Nam-Ho Kim
Dawn An
Joo-Ho Choi

Prognostics and Health Management of Engineering Systems

An Introduction

Springer

Prognostics and Health Management of Engineering Systems

Nam-Ho Kim · Dawn An · Joo-Ho Choi

Prognostics and Health Management of Engineering Systems

An Introduction

Springer

Nam-Ho Kim
Mechanical and Aerospace Engineering
University of Florida
Gainesville, FL
USA

Joo-Ho Choi
Aerospace & Mechanical Engineering
Korea Aerospace University
Goyang-City, Kyonggi-do
Republic of Korea

Dawn An
Daegyeong Division/Aircraft System Technology Group
Korea Institute of Industrial Technology
Yeongcheon-si, Gyeongbuk-do
Republic of Korea

ISBN 978-3-319-44740-7 ISBN 978-3-319-44742-1 (eBook)
DOI 10.1007/978-3-319-44742-1

Library of Congress Control Number: 2016948272

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To our families

Preface

A good maintenance strategy is essential to keep complex engineering systems safe. Historically, maintenance has evolved from post-failure repair to preventive maintenance to Condition-Based Maintenance (CBM). Preventive maintenance is an expensive and time-consuming process because it is carried out periodically regardless of the health state of systems. For modern complex systems with high reliability requirements, preventive maintenance has become a major expense of many industrial companies. CBM has recently received much attention as a cost-effective maintenance strategy, which is to perform maintenance only when needed. Prognostics and Health Management (PHM) is the key technology to accomplish CBM.

PHM is a new engineering approach that enables real-time health assessment of a system under its actual operating conditions, as well as the prediction of its future state based on up-to-date information, by incorporating various disciplines including sensing technologies, physics of failure, machine learning, modern statistics, and reliability engineering. It enables engineers to turn data and health states into information that will improve our knowledge on the system and provide a strategy to maintain the system in its originally intended function. While PHM has roots from the aerospace industry, it is now explored in many applications including manufacturing, automotive, railway, energy and heavy industry.

Since PHM is a relatively new research area, many researchers and students struggle to find a textbook that clearly explains basic algorithms and provides objective comparison between different algorithms. The objective of this book is to introduce the methods of predicting the future behavior of a system's health and the remaining useful life to determine an appropriate maintenance schedule. The uniqueness of this book lies not only in its introduction to various prognostics algorithms, but also in its explanations of their attributes and pros and cons in terms of model definition, model parameter estimation, and ability to handle noise and bias in data. Therefore, beginners in this field can select appropriate methods for their fields of application.

This book is suitable for graduate students in mechanical, civil, aerospace, electrical and industrial engineering, and engineering mechanics, as well as researchers and maintenance engineers in the above fields.

The textbook is organized into seven chapters. In Chap. 1, the basic ideas of PHM are introduced along with historical backgrounds, industrial applications, reviews of algorithms, and benefits and challenges of PHM. Before discussing individual prognostics algorithms in detail, Chap. 2 provides prognostics tutorials with a MATLAB code using simple examples. Even if simple polynomial models are used with the least-squares method, they contain most of important attributes of various prognostics algorithms. The tutorials include physics-based and data-driven prognostics algorithms to identify model parameters as well as to predict the remaining useful life. This chapter also introduces prognostics metrics to evaluate the performance of different algorithms as well as uncertainty due to noise in data.

A key step in prognostics is to convert the measured data from health monitoring systems into knowledge on damage degradation. Many prognostics algorithms utilize Bayes' theorem to update information on unknown model parameters using measured data. Chapter 3 introduces Bayesian inference with an explanation of uncertainty and conditional probability. For the purpose of prognostics, the chapter focuses on how to utilize prior information and likelihood functions from measured data in order to update the posterior probability density function (PDF) of model parameters. Depending on how information is updated, both recursive and total forms are discussed. The chapter ends with a method of generating samples from a posterior PDF.

When a physical model that describes the behavior of damage is available, it is always better to use it for prognostics. Chapter 4 presents physics-based prognostics algorithms, such as nonlinear least squares, Bayesian method, and particle filter. The major step in physics-based prognostics is to identify model parameters using measured data and to predict the remaining useful life using them. The chapter focuses on how to improve the accuracy of a degradation model and how to incorporate uncertainty in the future. The chapter ends by discussing issues in physics-based prognostics, which includes model adequacy, correlation between parameters, and quality of degradation data.

Even if physics-based approaches are powerful, many complex systems do not have a reliable physical model to describe the degradation of damage. Chapter 5 introduces data-driven approaches, which use information from observed data to identify the patterns of the degradation progress and predict the future state without using a physical model. As representative algorithms, the Gaussian process regression and neural network models are explained. Data-driven approaches share the same issues with physics-based approaches, such as model-form adequacy, estimation of optimal parameters, and quality of degradation data.

In Chap. 6, these prognostics algorithms are applied to fatigue crack growth problems to understand the attributes of different algorithms. In the case of physics-based approaches, correlation between model parameters, initial conditions, and loading conditions play an important role in the performance of algorithms. In the case of data-driven approaches, the availability of training data and the level of

noise are important. Chapter 7 presents several applications of prognostics in practical engineering systems, including wear in a revolute joint, fatigue crack growth in a panel, prognostics using accelerated life test data, and fatigue damage in bearings.

MATLAB programs for different algorithms as well as measurement data used in the book are available on the companion website of the book http://www2.mae.ufl.edu/nkim/PHM/. Each chapter contains a comprehensive set of exercise problems, some of which require MATLAB programs.

We thank the students who took various courses at the University of Florida and Korea Aerospace University. We are grateful for their valuable suggestions, especially those regarding the example and exercise problems. Finally, special thanks to Ms. Ting Dong for her outstanding work to correcting many errors in the manuscript.

Gainesville, USA — Nam-Ho Kim
Yeongcheon-si, Republic of Korea — Dawn An
Goyang-City, Republic of Korea — Joo-Ho Choi
June 2016

Contents

Chapter 1
Introduction

1.1 Prognostics and Health Management

Prognostics and health management (PHM) is a new engineering approach that enables real-time health assessment of a system under its actual operating conditions as well as the prediction of its future state based on up-to-date information by incorporating various disciplines including sensing technologies, failure physics, machine learning, modern statistics, and reliability engineering. It enables engineers to turn data and health state into information that will improve our knowledge on the system and provides a strategy to maintain the system in its originally intended function. While it has rooted from the aerospace industry, it is now explored in many applications including manufacturing, automotive, railway, energy, and heavy industry.

Because PHM can predict the system's actual remaining life during its operation, it enables the condition-based maintenance (CBM), a new maintenance strategy that can only repair/replace actually damaged parts, which can reduce the total life cycle costs. CBM consists of automated hardware and software systems that monitor, detect, isolate, and predict equipment performance and degradation without interrupting systems daily operation. In CBM, maintenance of systems/components is based on the actual condition of the equipment in contrast to breakdown or scheduled maintenance. In order to make a timely decision on maintenance, prognostics is a key enabling technology for CBM.

Every system deteriorates its performance over time subject to the stress or load in operation. Therefore, maintenance should be exercised to assure satisfactory level of reliability during the life of the system. The earliest maintenance has been corrective maintenance (also called reactive, unplanned or breakdown maintenance), which takes place only when failure has already occurred, hence, is passive in nature. Corrective maintenance, which is often called the first generation maintenance, has been placed since human made machines. Since corrective maintenance occurs after the useful life of the system is all consumed, there is no

N.-H. Kim et al., *Prognostics and Health Management of Engineering Systems*,
DOI 10.1007/978-3-319-44742-1_1

preparation time for maintenance. Unless replacement parts are already available, it takes the longest time for maintenance, and the forced outage cost is highest. Since it is hard to predict when the system will break down, the availability of the system is poor. However, it only replaces parts that are actually broken down, the number of replacement parts is the smallest.

A later maintenance technique is time-based preventive maintenance (also called planned maintenance or the second generation maintenance), which sets a periodic interval to prevent failures regardless of the health state of the system. Traditional reliability engineering has been applied to set the proper interval, which uses the reliability prediction methods based on the handbook or historical field data. This is the most popular maintenance strategy where most replacements are prescheduled. The most important aspect of the preventive maintenance is cost because it replaces all parts even if many of them may not need to replace. Preventive maintenance is cost-effective if all parts are expected to be failed about same time. However, this strategy is inefficient when only a small number of parts fail because this strategy enforces to replace many parts that are not going to be failed.

In order to illustrate the issue of wasted maintenance, consider the maintenance of repairing cracks in airplane panels. The regulation from Federal Aviation Administration requires to replace/repair cracks in the size of 0.1″ during type-C inspection, which is performed every 6,000 flight cycles. This regulation is designed from reliability assessment to keep the airplane frame safe in the reliability level of 10^{-7}, which means one failure out of ten millions. If there exists a crack in the size of 0.1″, the probability that the crack grows and becomes unstable in next 6,000 flights is in the order of 10^{-7}. Therefore, during type-C inspection, if any crack in the size of 0.1″ is detected, it has to be repaired or if several cracks are detected, the panel must be replaced. This is a typical example of preventive maintenance. But, an extreme scenario is that if there are ten millions of cracks in the size of 0.1″ in an airplane (this is a hypothetical assumption), only one of them will grow and become unstable. However, in order to keep the airplane safe, all ten millions of cracks should be repaired, which is a huge increase of maintenance cost!

With the rapid development of technology, modern systems become more and more complex while maintaining higher reliability, and this results in the higher maintenance cost. Eventually, preventive maintenance has become a major expense of many industrial companies. In order to reduce maintenance cost while maintaining the desired level of reliability and safety, the CBM has appeared as a promising solution, which is to exercise the maintenance only when needed, and the PHM is the key technology to accomplish this.

Figure 1.1 shows maintenance strategies versus cost plot. When the number of failed parts is very small, it would be cost-effective to perform corrective maintenance because only a small number of parts will need maintenance. When the number of failed parts is large; that is, when most of the parts will fail, it would be more efficient to perform preventive maintenance. However, in most engineering applications, it would be more cost beneficial to perform condition-based maintenance.

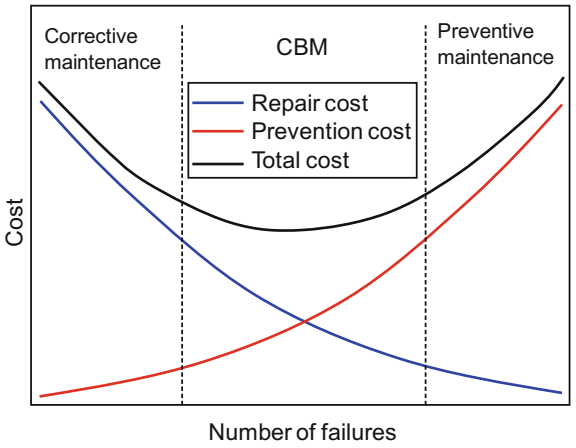


Fig. 1.1 Maintenance strategies versus cost

For the case of the previous explanation of cracks in airplane panels, CBM only repairs those cracks that actually grow and become unstable. Therefore, the maintenance cost can significantly be reduced. Even if CBM reduces the number of repaired or replaced panels the same as corrective maintenance, however, it still shares the same problem of no preparation time for maintenance and long maintenance time due to delay in parts supply systems. In order to take a full advantage of CBM, it is necessary to integrate it with prognostics. By predicting how cracks will grow in the future, it is possible to schedule appropriate maintenance time. Therefore, by combining CBM with prognostics, it is possible to take advantages from both corrective and preventive maintenances.

Figure 1.2 illustrates such an evolution of maintenance strategy, which explains that the maintenance is shifted from corrective (unscheduled, passive) or preventive (scheduled, active) maintenance strategy to the condition-based (predictive, proactive) ones.

The main steps of PHM include data acquisition, diagnostics, prognostics, and health management as shown in Fig. 1.3. The first step is data acquisition, which is to collect measurement data from the sensors and process them to extract useful features for diagnosis. The second step is the diagnostics, in which the fault is detected for any anomaly, isolated to determine which component is failing and to identify how severe it is with respect to the failure threshold. Third step is the prognostics that predicts how long it will take until failure under the current operating condition. The last step is the health management to manage in optimal manner the maintenance scheduling and logistics support. Among these, prognostics is the key enabler that permits the reliability of a system to be evaluated in its actual life cycle conditions. In other words, it predicts the time at which a system or a component will no longer perform its intended function, thus giving users the opportunity to mitigate system level risks while extending its useful life.

A number of different definitions of prognostics were proposed by several communities (Sikorska et al. 2011). Among them, the most encompassing description of prognostics is presented by ISO13381-1, which defines prognostics as '*an estimation of time to failure and risk for one or more existing and future*

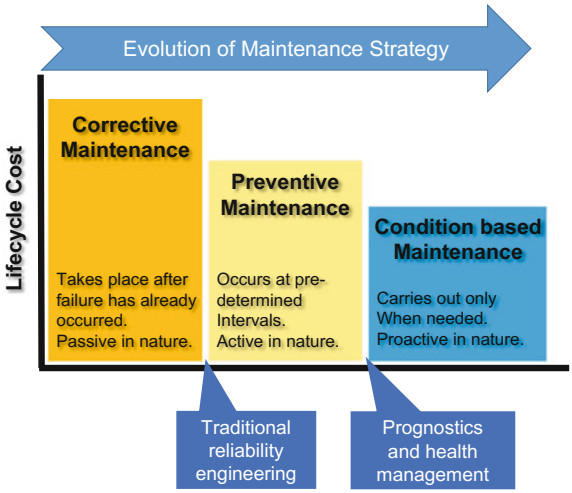


Fig. 1.2 Evolution of maintenance strategy

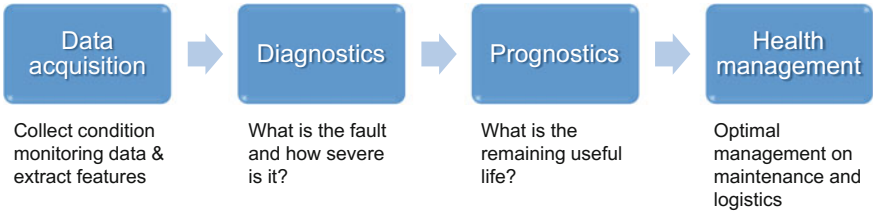


Fig. 1.3 Four main steps of the PHM

failure modes' (ISO 13381-1 2015). Standards or architectures of the PHM process were also proposed. Among them, the Open System Architecture for Condition Based Maintenance (OSA/CBM) is the most popular, which consists of six phases or layers: data acquisition, manipulation, condition monitoring, health assessment, prognostics, and decision-making (Callan et al. 2006; He and Lin 2012). The first layer is data acquisition and is concerned with the activity of measuring data using a variety of sensors. The data manipulation layer performs processing of raw data which then feeds into the condition monitoring layer that calculates condition indicators and provides alarm capabilities for anomaly. The health assessment layer uses the condition indicators to determine health status describing how bad it is in quantitative manner. An estimate of future progress is made including the remaining useful life by the prognostics layer. Decision-making layer is to produce suitable measures for replacement, and maintenance activities based on the data from the previous layers. A conceptual CBM system functional architecture illustrating the layers is shown in Fig. 1.4 (Callan et al. 2006). Comparing with Fig. 1.3, the first two layers belong to the data acquisition, the next two are the diagnostics, and the remaining two are the prognostics and health management, respectively.

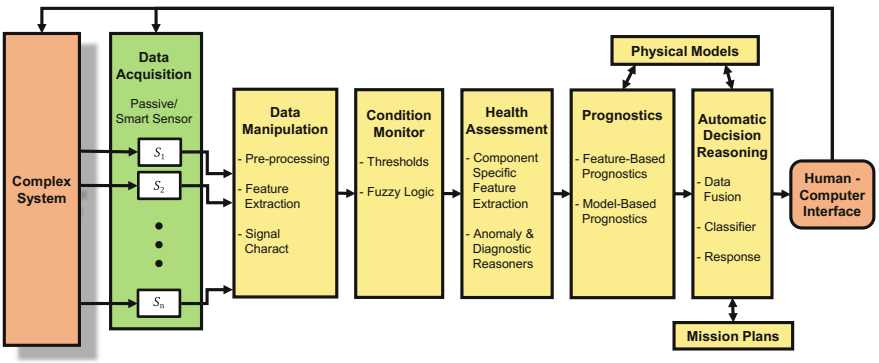


Fig. 1.4 Conceptual CBM system functional architecture (Callan et al. 2006)

1.2 Historical Background

Initially, PHM was started to reduce a helicopter accident rate by the Civil Aviation Authority of United Kingdom in the 1980s, and has been developed in 1990s based on health and usage monitoring system (HUMS) that measures health conditions and performance of helicopter. HUMS achieved good results to reduce accident rate more than a half by being set on in-service helicopter (see Fig. 1.5, excerpted from John Burt Associates Limited 2011 and Oil and Gas UK).

During 1990s, the concept of Vehicle Health Monitoring was adopted to the aerospace research of NASA in USA, which is to monitor the health of outer space vehicle. However, it was soon replaced by a more universal term Integrated Vehicle Health Management or System Health Management to incorporate the prognostics of various space systems (Pettit et al. 1999).

	Per 100,000 Flying Hours		Per 100,000 Sectors (Flight Stages)	
	Occupant Fatal Accident Rate	Non-Fatal Reportable Accident Rate	Occupant Fatal Accident Rate	Non-Fatal Reportable Accident Rate
1981-1990 years	5.61	2.24	2.39	0.96
1991-2000 years	1.13	0.82	0.49	0.35

Fig. 1.5 UKCS Offshore Helicopter Safety Record 1991–2000

In the 2000s, the defense advanced research projects agency (DARPA) in USA has developed the structural integrity prognosis system (SIPS) and condition-based maintenance plus (CBM+) (Space Daily 2003), which have the same purpose. The name prognostics and health management (PHM) was first adopted in the program for joint strike fighter (JSF) development (Joint Strike Fighter Program Office 2016) (see Fig. 1.6). The department of defense (DoD) made PHM system required by operation of the defense acquisition system from 2003 (DoD Instruction 5000.2, 2003), which states that "*program managers shall optimize operational readiness through affordable, integrated, embedded diagnostics and prognostics, and embedded training and testing, serialized item management, automatic identification technology (AIT), and iterative technology refreshment*".

Since then, the PHM technology has undergone significant development in various aspects, which includes the fundamental study of failure physics, sensor developments, feature extraction, diagnostics for fault detection and classification, and prognostics for failure prediction. These techniques have been explored and expanded to various industries. As the technologies are applied and matured in the industry, articles are increasing that addresses successful applications in various aspects (Sun et al. 2012; Yin et al. 2016).

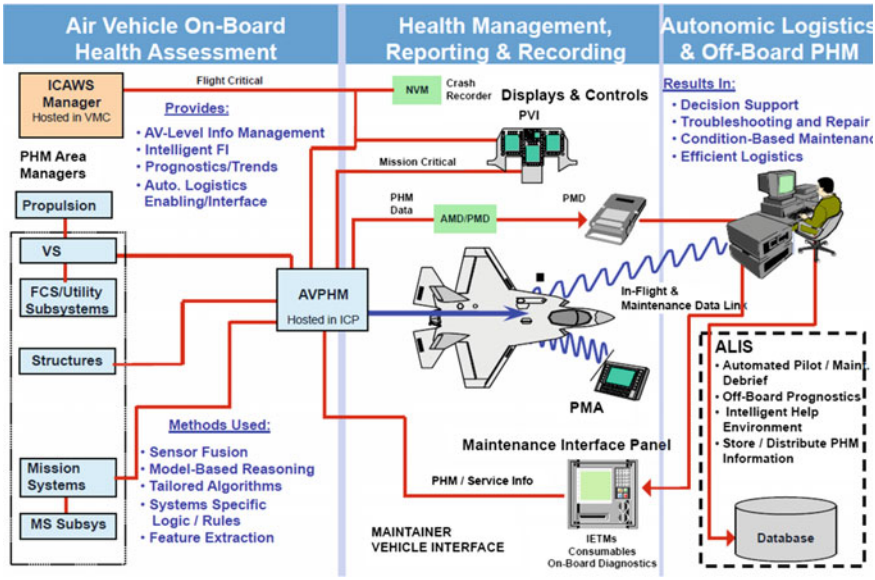


Fig. 1.6 PHM Architecture and Enabling Technologies (from Joint Strike Fighter Program Office)

In terms of success in the manufacturing sector for PHM solutions, there are some economic numbers that can be reported. For example, the National Science Foundation funded in independent economic impacts study on Industry/University Cooperative Research Centers (I/UCRC) and surveyed five industrial members of the Center for Intelligent Maintenance Systems; the five companies (predominantly manufacturing applications) reported a savings of over $855 Million U.S. dollars based on the successful implementation of the predictive monitoring and PHM solutions (Gray et al. 2012).

From few years ago, technical societies were established to gather and enhance the knowledge in many research fields. As representative ones, the PHM society holds the annual conference and publishes International Journal of Prognostics and Health Management (IJPHM) since it has been founded in 2009 (PHM Society 2009). Also, the reliability society in IEEE holds conference on PHM annually since 2011 (IEEE Reliability Society 2011). PHM study at the current time is being led by diverse institutes, some of which are briefly introduced here.

Intelligent Maintenance Systems (IMS) Center (2001): IMS center was established in 2001 by the University of Cincinnati, the University of Michigan, and Missouri University of Science and Technology as the NSF Industry/University Cooperative Research Center. The center focuses on the research for predictive analytics and industrial big data modeling for lifecycle performance of industrial systems. The center has coined the trademarked Watchdog Agent® prognostics tools and Device-to-Business (D2B) predictronics platform for e-maintenance applications.

Center for Advanced Life Cycle Engineering (CALCE) (1986): The CALCE, established in 1986 at the University of Maryland, is recognized as the leader in the reliability assessment of electronics based on the physics-based failure analysis. Recently, the center is actively working on the PHM research for electronics applications by the use of expendable devices, monitoring and reasoning of precursors to impending failure and life consumption modeling.

Prognostics Center of Excellence (PCoE) (2016): The PCoE, located at NASA Ames Research Center, provides an umbrella for prognostic technology development, specifically addressing prognostic technology gaps within the application areas of aeronautics and space exploration. The PCoE is currently investigating damage propagation mechanisms on safety-critical actuators for transport class aircraft, damage mechanisms on aircraft wiring insulation, and damage propagation mechanisms for critical electrical and electronic components in avionic equipment.

FEMTO-ST (2004): The FEMTO-ST Institute is a joint research institution, created in 2004 in France through the merger of five local laboratories, and restructured later into seven departments. The PHM team within the automatic control department develops advanced algorithms for classification, prediction, and decision on the problems such as the aging of fuel cells, composite materials, and observations from sensor networks.

Integrated Vehicle Health Management (IVHM) center (2008): The IVHM was established at Cranfield University by the support of Boeing in 2008 to act as a world leading research hub for the aircraft PHM study. The center has since then

offered the world's first IVHM M.Sc. course and hosts several Ph.D. students researching the application of IVHM to different fields.

1.3 PHM Applications

The PHM development was pioneered by the aerospace and defense industry due to its unique requirements and history for safety-critical nature and high-maintenance cost, as shown in Fig. 1.7 (Vachtsevanos et al. 2006). Since then, the technologies have been applied broadly and matured in the industry. Many reviews and overviews with different perspectives have been published in the literature over the past decade. Sun et al. (2012) have surveyed some of the PHM practices and case studies in their paper, which covers a wide range of applications like defense, aerospace, wind power, civil infrastructure, manufacturing, and electronics. Yin et al. (2016) have published a special section to collect the PHM applications in the industrial electronics.

In aerospace and defense systems, PHM techniques have seen the most significant advances in several applications. The health and usage management systems (HUMS) is an example of PHM solution for rotorcraft that detects several problems from shaft unbalance to gear and bearing deterioration (UTC Aerospace Systems 2013). Vachtsevanos et al. (2006) has implemented prognostics to predict the RUL of crack in the planetary gear plate of helicopter UH-60A (see Fig. 1.7). Pratt and Whitney has implemented advanced PHM systems in their engine for the F135 multipurpose fighter (VerWey 2009). General Electric Aviation has monitored aircraft engines for over 15 years and is providing diagnostic services to detect early symptoms of engine problems before downtime. The prognostics technology was included in the weapons platforms by the program from the US Army's Logistics Integration Agency (Greitzer et al. 2001), and in the transmission of SH-60 helicopters by the Naval air systems command (NAVAIR) (Hardman 2004).

In the heavy and energy industries, prognostic approach was applied to the gas turbine engines such as Rolls-Royce industrial AVON 1535 (Li and Nikitsaranont 2008).

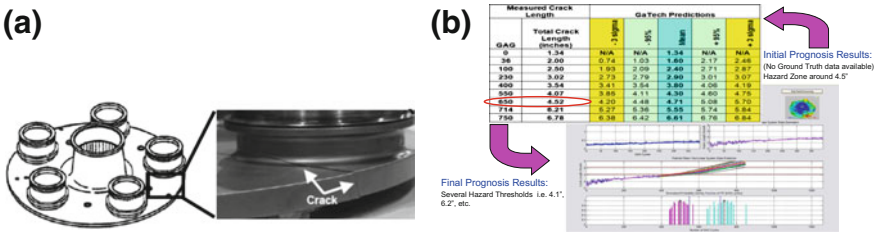


Fig. 1.7 Prognostics of crack on a planetary gear plate in UH-60A helicopter (Vachtsevanos et al. 2006). © John Wiley and Sons, 2007, reprinted with permission. **a** Crack on a planetary carrier plate, **b** prognosis result of crack growth

Komatsu and Caterpillar have developed the advanced data analysis algorithms to detect the vehicle problems at an early stage (Wang et al. 2007). In the renewable energy applications, wind turbine drivetrain condition monitoring systems have seen much progress (see Fig. 1.8) (Siegel et al. 2014; Bechhoefer and Mortom 2012).

In the manufacturing industry, in which the primary focus is on the reduced down time, numerous reviews have been devoted to address the state of the art of PHM technology for the manufacturing applications (e.g., see Jardine et al. 2006; Lee et al. 2014; Peng et al. 2010). Some of specific applications are the health monitoring of manufacturing equipment such as the spindle bearing of rotating machinery (Liao and Lee 2010), machine tool wear (see Fig. 1.9) (Kao et al. 2011), surge for air compressors (Sodemann et al. 2006) and the fleet health of industrial robots (Siegel et al. 2009). Erosion of induction furnace was predicted in a copper products manufacturing company (Christer et al. 1997). Failure of three water pumps was predicted using vibration data in a large soft drinks manufacturing plant (Wang et al. 2000).

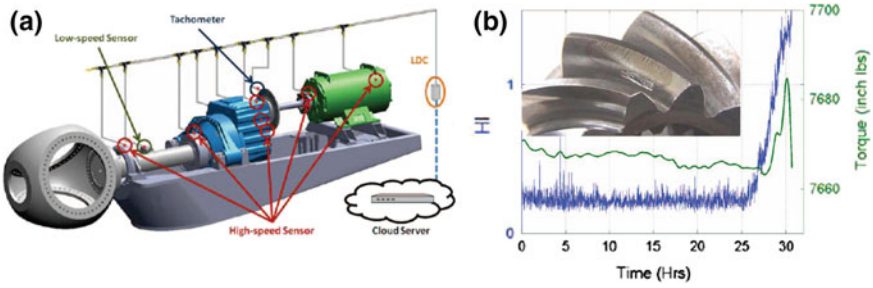


Fig. 1.8 Example application of prognosis in wind turbine (Bechhoefer and Mortom 2012). **a** System layout, **b** health index (HI) for gear run to failure

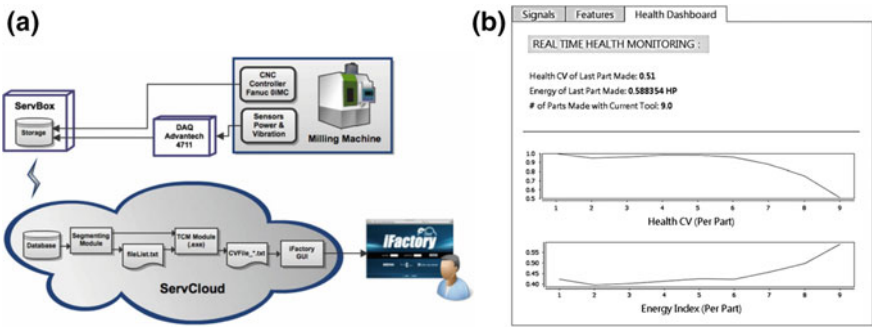


Fig. 1.9 Prognostics example of tool condition monitoring (TCM) application (Kao et al. 2011). *Engineering Asset Management Review*, "Deployment of prognostics technologies and tools for asset management: Platforms and applications," 2, 2011, 1–29, Lee, Jay et al., with permission of Springer. **a** Implementation framework of TCM, **b** screenshot of prognostics

In the electronics industry, the PHM study was the most active in the Center for Advanced Life Cycle Engineering (CALCE). Example applications are the failure prognosis of laptop computers (Vichare et al. 2004), and power electronics devices (IGBTs) used in avionics (Saha et al. 2009) to name a few.

Recently, there are also movements on the establishment of standards by the professional societies. In IEEE, standards for the PHM are discussed by the reliability subdivision and published a relevant article (Vogl et al. 2014). In the manufacturing side, similar activity has been taken by National Institute of Standards and Technology of USA, and published a report (Sheppard et al. 2009).

1.4 Review of Prognostics Algorithms

The PHM applications introduced in Sect. 1.3 can be facilitated based on a variety of data analysis techniques. In this section, the techniques of prognostics, as a part of PHM, are reviewed, which is what this book focuses on. Prognostics is to predict future damage/degradation and the remaining useful life (RUL) of in-service systems based on the measured damage data.

In general, prognostics methods can be categorized into physics-based and data-driven approaches based on the usage of information, which is illustrated in Fig. 1.10. Physics-based approaches assume that a physical model describing the behavior of degradation is available and combine the physical model with measured data and usage conditions to identify model parameters and to predict the future behavior. Since this approach needs to capture physical basis of failure in model that relates the forces that cause damage to their effect, it requires a detailed understanding of the problem.

The most important advantage of physics-based method is that the results tend to be intuitive because they are based on modeled phenomenon. Also, once a model is developed, it can be reused for different systems or different designs by tuning

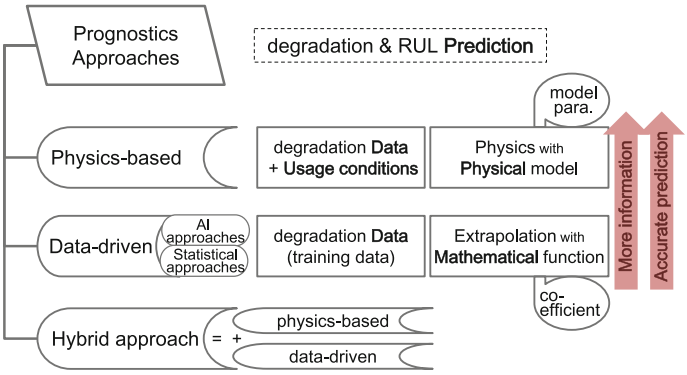


Fig. 1.10 Categorization and definition of prognostics methods

model parameters. If incorporated early enough in design process, it can drive sensor requirements; that is adding or removing sensors. The method is generally considered as computationally efficient than the data-driven method. However, the advantages of physics-based approaches can also work against them. For example, model development requires a thorough understanding of the system. If any important physical phenomenon is missed, then it can lead to failure of predicting degradation behavior. Also, high-fidelity models, especially for numerical models, can be computationally intensive.

Data-driven approaches use information from collected data at current and previous usage conditions (called training data) to identify the characteristics of the currently measured degradation state and to predict the future trend. The conceptual understanding for data-driven approaches is an extrapolation with mathematical function based on identified coefficients. Since this approach simply fits the trend of data, there is no guarantee that extrapolation will be meaningful since failure progression from current state forward may be unrelated to prior failure progression. The success of data-driven methods depends on collecting statistics of failures as a function of current state, which requires volumes of data. Without having a comprehensive understanding of the system, it is difficult to know how much data are good enough for the purpose of prognostics.

In the viewpoint of practicality, data-driven approaches are easy and fast to implement. In fact, several off-the-shelf packages are available for data mining and machine learning. By collecting enough data, it is possible to identify relationships that were not previously considered. Also, since the method works with objective data, it can consider all relationships without any prejudice. However, this method requires a lot of data that include all possible modes of failure for the same or similar systems. Since no physical knowledge is involved, the results may be counter-intuitive and it is dangerous to accept a result without understanding the cause of the problem. The method can be computationally intensive, in both analysis and implementation.

The application domains of the two methods can be illustrated in terms of reliability of physics model and availability of data, as shown in Fig. 1.11. When a highly reliable physics model is available, physics-based approaches will work well even with a small number of data. When much data are available, both methods can be applicable. The issue is when a reliable physics model is not available. In this case, if much data are available, then data-driven approach can be applicable without requiring physics model. However, if a small number of data are available, there is no reliable prognostics approaches and thus health assessment will be unreliable.

There are two main differences between physics-based and data-driven approaches; (1) availability of a physical model including usage conditions and (2) use of training data to identify the characteristics of the damage state. Since physics-based approaches require more information (physical model) than data-driven ones, more accurate and long-term prediction is possible. Physical degradation models are, however, rare in practice. Then data-driven approaches should be employed, but it is not easy to obtain several sets of degradation data due to expensive time and costs.

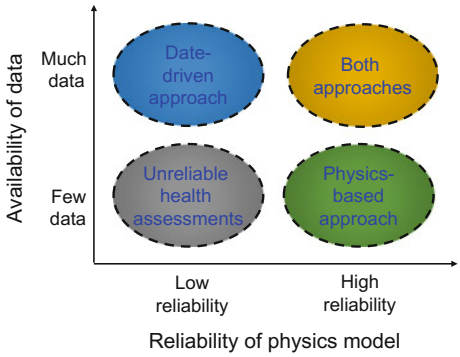


Fig. 1.11 Application domains of physics-based and data-driven prognostics algorithms

Hybrid approaches are to integrate advantages of both physics-based and data-driven methods to improve the prediction capability. For practical complex systems, it might not be efficient to use physics-based or data-driven approaches alone. Instead, both approaches are used together to maximize the prediction capability. For example, the knowledge on physical behavior can be used to determine the mathematical model (e.g., determining the order of polynomial or exponential functions) in data-driven method. It is also possible to use data-driven system model in conjunction with a physics-based fault model, or vice versa. This approach, however, depends on a specific application domain, hence would not be discussed in this book. More information about hybrid approaches can be found in a review material by Liao and Köttig (2014).

A simple example of prognostics is depicted in Fig. 1.12 with Paris model (Paris and Erdogan 1963) as a physics-based approach and a polynomial function as a data-driven approach. Even though the given information is different from each other, the prognostics procedure is almost the same: identifying parameters based on damage data, and predicting future damage by substituting the identified parameters to each physical degradation model or mathematical function. The damage growth depends on model parameters (m, C) and usage condition $(\Delta\sigma)$ in the Paris model. Equivalently, the coefficients (β's) correspond to model parameters

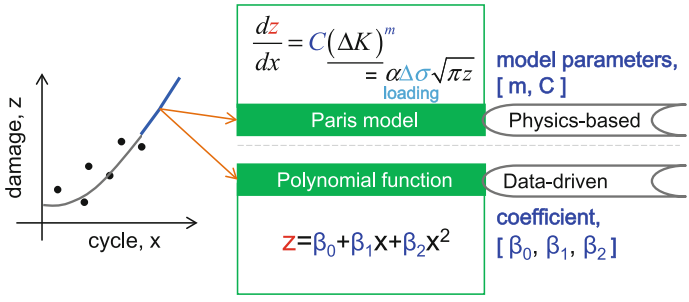


Fig. 1.12 Simple example of prognostics

in the data-driven approach. However, the usage condition is not necessarily required in data-driven approaches but can be utilized in the polynomial function as an input variable.

While the algorithms of physics-based approaches are limited to methods for parameter identification, data-driven approaches have a great variety of algorithms based on the type of mathematical functions and how to utilize given information. In fact, the algorithms of physics-based approaches can be used for data-driven approaches since a mathematical function can be employed instead of a physical model. Since, however, a form of the mathematical functions that can be employed in physics-based approaches is limited to polynomial or exponential functions, there is a typical categorization for algorithms between two approaches.

Since the behavior of damage depends on model parameters in physics-based approaches, identifying them is the most important issue in predicting future damage behaviors. Due to uncertainty in usage condition and noise in data, most algorithms identify model parameters as probability distributions rather than deterministic values based on Bayesian inference (Bayes and Price 1763) that is a statistical method in which observations are used to estimate and update unknown model parameters in the form of a probability density function. The most typical technique is the particle filter (Doucet et al. 2001; An et al. 2013) that expresses the distribution of parameters with a number of particles and their weights based on sequential Bayesian updating. The Kalman filter (Kalman 1960) is also a filtering method based on the sequential Bayesian updating, which gives the exact posterior distribution in the case of a linear system with Gaussian noise. Other Kalman filter family techniques, such as extended/unscented Kalman filter (Ristic et al. 2004; Julier and Uhlmann 2004) have been developed to improve the performance for nonlinear systems. Bayesian update also can be processed simultaneously, which is called a Bayesian method (Choi et al. 2010) in this book. Finally, the nonlinear least squares (Gavin 2016) that is a nonlinear version of the least squares is often utilized, but in which the parameters are estimated with deterministic values and their variance.

In data-driven approaches, the most common technique is the artificial neural network (so-called neural network) (Chakraborty et al. 1992; Ahmadzadeh and Lundberg 2013; Li et al. 2013) as an artificial intelligence method, in which a network model learns a way to produce a desired output, such as the level of degradation or lifespan, by reacting to given inputs, such as time and usage conditions. Also, the Gaussian process (GP) regression (Mackay 1998; Seeger 2004) is a commonly used method among regression-based data-driven approaches, which is a linear regression like the least squares method (Bretscher 1995) with the assumption that errors between a regression function and data are correlated. Including the least squares regression, there are a wide variety of algorithms such as the fuzzy logic (Zio and Maio 2010), relevance/support vector machine (RVM/SVM) (Tipping 2001; Benkedjouh et al. 2015), gamma process (Pandey and Noortwijk 2004), Wiener processes (Si et al. 2013), hidden Markov model (Liu et al. 2012), etc. A comprehensive review about various data-driven algorithms can be found in a reference by Si et al. (2011).

1.5 Benefits and Challenges for Prognostics

1.5.1 Benefits in Life-Cycle Cost

There are many different ways to view the benefits of PHM. The biggest benefit may be the reduction of total life-cycle cost as is addressed below.

Reduced operating cost: Because the maintenance costs of complex systems can be extremely large, especially for a fleet of systems, the cost can be reduced by the PHM substantially. More specifically, the PHM provides savings in two ways; the first is through the CBM practices and the second is through a more automated maintenance and logistics support system. In order to understand CBM, suppose a simple example to change oil in your car. The old philosophy is to change your oil at a specific period, every 3–5k miles, an example of a preventive maintenance. By continuously monitoring the health of the oil through sensor readings, the CBM is realized by replacing oil only when needed. The second is through the automated maintenance and logistics system, which will allow for a more just-in-time maintenance environment. Spare parts can be ordered and maintenance can be planned in advance when they are needed, instead of after a catastrophic failure. This will realize more cost savings for increasingly complex support systems such as military maintenance. Although it addresses the connected machine or internet of the things (IOT) which is wider than the PHM concept, General Electric is claiming that only 1 % improvement in efficiency can make $276 billion in the five major industries as shown in Fig. 1.13 (Evans and Annunziata 2012).

Increased revenue: Although this is not direct, the PHM can also help to increase the revenue stream of a business. If a product seller can offer a more reliable product that has equipped the PHM capability to their customers, they will be able to gain a larger market share and therefore increase revenue. In fact, this is taking place in the market such as the aircraft engines by GE and mining machines by Komatsu. They are not only selling their products, but also providing PHM services to the buyer companies from which more revenue can be made. An

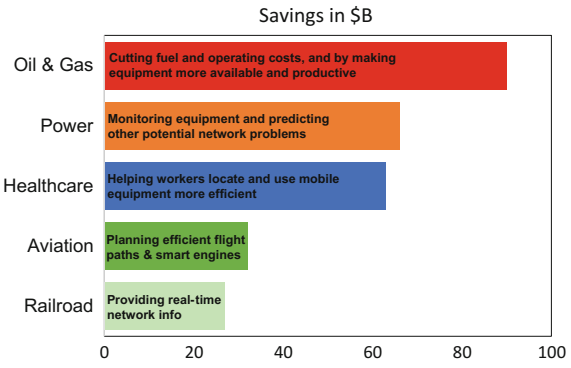


Fig. 1.13 Power of 1 % improvement by the connected machines in five industries

example is given in Fig. 1.14, in which the PHM system is imbedded in the mining machines of Komatsu company (Murakami et al. 2002).

Sun et al. (2012) made more detailed analysis on the benefits in view of the stages of the life cycle process, which are (a) system design and development, (b) production, (c) system operations, and (d) logistics support and maintenance. They are summarized as follows.

1.5.2 *Benefits in System Design and Development*

Optimum system design: When developing a new product, a large number of tests including accelerated life tests are needed, which is time-consuming and costly, but still is not truly representative of the actual conditions. Prognostics can provide those data from the actual conditions throughout the system life cycle. This information can be used to improve and optimize new design which is much less costly. Furthermore, if the new system includes the PHM capability, it does not have to be conservative in terms of safety and reliability because the faults are monitored and insured against failure by the PHM.

Improved reliability prediction: Reliability prediction is of importance in the design of safety-critical systems. Traditional methods are based on the database of handbooks such as MIL-HDBK-217 and its derivatives, but they are often proven to be

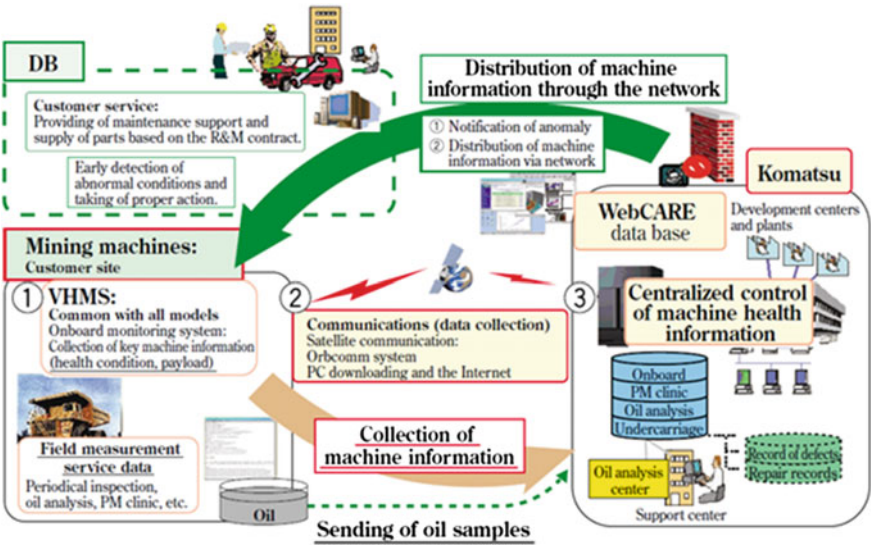


Fig. 1.14 PHM capability in the mining machines of Komatsu company. From Murakami, Taku, et al. "Development of vehicle health monitoring system (VHMS/WebCARE) for large-sized construction machine" (2002): 15–21. Reprinted with permission

misleading, providing incorrect life predictions. In contrast, the data collected from PHM reflect the actual life cycle conditions of the system, which facilitates more accurate damage and RUL assessment, hence, allows more accurate reliability prediction. A better reliability plan can be made accordingly with more cost-effective way.

Improved logistics support system: In general, when a new system is developed, logistics support systems are designed together, which has a great impact on the life-cycle cost. Prognostics can assist in constructing this system. For example, in the electronic systems, belief has been that electronics failures cannot be anticipated, and can only be dealt with as they occur. This has led to the large-scale supply resources with scattered spare parts depots and their management, which is another big source of increased cost. By using the PHM, the failure time and the number of electronics can be predicted in advance, which enables a significant reduction in the logistics.

1.5.3 Benefits in Production

Better process quality control: In practice, the quality deteriorates whenever the parameter of a piece of machining equipment deviates from its nominal or optimal value, such as the performance of a cutting tool, which degrades gradually. The monitoring and prognostics of manufacturing equipment status (e.g., vibration, strength, power, and operating mode), as well as the wear or fault status can provide more information about equipment itself than traditional quality control, thus promoting the quality control process and quality assurance.

Integrated maintenance development by OEMs: Most of the information used for high-level system prognostics comes from subsystems and assemblies. Therefore, it is necessary for system designers to work together with the suppliers to define the variables to be monitored and develop the algorithms for the efficient CBM. The suppliers then provide a component or subsystem level prognostics solution for system manufacturers. By sharing and integrating the PHM information, the system manufacturer can conduct CBM in practice.

1.5.4 Benefits in System Operation

Increased System Safety: Prognostics provides the ability to anticipate incipient faults prior to their progressing to catastrophic failure of the system. These capabilities enable more accurate management of the health of systems. Figure 1.15 shows an example of degrading health index and its prognosis. The prognostic "distance," which is the time interval between the advance failure warning time and the estimated failure time, is shown in Fig. 1.15. The required advance time before failure can vary from seconds to years. In the US space shuttle, the crew can have a 4-second window upon takeoff to eject for survival. In aircraft, prognostic warning

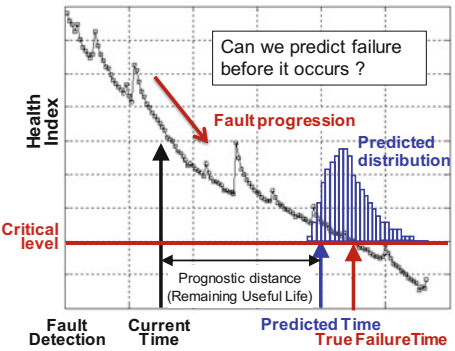


Fig. 1.15 Failure warning in advance by prognostics capability

may ensure a lead time of hours for unit replacement, while a couple of months for corrosion maintenance.

Improved Operational Reliability: In original sense, with proper design and effective production process control, the inherent reliability of a system can be made. However, under actual operating conditions, the environmental and operational loads may sometimes be quite different from what the system was designed for, and will affect the operational reliability of the system. In such cases, a system designed with intended reliability and life under predicted usage will be exposed to the risk of failure when subject to a severe usage. This is illustrated in Fig. 1.16. The monitoring capability of PHM makes it possible to take proper actions under different usage conditions, thus increasing service lifetime while maintaining the intended reliability and achieving mission success.

1.5.5 Benefits in Logistics Support and Maintenance

Condition-based maintenance: As mentioned earlier, the primary benefit of PHM is that it enables CBM, which minimizes unscheduled maintenance, eliminates redundant inspections, reduces scheduled maintenance, extends maintenance

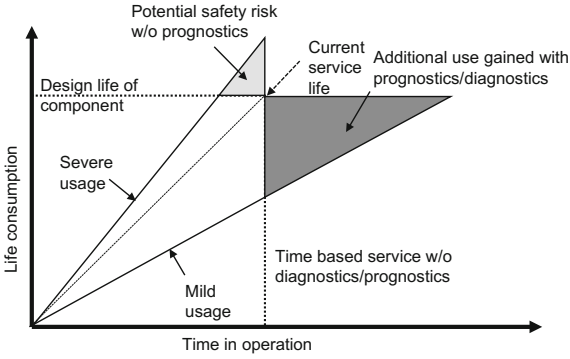


Fig. 1.16 Improved operational reliability by prognostics

interval, and most of all, reduces overall maintenance costs. Besides, it enhances identifying failed components, which helps preparation for maintenance in advance. In that case, the maintenance task can be prepared on ground while the airplane is still in flight, for example. There is a report that the prognostics system for the joint strike fighter (JSF) is expected to achieve a 20–40 % reduction in maintenance manpower, and a 50 % reduction in the logistics of machinery to support the plane (Scheuren et al. 1998).

Improved fleet-wide decision support: When the PHM is employed on a large number of systems (fleet), there are many more benefits than just multiplying the benefits for an individual system. Since the PHM will provide much more detailed information for the operation of every single component of the fleet, specific decisions can now be made in a variety of different situations. Where and when to deploy maintenance personnel, how many spare parts needs to be ordered, and where to perform opportunistic maintenance are only a few of the many fleet-wide benefits. Optimization techniques applied to the fleet-wide decisions can only help to realize greater overall system benefits of employing PHM.

Optimized logistics supply chain: Prognostics enables predictive logistics, which can improve the planning, scheduling, and control of activities in the supply chain. The main benefit comes from the fact that the operator can use information about the components' RUL to buy spare parts only when they are about to fail. By applying the prognostic information, fewer spare parts are used than before and they are delivered "just in time" thus, in-stock inventory levels are reduced, which leads to a substantial simplification of the supply chain.

Reduced maintenance-induced fault: When a mechanic works on a system to fix or replace a component, he may accidentally cause damage to another component, which is called a maintenance-induced fault. If this is not noticed, unexpected system downtime or even the catastrophic failure can result. Prognostics reduce the need for maintenance activities in the system, thus reducing the occurrence of human-related issues.

1.5.6 Challenges in Prognostics

While there are so many benefits in the PHM, there are a number of challenges to be explored in future studies, which are outlined as follows (Sun et al. 2012).

Implementing optimum sensor selection and localization: Data acquisition is a first step and an essential part of prognostics. It often requires the use of sensor systems to measure the environmental, operational, and performance parameters of a system. Inaccurate measurements by improper sensor selection and location can degrade the prognostic performance. The sensors should be able to accurately measure the change in the parameters linked to the critical failure mechanisms. The

possibility of sensor reliability and failures must also be taken into account. Some strategies to improve the reliability of sensors have been presented, such as using multiple sensors to monitor the same system (i.e., redundancy), and implementing sensor validation to access the integrity of a sensor system and adjust or correct it as necessary (Cheng et al. 2010).

Feature extraction: In order to have a meaningful prognostics, it is important to collect data that is directly related to damage. However, in many cases, it is difficult or impossible to collect damage data directly. For example, the cracks in the race of bearings are virtually impossible to measure because the bearing is constantly rotating. In such a case, a system response that is related to damage is measured to estimate the level of damage indirectly. For example, accelerometers are installed near the bearing to monitor the level of vibration due to cracks. In such a case, it is important to extract damage features out of vibration signals. Since the system vibration includes entire response of the system including noise, it is difficult to extract signals that are related to damage. Especially for complex systems, damage is only a tiny portion of the system. The signals related to damage tend to be very small compared to the signals related to system's response. Therefore, it is challenging to extract a small signals related to damage out of relatively large noise. In Chap. 7, a case study is presented to extract effective signals from noisy data of bearings.

Conditions for prognostics approaches: Generally, methods for prognostics can be grouped into physics-based and data-driven approaches as mentioned in Sect. 1.4. Physics-based approaches utilize knowledge of failure mechanism models or some other phenomenologically descriptive models of the system to assess the system's end of life. The advantage is its ability to predict the RUL accurately with a small number of data. However, sufficient information about the failure mode is needed for this. For example, in case of crack growth models, the materials, geometry, and operational and environmental loading conditions are required. In complex systems, these parameters may be difficult to obtain. Furthermore, the models require deep knowledge of the physical processes leading to failure, but in complex systems it is difficult to find such models, which is one of the limitations of physics-based approaches. Data-driven approaches use information from observed data to identify the characteristics of degradation progress and predict the future state without using any particular physical model. Therefore, the accuracy in RUL prediction results largely depends on the obtained data called training data. Usually many training data (especially up-to-failure data) are required to identify the degradation progress, but it is a challenge to obtain a large number of training data from in-service systems due to time and cost. In conclusion, it is advised to develop novel hybrid approaches by exploiting advantages of each approach to compensate the limitations. In this book, the attributes of typical algorithms of physic-based and data-driven approaches are introduced, which helps developing hybrid approaches after understanding intrinsic characteristics of each algorithm.

Addressing prognostic uncertainties and assessing its accuracy: Another major challenge for the use of prognostics is to develop methods that are capable of handling real-world uncertainties that lead to inaccurate predictions. Figure 1.17 shows some sources of uncertainty encountered in the prognostics, which are generally grouped into three different categories: (1) model uncertainty caused by model simplification and model parameters, (2) measurement and forecast uncertainty induced by environmental and operational loading conditions, and (3) inherent uncertainties with the geometry and materials of products mainly caused by the production process. These uncertainties can lead to the significant deviation of prognostics results from the actual situation. The development of methods that can be used to describe the uncertainty bounds and confidence levels for prognosis is very important. Method for prognostic accuracy assessment is also necessary for building and quantifying the confidence level of a prognostics system. In fact, the prognostic life of the models is expressed as a distribution as shown in Fig. 1.15. Although there is no general agreement as to an appropriate and acceptable metrics to assess the prognostic performance, a few researchers have suggested some ways. Leão et al. (2008) proposed a set of metrics to evaluate the performance of prognostics algorithms, including prognostics hit score, false alarm rate, missed estimation rate, correct rejection rate, prognostics effectivity, etc. Saxena et al. (2009) also suggested a list of metrics to assess critical aspects of RUL predictions, such as prognostic horizon, prediction spread, relative accuracy, convergence, horizon/precision ratio, etc., which is given in Fig. 1.18. Although efforts have been made to cover most PHM requirements, further refinements in concepts and definitions are expected as prognostics mature.

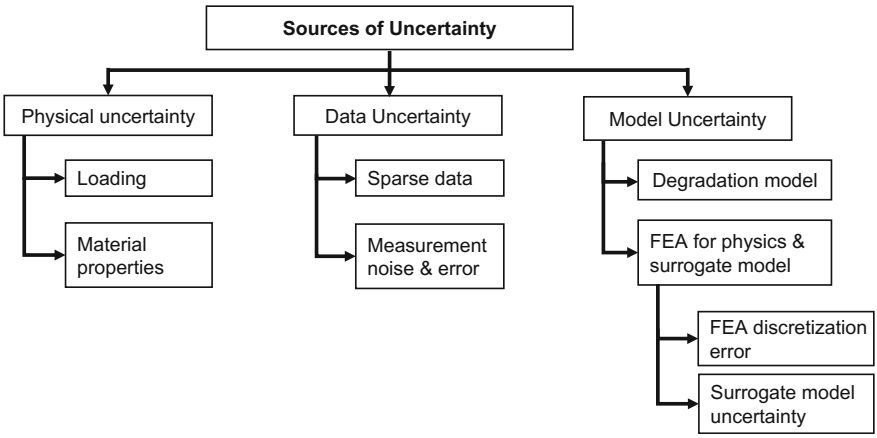


Fig. 1.17 Classification of uncertainty in the prognostics

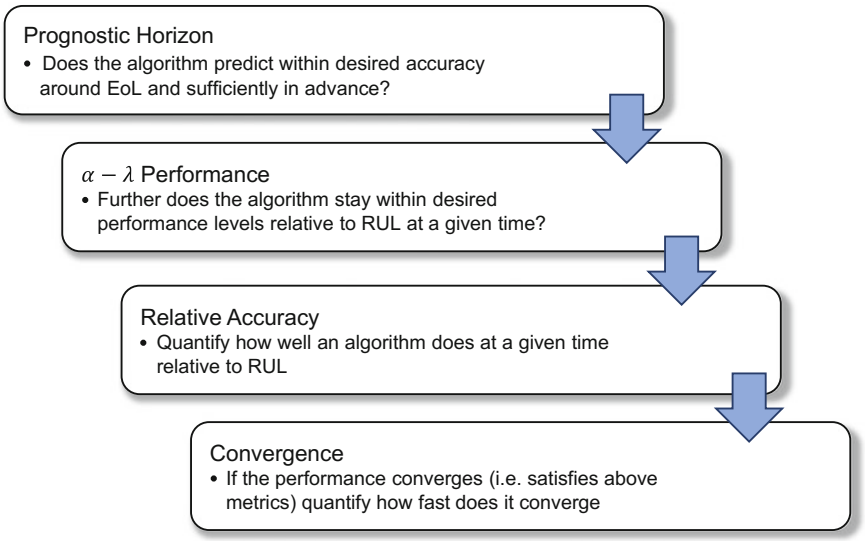


Fig. 1.18 Steps for the prognostics metrics. © (2012) IEEE. Reprinted, with permission, from E. Bechhoefer and B. Morton, "Condition monitoring architecture: To reduce total cost of ownership," *Prognostics and Health Management (PHM), 2012 IEEE Conference on*, Denver, CO, 2012, pp. 1–9

References

Ahmadzadeh F, Lundberg J (2013) Remaining useful life prediction of grinding mill liners using an artificial neural network. Miner Eng 53:1–8

An D, Choi JH, Kim NH (2013) Prognostics 101: a tutorial for particle filter-based prognostics algorithm using Matlab. Reliab Eng Syst Saf 115:161–169

Bayes T, Price R (1763) An Essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, communicated by Mr. Price, in a letter to John Canton, A. M. F. R. S. Philos Trans R Soc Lond 53:370–418. doi:10.1098/rstl.1763.0053

Bechhoefer E, Mortom B (2012) Condition monitoring architecture: to reduce total cost of ownership. In: Paper presented at the IEEE prognostics and health management conference, Denver, Colorado, USA, 18–21 June 2012

Benkedjouh T, Medjaher K, Zerhouni N et al (2015) Health assessment and life prediction of cutting tools based on support vector regression. J Intell Manuf 26(2):213–223

Bretscher O (ed) (1995) Linear algebra with applications. Prentice Hall, New Jersey

CALCE (1986) Center for advanced life cycle engineering. http://www.calce.umd.edu. Accessed 3 June 2016

Callan R, Larder B, Sandiford J (2006) An integrated approach to the development of an intelligent prognostic health management system. In: Paper presented at the IEEE aerospace conference, Big Sky, Montana, USA, 4–11 March 2006

Chakraborty K, Mehrotra K, Mohan CK et al (1992) Forecasting the behavior of multivariate time series using neural networks. Neural Netw 5:961–970

Cheng S, Azarian MH, Pecht MG (2010) Sensor systems for prognostics and health management. Sensors 10(6):5774–5797

Choi JH, An D, Gang J et al (2010) Bayesian approach for parameter estimation in the structural analysis and prognosis. In: Paper presented at the annual conference of the prog-nostics and health management society, Portland, Oregon, 13–16 Oct 2010

Christer AH, Wang W, Sharp J (1997) A state space condition monitoring model for furnace erosion prediction and replacement. Eur J Oper Res 101(1):1–14

DoD Instruction (2003) Operation of the defense acquisition system. DoD Instruction 5000.2. May 12, 2003. http://www.acq.osd.mil/dpap/Docs/new/5000.2%2005-12-06.pdf. Accessed 3 June 2016

Doucet A, Freitas DN, Gordon NJ (2001) Sequential Monte Carlo methods in practice. Springer, New York

Evans PC, Annunziata M (2012) Industrial internet: pushing the boundaries of minds and machines. GE report, 11.26.2012. http://www.ge.com/docs/chapters/Industrial_Internet.pdf. Accessed 3 June 2016

FEMTO-ST (2004) Franche-Comté Electronique, Mécanique, Thermique et Optique - Sciences et Technologies. http://www.femto-st.fr/en. Accessed 3 June 2016

Gavin HP (2016) The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems. Available via Duke University. http://people.duke.edu/∼hpgavin/ce281/lm.pdf. Accessed 28 May 2016

Gray DO, Drew R, George V (2012) Measuring the economic impacts of the NSF industry/university cooperative research centers program: a feasibility study. Final report for industry-university cooperative research center program. https://www.ncsu.edu/iucrc/PDFs/IUCRC_EconImpactFeasibilityReport_Final.pdf. Accessed 3 June 2016

Greitzer FL, Hostick CJ, Rhoads RE et al (2001) Determining how to do prognostics, and then determining what to do with it. In: Paper presented at the IEEE systems readiness technology conference, AUTOTESTCON proceedings, Valley Forge, Pennsylvania, USA, 20–23 Aug 2001

Hardman W (2004) Mechanical and propulsion systems prognostics: U.S. navy strategy and demonstration. J Miner Metals Mater Soc 56(3):21–27

He Y, Lin M. (2012) Research on PHM architecture design with electronic equipment. In: Paper presented at the IEEE conference on prognostics and health management, Beijing, China, 23–25 May 2012

IEEE Reliability Society (2011) International conference on prognostics and health management. http://rs.ieee.org. Accessed 3 Jun 2016

IMS Center (2001) Center for intelligent maintenance systems. http://www.imscenter.net. Accessed 3 June 2016

ISO 13381-1 (2015) Condition monitoring and diagnostics of machines—Prognostics—Part 1: General guidelines. International Standards Organization Available via http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=51436. Accessed 3 June 2016

IVHM (2008) Integrated vehicle health management center. https://www.cranfield.ac.uk/About/People-and-Resources/Schools-institutes-and-research-centres/satm-centres/Integrated-Vehicle-Health-Management-IVHM-Centre. Accessed 3 June 2016

Jardine AKS, Lin D, Banjevic D (2006) A review on machinery diagnostics and prognostics implementing condition-based maintenance. Mech Syst Signal Process 20(7):1483–1510

John Burt Associates Limited (2011) UK offshore commercial air transport helicopter safety record (1981–2010). Oil & Gas UK. http://oilandgasuk.co.uk/wp-content/uploads/2015/05/HS027.pdf . Accessed 3 June 2016

Joint Strike Fighter Program Office (2016) Joint strike fighter PHM vision. https://www.phmsociety.org/sites/phmsociety.org/files/PHMvision.pdf. Accessed 3 June 2016

Julier SJ, Uhlmann JK (2004) Unscented filtering and nonlinear estimation. Proc IEEE 92(3): 401–422

Kalman RE (1960) A new approach to linear filtering and prediction problems. Trans ASME J Basic Eng 82:35–45

Leão BP, Yoneyama T, Rocha GC (2008) Prognostics performance metrics and their relation to requirements, design, verification and cost-benefit. In: Paper presented at the international conference on prognostics and health management, Denver, Colorado, USA, 6–9 Oct 2008

Lee J, Wu F, Zhao W et al (2014) Prognostics and health management design for rotary machinery systems—reviews, methodology and applications. Mech Syst Signal Process 42(1):314–334

Li D, Wang W, Ismail F (2013) Enhanced fuzzy-filtered neural networks for material fatigue prognosis. Appl Soft Comput 13(1):283–291

Li YG, Nikitsaranont P (2008) Gas path prognostic analysis for an industrial gas turbine. Insight Non-Destruct Testing Condition Monitoring 50(8):428–435

Liao L, Köttig F (2014) Review of hybrid prognostics approaches for remaining useful life prediction of engineered systems, and an application to battery life prediction. IEEE Trans Reliab 63(1):191–207

Liao L, Lee J (2010) Design of a reconfigurable prognostics platform for machine tools. Expert Syst Appl 37:240–252. doi:10.1016/j.eswa.2009.05.004

Liu A, Dong M, Peng Y (2012) A novel method for online health prognosis of equipment based on hidden semi-Markov model using sequential Monte Carlo methods. Mech Syst Signal Process 32:331–348

Kao A, Lee J, Edzel E et al (2011) iFactory cloud service platform based on IMS tools and servolution. In: Paper presented at the world congress on engineering asset management, Cincinnati, Ohio, USA, 3–5 Oct 2011

Mackay DJC (1998) Introduction to Gaussian processes. NATO ASI Series F Comput Syst Sci 168:133–166

Murakami T, Saigo T, Ohkura Y et al (2002) Development of vehicle health monitoring system (VHMS/WebCARE) for large-sized construction machine. Komatsu's Technical Report 48 (150):15–21

Pandey MD, Noortwijk JMV (2004) Gamma process model for time-dependent structural reliability analysis. In: Paper presented at the second international conference on bridge maintenance, safety and management, Kyoto, Japan, 18–22 Oct 2004

Paris PC, Erdogan F (1963) A critical analysis of crack propagation laws. ASME J Basic Eng 85:528–534

PCoE (2016) Prognostics center of excellence. http://ti.arc.nasa.gov/tech/dash/pcoe. Accessed 3 June 2016

Peng Y, Dong M, Zuo MJ (2010) Current status of machine prognostics in condition-based maintenance: a review. Int J Adv Manuf Technol 50(1–4):297–313

Pettit CD, Barkhoudarian S, Daumann AG (1999) Reusable rocket engine advanced health management system architecture and technology evaluation—summary. In: Paper presented at the 35th AIAA/ASME/SAE/ASEE joint propulsion conference and exhibit, Los Angeles, California, USA, 20–24 June 1999

PHM Society (2009) The prognostics and health management society. http://www.phmsociety.org. Accessed 3 June 2016

Ristic B, Arulampalam S, Gordon N (2004) Beyond the Kalman filter: particle filters for tracking applications. IEEE Aerosp Electron Syst Mag 19(7):37–38

Saha B, Celaya J, Wysocki P et al (2009) Towards prognostics for electronics components. In: Paper presented at the IEEE aerospace conference, Big Sky, Montana, USA, 7–14 March 2009

Saxena A, Celaya J, Saha B et al (2009) On applying the prognostic performance metrics. In: Paper presented at the annual conference of the prognostics and health management society, San Diego, California, USA, 27 Sep–1 Oct 2009

Scheuren WJ, Caldwell KA, Goodman GA et al (1998) Joint strike fighter prognostics and health management. In: Paper presented at the AIAA joint propulsion conference, Cleveland, Ohio, 12–15 July 1998

Seeger M (2004) Gaussian processes for machine learning. Int J Neural Syst 14(2):69–106

Sheppard JW, Kaufman MA, Wilmer TJ (2009) IEEE standards for prognostics and health management. Aerospace Electr Syst Mag IEEE 24(9):34–41

Si XS, Wang W, Hu CH et al (2011) Remaining useful life estimation—a review on the statistical data driven approaches. Eur J Oper Res 213:1–14

Si XS, Wang W, Hu CH et al (2013) A Wiener process-based degradation model with a re-cursive filter algorithm for remaining useful life estimation. Mech Syst Signal Process 35(1–2): 219–237

Siegel D, Edzel L, AbuAli M et al (2009) A systematic health monitoring methodology for sparsely sampled machine data. In: Paper presented at the US society for machinery failure prevention technology (MFPT) conference, Dayton, Ohio, 28–30 Apr 2009

Siegel D, ZhaoW Lapira E et al (2014) A comparative study on vibration-based condition monitoring algorithms for wind turbine drive trains. Wind Energy 17(5):695–714

Sikorska JZ, Melinda H, Lin M (2011) Prognostic modelling options for remaining useful life estimation by industry. Mech Syst Signal Process 25(5):1803–1836

Sodemann A, Li Y, Lee J et al (2006) Data-driven surge map modeling for centrifugal air compressors. In: Paper presented at the ASME international mechanical engineering congress and exposition, Chicago, Illinois, 5–10 Nov 2006

Space Daily (2003) Prognosis program begins at DARPA. A newspaper report, Space Daily. http://www.spacedaily.com/news/materials-03zv.html. Accessed 3 June 2016

Sun B, Zeng S, Kang R et al (2012) Benefits and challenges of system prognostics. IEEE Trans Reliab 61(2):323–335

Tipping ME (2001) Sparse Bayesian learning and the relevance vector machine. J Machine Learning Res 1:211–244

UTC Aerospace Systems (2013) Health and Usage Management System (HUMS). http://utcaerospacesystems.com/cap/systems/sisdocuments/Health%20and%20Usage%20Management%20Systems%20(HUMS)/Health%20and%20Usage%20Management%20Systems%20(HUMS).pdf. Accessed 3 June 2016

Vachtsevanos G, Lewis F, Roemer M et al (2006) Intelligent fault diagnosis and prognosis for engineering systems. Wiley, Hoboken

VerWey J (2009) Airplane product strategy during a time of market transition. Boeing: e-newsletter June 2009. http://www.boeingcapital.com/p2p/archive/06.2009. Accessed 3 June 2016

Vichare N, Rodgers P, Eveloy V et al (2004) In-situ temperature measurement of a notebook computer—a case study in health and usage monitoring of electronics. IEEE Trans Device Mater Reliab 4(4):658–663

Vogl GW, Weiss BA, Donmez MA (2014) Standards related to prognostics and health management (PHM) for manufacturing. National Institute of Standards and Technology. http://dx.doi.org/10.6028/NIST.IR.8012. Accessed 3 June 2016

Wang W, Scarf PA, Smith MAJ (2000) On the application of a model of condition based maintenance. J Oper Res Soc 51(11):1218–1227

Wang H, Lee J, Ueda T et al (2007) Engine health assessment and prediction using the group method of data handling and the method of match matrix—autoregressive moving average. In: Paper presented at the ASME Turbo Expo, Montreal, Canada, 14–17 May 2007

Yin S, Ding S, Zhou D (2016) Diagnosis and prognosis for complicated industrial systems—Part I. IEEE Trans Industr Electron 63(4):2501–2505

Zio E, Maio FD (2010) A Data-driven fuzzy approach for predicting the remaining useful life in dynamic failure scenarios of a nuclear system. Reliab Eng Syst Saf 95:49–57

Chapter 2
Tutorials for Prognostics

2.1 Introduction

The performance of many engineering systems is gradually degraded, and eventually, the systems will fail under repeated usage conditions. Consider that a through-the-thickness center crack exists in an infinite plate under mode I fatigue loading condition. In aircraft structures, for example, this corresponds to a fuselage panel under repeated pressurization loadings (see Fig. 2.1), which is the main cause of fatigue failure. One flight corresponds to one cycle of loading, and cracks grow as the number of cycles increases. This is the major source of fatigue crack growth for fuselage panels. When the safety of aircraft structure is considered as a performance, it is gradually degraded as cracks grow. In this context, the crack is considered as damage, and the structural performance is degraded as the damage increases. That is, the structure is in a healthy state when there is no crack. Due to repeated loadings, micro-size cracks are formed and grow. As the cracks grow, damage increases and the structural performance is gradually degraded. Eventually, the structure will fail when the damage grows beyond a certain threshold.

In damage tolerance design, it is allowed to have damage in the system as long as it can be monitored and controlled so that it may not cause a system level failure. To avoid catastrophic failure, cracks should be inspected and monitored, and panels are required to be repaired or replaced before cracks grow beyond a certain level of threshold. The objective of prognostics is to predict *remaining cycles before the damage grows beyond the threshold (called remaining useful life, RUL)*. In order to predict the RUL, prognostics utilize the measured damage levels up to the current cycle.

The graph in Fig. 2.1 illustrates the prognostics process. The circles and squares are degradation/damage data such as, crack size, wear volume, and spall size, measured at different cycles. Since the degradation data cannot be directly measured in most cases, health monitoring data are obtained from sensors/actuators in the form of electrical or vibration signals, acoustic waves, thermography, etc. Then

N.-H. Kim et al., *Prognostics and Health Management of Engineering Systems*,
DOI 10.1007/978-3-319-44742-1_2

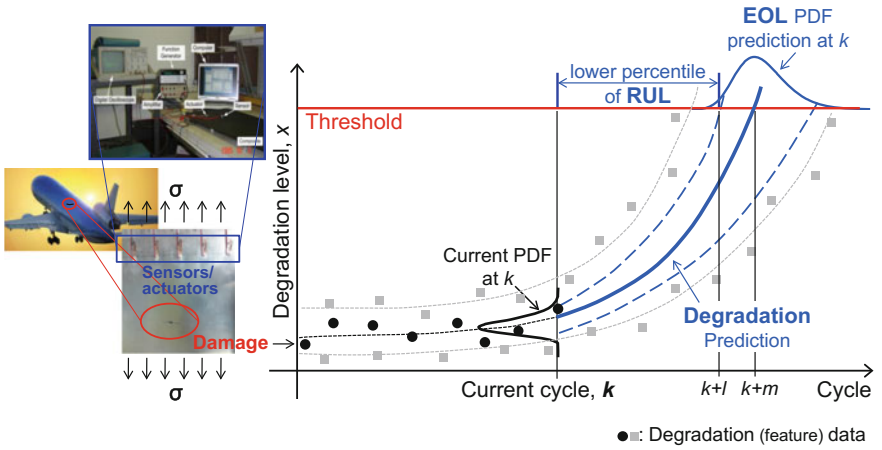


Fig. 2.1 Illustration of prognostics concept

these signals are converted into degradation data based on signal processing techniques, which is called featured data and will be discussed in Chap. 7. For example, in the case of crack size data, the actual measurements are transient elastic waves, and using calibration against known crack sizes, it is possible to estimate the crack size from wave signals.

In the figure, the circles represent degradation data up to the current cycle of the current system that are of interest to predict the RUL. These data are referred to as a prediction set in the text. The grey squares are degradation data from similar systems with the current one until the threshold (e.g., data from the same type of airplanes, or from the same airplane but different panels, which connotes that usage conditions can be different from the current one.), which are called training data and help to increase the prediction accuracy of the current one. In this text, these data are referred to as training set. Based on these data, future behavior of degradation is predicted as the solid curve in the figure.

The process of predicting the future behavior of degradation includes numerous sources of uncertainty, such as variability in material properties, data measurement noise/bias, current/future loading conditions, and not the least, the predicting process itself. Therefore, it is natural to consider the predicted degradation as a statistical distribution (the distribution is shown in the vertical line at the current cycle k in the figure), whose prediction interval is shown as the dashed blue curves in the figure. That is, even if degradation is measured as a single point at the current cycle, the degradation is represented as a distribution due to uncertainty.

For the purpose of prognostics, it is a failure of the system when the degradation level goes beyond the threshold (the red horizontal line in the figure). Due to uncertainty, the time to reach the threshold (end-of-life, EOL) is uncertain and can be represented as a probability distribution (blue bell-shaped curve). Then, for a

given time, the area under the distribution up to the time is considered as a failure probability or probability of failure. The failure probability increases as cycle increases. For example, the portion of the degradation distribution beyond the threshold (failure probability) at $k+m$ cycles is greater than one at $k+l$ cycles ($l > m$).

The end-of-life (EOL) is the corresponding cycle when the degradation reaches to the threshold and the maintenance should be ordered for the system. Since the degradation is a distribution, the cycle reaching to the threshold is not a deterministic value. Therefore, EOL is also a distribution because of the same sources of uncertainty, which is shown as the distribution on the threshold in the figure. The RUL, the remaining time to the maintenance from the current time, is also predicted as a distribution by subtracting the current cycle from the EOL distribution, as

$$t_{\mathrm{RUL}} = t_{\mathrm{EOL}} - t_k$$

The lower bound of RUL is considered as the maintenance time for a conservative purpose. *In a word, prognostics is to predict future behavior of degradation and the RUL of the system based on the measured data up to the current time.*

In general, prognostics methods can be categorized into two: physics-based and data-driven approaches. There are three main differences between the two approaches; (1) availability of a physical model that describes the behavior of damage, (2) availability of field operating conditions, and (3) damage degradation data from similar systems.

Physics-based approaches assume that a physical model describing the behavior of degradation is available with usage conditions such as loading information. Fatigue crack growth is the most common example used in this approach since the physical models are relatively well developed compared to other failure mechanisms. Consider again the crack on a plate ignoring the effect of finite plate size and the curvature of the plate in Fig. 2.1. When the stress range due to the pressure differential is $\Delta\sigma$, the rate of damage growth can be written using the Paris model (Paris and Erdogan 1963) as

$$\frac{\mathrm{d}a}{\mathrm{d}N} = C\left(\Delta\sigma\sqrt{\pi a}\right)^m \tag{2.1}$$

which can be rewritten by integrating it and solving for a as

$$a = \left[N \cdot C\left(1 - \frac{m}{2}\right)\left(\Delta\sigma\sqrt{\pi}\right)^m + a_0^{1-\frac{m}{2}}\right]^{\frac{2}{2-m}}, \tag{2.2}$$

where a_0 is the initial half crack size, a is the half crack size at the number of cycles N, and m, C are the model parameters. The model parameters govern the behavior of crack growth, which can be identified based on measured crack sizes at certain

interval of cycles with given loading information, $\Delta\sigma$. The crack size at future cycles can be predicted by substituting the identified parameters to the Paris model with future loading conditions. Since identifying model parameters is the most important step, algorithms of physics-based approaches can be considered as a parameter estimation method. The simplest method will be introduced with a simple example in Sect. 2.2, and more advanced methods will be discussed in Chap. 4.

Data-driven approaches can be considered when physical models are not available or the failure phenomenon is too complex to be expressed as a model. Now let us assume that degradation data only come from the dots in Fig. 2.1 and no physical model is available. In this case, no one may know what the future behavior would be without any information about degradation behavior with just a small number of data. Therefore, several sets of run-to-failure data (shown as the squares in the figure) are usually required to identify the degradation characteristics in data-driven approaches, which are called *training data*. In contrast with physics-based approaches identifying model parameters, there are a great variety of data-driven methods to use information from training data. One simple way will be introduced in Sect. 2.2.3, and typical algorithms will be discussed in Chap. 5.

This chapter provides a prognostics tutorial with a MATLAB code using simple examples. To simplify the process, uncertainty sources in prediction results are not considered at first. Therefore, degradation behavior and RUL at each cycle will be predicted as deterministic values rather than distributions. After the prognostics process is explained in terms of deterministic approach, probabilistic prediction is performed by considering the uncertainty in degradation data.

This chapter is organized as follows: in Sect. 2.2, degradation behaviors are predicted based on physics-based and data-driven approaches using least squares method; in Sect. 2.3, RUL is predicted and its results are evaluated based on the prognostics metrics; in Sect. 2.4, prognostics is performed by considering noise in degradation data, and in Sect. 2.5, issues in practical prognostics are briefly addressed, followed by exercises in Sect. 2.6.

2.2 Prediction of Degradation Behavior

2.2.1 Least Squares Method

In this section, a simple example of prognostics is introduced for both physics-based and data-driven approaches. Before we discuss about the two prognostics approaches, the least squares (LS) method (Bretscher 1995) is explained first, which is to find unknown parameters (or coefficients) by minimizing the sum of square errors (SS_E) between measured data and simulation outputs from the model/function.

Let y_k denote the measured data of degradation (e.g., crack size) at time index k and z_k the corresponding simulation output at the same time. The difference

between y_k and z_k is defined as error ε_k. Therefore, the relationship between measured data and simulation output can be written as

$$y_k = z_k + \varepsilon_k \tag{2.3}$$

In general, the error ε_k can represent measurement error in y_k as well as the error in simulation output z_k. For the moment, however, we assume that the simulation model is correct and the error only comes from measurement. We further assume that the measurement error does not include bias but noise, which is randomly distributed with the mean being zero; that is, the noise is unbiased.

Let us assume that the simulation model $z(t;\boldsymbol{\theta})$ is a linear function of input variable t (e.g., cycles) as

$$z(t;\boldsymbol{\theta}) = \theta_1 + \theta_2 t, \quad \boldsymbol{\theta} = \{\, \theta_1 \quad \theta_2 \,\}^{\mathrm{T}}, \tag{2.4}$$

where $\boldsymbol{\theta}$ is a vector of unknown parameters to be identified using the degradation data. The notation $z(t;\boldsymbol{\theta})$ is used to emphasize the fact that the simulation model takes time t as input and depends on its parameter $\boldsymbol{\theta}$. If there is no measurement error, then only two data points will be sufficient to identify unknown parameters. However, due to measurement error, these parameters are determined so that the sum of errors ε_k at all data points is minimized. This is the same as conventional regression. Since errors can be positive or negative, it is better to minimize the sum of square errors.

Let us consider the case that there are n_y data points, where the degradation is measured. The pair of input variable and measured degradation at data points is denoted as (t_k, y_k), $k = 1, \ldots, n_y$. A vector of measured degradation data can be denoted as $\mathbf{y} = \{\, y_1 \quad y_2 \quad \ldots \quad y_{n_y} \,\}^{\mathrm{T}}$. In the same way, the simulation model can be evaluated at data points as $z(t_k;\boldsymbol{\theta}) = z_k = \theta_1 + \theta_2 t_k$. The collected simulation outputs at all data points can be expressed as

$$\mathbf{z} = \begin{Bmatrix} z_1 \\ z_2 \\ \vdots \\ z_{n_y} \end{Bmatrix} = \begin{bmatrix} 1 & t_1 \\ 1 & t_2 \\ \vdots & \vdots \\ 1 & t_{n_y} \end{bmatrix} \begin{Bmatrix} \theta_1 \\ \theta_2 \end{Bmatrix} = \mathbf{X}\boldsymbol{\theta}, \tag{2.5}$$

where $\mathbf{X}$ is called the design matrix.

With the vectors of measured data and simulation outputs, the vector of errors in Eq. 2.3 can be defined as $\mathbf{e} = \{\, \varepsilon_1 \quad \varepsilon_2 \quad \ldots \quad \varepsilon_{n_y} \,\}^{\mathrm{T}} = \mathbf{y} - \mathbf{z}$. The sum of square errors can be defined in the following vector operation:

$$\mathrm{SS_E} = \mathbf{e}^{\mathrm{T}}\mathbf{e} = \{\mathbf{y} - \mathbf{z}\}^{\mathrm{T}}\{\mathbf{y} - \mathbf{z}\} = \{\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\}^{\mathrm{T}}\{\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\} \tag{2.6}$$

Note that the above $\mathrm{SS_E}$ is a quadratic function of parameters. Therefore, its minimum can be found from the stationary condition of $\mathrm{SS_E}$ with respect to $\boldsymbol{\theta}$ as

$$\frac{d(SS_E)}{d\boldsymbol{\theta}} = 2\left[\frac{d\mathbf{e}}{d\boldsymbol{\theta}}\right]^T \mathbf{e} = 2\mathbf{X}^T\{\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\} = \mathbf{0} \tag{2.7}$$

Therefore, the parameters that minimize SS_E can be obtained by solving the above equation for $\boldsymbol{\theta}$, which is called the estimated parameters as

$$\hat{\boldsymbol{\theta}} = [\mathbf{X}^T\mathbf{X}]^{-1}\{\mathbf{X}^T\mathbf{y}\} \tag{2.8}$$

The estimated parameters are used to predict the degradation level, $z(t;\hat{\boldsymbol{\theta}})$ at new time t from Eq. 2.4. The above process is called least squares method or linear regression.

In order to generalize the above derivations, we will use the following convention: $\mathbf{y}$ is a $n_y \times 1$ vector of measured data, $\boldsymbol{\theta}$ is a $n_p \times 1$ vector of the parameters associated with the model, and $\mathbf{X}$ is a $n_y \times n_p$ design matrix.

A sample MATLAB code for LS is given in **[LS]**, and it will be explained how to use it for different approaches in the following subsections.[1] Blanks in the code will be filled for a specific example.

[LS]: MATLAB code for Least Squares

```
%%Identify parameters (theta)
y=[ ]';                              % a vector of measured data
x=[ ]';                               % a vector/matrix of input
X=[ ];                          % a design matrix, e.g., Eq. 2.5
thetaH=(X'*X)\(X'*y)                                    % Eq.2.8
% This is the same as: thetaH=regress(y,X);

%%Degradation prediction at xNew
xNew=[ ]';
XNew=[ ];          % same form as X, but use xNew instead of x
zH=XNew*thetaH;                                  % Eq. 2.4 or 2.5

%%RUL prediction
thres=[ ];                                     % threshold level
currt=[ ];                                       % current cycle
syms xEOL
XEOL=[ ];          % same form as X, but use xEOL instead of x
eolFuc=thres-XEOL*thetaH;                              % EOL func.
eol=double(solve(eolFuc,'Real',true));
rul=min(eol(eol>=0))-currt
```

[1] All MATLAB codes in this book can be found in the companion website http://www2.mae.ufl.edu/nkim/PHM/. The naming convention is functionname.m. For example, the MATLAB code for least squares method is LS.m.

2.2.2 When a Degradation Model Is Available (Physics-Based Approaches)

Since damage is a part of physical phenomena, many researchers have tried to model the evolution of damage; i.e., degradation, using physical models, such as fatigue crack growth (An et al. 2012; Coppe et al. 2010; Sankararaman et al. 2009), wear of mechanical joints (An et al. 2011), recharging capability of battery (Dalal et al. 2011), etc. These models are developed based on understandings in physical phenomena through numerous test data. The advantage of having a degradation model is that we can expect the behavior of damage. However, since models are usually developed under idealized conditions with many assumptions, its applicability is limited. In addition, when damage is caused by interactions between many systems, it is difficult to develop a physical model that fully describes the degradation process. Since models are usually not perfect, it is possible to include the effect of model error. However, we will only consider the case that the model error is ignorable. We will discuss about the effect of model error in Chaps. 4 and 6.

2.2.2.1 Problem Definition

When a degradation model that describes the level of damage is available, the measured data can be used to identify (or calibrate) model parameters. Once the model parameters are identified, they can be used to predict the future behavior of the damage. In order to show how to identify model parameters of a degradation model, let us consider the following form of degradation model:

$$z(t;\boldsymbol{\theta}) = \theta_1 + \theta_2 L t^2 + \theta_3 t^3, \quad \boldsymbol{\theta} = \{\,\theta_1 \quad \theta_2 \quad \theta_3\,\}^{\mathrm{T}}, \tag{2.9}$$

where $z(t;\boldsymbol{\theta})$ is the degradation level (e.g., crack size) at time cycle t, $L = 1$ is a constant representing loading condition, and $\boldsymbol{\theta}$ is a vector of model parameters. Since a physical model is given with loading condition, the future degradation can be predicted after $\boldsymbol{\theta}$ is identified based on the degradation data, which is a physics-based approaches.

For the degradation model given in Eq. 2.9, let us assume that measured degradation data are given as in Table 2.1. Under the assumption that the model is perfect and the data do not have any noise, the data in Table 2.1 are generated by using the true values of $\boldsymbol{\theta}_{\text{true}} = \{\,5 \quad 0.2 \quad 0.1\,\}^{\mathrm{T}}$. However, the true values of parameters are only used to generate the data and are not used in the fitting process.[2] The objective is to identify model parameter $\boldsymbol{\theta}$ that fits the data best. After

[2]In this text, many data are generated from the assumed true parameters. This is useful to check if the identified parameters are accurate or not, compared to the true parameters.

Table 2.1 Degradation data given up to four cycles

Time index, k	1	2	3	4	5
Input, t_k (cycles)	0	1	2	3	4
Data, y_k	5.0	5.3	6.6	9.5	14.6

estimating the model parameters, the accuracy can be evaluated by comparing with the true values.

2.2.2.2 Parameter Estimation and Degradation Prediction

In order to use the MATLAB code **[LS]**, the pairs (t_k, y_k) of five data points are implemented as

```
set in [LS]
y=[5  5.3  6.6  9.5  14.6]';
x=[0  1  2  3  4]';
```

Also, the vector of degradation model (Eq. 2.5) at all data points can be written using Eq. 2.9, as

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_5 \end{bmatrix} = \begin{bmatrix} 1 & Lt_1^2 & t_1^3 \\ 1 & Lt_2^2 & t_2^3 \\ \vdots & \vdots & \vdots \\ 1 & Lt_5^2 & t_5^3 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} = \mathbf{X}\boldsymbol{\theta} \tag{2.10}$$

The design matrix $\mathbf{X}$ can be implemented as

```
set in [LS]
L=1;
X=[ones(length(x),1)  L*x.^2  x.^3];
```

Now since all variables required to use MATLAB code **[LS]** are available, the unknown model parameters can be identified using Eq. 2.8 as

```
in [LS]
thetaH=(X'*X)\(X'*y)
```

Finally, we obtain $\hat{varvectheta} = \{trix5.00.20.1\}^{\mathrm{T}}$, which is identical to the true values, and will predict the future prediction accurately. This is expected because the data were generated from the same model with the true model parameters. This case the case when both the model and data are accurate.

The major advantage of physics-based approach is that once the model parameters are identified, it is trivial to predict the behavior of damage in the future by providing future time cycles and loading conditions to the model. For example, the model parameters $\hat{varvectheta} = \{trix5.00.20.1\}^{\mathrm{T}}$ were identified using measured data given up to $t_5 = 4$ cycle in Table 2.1. If we want to predict the degradation behavior from $t = 4$ to $t = 15$ cycles, the following MATLAB code can be implemented in **[LS]** as

```
set in [LS]
xNew=[4:0.5:15]';
XNew=[ones(length(xNew),1)  L*xNew.^2  xNew.^3];
```

The variable, xNew, is used for prediction time. The red-dashed curve in Fig. 2.2 represents the degradation behavior predicted using five data (the dots), which is obtained by

```
in [LS]
zH=XNew*thetaH;
```

The prediction results (red dashed curve) can be compared with the true degradation curve (black solid curve) in Fig. 2.2, which can be obtained with the true parameter values of $varvectheta_{\mathrm{true}} = \{trix50.20.1\}^{\mathrm{T}}$ and Eq. 2.9 (the MATLAB code is given **[Fig. 2.2]**). Since the identified parameters are exactly the same as the true one, two degradation curves are overlapped each other. The following MATLAB code can be used to plot Fig. 2.2:

```
[Fig. 2.2]
thetaTrue=[5; 0.2; 0.1];
xTrue=[0:0.5:15]';
XTrue=[ones(length(xTrue),1)  L*xTrue.^2  xTrue.^3];
zTrue=XTrue*thetaTrue;

set(gca,'fontsize',18)
plot(x,y,'.k','markersize',30);
hold on; plot(xTrue,zTrue,'k','linewidth',3);
plot(xNew,zH,'--r','linewidth',3);
xlabel('Cycles'); ylabel('Degradation level');
legend('Data','True','Prediction',2);
```

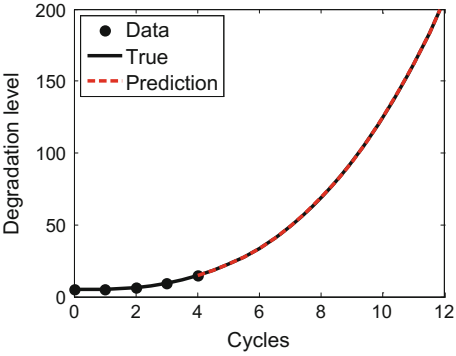


Fig. 2.2 Prediction of degradation behavior when degradation model is given with exact data

2.2.2.3 Effect of Noise in Data

In the previous example, data were generated from the exact model without noise, and thus, the parameters were identified exactly. However, degradation data almost always have a certain level of noise, which is called measurement error. When data have noise, it is not guaranteed to identify the exact model parameters. There will be an error in identifying model parameters. Since the noise is random, the identified parameters will be different if the fitting process is repeated with different sets of data. In order to show the effect of noise, uniformly distributed noise between −5 and 5 are added to the data in Table 2.1, which is shown in Table 2.2 (see the case of $L = 1$). The data in Table 2.2 are a particular realization due to the random noise. A similar process of identifying model parameters using **[LS]** yields a new vector of parameters $\hat{varvectheta} = \{trix4.28{-}0.290.25\}^{\mathrm{T}}$, which now are different from the true parameters. Figure 2.3a shows the prediction result of degradation behavior using data in Table 2.2 for L = 1. It is shown that prediction curve is different from the exact one. Due to the randomness of noise, if the above process is repeated with different sets of data but the same level of noise, different sets of degradation levels can be obtained. Using these different degradation levels, it is possible to obtain a probability distribution of degradation level at a given time cycle (refer to exercise

Table 2.2 Degradation data at three loading conditions with noise from $U(-5, 5)$

Time index, k	1	2	3	4	5	6	7	8	9	10	11
Input, x_k (cycles)	0	1	2	3	4	5	6	7	8	9	10
Data, y_k at $L = 1$	6.99	2.28	1.91	11.94	14.60	22.30	37.85	50.20	70.18	97.69	128.05
Data, y_k at $L = 0.5$	0.46	1.17	9.43	10.55	11.17	24.50	25.54	43.59	61.42	88.66	117.95
Data, y_k at $L = 2$	1.87	5.40	6.86	12.76	19.89	30.05	38.76	60.70	83.35	106.93	141.19

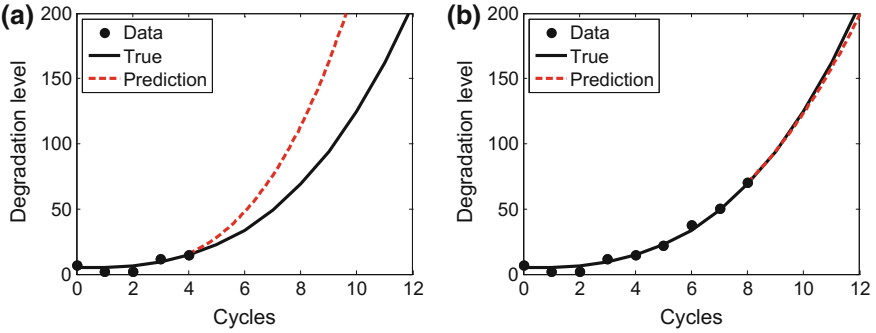


Fig. 2.3 Prediction of degradation behavior when degradation model is given with noisy data **a** five data **b** nine data

problem, P2.5). Since the random noise has a mean of zero, the error in prediction can be reduced when more data are used. For example, Fig. 2.3b shows the prediction curve with nine data points, whose prediction error is much smaller than that of five data case in Fig. 2.3a.

Example 2.1 Least squares method for determining model parameters Find model parameters in Eq. 2.9, (a) using data in Table 2.1 and (b) using data in Table 2.2 at L = 1.

Solution: (a) For no-noise data, the following vectors and matrices are obtained:

$$[\mathbf{X}] = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 4 & 8 \\ 1 & 9 & 27 \\ 1 & 16 & 64 \end{bmatrix}, \quad \{\mathbf{X}^{\mathrm{T}}\mathbf{y}\} = \begin{Bmatrix} 41.0 \\ 350.8 \\ 1249.0 \end{Bmatrix}$$

$$[\mathbf{X}^{\mathrm{T}}\mathbf{X}] = \begin{bmatrix} 5 & 30 & 100 \\ 30 & 354 & 1300 \\ 100 & 1300 & 4890 \end{bmatrix}$$

Therefore, the model parameter in Eq. 2.8 can be calculated by

$$\hat{\boldsymbol{\theta}}_{\text{no-noise}} = [\mathbf{X}^{\mathrm{T}}\mathbf{X}]^{-1}\{\mathbf{X}^{\mathrm{T}}\mathbf{y}\} = \begin{Bmatrix} 5.0 \\ 0.2 \\ 0.1 \end{Bmatrix}$$

Note that these identified parameters are identical to the true parameters because the data are generated from the same model.

(b) For data with noise in Table 2.2, the only vector that is changed is $\{\mathbf{X}^{\mathrm{T}}\mathbf{y}\}$:

$$\{\mathbf{X}^{\mathrm{T}}\mathbf{y}\} = \begin{Bmatrix} 37.7 \\ 351.0 \\ 1274.3 \end{Bmatrix}$$

The matrices $[\mathbf{X}]$ and $[\mathbf{X}^{\mathrm{T}}\mathbf{X}]$ remain unchanged. Therefore, the model parameter in Eq. 2.8 can be calculated by

$$\hat{\boldsymbol{\theta}}_{\text{with-noise}} = [\mathbf{X}^{\mathrm{T}}\mathbf{X}]^{-1}\{\mathbf{X}^{\mathrm{T}}\mathbf{y}\} = \begin{Bmatrix} 4.28 \\ -0.29 \\ 0.25 \end{Bmatrix}$$

Note that due to noise, the identified parameters are different from the true parameters.

Example 2.2 When an incorrect loading condition is given

(a) Repeat Example 2.1(a) by assuming the loading condition is given as $L = 2$. (b) Compare the degradation prediction with $L = 2$ to the true model ($L = 1$, $\boldsymbol{\theta}_{\text{true}} = \{5 \quad 0.2 \quad 0.1\}^{\mathrm{T}}$).

Solution:

(a) The second column of $\mathbf{X}$ in Example 2.1 is changed by applying $L = 2$ to the second order term in Eq. 2.9 as

$$[\mathbf{X}] = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 2 & 1 \\ 1 & 8 & 8 \\ 1 & 18 & 27 \\ 1 & 32 & 64 \end{bmatrix}$$

Therefore the model parameter can be calculated as

$$\hat{\boldsymbol{\theta}} = [\mathbf{X}^{\mathrm{T}}\mathbf{X}]^{-1}\{\mathbf{X}^{\mathrm{T}}\mathbf{y}\} = \begin{Bmatrix} 5.0 \\ 0.1 \\ 0.1 \end{Bmatrix}$$

Even if no-noise data are used, the identified parameters are not the same as the true ones because of the error in loading condition.

(b) The following figure can be obtained with the result in (a) by utilizing MATLAB code introduced previously.

```
% Example 2_2
L=2;                          % with a wrong loading condition
y=[5  5.3  6.6  9.5  14.6]';    % a vector of measured data
x=[0  1  2  3  4]';              % a vector/matrix of input
X=[ones(length(x),1)  L*x.^2  x.^3];      % a design matrix
thetaH=(X'*X)\(X'*y)                              % Eq.2.8
xNew=[0:0.5:15]';
XNew=[ones(length(xNew),1)  L*xNew.^2  xNew.^3];
zNew=XNew*thetaH;
%
LCorrect=1;                % with a correct loading condition
X=[ones(length(x),1)  LCorrect*x.^2  x.^3];
thetaCorrect=(X'*X)\(X'*y)
XCorrect=[ones(length(xNew),1)  LCorrect*xNew.^2  xNew.^3];
zCorrect=XCorrect*thetaCorrect;
%
set(gca,'fontsize',18)
plot(x,y,'.k','markersize',30);
hold on; plot(xNew,zCorrect,'k','linewidth',3);
plot(xNew,zNew,'--r','linewidth',3);
xlabel('Cycles'); ylabel('Degradation level');
legend('Data','\theta_{true} with L=1','\theta_{pred} with
L=2',2);
```

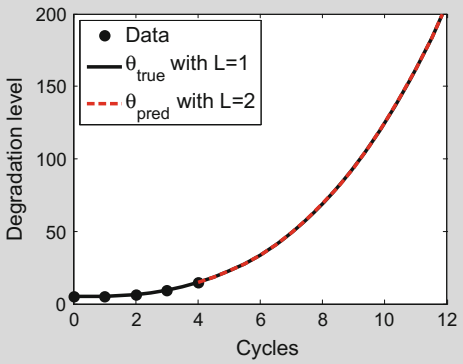


Fig. E2.1 Degradation prediction with incorrect loading condition

Note that the degradation prediction result under incorrect loading condition (red dashed curve) is exactly the same as the true behavior (black solid curve). This is because the model parameter θ_2 is correlated with the loading condition L, thereby model parameters are identified differently from the true ones under different loading condition. That is, $L\theta_2$ is the same for both results. The correlation issue in parameter identification will be discussed in more detail in Chap. 6.

2.2.3 When a Degradation Model Is NOT Available (Data-Driven Approaches)

When a physical model that describes the behavior of damage degradation is unavailable, the future behavior of damage has to be predicted using measured data. This happens when the damage degradation phenomenon is too complicated or when it is difficult to directly measure the damage. For example, in the case of the failure of bearing, even if the cause of damage starts from small defects in the race or in the rolling elements, it is very difficult to measure the crack sizes directly. Instead, the vibration of bearing assembly is measured using accelerometer, from which the level of damage is estimated. In such a case, it is true that the level of vibration is related to the level of damage, but it is difficult to make a physical relationship between the two because vibration can be caused by many different sources, such as misalignment or external excitations. More advanced algorithms for bearing prognostics will be introduced in Chap. 7.

2.2.3.1 Function Evaluation

Even if we want to predict the behavior of damage only using data, predictions still require a functional relationship between input variables and output degradation. In this case, however, the functional relationship does not have any physical meanings. Now we consider the case when the physical degradation model as in Eq. 2.9 is not available, which means that we need to build a relationship between input variables and output degradation using given data. For example, the data given in Table 2.2 has a single input variable (time cycle), which is also shown in Fig. 2.4. A typical way of making relationship between input variable and output is to assume a functional form: not a physical model but a pure mathematical function. For example, in the case of data given in Table 2.2, we can consider the following three different types of functional relationships:

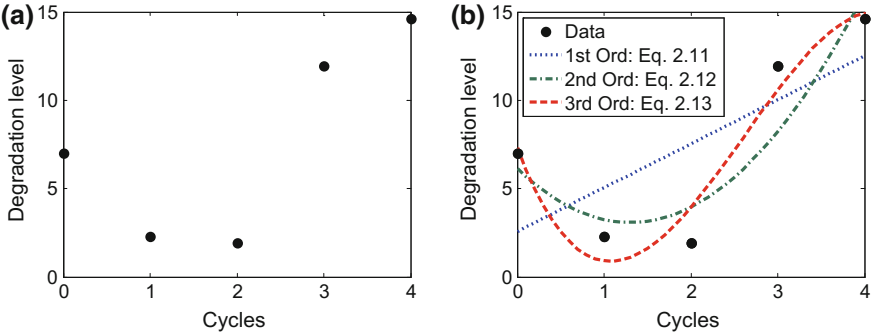


Fig. 2.4 Function fitting with five training data **a** given degradation data **b** function fitting results

$$z^{(1)} = \theta_1 + \theta_2 t, \tag{2.11}$$

$$z^{(2)} = \theta_1 + \theta_2 t + \theta_3 t^2, \tag{2.12}$$

$$z^{(3)} = \theta_1 + \theta_2 t + \theta_3 t^2 + \theta_4 t^3 \tag{2.13}$$

It is also noted that the above mathematical functions do not include loading conditions. It is obvious that the damage will evolve fast when the loading condition is severe. Therefore, if the loading condition is available, more accurate prediction is possible. But, for the moment, we only consider the case when the loading condition is not a part of mathematical functions.

Once a mathematical function is selected, the next step is to identify the unknown parameters in the function using given data, which is basically the same as the method used in Sect. 2.2.2. In general, data-driven approaches require training data obtained from similar systems. In this section, however, only previously measured data from the current system are used for the training purpose. The first five data from Table 2.2 when $L = 1$ are used for the training purpose (see Fig. 2.4a). The MATLAB code **`[LS]`** can be used to identify parameters (see Exercise P2.3). Figure 2.4b shows the fitting results using the identified parameters. The linear function shows a monotonic increase in degradation, but the quadratic and cubic functions show an initial decrease, which is difficult to explain physically (normally degradation is a monotonic function of time cycles). However, if a linear function is selected, then the future prediction will be significantly underestimated compared to the true degradation behavior in Fig. 2.2. Therefore, it would be hard to tell which function is better than the others. When the degradation model is not available, it is difficult to select an appropriate mathematical function as well as to identify parameters accurately due to the influence of noise in data. The quality of prediction depends on (1) the selection of function, (2) the number of data, and (3) the level of noise.

Possible ways of evaluating the quality of fitting for different mathematical functions are from the comparison between the function prediction and data, such as average and maximum errors, sum of squared errors, coefficient of determination, and cross-validation. Rigorous discussions are available in the literature of surrogate modeling (Kohavi 1995; Saed 2010). For simplicity, we only introduce the coefficient of determination, R^2, which is the ratio between the variation of function prediction to the variation of data as

$$R^2 = \frac{\mathrm{SS}_R}{\mathrm{SS}_T} = 1 - \frac{\mathrm{SS}_E}{\mathrm{SS}_T}, \tag{2.14}$$

$$\mathrm{SS}_R = \sum_{k=1}^{n_y} (z_k - \bar{y})^2, \quad \mathrm{SS}_T = \sum_{k=1}^{n_y} (y_k - \bar{y})^2, \quad \mathrm{SS}_E = \sum_{k=1}^{n_y} (y_k - z_k)^2 \tag{2.15}$$

where SS_T (the total sum of squares) is the variation of data with respect to the mean of data $\bar{y}$, SS_R (the regression sum of squares) is the variation of the function prediction z_k with respect to the mean of data, and SS_E (the residual sum of squares) is the sum of square of errors remaining after the fit. The total sum of squares is the sum of the regression sum of squares and the residual sum of squares, i.e., $\mathrm{SS}_T = \mathrm{SS}_R + \mathrm{SS}_E$, when the sum of y_k is equal to the sum of z_k.[3] The coefficient of multiple determination measures the fraction of the variation in the data that is captured by the function prediction. From the fact that an accurate function should have small error, R^2 close to one is considered an accurate function. However, the accuracy of the function estimated by R^2 only measures its accuracy in data points, which may not be related to the true accuracy of the function prediction. For example, if a cubic polynomial with four coefficients fits four data points, the polynomial can pass all four data points, which makes $\mathrm{SS}_E = 0$ and $R^2 = 1$, but that does not mean the polynomial is accurate at prediction points. The only true test to the predictive capability of a function is evaluating it at points not used in its construction.

The coefficient of determination, R^2, can be obtained from MATLAB using '`regress`' in **`[LS]`**:

```
[thetaH,~,~,~,stats]=regress(y,X);
R2=stats(1)
```

where the first component of the returned array `stats` contains the R-square statistic. The higher value means the better fitting with the given data. Having said that, R^2 is not directly used since it increases as the number of coefficients increase by fitting data more closely. This, however, does not mean a good prediction at other points.

[3]More specifically, the following condition is required: $\mathbf{y}^{\mathrm{T}}\bar{\mathbf{y}} = \mathbf{z}^{\mathrm{T}}\bar{\mathbf{y}}$, where $\bar{\mathbf{y}}$ is the constant vector all of whose elements are the mean data and $\mathbf{z}$ is a vector of model predictions at x_k.

Therefore, to prevent this phenomenon, the adjusted R^2 denoted as $\bar{R}^2$ is employed by penalizing the number of coefficients as

$$\bar{R}^2 = 1 - \left(1 - R^2\right) \frac{\left(n_y - 1\right)}{\left(n_y - n_p\right)}, \tag{2.16}$$

where n_y and n_p are, respectively, the number of data and coefficients. The results are listed in Table 2.3. Based on these results with five data, the third-order polynomial function is selected by expecting that this function with given data makes the best prediction of the future behavior. Since the data in data-driven approaches are used in determining the functional form and making a mathematical function to have a degradation feature, they are called training data.

In addition to the three functions showing quite different fitting results, the prediction beyond $t = 4$ may yield a much larger error. Figure 2.5 shows the prediction results up to time cycle $t = 10$ using the three polynomial functions in Fig. 2.4 along with the true degradation model. The difference between linear and quadratic polynomials was small in the range of training data, but now, the difference becomes huge in the prediction stage. When five data are used, the quality of fitting using the coefficient of determination showed that the cubic polynomial was the best fit, as shown in Table 2.3. However, in the prediction stage, the cubic polynomial shows the largest error. In fact, the cubic polynomial predicts the degradation decreases in the prediction stage even if the degradation is expected to monotonically increase. This happens because in the process of identifying parameters, the knowledge of monotonicity in degradation function is not used.

Table 2.3 $\bar{R}^2$ calculation results

	1st order: Eq. 2.11	2nd order: Eq. 2.12	3rd order: Eq. 2.13
$\bar{R}^2$ with five data	0.3071	0.6613	0.7482
$\bar{R}^2$ with nine data	–	0.9905	0.9887
$\bar{R}^2$ with 31 data	–	0.9589	0.9592

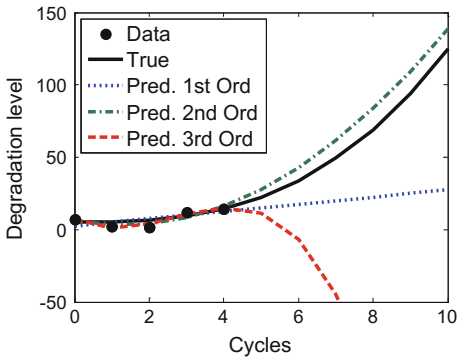


Fig. 2.5 Prediction of degradation behavior when five degradation data are given without a physical model

Therefore, when the degradation model is not available, the prediction far from the last measured data may be dangerous and possible to yield meaningless prediction results.

2.2.3.2 Overfitting

One important aspect in curve fitting is the relation between the number of data and the number of unknown coefficients in the function. Normally, it is expected that the number of data is much larger than that of unknown coefficients. In such a case, the least squares method tends to compensate for the noise in data and try to find the mean trend of the function. However, when the number of unknown coefficients increases, the least squares method tends to fit the noise, not the trend. This phenomenon is called *overfitting*. This is one of reasons that the cubic polynomial function does not show the best prediction result as shown in Fig. 2.5. As an extreme example, if a quartic polynomial is used to fit the five data points given in Fig. 2.4a, since the number of unknown coefficients and the number of data are the same, the polynomial perfectly fits all data points and the coefficient of determination becomes one. However, such a perfect fit does not mean perfect prediction at un-sampled points.

Overfitting is a modeling error which generally occurs when the proposed function is overly complex so that it fits noisy data (like the third-order function in this example), and when the function has no conformability with the data shape (Hawkins 2004). There are several techniques to avoid overfitting, such as cross-validation, regularization, early stopping and pruning (Wu and Lee 2011). These techniques try to explicitly penalize overly complex functions or to test the capability of the function to generalize by evaluating its performance on a set of data not used for training. However, these techniques are usually employed in the training stage, which does not guarantee good results in predicting at future cycles. Therefore, overfitting is usually prevented by two approaches in data-driven prognostics: (1) the behavior of degradation is expressed with a simple function, which will be explained in Chap. 5, and (2) more information, such as more training data and usage conditions, is used for reliable prognostics results.

2.2.3.3 Prognosis with More Training Data

Additional challenge in data-driven approaches is that removing overfitting does not always improve prediction accuracy at future time cycles. To use more data as a remedy for overfitting, assume nine data are given as in Fig. 2.3b. Since the trend of data clearly shows that the degradation behavior is no longer a linear function, the second- and third-order polynomial functions are considered. Using '`regress`' MATLAB function, these two functions are obtained as

$$z^{(2)} = 5.83 - 3.59t + 1.45t^2,$$
$$z^{(3)} = 5.99 - 3.93t + 1.56t^2 - 0.01t^3$$

Table 2.3 shows that both functions yield similar R^2 values. Indeed, the prediction results from these two functions are very similar as shown in Fig. 2.6, in which the prediction result is improved compared to ones using five data, but still less accurate than those from the physics-based approach in Fig. 2.3b.

One of the reasons that the prediction results of data-driven approaches were not as good as that from physics-based approaches is from the fact that the former does not have any information on the future behavior of degradation. That is, the selected function behaves similarly for the training region, but not for the prediction region. If a degradation history of identical systems up to failure is available, then that can greatly improve the prediction accuracy in data-driven approaches. In data-driven approaches, several sets of training data up to the failure are usually required to predict degradation behavior as accurate as the physics-based approaches. The best training data can be obtained from the identical system under identical usage condition. However, the data-driven approaches can utilize data from similar systems under different usage conditions.

In order to show the effect of training data from other systems on prediction accuracy, it is assumed that two sets of data are given, which are obtained under $L = 0.5$, 2 in Eq. 2.9 with the same level of noise (uniformly distributed between -5 and 5). These training data are shown as squares and the triangles in Fig. 2.7a. In addition to the two sets of training data, the nine data from the current system are given, which is also shown as dots in Fig. 2.7a (this is the same data in Fig. 2.6). These data are listed in Table 2.2, and a total 31 pairs of input x_k and data y_k are provided in `[LS]`, which is composed of two sets of 11 data and 9 data from the current system. Now using all 31 training data, R^2 and the degradation prediction results are shown in Table 2.3 and Fig. 2.7b, respectively. It is shown that the cubic function with higher values of R^2 makes the better results in degradation prediction in the case that additional training data sets are used. Additional sets of training data

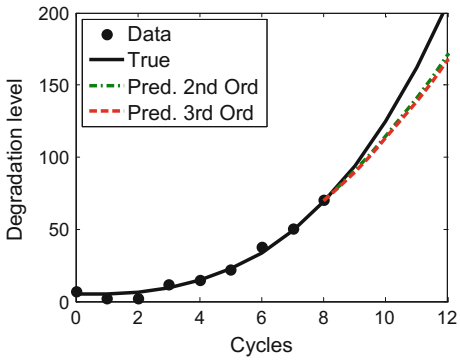


Fig. 2.6 Prediction of degradation behavior when nine degradation data are given without a physical model

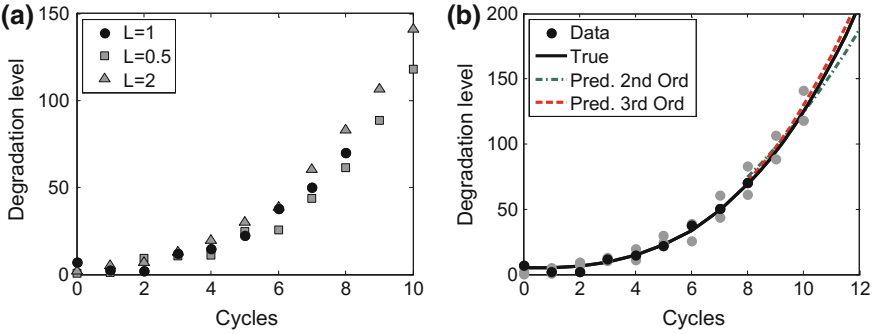


Fig. 2.7 Prediction of degradation behavior using two additional sets of training data without a physical model **a** training data **b** degradation predictions

can compensate for the absent of a physical model. Note that the reason for better results with additional training sets is that the degradation rate of the current system is in between the additional sets as well as they are close to each other. When there is a significant difference in degradation rate, loading information is additionally required to utilize the additional sets of training data to improve prediction accuracy, which will be discussed in Chap. 6.

2.3 RUL Prediction

Once the model parameters are identified in physics-based approaches or the mathematical function is trained in data-driven approaches, the model/function can be used to predict the remaining useful life (RUL), which is the remaining time until the degradation grows to a threshold. The threshold of degradation is determined so that the system is still safe but needs maintenance. Since determining the threshold depends on specific application with experience, we do not discuss about how to choose the threshold. Instead, we assume that the level of threshold is given and will discuss about how to predict RUL accurately. In this section, RUL prediction is performed based on the physics-based approach in Sect. 2.2.2. Also, prognostics metrics (Saxena et al. 2009) are introduced to evaluate RUL prediction results.

2.3.1 RUL

We used the physics-based degradation model in Eq. 2.9 to predict RUL. When data have uniformly distributed noise as in Table 2.2, the following parameters were identified: $\hat{\boldsymbol{\theta}} = \{\, 4.28 \quad -0.29 \quad 0.25 \,\}^{\mathrm{T}}$, which was different from the true parameters $\boldsymbol{\theta}_{\mathrm{true}} = \{\, 5.0 \quad 0.2 \quad 0.1 \,\}^{\mathrm{T}}$. Such an error in model parameters leads to an error

in predicting RUL. To illustrate the effect of the error in parameters on RUL prediction, it is assumed that the maintenance is ordered when the degradation level reaches 150. As an instance, from Fig. 2.3a, the current time cycle is 4, and the end-of-life (EOL), the cycle when the prediction result reaches 150, is 8.7 cycles. Therefore, RUL is predicted as 4.7 cycles (8.7(EOL) − 4(current cycle)), while the true RUL is 6.7 cycles (10.7(true EOL) − 4(current cycle)) (see Table 2.4).

In order to calculate the RUL, it is necessary to find the time cycle when the level of degradation reaches a threshold. The degradation model in Eq. 2.9 or Eq. 2.13 is given in such a way that the degradation can explicitly be calculated for a given cycle. However, calculating the time cycle when the degradation reaches a certain level is not straightforward as the relation is nonlinear and implicit. Therefore, in order to find the end-of-life, the following nonlinear equation has to be solved to find time cycle t_{EOL}:

$$y_{\text{threshold}} - z(t_{\text{EOL}}; \hat{\boldsymbol{\theta}}) = 0 \tag{2.17}$$

In general, the above nonlinear equation is solved numerically using Newton-Raphson iterative method. In MATLAB code **[LS]**, the function `solve` is used to find the time cycle t_{EOL} that satisfies the above relation, where the solution t_{EOL} is called the end-of-life (EOL). The remaining useful life (RUL) can be determined from $t_{\text{RUL}} = t_{\text{EOL}} - t_{\text{current}}$. Since $z(t; \hat{\boldsymbol{\theta}})$ is a monotonic function, the above equation will have a unique solution.

In order to calculate EOL, the MATLAB code **[LS]** is modified as shown in the following program list:

```
in [LS]
thres=150;
currt=4;
syms xEOL
L=1;
XEOL=[1  L*xEOL.^2  xEOL.^3];
eolFuc=thres-XEOL*thetaH; % EOL func.
eol=double(solve(eolFuc,'Real',true));
rul=min(eol(eol>=0))-currt
```

In the above code, the MATLAB code '`solve`' calculates a symbolic solution of algebraic equation with `xEOL` being symbolic variable for t_{EOL}. Since the

Table 2.4 Prediction of remaining useful life from data

Data	Table 2.1	Table 2.2
Parameters	[5.0, 0.2, 0.1]	[4.28, –0.29, 0.25]
Threshold	150	150
EOL (cycles)	10.7	8.7
RUL (cycles)	6.7	4.7

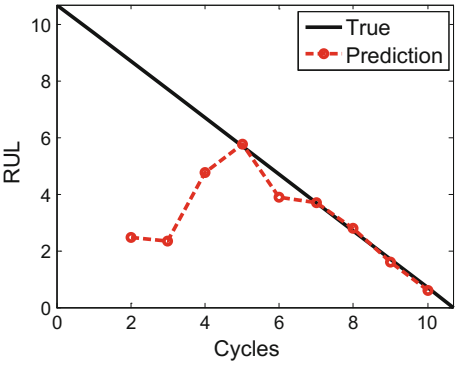


Fig. 2.8 RUL prediction at each cycle

degradation model is monotonic, Eq. 2.17 should have a single real solution. Since, however, the third-order polynomial function is employed in this example, it is possible to have more than one solution. Therefore, a minimum value of positive real numbers is considered as the EOL out of three possible solutions (see the last two lines). The RUL obtained from the above code is 4.76.

As shown aforementioned procedure, prognostics can be performed at every time cycle until the RUL becomes zero at which the maintenance has to be ordered. In an early time cycle, the prediction of model parameters and thus RUL might be inaccurate because of the lack of data. But, as more data are available the prediction results can be more accurate. Note that the RUL results at zeroth and first cycle are predicted as infinite because the number of data is smaller than that of unknown model parameters (proving this remains in the exercise problem, P2.9). Therefore, the results are shown from the second cycle, which is shown in Fig. 2.8. In Fig. 2.8, the black solid line represents the true RUL. The true RUL is a negative 45° line as the RUL decreases by one cycle at every cycle. The red-dashed line is the predicted RUL using the aforementioned procedure (data including later cycles are given in Table 2.2). Note that the prediction has significant error in early cycles because of inaccuracy in identifying model parameters. However, the prediction results converge to the true one after the seventh cycle.

Example 2.3 RUL prediction
Predict RUL at every measurement cycle using data in Table 2.2 when $L = 2$. The degradation model is given in Eq. 2.9 and the threshold is 150.

Solution:
Since the number of model parameters is three in Eq. 2.9, we can estimate them from the three cycles (three data). When the first three data are used, the unknown parameters are estimated as

$$\mathbf{X} = \begin{bmatrix} 1 & 0\times 2 & 0 \\ 1 & 1\times 2 & 1 \\ 1 & 4\times 2 & 8 \end{bmatrix}, \mathbf{y} = \begin{Bmatrix} 1.87 \\ 5.40 \\ 6.86 \end{Bmatrix} \Rightarrow \hat{\boldsymbol{\theta}} = [\mathbf{X}^{\mathrm{T}}\mathbf{X}]^{-1}\{\mathbf{X}^{\mathrm{T}}\mathbf{y}\} = \begin{Bmatrix} 1.87 \\ 2.91 \\ -2.28 \end{Bmatrix}$$

which is also obtained by adding the given information to the Matlab code [**LS**] as

```
% Example 2_3 RUL prediction using LS
L=2;
y=[1.87 5.40 6.86 12.76 19.89 30.05 38.76 60.70 83.35 106.93
141.19]';
x=[0:1:10]';
% At xk = 2
X=[ones(length(x(1:3)),1)  L*x(1:3).^2  x(1:3).^3];
thetaH=(X'*X)\(X'*y(1:3))
```

In the above MATLAB code, all data are given in array `y`, but only the first three, `y(1:3)`, are used for least squares. Using the estimated parameters, RUL can be predicted using the same way explained in the previous text around Eq. 2.17 and the MATLAB code **[LS]**. In this case, the current time needs to be changed as `currt=2;`.

However, the RUL result cannot be obtained in this case since the degradation predicted based on the estimated parameters does not reach the threshold, which is shown in Fig. E2.2 that is obtained from the MATLAB code **[LS]** with the following modifications:

```
% Degradation prediction at xNew
xNew=[0:0.1:12]';
XNew=[ones(length(xNew),1)  L*xNew.^2  xNew.^3];
yH2=XNew*thetaH;
%
% At xk = 3
X=[ones(length(x(1:4)),1)  L*x(1:4).^2  x(1:4).^3];
thetaH=(X'*X)\(X'*y(1:4))
yH3=XNew*thetaH;
%
% plotting degradation predictions (Fig. E.2.2)
figure(1);
plot(x(1:4),y(1:4),'.k','markersize',30)
hold on; plot (xNew,yH2,'k','linewidth',2)
plot(xNew,yH3,'--r','linewidth',2)
plot([0 10],[150 150],'g','linewidth',2)
axis([0 10 -50 160])
legend('Data','Pred. at xk=2','Pred. at xk=3','Threshold');
xlabel('Cycles');ylabel('Degradation level');
```

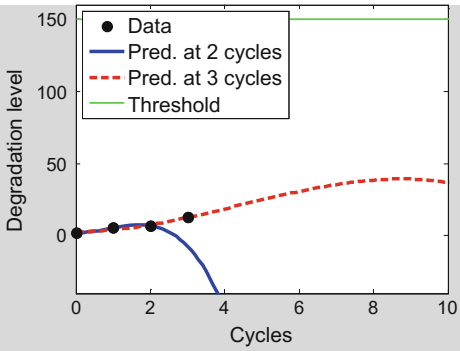


Fig. E2.2 Degradation prediction at two and three cycles

In this figure, the blue solid curve is the degradation prediction in this case (at two cycles), which goes down as cycles increase. That is, the number of data is too small to have any meaningful prediction. Likewise, the degradation prediction with one more data denoted as red-dashed curve dose not reach the threshold either. These two cases predict an infinite life, which is caused by insufficient number of data. Therefore, RUL is predicted from four cycles (five data) as the same way explained so far, and the prediction results are shown in the following table and figure. The following MATLAB code can be used to calculate and plot the RUL:

```
%
%RUL prediction
thres=150; % threshold level
rul=zeros(size(x));
syms xEOL
for i=4:11
  currt=i-1;
  X=[ones(length(x(1:i)),1)  L*x(1:i).^2  x(1:i).^3];
   thetaH=(X'*X)\(X'*y(1:i));
  XEOL=[ones(length(xEOL),1)  L*xEOL.^2  xEOL.^3];
  eolFuc=thres-XEOL*thetaH; % EOL func.
  eol=double(solve(eolFuc,'Real',true));
  rul(i)=min(eol(eol>=0))-currt;
end
%
figure(2)
plot(x(4:11),rul(4:11),'.--r','markersize',25,'linewidth',2);
hold on; plot([0 10],[10 0],'k','linewidth',2);
axis([0 10 0 10]);
legend('Prediction','True');
xlabel('Cycles');ylabel('RUL');
```

Table E2.1 RUL results

Cycle	0, 1, 2, 3	4	5	6	7	8	9	10
RUL	N/A	12.21	6.56	11.46	3.50	2.20	1.33	0.27

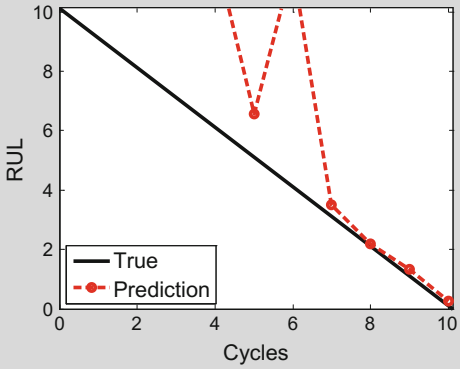


Fig. E2.3 RUL prediction when L = 2 in Eq. 2.9

Note that the predicted RULs still have a large error when the time cycle is up to 6 because the model parameters are not estimated accurately. However, starting from seventh time cycle, it is shown a good prediction of RUL. In general, it is true that more data can yield more accurate prediction.

2.3.2 *Prognostics Metrics*

In general, when different methods are employed to predict RUL, their performance can be compared with known true RUL using five prognostics metrics (Saxena et al. 2009): prognostic horizon (PH), $pha-\lambda$ accuracy, relative accuracy (RA), cumulative relative accuracy (CRA), and convergence. These metrics are illustrated in Fig. 2.9 with the RUL results in Fig. 2.8. It is noted that these metrics are possible only when the true RUL is available. Therefore, these can be used to evaluate the performance of prognostics algorithms with the true RUL.

2.3.2.1 Prognostic Horizon (PH)

Prognostic horizon (PH) is defined as the difference between the true EOL and the first time when the prediction result continuously resides in the *pha* accuracy zone, which is shown in Fig. 2.9a. The accuracy zone has a constant bound with a magnitude of $\pm pha$ error with respect to true EOL, shown as two parallel dotted lines when $pha = 5$ %. In this example (the RUL results in Fig. 2.8), the first time when the RUL

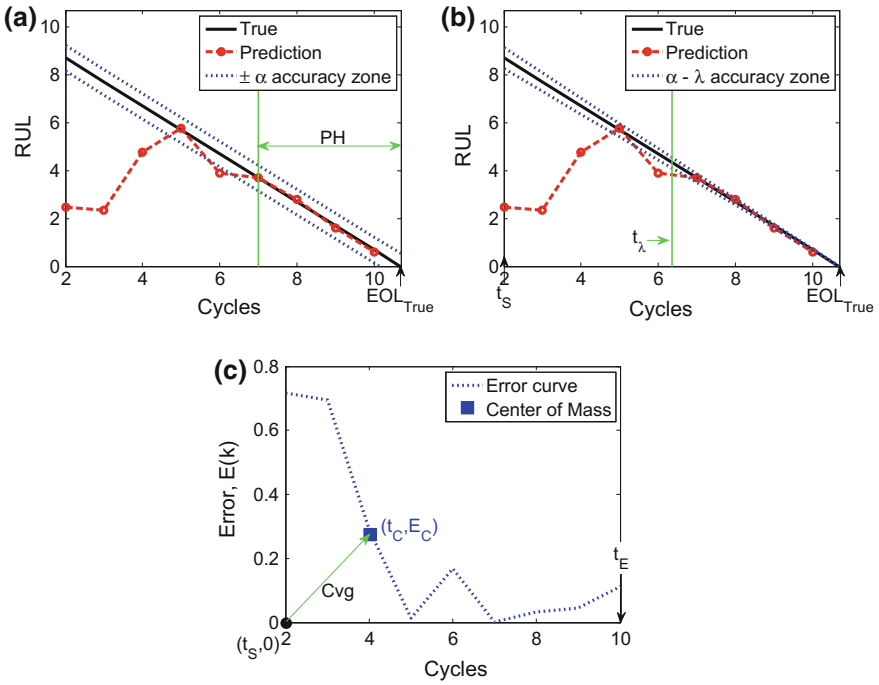


Fig. 2.9 Prognostics metrics with $\alpha = 0.05$, $\lambda = 0.5$ **a** PH, **b** $\alpha-\lambda$ accuracy, **c** Relative error and convergence

resides in the accuracy zone is at the seventh cycle, and the true EOL is 10.7 cycles, thereby PH is 3.7. The prognostics method with a larger PH indicates the better performance, which allows earlier prediction for EOL with more reliability.

2.3.2.2 $\alpha-\lambda$ Accuracy

The $\alpha-\lambda$ accuracy determines whether a prediction result falls within the α accuracy zone at a specific cycle t_λ, which is shown in Fig. 2.9b. The accuracy zone is not constant but varies with $\pm\alpha$ ratio to the true RUL, which is shown as two dotted lines when $\alpha = 0.05$ in the figure. It becomes smaller as cycle increases by reflecting that the prediction accuracy increases as more data are available. This expects that if the prediction results are proper, they will fall within the accuracy zone at any cycle. For this, the specific cycle, t_λ is employed, which is expressed with a fraction of λ between starting cycle of RUL prediction, t_S ($\lambda = 0$) and the true EOL ($\lambda = 1$), which is

$$t_\lambda = t_S + \lambda(\mathrm{EOL}_{\mathrm{True}} - t_S)$$

In this example, $t_S = 2$ cycles, and λ is usually considered as 0.5, thereby $t_\lambda = 6.35$. Since RUL is predicted at every cycle (no result at 6.35 cycles), interpolated RUL value from two RULs at both side of t_λ is used, or t_λ is adjust as the closest cycle. In this example, $t_\lambda = 6$ cycles is used as the adjusted one. The result of $\alpha-\lambda$ accuracy becomes true or false, and true is when a prediction result is included in the $\alpha-\lambda$ accuracy zone, otherwise it is false. It is false in this example as the RUL result at $t_\lambda = 6$ cycles does not fall in the zone.

2.3.2.3 (Cumulative) Relative Accuracy (RA, CRA)

Relative accuracy (RA) is the relative accuracy between the true and prediction RUL at t_λ, which is expressed as

$$\mathrm{RA} = 1 - \frac{|\mathrm{RUL}_{\mathrm{True}} - \mathrm{RUL}|}{\mathrm{RUL}_{\mathrm{True}}} \quad \text{at } t_\lambda$$

The relative error at each cycle is shown as the dotted curve in Fig. 2.9c. From this figure, the relative error is 0.17 at $t_\lambda = 6$ cycles, and therefore RA is 0.83 (=1−0.17). Cumulative relative accuracy (CRA) is the same as the average of RA values accumulated at every cycle from t_S to the last cycle of RUL prediction (end of prediction, t_E). In other words, CRA is the mean of unity minus the errors in Fig. 2.9c. When RA and CRA are close to one, the prediction accuracy is high.

2.3.2.4 Convergence

Finally, convergence can be considered with a nonnegative error metric of prediction accuracy or precision between the prediction and true RUL. In this example, the relative error, $E(k)$, shown as the dotted curve in Fig. 2.9c is employed. With this error curve, convergence is defined by the Euclidean distance between the point $(t_S, 0)$ and the center of mass of the area under the error curve (t_C, E_C). The center of mass is calculated as

$$\mathrm{CoM} = \sqrt{(t_C - t_S)^2 + E_C^2},$$

where

$$t_C = \frac{1}{2}\frac{\sum_{k=S}^{E}\left(t_{k+1}^2 - t_k^2\right)E(k)}{\sum_{k=S}^{E}(t_{k+1} - t_k)E(k)}, \quad E_C = \frac{1}{2}\frac{\sum_{k=S}^{E}(t_{k+1} - t_k)E^2(k)}{\sum_{k=S}^{E}(t_{k+1} - t_k)E(k)}$$

The lower the distance is, the faster the convergence is.

Table 2.5 Prognostics metrics

PH ($\alpha = 0.05$)	$\alpha-\lambda$ accuracy ($\alpha = 0.05$, $\lambda = 0.5$)	RA ($\lambda = 0.5$)	CRA	Convergence
3.6896	False	0.8310	0.7698	2.044

2.3.2.5 Results with MATLAB Code

The results from the five metrics are listed in Table 2.5, which can be obtained by setting the following information into **`[Metric]`**:

```
in [Metric]
rul=[inf  inf  2.4596  2.3471  4.7615  5.7584  3.8970  3.6922
2.7779  1.6149  0.6115]';
cycle=[0:10]';
eolTrue=10.6896;
t_s=2;
t_e=10;
alpha=0.05;
lambda=0.5;
```

Prognostics metrics for one method (physics-based approach with LS) are given in this section. However, these metrics are to compare different prognostics methods when the true RUL/EOL is known. The comparison between different prognostics methods are remained to exercise problem (see Exercise P2.12).

Example 2.4 Prognostics metrics
Obtain five prognostics metrics introduced in this section for the RUL results in Example 2.3.

Solution:
The usage of the MATLAB code **`[Metric]`** is explained in the previous text, but the RUL and the starting cycle for evaluation are changed to

```
rul=[0 0 0 0 12.2072 6.5588 11.4634 3.4997 2.2000 1.3270
0.2674]';
t_s=4;
```

Since the staring cycle (`t_s`) is four, it does not matter what values are used for the RUL at zero to three cycles. With this setting five metrics are calculated as the results in Table E2.2.

Table E2.2 Prognostics metrics when $L = 2$ in Eq. 2.9

PH ($\alpha = 0.05$)	$\alpha-\lambda$ accuracy ($\alpha = 0.05$, $\lambda = 0.5$)	RA ($\lambda = 0.5$)	CRA	Convergence
3.6896	False	0.9485	0.5025	2.8079

For this problem, the specific time t_λ(t_lam) is calculated as seven, so RA is high as shown in Fig. E2.3, but CRA is low.

[Metric]: MATLAB code for prognostics metrics

```
%=== Required Information ===================================
rul=[ ]';                                  %[k x 1] or [k x ns]
cycle=[ ]';                          %[k x 1] at rul prediction
eolTrue=[ ];                                        % true EOL
t_s=[ ];                                    % starting cycle
t_e=[ ];                                      % ending cycle
alpha=[ ];
lambda=[ ];
%=============================================================
%%% PH: Prognostic Horizon
rulTrue=eolTrue-cycle;
alphaZone=eolTrue*alpha;
loca=find( rulTrue-alphaZone<rul & rul<rulTrue+alphaZone );
a=find( [0 diff(loca')]>1,1,'last' ); if isempty(a); a=1; end
ph=eolTrue-cycle(loca(a));
disp(['PH= ' num2str(ph)]);
%%% ALA: Alpha-Lambda Accuracy
t_lam=t_s+(eolTrue-t_s)*lambda;
[~, idx_lam]=min(abs(cycle-t_lam)); t_lam=cycle(idx_lam);
a=find( rul(idx_lam)>rulTrue(idx_lam)*(1-alpha) ...
      & rul(idx_lam)<rulTrue(idx_lam)*(1+alpha) );
if isempty(a);
    disp('ALA: False');
else disp ('ALA: True');
end
%%% RA & CRA: Relative & Cumulative Relative Accuracy
ra=1-abs(rulTrue(idx_lam)-rul(idx_lam))/rulTrue(idx_lam);
si=find(cycle>=t_s,1); ei=find(cycle<=t_e,1,'last');
cra=mean(1-abs(rulTrue(si:ei)-rul(si:ei))./rulTrue(si:ei));
disp(['RA= ' num2str(ra) ', CRA= ' num2str(cra)])
%%% Cvg: Convergence
errCurve=abs( rulTrue(si:ei)-rul(si:ei) )./rulTrue(si:ei);
cycle1=[cycle(si+1:ei); eolTrue];
numer=sum( (cycle1 - cycle(si:ei)).*errCurve );
xc=0.5*sum( (cycle1.^2 - cycle(si:ei).^2).*errCurve )/numer;
yc=0.5*sum( (cycle1 - cycle(si:ei)).*errCurve.^2 )/numer;
cvg=sqrt( (xc-t_s)^2 + yc^2 );
disp(['Cvg= ' num2str(cvg)])
```

2.4 Uncertainty

Uncertainty in prognostics is an inevitable part. Numerous sources of uncertainty exist in practical cases, such as material variability, data measurement noise/bias, current/future loading conditions, error in model form, uncertainty in estimation algorithm, and

the prediction process itself. Therefore, it is critically important to treat these sources of uncertainty properly in order to have reliable prognostics results. Throughout this book, these sources of uncertainty will be addressed in detail. Among them, uncertainty due to noise in measured data will be discussed in this section.

Noise in measured data is caused by measurement variability, which represent uncertain measurement environment. That is, even if the same health monitoring system is used to measure the degradation, measured data can be different every time. Noise in measured data is random in nature, and often a statistical distribution is used to describe it. In particular, most of noise in this book is assumed to be Gaussian noise, which is statistical noise having a probability density function equal to that of the normal distribution. A special case is white Gaussian noise, in which the values at any pair of times are identically distributed and statistically independent (and hence uncorrelated). Principal sources of Gaussian noise arise during data acquisition, such as sensor noise caused by poor illumination and/or high temperature, and/or electronic circuit noise.

The objective is to estimate the level of uncertainty in estimated model parameters as well as the uncertainty in remaining useful life when measure data have noise. In the following, discussions are limited to the case when measured data are normally distributed with zero mean and variance σ^2. Depending on situations, sometime it is assumed that the variance of data is known in order to make the calculation simple. In reality, however, the variance of data is unknown and has to be identified along with the model parameters. In such a case, the number of unknown parameters increases from n_p to $n_p + 1$.

In order to show how to quantify uncertainty due to noise in measured data, the previous processes—parameter identification, degradation and RUL prediction and applying metrics—are repeated with the model in Eq. 2.9 and the data for $L = 1$ in Table 2.2. The uncertainty in model parameters will be identified first, followed by generating samples of model parameters. These samples can be used to generate samples of RUL through degradation model.

For uncertainty quantification using LS regression, it is assumed that the noise is normally distributed with zero mean and they are independent and identically distributed (i.i.d.). The variance of error, ε_k, in Eq. 2.3 can be estimated using the sum of square errors in Eq. 2.6 as follows:

$$\hat{\sigma}^2 = \frac{\mathrm{SS_E}}{n_y - n_p}, \tag{2.18}$$

where $n_y - n_p$ represents the degree of freedom, the number of unknown parameters n_p subtracted from the total number of data n_y. This is required to have unbiased estimation of variance. If the variance of data is also unknown, it has to be considered as an additional unknown parameter and the denominator in Eq. 2.18 should be changed to $n_y - n_p + 1$ (Refer to line 22 of MATLAB code **`[NLS]`** in Chap. 4). This is because the unknown variance is not included in the degrees of freedom for the regression process.

In order to quantify uncertainty in prognostics, it is necessary to quantify the uncertainty in model/function parameters first. For the case of a linear model with two parameters ($z(x) = \theta_1 + \theta_2 x$), the variance of parameters can be derived using the variance of the error in Eq. 2.18 and the parameter estimation in Eq. 2.8. The design matrix for a linear model is

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_{n_y} \end{bmatrix}.$$

Therefore

$$\mathbf{X}^{\mathrm{T}}\mathbf{X} = \begin{bmatrix} n_y & \sum_{i=1}^{n_y} x_i \\ \sum_{i=1}^{n_y} x_i & \sum_{i=1}^{n_y} x_i^2 \end{bmatrix}, \quad \mathbf{X}^{\mathrm{T}}\mathbf{y} = \begin{bmatrix} \sum_{i=1}^{n_y} y_i \\ \sum_{i=1}^{n_y} x_i y_i \end{bmatrix}$$

and from Eq. 2.8, the two estimated parameters become

$$\hat{\boldsymbol{\theta}} = \begin{Bmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \end{Bmatrix} = \begin{bmatrix} n_y & \sum_{i=1}^{n_y} x_i \\ \sum_{i=1}^{n_y} x_i & \sum_{i=1}^{n_y} x_i^2 \end{bmatrix}^{-1} \begin{Bmatrix} \sum_{i=1}^{n_y} y_i \\ \sum_{i=1}^{n_y} x_i y_i \end{Bmatrix} = \begin{Bmatrix} \bar{y} - \hat{\theta}_2 \bar{x} \\ \frac{\sum_{i=1}^{n_y} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n_y} (x_i - \bar{x})^2} \end{Bmatrix}.$$

Since $S_{xy} = \sum_{i=1}^{n_y} (x_i - \bar{x})(y_i - \bar{y})$ and $S_{xx} = \sum_{i=1}^{n_y} (x_i - \bar{x})^2$, the two parameters can be expressed as

$$\hat{\boldsymbol{\theta}} = \begin{Bmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \end{Bmatrix} = \begin{Bmatrix} \bar{y} - \bar{x} S_{xy}/S_{xx} \\ S_{xy}/S_{xx} \end{Bmatrix} \tag{2.19}$$

The estimated parameters in the above equation depend on measured data $\mathbf{y}$, which include measurement variability. Therefore, if a different set of data is used, the estimated model parameters will also be different. That is, the variability in data is propagated into the variability in model parameters. The uncertainty in model parameters can be calculated by considering y_i as a random samples and calculating variance. By using a theorem for variance calculation of a random variable (A) with a constant c,

$$Var(cA) = c^2 Var(A), \quad Var(A - c) = Var(A),$$

the variance of the two parameters can be obtained as

$$Var\left(\hat{\theta}_2\right) = Var\left(\frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{S_{xx}}\right) = \frac{\sum (x_i - \bar{x})^2 \sigma^2}{S_{xx}^2} = \frac{S_{xx}\sigma^2}{S_{xx}^2} = \frac{\sigma^2}{S_{xx}}$$
$$Var\left(\hat{\theta}_1\right) = Var\left(\bar{y} - \hat{\theta}_2 \bar{x}\right) = \frac{\sigma^2}{n_y} + \bar{x}^2 \frac{\sigma^2}{S_{xx}}$$

In the above equations, the terms related with x are constants because the source of uncertainty is from noise in data, and thus, the terms related with data y_i are random variables. Therefore, the variance of two parameters in a linear model is obtained as

$$Var\left(\hat{\boldsymbol{\theta}}\right) = \begin{bmatrix} \frac{\sigma^2}{n_y} + \bar{x}^2 \frac{\sigma^2}{S_{xx}} & \\ & \frac{\sigma^2}{S_{xx}} \end{bmatrix}, \quad S_{xx} = \sum_{i=1}^{n_y} (x_i - \bar{x})^2 \tag{2.20}$$

The above equation means that when the variance of data y_i is σ^2, the variance of estimated model parameters can be calculated using the above equation. It is obvious that when there is no variance in data; i.e., $\sigma = 0$, there is no variance in the estimated model parameters. Also, the variance in model parameters is linearly proportional to the variance of data. It is also interesting to note that having a large number of data reduces uncertainty in model parameters and eventually makes them deterministic because as $n_y \to \infty$, $S_{xx} \to \infty$.

The above derivations can be generalized to the case with many unknown parameters. In such a case, the derivation is complicated and the following equation can be employed to describe the correlation between parameters as[4]

$$\Sigma_{\hat{\boldsymbol{\theta}}} = \sigma^2 [\mathbf{X}^{\mathrm{T}} \mathbf{X}]^{-1}, \tag{2.21}$$

which is a $n_p \times n_p$ covariance matrix of the unknown parameters. Since the variance of error in data (σ) is unknown due to limited number of data, usually its estimated value ($\hat{\sigma}$) in Eq. 2.18 can be used instead. The diagonal elements represent the variance of each estimated parameters, and the off-diagonal elements are the covariance between each parameter, whose normalized version is correlation.

Example 2.5 Variance in a linear model
Calculate the variance of two parameters in a linear model, $z = \theta_1 + \theta_2 x$ using the data in Table 2.2 based on Eqs. 2.20 and 2.21 for each, and compare the results.

[4]More detailed information on the variance of the parameters are found in Montgomery et al. (1982).

Solution:
First, it is necessary to estimate the two parameters in order to estimate their variance. Using least squares method, the two parameters can be estimated as

$$\mathbf{X} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \end{bmatrix}, \mathbf{y} = \begin{Bmatrix} 6.99 \\ 2.28 \\ 1.91 \\ 11.94 \\ 14.60 \end{Bmatrix} \Rightarrow \hat{\boldsymbol{\theta}} = [\mathbf{X}^{\mathrm{T}}\mathbf{X}]^{-1}\{\mathbf{X}^{\mathrm{T}}\mathbf{y}\} = \begin{Bmatrix} 2.57 \\ 2.49 \end{Bmatrix}$$

Also the following MATLAB code can be used in **[LS]**:

```
y=[6.99  2.28  1.91  11.94  14.60]';
x=[0  1  2  3  4]';
X=[ones(length(x),1)  x];
thetaH=(X'*X)\(X'*y)
```

In order to estimate the variance in parameters, the variance of error in data, $\hat{\sigma}$, is first calculated using Eqs. 2.6 and 2.18:

$$\mathrm{SS_E} = \{\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\}^{\mathrm{T}}\{\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\} = 66.97$$
$$\hat{\sigma}^2 = \frac{\mathrm{SS_E}}{n_y - n_p} = \frac{66.97}{5-2} = 22.32$$

Also, the mean and variance of data points can be obtained as

$$\bar{x} = 2,\ S_{xx} = \sum_{i=1}^{n_y} (x_i - \bar{x})^2 = [-2 \quad -1 \quad 0 \quad 1 \quad 2] \begin{Bmatrix} -2 \\ -1 \\ 0 \\ 1 \\ 2 \end{Bmatrix} = 10$$

The above calculations can be done using the following MATLAB code

```
ny=length(y);
np=size(X,2);
sse=(y-X*thetaH)'*(y-X*thetaH)
sigmaH2=sse/(ny-np)
sxx=sum((x-mean(x)).^2);
```

Now, the variances and covariance matrix of two parameters can be calculated using Eqs. 2.20 and 2.21 as

$$Var\left(\hat{\boldsymbol{\theta}}\right) = \begin{bmatrix} \frac{\hat{\sigma}^2}{n_y} + \bar{x}^2 \frac{\hat{\sigma}^2}{S_{xx}} \\ \frac{\hat{\sigma}^2}{S_{xx}} \end{bmatrix} = \begin{bmatrix} \frac{22.32}{5} + 2^2 \times \frac{22.32}{10} \\ \frac{22.32}{10} \end{bmatrix} = \begin{bmatrix} 13.39 \\ 2.23 \end{bmatrix}$$

$$\Sigma_{\hat{\boldsymbol{\theta}}} = \hat{\sigma}^2 [\mathbf{X}^{\mathrm{T}}\mathbf{X}]^{-1} = 22.32 \begin{bmatrix} 0.6 & -0.2 \\ -0.2 & 0.1 \end{bmatrix} = \begin{bmatrix} 13.39 & -4.46 \\ -4.46 & 2.23 \end{bmatrix}$$

```
thetaVar=sigmaH2*[1/ny+mean(x)^2/sxx; 1/sxx]
thetaCov=sigmaH2*inv(X'*X)
```

The variances in Eq. 2.20 derived for the case of two parameters are the same as the diagonal components of covariance matrix obtained from Eq. 2.21. The covariance -4.46 in the matrix represents a measure of linear correlation between θ_1 and θ_2, whose normalized version is known as the correlation coefficient. Usually, the correlation between parameters makes estimating parameters difficult, which will be discussed in Chap. 6.

By using a covariance matrix in Eq. 2.21, the distribution of model parameters can be obtained. In order to do that, it is first necessary to understand the difference between population and samples in statistics. Let us assume that a random variable, A, is normally distributed with a mean, μ, and standard deviation, σ. In statistics, A is often referred to as a population. The basic assumption is that it is possible to generate samples from the population, but the mean and standard deviation of population are unknown. Then, the objective is to estimate the mean of population using a finite number of samples.

Let us assume that n_y number of samples are generated from the population, A. In general, the mean of samples, $\bar{A}$, may be different from the mean μ of population A. As the number of samples increases, however, the mean of samples converges to that of population. In addition, if different sets of n_y samples are used, the sample means may be different from each other. Therefore, it is natural to consider that the sample mean is also a random variable as it changes at every set of samples. It is known that when a random variable, A, is normally distributed with a population mean, μ, and standard deviation, σ, its sample mean, $\bar{A}$, from n_y data is also normally distributed with a mean μ and variance σ^2/n_y.[5]

[5]A sample mean of n_y data from the population A is not the same if another n_y data are collected, which means the sample mean is also a random variable and denoted by $\bar{A}$. In this case, the mean and standard deviation of $\bar{A}$ is μ and $\sigma/\sqrt{n_y}$, respectively. See a book by Haldar and Mahadevan (2000).

In practice, however, the mean and standard deviation of population are unknown. They have to be estimated from the mean and standard deviation of samples. In addition, it is hard to obtain different set of samples. Therefore, the objective is when the sample mean, $\bar{A}$, and sample standard deviation, s, are given from a single set of samples, it is required to estimate the mean and standard deviation of population. In such a case, an opposite argument can be made between population and samples. When the sample mean, $\bar{A}$, and standard deviation, s, are given, the mean of population follows a normal distribution with mean of $\bar{A}$ and variance of σ^2/n_y, i.e., $\mu \sim N(\bar{A}, \sigma^2/n_y)$. Since the populations' standard deviation is generally unknown, the standard deviation of samples, s is used instead of σ. In this case, the normalized distribution of μ with its mean and standard deviation follows the Student t-distribution (or t-distribution) with $n_y - 1$ degree of freedom

$$\frac{\mu - \bar{A}}{s/\sqrt{n_y}} \sim t_{n_y-1}$$

When the number of data n_y increases, the sample mean, $\bar{A}$, converges to the population mean, μ. It is clear that when $n_y \to \infty$, the sample mean converges to the population mean. More details of the relationship between sample distribution and population distribution will be discussed in Chap. 3.

The above discussion can be applied to estimating the population mean of parameters, $\boldsymbol{\theta}$. Note that the mean of parameters is given in Eq. 2.19 and variance in Eq. 2.20. These mean and variances are based on a single set of n_y measured data. Therefore, they can be considered as sample mean and sample variance. Therefore, the unknown population mean of parameters can be estimated by

$$\frac{\theta_i - \hat{\theta}_i}{\sigma_{\hat{\theta}_i}} \sim t_{n_y-n_p} \Rightarrow \theta_i \sim \hat{\theta}_i + t_{n_y-n_p} \cdot \sigma_{\hat{\theta}_i}, \tag{2.22}$$

where $t_{n_y-n_p}$ is the t-distribution with a degree of freedom $n_y - n_p$ and $\sigma_{\hat{theta}}$ is the square root of sample variance in Eq. 2.20; i.e., the standard deviation. Based on Eq. 2.22, samples of the parameter distribution can be obtained by generating samples from t-distribution, which are then applied to the model equation to predict the damage and RUL as distributions.

The bound between $\alpha\%$ and $(100 - \alpha)\%$ of the distributions, including parameters, degradation level and RUL, is called $(100 - 2\alpha)\%$ confidence interval. For example, 90 % confidence interval is the range of distribution from 5 to 95 %. The confidence interval reflects the error caused by limited number of data. As the number of data increase, the identified parameters converge to the deterministic true values, thereby, the confidence interval becomes zero. On the other hand, the prediction interval considers the measurement error (noise) of data in Eq. 2.18. As the number of data increase, the error in Eq. 2.18 converges to the population's one, and the prediction interval converges not to zero but to the magnitude of the noise.

The prediction interval is used to quantify the uncertainty in degradation level and RUL.[6]

Example 2.6 Uncertainty quantification using samples of the identified parameters

The model equation is given in Eq. 2.9 with known $\theta_1 = 5$ and unknown two parameters, θ_2, θ_3 as $z(t) = 5 + \theta_2 L t^2 + \theta_3 t^3$. Use five noisy data (i.e., the current time index is $k = 5$) in Table 2.2 in the case of $L = 1$ for the following problems.

(a) Identify the two unknown parameters θ_2, θ_3 and their covariance using Eq. 2.21.
(b) Estimate the distribution of the parameters with 5000 samples by using Eq. 2.22. Calculate the median and the 90 % confidence intervals of each parameter. Plot the histogram of each parameter and the correlation between the two parameters.
(c) Predict the future damage growth at every time from the current time ($t = 4$) to $t = 15$ using the results in (b), and plot the results of median, 90 % confidence and prediction intervals with the true one.
(d) Predict RUL when the degradation threshold is 150.

Solution

(a) Since θ_1 is known, Eq. 2.9 can be modified as

$$z^* = z - 5 = \theta_2 t^2 + \theta_3 t^3$$

Then, the design matrix, $\mathbf{X}$, and the vector of data, respectively, can be written as

$$\mathbf{X} = \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 4 & 8 \\ 9 & 27 \\ 16 & 64 \end{bmatrix}, \mathbf{y}^* = \mathbf{y} - 5 = \left\{ \begin{matrix} 6.99 \\ 2.28 \\ 1.91 \\ 11.94 \\ 14.60 \end{matrix} \right\} - 5$$

[6]The confidence/prediction intervals are meaningful under the assumption that the model form is correct. If most degradation data are obtained only at early stage, the intervals will be narrow, but the error of degradation prediction at future cycles can be large when the prediction performed based on a different model form from true behavior. Usually, however, more data reflect more degradation behavior, which means the degradation prediction taking account of uncertainty becomes more reliable as more data are employed.

Therefore, the unknown parameters can be identified as

$$\hat{\boldsymbol{\theta}} = [\mathbf{X}^{\mathrm{T}}\mathbf{X}]^{-1}\{\mathbf{X}^{\mathrm{T}}\mathbf{y}^{*}\} = \begin{Bmatrix} -0.58 \\ 0.31 \end{Bmatrix}$$

The square sum of errors and variance of data can be calculated using Eqs. 2.6 and 2.18

$$\mathrm{SS_E} = \{\mathbf{y}^{*} - \mathbf{X}\boldsymbol{\theta}\}^{\mathrm{T}}\{\mathbf{y}^{*} - \mathbf{X}\boldsymbol{\theta}\} = 35.78,$$
$$\hat{\sigma}^2 = \frac{\mathrm{SS_E}}{n_y - n_p} = \frac{35.78}{5-2} = 11.93$$

The covariance matrix of estimated parameters can be obtained from Eq. 2.21 as

$$\Sigma_{\hat{\boldsymbol{\theta}}} = \hat{\sigma}^2[\mathbf{X}^{\mathrm{T}}\mathbf{X}]^{-1} = \begin{bmatrix} 1.42 & -0.38 \\ -0.38 & 0.10 \end{bmatrix}, \sigma^2_{\hat{\boldsymbol{\theta}}} = \begin{Bmatrix} 1.42 \\ 0.10 \end{Bmatrix}$$

The following MATLAB code can calculate the estimated parameters as well as their covariance matrix

```
y=[6.99  2.28  1.91  11.94  14.60]';
ys=y-5;
x=[0  1  2  3  4]';
X=[x.^2  x.^3];
thetaH=(X'*X)\(X'*ys)

ny=length(y); % num. of data
np=size(X,2); % num. of unknown parameters

sse=(ys-X*thetaH)'*(ys-X*thetaH)
sigmaH2=sse/(ny-np)
thetaCov=sigmaH2*inv(X'*X) %covariance matrix using Eq.2.21
sigTheta=sqrt(diag(thetaCov)) %Standard error of theta
```

(b) First, samples are drawn by using multivariate t random numbers, `mvtrnd` with $\Sigma_{\hat{\boldsymbol{\theta}}}$, the degree of freedom, and the number of samples in MATLAB function. Then, samples of the two parameters are obtained by multiplying with the standard deviation of parameters, $\sigma_{\hat{\boldsymbol{\theta}}}$, and adding the

estimated parameters, $\hat{\boldsymbol{\theta}}$. The MATLAB program is given below with results obtained in (a).

```
ns=5000; % num. of samples
mvt=mvtrnd(thetaCov,ny-np,ns)';
thetaHat=repmat(thetaH,1,ns)+mvt.*repmat(sigTheta,1,ns);
```

The array **thetaHat** is a 2×5000 $(n_p \times n_s)$ matrix containing sampling results from the distributions of two parameters. The median is 50 percentile, and 5, 95 percentiles are calculated for 90 % confidence interval as

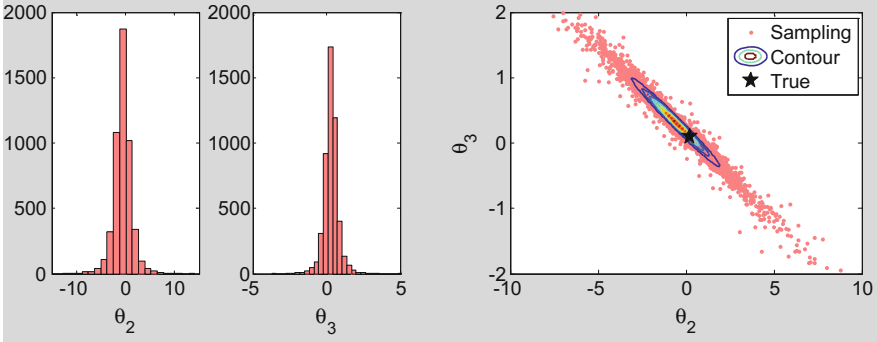
```
alpha=0.05;
thetaCI=prctile(thetaHat',[alpha 0.5 1-alpha]*100)'
```

The results are listed in Table E2.3, which can slightly vary since it is based on random number generation.

Table E2.3 Median and 90 % confidence intervals of identified parameters

	5 percentile	Median	95 percentile
$\hat{\theta}_2$	−3.40	−0.59	2.27
$\hat{\theta}_3$	−0.46	0.31	1.08

Finally, the distribution of the two parameters can be obtained using the following MATLAB code:



(a) histogram of each parameter (b) correlation

Fig. E2.4 Distribution of identified parameters

```
i=1; subplot(1,2,i); hist(thetaHat(i,:),30);
i=2; subplot(1,2,i); hist(thetaHat(i,:),30);

figure;
plot(thetaHat(1,:),thetaHat(2,:),'.','color',[0.5 0.5 0.5]);
```

(c) The future degradation level is obtained as the red-dotted (confidence interval) and blue dash-dotted (prediction interval) lines in Fig. E2.5 (they are overlapped each other). The 90 % confidence interval reflects the uncertainty in the parameters due to limited numbers of data with noise, which is obtained by substituting the identified parameters in the model equation as

$$\hat{z} = \hat{z}^* + 5 = \hat{\theta}_2 t^2 + \hat{\theta}_3 t^3 + 5$$

```
xNew=[4:15]'; XNew=[xNew.^2  xNew.^3];
nn=length(xNew);
zHat=XNew*thetaHat +5*ones(nn,ns);
zHatCI=prctile(zHat',[alpha 0.5 1-alpha]*100)'; % nn x 3
```

The blue dash-dot curve is the prediction interval that includes the noise level in data in Eq. 2.18. Since it was assumed that the error in data is normally distributed and since the variance of error is estimated, the predictive distribution would be a t-distribution with mean of $\hat{z}$ and standard deviation of $\hat{\sigma}$ in Eq. 2.18. However, since the estimated variance of error is given, the degree of freedom would be $n_y - n_p$.

```
zPredi= zHat+trnd(ny-np,nn,ns)*sqrt(sigmaH2);
zPrediPI=prctile(zPredi',[alpha 0.5 1-alpha]*100)';
```

The MATLAB code to obtain Fig. E2.5 is given as

```
thres=150;
xTrue=[0:15]';
XTrue=[ones(length(xTrue),1)  xTrue.^2  xTrue.^3];
zTrue=XTrue*[5 0.2 0.1]';
plot(x,y+5,'.k','markersize',30); hold on;
plot(xTrue,zTrue,'k');
plot(xNew,zHatCI(:,2),'--r'); %from samples
plot(xNew,zHatCI(:,1),':r');
plot(xNew,zPrediPI(:,1),'-.b');
plot([0 20],[thres thres],'g')
plot(xNew,zHatCI(:,3),':r');
plot(xNew,zPrediPI(:,3),'-.b');
xlabel('Cycles'); ylabel('Degradation level')
legend('Data','True','Median','90% C.I.','90% P.I.',
'Threshold')
axis([0 15 -50 200])
```

In this example, the error in data is relatively small compared to the error in parameters due to a small number of data (actually, the error itself is small compared to the degradation level), so the two intervals are very close each other.

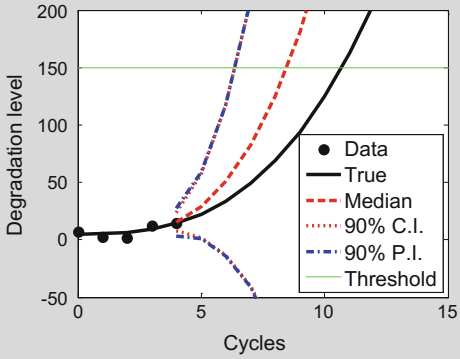


Fig. E2.5 Degradation prediction by considering uncertainty in data $k = 5$

(d) RUL can be predicted based on the program given in Sect. 2.3.1 by repeating the process for all samples of parameters. However, it is inefficient to solve the equation for 5000 times by 5000 samples of the parameters. Therefore, the following program is used instead. The line 6 is to find the first location where cycle is greater than or equals to the threshold. The line 7 is the case when RUL is infinite (this case is not included in the RUL results) because the degradation does not reach to the threshold, while the line 8 is when RUL is the same as the current

cycle, i.e., RUL = 0. Except for these two cases, RUL is predicted by interpolating the two points nearby the threshold, which is in the lines 10–11.

```
1   k=5; % current time index
2   currt=x(k); % current cycle
3   thres=150; % threshold
4   i0=0;
5   for i=1:ns
6    loca=find(zPredi(:,i)>=thres,1);
7    if isempty(loca); i0=i0+1;
8    elseif loca==1; rul(k,i-i0)=0;
9    else
10    rul(k,i-i0)=interp1([zPredi(loca,i) zPredi(loca-1,i)], ...
11    [xNew(loca) xNew(loca-1)],thres)-currt;
12   end
13  end
```

Finally, the distribution of RUL is obtained as shown in Fig. E2.6, which can be plotted by `hist(rul(k,:),30)`.

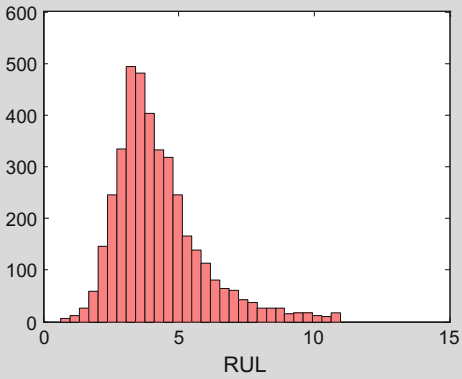


Fig. E2.6 Distribution of RUL prediction

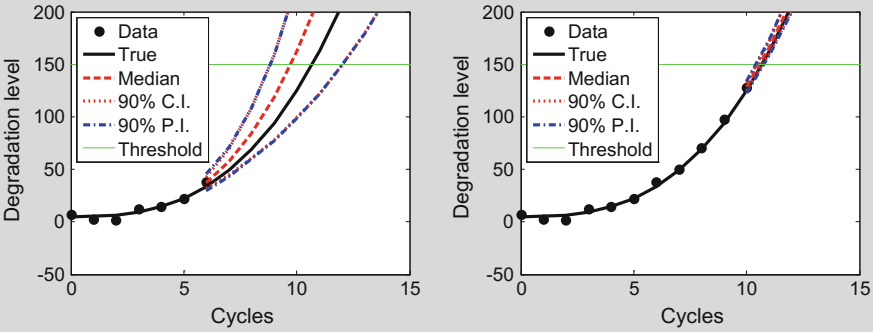
Minor modifications are required to employ the prognostics metrics for distributed RUL from the deterministic case explained in Sect. 2.3.2. PH is defined as the difference between the true EOL and the first time when more than $\beta\%$ of the predicted RUL distribution continuously resides in the α accuracy zone. In the $\alpha-\lambda$ accuracy, the result of $\alpha-\lambda$ accuracy becomes true when more than $\beta\%$ of the predictions distribution at t_λ is included in the accuracy zone. Finally, the median of the RUL distribution is used for RA, CRA, and the convergence.

Example 2.7 RUL prediction with prediction interval
Repeat Example 2.6 from identifying parameters to RUL prediction at every cycle from $k = 3$ to $k = 11$.

(a) Predict degradation and obtain confidence and prediction interval curves as given in Fig. E2.5 of Example 2.6 when $k = 7$ and $k = 11$.
(b) Predict RUL at every cycle from $k = 3$ and $k = 11$ and obtain RUL curve as given in Fig. E2.3, including prediction interval to account for uncertainty.
(c) Evaluate the RUL results in (b) using the prognostics metrics.

Solution

(a) The solution to obtain the degradation prediction like Fig. E2.5 is given in Example 2.6. The following figures can be obtained by using different data.



(a) $k = 7$ (b) $k = 11$

Fig. E2.7 Degradation prediction by considering uncertainty in data

Since the error in data is small compared to the degradation level, the prediction and confidence intervals are very close to each other. On the other hand, the difference between two intervals will be distinct when there is a large error in data (see Exercise P2.14). Both intervals become narrow as cycle/data increases.

(b) Example 2.6(d) explained how to predict RUL at a cycle. The following figure is obtained by repeating the RUL calculation for $k = 3 : 11$ and calculating 90 % prediction intervals. The samples of RUL at different time cycles are stored in array **`rul(11,ns)`**. The following program can be used to obtain Fig. E2.8 after calculating RUL for $k = 3 : 11$.

```
k1=3; k2=11;alpha=0.05;
x=0:10;
eolTrue=10.69;
rulPct=prctile(rul',[alpha 0.5 1-alpha].*100)';

plot([0 eolTrue],[eolTrue 0],'k','linewidth',3);
hold on; plot(x(k1:k2),rulPct(k1:k2,2),'--or','linewidth',3);
plot(x(k1:k2),rulPct(k1:k2,1:2:3),':r','linewidth',3);
xlabel('Cycles'); ylabel('RUL'); axis([0 eolTrue 0 eolTrue])
```

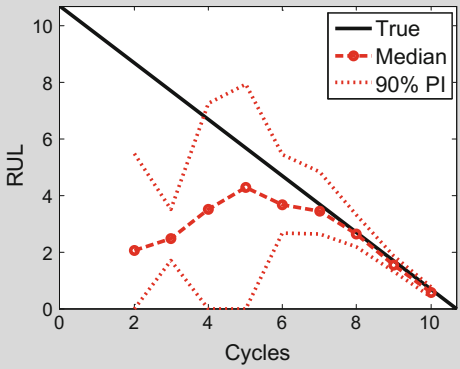


Fig. E2.8 RUL prediction results for $k = 3 : 11$

(c) Evaluating RUL prediction can be done by replacing several lines in **`[Metric]`**. This is because now the RUL is given as ns number of samples. After setting all required information, `beta=0.5` is added after line 8 in **`[Metric]`** for the fraction as a criterion when a distribution of RUL is considered (50 % is usually employed, but it can depend on users). This is equivalent to take the median of RUL samples.

```
cycle=[0:10]';
t_s=2;
eolTrue=10.6896;
t_e=10;
alpha=0.05;
lambda=0.5;
beta=0.5;
```

For the PH, lines 13–15 are replaced with

```
for i=1:size(rul,1)
 loca=find(rulTrue(i)-alphaZone<rul(i,:) ...
  & rul(i,:)<rulTrue(i)+alphaZone);
 probPH(i,:)=length(loca)/size(rul,2);
end
loca=find(probPH<beta,1,'last')+1;
ph=eolTrue-cycle(loca);
```

which is to find the first cycle when more than 50 % of samples continuously fall within the accuracy zone. For $\alpha-\lambda$ accuracy, lines 20–22 are replaced with

```
a=find( rul(idx_lam,:)>rulTrue(idx_lam)*(1-alpha) ...
      & rul(idx_lam,:)<rulTrue(idx_lam)*(1+alpha) );
probAL=length(a)/size(rul,2);
if probAL<beta;
```

Finally, for RA, CRA, and the convergence, line 27 is replaced with

```
ra=1-abs(rulTrue(idx_lam)-
median(rul(idx_lam,:)))/rulTrue(idx_lam);
```

and the following code is added after line 28, and `rul` after this line is replace by `rulA` to use median of the RUL distribution.

```
if size(rul,2)==1; rulA=rul; else rulA=median(rul')'; end
```

The metrics results are listed in Table E2.4.

Table E2.4 Prognostics metrics when RUL is predicted as a distribution

PH ($\alpha = 0.05$)	$\alpha-\lambda$ accuracy ($\alpha = 0.05$, $\lambda = 0.5$)	RA ($\lambda = 0.5$)	CRA	Convergence
3.6896	False	0.7794	0.6969	2.4684

2.5 Issues in Practical Prognostics

In this chapter, a least squares-based prognostics method is explained, which is a simple and effective method for the case of linear regression with Gaussian noise (normally distributed error). However, practical cases require more than polynomial

functions and Gaussian noise, which cannot be solved by the linear regression method. For more practical conditions, Bayesian-based approaches are usually employed, which will be introduced in Chaps. 3 and 4.

Also, there are several issues to be considered for a practical prognostics. First, the noise and bias have an effect on the prognostics results. Noise comes from various sources such as variable measurement environment and measurement variability, while the bias comes from calibration error in measurement equipment. The effect of small level of noise with a simple example is shown in Sect. 2.2.2. In reality, there can be a large level of noise and bias in complex systems. Also, it is difficult to identify model parameters accurately when they are correlated each other. Even with the simple example used in this chapter, the two parameters are correlated each other (see Example 2.6). Not only model parameters themselves but also loading conditions can be correlated with the parameters (see Example 2.2), whose effect on prognostics results will be discussed in the following chapters.

The abovementioned two issues are under the assumption that physic-based approaches are available. Physical degradation models are, however, rare in practice. When no physical model that describes the degradation behavior is available, data-driven approaches should be employed. In data-driven approaches, it is critical to utilize many sets of degradation data, but it is not easy to obtain several sets of degradation data due to expensive time and costs. More fundamental issue is that degradation data cannot be directly measured in most case and need to be extracted from sensor signals. These four issues, noise and bias, correlation, rare physical models and expensive data, and indirect data will be discussed in more detail through Chaps. 4 and 7.

2.6 Exercises

P2.1 What is the definition of prognostics? Use the terms: degradation/damage data, RUL, EOL, threshold. Also, define the meaning of these terms.

P2.2 Explain physics-based and data-driven approaches. What are the differences between the two approaches?

P2.3 Identify parameters in Eq. 2.11 using the first five data in Table 2.2 when $L = 1$. Repeat the process for Eqs. 2.12 and 2.13.

P2.4 The same data used in P2.3 are used for Fig. 2.3. Obtain the same results as Fig. 2.3 using `[LS]`.

P2.5 Repeat the process five times to obtain degradation prediction in Fig. 2.3a with different sets of data but the same level of noise such as Table 2.2. Obtain a probability distribution of degradation level at a given time cycle (4 cycles). Show how the distribution changes as this process is repeated more times, e.g., 100, 1000.

P2.6 Explain about overfitting. Which ways are preferred to avoid overfitting in data-driven approaches?

P2.7 Obtain the same results as listed in Table 2.3 using Eq. 2.16. Use Eqs. 2.14 and 2.15 to calculate the coefficient of determination, R^2, and compare the results to ones obtained from the MATLAB function, `regress`.

P2.8 Obtain the same plot in Fig. 2.7b. Additional data are given in Table 2.2.

P2.9 Predict the degradation behaviors and RUL at zeroth and first cycles with the degradation model given in Eq. 2.9 and data in Table 2.2 when $L = 1$.

P2.10 Obtain the RUL results in Fig. 2.8 with the same approach and the same data used for the figure.

P2.11 Repeat P2.10 with data-driven approach using the third-order polynomial function in Eq. 2.13 . Repeat this process again with additional two sets of data in Table 2.2 when $L = 0.5, 2$.

P2.12 Compare the RUL results from three different ways (results of P2.10 and P2.11) with five prognostics metrics. Which one is better in each metric?

P2.13 Derive Eq. 2.20 by using Eq. 2.21 with the moment matrix, $\mathbf{X}^T\mathbf{X}$ for a linear model.

P2.14 Repeat Example 2.7 with larger level of noise, U(-15,15) in data than them in Example 2.7, and compare confidence and prediction intervals from the different levels of noise.

P2.15 A random variable A is normally distributed: $A \sim N(10, 1^2)$. Generate 10, 20, 50, and 100 samples from the population distribution. Estimate the 90 % confidence intervals of the estimated mean based on four sets of samples.

P2.16 When crack sizes are measured as a function of time, as shown in the table, estimate the model parameters and their confidence intervals. Use a cubic polynomial function $z(t) = \theta_1 + \theta_2 t + \theta_3 t^2 + \theta_4 t^3$.

Time	0	100	200	300	400	500	600	700	800
Size	0.01	0.0104	0.0107	0.0111	0.0116	0.012	0.0125	0.0131	.0137

References

An D, Choi JH, Kim NH (2012) Identification of correlated damage parameters under noise and bias using Bayesian inference. Struct Health Monit 11(3):293–303

An D, Choi JH, Schmitz TL et al (2011) In-situ monitoring and prediction of progressive joint wear using Bayesian statistics. Wear 270(11–12):828–838

Bretscher O (1995) Linear algebra with applications. Prentice Hall, New Jersey

Coppe A, Haftka RT, Kim NH (2010) Uncertainty reduction of damage growth properties using structural health monitoring. J Aircraft 47(6):2030–2038

Dalal M, Ma J, He D (2011) Lithium-ion battery life prognostic health management system using particle filtering framework. Proc Inst Mech Eng, Part O: J Risk Reliab 225:81–90

Haldar A, Mahadevan S (2000) Probability, reliability, and statistical methods in engineering design. Wiley, New York

Hawkins DM (2004) The problem of overfitting. J Chem Inf Comput Sci 44(1):1–12

Kohavi R (1995) A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Paper presented at the international joint conference on artificial intelligence, Montreal, Quebec, Canada, 20–25 August 1995

Montgomery DC, Peck EA, Vining GG (1982) Introduction to linear regression analysis. Wiley, New York

Paris PC, Erdogan F (1963) A critical analysis of crack propagation laws. ASME J Basic Eng 85:528–534

Saed S (2010) Model evaluation—regression. http://www.saedsayad.com/model_evaluation_r.htm. Accessed 25 May 2016

Sankararaman S, Ling Y, Shantz C et al (2009) Uncertainty quantification in fatigue damage prognosis. In: Paper presented at the annual conference of the prognostics and health management society, San Diego, California, USA, 27 September–1 October 2009

Saxena A, Celaya J, Saha B et al (2009) On applying the prognostic performance metrics. In: Paper presented at the annual conference of the prognostics and health management society, San Diego, California, USA, 27 September–1 October 2009

Wu F, Lee J (2011) Information reconstruction method for improved clustering and diagnosis of generic gearbox signals. Int J Progn Health Manage 2:1–9

Chapter 3
Bayesian Statistics for Prognostics

3.1 Introduction to Bayesian Theory

One of the key ideas in prognostics is how to utilize the information obtained from health monitoring systems in order to predict the behavior of damage; i.e., degradation. In general, health monitoring systems measure the level of degradation (the size of crack, the depth of corrosion, the amount of wear, etc.) continuously or in fixed/variable time intervals. In the case of physics-based prognostics, the model that describes degradation is assumed to be known, but its model parameters are unknown. Then, the most important step in prognostics is to identify these unknown model parameters using health monitoring data. Once these model parameters are determined, it is possible to predict how damage will grow in the future, which is prognostics. In the case of the method of least-squares in the previous chapter, for example, the parameters of degradation model are determined by minimizing the error between measured damage sizes and model predictions. This process yields a set of deterministic model parameters that can represent measured degradation most closely.

The above-mentioned process of determining degradation model parameters works fine when the health monitoring systems measure the damage size accurately and when the degradation model is correct. In reality, however, the data always have measurement error and variability, and the degradation model may not be correct; that is, the model has so-called a model form error. In such a case, the identified degradation model parameters are not accurate and can be changed with different sets of measured data. Therefore, instead of identifying deterministic degradation model parameters, it makes more sense to characterize them probabilistically.

At this point, it might be a good idea to discuss about how to interpret the probabilistic representation of model parameters. In the viewpoint of frequentist, a probability distribution means that the variable is random in nature so that it can have different values every time when the variable is realized. For example, if we make many identical specimens of metal for tensile tests, the strength of each

N.-H. Kim et al., *Prognostics and Health Management of Engineering Systems*,
DOI 10.1007/978-3-319-44742-1_3

specimen is different. In such a case, we can represent the strength of the material using a statistical distribution. In this text, this type of uncertainty is called aleatory uncertainty, type-A uncertainty or variability.

The probabilistic representation of model parameters, however, has a different interpretation from the frequentist viewpoint. In the probabilistic viewpoint, there is a true deterministic value of model parameters, but their values are unknown. Therefore, the values are estimated using measured data, which include measurement error and noise. If the measured data have no error and the model is perfect, it is possible that the estimated model parameters are also perfect. With imperfect data and model, the estimated model parameters have uncertainty, which is represented by probability distribution. In such a case, the probability distribution does not mean that the model parameters are random. Rather, it represents the level of our knowledge on the unknown model parameters. For example, if the probability distribution of a parameter is wide, then it means that our knowledge on that parameter is not accurate. In this text, this type of uncertainty is called epistemic uncertainty, type-B uncertainty or reducible uncertainty. Different from aleatory uncertainty, since epistemic uncertainty represents the level of knowledge, it can be reduced by improving our knowledge. For example, if a model parameter is going to be estimated using measured data, the uncertainty with 10 data would be smaller than that with 5 data.

Historically, many different approaches have been proposed to identify model parameters using measured data. The least-squares introduced in Chap. 2 is one of them, which is a typical technique among regression-based methods. Also, the Bayesian statistics has been practically used to identify parameters as different techniques. There are popular algorithms such as Particle filter, Kalman filter-family, and Bayesian method, which are all based on the Bayesian approach. These algorithms can be used to identify unknown parameters and effectively reduce epistemic uncertainty in the identified parameters when more data are available. Therefore, the Bayesian statistics is introduced in this chapter before we consider prognostics methods in detail.

Bayes' theorem (Bayes and Price 1763) was named after the Thomas Bayes (1701–1761), who studied how to compute a distribution for the probability parameter of a binomial distribution using observed data, but his work was published by Richard Price later. For more than 100 years, however, Bayes' concept of "degree of belief" was rejected because it is vague and subjective. Instead, objective "frequency" was accepted in statistics for a long time. It was Jeffreys (1939) who rediscovered the Bayes' theorem and made it a modern theory (Jeffreys 1961). Until 1980s, however, Bayes' theorem has mostly stayed only as a theory because it requires a quite amount of computation for practical problems. It was starting from 1990s that the Bayes' theorem became practical due to the rapid growth of computers. Bayesian techniques for practical use have been developed by mathematicians and statisticians, and applied to areas of science (economics, medical) and engineering.

In this chapter, Bayesian inference is utilized to probabilistically identify unknown model parameters, from which the remaining useful life of the system can

be predicted. Here, inference is a process of determining a value of parameter based on collected evidence, which is in this case measured data using health monitoring systems. Bayesian inference is a statistical method to express degrees of belief on a certain event rather than a deterministic decision, which is based on information such as data and our knowledge using probability model (Gelman et al. 2004). Although there are different ways of using Bayesian inference, in this chapter it is used to update the probability distribution of unknown model parameters.

The Bayesian inference process is composed of building a prior distribution and likelihood. The former is the initial estimate of probability distribution of model parameters, which represents a prior knowledge or information of the parameter before taking any measurement. This can be established using expert's opinion, previous experience or physical reasoning. When there is no prior information, the prior distribution can be assumed to be constant for all range, which is called non-informative prior distribution. The latter, likelihood, represents the effect of measured data on the probability distribution. Its value is calculated by the probability to obtain the measure data with a given value of model parameter. The outcome of Bayesian inference is the posterior distribution, which can be obtained by combining the likelihood with the prior distribution.

In a prognostics point of view, Bayesian inference is used to estimate and update unknown parameters based on measured degradation data using health monitoring systems. The unknown model parameters are represented as a probability density function (PDF), which is updated with more data and prior knowledge or information. The more data are available, the more accurate estimation of model parameter would be possible. This PDF represents epistemic uncertainty; that is, the shape of knowledge regarding the model parameters. Once the PDF of model parameters are obtained, it can be used to predict the remaining useful life before the system fails. As expected, since the model parameters are given in the form of PDF, the remaining useful life will also be in the form of PDF. In order to maintain a certain level of safety, the operator may take a conservative value of remaining useful life to request for maintenance. Therefore, if the uncertainty in PDF is large, the conservative estimate of remaining useful life would be short, which means frequent maintenance. That is, if our knowledge on the model parameters that describe damage degradation is not good, then the system has to go through the maintenance process frequently. Therefore, it is important to reduce the uncertainty in remaining useful life, which comes from the uncertainty in model parameters. The performance of prognostics method is therefore determined based on the level of uncertainty in estimated model parameters.

The chapter is organized as follows. In Sect. 3.2, the concept of aleatory and epistemic uncertainty is explained. Section 3.3 introduces the conditional probability, which is a key concept used in the Bayesian theory. Section 3.4 explains the Bayesian theorem. Section 3.5 introduces two different methods of utilizing Bayesian update: recursive Bayesian update and overall Bayesian update. In Sect. 3.6, an example of the Bayesian parameter estimation is explained using the

simple example used in Chap. 2; and in Sect. 3.7, the inverse CDF method as one of sampling methods are introduced to utilize the posterior distribution for RUL prediction, followed by exercises in Sect. 3.8.

3.2 Aleatory Uncertainty versus Epistemic Uncertainty

3.2.1 Aleatory Uncertainty

The treatment of uncertainty is essential in prognostics because the exact value of model parameters are unknown, data from health monitoring system have measurement error and noise, and future loading/operating conditions are also uncertain. These sources of uncertainty contribute to the uncertainty in estimated model parameters and thus the remaining useful life. Therefore, how to quantify various sources of uncertainty and how to manage them are critically important in prognostics. In this section, two different ways of modeling uncertainty is discussed based on the nature of uncertainty.

In general, uncertainty is categorized into two different types, mainly because of convenience of handling them: aleatory and epistemic uncertainty. Aleatory uncertainty is also called as random variable, variability, type-A uncertainty, or irreducible uncertainty. Inherent randomness which cannot be reduced by further data is called aleatory uncertainty. Generally, a physical quantity with aleatory uncertainty is modeled by a random variable. For example, even if test specimens are made of a same material, the tensile strengths of different specimens are different. This happens because the micro-structures of grains are all different and random. Theoretically, although it is possible to model the behavior of micro-structures, it is often ignored and modeled as a homogeneous material with variability. In such a case, the variability in tensile strength of a material can be modeled as aleatory uncertainty. Variability in material properties, noise in measurement, and variability in applied loads can be considered as aleatory uncertainty. In general, a variable with aleatory uncertainty can be expressed using a probability distribution, which comes with a distribution type and distribution parameters. A probability distribution can be completely described using probability density function (PDF) or cumulative distribution function (CDF).

In general, aleatory uncertainty cannot be reduced by having more samples. The variability is still there, but having more samples can help to accurately estimate the true distribution parameters. When a variable is random, it is often required to take a conservative estimate. For example, when the failure strength is randomly distributed, it is safe to design based on a conservative value.

Example 3.1 Normal distribution
The variability of failure strength of aluminum material is given as a normal distribution with mean = 250 MPa and coefficient of variance = 10 %. Plot the PDF and CDF of the failure strength. Also, estimate the probability that the failure strength is larger than 200 MPa.

Solution:
First, the relationship between the coefficient of variance and the standard deviation is given as

$$\mathrm{CoV} = \frac{\sigma}{\mu}, \tag{3.1}$$

where μ is the mean and σ is the standard deviation of the failure strength. Therefore, the standard deviation can be calculated as $250 \times 0.1 = 25$ MPa. When a random variable X is normally distributed with mean = μ and standard deviation = σ, it is written as

$$X \sim N(\mu, \sigma^2). \tag{3.2}$$

The PDF of a normal distribution with mean = μ and standard deviation = σ can be given as

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]. \tag{3.3}$$

Figure E3.1 shows the plot of failure strength PDF. The probability is defined as the area under the PDF curve. For example, the probability of failure strength being between $\mu \pm 1\sigma$ is about 68 %, $\mu \pm 2\sigma$ about 95 %, and $\mu \pm 3\sigma$ about 99 %.

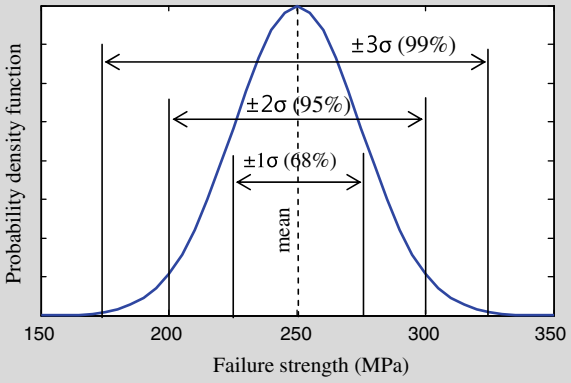


Fig. E3.1 Probability density function of normally distributed failure strength

The CDF of a distribution can be obtained by integrating the PDF. For example, the CDF, $F(x)$, is defined as the probability of failure strength X being less than x:

$$F(x) = P(X \leq x). \tag{3.4}$$

From the definition that the area under the PDF is probability, the CDF can also be written in terms of PDF as

$$F(x) = \int_{-\infty}^{x} f(\xi)\mathrm{d}\xi. \tag{3.5}$$

Unfortunately, there is no explicit integral form of the PDF $f(x)$ in Eq. 3.3. However, the following MATLAB commands can be used to plot the CDF function:

```
x=150:5:350;
y=cdf('normal',x,250,25);
plot(x,y);
```

Figure 3.1 shows the CDF of failure strength. Based on its definition, the probability that the failure strength is less than 200 MPa is 2.5 % because 200 MPa is 2σ location from the mean. Therefore, the probability that the failure strength is larger than 200 MPa is $P(X \geq 200) = 1 - P(X \leq 200) = 97.5\,\%$.

3.2.2 *Epistemic Uncertainty*

Epistemic uncertainty is different from randomness of a variable. In fact, the variable of interest may not be random at all, but the true value of the variable is unknown, and therefore, uncertain. For example, the elastic modulus for a material

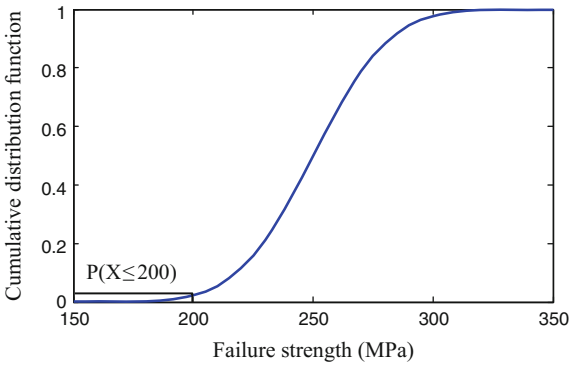


Fig. 3.1 Cumulative distribution function of failure strength

is considered to have a fixed value but unknown or poorly known. In such a case, the elastic modulus has epistemic uncertainty. Even if the variable is random, the exact distribution parameters can often be unknown. In such a case, epistemic uncertainty exists in the random variable, which is often called sampling uncertainty or statistical tolerance.

Epistemic uncertainty is often referred to as state of knowledge uncertainty, subjective uncertainty, type-B, or reducible uncertainty. Epistemic uncertainty, sometimes referred to as the lack of knowledge, is often more important than aleatory uncertainty in scientific computation and estimating the remaining useful life of a system. The epistemic uncertainty can be reduced through better understanding physics, or collecting more relevant data (Helton et al. 2010; Swiler et al. 2009).

In general, classifying aleatory and epistemic uncertainty is often unclear and depends on situation. An interesting observation between aleatory and epistemic uncertainty can be illustrated using a compressive strength of concrete material. For an existing building, the compressive strength of a concrete wall may have a deterministic value. However, since the strength is unknown, it can be said that the compressive strength of the wall has epistemic uncertainty. In order to find the value of compressive strength (i.e., to reduce or remove epistemic uncertainty), it is possible to take specimens from the building and test them to find the actual compressive strength. If the test itself is perfect, the true compressive strength can be found and the epistemic uncertainty can disappear. However, due to measurement error and noise, the epistemic uncertainty cannot completely be removed, but can be reduced if the measurement error and variability is relatively small. If the building is not built yet, on the other hand, then the compressive strength of a concrete wall has aleatory uncertainty because its strength can randomly be changed depending on its building process. Until the building will be realized, no additional testing will reduce variability of concrete strength. Therefore, aleatory uncertainty in the future building transforms into epistemic uncertainty as the building is constructed.

Another example is manufacturing tolerance. For a given manufacturing method, tolerance is considered as aleatory uncertainty because different parts may have different dimensions. Making more parts would not change the tolerance. However, if more advanced manufacturing method is employed and then it is possible to reduce the level of tolerance. Therefore, since some portion of tolerance can be reducible, it is not aleatory, but epistemic uncertainty.

Different approaches should be used to appropriately represent and model uncertainty according to its specific characteristics and information available about it. While the probability theory is widely used for modeling aleatory uncertainty, there is no dominant approach to model epistemic uncertainty. The probability theory (Helton and Breeding 1993) is the most prevalent due to its long history, sound theoretical foundation and deep root in the research on non-deterministic design. Non-probabilistic or imprecise probability approaches represent uncertainty without sharp numerical probabilities. Fuzzy sets or possibility theory (Zadeh 1999), and evidence theory (Shafer 1976; Dempster 1967) are in this category.

In this text, the probability theory is used to represent epistemic uncertainty. Even if the probability theory is used to represent both aleatory and epistemic uncertainty, the interpretation should be different. Figure 3.2 shows three examples of probability density function (PDF). For aleatory uncertainty, a large uncertainty means that when samples are generated, they tend to be widely distributed, while no uncertainty means that the variable or function is deterministic. For epistemic uncertainty, a large uncertainty means that the information related to a parameter or a quantity is ambiguous. Therefore, the shape of PDF is the shape of information for the parameter. No uncertainty means that the information is fully known, and there is no need to improve any knowledge regarding the parameter.

In this chapter, the main focus is on the epistemic uncertainty in degradation model parameters, which represents a lack of knowledge about the appropriate value to use for the model. This epistemic uncertainty is represented by a PDF. For example, material handbooks often show a lower and upper bound of a material parameters. In such a case, a uniform distribution can be used. This epistemic uncertainty represents the range of generic material. However, a particular batch of a material may have a narrow range of distribution. Therefore, if a specimen is taken from the same batch, a narrower range of material property can be found. In the later section, it will be shown that the epistemic uncertainty will be reduced using Bayesian inference when measurement data on degradation are available.

3.2.3 Sampling Uncertainty in Coupon Tests

When a material property has variability, it is important to characterize the statistical distribution of it so that designers can compensate for this aleatory uncertainty. In aerospace industry, it is common to take dozens of coupons to estimate the statistical distribution. However, since a finite number of coupons are used, the estimated statistical distribution may not be accurate, which means there exists epistemic uncertainty in the estimation process using coupons. Therefore, when designers want to compensate for uncertainty, it is important to consider both aleatory and epistemic uncertainty. In this subsection, a process of estimating

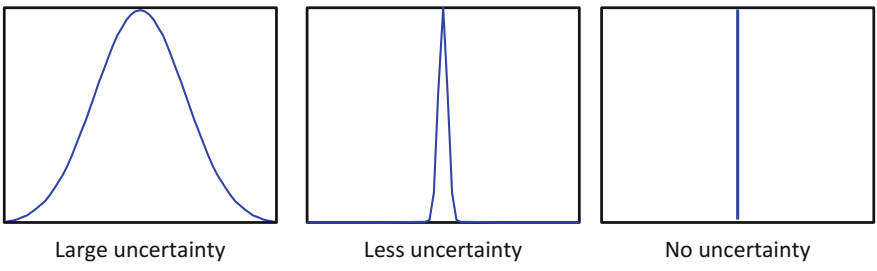


Fig. 3.2 Representation of epistemic uncertainty using probability distribution

conservative failure strength of a material is used to illustrate how to quantify aleatory and epistemic uncertainty from coupon tests.

The failure strength of a material has inherent variability. In order to find the distribution of the variability of material strength, a number of coupons (i.e., samples) are tested to measure the failure strength. However, the mean and standard deviation of coupons can be different from that of the true material distribution (i.e., population) because only a finite number of coupons are used. The population distribution can only be found when an infinite number of coupons are tested, which is practically impossible. Therefore, an important question is how to estimate the population material distribution based on the distribution obtained from coupons. This is traditionally called sampling uncertainty.

In order to explain sampling uncertainty, the process of characterizing the uncertainty in material strength is discussed as an example. Due to inherent variability of the material, it is assume that the failure strength is normally distributed; that is,

$$S_{\text{true}} \sim N\left(\mu_{\text{true}}, \sigma_{\text{true}}^2\right). \tag{3.6}$$

where μ_{true} and σ_{true} are, respectively, the mean and standard deviation of the true strength of the material, which are also called the distribution parameters. The uncertainty in S_{true} is inherent variability and called *aleatory uncertainty*.

Unfortunately, the true distribution parameters of failure strength are unknown, and they can only be estimated through tests. When n number of specimens are used to estimate the failure strength, the estimated distribution of the failure strength becomes

$$S_{\text{test}} \sim N\left(\mu_{\text{test}}, \sigma_{\text{test}}^2\right), \tag{3.7}$$

where μ_{test} and σ_{test} are, respectively, the sample mean and standard deviation of test. Due to sampling error, the distribution parameters from test are different from the true ones. Of course, when infinite test samples are used, S_{test} will converge to S_{true}. However, since a finite number of samples are used, the estimated distribution parameters have *epistemic uncertainty* (i.e., sampling uncertainty or statistical uncertainty). From the classical probability theory (Neyman 1937), the uncertainty in the estimated mean and variance can be expressed by

$$\mathrm{M}_{\text{est}} \sim N\left(\mu_{\text{test}}, \frac{\Sigma_{\text{est}}^2}{n}\right) \tag{3.8}$$

$$\Sigma_{\text{est}}^2 \sim \text{Inv-}\chi^2(n-1, \sigma_{\text{test}}^2), \tag{3.9}$$

where $\text{Inv-}\chi^2(n-1, \sigma_{\text{test}}^2)$ is the scaled inverse chi-squared distribution of $n-1$ degrees-of-freedom with the sample standard deviation as a scale parameter. Note that as the number of specimens increases, the uncertainty in Eqs. 3.8 and 3.9 decreases. When $n \to \infty$, it can be shown that $\mathrm{M}_{\text{est}} = \mu_{\text{test}}$ and $\Sigma_{\text{est}}^2 = \sigma_{\text{test}}^2$.

The source of epistemic uncertainty in Eqs. 3.8 and 3.9 are from sampling error. In general, however, different sources of epistemic uncertainty exist, such as modeling error, numerical error, etc. Although the statistical uncertainty in Eqs. 3.8 and 3.9 is given in the form of a distribution, epistemic uncertainty is not random by nature; that is, the true mean and standard deviation will be a single value, but their values are unknown. In this regard, the PDF of the distributions in Eqs. 3.8 and 3.9 should be interpreted as the shape of knowledge about the parameter. For example, Eq. 3.8 can be interpreted that the likelihood of μ_{test} being μ_{true} is higher than any other values.

When both aleatory and epistemic uncertainties exist, designer must determine conservative failure strength in order to compensate for both uncertainties. For example, when there exists aleatory uncertainty only, designer can determine the 90th percentile of the distribution as conservative failure strength. When epistemic uncertainty also exists, however, the 90th percentile cannot be determined as a single value, rather it becomes a distribution by itself. This is because the mean and standard deviation are uncertain. A designer must consider the effect of epistemic uncertainty when he or she chooses conservative failure strength.

The conservative estimate of failure strength for both aleatory and epistemic uncertainty has already been implemented aircraft design. When coupon tests are used to estimate conservative failure strength of an aluminum material, it is required to be estimated using A- or B-basis approach (U.S. Department of Defense 2002). In the case of B-basis, for example, the conservative failure strength is estimated by 90th percentile with 95 % confidence. The 90th percentile is for aleatory uncertainty, while 95 % confidence is for epistemic uncertainty.

Example 3.2 Failure strength

The distribution of failure strength and its estimated mean of population are given as in Table E3.1, calculate the mean and 90th percentile of probability of failure using double-loop MCS. Assume that there is no epistemic uncertainty in the estimated standard deviation; that is, the estimated standard deviation is the true one.

Table E3.1 Distribution parameters for aleatory and epistemic uncertainty

Failure strength	$S_{\text{est}} \sim N(\text{M}_{\text{est}}, \Sigma^2_{\text{est}})$
Estimated mean	$\text{M}_{\text{est}} \sim U(200, 250)$
True mean	$\mu_{\text{true}} = 225$
Estimated standard deviation	$\sigma_{\text{est}} = \sigma_{\text{true}} = 20$
Applied stress	$R = 190$

Solution:

The problem formulation is based on the situation where the form of probability distribution for an uncertain variable is known, but the distribution parameters governing the distribution are uncertain. In such a case, the estimated failure strength essentially becomes a distribution of distributions.

The estimated distribution of the failure strength can be obtained using a double-loop Monte Carlo simulation (MCS), as shown in Fig. E3.2. In the figure, the outer loop generates n samples from the estimated mean distribution, $M_{\text{est}} \sim U(200, 250)$, from which n sets of normal distributions, $S^i \sim N(\mu^i_{\text{est}}, \sigma_{\text{est}})$, of failure strength can be defined. In the inner-loop, m samples of failure strengths are generated from each $N(\mu^i_{\text{est}}, \sigma_{\text{est}})$, which represents aleatory uncertainty. Since the failure strength is normally distributed, it is also possible that the inner-loop can be analytically calculated without generating samples.

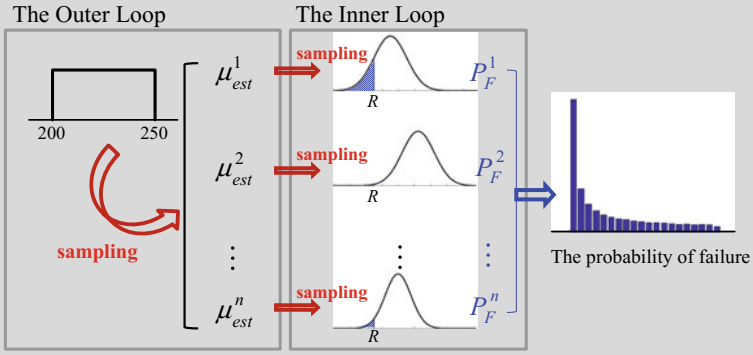


Fig. E3.2 Double-loop Monte Carlo simulation for estimated distribution of failure strength

For each given sample from epistemic uncertainty, the aleatory uncertainty is used to build a probability distribution, $N(\mu^i_{\text{est}}, \sigma^2_{\text{est}})$, from which the probability of failure, P^i_F, can be calculated.

$$P^i_F = P[S^i \le R], \quad i = 1, \ldots, n. \tag{3.10}$$

By collecting all samples, a distribution of probability of failure can be obtained, which represents the epistemic uncertainty. A conservative estimate of the probability of failure, P^{90}_F, can be obtained by taking the 90th percentile of the distribution. Therefore, the effect of aleatory uncertainty is considered by calculating P^i_F, while that of epistemic uncertainty is considered by calculating P^{90}_F.

For the given example, the PDF of the probability of failure and its 90th percentile conservative estimate is shown in Fig. 3.3. It is noted that since the PDF of the probability of failure is highly skewed, the conservative estimate, P^{90}_F, is far from its mean value, P^m_F. The following MATLAB code generates n = 100,000 samples of means and calculates probability of failure analytically.

```
m=1e5;sigma_est=20;
mu_est=50*rand(m,1)+200;
Pf=sort(cdf('norm',190,mu_est,sigma_est));
PDF=hist(Pf,50)/(m*max(Pf)/50);
x=linspace(0,max(Pf),50);
plot(x,PDF);
Pf_mean=mean(Pf)
Pf_90=Pf(90000)
```

In the probability-based method, the epistemic and aleatory uncertainties are treated separately, which can have both advantages and disadvantages. Disadvantages are the computational cost related to the double-loop MCS and the increase in dimensionality. That is, the number of uncertain input variables increases. Advantages are the separate treatment of epistemic and aleatory uncertainty such that it is clear to identify the sources of uncertainty. Also, it is possible to reduce epistemic uncertainty further using additional data.

It is also possible that aleatory and epistemic uncertainty can be combined to estimate the probability of failure. In the estimated distribution method, the epistemic and aleatory uncertainties are combined together and represented as a single distribution. Because of that, the advantages and disadvantages of the probability method are interchanged in this method. That is, the estimated distribution method is computationally inexpensive with a less number of uncertain input variables, while it cannot separate epistemic uncertainty from aleatory uncertainty.

If MCS-based sampling method is used to calculated the estimated true distribution, all $n \times m$ samples in Fig. E3.2 are used to obtain the estimated distribution of failure strength, which includes both aleatory and epistemic uncertainty. However, the real advantage of the estimated distribution method is when an

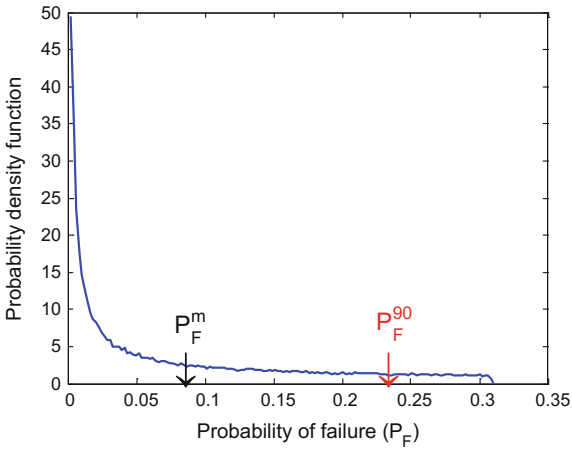


Fig. 3.3 PDF of the probability of failure and its 90 ‰ conservative estimate

analytical method is used to calculate the combined distribution, which eliminates sampling error. In order to model the above MCS process analytically, the estimated failure strength is first defined as a conditional distribution as

$$S_{\text{est}}|(\mu_{\text{est}}, \sigma^2_{\text{est}}) \sim N(\mu_{\text{est}}, \sigma^2_{\text{est}}), \tag{3.11}$$

where the left-hand side is a conditional random failure strength given μ_{est} and σ_{est}. The PDF of the estimated distribution can be calculated by integrating the conditional distribution with its parameters. Since only the uncertainty in the mean is considered, the PDF of the failure strength can be expressed as

$$f_S(s_{\text{est}}) = \int_{-\infty}^{\infty} \phi(s_{\text{est}}|\text{M}_{\text{est}} = \mu_{\text{est}})\, f_{\text{M}_{\text{est}}}(\mu_{\text{est}})\, \text{d}\mu_{\text{est}}, \tag{3.12}$$

where $\phi(s_{\text{est}}|\text{M}_{\text{est}} = \mu_{\text{est}})$ is the normal PDF of failure strength with given μ_{est}, and $f_{M_{\text{est}}}(\mu_{\text{est}})$ is the PDF of the mean parameter, which is uniformly distributed as given in Table E3.1. For mathematical details when both the mean and standard deviation have epistemic uncertainty, readers are referred to Park et al. (2014). Once the PDF of the estimated failure strength is obtained, the probability of failure can be calculated as

$$P_F^{\text{est}} = \int_{-\infty}^{R} f_S(s_{\text{est}})\, \text{d}s_{\text{est}}. \tag{3.13}$$

As the estimated distribution includes both aleatory and epistemic uncertainty, the probability of failure is a single value. Park et al. (2014) showed that Gauss quadrature with 50 segments are accurate enough when the level of probability of failure is in the order of 10^{-7}, while MCS has more than 200 % COV with a million samples.

It is interesting to note that the probability of failure in Eq. 3.13 is indeed the mean of the probability of failure distribution, P_F^m, from the probability method. It is relatively easy to show this fact by using MCS process, as

$$\frac{1}{n}\sum_{j=1}^{n}\left[\frac{1}{m}\sum_{i=1}^{m} I\left(S_j^i < R\right)\right] = \frac{1}{nm}\sum_{k=1}^{n\times m} I\left(\tilde{S}_k < R\right), \tag{3.14}$$

where $I(a) = 1$ when a is true, otherwise zero. In the above equation, S_j^i is the j-th sample of random variable $S^i \sim N(\mu_{\text{est}}^i, \sigma_{\text{est}})$, and $\tilde{S}_k = S_j^i$.

In the above equation, the term on the left-hand side corresponds to $P_F^m = 0.0789$ from the probability method, while the term on the right-hand side is $P_F^{\text{est}} = 0.0789$ in Eq. 3.13. Table 3.1 compares the estimated probability of failure

Table 3.1 Comparison of uncertainty in the probability of failure between different methods

Method	Estimated probability of failure
Probability method	$P_F^m = 0.0789,\ P_F^{90} = 0.23$
Combined distribution method	$P_F^{\text{est}} = 0.0789$

from the abovementioned two methods. The following MATLAB code can calculate the probability of failure using Eq. 3.14:

```
n=1000; m=10000; sigma_est=20;
mu_est=50*rand(n,m)+200;
R=normrnd(mu_est,sigma_est,n,m);
Pf=sum(sum(R<190))/(n*m)
```

In terms of computational cost, the estimated distribution method is more efficient than the probability method because the former can obtain the probability of failure through numerical integration. However, since the former can only estimate the expected value of epistemic uncertainty, it is difficult to take a conservative estimate. Especially when the distribution is severely skewed, such as the distribution of the probability of failure in Fig. 3.3, it can be dangerous to use a mean value. Therefore, it is not easy to find the confidence interval due to epistemic uncertainty.

3.3 Conditional Probability and Total Probability

3.3.1 Conditional Probability

Before introducing Bayesian inference, it is necessary to explain the conditional probability first, which is the foundation of Bayes' theorem. Let A and B are any two events, not necessarily mutually exclusive. In the theory of probability, the probability that A or B will occur is the sum of their separate probabilities minus the probability that they both occur; that is,

$$P(A \cup B) = P(A) + P(B) - P(A \cap B), \tag{3.15}$$

where $P(A)$ is the probability of event A being occurred, $P(A \cap B)$ is the probability of both A and B being occurred, and $P(A \cup B)$ is the probability of either A or B being occurred.

The above relation can easily be explained using the Venn diagram shown in Fig. 3.4. The numbers represent the frequencies that each event occurs, and the probability can be defined as the ratio between the frequency of an event and the

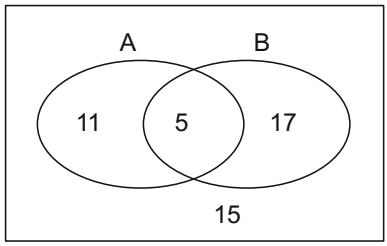


Fig. 3.4 Venn diagram for two events A and B (The numbers represent the frequencies that each event occurs)

total frequency. For example, the probability of occurring event A can be calculated by $P(A) = 16/48 = 0.333$. Similarly, $P(B) = 22/48 = 0.458$ and the intersection probability, $P(A \cap B) = 5/48 = 0.104$. Therefore, from Eq. 3.15 the union of A and B is $P(A \cup B) = 0.333 + 0.458 - 0.104 = 0.678$. This can be verified from direct calculation as $P(A \cup B) = (11 + 5 + 17)/48 = 0.687$.

The key point of Bayesian inference is how to calculate the intersection probability $P(A \cap B)$ in Eq. 3.15. The law of multiplication of probabilities states that if A and B are two events, then the probability that both A and B will occur is equal to the probability that A will occur multiplied by the conditional probability that B will occur given that A has occurred; that is

$$P(A \cap B) = P(A) \cdot P(B|A), \tag{3.16}$$

In the above equation, $P(B|A)$ is called the conditional probability, which is the probability that B will occur if we consider only those occasions on which A also occurs. In the case of Fig. 3.4, the conditional probability can be calculated by $P(B|A) = 5/16 = 0.3125$. Therefore, the intersection probability of A and B can be obtained as $P(A \cap B) = P(A) \cdot P(B|A) = 0.333 \times 0.3125 = 0.104$, which is the same result obtained before.

From the property of $P(A \cap B) = P(B \cap A)$, it is possible to express Eq. 3.16 based on the probability of event B as

$$P(A \cap B) = P(B) \cdot P(A|B). \tag{3.17}$$

Using the conditional probability $P(A|B) = 5/22 = 0.227$, the intersection probability of A and B can be obtained as $P(A \cap B) = P(B) \cdot P(A|B) = 0.458 \times 0.227 = 0.104$, which is again the identical result.

When the conditional probability of B given A is equal to the unconditional, or absolute, probability of B, the two events are said to be statistically independent. This means that the occurrence of A does not alter the probability that B will occur. If events A and B are statistically independent, the conditional probability becomes identical to the unconditional probability; that is, $P(B|A) = P(B)$, and thus,

$$P(A \cap B) = P(A) \cdot P(B). \tag{3.18}$$

For example, when two dice are thrown, the probability of yielding 6 of one die is independent of yielding 6 of another die. Therefore, the probability of both dices yielding 6 will be $\frac{1}{6} \times \frac{1}{6} = \frac{1}{36}$. Although the assumption of statistical independence is often applied for the purpose of simplicity, this is not desirable in the viewpoint of Bayesian update. When two events are statistically independent, no information can be obtained because one event cannot affect the other.

Example 3.3 Failure probability of turbine disk blades
Calculate the probability of failure of turbine disk blade. Field data are listed in Table E3.2. The events A and B, respectively, represent the number of failed blades and an inspection within 1.5 (normalized) operating hours, and $\bar{A}$ and $\bar{B}$ are their complementary events; i.e., the number of unbroken blades and an inspection after 1.5 (normalized) operating hours.

Solution:
Based on Table E3.2, the probability of each event is calculated as

- Blades failed: $P(A) = 66/520 = 0.1269$
- Blades not failed: $P(\bar{A}) = 454/520 = 0.8731$
- Inspection within 1.5 h: $P(B) = 280/520 = 0.5385$
- Inspection after 1.5 h: $P(\bar{B}) = 240/520 = 0.4615$

When A and B are independent, the probability of the intersection of A and B is the same as the multiplication of each probability of A and B. In this example,

- Blades failed within 1.5 h: $P(A \cap B) = 20/520 = 0.0385$
- $P(A) \cdot P(B) = 0.1269 \times 0.5385 = 0.0683$

which means that A and B are dependent on each other, and the conditional probability should be considered to calculate the probability of the intersection of A and B as

$$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A), \tag{3.19}$$

where $P(A|B)$ is the conditional probability of A given B, and $P(B|A)$ is the conditional probability of B given A, i.e.,

- $P(A|B) = \dfrac{\text{num. of A given B}}{\text{total num. of B}} = \dfrac{20}{280} = 0.0714$
- $P(B|A) = \dfrac{\text{num. of B given A}}{\text{total num. of A}} = \dfrac{20}{66} = 0.3030$

Now Eq. 3.19 can be verified with above results, as

$$P(A \cap B) = P(A|B)P(B) = 0.0714 \times 0.5385 = 0.0385$$
$$= P(B|A)P(A) = 0.3030 \times 0.1269 = 0.0385$$

The data in Table E3.2 can be represented as a Venn diagram in Fig. E3.3, and the probability calculations above can be done with the Venn diagram. More information on the basic of probability is found in a reference by Wackerly et al. (2008).

Table E3.2 Number of inspected turbine blades

	A (failed)	$\bar{A}$ (not failed)	Sum
B ($1.5 \geq$ hours of use)	20	260	280
$\bar{B}$ ($1.5 \leq$ hours of use)	46	194	240
sum	66	454	520

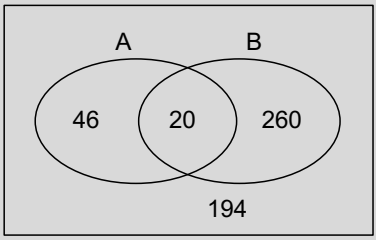


Fig. E3.3 Venn diagram for the data in Table E3.2

Example 3.4 Proof load testing
Due to manufacturing variability, the tensile strength of an aluminum bar is known as a random variable that follows a normal distribution with mean = 350 MPa and standard deviation = 50 MPa; that is, $S \sim N(350, 50^2)$. When a proof test finds that the aluminum bar is safe until s^* = 350 MPa, calculate the conditional probability of the strength given the proof test and compare it with the original probability of the strength. It is assumed that the bar is intact after the proof test; that is, no damage is occurred because of the proof test.

Solution:
Let S be the random variable that represents the strength of the bar, and s be a realization of S. Then, the probability of strength being greater than s is defined as $P(S > s)$, which can be a monotonically decreasing graph as a function of s, starting from one when $s = 0$ and ending zero when s approaches infinity. The following MATLAB commands can calculate the initial probability:

```
s=200:10:600;
P_ini=1-cdf('norm',s,350,50);
plot(s,P_ini,'b')
```

In the above MATLAB code, `cdf` function calculates the probability of $S < s$. Therefore, `1-cdf` becomes the probability of $S > s$. The solid curve in Fig. E3.4 shows the initial probability curve for the strength of the bar. In practice, the strength cannot be negative, but a random variable with a normal distribution can go to a negative infinity. However, since the zero strength corresponds to 7-sigma location, it can practically be considered as zero.

In order to calculate the conditional probability, the relation in Eq. 3.17 is used. In this case, event A corresponds to $S > s$, while event B to $S > s^*$. Therefore, the conditional probability can be expressed as

$$P(S > s|S > s^*) = \frac{P(S > s \cap S > s^*)}{P(S > s^*)}. \tag{3.20}$$

Since even A and event B are related to the strength of the bar, it would help to separate the calculate into two cases: when $s < s^*$ and $s > s^*$. When $s < s^*$, it is obvious to show that the probability of intersection becomes $P(S > s \cap S > s^*) = P(S > s^*)$ because the probability is monotonically decreasing function. On the other hand, when $s > s^*$, it can be concluded that $P(S > s \cap S > s^*) = P(S > s)$ because of the same reason. Therefore, the conditional probability can be summarized as

$$P(S > s|S > s^*) = \begin{cases} \frac{P(S>s)}{P(S>s^*)}, & \text{if } s > s^* \\ 1, & \text{if } s < s^* \end{cases}. \tag{3.21}$$

The following MATLAB commands can calculate the conditional probability, and plot the comparison between the two probabilities with and without proof test.

```
s_proof=350;
P_proof = 1-cdf('norm',s_proof,350,50);
P_post = P_ini/P_proof;
P_post(1:16) =1;
plot(s,P_ini,s,P_post)
legend('without proof test','with proof test');
xlabel('s'); ylabel('Probability')
```

In this simple example, the probability density functions (PDF) before and after the proof test can also be plotted. Initially, the PDF is normally distributed with mean of 350 MPa and standard deviation of 50 MPa. Since the proof test removes the probability of failing less than 350 MPa, the PDF becomes zero in this region. In addition, the remaining PDF needs to be

scaled in order to maintain the area under PDF curve to be one. The following MATLAB code can plot PDFs before and after the proof test.

```
s=[200:10:350 350:10:600];
p_ini=pdf('norm',s,350,50);
p_post(1:16) = 0;
p_post(17:42)=2*pdf('norm',s(17:42),350,50);
plot(s,p_ini,s,p_post)
legend('without proof test','with proof test');
xlabel('s'); ylabel('PDF')
```

Figure E3.4 shows the two probability plots. It is noted that the proof test cuts off the left half of probability density function because the probability of strength being less than 350 MPa is zero due to the proof test. In order to satisfy the property of probability, the probability density function after proof test is scaled so that the area should be one.

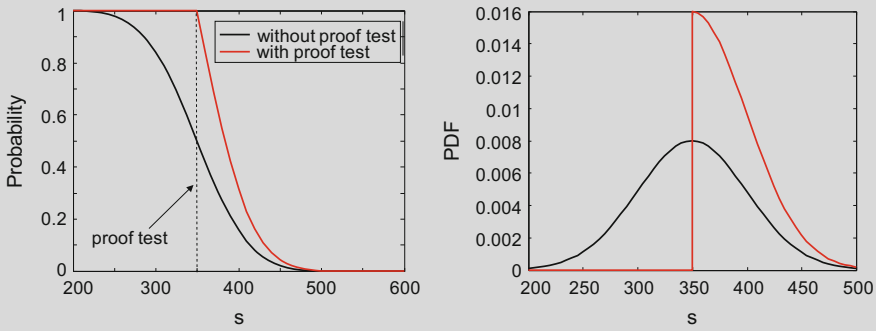


Fig. E3.4 Probability graph of the strength of a bar with and without proof test

The above example shows a very important nature of aleatory and epistemic uncertainty. In general, the variability of material strength can be considered as aleatory uncertainty; that is, if coupons are made from a batch of aluminum material, individual strengths will be different due to material variability. This is the traditional definition of aleatory uncertainty.

On the other hand, the material strength of a specific coupon is not random, but unknown before it is actually tested until it breaks. In that regards, the material strength of a specific coupon has epistemic uncertainty. If any evidence related to the strength of the coupon is collected, it is possible to improve our knowledge of the strength, or equivalently, the epistemic uncertainty can be reduced, as we demonstrated in the proof load test. This is the basic concept of Bayes' theorem, which will be discussed in the following section.

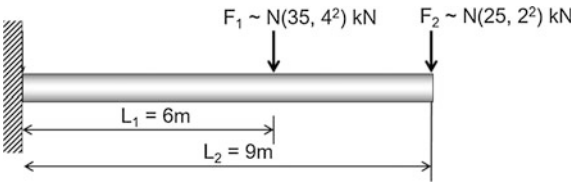


Fig. 3.5 Cantilevered beam under two random loads that are mutually exclusive

3.3.2 *Total Probability*

The conditional probability is a very useful concept when the occurrence of an event depends on the occurrence of other events. For example, a beam is under two possible loadings, F_1 and F_2, as shown in Fig. 3.5. Let the probability of occurring these two loads be $P(F_1)$ and $P(F_2)$, respectively. Also, let the probability of the beam being failed by each loading be $P(D|F_1)$ and $P(D|F_2)$, respectively. Then, the total probability of the beam being failed can be calculated by

$$P(D) = P(D|F_1)P(F_1) + P(D|F_2)P(F_2).$$

In calculating the total probability, it is important to satisfy the following two conditions. That is, all events should be non-overlapping (mutually exclusive) and their union constitutes the entire sample space (collectively exhaustive). The first condition requires that F_1 and F_2 should not occur at the same time, and the second condition requires that there is no other source of failures. The total probability can be generalized to n mutually exclusive and collectively exhaustive events E_i, $i = 1, \ldots, n$ as

$$P(D) = P(D|E_1)P(E_1) + P(D|E_2)P(E_2) + \cdots + P(D|E_n)P(E_n). \tag{3.22}$$

The above equation is called the theorem of total probability.

Example 3.5 Total probability
Consider the two loads applied to a cantilevered beam in Fig. 3.5. The two loads are mutually exclusive, and the beam can only failed by the two loads. The two loads are randomly distributed $F_1 \sim N(35, 4^2)$ kN and $F_2 \sim N(25, 2^2)$ kN. The beam is considered to be failed when the maximum bending moment reaches 235 kN-m. When the probability of occurring F_1 is $P(F_1) = 0.7$, and that of occurring F_2 is $P(F_2) = 0.3$, calculate the total probability of failure of the beam.

Solution:
The maximum bending moment occurs at the wall, whose magnitude becomes $M = L_1F_1$ when F_1 occurs or $M = L_2F_2$ when F_2 occurs. Since F_1 and F_2 are mutually exclusive, they cannot occur at the same time. When F_1 occurs, $M = L_1F_1$ is a random variable with mean $\mu_M = L_1 \times \mu_{F_1} = 6 \times 35 =$

210 kN m and variance $\sigma_M^2 = 6^2 \times \sigma_{F_1}^2 = 576 = 24^2$. Also due to linearity, the bending moment is also normally distributed; that is, $M|F_1 \sim N(210, 24^2)$. Using the same way, when F_2 occurs, $M|F_2 \sim N(225, 18^2)$.

Since the failure event is defined as $M \geq 235$ kN m, the conditional probabilities of failure when F_1 occurs can be calculated as

$$\begin{aligned} P(M \geq 235|F_1) &= P\left(\frac{M - \mu_M}{\sigma_M} \geq \frac{235 - \mu_M}{\sigma_M}\right) \\ &= P(X^* \geq 1.0417) \\ &= 1 - P(X^* \leq 1.0417) \\ &= 1 - \Phi(1.0417) \\ &= 0.149, \end{aligned}$$

where X^* is the standard normal random variable and Φ is the CDF of X^*. When F_2 occurs, the same process can be used to calculate the following conditional probability:

$$P(M \geq 235|F_2) = 1 - P(X^* \leq 0.5556) = 0.2893.$$

Therefore, the total probability can be calculated as

$$\begin{aligned} P(M \geq 235) &= P(M \geq 235|F_1)P(F_1) + P(M \geq 235|F_2)P(F_2) \\ &= 0.141 \times 0.7 + 0.2893 \times 0.3 = 0.1909. \end{aligned}$$

3.4 Bayes' Theorem

3.4.1 Bayes' Theorem in Probability Form

In general, there are two approaches to determine a probability of an event. The traditional method, often called the frequentist's method, is to generate a large number of samples (or experiments) and postulate its probability based on the number of times the event occurs. For example, when experiments are performed n times, out of which event A occurs k times, its relative frequency is k/n. Then, the probability of event A is postulated that

$$P(A) = \lim_{n \to \infty} \frac{k}{n}, \tag{3.23}$$

In the frequentist's approach, only the statistical evidence is used; that is only objective information is used in determining the probability of an event. No subjective information, such as a prior knowledge on the event, is used.

Different from the frequentist's approach, the Bayesian approach employs a degree-of-belief, which is subjective information. For example, the strength of an aluminum material is initially believed to be greater than 350 MPa with 50 % certainty. This belief may come from various sources, such as previous experience, expert's opinion or data from material handbook. As shown in Example 3.2, if tensile tests are performed using the aluminum material, the initial belief can change based on test results. This degree-of-belief can be increased or remain the same based on test results. That is, the prior knowledge or the initial degree-of-belief changes based on observations.

Even if there are many different ways of utilizing the Bayesian method, in this text, it is used in the context of Bayesian inference, which is a method of statistical inference in which Bayes' theorem expresses how a subjective degree-of-belief should rationally change to account for evidence. In particular, the subjective degree-of-belief is expressed in the form of probability density function (PDF) and observations are used to change or update the PDF. For example, epistemic uncertainty in material property is changed or reduced by using test results. In that sense, Bayesian inference is a rational method for updating beliefs.

In the case of estimating a probability of A given B, Eq. 3.19 can be used to derive the following relation:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}, \tag{3.24}$$

which is called Bayes' theorem (Bayes and Price 1763). The above equation is valid when $P(B) \neq 0$. In the viewpoint of Bayesian inference, event A is called postulate or hypothesis, and event B is called evidence. $P(A)$ is the initial degree-of-belief in event A or called the *prior*. $P(A|B)$ is the degree-of-belief after accounting for evidence event B or called the *posterior*. In this viewpoint, the Bayes' theorem is modifying or updating the prior probability $P(A)$ to the posterior probability $P(A|B)$ after accounting for evidence. Because of this nature, the Bayes' theorem is also referred to as Bayesian update. Note that when event A and event B are independent, from Eq. 3.18, $P(A|B) = P(A)$; that is, the prior probability is not improved by event B. $P(B|A)$, the conditional probability or likelihood, is the degree-of-belief in B, given that the proposition A is true.

In many applications, for instance in Bayesian inference, the event B is fixed in the discussion, and we wish to consider the impact of its having been observed on our belief in various possible events A. In such a situation the denominator of the last expression, the probability of the given evidence B, is fixed; what we want to vary is A. Bayes' theorem then shows that the posterior probabilities are proportional to the numerator

$$P(A|B) \propto P(B|A)P(A). \tag{3.25}$$

Example 3.6 Ebola epidemic
The outcome of Bayes' theorem can often be counterintuitive from our common sense. As an example, the recent epidemic of Ebola virus is considered. During an Ebola epidemic in Africa, it was estimated that a passenger arriving from Africa has 10^{-5} chance of being infected by the virus. In USA, the test for detecting Ebola virus has 99 % accuracy; that is, 1 % of the time a positive results is false and 1 % of the time a negative result is false. Calculate the probability that a person testing positive is indeed infected.

Solution:
Let event E_1 be the case infected by Ebola virus and E_2 be the case tested positive. Then the probability that a person testing positive is infected corresponds to $P(E_1|E_2)$. Using the Bayes' rule, it can be written as

$$P(E_1|E_2) = \frac{P(E_2|E_1)P(E_1)}{P(E_2)}.$$

In the above equation, it is known that $P(E_2|E_1) = 0.99$ because it is the accuracy of test when a person is infected and test can detect it. Also $P(E_1) = 10^{-5}$ because it is a chance to be infected by the virus. However, the probability $P(E_2)$ is not straightforward. In order to calculate it, the following property can be used:

$$\begin{aligned} P(E_2) &= P(E_2 \cap E_1) + P(E_2 \cap \bar{E}_1) = 0.99 \times 10^{-5} + 0.01 \times (1 - 10^{-5}) \\ &= 0.01001. \end{aligned}$$

Therefore, $P(E_1|E_2)$ can be calculated using Bayes' rule as

$$P(E_1|E_2) = \frac{P(E_2|E_1)P(E_1)}{P(E_2)} = \frac{0.99 \times 10^{-5}}{0.01001} = 0.000989.$$

The above result says that even if a person has a positive result from Ebola test with 99 % accuracy, the actual probability the person is infected is less than 0.1 %! This happens because the baseline probability of a person being infected is extremely small.

3.4.2 Bayes' Theorem in Probability Density Form

Bayes' theorem in Eq. 3.24 can be extended to the continuous probability distribution with probability density function (PDF), which is more appropriate for the purpose of the present text. Let f_X be a PDF of uncertainty variable X. If the test

measures a value Y, it is also a random variable, whose PDF is denoted by f_Y. Then, the joint PDF of X and Y can be written in terms of f_X and f_Y, as

$$f_{XY}(x,y) = f_X(x|Y=y)f_Y(y) = f_Y(y|X=x)f_X(x). \tag{3.26}$$

For example, X can be a fatigue life of a coupon, which has epistemic uncertainty, and Y is a test result of fatigue life. Normally test results have measurement variability, and thus, Y is considered as a random variable. $Y = y$ represents the case when test show a fatigue life of y.

When X and Y are independent, the joint PDF can be written as $f_{XY}(x,y) = f_X(x) \cdot f_Y(y)$ and Bayesian inference cannot be used to improve the probabilistic distribution of $f_X(x)$. Using the above identity, the original Bayes' theorem can be extended to the PDF as (Athanasios 1984)

$$f_X(x|Y=y) = \frac{f_Y(y|X=x)f_X(x)}{f_Y(y)}. \tag{3.27}$$

The interpretation of the above equation in the context of fatigue life is as follows. Initially the fatigue life of X has epistemic uncertainty in the form of $f_X(x)$. After measuring a fatigue life y of a specimen, our knowledge on fatigue life of X can be changed to $f_X(x|Y=y)$.

Note that it is trivial to show that the integral of $f_X(x|Y=y)$ is one by using the following property of marginal PDF:

$$f_Y(y) = \int_{-\infty}^{\infty} f_Y(y|X=\xi)f_X(\xi)\mathrm{d}\xi. \tag{3.28}$$

Thus, the denominator in Eq. 3.27 can be considered as a normalizing constant. By comparing Eqs. 3.24 with 3.27, $f_X(x|Y=y)$ is the posterior PDF of X given test $Y = y$, and $f_Y(y|X=x)$ is the likelihood function or the probability density value of test Y given $X = x$.

When the analytical expressions of the likelihood function, $f_Y(y|X=x)$, and the prior PDF, $f_X(x)$, are available, the posterior PDF in Eq. 3.27 can be obtained through simple calculation. In practical applications, however, they may not be in the standard analytical form. In such a case, either numerical integration or sampling method can be effectively used, which will be addressed in Sect. 3.7 in detail.

When the prior distribution is normal and the likelihood is also normal, the posterior distribution also follows a normal distribution. In this case, analytical calculation of posterior distribution is possible. In order to show this, let us consider that the prior distribution is given as $f_X(x) = N(\mu_0, \sigma_0^2)$ and the likelihood is defined using a normal distribution as $f_Y(y|X=x) = N(y, \sigma_y^2)$. Therefore, the posterior distribution can be calculated as

$$f_X(x|Y=y) = \frac{f_Y(y|X=x)f_X(x)}{f_Y(y)} \sim \exp\left[-\frac{(y-x)^2}{2\sigma_y^2} - \frac{(x-\mu_0)^2}{2\sigma_0^2}\right]. \tag{3.29}$$

Although derivation is complicated, but it is possible to show that the above expression can be rearranged to the PDF of a normal distribution. In order to do that, the following weight is defined first:

$$w = \frac{c_0}{c_0 + c}, \tag{3.30}$$

where $c_0 = 1/\sigma_0^2$ and $c = 1/\sigma_y^2$ are the inverse of variance of the prior and data. Then, the posterior distribution in 3.29 can be written as in the following normal distribution:

$$f_X(x|Y=y) \sim N(\mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right], \tag{3.31}$$

where the mean is $\mu = w\mu_0 + (1-w)y$ and variance is $\sigma^2 = 1/(c_0 + c)$. That is, the posterior mean is the weighted average of prior mean μ_0 and observed y with weights inversely proportional to the variance. It is interesting to note that when $\sigma_0 \to \infty$, $c_0 = w = 0$, and thus, $\mu = y$ and $\sigma^2 = \sigma_y^2$. That is, the posterior distribution is the same as sample distribution.

Example 3.7 Bayes' theorem for independent variables
Let X and Y are statistically independent. Show that Bayes' theorem yields the same posterior distribution with the prior distribution.

Solution:
When two variables are statistically independent, the joint PDF can be written as

$$f_{XY}(x,y) = f_X(x) \cdot f_Y(y).$$

Therefore, the Bayes' theorem in Eq. 3.27 can be written as

$$f_X(x|Y=y) = \frac{f_{XY}(x,y)}{f_Y(y)} = \frac{f_X(x) \cdot f_Y(y)}{f_Y(y)} = f_X(x).$$

That is, when two variables are statistically independent, having an observation in Y cannot affect the knowledge in X.

Example 3.8 Bayes' theorem for failure strength
A tensile test of a steel coupon shows 570 MPa for failure strength. Use Bayes' theorem to calculate the posterior distribution of failure strength. It is

known that test variability is normally distributed with standard deviation of 10 MPa, and the prior distribution is unknown (non-informative).

Solution:
In the view of Eq. 3.27, the Bayes' theorem can be modified as

$$f_X(x|Y=y) = \frac{1}{K} f_Y(y|X=x) f_X(x)$$

where the normalizing constant K can be calculated to satisfy the requirement of the area under a PDF being one. Since the prior distribution is unknown (non-informative), it can be possible to set it as a constant; that is, $f_X(x) =$ constant. Or, it can simply be ignored because it can be incorporated into the normalizing constant K. Therefore, the Bayes' theorem boils down to calculate the likelihood $f_Y(y|X=x)$, which means the value of PDF of Y at $y = 570$ MPa when $X = x$.

$$\begin{aligned} f_X(x|Y=570) &= f_Y(y=570|X=x) \\ &= \frac{1}{10\sqrt{2\pi}} \exp\left[-\frac{(x-570)^2}{200}\right]. \end{aligned}$$

Note that due to symmetry of normal distribution of f_Y, the posterior distribution is also a normal distribution centered at the test value and the standard deviation the same as test one.

Example 3.9 Conjugate distributions for failure strength
Repeat Example 3.8 when the prior distribution of failure strength is known to follow a normal distribution, $f_X(x) = N(550, 20^2)$.

Solution:
Since both the prior and likelihood follow normal distribution, they are conjugate and the posterior distribution is also a normal distribution. The weight from Eq. 3.30 becomes $w = 0.2$. That is, since the variance of prior is eight times larger than that of test data, more weight is given to the data. Therefore, the mean and variance of posterior distribution become $\mu = 566$ MPa and $\sigma = \sqrt{80}$ MPa, respectively. Figure E3.5 shows the plots of prior, likelihood and posterior distribution. It is noted that the variance of posterior distribution is smaller than that of prior as well as that of test data. The following MATLAB code can be used to calculate and plot the posterior distribution.

```
m0=550; s0=20; y=570; sy=10; x=500:1:620;
c0=1/s0^2; c=1/sy^2; w=c0/(c0+c);
m=w*m0+(1-w)*y; s=sqrt(1/(c0+c));
pdf_pr=exp(-((x-m0).^2)./(2*s0^2))/(sqrt(2*pi)*s0);
pdf_li=exp(-((x-y).^2)./(2*sy^2))/(sqrt(2*pi)*sy);
pdf_po=exp(-((x-m).^2)./(2*s^2))/(sqrt(2*pi)*s);
plot(x,pdf_pr,'-b',x,pdf_li,'-g',x,pdf_po,'-r');
```

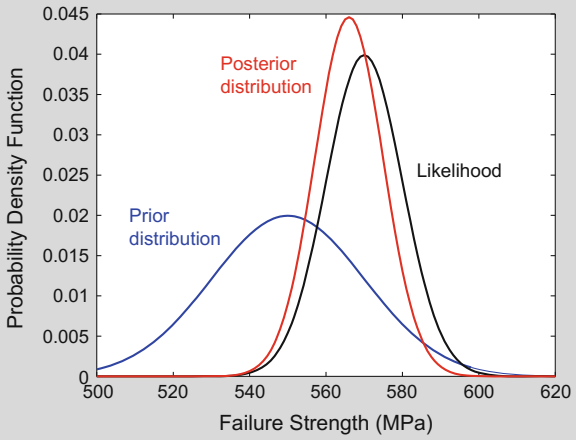


Fig. E3.5 Prior, likelihood and posterior distribution of normally distributed failure strength

In Bayes' theorem, if the posterior distribution $f_X(x|Y = y)$ is in the same family of distribution as the prior $f_X(x)$, the prior and posterior are then called conjugate distributions. In the above example, the normal distribution family is conjugate to itself (or self-conjugate) with respect to a normal likelihood function. That is, if the likelihood function is normal, choosing a normal prior over the mean will ensure that the posterior distribution is also a normal distribution. This is, however, a choice of convenience, which was useful in old days to obtain a closed-form expression for the posterior distribution. Due to the development of computation, this requirement is not important anymore.

3.4.3 *Bayes' Theorem with Multiple Data*

When multiple, independent tests are available, Bayesian inference can be applied either iteratively or all at once. When N number of tests are available; i.e., $\mathbf{y} = \{y_1, y_2, \ldots, y_N\}$, the Bayes' theorem in Eq. 3.27 can be modified to

$$f_X(x|Y = \mathbf{y}) = \frac{1}{K}\prod_{i=1}^{N}[f_Y(y_i|X = x)]f_X(x), \tag{3.32}$$

where K is a normalizing constant. In the above expression, it is possible that the likelihood functions of individual tests are multiplied together to build the total

likelihood function, which is then multiplied by the prior PDF followed by normalization to yield the posterior PDF. On the other hand, the one-by-one update formula for Bayes' theorem can be written in the recursive form as

$$f_X^{(i)}(x|Y=y_i) = \frac{1}{K_i} f_Y(y_i|X=x) f_X^{(i-1)}(x), \quad i=1,\ldots,N, \tag{3.33}$$

where K_i is a normalizing constant at i-th update and $f_X^{(i-1)}(x)$ is the PDF of X, updated using up to $(i-1)$th tests. In the above update formula, $f_X^{(0)}(x)$ is the initial prior PDF, and the posterior PDF becomes a prior PDF for the next update.

In the view of Eqs. 3.32 and 3.33, it is possible to have two interesting observations. First, the Bayes' theorem becomes identical to the maximum likelihood estimate when there is no prior information; e.g., $f_X(x) = \text{constant}$. Second, the prior PDF can be applied either first or last. For example, it is possible to update the posterior distribution without prior information and then to apply the prior PDF after the last update.

When there are multiple data whose variability is normally distributed with a known variance and when the prior is also normally distributed, it is possible to calculate the exact expression of posterior distribution using the concept of conjugate distributions. Let N be the number of test data $\mathbf{y} = \{y_1, y_2, \ldots, y_N\}$, whose mean is $\bar{y}$ and the standard deviation of data is known to be σ_y. Then, similar to Eq. 3.29, the posterior distribution can be written as

$$f_X(x|Y=\mathbf{y}) \sim \exp\left[-\sum_{i=1}^{N} \frac{(y_i - x)^2}{2\sigma_y^2} - \frac{(x-\mu_0)^2}{2\sigma_0^2}\right]. \tag{3.34}$$

Similar to the case of a single test, it can be shown that the posterior distribution is normally distributed as

$$f_X(x|Y=\mathbf{y}) \sim N(\mu, \sigma^2), \tag{3.35}$$

where the mean and variance are, respectively, $\mu = w_N\mu_0 + (1-w_N)\bar{y}$ and $\sigma^2 = 1/(c_0 + c_N)$. In the above equation, $w_N = c_0/(c_0 + c_N)$ is the weight and $c_0 = 1/\sigma_0^2$ and $c_N = N/\sigma_y^2$ are the inverse of variance of prior and test data.

It is interesting to note that when there is no prior knowledge, i.e., non-informative prior, the posterior distribution becomes

$$f_X(x|Y=\mathbf{y}) \sim N\left(\bar{y}, \frac{\sigma_y^2}{N}\right). \tag{3.36}$$

That is, the mean of posterior is the same as the mean of data, and the variance is divided by the number of data. Note that Eq. 3.36 is identical to Eq. 3.8 for sampling uncertainty.

Example 3.10 Conjugate distributions with multiple data
Four tensile tests of steel coupons show 558, 567, 573, and 582 MPa for failure strengths. Use Bayes' theorem to calculate the posterior distribution of failure strength. It is known that test variability is normally distributed with standard deviation of 10 MPa, and the prior distribution is known to follow a normal distribution, $f_X(x) = N(550, 20^2)$.

Solution:
The mean and variance of four data are $\bar{y} = 570$ and $\sigma_y^2/N = 100/4 = 25$. Note that the variance is not calculated from the data because it is assumed that the variance of test is known. Due to more number of data, the weight becomes $w_N = c_0/(c_0 + c_N) = 0.0588$. Therefore, the mean of posterior distribution will be close to the mean of data; i.e., $\mu = w_N\mu_0 + (1 - w_N)\bar{y} = 568.8$ MPa. The variance of posterior distribution also reduces to $\sigma^2 = 1/(c_0 + c_N) = 23.5$. Therefore, the posterior distribution is given as

$$f_X(x|Y=\mathbf{y}) = \frac{1}{23.5\sqrt{2\pi}} \exp\left[-\frac{(x-568.8)^2}{47}\right].$$

Figure E3.6 shows the plots of prior, likelihood and posterior distribution. Due to the fact that four test data are available, the product of likelihoods shows much narrower distribution, and thus, the posterior distribution. As more data are used, the effect of prior distribution is reduced.

```
x=500:1:620; m0=550; s0=20;
y=[558, 567, 573, 582]; sy=10;
c0=1/s0^2; c=4/sy^2; w=c0/(c0+c);
ybar=mean(y); sli=sqrt(sy^2/4);
m=w*m0+(1-w)*mean(y); s=sqrt(1/(c0+c));
pdf_pr=exp(-((x-m0).^2)./(2*s0^2))/(sqrt(2*pi)*s0);
pdf_li=exp(-((x-ybar).^2)./(2*sli^2))/(sqrt(2*pi)*sli);
pdf_po=exp(-((x-m).^2)./(2*s^2))/(sqrt(2*pi)*s);
plot(x,pdf_pr,'-b',x,pdf_li,'-g',x,pdf_po,'-r');
```

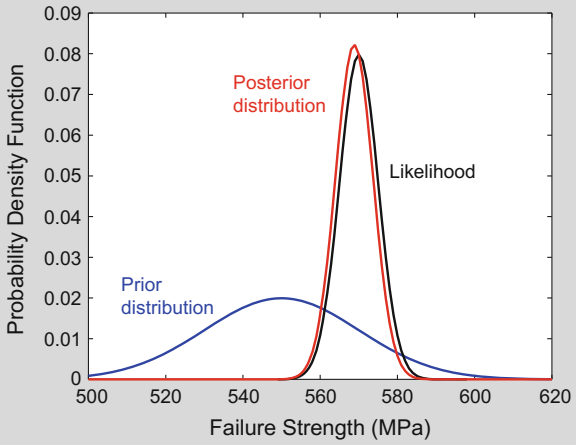


Fig. E3.6 Prior, likelihood and posterior distribution with multiple test data

An important advantage of Bayes' theorem over other parameter identification methods, such as the least-squares method and maximum likelihood estimate, is its capability to estimate the uncertainty structure of the identified parameters. These uncertainty structures depend on that of the prior distribution and likelihood function. Accordingly, the accuracy of posterior distribution is directly related to that of likelihood and prior distribution. Thus, the uncertainty in posterior distribution must be interpreted in that context.

3.4.4 Bayes' Theorem for Parameter Estimation

So far, Bayes' theorem has been used to obtain a posterior distribution using a prior distribution and experimental observation. It is important to note that in such a case, it is assumed that the direct observation is possible to update the uncertainty information. For example, to predict failure strength distribution, the failure strength of a coupon is directly measured. In many cases, however, measured quantity can be different from the quantity of interest. For example, in structural health monitoring, natural frequency is measured to estimate the change in stiffness due to damage. Therefore, engineers try to find the stiffness that can yield the same natural frequency observed in experiment. Conventionally, this is called calibration of parameters. This is very common in many engineering fields especially in engineering computation, because most engineering computation requires model parameters, which need to be identified based on experimental observations.

The major use of Bayes' theorem in this book is for the purpose of parameter estimation or calibration of model parameters. For example, as shown in Chap. 2, structural health monitoring systems can measure crack sizes at different flight

cycles. Then, using this information, the Paris model parameters are estimated first. Once the model parameters are identified, these parameters are used to predict the future behavior of crack sizes, which is the basic concept of prognostics. The conventional regression technologies can be used for identifying model parameters, but Bayes' theorem can provide more comprehensive information. It can provide not only the most likelihood value but also uncertainty associated with it. When multiple parameters are involved, the Bayes' theorem can also provide correlation between parameters.

For the purpose of parameter estimation, the vector of unknown model parameters is denoted as $\boldsymbol{\theta}$, while the vector of measured data is denoted as $\mathbf{y}$. Then, the Bayes' theorem in Eq. 3.27 can be written in the following form:

$$f(\boldsymbol{\theta}|\mathbf{y}) = \frac{f(\mathbf{y}|\boldsymbol{\theta})f(\boldsymbol{\theta})}{f(\mathbf{y})}. \tag{3.37}$$

As discussed before, the denominator in the above equation is independent of unknown parameters and it can be considered as a normalizing constant to make the integral of posterior PDF to be one. Therefore, the practical form of the Bayes' theorem can be written in the following form:

$$f(\boldsymbol{\theta}|\mathbf{y}) \propto f(\mathbf{y}|\boldsymbol{\theta})f(\boldsymbol{\theta}), \tag{3.38}$$

where $f(\mathbf{y}|\boldsymbol{\theta})$ is a likelihood function that is the PDF value at $\mathbf{y}$ conditional on given $\boldsymbol{\theta}$. $f(\boldsymbol{\theta})$ is the prior PDF of $\boldsymbol{\theta}$, which is updated to $f(\boldsymbol{\theta}|\mathbf{y})$, the posterior PDF of $\boldsymbol{\theta}$ conditional on $\mathbf{y}$.

The Bayes' theorem in Eq. 3.38 is also called Bayesian inference or Bayesian update. It is called Bayesian inference because the process is inferring unknown model parameters using observation. It is also called Bayesian update because the prior distribution is updated to the posterior distribution after having an observation.

The major difference in Bayesian parameter estimation from the previous Bayes' theorem is that the posterior distribution and likelihood represent different variables. That is, the posterior distribution is about model parameters, while the likelihood is measurement of a physical quantity. In the case of Bayes' theorem in the previous section, the posterior distribution and likelihood both are related to the same physical quantity, such as failure strength. In such a case, the likelihood simply represents test variability. However, when posterior and likelihood represent different quantities, a physical model must be employed to relate the model parameters with measured data. For example, in the case of estimating Paris model parameters using crack size data, the Paris model can be used to calculate the likelihood, that is the probability to obtain measured crack sizes $\mathbf{y}$ for given parameter values $\boldsymbol{\theta}$. This can be done by selecting different values of parameters $\boldsymbol{\theta}$ and calculating the probability of getting $\mathbf{y}$ when the parameters are given.

3.5 Bayesian Updating

In this section, a simple example is employed to demonstrate the process of Bayesian update. The basic process to obtain the posterior distribution is simply to multiply the likelihood function with the prior distribution. However, the readers should focus on the likelihood function, which looks similar to probability density function, but it is actually different from it. In the view of Eqs. 3.32 and 3.33, it is possible to implement Bayesian updating in two different ways when more than one data are available. Equation 3.32 shows the final expression of the posterior distribution that is obtained by multiplying likelihood of all data with the prior distribution. In this book, this method will be referred to as overall Bayesian method. On the other hand, Eq. 3.33 shows that Bayesian update is employed one data point at a time. In this case, the posterior distribution from the previous update is used as a prior distribution and likelihood includes only a single test data. In this book, this method will be referred to as recursive Bayesian method. Mathematically, these two Bayesian updating methods are equivalent and should yield the identical posterior distribution. However, they become different in numerical implementation, especially when a PDF is represented by samples. In this section, these two Bayesian updating methods are explained using a simple example.

3.5.1 Recursive Bayesian Update

In order to demonstrate the Bayesian updating process graphically, a simple case is considered where the prior is uniformly distributed as well as two data are assumed to be obtained when test variability is uniformly distributed. The left column in Fig. 3.6 graphically shows the Bayes' theorem in Eq. 3.38 with one unknown parameter (θ) and one data (y_1). For example, it can be assumed that y_1 is the measured crack size, and θ is the estimated mean of crack size.

The prior PDF of θ can be set based on the previous information/knowledge, or be non-informative. In the illustration, it is assumed that the prior is uniformly distributed between certain intervals. Any probability models, such as uniform, normal, or beta distribution can be employed for the likelihood function. It is assumed that the data are also uniformly distributed within the range of $\pm v$. That is, if the mean crack size is given as θ, then the test variability is uniformly distributed in the range of $[\theta - v,\ \theta + v]$. If the test data y_1 is within the range, then the likelihood will have a constant value, otherwise it will be zero. In order to build the posterior distribution, the likelihood needs to be calculated for all possible values of θ for given y_1.

As shown in the left column in Fig. 3.6, for example, when $\boldsymbol{\Theta} = \theta^a$, the test variability is in the range of $[\theta^a - v,\ \theta^a + v]$, but the test data y_1 is out of the range. Therefore, the likelihood function is zero; i.e., $f(y_1|\theta^a) = 0$. This means that if the

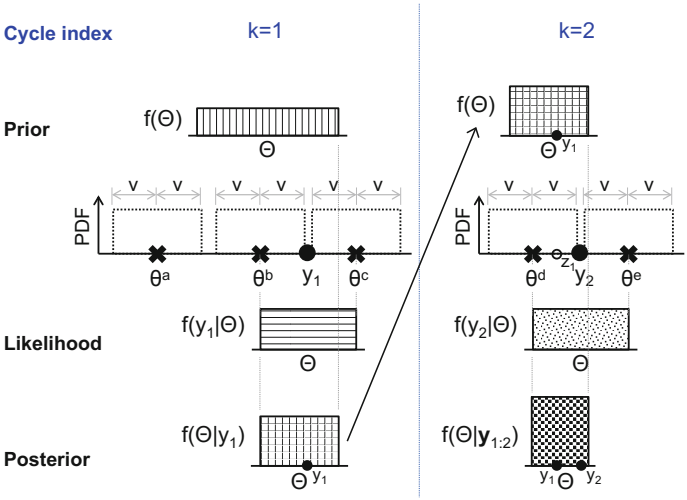


Fig. 3.6 Recursive Bayesian updating process

mean crack size is θ^a, the probability to have a measured crack size at y_1 is zero because the distance between θ^a and y_1 is larger than the possible variability v of test. The range of θ to be nonzero probability at y_1 is between θ^b and θ^c, which is in the distance of v from y_1. Therefore, the likelihood function as shown in the figure can be obtained. It is noted that the likelihood function looks like a PDF whose center is at y_1 and the range is $[y_1 - v, y_1 + v] = [\theta^b, \theta^c]$. Therefore, it might be confused that the likelihood is a PDF constructed around the test data with test variability. This happens because the uniform distribution of test variability is symmetry. If the test variability is not symmetry, the likelihood function would have been different from the PDF of test variability. Since the posterior is obtained by multiplying the prior and the likelihood, the posterior distribution becomes the overlapped range of θ from the two distributions, whose distribution is narrower than the prior one.

When another data y_2 is available, the distribution of the unknown parameter can be updated in two ways: recursive Bayesian update or overall Bayesian update. Figure 3.6 illustrates the recursive Bayesian update, where each test data is used to update the posterior distribution. Therefore, when n_y number of data are available, the recursive Bayesian update needs to be performed n_y times. The right column in Fig. 3.6 shows the recursive Bayesian process at cycle index $k = 2$. The posterior at $k = 1$ is used as the prior at $k = 2$, and the same process is repeated with a new data. For a given test data at y_2, the new likelihood function is nonzero when the parameter θ of mean crack size is in the range of $[\theta^d, \theta^e]$. Therefore, the posterior distribution is the common range between the prior distribution and likelihood function.

The recursive Bayesian update can be written as

$$\begin{aligned} f(\boldsymbol{\theta}|y_1) &\propto f(y_1|\boldsymbol{\theta})f(\boldsymbol{\theta}) \\ f(\boldsymbol{\theta}|\mathbf{y}_{1:2}) &\propto f(y_2|\boldsymbol{\theta})f(\boldsymbol{\theta}|y_1) \\ &\vdots \\ f(\boldsymbol{\theta}|\mathbf{y}_{1:k}) &\propto f(y_k|\boldsymbol{\theta})f(\boldsymbol{\theta}|\mathbf{y}_{1:k-1}). \end{aligned} \tag{3.39}$$

As explained before, in the recursive Bayesian update, the posterior distribution from the previous data is used as a prior distribution for the next data. In the above equation, the notation $\boldsymbol{\theta}|\mathbf{y}_{1:k}$ means $\boldsymbol{\theta}|y_1, y_2, \ldots, y_k$. Note that the posterior distribution of $\boldsymbol{\theta}$ becomes narrower as more data are used.

Example 3.11 Recursive Bayesian update for failure strength
Three tensile tests are performed to measure the failure strength of a material. After normalization, the three measured failure strengths are 1.05, 1.10, and 1.15. Use the recursive Bayesian update to calculate the posterior distribution of failure strength. The prior distribution is assumed to be uniformly distributed $\sim U(0.9, 1.1)$. Also, the test variability is uniformly distributed $\sim U(-0.15, 0.15)$.

Solution:
Figure E3.7a shows the prior distribution and the likelihood function for the first data $y_1 = 1.05$. The likelihood function is a constant in the range of [0.9, 1.2]. Since the prior is within the likelihood, the posterior becomes identical to the prior. That is, the first data cannot improve any information for the failure strength. Therefore, the posterior distribution is uniformly distributed in the range of [0.9, 1.1].

When the second data is used, Fig. E3.7b shows the prior, likelihood and posterior distribution. The prior is the same as the posterior in the previous update, that is [0.9, 1.1]. The likelihood is constant function in the range of [0.95, 1.25]. The multiplication of the prior and likelihood becomes the posterior distribution in the range of [0.95, 1.1].

For the third data, the prior is in the range of [0.95, 1.1], and the likelihood is in the range of [1.0, 1.3], which yields the posterior distribution in the range of [1.0, 1.1]. Note that the range of posterior distribution is reduced from the initial prior distribution by 50 %.

The following MATLAB code shows how to calculate and plot the posterior distributions using three measured failure strengths.

```
x=0.8:0.001:1.3;
prior=pdf('unif',x,0.9,1.1);
% test 1
a=0.15;
test1=1.05;
likelihood=pdf('unif',test1,x-a,x+a);
posterior=prior.*likelihood;
area=sum(posterior)*0.001;
posterior=posterior/area;
figure(1);
plot(x,prior,'r-',x,likelihood,'b--',x,posterior,'m-');
axis([0.85 1.25 0 7])
% test 2
test2=1.10;
prior=posterior;
likelihood=pdf('unif',test2,x-a,x+a);
posterior=prior.*likelihood;
area=sum(posterior)*0.001;
posterior=posterior/area;
figure(2);
plot(x,prior,'r-',x,likelihood,'b--',x,posterior,'m-');
axis([0.85 1.25 0 11])
% test 3
test3=1.15;
prior=posterior;
likelihood=pdf('unif',test3,x-a,x+a);
posterior=prior.*likelihood;
area=sum(posterior)*0.001;
posterior=posterior/area;
figure(3);
plot(x,prior,'r-',x,likelihood,'b--',x,posterior,'m-');
axis([0.85 1.25 0 11])
```

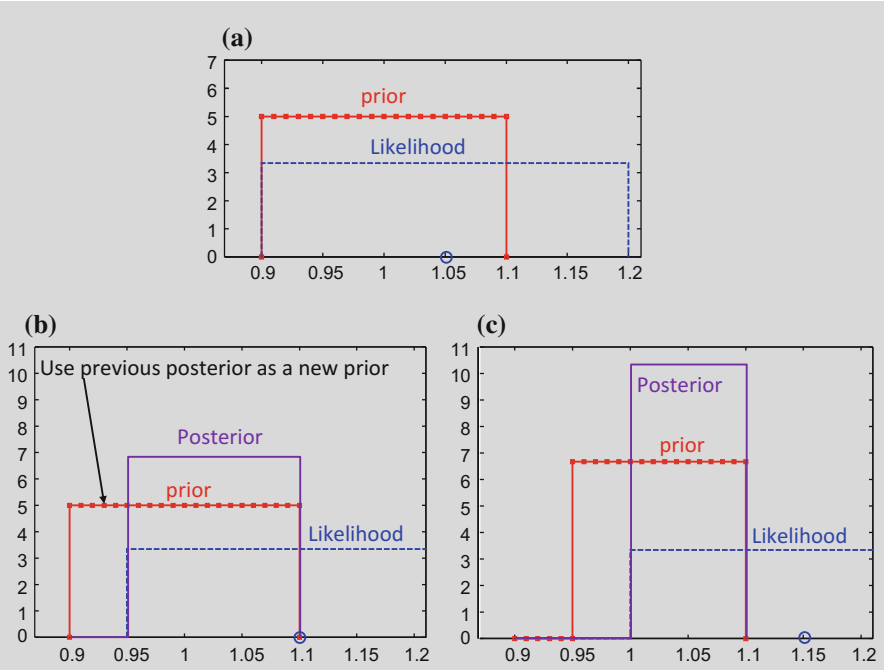


Fig. E3.7 Recursive Bayesian update of failure strength. **a** Update with the first data. **b** Update with the second data, **c** Update with the third data

3.5.2 *Overall Bayesian Update*

Different from the recursive Bayesian update, the overall Bayesian update utilizes all data simultaneously, and updates the posterior distribution at the same time, which is shown in Fig. 3.7. The procedure starts from the same prior and the same type of likelihood function that are used in Fig. 3.6. The difference is the data used in the likelihood; both of y_1 and y_2 are considered together to determine the likelihood function. As shown in Fig. 3.7, the lower and upper bounds of θ should be θ^f and θ^g to include both data y_1 and y_2, which corresponds to $f(y_1|\theta) \times f(y_2|\theta)$. In the case of k data (cycle index, k can be considered as the number of data, n_y in the overall update), the posterior distribution becomes

$$\begin{aligned} f(\boldsymbol{\theta}|\mathbf{y}_{1:k}) &\propto f(y_1|\boldsymbol{\theta}) \times f(y_2|\boldsymbol{\theta}) \times \cdots \times f(y_k|\boldsymbol{\theta}) \times f(\boldsymbol{\theta}) \\ &= f(\mathbf{y}_{1:k}|\boldsymbol{\theta}) f(\boldsymbol{\theta}). \end{aligned} \tag{3.40}$$

Eventually, the posterior obtained by multiplying the prior and the likelihood becomes the same as the one in Fig. 3.6.

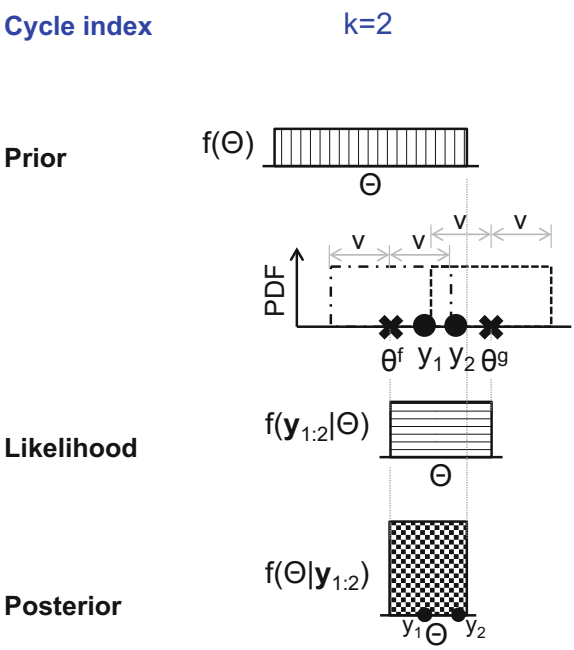


Fig. 3.7 Overall Bayesian updating process

Example 3.12 Overall Bayesian update
Repeat Example 3.11 using the overall Bayesian update.

Solution:
The posterior distribution is expressed by a single equation by multiplying the prior and all likelihood functions in the overall Bayesian update. Therefore, the prior distribution is uniformly distributed $\sim U(0.9, 1.1)$. The likelihood is also uniformly distributed $\sim U(1.0, 1.2)$ that is obtained by multiplying three likelihoods given in Fig. E3.7. Finally, the posterior is in the range of [1.0, 1.1], which is shown in Fig. E3.8 and can be obtained from the following MATLAB code by replacing it for the two lines, `test1` and `likelihood` given in Example 3.11.

```
x=0.8:0.001:1.3;
prior=pdf('unif',x,0.9,1.1);
test=[1.05 1.10 1.15]; ny=length(test);
likelihood=1;
for k=1:ny
 likelihood=pdf('unif',test(k),x-a,x+a).*likelihood;
end
areaL=sum(likelihood)*0.001;
likelihood=likelihood/areaL;
posterior=prior.*likelihood;
area=sum(posterior)*0.001;
posterior=posterior/area;
plot(x,prior,'r-',x,likelihood,'b--',x,posterior,'m-');
axis([0.85 1.25 0 11])
```

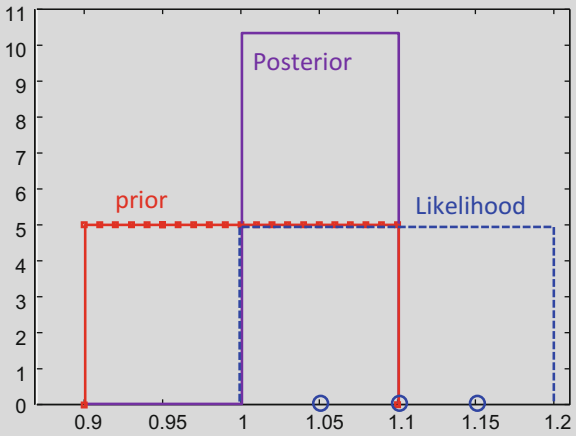


Fig. E3.8 Overall Bayesian update of failure strength

3.6 Bayesian Parameter Estimation

In the previous section, Bayesian update for a simple distribution parameter is shown. However, the main purpose of Bayesian update in this text is to update the model parameters. As an example for the Bayesian estimation of model parameters, the same example used in Chap. 2, Eq. 2.9 is considered again, which has three model parameters to be identified, as

$$z(t;\boldsymbol{\theta}) = \theta_1 + \theta_2 Lt^2 + \theta_3 t^3, \quad \boldsymbol{\theta} = \{\,\theta_1 \quad \theta_2 \quad \theta_3\,\}^{\mathrm{T}}. \tag{3.41}$$

First of all, the type of likelihood function is assumed as a normal distribution

$$f(y|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(\mu-y)^2}{2\sigma^2}\right), \tag{3.42}$$

where μ and σ are the mean and the standard deviation, which are probability parameters of normal distribution. The degradation data y is normally distributed with mean of μ and standard deviation of σ, which corresponds that the measurement error $\mu - y$ is normally distributed with zero mean and σ. Therefore, μ becomes the model output from Eq. 3.41, and σ is the measurement error that should be estimated along with the unknown parameters. Therefore, the equivalent form of Eq. 3.42 for this example can be expressed as

$$\begin{aligned} f(y_k|\boldsymbol{\theta}) &= \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(z_k-y_k)^2}{2\sigma^2}\right) \\ &= N\left(y_k; z_k, \sigma^2\right), \; z_k = \theta_1 + \theta_2 Lt_k^2 + \theta_3 t_k^3, \; \sigma = \theta_4. \end{aligned} \tag{3.43}$$

where k is time index and $N(y_k; z_k, \sigma^2)$ is the value at y_k of normal PDF $N(z_k, \sigma^2)$. The degradation data and model output depend on time as system is degraded, but the model parameters and the standard deviation are assumed as time-independent. Since the normal distribution is an exponential function, its multiplication of several terms can be expressed with a sum of all exponents. Therefore, the likelihood function for the overall Bayesian update can be written as

$$\begin{aligned} f\left(\mathbf{y}_{1:n_y}|\boldsymbol{\theta}\right) &= f(y_1|\boldsymbol{\theta}) \times f(y_2|\boldsymbol{\theta}) \times \cdots \times f\left(y_{n_y}|\boldsymbol{\theta}\right) \\ &= \frac{1}{\left(\sigma\sqrt{2\pi}\right)^{n_y}} \exp\left(-\frac{\sum_{k=1}^{n_y}\left(z_k - y_k\right)^2}{2\sigma^2}\right). \end{aligned} \tag{3.44}$$

The total number of unknown parameters to be identified is four including measurement error, σ. The prior distribution should be given, assumed, or not considered for each parameter. Let us assume the distribution of each parameter is the uniform distribution formulated as

$$f(\theta) = U(a,b) = \begin{cases} 1/(b-a) & \text{for } \theta \in [a,b] \\ 0 & \text{otherwise} \end{cases},$$

where a, b are the probability parameters of uniform distribution. The prior distribution is obtained by multiplying all prior distributions as

$$f(\boldsymbol{\theta}) = \prod_{i=1}^{n_p} f(\theta_i) = \prod_{i=1}^{n_p} U(a_i, b_i), \tag{3.45}$$

where n_p is the number of unknown parameters, in this example $n_p = 4$.

Consequently, the posterior becomes a multiplication of the likelihood and prior equations in Eqs. 3.43 and 3.45. When n_y data are given, the unknown parameters can be updated in two ways, a recursive method by using Eq. 3.39 and an overall method by using Eq. 3.40.

Let us assume that only θ_1 in Eq. 2.9 is the unknown parameter by using the true values for the others and $\sigma = 2.9$. The loading condition $L = 1$ and five no noisy data are given in Table 2.1. The likelihood function is a normal distribution, and the prior of the unknown parameter is uniformly distributed between 0 and 10. The posterior distribution based on the recursive method in Eqs. 3.39 and 3.43 can be obtained as

```
t=[0    1    2    3    4];
y=[5.0  5.3  6.6  9.5  14.6];

sigma=2.9;
theta=-5:0.01:15; ng=length(theta);

%[Recursive]
prior=unifpdf(theta,0,10);
for k=1:5;
 z(k,:)=theta+0.2*t(k)^2+0.1*t(k)^3;
 postS(k,:)=normpdf(y(k),z(k,:),sigma).*prior;
 prior=postS(k,:);
end
plot(theta,postS./repmat(0.01*sum(postS,2),1,ng));
```

The posterior distributions at each time are shown in Fig. 3.8a. When $t = 0$ ($k = 1$), the given model is the same as $\theta\,(f(0;\theta) = \theta)$. Therefore, the unknown parameter can be identified accurately with just one data without noise. The blue dashed curve shows that the most likely value for θ is 5, but the distribution represents uncertainty caused by using just one data whose measurement error, $\sigma = 2.9$. The cut-off at both ends is an effect of the prior distribution, $U(0, 10)$. The results show that as more data are used, the distribution becomes narrower.

The red solid curve at $k = 5$ is the finally updated posterior distribution with five data, which can be obtained with the overall method based on the Eqs. 3.40 and 3.44.

```
%[Overall]
ny=length(y);
for i=1:ng
 z=theta(i)+0.2*t.^2+0.1*t.^3;
 prior=unifpdf(theta(i),0,10);
 postO(1,i)=(sigma*sqrt(2*pi))^-ny* ...
                        exp(-0.5/sigma^2*(z-y)*(z-y)')*prior;
 %postO(1,i)=prod(normpdf(y,z,sigma))*prior;
end
plot(theta,postO/(0.01*sum(postO)));
```

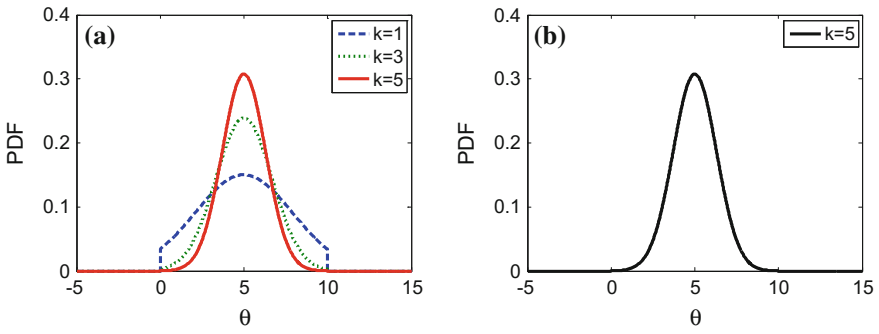


Fig. 3.8 Posterior distribution from Bayesian updating for one parameter. **a** recursive Bayesian update, **b** overall Bayesian update

The result is in Fig. 3.8b, which is the exactly same as the red solid curve in Fig. 3.8a.

Example 3.13
Parameter estimation based on Bayesian updating

Obtain and plot the posterior distribution of unknown parameters θ_2 and θ_3 in Eq. 2.9 based on Bayesian approach (a) recursive method and (b) overall method. Assume $\theta_1 = 5$, non-informative prior, and normal distribution with $\sigma = 2.9$ for the likelihood function. Use five noisy data in Table 2.2 in the case of $L = 1$ (this is the same problem as Example 2.6)

Solution:

The MATLAB code **[Recursive]** and **[Overall]** can be employed to obtain the posterior distribution of the unknown parameters. Since the number of unknown parameters is two here, a modification is required to obtain a 2-D joint posterior distribution. The 2-D grid points are generated with `meshgrid`, at which posterior probability is calculated in the same manner as **[Recursive]** and **[Overall]**, and then `contour` is used to plot the results. As the results, Figure E3.9 is obtained with the following code.

```
t=[0    1    2    3    4];
y=[6.99 2.28 1.91 11.94 14.60];
sigma=2.9;
ng=200;
th2B=[-10 10]; th3B=[-2 2];
[theta2 theta3]=meshgrid(linspace(th2B(1),th2B(2),ng)', ...
    linspace(th3B(1),th3B(2),ng));

%%% Recursive
prior=1;
for k=1:5;
 z(k,:)=5+theta2(:)*t(k)^2+theta3(:)*t(k)^3;
 postS(k,:)=normpdf(y(k),z(k,:),sigma).*prior;
 prior=postS(k,:);
 C=sum(postS(k,:))* ...
        (th2B(2)-th2B(1))/(ng-1)*(th3B(2)-th3B(1))/(ng-1);
 contour(theta2,theta3,reshape(postS(k,:)/C,ng,ng)); hold on;
end
plot(0.2,0.1,'pk','markersize',18);

%%% Overall
ny=length(y);
for i=1:ng; for j=1:ng
 z=5+theta2(i,j)*t.^2+theta3(i,j)*t.^3;
 postO(i,j)=(sigma*sqrt(2*pi))^-ny* ...
                      exp(-0.5/sigma^2*(z-y)*(z-y)');
end; end
C=sum(sum(postO))* ...
  (th2B(2)-th2B(1))/(ng-1)*(th3B(2)-th3B(1))/(ng-1);
figure;
contour(theta2,theta3,postO/C); hold on;
plot(0.2,0.1,'pk','markersize',18);
```

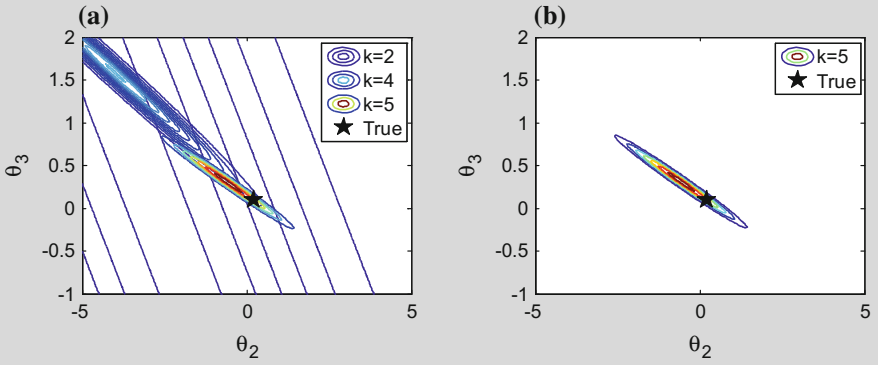


Fig. E3.9 Joint posterior distribution from Bayesian update for two parameters. **a** Recursive Bayesian update, **b** overall Bayesian update

3.7 Generating Samples from Posterior Distribution

The Bayesian parameter estimation in the previous section shows how to obtain the posterior distribution, i.e., PDF, of unknown parameters in terms of the prior distribution and likelihood. Except for a small number of exceptional cases when the prior and posterior are conjugate, the posterior distribution cannot be expressed in terms of standard probability distributions. It is often expressed using multiplication of different functions.

In prognostics, once the posterior distribution of degradation model parameters is obtained, it is used to calculate degradation behavior and predict the remaining useful life. In general nonlinear degradation models, it is difficult to analytically propagate uncertainty in the model parameters to degradation model. Instead, samples that follow the posterior distribution of model parameters are generated first, and then, each sample of model parameters is substituted into the degradation model to calculate a sample of remaining useful life. If this substitution is repeated for all samples of model parameters, they can represent the distribution of remaining useful life, which is the eventual goal of prognostics. Therefore, it is important to generate samples that follow the posterior distribution of model parameters. In this section, different ways of drawing samples from the posterior distribution are explained. The inverse CDF method is a well-known method to generate samples from non-standard probability distributions. The grid approximation method is a discrete version of the inverse CDF method (Gelman et al. 2004).

3.7.1 Inverse CDF Method

When a CDF of a continuous posterior distribution is available with a closed form; samples from the posterior can easily be drawn using the inverse CDF method,

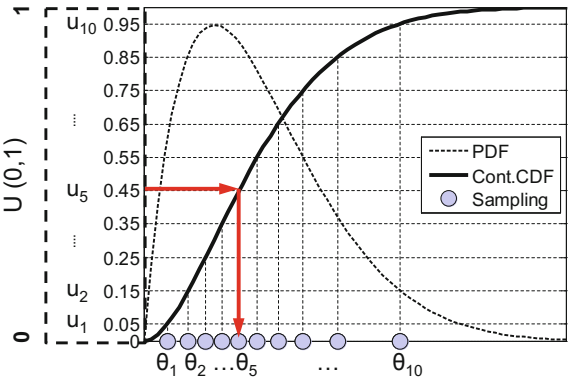


Fig. 3.9 Illustration of inverse *CDF* method

which is illustrated in Fig. 3.9. First, a CDF is constructed from the posterior PDF by integrating the area under the PDF, which is illustrated as solid curve in the figure. Therefore, this method is only applicable when a closed-form expression of CDF is available.

Second, a uniformly distributed random variable $u \sim U(0,1)$ is mapped into the CDF. The basic idea is that the range of CDF is the same as that of u, and it is a monotonically increasing function. Therefore, the following relation can be established:

$$F(\theta) = u \Leftrightarrow F^{-1}(u) = \theta. \tag{3.46}$$

This means that a sample θ can be obtained by calculating the inverse of a CDF. A random sample is generated from uniform distribution, which becomes a CDF value (e.g., $u = u_5 = 0.45$ in the figure). Then, a sample (θ_5) is drawn by using Eq. 3.46. By repeating this process n_s times, n_s samples are obtained (in the illustration, $n_s = 10$). It is shown that the samples are more concentrated in the region where the PDF is higher.

Example 3.14
Inverse CDF method

Draw 5000 samples from $N(5, 2.9^2)$ by using the inverse CDF method, and compare the result with exact PDF.

Solution:
Since the CDF of normal distribution and its inverse can be calculated from its closed-form equation or by using MATLAB function, this problem can be solved easily. With `mu=5, sig=2.9`, the first three lines of MATLAB code are a part of the inverse CDF method.

```
ns=5000; % num. of samples
u=rand(1,ns);
thetaS=norminv(u,mu,sig); % samples by inv. CDF

[fre val]=hist(thetaS,30); bar(val,fre/ns/(val(2)-val(1)))

%%% exact PDF
theta=linspace(-10,20,200);
pdf=normpdf(theta,mu,sig);
hold on; plot(theta,pdf,'r');
```

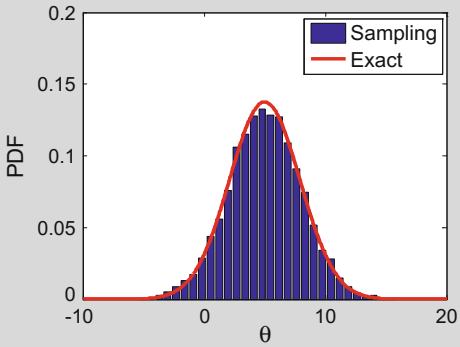


Fig. E3.10 Sampling result using the inverse CDF method

Note that since a closed-form expression of CDF is available for this example, samples can be drawn very easily by using inverse CDF method. It is, however, not easy to obtain a CDF equation and its inverse in most cases. In this case, approximation method is employed, which is explained in Sect. 3.7.2.

3.7.2 Grid Approximation Method: One Parameter

The inverse CDF method is a convenient method to generate samples from an arbitrary distribution, but the closed-form expression of CDF is rarely available in practice. Especially when the posterior distribution is given in terms of the product of likelihood and prior, the closed-form expression of PDF is available, but

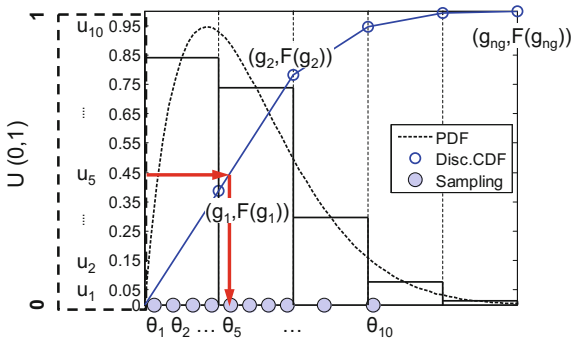


Fig. 3.10 Illustration of grid approximation: one parameter

analytical integration of PDF is not trivial. When a posterior CDF is not given as a closed-form expression, a discrete version of the inverse CDF method called the grid approximation method can be employed, which is illustrated in Fig. 3.10. This is the same concept as the inverse CDF method, but the CDF is approximated by piecewise linear polynomials at several grid points. For example, the CDF can be approximated by numerically integrating the posterior PDF. In Fig. 3.10, the PDF is divided into n_g ($n_g = 5$ in this illustration) to calculate CDF, which means available CDF information is at the five points denoted as blue circles. In this case, the samples in the inverse CDF method are approximately obtained, either selecting the closest CDF value or using an interpolation. For example, when $u = u_5$, $F(g_1)$ is selected as the closest values to u_5, then g_1 is obtained as θ_5, or θ_5 can be obtained by interpolating neighboring two CDF data as

$$\theta_5 = g_1 + \frac{u_5 - F(g_1)}{F(g_2) - F(g_1)} \times (g_2 - g_1). \tag{3.47}$$

Example 3.15
Grid approximation method for one parameter

Use the grid approximation to draw samples ($n_s = 5000$) from the posterior distribution in Fig. 3.8b when $n_g = 10$ and 50.

Solution:
The sampling results in Fig. E3.11 are obtained from the MATLAB code below, and the exact PDF can be plotted with the MATLAB code **`[Recursive]`** or **`[Overall]`**.

```
t=[0    1    2    3    4];
y=[5.0  5.3  6.6  9.5  14.6]; ny=length(y);
sigma = 2.9;

ng=10; % num. of grid
thL=-5; thU=15; theta = linspace(thL,thU,ng+1);
wt=(thU-thL)/ng/2; % half-grid width
thetaP=wt+theta(1:ng); % point for pdf

post=unifpdf(thetaP,0,10);
for k=1:ny;
 f=thetaP+0.2*t(k)^2+0.1*t(k)^3;
 post=normpdf(y(k),f,sigma).*post;
end
pdf=post/((thU-thL)/(ng-1)*sum(post));
cdf(2:ng+1)=cumsum(pdf)/sum(pdf);

ns=5000;
for i=1:ns
 u=rand;
 loca=find(cdf>=u,1);

 thetaS(i)=interp1([cdf(loca) cdf(loca -1)], ...
              [theta(loca) theta(loca -1)],u); % Eq. 3.47
end

[fre val]=hist(thetaS,30); bar(val,fre/ns/(val(2)-val(1)))
```

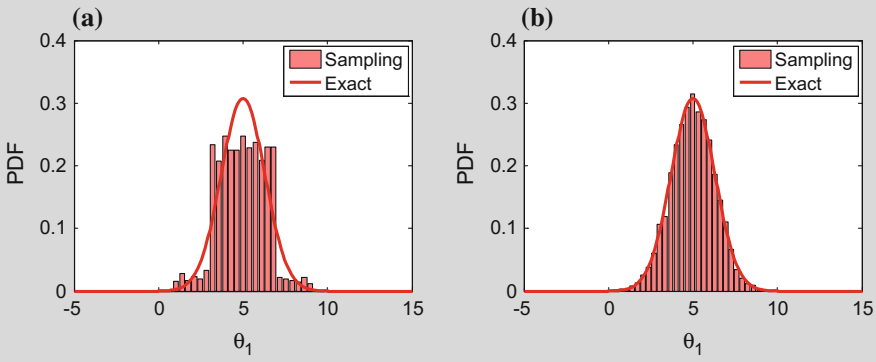


Fig. E3.11 Sampling results based on grid method: one parameter. **a** $n_g = 10$, **b** $n_g = 50$

As the number of grids increases, the sampling results become close to the exact PDF by reducing the approximation error. It is, however, inefficient when the number of gird increases especially with increase in the number of unknown parameters.

3.7.3 Grid Approximation: Two Parameters

In Sect. 3.7.2, the grid approximation method is presented when the posterior distribution is available for a single parameter. However, when the model has multiple parameters, the Bayesian update will yield the joint posterior distribution of all parameters. This joint posterior distribution also shows the correlation between parameters.

In this section, a method of generating samples from a joint PDF with two parameters is explained. A joint PDF of two parameters can be expressed using marginal PDF and conditional PDF as

$$f_{A,B}(a,b) = f_A(a|B=b)f_B(b) = f_B(b|A=a)f_A(a),$$

where $f_A(a)$ and $f_B(b)$ are marginal PDFs of A and B, as

$$f_A(a) = \int_{-\infty}^{\infty} f_{A,B}(a,b)\mathrm{d}b$$

$$f_B(b) = \int_{-\infty}^{\infty} f_{A,B}(a,b)\mathrm{d}a,$$

and $f_A(a|B=b)$ and $f_B(b|A=a)$ are conditional PDFs.

The following procedure explains the steps to generate samples from the grid approximation method for two parameters:

Step 1. Draw a sample from a marginal PDF of A as the same way explained in Sect. 3.7.2.
Step 2. Construct a PDF of B conditional on the sample of A in Step 1.
Step 3. Draw a sample from the PDF in Step 2 as the same way explained in Sect. 3.7.2.

Repeat Steps 1–3 n_s times.

Example 3.16
Grid approximation method for two parameters

Use the grid approximation to draw samples ($n_s = 5000$) from the posterior distribution in Fig. E3.9b when $n_g = 10$ and 50. Assume the ranges of the two parameters are $\theta_2 \in [-5, 5]$, $\theta_3 \in [-1, 2]$.

Solution:
The sampling results in Fig. E3.12 are obtained from the following MATLAB code that is based on the solutions in Examples 3.13 and 3.15. Additionally, the steps for marginal PDF of θ_2, given as $f_{\Theta_2}(\theta_2) = \int_{-1}^{2} f_{\Theta_2,\Theta_3}(\theta_2,\theta_3)d\theta_3$ and PDF of θ_3 conditional on θ_2 are required.

```
ng=50; % num. of grid
ns=5000; % num. of samples

t = [0    1    2    3    4];
y=[6.99 2.28 1.91 11.94 14.60]; ny=length(y);
sigma = 2.9;
thL=[-5; -1]; thU=[5; 2];

theta(1,:)=linspace(thL(1),thU(1),ng+1);
theta(2,:)=linspace(thL(2),thU(2),ng+1);

wt=(thU-thL)/ng; % grid width
thetaP=repmat(wt/2,1,ng)+theta(:,1:ng); % point for pdf
[thetaP2 thetaP3]=meshgrid(thetaP(1,:)',thetaP(2,:));

%%% joint PDF
post=1;
for k=1:ny;
 z=5+thetaP2(:)*t(k)^2+thetaP3(:)*t(k)^3;
 post=normpdf(y(k),z,sigma).*post;
end
C=sum(post)*wt(1)*wt(2);
jPdf=reshape(post/C,ng,ng);

%%% marginal PDF of theta 2
mPdf2=sum(jPdf)*wt(2);
mCdf2(2:ng+1)=cumsum(mPdf2*wt(1));

for i=1:ns
 u=rand;
 loca=find(mCdf2>=u,1);
 thetaS(1,i)=interp1([mCdf2(loca) mCdf2(loca-1)], ...
               [theta(1,loca) theta(1,loca-1)],u); % Eq. 3.47

 %%% PDF of theta 3 conditional on theta2
 post=1;
 for k=1:ny;
  z=5+thetaS(1,i)*t(k)^2+thetaP(2,:)*t(k)^3;
  post=normpdf(y(k),z,sigma).*post;
 end
 C=sum(post)*wt(2);
 mPdf3=post/C;
 mCdf3(2:ng+1)=cumsum(mPdf3*wt(2));
 u=rand;
 loca=find(mCdf3>=u,1);
```

```
 thetaS(2,i)=interp1([mCdf3(loca) mCdf3(loca-1)], ...
                [theta(2,loca) theta(2,loca-1)],u); % Eq. 3.47
end

figure;
[fre val]=hist(thetaS(1,:),30);
bar(val,fre/ns/(val(2)-val(1)))
figure;
[fre val]=hist(thetaS(2,:),30);
bar(val,fre/ns/(val(2)-val(1)))
figure;
plot(thetaS(1,:),thetaS(2,:),'.')
```

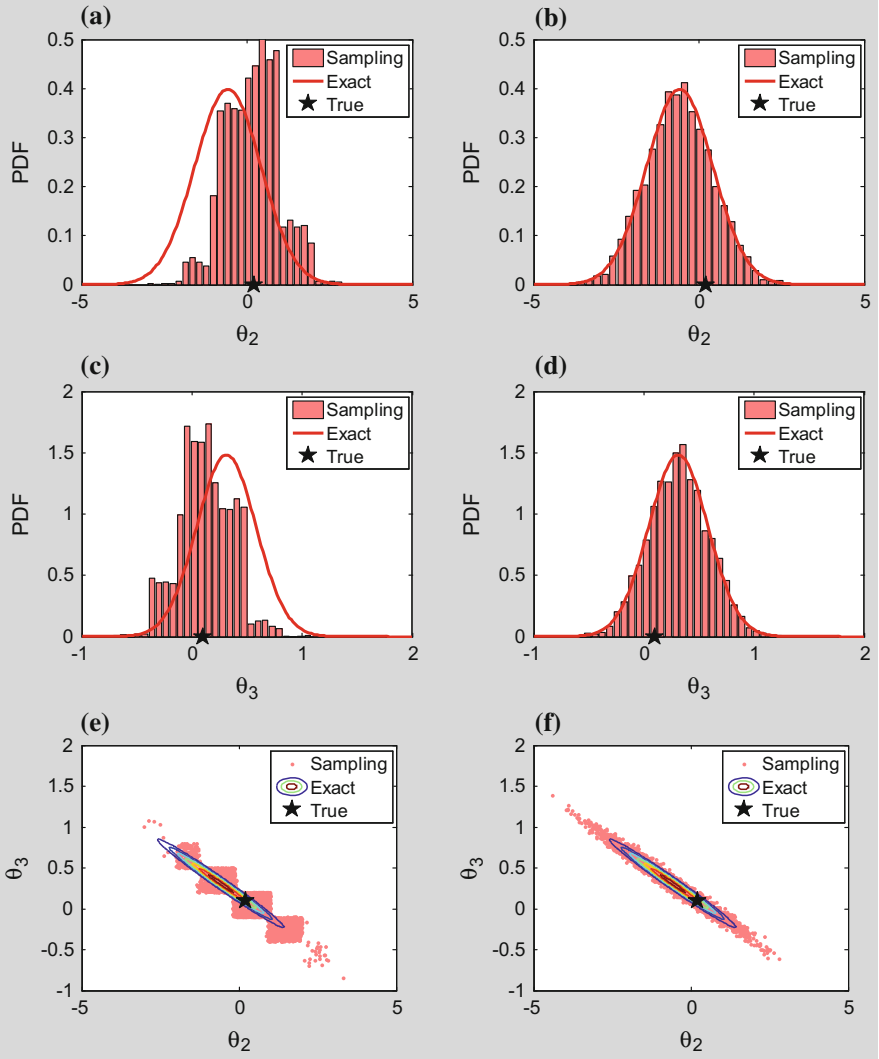


Fig. E3.12 Sampling results based on the grid method: two parameters. **a** θ_2, $n_g = 10$, **b** θ_2, $n_g = 50$, **c** θ_3, $n_g = 10$, **d** θ_3, $n_g = 50$, **e** correlation, $n_g = 10$, **f** correlation, $n_g = 50$

Although the grid approximation method is useful for many applications, it is difficult to find the correct location and scale for the grid points. Also, the method is relatively easy for one or two parameters, but it becomes quickly difficult for high-dimensional problems, where computing at every point in a dense multidimensional grid becomes prohibitively expensive. Lastly, the correlation between parameters can be complicated when several sets of correlation exist. In such a case, it is difficult to find interval visually in multidimensional space.

3.8 Exercises

P3.1 Repeat Example 3.2 when the estimated mean is normally distributed as $M_{est} \sim N(220, 10^2)$. Plot the PDF of probability of failure and calculate its mean and 90th percentile values.

P3.2 Random variable X has the PDF of $f_X(x) = 12(x^2 - x^3)$ for x in [0, 1]. Calculate the mean, median, mode and CDF of X.

P3.3 Show that when $n \to \infty$, $\mathrm{M_{est}} = \mu_{test}$ and $\Sigma^2_{\mathrm{est}} = \sigma^2_{\mathrm{test}}$ in Eqs. 3.8 and 3.9. Assume $\mu_{\mathrm{test}} = 250\,\mathrm{MPa}$ and $\sigma_{\mathrm{test}} = 25\,\mathrm{MPa}$.

P3.4 Generate 100 samples of failure strength from $S_{\mathrm{true}} \sim N(220, 20^2)$. Assuming that the true distribution is unknown, estimate the true mean and standard deviation using these samples. Using the estimated distribution of mean and standard deviation, plot the distribution of probability of failure using 100,000 samples from epistemic uncertainty. Use applied stress R = 190. Calculate the mean and 90th percentile of probability of failure

P3.5 Repeat Problem P3.4 using the combined method. Compare the calculated probability of failure with the mean of probability of failure calculated in Problem P3.2.

P3.6 A building can have structural damage (D) during its lifetime only from fire (F), high wind (W), and strong earthquakes (E). Thus, F, W, and E are collectively exhaustive events. Further assume that the building will not be structurally damages simultaneously by F, W, and E, thus making them mutually exclusive events. The probabilities of structural damage to the building if these events occur are estimated to be 0,005, 0.01, and 0.05, respectively. The probabilities of occurrence of F, W, and E during the life of the building are 0.5, 0.3, and 0.2, respectively. (a) What is the probability that the building will suffer structural damage during its lifetime? (b) If the building has suffered structural damage, what is the probability that it was caused by F? By W? By E?

P3.7 A concentrated load on a cantilever beam may be placed in either location A or B, with probabilities P(A) = 0.3 and P(B) = 0.7. If the load is placed at A, the probability of bending failure of the beam is 0.01, and the probability of shear failure is 0.001. If the load is placed at B, the probability of bending failure of the

beam is 0.02, and the probability of shear failure remains the same. If the beam has shear failure, then the probability of bending failure is 0.9. What is the overall probability of failure of the beam?

P3.8 Find how many samples of normally distributed numbers you need in order to estimate the mean and standard deviation with an error that will be less than 10 % of the true standard deviation most of the time.

P3.9 In the Ebola epidemic in Example 3.6, what is the probability of being infected when test shows negative?

P3.10 A passenger can travel from home to another city by car (C), ship (S), plane (F), or train (T). In a given year, the passenger made such a trip 80, 15, 100, and 50 times by C, S, F, and T, respectively. The probability of an accident (A) during a trip using these modes of transportation is estimated to be 1E-5, 5E-5, 1E-6, and 5E-5, respectively. (a) What is the probability of an accident during a trip? (b) What is the probability of an accident in the next 10 trips? (c) If there was an accident, what is the probability that the passenger was traveling by car?

P3.11 A building can suffer structural damage by fire (event F) or strong earthquakes (event E). Let F and E denote the events, with corresponding probabilities 0.005 and 0.05, respectively. F and E are statistically independent events. Calculate the probability of structural damage to the building.

P3.12 A bridge can be damaged by failure in the foundation (F) or in the superstructure (S). The corresponding failure probabilities for a particular bridge are estimated to be 0.05 and 0.01, respectively. Also, if there is foundation failure, then the probability that the superstructure will also suffer some damage is 0.50. Calculate probability $P(F \cup S)$.

P3.13 A patient who had received a flu shot shows up at a doctor's office complaining of flu symptoms. It is known that for his age group the vaccine is 70 % effective. It is also known that people with flu experience these symptoms 80 % of the time, and people free of flu experience these symptoms 20 % of the time. What is the probability that the patient does not have the flu? Use Bayes' rule.

P3.14 Repeat Example 3.9 with three different test variabilities: σ_y = 10, 100, 1000. Show that as σ_y increases, the posterior approaches the posterior with non-informative prior.

P3.15 When three test data 550, 570 and 590 MPa were obtained from tensile test of Aluminum, (a) use Bayesian method to calculate the posterior distribution of failure strength, (b) calculate the mode of posterior distribution, and (c) Calculate the ratios of posterior distribution P(550)/P(570) and P(590)/P(570). It is known that test variability is normally distributed with standard deviation of 10 MPa, and the prior distribution in unknown (non-informative).

P3.16 A failure strength of aluminum is known to be uniformly distributed between 400 and 600 MPa. Two AL tensile specimens failed at 520 and 570 MPa. Tensile

test is known to have uniformly distributed variability of ± 40 MPa. Using Bayesian inference, calculate 2 likelihood functions and the posterior distribution.

P3.17 Generate 20 samples from $N(2.9, 0.2^2)$. Using these samples, plot posterior PDF of unknown population mean conditional on the observations. Use (1) the analytical expression and (2) randomly generated 100,000 samples. Superpose the two PDFs in one graph. Also plot CDF of the two together and compute the maximum difference of the two.

P3.18 When a posterior PDF is given as

$$p_X(x) = \begin{cases} x, & 0 \le x \le 1 \\ 2 - x, & 1 \le x \le 2 \\ 0, & \text{otherwise.} \end{cases}$$

(a) Find CDF and the standard deviation of X analytically. (b) Generate 10,000 samples using the inverse CDF method and plot histogram with the interval 0.1 between grid points. Find a scale factor so that the histogram can approximate the exact PDF. Plot both scaled histogram and exact PDF.

P3.19 Repeat P3.18 when a posterior PDF is given as

$$p_X(x) = 30x(1-x)^4, \quad 0 \le x \le 1$$

P3.20 Repeat Example 3.11 when test variability is normally distributed. Calculate the posterior distribution using (a) recursive Bayesian update and (b) overall Bayesian update.

P3.21 Repeat Example 3.11 (a) when the prior uncertainty is increased from 10 % to 15 % and (b) when test variability is increased from 15 % to 20 %. Compare these results with that of Example 3.11.

P3.22 Using the grid approximation method, calculate 10,000 samples from the following $p(x) \propto 0.3exp(-0.2x^2) + 0.7\exp(-0.2(x-10)^2)$. Plot the PDF using samples and compare it with the exact PDF.

References

Athanasios P (ed) (1984) Probability, random variables, and stochastic processes. McGraw-Hill, New York

Bayes T, Price R (1763) An essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, communicated by Mr. Price, in a letter to John Canton, A. M. F. R. S. Philosophic Trans R Soc Lond 53:370–418. doi:10.1098/rstl.1763.0053

Dempster AP (1967) Upper and lower probabilities induced by a multivalued mapping. Ann Math Stat 38(2):325–339

Gelman A, Carlin JB, Stern HS, Rubin DB (eds) (2004) Bayesian data analysis. Chapman & Hall, New York

Helton JC, Breeding RJ (1993) Calculation of reactor accident safety goals. Reliab Eng Syst Saf 39:129–158

Helton JC, Johnson JD, Oberkampf WL et al (2010) Representation of analysis results involving aleatory and epistemic uncertainty. Int J Gen Syst 39(6):605–646

Jeffreys H (1939) Theory of probability. The Clarendon Press, Oxford

Jeffreys H (ed) (1961) Theory of probability. Oxford classic texts in the physical sciences. Oxford Univ. Press, Oxford

Neyman J (1937) Outline of a theory of statistical estimation based on the classical theory of probability. Philosophic Trans R Soc Lond A 236:333–380

Park CY, Kim NH, Haftka RT (2014) How coupon and element tests reduce conservativeness in element failure prediction. Reliab Eng Syst Saf 123:123–136

Shafer G (1976) A mathematical theory of evidence. Princeton University Press, Princeton

Swiler LP, Paez TL, Mayes RL (2009) Epistemic uncertainty quantification tutorial. Paper presented at the 27th international modal analysis conference, Orlando, Florida, USA, 9–12 Feb 2009

U.S. Department of Defense (2002) Guidelines for property testing of composites, composite materials handbook MIL-HDBK-17. DoD, Washington

Wackerly DD, Mendenhall W III, Scheaffer RL (eds) (2008) Mathematical statistics with applications. Thomson Brooks/Cole, Belmont

Zadeh LA (1999) Fuzzy sets as a basis for a theory of possibility. Fuzzy Sets Syst 100:9–23

Chapter 4
Physics-Based Prognostics

4.1 Introduction to Physics-Based Prognostics

In this chapter, physics-based prognostics approaches are discussed. The fundamental assumption is that there exists a physical model that describes the evolution of damage or degradation. For this reason, the physical model is often referred to as a degradation model, and the physics-based prognostics is often referred to as a model-based prognostics. The objective of this chapter is to introduce basic physics-based prognostics algorithms and to discuss challenges to apply the algorithms in practice.

If an accurate physical model that describes damage degradation as a function of time exists, then that basically completes prognostics because the future behavior of damage can be determined by progressing the degradation model in the future time. In practice, however, the degradation model is not complete and the usage condition in the future is uncertain. Therefore, the key issues in physics-based prognostics are how to improve the accuracy of degradation model and how to incorporate uncertainty in the future. For example, the Paris model in Chap. 2 describes how a crack grows as a function of fatigue loading, which is represented using the range of stress intensity factor. The crack growth rate depends on the level of the stress intensity factor, which may have some variability due to different usage conditions. Also, two model parameters, m and C, determine how fast the damage will grow. These parameters have uncertainty due to manufacturing variability. In addition, the Paris model itself is designed for an infinite plate under mode I fatigue loading. In practice, all plates are finite in size and under constrained boundary conditions with other parts. Therefore, the model itself may have some error (i.e., epistemic uncertainty). Therefore, the decision-making process of prognostics should include these sources of uncertainty and should be based on conservative estimate of damage degradation.

The process of the physics-based prognostics is illustrated in Fig. 4.1. The degradation model is expressed as a function of usage (or, loading) conditions,

N.-H. Kim et al., *Prognostics and Health Management of Engineering Systems*,
DOI 10.1007/978-3-319-44742-1_4

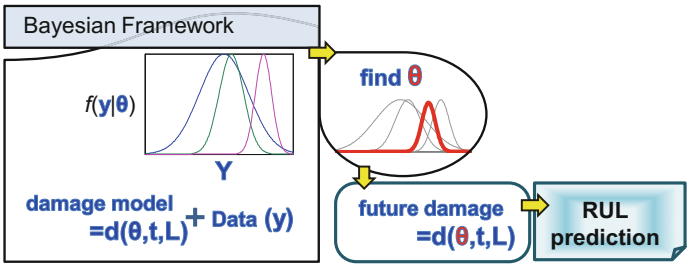


Fig. 4.1 Illustration of physics-based prognostics

L, elapsed cycle or time, t, and model parameters, $\boldsymbol{\theta}$. Although the most significant source of uncertainty is from usage conditions due to unknown future usages, it is often assumed that the usage conditions and time are given for the purpose of physics-based model development. Under this assumption, the major focus of this chapter is on identifying model parameters and predicting the future behavior of degradation.

Even if the model parameters can be obtained from laboratory tests, the actual model parameters that are used in a system might be different from the model parameters that are obtained from laboratory test. For example, different batches of materials show different properties. Therefore, the material tested in the laboratory may have different properties from the one used in a system. In order to acknowledge the variability of material properties for different batches and different companies, material handbook usually shows a wide variability in material properties. For example, the material handbook shows that the variability of Young's modulus of a steel is about 18 %. However, the specific material that is used for a system may have much smaller variability in material properties. Therefore, it is important to identify the actual material properties that are used for a system.

In particular, uncertainty in model parameters can significantly affect the performance of prognostics. For example, among the two Paris model parameters, the exponent, m, of aluminum alloys is known to be in the range of 3.6 and 4.2. This only corresponds to 16 % variability, but the life cycle can differ by 500 %. If a conservative estimated is made, then it is possible that maintenance has to be ordered only after 20 % of actual life has been consumed. Therefore, it is important to reduce the uncertainty in model parameters to predict more accurate remaining useful life, and thus, maintenance time.

As discussed in Chap. 3, the uncertainty in model parameters is mostly from epistemic uncertainty. In structural heal monitoring, on-board sensors and actuators are used to measure damage growth. The fundamental concept of physics-based prognostics is to use these measurements to reduce the uncertainty in degradation model parameters, which was the main focus of Chap. 3. Therefore, measurement data can be used to reduce the uncertainty in degradation model parameters using the Bayesian framework. Therefore, most physics-based prognostics methods have their foundation on Bayesian inference.

For the real-time estimation of the model parameters, the Bayesian statistics explained in Chap. 3 is employed. Once the model parameters are identified after updating process, the future behavior of degradation can be easily predicted by propagating the model to the future time, i.e., substitute the identified parameters to the degradation model with future time and loadings. Finally, the remaining useful life is predicted by propagating the degradation state until it reaches a threshold.

In fact, parameter estimation algorithms become criteria to classify different physics-based approaches. There are several methods for model parameter identification such as nonlinear least squares (NLS) (Gavin 2016), the Bayesian method (BM) (Choi et al. 2010), and several filtering-based techniques, such as Kalman filter (KF) (Kalman 1960), extended/unscented Kalman filter (Ristic et al. 2004; Julier and Uhlmann 2004), and particle filter (PF) (Doucet et al. 2001). NLS is a nonlinear version of the least squares method introduced in Chap. 2, and BM is the overall Bayesian approach given in Chap. 3. The filtering methods are based on the recursive Bayesian method. KF gives the exact posterior distribution in the case of a linear system with Gaussian noise, and other KF family techniques, such as extended/unscented Kalman filter, have been developed to improve the performance for nonlinear systems.

He et al. (2011) employed the extended Kalman filter for lithium-ion battery prognostics. Orchard and Vachtsevanos (2007) estimated the crack closure effect using PF for RUL prediction of a planetary carrier plate based on a vibration-based feature. Daigle and Goebel (2011) used PF to estimate wear coefficients by considering multiple damage mechanisms in centrifugal pumps. Zio and Peloni (2011) used PF for RUL prediction of nonlinear systems with a crack growth example. On the other hand, the Bayesian method is used to implement batch estimation using all the measurement data up to the current time. Choi et al. (2010) applied the Bayesian method to several structural problems for parameter estimation. An et al. (2011) estimated wear coefficients to predict the joint wear volume of a slider-crank mechanism using this method.

Among the aforementioned algorithms, Kalman filter-family and PF are based on the filtering technique that updates parameters recursively by taking one measurement data at a time. The performance of Kalman filter-family largely depends on the initial condition of the parameter and variance of the parameter in addition to the approximation in linearization. On the other hand, PF can be employed without any restrictions for systems and noise type. Therefore, this book focuses on PF as one of the filtering methods.

The organization of this chapter is as follows. Throughout the chapter the battery degradation model (Sect. 4.1.1) is used to explain physics-based prognostics algorithms. First, the nonlinear least squares method is used to identify model parameters and predict RUL in Sect. 4.2. The overall Bayesian update is explained in Sect. 4.3, followed by particle filter method in Sect. 4.4. Section 4.5 extends the battery degradation problem to other applications, such as crack growth prognostics. Finally, Sect. 4.6 presents challenges and issues in physics-based prognostics algorithms.

4.1.1 Demonstration Problem: Battery Degradation

As an example of physical models, a battery degradation model is employed. In a strict sense, there is no well-defined physical model describing battery degradation as a function of variable charging–discharging cycles. Empirical degradation models are widely used when a certain assumption is applied for usage conditions. In this chapter an empirical degradation model is considered as a physical model in physics-based approaches. In the degradation of a battery, it is well known that the capacity of a secondary cell such as a lithium-ion battery degrades over cycles in use, and the failure threshold is defined when the capacity fades by 30 % of the rated value.

A simple form of the empirical degradation model is expressed by an exponential growth model as follows (Goebel et al. 2008):

$$y = a\ \exp(-bt), \tag{4.1}$$

where a and b are model parameters, t is the time or cycles, and y is the internal battery performance, such as electrolyte resistance R_E or transfer resistance R_{CT}. When the usage condition of battery is fully charging–discharging cycles, the time t can be considered as the number of charging–discharging cycles. The internal battery performance is normally measured in terms of capacity. Also, there is a relationship between $R_E + R_{CT}$ and $C/1$ capacity (capacity at nominally rated current of 1A); $R_E + R_{CT}$ is typically inversely proportional to the $C/1$ capacity. In this case, the exponent $-b$ is changed to b without a change of the model form, but a sign of the exponent does not have an effect on the model since the parameter b can have a positive or negative value. Therefore, it is assumed that Eq. 4.1 represents the capacity degradation behavior, and measured data are given as a form of $C/1$ capacity in this chapter. Lastly, a is the initial degradation state (when $t = 0, y = a$), and the initial state can have a significant effect on the degradation rate. However, it is assumed to be known because the 30 % threshold is a relative value and independent of a. Since $C/1$ capacity is considered, it is assumed that $a = 1$ is given. Eventually, Eq. 4.1 can be rewritten as

$$z_k = \exp(-bt_k), \tag{4.2}$$

where z_k is the degradation level, i.e., $C/1$ capacity at a time index k.

The $C/1$ capacity data measured at every 5 charging–discharging cycles are given in Table 4.1. The data are generated by (a) assuming that the true model parameters $b_{\text{true}} = 0.003$; (b) calculating the true $C/1$ capacity according to Eq. 4.2 for the given time steps; and (c) adding Gaussian noise $\varepsilon \sim N(0, 0.02^2)$ to the true $C/1$ capacity data. The true values of parameters are only used to generate observed data. Then, the goal of prognosis is to estimate b using the data. As shown in Fig. 4.2, the end of life (EOL) of the battery is 118 cycles. If 45th cycle, the last measured data point, is considered as the current time, then the remaining useful life is 118 – 45 = 73 cycles.

Table 4.1 Measurement data for battery degradation problem

Time step, k	Initial, 1	2	3	4	5	6	7	8	9	10
Time (cycles)	0	5	10	15	20	25	30	35	40	45
Data, y (C/1 (Ahr))	1.00	0.99	0.99	0.94	0.95	0.94	0.91	0.91	0.87	0.86

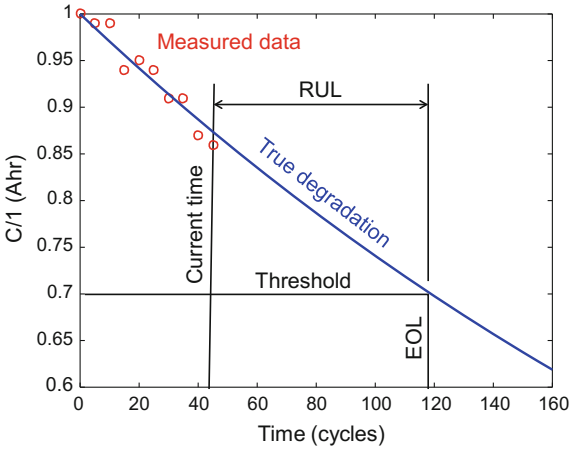


Fig. 4.2 Degradation model of battery with 10 measured data

4.2 Nonlinear Least Squares (NLS)

When a degradation model can be written as a linear combination of unknown model parameters, the regression process to find the unknown parameters is called (linear) least squares as explained in Chap. 2. When the degradation model is a nonlinear function of model parameters as shown in Eq. 4.2, the regression process is called nonlinear least squares (NLS). In other words, it is considered as NLS when the partial derivatives of the degradation model with respect to the model parameters/coefficients are functions of model parameters. The derivative of Eq. 4.2 with respect to b is $\partial z(t;b)/\partial b = -t\exp(-bt)$, where the derivative also depends on parameter b on the right-hand side. In linear model, for example, the derivatives of the linear problem in Eq. 2.9 are obtained as $\partial z/\partial\theta_1 = 1, \partial z/\partial\theta_2 = Lt^2, \partial z/\partial\theta_3 = t^3$, which are either constant or function of only the input variable, t (L is given loading information).

NLS finds model parameters by minimizing the sum of squared errors (SS_E) same as LS, but a weighted sum of square errors ($SS_{E,W}$) is considered as a general approach for NLS:

$$SS_{E,W} = \sum_{k=1}^{n_y} \frac{(y_k - z_k)^2}{w_k^2} = \{\mathbf{y} - \mathbf{z}\}^T \mathbf{W} \{\mathbf{y} - \mathbf{z}\}, \tag{4.3}$$

where w_k^2 is a weight at measurement point y_k, $\mathbf{W}$ is a diagonal matrix with the inverse of weights $1/w_k^2$ as diagonal elements, and z_k is a simulation output from the degradation model. When all weights are same for all measurements, $\mathbf{W}$ can be considered as a constant and can be ignored since it does not have any effect on the minimizing process to find parameters. For such a case, Eq. 4.3 shows the same form as Eq. 2.6. In this section, it is assumed that w_k^2 is a constant that can be determined by the estimated variance of noise in data in Eq. 2.18.

In the case of linear least squares, Eq. 2.6 is a quadratic form ($\{\mathbf{y} - \mathbf{X\theta}\}^{\mathrm{T}}\{\mathbf{y} - \mathbf{X\theta}\}$) and a global optimum can be found, e.g., it is a parabola for one-parameter case, and the global optimum is where the gradient is zero (see Eq. 2.7). The zero-derivative condition becomes a linear system of equations with model parameters as unknown. Therefore, by solving this linear system of equation can yield the optimum parameters. Then, Eq. 2.8 is given as the general form of the global optimum.

On the other hand, for nonlinear least squares, $\mathbf{z}$ in Eq. 4.3 is not a linear combination of unknown parameters anymore, and the zero-derivative conditions, $\mathrm{d}(\mathrm{SS}_{\mathrm{E,W}})/\mathrm{d}\boldsymbol{\theta} = 0$, cannot be expressed as a linear system of equations, such as Eq. 2.7. For this case, the parameters have to be determined using an iterative process based on optimization techniques. There are several algorithms, and the Levenberg–Marquardt (LM) method (Gavin 2016) that combines the gradient descent method and the Gauss–Newton method is a popular one, which is employed in this chapter. The detailed explanation for the optimization process is not introduced in this book, but the MATLAB function `lsqnonlin` is used instead. The process for parameter estimation using LM is explained with the MATLAB function in the following subsection.

In general, the optimization process is deterministic, which means that the process yields a single set of parameters' values that minimize the $\mathrm{SS}_{\mathrm{E,W}}$ error. However, the optimum parameters depend on measured data, which include measurement variability. Therefore, if a different set of data are used, the optimized parameters can be changed. That is, the estimated parameters have uncertainty. In nonlinear least squares, the uncertainty of estimated model parameters can be estimated based on its variance given in the following equation:

$$\boldsymbol{\Sigma}_{\hat{\boldsymbol{\theta}}} = [\mathbf{J}^{\mathrm{T}}\mathbf{W}\mathbf{J}]^{-1}, \tag{4.4}$$

where $\mathbf{J}$ is a Jacobian matrix, $[\partial z(t_k;\boldsymbol{\theta})/\partial\boldsymbol{\theta}]_{n_y \times n_p}$, which plays the same role as the design matrix $\mathbf{X}$ in Eq. 2.5 for linear problems. In order to make the following calculation simple, it is assumed that the magnitudes of the errors at all measured points are the same. Then, w_k^2 becomes constant and identical to the variance of noise in the measured data, $\hat{\sigma}^2$ in Eq. 2.18. That is,

$$w_k^2 = \hat{\sigma}^2 = \frac{\mathrm{SS}_{\mathrm{E}}}{n_y - n_p} = \frac{\{\mathbf{y} - \mathbf{z}\}^{\mathrm{T}}\{\mathbf{y} - \mathbf{z}\}}{n_y - n_p}. \tag{4.5}$$

Finally, Eq. 4.4 can be rewritten as

$$\Sigma_{\hat{\boldsymbol{\theta}}} = \hat{\sigma}^2[\mathbf{J}^{\mathrm{T}}\mathbf{J}]^{-1}. \tag{4.6}$$

Once the model parameters and its variance are obtained, the distribution of parameters also can be obtained based on multivariate *t*-random numbers (See Example 2.6 in Sect. 2.4 for more information). For the Jacobian matrix, it is calculated by numerical approximation when analytical expressions are not available. In this book, we exploit the Jacobian matrix provided from `lsqnonlin`. The degradation and RUL are predicted based on the samples of estimated parameters (explained in Sects. 2.3 and 2.4), and the overall procedure of NLS-based prognostics is explained with the MATLAB code **`[NLS]`** in the following subsection.

4.2.1 MATLAB Implementation of Battery Degradation Prognostics Using Nonlinear Least Squares

In this section, the usage of the MATLAB code **`[NLS]`** is explained. The code is divided into three parts: (1) problem definition for user-specific applications, (2) prognostics using NLS, and (3) post-processing for displaying results. Different algorithm can replace the second part, which will be discussed in later sections. Only the problem definition part needs to be modified for different applications, which are further divided into two subparts: variable definition and model definition. Once these two subparts are completed, users can obtain the results of parameter estimation and degradation and RUL prediction. In the following explanation, 'line' or 'lines' indicates the line number of the MATLAB code given in the last part of this section. Also, `i`, `j`, and `k` in MATLAB codes are indices for samples, parameters and time, respectively. This convention is same for Bayesian method and particle filter algorithms. Detailed explanations for using NLS are given in the following subsections with the example of battery degradation.

4.2.1.1 Problem Definition (Lines 5–14, 48)

For the battery degradation example, `Battery_NLS` is used for `WorkName`, which is the name of the result file. The capacity is measured at every 5 cycles, so `C/1 Capacity` and `cycles`, respectively, are used for `DegraUnit` and `TimeUnit`. The array `time` includes both the measurement and future times and should be long enough for RUL prediction. The C/1 capacity data in Table 4.1 are stored in `y`, which is a $n_y \times 1$ vector. The array size of `time` should be larger than that of `y`. According to the definition of failure threshold in Sect. 4.1.1, `0.7` (70 % of C/1 capacity) is used for `thres`. `ParamName` is the name of the unknown parameters to be estimated: model parameter `b` and the standard deviation of measurement noise `s`.

In Chap. 3, several examples are assumed to have a known variability of observed data. In reality, however, the variability of observed data is often unknown and it also has to be estimated during the parameter estimation process. In the case of **[NLS]**, the variability of observed data is not in the part of parameter estimation process. Rather, it is calculated after parameter estimation. However, it is included in `ParamName` in order to make the problem definition consistent with other algorithms.

Although `ParamName` is an array of character strings, it will be used as an actual variable name in the code using `eval()` function in MATLAB. Therefore, the name of parameters must satisfy the requirement of variable name in MATLAB. When determining the parameters' name, there are three cautions: (1) the user can define anything for the parameter's name but the length of parameters' name should be the same as each other, and when assigning a one letter name, be careful not to use `i`, `j`, `k` because they are already used in the code; (2) the parameter's name representing the model parameters should be used as the model equation in line 48; and (3) the parameter's name of the standard deviation of measurement error should be placed on the last row.

When the true values of parameters are available, `thetaTrue` can be used as a $n_p \times 1$ vector; if not, leave it as an empty array, `[]`. The rest of the required parameters are significance level `signiLevel` and the number of samples `ns`. The significance level is for calculating the confidence interval (C.I.) or prediction interval (P.I.), and the values of 5, 2.5, or 0.5 significance level means 90 %, 95 %, or 99 % intervals. Usually, 1,000–5,000 samples are used for `ns`. In this example, `5` and `5000` are set for `signiLevel` and `ns`, respectively.

```
WorkName='Battery_NLS';
DegraUnit='C/1 Capacity';
TimeUnit='Cycles';
time=[0:5:200]';
y=[1.00 0.99 0.99 0.94 0.95 0.94 0.91 0.91 0.87 0.86]';
thres=0.7;
ParamName=['b'; 's'];
thetaTrue=[0.003; 0.02];
signiLevel=5;
ns=5e3;
```

The damage model equation in Eq. 4.2 is used in line 48 as

```
z=exp(-b.*t);
```

In the damage model, the time t_k is expressed as `t` in the script. Also, the model parameter b and the degradation state z_k, respectively, are expressed as `b` (as defined in line 11) and `z`. Note that the parameter name `b` is identical to the string defined in

`ParaName`. The algebraic expressions should use componentwise operations (i.e., using '.') since degradation state is a vector with `ns` samples.

4.2.1.2 Prognosis Using NLS (Lines 17–32)

As mentioned before, `lsqnonlin` is utilized to estimate the battery model parameter, b. To use the LM method, the setting in line 19 is required.[1] The optimization process is performed in line 20 with objective function (`FUNC`) to be minimized, starting from the initial parameters (`theta0`). In `FUNC` (lines 39–43), the residual vector, $\{\mathbf{y} - \mathbf{z}\}$, for given model parameters is calculated at all measured `ny` data points. In order to calculate the model prediction, function `MODEL` (lines 44–50) is used. Line 48 should be modified for different degradation models. Then `lsqnonlin` MATLAB function builds the objective function SS_E using the residual vector and minimizes it. As the outputs from the process, the deterministic estimation of parameters (`thetaH`), residuals at the estimated parameters (`resid`), and the Jacobian matrix (`J`) are obtained. These results are used to predict the future damage behavior and to estimate the uncertainty of parameters.

The process to obtain the uncertainty of parameters in the form of samples was explained in Sect. 2.4. A brief summary of the process is as follows. (1) Calculate SS_E and the standard deviation in data (lines 21–23). (2) Obtain the standard deviation of parameters (line 25) from their covariance matrix (line 24). (3) Then, the distributions are obtained by adding the deterministic results and t-random samples multiplied with the standard deviation (lines 26, 27). In calculating the standard deviation of data, `dof = ny - np + 1` is used in line 22 because `np` includes the standard deviation, which is not used in the estimation process. Even though the standard deviation in data (`sigmaH`) is deterministically estimated based on the estimated parameters, it is included in the final sampling results (line 28) to maintain consistency in the results with other prognostics methods.

Once the distribution of model parameters are estimated using `ns` samples, the degradation prediction is done by using the results in the model equation (line 30), which reflects the parameter (or model) uncertainty and is related with the confidence intervals. On the other hand, the error in data is added to `degraPredi` (line 31), which is considered as the final degradation prediction results. The prediction of degradation behavior is obtained by repeating this process at other future times up to the last time given in line 8 (line 29), from which the prediction interval can be calculated.

---

[1]The number `0.01` in line 19 is a default value for the initial LM parameter λ. As λ is smaller, the Gauss–Newton update is more dominant in the LM method, while the gradient descent update is for the opposite case. Also, as the parameters are close to the optimal results, the update method is changed from the gradient descent to the Gauss–Newton update. Thereby, the LM parameter λ changes during the update process. See Gavin (2016) for more detailed explanation about the LM method, and see the Help in MATLAB for other options in the optimization process.

The Levenberg–Marquardt algorithm in MATLAB function `lsqnonlin` finds a local optimum, the parameter identification results depend on the initial parameters. It is expected that the users provide the initial values of parameters when **[NLS]** is called. To run the code **[NLS]**, enter the following code in the command window (In this example, `theta0=0.01`).

```
[thetaHat,rul]=NLS(0.01);
```

Although there are two parameters, `b` and `s`, only the initial estimate of `b` is required when **[NLS]** is called, because the standard deviation of measured data is not used in **[NLS]**.

4.2.1.3 Post-processing (Lines 34–37)

Once model parameters are identified and degradation is predicted, their graphical results as well as RUL predictions can be obtained by using the MATLAB code **[POST]** (line 36). Finally, all results are saved in the results file named with `WorkName` and the current cycle. In this example case, the saved file name is "Battery_NLS at 45.mat."

In MATLAB code **[POST]**, the distribution of identified parameters and degradation prediction results are plotted (lines 6–20). The PDF of the parameters in line 9 can be obtained from the histogram results in line 8 by dividing the frequency (`frq`) by the distribution area (`ns/(val(2)-val(1))`). This scaling makes the histogram consistent with the probability density function. Even if `NLS` does not calculate the distribution of standard deviation of measure data, it is plotted as a constant value: the square root of the variance of noise in Eq. 4.5.

The RUL should be calculated based on the degradation prediction results `degraPredi` (lines 21–33). The parameter `i0` (line 21) is used to count the number of samples not reaching to the threshold, which is displayed (line 26). This means that the end of prediction time is not long enough or the RUL is predicted as an infinite life. Therefore, when there are many samples not reaching to the threshold (`i0` is large compared to the total samples `ns`, such as more than 5 % of `ns`), it is required either to increase the ending time (line 8 in **[NLS]**) or to update model parameters with more data. Since degradation behaviors can monotonically increase (e.g., crack size) or decrease (e.g., battery capacity), `coeff` (line 22) reflects this difference: `coeff = -1` for decreasing degradation or `coeff = +1` for increasing degradation.

RUL prediction is performed for each sample (line 23) to find time/cycle where the degradation prediction meets the threshold. First, the time/cycle location where the degradation prediction from one sample `degraPredi(:,i)` meets the threshold first time is saved in `loca` (line 24). When `loca` is empty, it means the degradation of the particular sample does not reach the threshold (lines 25, 26) until

the end of time, while `loca==1` means the current time/cycle is the end of life (EOL), thereby RUL is zero (line 27). A general case for RUL calculation is in lines 28–30, where the precise RUL is interpolated using two degradation levels before and after the threshold. This process is repeated for all `ns` samples. Once the RUL is calculated for all samples, three percentiles (line 5) of RUL are calculated (line 33), and the PDF of RUL (lines 34–39) and its percentiles (lines 40–42) are displayed.

If the true values of parameters are given in line 12 in **`[NLS]`**, the true parameters (lines 44–47) and true degradation behavior (lines 49–52) are plotted with their prediction results. Also, the true RUL is calculated (lines 54–56) and plotted with the prediction result (line 57). The **`[POST]`** will be used throughout this book for other algorithms.

The results for battery degradation prognostics are shown in Figs. 4.3 and 4.4, which can be obtained again by typing `POST(thetaHat,yHat,degraTrue)` in the command window after loading the saved results with `Name` (line 37 in **`[NLS]`**). The standard deviation `s` of data is deterministically obtained as 0.0125, which is 37.26 % error with respect to the true value (`(thetaHat(2,1)-thetaTrue(2))/thetaTrue(2)*100`). In the figures, the star markers represent the true values.

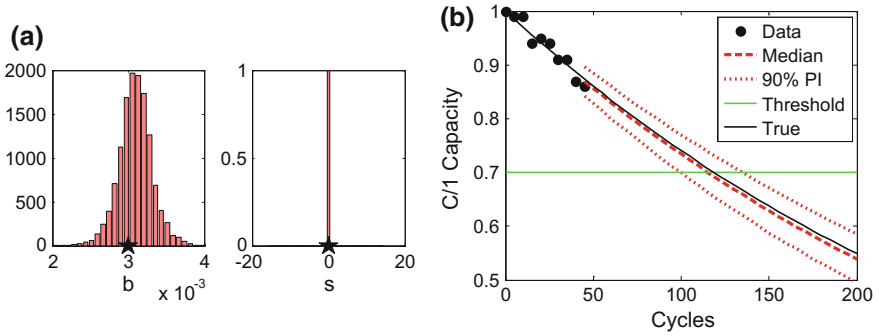
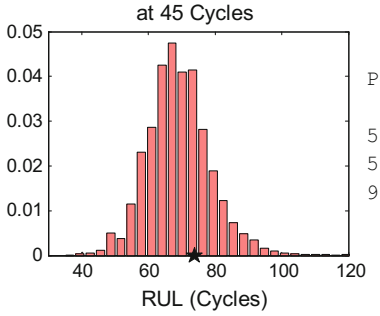


Fig. 4.3 Results of the battery problem: NLS. **a** Distribution of estimated parameter, **b** degradation prediction




```
Percentiles of RUL at 45 Cycles

5th: 54.5086,
50th (median): 68.5961,
95th: 85.8337
```

Fig. 4.4 RUL prediction results: NLS

[NLS]: MATLAB Code for Nonlinear Least Squares

```
function [thetaHat,rul]=NLS(theta0)
clear global; global DegraUnit TimeUnit ...
time y thres ParamName thetaTrue signiLevel ns ny nt np
%=== PROBLEM DEFINITION 1 (Required Variables) ==============
WorkName=' ';            % work results are saved by WorkName
DegraUnit=' ';                              % degradation unit
TimeUnit=' ';              % time unit (Cycles, Weeks, etc.)
time=[ ]';          % time at both measurement and prediction
y=[ ]';                                  %[nyx1]: measured data
thres= ;                             % threshold (critical value)
ParamName=[ ];      %[npx1]: parameters' name to be estimated
thetaTrue=[ ];            %[npx1]: true values of parameters
signiLevel= ;          % significance level for C.I. and P.I.
ns= ;                           % number of particles/samples
%=============================================================
% % % PROGNOSIS using NLS
ny=length(y); nt=ny;
np=size(ParamName,1);             %% Deterministic Estimation
Optio=optimset('algorithm',{'levenberg-marquardt',0.01});
[thetaH,~,resid,~,~,~,J]=lsqnonlin(@FUNC,theta0,[],[],Optio);
sse=resid'*resid;                     %% Distribution of Theta
dof=ny-np+1;
sigmaH=sqrt(sse/dof);                  % Estimated std of data
W=eye(ny)*1/sigmaH^2; thetaCov=inv(J'*W*J);
sigTheta=sqrt(diag(thetaCov));        % Estimated std of Theta
mvt=mvtrnd(thetaCov,dof,ns)';                % Generate t-dist
thetaHat=repmat(thetaH,1,ns)+mvt.*repmat(sigTheta,1,ns);
thetaHat(np,:)=sigmaH*ones(1,ns);    %% Final Sampling Results
for k=1:length(time(ny:end));        %% Degradation Prediction
 zHat(k,:)=MODEL(thetaHat,time(ny-1+k));
 degraPredi(k,:)=zHat(k,:)+trnd(dof,1,ns)*sigmaH;
end;
% % % POST-PROCESSING
degraTrue=[];                                %% True Degradation
if ~isempty(thetaTrue); degraTrue=MODEL(thetaTrue,time); end
rul=POST(thetaHat,degraPredi,degraTrue); %% RUL & Result Disp
Name=[WorkName ' at ' num2str(time(ny)) '.mat']; save(Name);
end
function objec=FUNC(theta)
 global time y ny
 z=MODEL(theta,time(1:ny));
 objec=(y-z);
end
function z=MODEL(theta,t)
 global ParamName np
 for j=1:np-1; eval([ParamName(j,:) '=theta(j,:);']); end
 %===== PROBLEM DEFINITION 2 (model equation) ===============


 %=============================================================
end
```

[POST]: MATLAB Code for RUL Calculation and Results Plot

```
function rul=POST(thetaHat,degraPredi,degraTrue)
global DegraUnit ...
TimeUnit time y thres ParamName thetaTrue signiLevel ns ny nt
np=size(thetaHat,1);
perceValue=[50 signiLevel 100-signiLevel];
figure(1);                          %% Distribution of Parameters
for j=1:np; subplot(1,np,j);
 [frq,val]=hist(thetaHat(j,:),30);
 bar(val,frq/ns/(val(2)-val(1)));
 xlabel(ParamName(j,:));
end;
figure(2);                                    %% Degradation Plot
degraPI=prctile(degraPredi',perceValue)';
f1(1,:)=plot(time(1:ny),y(:,1),'.k'); hold on;
f1(2,:)=plot(time(ny:end),degraPI(:,1),'--r');
f1(3:4,:)=plot(time(ny:end),degraPI(:,2:3),':r');
f2=plot([0 time(end)],[thres thres],'g');
legend([f1(1:3,:); f2],'Data','Median',...
              [num2str(100-2*signiLevel) '% PI'],'Threshold')
xlabel(TimeUnit); ylabel(DegraUnit);
i0=0;                                          %% RUL Prediction
if y(nt(1))-y(1)<0; coeff=-1; else coeff=1;end;
for i=1:ns;
 loca=find(degraPredi(:,i)*coeff>=thres*coeff,1);
 if isempty(loca); i0=i0+1;
  disp([num2str(i) 'th not reaching thres']);
 elseif loca==1; rul(i-i0)=0;
 else rul(i-i0)=...
  interp1([degraPredi(loca,i) degraPredi(loca-1,i)], ...
            [time(ny-1+loca) time(ny-2+loca)],thres)-time(ny);
 end
end;
rulPrct=prctile(rul,perceValue);
figure(3);                                %% RUL Results Display
[frq,val]=hist(rul,30);
bar(val,frq/ns/(val(2)-val(1))); hold on;
xlabel(['RUL' ' (' TimeUnit ')']);
titleName=['at ' num2str(time(ny)) ' ' TimeUnit];
title(titleName)
fprintf(['\n Percentiles of RUL at %g ' TimeUnit], time(ny))
fprintf('\n  %gth: %g,  50th (median): %g,  %gth: %g \n', ...
perceValue(2),rulPrct(2),rulPrct(1),perceValue(3),rulPrct(3))
if ~isempty(degraTrue);                   %% True Results Plot
 figure(1); % parameters
 for j=1:np; subplot(1,np,j); hold on;
  plot(thetaTrue(j),0,'kp','markersize',18);
 end;
 figure(2); % degradation
 sl=0; if ~isempty(nt); sl=ny-nt(end); end
 f3=plot(time(sl+1:sl+length(degraTrue)),degraTrue,'k');
```

```
 legend([f1(1:3,:); f2; f3],'Data','Median', ...
       [num2str(100-2*signiLevel) '% PI'],'Threshold','True')
 figure(3); % RUL
 loca=find(degraTrue*coeff>=thres*coeff,1);
 rulTrue=interp1([degraTrue(loca) degraTrue(loca-1)], ...
             [time(sl+loca) time(sl+loca-1)],thres)-time(ny);
 plot(rulTrue,0,'kp','markersize',18);
end
end
```

4.3 Bayesian Method (BM)

In this section, the overall Bayesian method (BM) is employed to estimate the PDF of model parameters using measured data up to the current time. As discussed in Chap. 3, the posterior distribution of parameters at the current step is obtained by a single equation, in which all the likelihood functions of measured data up to the current step are multiplied.

For conjugate distributions that have the same distribution type for the prior and posterior, it is possible to generate samples from standard probability distributions. For a single parameter estimation, it is also possible to generate samples using the inverse CDF of grid approximation methods as in Sect. 3.7. However, these methods are not general enough for practical engineering applications where the posterior distribution may not follow a standard probability distribution, or the posterior distribution is complicated due to correlation between multiple parameters.

In such a case, it would be necessary to use a sampling method that can generate samples from arbitrary posterior distribution. Several sampling methods are available, such as the grid approximation (Gelman et al. 2004), rejection sampling (Casella et al. 2004), importance sampling (Glynn and Iglehart 1989), and the Markov chain Monte Carlo (MCMC) method (Andrieu et al. 2003). In this section, the Markov chain Monte Carlo (MCMC) method is employed, which is known as an effective sampling method.

Once the samples of parameters are obtained from the posterior distribution, they can be substituted into the degradation model to predict the behavior of degradation and to find a future time when the degradation level reaches a threshold, which defines the remaining useful life.

4.3.1 Markov Chain Monte Carlo (MCMC) Sampling Method

The MCMC sampling method is based on a Markov chain model in random walk as shown in Fig. 4.5. It starts from an arbitrary initial sample (old sample) and a new

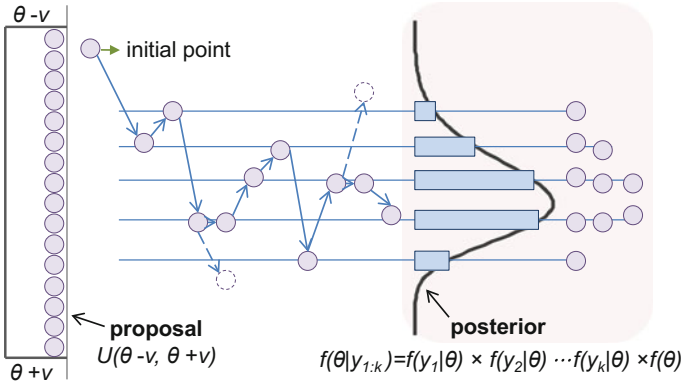


Fig. 4.5 Illustration of Markov chain Monte Carlo sampling

sample is drawn from an arbitrary proposal distribution centered at the old sample. The two consecutive samples are compared with each other based on an acceptance criterion, from which either the new sample is selected or the old sample is re-selected. In Fig. 4.5, two dashed circles mean that these new samples are not selected according to the criterion. In such a case, the old sample is selected again. This process is repeated as many times as necessary until a sufficient number of samples are obtained.

In order to generate samples, the following Metropolis–Hastings (M-H) algorithm is used:

1. Generate initial sample $\boldsymbol{\theta}^0$
2. For $i = 1$ to n_s
   - Generate sample from proposal distribution $\boldsymbol{\theta}^* \sim g\left(\boldsymbol{\theta}^* | \boldsymbol{\theta}^{i-1}\right)$
   - General acceptace sample $u \sim U(0,1)$
   - if $u < Q\left(\boldsymbol{\theta}^{i-1}, \boldsymbol{\theta}^*\right) = \min\left\{1, \frac{f(\boldsymbol{\theta}^*|\mathbf{y}) g\left(\boldsymbol{\theta}^{i-1}|\boldsymbol{\theta}^*\right)}{f\left(\boldsymbol{\theta}^{i-1}|\mathbf{y}\right) g\left(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{i-1}\right)}\right\}$ (4.7)

     $\boldsymbol{\theta}^i = \boldsymbol{\theta}^*$

     else

     $\boldsymbol{\theta}^i = \boldsymbol{\theta}^{i-1}$.

In the algorithm, $\boldsymbol{\theta}^0$ is a vector of initial values of the unknown model parameters to be estimated, n_s is the total number of samples, $f(\boldsymbol{\theta}|\mathbf{y})$ is the target distribution which is the posterior distribution after Bayesian update, and $g\left(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{i-1}\right)$ is an arbitrarily chosen proposal distribution which is used when a new sample $\boldsymbol{\theta}^*$ is drawn conditional on the previous point $\boldsymbol{\theta}^{i-1}$. When a symmetric distribution is selected for the proposal distribution, $g\left(\boldsymbol{\theta}^{i-1}|\boldsymbol{\theta}^*\right)$ is the same as $g\left(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{i-1}\right)$. For example, the PDF at b of $N(a, s^2)$ is the same as the PDF at a of $N(b, s^2)$. The same is true when a uniform distribution is used for the proposal distribution.

The proposal distribution, g, is usually chosen to be a uniform distribution $U(\boldsymbol{\theta}^* - \mathbf{w}, \boldsymbol{\theta}^* + \mathbf{w})$, where $\mathbf{w}$ is a vector of weights for setting the sampling interval and is selected arbitrarily based on the experience. For the case of symmetric proposal distribution, the acceptance criterion, $Q\left(\boldsymbol{\theta}^{i-1}, \boldsymbol{\theta}^*\right)$ can be reduced as

$$Q\left(\boldsymbol{\theta}^{i-1}, \boldsymbol{\theta}^*\right) = \min\left\{1, \frac{f(\boldsymbol{\theta}^*|\mathbf{y})}{f\left(\boldsymbol{\theta}^{i-1}|\mathbf{y}\right)}\right\}. \tag{4.8}$$

The selection process is based on comparing the above acceptance criterion with a randomly generated probability. If the result in Eq. 4.8 is greater than a randomly generated sample u from $U(0,1)$, $\boldsymbol{\theta}^*$ is accepted as a new sample. It is possible to think of two cases. First, when the PDF value at the new sample is greater than that of the old sample, the new sample is always accepted since Q is 1. This means that the new sample is always accepted if the new sample increases probability. Second, when the PDF value at the new sample is less than that of the old sample, the acceptance depends on the random sample u and the ratio of PDF values. If the new sample $\boldsymbol{\theta}^*$ is not accepted as an ith sample, the $i-1$th sample becomes the ith sample; that is, the particular sample is doubly counted. After sufficient iterations, this results in samples that approximate the target distribution.

MCMC simulation results are affected by the initial value of parameters and the proposal distribution. If the initial values are set with largely different values from the true ones, many iterations (samples) will be required to converge to the target distribution. Also, small weight means narrow proposal distribution, which can yield destabilization in the sampling results by not fully covering the target distribution, while large weight (wide proposal distribution) can yield many duplications in sampling result by not accepting new samples. The effects of the initial samples and weight are introduced in Example 4.1.

Example 4.1 MCMC method: one parameter
Draw 5000 samples from $A \sim N(5, 2.9^2)$ by using the MCMC method, and compare the scaled histogram with the exact PDF.

Solution:
This problem can be solved by following the steps in Eq. 4.7. First, in order to start with an initial sample, the mean of A can be utilized, i.e., setting `para0=5`. Next, one sample is drawn from $U(0,1)$ using `u=rand`, and then a new sample of the parameter is extracted from a proposal distribution. When the uniform distribution is employed for the proposal distribution, a new sample with given `para0` can be obtained as `para1=unifrnd (para0-weigh, para0+weigh)`, in which `weigh` determines the range of the proposal distribution, and thus it should be properly chosen. The weight is set as `5` based on the relation between the standard deviation and the interval of the uniform distribution ($\sigma = \text{interval}/\sqrt{12}$). Since the proposal

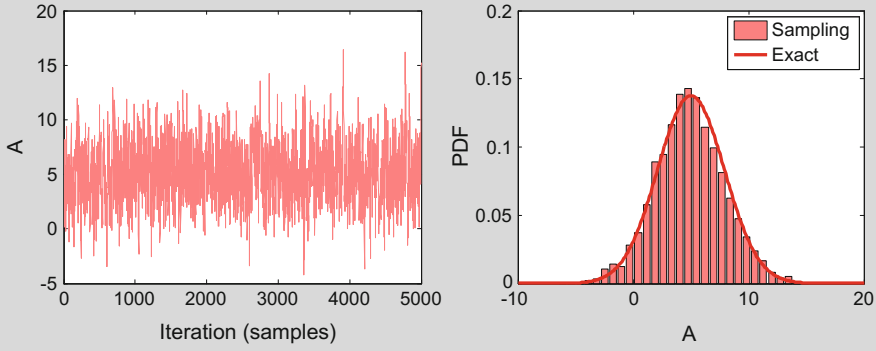
distribution is symmetric, Eq. 4.8 is used to assess the new sample. The target distribution, f is usually updated with given data $\mathbf{y}$, but it is already given as $N(5, 2.9^2)$. Therefore, the PDF values of target function can be calculated as

at new sample $f(\boldsymbol{\theta}^*|\mathbf{y})$, `pdf1=normpdf(para1,5,2.9);`
at old (initial) sample $f\left(\boldsymbol{\theta}^{i-1}|\mathbf{y}\right)$, `pdf0=normpdf(para0,5,2.9);`

and then, $Q\left(\boldsymbol{\theta}^{i-1}, \boldsymbol{\theta}^*\right)$ can be calculated. If `u` is less than $Q\left(\boldsymbol{\theta}^{i-1}, \boldsymbol{\theta}^*\right)$, the new sample is accepted as a current one, if not the old (initial) sample is stored again. The accepted sample becomes the old one for the next step, and this process is repeated 5000 times. The overall process for the MCMC simulation is given as

```
para0=5; weigh=5;
for i=1:5000
    para1=unifrnd(para0-weigh, para0+weigh);
    pdf1=normpdf(para1,5,2.9);
    pdf0=normpdf(para0,5,2.9);
    Q=min(1, pdf1/pdf0);
    u=rand;
    if u<Q; sampl(:,i)=para1; else sampl(:,i)=para0; end
    para0=sampl(:,i);
end
```

The sampling results are stored in `sampl`, and Fig. E4.1 is obtained using the following code:

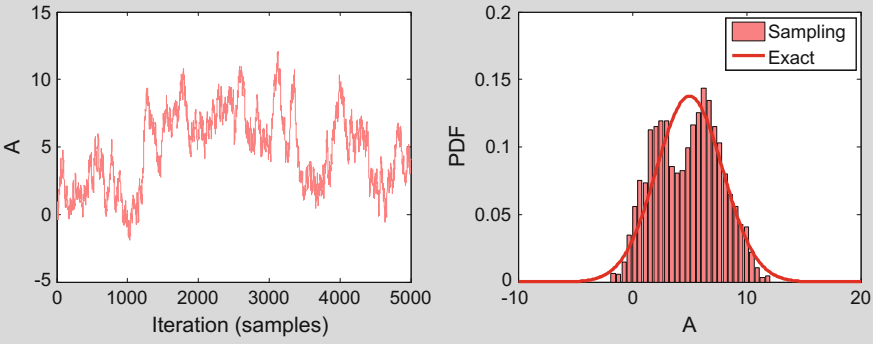


(a) Sampling trace (b) PDF

Fig. E4.1 MCMC sampling results for $N(5, 2.9^2)$. **a** Sampling trace, **b** PDF

```
plot(sampl); xlabel('Iteration (samples)'); ylabel('A')
figure; [frq,val]=hist(sampl,30);
bar(val,frq/5000/(val(2)-val(1))); hold on;
a=-10:0.1:20; plot(a,normpdf(a,5,2.9),'k')
xlabel('A'); legend('Sampling','Exact')
```

The results in Fig. E4.1 show that the initial sample and the weight are properly chosen. For example, Fig. E4.1a shows fluctuating samples within a constant bound around the mean value, which means the sampling results are stabilized and Fig. E4.1b proves this by agreeing the histogram with the exact PDF. However, it is not easy to know the proper values for the initial sample and the weight in practical problems, and improper settings may yield destabilized results. For example, when `para0=0.5, weigh=0.5` are used for this problem, the results in Fig. E4.2 are obtained. As can be seen in Fig. E4.2a, the sampling trace is not fluctuating with respect to the mean, and as a result, the histogram in Fig. E4.2b does not agree with the exact PDF.



(a) Sampling trace (b) PDF

Fig. E4.2 MCMC sampling results for $N(5, 2.9^2)$ with improper settings (destabilization). **a** Sampling trace, **b** PDF

One of important advantages of MCMC sampling method is that it is convenient to generate samples from multidimensional, correlated joint PDF. Example 4.2 shows how to generate samples from two-parameter joint PDF with five measurements.

Example 4.2 MCMC method: two parameters

A posterior joint PDF of two model parameters, $\boldsymbol{\theta} = \{\theta_1, \theta_2\}^{\mathrm{T}}$, is given as

$$f\left(\boldsymbol{\theta}|\mathbf{y}_{1:n_y}\right) = \frac{1}{\left(\sigma\sqrt{2\pi}\right)^5}\exp\left(-\sum_{k=1}^{5}\frac{(z_k - y_k)^2}{2\sigma^2}\right), \quad \sigma = 2.9,$$

where the degradation model and five measured data are given as

$$z_k = 5 + \theta_1 t_k^2 + \theta_2 t_k^3, \quad \mathbf{t} = [0 \quad 1 \quad 2 \quad 3 \quad 4]$$
$$\mathbf{y} = [6.99 \quad 2.28 \quad 1.91 \quad 11.94 \quad 14.60].$$

Draw 5000 samples of parameters by using the MCMC method, and compare the result with exact PDF.

Solution:

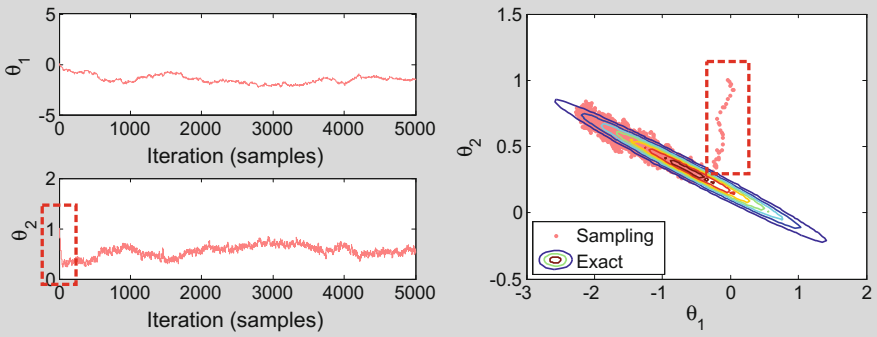
The process is the same as the one-parameter case in Example 4.1, but the initial samples and weights are vectors, e.g., `para0=[0; 1], weigh=[0.05; 0.05]`, and accordingly, the sampling results `sampl` become a 2×5000 matrix. In this problem, the target distribution is a posterior distribution with normal likelihood function and five data, which is the same problem as Example 3.13. The joint PDF at old and new samples can be calculated as

```
para0=[0; 1]; weigh=[0.05; 0.05];
t = [0    1    2    3    4];
y=[6.99 2.28 1.91 11.94 14.60];
s=2.9;

f0=5+para0(1)*t.^2+para0(2)*t.^3;
pdf0=(s*sqrt(2*pi))^-5*exp(-(f0-y)*(f0-y)'/(2*s^2));

para1=unifrnd(para0-weigh, para0+weigh);
f1=5+para1(1)*t.^2+para1(2)*t.^3;
pdf1=(s*sqrt(2*pi))^-5*exp(-(f1-y)*(f1-y)'/(2*s^2));
```

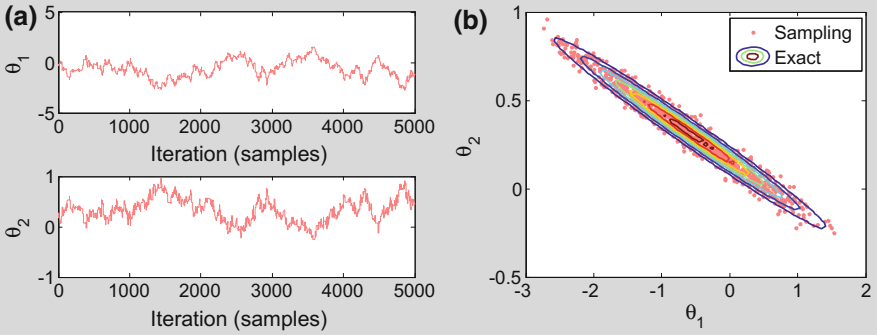
The sampling results are shown in Fig. E4.3, (see Example 3.13 to plot the results). The samples in the red rectangles are caused by improper initial value. To prevent this, an initial portion of the samples has to be discarded, which is called the burn-in. Also, there are very small fluctuations in the sampling trace, which means the magnitude of weight is too small.



(a) Sampling trace (b) Sampling results: correlation

Fig. E4.3 MCMC sampling results for two parameters

Instead of discarding the samples at the initial stage, the initial values are adjusted and the weight are increased as `para0=[0; 0], weigh=[0.3; 0.3]`, whose results are shown in Fig. E4.4. Since the two parameters are correlated each other, it is not easy to obtain the PDF of each parameter accurately and the trace results of each parameter are not stabilized. However, the correlation can be identified well, which is more important to predict degradation and RUL accurately. The issue of identifying correlated parameters will be discussed in Chaps. 6 and 7 in more detail.



(a) Sampling trace (b) Sampling results: correlation

Fig. E4.4 MCMC sampling results for two parameters.

4.3.2 MATLAB Implementation of Bayesian Method for Battery Prognostics

According to the Bayesian parameter estimation method explained in Chap. 3, the likelihood and prior distribution should be defined in order to obtain the posterior distribution. In this section, normal and uniform distributions are employed for the likelihood and prior distribution, respectively. The likelihood function can be expressed in the following form for the battery model given in Eq. 4.2:

$$f(y_k|\boldsymbol{\theta}) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(z_k - y_k)^2}{2\sigma^2}\right), \quad z_k = \exp(-bt_k), \tag{4.9}$$

where $\boldsymbol{\theta} = \{b, \sigma\}^{\mathrm{T}}$, the model parameter and the standard deviation of measurement noise, are the unknown parameters to be estimated, and their prior distributions are assumed as:

$$f(\boldsymbol{\theta}) = f(b) \times f(\sigma), \quad f(b) \sim U(0, 0.02), \quad f(\sigma) \sim U(1\text{e-}5, 0.1), \tag{4.10}$$

which means that both parameters are initially assumed to be independent.

In the MATLAB code **[BM]** at the end of this section, the default options for the prior distribution (lines 56–57) and the likelihood function (lines 58–59), respectively, are uniform and normal distribution as introduced here. However, the other probability distributions can also be employed, and this will be explained in Sect. 4.5.

4.3.2.1 Problem Definition (Lines 5–16, 52)

Most parameters in the problem definition part are identical to that of **[NLS]** code in Sect. 4.2.1, but two more parameters are required to be set in lines 12 and 16 in **[BM]** as

```
initDisPar=[0 0.02; 1e-5 0.1];
burnIn=0.2;
```

`initDisPar` that is a $n_p \times 2$ matrix of probability distribution parameters of initial/prior distributions, where n_p is the number of unknown parameters. For example, the probability parameters of uniform distribution and normal distribution are lower and upper bounds and mean and standard deviation, respectively. Even though there are distributions having three and four probability parameters, such as exponentiated Weibull and four-parameter beta, the number of probability

distribution parameters are fixed as two since most types of distribution have two parameters. The battery problem has two unknown parameters to be estimated. As given in Eq. 4.10, `initDisPar=[0 0.02; 1e-5 0.1]` is set in line 12. The first row is the lower- and upper bounds of uniformly distributed parameter `b`, while the second row is the bounds for standard deviation of data `s`. It is noted that the lower bound of `s` has a small number because zero standard deviation is not allowed. As a MCMC-specific parameter, `burnIn` (line 16) is the ratio for burn-in that is to discard the initial samples before stabilizing. When it is set 20 %, `burnIn=0.2`. The degradation model is the same as the case of NLS with Eq. 4.2, which should be given in line 52.

4.3.2.2 Prognosis Using BM with MCMC (Lines 19–41)

The structure of **`[BM]`** is based on the MCMC simulation algorithm in Eq. 4.7. As explained in Sect. 4.3.1, the sampling results depend on the initial parameter (`para0`) and the weight in a proposal distribution (`weigh`); thereby these two parameters are used as the input variables in the function (line 1). For the battery problem, the initial parameters `[0.01 0.05]'` and the weight `[0.0005 0.01]'` are used for `para0` and `weigh`, respectively. The following command can be used to run **`[BM]`** in MATLAB command prompt:

```
[thetaHat,rul]=BM([0.01  0.05]',[0.0005  0.01]');
```

Different from **`[NLS]`**, the function `MODEL` (lines 48–62) returns the degradation prediction z as well as posterior PDF value `poste` for each given sample of model parameter `param` and time/cycle `t`. The degradation prediction is the same as **`[NLS]`**, where the physical model is used to predict the degradation at given time/cycle. If an array of times/cycles is given, the function `MODEL` returns an array of predictions. The function `MODEL` can also calculate the posterior PDF value using `ny` measured data. Using the prior given in Eq. 4.10 (line 56) and the likelihood in Eq. 4.9 (line 58), the posterior is calculated by multiplying these two (line 60). It is noted that the likelihood uses all `ny` data simultaneously, which is why it is called the overall Bayesian method. In the function MODEL, it is assumed that the prior is the product of independent uniform distributions, and the likelihood is from normally distributed noise. Other types of distributions will be handled in Sect. 4.5.

The function `MODEL` can be used for two purposes. The first usage is to call `MODEL` with the first `ny`×`1` times/cycles, which is the parameter estimation stage. In this stage, `MODEL` will return the posterior PDF value (lines 22 and 27) for given parameter values. This stage is used to generate the posterior samples in MCMC.

The other usage is to call `MODEL` with future times, where `MODEL` will return degradation predictions in the future times (line 39).

The generated posterior PDF samples are stored in `sampl` (line 32), and it starts with the initial sample of model parameters `para0` (line 21). After setting the initial values of parameters and the joint PDF value at the initial values (line 22), the same calculations are repeated for new samples. A new sample of parameters is drawn from a proposal distribution (it is the uniform distribution in the code) with `weigh` (line 25), and a joint posterior PDF value is calculated for the new sample (line 27). The criterion in Eq. 4.8 is used to determine whether the new sample is accepted or not, by comparing with a random sample from $U(0,1)$ (lines 29–31). If the Q in Eq. 4.8 is greater than a random sample, `rand` (line 29), the new sample is accepted (line 30) and stored in `sampl` (line 32). Otherwise, the current sample is duplicated in `sampl` without replacing with the new sample (line 32). Since the joint PDF at the current samples, `jPdf0` already exists with `para0`, the sampling process is repeated by drawing a new sample and the joint PDF. The number of total samples is `ns/(1-burnIn)` in line 23 and the number of burn-in samples is `nBurn=ns/(1-burnIn)-ns`, which makes the number of final samples becomes `ns`.

The sampling traces of the two parameters are plotted in lines 34–36, as shown in Fig. 4.6, in which the red vertical lines indicate the burn-in location. By using sampling results after the lines, the initial effects can be avoided, which is stored in `thetaHat` (line 37). Once the final samples of the parameters are obtained, degradation levels at future cycles can be predicted as the same way explained in Sect. 4.2.1 (lines 38–41).

4.3.2.3 Post-processing (Lines 43–46)

The post-processing is basically the same as that of **`[NLS]`**. Therefore, the same MATLAB code **`[POST]`** can be used. The results from BM are shown in Figs. 4.7 and 4.8, which are similar with them from **`[NLS]`** except for the standard deviation

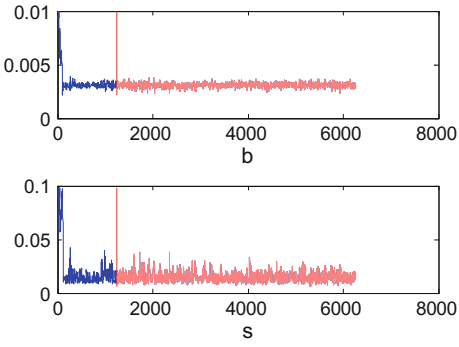


Fig. 4.6 Sampling trace of estimated parameters in the battery problem

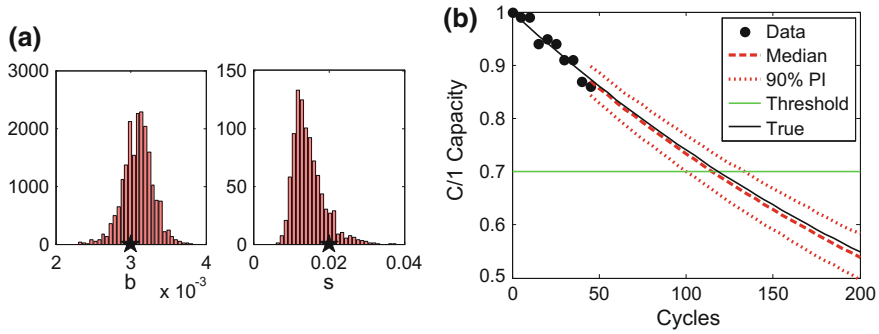
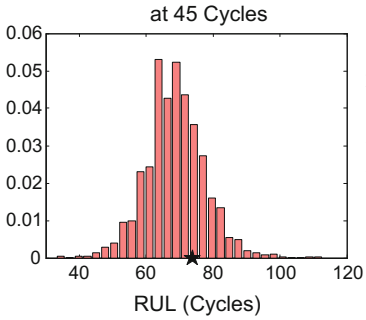


Fig. 4.7 Results of the battery problem: BM. **a** Distribution of estimated parameter, **b** degradation prediction




```
Percentiles of RUL at 45 Cycles

5th: 54.3859,
50th (median): 68.3619,
95th: 82.8892
```

Fig. 4.8 RUL prediction results: BM

σ of data noise that is obtained as a distribution as shown in Fig. 4.7a. The median of σ is 0.0137 that can be calculated with `median(thetaHat(2,:))`, which is closer to the true values than NLS result. Since the distribution has a positive skewness (lower values of σ), the prediction bound from BM in Fig. 4.7b is not wider than that from NLS in Fig. 4.3b even if σ is identified as a distribution. The 90 % prediction interval of BM is about 10 % smaller than that of NLS.

[BM]: MATLAB Code for Bayesian Method

```
function [thetaHat,rul]=BM(para0,weigh)
clear global; global DegraUnit initDisPar TimeUnit ...
time y thres ParamName thetaTrue signiLevel ns ny nt np
%=== PROBLEM DEFINITION 1 (Required Variables) ==============
WorkName=' ';            % work results are saved by WorkName
DegraUnit=' ';                              % degradation unit
TimeUnit=' ';              % time unit (Cycles, Weeks, etc.)
time=[ ]';          % time at both measurement and prediction
y=[ ]';                                  %[nyx1]: measured data
thres= ;                          % threshold (critical value)
ParamName=[ ];      %[npx1]: parameters' name to be estimated
initDisPar=[ ]; %[npx2]: prob. parameters of init./prior dist
thetaTrue=[ ];            %[npx1]: true values of parameters
signiLevel= ;          % significance level for C.I. and P.I.
ns= ;                           % number of particles/samples
burnIn= ;                                  % ratio for burn-in
%============================================================
% % % PROGNOSIS using BM with MCMC
ny=length(y); nt=ny;
np=size(ParamName,1);
sampl(:,1)=para0;                             %% Initial Samples
[~,jPdf0]=MODEL(para0,time(1:ny));     %% Initial (joint) PDF
for i=2:ns/(1-burnIn);                          %% MCMC Process
 % proposal distribution (uniform)
 para1(:,1)=unifrnd(para0-weigh,para0+weigh);
 % (joint) PDF at new samples
 [~,jPdf1]=MODEL(para1,time(1:ny));
 % acception/rejection criterion
 if rand<min(1,jPdf1/jPdf0)
  para0=para1; jPdf0=jPdf1;
 end;
 sampl(:,i)=para0;
end;
nBurn=ns/(1-burnIn)-ns;                        %% Sampling Trace
figure(4);for j=1:np;subplot(np,1,j);plot(sampl(j,:));hold on
plot([nBurn nBurn],[min(sampl(j,:)) max(sampl(j,:))],'r');end
thetaHat=sampl(:,nBurn+1:end);       %% Final Sampling Results
for k=1:length(time(ny:end));        %% Degradation Prediction
 [zHat(k,:),~]=MODEL(thetaHat,time(ny-1+k));
 degraPredi(k,:)=normrnd(zHat(k,:),thetaHat(end,:));
end;
% % % POST-PROCESSING
degraTrue=[];                                %% True Degradation
if ~isempty(thetaTrue); degraTrue=MODEL(thetaTrue,time); end
rul=POST(thetaHat,degraPredi,degraTrue); %% RUL & Result Disp
Name=[WorkName ' at ' num2str(time(ny)) '.mat']; save(Name);
end
```

```
function [z,poste]=MODEL(param,t)
 global y ParamName initDisPar ny np
 for j=1:np; eval([ParamName(j,:) '=param(j,:);']); end
 %===== PROBLEM DEFINITION 2 (model equation) ===============

 %============================================================
 if length(y)~=length(t); poste=[];
 else
  prior= ...                                          %% Prior
        prod(unifpdf(param,initDisPar(:,1),initDisPar(:,2)));
  likel=(1./(sqrt(2.*pi).*s)).^ny.* ...          %% Likelihood
          exp(-0.5./s.^2.*(y-z)'*(y-z));
  poste=likel.*prior;                             %% Posterior
 end
end
```

4.4 Particle Filter (PF)

Particle filter (PF) is the most popular method in the physics-based prognostics, whose fundamental concept is the same as the recursive Bayesian update in Sect. 3.5.1. The difference is that in PF the prior and posterior PDF of parameters are represented by particles (samples). When a new measurement is available, the posterior at the previous step is used as the prior at the current step, and the parameters are updated by multiplying it with the likelihood from the new measurement. Therefore, PF is also known as the sequential Monte Carlo method. In this section, the algorithm for PF is explained with MATLAB code.

It would be better to explain the importance sampling method (Glynn and Iglehart 1989) first to understand PF. In sampling-based methods, a number of samples are used to approximate the posterior distribution. In order to approximate the distribution better with a limited number of samples, the importance sampling method assigns a weight to each sample (or particle) in proportion to an arbitrarily chosen importance distribution; therefore, the quality of estimation depends on the selected importance distribution. The weight of particle $\boldsymbol{\theta}^i$ is expressed as

$$w(\boldsymbol{\theta}^i) = \frac{f(\boldsymbol{\theta}^i|y)}{g(\boldsymbol{\theta}^i)} = \frac{f(y|\boldsymbol{\theta}^i)f(\boldsymbol{\theta}^i)}{g(\boldsymbol{\theta}^i)}, \tag{4.11}$$

where $f(\boldsymbol{\theta}^i|y)$ and $g(\boldsymbol{\theta}^i)$ are the ith particle's PDF value of the posterior distribution and an arbitrarily chosen importance distribution, respectively. The second equality is from the Bayes' theorem, $f(\boldsymbol{\theta}^i|y) = f(y|\boldsymbol{\theta}^i)f(\boldsymbol{\theta}^i)$, where the posterior is expressed as the product of the likelihood and the prior distribution. From the viewpoint of Bayes' theorem, it is possible to use the prior distribution as an importance distribution because it is already available and close to the posterior distribution. Then, Eq. 4.11 is reduced to the likelihood function by substituting the prior $f(\boldsymbol{\theta}^i)$

for $g(\boldsymbol{\theta}^i)$; this is called the Condensation (CONditional DENSity propagATION) algorithm, which is employed here. In this method, the weight in Eq. 4.11 becomes the likelihood.

PF can be considered as a sequential importance sampling method, in which the weights are continuously updated whenever a new observation is available. However, updating weights with the same initial samples yield the degeneracy phenomenon that decreases the accuracy in the posterior distribution with a large variance. This degeneracy phenomenon will be explained in the next section in detail. A resampling process called sequential importance resampling (SIR) (Kim and Park 2011) can be employed to resolve the degeneracy phenomenon. In SIR, those particles with a small weight are eliminated, while those particles with a high weight are duplicated. Even though there is a particle depletion problem in SIR caused by the elimination/duplication process, SIR is a typical PF method and is employed in this book.

Conventionally, PF has been used to estimate the system state using measured data based on model parameters that are obtained from laboratory tests. However, the parameters from laboratory tests can be different from those in service due to environmental conditions. In such a case, PF can also be used to estimate both the system state and model parameters; the detailed procedure can be found in the literature (Zio and Peloni 2011; An et al. 2013), which will be explained here.

The general process of PF is based on the state transition function d that is referred to a degradation model and the measurement function h:

$$z_k = d(z_{k-1}, \boldsymbol{\theta}) \tag{4.12}$$

$$y_k = h(z_k) + \varepsilon, \tag{4.13}$$

where k is the time step index, z_k is the system state (degradation level), $\boldsymbol{\theta}$ is a vector of model parameters, and ε is the measurement noise. In a classic PF, the current system state is a function of the previous system state, z_{k-1}, and process noise that is required for the case of using given model parameters. However, in the purpose of prognostics in this chapter, model parameters are updated and identified with measured data to handle uncertainty, and therefore, the process noise is ignored. On the other hand, measurement noise ε is considered in the measurement system. In the case of Gaussian noise, the noise is represented as $\varepsilon \sim N(0, \sigma^2)$, where σ is the standard deviation of measurement noise. The measurement function in Eq. 4.13 is needed when the system state cannot be measured directly. When the system state can be directly measured, $h(z_k)$ in Eq. 4.13 simply becomes z_k.

In order to use PF, the battery degradation model in Eq. 4.2 needs to be rewritten in the form of the transition function. First, the degradation states at $k-1$ and k are defined as

$$z_{k-1} = \exp(-bt_{k-1}), \quad z_k = \exp(-bt_k), \tag{4.14}$$

and then z_k/z_{k-1} can be calculated as:

$$z_k/z_{k-1} = \exp(-bt_k)/\exp(-bt_{k-1}) = \exp(-bt_k + bt_{k-1}). \tag{4.15}$$

Finally, the state transition function, i.e., degradation function for PF can be obtained from Eq. 4.15 as:

$$z_k = \exp(-b\Delta t)z_{k-1}, \quad \Delta t = t_k - t_{k-1}. \tag{4.16}$$

Note that it is generally possible to convert any degradation model in the form of transition function.

4.4.1 SIR Process

The process of PF is illustrated in Fig. 4.9 with one parameter estimation. Since PF can update the degradation state and model parameter, the parameter θ in the figure can be understood either as a degradation state or a model parameter. In the following, the subscript k is used to denote the time step t_k.

The process starts from assuming the initial distribution of degradation state z_1 based on initial measurement data y_1. The initial distribution of degradation state can be determined with an assumed distribution type and the level of noise. For example, if the level of noise is assumed to be v, then the initial distribution of degradation state can be defined as $f(z_1) \sim U(y_1 - v, y_1 + v)$. The initial distribution of model parameters $\boldsymbol{\theta}_1$ at $t_1 = 0$[2] can also be defined using the prior distribution based on experience or expert opinion as $f(\boldsymbol{\theta}_1)$. Then, n_s particles/samples ($n_s = 10$ in the illustration) are randomly generated from the initial distributions for both degradation states and model parameters. In the initial step, all particles/samples are assumed to have the same weight, $w(\boldsymbol{\theta}^i) = 1/n_s$.

(1) **Prediction**. The initial distributions at $k = 1$ are utilized as a prior distribution at $k = 2$ in the prediction process. In this step, it is assumed that the information from previous step is fully available. That is, n_s samples (or particles) that represent the two PDFs, $f(\boldsymbol{\theta}_{k-1}|\mathbf{y}_{1:k-1})$ and $f(z_{k-1}|\mathbf{y}_{1:k-1})$, are available. The subscript $1:k-1$ means $1, 2, \ldots, k-1$, which is the convention used in MATLAB. In this case, since $k = 2$, it is equivalent to y_1. These are the posterior distributions from $k = 1$, now used as prior distributions at $k = 2$.

In this step, the degradation state at the previous time t_{k-1} is propagated to the current time t_k based on the degradation model such as the transition function in Eq. 4.16. For this, samples of the prior distribution need to be generated. First, n_s samples of $\boldsymbol{\theta}_k$ are generated from $f(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{k-1})$, which means

[2]Model parameters are considered as time independent variables. However, the model parameters undergo updating process, and the subscripts are just to denote the time step during the process.

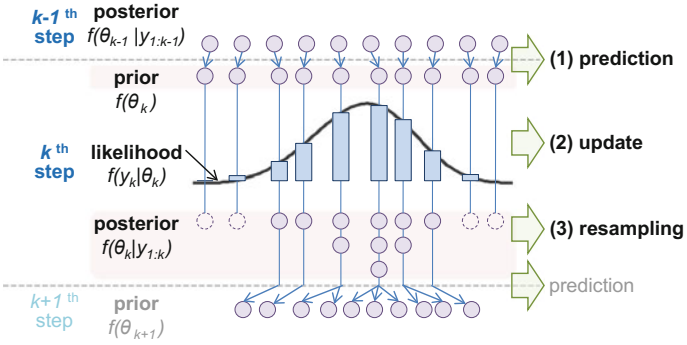


Fig. 4.9 Illustration of Particle Filter process

a predicted distribution of the $\boldsymbol{\theta}_k$ conditional on the previous distribution, $f(\boldsymbol{\theta}_{k-1})$. It, however, should be noted that model parameters inherently do not depend on time evolution. Therefore, n_s samples of $\boldsymbol{\theta}_k$ become the same as the previous n_s particles. In other words, there is no transition between $\boldsymbol{\theta}_{k-1}$ and $\boldsymbol{\theta}_k$. Next, draw n_s samples of z_k based on $f(z_k|z_{k-1}, \boldsymbol{\theta}_k)$, that is, n_s samples of the previous degradation level, z_{k-1} and model parameters, $\boldsymbol{\theta}_k$ are used in the degradation model (Eq. 4.16) to propagate n_s samples of z_k. In summary, the posterior distributions at the previous (k–1)-th step are used to calculate the prior distributions at the current k-th step in the prediction step.

(2) **Update**. Model parameters and degradation state are updated (i.e., corrected) based on the likelihood function from measured data y_k at the current step. For this, the probability of obtaining y_k is calculated conditional on each sample in the prediction step based on the likelihood function. The weight is defined by normalizing the likelihood so that the sum of weights is equal to one. In the case of normally distributed noise, $\varepsilon \sim N(0, \sigma^2)$, the likelihood function and weight become

$$f\left(y_k|z_k^i, \boldsymbol{\theta}_k^i\right) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\frac{\left(y_k - h(z_k^i)\right)^2}{\sigma^2}\right], \quad i = 1, \ldots, n_s \tag{4.17}$$

$$w_k^i = \frac{f\left(y_k|z_k^i, \boldsymbol{\theta}_k^i\right)}{\sum_{j=1}^{n_s} f\left(y_k|z_k^j, \boldsymbol{\theta}_k^j\right)}. \tag{4.18}$$

The length of the vertical bars of the likelihood in the update process in Fig. 4.9 represents the magnitude of weights, and the samples from the prediction step are assigned with the weights, which is the update process. Once the update process is done, the posterior distribution is given in the form of samples and their weights.

Ideally, this process can be repeated for additional measurements in following times. This update process, however, causes the degeneracy

phenomenon that decreases the accuracy in the posterior distribution and causes the lack of reducing the range of distribution by keeping samples with low weights. To solve this problem, the resampling process is required. The ideal situation is that all samples have the same weight so that samples, not weight, can represent the posterior distribution of model parameters and degradation state.

(3) **Resampling**. The idea of resampling is to maintain all samples (i.e., particles) with the same weight. In order to do that, samples at the update step are duplicated or eliminated based on the magnitude of the weights. Among several methods, the so-called inverse CDF method introduced in Chap. 3 is used (see Fig. 3.9/3.10), which is explained as follows: (a) Construct CDF of y_k from the likelihood function in Eq. 4.17. In other words, CDF is based on the weights in Eq. 4.18 which corresponds to the PDF of the y_k conditional on $(z_k^i, \boldsymbol{\theta}_k^i)$. (b) Find samples of $(z_k, \boldsymbol{\theta}_k)$ which make the CDF of y_k be the same value as (or the closest to) the randomly chosen value from $U(0, 1)$. By repeating this process n_s times, n_s samples of $(z_k, \boldsymbol{\theta}_k)$ with a constant weight are obtained, which represent an approximation of the posterior distribution, which was based on the weighted samples $\left(z_k^i, \boldsymbol{\theta}_k^i | w_k^i\right)$. After resampling process, every sample is assigned with the same weight of $w_k^i = 1/n_s$.

Note that in Bayesian method the posterior distribution is composed of the product of the prior distribution with the likelihood. In PF, the prior distribution is represented using particles, and the likelihood information is stored in weights. The resampling process is then equivalent to generate a new set of samples that follow the posterior distribution. Those particles with a large weight are likely duplicated, and those particles with a small weight are likely removed. Since these new samples follow the posterior distribution, all samples are assigned with the same weight.

The posterior distribution at the current step is used as a prior distribution at the next $(k + 1)$-th step, which means that the Bayesian update is processed recursively in PF.

Example 4.3 PF process with 10 samples

Demonstrate the PF process for the battery problem using Eq. 4.16 with ten samples. It is assumed that the initial distributions are given as $z_1 \sim U(0.9, 1.1), b_1 \sim U(0, 0.005)$, and the noise in data is normally distributed with $\sigma = 0.02$. Use three capacity and time data at $k = 1, 2, 3$ in Table 4.1 to update the model parameter b and degradation state z_k.

Solution:

From Table 4.1, the first three measured data are given as .

```
y=[1.00 0.99 0.99]';.
```

At $k = 1$:

The initial distribution of the degradation is uniformly distributed between 0.9 and 1.1, which is based on the information from the data `y(1)` with

assumed range of data noise of 0.1. In addition, the initial distribution of the model parameter needs to be properly assumed. Here the initial distribution of model parameters is assumed to be uniformly distributed between 0 and 0.005. The PF process starts from generating ten samples from the given initial distribution, which is shown in Fig. E4.5 and obtained from the following code. Note that the actual values of samples will be different due to random number generation.

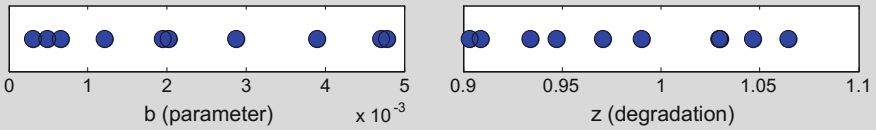


Fig. E4.5 Sampling results of initial distributions (posterior at $k = 1$)

```
k=1;
ns=10;
b(k,:)=unifrnd(0,0.005,ns,1);
z(k,:)=unifrnd(0.9,1.1,ns,1);
figure;
subplot(1,2,1);
plot(b(k,:),zeros(1,ns),'ok'); xlabel('b (parameter)')
subplot(1,2,2);
plot(z(k,:),zeros(1,ns),'ok'); xlabel('z (degradation)')
```

At $k = 2$,

(1) Prediction: The samples of model parameters are the same as the previous time $k = 1$, while degradation state is propagated based on the model equation in Eq. 4.16. The samples of model parameter are in `b(2,:)` and that of the previous degradation level are in `z(1,:)`. In Fig. E4.6, the samples of parameter are exactly same as those in Fig. E4.5, but the samples of degradation are not, which is obtained from the following code.

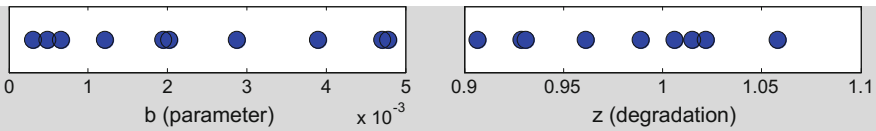


Fig. E4.6 Sampling results of prediction step (prior at $k = 2$)

```
k=2;
dt=5;
b(k,:)=b(k-1,:);
z(k,:)=exp(-b(k,:).*dt).*z(k-1,:);
% for figure plot, use the same code given in k=1
```

(2) Update: Since the error in data is normally distributed, the likelihood of data is calculated based on Eq. 4.17, in which $h(z_k^i) = z_k^i$, $\sigma = 0.02$, $i = 1, \ldots, 10$. At this stage, the posterior distribution is in the form of samples (prior) with weights (likelihood). The sum of weights is one by normalizing the likelihood as shown in Eq. 4.18. In order to simplify the resampling process, the weights are expressed as the number of samples by multiplying the total number of samples to the weights. Therefore, the weights in this example represent the number of samples to be duplicated, which is shown in Fig. E4.7 and can be obtained in the following code.

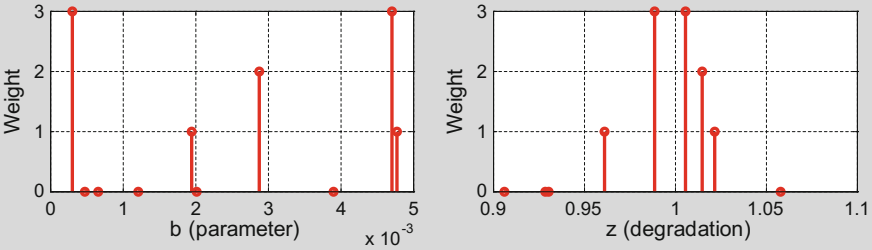


Fig. E4.7 Weight calculation (likelihood at $k = 2$)

```
likel=normpdf(y(k,:),z(k,:),0.02);
weigh=round(likel/sum(likel)*ns);
figure;
subplot(1,2,1); stem(b(k,:),weigh,'r');
xlabel('b (parameter)'); ylabel('Weight')
subplot(1,2,2); stem(z(k,:),weigh,'r');
xlabel('z (degradation)'); ylabel('Weight')
```

(3) Resampling: As explained before, the inverse CDF method can be employed as a practical resampling method. However, to make the resampling concept easier, samples having zero weight are simply eliminated, and other samples are duplicated as many as the weight in

Fig. E4.7. The results are shown in Fig. E4.8, which is obtained from the following code. In the figure, several samples are overlapped, which yields the particle depletion phenomenon.

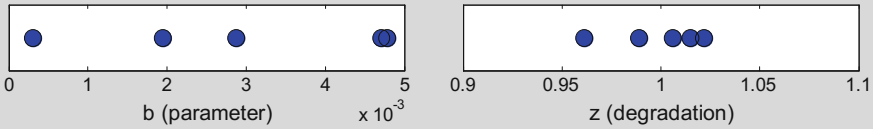


Fig. E4.8 Resampling results (posterior at $k = 2$)

```
sampl=[b(k,:); z(k,:)];
loca=find(weigh>0); ns0=1;
for i=1:length(loca);
 ns1=sum(weigh(loca(1:i)));
 b(k,ns0:ns1)=sampl(1,loca(i))*ones(1,weigh(loca(i)));
 z(k,ns0:ns1)=sampl(2,loca(i))*ones(1,weigh(loca(i)));
 ns0=ns1+1;
end
% for figure plot, use the same code given in k=1
```

At $k = 3$,

The same process (prediction, update, and resampling) is done at $k = 3$, and the results are given in Fig. E4.9 that is updated results from Fig. E4.5.

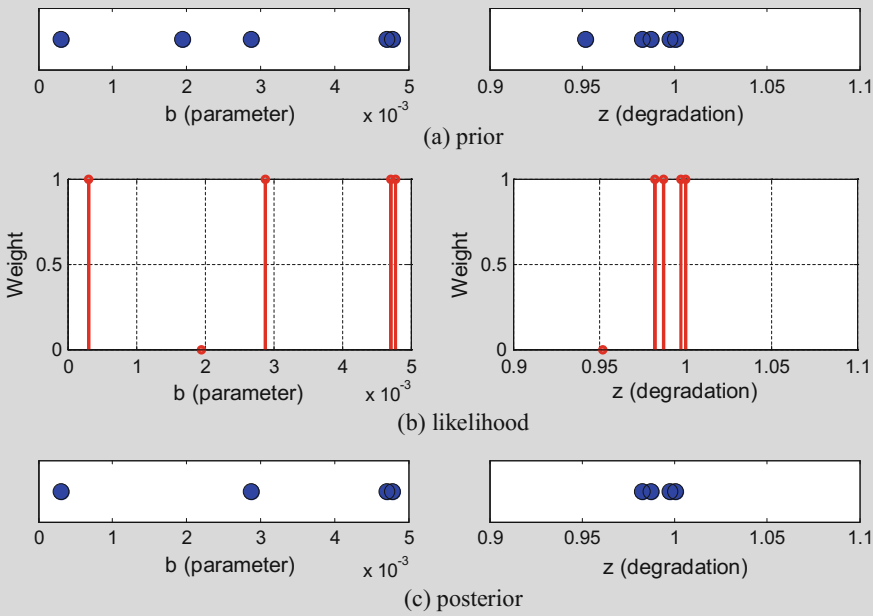


Fig. E4.9 Sampling and weight at $k = 3$

Once the model parameters and degradation state are updated using given data up to the current time, the degradation state z_{k+l} at a future time t_{k+l} is predicted based on the degradation function with the degradation state at the previous time, z_{k+l-1} and estimated model parameters, $\boldsymbol{\theta}_k$. In the prediction stage, the model parameters are not updated any more. The degradation state is continuously propagated until reaching the threshold. Let us assume that the model parameter b in Eq. 4.16 is updated as 0.004 based on measured data up to the current time (time index is k), the degradation state $z_k = 0.85$, and time interval $\Delta t = 5$. This case can be considered as $n_s = 1$, and future degradation can be propagated as:

at $k+1$, $z_{k+1} = \exp(-b_k \Delta t) z_k = \exp(-0.004 \times 5) \times 0.85 = 0.83$,
at $k+2$, $z_{k+2} = \exp(-b_k \Delta t) z_{k+1} = \exp(-0.004 \times 5) \times 0.83 = 0.81$.
which is repeated until the degradation state reaches the threshold. When the threshold is 0.7, the degradation propagated up to $k+10$ as below.
at $k+10$, $z_{k+10} = \exp(-b_k \Delta t) z_{k+9} = \exp(-0.004 \times 5) \times 0.71 = 0.70$

Since the time interval is 5 cycles, and the degradation is propagated 10 steps further, RUL can be predicted as 50 (5×10) cycles.

4.4.2 *MATLAB Implementation of Battery Prognostics*

In this section, the MATLAB implementation of PF is explained using the same battery problem in Sect. 4.1.1. Problem definition and post-processing parts are similar to BM with minor modifications. Therefore, focus is on the prognosis part of the code.

4.4.2.1 Problem Definition (Lines 5–16, 64)

The required variables for PF are almost the same as those of BM, but the time interval for degradation propagation `dt` is introduced because PF uses incremental form of degradation. It can be different from the measurement time interval, but the current implementation assumes that the interval of health monitoring is an integer multiple of `dt` (variable `nk` in line 27). The model error is reduced with a smaller interval. Mathematically, this time interval is used to integrate the differential form of degradation model using the forward finite difference method, which is stable and accurate when the time interval is small. In the PF process, the degradation state, z, is included in the parameters to be estimated. Therefore, the term for degradation should be placed on the last row in `ParamName`, `initDisPar`, and `thetaTrue` as listed below.

```
dt=5;
ParamName=['b'; 's'; 'z'];
initDisPar=[0 0.02; 1e-5 0.1; 1 1];
thetaTrue=[0.003; 0.02; 1];
```

The degradation variable is named as `z`, and its true initial value `1` is assigned to `thetaTrue(3)`. For the initial distribution, the lower and the upper bounds are set as the same value as the true initial value, that is $f(z) \sim U(1, 1)$. This is because it was assumed that the initial capacity is given as 1 (C/1 capacity) without considering uncertainty in it.

The degradation model equation in Eq. 4.16 is used in line 64. In the equation, the time interval Δt is expressed as `dt` in the script, which is defined in line 9. Also, the degradation state at the previous step z_{k-1} and the current step z_k, respectively, are denoted as `z` (should be the same as the notation defined in line 12) and `z1` (can vary, but should be consistent with `z1` in line 60). The following code is used to express the battery degradation model for PF.

```
z1=exp(-b.*dt).*z;
```

4.4.2.2 Prognosis Using PF (Lines 19–45)

Once problem is properly defined, the code can be run with the following command in MATLAB:

```
[thetaHat,rul]=PF;
```

The prognostics process is started by drawing `ns` samples from initial/prior distributions of the parameters (lines 21–23), and then degradation is propagated until reaching the ending time in the for-loop (lines 24–41). Within the for-loop, the parameters are updated when the measured data are available (lines 29–37). During the for-loop, since there are `ny` data are available, the model parameters are updated when time step `k <= ny` (training stage). On the other hand, when `k > ny`, there is no measurement data available and the degradation level is propagated without updating model parameters (prediction stage) (line 39).

As explained in the previous section, there are three steps in the training stage. At the prediction step (lines 26–28), the parameters at the previous step `param(:,:,k-1)` are copied to the parameters at the current step, `paramPredi` (line 26). On the other hand, the degradation levels, `paramPredi(np,:)` need to be propagated from the previous step using `MODEL` function. This propagation is based

on the previous model parameters and time interval `dt` (line 28). Since the propagation has to be done from `time(k-1)` to `time(k)`, this propagation has to be repeated `nk` times with `dt` interval. The samples in this step correspond to $f(\theta_k)$ in Fig. 4.9.

Next is the updating step (line 31), which is related to the likelihood of measurement data given in Eq. 4.17 and $f(y_k|\theta_k)$ in Fig. 4.9. This likelihood is calculated for every sample of degradation level `paramPredi(np,:)` with the sample of measurement noise `paramPredi(np-1,:)`.

Finally, the resampling step based on the inverse CDF method is given in lines 33–37. In this step, the cumulative CDF of likelihood samples is calculated first (line 33), and then, a sample from uniform distribution between zero and one is used to find the approximate location of likelihood sample based the inverse CDF method (line 36). The location found in the inverse CDF becomes a new sample. This inverse CDF method is repeated by `ns` times to generate new `ns` samples, which is the resampling step. In this way, a sample with a large likelihood (weight) has more probability to be duplicated, while a sample with a small likelihood may not be reproduced. This process is repeated for all measurement data.

After both the training and prediction stages are finished, model parameters and the standard deviation of measurement noise are stored in `thetaHat` (line 42). For the purpose of calculating prediction interval, first the samples of degradation levels are stored in array `zHat(41,ns)` in line 44. Since the prediction interval includes the effect of measurement noise in the future, normal random noise is added to the degradation levels in the prediction stage (line 45).

4.4.2.3 Post-processing (Lines 47–58)

The post-processing in PF is the same as the NLS and BM. When the true degradation is given, it can be used in the same was as in NLS and BM. However, if the true degradation is not available but only true model parameters are available, the degradation model is integrated to generated the true degradation level (lines 47–56). As with NLS and BM, the MATLAB function POST can be used to plot the histograms of model parameters, the degradation history as a function of time cycles, and the histogram of RUL. The distributions in the estimated parameters in Fig. 4.10a are not as smooth as ones in Fig. 4.7a from BM, which is because of the particle depletion phenomenon. It does not have an effect on the prediction results in this problem though as shown in Figs. 4.10b and 4.11. The prediction results of degradation and RUL are similar with ones from NLS and BM.

[PF]: MATLAB Code for Particle Filter

```
function [thetaHat,rul]=PF
clear global; global DegraUnit initDisPar TimeUnit ...
time y thres ParamName thetaTrue signiLevel ns ny nt np
%=== PROBLEM DEFINITION 1 (Required Variables) ==============
WorkName=' ';            % work results are saved by WorkName
DegraUnit=' ';                              % degradation unit
TimeUnit=' ';               % time unit (Cycles, Weeks, etc.)
time=[ ]';          % time at both measurement and prediction
dt= ;             % time interval for degradation propagation
y=[ ]';                                  %[nyx1]: measured data
thres= ;                          % threshold (critical value)
ParamName=[ ];      %[npx1]: parameters' name to be estimated
initDisPar=[ ]; %[npx2]: prob. parameters of init./prior dist
thetaTrue=[ ];             %[npx1]: true values of parameters
signiLevel= ;          % significance level for C.I. and P.I.
ns= ;                           % number of particles/samples
%============================================================
% % % PROGNOSIS using PF
ny=length(y); nt=ny;
np=size(ParamName,1);
for j=1:np;                               %% Initial Distribution
 param(j,:,1)=unifrnd(initDisPar(j,1),initDisPar(j,2),1,ns);
end;
for k=2:length(time);          %% Update Process or Prognosis
 % step1. prediction (prior)
 paramPredi=param(:,:,k-1);
 nk=(time(k)-time(k-1))/dt;
 for k0=1:nk; paramPredi(np,:)=MODEL(paramPredi,dt); end
 if k<=ny                                      % (Update Process)
  % step2. update (likelihood)
  likel=normpdf(y(k),paramPredi(np,:),paramPredi(np-1,:));
  % step3. resampling
  cdf=cumsum(likel)./sum(likel);
  for i=1:ns;
   u=rand;
   loca=find(cdf>=u,1); param(:,i,k)=paramPredi(:,loca);
  end;
 else                                             % (Prognosis)
  param(:,:,k)=paramPredi;
 end
end
thetaHat=param(1:np-1,:,ny);        %% Final Sampling Results
paramRearr=permute(param,[3 2 1]);  %% Degradation Prediction
zHat=paramRearr(:,:,np);
degraPredi=normrnd(zHat(ny:end,:),paramRearr(ny:end,:,np-1));
% % % POST-PROCESSING
degraTrue=[];                                %% True Degradation
```

```
if ~isempty(thetaTrue); k=1;
 degraTrue0(1)=thetaTrue(np); degraTrue(1)=thetaTrue(np);
 for k0=2:max(time)/dt+1;
  degraTrue0(k0,1)= ...
              MODEL([thetaTrue(1:np-1); degraTrue0(k0-1)],dt);
  loca=find((k0-1)*dt==time,1);
  if ~isempty(loca); k=k+1; degraTrue(k)=degraTrue0(k0);end
 end
end
rul=POST(thetaHat,degraPredi,degraTrue); %% RUL & Result Disp
Name=[WorkName ' at ' num2str(time(ny)) '.mat']; save(Name);
end
function z1=MODEL(param,dt)
 global ParamName np
 for j=1:np; eval([ParamName(j,:) '=param(j,:);']); end
 %===== PROBLEM DEFINITION 2 (model equation) ===============

 %============================================================
end
```

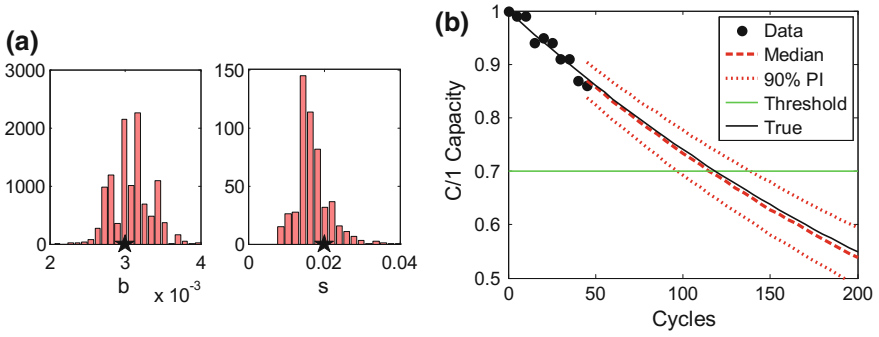
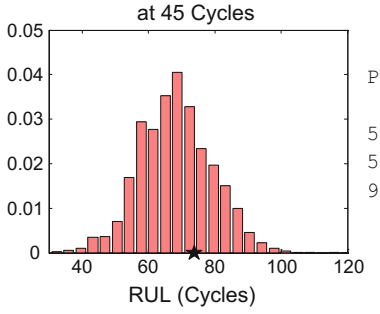


Fig. 4.10 Results of the battery problem: PF. **a** Distribution of estimated parameter, **b** degradation prediction




```
Percentiles of RUL at 45 Cycles

5th: 50.7712,
50th (median): 68.0799,
95th: 86.7071
```

Fig. 4.11 RUL prediction results: PF

4.5 Practical Application of Physics-Based Prognostics

Although the three programs, **[NLS]**, **[BM]** and **[PF]**, are written for battery degradation prognostics, they can easily be modified by users for different applications. As an example, the usage of three algorithms with a crack growth example is discussed in the following subsections. Users are expected to modify the problem definition section as well as model equation in function `MODEL`.

4.5.1 Problem Definition

4.5.1.1 Crack Growth Model

The crack growth model introduced in Chap. 2 is employed here to explain how this model can be used in the three prognostics algorithms. For **[NLS]** and **[BM]**, it is necessary to have a model that can calculate the crack size as a function of time/cycle, while a transition function (incremental form) is required for **[PF]**. Equation 2.2 can be used as a degradation model for NLS and BM as

$$a_k = \left[N_k C\left(1-\frac{m}{2}\right)\left(\Delta\sigma\sqrt{\pi}\right)^m + a_0^{1-\frac{m}{2}}\right]^{\frac{2}{2-m}}, \tag{4.19}$$

which expresses the crack size a_k after N_k cycles of fatigue loading (this is the same as Eq. 2.2). On the other hand, this model can be rewritten in the form of the state transition function for PF:

$$a_k = C(\Delta\sigma\sqrt{\pi a_{k-1}})^m dN + a_{k-1}. \tag{4.20}$$

It is assumed that the crack size y_k is measured at every 100 cycles under loading condition $\Delta\sigma$ = 75 MPa, which is listed in Table 4.2, and is obtained with the following steps. First, the true crack size data are generated at every 100 cycles using Eq. 4.19 with $m_{\text{true}} = 3.8, C_{\text{true}} = 1.5 \times 10^{-10}$ and $a_{0,\text{true}} = 0.01$ m. The measured crack size data are then generated by adding Gaussian noise

Table 4.2 Measurement data for crack growth problem

Time step, k	Initial, 1	2	3	4	5	6	7	8
Time (cycles)	0	100	200	300	400	500	600	700
Crack size (m)	0.0100	0.0109	0.0101	0.0107	0.0110	0.0123	0.0099	0.0113
Time step, k	9	10	11	12	13	14	15	16
Time (cycles)	800	900	1000	1100	1200	1300	1400	1500
Crack size (m)	0.0132	0.0138	0.0148	0.0156	0.0155	0.0141	0.0169	0.0168

$\varepsilon \sim N(0, \sigma^2), \sigma = 0.001m$ to the true crack sizes. The generated data are used to identify two model parameters, $\boldsymbol{\theta} = \{m \ \ln(C)\}^{\mathrm{T}}$. In Paris model, the y-intercept C is very small but changes its magnitude by several orders. Therefore, it would be better to identify logarithm of C. For simplicity of presentation, the initial crack size, a_0, and the magnitude of noise, σ, are assumed to be known. For RUL calculation, the critical crack size is determined as 0.050 m.

4.5.1.2 Likelihood and Prior Distribution

For the Bayesian method and particle filter, it is necessary to define the prior distribution and the likelihood function. In the previous chapters, it is often assumed that noise in data follows a normal distribution. However, when the distribution type of measurement noise is unknown, it is possible that the likelihood function might be different from the true noise distribution. The same is true for the prior/initial distribution. Therefore, it would be a good exercise to study the effect of different distribution types by changing the MATLAB codes. In this example, the lognormal distribution is employed for the likelihood function as:

$$f\left(y_k | m_k^i, C_k^i\right) = \frac{1}{y_k \sqrt{2\pi} \zeta_k^i} \exp\left[-\frac{1}{2}\left(\frac{\ln y_k - \eta_k^i}{\zeta_k^i}\right)^2\right], \quad i = 1, \ldots, n_s, \tag{4.21}$$

where

$$\zeta_k^i = \sqrt{\ln\left[1 + \left(\frac{\sigma}{a_k^i(m_k^i, C_k^i)}\right)^2\right]}$$

and

$$\eta_k^i = \ln\left[a_k^i(m_k^i, C_k^i)\right] - \frac{1}{2}\left(\zeta_k^i\right)^2$$

are standard deviation and mean of lognormal distribution, respectively. In the above equations, $a_k^i(m_k^i, C_k^i)$ is the model prediction from Eq. 4.19 at time t_k with given model parameters m_k^i and C_k^i.

Also, the prior/initial distributions of the parameters are assumed as normal distributions as:

$$f(m) = N\left(4, 0.2^2\right), \quad f(\log C) = N\left(-23, 1.1^2\right). \tag{4.22}$$

4.5.2 Modifying the Codes for the Crack Growth Example

4.5.2.1 NLS

Based on the given information in Sect. 4.5.1, the problem definition part (lines 5–14) in the code **[NLS]** is changed as follows:

```
WorkName='Crack_NLS';
DegraUnit='Crack size (m)';
TimeUnit='Cycles';
time=[0:100:3500]';
y=[0.0100 0.0109 0.0101 0.0107 0.0110 0.0123 0.0099 0.0113...
   0.0132 0.0138 0.0148 0.0156 0.0155 0.0141 0.0169 0.0168]';
thres=0.05;
ParamName=['m'; 'C'; 's'];
thetaTrue=[3.8; log(1.5e-10); 0.001];
signiLevel=5;
ns=5e3;
```

Even though the standard deviation of noise in data, σ, is not an unknown parameter in this example and it is calculated based on the identified parameters in NLS. Also as mentioned in Sect. 4.2, it is added to `ParamName` to maintain consistency with other codes. Accordingly, the true value of σ is added to `thetaTrue`.

For the model equation, the following code is used in line 48:

```
a0=0.01; dsig=75;
z=(t.*exp(C).*(1-m./2).*(dsig*sqrt(pi)).^m ...
                                  +a0.^(1-m./2)).^(2./(2-m));
loca=imag(z)~=0; z(loca)=1;
```

This corresponds to Eq. 4.19, but note that $\log(C)$ is used instead of C. The above model equation from Eq. 4.19 can yield a complex number when crack size grows too large. Therefore, when a crack size becomes a complex number, it is replaced with an arbitrary large crack size, which is taken here as 1.0 m.

Even though the likelihood function is not necessary for parameter estimation process in NLS, it is used to predict degradation in line 31. The lognormal distribution in Eq. 4.21 is used instead of normal distribution as:

```
mu=zHat(k,:);
C=chi2rnd(ny-1,1,ns);
s=sqrt((ny-1)*thetaHat(np,:).^2./C);
zeta=sqrt(log(1+(s./mu).^2)); eta=log(mu)-0.5*zeta.^2;
degraPredi(k,:)=lognrnd(eta,zeta);
```

Once all required information are defined, the following code is used to run **[NLS]**. Note that only the initial values of model parameters are used in `theta0` (line 1) since the measurement error σ is calculated after the model parameters are estimated regardless of whether σ is known or unknown.

```
[thetaHat,rul]=NLS([4; -23]);
```

The results of NLS algorithm will be discussed with other algorithms in the result section.

4.5.2.2 BM

The required variables (lines 5–16) and model definition (line 52) for BM are the same as those for NLS, but the following three are additionally required for BM.

```
WorkName='Crack_BM';
initDisPar=[4.0  0.2;  -23  1.1;  0.001  1e-5];
burnIn=0.2;
```

The `initDisPar` corresponds to Eq. 4.22, which are the probability parameters of normal distribution, the mean, and the standard deviation. The last two values, `0.001` and `1e-5` are for σ (=s). In this example, even if σ is a deterministic value (the true value is known and/or not need to be estimated), it should be included in `initDisPar`. In order to have the same effect of known deterministic σ, its standard deviation is set to be zero (1e-5 is used to prevent numerical error).

The degradation prediction in line 40 is also modified with the same code as given in NLS part based on Eq. 4.21. The difference between BM and NLS is that the likelihood and prior distribution are used to identify the model parameters not only for the degradation prediction in BM, thereby lines 56–59 are replaced with the following code:

```
prior=prod(normpdf(param,initDisPar(:,1),initDisPar(:,2)));
likel=1;
for k=1:ny;
 zeta=sqrt(log(1+(s./z(k)).^2)); eta=log(z(k))-0.5*zeta.^2;
 likel=lognpdf(y(k),eta,zeta).*likel;
end
```

To run the code **[BM]**, the initial samples of parameters `para0` and the weights `weigh` are typed in line 1 as

```
[thetaHat]=BM([4  -23  0.001]',[0.1  0.2  0]');
```

Note that σ is included here but the weight for σ is zero to fix it during the sampling process, which corresponds to `unifrnd(0.001-0,0.001+0)` in line 25. In this example, the proposal distribution is still a uniform distribution, but it can be other forms, e.g., for the case of normal distribution, line 25 can be replaced with `normrnd(para0,abs(para0.*weigh));`.

4.5.2.3 PF

The required variables (lines 5–16) for PF are given below (the variables other than these are the same as NLS/BM.):

```
WorkName='Crack_PF';
dt=20;
ParamName=['m'; 'C'; 's'; 'a'];
initDisPar=[4.0  0.2;  -23  1.1;  0.001  0;  0.01  0];
thetaTrue=[3.8;log(1.5e-10); 0.001; 0.01];
```

As explained before, the degradation state term `a` is included in `ParamName`, `initDisPar`, and `thetaTrue`. Since the initial crack is assumed as 0.01 m, its standard deviation is also set to be zero.[3] The model definition in line 64 is replaced with the code corresponding Eq. 4.20 as

[3]In PF, zero standard deviation is not need to be corrected as 1e-5 because it is related with random sampling rather than a PDF calculation.

```
dsig=75;
z1=exp(C).*(dsig.*sqrt(pi*a)).^m.*dt+a;
```

In the same manner as NLS and BM, lognormal distribution is employed to predict degradation in line 45.

```
mu=zHat(ny:end,:); s=paramRearr(ny:end,:,np-1);
zeta=sqrt(log(1+(s./mu).^2)); eta=log(mu)-0.5*zeta.^2;
degraPredi=lognrnd(eta,zeta);
```

For the initial distribution in line 22, the following code is used,

```
param(j,:,1)=normrnd(initDisPar(j,1),initDisPar(j,2),1,ns);
```

and line 31 is replaced with the following code.

```
mu=paramPredi(np,:); s=paramPredi(np-1,:);
zeta=sqrt(log(1+(s./mu).^2)); eta=log(mu)-0.5*zeta.^2;
likel=lognpdf(y(k),eta,zeta);
```

There are no specific inputs required in PF, therefore the code to run **`[PF]`** is the same regardless of different applications as.

```
[thetaHat,rul]=PF;
```

4.5.3 Results

The results of parameter identification and degradation prediction from NLS, BM, and PF are shown in Figs. 4.12, 4.13 and 4.14, respectively. The correlation plots are obtained from the final sampling results in `thetaHat` with `plot(thetaHat(1,:),thetaHat(2,:),'.')`. The most notable difference in results is that there are very large uncertainties in the results from NLS compared to the others. This is because there is a very wide range of correlation between m and C, but it cannot be reduced since the prior information cannot be employed in NLS.

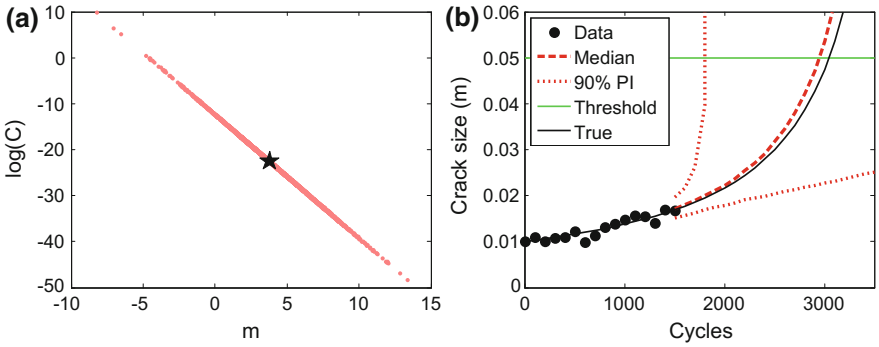


Fig. 4.12 Results of the crack growth problem: NLS. **a** Correlation, **b** degradation prediction

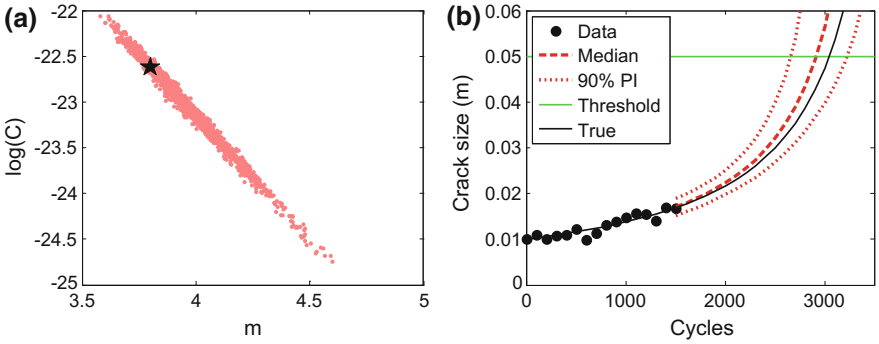


Fig. 4.13 Results of the crack growth problem: BM. **a** Correlation, **b** Degradation prediction

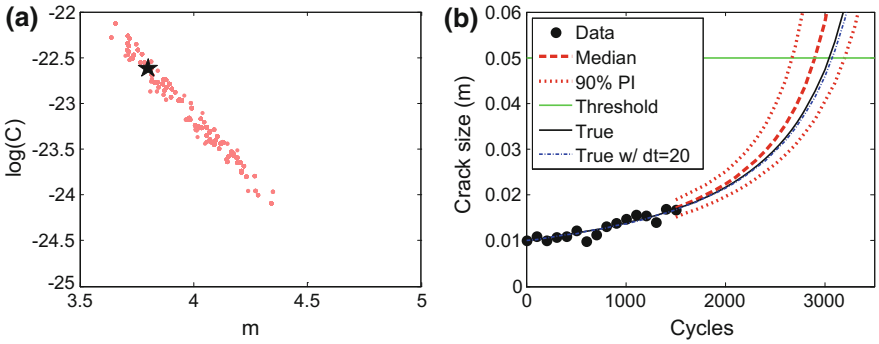


Fig. 4.14 Results of the crack growth problem: PF. **a** Correlation, **b** degradation prediction

The parameters estimated in a very wide range can yield a huge difference in the degradation prediction. On the other hand, the parameters from BM and PF are identified in a narrow range compared to NLS by employing the prior information, and thereby the prediction uncertainty is much smaller than that of NLS.

Another important issue needed to be mentioned is about the model error caused by discontinuous transient function in PF. In this example, the time interval `dt=20` is used, but there is a difference in the degradation rate between the models in Eq. 4.19 and in Eq. 4.20, which is shown in Fig. 4.14b. In the figure, the black solid curve and the blue dot-dashed curves correspond to Eq. 4.19 and Eq. 4.20, respectively. If the transition function is viewed as a differential equation, Eq. 4.20 corresponds to the forward Euler finite difference method, which requires small time intervals to be accurate. This gap can be reduced with smaller time interval.[4] More in-depth comparison between each method will be discussed in Chap. 6.

4.6 Issues in Physics-Based Prognostics

Compared to data-driven methods that will be discussed in Chap. 5, physics-based prognostics algorithms have several advantages. First, physics-based methods can make a long-term prediction. Once the model parameters are identified accurately, it is possible to predict the remaining useful life by propagating the physical model until degradation reaches a predetermined threshold. Second, physics-based methods require relatively small number of data. Theoretically, it is possible to identify model parameters when the number of data is the same as the number of unknown model parameters. In reality, however, due to noise in data and due to insensitivity of degradation behavior to parameters, a more number of data are required. Still, the required number of data in physics-based prognostics algorithms is much less than that of data-driven methods.

On the negative side, there are three important issues in physics-based prognostics to be practical: model adequacy, parameter estimation, and quality of data. In this section, these issues are addressed with some literature reviews. More comprehensive discussions will be presented in Chaps. 6 and 7.

[4]Smaller time interval means more iterations, which requires more computational time and prone to yield numerical error. To prevent this error, the initial distribution of model parameters should be set properly.

4.6.1 Model Adequacy

Model adequacy is an issue addressing if the physical model is good enough to predict the future behavior of degradation. This is slightly different from the conventional curve-fitting issue in regression because regression pays attention to accuracy between data, which is in a sense the error in interpolating region. In prognostics, however, the interest is in beyond measure data points, which is the error in extrapolation.

A model that can fit the data very well in the interpolating region does not mean it can also predict the trend in the extrapolating region. A crack growth example is used to demonstrate the model adequacy issue. In the following MATLAB code, a Paris model is used to generate six crack size data at flight cycles between 10,000 and 20,000 with intervals of 2,000. These data are shown as blue circles in Fig. 4.15. For a physical model, a cubic polynomial with four unknown coefficients is selected for the purpose of explanation. The MATLAB function `regress` can find the unknown coefficients that minimize error between the six data and polynomial prediction. The solid black curve in Fig. 4.15 shows the model prediction, which is very accurate as the difference between data and model prediction is almost zero. Therefore, in the viewpoint of curve-fitting, this model is considered to be good. However, in the extrapolating region, the conclusion can be completely different. In the figure, the same model is used to predict the crack growth at flight cycles between 20,000 and 30,000 with the same coefficients, which is shown as a red curve in the figure. However, as shown in red circles, the actual crack grows much faster than the model prediction. Since the model predicts a slower crack growth, it would be dangerous to rely on it. Therefore, the adequate model in the interpolating region can be inadequate in the extrapolating region. Therefore, it is

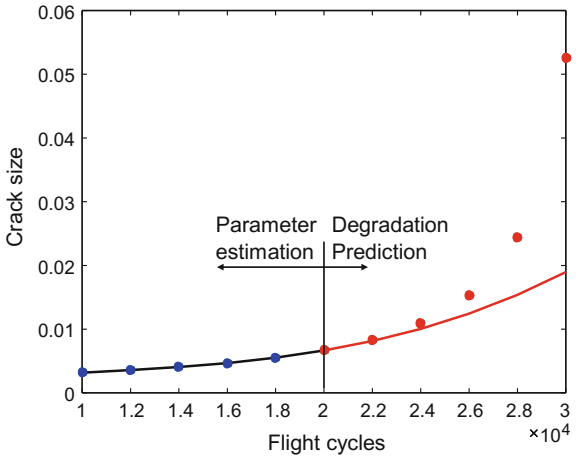


Fig. 4.15 Model adequacy in physics-based prognostics

very important to validate that the physical model is accurate for the purpose of prognostics.

```
m=3.6; C=1E-10;a0=2E-3;dels=80;      % Paris model parameters
N=[10000:2000:20000]';                    % Training cycles
a=(N.*C.*(1-m/2)*(dels*sqrt(pi)).^m+a0^(1-m/2)).^(2/(2-m));
X=[ones(size(N)) N N.*N N.^3];
b=regress(a,X);                   % Fitting cubic polynomials
afit=b(1) + b(2).*N + b(3).*N.^2 + b(4).*N.^3;
plot(N,a,'o',N,afit,'b'); hold on;
M=[20000:2000:30000]';                    % Prediction cycles
ap=(M.*C.*(1-m/2)*(dels*sqrt(pi)).^m+a0^(1-m/2)).^(2/(2-m));
apfit=b(1) + b(2).*M + b(3).*M.^2 + b(4).*M.^3;
plot(M,ap,'or',M,apfit,'r');
```

Since physics-based approaches employ a physical model describing the behavior of damage, they have advantages in predicting long-term behaviors of damage. However, model validation should be carried out first, since most models contain assumptions and approximations. There has been much literature on model validation using statistical methods, such as the hypothesis test and the Bayesian approach (Oden et al. 2013; Rebba et al. 2006; Sargent 2013; Ling and Mahadevan 2013).

In general, as the model complexity increases, the number of model parameters increases, and estimation of parameters becomes more difficult. Recently, Coppe et al. (2012) showed that the issue of model adequacy can be relaxed by identifying equivalent parameters from a simpler model. A simple Paris model was used with an assumed stress intensity factor to predict crack growth of complex geometries in which the model parameters were adjusted to compensate for the error in the stress intensity factor. Although this is limited to the case of a similar damage behavior between the simple and complex model, additional efforts to validate model accuracy can be avoided.

4.6.2 Parameter Estimation

Parameter estimation is the most important step in physics-based prognostics because once model parameters are determined, predicting the remaining useful life is straightforward. There are two specific issues in physics-based parameter estimation: (1) estimation accuracy that is related to characteristics of different algorithms, such as NLS, BM, and PF in this chapter, and (2) correlations between model parameters and also between model parameters and loading conditions that hinders identifying accurate parameters. A good prognostics algorithm can identify

accurate model parameters with a less number of data. For correlation, Chaps. 6 and 7 will show that even if the accurate identification of parameters might be difficult, it is possible that accurate prediction in degradation and remaining useful life are still possible. Since comparison and case study are required to introduce these two issues, more details will be discussed in Chaps. 6 and 7.

4.6.3 Quality of Degradation Data

In order to estimate model parameters of system in service, structural health monitoring (SHM) data are often used to collecting data that are used for the purpose of prognosis. SHM data could include a high level of noise and bias due to the sensor equipment and measurement environment. Noise is a random fluctuation in measured data or signal due to interference or unwanted electromagnetic fields in electronic devices. Bias is static deviation of signals from correct ones, which can happen due to calibration error. Noise makes it difficult to identify signals related to degradation, while bias induces error in prediction. In fact, filtering noise and compensating bias has been the major research issues in prognostics using SHM data.

Gu et al. (2008) presented a prognostics approach that detects the performance degradation of multilayer ceramic capacitors under temperature–humidity-bias conditions. Coppe et al. (2009) showed that the uncertainty in structure-specific damage growth parameters can be progressively reduced, in spite of noise and bias in sensor measurements. Guan et al. (2009) considered various uncertainties from measurements, modeling and parameter estimations to describe the stochastic process of fatigue damage accumulation based on a maximum entropy-based general framework. It was found that large noise induces slow convergence, and bias shifts parameter distribution.

Noise in sensor signals hinders the detection of degradation features, which adversely affect the prognostic capability in both the physics-based and data-driven approaches. To relieve this, de-noising is usually conducted in signal processing. Zhang et al. (2009) proposed a de-noising scheme for improving the signal-to-noise ratio and applied it to vibration signals obtained from a helicopter gearbox test-bed. Qiu et al. (2003) introduced an enhanced robust method for bearing prognostics, which includes a wavelet filter-based method for extraction of a weak fault feature from vibration signals and a self-organizing map based method for bearing degradation assessment. Abouel-seoud et al. (2011) introduced a robust prognostics concept based on an optimal wavelet filter for fault identification of a gear tooth crack by acoustic emission. The quality of data issues will be discussed further in Chaps. 6 and 7.

4.7 Exercise

P4.1 Find the model parameter and its variance of $z_k = \exp(-bt_k)$ when the following three data are available: $\{(t_k, z_k)\} = \{(0, 1),\ (25, 0.94),\ (45, 0.86)\}$. In order to find the model parameter, plot the sum of squared errors as a function of model parameter b and find the minimum point in the plot. Using the optimum parameter, plot the degradation model in the range of $t = [0, 45]$ overlapped data. Assume that weights are constant.

P4.2 From true model of battery degradation $z_k = \exp(-0.003t_k)$, generate exact degradation data between cycles $t = [0, 45]$ at every five cycles and add random noise from a normal distribution $N(0, \sigma^2)$. By changing the standard deviation of noise by $\sigma = \{0.0,\ 0.01,\ 0.02,\ 0.05\}$, compare the identified model parameter as well as the remaining useful life, especially for their confidence and prediction intervals. Use a threshold of 0.7.

P4.3 Sometimes nonlinear model can be transformed into a linear one. For example, the battery degradation model in Eq. 4.2 can be transformed into a linear model by taking logarithms: $\log z_k(t_k, b) = -bt_k$, where b is a model parameter. Perform a linear regression using **`[LS]`** to find the model parameter b and find the distribution of remaining useful life after taking logarithms of data in Table 4.1. Compare these results with the results from nonlinear least squares **`[NLS]`** in Sect. 4.2.1.

P4.4 When a degradation model is given as $z(t) = a_0^2 - a_1^2 t$, it is possible to estimate the two parameters using nonlinear least squares (treat a_0 and a_1 as unknown parameters) or linear least squares (treat a_0^2 and a_1^2 as unknown parameters). Identify the two model parameters using linear and nonlinear regressions and compare their results. Use the following degradation data:$\{(t_k, z_k)\} = \{(0, 1),\ (2, 0.91),\ (3, 0.78), (4, 0.71)\}$.

P4.5 Markov chain Monte Carlo (MCMC) sampling can be used to generate sample from nonstandard PDF. Consider a PDF, $p(x) \propto 0.3exp(-0.2x^2) + 0.7\exp(-0.2(x-10)^2)$. Use a normal distribution with the standard deviation of 10 as a proposal distribution. Generate 10000 samples and plot the PDF from samples with the exact PDF.

P4.6 Repeat P4.5 with 100 and 500 samples and compare the sample PDFs. Also repeat the same problem with the standard deviation of the proposal distribution of 1 and 100.

P4.7 Generate the PDF of $U(0, 20)$ using Markov chain Monte Carlo (MCMC) sampling. Use the proposal PDF of $U(x-1, x+1)$. Plot the sample PDF with 1000, 10,000 and 100,000 samples.

P4.8 Generate 3 samples of PDF $p(x) = 2x,\ x \in [0, 1]$ manually using MCMC, starting at $x = 0.1$. Use the proposal distribution of $U(x - 0.25, x + 0.25)$. Use MATLAB program to generate 1000 samples and plot the sample PDF.

P4.9 Markov chain Monte Carlo (MCMC) sampling can be extended to multiple variables. In the case of two variables, the following steps are followed to generate a sample of x_1 and x_2. At a current iteration (i), two sample values from (i − 1) iteration are availble. (1) First draw $x_1^{(i)}$ conditional on given $x_2^{(i-1)}$; i.e., $p(x_1^{(i)}|x_2^{(i-1)})$. (2) Next draw $x_2^{(i)}$ conditional on new $x_1^{(i)}$; i.e., $p(x_2^{(i)}|x_1^{(i)})$. Using this method, generate 500 samples of bivariate normal distribution

$$\begin{pmatrix}\theta_1\\ \theta_2\end{pmatrix}\Bigg|\mathbf{y}\sim N\left(\begin{pmatrix}y_1\\ y_2\end{pmatrix},\begin{pmatrix}1 & \rho\\ \rho & 1\end{pmatrix}\right),$$

where $\mathbf{y}=\{0,\ 0\}^{\mathrm{T}}$ and the correlation coefficient $\rho = 0.8$. Hint: Conditional distribution can be given as

$$\theta_1|\theta_2,\mathbf{y}\sim N(y_1+\rho(\theta_2-y_2),1-\rho^2)$$
$$\theta_2|\theta_1,\mathbf{y}\sim N(y_2+\rho(\theta_1-y_1),1-\rho^2).$$

P4.10 The joint PDF of two random variables is given as $p(x,y)=4xy,\ x\in[0,1],\ \ y\in[0,1]$. Explain how to generate n_s samples of $(x,\ y)$.

P4.11 Proceed the update process up to $k = 6$ in Example 4.3. Discuss the results.

P4.12 Perform the battery prognostics with PF, but without the resampling process. In this case, the posterior distribution is expressed with the initial samples and updated weights. Compare the results with ones in Figs. 4.10 and 4.11.

References

Abouel-seoud SA, Elmorsy MS, Dyab ES (2011) Robust prognostics concept for gearbox with artificially induced gear crack utilizing acoustic emission. Energy Environ Res 1(1):81–93

An D, Choi JH, Schmitz TL et al (2011) In-situ monitoring and prediction of progressive joint wear using Bayesian statistics. Wear 270(11–12):828–838

An D, Choi JH, Kim NH (2013) Prognostics 101: a tutorial for particle filter-based prognostics algorithm using Matlab. Reliab Eng Syst Saf 115:161–169

Andrieu C, Freitas DN, Doucet A et al (2003) An introduction to MCMC for machine learning. Mach Learn 50(1):5–43

Casella G, Robert CP, Wells MT (2004) Generalized accept-reject sampling schemes. Lecture Notes-Monograph Series, vol 45. Institute of Mathematical Statistics, Beachwood, pp 342–347

Choi JH, An D, Gang J et al (2010) Bayesian approach for parameter estimation in the structural analysis and prognosis. In: Paper presented at the annual conference of the prognostics and health management society, Portland, Oregon, 13–16 October 2010

Coppe A, Haftka RT, Kim NH et al (2009) Reducing uncertainty in damage growth properties by structural health monitoring. In: Paper presented at the annual conference of the prognostics and health management society, San Diego, California, USA, 27 September–1 October 2009

Coppe A, Pais MJ, Haftka RT et al (2012) Remarks on using a simple crack growth model in predicting remaining useful life. J Aircr 49:1965–1973

Daigle M, Goebel K (2011) Multiple damage progression paths in model-based prognostics. In: Paper presented at IEEE aerospace conference, Big Sky, Montana, USA, 5–12 March 2011
Doucet A, Freitas DN, Gordon NJ (2001) Sequential Monte Carlo methods in practice. Springer, New York
Gavin HP (2016) The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems. Available via Duke University. http://people.duke.edu/~hpgavin/ce281/lm.pdf. Accessed 28 May 2016
Gelman A, Carlin JB, Stern HS et al (eds) (2004) Bayesian data analysis. Chapman & Hall, New York
Glynn PW, Iglehart DL (1989) Importance sampling for stochastic simulations. Manag Sci 35 (11):1367–1392
Goebel KB, Saha A, Saxena JR et al (2008) Prognostics in battery health management. IEEE Instrumen Meas Mag 11(4):33–40
Gu J, Azarian MH, Pecht MG (2008) Failure prognostics of multilayer ceramic capacitors in temperature-humidity-bias conditions. In: Paper presented at the international conference on prognostics and health management, Denver, Colorado, USA, 6–9 October 2008
Guan X, Liu Y, Saxena A et al (2009) Entropy-based probabilistic fatigue damage prognosis and algorithmic performance comparison. Paper presented at the annual conference of the prognostics and health management society, San Diego, California, USA, 27 September – 1 October 2009
He W, Williard N, Osterman M et al (2011) Prognostics of lithium-ion batteries using extended Kalman filtering. In: Paper presented at IMAPS advanced technology workshop on high reliability microelectronics for military applications, Linthicum, Maryland, USA, 17–19 May 2011
Julier SJ, Uhlmann JK (2004) Unscented filtering and nonlinear estimation. Proc IEEE 92(3): 401–422
Kalman RE (1960) A new approach to linear filtering and prediction problems. Trans ASME J Basic Eng 82:35–45
Kim S, Park JS (2011) Sequential Monte Carlo filters for abruptly changing state estimation. Probab Eng Mech 26:194–201
Ling Y, Mahadevan S (2013) Quantitative model validation techniques: New insights. Reliab Eng Syst Saf 111:217–231
Oden JT, Prudencio EE, Bauman PT (2013) Virtual model validation of complex multiscale systems: applications to nonlinear elastostatics. Comput Methods Appl Mech Eng 266: 162–184
Orchard ME, Vachtsevanos GJ (2007) A particle filtering approach for on-line failure prognosis in a planetary carrier plate. Int J Fuzzy Logic Intell Syst 7(4):221–227
Qiu H, Lee J, Linc J et al (2003) Robust Performance degradation assessment methods for enhanced rolling element bearing prognostics. Adv Eng Inform 17:127–140
Rebba R, Mahadevan S, Huang S (2006) Validation and error estimation of computational models. Reliab Eng Syst Saf 91:1390–1397
Ristic B, Arulampalam S, Gordon N (2004) Beyond the Kalman filter: particle filters for tracking applications. IEEE Aerosp Electron Syst Mag 19(7):37–38
Sargent RG (2013) Verification and validation of simulation models. J Simul 7:12–24
Zhang B, Khawaja T, Patrick R et al (2009) Application of blind deconvolution denoising in failure prognosis. IEEE Trans Instrum Meas 58(2):303–310
Zio E, Peloni G (2011) Particle filtering prognostic estimation of the remaining useful life of nonlinear components. Reliab Eng Syst Saf 96(3):403–409

Chapter 5
Data-Driven Prognostics

5.1 Introduction to Data-Driven Prognostics

The physics-based prognostics approaches in Chap. 4 is a powerful tool for predicting the future behavior of damage degradation with a relatively small number of observed data. However, they are limited to the case when a physical model that describes the damage degradation behavior is available. In addition, the measured data must be directly related to the physical model. In the case of crack growth prognostics, for example, the measured data (crack size) is the same as model prediction from the Paris model. Therefore, the physics-based prognostics may anticipate challenges when predicting failure of a complex system without well-defined physical model to describe degradation and when direct measurement of damage is impossible.

In the case of bearing failure prognostics, for example, common damage is caused by a spall phenomenon where the rolling elements or race surface is damaged due to surface cracks. An initial small surface crack quickly propagates into other regions of surface and gradually increases vibration of the shaft and system. The excessive vibration eventually leads to the system failure. For structural health monitoring purpose, however, these surface cracks cannot be measured directly while the bearing is rotating. Therefore, accelerometers are often installed in the stationary parts of the system to monitor the level of vibration. That is, the damage degradation is measured indirectly through the level of vibration. In addition, there is no clear physical relationship between the degradation level and vibration level. In such a case, physics-based prognostics may not be applicable. Instead, based on experience, if the level of vibration of a particular bearing system increases beyond a certain level, engineers learned that the system is going to fail. Therefore, without having a physical model to describe degradation, it is possible to determine when the system needs maintenance. This is the basic concept of data-driven prognostics. Of course, in order to make a reliable prediction, it requires many failure data of similar systems.

N.-H. Kim et al., *Prognostics and Health Management of Engineering Systems*,
DOI 10.1007/978-3-319-44742-1_5

Data-driven approaches in prognostics use information from observed data to identify the patterns of degradation progress and predict the future state without using a physical model. Even if a physical model is not used, data-driven approaches still use some sort of mathematical model that works only for a particular system under monitoring. Similar to physics-based prognostics, the data-driven approaches can be considered as an extrapolation (prediction) method based on mathematical functions (degradation progress).

In general, several sets of additional degradation data up to the end of life as well as the data from the current system are required to identify the degradation characteristics, which is called training data. Even though degradation data are also required to identify model parameters in physics-based approaches, which are not called training data. Training data are used to make mathematical models learn the degradation behavior without a physical model, which is a main difference from physics-based approaches. It is clear that physics-based approaches are more accurate than data-driven ones in prediction results with the same data since they have more information, i.e., physical models and loading conditions. Nevertheless, data-driven approaches are practical since physical degradation models are rare in practice.

The process of the least squares-based data-driven approach was introduced in Sect. 2.2.3, and Fig. 5.1 illustrates an overall process of data-driven approaches. A mathematical model output (z) is a function of input variables (x) and parameters (θ) associated with the mathematical model. First, the mathematical model that represents the relation between input variables and degradation state as the output should be determined, which can be based on polynomial/sigmoid functions or correlation functions. Different data-driven algorithms are distinguished from how to express the relation between inputs and output. Once the mathematical model is determined, parameters associated with the mathematical model are identified by combining with training data via an optimization process. The training data are obtained under various usage conditions as shown in gray squared markers from similar systems (let us call them training sets) or at previous times under a given usage condition as shown in black dotted markers from the current system (let us call them prediction set) in Fig. 2.1. Usually, many sets of training data are required to prevent overfitting (see Sect. 2.2.3). The future degradation state is predicted based on the identified parameters and the mathematical model. Since the parameters are usually obtained as deterministic values, prediction uncertainties are separately included to the deterministic degradation prediction based on individual

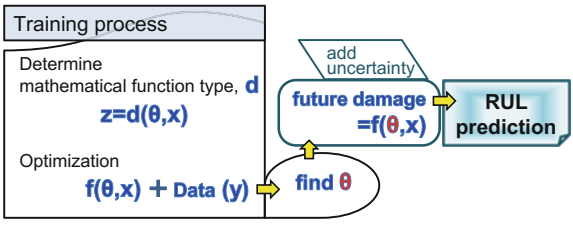


Fig. 5.1 Illustration of data-driven prognostics

algorithms. Lastly, Based on identified parameters and uncertainty, RUL is predicted as the same manner with physics-based approaches.

There are a whole variety of algorithms in data-driven approaches since any extrapolation method can be used for the data-driven prognostics. In general, data-driven approaches are divided into two categories: (1) the artificial intelligence approaches that include neural network (NN) (Chakraborty et al. 1992; Ahmadzadeh and Lundberg 2013; Li et al. 2013) and fuzzy logic (Zio and Maio 2010; Silva et al. 2014); and (2) the statistical approaches that include the Gaussian process (GP) regression (Mackay 1998; Seeger 2004), relevance/support vector machine (Tipping 2001; Benkedjouh et al. 2015), least squares regression (Bretscher 1995; Coppe et al. 2011), the gamma process (Pandey and Noortwijk 2004), the Wiener processes (Si et al. 2013), hidden Markov model (Liu et al. 2012a), etc. Among these algorithms, NN as an artificial intelligence method has been used in wide fields and applications, such as time series prediction, classification/pattern recognition, and robotics/control. It is also the most common algorithm used for prognostics, while the preference for the others might depend on researchers. A comprehensive review about various data-driven algorithms can be found a reference by Si et al. (2011).

Among aforementioned algorithms, GP and NN will be discussed in this chapter. These two algorithms are popular for prognostics. The algorithms will be explained with the same battery example as in Chap. 4, and their practical use will be discussed in Sect. 5.4.

5.2 Gaussian Process (GP) Regression

5.2.1 Surrogate Model and Extrapolation

When many measured data are available at different times of health monitoring, a simple way is to fit the data using a mathematical function. For example, the least squares method in Chap. 2 fits data with a polynomial function by finding coefficients that minimizes the error between data and predictions from the model. In general, fitting data with a mathematical function is referred to as a surrogate model. Since the surrogate model is approximation of physical phenomena, accuracy is uttermost important criterion. If more data are available and if the surrogate model is flexible enough to follow all data, then an accurate surrogate model can be obtained. Once a surrogate model is fitted with given data, evaluating function values at different input values are extremely cheap. The main purpose of surrogate modeling is to reduce the computational time for numerous calculation of the quantity of interest at different input values.

Surrogate modeling is popular in various engineering fields, such as engineering design and reliability assessment. In design applications, the cost and constraint functions are modeled as a function of multiple design variables. Then, during

optimization, design variables are repeatedly changed to find an optimum design, which requires numerous evaluations of functions. Also, reliability analysis requires generating numerous samples of input random variables and estimates the probability that the function output cannot satisfy a requirement.

In prognostics, however, the surrogate model is used for a different purpose. First, time is often used as an input variable because the main purpose of prognostics is to predict the remaining useful life of a system in service. Therefore, measured data are given as a pair of (time, degradation). Second, the main purpose of surrogate is not reducing computational time, but accurate prediction. Even if many predictions are performed in order to estimate the statistical distribution of the remaining useful life, still accurate prediction of the remaining useful life is the main objective of surrogate in prognostics. Lastly, surrogate models in other applications usually predict the quantity of interest within the range of input data (interpolation[1]). However, the usage of surrogate model in prognostics is to predict the degradation level in future time (extrapolation). If degradation levels are measured up to the current time, they are used to fit a surrogate model to predict the degradation level in future time. It is noted that a surrogate behaves quite different between interpolation and extrapolation region. A surrogate model with a good fit in interpolation region can miserably fail in predicting in the extrapolation region.

As an example, Fig. 5.2 shows an approximation of function $y(x) = x + 0.5\sin(5x)$. First, 21 equally spaced samples are generated from the exact function expression in the range of [1, 9]. And then, the MATLAB function `newrb` is used to fit 21 samples with a radial basis network surrogate model. The exact function and surrogate predictions are plotted in the range of [0, 10] as shown in Fig. 5.2. The figure shows that the surrogate predictions are extremely accurate in the interpolation region [1, 9]—in fact the root-mean-square error in this range is zero, while it miserably fails at the extrapolation regions [0, 1] and [9, 10], even if the extrapolation region is very close to the sample locations. The following MATLAB code is used to generate the surrogate:

```
x=linspace(1,9,21); y=x+0.5*sin(5*x);
net=newrb(x,y);
xf=linspace(0,10,101); yf=xf+0.5*sin(5*xf);
ysim=sim(net,xf);
plot(xf,yf); hold on; plot(xf,ysim,'ro')
```

Therefore, it is important to evaluate surrogates for the perspective of extrapolation in prognostics. As shown in Fig. 5.2, the radial basis network may not be a good surrogate as it does not carry the trend of degradation. Also, the polynomial response surface may follow the trend relatively well, but it tends to change the

[1]In high-dimensional problem, though, the conventional surrogate model often ends up extrapolating most of regions. In 10-dimensional space, for example, if 1024 samples (data) are generated randomly, then it is likely more than 90 % of domain is in the extrapolation region.

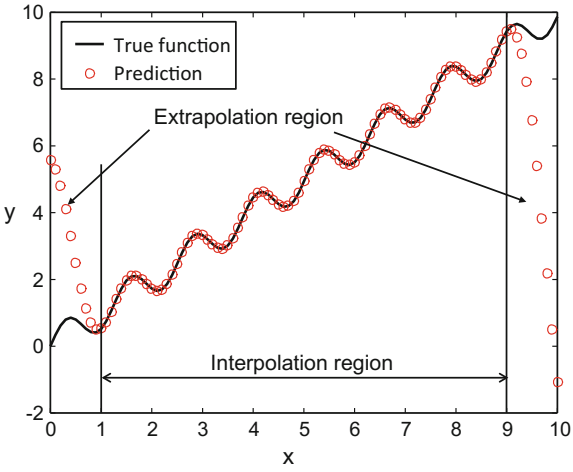


Fig. 5.2 Illustration of surrogate model in interpolation and extrapolation regions

curvature quickly in the extrapolation region. Therefore, based on the numerical tests for different surrogates, the GP regression surrogate model turns out to be most robust in terms of extrapolation. In this section, the GP regression is used for predicting the progress of degradation in future time. Since no physical model is used, this is considered as a data-driven approach.

5.2.2 *Gaussian Process Simulation*

The Gaussian process (GP) regression is one of regression-based methods used for data-driven prognostics, which is a linear regression like the least squares method. The difference between GP and ordinary linear regression is whether the correlation in errors between a regression function and data are considered or not. It is assumed that errors are independent and identically distributed (i.i.d.) in the ordinary linear regression, while they are assumed to be correlated in GP.

Correlation is an important concept in GP, which is directly related to covariance of two random variables. The covariance of two random variables, X and Y, can be defined as

$$\mathrm{cov}(X, Y) = E[(X - \mu_X)(Y - \mu_Y)] = E[XY] - \mu_X \mu_Y, \tag{5.1}$$

where $E[\cdot]$ is the expected value of a random variable, μ_X and μ_Y are mean values of X and Y, respectively. The covariance of a random variable itself is the square of the standard deviation, which is also called variance. Note that the variance of a random variable is the second moment around its mean, or the square of the standard deviation. The covariance of a pair of random variables is the mixed moment of the two variables.

Using covariance, the correlation of two random variables can be defined as

$$R(X,Y) = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y}, \tag{5.2}$$

where σ_X and σ_Y are the standard deviation of X and Y, respectively. Note that $-1 \leq R(X,Y) \leq 1$. The correlation measures their propensity to move together. So a correlation of 1 means that they increase and decrease in perfect synchrony. The most common case is when $X = \alpha Y$ with a positive constant α. Similarly, a correlation of -1 also means perfect synchrony, but in the opposite direction, and in the case of $X = \alpha Y$ with a negative constant α. Finally, a correlation of 0 means that they are not related. The common case is when the random variables are independent, but not all uncorrelated variables are independent. Zero correlation relates to the first moment, but there may be dependence through higher order effects.

Example 5.1 Correlation between function values
The fitting process in GP needs to estimate how fast the correlation between function values decays with distance. This depends on the wavelength of the function; that is, the correlation quickly decays for a function with a short wavelength. In order to check how fast the correlation decays for $\sin(x)$ that has a wave length of 2π, (a) generate 10 random numbers, $x_i, i = 1, \ldots, 10$, between 0 and 10, (b) move x_i by a small amount, $x_i^{\text{near}} = x_i + 0.1$, and a large amount, $x_i^{\text{far}} = x_i + 1.0$, and (c) calculate correlations between $y = \sin(x_i)$ and $y = \sin(x_i^{\text{near}})$ and between $y = \sin(x_i)$ and $y = \sin(x_i^{\text{far}})$.

Solution:
Different random numbers may be generated, but the following MATLAB code can be used to generate random numbers and calculate correlations:

```
x=10*rand(1,10);
xnear=x+0.1; xfar=x+1;
y=sin(x);
ynear=sin(xnear)
yfar=sin(xfar)
rnear=corrcoef(y, ynear)        %rnear=0.9894;
rfar=corrcoef(y, yfar)          %rfar=0.4229
```

The following table shows 10 values of random numbers along with $y = \sin(x_i), y = \sin(x_i^{\text{near}})$ and $y = \sin(x_i^{\text{far}})$.

x	8.147	9.058	1.267	9.134	6.324	0.975	2.785	5.469	9.575	9.649
ynear	0.9237	0.2637	0.9799	0.1899	0.1399	0.8798	0.2538	-0.6551	-0.2477	-0.3185
y	0.9573	0.3587	0.9551	0.2869	0.0404	0.8279	0.3491	-0.7273	-0.1497	-0.2222
yfar	0.2740	-0.5917	0.7654	-0.6511	0.8626	0.9193	-0.5999	0.1846	-0.9129	-0.9405

It is clear that the pattern between y and ynear is similar, but not between y and yfar. That is, y and ynear is strongly correlated, but not for y and yfar. Indeed, the correlation coefficient between y and ynear is 0.9894, while it is 0.4229 between y and yfar. This reflects the change in function values, as illustrated by the pairs of points marked in red in the table. In GP, finding the rate of correlation decay is part of the fitting process. This example shows that with a wavy function the correlation decays to about 0.4 over one sixth of the wavelength, $(1 : 2\pi)$.

In GP, a simulation output, $z(x)$, is composed of a global function output, $\boldsymbol{\xi}(\boldsymbol{x})\boldsymbol{\theta}$, that often takes the form of constant or polynomials and its local departure, $s(\mathbf{x})$:

$$z(\mathbf{x}) = \boldsymbol{\xi}(\mathbf{x})\boldsymbol{\theta} + s(\mathbf{x}), \tag{5.3}$$

where $\mathbf{x}$ is a row vector of input variables, $\boldsymbol{\xi}(\mathbf{x})$ is a $1 \times n_p$ vector of bases associated with $n_p \times 1$ vector of global function parameters/coefficients, $\boldsymbol{\theta}$, and $s(\mathbf{x})$ is a local departure, i.e., an error between the global function and measurement data. For example, when the global function is assumed to be a linear polynomial, $a_0 + a_1 x$ in one-dimensional GP, $\boldsymbol{\xi}(x) = [1,\ x]$ and $\boldsymbol{\theta} = [a_0,\ a_1]^{\mathrm{T}}$. In practice, a simple polynomial function is used for the global function.

One may think that Eq. 5.3 is similar to Eq. 2.3, but they are significantly different in terms of the fundamental assumptions. The polynomial-based least squares method in Eq. 2.3 assumes that the model form $z(\mathbf{x}) = \boldsymbol{\xi}(\mathbf{x})\boldsymbol{\theta}$ is correct, but the data have errors ε_k that are normally distributed as $N(0, \sigma^2)$. These errors are assumed to be statistically independent; that is, the error in one point is not related to the error in other points. On the other hand, GP assumes that the measured data are correct but the model form is uncertain. Since data are correct, GP fits all data. That is, the model prediction and measured data match at data points: $y_k = z(\mathbf{x}_k) = \boldsymbol{\xi}(\mathbf{x}_k)\boldsymbol{\theta} + s(\mathbf{x}_k)$. In prediction points, the uncertainty or error in GP can be described using the local departure, $s(\mathbf{x})$, in Eq. 5.3, which is a realization of the Gaussian stochastic process with zero mean, variance σ^2, and nonzero covariance as

$$s(\mathbf{x}) \sim N\left(0, \sigma^2\right), \tag{5.4}$$

where σ is the standard deviation of data with respect to the global function. The standard deviation is small when the data is dense. The data is considered to be dense, when there are more than enough points to describe the waviness of the function. In order to be dense, the distance from one point to the nearest one should be an order of magnitude less than the wavelength of the function.

The key concept of GP is that the functional form of $z(\mathbf{x})$ is unknown, except for the form of correlation between the value of the function at nearby points. In particular, the correlation depends only on the separation distance between points and decays as they are further apart. The correlation matrix between n_y data can be defined as

$$\mathbf{R} = [R(\mathbf{x}_k, \mathbf{x}_l)],\ k, \quad l = 1, \ldots, n_y, \tag{5.5}$$

where $R(\mathbf{x}_k, \mathbf{x}_l)$ is correlation between two data points, which will be defined using correlation function later. Note that the diagonal components of $\mathbf{R}$ are one because they represent correlation between the same point. The correlation is strong between two points nearby and weak between two points far away. The correlation between a prediction point, $\mathbf{x}$, and data points can be defined in a similar way as

$$\mathbf{r}(\mathbf{x}) = [R(\mathbf{x}_k, \mathbf{x})], \quad k = 1, \ldots, n_y, \tag{5.6}$$

which is a $n_y \times 1$ vector that represents correlation between the data points $\mathbf{x}_k$ and prediction point $\mathbf{x}$, which makes different magnitude in departure at different prediction points.

Because of the correlation in the local departure, GP has an outstanding feature that the simulated output, denoted by the blue-dashed curve in Fig. 5.3, passes through the set of measured data (training data) $[\mathbf{x}_{1:n_y}, \mathbf{y}_{1:n_y}]$. When the prediction point (a new input), x^{new}, is located at a measurement point A, $x_i\left(i = 1, 2, \ldots, n_y\right)$, the magnitude of departure is the same as the difference between measured data, $y_i\left(i = 1, 2, \ldots, n_y\right)$ and the global function output (the global function is constant in the figure); hence, the prediction output $z(x_i)$ becomes measured data, y_i. When the prediction point is different from measured points as in point B, the magnitude of departure is changed based on the correlation between B and the measured points. As a result, if the prediction point is in between measured points, the simulated output smoothly interpolates the measured data. If, however, the prediction point

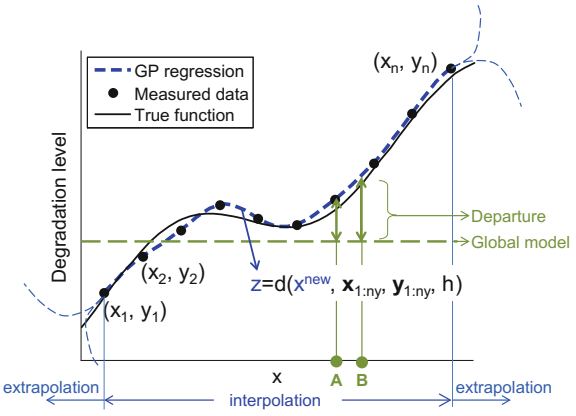


Fig. 5.3 Illustration of Gaussian Process regression

moves away from measured points, which is extrapolation, the influence of this departure is reduced as correlation decreases and the GP becomes closer to the global function. Therefore, the case of extrapolation is not so different from an ordinary linear regression, which follows the global model. For interpolation, correlation between data points is an important factor to characterize GP's attribute, which is determined by selecting an appropriate type of correlation function and estimating its hyperparameters. The hyperparameters in GP control the effect of correlation between two points, which affect the smoothness of the surrogate model. The hyperparameters are usually determined using an optimization algorithm based on the measured data (training data). A typical correlation function and estimating hyperparameters will be discussed later part in this section.

5.2.3 GP Simulation

5.2.3.1 Global Function Parameter and Distribution of Errors

In this section, GP simulation function is derived using measured data points. First, the global function parameters, $\boldsymbol{\theta}$, in Eq. 5.3 and the variance of data with respect to the global function, σ^2, in Eq. 5.4 can be estimated by using maximum likelihood estimation (MLE). Once the global function is determined, the local departure is calculated by minimizing the unbiased mean squares error.

As explained in Eqs. 5.3 and 5.4, the errors between the global function and data are assumed to be normally distributed. At n_y data point, the errors can be defined as

$$\mathbf{e} = \mathbf{y} - \mathbf{X}\boldsymbol{\theta} = \begin{Bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n_y} \end{Bmatrix} - \begin{bmatrix} -\xi(\mathbf{x}_1)- \\ -\xi(\mathbf{x}_2)- \\ \vdots \\ -\boldsymbol{\xi}(\mathbf{x}_{n_y})- \end{bmatrix} \begin{Bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_{n_p} \end{Bmatrix},$$

where $\mathbf{X}$ is the design matrix, the values of basis vector at all data points. It is noted that the above errors depend on the global function parameters, $\boldsymbol{\theta}$.

The assumption in GP is that the error $\mathbf{e}$ follows a Gaussian distribution with zero mean, variance σ^2 and correlation between data points. As explained in Bayesian inference in Chap. 3, the likelihood is the value of probability density of obtaining measured data $\mathbf{y}$ for given parameters, $\boldsymbol{\theta}$ and σ^2. Therefore, the joint probability density function of correlated n_y data points can be used to define the likelihood as

$$f\left(\mathbf{y}|\boldsymbol{\theta}, \sigma^2\right) = \frac{1}{\sqrt{(2\pi)^{n_y}(\sigma^2)^{n_y}|\mathbf{R}|}} \exp\left(-\frac{(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})}{2\sigma^2}\right) \tag{5.7}$$

which is multiplication of n_y Gaussian distributions. This is the likelihood function of $\mathbf{y}$ with given $\boldsymbol{\theta}$ and σ^2 with a the correlation matrix, $\mathbf{R}$. In the above equation, $|\mathbf{R}|$ is the determinant of the correlation matrix.

The parameters, $\boldsymbol{\theta}$ and σ^2, can be found by maximizing the likelihood function. For the purpose of handling an algebraic function instead of exponential function, a logarithm of the likelihood is often used. Since logarithm is a monotonically increasing function, it may change the value of likelihood, but it will not change the parameters' location that yields the maximum value. The logarithmic likelihood is defined as

$$\begin{aligned}\ln\left[f\left(\mathbf{y}|\boldsymbol{\theta},\sigma^2\right)\right] = &-\frac{n_y}{2}\ln(2\pi)-\frac{n_y}{2}\ln(\sigma^2)\\ &-\frac{1}{2}\ln|\mathbf{R}|-\frac{(\mathbf{y}-\mathbf{X}\boldsymbol{\theta})^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{y}-\mathbf{X}\boldsymbol{\theta})}{2\sigma^2}.\end{aligned} \tag{5.8}$$

The maximum location can be found by differentiating Eq. 5.8 with respect to $\boldsymbol{\theta}$ and σ as

$$\begin{aligned}\frac{\partial \ln f}{\partial \boldsymbol{\theta}} &= \frac{\mathbf{X}^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{y}-\mathbf{X}\boldsymbol{\theta})}{\sigma^2}=0,\\ \frac{\partial \ln f}{\partial \sigma^2} &= -\frac{n_y}{2}\frac{1}{\sigma^2}+\frac{(\mathbf{y}-\mathbf{X}\boldsymbol{\theta})^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{y}-\mathbf{X}\boldsymbol{\theta})}{2\sigma^4}=0.\end{aligned} \tag{5.9}$$

Therefore, the estimated parameters that maximize the likelihood function can be obtained by solving the above equation for $\boldsymbol{\theta}$ and σ^2 as

$$\begin{aligned}\hat{\boldsymbol{\theta}} &= \left(\mathbf{X}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{X}\right)^{-1}\left\{\mathbf{X}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{y}\right\}\\ \hat{\sigma}^2 &= \frac{(\mathbf{y}-\mathbf{X}\hat{\theta})^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\theta}})}{n_y-n_p}.\end{aligned} \tag{5.10}$$

Note that the estimated parameter, $\hat{\boldsymbol{\theta}}$, is the same form as Eq. 2.8 except for the inverse of correlation matrix, and the degree of freedom $n_y - n_p$ is used instead of the number of data n_y in $\hat{\sigma}^2$ for the unbiased estimation, which is basically the same with Eq. 2.18 except for the inverse of correlation matrix. It is noted that the estimated parameters depend on the correlation matrix $\mathbf{R}$, which will be discussed later.

In deriving Eq. 5.10, it is assumed that the wavelength of the function is uniform in all regions of input space. In such a case, the covariance matrix that is found using the initial data is constant throughout the input space, which is called stationary covariance. For nonstationary covariance, readers are referred to Xiong et al. (2007).

Polynomial functions are often used for the global function. Kriging surrogate is a different name of Gaussian process because a South African geostatistical engineer Daniel G. Krige developed the surrogate to estimate distance-weighted average

gold grades. A specific term, ordinary Kriging, is provided when a constant function is used for the basis; that is the estimated parameter $\hat{\boldsymbol{\theta}}$ is a scalar. When the main purpose of the surrogate is for estimating function in interpolating region, the choice of global function might not be important as the local departure will model such a way that the prediction passes through data points and prediction between data points are determined based on correlation between data points. However, for the purpose of prognostics, this is not a good choice of the global function because the purpose of GP in prognostics is for extrapolation where the correlation between data points are quickly diminished as the extrapolation distance increases and the prediction goes back to the global function. Knowing that degradation shows a monotonically increasing or decreasing behavior, it would be better to choose bases with such a property. In this book, either linear or quadratic polynomial is suggested, which shows a monotonic behavior when input variable is positive.

5.2.3.2 Local Departure

In the previous subsection, the global function parameters, $\hat{\boldsymbol{\theta}}$, and the variance, $\hat{\sigma}^2$, were estimated using maximum likelihood estimation, whose results are given in Eq. 5.10. Therefore, at this point, the form of global function in Eq. 5.3, is determined. Now then, the local departure term $s(\mathbf{x})$ in Eq. 5.3 is considered, which eventually makes a GP simulation function. From the property that GP simulation passes measured data point, GP simulation can be expressed by a linear combination of measured data and weight functions. Then, the weight function can be found by minimizing the mean squared error (MSE) between outputs from a GP simulation function and the true function. First, the error between GP simulation and true function is defined as

$$\varepsilon(\mathbf{x}) = \hat{z}(\mathbf{x}) - z(\mathbf{x}) = \mathbf{w}(\mathbf{x})^{\mathrm{T}}\mathbf{y} - z(\mathbf{x}), \tag{5.11}$$

where $\hat{z}(\mathbf{x})$ and $z(\mathbf{x})$ are, respectively, the GP simulation function and true function. The GP simulation function can be expressed as the dot product of a $n_y \times 1$ vector of weight functions, $\mathbf{w}(\mathbf{x})$, and a $n_y \times 1$ vector of measurement data, $\mathbf{y}$ (i.e., $\mathbf{w}(\mathbf{x})^{\mathrm{T}}\mathbf{y}$). The goal is to determine the weight function that minimizes the error in Eq. 5.11. Even if the GP simulation function is shown in the form of $\hat{z}(\mathbf{x}) = \mathbf{w}(\mathbf{x})^{\mathrm{T}}\mathbf{y}$, this form will be expressed in terms of the global function and local departure as shown in Eq. 5.3.

From the basic assumption that there is no error in data, the data can be expressed in the form of GP simulation function in Eq. 5.3. That is, $\mathbf{y} = \mathbf{X}\hat{\boldsymbol{\theta}} + \mathbf{s}$, where $\mathbf{s} = \{s(\mathbf{x}_1),\ s(\mathbf{x}_2),\ \ldots,\ s(\mathbf{x}_{n_y})\}^{\mathrm{T}}$ is the vector of local departures at data points. Note that $s(\mathbf{x})$ is a function, while $\mathbf{s}$ is a vector of $s(\mathbf{x})$ at all sampling points. Similarly, the true function can also be expressed as $z(\mathbf{x}) = \boldsymbol{\xi}(\mathbf{x})\hat{\boldsymbol{\theta}} + s(\mathbf{x})$. Therefore, Eq. 5.11 can be expressed as

$$\begin{aligned}\varepsilon(\mathbf{x}) &= \mathbf{w}(\mathbf{x})^{\mathrm{T}}\{\mathbf{X}\hat{\boldsymbol{\theta}} + \mathbf{s}\} - \left(\boldsymbol{\xi}(\mathbf{x})\hat{\boldsymbol{\theta}} + s(\mathbf{x})\right) \\ &= \left(\mathbf{w}(\mathbf{x})^{\mathrm{T}}\mathbf{X} - \boldsymbol{\xi}(\mathbf{x})\right)\hat{\boldsymbol{\theta}} + \mathbf{w}(\mathbf{x})^{\mathrm{T}}\mathbf{s} - s(\mathbf{x}).\end{aligned} \tag{5.12}$$

In Eq. 5.12, $\left(\mathbf{w}(\mathbf{x})^{\mathrm{T}}\mathbf{X} - \boldsymbol{\xi}(\mathbf{x})\right)\hat{\boldsymbol{\theta}}$ and $\mathbf{w}(\mathbf{x})^{\mathrm{T}}\mathbf{s} - s(\mathbf{x})$ are the global and the departure error terms, respectively. To keep the global function unbiased, a constraint for the weight function is demanded so that the global error term is zero. Since the global function parameters $\hat{\boldsymbol{\theta}}$ cannot be all zero, the following equation is obtained as the constraint:

$$\mathbf{w}(\mathbf{x})^{\mathrm{T}}\mathbf{X} - \boldsymbol{\xi}(\mathbf{x}) = \mathbf{0}. \tag{5.13}$$

Now the MSE of Eq. 5.12 becomes[2]

$$\mathrm{MSE} = \mathrm{E}\left[\varepsilon(\mathbf{x})^2\right] = \mathrm{E}\left[\left(\mathbf{w}^{\mathrm{T}}\mathbf{s} - s\right)^2\right] = \mathrm{E}\left[\mathbf{w}^{\mathrm{T}}\mathbf{s}\mathbf{s}^{\mathrm{T}}\mathbf{w} - 2\mathbf{w}^{\mathrm{T}}\mathbf{s}s + s^2\right]. \tag{5.14}$$

In the above equation, E[•] is the expectation operator. By using the definition of the second central moment (the variance) and the covariance,[3] the MSE can be obtained as

$$\mathrm{MSE} = \sigma^2\left(\mathbf{w}^{\mathrm{T}}\mathbf{R}\mathbf{w} - 2\mathbf{w}^{\mathrm{T}}\mathbf{r} + 1\right), \tag{5.15}$$

which represents uncertainty in simulated GP outputs, and thus, it becomes the variance of the GP simulation.

The weight function can be found by minimizing the MSE in Eq. 5.15, while satisfying the unbiased constraint in Eq. 5.13. The method of Lagrange multipliers that is an optimization method with equality constraints is employed to find $\mathbf{w}$ by minimizing the MSE in Eq. 5.15 with the constraint in Eq. 5.13. The Lagrange function is defined by introducing a Lagrange multiplier, $\boldsymbol{\lambda}$ that is a $1 \times n_p$ vector in this case:

$$L(\mathbf{w}, \boldsymbol{\lambda}) = \sigma^2\left(\mathbf{w}^{\mathrm{T}}\mathbf{R}\mathbf{w} - 2\mathbf{w}^{\mathrm{T}}\mathbf{r} + 1\right) - \boldsymbol{\lambda}\left(\mathbf{X}^{\mathrm{T}}\mathbf{w} - \boldsymbol{\xi}^{\mathrm{T}}\right). \tag{5.16}$$

The minimum can be found by differentiating Eq. 5.16 with respect to $\mathbf{w}$ and $\boldsymbol{\lambda}$ as

$$\frac{\partial L(\mathbf{w}, \boldsymbol{\lambda})}{\partial \mathbf{w}} = 2\sigma^2(\mathbf{R}\mathbf{w} - \mathbf{r}) - \mathbf{X}\boldsymbol{\lambda}^{\mathrm{T}} = \mathbf{0}. \tag{5.17}$$

[2]In this equation, even though $\mathbf{w}$ and s are function of inputs, it is not indicated to simplify the equation.

[3]The covariance between different inputs are as follows: $\mathrm{Cov}[s(\mathbf{x}), s(\mathbf{x})] = \sigma^2$, $\mathrm{Cov}[s(\mathbf{x}_k), s(\mathbf{x})] = \sigma^2\mathbf{r}$, $\mathrm{Cov}[s(\mathbf{x}_k), s(\mathbf{x}_l)] = \sigma^2\mathbf{R}$, $k, l = 1, \ldots, n_y$.

$$\frac{\partial L(\mathbf{w},\boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} = \mathbf{X}^{\mathrm{T}}\mathbf{w} - \boldsymbol{\xi}^{\mathrm{T}} = \mathbf{0}. \tag{5.18}$$

By solving Eq. 5.17 for $\mathbf{w}$, the following equation is obtained:

$$\mathbf{w} = \mathbf{R}^{-1}\mathbf{r} + \mathbf{R}^{-1}\mathbf{X}\frac{\boldsymbol{\lambda}^{\mathrm{T}}}{2\sigma^2}. \tag{5.19}$$

To obtain $\boldsymbol{\lambda}^{\mathrm{T}}/2\sigma^2$, Eq. 5.19 is substituted into Eq. 5.18 and solve for $\boldsymbol{\lambda}^{\mathrm{T}}/2\sigma^2$ as

$$\frac{\boldsymbol{\lambda}^{\mathrm{T}}}{2\sigma^2} = (\mathbf{X}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{X})^{-1}\{\boldsymbol{\xi}^{\mathrm{T}} - \mathbf{X}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{r}\}. \tag{5.20}$$

Therefore, the weight vector, $\mathbf{w}$ can be obtained with Eqs. 5.19 and 5.20, which is used for the GP simulation in Eq. 5.11 as

$$\begin{aligned}\hat{z}(\mathbf{x}) &= \mathbf{w}(\mathbf{x})^{\mathrm{T}}\mathbf{y} \\ &= \left(\mathbf{R}^{-1}\mathbf{r} + \mathbf{R}^{-1}\mathbf{X}\frac{\boldsymbol{\lambda}^{\mathrm{T}}}{2\sigma^2}\right)^{\mathrm{T}}\mathbf{y} \\ &= \mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{y} + \frac{\boldsymbol{\lambda}}{2\sigma^2}\mathbf{X}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{y} \\ &= \mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{y} + \left(\boldsymbol{\xi} - \mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{X}\right)\underline{\left(\mathbf{X}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{X}\right)^{-1}\left\{\mathbf{X}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{y}\right\}}.\end{aligned} \tag{5.21}$$

Note that the symmetric property of the correlation matrix is used. In Eq. 5.21, the underlined term is the estimated parameter, $\hat{\boldsymbol{\theta}}$ in Eq. 5.10. Finally, the equation is obtained for GP simulation as

$$\hat{z}(\mathbf{x}) = \boldsymbol{\xi}(\mathbf{x})\hat{\boldsymbol{\theta}} + \mathbf{r}(\mathbf{x})^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\theta}}), \tag{5.22}$$

where $\mathbf{y}$ is a $n_y \times 1$ vector of measured data, $\mathbf{X}$ is a $n_y \times n_p$ design matrix, $\mathbf{R}$ is a $n_y \times n_y$ correlation matrix between the measurement points, and $\hat{\boldsymbol{\theta}}$ is a $n_p \times 1$ vector of global function parameters/coefficients, which are determined based on the measurement data, while $\mathbf{r}(\mathbf{x})$ is a $n_y \times 1$ correlation vector between the prediction point and the measurement points. In Eq. 5.22, the first term, $\boldsymbol{\xi}(\mathbf{x})\hat{\boldsymbol{\theta}}$ represents a global function term and the second term is the local departure. To calculate the departure at a new input, the errors between the data and the global function outputs, $\left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\theta}}\right)$ is weighted and summed based on the correlation term, $\mathbf{r}(\mathbf{x})^{\mathrm{T}}\mathbf{R}^{-1}$.

If Eq. 5.22 is written in the form of $\hat{z}(\mathbf{x}) = \boldsymbol{\xi}(\mathbf{x})\hat{\boldsymbol{\theta}} + \mathbf{r}(\mathbf{x})^{\mathrm{T}}\boldsymbol{\beta}$, only $\boldsymbol{\xi}(\mathbf{x})$ and $\mathbf{r}(\mathbf{x})$ are a function of prediction point $\mathbf{x}$; all other terms are fixed for a given set of sampling data. Therefore, GP is similar to linear regression, which is a linear combination of unknown coefficients multiplied by known basis functions. However, there are two important differences. First, the basis function, $\mathbf{r}(\mathbf{x})$,

is unknown, in that the correlation matrix, **R**, needs to be estimated from the data (in the next subsection). Second, the coefficients are not found by minimizing the root-mean-square-error (RMSE) at the data points; in fact, RMSE is zero because the prediction interpolates the data exactly.

5.2.3.3 Correlation Function and Hyperparameters

There are different types of correlation functions available, including radial basis (or squared exponential), rational quadratic, neural network, Matern, periodic, constant, linear functions, and a combination of these (Rasmussen and Williams 2006). Even though the quality of GP simulation depends on the correlation function, one-parameter radial basis function that is common and simple is employed in this book. The radial basis function depends on the distance between inputs, i.e., the Euclidean norm as

$$R(\mathbf{x}, \mathbf{x}^*) = \exp\left(-(x_d/h)^2\right), \quad x_d = \|\mathbf{x} - \mathbf{x}^*\|, \tag{5.23}$$

where h is a hyperparameter that is a scale parameter in this function, which is used to control the smoothness of the function.

A small h means a fast decay of the correlation with distance. The correlation should decay to about 0.4 at one-sixth of the wavelength, ℓ (see Example 5.1). Since $e^{-1} = 0.37$ is close to 0.4, the hyperparameter must satisfy $(\ell/6h)^2 \approx 1$, which yields $h \approx \ell/6$. That is, if a function has a wavelength ℓ, then the hyperparameter should be close to $h \approx \ell/6$. For example, for $z(x) = \sin(x)$, $h \approx 1$, while for $z(x) = \sin(5x)$, $h \approx 0.2$.

In general, the hyperparameters are determined via an optimization algorithm by maximizing a log-likelihood (or minimizing a negative log-likelihood) function corresponding to the error between global function outputs and data, which is given in Eq. 5.8. The following equation is obtained by omitting the first and the fourth[4] terms in Eq. 5.8 since they are constant with respect to the correlation matrix, i.e., the hyperparameters (Toal et al. 2008). Therefore, the hyperparameter, h, can be found by maximizing the following equation:

$$h = \arg\max\left[-\frac{n_y}{2}\ln(\hat{\sigma}^2) - \frac{1}{2}\ln|\mathbf{R}|\right]. \tag{5.24}$$

The optimum value of h can be obtained by maximizing Eq. 5.24 or minimizing equivalent function of Eq. 5.24 as

[4]Even if the fourth term include the inverse of correlation matrix, it will be canceled out when σ^2 is substituted with $\hat{\sigma}^2$ in Eq. 5.10.

$$h = \arg\min\left[\ln\left(\hat{\sigma}^{2\times(n_y-n_p)} \times |\mathbf{R}|\right)\right]. \tag{5.25}$$

In this equation, the degree of freedom, $n_y - n_p$ is used instead of the number of data n_y for the purpose of unbiased estimation in variance.

Maximizing a log-likelihood function is a challenging optimization problem because the likelihood function normally varies slowly in a wide range of argument. Instead of using a maximum likelihood approach, cross-validation can also be used. The cross-validation method leaves out one of the points, fits the rest with the parameters, and calculates the error at the left-out point. This process is repeated for every point to produce a total error measure. Then, the set of parameters that yields the lowest cross-validation error is chosen. The optimization problem to solve for maximum likelihood or minimum cross-validation error in GP regression is often ill-conditioned, which can lead to either a poor fit or poor estimate of the prediction variance. Poor estimate of the prediction variance can be checked by comparing it to the cross-validation error. A poor fit can often be found in GP regression when the curvature changes significantly near data points.

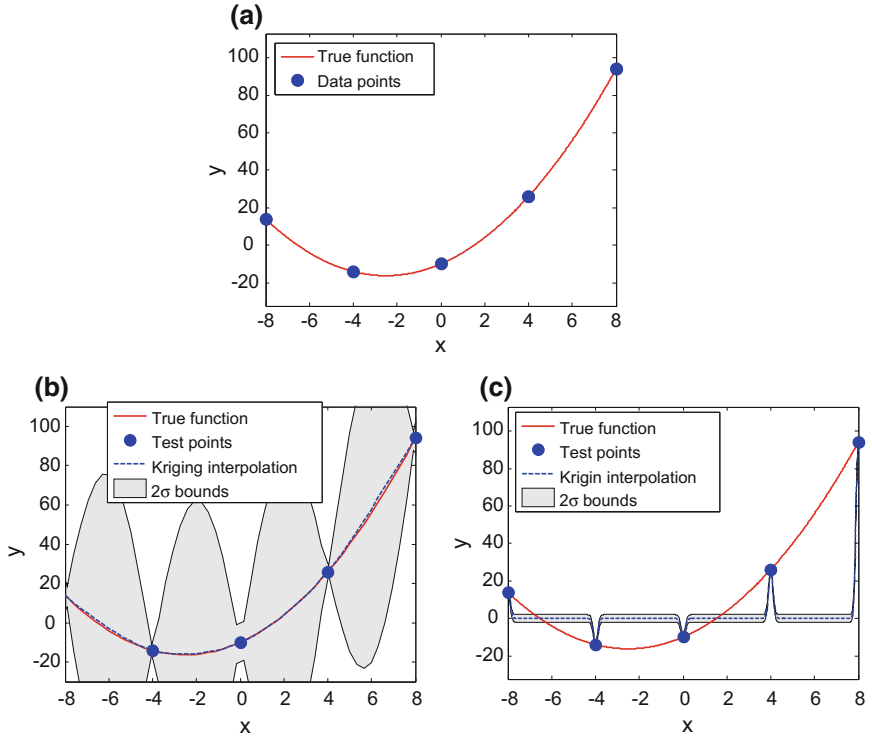


Fig. 5.4 Effect of hyperparameters in Gaussian process regression. **a** True function and sample locations, **b** GP regression with too large hyperparameter, **c** GP regression with too small hyperparameter

For example, Fig. 5.4a shows a simple quadratic polynomial $y(x) = x^2 + 5x - 10$ where the curvature changes relatively quickly. Nine data points are generated from the polynomial (only a portion of data are shown), and GP regression is used to fit the data with a constant global function $\xi(x) = 0$. Figure 5.4b shows GP regression with unreasonably large hyperparameter, h, while Fig. 5.4c shows the case with too small hyperparameter. When h is too large, the fit is good, but the estimated prediction variance is too large. On the other hand, when h is too small, the fit is poor and the prediction variance fails to cover the true function. Therefore, it is always recommended to visualize by plotting the GP regression with its uncertainty.

Example 5.2 Deterministic GP simulation
Using GP simulation, perform the following steps with the first five data given in Table 4.1.

(a) Calculate the global function parameter for a constant form and the variance of data with respect to the global function. Use the radial basis correlation function given in Eq. 5.23 and assume $h = 5.2$.
(b) Obtain an optimum value of the scale parameter, h.
(c) Calculate the GP simulation results at $t = 10, 14$ after obtaining the global and departure terms separately.

Solution:
This problem can be solved by using the given equations in this section. Before solving the problems, the data and design matrix need to be defined as

$$\mathbf{y} = [1 \quad 0.99 \quad 0.99 \quad 0.94 \quad 0.95]^{\mathrm{T}}$$
$$\mathbf{x}(=\mathbf{t}) = [0 \quad 5 \quad 10 \quad 15 \quad 20]^{\mathrm{T}}$$
$$\mathbf{X} = [1 \quad 1 \quad 1 \quad 1 \quad 1]^{\mathrm{T}}$$

Note that the global function is defined as a constant in the problem, which means the global function outputs will be the same for any input and there will be one parameter. That is, $\boldsymbol{\xi}(\mathbf{x}) = \{1\}$ and $\boldsymbol{\theta} = \{\theta\}$. In MATLAB, these terms are defined as

```
y=[1 0.99 0.99 0.94 0.95]';                % measurement data
x=[0 5 10 15 20]';                        % input variable, t
X=ones(5,1);                                 % design matrix
ny=length(y); np=size(X,2);
```

(a) The equations for solving this problem are given in Eq. 5.10, and all terms were defined previously except for the correlation matrix, $\mathbf{R}$ that is obtained from Eq. 5.23 as $\mathbf{R} = [R(x_k, x_l)]$, $k, l = 1, \ldots, n_y$. This following MATLAB commands calculate the correlation matrix:

```
h=5.2;
for k=1:ny; for l=1:ny;
    R(k,l)=exp(-(norm(x(k,:)-x(l,:))/h)^2);
end; end;
```

and the results become

$$\mathbf{R} = \begin{bmatrix} 1 & 0.3967 & 0.0248 & 0.0002 & 0 \\ 0.3967 & 1 & 0.3967 & 0.0248 & 0.0002 \\ 0.0248 & 0.3967 & 1 & 0.3967 & 0.0248 \\ 0.0002 & 0.0248 & 0.3967 & 1 & 0.3967 \\ 0 & 0.0002 & 0.0248 & 0.3967 & 1 \end{bmatrix}.$$

The diagonal elements should be one because they represent the correlation between themselves, e.g., $\mathbf{R}(1,1)$ and $\mathbf{R}(2,2)$ are correlations between x_1 and x_1, and between x_2 and x_2, respectively. On the other hand, the off-diagonal elements are determined based on the distance between two different inputs; e.g., $\mathbf{R}(1,2)$ and $\mathbf{R}(2,1)$ are calculated from x_1 and x_2, which makes the correlation matrix symmetric.

Now the global function parameter and the variance of data with respect to the global function can be calculated by using Eq. 5.10 as

$$\hat{\theta} = \left(\mathbf{X}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{X}\right)^{-1}\left\{\mathbf{X}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{y}\right\} = 3.0989^{-1} \times 3.0226 = 0.9754$$

$$\hat{\sigma}^2 = \frac{(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\theta}})^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\theta}})}{n_y - n_p} = 7.28 \times 10^{-4}, \hat{\sigma} = 0.0270$$

```
Rinv=inv(R);
thetaH=(X'*Rinv*X)\(X'* Rinv*y);
sigmaH=sqrt(1/(ny-np)*((y-X*thetaH)'*Rinv*(y-X*thetaH)));
```

(b) During the process in part (a), the value of $h = 5.2$ is given. In fact, this is the optimum results, which can be obtained by minimizing Eq. 5.25. Note that the two variables, $\hat{\sigma}$ and $\mathbf{R}$ depend on h as shown in Eq. 5.10 and Eq. 5.23, and how to calculate these variables are shown in part (a). Since this is a one-parameter optimization problem, it can be solved

approximately by plotting the objective function (Eq. 5.25) at different values of h. By calculating Eq. 5.25 for $h > 0$, Fig. E5.1 is obtained. In the figure, the dot marker represents the optimum point, from which the optimum value of the scale parameter is approximately obtained as $h_{\text{opt}} = 5.2$. The following MATLAB code can be used to calculate Fig. E5.1.

```
h=zeros(20,1); Obj=zeros(20,1);
for i=1:20
  h(i)=0.5*i;
  for k=1:ny; for l=1:ny;
    R(k,l)=exp(-(norm(x(k,:)-x(l,:))/h(i))^2);
  end; end;
  Rinv=inv(R);
  thetaH=(X'*Rinv*X)\(X'* Rinv*y);
  sigmaH=sqrt(1/(ny-np)*((y-X*thetaH)'*Rinv*(y-X*thetaH)));
  Obj(i)=log(sigmaH^(2*(ny-np))*det(R));
end
plot(h,Obj,'linewidth',2); grid on;
```

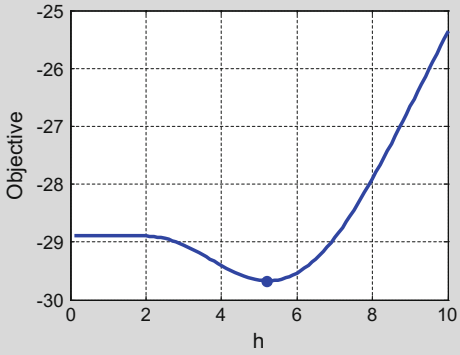


Fig. E5.1 Objective function with respect to h

(c) Since the global function is a constant, the design vector, $\boldsymbol{\xi} = [1]$ and there is a single function parameter. Therefore, the global term $\boldsymbol{\xi}\hat{\boldsymbol{\theta}}$ in Eq. 5.22 becomes $\xi\hat{\theta} = 1 \times 0.9754$. For the departure term, $\mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\theta}}\right)$ in Eq. 5.22, the correlation vector between $t = 10, 14$ (t is input variable, x) and the measurement data points should be calculated as (R is given in Eq. 5.23):

$$\mathbf{r} = \{R(x_k, x)\}, \quad k = 1, \ldots, n_y$$

and thus at $t = 10$;

$$\mathbf{r} = [\,0.0248 \quad 0.3967 \quad 1 \quad 0.3967 \quad 0.0248\,]^{\mathrm{T}}$$
$$\mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\theta}}\right) = 0.0146$$

Note that since $t = 10$ is the same as one of the measurement points (the third point), the correlation vector, $\mathbf{r}$, is identical to the third column of correlation matrix, $\mathbf{R}$.

In the same manner, at $t = 14$

$$\mathbf{r} = [\,0.0007 \quad 0.05 \quad 0.5534 \quad 0.9637 \quad 0.2641\,]^{T}$$
$$\mathbf{r}^{T}\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\theta}}\right) = -0.0272$$

The correlation vectors can be calculated with the following MATLAB code:

```
xNew=10; %or xNew=14
for k=1:ny; r(k,1)=exp(-(norm(x(k,:)-xNew)/h)^2); end;
gpDepar=r'*Rinv*(y-X*thetaH);
```

Therefore, the GP simulation results in Eq. 5.22 become as follows:

at $t = 10$; $\hat{z}(10) = 0.9754 + 0.0146 = 0.99$
at $t = 14$; $\hat{z}(14) = 0.9754 - 0.0272 = 0.9482$

The results are shown as blue star markers in Fig. E5.2a, which are departures from the global function. By repeating the aforementioned process for $t \in [-5, 25]$, the red solid curve in Fig. E5.2a is obtained. Note that the GP simulation results pass through the measurement data and make a smooth curve.

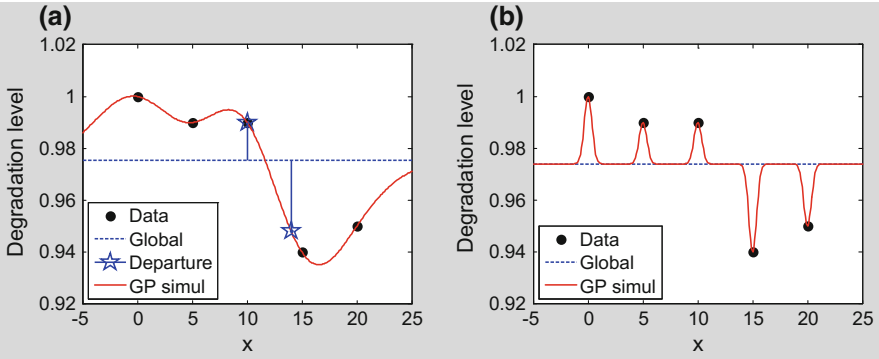


Fig. E5.2 GP regression with five data in Table 4.1. **a** $h = 5.2$. **b** $h = 0.5$

Figure E5.2b shows the GP simulation results when an inappropriate scale parameter is used. Too small value, $h = 0.5$ in this case, reduces the correlation between inputs. When the correlation is reduced, GP regression approaches a linear regression, and thus follows the global function. The both ends and in the middle of data points are overlapped with the global function. But still, the GP regression pass through the data points since the distance between the same points is zero, which makes the correlation value one from Eq. 5.23. It is given as an exercise problem (see P5.4) to consider the opposite case, i.e., when the scale parameter is very large.

Note that since the exponential form of correlation decays away from the data points, when the prediction point is far from all data points, it will converge to the global function $\boldsymbol{\xi}(\mathbf{x})\hat{\boldsymbol{\theta}}$. Therefore, GP is not good for long-term prognostics unless there is enough training data from other similar systems.

5.2.3.4 Uncertainty

An important attribute for GP compared to other surrogate models is that it allows to estimate uncertainty in prediction points. In the viewpoint of uncertainty, Eq. 5.22 provides a mean prediction of GP regression. The variance of the GP simulation is introduced in the form of MSE in Eq. 5.15. The uncertainty in GP simulation follows a Gaussian distribution with mean at the deterministic model in Eq. 5.22 and the variance given in Eq. 5.15 as

$$Z(\mathbf{x}) \sim N\left(\boldsymbol{\xi}\hat{\boldsymbol{\theta}} + \mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\theta}}\right),\quad \sigma^2\left(\mathbf{w}^{\mathrm{T}}\mathbf{R}\mathbf{w} - 2\mathbf{w}^{\mathrm{T}}\mathbf{r} + 1\right)\right). \tag{5.26}$$

Note that the true variance of the GP simulation is unknown but estimated from the limited number of data. When the variance is estimated from a small number of

samples, the distribution of mean follows Student's t-distribution. Therefore, Eq. 5.26 is modified with the Student's t-distribution with a degree of freedom, $n_y - n_p$ as

$$Z(\mathbf{x}) \sim \boldsymbol{\xi}\hat{\boldsymbol{\theta}} + \mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\theta}}\right) + t_{n_y - n_p} \cdot \hat{\sigma}\sqrt{\mathbf{w}^{\mathrm{T}}\mathbf{R}\mathbf{w} - 2\mathbf{w}^{\mathrm{T}}\mathbf{r} + 1}, \tag{5.27}$$

which represents the distribution of simulation outputs that is caused by the uncertainty in function parameters. Due to the assumption in GP, the above uncertainty does not include any noise in data, and thus, the prediction interval is not considered in GP simulation. Equation 5.27 can be used to calculate the confidence interval of simulation outputs. The confidence interval, however, can be considered as a prediction interval at the same time because the simulation results pass through measurement points. Therefore, the prediction uncertainty for prognostics is performed based on Eq. 5.27, whose interval is called a prediction interval.

It is interesting to note that the prediction variance is zero at data points in GP. In order to show this, the prediction point is chosen at a data point; $\mathbf{x} = \mathbf{x}_k$. From the expression of Lagrange multiplier in Eq. 5.20, it can be shown that

$$\boldsymbol{\xi}(\mathbf{x}_k)^{\mathrm{T}} - \mathbf{X}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}_k) = \mathbf{0}.$$

This is because $\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}_k)$ becomes a unit vector whose kth component is one and all other components are zero, and kth column of design matrix $\mathbf{X}$ is nothing but $\boldsymbol{\xi}(\mathbf{x}_k)$. With zero Lagrange multiplier at kth data point, the interpolating weight $\mathbf{w}$ in Eq. 5.19 becomes a unit vector whose kth component is one and all other components are zero. Finally, the MSE at the kth data point becomes

$$\mathrm{MSE}(\mathbf{x}_k) = \sigma^2\left(\mathbf{w}^{\mathrm{T}}\mathbf{R}\mathbf{w} - 2\mathbf{w}^{\mathrm{T}}\mathbf{r} + 1\right) = 0.$$

In the above equation, $\mathbf{w}^{\mathrm{T}}\mathbf{R}\mathbf{w} = 1$ because the diagonal component of correlation matrix is one, and $\mathbf{w}^{\mathrm{T}}\mathbf{r} = 1$ because the kth component of correlation vector is also one. Therefore, at data points, the prediction variance becomes zero and there is no uncertainty at data points.

Example 5.3 Uncertainty in GP simulation
Calculate 90 % confidence interval at $t = 10$, 14 for the problem given in Example 5.2.

Solution:
To calculate the prediction interval, Eq. 5.27 can be used, and the mean of the GP regression is obtained in Example 5.2. For the uncertainty part, the weight

function, $\mathbf{w}$, is given in Eq. 5.19, and then the standard deviation of the prediction uncertainty, $\hat{\sigma}_z = \hat{\sigma}\sqrt{\mathbf{w}^T\mathbf{R}\mathbf{w} - 2\mathbf{w}^T\mathbf{r} + 1}$ can be calculated as

```
xi=1;
w=Rinv*r+Rinv*X*((X'*Rinv*X)\(xi'-X'*Rinv*r));
zSigmaH=sigmaH*sqrt(w'*R*w-2*w'*r+1);
```

Since there is no uncertainty at measurement points, $\hat{\sigma}_z$ should be zero at $t = 10$, and the 5th and 95th percentiles are the same as the mean value, 0.99 (given in Example 5.2). On the other hand, $\hat{\sigma}_z$ is calculated as 0.0041 at $t = 14$, and the confidence interval can be obtained with two ways based on the inverse CDF of Student's t or the random samples from the t-distribution. The following is MATLAB commands to calculate the confidence interval.

```
% using the inverse calculation
gpMean=0.9482; % from Example 5.2
PI=[ gpMean + tinv(0.05,ny-np)*zSigmaH, ...
          gpMean + tinv(0.95,ny-np)*zSigmaH]
% using the random samples
ns=5e3;
tDist=trnd(ny-np,1,ns);
yHat=gpMean+tDist*zSigmaH;
PI=prctile(yHat,[5 95])
```

The results are listed in Table E5.1, and the red-dashed curves in Fig. E5.3 are obtained by repeating the process for $t \in [-5, 25]$. Note that the confidence intervals are zero at data points, and they increases at the extrapolation region.

Table E5.1 Prediction intervals in GP regression

x_{new}	5 percentile	95 percentile	90 % P.I.
$t = 10$	0.99	0.99	0
$t = 14$	0.9394	0.9570	0.0176

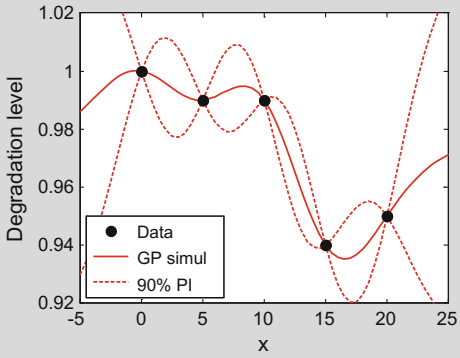


Fig. E5.3 Uncertainty in GP regression with five data in Table 4.1

5.2.4 *MATLAB Implementation of Battery Prognostics Using Gaussian Process*

In this section, a MATLAB code **[GP]** is introduced that can utilize GP simulation to predict the degradation of damage and the remaining useful life. The usage of the code is explained with the same battery prognostics introduced in Chap. 4. The code is divided into three parts: (1) problem definition for user-specific applications, (2) prognostics using GP, and (3) postprocessing for displaying results, which is the same as physics-based algorithms in Chap. 4.

5.2.4.1 Problem Definition (Lines 5–15, 17, 27–31)

The MATLAB code for GP simulation is given in the code **[GP]**. Most variables in problem definition section were explained in Sect. 4.2.1, and a few ones are newly introduced for data-driven algorithms. In lines 10 and 11, `nT` is the number of training sets including prediction set, `nt` is a `nT` × 1 vector of the number of training data in each data set. It is possible that data-driven approaches can take multiple sets of data in GP regression. For example, let us assume that two batteries have been measured for degradations until they fail, and we want to predict the remaining useful life of the third battery. Let us also assume that 20 degradation

data are available for each of battery 1 and 2 until they fail and the current battery has been measured up to 10 degradation data. In such a scenario, `nT = 3` because it includes all batteries that have degradation data, including the current one. Since the number of data are different for different batteries, specifies the number of available data at each set. When multiple data sets are used, the last data set is always considered as the prediction set; i.e., the current system of interest. For multiple training data sets, readers are referred to Sect. 5.4.

Instead of true parameters in physics-based methods, true degradation level of the prediction set can be given in line 13, same array size with `time`. If the true degradation is unknown, `degraTrue` can be an empty array. The variable `signiLevel` is used to calculate confidence intervals using `ns` number of samples. For the battery problem, a code for the first parameter definition part (lines 5–15) is given as

```
WorkName='Battery_GP';
DegraUnit='C/1 Capacity';
TimeUnit='Cycles';
time=[0:5:200]';
y=[1.00 0.99 0.99 0.94 0.95 0.94 0.91 0.91 0.87 0.86]';
nT=1;
nt=[10]';
thres=0.7;
degraTrue=exp(-0.003.*time);
signiLevel=5;
ns=5e3;
```

Since there is no physical model available, the given information, i.e., measurement data and/or usage conditions[5] need to be properly utilized. Prediction quality depends on the utilization of given information, which is related to the definition of input and output matrices. Therefore, it is one of important tasks to define input and output matrices in the data-driven approaches. Because of its importance, the definition of input and output matrices are in the problem definition section in lines 17 and 27–31. In this section, measurement time and the measurement data at the time are, respectively, used in inputs (`xTrain`) and outputs (`yTrain`) as a simple example. In this case, the line 17 can be modified with

[5]The data-driven approach with usage conditions is not handled in this chapter, but its effect on the prediction results will be discussed in Chaps. 6 and 7.

```
y0=(y-min(y))/(max(y)-min(y));
time0=(time-min(time))/(max(time)-min(time));
xTrain=time0(1:length(y)); yTrain=y0;
```

Note that the first two lines in the above code normalize the input and output data between zero and one, which is required to find the scale parameter properly. New input values are required to predict degradation, which is defined in line 28 as

```
xNew=time0(ny-1+k);
```

Again the normalize time, `time0` is used instead of `time`.

5.2.4.2 Prognosis Using GP

The process for GP is based on the derived equations in Sect. 5.2.3, and simple example was introduced in Example 5.2 to obtain the mean prediction using GP model and Example 5.3 to quantify the uncertainty in the model. Since the prediction results can largely depend on the scale parameter and the global function, they are used as the input variables in the function (line 1). For the battery problem, the following code is used to run **`[GP]`**.

```
rul=GP(1,0.05);
```

The order of global function, `funcOrd` defines global function with the `GLOBAL` in lines 42–48. There are three polynomial functions that is determined by `functOrd = 0, 1, or 2`, but this part can be modified with any polynomial function. Since `functOrd = 1` in this example, the first-order polynomial function in line 45 is used, which makes the design matrix from training data (line 21) and new inputs (line 73).

The initial value of scale parameter, `h0` is set as `0.05` to find the optimum value of `hH` (lines 23–24) that is estimated based on an optimization function `fmincon` to find minimum value of the scale parameter between −2 and 2. The maximum magnitude of the scale parameter, `hMax = 2` (line 23) make a correlation of the maximum distance become 0.7788.[6] This is to prevent ridiculously large value of

[6]The maximum distance, $x_{d,\max}$ in Eq. 5.23 is one since the training data are normalized between zero and one. Therefore, when $h = 2\,(\text{or } -2)$, the correlation is calculated as $\exp\left(-(1/2)^2\right) = 0.7788$.

the scale parameter (when it is large, the correlation of the maximum distance close to one), and `hMax` can be adjusted. The objective function is given in Eq. 5.25 (line 53), which is minimized in the `FUNC` (lines 50–54). The required parameters to calculate the objective function are the correlation matrix, `R` and the magnitude of error between global function and data, `sigma` that calculated in the `RTHESIG` (lines 56–61). The correlation matrix, `R` (line 58) is calculated in the `CORREL` (lines 63–68) based on the correlation function given in Eq. 5.23 (line 66). Then, the function parameters, `theta` and the error, `sigma` (lines 59, 60) can be calculated by using Eq. 5.10. These three parameters depend on the scale parameter, and they are finally calculated (line 25) after finding the optimum value of `hH` (line 24).

Now that GP simulation results in terms of mean, `zMean`, and the magnitude of simulation error, `zSig` (line 29), can be calculated via `GPSIM` (lines 70–78). The `xi` (line 73) and `r` (line 74) are, respectively, a design vector and a correlation vector, which depend on the new inputs, `x`. The mean of GP simulation, `zMean` (line 75), the weight vector, `w` (line 76) and the magnitude of simulation error, `zSig` (line 77) are given in the Eqs. 5.15, 5.19, 5.20 and 5.21.

Once the mean and the error are obtained, the prediction uncertainty (lines 33–36) is quantified based on Eq. 5.27 and the random sampling method explained in Example 5.3. Since normalized data are used, the final results of degradation prediction are rescaled to the original values (line 35). As mentioned before, the error in simulation results is considered the prediction uncertainty (line 36).

5.2.4.3 Postprocessing

The postprocessing is the same as that used for the physics-based approaches in Chap. 4, **`[POST]`**, but `thetaHat` in the code for physics-based approaches is replaced with a blank here since the function parameters are deterministically estimated and the comparison with the true values is not possible. As the results, the degradation and RUL are predicted as shown in Fig. 5.5, and the percentiles of RUL distribution at 45 cycles are obtained as

```
Percentiles of RUL at 45 Cycles
 5th: 39.5361,  50th (median): 49.9718,  95th: 58.7455
```

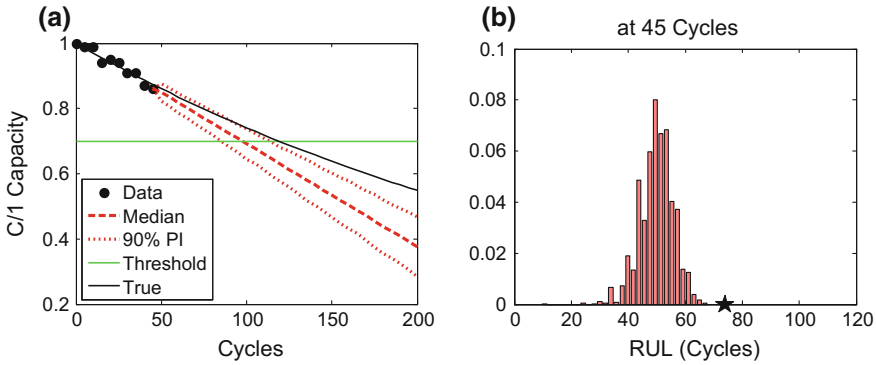


Fig. 5.5 GP prediction results for the battery problem

[GP]: MATLAB Code for Gaussian Process

```
function rul=GP(funcOrd,h0)
clear global; global DegraUnit ...
TimeUnit time y thres signiLevel ns ny nt xTrain yTrain X dof
%=== PROBLEM DEFINITION 1 (Required Variables) ==============
WorkName=' ';              % work results are saved by WorkName
DegraUnit=' ';                                % degradation unit
TimeUnit=' ';                % time unit (Cycles, Weeks, etc.)
time=[ ]';     %[cv]: time at both measurement and prediction
y=[ ]';                                  %[ny x 1]: measured data
nT= ;         % num. of training set including the current one
nt=[ ]';%[nT x 1]: num. of training data in each training set
thres= ;                             % threshold (critical value)
degraTrue=[ ]';               %[cv]: true values of degradation
signiLevel= ;           % significance level for C.I. and P.I.
ns= ;                               % number of particles/samples
%=== PROBLEM DEFINITION 2 (Training Matrix) =================
xTrain=[ ]; yTrain=[ ];
%============================================================
% % % PROGNOSIS using GP
ny=length(y);
X=GLOBAL(xTrain,funcOrd);          %% Determine Hyperparameter
dof=size(yTrain,1)-size(X,2);
hMax=2; hBound=hMax*ones(1,length(h0));
hH=fmincon(@FUNC,h0,[],[],[],[],-hBound,hBound);
[R,thetaH,sigmaH]=RTHESIG(hH);          %% R, Theta, and Sigma
%=== PROBLEM DEFINITION 2 (Input Matrix for Prediction ======
for k=1:length(time(ny:end));        %% Degradation Prediction
 xNew=[ ];
 [zMean(k,:),zSig(k,:)]= ...
                       GPSIM(xNew,funcOrd,hH,R,thetaH,sigmaH);
```

```
end;
%=============================================================
tDist=trnd(dof,size(zMean,1),ns);   %% Prediction Uncertainty
zHat0=repmat(zMean,1,ns)+tDist.*repmat(zSig,1,ns);
zHat=zHat0*(max(y)-min(y))+min(y);
degraPredi=zHat;
% % % POST-PROCESSING
rul=POST([],degraPredi,degraTrue);       %% RUL & Result Disp
Name=[WorkName ' at ' num2str(time(ny)) '.mat']; save(Name);
end
% % % GLOBAL FUNCTION
function X=GLOBAL(x0,funcOrd)
 nx=size(x0,1);
 if funcOrd==0; X=ones(nx,1);
 elseif funcOrd==1; X=[ones(nx,1) x0];
 elseif funcOrd==2; X=[ones(nx,1) x0 x0.^2];
 end
end
% % % OBJECTIVE FUNCTION TO FIND H
function objec=FUNC(h)
 global dof
 [R,~,sigma]=RTHESIG(h);
 objec=log(sigma^(2*dof)*det(R));
end
% % % R, THETA, and SIGMA
function [R,theta,sigma]=RTHESIG(h)
 global xTrain yTrain X dof
 R=CORREL(xTrain,h); Rinv=R^-1;
 theta=(X'*Rinv*X)\(X'*Rinv*yTrain);
 sigma=sqrt(1/dof*((yTrain-X*theta)'*Rinv*(yTrain-X*theta)));
end
% % % CORRELATION FUNCTION
function R=CORREL(x0,h)
 global xTrain
 for k=1:size(xTrain,1); for l=1:size(x0,1);
  R(k,l)=exp(-(norm(xTrain(k,:)-x0(l,:))/h)^2);
 end; end;
end
% % % GP SIMULATION at NEW INPUTS
function [zMean,zSig]=GPSIM(x,funcOrd,h,R,theta,sigma)
 global yTrain X
 Rinv=R^-1;
 xi=GLOBAL(x,funcOrd);
 r=CORREL(x,h);
 zMean=xi*theta+r'*Rinv*(yTrain-X*theta);
 w=Rinv*r+Rinv*X*((X'*Rinv*X)\(xi'-X'*Rinv*r));
 zSig=sigma*sqrt(w'*R*w-2*w'*r+1);
end
```

5.3 Neural Network (NN)

The earliest artificial neural network was introduced in 1943 by Warren McCulloch, a neurophysiologist, and a mathematician Walter Pitts. In 1949, Donald Hebb published a book, "*The Organization of Behavior*," where he formulated the classical Hebbian rule, which is proved to be the basis of almost all neural learning procedures. Perceptron, a simple mathematical representation of neuron, was developed by Frank Rosenblatt in 1958. Unfortunately, this concept is proved to be limited and can only solve linearly separable problems in Marvin Minsky and Seymour Papert's book Perceptrons, resulting research on neural network to an abrupt halt.

After long silence in this area, John Hopfield invented associative neural network in 1982, which is now more commonly known as Hopfield Network. He applied this technology into useful device and persuaded many scientists to join this area. At the same time, backpropagation algorithm, a generalization of Widrow-Hoff learning algorithm, was invented by Rumelhart, Hinton, and Williams. These events caused the renaissance of research in neural network. Today, neural network is used to solve wide variety of problems such as forecasting, classification and statistical pattern recognition. In this section, the neural network is introduced as a tool for prognostics.

The Neural Network (NN) algorithm is a representative data-driven prognostics method, in which a network model learns a way to produce desired outputs, such as the level of degradation or lifespan, by reacting to given inputs, such as time and usage conditions. The relationship between inputs and outputs depends on how the NN architecture is constructed and which functions (called the transfer or activation function) are associated with the architecture. Once the network model learns enough about the relationship between inputs and outputs, it can be used for the purpose of prognosis.

There are many different types of NN, such as the feedforward NN (Svozil et al. 1997) that conveys the information in one direction, the radial basis function network (Orr 1996) that usually uses the Gaussian radial basis function, the recurrent NN (Bodén 2001) that has local feedback connections between the input and outputs, and so forth. There are other neural networks, such as fuzzy-neural (Liu and Li 2004), wavelet (He et al. 2004), associative-memory (Bicciato et al. 2001), modular (Happel and Murre 1994), and hybrid (Zhang and Yuen 2013; Rovithakis et al. 2004) neural networks. Among these types, the feedforward NN is the most common one, and it is introduced in this book.

5.3.1 *Feedforward Neural Network Model*

5.3.1.1 Concept of Feedforward Neural Network

The basic processing units in neural network is called neurons or nodes. For different functions of nodes, they can be grouped into different layers such as input layer, output layer and hidden layer. The first layer is input layer which is used for receiving input data from outside, while the last layer is output layer used to send processed data out of the neural network. Hidden layer, existing between input and output layer and as the name suggested, is invisible from outside and do not have any interaction with outside.

The most popular architecture of NN is feedforward neural network (FFNN), which means that information only moves in forward direction from input layer, through hidden layer and to output layer. There is no feedback or flow within layers. A two-layer feedforward neural network (FFNN) is illustrated in Fig. 5.6, which is a basic form of it. In the figure, circles represent nodes, which can belong to three different layers: input, hidden, and output layers. The input nodes and output nodes represent the input and output variables, respectively. The hidden nodes connect the input and output variables; i.e., they are fed input information, and then forward the information to the output node. This is why it is called a feedforward neural network. The number of nodes in the hidden layer should be determined to properly express the mechanism between the inputs and output.

There are two different ways of defining layers in NN. First, each set of nodes in the same column can be called a layer, i.e., input, hidden, and output layers. Also, a

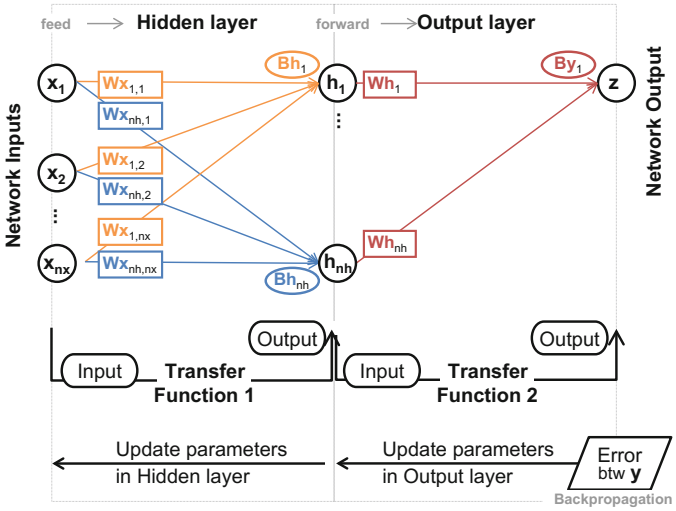


Fig. 5.6 Illustration of a typical network model: feedforward-backpropagation neural network

layer can be understood as an interface between different columns of nodes. These two definitions are basically the same, but we follow the second one as shown in Fig. 5.6. That is, the hidden layer is located between input and hidden nodes, while the output layer is between hidden and output nodes.

There are two types of parameters to be estimated in Fig. 5.6; the rectangles and the ellipses are called weights and biases, respectively. Weight is an interconnection parameter between different nodes, which represents the contribution for each input data when calculating the weighted sum of all inputs. It can be either positive or negative, where positive weight is considered as excitation and negative one as inhibition. For each node in hidden nodes and output node, there is a bias attached on it. Bias is usually used as threshold value added after weighted sum of inputs from previous layer is calculated.

We define the name of weights and biases as follows. Weights in the hidden layer are input weights (w_x) since they are associated with the input nodes. On the other hand, weights in the output layer are hidden weights (w_h). Also, there are hidden biases (b_h) and an output bias (b_z), which are biases for function inputs in each layer. These parameters are integrated into some functions called transfer or activation functions to determine the relation between input and output variables. The input variables with input weights and hidden biases become the input for a transfer function in the hidden layer, and whose outputs are assigned into the hidden nodes. Again, the hidden nodes (outputs in the hidden layer) with the hidden weights and output bias are used as the input for a transfer function in the output layer, and whose outputs are the final network output. The mechanism of FFNN will be explained in the following section in more detail.

The training process is equivalent to finding the optimal parameters, weights and biases, such that the network model accurately represents the relationship between inputs and outputs. Once the network model is trained enough, the model is functionalized using transfer functions, weight and bias parameters. For this process, the backpropagation is typically employed, which propagates the error between training data and the network outputs backward. That is, updating-related parameters (gradient or Jacobian) for hidden weights and output bias (output layer) are updated first, and then the parameters for input weights and hidden biases (hidden layer) are updated next based on the updated parameters in the output layer. This is basically an optimization process to find the best weights and biases to minimize the mean-square-error between network predictions and training data. Since the FFNN is usually trained with backpropagation method, it is often called the backpropagation neural network. The process of the backpropagation will be explained in the following section.

5.3.1.2 Feedforward Mechanism

The basic mathematical relationship in the feedforward network model is a linear combination of input nodes with weights and biases as an input to the transfer function in the next layer. The weights are usually multiplied with the value at their

associated nodes, and then the biases are added to the sum of the multiplication results to be an input to the transfer function. The feedforward mechanism can be explained as

$$\mathbf{h} = d_h(\mathbf{W}_x\mathbf{x} + \mathbf{b}_h), \quad \mathbf{z} = d_z(\mathbf{W}_h\mathbf{h} + \mathbf{b}_z), \tag{5.28}$$

where $\mathbf{W}_x$ is an $n_h \times n_x$ (n_h and n_x are the number of hidden nodes and input variables, respectively) input weight matrix associated with a $n_x \times 1$ vector of input variables, $\mathbf{x}$, and $\mathbf{b}_h$ is a $n_h \times 1$ hidden bias vector, which is the input for a transfer function, d_h in the hidden layer, whose function outputs are in $\mathbf{h}$, a $n_h \times 1$ vector of hidden layer outputs. The $\mathbf{h}$ is combined with $\mathbf{W}_h$ that is a $n_z \times n_h$ (n_z is the number of output variables, and $n_z = 1$ in Fig. 5.6) hidden weight matrix to be the input for the transfer function, d_z in the output layer with a $n_z \times 1$ output bias vector, $\mathbf{b}_z$. Then, $\mathbf{z}$ is a $n_z \times 1$ vector of the output variables.

For the purpose of prognostics, the number of output variables, n_z, is usually considered as one for the degradation level. During the training step, the vector of n_y training data, $\mathbf{y}$, is available. These data measure the same degradation at n_y different times. For a given weight and bias parameters, the vector of outputs, $\mathbf{z}$, can be calculated at the same n_y times. Then, the optimization algorithm finds the weights and bias parameters that minimize the difference between the outputs, $\mathbf{z}$, and the training data, $\mathbf{y}$. The training process utilizes n_y training data in a batch process. Therefore, Eq. 5.28 that is for one set of data can be rewritten by considering the number of measurement/training data, n_y as:

$$\mathbf{H} = d_h(\mathbf{W}_x\mathbf{X} + \mathbf{B}_h), \quad \mathbf{z} = d_z(\mathbf{w}_h\mathbf{H} + \mathbf{b}_z), \tag{5.29}$$

where $\mathbf{W}_x$ is a $n_h \times n_x$ input weight matrix, $\mathbf{X}$ is a $n_x \times n_y$ matrix of training input data, $\mathbf{B}_h = [\mathbf{b}_h \quad \mathbf{b}_h \quad \ldots \quad \mathbf{b}_h]_{n_h \times n_y}$ is a hidden bias matrix, $\mathbf{H}$ is a $n_h \times n_y$ matrix of hidden layer outputs, $\mathbf{w}_h$ is a $1 \times n_h$ hidden weight row vector, $\mathbf{b}_z = [b_z \quad b_z \quad \ldots \quad b_z]_{1 \times n_y}$ is a $1 \times n_y$ output bias vector, and $\mathbf{z}$ is a $1 \times n_y$ vector of network simulation outputs.

Example 5.4 Two-layer FFNN with linear transfer functions
Write the expression of the output of a two-layer FFNN with two input and one hidden nodes as a function of input variables, weights, and biases. Use a pure linear function, $z = x$ for the transfer functions in both layers.

Solution:
This problem can be modeled as shown in Fig. E5.4, which can be formularized with $n_x = 2$, $n_h = 1$ and $n_z = 1$ in Eq. 5.28 (or $n_y = 1$ in Eq. 5.29).

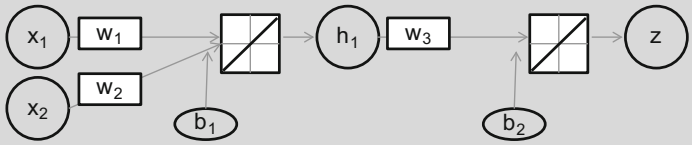


Fig. E5.4 Illustration of two-layer FFNN with two input and one hidden nodes

The input, weight and bias matrices/vectors are given as:

$$\mathbf{W}_x = [\,w_1 \quad w_2\,],\ \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad \mathbf{b}_h = b_1,\ \mathbf{W}_h = w_3,\ \mathbf{b}_z = b_2$$

The input for the transfer function in the hidden layer is the same as its output since the function is a pure linear, therefore the following equation can be obtained:

$$\mathbf{h} = \mathbf{W}_x\mathbf{x} + \mathbf{b}_h = w_1x_1 + w_2x_2 + b_1.$$

Then, the input and output in the output layer are obtained as

$$\begin{aligned} \mathbf{z} &= \mathbf{W}_h\mathbf{h} + \mathbf{b}_z \\ &= w_3(w_1x_1 + w_2x_2 + b_1) + b_2 \\ &= (w_1w_3x_1 + w_2w_3x_2) + w_3b_1 + b_2. \end{aligned}$$

Note that the two-layer FFNN with linear transfer functions can be reduced to a one-layer network model with a pure linear function, as

$$\mathbf{z} = w_1^*x_1 + w_2^*x_2 + b_1^*, w_1w_3 = w_1^*, w_2w_3 = w_2^*, w_3b_1 + b_2 = b_1^*$$

The above example can be generalized for pure linear transfer function. In such a case, Eq. 5.29 can be rewritten as

$$\mathbf{z} = \mathbf{w}_h(\mathbf{W}_x\mathbf{X} + \mathbf{B}_h) + \mathbf{b}_z = (\mathbf{w}_h\mathbf{W}_x)\mathbf{X} + (\mathbf{w}_h\mathbf{B}_h + \mathbf{b}_z), \tag{5.30}$$

which is a one-layer network model with $(\mathbf{w}_h\mathbf{W}_x)_{1\times n_x}$ being input weights and $(\mathbf{w}_h\mathbf{B}_h + \mathbf{b}_z)_{1\times n_y}$ being the output bias. In general, for any multilayer network model with pure linear transfer functions, it is always possible to convert it to a single-layer network model. When the transfer function is nonlinear, such as tangent sigmoid function, such a simplification cannot be possible.

In the case of prognostics, structural health monitoring system measures damage degradation in a sequence of times. Let us consider that $n_y + 3$ degradation data are available as $\mathbf{y} = \{y_1, y_2, \ldots, y_{n_y}, y_{n_y+1}, y_{n_y+2}, y_{n_y+3}\}$ that are measured at times $\mathbf{t} = \{t_1, t_2, \ldots, t_{n_y}, t_{n_y+1}, t_{n_y+2}, t_{n_y+3}\}$. It will be shown later that $n_y + 2$ data are

required to generate n_y sets of training data. The simplest network model would have the time as an input and the level of degradation as an output. However, this may not be a good idea because different systems under different loading conditions may have different degradation for the same time.

From the fact that damage degradations are given in a sequence of times, it might be a good idea to estimate the degradation at time t_k using degradation data at previous times. For example, in order to predict degradation z_k at time t_k, the previous three measured degradations, $\mathbf{x} = \{y_{k-1}, y_{k-2}, y_{k-3}\}^{\mathrm{T}}$ can be used as an input ($n_x = 3$). That is, the network model predicts the degradation level at next time step using measured degradations at previous times. As a sequence, z_{k+1} can be predicted using $\mathbf{x} = \{y_k, y_{k-1}, y_{k-2}\}^{\mathrm{T}}$. This sequence can be continued until z_{n_y+3}. In such a case, the input matrix in Eq. 5.29 can be defined as

$$\mathbf{X} = \begin{bmatrix} y_1 & y_2 & \cdots & y_{n_y} \\ y_2 & y_3 & \cdots & y_{n_y+1} \\ y_3 & y_4 & \cdots & y_{n_y+2} \end{bmatrix}_{n_x \times n_y.}$$

Even if a total $n_y + 3$ measured data are available, only n_y columns can be used because the first n_x data cannot be used as a prediction. For the training purpose, the difference between n_y number of predictions, $\mathbf{z} = \{z_4, z_5, \ldots, z_{n_y+3}\}$ and the same number of measured data $\mathbf{y} = \{y_4, y_5, \ldots, y_{n_y+3}\}$ are defined as an error, which needs to be minimized by changing weights and biases. Therefore, in prognostics, $n_y + n_x$ number of sequential data are required to have n_y number of training set.

5.3.1.3 Transfer Function

The transfer functions characterize the relationship between two adjacent layers. For mathematical reason, the transfer functions should be differentiable because the gradient of mean-square-error with respect to weight parameters is needed in backpropagation training process. There are several types of transfer functions available, such as the sigmoid, inverse, and linear functions (Duch and Jankowski 1999). The choice of transfer functions mainly depend on the complexity and performance of neural network model. In the viewpoint of prognostics, the degradation normally does not have a complex behavior. Most cases, the degradation monotonically increases or decreases. Even though a multilayer network model with different types of transfer functions is available, a two-layer model with sigmoid and linear functions will be enough to characterize degradation behavior. By combining these two transfer functions and linear combination of input variables, it is possible to represent a variety of model forms in a two-layer network model.

The pure linear and tangent sigmoid functions are shown in Fig. 5.7, which can be obtained with the following equations:

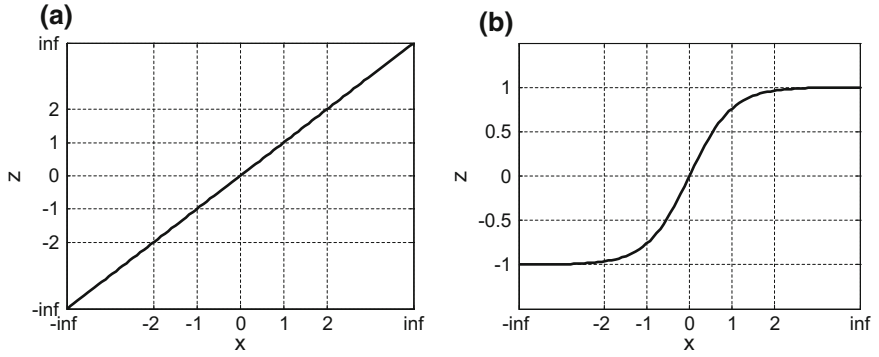


Fig. 5.7 Typical transfer functions. **a** Pure linear function (Eq. 5.31), **b** tangent sigmoid function (Eq. 5.32)

$$z = x. \tag{5.31}$$

$$z = \frac{1 - e^{-2x}}{1 + e^{-2x}}. \tag{5.32}$$

In order to show the capability of representing a complex behavior using a simple transfer function, a two-layer network model is considered that has one input node, one (or five) hidden node, and one output node. The weights and biases are randomly generated from a uniform distribution: $U(-5, 5)$, which is used to calculate the network model output at an arbitrary input based on Eq. 5.28 when $n_z = 1$. Then Fig. 5.8 can be obtained for $x \in [-1, 1]$ by repeating the process to calculate the network output five times with randomly generated weights and biases

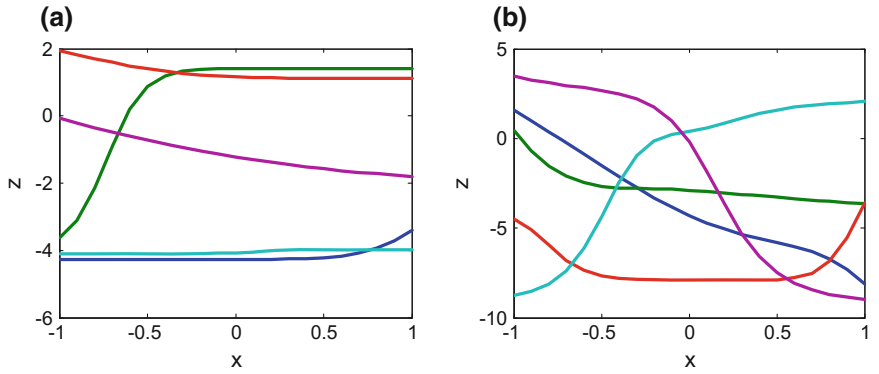


Fig. 5.8 Various shapes of the network outputs. **a** Network output with one hidden node **b** Network output with five hidden node

(Obtaining this figure remains as the Exercise problem P5.11), Fig. 5.8a, b are when the number of hidden nodes is one and five, respectively. The model complexity depends on the number of hidden nodes. Considering that the degradation behavior is in general monotonic, a two-layer network model with the tangent sigmoid and pure linear functions should be general enough to model the degradation behaviors. Therefore, the neural network model for the purpose of prognostics is not expanded with more than one hidden layer in this book. For the transfer functions, other forms can be used, but the pure linear and the tangent sigmoid functions are usually employed.

5.3.1.4 Backpropagation Process

The backpropagation is a training method to determine weights and biases through a learning/optimization algorithm by backward propagation of the errors between the training data and the network outputs. The backpropagation can be understood as a layer-by-layer updating, which is more effective to find optimum values of the parameters than updating all of them together. The steps for training process with a $n_x \times n_y$ matrix of training input data $\mathbf{X}$ and a $1 \times n_y$ vector of training output data $\mathbf{y}$ as follows:

Step 1. Initial values of the parameters (weights and biases) are set. Selection of initial weight parameters is important as it affects whether neural network can find the global optimum and how fast the convergence rate is. Too large initial weights are likely to make them fall in the saturation region easily and the network barely learns, while too small weights lead to a low learning rate. For optimum results, initial weight parameters are usually set as random numbers chosen between -1 and 1.

In the case of damage prognostics in this book, damage is monotonically increases or decreases, but it stays in a positive number. In addition, the level of damage in previous times are often used as inputs to the next time. Therefore, it is difficult to end up negative weights, which means that the damage in next time is in different sign from previous times. An increasing damage may have $w_x > 1$, while decreasing damage may have $w_x < 1$. Because of this reason, the initial estimates of weight parameters are set to be one.

Step 2. Network simulation outputs $\mathbf{z}$ are calculated with the weights and biases in Step 1 through the feedforward process (see Fig. 5.6 and Eq. 5.29). This step is performed for all $n_x \times n_y$ training data. The dimension of $\mathbf{z}$ is a $1 \times n_y$.

Step 3. The errors between the training output data $\mathbf{y}$ and the simulation outputs $\mathbf{z}$ are calculated. Different errors can be used, such as mean squared error (MSE), mean-absolute-error, and sum of squared errors. For example, the following MSE can be used:

$$\text{MSE} = \frac{1}{n_y}\sum_{i=1}^{n_y}[y_i - z_i(\mathbf{W}_x, \mathbf{B}_h, \mathbf{w}_h, \mathbf{b}_z)]^2. \tag{5.33}$$

Step 4. The changes of weights and biases, $\Delta\mathbf{w}_h$ and $\Delta\mathbf{b}_z$, of the output layer are calculated. In the backpropagation stage, the weights and biases at the output layer are updated first. Different optimization algorithms have different methods of calculating $\Delta\mathbf{w}_h$ and $\Delta\mathbf{b}_z$. For example, the gradient descent and the Levenberg–Marquardt (LM) algorithms respectively, calculate a gradient and a Jacobian of MSE with respect to the parameters (see the reference by Yu and Wilamowski (2011) for more details). The gradient descent is a basic learning algorithm used with the backpropagation, but the LM algorithm is commonly used since it is fast and stable in convergence.

Step 5. The changes of weights and biases, $\Delta\mathbf{W}_x$ and $\Delta\mathbf{B}_h$, of the hidden layer are calculated. Calculation of $\Delta\mathbf{W}_x$ and $\Delta\mathbf{B}_h$ depends on $\Delta\mathbf{w}_h$ and $\Delta\mathbf{b}_z$ in Step 4 and also MSE in Step 3.

Step 6. The weights and biases are updated by adding the changes in Step 4 and Step 5.

Step 7. Steps 1–6 are repeated until the network performance reaches to the stopping criterion.

5.3.1.5 Introduction to MATLAB Functions for Feedforward Neural Network

A sequence of MATLAB functions can be used to define a network model, to set up configurations, to train the network model (i.e., to determine weights and biases), and to predict future degradation behavior. The following is four steps of making predictions using FFNN.

In the following steps, the italic letters represents options that can be modified, but the default options explained so far will be mostly adopted in this book. For more options and understanding the neural network process, see the MATLAB user's guide (Beale et al. 2015).

(1) The function `feedforwardnet` creates a two-layer network model with user-defined number of hidden nodes. *Transfer functions* have default options; the tangent sigmoid and the pure linear functions are used for the hidden and the output layers, respectively. Once a network model is created, all other required information are automatically assigned with default options, which is explained in the following steps.

(2) The function `configure` normalizes input and output data and assigns them to each node. This step is optional. It automatically *normalizes* the input and output data between −1 and 1, and the *initial values* of weights and biases are randomly selected. Configuration is automatically done when the next step (training) is run, but this step can be added to assign the initial values of the parameters instead of random selection.
(3) The function `train`[7] trains the network model with the LM backpropagation for *training method* and the MSE for the *error function* (or performance evaluation function). For MATLAB implementation, early stopping option has several criteria. One of stopping criteria is based on validation error, which is the most critical criterion of preventing overfitting. In this criterion, training data are divided into three sets based on random *data division method*, 70 % for training, 15 % for validation, and 15 % for test. The training sets are used to update the weights and biases based on their error in the network output. The error of the training set keeps decreasing during the training process, and the error in the validation set also decreases, which means the training process goes well. On the other hand, the error in the validation set starts to increase after some phase of training process, i.e., good performance at the training set but bad performance at the validation set (new points, prediction points), which means that the network model overfits the training data. Therefore, the training process stops when the validation error starts increasing. The test data sets are not used during the training process but used for testing the prediction accuracy with trained parameters.
(4) The function `sim` simulates/predicts the network outputs at new inputs based on the identified weights and biases during the training process.

Example 5.5 `feedforwardnet`
Consider the same problem given in Example 5.2. Perform the feedforward neural network by using the MATLAB function `feedforwardnet` with one hidden node and plot predictions at $x_{\text{new}} \in [-5, 25]$.

Solution:
The following MATLAB script shows how to use the `feedforwardnet` MATLAB toolbox to perform FFNN introduced in Sect. 5.3.1.

[7]The function train uses all data at once, while `adapt` that is another learning method uses data one by one and is suitable for online learning. For real problems, `adapt` can be utilized since it does not require to store tremendous amount of monitoring data, but `train` is used in this book since it is more efficient for most practical problems.

```
x=[0    5   10   15   20];       %[1 x 5] training input data
y=[1 0.99 0.99 0.94 0.95];      %[1 x 5] training output data
nh=1;                                  % num. of hidden node
net=feedforwardnet(nh);       % creat two-layer network model
net=configure(net,x,y);                  % this is optional
[netModel,trainRecor]=train(net,x,y); % train the model 'net'
xNew=-5:0.1:25;                          % new input points
z=sim(netModel,xNew);                  % simulation results
```

The process itself is quite simple. First, the measurement times (cycles) and the degradation data are defined as training input data, `x`, and output data, `y`, respectively. The network model, `net` is created with function `feedforwardnet` after defining the number of hidden node in `nh = 1`. The next configuration step, `configure`, is optional, but with this we can extract the initial values of weights and biases by using the following variables: `net.iw` for input weights, `net.lw` for hidden weights, and `net.b` for biases, which are shown in Fig. E5.5 (blue-colored values). Note that the initial values can be different every time when `configure` is called since they are randomly generated.

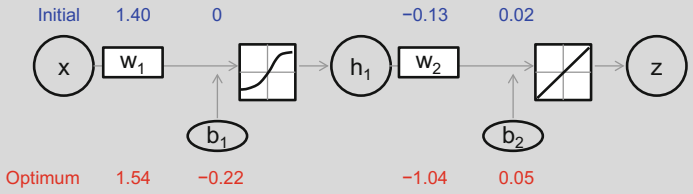


Fig. E5.5 Feedforward network model with initial and, optimum values of parameters

Now, the network model is trained based on function `train` and the trained model is named as `netModel`. The optimum values of parameters can be shown as the same way, but using `netModel` instead of `net`, which are also shown in Fig. E5.5 (red-colored values). The results can also vary every time since the initial values of parameters and the training sets will be different.

The number of training data that is three in this problem is too small, which is even less than the number of parameters to be estimated. In such a case, the estimated network model is not accurate. In general, it is required that the number of training data must be larger than that of network model parameters. Especially, since only 70 % of training data are used for the purpose of training, it is important to have enough training data to have a robust training results.

Training records are stored in `trainRecor`. For example, the indexes for training, validation, and test sets can be shown by using `trainRecor.trainInd`, `trainRecor.valInd`, and `trainRecor.testInd`, respectively, whose results are 2, 3, 4 for training, 5 for validation, 1 for test sets (again, these results can vary). Also, the performance (MSE) of training, validation, and test sets are recoded, and it can be plotted with `plotperf(trainRecor);` which is shown in Fig. E5.6. There is a total 8 epochs that can be known with `trainRecor.num_epochs`, and the training stops at 2 epochs that can be known with `trainRecor.best_epoch` based on the early stopping. Since the performance of validation sets are recoded in `trainRecor.vperf`, the circle in the figure can be plotted with the following codes:

```
bestE=trainRecor.best_epoch;
bestP=trainRecor.best_vperf; % or trainRecor.vperf(bestE+1)
hold on; plot(bestE,bestP,'o');
```

Note that the performance of the validation set starts to increase after the best epoch at the circle in the figure, while that of the training set still decreases.

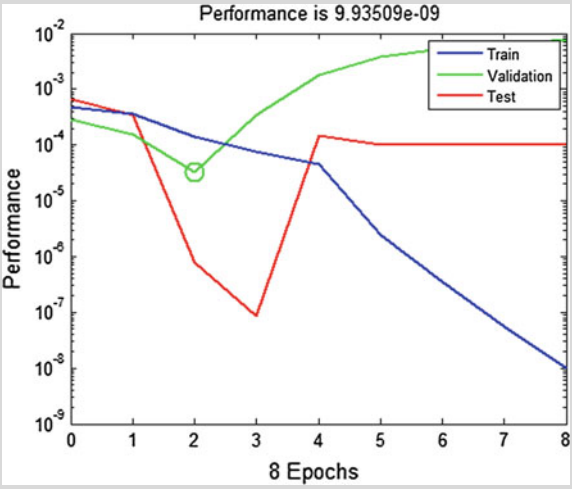


Fig. E5.6 Performance plot for training, validation and test sets

Once training the network model is completed, simulation outputs can be obtained with `sim` for $x_{new} \in [-5, 25]$, which is shown in Fig. E5.7.

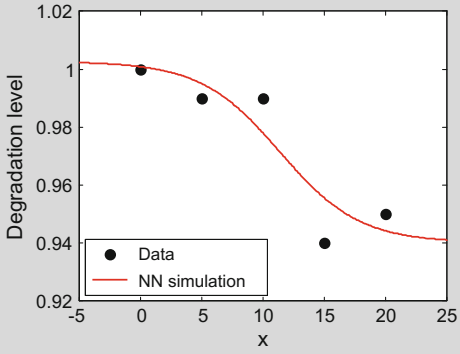


Fig. E5.7 FFNN simulation results

5.3.1.6 Uncertainty

As explained earlier, the network model prediction can vary due to the randomly chosen initial values of weight and bias parameters as well as randomly chosen 70 % of training data set. Because of these randomness, the network model prediction has uncertainty. There are several ways to consider uncertainty in NN simulation results, which will be discussed in Sect. 5.5.3. In this book, uncertainty in the prediction results is obtained by repeating the NN process several times. Detailed process is explained in Example 5.6.

Example 5.6 Uncertainty in NN simulation
Plot all simulation results by repeating the process in Example 5.5 5 times. Record the initial and optimum values of weights and biases, and the indexes of train, validation, and test sets.

Solution:
Whenever the network model is configured or initialized with `net = init(net)`, the initial values of parameters are assigned differently. Also, the train, validation, and test sets change when the training is restarted. These two are the sources of uncertainty in the neural network process and make different results. Figure E5.8 can be obtained by repeating the process given in the solution in Example 5.5, and the initial/optimum values of parameters and the indexes of data sets are listed in Table E5.2. In the figure, most results

follow the trend of data except for the fifth simulation result (magenta dashed curve). This is because of the early stopping, which sometimes hinders to obtain proper optimization results. As shown in Fig. E5.9 and the last column in Table E5.2, it practically stops at the initial stage. The optimization algorithm can be modified to resolve this problem, e.g., modifying the parameters associated with the optimization algorithm or using other optimization algorithms. In this book, however, regulations are applied based on the property that degradation should increase or decrease along with the cycles rather than modifying the optimization algorithm. This will be further explained in MATLAB code **[NN]**.

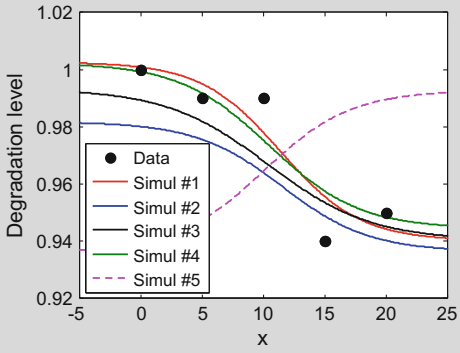


Fig. E5.8 FFNN simulation results with 5 repetitions

Table E5.2 Prediction intervals in NN simulation

Repetition	1	2	3	4	5
$w_{1,init}$	1.4000	−1.4000	1.4000	−1.4000	1.4000
$b_{1,init}$	0.0000	0.0000	0.0000	0.0000	0.0000
$w_{2,init}$	−0.1332	0.6697	0.7417	−0.3831	0.9438
$b_{2,init}$	0.0179	−0.9266	−0.8994	0.6767	−0.1827
$w_{1,opti}$	1.5431	−1.4141	1.1967	−1.3651	1.4000
$b_{1,opti}$	−0.2173	0.2205	−0.0467	0.0674	0.0000
$w_{2,opti}$	−1.0414	0.7558	−0.8837	0.9658	0.9438
$b_{2,opti}$	0.0480	−0.3632	−0.1038	0.1140	−0.1827
Train set	2, 3, 4	1, 3, 5	1, 4, 5	3, 4, 5	1, 2, 3
Validation set	5	4	3	1	4
Test set	1	2	2	2	5

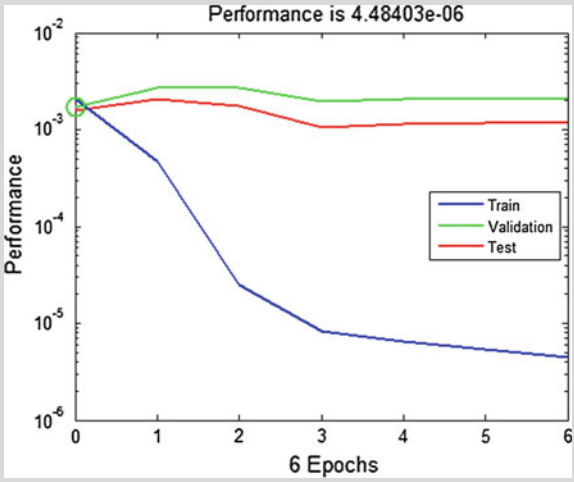


Fig. E5.9 Performance plot for fifth result

5.3.2 *MATLAB Implementation of Battery Prognostics Using Neural Network*

In this section, the MATLAB implementation of neural network code **[NN]** is explained using the same battery problem in Sect. 4.1.1. Problem definition and post-processing parts are similar to that of **[NN]** with minor modifications. Therefore, focus is on the prognosis part of the code.

5.3.2.1 Problem Definition (Lines 5–15, 17, 30–33)

All required variables for NN are the same as that of GP. It is noted that even though the repetition of the NN process replaces for the uncertainty quantification, thousands of repetition may not be appropriate. Therefore, 30–50 repetitions are employed for the number of samples, `ns` in NN (line 15), which might be the proper repetition number to capture different simulation results caused by the different data sets and initial values of weights and biases (`ns = 30` in this problem). Since NN automatically normalize the input and output matrix between −1 and 1, additional process is not required as in the GP. Therefore, codes for the input training matrix (line 17) and the new input (line 31), respectively, become

```
xTrain=time(1:length(y)); yTrain=y;
```

and

```
xNew=time(ny-1+k);
```

5.3.2.2 Prognosis Using NN

The MATLAB functions for NN are already introduced in Example 5.5, but some modifications are included in the MATLAB code **[NN]**. First of all, the following command line is used to run the battery prognostics problem (line 1).

```
rul=NN({'purelin'; 'purelin'},1,2);
```

There are three input variables to run **[NN]**. For this problem, two linear functions are used for `TransFunc` with one hidden node, `nh=1`.[8] Also, `outliCrite=2` is the criterion to handle outliers that are improper prediction results such as the magenta dashed curve in Fig. E5.8. Its value represents the level of standard deviation to remove predictions. That is, `outliCrite=2` means the results beyond the bound of 2σ are eliminated from the final results (will be explained later).

The transfer functions are assigned after creating the network model (lines 24–25). Since a test set is not used for training but for comparison between different models, it is not considered. Instead, the test set is added to the validation set by adjusting the ratio of validation data to 30 % (line 26). This can improve prediction results with small number of training data. Then, training process can be performed (line 28) with the input and output matrices defined in line 17.

Even though the early stopping is employed to prevent overfitting, it cannot control the underfitting in which the fitting results cannot follow the data trend well with a contrast to the overfitting. The magenta dashed curve/lines in Figs. E5.8 and 5.9a show that case. All samples for degradation prediction including gray and magenta lines in Fig. 5.9a can be plotted with `plot(time(ny:end),zHat)`

[8]Since two linear functions are used, it is always a linear function, and thus more than one hidden node are not necessary. Besides, increasing the number of hidden nodes increases the number of parameters to be estimated, which can make prediction results worse.

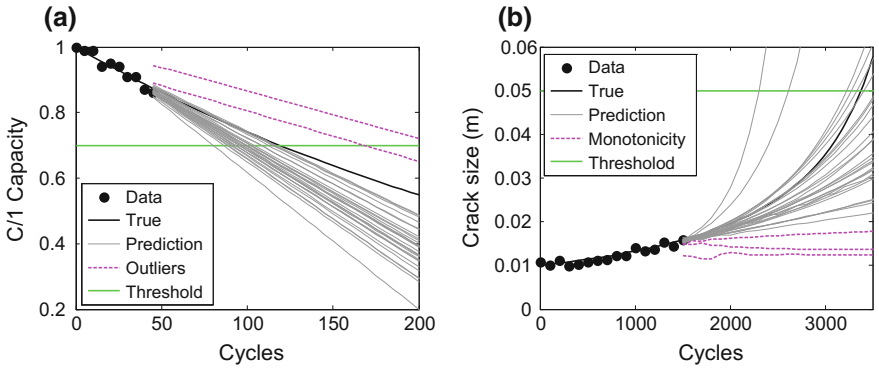


Fig. 5.9 Illustration of prediction outliers. **a** Case 1, **b** case 2

after loading the saved work file. To prevent having these improper results, regulations are added in lines 36–44. If a prediction among `ns` repetitions cannot satisfy the regulations, it is removed from the prediction results. First, the outliers deviated from the majority of prediction results are determined based on the results at the last prediction step, `z1` (line 36). The samples located within 2σ that is from the criterion option, `outliCrite = 2` are selected in `loca1` (line 37), which can be considered as a 97.72 % confidence interval. Those predictions that are out of 2σ confidence interval are removed from the prediction samples. Also, another regulation is added (lines 38–44) to consider the case that simulation results do not monotonically increase/decrease as shown in Fig. 5.9b. Based on the codes in lines 38–44, samples that monotonically increase/decrease in the prediction results are selected in `loca2` (line 43).

The final prediction results that satisfy both two regulations are saved as `degraPredi` (lines 45–46). The number of final samples is displayed to check how many outliers are eliminated (lines 47–48). In this case 3 our of 30 are eliminated as outliers (can vary). The remaining process is the same as GP.

5.3.2.3 Postprocessing

As results, the degradation and RUL are predicted as shown in Fig. 5.10, and the percentiles of RUL distribution at 45 cycles are obtained as

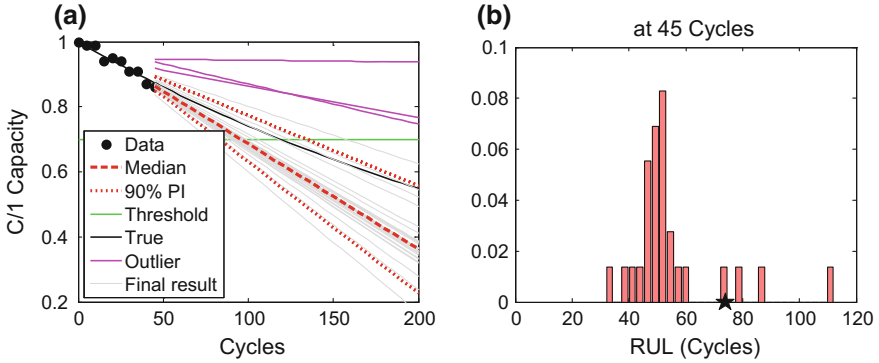


Fig. 5.10 NN prediction results for the battery problem. **a** Degradation **b** RUL

```
Percentiles of RUL at 45 Cycles
 5th: 38.0446,  50th (median): 50.8084,  95th: 89.8012
```

In Fig. 5.10, all prediction results from 30 repetitions are plotted along with the median and the 90 % prediction interval. Three magenta solid lines in the figure represent outliers excluded to calculate the median and the prediction interval. The final prediction results (gray lines) and excluded ones (magenta lines) can be plotted after loading the saved work file with the following code.

```
plot(time(ny:end),zHat,'m');
plot(time(ny:end),degraPredi,'color',[.5 .5 .5]);
```

From Figs. 5.5 and 5.10, it is shown that the uncertainty in the prediction results is larger than that of GP. The main reason is that in NN, there are more parameters to be estimated with a less number of training data than GP. In NN, there are four parameters (one input weight, one hidden weight, one hidden bias, and one output bias), and seven data in the training sets (70 % of total ten training data), while there are three parameters (one scale parameter and two coefficients) and ten training data in GP.

[NN]: MATLAB Code for Neural Network

```
function rul=NN(TransFunc,nh,outliCrite)
clear global;
global DegraUnit TimeUnit time y thres signiLevel ns ny nt
%=== PROBLEM DEFINITION 1 (Required Variables) ==============
WorkName=' ';            % work results are saved by WorkName
DegraUnit=' ';                          % degradation unit
TimeUnit=' ';              % time unit (Cycles, Weeks, etc.)
time=[ ]';     %[cv]: time at both measurement and prediction
y=[ ]';                               %[ny x 1]: measured data
nT= ;        % num. of training set including the current one
nt=[ ]';%[nT x 1]: num. of training data in each training set
thres= ;                        % threshold (critical value)
degraTrue=[ ]';             %[cv]: true values of degradation
signiLevel= ;          % significance level for C.I. and P.I.
ns= ;                 %  number of repetition for NN process
%=== PROBLEM DEFINITION 2 (Training Matrix) =================
xTrain=[ ]; yTrain=[ ];
%============================================================
% % % PROGNOSIS using NN
ny=length(y);
for i=1:ns;                                       %% NN Process
 disp(['repetition: ' num2str(i) '/' num2str(ns)]);
 % create FFNN
 net=feedforwardnet(nh);
 net.layers.transferfcn=TransFunc;
 net.divideParam.valRatio=0.3; net.divideParam.testRatio=0;
 % train FFNN
 [netModel,trainRecod]=train(net,xTrain',yTrain');
%=== PROBLEM DEFINITION 2 (Input Matrix for Prediction) =====
 for k=1:length(time(ny:end));      %% Degradation Prediction
  xNew=[ ];
  zHat(k,i)=sim(netModel,xNew');
 end;
%============================================================
end
z1=zHat(end,:);              %% Prediction Results Regulation
loca1=find(abs(z1-mean(z1)) < outliCrite*std(z1));
i0=1;
for i=1:ns;
 if y(1)-y(nt(1))>0; z2=wrev(zHat(:,i));
 else z2=zHat(:,i);
 end;
 if issorted(z2)==1; loca2(i0)=i; i0=i0+1; end
end
loca=intersect(loca1,loca2);                %% Final Results
degraPredi=zHat(:,loca);
ns=size(degraPredi,2);
```

```
disp(['Final num. of samples: ' num2str(ns)])
% % % POST-PROCESSING
rul=POST([],degraPredi,degraTrue);        %% RUL & Result Disp
Name=[WorkName ' at ' num2str(time(ny)) '.mat']; save(Name);
end
```

5.4 Practical Use of Data-Driven Approaches

In the previous section of Gaussian process regression, cycle or time is used as the input variable, which might work well for a simple problem with one data set. It, however, cannot be proper for most practical cases with complex behaviors. It is not reasonable to use cycle/time as an input without additional input information when there are training data sets obtained under different usage conditions, because different operating conditions may lead to different degradation rates. Therefore, a practical method for data-driven approaches is introduced in this section, i.e., degradation levels are used as both input and output. The current degradation state as an output can be expressed with several previous degradation states as the inputs, which can utilize the information included in different degradation rates from the different data sets. These inputs are similar to that of NN model. Also, having previous degradation states as inputs can allow to use a simpler functional form than when times/cycles are used as inputs, which can help to prevent overfitting. Therefore, prediction results can be more accurate with given information. Detailed description is explained in the following subsections.

5.4.1 Problem Definition

The crack growth problem introduced in Sect. 4.5.1 is employed again for the practical use of data-driven approaches. Degradation data are generated with the same manner as the data given in Table 4.2, but the Gaussian noise is not added this time. These data are considered as the prediction set. For the training set, one additional data set are generated with the same model parameters, $m_{\text{true}} = 3.8$, $C_{\text{true}} = 1.5 \times 10^{-10}$ and initial crack size, $a_{0,\text{true}} = 0.01\text{m}$ but different loading condition, $\Delta\sigma$ = 79 MPa ($\Delta\sigma$ = 75 MPa for the prediction set). This set has 26 data measured at every 100 cycles from zero to 2500 cycles. The training data including both prediction and training sets are shown in Fig. 5.11 along with the true degradation of the prediction set.

As the method used by Chakraborty et al. (1992), the previous degradation data are employed as input variables. When the previous three data are used as inputs, the fourth data become the output, and this is performed sequentially to make input and output matrices, which is explained in Table 5.1 for the case of one data set.

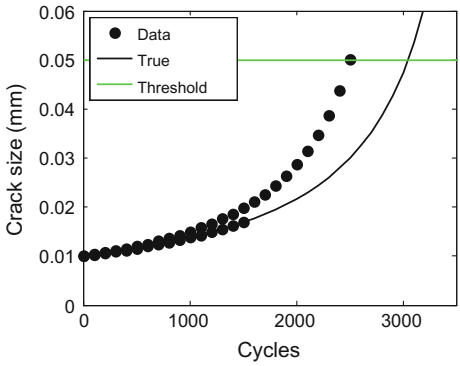


Fig. 5.11 Training data for crack growth problem

Table 5.1 Input and output data

	Training				Prediction			
Cycle	4	5	…	k	$k+1$	$k+2$	$k+3$	…
Input	y_1, y_2, y_3	y_2, y_3, y_4	…	y_{k-3}, y_{k-2}, y_{k-1}	y_{k-2}, y_{k-1}, y_k	y_{k-1}, y_k, z_{k+1}	y_k, z_{k+1}, z_{k+2}	…
Output	y_4	y_5	…	y_k	z_{k+1}	z_{k+2}	z_{k+3}	…

In the table, k is the current time index, and there are $k-3$ training data sets since the previous three data are the input, which makes input and output training matrices. After the training process is completed, inputs for prediction are set with the same manner. At $k+1$th cycles, the simulation output, z_{k+1} is predicted with the three most recent measurement data, $\mathbf{y}_{k-2:k}$.

Two different predictions are possible. The first one is called short-term prediction, where measured three data at previous cycles are used to predict the degradation at the current cycle. This is one-step ahead prediction. The difficulty of short-term prediction is that it is not possible to predict the degradation in future time because measurement data are not available. However, short-term prediction is accurate because it uses actual measured data and extrapolation distance is short. The second one is called long-term prediction, where predicted degradations are used to predict the degradation at further future time. In Table 5.1, training data are available up to kth cycle. Starting from $k+2$th cycle, there is no three measurement data available. In this case, the simulation output (prediction result at the previous step) is utilized (see $k+2$th and $k+3$th cycles) for a long term prediction. Of course, since prediction degradation is used for inputs, the accuracy will be worse than short-term prediction, but it allows to predict the degradation in long future time. This book considers the long-term prediction only.

5.4.2 MATLAB Codes for the Crack Growth Example

Based on the given information in Sect. 5.4.1, the problem definition part (lines 5–15) in the codes **[GP]** and **[NN]** is changed as follows:

```
WorkName='Crack_GP'; % or 'Crack_NN'
DegraUnit='Crack size (m)';
TimeUnit='Cycles';
time=[0:100:2500, 0:100:1600, 1700:100:4000]';
y=[0.01 0.0104 0.0107 0.0111 0.0116 0.0120 0.0125  ...
 0.0131 0.0137 0.0143 0.0150 0.0158 0.0166 0.0176   ...
 0.0186 0.0198 0.0211 0.0226 0.0243 0.0263 0.0287   ...
 0.0314 0.0347 0.0387 0.0437 0.0501 ...
 0.0100 0.0103 0.0106 0.0109 0.0112 0.0116 0.012 0.0124 ...
 0.0128 0.0133 0.0138 0.0143 0.0149 0.0155 0.0162 0.0169]';
nT=2;
nt=[26 16]';
thres=0.05;
mTrue=3.8; cTrue=1.5e-10; aTrue=0.01; dsig=75; t=0:100:3500;
degraTrue=(t'.*cTrue.*(1-mTrue./2).*(dsig*sqrt(pi)).^mTrue...
 +aTrue.^(1-mTrue./2)).^(2./(2-mTrue));
signiLevel=5;
ns=5e3; % or 30 for NN
```

There are 42 measurement data in `y`, 26 from the training set (first four lines) and 16 from the prediction set (last two lines). The measurement data from the training set should be inputted first, and followed by data from the prediction set. This is the same for the cycle/time, the first (`0:100:2500`) and the second (`0:100:1600`) portions in `time` correspond to the training set and the prediction set, respectively. The last one (`1700:100:4000`) is for the degradation prediction, i.e., it is predicted at every 100 cycles from 1700 cycles to 4000 cycles. For this problem, the total number of data sets, `nT = 2`, and the number of training data in each set is inputted as `nt = [26 16]'`.

In data-driven approaches, one of the important things is to properly build the input and output matrices to make good use of given information, which is related to modify the second problem definition parts, lines 17 and 27–31 in the code **[GP]** and lines 17 and 30–33 in the code **[NN]**. Any variation will be possible based on users' application and novel ideas, but the way introduced in Sect. 5.4.1 is implemented. The training input and output matrices explained in Table 5.1 can be programed as

```
nx=3;                      % num. of previous data as the input
y0=(y-min(y))/(max(y)-min(y));        % normalization of data
nm=size(y,2); a=0; b=0;
for l=1:nT;
 for k=1:nt(l)-nx;
  xTrain(k+a,1:nx*nm)=reshape(y0(k+b:(k-1)+nx+b,:),1,nx*nm);
  yTrain(k+a,:)=y0(k+nx+b,:);
 end; a=size(xTrain,1); b=sum(nt(1:l));
end
```

which replaces line 17 in the code **[GP]**. The number of input (previous data) can be modified with `nx` in the code above. Note that even if `time` is defined in the problem definition section, it is not used as an input. Instead, previous measured data are used as inputs. Therefore, the GP predicts the degradation level when `nx` previous degradation data are given as inputs. Therefore, the functional form of GP becomes

$$\hat{z}_k = \mathrm{GP}(y_{k-3}, y_{k-2}, y_{k-1})$$

For the input for prediction part, the lines 27–31 in **[GP]** are replaced with the following code.

```
input=y0(end-nx:end,:);
for k=1:length(time(ny:end));
 xNew=reshape(input(k:(k-1)+nx,:),1,nx*nm);
 [zMean(k,:),zSig(k,:)]= ...
                      GPSIM(xNew,funcOrd,hH,R,thetaH,sigmaH);
 if k>1 input(k+nx,:)=zMean(k,:); end
end;
```

The code modification for NN is basically the same as that of GP. For input and output training matrices, normalization is not considered in this case, and the following code is applied to line 17 in the code **[NN]**.

```
nx=3;                      % num. of previous data as the input
nm=size(y,2); a=0; b=0;
for l=1:nT;
 for k=1:nt(l)-nx;
  xTrain(k+a,1:nx*nm)=reshape(y(k+b:(k-1)+nx+b,:),1,nx*nm);
  yTrain(k+a,:)=y(k+nx+b,:);
 end; a=size(xTrain,1); b=sum(nt(1:l));
end
```

For the input for prediction part, lines 30–33 in **[NN]** are replaced with the following code.

```
input=y(end-nx:end,:);
for k=1:length(time(ny:end));
 xNew=reshape(input(k:(k-1)+nx,:),1,nx*nm);
 zHat(k,i)=sim(netModel,xNew');
 if k>1, input(k+nx,:)=zHat(k,i); end
end;
```

There is an addition modification required in GP. A value of the exponent, 2, in the correlation equation in Eq. 5.23 often causes singularity in the correlation matrix. Therefore, it usually adjusted with a value less than 2. In this problem, the exponent is adjusted with 1.9, and line 66 in **[GP]** is replaced with the following code.

```
R(k,l)=exp(-(norm(xTrain(k,:)-x0(l,:))/h)^1.9);
```

Now the codes **[GP]** and **[NN]** can respectively be run with

```
rul=GP(1,0.1);
```

and

```
rul=NN({'purelin'; 'purelin'},1,2);
```

in which linear functions are used for both hidden and output layers in **[NN]**. Also, a linear global function is used in **[GP]**.

5.4.3 Results

The results for degradation predictions are shown in Fig. 5.12 for both GP and NN. Note that linear functions are used for both global function in GP and transfer functions in NN, but they can predict nonlinear behavior of degradation well. This is because the relation between previous three degradation data and the current degradation data can be captured by a linear function. The GP predicts degradation accurately with very small uncertainty as shown in Fig. 5.12a. However, the

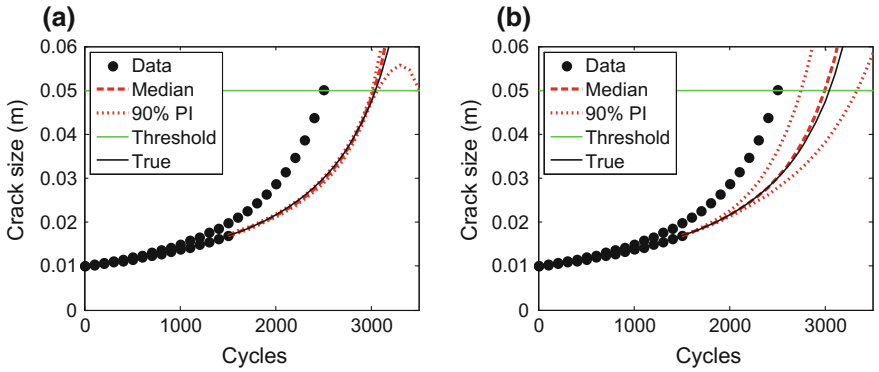


Fig. 5.12 Degradation prediction. **a** GP, **b** NN

prediction lower bound after 3000 cycles does not follow the true behavior. This may happen because the largest input crack size is 0.05 m, and beyond this size, the GP simulation becomes extrapolation. In the extrapolation region, the correlation is reduced, and GP tends to follow the global function. Also, the results are largely affected by the correlation function including different scale parameter and the exponent in Eq. 5.23. The degradation result from NN is shown in Fig. 5.12b. Out of 30 repetitions (ns = 30), 19 of them failed to satisfy two regulations: outliers and monotonicity. Therefore, only 11 degradation results are used to calculate the 90 % prediction intervals in the figure.

RUL prediction results from GP and NN are shown in Fig. 5.13, and the percentiles of RUL distribution at 1500 cycles are obtained from GP:

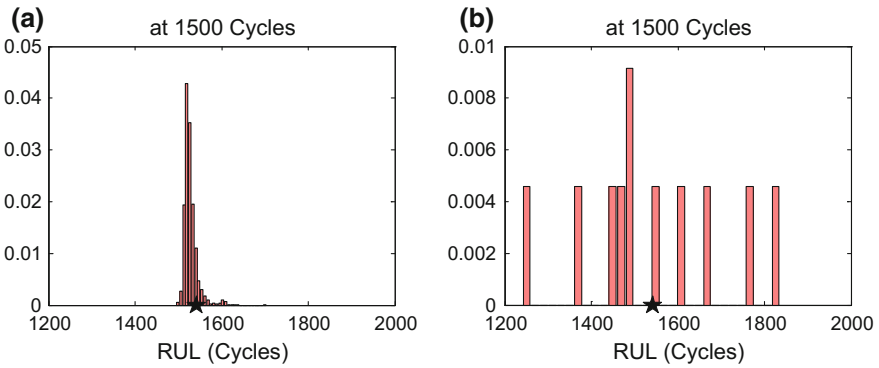


Fig. 5.13 RUL prediction at 1500 cycles. **a** GP, **b** NN

```
Percentiles of RUL at 1500 Cycles
 5th: 1511.83,  50th (median): 1524.44,  95th: 1554.58
```

And from NN:

```
Percentiles of RUL at 1500 Cycles
 5th: 1246.77,  50th (median): 1495.7,  95th: 1832.39
```

5.5 Issues in Data-Driven Prognostics

In the following, three important issues are discussed for GP- and NN-based prognostics, which are basically the same as the issues in physics-based prognostics. Many options and variations are possible for data-driven algorithms, but this book introduces the basic ones. Therefore, it is important that users are aware of these issues in order to use the algorithm properly.

5.5.1 Model-Form Adequacy

5.5.1.1 GP: Global and Covariance Functions

In GP-based prognostics, the performance largely depends on the selection of global and covariance functions. Since GP has usually been employed in interpolation, a global function often takes the form of constant, and it has been considered as less important than the covariance function. In prognostics, which is equivalent to extrapolation, the global function in GP is as much important as the covariance function. However, there is not much literature on selecting or updating the global function to improve the prediction capability in the extrapolation region.

On the other hand, covariance functions determine GP simulation quality for entire regions, and there are many researches to study the effect of the covariance function and to use a better covariance function. Mohanty et al. (2009) compared the prediction results of crack length under variable amplitude loading using the radial basis function (RBF) and NN-based covariance function, and showed that RBF-based GP model outperformed the NN-based model in their application. A few articles have also introduced nonstationary covariance function, which adapts to variable smoothness by adding or multiplying simple covariance functions. Paciorek and Schervish (2004) showed that the results by nonstationary covariance functions are better than those by the stationary GP, but pointed out the loss of simplicity of the algorithm, as the nonstationary GP requires more parameters. Brahim-Belhouari and Bermak (2004) employed nonstationary GP to predict respiration signals and compared them with those from an exponential covariance

function. Liu et al. (2012b) used the combination of three covariance functions to predict lithium-ion battery degradation.

5.5.1.2 NN: Network Model Definition

The definition of the network model includes selecting the number of hidden nodes, hidden layers, and input nodes. Although there is no general selection procedure for the number of hidden nodes, Lawrence et al. (1996) investigated the usage of mean squared errors in order to find the optimal number of hidden nodes. Gómez et al. (2009) used the idea of measuring complexity of function to determine the number of nodes and showed that the results were close to the optimum. More studies on the number of hidden nodes are summarized in the literature by Sheela and Deepa (2013). While one or two hidden layers are commonly used, Ostafe (2005) presented a method using pattern recognition to determine the number of hidden layers.

The problem of determining the number of input nodes always exists in the data-driven approach since all available information, such as time, loading conditions, and degradation data can be considered as inputs. It is natural to use degradation data as inputs, but Chakraborty et al. (1992) concluded that it was unclear how many past values should be used as inputs. Chang and Hsieh (2011) explored the optimal number of input nodes using particle swarm optimization. In conclusion, there is no universal procedure to establish a proper NN model, which may be difficult for new users without much experience. In recent years, there have been efforts to find proper model/method for each application rather than a study of selecting number of nodes and layers (Guo et al. 2012; Hodhod and Ahmed 2013; Kang et al. 2014).

5.5.2 Optimal Parameters Estimation

5.5.2.1 GP: Scale Parameters

Determining the scale parameters (or hyperparameters) in the covariance function is also important, since they determine the smoothness of the GP function. For example, in Eq. 5.23, the function becomes smoother as h is increased, but singularity is encountered for the correlation matrix if it is too high. In general, the parameters are determined via an optimization algorithm by minimizing a likelihood function corresponding to the error between a global function and data. Finding optimal parameters, however, is not guaranteed, and even if they are found, they are not always the best selection (An and Choi 2012). Since the scale parameters are seriously affected by the magnitudes of input and output values, a common practice is to normalize the input and output variables. Mohanty et al. (2009) studied the performance of predicting crack growth according to three

different types of scaling: logarithmic, normalized and log-normalized scaling. Neal (1998) considered the scale parameters as distributions rather than deterministic values, and An and Choi (2012) found that the result with distributions outperforms the one by deterministic values.

The current parameter estimation schemes were developed for the purpose of surrogate modeling where the main objective is to predict accurately in the interpolation region. There is no literature available to estimate the scale factor for the purpose of extrapolation, which is the main interest in prognostics. For example, if many health monitoring data are available with a small time interval, it is likely that the scale parameter becomes a small value because enough data are available with a small distance. However, when extrapolation is used with a small-scale parameter, the prediction quickly converges to the global model because the correlation decreases quickly. Therefore, it is likely that the prediction in the extrapolation region will be inaccurate.

5.5.2.2 NN: Weights and Biases

Once a network model is defined, the next issue is to find the weights and biases that are related with the model using a learning/optimization algorithm. In NN, no matter how complex the relationship between the input and output layer is, it is possible to express the relationship by augmenting the number of hidden layers and hidden nodes. However, more complex NN model ends up with more numbers of unknown parameters, which requires more training data. When the backpropagation algorithm is used especially, the following problems occur: (1) the global optimum is extremely difficult to find in the case of many parameters and (2) the convergence rate is very low and depends on the initial estimates. For these reasons, there have been many efforts to improve the drawbacks of this algorithm, such as a dynamic self-adaptation algorithm (Salomon and Hemmen 1996), a simulated annealing algorithm (Chen et al. 2006), combined genetic and differential evolution algorithm (Subudhi et al. 2008), and a technique combining the conjugate gradient optimization algorithm with the backpropagation algorithm (Nawi et al. 2008). There are many ensemble techniques to improve the performance of algorithms (Soares et al. 2013; Jacobs 1995; Drucker et al. 1994; Krogh and Vedelsby 1995; Chao et al. 2014; Naftaly et al. 1997), as well as other efforts in Refs. (Wilamowski et al. 2001; Salomon and Hemmen 1996; Nawi et al. 2008).

While the aforementioned methods are to find the parameters in a deterministic sense, there have been probabilistic approaches based on the Bayesian learning technique (Hernández-Lobato and Adams 2015; de Freitas 2003), in which uncertainty in the parameters are handled with an analytical method or a sampling method. Even though probabilistic approaches can cover the local optimal problem, the analytical method is limited to a specific case and the sampling error grows as the number of weight parameters increases. Therefore, finding optimal weights and biases is still challenging, and the performance of the NN algorithm quickly deteriorates with local optimal weight parameters.

5.5.3 *Quality of Degradation Data*

Last but not the least, uncertainty caused by bias and noise in data is an important issue regardless of prognostics approaches, which has a significant effect on the prediction results. Unfortunately, the bias cannot be handled with data-driven approaches because there are no parameters related to it, which is one of the drawbacks of data-driven approaches. The data quality issues for noise will be discussed in Chap. 6. Here, several issues related with data quality are addressed.

5.5.3.1 GP: Number of Data and Uncertainty in Data

Even though a large number of training data are usually good for increasing the accuracy of prediction results, it is not always true for GP because it also increases computational cost to calculate the inverse of the covariance matrix and may cause singularity as well. The direct matrix inversion may become computationally prohibitive when the number of data is greater than 1000 (MacKay 1997). To relieve this problem, only a portion of the whole data sets can be used (Lawrence et al. 2003; Foster et al. 2009). Melkumyan and Ramos (2009) suggested also a new covariance function based on the cosine function that inherently provides a sparse covariance matrix. Sang and Huang (2012) proposed an approximation method of a covariance function for both large and small number of training data based on a reduced rack covariance.

Even though the typical GP assumes that the data is accurate, the nugget effect representing the noise in data is sometimes considered by adding a value greater than zero to the diagonal terms of the covariance matrix, so that the simulated outputs do not pass noisy data points (Gramacy and Lee 2012; Andrianakis and Challenor 2012). Even though the value of the nugget effect is also found via optimization along with the scale parameters, the value cannot be determined uniquely due to the correlation with scale parameters. In addition, in the case of small data, it is inherently difficult to figure out how much noise exists in the data. Considering most health monitoring data include noise (often a significant level), GP inherently anticipates a difficulty to compensate for the effect of noise.

5.5.3.2 NN: Uncertainty in Prediction Results

It is common to provide confidence bounds based on nonlinear regression and/or the error between NN outputs and training data (Chryssoloiuris et al. 1996; Veaux et al. 1998; Rivals and Personnaz 2000). It, however, is difficult to find global optimum of parameters due to measurement noise, a small number of data compared to the number of parameters, and the complexity of damage growth, which can yield a significant error in prediction results. Bootstrapping (Efron and Tibshirani 1994; Khosravi et al. 2013) is also employed, which can easily be

implemented by running MATLAB NN toolbox several times because MATLAB uses different subsets of the training data to obtain weights and biases. Furthermore, this process can relieve the concern related to selecting initial weight parameters for optimization because the process automatically selects different initial parameters. For example, Liu et al. (2010) used this method with 50 attempts to predict a battery's RUL with uncertainty. Actually, a systematic method to handle the uncertainty in NN is the probabilistic neural network (PNN) (Specht 1990) using the Parzen estimator (Parzen 1962). However, most papers employ the PNN for classification or risk diagnosis (Giurgiutiu 2002; Mao et al. 2000), and the PNN for prognosis is rarely found, except for the study by Khawaja et al. (2005). They introduced a way to obtain not only confidence bounds but also confidence distributions based on PNN to predict a crack on a planetary gear plate. Khosravi et al. (2011) reviewed aforementioned methods, and considered combined intervals from the methods. On the other hand, Bayesian NN (Hernández-Lobato and Adams 2015; de Freitas 2003) has been proposed to resolve local optimum problem, which provides distribution of prediction results caused by measurement error and uncertainty in parameters that are identified as distributions based on Bayes' theorem instead of deterministic values given by an optimization process.

5.6 Exercise

P5.1 Repeat fitting function $y(x) = x + 0.5\sin(5x)$ in Sect. 5.2.1 with radial basis networks surrogate using 21 equal-interval points in the range of [1, 9] and the goal of mean squared error of 0.1. Compare the prediction results in the range of [0, 10] with Fig. 5.2.

P5.2 Generate 10 random numbers, $\mathbf{x}$, in [0, 10], shit them by 0.1 to define $\mathbf{x}_{\text{near}} = \mathbf{x} + 0.1$ and by 1.0 to define $\mathbf{x}_{\text{far}} = \mathbf{x} + 1.0$. Calculate the sine functions at the three sets: $\mathbf{y} = \sin(\mathbf{x})$, $\mathbf{y}_{\text{near}} = \sin(\mathbf{x}_{\text{near}})$, $\mathbf{y}_{\text{far}} = \sin(\mathbf{x}_{\text{far}})$. Compare correlation between $\mathbf{y}$ and $\mathbf{y}_{\text{near}}$, as well as the correlation between $\mathbf{y}$ and $\mathbf{y}_{\text{far}}$.

P5.3 Obtain the same results as Fig. E5.2 in Example 5.2

P5.4 Repeat Example 5.2 for $h = 140$. Discuss the results.

P5.5 Repeat Example 5.2 for the case of linear global function, $\theta_1 + \theta_2 x$.

P5.6 When a true function is given as $z(x) = x^2 + 5x - 10$ with $x \in [-8, 8]$, use five data at $x = \{-8, -4, 0, 4, 8\}$ to fit GP simulations with different scale parameters $h = 1, 10$ and different constant global functions $\xi\theta = 0, 10$. Plot them together and compare them.

P5.7 Derive the following equation for uncertainty in linear regression model referring the GP case. Compare it with the results from sampling method introduced in Chap. 2.

$$std(\hat{z}) = \sigma\sqrt{\frac{1}{n_y} + \frac{(x-\bar{x})^2}{S_{xx}}}$$

P5.8 For each case of Problem 5, plot the prediction intervals and actual error together.

P5.9 Implement the GP for the same problem in Sect. 5.2.4 but use different global functions, (a) constant and (b) second-order function. Compare and discus about the three results including in Fig. 5.5a.

P5.10 The true degradation of battery is given as $d = \exp(-0.003t)$. Generate 10 training data at time t = {0, 5, 10, …, 50} using the true degradation equation and add random noise that follows a normal distribution $N(0, \sigma^2)$. Using these training data, predict the median and 90 % confidence intervals of remaining useful life using Gaussian process regression. Use $\sigma = 0.01, 0.02, 0.05$. Use 0.7 for the threshold of degradation.

P5.11 Repeat the process to obtain different functions like given in Fig. 5.8 by changing the number of hidden nodes. Note that the results are not the exactly same as the figure because of the random weight and bias parameters.

P5.12 Consider one layer (no hidden layer) with one input node, and input and output data are given as

$$\mathbf{x} = \{0 \quad 1 \quad 2 \quad 3 \quad 4\}^{\mathrm{T}}, \quad \mathbf{y} = \{1.27 \quad 3.92 \quad 3.87 \quad 7.43 \quad 9.16\}^{\mathrm{T}}$$

Find weight and bias parameters in the network function `newlind` that designs a linear model and compare the results with the least squares method.

P5.13 Obtain Fig. E5.7 in Example 5.5 by a manual calculation with the optimum results of weights and biases that is given in Table E5.2. Note that the optimum results are based on the input and output ranged between −1 and 1.

P5.14 Repeat Example 5.6 without the test set, but 70 % of training set and 30 % of validation set instead. Compare the results with Fig. E5.8 and Table E5.2.

P5.15 Compare the degradation prediction results from (a) using time information and (b) using previous n data as the input. Solve this problem by using both GP and NN with only the data in prediction set in Fig. 5.11.

P5.16 There is a relatively large uncertainty in the prediction result from NN in Fig. 5.12b even though no noise data are used. Discuss what the uncertainty sources are in this case.

References

Ahmadzadeh F, Lundberg J (2013) Remaining useful life prediction of grinding mill liners using an artificial neural network. Miner Eng 53:1–8

An D, Choi JH (2012) Efficient reliability analysis based on Bayesian framework under input variable and metamodel uncertainties. Struct Multidiscip Optim 46:533–547

Andrianakis I, Challenor PG (2012) The effect of the nugget on Gaussian process emulators of computer models. Comput Stat Data Anal 56:4215–4228

Beale MH, Hagan MT, Demuth HB (2015) Neural Network Toolbox™ 8.4. User's Guide. The MathWorks, Inc., Natick

Benkedjouh T, Medjaher K, Zerhouni N et al (2015) Health assessment and life prediction of cutting tools based on support vector regression. J Intell Manuf 26(2):213–223

Bicciato S, Pandin M, Didonè G, Bello CD (2001) Analysis of an associative memory neural network for pattern identification in gene expression data. In: Biokdd01, workshop on data mining in bioinformatics, pp. 22–30

Bodén M (2001) A guide to recurrent neural networks and backpropagation. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.16.6652&rep=rep1&type=pdf. Accessed 29 May 2016

Brahim-Belhouari S, Bermak A (2004) Gaussian process for nonstationary time series prediction. Comput Stat Data Anal 47:705–712

Bretscher O (ed) (1995) Linear algebra with applications. Prentice Hall, New Jersey

Chakraborty K, Mehrotra K, Mohan CK et al (1992) Forecasting the behavior of multivariate time series using neural networks. Neural Networks 5:961–970

Chang JF, Hsieh PY (2011) Particle swarm optimization based on back propagation network forecasting exchange rates. Int J Innov. Comput. Inf. Control 7(12):6837–6847

Chao M, Zhi S, Liu X et al (2014) Neural network ensembles based on copula methods and distributed multiobjective central force optimization algorithm. Eng Appl Artif Intell 32:203–212

Chen SC, Lin SW, Tseng TY et al (2006) Optimization of back-propagation network using simulated annealing approach. In: Paper presented at the IEEE international conference on systems, man, and cybernetics, Taipei, Taiwan, 8–11 October 2006

Chryssoloiuris G, Lee M, Ramsey A (1996) Confidence interval prediction for neural network models. IEEE Trans Neural Networks 7(1):229–232

Coppe A, Haftka RT, Kim NH (2011) Uncertainty identification of damage growth parameters using nonlinear regression. AIAA Journal 49(12):2818–2821

de Freitas JFG (2003) Bayesian methods for neural networks. In: Dissertation, University of Cambridge

Drucker H, Cortes C, Jackel LD et al (1994) Boosting and other ensemble methods. Neural Comput 6(6):1289–1301

Duch W, Jankowski N (1999) Survey of neural transfer functions. Neural Comput Surveys 2:163–212

Efron B, Tibshirani RJ (1994) An introduction to the bootstrap. Chapman & Hall/CRC, Florida

Foster L, Waagen A, Aijaz N et al (2009) Stable and efficient Gaussian process calculations. J Mach Learn Res 10:857–882

Giurgiutiu V (2002) Current issues in vibration-based fault diagnostics and prognostics. In: Paper presented at the SPIE's 9th annual international symposium on smart structures and materials and 7th annual international symposium on NDE for health monitoring and diagnostics, San Diego, California, USA, 17–21 Mar 2002

Gómez I, Franco L, Jérez JM (2009) Neural network architecture selection: Can function complexity help? Neural Process Lett 30:71–87

Gramacy RB, Lee HK (2012) Cases for the nugget in modeling computer experiments. Stat Comput 22(3):713–722

Guo Z, Zhao W, Lu H et al (2012) Multi-step forecasting for wind speed using a modified EMD-based artificial neural network model. Renewable Energy 37(1):241–249

Happel BLM, Murre JMJ (1994) The design and evolution of modular neural network architectures. Neural Networks 7:985–1004

He Y, Tan Y, Sun Y (2004) Wavelet neural network approach for fault diagnosis of analogue circuits. IEE Proc Circuits Devices Syst 151(4):379–384

Hernández-Lobato JM, Adams RP (2015) Probabilistic backpropagation for scalable learning of bayesian neural networks. arXiv preprint arXiv:1502.05336

Hodhod OA, Ahmed HI (2013) Developing an artificial neural network model to evaluate chloride diffusivity in high performance concrete. HBRC J 9(1):15–21

Jacobs RA (1995) Methods for combining experts' probability assessments. Neural Comput 7 (5):867–888

Kang LW, Zhao X, Ma J (2014) A new neural network model for the state-of-charge estimation in the battery degradation process. Appl Energy 121:20–27

Khawaja T, Vachtsevanos G, Wu B (2005) Reasoning about uncertainty in prognosis: a confidence prediction neural network approach. In: Paper presented at the annual meeting of the north American fuzzy information processing society, Ann Arbor, Michigan, USA, 22–25 June 2005

Khosravi A, Nahavandi S, Creighton D et al (2011) Comprehensive review of neural network-based prediction intervals and new advances. IEEE Trans Neural Networks 22 (9):1341–1356

Khosravi A, Nahavandi S, Creighton D (2013) Quantifying uncertainties of neural network-based electricity price forecasts. Appl Energy 112:120–129

Krogh A, Vedelsby J (1995) Neural network ensembles, cross validation, and active learning. In: Tesauro G, Touretzky D, Leen T (eds) Advances in neural information processing systems, vol 7. The MIT Press, Massachusetts, pp 231–238

Lawrence N, Seeger M, Herbrich R (2003) Fast sparse Gaussian process methods: The information vector machine. In: Becker S, Thrun S, Obermayer K (eds) Advances in neural information processing systems, vol 15. MIT Press, Massachusetts, pp 625–632

Lawrence S, Giles CL, Tsoi AC (1996) What size neural network gives optimal generalization? convergence properties of backpropagation. In: Technical report UMIACS-TR-96-22 and CS-TR-3617. Institute for Advanced Computer Studies. https://clgiles.ist.psu.edu/papers/UMD-CS-TR-3617.what.size.neural.net.to.use.pdf. Accessed 29 May 2016

Li D, Wang W, Ismail F (2013) Enhanced fuzzy-filtered neural networks for material fatigue prognosis. Appl Soft Comput 13(1):283–291

Liu A, Dong M, Peng Y (2012a) A novel method for online health prognosis of equipment based on hidden semi-Markov model using sequential Monte Carlo methods. Mech Syst Signal Process 32:331–348

Liu D, Pang J, Zhou J et al (2012b) Data-driven prognostics for lithium-ion battery based on Gaussian process regression. In: Paper presented at the 2012 prognostics and system health management conference, Beijing, China, 23–25 May 2012

Liu J, Saxena A, Goebel K et al (2010) An adaptive recurrent neural network for remaining useful life prediction of lithium-ion batteries. In: Paper presented at the annual conference of the prognostics and health management society, Portland, Oregon, USA, 10–16 Oct 2010

Liu P, Li H (2004) Fuzzy neural network theory and application. World Scientific, Singapore

MacKay DJC (1997) Gaussian processes-A replacement for supervised neural networks? http://www.inference.eng.cam.ac.uk/mackay/gp.pdf. Accessed 29 May 2016

Mackay DJC (1998) Introduction to Gaussian processes. NATO ASI Ser F Comput Syst Sci 168:133–166

Mao KZ, Tan K-C, Ser W (2000) Probabilistic neural-network structure determination for pattern classification. IEEE Trans Neural Networks 11(4):1009–1016

Melkumyan A, Ramos F (2009) A sparse covariance function for exact Gaussian process inference in large datasets. In: Paper presented at the 21st international joint conference on artificial intelligence, Pasadena, California, USA, 11–17 July 2009

Mohanty S, Das D, Chattopadhyay A et al (2009) Gaussian process time series model for life prognosis of metallic structures. J Intell Mater Syst Struct 20:887–896

Naftaly U, Intrator N, Horn D (1997) Optimal ensemble averaging of neural networks. Network: Comput Neural Syst 8(3):283–296
Nawi NM, Ransing RS, Ransing MR (2008) An improved conjugate gradient based learning algorithm for back propagation neural networks. Int J Comput Intell 4(1):46–55
Neal RM (1998) Regression and classification using Gaussian process priors. In: Bernardo JM, Berger JO, Dawid AP et al (eds) Bayesian statistics, vol 6. Oxford University Press, New York, pp 475–501
Orr MJL (1996) Introduction to radial basis function networks. http://www.cc.gatech.edu/~isbell/tutorials/rbf-intro.pdf. Accessed 29 May 2016
Ostafe D (2005) Neural network hidden layer number determination using pattern recognition techniques. In: Paper presented at the 2nd Romanian-Hungarian joint symposium on applied computational intelligence, Timisoara, Romania, 12–14 May 2005
Paciorek CJ, Schervish MJ (2004) Nonstationary covariance functions for Gaussian process regression. Adv Neural Informa Process Syst 16:273–280
Pandey MD, Noortwijk JMV (2004) Gamma process model for time-dependent structural reliability analysis. In: Paper presented at the second international conference on bridge maintenance, safety and management, Kyoto, Japan, 18–22 Oct 2004
Parzen E (1962) On estimation of a probability density function and mode. Ann Math Stat 33 (3):1065–1076
Rasmussen CE, Williams CKI (2006) Gaussian processes for machine learning. The MIT Press, Massachusetts
Rivals I, Personnaz L (2000) Construction of confidence intervals for neural networks based on least squares estimation. Neural Networks 13(4–5):463–484
Rovithakis GA, Maniadakis M, Zervakis M (2004) A hybrid neural network/genetic algorithm approach to optimizing feature extraction for signal classification. IEEE Trans on Syst Man Cybernetics-Part B: Cybernet 34(1):695–702
Salomon R, Hemmen JLV (1996) Accelerating Backpropagation through dynamic self-adaptation. Neural Networks 9(4):589–601
Sang H, Huang JZ (2012) A full scale approximation of covariance functions for large spatial data sets. J R Stat Soc: Ser B (Statistical Methodology) 74(1):111–132
Seeger M (2004) Gaussian processes for machine learning. Int J Neural Syst 14(2):69–106
Sheela KG, Deepa SN (2013) Review on methods to fix number of hidden neurons in neural networks. Math Probl Eng 2013:1–11
Si XS, Wang W, Hu CH et al (2011) Remaining useful life estimation—A review on the statistical data driven approaches. Eur J Oper Res 213:1–14
Si XS, Wang W, Hu CH et al (2013) A Wiener process-based degradation model with a recursive filter algorithm for remaining useful life estimation. Mech Syst Signal Process 35(1–2):219–237
Silva RE, Gouriveau R, Jemeï S et al (2014) Proton exchange membrane fuel cell degradation prediction based on adaptive neuro-fuzzy inference systems. Int J Hydrogen Energy 39 (21):11128–11144
Soares S, Antunes CH, Araújo R (2013) Comparison of a genetic algorithm and simulated annealing for automatic neural network ensemble development. Neurocomputing 121:498–511
Specht DF (1990) Probabilistic neural networks. Neural Networks 3:109–118
Subudhi B, Jena D, Gupta MM (2008) Memetic differential evolution trained neural networks for nonlinear system identification. In: Paper presented at the IEEE region 10 colloquium and the third international conference on industrial and information systems, Kharagpur, India, 8–10 Dec 2008
Svozil D, Kvasnička V, Pospíchal J (1997) Introduction to multi-layer feed-forward neural networks. Chemometr Intell Lab Syst 39:43–62
Tipping ME (2001) Sparse Bayesian learning and the relevance vector machine. J Mach Learn Res 1:211–244
Toal DJJ, Bressloff NW, Keane AJ (2008) Kriging hyperparameter tuning strategies. AIAA J 46 (5):1240–1252

Veaux RD, Schumi J, Schweinsberg J et al (1998) Prediction intervals for neural networks via nonlinear regression. Technometrics 40(4):273–282

Wilamowski BM, Iplikci S, Kaynak O et al (2001) An algorithm for fast convergence in training neural networks. Proc Int Joint Conf Neural Netw 3:1778–1782

Xiong Y, Chen W, Apley D, Ding X (2007) A non-stationary covariance-based Kriging method for metamodelling in engineering design. Int J Numer Meth Eng 71(6):733–756

Yu H, Wilamowski BM (2011) Levenberg–marquardt training. Industrial Electron Handb 5(12): 1–16

Zio E, Maio FD (2010) A Data-driven fuzzy approach for predicting the remaining useful life in dynamic failure scenarios of a nuclear system. Reliab Eng Syst Saf 95:49–57

Zhang G, Yuen KKF (2013) Toward a hybrid approach of primitive cognitive network process and particle swarm optimization neural network for forecasting. Procedia Comput Sci 17:441–448

Chapter 6
Study on Attributes of Prognostics Methods

6.1 Introduction

In this chapter, the attributes of various prognostics algorithms introduced in Chaps. 4 and 5 are discussed in details. Specifically, the following five algorithms will be considered in this chapter: three physics-based methods including nonlinear least squares (NLS), Bayesian method (BM) and particle filter (PF), and two data-driven methods including Gaussian process (GP) regression and neural network (NN). The goal is to provide guidelines so that the readers can choose an appropriate algorithm for their field of application.

Since there are many variations of each algorithm, the most common and basic ones are employed to understand each algorithm's attributes. Therefore, the five MATLAB codes that were provided in Chaps. 4 and 5 will be utilized in this chapter. Each algorithm is tested using crack growth problems including simple and complex degradation behaviors, which are explained in Sect. 6.2. Their attributes, advantages, and disadvantages are discussed based on the logic of their algorithms. The choice of the crack growth problem is because the damage growth model is available, and hence, it enables comparison of all the algorithms. The discussions are not limited to the crack growth problem and can be interpreted in a general context, since the results are caused by the algorithms' inherent attributes.

Even if all prognostics algorithms have the same goal of predicting the future behavior of degradation, different prognostics algorithms have different characteristics. For example, in physics-based methods, some algorithms use an explicit form of degradation as a function of model parameters, loading/usage condition and time/cycle. On the other hand, some algorithms use a differential equation form of degradation model, where degradation level can be obtained by integrating the differential equation. The former is referred to as a total form, while the latter is an incremental form. In the case of battery degradation model in Chap. 4, Eq. 4.2 is an example of a total form degradation model and Eq. 4.16 is an incremental form. Although the total and incremental forms are mathematically equivalent, they are

N.-H. Kim et al., *Prognostics and Health Management of Engineering Systems*,
DOI 10.1007/978-3-319-44742-1_6

different in numerical implementation. In particular, the incremental form will have an error in time integration. Therefore, it is important to make sure that the time interval of health monitoring should be small enough so that the error in integrating the differential equation is in an acceptable range. Otherwise, the incremental form of physical model has to be integrated with a different time interval than the interval of health monitoring.

An important question in physics-based approaches is what if the degradation model is not perfect; that is, how to handle model-form errors. For example, in the case of Paris model for crack growth, the original model is presented for an infinite flat panel under mode I fatigue loading condition. In reality, however, there is no infinite flat panel; all panels are in a finite size, and in airplanes most panels are not flat. Unfortunately, many model parameters are not intrinsic material properties; they depend on boundary conditions and loading conditions. This is why the model parameters that are estimated in laboratory tests are different from the model parameters of the panel used in an actual system. Therefore, it is important how physics-based prognostics algorithms can correct or compensate for the model-form error.

In physics-based prognostics, it is important to understand how different algorithms identify model parameters, especially uncertainty in model parameters and correlation between them. Some algorithms represent this uncertainty in the form of samples, while other algorithms assume Gaussian distribution with estimated variance. In addition, the prior knowledge on model parameters plays an important role in the performance of prognostics algorithm. Bayesian inference-based algorithms have a systematic way of considering the prior knowledge, but regression-based algorithms do not have a feasible way of considering it. Even if Bayesian inference-based algorithms have the same theoretical basis, numerical implementation can yield quite different prediction outcomes. An example is the Bayesian method versus particle filter where both methods are based on Bayesian inference but their prediction process and outcomes are quite different. The attributes of physics-based algorithms will be discussed in Sect. 6.3.

In the first look, data-driven approaches seem to be more attractive than physics-based approaches because they do not require accurate physical model. In fact, many complex systems, there is no good physical model available to describe the degradation behavior. However, data-driven approaches have their own issues. It sounds like data-driven approaches do not use a model, but in reality data-driven approaches also employ a mathematical model to represent the degradation behavior. The difference is that the mathematical model does not have any physical meaning. However, since a mathematical model is used to describe the behavior of degradation, data-driven approaches also have a similar question as with physics-based approaches; is the mathematical model good enough to represent the behavior of degradation? Of course, a straightforward answer to this question is to use a complex model so that any kinds of degradation behavior can be modeled. However, a complex model comes with many parameters, which need to be identified using measured data. A model with many parameters causes many challenges because it requires more data and it is difficult to identify them unless the

mode is linear with respect to the parameters. Also, when the available number of data is not enough, it often causes an overfitting phenomenon. In the theory of regression, it is normally suggested to have about 2–3 times more number of data than the number of unknown coefficients. For example, if a model is composed of polynomial function with 10 coefficients and if 10 data are available, it is possible to find 10 coefficients and the model exactly passes the 10 data points, but it does not mean the model is accurate at prediction points. Therefore, data-driven approaches move the decision of selecting model form to users. In order to find an appropriate form of mathematical model in data-driven approaches, it requires in-depth understandings of the degradation behavior as well as data-driven prognostics algorithms. The attributes of data-driven algorithms will be discussed in Sect. 6.4.

Using a complex crack growth example, both physics-based and data-driven prognostics algorithms will be compared in Sect. 6.5. In general, physics-based approaches always have advantage if an accurate physical model is available. This is because physics-based algorithms usually require a less number of data to identify model parameters. However, it would be informative to estimate how many data would be required by the data-driven approaches to produce a similar level of accuracy with physics-based algorithms so that the users can estimate how many data would be required.

6.2 Problem Definition

Physics-based approaches assume that a physical model describing the behavior of degradation is available with usage conditions such as loading information. Fatigue crack growth problem is the most common example used in these approaches since the physical models are relatively well developed compared to other failure mechanisms. In this section, two different physical models, the Paris model (Paris and Erdogan 1963) and Huang's model (Huang et al. 2008), are employed to describe the crack growth behavior under constant and variable amplitude loading conditions, respectively.

6.2.1 Paris Model for Fatigue Crack Growth

The Paris model introduced in Sects. 2.1 and 4.5 describes the rate of crack growth under constant amplitude cyclic mode I loading of an infinite plate, as shown in Fig. 6.1. A common application is a fatigue crack growth in fuselage panel of airplanes. Due to repeated pressurizations during take-off and landing, fuselage panels experience fatigue loading. When the stress range due to the pressure differential is $\Delta\sigma$, the rate of crack growth can be written using the Paris model (Paris and Erdogan 1963) as

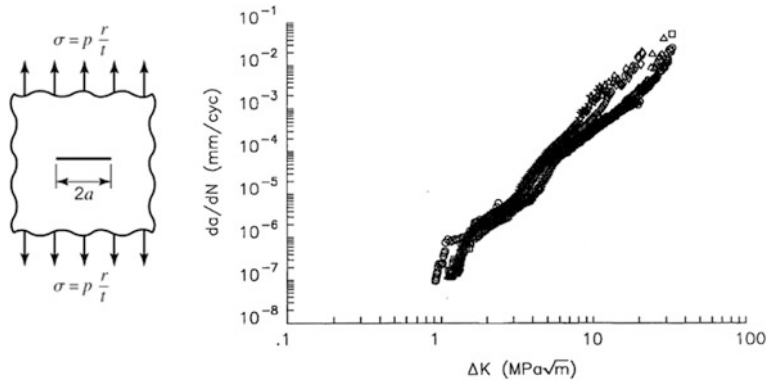


Fig. 6.1 Fatigue crack growth under mode I loading (Paris model)

$$\frac{\mathrm{d}a}{\mathrm{d}N} = C\left(\Delta\sigma\sqrt{\pi a}\right)^m, \tag{6.1}$$

where a is a half crack size shown in Fig. 6.1, m and C are model parameters, N is the number of loading cycles, and $\Delta K = \Delta\sigma\sqrt{\pi a}$ is the range of stress intensity factor. Although the number of cycles N is a discrete number, it is considered as a continuous number. Equation 6.1 is a differential equation, which needs to be integrated to obtain a crack size as a function of loading cycles. In order to integrate Eq. 6.1, the initial condition (initial crack size, a_0) is required. When the initial crack size is also unknown, it has to be included as unknown parameters. Therefore, the Paris model in Eq. 6.1 has three unknown parameters (m, C, a_0).

The slope in Fig. 6.1 corresponds to exponent m, while C is the y-intercept at $\Delta K = 1$. Note that the figure is in log–log scale. It is well known that the stress intensity factor is the main contributor to fatigue crack growth. In the case of airplanes, the fuselage is often approximated by a cylinder, and the range of hoop stress can be calculated using the pressure differential as $\Delta\sigma = \Delta p \cdot r/t$, where r and t are radius and thickness of fuselage panel, respectively.

Note that the above physical model is given in a rate form. A total form of physical model can be obtained by integrating Eq. 6.1 and solving for a as

$$a(N) = \left[NC\left(1 - \frac{m}{2}\right)\left(\Delta\sigma\sqrt{\pi}\right)^m + a_0^{1-\frac{m}{2}}\right]^{\frac{2}{2-m}}, \tag{6.2}$$

where a_0 is the initial half crack size, a(N) is the half crack size at the number of cycles N. In aluminum material, for example, the exponent m is in the range of 3.0–4.0. Therefore, the exponent in Eq. 6.2 is negative. In addition, the first term in the bracket is negative, while the second term is positive. In order to make the crack size to be a real number, it is necessary that the term in the bracket should be positive. As the number of cycles N increases, the term in the bracket becomes negative and the crack size a becomes a complex number, which is physically not possible. This

happens because the crack will grow faster as N increases. That is, the crack growth becomes unstable and the crack size goes infinity. Therefore, Eq. 6.2 is valid only up to $N_{\max}$ cycles that makes the term inside the bracket positive.

In practice, the crack is considered unstable when the crack grows in a size called a critical crack size. A critical crack size is defined when the stress intensity factor reaches the fracture toughness, K_{IC}, which describes the ability of a material containing a crack to resist fracture, and is one of the most important properties of any material for many design applications. In the case of mode I loading, the critical crack size is defined as

$$a_C = \left(\frac{K_{IC}}{\Delta\sigma\sqrt{\pi}}\right)^2. \tag{6.3}$$

The model parameters govern the behavior of crack growth, which need to be identified using measured crack size data at different loading cycles with given loading information, $\Delta\sigma$. As shown in Fig. 6.1, the test results show some level of variability. Therefore, it is expected that the identified model parameters may show a statistical distribution. The crack size at future cycles can be predicted by substituting the identified parameters to the Paris model with future loading conditions. Therefore, algorithms of physics-based approaches can be considered as a parameter estimation method.

For simplicity, the effect of finite plate size and the curvature of the plate is ignored in the above Paris model. However, these can be the source of model-form error, which is an important issue in physics-based prognostics.

6.2.2 *Huang's Model for Fatigue Crack Growth*

The basic assumption in the Paris model is that the fatigue loading is in constant amplitude. Therefore, the range of stress, $\Delta\sigma$, or the range of stress intensity factor, ΔK, is good enough to describe loading conditions. In practice, however, loadings with different amplitudes are often applied to the system, which makes the crack growth much more complicated than the Paris model. Huang et al. (2008) proposed a fatigue crack growth model under variable amplitudes using an equivalent stress intensity factor and a modified Wheeler model. This method leads to a fatigue life prediction model that depends mainly on the stress ratio and the plastic zone size ahead of the crack tip. An important capability of this model is that it can describe the phenomena of retardation and arrest due to overload, and the acceleration due to a state of underload following an overload. In this chapter Huang's fatigue model is used as a complex physical model, compared to the Paris model.

In Huang's model, the governing differential equation for crack growth is defined as

$$\frac{da}{dN} = C\left[(\Delta K_{eq})^m - (\Delta K_{th})^m\right], \tag{6.4}$$

where C and m are model parameters and ΔK_{th} is the threshold of stress intensity factor range. In practice, since the crack size cannot be reduced, the right-hand side of Eq. 6.4 must be positive. That is, when $\Delta K_{eq} < \Delta K_{th}$, it is considered that $da/dN = 0$. In Eq. 6.4, the range of equivalent stress intensity factor includes the effect of crack tip plasticity as well as crack closure after overloading, and is defined as

$$\Delta K_{eq} = M_R M_P \Delta K, \quad \Delta K = Y \Delta\sigma\sqrt{\pi a}, \tag{6.5}$$

where Y is the geometry factor related with the specimen geometry, and M_R and M_P, respectively, are scaling and correction factors for the loading ratio and the loading sequence interaction. These factors are defined as

$$M_R = \begin{cases} (1-R)^{-\beta_1} & (-5 \le R < 0) \\ (1-R)^{-\beta} & (0 \le R < 0.5) \\ (1.05 - 1.4R + 0.6R^2)^{-\beta} & (0.5 \le R < 1) \end{cases}, \tag{6.6}$$

$$M_P = \begin{cases} \left(\frac{r_y}{a_{OL} + r_{OL} - a - r_\Delta}\right)^n & \left(a + r_y < a_{OL} + r_{OL} - r_\Delta\right) \\ 1 & \left(a + r_y \ge a_{OL} + r_{OL} - r_\Delta\right) \end{cases}, \tag{6.7}$$

$$r_y = \alpha\left(\frac{K_{\max}}{\sigma_y}\right)^2 \quad \text{and} \quad r_{OL} = \alpha\left(\frac{K_{\max}^{OL}}{\sigma_y}\right)^2,$$

$$\alpha = 0.35 - \frac{0.29}{1 + \left[1.08 K_{\max}^2/(t\sigma_y^2)\right]^{2.15}}, \tag{6.8}$$

where $R = \sigma_{\max}/\sigma_{\min}$ is the load ratio; $\sigma_{\max}$ and $\sigma_{\min}$, respectively, are the maximum and minimum stress at every cycle; β and β_1 are shaping exponents; r_y is the plastic zone size ahead of the crack tip; r_Δ is the increment in the plastic zone size ahead of the crack tip caused by under loading, which is ignored here since there is no under loading in this example; n is the shaping exponent in the present model; α is the plastic zone size factor; $K_{\max}$ is maximum stress intensity factor; σ_y is tensile yield stress; and OL placed on subscript or superscript is the case at which the overload occurs.

Note that there is no total form corresponding to Huang's model because analytical integration of 6.4 is not available for general variable loadings. The rate form of physical model in Eq. 6.4 can be calculated using different integration algorithms. In this section, a simple forward Euler integration method is used to calculate the crack size at a given cycle. The following MATLAB code **[Huang]** can be used to calculate crack sizes for given model parameters. In the code **[Huang]**, problem definition is already filled to generate data in Set 1 in Sect. 6.2.3. It is assumed that the plate has a 4 mm-thickness and 150 mm-width (line 12), and a

geometry factor for finite plate is employed (lines 30–31). Other than these, the code **[Huang]** can be understood with Eqs. 6.4 and 6.8.

[Huang]: MATLAB Code for Huang's Model

```
%=== PROBLEM DEFINITION (Loading and Model Parameters) ======
dN=1;                              % transition cycle interval
dtMeasu=1000;                     % measurement cycle interval
dSmin=5;                                          % min. load
sNomin=65;                                     % nominal load
sOverl=[125 100 100 125 125];                      % overload
cNomin=[5 5 5 5 45]; % num. of cycle in a block: nominal load
cOverl=[45 45 95 95 5];  % num. of cycle in a block: overload
cBlock=[5000 10000 15000 20000 25000];% cycles for each block
a0=0.01;          % true model param including initi crack, a0
m=3.1; C=log(5.5e-11); dKth=5.2; beta=0.2; n=2.8; sY=580;
thick=4e-3; width=150e-3/2;               % geometry dimension
%=============================================================
cycle=[0:dtMeasu:cBlock(end)]';
% % % LOAD RATIO, R
nb=length(cBlock);
cInter(1)=cBlock(1);
if nb>1; for i=2:nb; cInter(i)=cBlock(i)-cBlock(i-1); end;end
dSmax=[];
for i=1:nb;
 B=[sNomin*ones(cNomin(i),1); sOverl(i)*ones(cOverl(i),1)];
 dSmax(length(dSmax)+1:cBlock(i),1)= ...
                repmat(B,cInter(i)/(cNomin(i)+cOverl(i)),1);
end;
dSmax=dSmax(1:dN:end); R=dSmin./dSmax;
% % % HUANG's MODEL
aCurre(1)=a0; akOL(1)=a0; rOL(1)=0;
for k=1:max(cycle)/dN;
 ak=aCurre(k);
 Y=(1-ak./(2*width)+0.326.*(ak./width).^2)...%geometry factor
  ./sqrt(1-ak./width); loca=ak./width>1; Y(loca)=0;
 Kmax=Y.*dSmax(k).*sqrt(pi*ak);              %SIF from Eq. 6.5
 Kmin=Y.*dSmin.*sqrt(pi*ak);
 % Scaling factor, Mr (Eq. 6.6)
 Mr=(1-R(k)).^(-beta);
 % Correction factor, Mp (Eqs. 6.7 and 6.8)
 alpha=0.35-0.29./(1+(1.08.*Kmax.^2./(thick.*sY.^2)).^2.15);
 ry=alpha.*(Kmax./sY).^2;
 if R(k)~=dSmin/sNomin;
  loca=ry>rOL; rOL(loca)=ry(loca); akOL(loca)=ak(loca);
 end
 Mp=(ry./(akOL+rOL-ak)).^n; loca=ak+ry>=akOL+rOL; Mp(loca)=1;
 % Calculate crack size
 dKeq=Mr.*Mp.*(Kmax-Kmin);                          % Eq. 6.5
 rate=exp(C).*(dKeq.^m - dKth.^m);                  % Eq. 6.4
 loca=dKeq<=dKth; rate(loca)=0;
 aCurre(k+1)=aCurre(k)+rate.*dN;
end;
crackHuang=aCurre(1:dtMeasu/dN:max(cycle)/dN+1);
% % % DATA PLOT
plot(cycle,crackHuang);
```

The shaping exponent β_1 in Eq. 6.6 is not considered since the load ratio, R is between 0 and 0.5 in the following example. Also, uncertainties in loading, thickness, and width of plate are ignored. Therefore, m, C, ΔK_{th}, β, n, and σ_Y are selected as model parameters in this example. As expected, Huang's model can describe a complicated behavior of crack growth under variable amplitude loadings. However, the number of model parameters increases to six, compared to two in the Paris model. Therefore, it would be more difficult to identify parameters from Huang's model. In addition, the correlation structure between parameters would be more complicated than that of Paris model.

6.2.3 Health Monitoring Data and Loading Conditions

Instead of measured data from health monitoring system, synthetic data are used in this chapter. Synthetic data can be generated based on the following three steps:

(1) The true values of model parameters and loading conditions are assumed
(2) The true parameters and loading conditions are substituted in the physical model at different times to generate true degradation data
(3) Random noise and deterministic bias are added to the true degradation data.

There are several advantages of synthetic data. First, it is possible to compare the identified parameters from data with the true parameters. Second, since the true degradation data are given, it is possible to evaluate the accuracy of the predicted remaining useful life. Finally, it is possible to control the level of noise and bias such that their effect can be studied.

The following true parameters are used to generate synthetic data:

Paris model parameters: $m = 3.8$, $C = 1.5\text{e}{-10}$
Huang's model parameters:

$$m = 3.1,\ C = 5.5\text{e} - 11,\ \Delta K_{th} = 5.2,\ \beta = 0.2,\ n = 2.8,\ \sigma_Y = 580.$$

For the Paris model (simple model), ten sets of true crack growth data are generated using the above model parameters with an interval of 100 cycles under ten different loading cases. These ten sets are generated by gradually increasing the stress range from 65 to 83 MPa with 2 MPa increments. Results are given in Fig. 6.2a, in which the 26 data points are available in a set until the end of 2500 cycles, and ten sets are available with the set ID denoted by the number in a circle. Set 1 corresponds to $\Delta\sigma = 65$ MPa, while Set 10 corresponds to $\Delta\sigma = 83$ MPa. For the critical crack size, $a_C = 0.05$ m is used throughout all cases.

For Huang's model (complex model), ten data sets are given in Fig. 6.2b, which exhibit crack retardation and acceleration due to complex loadings as defined by Fig. 6.3. In the figure, a loading block is composed of the minimum load fixed at a

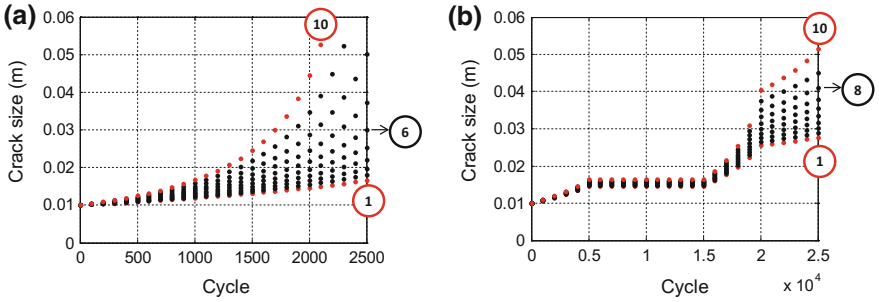
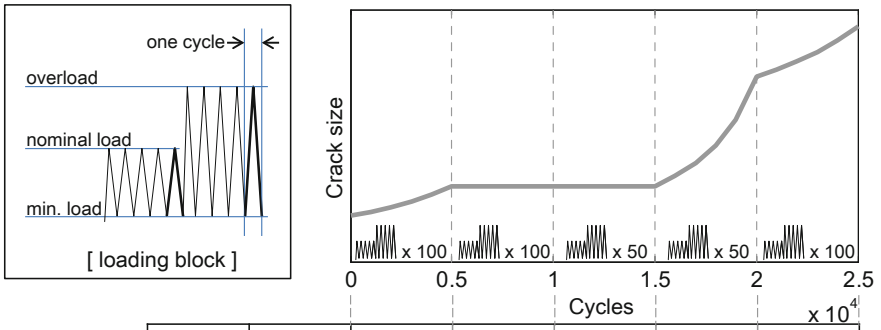


Fig. 6.2 Ten data sets for crack growth from two different models. **a** simple model (Paris model with two parameters), **b** complex model (Huang's model with six parameters)



overload	MPa (# cycles)	125~134 (45 cycles)	100~109 (45 cycles)	100~109 (95 cycles)	125~134 (95 cycles)	125~134 (5 cycles)
nominal load	MPa (# cycles)	65~74 (5 cycles)	65~74 (5 cycles)	65~74 (5 cycles)	65~74 (5 cycles)	65~74 (45 cycles)
min. load	MPa	5				

Fig. 6.3 Variable load amplitude spectrum for Huang's model

constant of 5 MPa and two maximum loads (we call these nominal load and overload, respectively) varying in terms of cycles. The loading block is applied with five different scenarios in terms of the cycles as specified in the table, each block lasting over the period of 5000 cycles. In each of the scenarios, load ranges are given, all with the width of 10 MPa, which represent the values of ten loading cases with increment of 1.0 MPa. For example, the first loading case consists of a loading block of the nominal load 65 MPa and overload 125 MPa lasting 5 and 45 cycles, respectively, during the period of 0–5000 cycles. This is applied the same way, as indicated in the table, over the rest of the period until it reaches 25,000 cycles. Other loading cases are made by increasing the load magnitude with 1.0 MPa increments in each of the scenarios, which makes ten loading sets, and yields the crack growth results in Fig. 6.2b.

Figure 6.2b shows a complicated behavior of crack growth. The rate of crack growth is different at different loading blocks and the crack does not grow at all between 5000 and 15,000 cycles. Due to the high amplitude of overloading up to 5000 cycles, retardation (crack arrest) occurs between 5000 and 15,000 cycles, where crack does not grow. After 15,000 cycles, the crack grows fast because underloads are followed by overloads. The crack grows slowly after 20,000 cycles because the number of cycles for overloads is reduced. These data are available in the companion website.[1] In the companion website, CrackData.m file stores matrices of these data. In CrackData.m file, the array CrackSimple is 10×26 array that stores 10 sets of crack size data from the Paris model, while the array CrackHuang is 10×26 array from the Huang's model. Note that these arrays store exact crack size data.

In order to simulate measured crack size data, different levels of noise and bias are added to the true crack sizes that are calculated from the model with true parameters. A deterministic bias of -3mm is added first, after which random noise is uniformly distributed between $-v$ mm and $+v$ mm, where v of 0, 1, and 5 mm are considered. For data-driven approaches, Set 6 and Set 8 are considered as the prediction sets in Paris and Huang's model, respectively, while other sets are considered as training sets. The following MATLAB commands add the above-mentioned noise and bias:

```
b=-0.003; v=0.001;
crackSimpleMeas=crackSimple + v*(2*rand(10,26)-1) + b;
crackHuangMeas=crackHuang + v*(2*rand(10,26)-1) + b;
```

6.3 Physics-Based Prognostics

In this section, physics-based prognostics algorithms are compared in terms of accuracy in parameter estimation and RUL prediction. In particular, focus is on the uncertainty and correlation in model parameters as well as model-form errors. For BM and PF in physics-based approaches, a normal distribution is employed for the likelihood function, and the prior distributions of unknown parameters (model parameters and noise) are assumed as listed in Table 6.1. The distributions of parameters are represented using 5000 samples in the three methods, NLS, BM, and PF, and future damage growth and RUL are predicted by applying individual samples to the physical model. In the case of NLS, the deterministic values of parameters and their covariance matrix are identified from optimization. Then 5000 samples are generated from Student's t-distribution, which is multiplied by the standard deviation that is the square root values of diagonal terms in the covariance

[1]All MATLAB codes and data used in the book can be found in the companion website http://www2.mae.ufl.edu/nkim/PHM/.

Table 6.1 Prior distributions of model parameters and noise

Paris model	$f(m) = U(3.3, 4.3)$ (true: 3.8)	$f(\ln C) = U(\ln(5 \times 10^{-11}), \ln(5 \times 10^{-10}))$ (true: 1.5e-10)
Huang's model	$f(m) = U(2.8, 3.4)$ (true: 3.1) $f(\Delta K_{th}) = U(2, 8)$ (true: 5.2) $f(n) = U(1.5, 4)$ (true: 2.8)	$f(\ln C) = U(\ln(3 \times 10^{-11}), \ln(8 \times 10^{-11}))$ (true: 5.5e-11) $f(\beta) = U(0.1, 0.4)$ (true: 0.2) $f(\sigma_Y) = U(400, 600)$ (true: 580)
Noise	$f(\sigma) = U(0, 0.02)$	
Initial crack size	$f(a_0) = U(0.008, 0.012)$ (true: 0.01)	

matrix and added by the identified parameters as a mean. In the case of BM, 6250 MCMC samples are generated based on the posterior joint PDF, and then first 1250 samples are removed as burn-in samples. In the case of PF, 5000 particles are generated from the initial distributions of each parameter as the prior PDFs. The samples as the prior PDFs are evaluated by the likelihood function, and weight values are assigned to the samples. The joint PDF is obtained after resampling based on the evaluated weights.

6.3.1 Correlation in Model Parameters

Since a physical model that describes the degradation behavior is available in physics-based approaches, the performance of different algorithms can be evaluated based on their ability to identify model parameters. This is reasonable because the prediction of RUL in physics-based approaches is simply substituting samples of model parameters in the physical model to obtain samples of RUL. Therefore, main focus on physics-based prognostics is how to identify model parameters accurately with a less number of measured data.

One of the most challenging aspects of model parameter identification is the correlation between parameters. In general, due to uncertainty in model form and noise in data, model parameters are identified as a probability distribution. It is possible to estimate the probability distribution of individual parameters, but this is not enough in general. Not only probability distribution, but also the correlation between different parameters should be identified. Without properly identifying the correlation, the predicted RUL can be significantly different from the real one.

An important observation with correlated parameters is that even if the accurate value of each parameter may not be identified, the prediction results of degradation and RUL can be reliable when all combinations of the correlated parameters can yield the same prediction results. Before we compare three physics-based algorithms, the attributes of the correlation are addressed with the Paris model.

6.3.1.1 Correlation Between Model Parameters

As shown in Fig. 6.4a, It is well known that Paris model parameters, m and C, are strongly correlated, and parameters can be identified with their correlation rather than individual values. In log–log scale graph shown in Fig. 6.4a, the exponent m is the slope and C is y-intercept at $\Delta K = 1$. Therefore, when a crack growth rate is measured at a given ΔK, it is possible to generate multiple lines that pass through the measured point as shown in Fig. 6.4a. This is the basic property of correlated parameters. It would be interesting if a functional relationship can be found between parameters; in other words, if their correlation is unique under given usage condition. In this text, the usage condition and loading condition mean the applied load. In the case of crack growth, it means the range of stress intensity factor, ΔK, or the range of stress, $\Delta\sigma$. If the correlation changes over cycles, it is also doubted how the prediction results that are based on current correlation can be affected. To find the answer to this question, correlations between m and $\ln(C)$ are considered, which can be obtained by taking logarithm of Eq. 6.1 as

$$\ln(C) + m\,\ln\left(\Delta\sigma\sqrt{\pi a}\right) = \ln(\mathrm{d}a/\mathrm{d}N). \tag{6.9}$$

Since a crack growth rate at a cycle can be expressed by a combination of m and $\ln(C)$ (see Fig. 6.4a), the same crack growth rate in Eq. 6.9 can also be expressed with the true model parameters as

$$\ln(C_{\text{true}}) + m_{\text{true}}\,\ln\left(\Delta\sigma\sqrt{\pi a}\right) = \ln(\mathrm{d}a/\mathrm{d}N). \tag{6.10}$$

By subtracting Eq. 6.10 from Eq. 6.9, the following equation can be obtained:

$$\ln(C) = \ln(C_{\text{true}}) + (m_{\text{true}} - m)\,\ln\left(\Delta\sigma\sqrt{\pi a}\right). \tag{6.11}$$

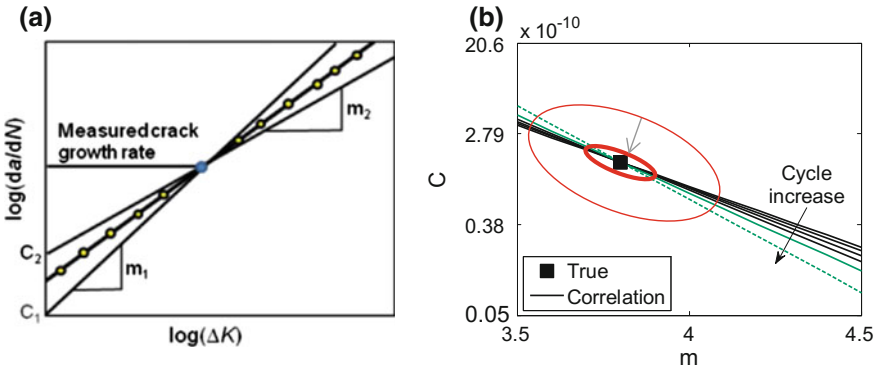


Fig. 6.4 Illustration of correlation attribute between m and C. **a** same crack growth rate with different combinations of parameters **b** correlation change with cycle

The above equation can be considered as a linear relationship between m and $\ln(C)$ for a given cycle. Note that $m_{\text{true}} = 3.8$, $C_{\text{true}} = 1.5 \times 10^{-10}$ and $\Delta\sigma = 83$ MPa can be considered constants, but the crack size a can change as cycle N increases, which can be calculated from Eq. 6.2 with m_{true} and C_{true}. Therefore, the correlation in Eq. 6.11 may gradually change as crack grows. That is, the relationship between m and $\ln(C)$ is explicit for a given ΔK but it gradually changes as the crack grows.

The correlation between m and $\ln(C)$ can be plotted with respect to cycles by changing the crack size, a. The results are shown in Fig. 6.4b, in which the square marker is the true parameters, and the six different lines represent the correlation between m and $\ln(C)$ based on Eq. 6.11 at different cycles: from 0 to 2500 cycles with interval of 500 cycles (green dashed line is at 2500 cycles). As shown in the figure, the correlation between m and $\ln(C)$ changes as cycle increases. Therefore, there is no deterministic relationship between parameters, and the relationship evolves as the crack size increases.

As the cycle increases, the correlation gradually evolves with the true values at the center, which makes the correlation to be identified as a narrow-banded ellipse as shown in Fig. 6.4b. Therefore, the true parameters can eventually be identified since the correlation converges to the true value by intersecting many correlation lines during the update process as more data are used.

Because of this strong linear correlation, the true values of two model parameters m and C can be obtained with only three exact degradation data (crack size at zero cycle is independent on m and C, therefore three data is required to estimate two model parameters). The exact three data in Set 6 can be calculated from Eq. 6.2 with `N={0, 100, 200}` as a=`{0.010000000000000, 0.010286799119103, 0.010589640425939}`. Note that the exact crack size data have 15 digits of accuracy, which is impractical, but for the moment it is assumed that such an accurate data are available. In this case, even if the three data are in early stage of degradation, accurate parameters as well as accurate RUL can be predicted. The model parameters are identified as $\hat{m} = 3.8$ and $\ln(\hat{C}) = -22.6204$ with very small values of the covariance matrix as

$$\Sigma_{\hat{\boldsymbol{\theta}}} = \begin{bmatrix} 0.1825 & -0.4746 \\ -0.4746 & 1.2339 \end{bmatrix} \times 10^{-18},$$

which makes the degradation prediction very accurate as shown in Fig. 6.5.

In practice, however, most measured data include noise. When degradation data include noise, the linear relationship as in Eq. 6.11 cannot be obtained. To explain the effect of noise in parameter identification, let us consider measured crack size data at two different cycles. Since crack sizes are different, the slopes in correlation Eq. 6.11 are slightly different. Figure 6.6a shows the case when there is no noise in measure crack sizes. In this case, since the two correlation equations have different slopes, it is possible to find an intersection between two lines, which is the true

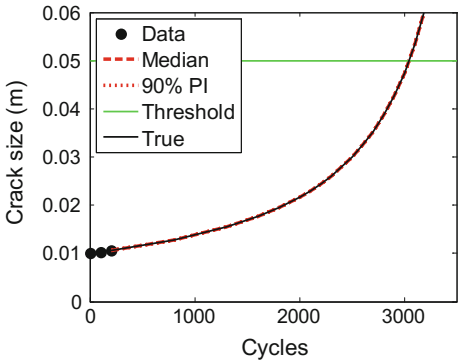


Fig. 6.5 Degradation prediction result from NLS with three exact data

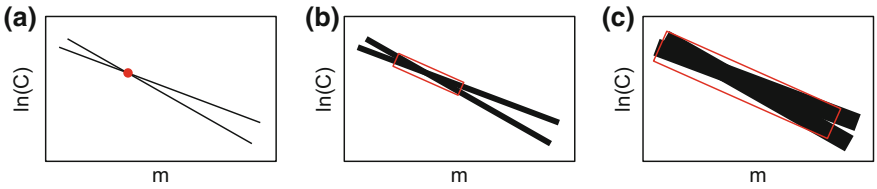


Fig. 6.6 Parameter identification along with the level of noise. **a** small, **b** medium, **c** large

values of the two parameters. In this case, there is no uncertainty, and the two parameters can be identified deterministically.

On the other hand, when data have measurement noise, the correlation relationship in Eq. 6.11 has uncertainty, which can be represented by a line with thickness. The thickness of line is proportional to the level of noise. As shown in Fig. 6.6b, c, when data have measurement noise, the intersection of two correlation lines is not a point, but an area. The model parameters can be identified anywhere inside the area. Therefore, there will be uncertainty in identified model parameters, which can be represented using probability distribution or samples. Note that the uncertainty in identified model parameters increases when the level of noise is large and when the two slopes are not much different. The latter case happens when two measured crack sizes are not much different.

In order to show the sensitivity of parameter identification with a small number of data, the three exact data used in Fig. 6.5 are used after rounding off at fourth decimal places, as a=`{0.0100, 0.0103, 0.0106}`. Therefore, the level of noise is in the order of 10^{-4} m, which is smaller than the capability of modern structural health monitoring systems. Therefore, the error in data can be considered to be very small. Using MATLAB code NLS, Fig. 6.7 shows the samples of identified model parameters and the predicted RUL with confidence intervals. Even if the measured data have very small errors, the identified parameters are distributed

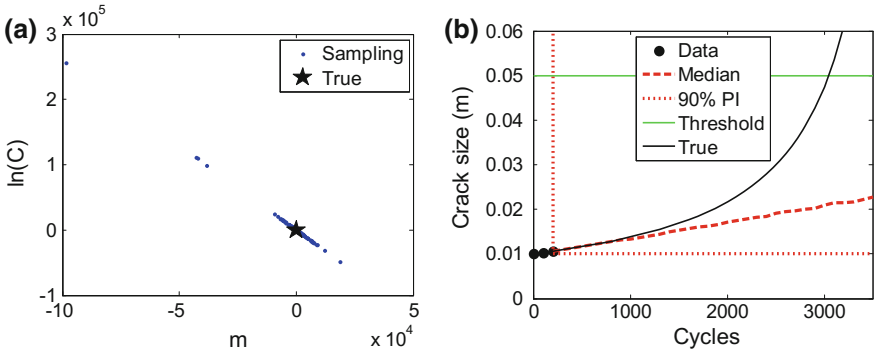


Fig. 6.7 Degradation prediction with exact data up to four decimal places. **a** parameter identification, **b** degradation prediction

in a very wide range. Based on uncertainty in model parameters, the negative values of exponent m happen, which means the crack barely grows. This happens because only three data are used, and the slopes of correlation lines are very close to each other; that is, the crack does not grow much during the three cycles.

Due to a wide range of identified parameters, the predictions in Fig. 6.7b show meaningless results; the 90 % confidence intervals are too wide, and the median failed to follow the trend of the true degradation. As the level of noise increases, the correlation itself cannot be found well. Since measurement error is always involved in the data in practice, more data are required to identify the correlation well, which may depend on the level of noise. In addition, it is important to have data with significantly different crack sizes because the slope in correlation equation in Eq. 6.11 is proportional to crack size. As the crack grows slowly in early stage, it is likely that the uncertainty would be large in early stage. As the crack grows fast, the slope changes significantly, and thus, a better parameter identification can be possible.

6.3.1.2 Correlation Between Model Parameters and Loading Condition

In physics-based approaches, there is also a correlation between model parameters and loading conditions. In Fig. 6.8, the true behavior of crack growth is shown in black solid line, where the correct model parameters and loading condition are used. Note that the two parameters for Paris model represent the slope, m, of the line and y-intercept, C, at $\Delta K = 1$. In this case, loading condition means the range of stress intensity factor, $\Delta K(=\Delta\sigma\sqrt{\pi a})$. If a smaller ΔK than the correct one is used and if the same crack growth is observed from measured data, then the black solid line will shift to the left, as shown in red dashed line in the figure. This can happen by increasing parameter C. That is, in order to predict the same crack growth rate,

different combinations of ΔK and C can be used, which means these two are correlated. More specifically, ΔK and $\ln C$ are negatively correlated.

The notion of this correlation is somewhat different implication than the correlation between parameters, m and C. In the case of correlation between parameters, it was difficult to identify the true values of parameters. However, it was shown that an accurate RUL can be predicted even if the true values of parameters may not be identified. However, in the case of correlation between model parameters and loading conditions, the loading conditions are normally input, not something to be identified. Therefore, the issue occurs when incorrect loading conditions are used in prognostics. If a smaller stress intensity factor than the correct one is used, the two square markers move to two circle markers in the figure. Therefore, m and C_1 that make a line passing through two circle markers are considered as equivalent parameters under incorrect loadings. As long as the loading is constant, this is the same as identifying correlation between (equivalent) model parameters. Consequently, prediction results from physics-based approaches can be accurate even under incorrect loadings because the model parameters are updated to the equivalent parameters to the incorrect loading (smaller one) conditions.

In order to show how prognostics algorithm can identify equivalent model parameters and accurately predict the remaining useful life when an incorrect loading condition is given, data Set 10 for Paris model (simple model) given in Sect. 6.2.3 is employed here. Even if data Set 10 is obtained under 83 MPa loading condition, it is assumed that incorrect loading condition of 65 MPa is used, which is the loading condition of data Set 1. Based on the previous discussion of Paris model given in Fig. 6.8, it is expected that y-intercept C will be identified with a higher value than the true one, while m will remain the same. Figure 6.9 shows the prognostics results using NLS for identified model parameters as well as the remaining useful life. In Fig. 6.9a, the true parameter values are at the star marker location, while blue dots are samples from estimated model parameters with correlation. Even though the samples are distributed due to correlation, the median of m is 3.8020, which is close to the true value, 3.8, while the result of C is higher than the true value. Since the equivalent parameters are identified under incorrect loading conditions, the prediction result can still be accurate as shown in Fig. 6.9b.

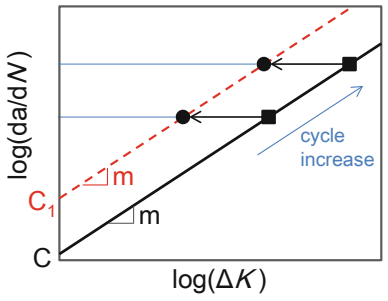


Fig. 6.8 Illustration of equivalent parameters to different loading

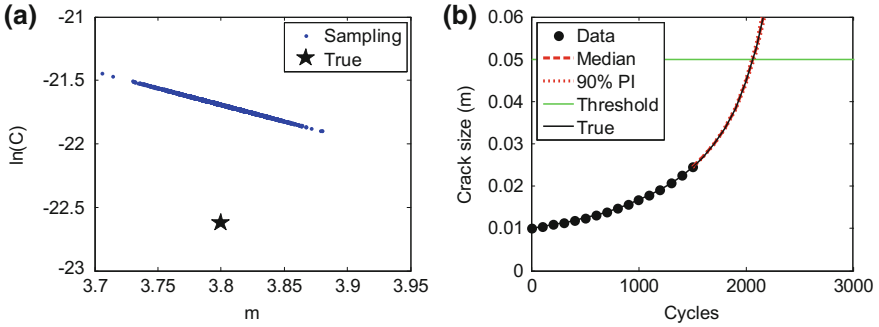


Fig. 6.9 Prediction data Set 10 with incorrect loading condition given in Set 1. **a** parameter identification, **b** degradation prediction

The aspect of equivalent model parameters can further be discussed using the following two logarithms of Paris model:

$$\ln(C) + m\,\ln\left(\Delta\sigma\sqrt{\pi a}\right) = \ln(\mathrm{d}a/\mathrm{d}N),$$

$$\ln(C') + m'\,\ln\left(\Delta\sigma'\sqrt{\pi a}\right) = \ln(\mathrm{d}a/\mathrm{d}N).$$

The first equation is with the correct loading condition $\Delta\sigma = 83\,\mathrm{MPa}$, while the second one is with the incorrect one $\Delta\sigma' = 65\,\mathrm{MPa}$. When the same measured data are used, the crack growth rates on the right-hand side of both equations are the same. The goal is to obtain the relationship between two sets of parameters. As we discussed previously, let us assume that the slope parameter remains the same; i.e., $m = m'$, By equating the above two equations, we have

$$\ln(C^{'}) = \ln(C) + m\,\ln\left(\frac{\Delta\sigma}{\Delta\sigma^{'}}\right). \tag{6.12}$$

Using the true value of $m = 3.8$ and the ratio between loading conditions, the equivalent y-intercept is increased by $\ln(C^{'}) = \ln(C) + 0.93$. In Fig. 6.9a, if the true values of parameters (star marker) are shifted by 0.93 in the vertical direction, it will exactly match with the samples of identified equivalent parameters.

In practical point of view, this correlation works favorable to physics-based prognostics. As mentioned before, the most important issue in physics-based approaches is the model-form error. That is, the actual behavior of degradation is different from the physical model due to simplifications and assumptions imposed in developing the physical model. The error in model-form can often be transferred to an error in loading conditions. For example, in the case of crack growth, the Paris model is considered to be accurate when mode I fatigue loading is applied to a flat plate with infinite size. In reality, there is no infinite plate, and the accuracy of Paris model depends on geometrical effects, boundary conditions, crack shape, and crack

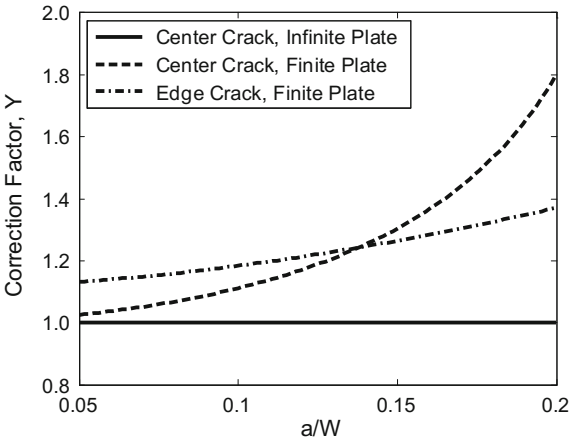


Fig. 6.10 Correction factors for several plate geometries and crack sizes for a plate width of 200 mm

location. A more general expression of the range of stress intensity factor can be written as

$$\Delta K^{'} = Y\Delta K,$$

where Y is the correction factor, given as the ratio of the true stress intensity factor to the value predicted by $\Delta K = \Delta\sigma\sqrt{\pi a}$. The correction factor depends on the geometry of the crack and plate and the loading conditions. Therefore, the apparent stress intensity factor ΔK is different from correct stress intensity factor $\Delta K^{'}$. However, the process of identifying model parameters can compensate for this error by identifying equivalent model parameters that can yield the same remaining useful life.

The capability of compensating error in loading condition by identifying equivalent model parameters is limited to the case when the correction factor remains constant. In reality, however, the correction factor is a function of crack size, and it will evolve as the crack grows, as shown in Fig. 6.10. Therefore, the capability is limited but still it will help alleviating the error.

6.3.1.3 Correlation Between Initial Crack Size and Bias in Data

Bias occurs in measured data due to various reasons, such as calibration error in sensor signals. In structural health monitoring, bias error often occurs when the location of crack is unknown. In actuator/sensor-based health monitoring systems, the actuator sends signals in the form of wave, and the sensor receives reflected signals from a crack. In such a system, the amplitude of signals is often used to determine the size of crack. However, the amplitude of signals quickly diminishes as a function of distance from the actuator to the crack to the sensor. Therefore, the distance has to be considered in calibrating the signal strength with crack size. If the

actual distance is shorter than the used distance, the structural health monitoring system may estimate the crack size larger than the actual one, and vice versa. Another example is the orientation of crack. When the crack direction is perpendicular to the sensor/actuator, the signal from reflected wave is the strongest. The same size crack with different directions may have different signal strengths. Therefore, it is possible to have a bias due to crack orientation.

Bias in measured data can be handled in physics-based approaches but not in data-driven approaches. The mathematical model in data-driven approaches does not have any physical meaning, and thus, no distinction between actual damage and bias. Theoretically, it is possible to include bias in data-driven approaches, but practically it does not mean much because data-driven approaches focus on finding the trend of data. For example, if all measured data are shifted by a constant, data-driven approaches may yield the same prognostics results because the trend of data is identical. The only difference would be the threshold, but the threshold will also be shifted in data-driven approaches because it is also calculated by experiment using the same system.

In physics-based approaches, however, the bias can play an important role. In the case of crack growth, for example, let us assume that there is a positive bias in the measured crack size. The effect of this positive bias ends up overestimating the range of stress intensity factor, $\Delta K = \Delta\sigma\sqrt{\pi a}$. Even if the crack size is overestimated, the crack growth may remain the same. Therefore, from Eq. 6.9 the exponent m may be underestimated. For the same reason, if negative bias is applied, the exponent will be overestimated.

In order to identify the bias systematically, it is necessary to include a deterministic bias to the physical model. In the case when the physical model is given in the rate form, Eq. 6.1, the measured crack size and actual crack size have the following relationship:

$$a_N^{\text{meas}} = a_N + b,$$

where b is the deterministic bias, and a_N is the actual crack size at loading cycle N. When the measured crack size has a bias, the actual crack size has to be used in calculating the range of stress intensity factor; that is, $a_N = a_N^{\text{meas}} - b$ has to be used. In general, the magnitude of bias is unknown, and it has to be included to the unknown parameters. Therefore, the total number of parameters are now four (m, C, a_0, b).

In the case when the physical model is given in the total form, the measured crack size can be expressed by adding a constant bias to crack size in Eq. 6.2 as

$$a_k^{\text{meas}} = a_k + b = \left[N_k C\left(1 - \frac{m}{2}\right)\left(\Delta\sigma\sqrt{\pi}\right)^m + a_0^{1-\frac{m}{2}}\right]^{\frac{2}{2-m}} + b. \tag{6.13}$$

An interesting observation at the parameter estimation stage is that the bias is correlated with the initial crack size in the physical model that has an effect on the degradation behavior. Let the initially detected crack size be a_0^{meas} and let the

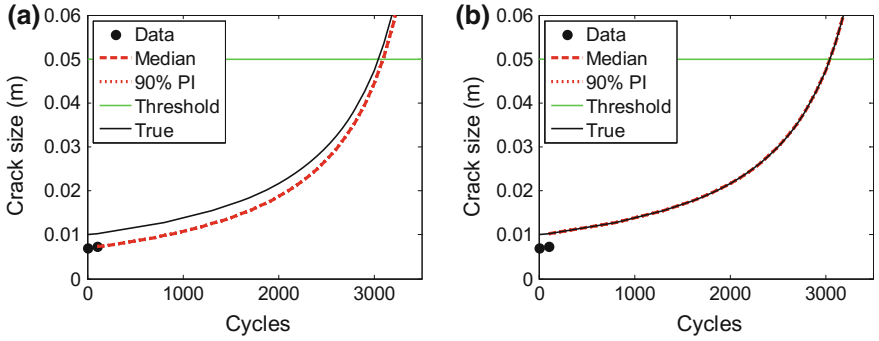


Fig. 6.11 Degradation prediction with biased data. **a** before bias correction, **b** after bias correction (final result)

measurement data have no noise. This measured size is the outcome of the initial damage size and bias:

$$a_0^{\text{meas}} = a_0 + b. \tag{6.14}$$

For a given measured crack size, there is an infinitely possible combinations of initial crack sizes and biases that can yield the same measured crack size. Therefore, there is a linear correlation between them.

When two Paris model parameters are known, true values of initial crack size and bias can be found with two exact measured data. In Fig. 6.11, two measured data are biased by −3 mm; measured crack sizes are consistently less than the true sizes. NLS is employed by setting the initial values of $a_{0,\text{initial}} = 0.02\,\text{m}$ and $b_{\text{initial}} = 0.0\,\text{m}$ with two exact damage data. After optimization, the two parameters are estimated as $\hat{a}_0 = 0.01\text{m}$ and $\hat{b} = -0.003\text{m}$, which are the same as the true values. If the bias is included in predicting the degradation, the result follows the biased data as shown in Fig. 6.11a. Therefore, once the bias is identified, the bias term has to be subtracted in prediction, as shown in Fig. 6.11b, which the prediction curve is offset by $-b = 0.003\,\text{m}$ from the measure data.

As mentioned before, identifying parameters is hindered by noise in data. Also, several sets of correlation, e.g., the case that model parameters, initial crack size, bias, and loading condition are considered as unknown parameters together, make identifying parameters challenging. In the following section, three physics-based algorithms are compared based on identifying model parameters in the Paris model. Further study for identification of correlated parameter under noise and bias will be discussed in Sect. 7.3.

6.3.2 Comparison of NLS, BM, and PF

In the previous section, it is shown that correlations are intrinsic nature in the physics-based prognostics. However, different algorithms show different performances in identifying correlations. In this section, the three physics-based algorithms, NLS, BM, and PF, are compared in terms of their capability in identifying correlations as well as predicting degradation in terms of samples.

6.3.2.1 Prior Information and Sampling Error

An important distinction between regress-based methods (e.g., NLS) and Bayesian-based methods (e.g., BM and PF) is the usage of prior information. Prior information is preexisting knowledge for unknown parameters. This includes expert's opinion, previous experience, or knowledge from similar systems. In the case of initial crack size, for example, the traditional nondestructive inspection technology can provide an approximate range of initial crack size existing in airplane panels. For the case of bias, an approximate range of bias can be estimated by testing measurement equipment at known crack sizes. In the case of Paris model parameters, C and m, laboratory tests for different batches from different manufactures can be used to estimate the approximate range. In fact, the ranges of Paris model parameters are available in several material handbooks.

Utilizing the prior information is a unique characteristics of Bayesian-based methods because there is no similar concept in regress-based methods. In that sense, regress-based methods are similar to the frequentist's method in Chap. 3, where only objective information (i.e., measured data) is used in estimating parameters. On the other hand, Bayesian-based methods use subjective information, which is the prior information. In Bayesian-based methods, this prior information is represented in the form of prior distribution. Table 6.1 shows the prior distributions that are used in this chapter. Without prior distribution, parameter estimation using Bayesian methods becomes identical to maximum likelihood estimation, which becomes the same as regression-based parameter estimation when variability in measured data has a Gaussian distribution.

The performance of Bayesian inference is determined by the uncertainty in prior and the variability in measured data. If uncertainty in the prior is very small, measured data may not contribute in improving the uncertainty in the posterior distribution. In this case, the prior distribution contributes the most to the posterior distribution. If the variability in measured data is very small, there is not much use for the prior. In this case, the posterior distribution is dominated by measured data.

The prior distribution can affect the posterior distribution in both positive and negative ways. If the uncertainty in prior is small and it covers the true value of parameter, the prior can help accelerating the convergence of the posterior distribution. If the uncertainty in prior distribution is too big, it will not help to reduce uncertainty in the posterior distribution. In fact, when the uncertainty in the prior is

infinity, it is called noninformative prior; that is, it is unnecessary to consider the prior distribution in Bayesian update. In such a case, only likelihood functions contribute to the posterior, and that is why the estimation result is identical to the maximum likelihood estimation.

When the prior information is wrong or inconsistent with measured data, it can actually prevent parameters from being accurately identified or slow down the parameter estimation process. In the case of Paris model exponent, m, for example, let us assume that the prior distribution is given as a uniform distribution, $m \sim U(3.0, 3.5)$, but the true value is supposed to be $m_{\text{true}} = 3.8$. In such a case, no matter how many measured data are used, there is no way the posterior distribution to include the true value because the value of prior distribution is zero at the true value. This is an extreme case, but in general a wrong prior can make the posterior distribution converge slowly or converge to a wrong value. More detailed discussion can be found in Sect. 7.2

In order to compare the effect of prior distribution, NLS, BM, and PF are used to predict the degradation behavior of Paris model using data Set 6 with a level of noise, $v = 1$ mm in Fig. 6.2a. It is assumed that 1500 cycle is current, and the future crack growth is predicted using the three different methods in Fig. 6.12. It is shown that the predictions are similar between BM and PF, but NLS clearly shows different prediction with much wider uncertainty, and the median is far away from the true degradation curve. Such an inferior performance of NLS can be explained by its inability to utilize the prior information. In order to explain the effect of prior information clearly, the identified parameters from three methods are shown in Fig. 6.13. In the figure, the black rectangular box represents the bounds of uniform prior distributions given in Table 6.1. Since the prior is given as a uniform distribution, the posterior distribution cannot go beyond this box. Because of this bounding prior distribution, the parameters can be identified within the box for BM and PF, and thus, the prediction results can be accurate with small uncertainty.

The influence of prior distribution, however, is gradually reduced as more data are used. For example, when measured data up to 2500 cycles are used, the prediction results from NLS and BM/PF become similar to each other, which is shown in Fig. 6.14. The difference in the distribution of updated parameters between NLS and BM is significantly reduced as shown in Fig. 6.15a.The difference will further be reduced as more data are used.

On the other hand, when prior information is not available and there is small number of data available, NLS can actually perform better than BM and PF. This is because the limited number of samples in BM and PF cannot cover the wide range of correlation as with NLS result shown in Fig. 6.13a, and there will be a huge sampling error.

An interesting observation from Fig. 6.14c is that the prediction result from PF is worse than the other two. This is due to accumulated sampling error during the updating process; i.e., the particle depletion phenomenon introduced in Chap. 4. As shown in Fig. 6.15b, only a few samples remain after 25 updating process for PF (blue dots). In fact, the prediction results can be accurate even with few different samples when they are laid on the correlation line close to the true values. However,

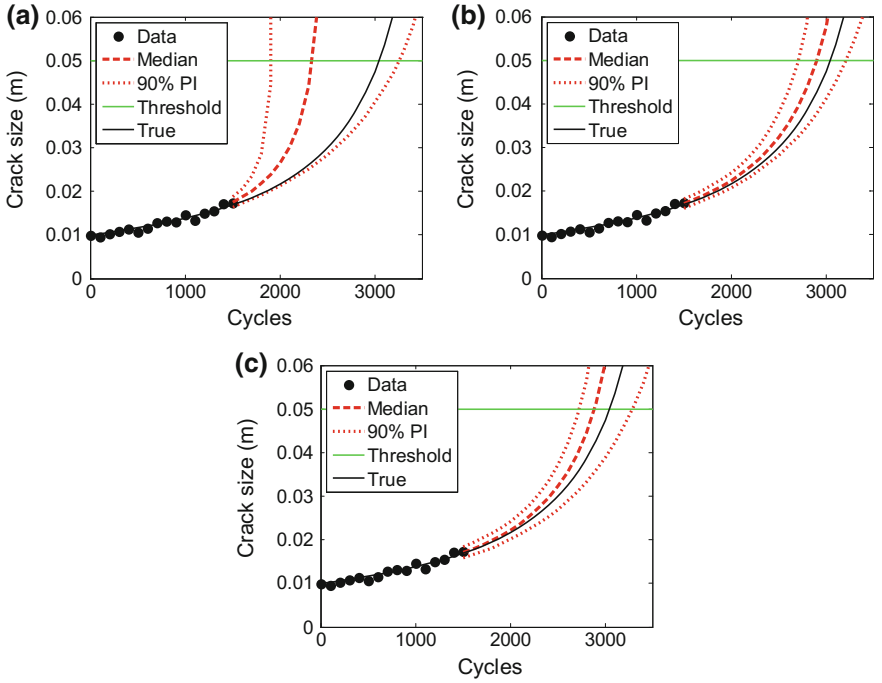


Fig. 6.12 Degradation prediction at 1500 cycles. **a** NLS, **b** BM, **c** PF

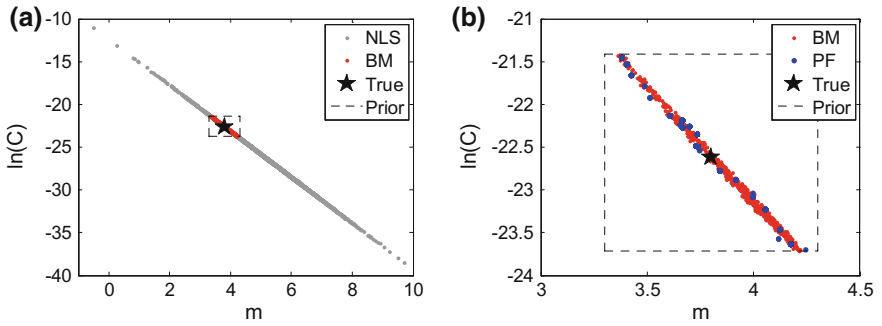


Fig. 6.13 Comparison of the identified parameters from NLS, BM, and PF at 1500 cycles. **a** NLS and BM, **b** BM and PF

the location of few samples cannot be ensured, and the prediction result easily varies whenever PF is performed.

The phenomenon of particle depletion has been the major drawback of PF. A common practice to avoid particle depletion is to add random samples from an arbitrary distribution during the prediction step so that duplicated particles are not generated (Kitagawa 1987, Higuchi 1997, Wang et al. 2009). This method,

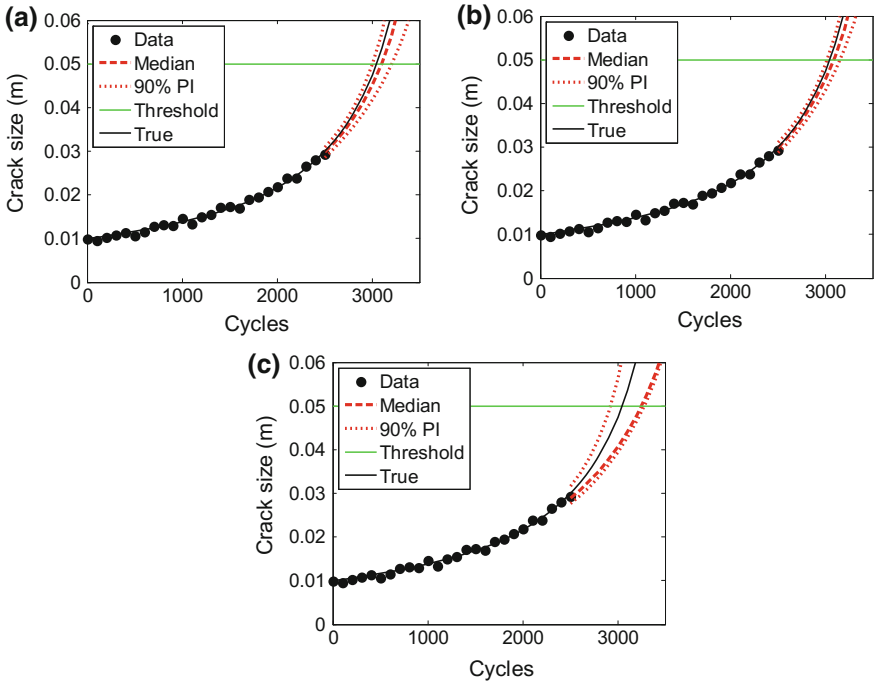


Fig. 6.14 Degradation prediction at 2500 cycles. **a** NLS, **b** BM, **c** PF

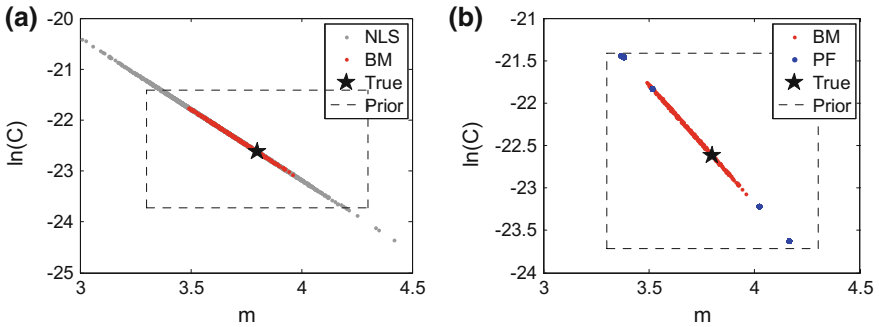


Fig. 6.15 Comparison of the identified parameters from NLS, BM, and PF at 2500 cycles. **a** NLS and BM, **b** BM and PF

however, can change probabilistic characteristics of parameters and can increase the variance of parameters. Gilks and Berzuini (2001) proposed a resample-move algorithm based on the PF and Markov chain Monte Carlo (MCMC) method. Kim and Park (2011) introduced the maximum entropy particle filter and demonstrated its effectiveness by applying it to highly nonlinear dynamical systems.

In BM, even though there is no accumulated sampling error like the particle depletion phenomenon, more sampling error exists compared to NLS due to the random walk process in MCMC sampling. The location of the initial sample, the proposal distribution to draw a new sample and the acceptance ratio to the old sample have an effect on the sampling results; improper settings or inadequate selection may lead to a convergence failure or show a stationary chain where the old sample is selected constantly. Rubin (1998) and An and Choi (2013) used the marginal density function for the proposal distribution in order to reduce these effects. Gelfand and Sahu (1994) presented two distinct adaptive strategies to accelerate the convergence of the MCMC algorithm. More literature can be found in the reference by Andrieu et al. (2003).

6.3.2.2 Uncertainty Representation and Updating Process

An important comparison between different physics-based methods is the way to representing uncertainty in identified parameters. For this aspect, NLS handles uncertainty differently from BM and PF. In BM and PF, the uncertainties in the unknown parameters are considered with a joint posterior distribution, and thus, the correlation between parameters as well as the distribution of each parameter can be identified at once. In NLS, on the other hand, each parameter is deterministically estimated via optimization process, and then, a linear correlation is added to the deterministic results based on the covariance matrix. The deterministic parameters are considered as the mean of distribution. Since samples are generated from Student's t-distribution directly, there is negligible sampling error in the process. Therefore, NLS can be an efficient and effective prediction method when there is a strong linear correlation between unknown parameters. However, correlations may not be linear (see Exercise problem P6.4), and it is not easy to find accurate correlations when there are many unknown parameters in a degradation model.

Also, the posterior distributions for parameters from BM and PF are not limited to standard distribution types, such as normal or exponential distribution. They may not have any particular type of distribution. In fact, the posterior distribution is represented using samples or particles. However, in the case of NLS, the posterior distributions of parameters are assumed, especially, as a normal distribution, and a t-distribution is practically employed with estimated variance. The issue of distribution type, however, is less important than accurately identifying correlation between parameters since prediction accuracy is more sensitive to correlation than uncertainty type. Also, as more data are used, the uncertainty in the parameter is reduced and the error distribution approaches to a normal distribution, which makes the assumption for parameter distribution in NLS is valid.

The difference between BM and PF is usually negligible in terms of prediction results because the two methods have the same theoretical foundation with the same physical model. The key difference between BM and PF is the way of updating distributions. When the degradation model is given as a closed form (i.e., total form) like the Paris model, BM is faster than PF because the posterior distribution is

given as a single expression. However, BM can be impractical when the degradation model is given in the rate form, like Huang's model because of tremendous computational costs caused by integrating damage for every cycle with thousands of samples separately. This also applies to NLS. Therefore, PF that is based on sequential updating process is only a practical method for the rate form of degradation model.

The sequential updating process in PF is straightforward when updating Paris model parameters or bias. However, when the initial damage size is unknown parameter, it is difficult because when a new distribution of initial damage is considered, the samples have to be propagated from the start. Since the degradation rate depends on the initial damage, the initial distribution of initial damage should be properly assumed based on the measurement data at zero cycle.

6.3.2.3 Results Summary

Based on discussions in the previous sections, the following conclusions can be made for physics-based prognostics algorithms. NLS can be a simple and accurate prediction method when the parameters are linearly correlated. Since it assumes that uncertainty in parameters is a Gaussian distribution, this method is good when the distribution of noise in data is Gaussian and nonlinearity in the physical model is relatively small. This is because when input distribution is Gaussian in a linear system, the output distribution is also Gaussian.

BM is an efficient method when the degradation model is given as a total form equation, because the degradation model is used in calculating likelihood at individual measured data. Since this likelihood is calculated at each samples of MCMC simulation multiplied by the number of data, the degradation model has to be computationally inexpensive. If the number of parameters is small, such as two or three, the grid approximation method in Sect. 3.7.3 can be used. However, since a simple Paris model can have five parameters, m, C, a_0, b, σ, almost all practical models use a family of MCMC sampling methods. When the number of parameters are too many, the joint posterior distribution becomes a high dimension, and many MCMC samples are required to have a good convergence.

For practical purpose, the real challenge in BM is to choose appropriate options for MCMC sampling process, such as initial values of unknown parameters and the width of proposal distribution. If the initial values of parameters are far away from the true parameters, MCMC requires many iterations (samples) to converge to the target distribution. Too small width of proposal distribution can yield destabilization in the sampling results by not fully covering the target distribution, while too large width can yield many duplications in sampling result by not accepting new samples. The rule of thumb is to choose the width of proposal distribution as 1 % of the initial values of parameters, and then to add a factor to consider the variability in data as well as the number of data. That is, the width of proposal distribution depends on the expected uncertainty in unknown parameters. Therefore, BM

Table 6.2 Summary of the difference between the three methods

	NLS	BM	PF
Likelihood (distribution of noise)	No	Yes	Yes
Prior information	No	Yes	Yes
Ability for correlated parameter identification	Limited to linear correlation Gaussian distributions	Unlimited	Unlimited
Levels of difficulty for execution	Low (need to start with good initial parameters)	High (difficult to find initial parameters and weights)	Medium/low (need to start with good initial distributions)
Computational costs (rate-form model)	High (number of function calls × number of cycles)	Very high (number of samples × number of cycles)	Low (number of cycles)
Proper usage	Total-form model linear correlation	Total-form model general correlation	Rate-form model general correlation

requires a good level of knowledge on parameters' estimated values as well as the level of noise in measured data.

PF is considered easy to use, as much as NLS, because it does not require to select many options as with BM. PF uses the state transition function as a physical model, which is given in the form of differential equation (i.e., rate form). However, this cannot be a limitation because if the degradation model is given in the total form, it is often possible to convert the total form into the rate form. The only difficulty in PF is particle depletion problem when many measured data are available. Therefore, PF is the most popular method in practical applications. The differences between three methods are listed in Table 6.2.

6.4 Data-Driven Prognostics

In this section, data-driven prognostics algorithms, Gaussian process (GP) and neural network (NN), are compared in terms of accuracy and uncertainty in degradation prediction. In particular, focuses are on the selection of model form, performance under noise, and the number of training data. For GP, a first-order polynomial is used for the global function. This choice is made because the main purpose of GP in prognostics is for extrapolating degradation in future time, where the correlation between data points are quickly diminished as the extrapolation distance increases and the prediction goes back to the global function. Knowing that degradation shows a monotonically increasing or decreasing behavior, it would be

better to choose bases with such a property. In this book, either linear or quadratic polynomial is suggested, which shows a monotonic behavior when input variable is positive. The distributions and prediction intervals of degradation are calculated using 5000 samples from Student's t-distribution.

For NN, pure linear functions are used for both hidden and output layers. This choice is made because the degradation of crack size shows a monotonically increasing behavior. For the network model, only one hidden node is considered with the degradation information (either measured data or predicted value) from previous three time increments that is used as inputs for both GP and NN. In order to screen out outliers, `outliCrite=2` is used, which means that the results beyond the bound of 2σ are eliminated from the final results. The distributions and prediction intervals of degradation are calculated by repeating NN process 30 times.

6.4.1 Comparison Between GP and NN

In order to compare the prediction capability of GP and NN for the simple crack growth, data Set 6 in Fig. 6.2a are used. Since data-driven approaches do not require loading condition, the information of stress range is not used. In this case, it is assumed that there is no training data set, and only prediction data up to 1500 cycles are used. In order to see the effect of a small level of noise, uniformly distributed random noise $U(-0.001, 0.001)$ is added to the exact damage growth data using the following MATLAB commands:

```
v=0.001;
crackSimpleMeas=crackSimple(6,1:16) + v*(2*rand(1,16)-1);
```

Figure 6.16 shows the comparison between GP and NN using measured data up to 1500 cycles under small level of noise. GP shows little more accurate prediction result with much smaller prediction interval than that of NN. As this is an extrapolation without any training data, the prediction result is considered to be good for both methods. This is because three degradation data at previous times are used as input rather than cycles (comparing the difference remains in Exercise problems in P6.19 and P6.20). The reason for the lower performance of NN compared to GP is that NN has a lower ratio of data to parameters. In the case of GP, two global function parameters and one scale parameter. Therefore, these three parameters need to be identified using 16 data, which makes the data to parameters ratio to be more than five. On the other hand, NN has six parameters (three input weights, one hidden weight, one hidden bias and one output bias). In addition, NN uses 70 % of data for the purpose of training, in which only 11 data are used, which makes the data to parameters ratio to be less than two. In such a case, it is highly

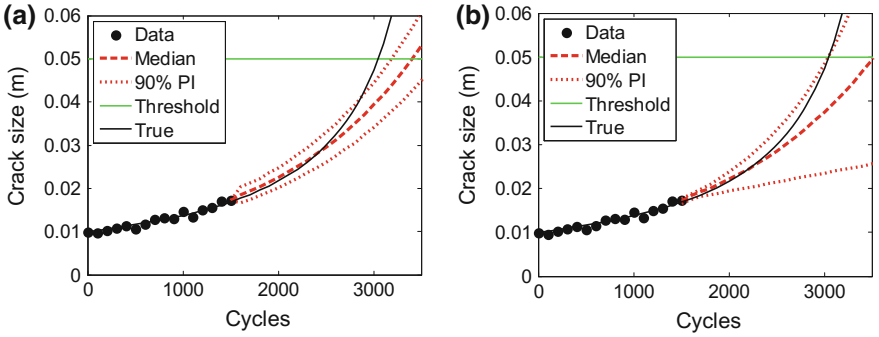


Fig. 6.16 Degradation prediction at 1500 cycles under small noise in data. **a** GP with `h0` = 0.05, **b** NN with `ns` = 26

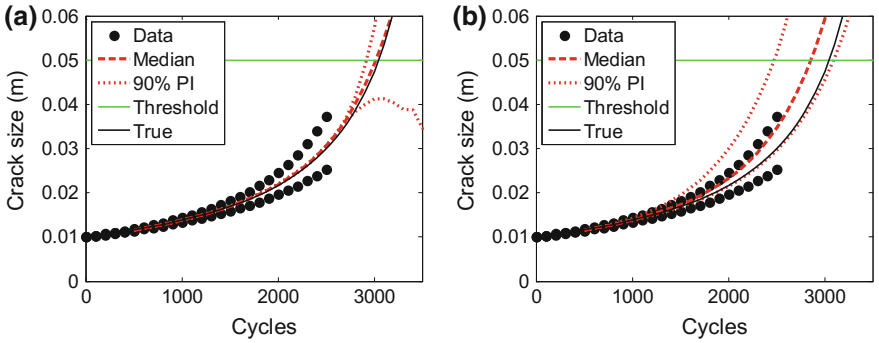


Fig. 6.17 Degradation prediction at 500 cycles of data Set 6 under zero noise with training Sets 5 and 7. **a** GP with `h0` = 0.5, **b** NN with `ns` = 28

likely the training process ends up a local minimum, not the global minimum. Therefore, when the number of data is small and the level of noise is low, GP is favorable than NN.

Figure 6.17 is the prediction results using data up to 500 cycles of data Set 6 under zero noise (rounded to four decimal places) with training Sets 5 and 7. Up to 2500 cycles, this is the case of interpolation, since the prediction set (data Set 6) is in-between training sets (data Sets 5 and 7) and the level of input values of the prediction points is within the training input points. Basically, data-driven approaches can yield good prediction results in the interpolation regions. Especially, the prediction results from GP can be very accurate in the interpolation region with a small level of noise in data since correlation can be well defined, which is proved in Fig. 6.17a. In the figure, the median of GP prediction is very close to the true degradation with a small prediction interval. This is, however, up to around 2500 cycles that is the last cycle of training set. Beyond the cycle, the input

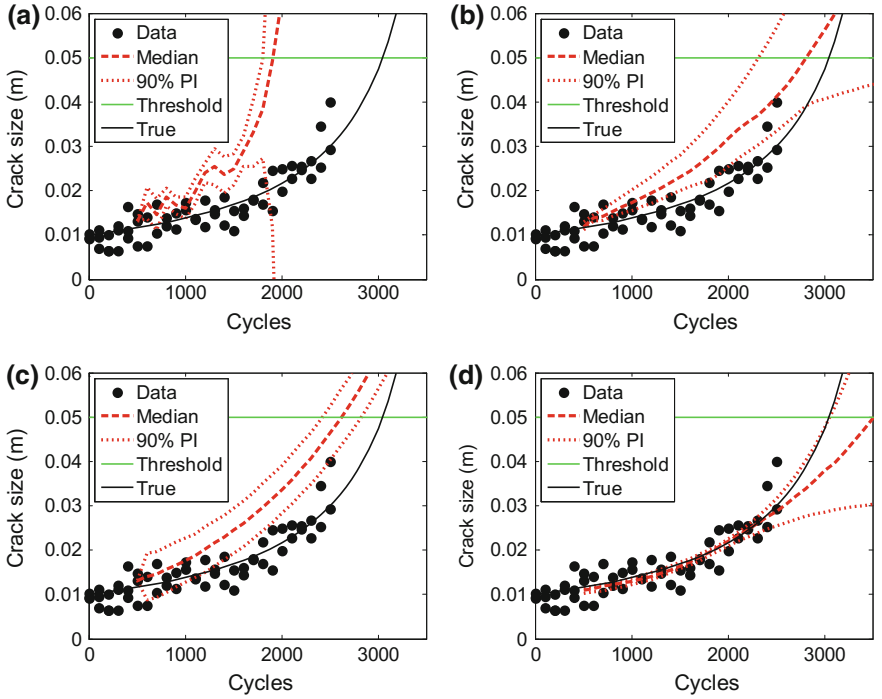


Fig. 6.18 Degradation prediction at 500 cycles under large noise with training Sets 5 and 7. **a** GP with `h0` = 0.5, **b** NN with `ns` = 28, **c** GP with `h0` = 0.001, **d** NN with `ns` = 27, time input and sigmoid function

points for prediction approach to the extrapolation region and correlation is reduced, thereby the quality of the results deteriorated. When the regulation of monotonicity is applied up to 4000 cycles, none of prediction results satisfy the regulation. Therefore, the results beyond 2500 cycles cannot be improved in GP. On the other hand, the results from NN are not deteriorated since they depend on function form, but the accuracy (median) is worse than GP. This result shows that GP is good for the purpose of interpolation in terms of prediction accuracy and prediction intervals, while NN performs better in extrapolation region.

The good results as in Fig. 6.17 cannot be obtained when a large level of noise exists as shown in Fig. 6.18, where uniformly distributed random noise $U = (-0.005, 0.005)$ is applied to all training and prediction data sets. Especially, as shown in Fig. 6.18a, the prediction performance of GP is significantly deteriorated since input variables fail to show a consistent behavior due to the large level of noise; thus, the correlation cannot clearly be identified. In this case, NN outperforms GP, as shown in Fig. 6.18b.

When the initial value of scale parameter changes to a low value like 0.001 in GP, the prediction results can be improved as shown in Fig. 6.18c. However, this result is from zero correlation matrix due to very low scale parameter (the optimum

value of the scale parameter remains the same as the initial value), and thus the prediction results totally depend on the linear global function (compare the results with different global model in Exercise problem P6.21). When there is a large level of noise in data, having training sets may not help much in improving prediction accuracy. However, when the level of noise is large, it may help to use time/cycle as an input in NN instead of using the previous data as input as shown in Fig. 6.18d, whose prediction is much better than the one in Fig. 6.18b. The transfer functions for Fig. 6.18d are the tangent sigmoid and the pure linear function with one hidden node. This result show that when the level of noise is large, NN with time input is favorable than GP.

6.4.1.1 Results Summary

Based on numerical tests in this section, the following conclusions can be made. GP is an efficient and accurate prediction method when a small number of data are given with a small level of noise. This is strong foundation when the prediction point is within the range of training points with a properly defined correlation matrix. It is, however, not an easy task to build a proper correlation matrix since not only a large level of noise, but also a large number of data adversely affect handling the correlation matrix. On the other hand, NN is less sensitive in the level of noise than GP, and a large number of data increase the accuracy in prediction results because the transfer functions in NN fit degradation behavior based on the trend of data rather than fluctuation of each data point. Finally, it is important to consider types of input variables and global/transfer functions in data-driven prognostics. Table 6.3 summarizes the difference between GP and NN for the purpose of predicting degradation.

Table 6.3 Summary of the difference between GP and NN

	GP	NN
Levels of difficulty for execution	Medium (not easy to find proper vales of the scale parameter)	low
Computational costs	Low	Medium (depends on number of repetitions)
Proper usage	Small noise Small number of data Simple degradation behavior	Large noise Large number of data Complex degradation behavior (but not limited to)

6.5 Comparison Between Physics-Based and Data-Driven Prognostics

In this section, the performance and attributes of physics-based and data-driven approaches are compared using the damage growth data Set 8 of Huang's model (complex model) in Fig. 6.2b. As discussed before, since the physical model is given in a rate form, it is expected that NLS and BM are not computationally attractive. Therefore, only PF is considered as a physics-based approach. For a data-driven approach, NN is chosen because GP becomes the same as the global function (a polynomial function) at the extrapolation cycles, while different combinations of transfer functions of NN can predict better than a polynomial function.

The prediction result from PF using data up to 16,000 cycles is presented in Fig. 6.19a. Seven parameters including six model parameters and standard deviation are identified with 17 data. Therefore, the ratio between data and model parameters are about 2.5. A total of 5000 particles are used to represent the correlated joint PDF of seven parameters.

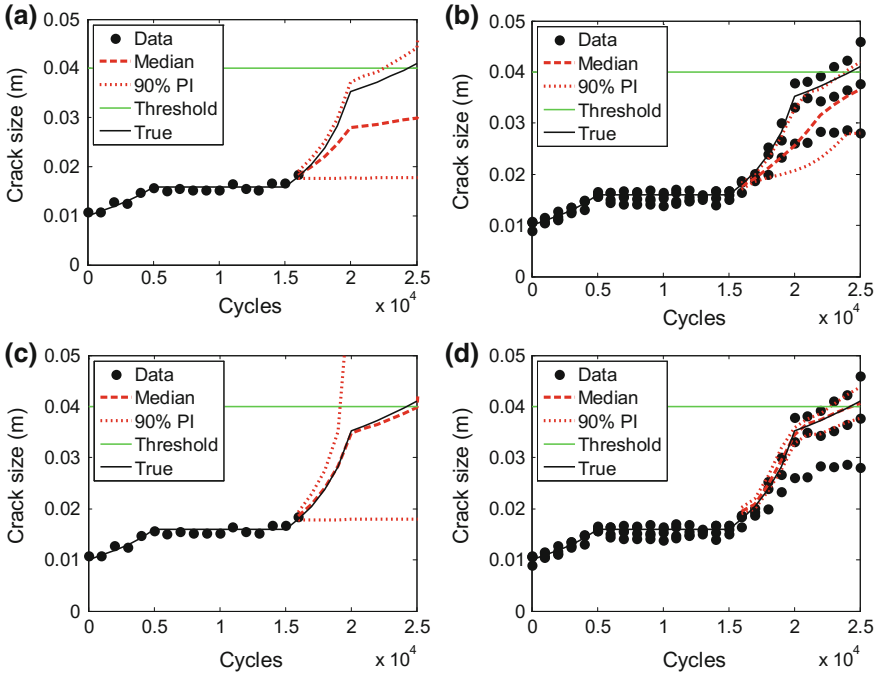


Fig. 6.19 Comparison of PF and NN for a complex degradation growth at 16,000 cycles under small noise. **a** PF with distributed initial damage size, **b** NN with `ns` = 8 training Sets 2, 7 and 9, **c** PF with true value of initial damage size, **d** NN with `ns` = 23 training Sets 2, 7 and 9 and loading conditions

It is obvious that data-driven approaches may not perform as good as physics-based approaches because the latter can take advantage of important physical model information. Therefore, it would be natural to provide more measured data to the data-driven approach. The question is how many additional data are required so that the prediction accuracy of data-driven approach can be comparable with that of physics-based approach. Based on trial-and-error approach, it is found that the three training data set can provide comparable prediction accuracy with that of physics-based approach. Therefore, in this section, NN is used with three training data sets.

Next question is the proximity of training data with the prediction data, which plays a critically important role for the prediction accuracy. For example, when data Set 8 is prediction set, if data Sets 7 and 9 are provided as training sets, then it is expected that the prediction accuracy will be good because the behavior of data Set 8 is close interpolation between data Sets 7 and 9. On the other hand, if data Sets 1 and 2 are provided, the prediction accuracy may not be as good as the case with data Sets 7 and 9. Therefore, three training sets are randomly selected from available ten data sets. Based on this process, Fig. 6.19b shows the prediction of degradation of Huang's model using NN. For this prediction, tangent sigmoid and pure linear functions are employed with five hidden nodes. The result from NN can even be better, but the result is obtained based on eight samples out of 30 repetitions because remaining 22 cases did not satisfy regulations. This means that prediction of complex behavior with only damage size information is not straightforward. In this case, the regulation of monotonicity in degradation should be used to select valid predictions.

The results in Fig. 6.19a, b which can further be improved. If the distribution of initial damage size in PF is updated to be close to the true initial damage, the median will be close to the true curve because the physical model largely depends on the initial damage size. It is, however, a time-consuming process to update the initial damage in PF. Therefore, Fig. 6.19c shows the results using the true value of initial damage instead. For NN, if the loading conditions are added to training in NN, the prediction behaves can be improved, as shown in Fig. 6.19d. Even though physics-based approaches undoubtedly outperform data-driven ones if both a physical model and loading conditions are available, data-driven approaches with more training data (three training sets in this case) and additional information (loading condition in this case) can outperform physics-based ones, as shown in Fig. 6.19d.

6.6 Results Summary

In terms of algorithms, results from case studies can be summarized as follows. Physics-based methods are less affected by the level of noise and model complexity, but they can be employed only when a physical model and loading conditions are available. NLS is different from BM and PF in the process of identifying correlation

between parameters, which works favorable for a linear correlation case. In the comparison between BM and PF, the results from the two methods are not much different, but PF has a wide range of applications and BM is fast when the expression of posterior distribution is available explicitly.

The GP works well when the covariance function is well defined, such as the case of low noise in data and simple behavior of the damage growth. However, all other algorithms work as well for such a benign case. The GP can outperform other algorithms if interpolation conditions can be considered with the given information. The GP is easy to implement and quick to calculate, which gives prediction results with interval within around 2 σ. This is, however, limited to the case that the process to find an optimum scale parameter works well. On the other hand, NN does not require choosing various tuning parameters, even though the computational cost depends on each training case (a few seconds to minutes for one training) and the number of repetitions to account for prediction uncertainty. The NN is advantageous for cases with high levels of noise and complex models with many training data sets.

Finally, here is a note about prognostics methods. When loading conditions and physical models are not available, prediction can be done by using data-driven approaches with at least three data sets, and loading conditions improve the prediction results. However, it would be challenging to obtain many sets of training data, and also physical degradation models are rare in practice. Hybrid approaches or fusion methods that combine the two approaches or prediction results can be considered to improve the prediction performance, detailed explanation can be found a review material by Liao and Köttig (2014). But still there is no effective method to handle the case of limited number of data without physical model.

6.7 Exercise

P6.1 Using the true model parameters given in Sect. 6.2.3, generate 10 sets of crack growth data as shown in Fig. 6.2a for the Paris model.

P6.2 Using the true model parameters given in Sect. 6.2.3, generate 10 sets of crack growth data as shown in Fig. 6.2b for the Huang's model.

P6.3 A physical model is given as $z = \theta_1 + \theta_2$. The measurement data is given as $y = 5$. When $\theta_1 \sim N(0, 2^2)$, Find the correlation between θ_1 and θ_2 (a) by generating 20 samples of θ_1 and θ_2 and plotting them in 2D graph, and (b) by calculating correlation coefficients.

P6.4 When the degradation model of battery is defined as $z(t) = a \exp(-bt)$ with a and b being two model parameters, identify correlation between the two model parameters at a given cycle/time t_k. Using the data given in Table 4.1, identify two model parameters. Discuss why in this case, the accurate model parameters can be identified.

P6.5 Sinclair and Pierie (1990)[2] showed that the two Paris model parameters are related by $\ln(C) = -3.2m - 12.47$ when stress and crack size are in the unit of MPa and meters. Therefore, it is possible to reduce the number of Paris model parameters from two to one. Using NLS and data Set 6 in Fig. 6.2, compare the estimated parameters and the remaining useful life for one and two parameter Paris model.

P6.6 Fig. 6.7 shows the difficulty of identifying model parameters and remaining useful life when three measured data a=`{0.0100, 0.0103, 0.0106}` at N=`{0, 100, 200}` are available. Repeat the same process with three data a=`{0.0100, 0.0116, 0.0138}` at N=`{0, 500, 1000}` and explain why the estimated parameters and remaining useful life are significantly improved.

P6.7 Using NLS and data Set 6 in Fig. 6.2a up to 1500 cycles, identify the two Paris model parameters and covariance matrix. (a) Assuming that the two parameters are independent, calculate the median and 90 % confidence interval of RUL. (b) Using the covariance matrix, calculate the median and 90 % confidence interval of RUL and compare them with the results from (a).

P6.8 Using NLS and data Set 10 in Fig. 6.2a, identify the equivalent model parameters $C^{'}$ when an incorrect loading condition of $\Delta\sigma = 65\,\text{MPa}$ is used, and compare the identified parameter value with that in Eq. 6.12.

P6.9 The correction factor for a center crack of a plate with finite width can be given as

$$Y = \sqrt{\sec\left(\frac{\pi\lambda}{2}\right)\left(1 - \frac{\lambda^2}{40} + \frac{3\lambda^4}{50}\right)},$$

where $\lambda = a/W$ is the ratio between crack size and plate width. When plate width is 0.1 m, initial crack size is 0.01 m, the exact model parameters are $m_{\text{true}} = 3.8$ and $C_{\text{true}} = 1.5 \times 10^{-10}$, (a) generate true crack growth data at N = 0:100:2500 and identify equivalent model parameters using the Paris model, and (b) compare the crack growth curve with estimated parameter with the exact one.

P6.10 In data Set 6 in Fig. 6.2a, add normally distributed random noise $N(0, 0.005^2)$ and a positive deterministic bias of $b = 0.01\text{m}$. Using Bayesian method, identify exponent m of Paris model. Use the prior as a uniform distribution $U(3.3, 4.3)$, $C = C_{\text{true}} = 1.5 \times 10^{-10}$ and $a_0 = 0.01\text{m}$. Compare the mode of identified parameter m with its true value $m_{\text{true}} = 3.8$.

[2]G.B. Sinclair and R.V. Pierie (1990) On obtaining fatigue crack growth parameters from the literature, International Journal of Fatigue, Vol. 12, No. 1, pp. 57–42.

P6.11 Repeat P6.10 with a negative deterministic bias of $b = 0.01$m.

P6.12 In data Set 6 in Fig. 6.2a, add normally distribute random noise $N(0, 0.005^2)$ and a positive deterministic bias of $b = 0.01$m. Using Bayesian method, identify the initial crack size and bias with their correlation. Assume that the true model parameters are given; that is, $m = m_{\text{true}} = 3.8$ and $C = C_{\text{true}} = 1.5 \times 10^{-10}$. Compare the identified results with their true values.

P6.13 For data Set 6 in Fig. 6.2a, add normally distributed random noise $N(0, 0.005^2)$ and estimate the posterior distribution of Paris model exponent m using Bayesian method when the prior distribution is given as $U(3.0, 3.5)$. Assume the true values of other parameters.

P6.14 Repeat P6.13 with the prior distribution of $U(3.7, 4.2)$. Compare the two results in terms of mean, median, and mode of estimated parameter distribution.

P6.15 Use BM and PF without prior to predict the degradation of a simple crack growth with three degradation data at previous times as inputs. Use data Set 6 in Fig. 6.2a up to 1500 cycles with uniformly distributed random noise $U(-0.001, 0.001)$. Compare the prediction results with that from NLS in Fig. 6.12a.

P6.16 Use NLS to identify four unknown parameters, i.e., the initial crack size, bias, and two model parameters in the Paris model. Use data Set 6 in Fig. 6.2a up to 1500 cycles with uniformly distributed random noise $U(-0.001, 0.001)$, and training data sets if they are needed.

P6.17 Repeat P6.16 using BM.

P6.18 Repeat P6.16 using PF without updating the initial crack size. Compare the result with those from NLS and BM in P6.16 and P6.17.

P6.19 Use GP to predict the degradation of a simple crack growth when input is the number of cycles. Use data Set 6 in Fig. 6.2a up to 1500 cycles with uniformly distributed random noise $U(-0.001, 0.001)$. Compare the prediction results with that in Fig. 6.16a when three previous degradations are used as inputs.

P6.20 Repeat P6.19 using NN.

P6.21 Consider the same problem in Fig. 6.18c. Predict the degradation using GP with a quadratic global function.

P6.22 Compare the performance of five algorithms, i.e., NLS, BM, PF, GP, and NN based on the prognostics metric introduced in Chap. 2. Use data Set 6 in Fig. 6.2a up to 1500 cycles with uniformly distributed random noise $U(-0.001, 0.001)$.

P6.23 Based on the differences of each algorithm listed in Tables 6.2 and 6.3, make a selection tree for proper use of different algorithm in terms of given information, such as physical model, loading conditions, noise level, etc.

References

An D, Choi JH (2013) Improved MCMC method for parameter estimation based on marginal probability density function. J Mech Sci Technol 27(6):1771–1779

Andrieu C, Freitas DN, Doucet A et al (2003) An introduction to MCMC for machine learning. Mach Learn 50(1):5–43

Gelfand AE, Sahu SK (1994) On Markov chain Monte Carlo acceleration. J Comput Graph Stat 3:261–276

Gilks WR, Berzuini C (2001) Following a moving target-Monte Carlo inference for dynamic Bayesian models. Roy Stat Soc B 63(Part 1):127–146

Higuchi T (1997) Monte Carlo filter using the genetic algorithm operators. J Stat Comput Simul 59 (1):1–23

Huang X, Torgeir M, Cui W (2008) An Engineering model of fatigue crack growth under variable amplitude loading. Int J Fatigue 30(1):2–10

Kim S, Park JS (2011) Sequential Monte Carlo filters for abruptly changing state estimation. Probab Eng Mech 26:194–201

Kitagawa G (1987) Non-Gaussian state space modeling of nonstationary time series (with discussion). J Am Stat Assoc 82(400):1032–1063

Liao L, Köttig F (2014) Review of hybrid prognostics approaches for remaining useful life prediction of engineered systems, and an application to battery life prediction. IEEE Trans Reliab 63(1):191–207

Paris PC, Erdogan F (1963) A critical analysis of crack propagation laws. ASME J Basic Eng 85:528–534

Rubin DB (1988) Using the SIR algorithm to simulate posterior distributions. Bayesian Stat 3 (1):395–402

Wang WP, Liao S, Xing TW (2009) Particle filter for state and parameter estimation in passive ranging. In: Paper presented at the IEEE international conference on intelligent computing and intelligent systems, Shanghai, China, 20–22 November 2009

Chapter 7
Applications of Prognostics

7.1 Introduction

This chapter introduces several prognostics applications based on the real test data or data that are generated by simulating practical cases. Three major challenges are addressed for prognostics methods to be practical throughout this chapter.

(1) In physics-based approaches, the degradation behavior depends on model parameters. It is, however, difficult to accurately identify these parameters due to the correlation between them and a large level of noise and bias in data. An interesting observation is that even if accurate values of individual parameters may not be found, the prediction of remaining useful life (RUL) can still be accurate. Also, when the physical model has a model form error, prognostics algorithms can identify equivalent values of parameters that can compensate for the model form error. Most physics-based algorithms use Bayesian method in estimating model parameters. One of important advantages as well as disadvantages of Bayesian method is its capability to utilize the prior information. A good prior information can help to reduce uncertainty in the estimated parameters, but inaccurate prior can prevent from converging accurate values.
(2) Data-driven approaches are also affected by the level of noise, but obtaining a large number of training data is a challenge because it is expensive to obtain data due to time and cost. Instead of obtaining data from the same or similar systems, it is possible to utilize the data obtained from accelerated life tests during the development of the current system design. Since accelerated life tests are performed much severe operating conditions, it is necessary to convert them corresponding to the nominal operating conditions in order to be used as training data.
(3) The third challenge is prognostics using data indirectly related to damage. In many cases, the damage data cannot be measured directly, but system responses that are affected by damage are measured instead. For example, vibration data are often measured to monitor any cracks on bearings and gears,

N.-H. Kim et al., *Prognostics and Health Management of Engineering Systems*,
DOI 10.1007/978-3-319-44742-1_7

from which the level of damage in bearings is to be predicted. It is difficult to extract degradation features from indirectly measured data. In such a case, it is important to separate signals from noise. Therefore, signal-to-noise ratio becomes an important issue.

This chapter is organized as follows: in Sect. 7.2, wear volume in a revolute joint is predicted based on the Bayesian method with Archard's wear model; in Sect. 7.3, crack growth parameters are identified under various levels of noise and bias; in Sect. 7.4, practical usage of accelerated test data is presented for the purpose of prognostics; in Sect. 7.5, bearing prognostics is performed based on featured data extracted from vibration signals; and other applications are introduced in Sect. 7.6.

7.2 In Situ Monitoring and Prediction of Joint Wear[1]

In this section, the prognostics of wear volume in a revolute joint is presented using a physics-based approach; i.e., the Bayesian method. Special in situ measurement devices are designed to measure the progress of wear as well as loading conditions. The Bayesian inference technique is used to update the distribution of wear coefficients, which incorporates in situ measurement data to obtain the posterior distribution. The Markov chain Monte Carlo (MCMC) technique is employed to generate samples from the given distribution. The results show that it is possible to narrow the distribution of wear coefficients and to predict the future wear volume with reasonable confidence. The effect of the prior distribution on the wear coefficient is discussed by comparing with the noninformative case.

7.2.1 Motivation and Background

Most mechanical systems are characterized by motion. In order to fulfill their design function, the individual components of a system must move relative to one another, which inevitably produces sliding along the mating surfaces and causes wear. Wear is the gradual removal of material from contacting surfaces in relative motion, which eventually causes failure of the system. Since mechanical wear occurs for most systems in motion, it is important to predict its effects and estimate the service life of the system before failure.

The common approaches for predicting wear volume are (a) estimating wear coefficient from tribometer tests, (b) calculating contact pressure and sliding distance, and (c) using a wear model to calculate wear depth as well as wear volume. Although this process is commonly applied, the fundamental limitation of wear

[1]An D, Choi JH, Schmitz TL, Kim NH (2011) In situ monitoring and prediction of progressive joint wear using Bayesian statistics. Wear 270(11–12):828–838.

prediction is that it applies only when the actual contact pressure condition matches the tribometer test (constant) pressure condition. In practice, however, the contact pressure usually varies as a function of time. Furthermore, it often varies over the contact surface. Since the wear coefficient is not an intrinsic material property, it depends on the operating conditions. Calculating wear coefficients at all possible operating conditions requires numerous wear tests and is extremely time-consuming. In addition, the variability of wear coefficients is significant even if different parts are made of the same material (Schmitz et al. 2004). Thus, it is a preferred approach to measure the wear coefficient directly from the mechanical component in question. Since calculating the wear coefficient requires kinematic information (wear volume and sliding distance) as well as kinetic information (contact force/pressure), it is important to design an in situ measurement apparatus to measure both factors. In this study, an instrumented slider-crank mechanism (Mauntler et al. 2007) is used to measure these factors. Since in situ measurements inherently include uncertainty, the statistical distribution of wear coefficient is updated based on the Bayesian method, and future wear is predicted based on the updated wear coefficient.

This section is organized as follows. In Sect. 7.2.2, the simple wear model used in this section is summarized. Although a linear model is used, the main concept of this study can be extended to more complex wear models. In Sect. 7.2.3, in situ measurements of joint force and wear volume are presented. Section 7.2.4 describes the Bayesian inference technique along with the MCMC sampling method. In Sect. 7.2.5, the wear coefficient is statistically identified and validated. Section 7.2.6 includes a discussion of the results and conclusions.

7.2.2 *Wear Model and Wear Coefficient*

Wear is a common physical phenomenon when two or more mechanical parts are under relative motion (sliding) on the interface with contact pressure. When plastic deformation is a dominant mechanism of material removal, Archard's wear model (Archard 1953) can be applicable as discussed by Lim and Ashby (1990) and Cantizano et al. (2002). Archard's wear model assumes that the volume of material removed is linearly proportional to the product of the sliding distance and the normal load. The traditional method for calculating the wear coefficient is based on plotting wear volume versus sliding distance as shown in Fig. 7.1. In this model, the worn volume is considered to be proportional to the normal load. The model is expressed mathematically as

$$\frac{V}{l} = K\frac{F_n}{H}, \tag{7.1}$$

where V is the worn volume, l the slip distance, K the dimensionless wear coefficient, H the Brinell hardness of the softer material, and F_n the applied contact force.

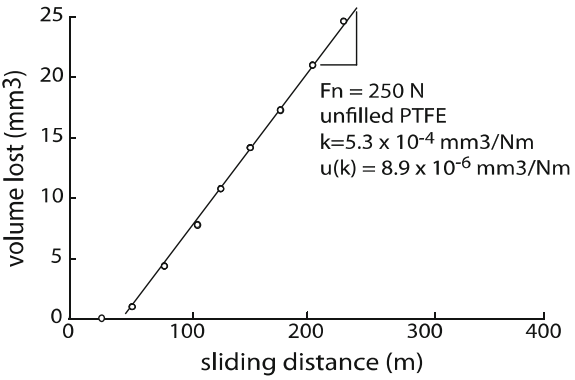


Fig. 7.1 Example of wear coefficient data from an unfilled PTFE polymer system (Schmitz et al. 2004)

Since the wear coefficient is the quantity of interest, Eq. 7.1 is often written in the following form

$$\kappa = \frac{V}{F_n \cdot l}. \tag{7.2}$$

The nondimensioned wear coefficient K and the hardness are grouped into the dimensioned wear coefficient κ. Therefore, the main objective of wear analysis is to identify the wear coefficient for given normal load and slip distance. Since the worn volume is relatively small, it is often measured in the unit of mm^3. Thus, the units of κ become mm^3/N m.

In calculating wear coefficient, the applied normal force and contact area remain constant through the entire process. If the normal force varies within the slip distance, the definition of wear coefficient in Eq. 7.2 must be modified as

$$\kappa = \frac{V}{\int_0^l F_n(l)\mathrm{d}l}. \tag{7.3}$$

In this definition, it is assumed that the wear coefficient is independent of normal force, which is not true in general. However, the wear coefficient κ in Eq. 7.3 can be interpreted as an average wear coefficient for a given load profile.

The wear coefficient is not an intrinsic material property, but instead depends on operating conditions, such as the normal force and slip speed. The value of κ for a specific operating condition and given pair of materials may be obtained through experiments (Kim et al. 2005). However, experimentation does not often represent the real conditions of a machine, especially when loading conditions vary due to the progress of wear. Thus, a difference may exist between the wear coefficient measured in the tribometer test and that observed from the real machine.

The wear coefficient from the tribometer test is reliable as it is performed under a well-controlled environment, but it may not reflect the real operating conditions. On the other hand, the wear coefficient observed from the real machine reflects real

operating conditions, but uncertainty in the field measurements is relatively large as it is not performed under laboratory conditions. The main objective of this study is to reduce the effect of uncertainty in the field measurements by using a statistical tool to identify the wear coefficient more accurately. The identified wear coefficient can be used to predict the wear volume in the future and, thus, to schedule maintenance intervals.

7.2.3 *In Situ Measurement of Joint Wear for a Slider-Crank Mechanism*

The slider-crank test apparatus used in this study is shown in Fig. 7.2. In order to minimize dynamic contributions from the other components in the mechanism, porous carbon air bearings are used for the revolute joint between the follower link and the slide stage, as well as the prismatic joint for the linear slide.

The revolute joint under study consists of a 19.0 mm diameter instrumented steel pin and a polymer bushing. The pin is clamped in the crank link at one end and rotates, subject to sliding friction, in the bushing clamped in the follower link. The pin is made of hardened steel and is assumed to be hard enough so that no appreciable wear occurs on its surface. The bushing, on the other hand, is made of polytetrafluoroethylene (PTFE) which is soft and is subject to considerable wear. To enable the wear debris to escape the contact area and prevent it from affecting the wear progression, grooves are machined into the bushing.

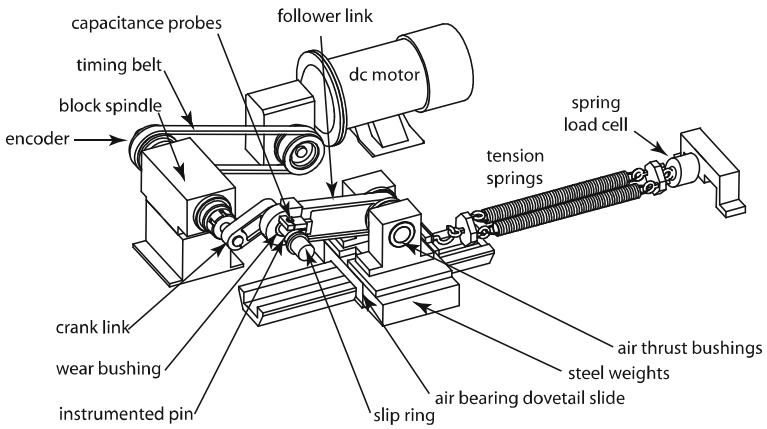


Fig. 7.2 The layout of slider-crank mechanism used in the experiments. Reprinted from *Wear*, 270/11–12, Dawn An, Joo-Ho Choi, Tony L. Schmitz and Nam H. Kim, "In situ monitoring and prediction of progressive joint wear using Bayesian statistics," 828–838, © (2011), with permission from Elsevier

The added mass and springs affect the joint force, which can accelerate wear. In practice, mechanisms are usually operated under added masses and additional constraint forces. The wear pattern or process depends on three factors: (1) wear coefficient, (2) joint force, and (3) relative motion in the interface. Because different joints may have different joint forces and relative motions, they need to be measured.

Forces transmitted through the joint of interest are measured via a load cell built into a steel pin. Two full-bridge arrays of strain gages mounted to a necked down portion of the pin monitor transverse loads while canceling out bending stresses. The necked portion of the pin, along with a hollow cross section, also serves to localize the strain to the region where the gages are attached. A slip ring mounted to the free end of the pin allows power and signals to be transmitted to and from the strain gages. The load cell is deadweight calibrated and has a full scale capacity of 400 N with a resolution of 2 N.

Simultaneously, two orthogonally mounted capacitance probes monitor the position of the pin relative to bushing. These probes are clamped to the follower arm and are electrically insulated by polymer bushings. These probes have a range of 1,250 μm and a resolution of 40 nm. Additionally, the pin and target are electrically grounded. The angular position of the crank is measured by a hollow shaft incremental encoder attached to the spindle shaft.

Figure 7.3 shows the joint force as a function of crank angle at Cycle 1 and Cycle 20,500. The measured joint force agrees well with the multibody dynamic simulation with the coupled evolution wear model (CEWM) (Mukras et al. 2010). High-frequency oscillation is observed when the slider changes its velocity direction. However, there is no significant variation of joint forces between different cycles. Thus, the profile of joint forces is fixed throughout all cycles.

It is well known that uncertainty in applied loading is the most significant factor in prognosis. Without knowing future loadings, the uncertainty in prediction can be so wide that the prediction may not have significant meaning. This issue can be

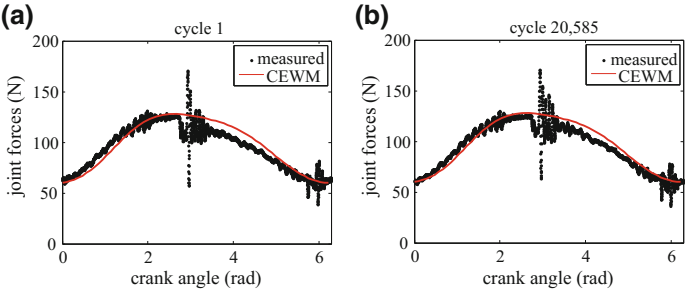


Fig. 7.3 Joint force predictions and measured data. **a** Cycle 1, **b** cycle 20,500

addressed in two ways. First, although the loading condition is variable, if there is enough proof that the future loadings will be similar to the past loadings, then the collected data for past joint forces can be used to predict the future loadings. In this case, the future loadings can be represented using statistical methods. Of course, the uncertainty in predicted life will be increased due to the added uncertainty. Second, wear parameters can be characterized using only the past loading history. This approach is applied here. Note that this method will also work for a variable load history. In this case, the computation will be more expensive than the current example.

Figure 7.4 shows measured displacements of the pin center using the capacitance probe. The wear volume is computed based on the values of δ_x and δ_y. Due to the pretensioned springs, the contact points are located at only one side of the bushing. However, the location of pin center varies according to crank angle. This can be explained by the rounded pin surface and different amounts of elastic deformation due to variation in spring forces with crank angle. The measured forces, computed wear volume from the measured displacement, sliding distance, and wear coefficient are shown at six cycles for crank angles 0 and π radians in Tables 7.1 and 7.2, respectively. Although the wear coefficient converges for both cases, the converged results do not match. This is because the joint force is not constant with angle and the measured wear volume includes uncertainty. In the following section, a statistical approach is introduced to improve the wear coefficient estimate. The idea is to estimate a wear coefficient incorporating uncertainty based on the first five sets of measured data and then predict the wear volume in the sixth cycle using the information. Since the actual data at this cycle are also measured, we can evaluate the accuracy of the method by comparing the measured and predicted results.

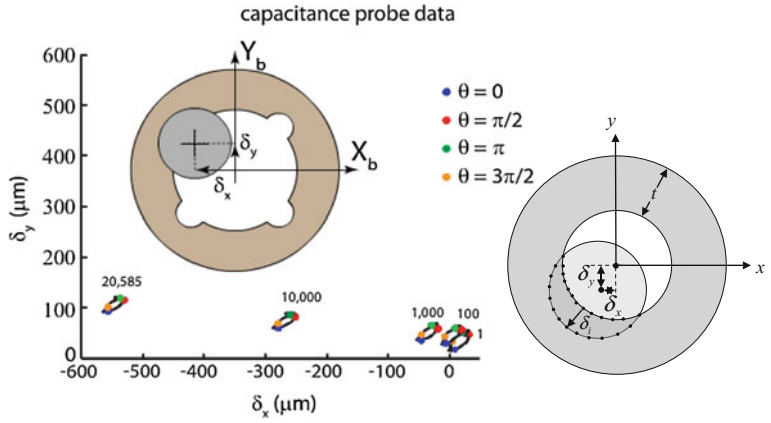


Fig. 7.4 In situ measurements of pin displacement. **a** Profile of pin center locations, **b** wear volume from the overlapped area

Table 7.1 Wear coefficient calculation using pin locations at 0 rad

Cycles	Force (N)	Volume (mm^3)	Slip distance (m)	$\kappa \times 10^4$ (mm^3/N m)
1	64.41	1.59	0.06	4134.80
100	62.80	2.57	5.99	68.25
1000	63.17	8.10	59.85	21.44
5000	64.77	24.48	299.24	12.63
10,000	62.65	46.21	598.47	12.32
20,585	59.96	93.90	1232.00	12.71

Table 7.2 Wear coefficient calculation using pin locations at π radian

Cycles	Force (N)	Volume (mm^3)	Slip distance (m)	$\kappa \times 10^4$ (mm^3/N m)
1	103.87	7.29	0.06	11731.00
100	106.90	7.64	5.99	119.43
1000	114.04	9.55	59.85	13.99
5000	138.77	23.87	299.24	5.75
10,000	143.50	44.41	598.47	5.17
20,585	147.85	91.56	1232.00	5.03

7.2.4 Bayesian Inference for Predicting Progressive Joint Wear

7.2.4.1 Likelihood and Prior

The Bayesian technique introduced in Chap. 3 is employed to identify the wear coefficient κ using wear volume measurements. The wear volume corresponds to the observed data, and the wear coefficient corresponds to the model parameters. The procedure to obtain the posterior distribution consists of proper definition of the probability distribution for the likelihood and prior. The choice of likelihood can affect the analysis results. In this context, Martín and Pérez (2009) studied about a generalized lognormal distribution to provide flexible fits to many types of experimental or observational data. Moreover, there are many approaches to select the appropriate distribution type for likelihood. For example, Walker and Gutiérrez-Peña (1999) suggested a simple one to select a model when no information on variability of experimental data is available.

In this study, two types of likelihood are assumed for the simplicity: normal and lognormal distributions. In addition to the wear coefficient, the standard deviation of the measured wear volume is also considered as an unknown model parameter. In likelihood calculation, the actual wear volume at a given cycle is computed by averaging the values at 0 and π radians (see Tables 7.1 and 7.2). Denoting the wear

volume measured at the specific cycle as V, the likelihoods of the data for a given wear coefficient and standard deviation can be defined as

$$f(V|\kappa,\sigma) \sim N\left(\mu,\sigma^2\right) \tag{7.4}$$

and

$$f(V|\kappa,\sigma) \sim LN\left(\eta,\zeta^2\right), \tag{7.5}$$

where μ and σ are the mean and standard deviation of wear volume, and η and ζ are two parameters of the lognormal distribution. Note that $N(\mu,\sigma^2)$ represents a normal distribution, while $LN(\eta,\zeta^2)$ indicates a lognormal distribution. In Eq. 7.4, the likelihood is defined with mean and standard deviation. In reality, a single wear volume is measured using in situ capacitance probes, but this measured data has errors. In the likelihood definition, the measured volume is used as a mean value, but the error is unknown. Therefore, the standard deviation of the likelihood is considered unknown and needs to be updated through the Bayesian inference. Therefore, the distribution of standard deviation represents the error in data.

According to Eq. 7.3, the mean wear volume is expressed as the integral of contact force and slip, multiplied by κ. In practice, this integral is computed discretely by dividing the cycle into q equal intervals:

$$\mu = \kappa N\left(\sum_{k=1}^{q} F_{n,k}\Delta l_k\right), \tag{7.6}$$

where $F_{n,k}$ and Δl_k are the contact force and incremental slip at kth segment, respectively, and N is the number of cycles. In order to simplify the above expression, it is assumed that the sum of contact force and sliding distance can be expressed by an average normal contact force times the accumulated sliding distance. That is,

$$\mu = \kappa F_n l, \tag{7.7}$$

where F_n is the average contact force and $l = \sum \Delta l_k$ is the accumulated sliding distance. Because the force profile is consistent from one cycle to the next, the sum in Eq. 7.6 is obtained as a constant with the value 5.966 (N m) based on experimental data. In Eq. 7.5, η and ζ are given as

$$\eta = \ln\mu - \frac{1}{2}\zeta^2 \quad \text{and} \quad \zeta = \sqrt{\ln\left(1+\frac{\sigma^2}{\mu^2}\right)}. \tag{7.8}$$

Note that the mean is only a function of κ since all other terms are given or fixed in Eq. 7.6. This makes sense given that κ and σ are unknown parameters to be estimated conditional on the observed data V.

For the prior distribution of κ, the wear coefficient from the literature (Schmitz et al. 2004) is employed:

$$f(\kappa) \sim N(5.05, 0.74) \times 10^{-4}. \tag{7.9}$$

This prior distribution was obtained from tribometer tests under a constant contact pressure to determine κ for the bushing with the same material as the current study. In the Bayesian technique, the posterior distribution of κ is obtained by multiplying equation 7.4 or 7.5 with the prior distribution $f(\kappa)$ provided in Eq. 7.9. Since the Bayesian technique can be sensitive to the prior, the case with no prior knowledge (a noninformative prior) is also considered to study the effect of prior information. Additionally, the noninformative prior is considered for the standard deviation σ of likelihood because no knowledge is available. The noninformative prior is equivalent to the uniformly distributed prior that covers entire range. In practice, however, it is possible to consider the first likelihood as a prior distribution.

7.2.4.2 Markov Chain Monte Carlo (MCMC) Simulation

The MCMC simulation is employed to realize the posterior distribution, which is introduced in Chap. 4. Since the MCMC method is a sampling-based one, it has to include enough samples so that the statistical characteristics of the distribution can be well captured. There are some methods available to determine the convergence condition, such as Adlouni et al. (2006) and Plummer et al. (2006). In this study, the convergence condition is determined graphically as the simplest method. It involves discarding the values at the initial stage of iteration, and monitoring the traces and histogram plots for later iterations from which the subjective judgment is made as to the convergence to a stationary chain. As an example of MCMC simulation, we consider a posterior distribution which is given as

$$p(\kappa, \sigma) \propto \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^5 \exp\left[-\frac{1}{2}\sum_{k=1}^{5}\frac{(V_k - \kappa F_n l_k)^2}{\sigma^2}\right]. \tag{7.10}$$

Equation 7.10 is the posterior distribution with noninformative prior when the error, ε, between obtained data (in situ wear volume, Vol_k) and estimated values from wear volume equation ($\kappa F_n l_k$) is normally distributed with zero mean and standard deviation, σ. Therefore, the measured wear volume can be represented by

$$\mathrm{Vol}_k = \kappa F_n l_k + \varepsilon, \quad \varepsilon \sim N\left(0, \sigma^2\right). \tag{7.11}$$

When n_y measured data are used, Eq. 7.10 can be written in a general form as

$$p(\kappa, \sigma) \propto \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{n_y} \exp\left[-\frac{1}{2}\sum_{k=1}^{n_y}\frac{(V_k - \kappa F_n l_k)^2}{\sigma^2}\right]. \tag{7.12}$$

If n_y is one, the equation is exactly same with normal distribution. Equation 7.10 is the case that n_y is five.

Figure 7.5 and Table 7.3 show the sampling result of $[\kappa, \sigma]$ for Eq. 7.10. Figure 7.5a represents traces of 10,000 iterations. As mentioned previously, the initial stages are discarded (burn-in) because they are not converged; the discarding value can be selected as an arbitrary value, in this case we select 4,000 as the discarding value (40 % of total samples). Figure 7.5b gives the estimated PDF with 6,000 samples starting from 4,001 iterations, while Fig. 7.5c shows the analytical PDF from Eq. 7.10. It can be seen that the MCMC sampling result follows the

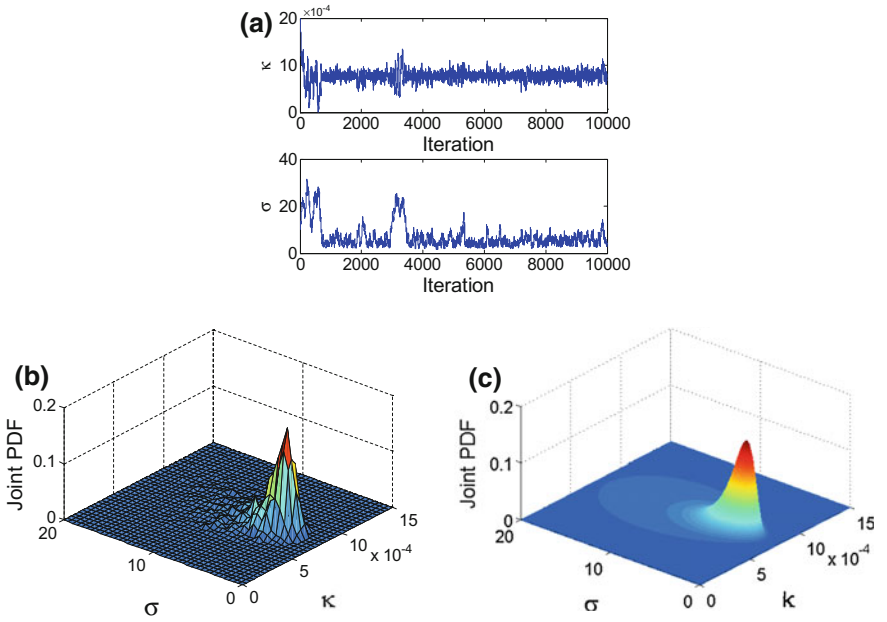


Fig. 7.5 Joint posterior PDF. **a** Trace of iteration, **b** using MCMC, **c** exact solution

Table 7.3 Statistical moments

	μ_κ	μ_σ	σ_κ	σ_σ	$\mathrm{cov}(\kappa, \sigma)$
MCMC	7.82×10^{-4}	5.61	0.93×10^{-4}	2.59	0
Exact sol.	7.76×10^{-4}	5.63	0.92×10^{-4}	2.51	0
Error (%)	0.79	0.47	1.09	3.13	0

analytical distribution quite well. Table 7.3 shows that the first statistical moments have less than 1 % error.

Once the samples are obtained from the posterior distribution of $[\kappa, \sigma]$, the samples of wear volume can be obtained by using Eq. 7.3 or Eq. 7.6. In Eq. 7.6, μ represents the wear volume which is obtained as a distribution due to the uncertainty of unknown parameter, κ, and the 90 % interval is defined as a confidence interval (CI). The level of prediction interval (PI) can also be calculated by adding the measurement error σ to the wear volume in Eq. 7.6.

7.2.5 *Identification of Wear Coefficient and Prediction of Wear Volume*

7.2.5.1 Posterior Distribution of Wear Coefficient

The posterior distributions of κ and σ are obtained as the results of five times updating using the MCMC technique with the first five sets of data in Tables 7.1 and 7.2. The last data set is remotely located and is used for the prediction validation. In the MCMC process, the number of iterations is fixed at 10,000. The resulting PDFs are given in Fig. 7.6. Both normal and lognormal distributions are considered for the likelihood, and both noninformative and normal distribution are considered for the prior. In the case of a noninformative prior, the PDF shape in Fig. 7.6c with the lognormal likelihood is narrower than that of Fig. 7.6a with the normal likelihood. As shown in the fifth set of Table 7.4, the standard deviation from the lognormal likelihood ($0.51 \times 10^{-4}\,\mathrm{mm^3/N\,m}$) is 43 % less than that of normal likelihood ($0.9 \times 10^{-4}\,\mathrm{mm^3/N\,m}$), whereas the mean values are nearly equal; i.e., 7.89 and 7.88, respectively. These results show that the lognormal likelihood exhibits increased accuracy over the normal likelihood. The reason that the former is better than the latter may be attributed to the nonnegative nature of the lognormal distribution, which is the case of the wear coefficient.

By comparing the results for the two priors, i.e., Figs. 7.6a versus b and 7.6c versus d, it is seen that the use of a normal prior leads to underestimation of the mean for κ. It is noted that the actual κ values at the last data set are found to vary between 5.03×10^{-4} and 12.71×10^{-4} (see Tables 7.1 and 7.2), of which the average is 8.5×10^{-4}. The reason that the results with a prior are worse than the noninformative results may attribute to the inaccurate prior distribution; as mentioned previously, the tribometer wear tests were performed under the uniform pressure condition, while the contact pressure in the bushing is not constant. In addition, the wear coefficient is not an intrinsic material property, but depends on contact pressure and contact area. It is also observed that the posterior distributions of κ in Fig. 7.6a, c are close to Laplacian distribution with heavy tails. Although the case with lognormal likelihood has narrower distribution, the shape of posterior distribution is close. On the other hand, the posterior distributions with the normal

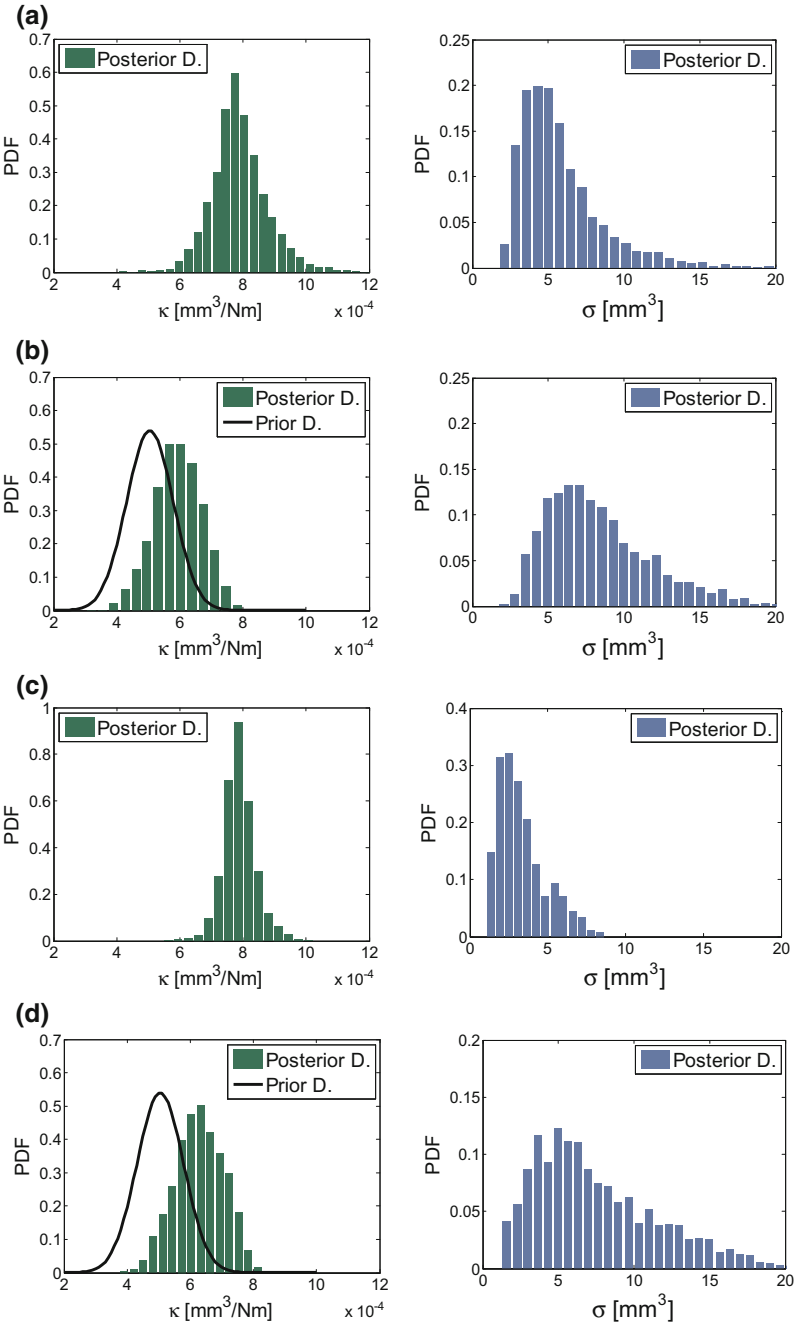


Fig. 7.6 Posterior distribution when using the first five data sets. **a** Normal likelihood with non-informative prior, **b** Normal likelihood with normal prior, **c** Lognormal likelihood with non-informative prior, **d** Lognormal likelihood with normal prior

Table 7.4 Mean and standard deviation of κ (10^{-4} mm^3/N m)

Likelihood	Prior	Data set no.	3	4	5	6
Normal	Non-inform	Mean	17.49	8.28	7.89	7.57
		Std.	10.02	2.32	0.90	0.40
	Prior	Mean	5.11	5.42	5.93	6.89
		Std.	0.73	0.73	0.76	0.57
Lognormal	Non-inform	Mean	20.81	8.89	7.88	7.64
		Std.	5.82	1.17	0.51	0.25
	Prior	Mean	5.46	5.67	6.29	7.41
		Std.	0.72	0.75	0.77	0.25

prior in Fig. 7.6b, d show quite different distribution shape from that of Fig. 7.6a, c. Therefore, it can be concluded that the prior contributes significantly to the posterior distribution. This is partly related to the fact that the prior information is not consistent with observed data.

In order to investigate the effects of the prior in more detail, the posterior distributions of κ are obtained after updating at each data set. Values of the mean and standard deviation are given in Table 7.4, and 5–95 confidence intervals (CI) as well as maximum likelihood values are plotted in Fig. 7.7. In Fig. 7.7, the star maker denotes the mean value of the distribution after the last update; i.e., using the sixth data set. This is used as a target value for correct prediction of the earlier stage. In Fig. 7.7a, b, the CIs with the normally distributed prior are located much lower than the ones without a prior (noninformative) and do not include the target value. What is remarkable from this observation is that, although the use of prior knowledge is usually recommended for reducing uncertainty and accelerating convergence, it should be used with caution. In this study, the wear coefficient is not an intrinsic material property but can vary with operating conditions. This turned out to be the cause of incorrect prediction, and should be avoided. It should be

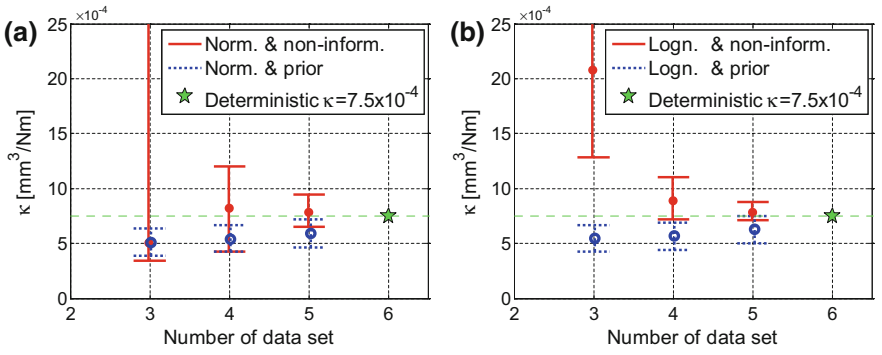


Fig. 7.7 Confidence intervals of the wear coefficient in terms of the data set number. **a** Likelihood: normal distribution, **b** likelihood: lognormal distribution

noted, however, that this is not always the case. Unlike the present case, if only limited data are available, the user may have to depend more on prior knowledge than the likelihood of the data.

7.2.5.2 Prediction of Wear Volume

Once the posterior distribution of κ and σ are obtained, the information can be used to predict the wear volume at the next stage. For that purpose, the wear volume for the sixth data set (at 20,585 cycles) is predicted using the posterior distribution for the wear coefficient from previous data sets. Due to uncertainty in κ and σ, the predicted wear volume becomes a probability distribution. Figure 7.8 shows 5–95 CI and maximum likelihood values of the wear volume distribution. In the figure, the actual value of wear volume measured at the sixth data set (92.73 mm^3) is used as a target value. As noted previously, the use of selected prior does not predict the wear volume well. Even the upper confidence limit is below the target value, which can cause unexpected failure if the value is carelessly used in the design decision.

Comparing the results of normal and lognormal likelihood, the size of the interval is quite large at the third stage of data in the case of the normal likelihood. However, the size of the interval is quickly reduced as the number of data set increases. Overall, the size of the interval of the lognormal likelihood is smaller than that of the normal likelihood.

In Fig. 7.9, the predicted distributions of the wear volume at the sixth stage using the posterior distributions from the fifth stage are given with different combinations of likelihoods and priors; the vertical line indicates the measured value. It can be seen that the result with the noninformative prior is better than that with the normally distributed prior. Due to the underestimation of the wear coefficient in the normally distributed prior, the predicted wear volumes with the prior are less than the actual one.

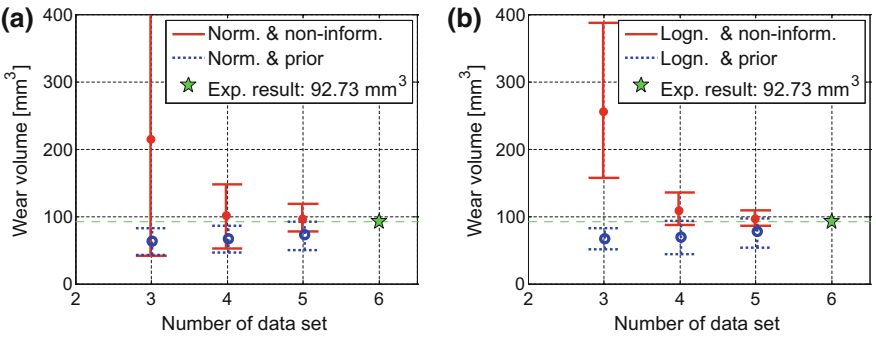


Fig. 7.8 Prediction intervals of the wear volume at 20,585 cycles in terms of the data set number. **a** Likelihood: normal distribution, **b** likelihood: lognormal distribution

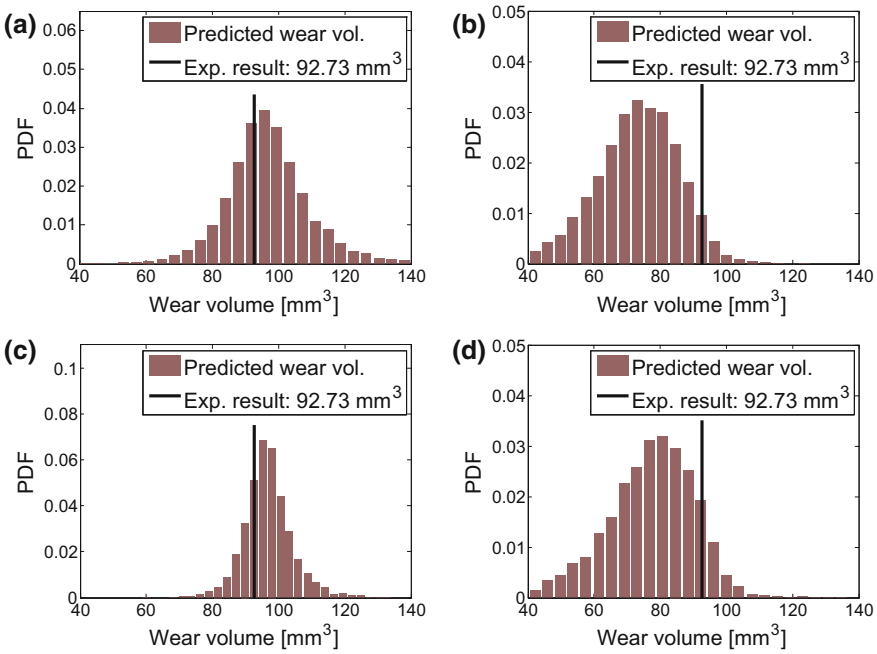


Fig. 7.9 Predicted distributions of wear volume at cycle = 20,565. **a** Likelihood: *N* and noninform, **b** likelihood: *N* and prior, **c** likelihood: *LN* and noninform, **d** likelihood: *LN* and prior

In Fig. 7.10, the CI and PI of the wear volumes at all stages are given using the posterior distribution at each stage along with the measured data (denoted by dots). In both cases of likelihoods, the CI and PI have a tendency to gradually decrease from 1,000 cycles as the cycles increase. The results at the first and second stages, which are 1 cycle and 100 cycles respectively, exhibit smaller intervals than other high cycles even if they are results with small number of data. The reason is that the variances of the parameters are large, but the wear volume itself is very small at low cycle compared to higher cycles. Although these early stages are not of interest in terms of prognosis, the estimation results are fairly exact. In Table 7.5, the numerical values of CI and PI are provided. The CIs and PIs of the lognormal case are smaller than that of the normal case, which demonstrates improved accuracy.

7.2.6 Discussion and Conclusions

In this study, the Bayesian inference technique is used to estimate the probability distribution of wear coefficients from in situ measurements. The first five sets of

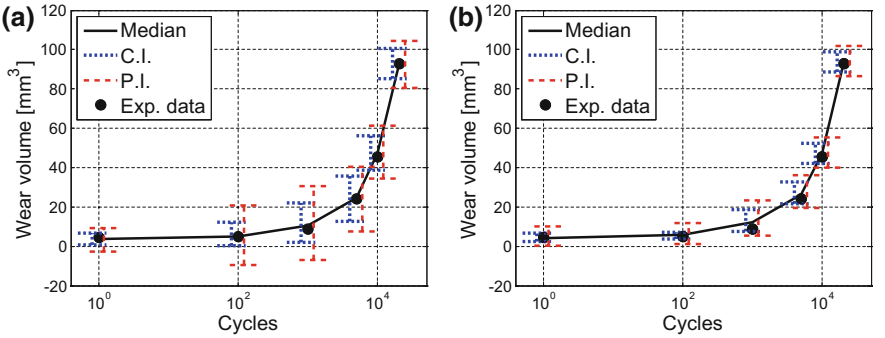


Fig. 7.10 Confidence and prediction intervals of wear volume. **a** Likelihood: normal distribution, **b** likelihood: lognormal distribution

Table 7.5 C.I. and P.I. of wear volume

Data set no.			1	2	3	4	5	6
Measured wear V			4.44	5.105	8.825	24.175	45.31	92.73
C.I.	N	95 %	6.61	12.13	22.14	35.75	56.25	100.38
		5 %	0.75	0.35	2.03	12.78	38.87	85.09
		Inter	5.86	11.77	20.11	22.97	17.38	15.28
	LN	95 %	6.61	7.06	18.73	32.88	52.27	99.04
		5 %	2.50	3.60	7.66	21.52	42.31	88.78
		Inter	4.11	3.46	11.07	11.36	9.96	10.26
P.I.	N	95 %	9.19	20.32	30.96	40.34	60.64	104.04
		5 %	−2.84	−9.18	−7.01	7.56	33.78	81.22
		Inter	12.03	29.50	37.97	32.78	26.87	22.82
	LN	95 %	9.90	12.22	23.72	36.06	55.18	101.86
		5 %	0.24	1.23	5.18	19.36	39.87	86.70
		Inter	9.66	10.99	18.53	16.70	15.30	15.15

data up to 10,000 cycles are used to reduce uncertainty in wear coefficient and the last set of data at 20,585 cycles is used for the prediction validation. The numerical results show that the posterior distribution with a noninformative prior is more accurate than that with the prior distribution from the literature. This result is obtained because the converged posterior distribution is quite different from the prior distribution. In order to predict the wear coefficient of a mechanical component, it has been suggested that the wear volume, slip distance, and applied load must be measured simultaneously.

7.3 Identification of Correlated Damage Parameters Under Noise and Bias Using Bayesian Inference[2]

7.3.1 Motivation and Background

In Chaps. 4 and 6, we discussed that the most important step of physics-based prognostics is to identify model parameters. In this section, statistical model parameter identification is presented based on the Bayesian method when parameters are correlated and observed data have noise and bias. For Paris crack growth model, it will be shown that a strong correlation exists (a) between two parameters of the Paris model, and (b) between initially measured crack size and bias. As the level of noise increases, the Bayesian inference is not able to identify the correlated parameters. However, the RUL is predicted relatively accurately. It will also be shown that the Bayesian identification process converges slowly when the level of noise is high.

In this study, a physics-based model for structural degradation due to fatigue damage is applied for prognostics, since damage grows slowly and the physics governing its behavior is relatively well known. The main purpose of this study is to present the usage of Bayesian inference in identifying model parameters and predicting the RUL—the remaining cycles before maintenance. The study focuses on crack growth in a fuselage panel under repeated pressurization loading, which can be considered regular loading cycles. In this type of application, the uncertainty in applied loading is small compared to other uncertainties. Therefore, the crack growth behavior and the RUL can be predicted based on the identified model parameters before the crack becomes dangerous. The improved accuracy in these model parameters allows more accurate prediction of the RUL of the monitored structural component.

Identifying the model parameters and predicting damage growth, however, is not a simple task due to the noise and bias of data from SHM systems and the correlation between parameters, which is prevalent in practical problems. The noise comes from variability of random environments, while the bias comes from systematic departures of measurement data, such as calibration error. However, research for identifying model parameters under noise and bias, without mentioning correlated parameters, is limited.

The main objective of this study is to demonstrate how Bayesian inference can be used to identify model parameters and to predict RUL, especially when the model parameters are correlated. In order to find the effects of noise and bias on the identified parameters, numerical studies utilize synthetic data; i.e., the measurement data are produced from the assumed model of noise and bias. The key interest is how the Bayesian inference identifies the correlated parameters under noise and bias in data.

---

[2]An D, Choi JH, Kim NH (2012) Identification of correlated damage parameters under noise and bias using Bayesian inference. Structural Health Monitoring 11(3):293–303.

This section is organized as follows: in Sect. 7.3.2, a simple damage growth based on Paris model is presented in addition to the uncertainty model of noise and bias; in Sect. 7.3.3, parameter identification and RUL prediction using the Bayesian method are presented with different levels of noise and bias; and conclusions are presented in Sect. 7.3.4.

7.3.2 *Damage Growth and Measurement Uncertainty Models*

7.3.2.1 Damage Growth Model

In this study, a simple damage growth model is used to demonstrate how to characterize damage growth parameters. It is assumed that a through-the-thickness center crack exists in an infinite plate under the mode I loading condition. Loading conditions and fracture parameters of 7075-T651 aluminum alloy are shown in Table 7.6. In Table 7.6, the two model parameters are assumed to be uniformly distributed, whose lower- and upper bounds were obtained from the scatter of experimental data (Newman et al. 1999). They can be considered as the damage growth parameters of generic Al 7075-T651 material. In general, it is well known that the two Paris parameters are strongly correlated (Sinclair and Pierie 1990), but it is assumed initially that they are uncorrelated because there is no prior knowledge on the level of correlation. Using measured data of crack sizes, the Bayesian inference will show the correlation structure between these two parameters. Since the scatter is so wide, the prediction of RUL using the initial distributions of parameters is meaningless. The specific panel being monitored using SHM systems may have much narrower distributions of the parameters, or even deterministic values.

7.3.2.2 Measurement Uncertainty Model

In SHM-based inspection, the sensors installed on the panel are used to detect the location and size of damage. A crack in the fuselage panel grows according to the applied load, pressurizations in this case. Then, the structural health monitoring (SHM) systems detect the crack. In general, the SHM system cannot detect a crack

Table 7.6 Loading and fracture parameters of 7075-T651 aluminum alloy

Property	Nominal stress $\Delta\sigma$ (MPa)	Fracture toughness (K_{IC} (MPa$\sqrt{\text{m}}$))	Damage parameter m	Damage parameter $\ln(C)$
Distribution type	Case 1: 86.5 Case 2: 78.6 Case 3: 70.8	Deterministic 30	$U(3.3, 4.3)$	$U(\ln(5e{-}11), \ln(5e{-}10))$

when it is too small. Many SHM systems can detect a crack as small as the size of 5–10 mm. Therefore, the necessity of identifying the initial small crack size becomes unimportant. In practice when SHM system detects a crack with size a_0, this can be considered as the initial crack size with that time being the initial cycle. However, a_0 may still include noise and bias from the measurement. In addition, the fracture toughness, K_{IC}, is also unimportant because airliners may want to send the airplane for maintenance before the crack becomes critical.

The main objective of this study is to show that the measured data can be used to identify crack growth parameters, and then, to predict the future behavior of the cracks. We simulate the measured crack sizes from SHM. In general, the measured damage includes the effect of bias and noise. The former is deterministic and represents a calibration error, while the latter is random and represents noise in the measurement environment. The synthetic measurement data are useful for parameter study, that is, the different noise and bias levels show how the identification process is affected. In this context, bias is considered as two different levels, ± 2 mm, and noise is uniformly distributed between $-v$ mm and $+v$ mm. Four different levels of v are considered: 0, 0.1, 1, 5 mm. The varying levels of noise represent the quality of SHM systems.

The synthetic measurement data are generated by (a) assuming that the true parameters, m_{true} and C_{true}, and the initial half crack size, a_0, are known; (b) calculating the true crack sizes according to Eq. 4.19 for a given N_k and $\Delta\sigma$; and (c) adding a deterministic bias and random noise to the true crack size data including the initial crack size. Once the synthetic data are obtained, the true values of crack sizes as well as the true values of parameters are not used in the prognostics process. In this study, the following true values of parameters are used for all numerical examples: $m_{\text{true}} = 3.8$, $C_{\text{true}} = 1.5 \times 10^{-10}$, and $a_0 = 10\,\text{mm}$.

Table 7.6 shows three different levels of loading; the first two ($\Delta\sigma = 86.5$ and 78.6 MPa) are used for estimating model parameters, while the last ($\Delta\sigma = 70.8$) is used for validation purposes. The reason for using two sets of data to estimate damage growth parameters is to utilize more data having damage propagation information at an early stage. Theoretically, the true values of parameters can be identified using a single set of data because the Paris model is a nonlinear function of parameters. However, random noise can make the identification process slow, especially when parameters are correlated; i.e., many different combinations of correlated parameters can achieve the same crack size. This property delays the convergence of Bayesian process such that meaningful parameters can only be obtained toward the end of RUL. Based on preliminary study, two sets of data at different loadings can help the Bayesian process converge quickly. This situation corresponds to two fuselage panel with different thickness.

Figure 7.11 shows the true crack growth curves for three different levels of loading (solid curves) and synthetic measurement data a_i^{meas} (triangles) with noise and bias. It is noted that the positive bias shifts the data above the true crack growth. On the other hand, the noises are randomly distributed between measurement cycles. It is assumed that the measurements are performed at every 100 cycles. Let

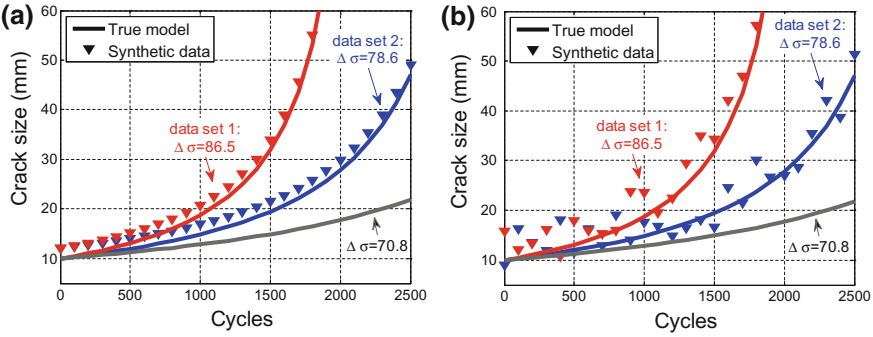


Fig. 7.11 Crack growth of three different loading conditions and two sets of synthetic data. **a** bias = +2 mm and noise = 0 mm, **b** bias = +2 mm and noise = 5 mm

there be n_y measurement data. Then, the measured crack sizes and corresponding cycles are represented by

$$\begin{aligned} \mathbf{a}^{\text{meas}} &= \{a_0^{\text{meas}}, a_1^{\text{meas}}, a_2^{\text{meas}}, \ldots, a_{n_y}^{\text{meas}}\} \\ \mathbf{N} &= \{N_0 = 0, N_1 = 100, N_2 = 200, \ldots, N_{n_y}\}. \end{aligned} \tag{7.13}$$

It is assumed that after N_{n_y}, the crack size becomes larger than the threshold and the crack is repaired.

7.3.3 *Bayesian Inference for Characterization of Damage Properties*

7.3.3.1 Damage Growth Parameters Estimation

Once the synthetic data (damage sizes vs. cycles) are generated, they can be used to identify unknown damage growth parameters. As mentioned before, m, C, and a_0 can be considered as unknown damage growth parameters. In addition, the bias and noise are also unknown because they are only assumed to be known in generating crack size data. In the case of noise, the standard deviation, σ, of the noise and deterministic bias, b, are considered as an unknown parameters. The identification of σ will be important as the Bayesian process depends on it. Therefore, the objective is to identify (or, improve) these five parameters using the measured crack size data. The vector of unknown parameters is defined by $\boldsymbol{\theta} = \{m, C, a_0, b, \sigma\}$.

Bayesian method introduced in Chap. 4 is used to identify the unknown parameters as well as the level of noise and bias based on the measured crack sizes

$\mathbf{a}^{\text{meas}}$. The joint posterior PDF is obtained by multiplying the prior PDF with the likelihood as

$$f(\boldsymbol{\theta}|\mathbf{a}^{\text{meas}}) \propto f(\mathbf{a}^{\text{meas}}|\boldsymbol{\theta}) f(\boldsymbol{\theta}) \tag{7.14}$$

For prior distribution, the uniform distribution is used for the damage growth parameters, m and C, as described in Table 7.6. For other parameters, no prior distribution is used; i.e., noninformative. Therefore, the prior PDF becomes $f(\boldsymbol{\theta}) = f(m)f(C)$. The likelihood is the probability of obtaining the observed crack sizes $\mathbf{a}^{\text{meas}}$ given values of parameters. For the likelihood, it is assumed to be a normal distribution for the given five parameters including the standard deviation of the measured size, σ:

$$f(\mathbf{a}^{\text{meas}}|\boldsymbol{\theta}) \propto \left(\frac{1}{\sqrt{2\pi}\theta_5}\right)^{n_y} \exp\left[-\frac{1}{2}\sum_{i=1}^{n_y}\left(\frac{a_i^{\text{meas}} - a_i(\boldsymbol{\theta}_{1:4})}{\theta_5}\right)^2\right], \tag{7.15}$$

where $\boldsymbol{\theta} = \{m, C, a_0, b, \sigma\}$ and

$$a_i(\boldsymbol{\theta}_{1:4}) = \left[N_i C\left(1 - \frac{m}{2}\right)\left(\Delta\sigma\sqrt{\pi}\right)^m + a_0^{1-\frac{m}{2}}\right]^{\frac{2}{2-m}} + b \tag{7.16}$$

is the crack size from the Paris model with bias and a_i^{meas} is the measurement crack size at cycle N_i. In general, it is possible that the normal distribution in Eq. 7.15 may have a negative crack size, which is physically impossible; therefore, the normal distribution is truncated at zero.

A primitive way of computing the posterior PDF is to evaluate Eq. 7.14 at a grid of points after identifying the effective range. This method, however, has several drawbacks such as the difficulty in finding correct location and scale of the grid points, the spacing of the grid, and so on. Especially, when a multivariable joint PDF is required, which is the case in this study, the computational cost is proportional to M^5, where M is the number of grids in one dimension. On the other hand, the MCMC simulation can be an effective solution as it is less sensitive to the number of variables (Andrieu et al. 2003). Using the expression of posterior PDF in Eq. 7.14, 5,000 samples of parameters are drawn by using the Metropolis–Hastings (M–H) algorithm, which is a typical method of MCMC.

7.3.3.2 The Effect of Correlation Between Parameters

Since the original data of crack sizes are generated from the assumed true values of parameters, the objective of Bayesian inference is to make the posterior joint PDF converge to the true values. Therefore, it is expected that the PDF becomes narrower as n_y increases; i.e., more data are used. This process seems straightforward, but the preliminary study shows that the posterior joint PDF may converge to

values different from the true ones. It is found that this phenomenon is related to the correlation between parameters, which is explored in Chap. 6. There are many correlations between unknown parameters, but only the strongest correlated relationships are considered: the two Paris model parameters and a_0 and b. First, it was well known that the two Paris model parameters, m and C, are strongly correlated (Carpinteri and Paggi 2007). Their true values can be identified well with only three data with perfect data in Chap. 6. However, the embedded noise can make it difficult to identify the two model parameters because the crack growth rate may not be consistent with the noisy data. In addition, this can slow down the convergence in the posterior distribution, because when the crack is small, there is no significant crack growth rate. The effect of noise is relatively diminished as the crack growth rate increases, which occurs toward the end of life (EOL). Next, let the initially detected crack size be $\bar{a}_0^{\text{meas}}$ when the measurement environment has no noise. This measured size is the outcome of the initial crack size and bias:

$$\bar{a}_0^{\text{meas}} = a_0 + b. \tag{7.17}$$

Therefore, there exist infinite possible combinations of a_0 and b to obtain the measured crack size. It is generally infeasible to identify the initial crack size and bias with a single measurement when the measured data is linearly dependent on both parameters.

In order to overcome the above-mentioned difficulty in identifying correlated parameters, two different strategies are used in this study. First, the two Paris model parameters are kept because they can be identified as the crack grows. Second, the relationship between a_0 and b has different characteristics from the model parameters (m, C). a_0 and b are time independent, and the sum of the two parameters are constant. Therefore, the bias is removed from the Bayesian identification process using Eq. 7.17, assuming that the bias and initial crack size are perfectly correlated. This process seems straightforward, but the difficulty exists from the fact that the constant $\bar{a}_0^{\text{meas}}$ in Eq. 7.17 is unknown. Below is the procedure of estimating $\bar{a}_0^{\text{meas}}$.

(a) Assume that the measured initial crack size is $a_0 = a_0^{\text{meas}}$
(b) With given a_0, use the Bayesian method to update the posterior joint PDF of m, C, b, σ
(c) Calculate the maximum likelihood value b^* of b from the posterior joint PDF
(d) Estimate $\bar{a}_0^{\text{meas}} = a_0 + b^*$
(e) Eliminate b using $b = \bar{a}_0^{\text{meas}} - a_0$ and update the posterior joint PDF of m, C, a_0, σ.

Figure 7.12 shows the posterior PDFs for the case of true bias of +2 mm (a) when $n_y = 13$ (N_{12} = 1,200 cycles) and (b) when $n_y = 17$ (N_{16} = 1,600 cycles). The posterior joint PDFs are plotted separately into three groups for the plotting purpose. In this case, it is assumed that there is no noise in the data. The true values of parameters are marked using a star symbol. Similar results were also obtained in the case with bias = −2 mm. First, it is clear that the two Paris model parameters are strongly correlated. The same is true for the initial crack size and bias; in fact, the

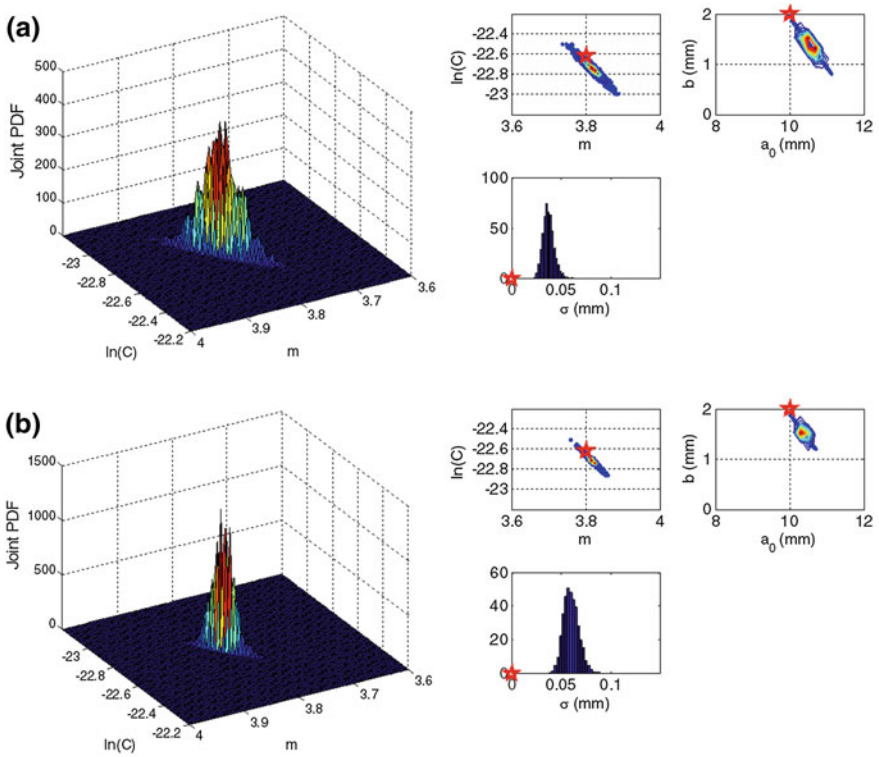


Fig. 7.12 Posterior distributions of parameters with zero noise and true bias of +2 mm. **a** $n_y = 13$, **b** $n_y = 17$

PDF of bias is calculated from Eq. 7.17 with the initial crack size. Second, it can be observed that the PDFs at $n_y = 17$ is narrower than that of $n_y = 13$, although the PDFs at $n_y = 13$ is quite narrow compared to the prior distribution. Lastly, the identified results look different from the true values due to the scale, but the errors between the true values and the median of identified results are at a maximum of around 5 %, except for bias. The error in bias looks large, but that is because the true value of bias is small. The error in bias is about 0.5 mm. The same magnitude of error exists for the initial crack size due to the perfect correlation between them. Table 7.7 lists all six cases considered in this study, and all of them show a similar level of errors. It is noted that the identified standard deviation of noise, σ, does not converge to its true value of zero. Zero noise data can cause a problem in the likelihood calculation, as the denominator becomes zero in Eq. 7.15. However, this would not happen for practical cases in which noise always exists.

The next example is to investigate the effect of noise on the posterior PDFs of parameters. The results of identified posterior distributions with different levels of noise are shown in Fig. 7.13 when the true bias was +2 mm. Similar results were obtained when bias is −2 mm. The black, blue and red colors, respectively,

Table 7.7 The median of identified parameters and the errors with the true values

		$n_y = 13$				$n_y = 15$				$n_y = 17$			
		m	$\ln(C)$	a_0	b	m	$\ln(C)$	a_0	b	m	$\ln(C)$	a_0	b
True values		3.8	−22.6	10	±2	3.8	−22.6	10	±2	3.8	−22.6	10	±2
$b = +2$mm	Median	3.82	−22.8	10.6	1.37	3.81	−22.7	10.4	1.53	3.82	−22.7	10.4	1.52
	Error (%)	0.49	0.57	5.67	31.7	0.32	0.37	4.00	23.6	0.47	0.44	3.84	24.2
$b = -2$mm	Median	3.78	−22.5	9.50	−1.44	3.78	−22.5	9.51	−1.41	3.78	−22.5	9.49	−1.35
	Error (%)	0.40	0.50	4.96	28.0	0.40	0.48	4.94	29.5	0.55	0.55	5.11	32.7

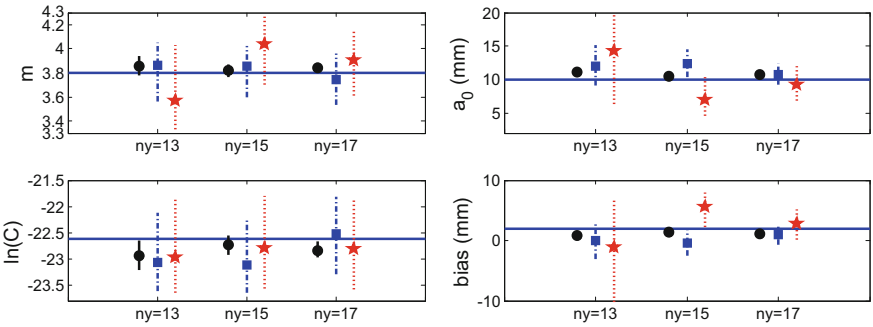


Fig. 7.13 Posterior distributions with three different levels of noise (bias = +2 mm)

represent noise levels of 0.1, 1, and 5 mm. The median location is denoted by a symbol (a circle for 0.1 mm noise, a square for 1 mm noise, and a star for 5 mm noise). Each vertical line represents a 90 % CI of posterior PDF. The solid horizontal line is the true value of the parameter. In the case of noise level = 0.1 mm, all parameters were identified accurately with very narrow CIs. In the case of noise level = 1 mm, the initial crack size and bias were identified accurately as the number of data is increased, whereas the CIs of two Paris parameters were not reduced. In addition, the median values were somewhat different from the true parameter values. This happened because the large noise prevents identifying correlated parameters accurately. Increasingly inaccurate results were observed as the level of noise increased to 5 mm. Therefore, it is concluded that the level of noise plays an important role in identifying correlated parameters using Bayesian inference. However, this does not mean that it is not able to predict RUL. Even if these parameters were not accurately identified because of correlation, the predicted RUL was relatively accurate, which will be discussed in detail next subsection.

7.3.3.3 Damage Propagation and RUL Prediction

Once the parameters are identified, they can be used to predict the crack growth and estimate RUL. Since the joint PDF of parameters are available in the form of 5,000 sample, the crack growth and RUL will also be estimated using the same number of sample. First, using 5,000 sets of parameters obtained from the MCMC method, Eq. 4.19 is utilized to calculate 5,000 numbers of crack size a_k after N_k cycles. Then, random measurement errors are added to the predicted crack sizes. For that purpose, 5,000 samples of measurement errors are generated from normal distribution with zero mean and identified 5,000 samples of σ. Then, the quality of prediction can be evaluated in terms of how close the median is to the true crack growth and how large the PI is. The results of crack growth are shown in Fig. 7.14 when the true bias is +2 mm. Different colors represent the three different loading conditions. The solid curves are true crack growth, while the dashed curves are

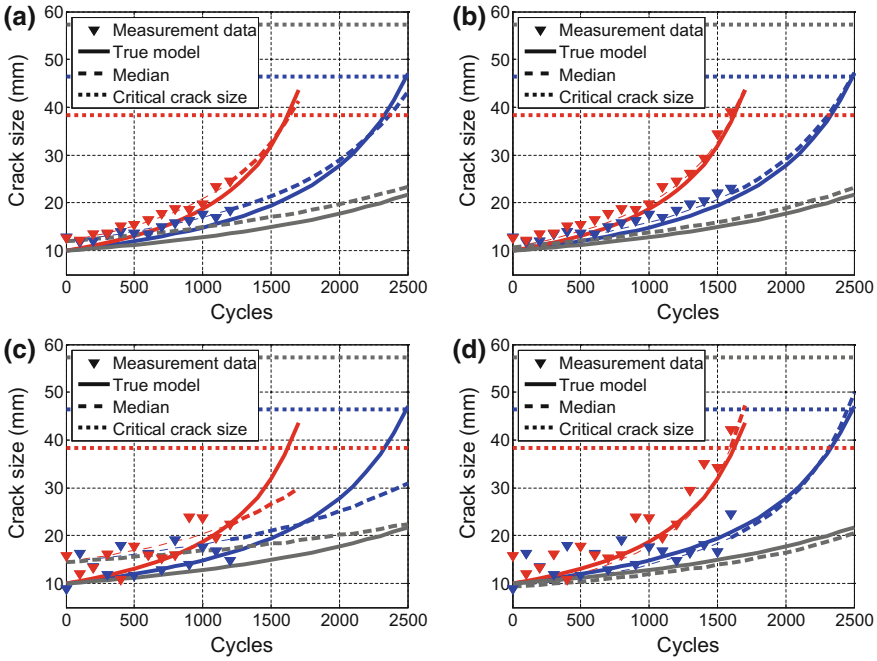


Fig. 7.14 Prediction of crack growth with bias = +2 mm. **a** noise = 1 mm, $n_y = 13$, **b** noise = 1 mm, $n_y = 17$, **c** noise = 5 mm, $n_y = 13$, **d** noise = 5 mm, $n_y = 17$

medians of predicted crack growth distribution. The results are obtained as a distribution due to the uncertainty in parameters, but the medians of predicted crack growth are only shown in the figures for visibility. In addition, the critical crack sizes with different loadings use horizontal lines. Different from the previous chapters, different critical crack sizes are used here for different loading conditions. Since the posterior distributions of parameters are symmetric, there was no difference between the mean, mode, and maximum likelihood estimates.

Figure 7.14 shows that the results closely predicted the true crack growth when noise is less than 1 mm. Even if the level of noise is 5 mm, the results of predicted crack growth become close to the true one as the number of data increases. This means that if there are many data (much information about crack growth), the future crack growth can be predicted accurately, even if there is much noise. However, when the level of noise is large, the convergence is slow such that the accurate prediction happened almost at the EOL.

As can be seen from Fig. 7.14, crack growth and RUL can be predicted with reasonable accuracy, even though the true values of the parameters are not accurately identified. The reason is that the correlated parameters m and C work together to predict crack growth in Eq. 4.19. For example, if m is underestimated, then the Bayesian process overestimates C to compensate for it. In addition, if there is large noise in the data, the distribution of estimated parameters becomes wider, which

can cover the risk that comes from the inaccuracy of the identified parameters. Therefore it is possible to safely predict crack growth and RUL.

In order to see the effect of the noise level on the uncertainty of predicted RUL, Fig. 7.15 plots the median and 90 % PI of the RUL and compared them with the true RUL. The RUL can be calculated by solving Eq. 4.19 for N when the crack size becomes critical; $a = a_C$:

$$N_f = \frac{a_C^{1-m/2} - a_k^{1-m/2}}{C(1-\frac{m}{2})(\Delta\sigma\sqrt{\pi})^m}. \tag{7.18}$$

The RUL is also expressed as a distribution due to the uncertainty of the parameters, which is obtained by replacing a_k and model parameters, m and C in Eq. 7.18 with 5,000 predicted crack growth and identified model parameters. In Fig. 7.15, the solid diagonal lines are the true RULs at different loading conditions $(\Delta\sigma = 86.5, 78.6, 70.8)$. The precision and accuracy are fairly good when the noise is less than 1 mm, which is consistent with the crack growth results. In the case of a large noise, 5 mm, the medians are close to the true RUL, and the wide intervals are gradually reduced as more data are used. Therefore, it is concluded that the RULs are predicted reasonably in spite of large nose and bias in data.

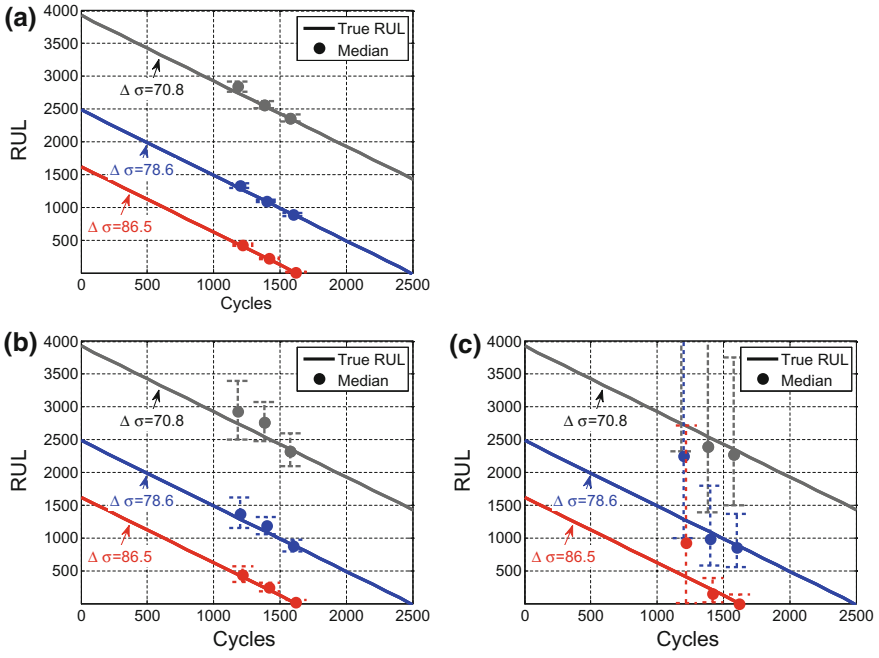


Fig. 7.15 Median and 90 % of prediction interval of the predicted RUL (bias = +2 mm). **a** noise = 0.1 mm, **b** noise = 1 mm, **c** noise = 5 mm

7.3.4 Conclusions

In this study, Bayesian method is used for identifying the Paris model parameters that govern the crack growth in an aircraft panel using SHM systems that measure crack sizes with noise and bias. Focuses have been given to the effect of correlated parameters and the effect of noise and bias. The correlation between the initial crack size and bias was explicitly imposed using analytical expression, while the correlation between two Paris parameters was identified through Bayesian inference. It is observed that the correlated parameter identification is sensitive to the level of noise, while predicting the RUL is relatively insensitive to the level of noise. It is found that greater numbers of data are required to narrow the distribution of parameters when the level of noise is high. When parameters are correlated, it is difficult to identify the true values of the parameters, but the correlated parameters work together to predict accurate crack growth and RUL.

7.4 Usage of Accelerated Test Data for Predicting Remaining Useful Life at Field Operating Conditions[3]

Prognostics predicts future damage/degradation and the RUL of in-service systems based on damage data obtained during previous usage. General prognostics methods are physics-based approaches when physical models and loading conditions are available and data-driven approaches when only the damage data are available. Damage data are of great importance regardless of prognostics methods used, but it is expensive to obtain data from in-service systems because of time and cost. Instead, companies frequently perform accelerated life tests under severe operating conditions for the purpose of design validation. This section presents a method of utilizing degradation data obtained during accelerated life tests for the purpose of prognostics. As an example, crack growth data are synthetically generated under overloaded conditions, which are utilized to predict damage growth and RUL at field operating conditions. Four different scenarios are considered based on the availability of a physical model and field loading conditions. Using accelerated test data increases prediction accuracy in the early stages of physics-based prognostics and also compensates for the insufficient data problem in data-driven prognostics.

[3]An D, Choi JH, Kim NH (2013) Practical use of accelerated test data for the prognostics methods, Annual conference of the Prognostics and Health Management Society, Oct. 14–17, 2013, New Orleans, LA.

7.4.1 Motivation and Background

Prognostics predicts future damage/degradation and the RUL of in-service systems based on the damage data obtained during previous usage. Even though prognostics facilitates condition-based maintenance that is considered as a cost-effective maintenance strategy compared to periodic preventive maintenance, there are several challenges in order for it to be viable in practice. One of them to be considered in this study is a limited number of damage data in-service systems, since it is expensive to obtain data due to time and cost. Instead, it is easy to obtain damage data from accelerated life tests under severer operating conditions than in-service ones, since companies frequently use them for the purpose of design validation.

Even though studies on life estimation using accelerated life test data are available in the literature (Nelson 1990; Park and Bae 2010), they are mostly used to estimate the average life of a fleet of systems, not the life of a specific system that can experience different loading histories from other systems. It is difficult to use them for the purpose of prognostics because in-service usage conditions are not reflected. Prognostics may produce too conservative life prediction if the accelerated life test data are directly used as degradation data by considering accelerated loading conditions as field operation conditions. This issue has been recognized by researchers in the prognostics field (Celaya et al. 2011). As another case of utilization of accelerated life test data, the data are used to build a degradation model when physical models are not available (Skima et al. 2014). In such a case, however, the degradation model is good for predicting the RUL at the accelerated loading conditions, but not for the nominal operating conditions.

This study presents several methods of utilizing progressive damage data obtained from accelerated life tests in predicting the RUL of a particular system operating in the field. It is assumed that during the accelerated life tests, sensors are placed in the system or periodic inspections are performed such that the history of damage growth is measured until the damage grows up to a threshold. The threshold is determined at which the system cannot be operated further without maintenance. The time when the damage grows to the threshold is called the EOL. It is also assumed that the current system in the field has a similar damage data up to the current time. Then the objective is to predict the EOL of the current system so that maintenance can be ordered before the system reaches inoperable situation. Since this prediction involves numerous uncertainty, the prediction needs to be given in the framework of probability. Four different scenarios are considered based on available information, such as physical models and field loading conditions. Especially, this study focuses on the last scenario when neither a physical model nor loading conditions are available and presents a mapping method to compensate for the lack of accuracy due to insufficient data in data-driven approaches.

This section is organized as follows: in Sect. 7.4.2, the problem is defined with a crack growth example; in Sect. 7.4.3, it is suggested how accelerated condition data can be used under different scenarios, followed by conclusions in Sect. 7.4.4.

7.4.2 Problem Definition

7.4.2.1 Prognostics Approaches

Physics-based approaches assume that a physical model describing the behavior of damage is available and combine the physical model with measured data to predict the future behavior of damage. The physical degradation model is expressed as a function of usage conditions, elapsed cycle or time and model parameters. Normally, usage conditions and time are given, while the model parameters should be identified. Since the behavior of damage depends on model parameters, identifying them is the most important issue in predicting future damage behaviors. Among several algorithms, the particle filter (PF) explained in Chap. 4 is used in this study.

Data-driven approaches use information from collected data (training data) to identify the damage state and predict the future state without using any specific physical model. In general, previous damage states are employed for the input variables, which allow to create a relation between damage data without requiring information from physical model or loading conditions. The selection of input variables, however, is flexible: for example, loading information can be included if it is available, which increases the accuracy of prediction.

Among several algorithms, the neural network (NN) explained in Chap. 5 is used, but the Bayesian batch method is employed in this study. Therefore, the weights and biases are identified as distributions based on Bayes' theorem instead of deterministic values given by an optimization process. In implementation, the network model is constructed with two input nodes (the previous two damage data sets are used for input variables), one hidden node with one hidden layer and one output node, and the tangent sigmoid and pure linear functions are employed. Two types of predictions are considered: short-term and long-term predictions. The former is to predict just one step ahead; that is, the damage state at time $k+1$ is predicted using measured damage information up to time k. On the other hand, the latter is to predict several steps ahead by using the predicted damage states for the input instead of real measurement data, which is implemented in Chap. 5.

The performance of data-driven approaches depends on the number and quality of training data. If damage growth data of the same system under same loading condition are available, then the data-driven approaches can predict the damage growth of the current system accurately. In practice, however, such data are rarely available. An important objective of this study is to show that accelerated life test data can be used as training data for data-driven approaches. Then, a technical challenge is that how to use the accelerated life test data, which are obtained in severer loading conditions, can be used for training the current system that is being operated under nominal loading conditions.

7.4.2.2 Crack Growth Example

To illustrate the main idea of this study, a crack growth problem in a fuselage panel of aircraft under repeated pressurization loadings is employed, whose damage growth is based on the Paris model given in Eq. 2.1. Synthetic measurement data are generated using Eq. 2.2/4.19 with assumed true damage growth parameters, $m_{\text{true}} = 3.5$ and $C_{\text{true}} = 6.4 \times 10^{-11}$, and the initial half crack size, $a_0 = 10$ mm. These true parameters are only used to generate the synthetic data. It is assumed that accelerated life tests are performed under three high stress ranges, $\Delta\sigma = 145$, 140, and 135 MPa, and the real operating condition is under $\Delta\sigma = 68$ MPa. In order to simulate the real-life situation, a uniformly distributed random noise between $\pm v$ mm is added to the synthetic data generated using Eq. 4.19. Since accelerated life test environment in a laboratory is more controlled than that of a field, $v = 0.7$ mm is used for the accelerated life test data, while $v = 1.5$ mm for the field operating condition. Figure 7.16 shows synthetic data that are used in the numerical example. The three sets of accelerated life test data in Fig. 7.16a are utilized to increase the accuracy of damage prediction of an in-service system, shown as star markers in Fig. 7.16b. The uniqueness of the current work is to use the accelerated test data in Fig. 7.16a to predict the damage state under field operating conditions with much lower stress levels in Fig. 7.16b.

7.4.3 Utilizing Accelerated Life Test Data

Four methods of utilizing accelerated life test data are presented in the following subsections. They are based on the availability of a physical model and field loading conditions, as listed in Table 7.8.

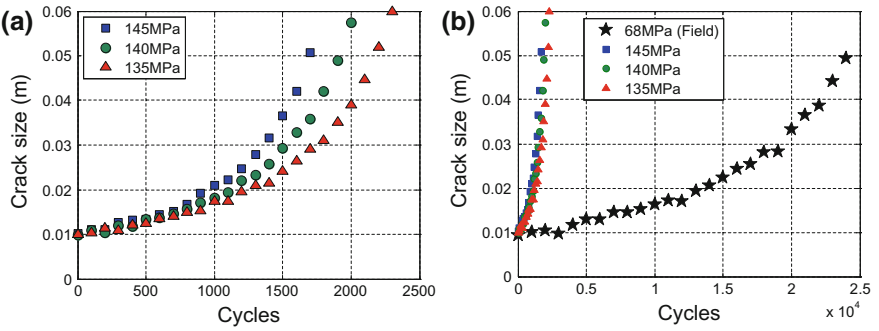


Fig. 7.16 Synthetic crack growth data under different loading conditions with random noise. **a** Accelerated life tests, **b** field operating conditions

Table 7.8 Four scenarios considered in the numerical study

	Case 1	Case 2	Case 3	Case 4
Physical model	O	O	X	X
Field loading	O	X	O	X
Available method	Physics-based (PF)		Data-driven (NN)	

7.4.3.1 Case 1: When Both Physical Model and Loading Conditions Are Available

This is the conventional case of physics-based prognostics using the Paris model given in Eq. 2.1. In such a case, accelerated life test data are unnecessary for prognosis; that is, the future damage state can be predicted using only the field data. The accelerated test data, however, can be used to improve prognostics accuracy and to reduce uncertainty in the early stage because the additional data can give better prior information for the damage parameters.

In order to illustrate the effect of accelerated life test data, it is assumed that the Paris model parameters, m and C, are unknown for a specific fuselage panel of interest. Therefore, it is natural to start with the values of these parameters from a generic Al 7075-T651 material. From Newman et al. (1999), it is assumed that these material parameters are uniformly distributed due to manufacturing variability: $m \sim U(3.3, 4.3)$ and $\ln(C) \sim U(\ln(5 \times 10^{-11}), \ln(5 \times 10^{-10}))$. A total of 5,000 random samples of these parameters are shown in Fig. 7.17a. The star mark in the figure represents the true value of the parameters, which needs to be found using crack growth measurement data. The initial distribution of model parameters in Fig. 7.17a is so large that the crack growth prediction using them does not provide useful information.

The objective is to narrow the initial parameter distribution by incorporating the accelerated life test data into the PF. Figure 7.17b shows the narrowed distribution using the three accelerated life test data in Fig. 7.16a. It is noted that the scatter of distribution is much narrower than that of the initial distribution. It is also noted that

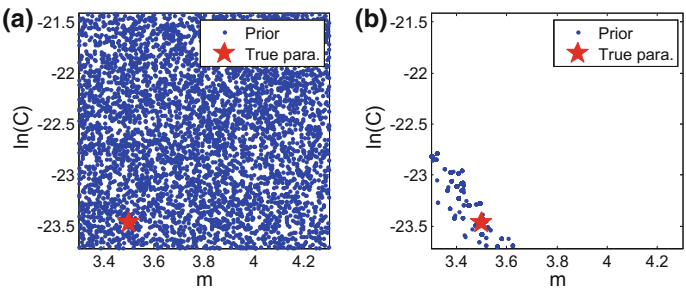


Fig. 7.17 Prior distribution of two Paris model parameters. **a** From the literature, **b** from accelerated life test

due to a strong correlation between the two Paris parameters, the Bayesian process cannot identify the true value of the parameters. Instead, the process converges to a narrow band between the two parameters. As pointed out in Sect. 7.3, the crack propagation behavior will be similar for different combinations of the two Paris parameters along this band. The advantage of accelerated life test data is that the prognostics of the current system can start with much better initial distribution, as shown in Fig. 7.17b.

For the purpose of comparison, the two distributions in Fig. 7.17a, b are used as an initial distribution, and the PF is used to update the distributions at each measurement cycle with the data in the field operating condition. Then, the updated distributions are used to predict the distribution of RUL, as shown in Fig. 7.18a, b. Due to the uncertainty in model parameters, the predicted RUL is also uncertain, and the figure shows the median as well as 90 % PIs. Figure 7.18b is the case when the prior information from the accelerated life test data is used, which shows fairly accurate and precise results starting from an early stage. Also, Table 7.9 shows the difference between Fig. 7.18a, b quantitatively based on the prognostics metrics (Saxena et al. 2009) that is introduced in Chap. 2, in which the performance is better when the values of prognostics metrics are larger.

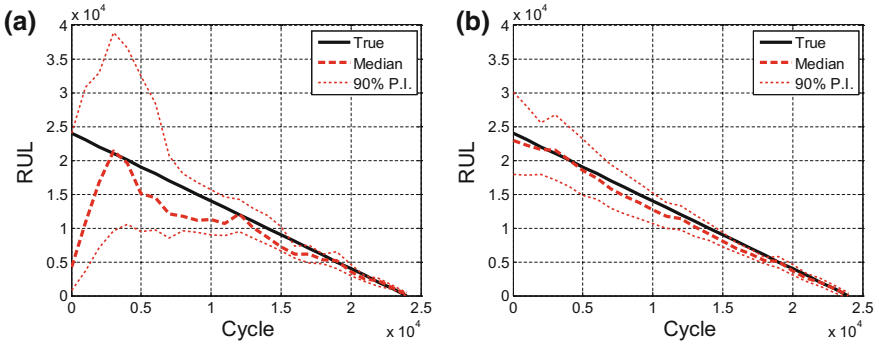


Fig. 7.18 RUL prediction of Case 1. **a** Based on prior from literature, **b** based on prior from accelerated life tests

Table 7.9 Prognostics metrics for Case 1

	PH ($\alpha = 10$ %)	$\alpha - \lambda$ accuracy ($\alpha = 10$ %, $\lambda = 0.5$)	RA ($\lambda = 0.5$)	CRA
Prior from literature	11,000	False (0.2504)	0.9161	0.8479
Prior from test data	**22,000**	**True (0.6046)**	**0.9240**	**0.9285**

7.4.3.2 Case 2: When Only Physical Model Is Available but not Loading Condition

In prognostics, loading conditions are as important as physics model because the loading conditions are input to the model. In the accelerated life tests, it is a standard process that a specific load (or the history of loads) is applied to the system. In the field operation, however, it is difficult to measure the real operating loads, and the loads may be different at different times. Therefore, the second case is when the physics model is available (its model parameters are still unknown) but the real operating loadings are unknown. In the following, the case when the real loading is constant but its magnitude is unknown is considered.

When real operating loading is uncertain, it makes a sense to assume its range. It is assumed that the uncertain loadings are uniformly distributed between 50 and 90 MPa, while the true loading is 68 MPa. With a given physical model, there are three ways to predict RUL: (A) updating both parameters and uncertain loadings, (B) updating loadings with a given distribution of parameters, and (C) updating parameters with given uncertain loadings. However, the last case, Case 2-C, does not make sense because there are no ways to reduce the uncertainty from uncertain loadings. Therefore, Case 2-A and Case 2-B are considered in the following.

For Case 2-A, the accelerated test data are used to improve the prior distribution of the two parameters, and then, the field data are used to update both the uncertain loading and the two parameters. Therefore, Bayesian inference updates the joint PDF of all three variables. Comparison of RUL prediction using the two different priors (Fig. 7.17a, b) as in the previous section is shown in Fig. 7.19 and Table 7.10. Although it is difficult to figure out the differences between the two clearly, numerical results in Table 7.10 show the case using prior from accelerated life test data yields slightly better results than that using the prior from literature. It is interesting that using a much narrower prior only slightly improves the RUL prediction. It is because of the correlation between model parameters and loading. Even if the case with a uniform prior from literature may converges slower in

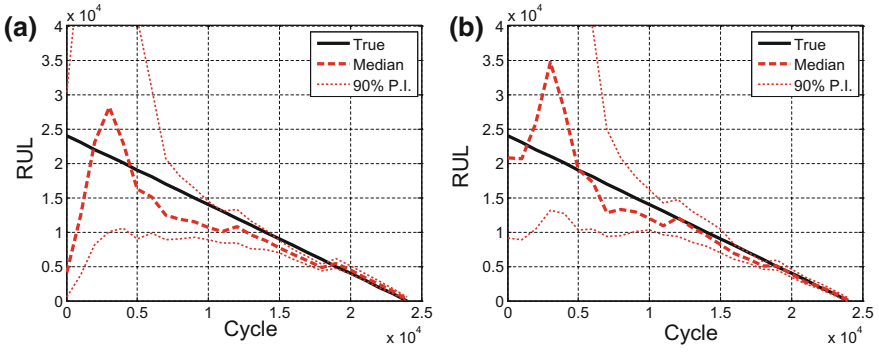


Fig. 7.19 RUL prediction of Case 2-A. **a** Prior from literature, **b** prior from test data

Table 7.10 Prognostics metrics for Case 2-A

	PH ($\alpha = 10\,\%$)	$\alpha-\lambda$ accuracy ($\alpha = 10\,\%$, $\lambda = 0.5$)	RA ($\lambda = 0.5$)	CRA
Prior from literature	11,000	False (0.3246)	0.8739	0.8116
Prior from test data	**13,000**	False (**0.4728**)	**0.9704**	**0.8643**

finding accurate values of parameters and loading than that of the prior with accelerate life test data, the prediction of RUL can be accurate when the correlation is well identified. If there are more model parameters and many correlations between parameters, however, it is possible that narrow priors may outperform uniform priors.

The best way to utilize accelerated life test data for Case 2 would be Case 2-B. Parameters are given as distributions, as shown in Fig. 7.17, and only the loading condition is updated with measured data. Figure 7.20 and Table 7.11 show that the results using prior information from accelerated life test data are significantly better than ones using the prior information from the literature, and show the best prediction among all cases (see Tables 7.10 and 7.11). Also, since the distributions of parameters are obtained from correct loadings (accelerated life tests), the true loading can be identified.

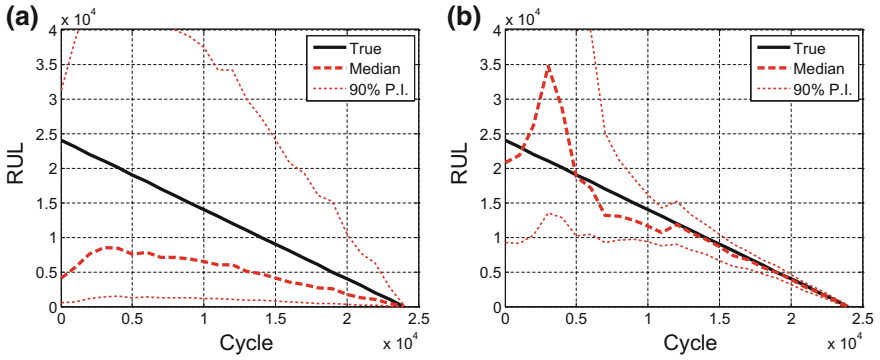


Fig. 7.20 RUL prediction of Case 2-B. **a** Prior from literature, **b** prior from accelerated life tests

Table 7.11 Prognostics metrics for Case 2-B

	PH ($\alpha = 10\,\%$)	$\alpha - \lambda$ accuracy ($\alpha = 10\,\%$, $\lambda = 0.5$)	RA ($\lambda = 0.5$)	CRA
Prior from literature	2000	False (0.0578)	0.4627	0.4450
Prior from test data	**11,000**	**True (0.5182)**	**0.9727**	**0.8751**

7.4.3.3 Case 3: When Only Loading Conditions Are Given, but not a Physical Model

As a comparison to the previous case, this corresponds to the case when the physical model is not available, but accurate loading information is. Since the physics model is not available, a data-driven approach would be a natural choice. This case can be handled with NN by adding loading conditions to the input variables. First, Fig. 7.21 shows damage prediction results without using the accelerated life test data; that is, training data are available only from field operating conditions. In this case, short-term predictions become valid after 13,000 cycles, as shown in Fig. 7.21a, but long-term predictions diverge from the true damage growth. This trend is similar to the data up to 19,000 cycles, as shown in Fig. 7.21b. Even if the PIs are narrowed than that of 13,000 cycles, the median of long-term prediction diverges from the true damage growth, and the 90 % PIs failed to include the true damage growth.

In contrast, when the accelerated life test data are used as training data, good results are obtained for short-term predictions at early cycles, as shown in Fig. 7.22a. Also, long-term predictions become close to the true damage growth with a narrow CI at 19,000 cycles, as shown in Fig. 7.22b. These results show that the accelerated life test data can be used as training data, even if the loading conditions in accelerated life tests are different from that of the nominal loading conditions.

7.4.3.4 Case 4: When Neither a Physical Model nor Loading Conditions Are Available

This case is also data-driven prognostics, like Case 3, but the loading condition is not available and cannot be updated since there is no physical model. In such a case, many training data under similar loading conditions to the field operating condition

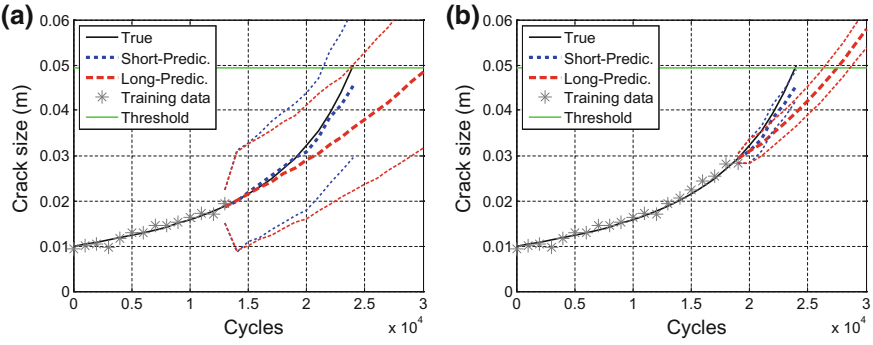


Fig. 7.21 Damage prediction without using the accelerated life test data. **a** At 13,000 cycles, **b** at 19,000 cycles

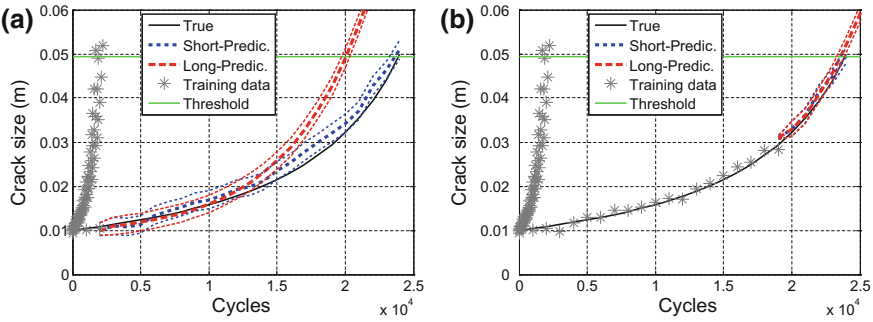


Fig. 7.22 Damage prediction using the accelerated life test data. **a** At 2000 cycles, **b** at 19,000 cycles

are required, but it would be expensive to obtain such data because of time and cost. In this section, the damage data at the accelerated life test conditions are mapped into the damage data at the field operating condition so that they can be used as training data.

The inverse power model is employed for mapping between the accelerated test data and field data. This model is widely used in problems such as an electric insulator, bearing, and metal fatigue. The inverse power model has a linear relation between the logarithm of life and load (Nelson 1990):

$$\text{Life} = \frac{\beta_1}{\text{Load}^{\beta_2}}, \quad \log(\text{Life}) = \log(\beta_1) - \beta_2 \log(\text{Load}), \tag{7.19}$$

where Life is the life span of a system and β_1, β_2 are coefficients, which need to be found using life data under at least two different loading conditions. Once the coefficients are identified, the life span under field operating conditions can be calculated from Eq. 7.19.

To apply the inverse power model for prognosis, the two coefficients in Eq. 7.19 should be determined. First, the life span is defined as when the damage size reaches a threshold. For example, when the damage threshold is 50 mm (dashed line in Fig. 7.23a), the life span of three loads, 145, 140 and 135 MPa, are 1734, 1947 and 2241 cycles, respectively. The three different markers in Fig. 7.23b show these three points. The slope of this line corresponds to the coefficient β_2, while the y-intercept at $\log(\text{Load}) = 0$ corresponds to $\log(\beta_1)$. When only two sets of data are available, the two parameters can be found by connecting a straight line in the logarithmic scale graph. When more than two sets of data are available, regression can be used to find these parameters.

Once these parameters are identified, the life span at the real operating load can be calculated using Eq. 7.19, whose examples at loadings from 50 to 90 MPa with an equal interval of 5 MPa are shown as star markers in Fig. 7.23c. This process can be repeated for different thresholds as shown by many dashed lines in Fig. 7.23a, b, and then the solid curves in Fig. 7.23c can be obtained by connecting

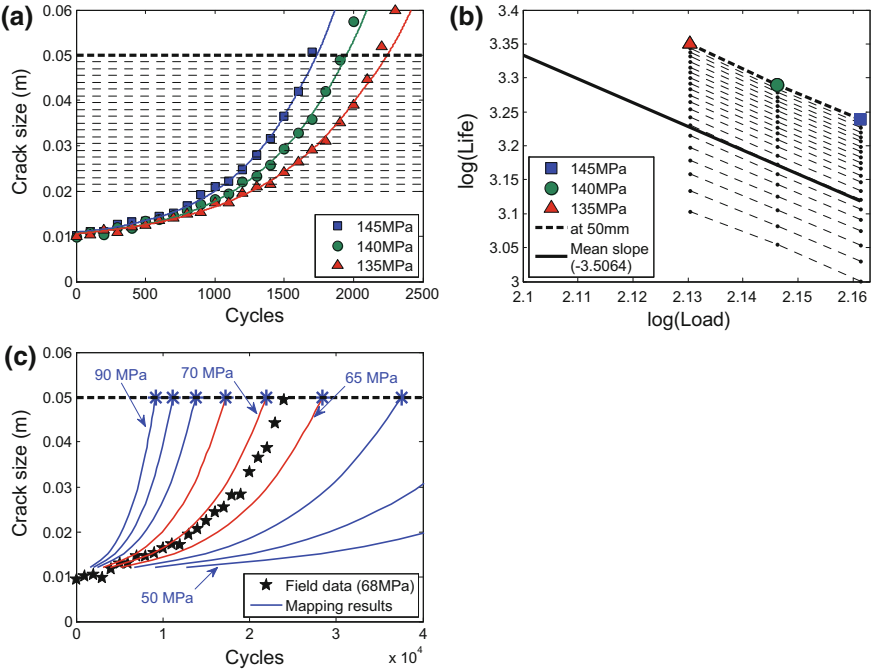


Fig. 7.23 Mapping process based on linear relation between logarithm of life and load. **a** Accelerated test data, **b** inverse power model, **c** mapping with estimated life

the lifespan results having the same loads. The uncertain loading range, 50–90 MPa, however, turned out to be too wide because Fig. 7.23c shows significantly different rates of damage growth over the range. Instead, if there are field damage data up to 10,000 cycles, it is easy to reduce the range of the real operating load between 65 and 75 MPa. In this study, therefore, three sets of mapping data are generated at three loads, 65, 70 and 75 MPa, and used for training data.

There are three uncertainty sources caused by three mapping steps, which correspond to Fig. 7.23a–c:

(1) Uncertainty in regression of accelerated life test data: Since accelerated test data (markers in Fig. 7.23a) are given as damage size at every 100th cycle, they are required to make regression models for calculating lifespan (cycle) at given threshold (damage size). Thirty regression models for each loading condition are constructed by randomly selecting initial weight parameters in NN, and the medians of the regression results at each loading condition are shown as three curves in Fig. 7.23a.

(2) Uncertainty in the inverse power model: Uncertainty in calculated life span data has an effect on identifying the coefficients of mapping. In fact, a small change in coefficients can yield a large difference in mapping results due to the logarithm relation. However, only the uncertainty in the y-intercept, $\log(\beta_1)$,

will be considered due to the following reasons: (a) as shown in Fig. 7.23b, the slopes at different thresholds are almost the same, and (b) the slope and the y-intercept have a strong linear relation. The mean slope of −3.5 is determined out of 630 slopes caused by 21 thresholds between 20 and 50 mm and 30 sets of regression models, which is illustrated as a solid line in Fig. 7.23b with an arbitrary y-intercept. The uncertainty in $\log(\beta_1)$ with a fixed slope is identified from 30 samples using the Bayesian approach.

(3) Uncertainty in inverse regression: The first regression model is to calculate the life span cycle for a given damage threshold. However, in order to estimate the uncertainty in damage size for a given cycle, it is necessary to make an inverse regression model. Therefore, 30 sets of second regression models are constructed to calculate damage size at given cycles in common with real load data.

Finally, mapping uncertainty from 27,000 possibilities ($30 \times 30 \times 30$) are shown in Fig. 7.24a. In the figure, the effect of uncertainty is represented by the height of markers, which is the 90 % CI. For reference, the true damage growths are shown using solid curves at three different load levels. The mapped data are close to the true ones, and the level of uncertainty is comparable to the level of noise in the field data. For final mapping data to be used for training, one set of damage growth can be randomly selected out of 27,000 sets, or other methods considering uncertainty in damage data can be developed. Nevertheless, note that mapping results are close to the true damage growth with narrow uncertainties, and training data having a level of noise similar to data under real operating loads are of help to predict damage growth at a real operating load. Even though the possible sources of uncertainty are discussed for academic purposes, it would be fine to consider only the median in the mapping process and add noise. Figure 7.24b shows the final mapping data to be used for training, which is obtained by adding noise to the median of mapping results. The noise is uniformly distributed between ±1 mm, whose level can be roughly determined based on field data up to current time. Figure 7.25 shows damage prediction results, where using mapped data is much better than not using mapped data in both short-term and long-term predictions.

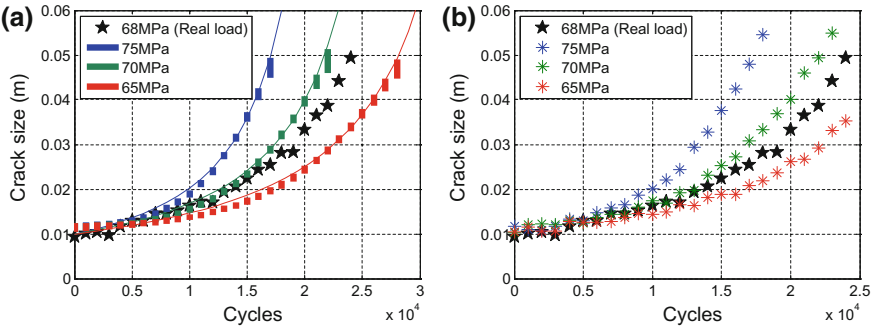


Fig. 7.24 Mapping between nominal and accelerated conditions data. **a** Mapping uncertainty, **b** mapping into field operating conditions

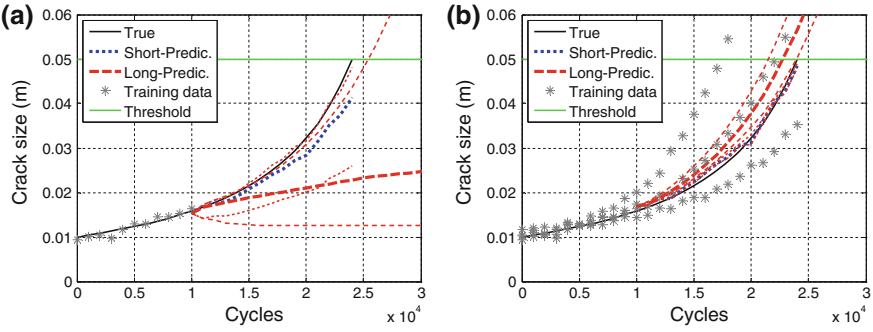


Fig. 7.25 Damage prediction results for Case 4. **a** Without using mapping data, **b** using mapping data

These results show that accelerated test data can show damage growth and RUL under field operation conditions through mapping and can be utilized as training data for nominal conditions through proper mapping methods.

7.4.4 Conclusions

This section presents four different methods of utilizing accelerated life test data for the purpose of prognostics in order to compensate for the limited number of in-service damage data. Four different cases are: (1) both the physical model and loading conditions are given, (2 and 3) either the physical model or loading conditions is given, and (4) neither the physical model nor loading conditions are given. Using accelerated life test data increases prediction accuracy in the early stages in all cases. Especially, proper mapping between nominal and accelerated conditions data can cover insufficient data for the last case, which is the most practical situation.

7.5 Bearing Prognostics Method Based on Entropy Decrease at Specific Frequencies[4]

7.5.1 Motivation and Background

Spall failure in bearing is one of the foremost causes of failure in rotating machineries, as 80–90 % of aircraft engine failures are caused by bearing failure. If

[4]An D, Choi JH, Kim NH (2014) Bearing prognostic method based on entropy decrease at specific frequency, Second European Conference of the Prognostics and Health Management Society 2014, July 8–10, 2014, Nantes, France.

bearings are not repaired properly, it can lead to a catastrophic failure. Since bearings cannot be disassembled once they are installed, it is not easy to know how much degradation is progressed and when the maintenance is required. System responses, such as vibration signals, oil debris and thermography, that are influenced by damage are measured to evaluate the level of damage indirectly. Especially, vibration signals have been used for damage evaluation for several decades (Haward 1994). These signals can be used to find the relationship between bearings' failure mechanism and signal strength at a specific frequency, from which the current damage level can be estimated. In the case of spall damage, however, since the damage grows rapidly, it is difficult to detect the damage early enough to prepare for maintenance.

Many efforts have been made to predict bearing's life. Yu and Harris (2001) proposed a stress-based fatigue life model for ball bearings, which outperformed the previous bearing life models (the Lundberg and Palmgmn model and the Inannidcs-Hnrris model), but the uncertainties during in-service were not considered. Bolander et al. (2009) developed a physical degradation model for the aircraft engine bearing, but this is limited to a spall on the outer raceway of a typical roller bearing. Since it is challenging to develop a physical degradation model, most researches on bearing prognostics rely on degradation feature extraction methods from vibration signal (Loutas et al. 2013; He and Bechhoefer 2008; Li et al. 2012; Kim et al. 2012; Boškoski et al. 2012; Sutrisno et al. 2012). Some of the results from previous studies show good RUL prediction results, but they are difficult to generalize and largely depend on the given vibration signals that can differ even from the nominally identical bearings under the same usage conditions.

Figure 7.26 from FEMTO bearing experimental data (Nectoux et al. 2012) shows the challenges in predicting bearing failure. The data shown as blue in the figure are vibration signals measured by accelerometer until failure occurs, and the red horizontal line represents a threshold (20 g). Note that the two quite different signals are obtained under the same usage conditions and from the nominally identical bearings. In general, it is difficult to find any consistency in the behavior of signals, life span and even a threshold. The life span of test 1 (Fig. 7.26a) is around 2800 cycles, but it is 870 cycles for test 2 (Fig. 7.26b) even before the acceleration signal reaches the threshold.

In this section, a new method is introduced to extract degradation feature of bearings from vibration signals, which is based on entropy change in the frequency

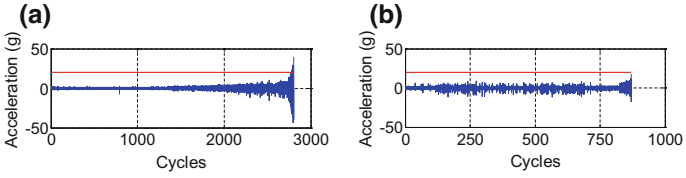


Fig. 7.26 Vibration signal from the nominally identical baring and the same usage conditions. **a** Test 1, **b** test 2

domain. Entropy of a system changes when system's characteristics change, such as mechanical and/or chemical energies. However, it is difficult to find these changes from raw data that is a result of combination of system dynamics, damage, and noise signals. From the observation that the nature of vibration signals changes as the mechanical condition changes, the signals are decomposed into various frequencies, and those frequencies that show a consistently change with respect to damage are selected and analyzed. More detailed explanations and procedures are presented in Sect. 7.5.2.

FEMTO bearing experimental data (Nectoux et al. 2012) are employed to demonstrate the entropy-based method, whose experimental platform and bearing information are shown in Fig. 7.27 (more detailed explanation of test apparatus can be found in IEEE PHM 2012 Prognostic challenge 2012). Vibration signals are monitored during 0.1 s with a sampling rate of 25.6 kHz every 10 s. Therefore, the 0.1 s in every 10 s is considered as one cycle, and there are 2560 samples in each cycle. This setting is necessary because otherwise the amount of data would be huge.

The information of usage conditions and the number of experimental data are listed in Table 7.12. Two different usage conditions in terms of the radial force applied on the tested bearing and rotating speed are used in this study. The usage conditions of Conditions 1 and 2 are respectively, 4 kN and 1800 RPM and 4.2 kN and 1650 RPM. Seven sets of experimental results from each condition are obtained until failure occurs. Three sets of data from each condition are used as training data to predict RUL of other bearings. Based on the raw vibration signals, it is considered that failure is occurred when the magnitude of acceleration reaches 20 g.

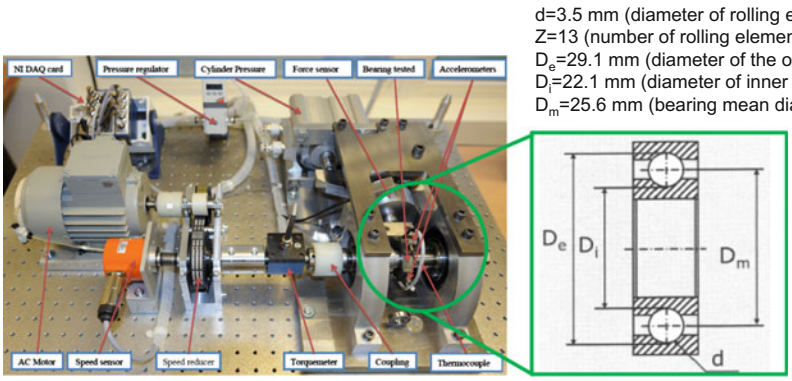


Fig. 7.27 Experimentation platform: PRONOSTIA (Nectoux et al. 2012)

Table 7.12 Experimental condition and data usage

	Condition 1	Condition 2
Radial force (N)	4000	4200
Rotating speed (RPM)	1800	1650
Num. of data set	7 (Set 1–Set 7)	7 (Set 1–Set 7)

With the experimental information, this section is organized as follows: in Sect. 7.5.2, the method for degradation feature extraction and its attributes are explained; in Sect. 7.5.3, prognostics is performed based on the results from Sect. 7.5.2; in Sect. 7.5.4, a generality of the proposed method is discussed; and conclusions and future works with limitation of the proposed method are presented in the final section.

7.5.2 *Degradation Feature Extraction*

It is difficult to define vibration signals from raw data as a degradation feature since there is no change in signals even just before failure in many cases. In this study, a new method is proposed to extract degradation feature from the raw data based on entropy changes in the frequency domain.

7.5.2.1 Information Entropy for Degradation Feature Extraction

Entropy is a measure of disorder and randomness of the system. In physical interpretation, entropy change is explained using energy flow. When a system absorbs energy from others, its entropy increases and the counterpart decreases. The total entropy of an isolated system, however, never decreases. There is also a mathematical concept of entropy called information entropy or Shannon entropy, which is the average amount of information (Shannon 1948). In this concept, the increase in entropy means the increase in uncertainty by missing information contained in data. Many researchers argue that physical and information entropy are related to each other (Brillouin 1956; Bao et al. 2010), but an opposite argument also exists (Frigg and Werndl 2010). For the bearing problem, the total energy in bearing increases as the work done by external force is consumed by damage initiation/propagation (Wang and Nakamura 2004), which makes entropy increases (Bao et al. 2010). However, the information or data used in this research are decomposed vibration signals rather than the total thermodynamic energy during degradation process of bearings, which means that the information entropy fits better. Since it is a debatable topic whether there is a relation between physical and information entropy, information entropy is only considered here as a tool to express changes in vibration signals rather than to connect physical interpretation with information entropy.

Information Entropy

In information theory, entropy is calculated based on the following equation (Shannon 1948):

$$H(X) = -\sum_{i=1}^{n} f(x_i) \log_2 f(x_i), \tag{7.20}$$

where X is an information source, n is the number of possible outcomes from X, and $f(x_i)$ is the probability of each outcome. In this case, X in bearing problem represents bins of acceleration data. After normalizing acceleration data between minimum and maximum values, the normalized range between 0 and 1 is divided by 255 intervals, which are referred to as bins as shown in Fig. 7.28. There are total 256 bins initially and the magnitude of data that can be located in each bin is also listed in the figure. Data whose magnitude is less than 1/500 and in between 1/500 and 3/500 are, respectively, included in the first and the second bin, and the same for the last bins (the reason for the criteria of number of initial bins and data magnitude is that Eq. 7.20 is developed based on computer usage). After data allocation, the number of bins having nonempty data becomes the number of possible outcomes, n; that is, $256 - n$ bins are empty. The probability, $f(x_i)$ is the number of data in ith bin divided by total number of data; this is basically the same concept as the probability mass function in statistics.

Equation 7.20 denotes that entropy increases as the number of bin, n, increases and the probability of each bin is even, which corresponds to the case that the data are spread from zero to one as illustrated in Fig. 7.29a. The entropy at $k+1$ cycle is higher than that of k cycle, since the amplitude data are allocated in more bins with even probability. Note that entropy is calculated using all data from cycle zero to the current cycle. That is, entropy calculation uses accumulated data. Figure 7.29b shows where opposite case. The number of bins at k and $k+1$ cycles are the same, but the data at $k+1$ cycle are more located in the middle bins, which makes entropy decrease.

Entropy as a Degradation Feature

Figure 7.30 shows normalized raw data for Sets 1 and 2 at Condition 1 and the change in their entropy in the time domain using accumulated data up to the current

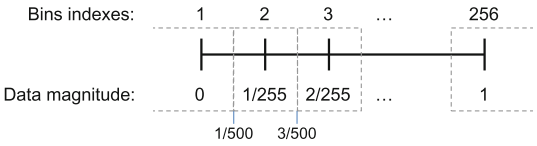


Fig. 7.28 Illustration of information source of bearing problem

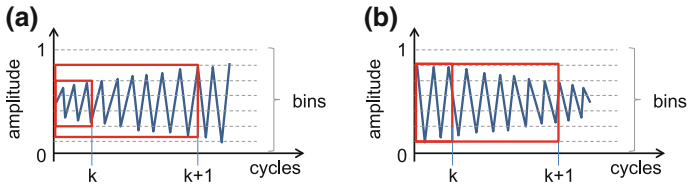


Fig. 7.29 Illustration of two cases of entropy change. **a** Entropy increase, **b** entropy decrease

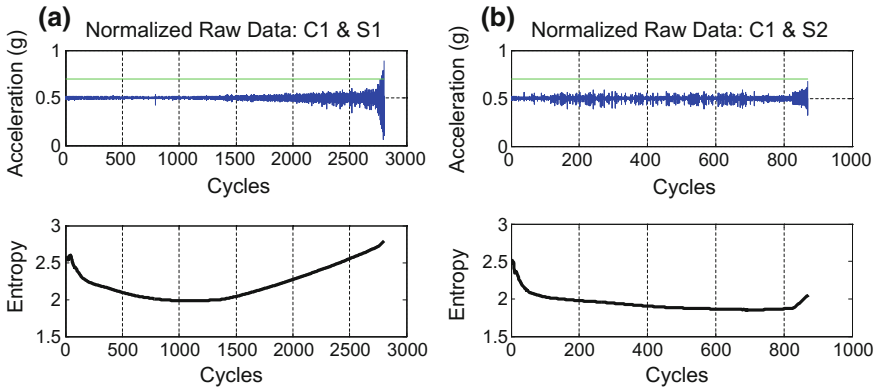


Fig. 7.30 Entropy calculation with raw data. **a** Condition 1 and Set 1, **b** Condition 1 and Set 2

cycle. In Fig. 7.30a, the entropy increases as the vibration energy increases (the magnitude of amplitude increases). Since the entropy change is related to vibration energy, which is related to damage degradation, it seems that the entropy can be used as a degradation feature. However, note that the entropy starts changing as vibration signals change, which means entropy change cannot be observed unless there is a change in vibration signals. However, many vibration signals from bearing test do not show a significant change until just before failure under large noise as shown in Fig. 7.30b. Therefore, it seems difficult to use the entropy change in time domain as a damage feature.

Instead of entropy change in time domain, 2560 raw data at each cycle (0.1 s) of a bearing are transformed to the frequency domain, and the entropy at each frequency can be plotted with respect to cycles. If there are Nf frequencies in fast Fourier transformation (FFT), there are Nf curves of entropy versus cycle. The idea is that among these entropy curves at different frequencies, those frequencies that show consistent change in entropy for all bearing test sets are selected as a damage feature. When the vibration signals are decomposed into various frequencies, the amplitude at some frequencies increases and that of other frequencies decreases as cycle increases. Figure 7.31a, b show, respectively, an example of increasing and decreasing amplitude and entropy at a frequency of Condition 1 and Set 1. The amplitude in the frequency domain more clearly increases than that in the time domain, which is because the signals in time domain are the sum of amplitudes of all frequencies including both increasing and decreasing amplitudes. This is more clearly exhibited in the case of Condition 1 and Set 2 in Fig. 7.30b with Fig. 7.31c, d.

It is expected that the amplitude increases as the level of damage increases. This is true in the time domain even if it is difficult to observe it at early stages. In frequency domain, this may true only for localized defects. In practice, system vibration characteristics continue to change as degradation progresses. The natural frequency changes as the stiffness decreases due to damage growth, and the periodic

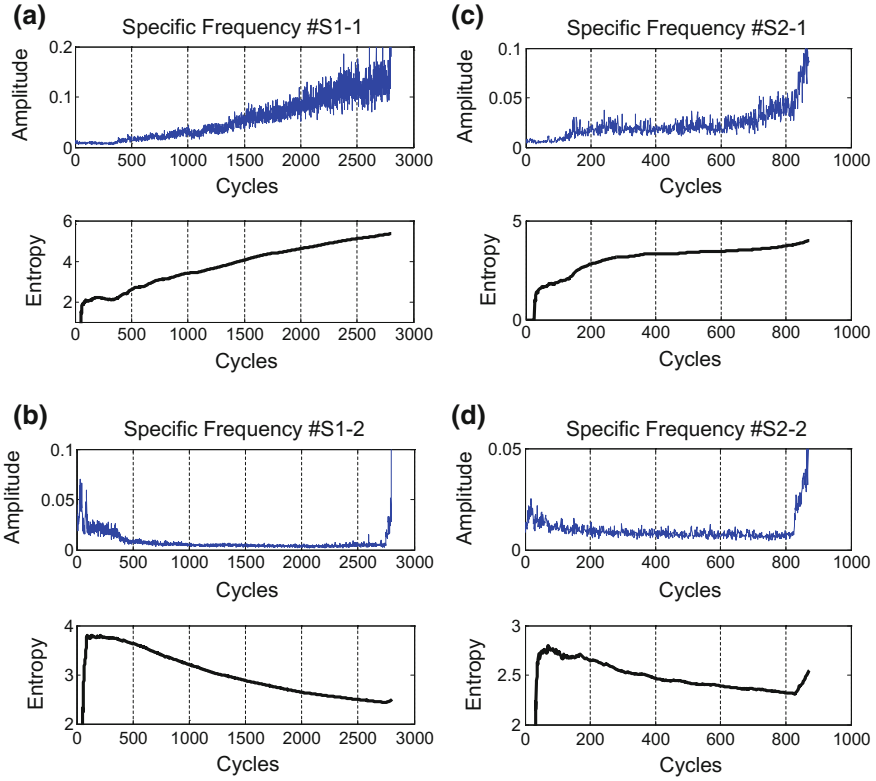


Fig. 7.31 Entropy calculation with amplitude at specific frequency. **a** Cond. 1 and Set 1: increasing entropy, **b** Cond. 1 and Set 1: decreasing entropy, **c** Cond. 1 and Set 2: increasing entropy, **d** Cond. 1 and Set 2: decreasing entropy

nature of localized defects can diminish since the motion of rolling elements becomes irregular and disturbed as damage progresses (Haward 1994).

The following illustrative example can be used to explain different behaviors of amplitude change in frequency domain due to damage growth. Let us assume that as damage grows, the natural frequency of system gradually changes. Figure 7.32 shows the change in vibration characteristics in terms of natural frequency from 2.4 to 2.2 Hz. If the time domain data in Fig. 7.32a are continuous, then the frequency amplitude in Fig. 7.32b will be a Dirac delta function with amplitude of 5 at 2.4, 2.3 and 2.2 Hz. Since the resolution of this example is 0.5 Hz and the frequencies are not multiples of 0.5, noisy responses exist in Fig. 7.32b, which can also be common for real applications. From Fig. 7.32b, it is shown that the amplitude at frequency 2.0 increases (like Fig. 7.31a, c), while the one at frequency 2.5 decreases (like Fig. 7.31b, d) as the natural frequency changes from 2.4 to 2.2 Hz.

Based on entropy plots in Fig. 7.31 and the discussion in Fig. 7.32, it seems that amplitude increases at some frequencies, while it decreases at other frequencies. In

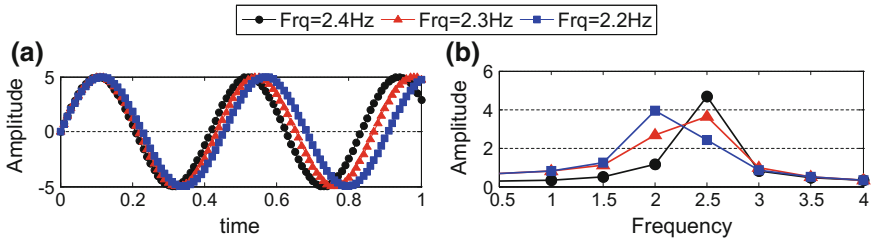


Fig. 7.32 Amplitude change at specific frequency as vibration characteristics changes. **a** Time domain, **b** frequency domain

fact, both can be used as a degradation feature. After examining 14 FEMTO data sets, it is found that the decreasing trends in entropy are consistent for all data sets, and some important attributes can be found. On the other hand, no common characteristics can be found from increasing trends in entropy. Even though the entropy clearly increases, its behavior is unpredictable. Therefore, decreasing amplitudes are more stable and consistent than increasing ones. In addition, it is found that the frequencies showing decreasing entropy are around 4 kHz for all data sets, which may depends on the structure of the systems, such as the bearing types and number of rolling elements. Therefore, the decreasing entropy in the frequency domain is used as a degradation feature. The detailed procedure to extract degradation feature based on entropy change in the frequency domain is explained in the following section.

7.5.2.2 Procedure of Degradation Feature Extraction

The procedure of the proposed degradation feature extraction method is illustrated in Fig. 7.33, whose detailed explanation is as follows:

Step 1: Convert signals in time domain into frequency domain using FFT. As mentioned before, one cycle includes 2560 samples of vibration signals (data acquisition during 0.1 s with sampling rate 25.6 kHz), which is converted into frequency domain using FFT (Walker 1996). There are 1401 different FFT results from different cycles (0–1400 cycles), such as the example in Fig. 7.33.

Step 2: Reshape FFT results frequency-wise (frequency-wise plot). At a fixed frequency (e.g., Frq: 1 in the figure), plot acceleration amplitude versus cycle graph. Amplitudes at different cycles are collected at a fixed frequency, which is called frequency-wise plot here. Since FFT results are symmetric with 2560 different frequencies between 0 Hz and 25.6 kHz, there are 1280 frequencies to be considered as candidates for degradation feature; that is, there are 1280 amplitude cycle plots.

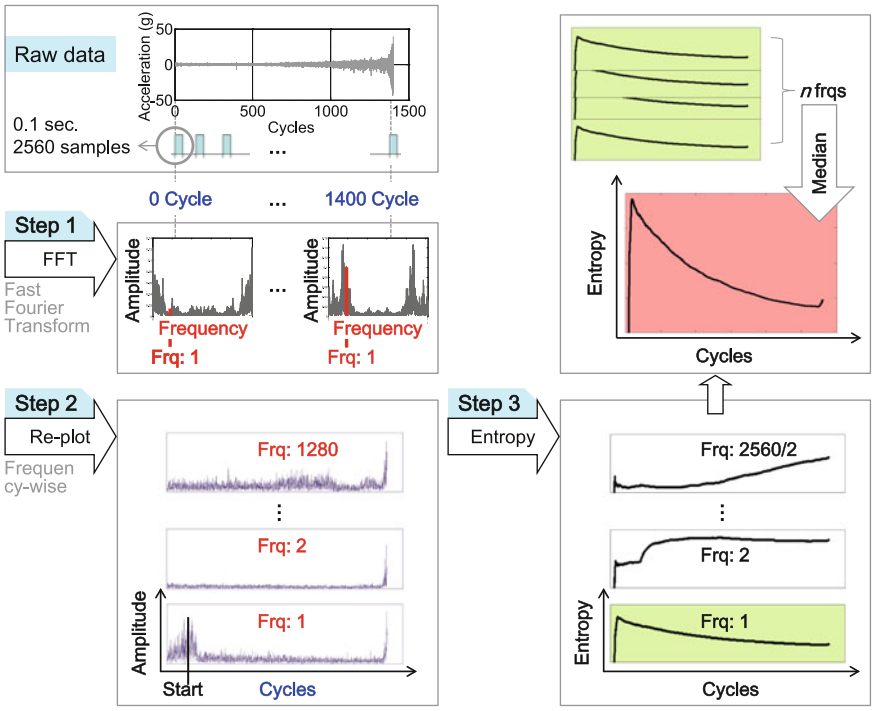


Fig. 7.33 Illustration of degradation feature extraction

Step 3: Calculate entropy and select frequencies showing decreasing entropy. The frequency-wise amplitude versus cycle graphs from Step 2 are used to calculate entropy using Eq. 7.20, which ends up 1280 graphs for entropy versus cycle. Different traces of entropy are illustrated in Step 3 of Fig. 7.33. Among them, those frequencies showing decreasing entropy, such as Frq: 1, are selected. If multiple frequencies show decreasing entropy, then a median of these entropies at each cycle is taken as a damage feature.

There are two points to be considered in calculating entropy. First, a starting cycle for entropy calculation needs to be determined based on amplitude at frequency domain, which is to avoid the initial effect. For example, a point where amplitude or trend changes significantly at an early cycle is selected as a starting cycle since this can be related to a large vibration due to initial misalignment of the system, as shown in Frq: 1 at Step 2 in Fig. 7.33. The starting cycle can differ for different sets, but it is usually located between 12 and 100 cycles. Second, the method to select frequencies is based on a slope of linear regression using entropy data. If too few frequencies whose slope is large are selected, then the degradation feature becomes clear, but at the same time common attributes from different test sets cannot be clearly found since the number of frequencies is too small to represent the general characteristics. In this study, frequencies whose entropy slope is

in top 25 are selected. The starting cycle and selected frequencies have an effect on calculation results of a particular data set, but overall attributes (will be discussed later in this section) of bearing problem from the proposed methods do not change.

7.5.2.3 Results of Feature Extraction and Its Attributes

Degradation features extracted based on the procedure in the previous section are shown in Fig. 7.34, in which entropy of all 14 sets decreases as cycle increases. Each curve is obtained by taking a median of 25 entropy values at each cycle that are calculated from selected frequencies, showing a consistent decrease in entropy. From the entropy curves, the maximum and minimum entropy and EOL is defined as shown in Fig. 7.35.

There are two main findings when the proposed method is used. First, EOL is proportional to the maximum entropy, which is shown in Fig. 7.36a. It is possible that a higher energy at initial stage may be related to a longer life. The RUL can be predicted by utilizing the linear relationship between the maximum entropy and

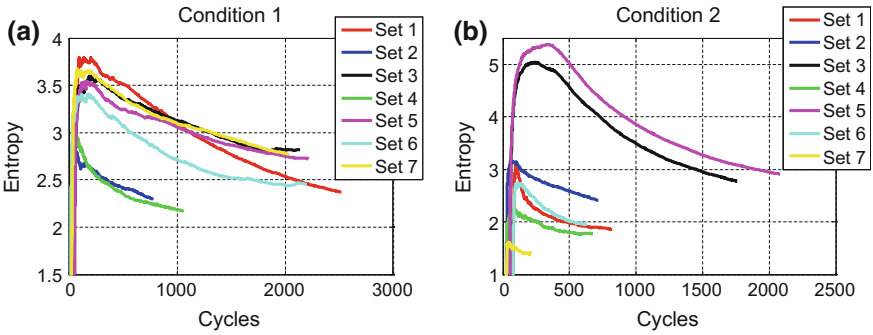


Fig. 7.34 Results of the degradation feature. **a** Condition 1, **b** condition 2

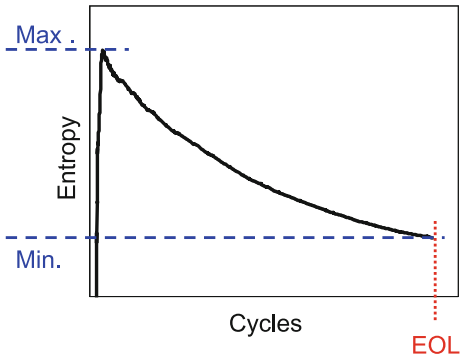


Fig. 7.35 Definition of maximum and minimum entropy and EOL

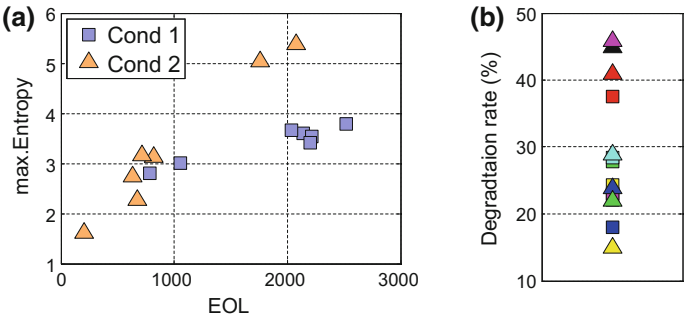


Fig. 7.36 Two important attributes from the extracted feature. **a** Relation between maximum entropy and EOL, **b** two groups of degradation rate

EOL based on training data. Second, when the degradation rate is defined using the maximum and minimum entropy, as

$$\mathrm{dr} = 1 - \frac{\text{min. Entropy}}{\text{max. Entropy}} \tag{7.21}$$

It turns out that the degradation rate can be classified into two groups, as shown in Fig. 7.36b. It is possible that the two groups are related to two different failure mechanisms. The two groups of the degradation rate are distributed around 20 and 40 %, which can be used for a threshold. In the following section, RUL is predicted based on the two approaches using the linear relation and the degradation rate with the entropy trend.

7.5.3 *Prognostics*

Even though the vibration signals until failure (true EOL) are available, it is assumed that the threshold is 90 % of the true EOL for the purpose of maintenance scheduling. The procedure of feature extraction in Fig. 7.33 is repeated every 50 cycles to select featured frequencies. Even though the selected frequencies can be different at each cycle, they gradually become stable and converge to around 4 kHz as cycles increase, as shown in Fig. 7.37. Therefore, RUL prediction is performed after the selected frequencies are converged; that is, after the red vertical line in the figure. RUL is predicted in two methods: (1) using the linear relationship between the maximum entropy and EOL and (2) using the entropy trend with a threshold.

As mentioned before, three sets of data (Sets 1, 2, and 3) from each condition are used as training data. These sets are utilized to construct a linear relationship between the maximum entropy and EOL, which is shown in Fig. 7.38a. In the figure, three different colors represent different sets (red: Set 1, blue: Set 2, and

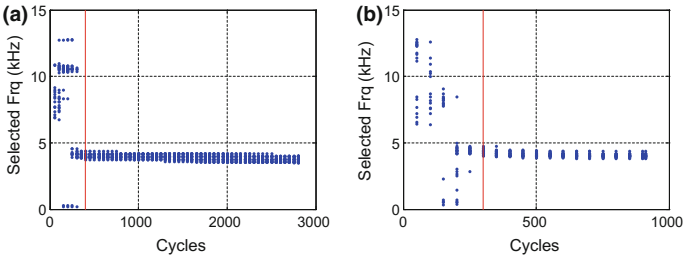


Fig. 7.37 Selected frequencies at every 50 cycles. **a** Condition 1 and Set 1, **b** Condition 2 and Set 1

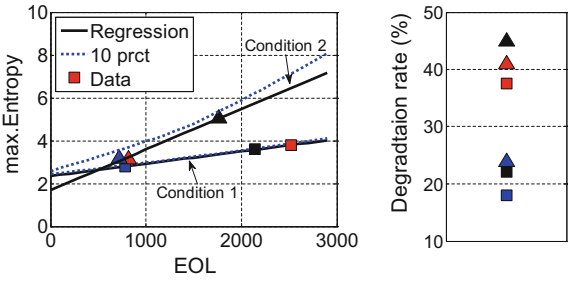


Fig. 7.38 Information from three sets of training data. **a** Linear relationship between the maximum entropy and EOL, **b** degradation rate (threshold)

black: Set 3), and square and triangle markers, respectively, represent Conditions 1 and 2. The black solid and blue dashed lines are, respectively, mean of regression results and 10th percentile lower confidence bound obtained by using three data from each condition. The 10th percentile lower confidence bound is used for a conservative prediction. For example, when the maximum entropy is obtained as 4 at the current 700 cycles, the EOL and the RUL are predicted as 1000 and 300 cycles, respectively.

7.5.3.1 E.trend Method: Entropy Trend with Threshold

In this method, the future behavior of entropy is predicted based on the following model:

$$\text{Entropy} = \beta_1 \exp\left[\beta_2 (\text{Cycle})^{\beta_3}\right], \tag{7.22}$$

whose expression can follow the entropy trend in Fig. 7.34. Three unknown parameters, $\beta_1, \beta_2, \beta_3$ in the equation are identified based on nonlinear regression using the data from the cycle at the maximum entropy to the current cycle. The RUL is predicted by extrapolating the model with identified parameters until it reaches the threshold. The threshold is determined using the degradation rate from

three sets of data (Sets 1, 2, and 3) from each condition, as shown in Fig. 7.38b. Degradation rates of total six training data are classified into two groups, and the mean of each group is considered as a threshold; they are 21 and 41 %.

It can be determined whether the current data trend belongs to either 21 or 41 % threshold based on the linear regression model, as shown in Fig. 7.38a. As an illustration, the prediction result of future entropy trend of Condition 2 and Set 3 is shown in Fig. 7.39. In Fig. 7.39a, black solid curve, black circles and green dotted horizontal line are, respectively, true trend up to EOL, used data to identify $\beta_1, \beta_2, \beta_3$ in Eq. 7.22 and the threshold calculated from 21 % degradation rate. The red dashed curve from the current cycle (700 cycles) is a predicted entropy trend based on Eq. 7.22 with identified parameters ($\beta_1 = 6.15 \times 10^6$, $\beta_2 = -12.14$, $\beta_3 = 0.0244$). The intersection of the predicted entropy trend and the threshold is a predicted EOL, which is shown as the red dashed vertical line. This result shows RUL is −9 cycles using 21 % threshold (i.e., a total of 691 life cycles), which does not make sense because the degradation rate at the current 700 cycle is already more than 21 %.

The outcome can be different if the threshold is estimated using the linear regression in Fig. 7.38a. The maximum entropy is 5.08 from Fig. 7.39, which corresponds to 1780 cycles EOL from the mean of regression model (max. Entropy $= 1.70 + 0.0019 \cdot$ EOL, since this is just for threshold classification, the solid line is used) in Fig. 7.38a. Compared to this result, 691 cycles is too short as the EOL. The new threshold is estimated as 43 % by considering minimum entropy at 1780 cycles, which is obtained by extrapolating entropy trend (red dashed curve in Fig. 7.39a) to 1780 cycles. Since the newly estimated threshold is close to the threshold in the other group (41 %) from the training data, the RUL is re-predicted using 41 % threshold, which is shown in Fig. 7.39b. With 41 % threshold, the RUL is predicted as 833 cycles whose error with the true RUL (1059 cycles) is 0.2137, which is calculated by dividing the true RUL minus the predicted one by the true one.

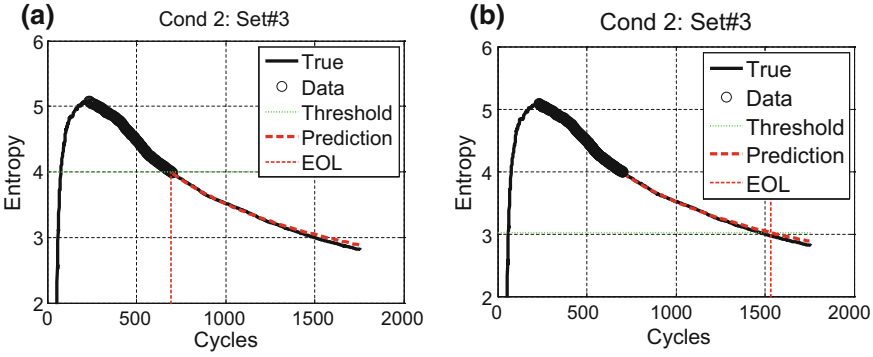


Fig. 7.39 Results of degradation prediction. **a** Threshold 21 %, **b** threshold 41 %

7.5.3.2 RUL Prediction Results

The results of RUL prediction for all prediction sets are shown in Fig. 7.40, in which black solid lines are the true RUL, green vertical line indicate the cycles when the frequencies are converged, and blue dotted lines and red dashed lines are prediction results based on Max.E-EOL and E.trend methods, respectively. The results from the Max.E-EOL method of Condition 1 are closer to the true RUL than those from the E.trend method, while the Max.E-EOL method does not perform well for Condition 2 since EOL of Condition 2 is short and maximum entropy is small, but 10-‰ at small entropy gives very conservative prediction (see in Fig. 7.38a).

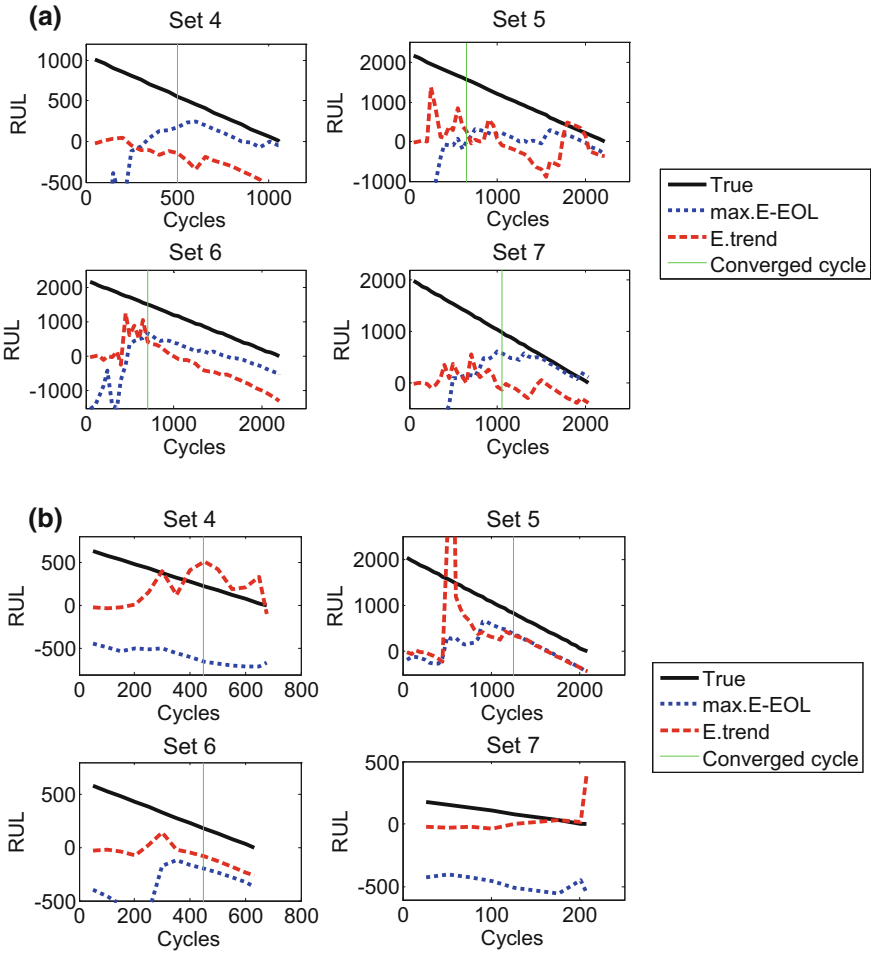


Fig. 7.40 RUL prediction using three training data. **a** Condition 1, **b** Condition 2

It is considered that the RUL results can be reliable after selected frequencies are converged (the green vertical lines). However, the RUL prediction of some cases change from negative value to positive. For example, E.trend result of Set 5 in Fig. 7.40a is negative at 1500 cycles, but it becomes positive at 1800 cycles. Since RUL at future cycles is unknown, it is assumed that when RUL is less than 50 cycles, maintenance is ordered. In this case, the used life is calculated at the cycle when maintenance is ordered. The ratio of used life to the EOL is listed in Table 7.13; the higher the ratio is, the better the prediction is. However, it is failed to predict the RUL of Condition 2 and Set 7, in which failure occurs before the selected frequencies are converged. By considering the EOL of this set is very short, it seems that there were significant initial defect in this bearing. Except for this bearing, the mean of the ratio from seven results is calculated. The mean of the ratio of used life is 0.56 by considering conservative results between the Max. E-EOL and the E.trend methods (0.47, 0.29, 0.45, 0.52 from Condition 1 and 0.67, 0.78, 0.71 from Condition 2). On the other hand, when the optimistic results are selected by ignoring conservative results and going to next prediction steps, the mean is 0.78 (0.81, 0.32, 0.72, 1.08 from Condition 1 and 1.04, 0.79, 0.71 from Condition 2). That is, bearings can be used 56 or 78 % of their whole life in average. The choice between the Max.E-EOL and the E.trend methods can be made depending on the trade-offs between maintenance cost and risk.

So far, the first three data sets are used for training. Lastly, the prediction results from different combination of three training data sets are provided to validate the proposed method, which is listed in Table 7.14. There are total ten cases among total 35 possible combinations (selecting 3 out of 7). In each case, three training sets are randomly selected, but the combinations of which three data do not show proportional relation between maximum entropy and EOL are excluded. From the three sets, threshold levels and the linear relation between maximum entropy and EOL (see Fig. 7.38) are determined for each condition and utilized to predict RUL of remaining four bearings in each condition, which is repeated for all cases.

The mean of conservative and optimistic results from each case are listed in Table 7.14, which is calculated in the same manner as in Table 7.13. The statistical results (3 ‰, minimum and maximum values) from the ten cases are also

Table 7.13 Ratio of used life to EOL using three training data

		Set 4	Set 5	Set 6	Set 7
Condition 1	Max. E-EOL	0.81	0.29	0.72	1.08
	E.trend	0.47	0.32	0.45	0.52
Condition 2	Max. E-EOL	0.67	0.79	0.71	Fail
	E.trend	1.04	0.78	0.71	Fail
Conservative mean		0.56			
Optimistic mean		0.78			

Table 7.14 Ratio of used life to true EOL using different combination of three training data

	Case 1	Case 2	Case 3	Case 4	Case 5
Training set #	[1 2 3]	[3 4 5]	[1 3 4]	[2 5 7]	[1 3 7]
Conservative mean	0.56 (7)	0.67 (7)	0.60 (7)	0.58 (8)	0.54 (8)
Optimistic mean	0.78 (7)	0.71 (5)	0.72 (7)	0.71 (7)	0.68 (8)
Fail to be predicted with optimistic result		Two sets		One set	
	Case 6	Case 7	Case 8	Case 9	Case 10*
Training set #	[1 2 5]	[2 3 5]	[1 4 5]	[2 6 7]	[4 5 6]
Conservative mean	0.63 (7)	0.56 (7)	0.70 (7)	0.45 (8)	0.64 (6)
Optimistic mean	0.87 (7)	0.84 (6)	0.75 (6)	0.55 (6)	0.80 (3)
Fail to be predicted with optimistic result		One set	One set	Two set	Three sets
	Min	25th prc	Median	75th prc	Max
Conservative mean	0.45	0.56	0.59	0.64	0.70
Optimistic mean	0.55	0.71	0.74	0.80	0.87

*One set fails to be predicted with the proposed method

calculated. Based on these results, 56–64 and 71–80 % of bearings' life from conservative way and optimistic way, respectively, can be used in average. However, note that there are several sets that are failed to be predicted by relying on the optimistic results, and the mean results in Table 7.14 are obtained by excluding those ones (the numbers in the parenthesis denote the total number of bearings to calculate the mean values). It is required to be very careful when relying on the optimistic results with high safety systems. Also, the proposed method cannot predict the RUL for one set in Case 10.

7.5.4 Discussions on Generality of the Proposed Method

The main findings from the proposed method are that the entropy gradually decreases to a certain level of thresholds, and the EOL is proportional to the maximum entropy. If these attributes are also found in the other applications, the proposed method might be widely applicable. In addition, if the effect of usage conditions on these attributes is found, it is expected that more accurate prediction is possible with less number of training data. In this section, these three issues are discussed.

7.5.4.1 Another Bearing Application

Additional experimental results from another bearing application are employed to validate the attributes of the proposed method, which are provided by the NSF I/UCR Center for Intelligent Maintenance Systems (IMS) (Lee et al. 2007). Four

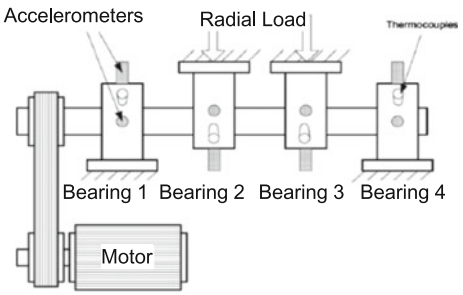


Fig. 7.41 Illustration of IMS bearing (Qiu et al. 2006)

double row bearings (16 rollers) are installed on a shaft, and the rotating speed and radial load are, respectively, 2000 RPM and 6000 lbs (see Fig. 7.41). The test stops when the accumulated debris exceeds a certain level, and there are three sets of experimental data available. A run-and-stop operation is repeated for experimental Set 1 and Set 3 (Set 2 is continuously monitored to EOL).

Threshold, EOL, and maximum entropy of the failed bearings in each set are listed in Table 7.15. The maximum entropy of Set 3 is too small compared to its EOL. However, it is difficult to conclude that the EOL is not proportional to the maximum entropy since three sets of data are very small and there were several run-and-stop during the operation. At least, failure occurs at bearings having the lowest maximum entropy among four bearings in each set. Also, the thresholds of Set 2, Set 3 and bearing 4 of Set 1 are around 70 %. The difference between bearing 3 (48 %) of Set 1 and all other bearings (70 %) is similar to the difference in FEMTO bearing (20 and 40 %). Since the number of data set is too small, the run-and-stop of the operation has an effect on vibration signal and entropy calculation, and four bearings on a shaft can interact with each other in the failure process, it is difficult to make any definite conclusions.

7.5.4.2 The Relation Between Threshold/Max.E-EOL and Usage Conditions

In the previous section, the threshold of the IMS bearing is around 50–70 %, which differs from that of the FEMTO bearing, 20–40 %. The cycle of the FEMTO

Table 7.15 Failure summary of three data sets

Data set	Minimum Max.E	Failed bearing	Failed element	Threshold (%)	EOL	Max. E
Set 1	Bearing 4	Bearing 3 Bearing 4	Inner race Roller element	48 65	1940	1.37 1.00
Set 2	Bearing 1 Bearing 4	Bearing 1	Outer race	73	886	0.81
Set 3	Bearing 3	Bearing 3	Outer race	71	4003	0.68

bearing is based on seconds, and the EOL is around 2–7 h, while the EOL of IMS bearing is around 7–30 days. That means, FEMTO bearings are under the accelerated test conditions, while IMS bearings are under the nominal operation conditions. Therefore, it seems that there is a relationship between the degradation rate (threshold) and the applied load, which cannot be found in the research due to the lack of test data. However, if this relationship can be established, then it can help to determine a threshold with a small number of training data.

It seems like the slope is proportional to the load based on Conditions 1 and 2 in Fig. 7.36a. If a relation between the slope and usage conditions can be found, it can help decision-making to utilize the relation between maximum entropy and EOL for prediction. However, two conditions are not enough to validate the relation. This will be considered in near future with more data sets under various usage conditions.

7.5.5 Conclusions and Future Works

As a summary, the proposed method is based on entropy changes at fixed frequencies to extract degradation feature from vibration signals and to predict the RUL of bearing applications. The main contributions and attributes of the proposed method are as follows:

- Degradation feature is found from vibration signals, which gradually decreases as cycle increases.
- Degradation rate of different experimental sets from the same application are similar each other, which can be used as a threshold.
- EOL is proportional to the maximum entropy, which can be used another prediction method without a threshold.

The proposed method is demonstrated using 14 sets of bearing experimental data under two different conditions. By considering used life, 59–74 % of bearings' life can be used based on the proposed method.

The results from the proposed method are noticeable by considering the current level of prediction capability in the literature, but there are still several limitations to be solved. First of all, the entropy trend decreases exponentially, which can make a large difference in life prediction with a small perturbation of threshold. Second, the proposed method is based on the accumulated vibration data. Since the bearing systems under real operating conditions may last very long time, tremendous amount of data should be stored. Lastly, the proposed method has been developed based on the vibration signal. Physical explanations of the observed attributes are not available yet.

As future works, the proposed method may be improved by resolving the aforementioned limitations. Also, a generality of the method will be demonstrated by studying the effect of usage conditions on the level of threshold and the slope of

Max.E-EOL. Even though the proposed method has not been demonstrated using the other bearing applications yet, the general usage of this method is promising by judging the results from 14 sets of bearing data, and the results in Sect. 7.5.4 show the possibility.

7.6 Other Applications

There are several sets of real test data available in the NASA website (http://ti.arc.nasa.gov/tech/dash/pcoe/prognostic-data-repository/). For example, wear data from a milling machine are available (Agogino and Goebel 2007), in which both directly measured wear depth and indirectly measured signals from three different types of sensors (acoustic emission, vibration and current sensors) are provided. These data can be a good source to extract degradation feature or health index from sensor signals. Also, the data of two bearing applications in Sect. 7.5 are provided in the website. If the sensor signals show some changes with respect to cycles, they can be used as degradation data. Most of the data given in the website require feature extraction processes as with the bearing problems in Sect. 7.5 since they are measured by sensors as practical cases.

In practical applications, heath monitoring data may not be a deterministic value but can be distributed. There are cases when health monitoring data for prognostics can be distributed. For example, many sets of damage data are given at the same usage conditions from the same system, and usage conditions such as loading conditions can also have uncertainties and need to be considered as distributions. In this section, damage feature extraction and prognostics will be discussed with distribution-type data.

When distributed data are given at a given time cycle, the best way is to parameterize the distribution and to use the distribution parameters, which are deterministic, for the purpose of prognostics. Since the type of distribution may change in the course of damage degradation, it is important to use a flexible distribution type so that it can model different distribution shapes. Among many standard distribution types, Johnson distribution has four parameters and very flexible in terms of representing different distribution shapes, and thus, it will be used here.

The distributed data can be replaced with four parameters based on the Johnson distribution (Johnson 1949), in which 4 ‰ corresponding to probabilities 0.0668, 0.3085, 0.6915 and 0.9332 (let us call these four quantiles) are employed to represent the distribution. Figure 7.42 shows examples of Johnson distribution for normal and beta distribution. The black solid curves are exact probability density function (PDF) from each distribution, and the bars are the results from Johnson

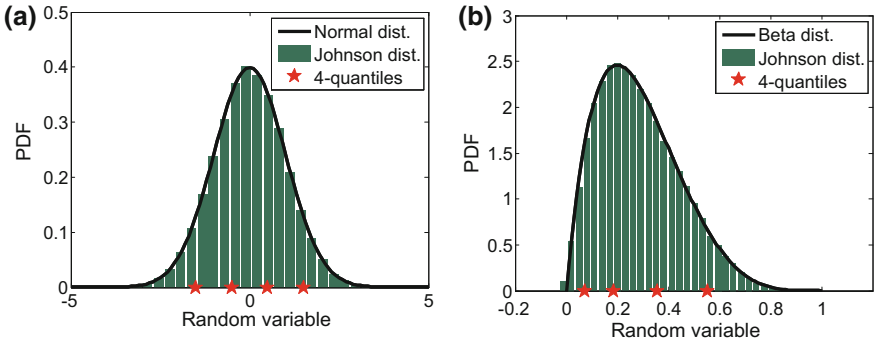


Fig. 7.42 Example of Johnson distribution. **a** Standard normal distribution, **b** beta distribution with $\alpha = 2$, $\beta = 5$

distribution using four quantiles represented as red star markers. Johnson distribution can express any other distribution types when the four quantiles are correctly given.

As an example of distributed data, the crack growth example is considered. Distributed synthetic measurement data are generated using Eq. 4.19 with assumed true damage growth parameter, $C_{\text{true}} = 1.5 \times 10^{-10}$, the initial half crack size, $a_0 = 10\,\text{mm}$, load magnitude, $\Delta\sigma = 78\,\text{MPa}$, and the distributed model parameter, $m \sim U(3.8 - 0.027,\ 3.8 + 0.027)$, and then a small level of noise uniformly distributed between -1 and 1 mm is added, whose result is shown in Fig. 7.43. Each cycle has 5000 samples as the measurement data, whose distribution at 0, 800, 1500, and 2200 cycles are shown in Fig. 7.43b with their true damage size (when $m = 3.8$) shown as black square marker. It is shown that the shape of distributions is changed as cycle increases.

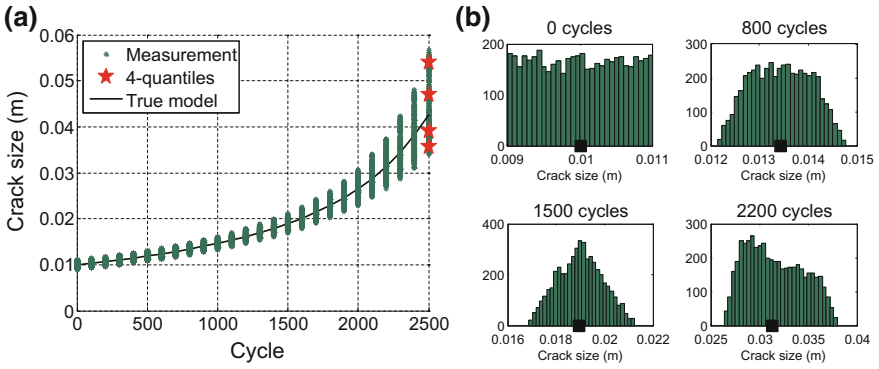


Fig. 7.43 Distributed synthetic data. **a** Measured data at every 100 cycles, **b** distribution of measured data

Degradation prediction with distributed data is performed with one of data-driven approaches, the NN that can easily handle multivariate outputs by adding output nodes. In this example, the four quantiles of Johnson distribution are considered as outputs. Four quantiles whose example at 2500 cycles is shown as red star markers in Fig. 7.43a represent degradation data at each cycle, which makes the number of input and output variables increase by four times. When the previous two data are considered as input variables, total numbers of inputs and outputs become eight (2×4) and four (1×4), respectively.

Figure 7.44 shows damage prediction results from NN with Johnson distribution at 1500 cycles. In Fig. 7.44a, the median of future damage growth is very close to the true one, and 90 % PI also covers damage distribution at every cycle. Figure 7.44b, c show comparison of damage distribution between predicted one and measured one at 1600 and 2400 cycles, and their errors are listed in Table 7.16. The maximum magnitude of error is 5.75 % at 2400 cycles that is 900 cycles ahead prediction from 1500 cycles. These results show that NN using Johnson distribution is applicable for predicting damage distribution with distributed data.

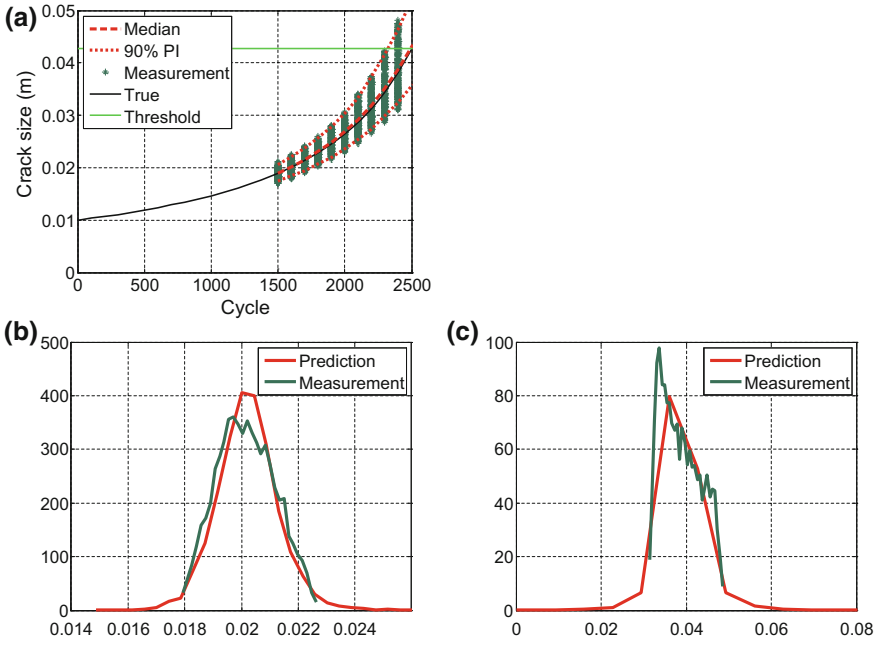


Fig. 7.44 Damage prediction results. **a** Damage growth prediction, **b** damage distribution at 1600 cycles, **c** damage distribution at 2400 cycles

Table 7.16 Errors between prediction results and measurement at 1500 cycles

	Cycles	1600	1800	2000	2200	2400
6.7Q	Measurement	0.0186	0.0210	0.0239	0.0276	0.0327
	Prediction	0.0186	0.0210	0.0240	0.0279	0.0332
	Error (%)	0.03	0.02	0.58	0.95	1.82
30.9Q	Measurement	0.0196	0.0221	0.0253	0.0296	0.0354
	Prediction	0.0197	0.0225	0.0261	0.0308	0.0374
	Error (%)	0.63	1.54	2.89	4.18	5.75
69.1Q	Measurement	0.0207	0.0237	0.0278	0.0333	0.0413
	Prediction	0.0207	0.0237	0.0276	0.0329	0.0402
	Error (%)	0.10	0.01	0.59	1.35	2.66
93.3Q	Measurement	0.0218	0.0251	0.0296	0.0363	0.0463
	Prediction	0.0218	0.0253	0.0299	0.0363	0.0453
	Error (%)	0.32	0.82	0.78	0.09	2.20

References

Adlouni SEl, Favre AC, Bobe B (2006) Comparison of methodologies to assess the convergence of Markov chain Monte Carlo methods. Comput Stat Data Anal 50(10):2685–2701

Agogino A, Goebel K (2007) Mill data set. BEST lab, UC Berkeley. NASA Ames Prognostics Data Repository. Available via http://ti.arc.nasa.gov/tech/dash/pcoe/prognostic-data-repository. Accessed 6 June 2016

Andrieu C, de Freitas N, Doucet A et al (2003) An introduction to MCMC for machine learning. Mach Learn 50(1):5–43

Archard JF (1953) Contact and rubbing of flat surfaces. J Appl Phys 24:981–988

Bao T, Peng Y, Cong P et al (2010) Analysis of crack propagation in concrete structures with structural information entropy. Sci China Technol Sci 53(7):1943–1948

Bolander N, Qiu H, Eklund N et al (2009) Physics-based remaining useful life prediction for aircraft engine bearing prognosis. Paper presented at the annual conference of the prognostics and health management society, San Diego, California, USA, 27 Sept–1 Oct 2009

Boškoski P, Gašperin M, Petelin D (2012) Bearing fault prognostics based on signal complexity and Gaussian process models. Paper presented at the IEEE international conference on prognostics and health management, Denver, Colorado, USA, 18−21 June 2012

Brillouin L (1956) Science and information theory. Dover Publications Inc, New York

Cantizano A, Carnicero A, Zavarise G (2002) Numerical simulation of wear-mechanism maps. Comput Mater Sci 25:54–60

Carpinteri A, Paggi M (2007) Are the Paris' law parameters dependent on each other? Frattura ed Integrità Strutturale 2:10–16

Celaya JR, Saxena A, Saha S et al (2011) Prognostics of power MOSFETs under thermal stress accelerated aging using data-driven and model-based methodologies. Paper presented at the annual conference of the prognostics and health management society, Montreal, Quebec, Canada, 25–29 Sept 2011

Frigg R, Werndl C (2010) Entropy—a guide for the perplexed. In: Beisbart C, Hartmann S (eds) Probabilities in physics. Oxford University Press, Oxford

Haward I (1994) A review of rolling element bearing vibration "detection, diagnosis and prognosis". No. DSTO-RR-0013. Defense Science and Technology Organization Canberra. Available via http://dspace.dsto.defence.gov.au/dspace/bitstream/1947/3347/1/DSTO-RR-0013%20PR.pdf. Accessed 6 June 2016

He D, Bechhoefer E (2008) Development and validation of bearing diagnostic and prognostic tools using HUMS condition indicators. Paper presented at the IEEE aerospace conference, Big Sky, Montana, USA, 1−8 March 2008

IEEE PHM (2012) Prognostic challenge: outline, experiments, scoring of results, winners. Available via http://www.femto-st.fr/f/d/IEEEPHM2012-Challenge-Details.pdf. Accessed 6 June 2016

Johnson NL (1949) Systems of frequency curves generated by methods of translation. Biometrika 36:149–176

Kim NH, Won D, Buris D et al (2005) Finite element analysis and validation of metal/metal wear in oscillatory contacts. Wear 258(11–12):1787–1793

Kim HE, Tan A, Mathew J et al (2012) Bearing fault prognosis based on health state probability estimation. Expert Syst Appl 39:5200–5213

Lee J, Qiu H, Yu G et al (2007) Rexnord technical services. Bearing data set. IMS, University of Cincinnati. NASA Ames Prognostics Data Repository. Available via http://ti.arc.nasa.gov/tech/dash/pcoe/prognostic-data-repository. Accessed 6 June 2016

Li R, Sopon P, He D (2012) Fault features extraction for bearing prognostics. J Intell Manuf 23:313–321

Lim SC, Ashby MF (1990) Wear-mechanism maps. Scr Metall Mater 24(5):805–810

Loutas TH, Roulias D, Georgoulas G (2013) Remaining useful life estimation in rolling bearings utilizing data-driven probabilistic e-support vectors regression. IEEE Trans Reliab 62 (4):821–832

Martín J, Pérez C (2009) Bayesian analysis of a generalized lognormal distribution. Comput Stat Data Anal 53:1377–1387

Mauntler N, Kim NH, Sawyer WG et al (2007) An instrumented crank-slider mechanism for validation of a combined finite element and wear model. Paper presented at the 22nd annual meeting of American society of precision engineering, Dallas, Texas, 14–19 Oct 2007

Mukras S, Kim NH, Mauntler NA et al (2010) Analysis of planar multibody systems with revolute joint wear. Wear 268(5–6):643–652

Nectoux P, Gouriveau R, Medjaher K et al (2012) Pronostia: an experimental platform for bearings accelerated degradation test. Paper presented at the IEEE international conference on prognostics and health management, Denver, Colorado, USA, 18–21 June 2012

Nelson W (1990) Accelerated testing: statistical models, test plans, and data analysis. Wiley, Hoboken

Newman JC Jr, Phillips EP, Swain MH (1999) Fatigue-life prediction methodology using small-crack theory. Int J Fatigue 21:109–119

Park JI, Bae SJ (2010) Direct prediction methods on lifetime distribution of organic light-emitting diodes from accelerated degradation tests. IEEE Trans Reliab 59(1):74–90

Plummer M, Best N, Cowles K et al (2006) CODA: convergence diagnosis and output analysis for MCMC. R News 6(1):7–11

Qiu H, Lee J, Lin J (2006) Wavelet filter-based weak signature detection method and its application on roller bearing prognostics. J Sound Vib 289:1066–1090

Saxena A, Celaya J, Saha B et al (2009) On applying the prognostic performance metrics. Paper presented at the annual conference of the prognostics and health management society, San Diego, California, USA, 27 Sept–1 Oct 2009

Schmitz TL, Sawyer WG, Action JE et al (2004) Wear rate uncertainty analysis. J Tribol 126:802–808

Shannon CE (1948) A mathematical theory of communication. Bell Syst Tech J 27(379–423):623–656

Sinclair GB, Pierie RV (1990) On obtaining fatigue crack growth parameters from the literature. Int J Fatigue 12(1):57–62

Skima H, Medjaher K, Zerhouni N (2014) Accelerated life tests for prognostic and health management of MEMS devices. Paper presented at the 2nd European conference of the prognostics and health management society. Nantes, France, 8–10 July 2014

Sutrisno E, Oh H, Vasan ASS et al (2012) Estimation of remaining useful life of ball bearings using data driven methodologies. Paper presented at the IEEE international conference on prognostics and health management, Denver, Colorado, USA, 18–21 June 2012

Walker JS (1996) Fast Fourier transforms. CRC Press, Boca Raton

Walker SG, Gutiérrez-Peña E (1999) Robustifying Bayesian procedures. In: Bernardo JM, Berger JO, Darwin AP et al (eds) Bayesian statistics, vol 6. Oxford University Press, Oxford, pp 685–710

Wang Z, Nakamura T (2004) Simulations of crack propagation in elastic–plastic graded materials. Mech Mater 36:601–622

Yu WK, Harris TA (2001) A new stress-based fatigue life model for ball bearings. Tribol Trans 44 (1):11–18

Index

N.-H. Kim et al., *Prognostics and Health Management of Engineering Systems*,
DOI 10.1007/978-3-319-44742-1