APPLIED MATHEMATICS FOR THE
ANALYSIS OF
BIOMEDICAL DATA
Models, Methods, and MATLAB®
PETER J. COSTA
WILEY
APPLIED MATHEMATICS FOR THE ANALYSIS OF BIOMEDICAL DATA
APPLIED MATHEMATICS
FOR THE ANALYSIS OF
BIOMEDICAL DATA

Models, Methods, and MATLAB®

PETER J. COSTA

WILEY
IN DEDICATION

Per la mia bella Anne.
L’amore della mia vita,
una luce per la nostra famiglia,
un faro per il mondo.

To William J. Satzer
A great mathematical scientist.
A better friend.
IN MEMORIAM

I was the beneficiary of many superlative educators. Among them the late Professors George Craft, Allen Ziebur (SUNY @ Binghamton), and Edward J. Scott (University of Illinois) are owed more than I can possibly repay. It is impossible for me to adequately thank my dissertation advisor, the late Professor Melvyn S. Berger (University of Massachusetts), for his profound influence on my mathematical and personal development. But thanks are all I can presently offer. My great and avuncular late colleague, Frank P. Morrison, was an enormously positive presence in my life.

Finally, to my long deceased grandparents, I offer my most profound gratitude. You braved an ocean to come to a country where you understood neither the language nor the culture. And yet you persevered, raised families, and lived to see your grandson earn a doctorate in applied mathematics. I hope that this book honors your sacrifices, hardships, and accomplishments. *Molto grazie Nonna e Nonno. Io non ho dimenticato.*
CONTENTS

Preface xi
Acknowledgements xiii
About the Companion Website xv
Introduction xvii

1 Data 1
   1.1 Data Visualization, 1
   1.2 Data Transformations, 3
   1.3 Data Filtering, 7
   1.4 Data Clustering, 17
   1.5 Data Quality and Data Cleaning, 25
       References, 28

2 Some Examples 29
   2.1 Glucose–Insulin Interaction, 30
   2.2 Transition from HIV to AIDS, 33
   2.3 Real-Time Polymerase Chain Reaction, 37
       References, 45
       Further Reading, 45
3 SEIR Models 47

3.1 Practical Applications of SEIR Models, 50
References, 88
Further Reading, 90

4 Statistical Pattern Recognition and Classification 93

4.1 Measurements and Data Classes, 94
4.2 Data Preparation, Normalization, and Weighting Matrix, 98
4.3 Principal Components, 104
4.4 Discriminant Analysis, 107
4.5 Regularized Discriminant Analysis and Classification, 112
4.6 Minimum Bayes Score, Maximum Likelihood, and Minimum Bayes Risk, 116
4.7 The Confusion Matrix, Receiver–Operator Characteristic Curves, and Assessment Metrics, 122
4.8 An Example, 127
4.9 Nonlinear Methods, 131
References, 139
Further Reading, 140

5 Biostatistics and Hypothesis Testing 141

5.1 Hypothesis Testing Framework, 142
5.2 Test of Means, 157
5.3 Tests of Proportions, 179
5.4 Tests of Variances, 212
5.5 Other Hypothesis Tests, 232
References, 268
Further Reading, 270

6 Clustered Data and Analysis of Variance 271

6.1 Clustered Matched-Pair Data and Non-Inferiority, 273
6.2 Clustered Data, Assessment Metrics, and Diagnostic Likelihood Ratios, 278
6.3 Relative Diagnostic Likelihood Ratios, 286
6.4 Analysis of Variance for Clustered Data, 291
6.5 Examples for Anova, 300
6.6 Bootstrapping and Confidence Intervals, 314
References, 316
Further Reading, 316

Appendix: Mathematical Matters 317

Glossary of MATLAB Functions 335

Index 407
This is the book I wanted. Or rather I should write, this book would have greatly benefited a considerably younger me.

Just two months after completing my graduate studies, I began my career as a professional mathematician at a prestigious research laboratory. At that time, I was well prepared to prove theorems and to make complicated analytical computations. I had no clue, however, how to model and analyze data. It will come as no surprise, then, that I was not wildly successful in my first job.

But I did learn and mostly that I needed to devise a new approach and new set of tools to solve problems which engineers, physicists, and other applied scientists faced on a day–to–day basis. This book presents some of those tools. It is written with the “new approach” that I often learned the hard way over several decades.

The approach is deceptively simple. Mathematics needs to resemble the world and not the other way around. Most of us learn iteratively. We try something, see how well it works, identify the faults of the method, modify, and try the resulting variation. This is how industrial mathematics is successfully implemented. It has been my experience that the formula of data + mathematical model + computational software can produce insightful and even powerful results. Indeed, this process has been referred to as industrial strength mathematics.

This book and its complimentary exercises have been composed to follow this methodology. The reader is encouraged to “play around” with the data, models, and software to see where those steps lead. I have also tried to streamline the presentation, especially with respect to hypothesis testing, so that the reader can locate a technique which “everyone knows” but rarely writes down.

Most of all, I hope that the reader enjoys this book. Applied mathematics is not a joyless pursuit. Getting at the heart of a matter via mathematical principles has proven most rewarding for me. Please have some fun with this material.
There are 4,632 humans (and a few avians) who found their way into this book. In particular, I wish to thank (at least) the following people.

No two people were more influential and supportive in the development of this work than Anne R. Costa and Dr. William J. Satzer. Anne is my shining light and wife of 30 years. She is aptly named as she greets my typically outlandish suggestions “sweetheart, I have this idea …” with grace and aplomb. Her good humor and editorial skills polished the book’s presentation and prevented several ghastly errors. Bill Satzer read through the entire manuscript, made many insightful recommendations, and helped give the book a coherent theme. He is due significantly more than my thanks.

Dr. Vladimir Krapchev (MIT) first introduced me to the delicate dance of mathematical models and data while Dr. Laurence Jacobs (ADK) showed me the power of computational software. Dr. Adam Feldman (Massachusetts General Hospital) and Dr. James Myrtle (who developed the PSA test) greatly enhanced my understanding of prostate specific antigen levels and the associated risk of prostate cancer. Dr. Clay Thompson (Creative Creek) and Chris Griffin (St. Jude’s Medical) taught me how to program in MATLAB and create numerous data analysis and visualization tools. Thomas Lane and Dr. Thomas Bryan (both of The MathWorks) helped with subtle statistical and computational issues. Professor Charles Roth (Rutgers University) guided me through the mathematical model for real–time polymerase chain reaction. Victoria A. Petrides (Abbott Diagnostics) encouraged the development of outlier filtering and exact hypothesis testing methods. Michelle D. Mitchell helped to develop the HIV/AIDS SEIR model. William H. Moore (Northrup Grumman Systems) and Constantine Arabadjis taught me the fundamentals of the extended Kalman–Bucy filter and helped me implement an automated outbreak detection method.
Dr. Robert Nordstrom (NIH) first introduced me to and made me responsible for the development of statistical pattern recognition techniques as applied to the detection of cervical cancer. His influence in this work cannot be overstated. Dr. Stephen Sum (infraredx) and Professor Gilda Garibotti (Centro Regional Universitario, Bariloche) were crucial resources in the refinement of pattern recognition techniques. Professor Rüdiger Seydel (Universität Köln) invited me to his department and his home so that I could give lectures on my latest developments. Dr. Cleve Moler (The MathWorks) contributed an elegant M–file (lambertw.m) for the computation of the Lambert W function. Professor Brett Ninness (University of Newcastle) permitted the use of his team’s QPC package which allowed me to compute support vector machine boundaries. Sid Mayer (Hologic) and I discussed hypothesis testing methods until we wore out several white boards and ourselves. Professor Richard Ellis (University of Massachusetts) provided keen mathematical and personal insight.

For their ontological support and enduring friendship I thank Carmen Acuña, Gus & Mary Ann Arabadjis, Elizabeth Augustine & Robert Praetorius, Sylvan Elhay & Jula Szuster, Alexander & Alla Eydeland, Alfonso & Franca Farina, Vladimir & Tania Krapchev, Stephen & Claudia Krone, Bill & Carol Link, Jack & Lanette McGovern, Bill & Amy Moore, Ernest & Rae Selig, Jim Stefanis & Cindy Sacco, Rüdiger & Friederike Seydel, Uwe Scholz, Clay & Susan Thompson, and many others. To my family, including my parents (Marie and Peter), brothers (MD, Lorenzo, JC, and E), and sisters (V, Maria, Jaki, and Pam), thank you for understanding my decidedly different view of the world. To my nieces (Coral, Jamie, Jessica, Lauren, Natalie, Nicole, Shannon, Teresa, and Zoë the brave hearted) and nephews (Anthony, Ben, Dimitris, Jack, Joseph, Matthew, and Michael) this book explains, in part, why your old uncle is forever giving you math puzzles: Now go do your homework. A hearty thanks to my mates at the Atkinson Town Pool in Sudbury, Massachusetts. There is no better way to start the day than with a swim and a laugh. Special thanks to the Department of Mathematics & Statistics at the University of Massachusetts (Amherst) for educating and always welcoming me.

To everyone mentioned above, and a those (4562?) I have undoubtedly and inadvertently omitted, please accept my sincere gratitude. This work could not exist without your support.

All mistakes are made by me and me alone, without help from anyone.

P.J. Costa  
Hudson, MA
ABOUT THE COMPANION WEBSITE

This book is accompanied by a companion website:

www.wiley.com/go/costa/appmaths_biomedical_data/

The website includes:

- MATLAB® code
INTRODUCTION

The phrase, *mathematical analysis of biomedical data*, at first glance seems impossibly ambitious. Does the author assert that inherently irregular biological systems can be described with any consistency via the rigid rules of mathematics? Add to this expression *applied mathematics* and a great deal of skepticism will likely fill the reader’s mind. What is the intention of this work?

The answer is, in part, to provide a record of the author’s 30-year career in academics, government, and private industry. Much of that career has involved the analysis of biological systems and data via mathematics. More than this, however, is the desire to provide the reader with a set of tools and examples that can be used as a basis for solving the problems he/she is facing. Some uncommon “tricks of the trade” and methodologies rarely broached by university instruction are provided.

Too often, books are written with only an academic audience in mind. This effort is aimed at working scientists and aspiring apprentices. It can be viewed as a combined textbook, reference work, handbook, and user’s guide. The program presented here will be example driven. It would be disingenuous to say that the mathematics will not be emphasized (the author is, after all, a mathematician). Nevertheless, each section will be motivated by the underlying biology. Each example will contain the MATLAB® code required to produce a figure, result, and/or numerical table.

The book is guided by the idea that applied mathematical models are iterative. Develop a set of equations to describe a phenomenon, measure its effectiveness against data collected to measure the phenomenon, and then modify the model to improve its accuracy. The focus is on solving real examples by way of a mathematical method. Sophistication is not the primary goal. A symbiosis between the rigors of mathematical techniques and the unpredictable nature of biological systems is the point of emphasis.
The book reflects the formula that “mathematics + data + scientific computing = genuine insight into biological systems.” The computing software of choice in this work is MATLAB. The reader can think of MATLAB as another important mathematical tool, akin to the Fourier transform. It (that is, MATLAB) helps transform data into mathematical forms and vice versa.

The presentation of concepts is as follows.

This introduction gives an overview of the book and ends with a representative example of the “mathematics + data + software” paradigm. The first chapter lists a set of guidelines and methods for obtaining, filtering, deciphering, and ultimately analyzing data. These techniques include data visualization, data transformations, data filtering/smoothing, data clustering (i.e., splitting one collection of samples into two or more subclasses), and data quality/data cleaning. In each case, a topic is introduced along with a data set. Mathematical methods used to examine the data are explained. Specific MATLAB programs, developed for use in an industrial setting, are applied to the data. The underlying assumption of this book is that, unlike most academic texts, data must be examined, verified, and/or filtered before a model is applied.

Following the discussion of data, the second chapter provides a view of the utility of differential equations as a modeling method on three distinct medical issues. The interaction of glucose and insulin levels within a human body is described by way of an elementary interaction model. This same approach is applied to the transition of HIV to AIDS within a patient. The HIV/AIDS example portends the susceptible—exposed—infected—recovered/removed models detailed in Chapter 3. The renowned polymerase chain reaction is presented as a coupled set of differential equations. In all of the cases above, the models are either applied to real clinical data or tested for their predictive value. MATLAB functions and code segments are included.

Chapter 3 focuses on mathematical epidemiology. The approach here is decidedly more involved than the examples provided in Chapter 2. The first section concerns a model, built on reported clinical data, that governs the transmission of HIV/AIDS through a population. It is, to the author’s knowledge, the only such unified approach to the spread of a contagious disease. The second example within Chapter 3 concerns a mathematical method developed to predict the outbreak of a contagious disease based on simulated data (that mimic clinical data) of respiratory infections recorded at Boston Children’s Hospital (BCH). Due to HIPAA (Health Insurance Portability and Accountability Act) laws, the author was unable to include the actual BCH data in the analysis. The simulated data, however, very closely resemble the clinical data. In each case, the need to estimate certain parameters crucial to the overall model reflecting real measurements and recorded populations is emphasized.

The fourth chapter concerns statistical pattern recognition methods used in the classification of human specimen samples for disease identification. Again, the application of mathematics to clinical data is the central focus of the exposition. All of the methods described are applied to data. Numerous figures and MATLAB code segments are included to aid the reader. The chapter ends with a presentation of support vector machines and their applications to the classification problem. Special software, developed at the University of Newcastle (Australia), is used with the permission of the design team to calculate the support vector boundaries. This cooperation itself is
an example of how things are done in industry: collaborate with other experts; do not
reinvent the wheel.

Chapter 5 is dedicated to a key component of biostatistics: hypothesis testing. The
mathematical infrastructure is developed to produce the calculations (sample size,
test statistic, hypothesis test, \( p \)-values) required by review agencies for submissions.
This is an encyclopedic chapter listing the most important hypothesis tests and their
variations including equivalence, non-inferiority, and superiority tests. As a point of
reference, note that the author is on the organizing committee for the annual statistical
issues workshop attended by industrial and review agency scientists, statisticians,
and policy analysts. Thus, some of the key statistical matters as presented in these
workshops are included in the chapter.

The final chapter examines clustered (that is, multi-reader/multi-category) data
and the mathematical methods developed to render scientifically justified conclu-
sions. The techniques include hypothesis testing and analysis of variance on clustered
data.

0.1 HOW TO USE THIS BOOK

Throughout these chapters, every effort has been made to present the material in as
direct and clear a manner as possible. It is assumed that the reader is familiar with
elementary differential equations, linear algebra, and statistics. An appendix includes
a brief review of the mathematical underpinnings of these subjects. Further, it is
hoped that the reader has some familiarity with MATLAB. In order to use the M-
files written for this text, the reader must have access to MATLAB and the MATLAB
Statistics Toolbox. A summary of most of the pertinent M-files and MAT-files which
contain the data sets are provided in the Glossary of MATLAB Functions located at
the end of the book. Also within this section is a recommendation for setting up a
workspace to access the M- and MAT-files associated with this text. The reader is
strongly urged to follow the recommendations contained therein. To gain access to
the quadratic programming solver implemented in Chapter 4, the reader must con-
tact Professor Brett Ninness of the University of Newcastle in New South Wales,
Australia (http://sigpromu.org/quadprog/index.html). Expertise in any of the afore-
mentioned areas, however, is not crucial to the use and understanding of this work.

The chapters are, by design, independent units. While methods developed in each
chapter can be applied throughout the book (especially the chapter on data), each
topic can be read without reliance on its predecessor. Whenever possible, it is recom-
mended that the reader have MATLAB at the avail so that the examples can be traced
along with the provided code.

0.2 DATA AND SOLUTIONS

With these ideas in mind, a few words about the source of all modeling efforts
are presented: data. How does an industrial scientist deal with data? As a
INTRODUCTION

A theoretical physicist once noted, *if the data do not fit your model, change the data.* Naturally, this comment was made with tongue firmly implanted in cheek. Here are some guidelines.

(i) **Data Normalization.** When dealing with time-dependent data, it is often advisable to “center time” based upon the given interval. That is, if \( t \) ranges over the discrete values \( \{t_0, t_1, \ldots, t_n\} \), then make calculations on times which start at 0 by subtracting off the initial time \( t_0 \). More precisely, map \( t \) into \( t' = t - t_0 \): \( t_k \mapsto t'_k = t_k - t_0 \). Similarly, some large measurements (say, population) can be given as 6,123,000, 6,730,000, etc. Rather than reporting such large “raw” numbers (which can cause overflow errors in computations), compute in smaller units; that is, 6.123 million. Finally, some data are rendered more stable (numerically) when normalized by their mean \( \bar{x} \) and standard deviation \( s \): \( x_k \mapsto x'_k = \frac{x_k - \bar{x}}{s} \).

(ii) **Data Filtering.** Some collections of measurements are so noisy that they need to be smoothed. This can be achieved via the application of an elementary function (such as the natural logarithm \( \ln \)). Also, smoothing techniques (e.g., the Fourier transform) can be applied. In other cases, a “distinctly different” measurement or outlier from the data can be identified and removed.

(iii) **Think Algorithmically.** A single equation, transformation, or statistic is unlikely to extract a meaningful amount of information from data. Rather, a systematic and consistent algorithmic method will generally be required to understand the underlying information contained within the data. Consequently, the most realistic and hence insightful models will be those that incorporate mathematics and data processing. Some examples of this process are real-time polymerase chain reaction, the extended Kalman–Bucy filter applied to infectious disease transmission, and disease state diagnosis via statistical pattern recognition.

(iv) **Solution Scale.** Engineers are famous for producing “back of the envelope” calculations. Sometimes, this is all that a particular matter requires. Sometimes, however, it is not. There are (at least) three types of “solutions” the industrial scientist is asked to provide: simple calculation, formal study, and “research effort.”

Consider the following example. When processing tissue samples (for pathologists), certain stains are applied to the sample. More specifically, the tissue sample is affixed to a glass slide, the stain is applied, and then the slide is sent off to pathology for review. When this is done on a large scale (say, for cervical cancer screening), the process can be automated. During the automation process, some cells from one slide can migrate to a neighboring slide. Is this “cross-contamination” a serious concern?

The first “solution” could be a basic probability model. What is the probability that a slide sheds a cell? What is the probability that a cell migrates to a neighboring slide? What is the probability that a migrating cell determines the outcome of the pathologist’s review? Multiply these three probabilities together and a “first-order” estimate of the matter’s impact is produced.
The second industrial solution is to design an experiment to test whether the cell migration problem is a cause for cross-contamination from an “abnormal” specimen to a “healthy” one. This entails writing a study protocol complete with sample size justification and the mathematical tools used to determine the outcome of the data analysis. Typically, this is addressed via hypothesis testing. The details can be found in Chapter 5.

The “research effort” solution would require a large-scale mathematical modeling effort. How does a cell migrate from a slide and attach itself to a neighbor? If the stain is applied as a liquid, computation fluid dynamics can come into play. This could easily become a master’s thesis or even doctoral dissertation.

The industrial scientist should be aware that all three solutions are actively pursued within the course of a career. It should never be the case that only one type of solution is considered. Each matter must be approached openly. Sometimes a problem undergoes all three solution strategies before it is “solved.”

Whenever data are part of a modeling effort, these ideas will be kept in mind throughout this book.

0.3 AN EXAMPLE: PSA DATA

The first assignment a scientist is likely to encounter in industry is to “build a model based on these data.” Such a straightforward request should have an equally direct answer. This is rarely the case. Examine the set of measurements listed in Table 0.1.

<table>
<thead>
<tr>
<th>Date</th>
<th>PSA level</th>
</tr>
</thead>
<tbody>
<tr>
<td>16 November 2000</td>
<td>1.7</td>
</tr>
<tr>
<td>21 November 2001</td>
<td>1.9</td>
</tr>
<tr>
<td>24 October 2003</td>
<td>2.0</td>
</tr>
<tr>
<td>6 January 2005</td>
<td>3.6</td>
</tr>
<tr>
<td>23 May 2007</td>
<td>2.31</td>
</tr>
<tr>
<td>25 July 2008</td>
<td>2.98</td>
</tr>
<tr>
<td>28 July 2009</td>
<td>3.45</td>
</tr>
<tr>
<td>30 July 2010</td>
<td>3.56</td>
</tr>
<tr>
<td>19 August 2011</td>
<td>5.35</td>
</tr>
<tr>
<td>27 September 2011</td>
<td>4.51</td>
</tr>
<tr>
<td>7 December 2011</td>
<td>5.77</td>
</tr>
<tr>
<td>20 March 2012</td>
<td>5.01</td>
</tr>
<tr>
<td>9 June 2012</td>
<td>7.19</td>
</tr>
<tr>
<td>3 July 2012</td>
<td>6.23</td>
</tr>
<tr>
<td>21 July 2012</td>
<td>5.98</td>
</tr>
<tr>
<td>21 August 2012</td>
<td>9.11</td>
</tr>
<tr>
<td>3 December 2012</td>
<td>6.47</td>
</tr>
<tr>
<td>6 April 2013</td>
<td>5.68</td>
</tr>
<tr>
<td>7 August 2013</td>
<td>4.7</td>
</tr>
</tbody>
</table>
These are the prostate specific antigen (PSA) levels of a man taken over a 13-year span.

To begin the process of modeling these data, it is reasonable to plot them. Even this relatively simple idea, however, is no easy task. First, the dates need to be converted into units that do not introduce numerical errors when fitting curves. Thus, the first date should correspond to time 0 while the final date should be in the units of years after the initial time. More precisely, set $t_0 = 0$ and $t_n = \text{time (in years) after } t_0$. Figure 0.1 displays the data in these units, and the MATLAB code used to create the plot is listed immediately thereafter.

![PSA measurements](image)

**FIGURE 0.1** PSA measurements.

### MATLAB Commands

(For Figure 0.1)

% Dates measurements taken
% PSA levels
y = [1.7,1.9,2.0,3.6,2.98,3.45,3.56,5.35,4.51,5.77,5.01,7.19,6.23,5.98,9.11,6.47,5.68,4.7];
% Convert dates to date strings (seconds) ...
n = cellfun(@datenum,dates);
% ... and the date strings to years centered at $t_0 = 16$ November 2000
$t = (n - n(1))/365.25$;
% Plot the scores vs. time
h = plot(t,y,'ro'); grid('on');
set(gca,'FontSize',16,'FontName','Times New Roman');
set(h,'MarkerSize',12,'MarkerFaceColor','r');
xlabel('Time (\textit{years})'); ylabel('\textit{PSA Level}');
Tstart = ...
\textsf{strrep(dates{1},',',' - ')} - \textsf{strrep(dates\{end\},',',' ')};
title(Tstart);
axis([0,ceil(t(end)),0,ceil(max(y))]);

For emphasis, note that time 0 is the 16th of November 2000: \( t_0 = 16 \) November 2000. The next step in the modeling effort is to postulate what function would best fit these data. A preliminary examination of the shape of the data in Figure 0.1 suggests (at least) two curves: a parabola and an exponential function. These choices are now considered from a mathematical perspective.

**Curve 1** Quadratic polynomial \( f_2(t) = a_0 + a_1 t + a_2 t^2 \)
The times and PSA levels in Table 0.1 can be mapped into the vectors \( t = [t_0, t_1, \ldots, t_n]^T \) and \( y = [y_0, y_1, \ldots, y_n]^T \), respectively. Here, \( n = 18 \) but notice there are 19 measurements, the first occurring at time \( t_0 \). At the time points \( t \), the curve takes on the values \( y \). Therefore, the coefficients \( a = [a_0, a_1, a_2]^T \) must satisfy the following set of equations

\[
\begin{align*}
y_0 &= a_0 + a_1 \cdot t_0 + a_2 \cdot t_0^2 \\
y_1 &= a_0 + a_1 \cdot t_1 + a_2 \cdot t_1^2 \\
\vdots \\
y_n &= a_0 + a_1 \cdot t_n + a_2 \cdot t_n^2
\end{align*}
\]

(0.1a)

The matrix

\[
V_2(t) = \begin{bmatrix} 1 & t_0 & t_0^2 \\
1 & t_1 & t_1^2 \\
\vdots & \vdots & \vdots \\
1 & t_n & t_n^2 \end{bmatrix}
\]

(0.1b)

and vectors \( a = [a_0, a_1, a_2]^T, y = [y_0, y_1, \ldots, y_n]^T \) permit (0.1a) to be written in the compact form

\[
V_2(t)a = y
\]

(0.2)

In the special case of \( n = 2 \), \( V_2(t) \) is called the Vandermonde matrix. Equation (0.1b) represents a generalized extension of the Vandermonde system. Typically, the vector \( a \) in (0.2) is determined by inverting the matrix \( V_2(t) \) and calculating its product with \( y: a = V_2(t)^{-1} y \). Observe, however, that \( V_2(t) \) is not a square matrix so that
its inverse must be computed by a generalized method. From this vantage point, it is more efficient to first factor the matrix into a product of other matrices that have easy-to-calculate inversion properties. One such factorization is called the QR decomposition. Using this method the matrix, $V_2(t) \in \mathcal{M}_{m \times 3}(\mathbb{R})$, can be written as the product of an orthogonal matrix $Q \in \mathcal{M}_{m \times 3}(\mathbb{R})$ and a positive upper triangular matrix $R \in \mathcal{M}_{m \times 3}(\mathbb{R})$ so that $V_2(t) = QR$. Consequently, $a = R^{-1}Q^Ty$. For details, see Trefethen and Bau [5], Demmel [1], or Moler [4]. While MATLAB does indeed have a QR decomposition function, it is more direct to fit a second-order polynomial to the data $t, y$ using the MATLAB function polyfit.m as below.

### MATLAB Commands

(For Curve 1)

```matlab
% Fit a second-order polynomial to t, y
a = polyfit(t, y, 2);
```

This produces the coefficients $a = [0.051, -0.2535, 2.187]^\top$. The error obtained from fitting a quadratic polynomial to these data is defined via (0.3a). This is known as the least squares error and the coefficients $a$ are selected to minimize (0.3a). Figure 0.2a displays the fitted curve $f_2(t)$ to the data.

$$E[a] = \sum_{k=0}^{n} [f_2(t_k) - y_k]^2 \quad (0.3a)$$

A variation of the least squares error, called the root mean square error, is preferred by the engineering community. For a function $f$ defined via the parameters $a$, the RMS error is written as

$$RMS[a] = \sqrt{\frac{1}{n+1} \sum_{k=0}^{n} \left(f(t_k; a) - y_k\right)^2} \quad (0.3b)$$

This error will be more prominently used in succeeding sections.

Observe that, while the fit appears to be “close,” there is a cumulative least squares error of 21.7 and a RMS error of 1.07. The initial value of the curve is more than 16% larger than the initial measured value: $f_2(t_0) = 1.97 > y_0 = 1.7$, whereas the model produces only 65% of the maximum PSA score. Specifically, $f_2(t_{16}) = 5.96$ versus max$(y) = y_{16} = 9.11$. The terminal value of the quadratic fitted curve, however, is 43% larger than the final PSA reading: $f_2(t_n) = 6.73 > y_n = 4.7$. Therefore, this model is at best a crude approximation to the data. This leads to a second model.

**Curve 2** Exponential $f(t) = a_0e^{\alpha(t-t_0)}$

Since it is desirable to have the model equal the data at the initial time point, take $y_0 = f(t_0) = a_0$. How can the exponent $\alpha$ be estimated? One approach is to use calculus to minimize the least squares error functional $E[a] = \sum_{k=0}^{n} [y_0e^{\alpha(t_k-t_0)} - y_k]^2$. This is
achieved by differentiating $E[\alpha]$ with respect to the exponent $\alpha$ and then setting the derivative to 0. The resulting equation

$$y_0 \sum_{k=0}^{n} (t_k - t_0) e^{2\alpha(t_k - t_0)} = \sum_{k=0}^{n} y_k(t_k - t_0) e^{\alpha(t_k - t_0)}$$

resists a closed form solution for $\alpha$. Consider, instead, transforming the data via the natural logarithm. In this case, \( \ln(f(t)) = \ln(y_0) + \alpha(t - t_0) \). Set $\Delta t = (t - t_0)$, $\Delta z = (z - z_0)$, and $z_k = \ln(y_k)$ for $k = 0, 1, 2, \ldots, n$. Then the transformed error functional becomes $E_{\ln}[\alpha] = \sum_{k=0}^{n} [\Delta z_k - \alpha \cdot \Delta t_k]^2$. To minimize the error with respect to the exponent, differentiate with respect to $\alpha$, and set the corresponding derivative equal to zero.

$$\frac{dE_{\ln}[\alpha]}{d\alpha} = -2 \sum_{k=0}^{n} [\Delta z_k - \alpha \cdot \Delta t_k] \cdot \Delta t_k$$

$$\frac{dE_{\ln}[\alpha]}{d\alpha} = 0 \Rightarrow \alpha = \frac{\sum_{k=0}^{n} \Delta z_k \cdot \Delta t_k}{\sum_{k=0}^{n} (\Delta t_k)^2} \tag{0.4b}$$

For the data of Table 0.1, the value of the exponent is $\alpha = 0.102$ and the value of the error functional is $E_1[0.102] = 23.27$ with a corresponding RMS error of 1.11. These errors are larger than those of the quadratic function. Figure 0.2b shows the fit via the solid curve.

**Warning:** The critical value of $\alpha$ attained via (0.4b) minimizes the error functional $E_{\ln}[\alpha]$ in natural logarithm space. It does not necessarily minimize $E[\alpha]$ in linear space. To find such a value, use the MATLAB commands below to see that $a \approx 0.106$ minimizes $E[\alpha]$. In this case, the value of $E_2[0.106] = 22.59 < 23.27 = E_1[0.102]$ as determined via equation (0.4b). Figure 0.2b shows this corrected fit via the dashed curve. For the exponential model, the parameter vector $a$ corresponds to the values of $a_0 = y_0$ and the estimated exponent $\alpha$.

**MATLAB Commands**

(Figure 0.2a)

```matlab
% Evaluate the second-order polynomial at the times t
F = polyval(a,t);
% Compute the least squares error of the difference in-between the polynomial and data y
E = sum((F-y).^2)
E = 21.6916
% Calculate the RMS error as well
RMS = sqrt((sum((F-y).^2))/numel(y))
RMS = 1.0685
```
MATLAB Commands
(Figure 0.2b)

% Compute the values of the transform data \( z = \ln(y) \) and the time and data centered at \( t_0 \) and \( z_0 \), respectively, \( \Delta t = (t - t_0) \), \( \Delta z = (z - z_0) \)
\[
\begin{align*}
Dt &= t - t(1); \quad z = \log(y); \quad Dz = z - z(1); \\
\end{align*}
\]
% Calculate the “optimal” \( \alpha \) via equation (0.4b)
\[
\begin{align*}
alpha1 &= \frac{\text{sum}(Dz.*Dt)}{\text{sum}(Dt.^2)}; \\
\end{align*}
\]
alpha1 = 0.1020
% Form the function \( f(t) = y_0 \exp(\alpha(t - t_0)) \) with the value of \( \alpha_1 \) above
\[
\begin{align*}
F1 &= y(1)*\exp(alpha1*Dt) ; \\
\end{align*}
\]
% Find the corresponding least squares ...
\[
\begin{align*}
E1 &= \text{sum}( (F1 - y).^2 ) ; \\
E1 &= 23.2671
\end{align*}
\]
% and RMS errors
\[
\begin{align*}
\text{RMS1} &= \sqrt{E1/\text{numel}(y)} \\
\text{RMS1} &= 1.1066
\end{align*}
\]
% Create an in-line error function \( E_2[\alpha] = \sum_k (y_k \exp(\alpha(t_k - t_0)) - y_k)^2 \)
\[
\begin{align*}
Dt &= t - t(1); \\
E2 &= @(alpha) ( \text{sum}((y(1)*\exp(alpha*Dt) - y).^2) ) ; \\
\end{align*}
\]
% Find the value of \( \alpha \) that leads to the minimum of \( E_2[\alpha] \)
% Assume an initial estimate of \( \alpha_0 = 0.1 \)
ao = 0.1;
Ao = fminsearch(E2,ao);
Equations (0.4a) and (0.4b) yield an estimate of the key parameter $\alpha$ for the exponential curve fit $f(t) = a_0 e^{\alpha(t-t_0)}$ by transforming the equation via the natural logarithm ln. An alternate method of achieving such a fit is to first transform the data via ln, fit the data in the transformed space, and finally map the fit back into the original space by inverting the transformation. The transformed data can then be fit with a line. This can be accomplished via linear regression, a common technique used in many data fitting applications. The basic idea of linear regression is that a line $L(t) = a_0 + a_1 t$ models the data. The parameters $\mathbf{a} = [a_0, a_1]^T$ are estimated from the measured information $\mathbf{t} = [t_0, t_1, \ldots, t_n]^T$ and $\mathbf{y} = [y_0, y_1, \ldots, y_n]^T$. This is often written as the paired measurements $\{(t_0, y_0), (t_1, y_1), \ldots, (t_n, y_n)\}$. In minimizing the error functional

\[
E[\mathbf{a}] = \sum_{k=0}^{n} [y_k - a_0 - a_1 \cdot t_k]^2 \tag{0.5a}
\]

over $\mathbf{a}$, it can be shown (see, e.g., Hastie, Tibshirani, and Friedman [3]) that the parameters are estimated as

\[
\hat{a}_1 = \frac{\sum_{k=0}^{n} (t_k - \bar{t})(y_k - \bar{y})}{\sum_{k=0}^{n} (t_k - \bar{t})^2} \tag{0.5b}
\]

\[
\hat{a}_0 = \bar{y} - \hat{a}_1 \cdot \bar{t}
\]

\[
\bar{t} = \frac{1}{n} \sum_{k=0}^{n} t_k, \quad \bar{y} = \frac{1}{n} \sum_{k=0}^{n} y_k \tag{0.5c}
\]

This constitutes a least squares solution of (0.5a) for $\mathbf{a} = [a_0, a_1]^T$. It should be observed, however, that simple linear regression assumes that modeling errors are compensated only in the regression coefficients $\mathbf{a}$. More specifically, the errors occur only with respect to the $x$-axis (which, in this case, corresponds to the time measurements $t_0, t_1, \ldots, t_n$).
Deming regression takes into account errors in both the x-axis \((t_0, t_1, \ldots, t_n)\) and the y-axis \((y_0, y_1, \ldots, y_n)\). It comprises a maximum likelihood estimate for the coefficients \(a\) and is summarized in the formulae below.

\[
\hat{a}_1 = \frac{s_y^2 - s_t^2 + \sqrt{(s_y^2 - s_t^2)^2 + 4s_{t,y}^2}}{2s_{t,y}} \\
\hat{a}_0 = \bar{y} - \hat{a}_1 \cdot \bar{t} \\
\hat{s}_y^2 = \frac{1}{n} \sum_{k=0}^{n} (y_k - \bar{y}), \hat{s}_t^2 = \frac{1}{n} \sum_{k=0}^{n} (t_k - \bar{t}), \hat{s}_{t,y} = \frac{1}{n} \sum_{k=0}^{n} (t_k - \bar{t})(y_k - \bar{y}) \quad (0.6a)
\]

\[
\hat{s}_y^2 = \frac{1}{n} \sum_{k=0}^{n} (y_k - \bar{y}), \hat{s}_t^2 = \frac{1}{n} \sum_{k=0}^{n} (t_k - \bar{t}), \hat{s}_{t,y} = \frac{1}{n} \sum_{k=0}^{n} (t_k - \bar{t})(y_k - \bar{y}) \quad (0.6b)
\]

Note: Since there are \(n + 1\) measurements, the factor normalizing the sample variance \(s_y^2\) and sample covariance \(s_{t,y}\) is \(1/n\). If only \(n\) measurements were available (i.e., \(\{(t_1, y_1), \ldots, (t_n, y_n)\}\)), then the normalizing factor would be \(1/(n - 1)\).

Figures 0.3a and 0.3b show the result of computing a Deming regression fit \(L(t)\) to the transformed data \(\{(t_0, \ln(y_0)), (t_1, \ln(y_1)), \ldots, (t_n, \ln(y_n))\}\) and then mapping the line back into linear space via the inverse transformation \(f(t, a) = \exp(L(t)) = A_0 e^{\hat{a}_1(t-t_0)}\) where \(A_0 = e^{\hat{a}_0}\). As can be seen from the figure, the error in the transformed logarithm space is 0.83, whereas the error in linear space is 22.89. As before, a “good” linear fit in logarithm space does not guarantee the best fit in the original space.

**FIGURE 0.3** Data and model in (a) logarithm space and (b) linear space.