

## Chapter 2

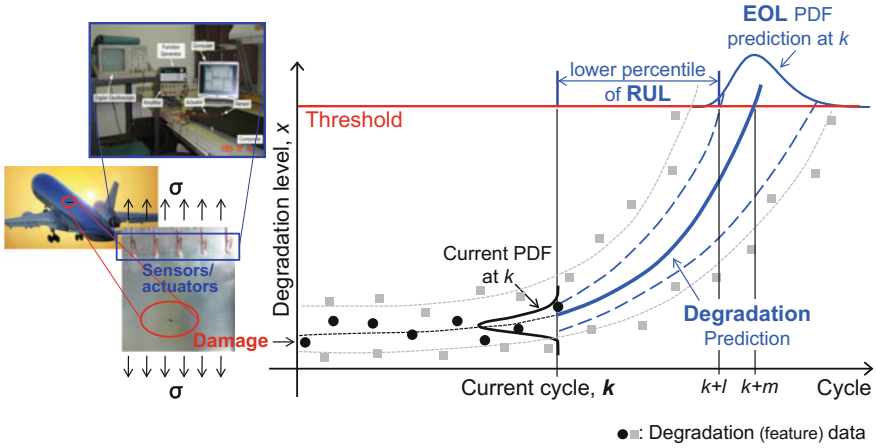
# Tutorials for Prognostics

### 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



**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

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

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

$$\frac{da}{dN} = C(\Delta\sigma\sqrt{\pi a})^m \quad (2.1)$$

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

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

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  $\varepsilon_k$ . Therefore, the relationship between measured data and simulation output can be written as

$$y_k = z_k + \varepsilon_k \quad (2.3)$$

In general, the error  $\varepsilon_k$  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

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

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  $\varepsilon_k$  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  $(t_k, y_k)$ ,  $k = 1, \dots, n_y$ . A vector of measured degradation data can be denoted as  $\mathbf{y} = \{ y_1 \quad y_2 \quad \dots \quad y_{n_y} \}^T$ . 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

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

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  $\mathbf{e} = \{ \varepsilon_1 \quad \varepsilon_2 \quad \dots \quad \varepsilon_{n_y} \}^T = \mathbf{y} - \mathbf{z}$ . The sum of square errors can be defined in the following vector operation:

$$SS_E = \mathbf{e}^T \mathbf{e} = \{ \mathbf{y} - \mathbf{z} \}^T \{ \mathbf{y} - \mathbf{z} \} = \{ \mathbf{y} - \mathbf{X}\boldsymbol{\theta} \}^T \{ \mathbf{y} - \mathbf{X}\boldsymbol{\theta} \} \quad (2.6)$$

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

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

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

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

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.<sup>1</sup> Blanks in the code will be filled for a specific example.

```
[LS]: MATLAB code for Least Squares

1  %%Identify parameters (theta)
2  y=[ ]'; % a vector of measured data
3  x=[ ]'; % a vector/matrix of input
4  X=[ ]; % a design matrix, e.g., Eq. 2.5
5  thetaH=(X'*X)\(X'*y) % Eq.2.8
6  % This is the same as: thetaH=regress(y,X);
7
8  %%Degradation prediction at xNew
9  xNew=[ ]';
10 XNew=[ ]; % same form as X, but use xNew instead of x
11 zH=XNew*thetaH; % Eq. 2.4 or 2.5
12
13 %%RUL prediction
14 thres=[ ]; % threshold level
15 currt=[ ]; % current cycle
16 syms xEOL
17 XEOL=[ ]; % same form as X, but use xEOL instead of x
18 eolFuc=thres-XEOL*thetaH; % EOL func.
19 eol=double(solve(eolFuc, 'Real', true));
20 rul=min(eol(eol>=0))-currt
```

<sup>1</sup>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:

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

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  $\boldsymbol{\theta}_{\text{true}} = \{ 5 \quad 0.2 \quad 0.1 \}^T$ . However, the true values of parameters are only used to generate the data and are not used in the fitting process.<sup>2</sup> The objective is to identify model parameter  $\boldsymbol{\theta}$  that fits the data best. After

---

<sup>2</sup>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

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

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  $\text{varvec{\hat{\theta}}} = \{\text{trix}5.00.20.1\}^T$ , 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  $\text{varvec{\hat{\theta}}} = \{\text{trix}5.00.20.1\}^T$  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

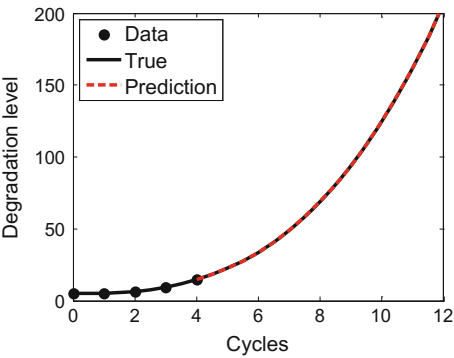
```
in [LS]
zH=XNew*thetaH;
```

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  $\text{varvec{\theta}}_{\text{true}} = \{\text{trix}50.20.1\}^T$  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

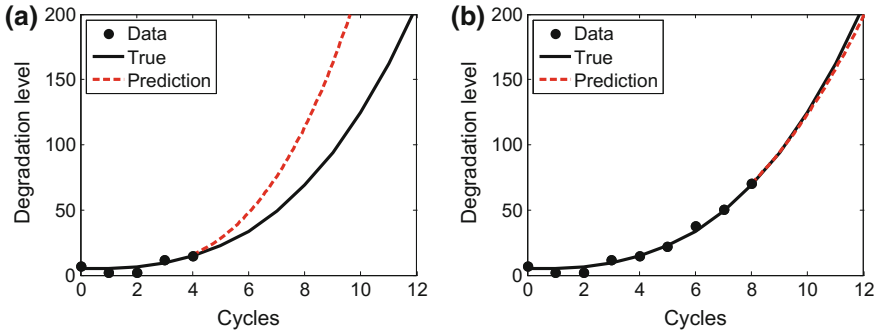


2.2.2.3 Effect of Noise in Data

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  $varvec{\hat{\theta}} = \{trix{4.28-0.290.25}\}^T$ , 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:

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

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

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

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

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  $\{\mathbf{X}^T \mathbf{y}\}$ :

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

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

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

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 ( $L = 1$ ,  $\boldsymbol{\theta}_{\text{true}} = \{5 \quad 0.2 \quad 0.1\}^T$ ).

#### 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

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

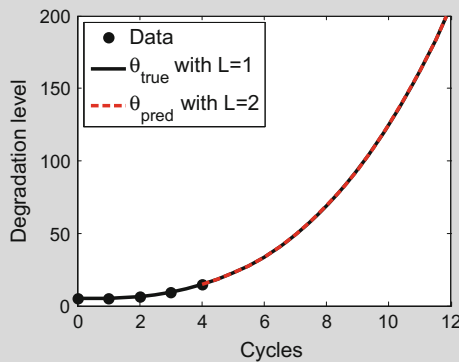
Therefore the model parameter can be calculated as

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

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

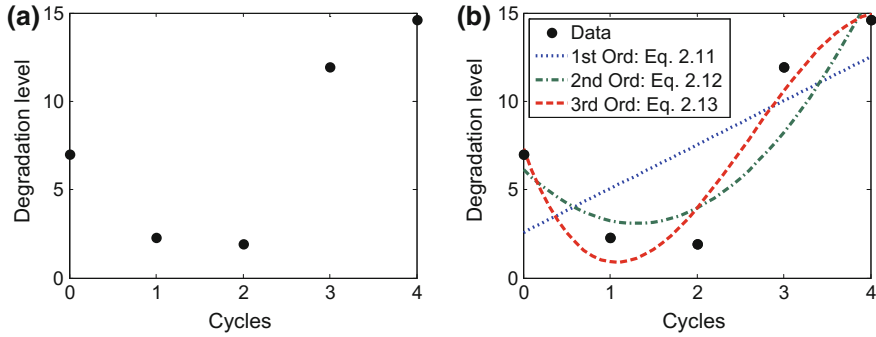
Note that the degradation prediction result under incorrect loading condition (red dashed curve) is exactly the same as the true behavior (black solid curve). This is because the model parameter  $\theta_2$  is correlated with the loading condition  $L$ , thereby model parameters are identified differently 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

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

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

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

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

$$R^2 = \frac{SS_R}{SS_T} = 1 - \frac{SS_E}{SS_T}, \quad (2.14)$$

$$SS_R = \sum_{k=1}^{n_y} (z_k - \bar{y})^2, \quad SS_T = \sum_{k=1}^{n_y} (y_k - \bar{y})^2, \quad SS_E = \sum_{k=1}^{n_y} (y_k - z_k)^2 \quad (2.15)$$

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.,  $SS_T = SS_R + SS_E$ , when the sum of  $y_k$  is equal to the sum of  $z_k$ .<sup>3</sup> 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  $SS_E = 0$  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.

---

<sup>3</sup>More specifically, the following condition is required:  $\mathbf{y}^T \bar{\mathbf{y}} = \mathbf{z}^T \bar{\mathbf{y}}$ , 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

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

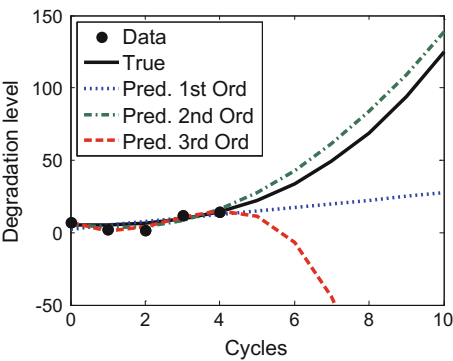
where  $n_y$  and  $n_p$  are, respectively, the number of data and coefficients. The results are listed in Table 2.3. Based on these results with five data, the third-order polynomial function is selected by expecting that this function with given data makes the best prediction of the future behavior. 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

$$z^{(2)} = 5.83 - 3.59t + 1.45t^2,$$

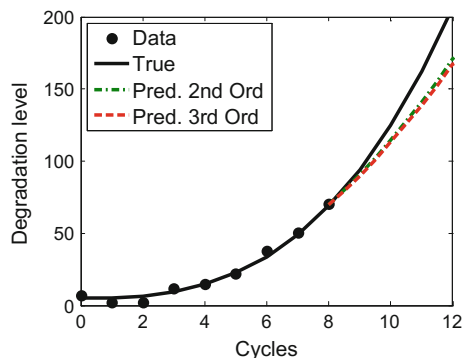
$$z^{(3)} = 5.99 - 3.93t + 1.56t^2 - 0.01t^3$$

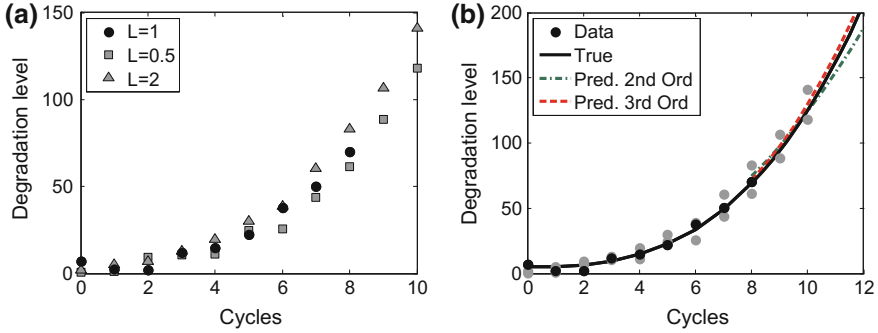
Table 2.3 shows that both functions yield similar  $R^2$  values. Indeed, the prediction results from these two functions are very similar as shown in Fig. 2.6, in which the prediction result is improved compared to ones using five data, but still less accurate than those from the physics-based approach 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 absence 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:  $\hat{\theta} = \{4.28 \quad -0.29 \quad 0.25\}^T$ , which was different from the true parameters  $\theta_{\text{true}} = \{5.0 \quad 0.2 \quad 0.1\}^T$ . 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 ( $8.7(\text{EOL}) - 4(\text{current cycle})$ ), while the true RUL is 6.7 cycles ( $10.7(\text{true EOL}) - 4(\text{current cycle})$ ) (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  $t_{\text{EOL}}$ :

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

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  $t_{\text{EOL}}$  that satisfies the above relation, where the solution  $t_{\text{EOL}}$  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  $z(t; \hat{\theta})$  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:

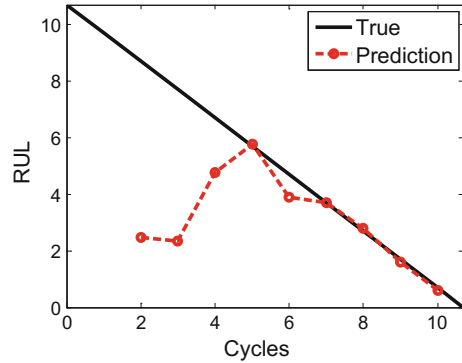
```
in [LS]
thres=150;
currt=4;
syms xEOL
L=1;
XEOL=[1 L*xEOL.^2 xEOL.^3];
eolFuc=thres-XEOL*thetaH; % EOL func.
eol=double(solve(eolFuc,'Real',true));
rul=min(eol(eol>=0))-currt
```

In the above code, the MATLAB code ‘`solve`’ calculates a symbolic solution of algebraic equation with `xEOL` being symbolic variable for  $t_{\text{EOL}}$ . 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  $45^\circ$  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

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

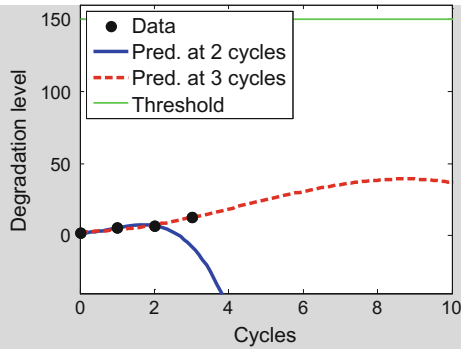
which is also obtained by adding the given information to the Matlab code **[LS]** as

```
% 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  $\mathbf{y}$ , but only the first three,  $\mathbf{y}(1:3)$ , 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 `curr=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');
```



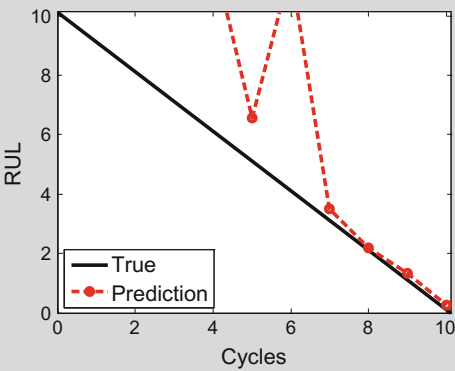
**Fig. E2.2** Degradation prediction at two and three cycles

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 does 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');
```



<b>Table E2.1</b> 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

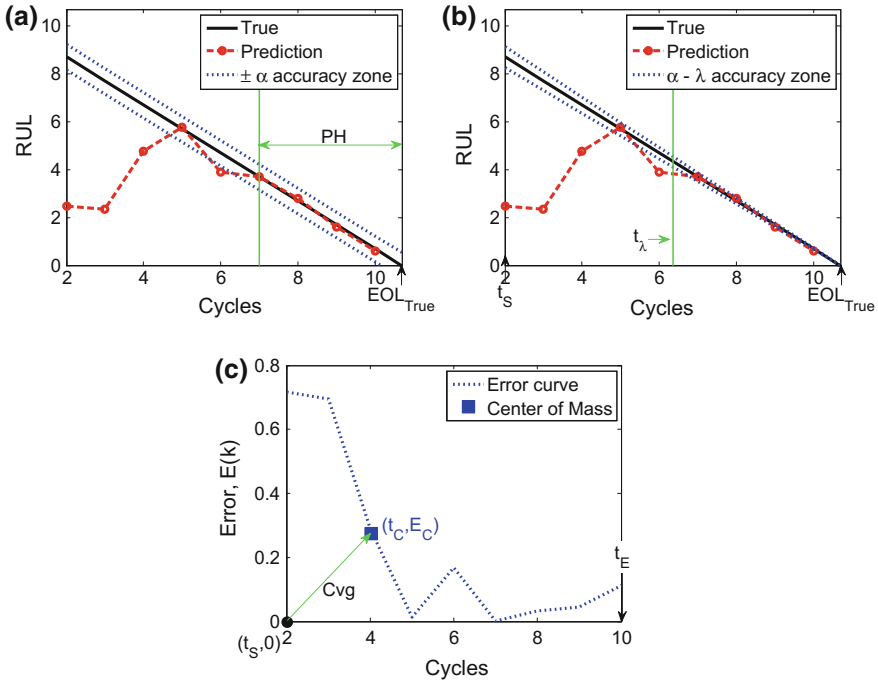
Note that the predicted RULs still have a large error when the time cycle is up to 6 because the model parameters are not estimated accurately. However, starting from seventh time cycle, it is shown a good prediction of RUL. In general, it is true that more data can yield more accurate prediction.

**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  $\alpha$  accuracy zone at a specific cycle  $t_\lambda$ , 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,  $t_\lambda$  is employed, which is expressed with a fraction of  $\lambda$  between starting cycle of RUL prediction,  $t_S$  ( $\lambda = 0$ ) and the true EOL ( $\lambda = 1$ ), which is

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

In this example,  $t_S = 2$  cycles, and  $\lambda$  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  $t_\lambda$  is used, or  $t_\lambda$  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  $t_\lambda$ , which is expressed as

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

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

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

where

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

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 starting 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  $t_{\lambda}(t_{\text{lam}})$  is calculated as seven, so RA is high as shown in Fig. E2.3, but CRA is low.

```
[Metric]: MATLAB code for prognostics metrics
1  %=== Required Information =====
2  rul=[ ]';                                     %[k x 1] or [k x ns]
3  cycle=[ ]';                                  %[k x 1] at rul prediction
4  eolTrue=[ ]';                               % true EOL
5  t_s=[ ]';                                   % starting cycle
6  t_e=[ ]';                                   % ending cycle
7  alpha=[ ]';
8  lambda=[ ]';
9  %=====
10 %%% PH: Prognostic Horizon
11 rulTrue=eolTrue-cycle;
12 alphaZone=eolTrue*alpha;
13 loca=find( rulTrue-alphaZone<rul & rul<rulTrue+alphaZone );
14 a=find( [0 diff(loca')]>1,1,'last' ); if isempty(a); a=1; end
15 ph=eolTrue-cycle(loca(a));
16 disp(['PH= ' num2str(ph)]);
17 %%% ALA: Alpha-Lambda Accuracy
18 t_lam=t_s+(eolTrue-t_s)*lambda;
19 [~, idx_lam]=min(abs(cycle-t_lam)); t_lam=cycle(idx_lam);
20 a=find( rul(idx_lam)>rulTrue(idx_lam)*(1-alpha) ...
21         & rul(idx_lam)<rulTrue(idx_lam)*(1+alpha) );
22 if isempty(a);
23     disp('ALA: False');
24 else disp('ALA: True');
25 end
26 %%% RA & CRA: Relative & Cumulative Relative Accuracy
27 ra=1-abs(rulTrue(idx_lam)-rul(idx_lam))/rulTrue(idx_lam);
28 si=find(cycle>=t_s,1); ei=find(cycle<=t_e,1,'last');
29 cra=mean(1-abs(rulTrue(si:ei)-rul(si:ei))./rulTrue(si:ei));
30 disp(['RA= ' num2str(ra) ', CRA= ' num2str(cra)]);
31 %%% Cvg: Convergence
32 errCurve=abs( rulTrue(si:ei)-rul(si:ei) )./rulTrue(si:ei);
33 cycle1=[cycle(si+1:ei); eolTrue];
34 numer=sum( (cycle1 - cycle(si:ei)).*errCurve );
35 xc=0.5*sum( (cycle1.^2 - cycle(si:ei).^2).*errCurve )/numer;
36 yc=0.5*sum( (cycle1 - cycle(si:ei)).*errCurve.^2 )/numer;
37 cvg=sqrt( (xc-t_s)^2 + yc^2 );
38 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  $\sigma^2$ . 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,  $\varepsilon_k$ , in Eq. 2.3 can be estimated using the sum of square errors in Eq. 2.6 as follows:

$$\hat{\sigma}^2 = \frac{SS_E}{n_y - n_p}, \quad (2.18)$$

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

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

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

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

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  $c$ ,

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

the variance of the two parameters can be obtained as

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

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

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

The above equation means that when the variance of data  $y_i$  is  $\sigma^2$ , 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  $n_y \rightarrow \infty$ ,  $S_{xx} \rightarrow \infty$ .

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<sup>4</sup>

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

which is a  $n_p \times n_p$  covariance matrix of the unknown parameters. Since the variance of error in data ( $\sigma$ ) 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.

<sup>4</sup>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

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

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:

$$SS_E = \{\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\}^T \{\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\} = 66.97$$

$$\hat{\sigma}^2 = \frac{SS_E}{n_y - n_p} = \frac{66.97}{5 - 2} = 22.32$$

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

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

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

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

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

```
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  $\theta_1$  and  $\theta_2$ , 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,  $\mu$ , and standard deviation,  $\sigma$ . 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  $\mu$  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,  $\mu$ , and standard deviation,  $\sigma$ , its sample mean,  $\bar{A}$ , from  $n_y$  data is also normally distributed with a mean  $\mu$  and variance  $\sigma^2/n_y$ .<sup>5</sup>

<sup>5</sup>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  $\mu$  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  $\sigma^2/n_y$ , 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  $\sigma$ . In this case, the normalized distribution of  $\mu$  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,  $\mu$ . It is clear that when  $n_y \rightarrow \infty$ , 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,  $\theta$ . 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

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

where  $t_{n_y-n_p}$  is the  $t$ -distribution with a degree of freedom  $n_y - n_p$  and  $\sigma_{\hat{\theta}_i}$  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.<sup>6</sup>

### 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,  $\theta_2, \theta_3$  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.

- Identify the two unknown parameters  $\theta_2, \theta_3$  and their covariance using Eq. 2.21.
- 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.
- 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.
- Predict RUL when the degradation threshold is 150.

### Solution

- Since  $\theta_1$  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

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

<sup>6</sup>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

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

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

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

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

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

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,  $\hat{\theta}$ . 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  $2 \times 5000$  ( $n_p \times n_s$ ) 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

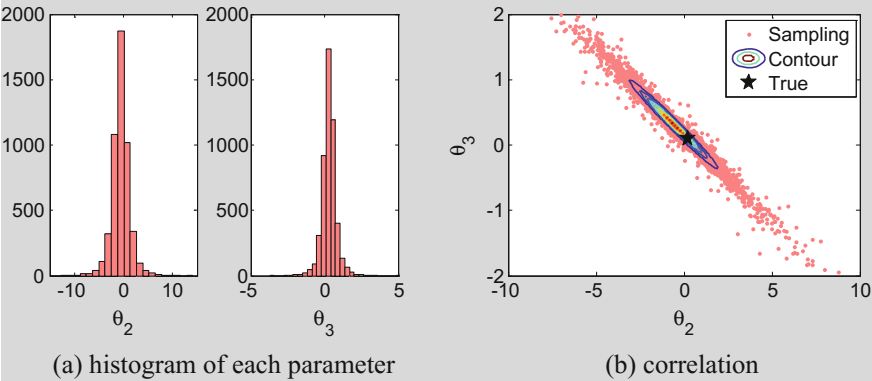
```
alpha=0.05;
thetaCI=prctile(thetaHat',[alpha 0.5 1-alpha]*100)'
```

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:



**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)';
```

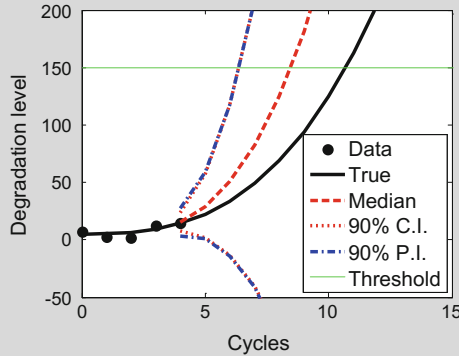
The MATLAB code to obtain Fig. E2.5 is given as

```

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



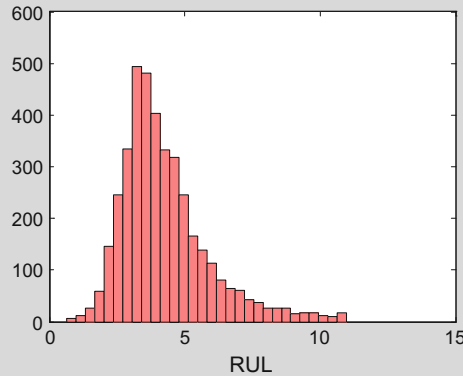
cycle, i.e.,  $RUL = 0$ . Except for these two cases, RUL is predicted by interpolating the two points nearby the threshold, which is in the lines 10–11.

```

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  $\alpha$  accuracy zone. In the  $\alpha-\lambda$  accuracy, the result of  $\alpha-\lambda$  accuracy becomes true when more than  $\beta\%$  of the predictions distribution at  $t_\lambda$  is included in the accuracy zone. Finally, the median of the RUL distribution is used for RA, CRA, and the convergence.

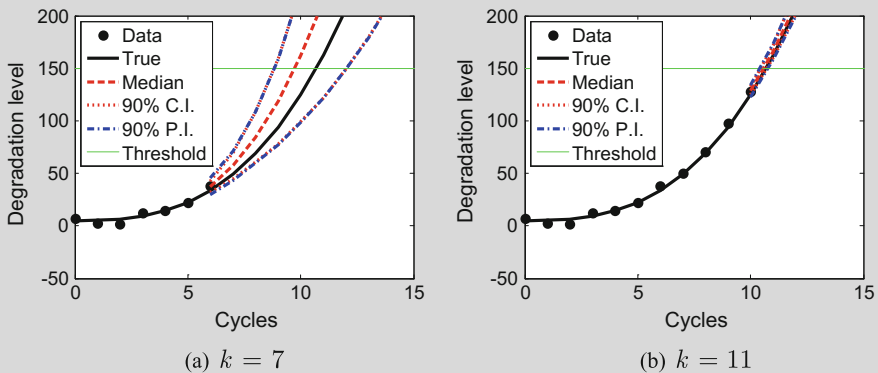
### Example 2.7 RUL prediction with prediction interval

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

- Predict degradation and obtain confidence and prediction interval curves as given in Fig. E2.5 of Example 2.6 when  $k = 7$  and  $k = 11$ .
- 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.
- Evaluate the RUL results in (b) using the prognostics metrics.

#### Solution

- 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.



**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.

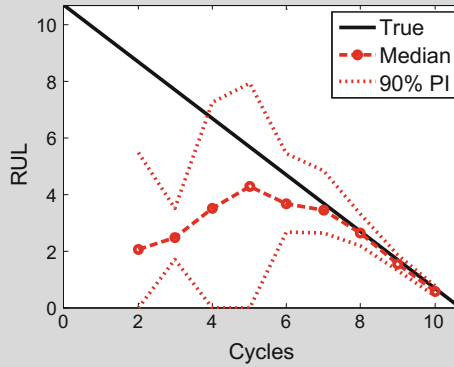
- Example 2.6(d) explained how to predict RUL at a cycle. The following figure is obtained by repeating the RUL calculation for  $k = 3 : 11$  and calculating 90 % prediction intervals. The samples of RUL at different time cycles are stored in array `rul(11,ns)`. The following program can be used to 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,  $\text{beta}=0.5$  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, Hafka 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](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

Prognostics and Health Management of Engineering  
Systems

An Introduction

Kim, N.-H.; An, D.; Choi, J.-H.

2017, XIV, 347 p. 166 illus., 155 illus. in color.,

Hardcover

ISBN: 978-3-319-44740-7