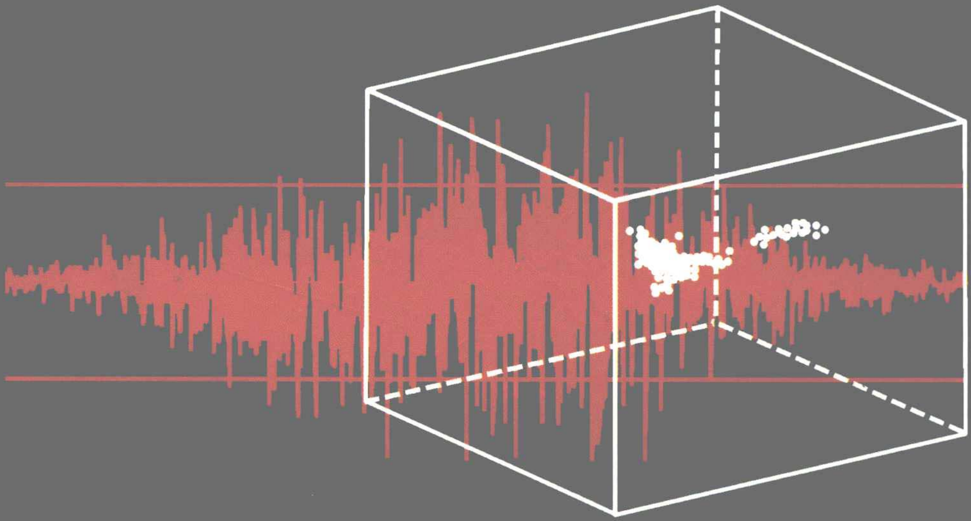
Environmental Data Analysis with MATLAB®



William Menke
Joshua Menke

Environmental Data Analysis with *MatLab*

William Menke
Professor of Earth and Environmental Sciences, Columbia University

Joshua Menke
Software Engineer, JOM Associates

Amsterdam • Boston • Heidelberg • London • New York • Oxford
ELSEVIER Paris • San Diego • San Francisco • Singapore • Sydney • Tokyo

Elsevier
The Boulevard, Langford Lane, Kidlington, Oxford OX5 1GB, UK
Radarweg 29, PO Box 211, 1000 AE Amsterdam, The Netherlands

First edition 2012

Notice
No responsibility is assumed by the publisher for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products, instructions or ideas contained in the material herein.

Library of Congress Cataloging-in-Publication Data
Menke, William.
Environmental data analysis with MatLab/William Menke, Joshua Menke.
p. cm.
Includes bibliographical references and index.
ISBN 978-0-12-391886-4 (alk. paper)
1. Environmental sciences–Mathematical models. 2. Environmental sciences–Data processing. 3. MATLAB. I. Menke, Joshua E. (Joshua Ephraim), 1976- II. Title.
GE45.M37M46 2012
363.7001'5118–dc22
2011014689

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

Printed and bound in USA
12 13 14 10 9 8 7 6 5 4 3 2 1

ISBN: 978-0-12-391886-4

Environmental Data Analysis with *MatLab*

Dedication

FOR
H1

Preface

The first question that I ask an environmental science student who comes seeking my advice on a data analysis problem is *Have you looked at your data?* Very often, after some beating around the bush, the student answers, *Not really*. The student goes on to explain that he or she loaded the data into an analysis package provided by an advisor, or downloaded off the web, and it didn't work. The student tells me, *Something is wrong with my data!* I then ask my second question: *Have you ever used the analysis package with a dataset that did work?* After some further beating around the bush, the student answers, *Not really*. At this point, I offer two pieces of advice. The first is to spend time getting familiar with the dataset. Taking into account what the student has been able to tell me, I outline a series of plots, histograms, and tables that will help him or her prise out its general character and some of its nuances. Second, I urge the student to create several simulated datasets with properties similar to those expected of the data and run the data analysis package on them. The student needs to make sure that he or she is operating it correctly and that it returns the right answers. We then make an appointment to review the student's progress in a week or two. Very often the student comes back reporting, *The problem wasn't at all what I thought it was!*

Then the real works begins, either to solve the problem or if the student has already solved it—which often he or she has—to get on with the data analysis.

Environmental Data Analysis with MatLab is organized around two principles. The first is that real proficiency in data analysis requires analyzing realistic data on a computer, and not merely working through ultra-simplified examples with pencil and paper. The second is that the skills needed to perform data analysis are best learned in a series of steps that alternate between theory and application and that start simple but rapidly expand as one's toolkit of skills grows. The real world puts many impediments in the way of analyzing data—errors of all sorts, missing information, inconvenient units of measurements, inscrutable data formats, and more. Consequently, real proficiency is as much about confidence and experience as it is about formal knowledge of techniques. This book teaches a core set of techniques that are widely applicable across all of Environmental Science, and it reinforces them by leading the student through a series of case studies on real-world data that has both the richness and the blemishes inherent in real-world things.

Two fundamental themes are used to tie together many different data analysis techniques:

The first is that measurement *error* is a fundamental aspect of observation and experiment. Error has a profound influence on the way that knowledge is distilled from data. We use probability theory to develop the concept of *covariance*, the key tool for quantifying error. We then show how covariance propagates through a chain of calculations leading to a result that possesses uncertainty. Dealing with that uncertainty is as important a part of data analysis as arriving at the result, itself. From Chapter 3, where it is introduced, through the book's end, we are always returning to the idea of the propagation of error.

The second is that many problems are special cases of a *linear model* linking the observations to the knowledge that we aspire to derive from them. Measurements of the world around us create data, numbers that describe the results of observations and experiments. But measurements, in and of themselves, are of little utility. The purpose of data analysis is to distill them down to a few significant and insightful *model parameters*. We develop the idea of the linear model in Chapter 4 and in subsequent chapters show that very many, seemingly different data analysis techniques are special cases of it. These include curve fitting, Fourier analysis, filtering, factor analysis, empirical function analysis and interpolation. While their uses are varied, they all share a common structure, which when recognized makes understanding them easier. Most important, covariance propagates through them in nearly identical ways.

As the title of this book implies, it relies very heavily on *MatLab* to connect the theory of data analysis to its practice in the real world. *MatLab*, a commercial product of *The MathWorks, Inc.*, is a popular scientific computing environment that fully supports data analysis, data visualization, and data file manipulation. It includes a scripting language through which complicated data analysis procedures can be developed, tested, performed, and archived. *Environmental Data Analysis with MatLab* makes use of scripts in three ways. First, the text includes many short scripts and excerpts from scripts that illustrate how particular data analysis procedures are actually performed. Second, a set of complete scripts and accompanying datasets is provided as a companion to the book. They implement all of the book's figures and case studies. Third, each chapter includes recommended homework problems that further develop the case studies. They require existing scripts to be modified and new scripts to be written.

Environmental Data Analysis with MatLab is a relatively short book that is appropriate for a one-semester course at the upper-class undergraduate and graduate level. It requires a working knowledge of calculus and linear algebra, as might be taught in a two-semester undergraduate calculus course. It does *not* require any knowledge of differential equations or more advanced forms of applied mathematics. Students with some familiarity with the practice of environmental science and with its underlying issues will be better able to put examples in context, but detailed knowledge of the science is not required. The book is self-contained; it can be read straight through, and profitably, even by someone with no access to *MatLab*. But it is meant to be used in a setting where students are actively using *MatLab* both as an aid to studying (i.e., by reproducing the case studies described in the book) and as a tool for completing the recommended homework problems.

Environmental Data Analysis with MatLab uses six exemplary environmental science datasets:

Air temperature,
Chemical composition of sea floor samples,
Ground level ozone concentration,
Sea surface temperature,
Stream flow, and
Water quality.

Most datasets are used in several different contexts and in several different places in the text. They are used both as a test bed for particular data analysis techniques and to illustrate how data analysis can lead to important insights about the environment.

Chapter 1, *Data Analysis with MatLab*, is a brief introduction to *MatLab* as a data analysis environment and scripting language. It is meant to teach *barely enough* to enable the reader to understand the *MatLab* scripts in the book and to begin to start using and modifying them. While *MatLab* is a fully featured programming language, *Environmental Data Analysis with MatLab* is not a book on computer programming. It teaches scripting mainly by example and avoids long discussions on programming theory.

Chapter 2, *A First Look at Data*, leads students through the steps that, in our view, should be taken when first confronted with a new dataset. Time plots, scatter plots, and histograms, as well as simple calculations, are used to examine the data with the aim both of understanding its general character and spotting problems. We take the position that *all data*sets have problems—errors, data gaps, inconvenient units of measurement, and so forth. Such problems should not scare a person away from data analysis! The chapter champions the use of the *reality check*—checking that observations of a particular parameter really have the properties that we know it must possess. Several case study datasets are introduced, including a hydrograph from the Neuse River (North Carolina, USA), air temperature from Black Rock Forest (New York), and chemical composition from the floor of the Atlantic Ocean.

Chapter 3, *Probability and What It Has to Do with Data Analysis,* is a review of probability theory. It develops the techniques that are needed to understand, quantify, and propagate measurement error. Two key themes introduced in this chapter and further developed throughout the book are that error is an unavoidable part of the measurement process and that error in measurement propagates through the analysis to affect the conclusions. Bayesian inference is introduced in this chapter as a way of assessing how new measurements improve our state of knowledge about the world.

Chapter 4, *The Power of Linear Models*, develops the theme that making inferences from data occurs when the data are distilled down to a few parameters in a quantitative model of a physical process. An integral part of the process of analyzing data is developing an appropriate quantitative model. Such a model links to the questions that one aspires to answer to the parameters upon which the model depends, and ultimately, to the data. We show that many quantitative models are *linear* in form and, thus, are very easy to formulate and manipulate using the techniques of linear algebra. The method of least squares, which provides a means of estimating model parameters from data, and a rule for propagating error are introduced in this chapter.

Chapter 5, *Quantifying Preconceptions*, argues that we usually know things about the systems that we are studying that can be used to supplement actual observations.

Temperatures often lie in specific ranges governed by freezing and boiling points. Chemical gradients often vary smoothly in space, owing to the process of diffusion. Energy and momentum obey conservation laws. The methodology through which this prior information can be incorporated into the models is developed in this chapter. Called generalized least squares, it is applied to several substantial examples in which prior information is used to fill in data gaps in datasets.

Chapter 6, *Detecting Periodicities*, is about spectral analysis, the procedures used to represent data as a superposition of sinusoidally varying components and to detect periodicities. The key concept is the Fourier series, a type of linear model in which the data are represented by a mixture of sinusoidally varying components. The chapter works to make the student completely comfortable with the Discrete Fourier Transform (DTF), the key algorithm used in studying periodicities. Theoretical analysis and a practical discussion of *MatLab's* DFT function are closely interwoven.

Chapter 7, *The Past Influences the Present*, focuses on using past behavior to predict the future. The key concept is the *filter*, a type of linear model that connects the past and present states of a system. Filters can be used both to quantify the physical processes that connect two related sets of measurements and to predict their future behavior. We develop the *prediction error filter* and apply it to hydrographic data, in order to explore the degree to which stream flow can be predicted. We show that the filter has many uses in addition to prediction; for instance, it can be used to explore the underlying processes that connect two related types of data.

Chapter 8, *Patterns Suggested by Data*, explores linear models that characterize data as a mixture of a few significant patterns, whose properties are determined by the data, themselves (as contrasted to being imposed by the analyst). The advantage to this approach is that the patterns are a distillation of the data that bring out features that reflect the physical processes of the system. The methodology, which goes by the names, *factor analysis* and *empirical orthogonal function (EOF)* analysis, is applied to a wide range of data types, including chemical analyses and images of sea surface temperature (SST). In the SST case, the strongest pattern is the El Niño climate oscillation, which brings immediate attention to an important instability in the ocean–atmosphere system.

Chapter 9, *Detecting Correlations Among Data*, develops techniques for quantifying correlations within datasets, and especially within and among time series. Several different manifestations of correlation are explored and linked together: from probability theory, covariance; from time series analysis, cross-correlation; and from spectral analysis, coherence. The effect of smoothing and band-pass filtering on the statistical properties of the data and its spectra is also discussed.

Chapter 10, *Filling in Missing Data*, discusses the interpolation of one and two dimensional data. Interpolation is shown to be yet another special case of the linear model. The relationship between interpolation and the gap-filling techniques developed in Chapter 5 are shown to be related to different approaches for implementing prior information about the properties of the data. Linear and spline interpolation, as well as kriging, are developed. Two-dimensional interpolation and Delaunay triangulation, a critical technique for organizing two-dimensional data, are explained. Two dimensional Fourier transforms, which are also important in many two-dimensional data analysis scenarios, are also discussed.

Chapter 11, *Are My Results Significant?*, returns to the issue of measurement error, now in terms of *hypothesis testing*. It concentrates on four important and very widely applicable statistical tests—those associated with the statistics, Z, χ^2, t, and F. Familiarity with them provides a very broad base for developing the practice of *always* assessing the significance of *any* inference made during a data analysis project. We also show how empirical distributions created by *bootstrapping* can be used to test the significance of results in more complicated cases.

Chapter 12, *Notes*, is a collection of technical notes that supplement the discussion in the main text.

William Menke
December, 2010

Advice on scripting for beginners

For many of you, this book will be your first exposure to scripting, the process of instructing *MatLab* what to do to your data. Although you will be learning something new, many other tasks of daily life will have already taught you relevant skills. Scripts are not so different than travel directions, cooking recipes, carpentry and building plans, and tailoring instructions. Each is in pursuit of a specific goal, a final product that has value to you. Each has a clear starting place and raw materials. And each requires a specific, and often lengthy, set of steps that need to be seen to completion in order to reach the goal. Put the skills that you have learned in these other arenas of life to use!

As a beginner, you should approach scripting as you would approach giving travel directions to a neighbor. Always focus on the goal. Where does the neighbor want to go? What analysis products do you want *MatLab* to produce for you? With a clear goal in mind, you will avoid the common pitfall of taking a drive that, while scenic, goes nowhere in particular. While *MatLab* can make pretty plots and interesting tables, you should not waste your valuable time creating any that does not support your goal.

When starting a scripting project, think about the information that you have. How did you get from point A to point B, the last time that you made the trip? Which turns should you point out to your neighbor as particularly tricky? Which aspects of the script are likely to be the hardest to get right? It is these parts on which you want to focus your efforts.

Consider the value of good landmarks. They let you know when you are on the right road (you will pass a firehouse about halfway) and when you have made the wrong turn (if you go over a bridge). And remember that the confidence-building value of landmarks is just as important as is error detection. You do not want your neighbor to turn back, just because the road seems longer than expected. Your *MatLab* scripts should contain landmarks, too. Any sort of output, such as a plot, that enables you to judge whether or not a section of a script is working is critical. You do not want to spend time debugging a section of your script that already works. Make sure that every script that you write has landmarks.

Scripts relieve you from the tedium of repetitive data analysis tasks. A finished script is something in which you can take pride, for it is a tool that has the potential for helping you in your work for years to come.

Joshua Menke
February, 2011

Contents

1 Data analysis with *MatLab*

1.1 Why *MatLab*?

Data analysis requires computer-based computation. While a person can learn much of the *theory* of data analysis by working through short pencil-and-paper examples, he or she cannot become proficient in the *practice* of data analysis that way—for reasons both good and bad. Real datasets, which are almost always too large to handle manually, are inherently richer and more interesting than stripped-down examples. They have more to offer, but an expanded skill set is required to successfully tackle them. In particular, a new kind of judgment is required for selecting the analysis technique that is right for the problem at hand. These are good reasons. Unfortunately, the practice of data analysis is littered with bad reasons, too, most of which are related to the very steep learning curve associated with using computers. Many practitioners of data analysis find that they spend rather too many frustrating hours solving computer-related problems that have very little to do with data analysis, *per se*. That's bad, especially in a classroom setting where time is limited and where frustration gets in the way of learning.

One approach to dealing with this problem is to conduct all the data analysis within a single software environment—to *limit the damage*. Frustrating software problems will still arise, but fewer than if data were being shuffled between several different

Environmental Data Analysis with *MatLab*. DOI: 10.1016/B978-0-12-391886-4.00001-5

environments. Furthermore, in a group setting such as a classroom, the memory and experience of the group can help individuals solve commonly encountered problems. The trick is to select a single software environment that is capable of supporting *real* data analysis.

The key decision is whether to go with a spreadsheet or a scripting language-type software environment. Both are viable environments for computer-based data analysis. Stable implementations of both are available for most types of computers from commercial software developers at relatively modest prices (and especially for those eligible for student discounts). Both provide support for the data analysis itself, as well as associated tasks such as loading and writing data to and from files and plotting them on graphs. Spreadsheets and scripting languages are radically different in approach, and each has advantages and disadvantages.

In a spreadsheet-type environment, typified by *Microsoft Excel*, data are presented as one or more *tables*. Data are manipulated by selecting the rows and columns of a table and operating on them with functions selected from a menu and with formulas entered into the cells of the table itself. The immediacy of a spreadsheet is both its greatest advantage and its weakness. You see the data and all the intermediate results as you manipulate the table. You are, in a sense, touching the data, which gives you a great sense of what the data are like. More of a problem, however, is keeping track of what you did in a spreadsheet-type environment, as is transferring useful procedures from one spreadsheet-based dataset to another.

In a scripting language, typified by *The MathWorks MatLab*, data are presented as one or more *named variables* (in the same sense that the "c" and "d" in the formula $c = \pi d$ are named variables). Data are manipulated by typing formulas that create new variables from old ones and by running *scripts*, that is, sequences of formulas stored in a file. Much of data analysis is simply the application of well-known formulas to novel data, so the great advantage of this approach is that the formulas that you type usually have a strong similarity to those printed in a textbook. Furthermore, scripts provide a way of both documenting the sequence of formulas used to analyze a particular dataset and transferring the overall data analysis procedure from one dataset to another. The main disadvantage of a scripting language environment is that it hides the data within the variable—not absolutely, but a conscious effort is nonetheless needed to display it as a table or as a graph. Things can go badly wrong in a script-based data analysis scheme without the practitioner being aware of it. Another disadvantage is that the parallel between the syntax of the scripting language and the syntax of standard mathematical notation is nowhere near perfect. One needs to learn to translate one into the other.

While both spreadsheets and scripting languages have *pros* and *cons*, our opinion is that, on balance, a scripting language wins out, at least for the data analysis scenarios encountered in Environmental Science. In our experience, these scenarios often require a long sequence of data manipulation steps before a final result is achieved. Here, the self-documenting aspect of the script is paramount. It allows the practitioner to review the data processing procedure both as it is being developed and years after it has been completed. It provides a way of communicating *what you did*, a process that is at the heart of science.