

Statistical Modeling

A FRESH APPROACH

DANIEL T. KAPLAN

MACALESTER COLLEGE

Copyright (c) 2009 by Daniel Kaplan.

All rights reserved. No part of the material protected by this copyright notice may be reproduced or utilized in any form, electronic or mechanical, including photocopying, recording, or scanning, or stored on any information storage or retrieval system, without written permission from the author.

Cover Photo: The trunk of a scribbly gum eucalyptus tree on Fraser Island in Queensland, Australia. The scribbly gum moth lays its eggs between the old and new bark layers. The larvae burrow between the bark layers, leaving a winding tunnel that is revealed when the old bark falls away. [Photo credit: the author.]

To my wife, Maya.

Contents

1	Statistical Models	3
1.1	Models and their Purposes	6
1.2	Observation and Knowledge	7
1.3	The Main Points of this Book	10
1.4	Introduction to Computation with R	11
2	Data: Cases, Variables, Samples	31
2.1	Kinds of Variables	32
2.2	Data Frames and the Unit of Analysis	34
2.3	Populations and Samples	35
2.4	Longitudinal and Cross-Sectional Samples	37
2.5	Computational Technique	39
3	Describing Variation	43
3.1	Coverage Intervals	45
3.2	The Variance and Standard Deviation	47
3.3	Displaying Variation	51
3.4	Normal and Non-normal Distributions	55
3.5	Categorical Variables	56
3.6	Computational Technique	60
4	The Language of Models	67
4.1	Models as Functions	68
4.2	Multiple Explanatory Variables	70
4.3	Reading a Model	73
4.4	Choices in Model Design	75
4.5	Model Terms	76
4.6	Standard Notation for Describing Model Design	83
4.7	Computational Technique	84

5	Model Formulas and Coefficients	93
5.1	The Linear Model Formula	94
5.2	Linear Models with Multiple Terms	95
5.3	Formulas with Categorical Variables	95
5.4	Model Coefficients Describe Relationships	97
5.5	Model Values and Residuals	98
5.6	Coefficients of Basic Model Designs	98
5.7	Coefficients have Units	102
5.8	Untangling Explanatory Variables	103
5.9	Why Linear Models?	106
5.10	Computational Technique	110
6	Fitting Models to Data	113
6.1	The Least Squares Criterion	113
6.2	Partitioning Variation	117
6.3	Redundancy	119
6.4	Computational Technique	122
7	Measuring Correlation	127
7.1	Properties of R^2	127
7.2	Simple Correlation	129
7.3	Nested Models	132
7.4	Toward Statistical Models	134
7.5	Computational Technique	137
8	Total and Partial Relationships	141
8.1	Total and Partial Relationships	142
8.2	Explicitly Holding Covariates Constant	153
8.3	Adjustment and Truth	154
8.4	Computational Technique	156
9	Model Vectors	163
9.1	Vectors	164
9.2	Vectors as Collections of Numbers	166
9.3	Model Vectors	167
9.4	Model Vectors in the Linear Formula	169
9.5	Model Vectors and Redundancy	170
9.6	Geometry by Arithmetic	171
9.7	Computational Technique	173
10	Statistical Geometry	177
10.1	Case Space and Variable Space	178
10.2	Subspaces and Geometrical Operations	179
10.3	The Model Triangle	185
10.4	Simple Statistics: Geometrically	186
10.5	Drawing Vector Diagrams	189
10.6	Computational Technique	191

11 Geometry with Multiple Vectors	195
11.1 Finding Coefficients	198
11.2 The Coefficient of Determination	199
11.3 Collinearity	200
11.4 Redundancy	203
11.5 Multi-Collinearity	203
11.6 Higher Dimensions	206
11.7 Computational Technique	206
12 Modeling Randomness	207
12.1 Describing Pure Randomness	207
12.2 Settings for Probability Models	210
12.3 Models of Counts	210
12.4 Common Probability Calculations	213
12.5 Continuous Probability Models	214
12.6 The Normal Distribution	217
12.7 Computational Technique	219
12.8 Simulations	224
13 Geometry of Random Vectors	227
13.1 Random Angles	227
13.2 Random Models	230
13.3 Random Walks	231
14 Confidence in Models	235
14.1 The Sampling Distribution	237
14.2 Standard Errors and the Regression Report	240
14.3 Confidence Intervals	242
14.4 Interpreting the Confidence Interval	244
14.5 Confidence in Predictions	245
14.6 Finding the Resampling Distribution	248
14.7 Confidence and Collinearity	258
14.8 Confidence and Bias	260
14.9 Computational Technique	261
15 The Logic of Hypothesis Testing	269
15.1 An Example of a Hypothesis Test	271
15.2 Inductive and Deductive Reasoning	271
15.3 The Null Hypothesis	275
15.4 The p-value	276
15.5 Rejecting by Mistake	277
15.6 Failing to Reject	279
15.7 A Glossary of Hypothesis Testing	281
15.8 Computational Technique	282

16 Hypothesis Testing on Whole Models	287
16.1 The Permutation Test	288
16.2 R^2 and the F Statistic	290
16.3 The ANOVA Report	296
16.4 Visualizing p-values	298
16.5 Tests of Simple Models	300
16.6 Interpreting the p-value	303
16.7 Computational Technique	307
17 Hypothesis Testing on Parts of Models	317
17.1 The Term-by-Term ANOVA Table	318
17.2 Covariates Soak Up Variance	319
17.3 Measuring the Sum of Squares	322
17.4 ANOVA, Collinearity, and Multi-Collinearity	325
17.5 Hypothesis Tests on Single Coefficients	334
17.6 Non-Parametric Statistics	337
17.7 Sample Size and Power	338
17.8 Computational Technique	341
18 Models of Yes/No Variables	345
18.1 The 0-1 Encoding	345
18.2 Inference on Logistic Models	350
18.3 Model Probabilities	353
18.4 Computational Technique	356
19 Causation	361
19.1 Interpreting Models Causally	362
19.2 Causation and Correlation	364
19.3 Hypothetical Causal Networks	367
19.4 Networks and Covariates	369
20 Experiment	383
20.1 Experiments	383
20.2 When Experiments are Impossible	394
20.3 Conclusion	403
Further Readings & Bibliography	405
Datasets	412
Dataset Index	418
R Operator Index	419
Concept Index	422

Preface

The purpose of this book is to provide an introduction to statistics that gives readers a sufficient mastery of statistical concepts, methods, and computations to apply them to authentic systems. By “authentic,” I mean the sort of multivariable systems often encountered when working in the natural or social sciences, commerce, government, law, or any of the many contexts in which data are collected with an eye to understanding how things work or to making predictions about what will happen.

The world is complex and uncertain. We deal with the complexity and uncertainty with a variety of strategies including the scientific method and the discipline of statistics.

Statistics helps to deal with uncertainty, quantifying it so that you can assess how reliable — how likely to be repeatable — your findings are. The scientific method helps to deal with complexity: reduce systems to simpler components, define and measure quantities carefully, do experiments in which some conditions are held constant but others are varied systematically.

Beyond helping to quantify uncertainty and reliability, statistics provides another great insight of which most people are unaware. When dealing with systems involving multiple influences, it is possible and best to deal with those influences simultaneously. By appropriate data collection and analysis, the confusing tangle of influences can sometimes be straightened out (and it is possible to know when the attempt gives ambiguous and unreliable results). In other words, statistics goes hand-in-hand with the scientific method when it comes to dealing with complexity and understanding how systems work.

The statistical methods that can accomplish this are often considered advanced: multiple regression, analysis of covariance, logistic regression, etc. With appropriate software, any method is accessible in the sense of being able to produce a summary report on the computer. But a method is useful only when the user has a way to understand whether the method is appropriate for the situation, what the method is telling about the data, and what the method is not capable of revealing. Computer scientist Richard Hamming (1915-1998) said: “The purpose of computing is insight, not numbers.” Without a solid understanding of the theory that underlies a method, the numbers generated by the computer may not give insight.

The methods of statistics can give tremendous insight, particularly the so-called advanced methods. For this reason, these methods need to be accessible

both computationally and theoretically to the widest possible audience. Historically, the audience has been limited because few people have the algebraic skills needed to approach the methods in the way they are usually presented. But there are many paths to understanding and I have undertaken to find one that takes the greatest advantage of the actual skills that most people already have in abundance.

In trying to meet that challenge, I have made many unconventional choices. Theory becomes simpler when there is a unified framework for treating many aspects of statistics, so I have chosen to present just about everything in the context of models: descriptive statistics as well as inference as well as experimental design.

Another choice is the use of simple geometry to present theory. The underlying geometrical concepts are elementary: lengths, angles, projection, the Pythagorean theorem. Much of the statistical theory can be displayed in a two-dimensional sketch; once in a while three-dimensional visualization is needed. When I have presented the geometrical ideas in this manner to professionals in research groups or at statistics conferences and workshops, I find uniformly that people gain significant insight into their work. A common reaction is, "It can't be that easy." Yes, it can be and it should be.

Also unconventional, but hardly innovative, is the use of resampling and randomization to motivate the concepts of statistical inference. George Cobb [1] cogently describes the logic of statistical inference as the three-Rs: "Randomize, Repeat, Reject." In a decade of teaching statistics, I have found that students can understand this algorithmic logic much better than the derivations of algebraic formulas for means and standard deviations of sampling distributions.

As you might expect from the preceeding comments, algebraic notation and formulas are strongly de-emphasized in this book. I find that most people are not skilled in interpreting them and extracting meaning from them. In any event, formulas are no longer needed as a way to describe calculations since statistical work is now done on the computer.

And then there is software. Some people think that statistics should be taught without computers in order to help develop conceptual understanding. Others think that it is silly to ignore a technology that is universally used in practice and greatly expands our capabilities. Both points of view are right.

The main body of this book is presented in a way that makes little or no reference to software; the statistical concepts are paramount. But each chapter has a section on computational technique that shows how to get things done and aims to give the reader concrete skills in the analysis of data. An extensive set of exercises and classroom activities is published in workbook form, via the Internet. The computer is an effective teaching tool, so many of the exercises and activities make heavy use of it.

The software used is R, a modern, powerful and freely available system for statistical computations and graphics. The book assumes that you know nothing at all about scientific software and, accordingly, introduces R from basics. If you have experience with statistics, you probably already have a preferred software package. So long as that software will fit linear models with multiple

explanatory variables and produce a more-or-less standard regression report, it can be used to follow this book. That said, I strongly encourage you to think about learning and using R. You can learn it easily by following the examples and can be doing productive statistics very quickly. Not only will you easily be able to fit models and get reports, but you can use R to explore ideas such as re-sampling and randomization. If you now use “educational” software, learning R will give you a professional-level tool for use in the future.

For many instructors, this book can support a nice *second* course in statistics — a follow-up to a conventional first introductory course. Increasingly, such a course is needed as more and more young people encounter basic statistical ideas in grade school and many of the topics of the conventional university course are absorbed into the high-school curriculum. At Macalester College, where I developed this book, mainstream students of biology, economics, political science, and so on use this book for their *first* statistics course. Accordingly, the book is written to be self-contained, making no assumption that readers have had any previous formal study in statistics.

Thanks and acknowledgments ...

I have been fortunate to have the assistance and support of many people. Some of the colleagues who have played important roles are David Bressoud, George Cobb, Dan Flath, Tom Halverson, Gary Krueger, Weiwen Miao, Phil Poronnik, Victor Addona, Karen Saxe, Michael Schneider, and Libby Shoop. Critical institutional support was given by Brian Rosenberg, Jan Serie, Dan Hornbach, Helen Warren, and Diane Michelfelder at Macalester and Mercedes Talley at the Keck Foundation.

I received encouragement from many in the statistics education community, including George Cobb, Joan Garfield, Dick De Veaux, Bob delMas, Julie Legler, Milo Schield, Paul Alper, Andy Zieffler, Sharon Lane-Getaz, Katie Makar, Michael Bulmer, and the participants in the monthly “Stat Chat” sessions. Helpful suggestions came from Simon Blomberg, Dominic Hyde, Erik Larson, Julie Dolan, and Kendrick Brown. Michael Edwards helped with proofreading. Nick Trefethen and Dave Saville provided important insights about the geometry of fitting linear models.

Thanks also go to the hundred or so students at Macalester College who enrolled in the early, experimental sessions of Math 155 where many of the ideas in this book were first tried out. Among those students, I want to acknowledge particular help from Alan Eisinger, Caroline Ettinger, Bernd Verst, Wes Hart, Sami Saqer, and Michael Snavely.

Crucial early support for this project was provided by a grant from the Howard Hughes Medical Institute. An important Keck Foundation grant was crucial to the continuing refinement of the approach and the writing of this book.

Finally, my thanks and love to my wife, Maya, and daughters, Tamar, Liat, and Netta, who endured the many, many hours during which I was preoccupied by some or another statistics-related enthusiasm, challenge, or difficulty.

Chapter 1

Statistical Models

All models are wrong. Some models are useful. — George Box

Art is a lie that tells the truth. — Pablo Picasso

This book is about **statistical modeling**, two words that themselves require some definition.

“Modeling” (note the “ing” ending) is a process of asking questions. “Statistical” refers in part to data — the statistical models you will construct will be rooted in data. But it refers also to a distinctively modern idea: that you can measure what you *don’t* know and that doing so contributes to your understanding.

There is a saying, “A person with a watch knows the time. A person with two watches is never sure.” The statistical point of view is that it’s better not to be sure. With two watches you can see how they disagree with each other. This provides an idea of how precise the watches are. You don’t know the time exactly, but knowing the precision tells you something about what you don’t know. The non-statistical certainty of the person with a single watch is merely an uninformed self-confidence: the single watch provides no indication of what the person doesn’t know.

The physicist Ernest Rutherford (1871-1937) famously said, “If your experiment needs statistics, you ought to have done a better experiment.” In other words, if you can make a good enough watch, you need only one: no statistics. This is bad advice. Statistics never hurts. A person with two watches that agree perfectly not only knows the time, but has evidence that the watches are working at high precision. Sensibly, the official world time is based on an average of many atomic clocks. The individual clocks are fantastically precise; the point of averaging is to know when one or more of the clocks is drifting out of precision for some reason.

Why “statistical modeling” and not simply “statistics” or “data analysis?” Many people imagine that data speak for themselves and that the purpose of statistics is to extract the information that the data carry. Such people see data

analysis as an objective process in which the researcher should, ideally, have no influence. This can be true when very simple issues are involved, for instance how precise is the average of the atomic clocks used to set official time or what is the difference in time between two events. But many questions are much more complicated; they involve many variables and you don't necessarily know what is doing what to what.[2]

The conclusions you reach from data depend on the specific questions you ask. Like it or not, the researcher plays an active and creative role in constructing and interrogating data. This means that the process involves some subjectivity. But this is not the same as saying anything goes. Statistical methods allow you to make objective statements about how the data answer your questions. In particular, the methods help you to know if the data show anything at all.

The word “modeling” highlights that your goals, your beliefs, and your current state of knowledge all influence your analysis of data. The core of the scientific method is the formation of hypotheses that can be tested and perhaps refuted by experiment or observation. Similarly, in statistical modeling you examine your data to see whether they are consistent with the hypotheses that frame your understanding of the system under study.

Example 1.1: Grades A woman is applying to law school. The schools she applies to ask for her class rank, which is based on the average of her college course grades.

A simple statistical issue concerns the precision of the grade-point average. This isn't a question of whether the average was correctly computed or whether the grades were accurately recorded. Instead, imagine that you could send two essentially identical students to essentially identical schools. Their grade-point averages might well differ, reflecting perhaps the grading practices of their different instructors or slightly different choices of subjects or random events such as illness or mishaps or the scheduling of classes. One way to think about this is that the students' grades are to some extent random, contingent on factors that are unknown or perhaps irrelevant to the students' capabilities.

How do you measure the extent to which the grades are random? There is no practical way to create “identical” students and observe how their grades differ. But you can look at the variation in a single student's grades — from class to class — and use this as an indication of the size of the random influence in each grade. From this, you can calculate the likely range of the random influences on the overall grade-point average.

Statistical models let you go further in interpreting grades. It's a common belief that there are easy- and hard-grading teachers and that a grade reflects not just the student's work but the teacher's attitude and practices. Statistical modeling provides a way to use data on grades to see whether teachers grade differently and to correct for these differences between teachers. Doing this involves some subtlety, for example taking into account the possibility that strong students take different courses than weaker students.

Example 1.2: Nitrogen Fixing Plants Biologist Michael Anderson studies how plants fix nitrogen in the soil. All plants need nitrogen to grow. Since nitrogen is the primary component of air, there is plenty around. But it's hard for plants to get nitrogen from the air; they get it instead from the soil. Some plants, like alder and soybean, support nitrogen-fixing bacteria in nodules on the plant roots. The plant creates a hospitable environment for the bacteria; the bacteria, by fixing nitrogen in the soil, create a good environment for the plant. In a word, symbiosis.

Anderson is interested in how genetic variation in the bacteria influences the success with which they fix nitrogen. One can imagine using this information to breed plants and bacteria that are more effective at fixing nitrogen and thereby reducing the need for agricultural fertilizer.

Anderson has an promising early result. His extensive field studies indicate that different genotypes of bacteria fix nitrogen at different rates. Unfortunately, the situation is confusing since the different genotypes tend to populate different areas with different amounts of soil moisture, different soil temperatures, and so on. How can he untangle the relative influences of the genotype and the other environmental factors in order to decide whether the variation in genotype is genuinely important and worth further study?

Example 1.3: Sex Discrimination A large trucking firm is being audited by the government to see if the firm pays wages in a racially or sexually discriminatory way. The audit finds wage discrepancies between men and women for "office and clerical workers" but not for other job classifications such as technicians, supervisors, sales personnel, or "skilled craftworkers." It finds no discrepancies based on race.

A simple statistical question is whether the observed difference in average wages for men and women office and clerical workers is based on enough data to be reliable. In answering this question, it actually makes a difference what other groups the government auditors looked at when deciding to focus on sex discrimination in office and clerical workers.

Further complicating matters are the other factors that contribute to people's wages: the kind of job they have, their skill level, their experience. Statistical models can be used to quantify how these various factors contribute and to see whether they account for some or all of the wage discrepancy associated with sex. For instance, it turns out that men on average tend to have more job experience than women, and some or all of the men's higher average wages might be due to this.

Models can help you decide whether this potential explanation is plausible. For instance, if you see that both men's and women's wages increase with experience in the same way, you might be more inclined to believe that job experience is a legitimate factor rather than just a mask for discrimination.

1.1 Models and their Purposes

Many of the toys you played with as a child are models: dolls, balsa-wood airplanes with wind-up propellers, wooden blocks, model trains. But so are many serious objects of the adult world: architectural plans, bank statements, train schedules, the results of medical diagnostic tests, the signals transmitted by a telephone, the equations of physics, the genetic sequences used by biologists. There are too many to list.

What all models have in common is this:

A model is a representation for a particular purpose.

A model might be a physical object or it might be an idea, but it always stands for something else: it's a representation. Dolls stand for babies and other creatures, architectural plans stand for buildings and bridges, a white blood-cell count stands for the function of the immune system.

When you create a model, you have (or ought to have) a purpose in mind. Toys are created for the entertainment and (sometimes) edification of children. The various kinds of toys — dolls, blocks, model airplanes and trains — have a form that serves this purpose. Unlike the things they represent, the toy versions are small, safe, and inexpensive.

Models always leave things out and get some things — many things — wrong. Architectural plans are not houses; you can't live in them. But they are easy to transport, copy, and modify. That's the point. Telephone signals — unlike the physical sound waves that they represent — can be transported over long distances and even stored. A train schedule tells you something important but it obviously doesn't reproduce every aspect of the trains it describes; it doesn't carry passengers.

Statistical models revolve around data. But even so, they are first and foremost models. They are created for a purpose. The intended use of a model should shape the appropriate form of the model and determines the sorts of data that can properly be used to build the model.

There are three main uses for statistical models. They are closely related, but distinct enough to be worth enumerating.

Description. Sometimes you want to describe the range or typical values of a quantity. For example, what's a "normal" white blood cell count? Sometimes you want to describe the relationship between things. Example: What's the relationship between the price of gasoline and consumption by automobiles?

Classification or prediction. You often have information about some observable traits, qualities, or attributes of a system you observe and want to draw conclusions about other things that you can't directly observe. For instance, you know a patient's white blood-cell count and other laboratory measurements and want to diagnose the patient's illness.

Anticipating the consequences of interventions. Here, you intend to do something: you are not merely an observer but an active participant in the sys-

tem. For example, people involved in setting or debating public policy have to deal with questions like these: To what extent will increasing the tax on gasoline reduce consumption? To what extent will paying teachers more increase student performance?

The appropriate form of a model depends on the purpose. For example, a model that diagnoses a patient as ill based on an observation of a high number of white blood cells can be sensible and useful. But that same model would give absurd predictions about intervention: Do you really think that lowering the white blood cell count by bleeding a patient will make the patient better?

To anticipate correctly the effects of an intervention you need to get the direction of cause and effect correct in your models. But for a model used for classification or prediction, it may be unnecessary to represent causation correctly. Instead, other issues, e.g. the reliability of data, can be the most important. One of the thorniest issues in statistical modeling — with tremendous consequences for science, medicine, government, and commerce — is how you can legitimately draw conclusions about interventions from models based on data collected without performing these interventions.

1.2 Observation and Knowledge

How do you know what you know? How did you find it out? How can you find out what you don't yet know? These are questions that philosophers have addressed for thousands of years. The views that they have expressed are complicated and contradictory.

From the earliest times in philosophy, there has been a difficult relationship between knowledge and observation. Sometimes philosophers see your knowledge as emerging from your observations of the world, sometimes they emphasize that the way you see the world is rooted in your innate knowledge: the things that are obvious to you.

This tension plays out on the pages of newspapers as they report the controversies of the day. Does the death penalty deter crime? Does increased screening for cancer reduce mortality? Will paying teachers more improve student outcomes?

Consider the simple, obvious argument for why severe punishment deters crime. Punishments are things that people don't like. People avoid what they don't like. If crime leads to punishment, then people will avoid committing crime.

Each statement in this argument seems perfectly reasonable, but none of them is particularly rooted in observations of actual and potential criminals. It's artificial — a learned skill — to base knowledge such as "people avoid punishment" on observation. It might be that this knowledge was formed by our own experiences, but usually the only explanation you can give is something like, "that's been my experience" or give one or two anecdotes.

When observations contradict opinions — opinions are what you think you know — people often stick with their opinions. Put yourself in the place of

someone who believes that the death penalty really does deter crime. You are presented with accurate data showing that when a neighboring state eliminated the death penalty, crime did not increase. So do you change your views on the matter? Possibly, but possibly not. A skeptic can argue that it's not just punishment but also other factors that influence the crime rate, for instance the availability of jobs. Perhaps it was that a generally improving economic condition in the other state kept the crime rate steady even at a time when society is imposing lighter punishments.

It's difficult to use observation to inform knowledge because relationships are complicated and involve multiple factors. It isn't at all obvious how people can discover or demonstrate causal relationships through observation. Suppose one school district pays teachers well and another pays them poorly. You observe that the first district has better student outcomes than the second. Can you legitimately conclude that teacher pay accounts for the difference? Perhaps something else is at work: greater overall family wealth in the first district (which is what enabled them to pay teachers more), better facilities, smaller classes, and so on.

Historian Robert Hughes concisely summarized the difficulty of trying to use observation to discover causal relationships. In describing the extensive use of hanging in 18th and 19th century England, he wrote, "One cannot say whether public hanging did terrify people away from crime. Nor can anyone do so, until we can count crimes that were never committed." [3, p.35] To know whether hanging did deter crime, you would need to observe a **counterfactual**, something that didn't actually happen: the crimes in a world without hanging. You can't observe counterfactuals. So you need somehow to generate observations that give you data on what happens for different levels of the causal variable.

A modern idea is the **controlled experiment**. In its simplest ideal form, a controlled experiment involves changing one thing — teacher pay, for example — while *holding everything else constant*: family wealth, facilities, etc.

The experimental approach to gaining knowledge has had great success for example in medicine and science. For many people, experiment is the essence of science. But experiments are hard to perform and sometimes not possible at all. How do you "hold everything else constant?" Partly for this reason, you rarely see reports of experiments when you read the newspaper, unless the article happens to be about a scientific discovery.

Scientists pride themselves on recording their observations carefully and systematically in lab notebooks. Laboratories are filled with high-precision instrumentation. The quest for precision culminates perhaps in the physicist's fundamental quantities: the speed of light is reported to be $299,792,500 \pm 1000$ meters per second, the mass of the electron reported as $9.10938215 \pm 0.00000045 \times 10^{-31}$ kg. Each of these is precise to about 50 parts in a billion. Contrast this extreme precision with the humble speed measurements from a policeman's radar gun (perhaps a couple of miles or kilometers per hour — one part in 50) or the weight indicated on a bathroom scale (give or take a kilogram or a couple of pounds — about one part in 100 for an adult).

All such observations and measures are the stuff of **data**, the records of ob-

servations. Observations do not become data by virtue of high precision or expensive instrumentation or the use of metric rather than traditional units. For many purposes, data of low precision is used. An ecologist's count of the number of mating pairs of birds in a territory is limited by the ability to find nests. A national census of a country's population, conducted by the government can be precise to only a couple of percent. The physicist counting neutrinos in huge observatories buried under mountains to shield them from extraneous events waits for months for her results. These results are precise to only one part in two.

The precision that is needed in data depends on the purpose for which the data will be used. The important question for the person using the data is whether the precision, whatever it be, is adequate for the purpose at hand. To answer this question, you need to know how to measure precision and how to compare this to a standard reflecting the needs of your task. The scientist with expensive instrumentation and the framer of social policy both need to deal with data in similar ways to understand and interpret the precision of their results.

It's common for people to believe that conclusions drawn from data apply to certain areas — science, economics, medicine — but aren't terribly useful in other areas. In teaching, for example, almost all decisions are based on "experience" rather than observation. Indeed, there is often strong resistance to making formal observations of student progress as interfering with the teaching process.

This book is based on the idea that techniques for drawing valid conclusions from observations — data — are valuable for two groups of people. The first group is scientists and others who routinely need to use statistical methods to analyze experimental and other data.

The second group is everybody else. All of us need to draw conclusions from our experiences, even if we're not in a laboratory. It's better to learn how to do this in valid ways, and to understand the limitations of these ways, than to rely on an informal, unstated process of opinion formation. It may turn out that in any particular area of interest there are no useful data. In such situations, you won't be able to use the techniques. But at least you will know what you're missing. You may be inspired to figure out how to supply it or to recognize it when it does come along, and you'll be aware of when others are misusing data.

As you will see, the manner in which the data are collected plays a central role in what sorts of conclusions can be legitimately made; data do not always speak for themselves. You will also see that strongly supported statements about causation are difficult to make. Often, all you can do is point to an "association" or a "correlation," a weaker form of statement.

Statistics is sometimes loosely described as the "science of data." This description is apt, particularly when it covers both the collection and analysis of data, but it does not mean much until you understand what data are. That's the subject of the next chapter.

1.3 The Main Points of this Book

Statistics is about variation. Describing and interpreting variation is a major goal of statistics.

You can create empirical, mathematical descriptions not only of a single trait or variable but also of the relationships between two or more traits. Empirical means based on measurements, data, observations.

Models let you split variation into components: “explained” versus “unexplained.” How to measure the size of these components and how to compare them to one another is a central aspect of statistical methodology. Indeed, this provides a definition of statistics:

Statistics is the explanation of variation in the context of what remains unexplained.

By collecting data in ways that require care but are quite feasible, you can estimate how reliable your descriptions are, e.g., whether it’s plausible that you should see similar relationships if you collected new data. This notion of reliability is very narrow and there are some issues that depend critically on the context in which the data were collected and the correctness of assumptions that you make about how the world works.

Relationships between pairs of traits can be studied in isolation only in special circumstances. In general, to get valid results it is necessary to study entire systems of traits simultaneously. Failure to do so can easily lead to conclusions that are grossly misleading.

Descriptions of relationships are often **subjective** — they depend on choices that you, the modeler, make. These choices are generally rooted in your own beliefs about how the world works, or the theories accepted as plausible within some community of inquiry.

If data are collected properly, you can get an indication of whether the data are consistent or inconsistent with your subjective beliefs or — and this is important — whether you don’t have enough data to tell either way.

Models can also be used to check out the sensitivity of your conclusions to different beliefs. People who disagree in their views of how the world works often may not be able to reconcile their differences based on data, but they will be able to decide objectively whether their own or the other party’s beliefs are reasonable given the data.

Notwithstanding everything said above about the strong link between your prior, subjective beliefs and the conclusions you draw from data, by collecting data in a certain context — experiments — you can dramatically simplify the interpretation of the results. It’s actually possible to remove the dependence on identified subjective beliefs by intervening in the system under study experimentally.

This book takes a different approach than most statistics texts. Many people want statistics to be presented as a kind of automatic, algorithmic way to process data. People look for mathematical certainty in their conclusions. After all, there are right-or-wrong answers to the mathematical calculations that peo-

ple (or computers) perform in statistics. Why shouldn't there be right-or-wrong answers to the conclusions that people draw about the world?

The answer is that there can be, but only when you are dealing with narrow circumstances that may not apply to the situations you want to study. An insistence on certainty and provable correctness often results in irrelevancy.

The point of view taken in this book is that it is better to be useful than to be provably certain. The objective is to introduce methods and ideas that can help you deal with drawing conclusions about the real world from data. The methods and ideas are meant to guide your reasoning; even if the conclusions you draw are not guaranteed by proof to be correct, they can still be more useful than the alternative, which is the conclusions that you draw without data, or the conclusions you draw from simplistic methods that don't honor the complexity of the real system.

1.4 Introduction to Computation with R

Modern statistics is done on the computer. There was a time, 60 years ago and before, when computation could only be done by hand or using balky mechanical calculators. The methods of applied statistics developed during this time reflected what could be done using such calculators, not necessarily what was best for illuminating the system under study. These methods took on a life of their own — they became the operational definition of statistics. They continue to be taught today, using electronic calculators or personal computers or even just using paper and pencil. For the old statistical methods, computers are merely a labor saving device.

But not for modern statistics. The statistical methods at the core of this book cannot be applied in an authentic and realistic way without powerful computers. Thirty years ago, many of the methods could not be done at all unless you had access to the resources of a government agency or a large university. But with the revolutionary advances in computer hardware and numerical algorithms over the last half-century, modern statistical calculations can be performed on an ordinary home computer or laptop. (Even a cell phone has phenomenal computational power, often besting the mainframes of thirty years ago.) Hardware and software today pose no limitation; they are readily available.

Each chapter of this book includes a section on computational technique. Many readers will be completely new to the use of computers for scientific and statistical work, so the first chapters cover the foundations, techniques that are useful for many different aspects of computation. Working through the early chapters is essential for developing the skills that will be used later in actual statistical work. It will take a few hours, but this investment will pay off handsomely.

Chances are, you use a computer almost every day: for email, word-processing, managing your music or your photograph collection, perhaps even using a spreadsheet program for accounting. The software you use for such activities makes

it easy to get started. Possibly you have never even looked at an instruction manual or used the “help” features on your computer.

When you use a word processor or email, the bulk of what you enter into the computer — the content of your documents and email — is without meaning to the computer. This is not at all to say that it is meaningless. Your documents and letters are intended for human readers; most of the work you do is directed so that the recipients can understand them. But the computer doesn’t need to understand what you write in order to format it, print it, or transmit it over the Internet. Indeed, the computer would be equally effective at handling random text generated by typing monkeys.

When doing scientific and statistical computing, things are different. What you enter into the computer is instructions to the computer to perform calculations and re-arrangements of data. Those instructions have to be comprehensible to the computer. If they make no sense or if they are inconsistent or ill formed, the computer won’t be able to carry out your instructions. Worse, if the instructions make sense in some formal way but don’t convey your actual intentions, the computer will perform some operation but the result will mislead you.

The difficulty with using software for mathematics and statistics is in making sure that your instructions make sense and do what you want them to do. This difficulty is not a matter of bad software design; it’s intrinsic to the problem of communicating your intentions to the computer. The same difficulty would arise in word processing if the computer had to make sense of your writing, rejecting it when a claim is unconvincing or when a sentence is ambiguous. Statistical computing pioneer John Chambers refers to the “Prime Directive” of software[4]: “to program in such a way that computations can be understood and trusted.”

Much of the design of packages for scientific and statistical work is oriented around the difficulty of communicating intentions. A popular approach is based on the computer mouse: the program provides a list of possible operations — like the keys on a calculator — and lets the user choose which operation to apply to some selected data. This style of user interface is employed, for example, in spreadsheet software, letting users add up columns of numbers, make graphs, etc. The reason this style is popular is that it can make things extremely easy ... so long as the operation that you want has been included in the software. But things get very hard if you need to construct your own operation and it can be difficult to understand or trust the operations performed by others.

Another style of scientific computation — the one used in this book — is based on language. Rather than selecting an option with a mouse, you construct a **command** that conveys both the operation that you want and the data to which you want to apply that operation. There are dramatic advantages to this language-based style of computation:

- It lets you **connect** computations to one another, so that the output of one operation can become the input to another.
- It lets you **repeat** the operation on new or modified data, allowing you to

automate tedious tasks and, importantly, to verify the correctness of your computations on data where you already know the answer.

- It lets you **accumulate** the results of previous operations, treating those results as new data.
- It lets you **document** concisely what was done so that you can demonstrate that what you said you did is what you actually did. In this way, you or others can repeat the analysis later if necessary to confirm your results.
- It lets you **modify** the computation in a controlled way to correct it or to vary some aspect of it while holding other aspects exactly the same as in the original.

In order to use the language-based approach, you will need to learn a few principles of the language itself: some vocabulary, some syntax, some grammar. This is much, much easier for the computer language than for a natural language like English or Chinese; it will take you only a couple of hours before you are fluent enough to do useful computations. In addition to letting you perform scientific computations in ways that use the computer and your own time and effort effectively, the principles that you will learn are broadly applicable to many computer systems and can provide significant insight even to how to use mouse-based interfaces.

1.4.1 The R Environment

The software package used in this book is called R. The R package provides an environment for doing statistical and scientific computation at a professional level. It was designed for statistics work, but suitable for other forms of scientific calculations and the creation of high-quality scientific graphics.[5]

There are several other major software packages widely used in statistics. Among the leaders are SPSS, SAS, and STATA. Each of them provides the computational power needed for statistical modeling. Each has its own advantages and its own devoted group of users.

One reason for the choice of R is that it offers a command-based computing environment. That makes it much easier to write about computing and also reveals better the structure of the computing process.[6] Also nice is that R is available for free and works on the major types of computers, e.g., Windows, Macintosh, and Unix/Linux. You can get information about how to install R on your computer at www.r-project.org.

In making your own choice, the most important thing is this: *choose something!* Readers who are familiar with SPSS, SAS, or STATA can use the information in each chapter's computational technique section to help them identify the facilities to look for in those packages.

Another package that's often used in statistical work is the spreadsheet program Excel. This package has its own advantages. It's effective for entering data and has nice facilities for formatting tables. The visual layout of the data seems to be intuitive to many people. Many businesses use Excel and it's widely taught

in high schools. Unfortunately, it's very difficult to use for statistical analyses of any sophistication. Indeed, even some very elementary tasks such as making a histogram are difficult to do in Excel and the results are usually unsatisfactory from a graphical point of view. Worse, Excel is very hard to use reliably. There are lots of opportunities to make mistakes that will go undetected; Excel encourages bad programming practices that make software unreliable.

1.4.2 Setting R Up

You can skip this section if you want to jump ahead to read about how R works.

Better, though, if you try out the commands as they are shown, using the R software on your own computer.

To do this, you will need to start the R software. If R is not already installed on your computer (this is likely if you are using your own computer and have never used R before), the first step is to install it.

If you are used to installing software, you will find R follows the usual pattern.

1. Use a web browser to go to `www.r-project.org`. Select the Download/CRAN link on the left of the page. This will bring you to a list of download sites.
2. Choose one in your own region of the world. This will bring up a page with a choice of Linux, Mac OS X, or Windows. Choose whichever is appropriate for your computer.
3. For Windows: choose the link for the "base" distribution, and then download the link that looks like "R-2.9.0-win32.exe." The name may be slightly different, depending on what new versions have been released. Run this program, accepting the defaults.

For Macintosh, follow the link that looks like "R-2.9.0.dmg" and follow the instructions. Again, the name may be slightly different, depending on what new versions have been released.

If you are using Linux, you probably don't need any instructions on how to install software.

Once R is installed, you start it like most programs, by clicking on an icon. On a Windows computer, this will be under the "start" button, on a Macintosh this will be in the applications folder.

When you start R, a new window appears on your screen. It will look something like Figure 1.1. You're ready to go!

Actually, there is one more thing that you can do that will make things easier later on, when you start to analyze the data sets that go along with this book. Download one file from a web site and put it in some convenient place on your computer, for instance your "desktop" or a folder that you make for this book. The file is at

`www.macalester.edu/~kaplan/ISM/ISM.Rdata`

This file contains various data sets that you will be using.

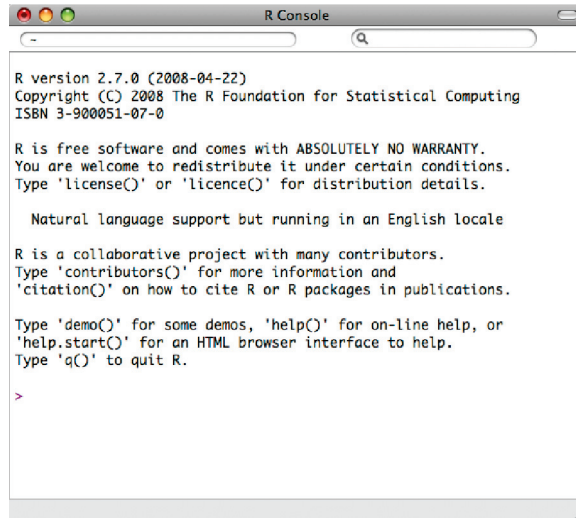


Figure 1.1: The R command window.

Once you have downloaded the file, called `ISM.Rdata`, double-clicking on it in the ordinary way will start R or load the data into an already started session of R.

1.4.3 Invoking an Operation

People often think of computers as *doing* things: sending email, playing music, storing files. Your job in using a computer is to tell the computer *what* to do. There are many different words used to refer to the “what”: a procedure, a task, a function, a routine, and so on. I’ll use the word **computation**. Admittedly, this is a bit circular, but it is easy to remember: computers perform computations.

Complex computations are built up from simpler computations. This may seem obvious, but it is a powerful idea. An **algorithm** is just a description of a computation in terms of other computations that you already know how to perform. To help distinguish between the computation as a whole and the simpler parts, it is helpful to introduce a new word: an **operator** performs a computation.

It’s helpful to think of the computation carried out by an operator as involving four parts:

1. The name of the operator
2. The input arguments
3. The output value
4. Side effects

A typical operation takes one or more **input arguments** and uses the information in these to produce an **output value**. Along the way, the computer might

take some action: display a graph, store a file, make a sound, etc. These actions are called **side effects**.


To tell the computer to perform an computation — call this **invoking an operation** or giving a **command** — you need to provide the name and the input arguments in a specific format. The computer then returns the output value. For example, the command `sqrt(25)` invokes the square root operator (named `sqrt`) on the argument 25. The output from the computation will, of course, be 5.

The syntax for invoking an operation consists of the operator's name, followed by round parentheses. The input arguments go inside the parentheses.

The software program that you use to invoke operators is called an **interpreter**. (The interpreter is the program you are running when you start R.) You enter your commands as a dialog between you and the interpreter. To start, the interpreter prints a prompt, after which you type your command:

PROMPT →  `sqrt(25)` ← COMMAND

When you press “Enter,” the interpreter reads your command and performs the computation. For commands such as this one, the interpreter will print the output value from the computation:

 `sqrt(25)`

OUTPUT MARKER → `[1]` 5 ← OUTPUT VALUE

NEXT PROMPT → 

The dialog continues as the interpreter prints another prompt and waits for your further command.

To save space, I'll usually show just the give-and-take from one round of the dialog:

```
> sqrt(25)
[1] 5
```

(Go ahead! Type `sqrt(25)` after the prompt in the R interpreter, press “enter,” and see what happens.)

Often, operations involve more than one argument. The various arguments are separated by commas. For example, here is an operation named `seq` that produces a sequence of numbers:

```
> seq(3,10)
[1] 3 4 5 6 7 8 9 10
```

The first argument tells where to start the sequence, the second tells where to end it.

The order of the arguments is important. Here is the sequence produced when 10 is the first argument and 3 the second:

```
> seq(10,3)
[1] 10 9 8 7 6 5 4 3
```

For some operators, particularly those that have many input arguments, some of the arguments can be referred to by name rather than position. This is particularly useful when the named argument has a sensible default value. For example, the `seq` operator can be instructed how big a jump to take between successive items in the sequence. This is accomplished using an argument named `by`:

```
> seq(3,10,by=2)
[1] 3 5 7 9
```

Depending on the circumstances, all four parts of a operation need not be present. For example, the `date` operation returns the current time and date; no input arguments are needed.

```
> date()
[1] "Wed Apr 16 06:18:06 2008"
```

Note that even though there are no arguments, the parentheses are still used. Think of the pair of parentheses as meaning, “Do this.”

Naming and Storing Values

Often the value returned by an operation will be used later on. Values can be stored for later use with the **assignment operator**. This has a different syntax that reminds the user that a value is being stored. Here’s an example of a simple assignment:

```
> x = 16
```

This command has stored the value 16 under the name `x`. The syntax is always the same: an equal sign (=) with a name on the left and a value on the right.

Such stored values are called **objects**. Making an assignment to an object defines the object. Once an object has been defined, it can be referred to and used in later computations.

Notice that an assignment operation does not return a value or display a value. Its sole purpose is to have the side effects of defining the object and thereby storing a value under the object’s name.

To refer to the value stored in the object, just use the object’s name itself. For instance:

```
> x
[1] 16
```

Doing a computation on the value stored in an object is much the same:

```
> sqrt(x)
[1] 4
```

You can create as many objects as you like and give them names that remind you of their purpose. Some examples: `wilma`, `ages`, `temp`, `dog.houses`, `foo3`. There are some rules for object names:

- Use only letters and numbers and the two punctuation marks “dot” (.) and “underscore” (_).
- Do NOT use spaces anywhere in the name.
- A number or underscore cannot be the first character in the name.
- Capital letters are treated as distinct from lower-case letters. The objects named `wilma` and `Wilma` are different.

For the sake of readability, keep object names short. But if you really must have an object named something like `agesOfChildrenFromTheClinicalTrial`, feel free.

Objects can store all sorts of things, for example a sequence of numbers:

```
> x = seq(1,7)
```

When you assign a new value to an existing object, as just done to `x`, the former value of that object is erased from the computer memory. The former value of `x` was 16, but after the above assignment command it is

```
> x
[1] 1 2 3 4 5 6 7
```

The value of an object is changed only *via* the assignment operator. Using an object in a computation does not change the value. For example, suppose you invoke the square-root operator on `x`:

```
> sqrt(x)
[1] 1.00 1.41 1.73 2.00 2.24 2.45 2.65
```

The square roots have been returned as a value, but this doesn’t change the value of `x`:

```
> x
[1] 1 2 3 4 5 6 7
```

If you want to change the value of `x`, you need to use the assignment operator:

```
> x = sqrt(x)
> x
[1] 1.00 1.41 1.73 2.00 2.24 2.45 2.65
```

Connecting Computations

The brilliant thing about organizing operators in terms of input arguments and output values is that the output of one operator can be used as an input to another. This lets complicated computations be built out of simpler ones.

For example, suppose you have a list of 10000 voters in a precinct and you want to select a random sample of 20 of them for a survey. The `seq` operator can

Aside. 1.1 Assignment vs Algebra

An assignment command like `x=sqrt(x)` can be confusing to people who are used to algebraic notation. In algebra, the equal sign describes a relationship between the left and right sides. So, $x = \sqrt{x}$ tells us about how the quantity x and the quantity \sqrt{x} are related. Students are usually trained to “solve” such relationships, going through a series of algebraic steps to find values for x that are consistent with the mathematical statement. (For $x = \sqrt{x}$, the solutions are $x = 0$ and $x = 1$.) In contrast, the assignment command `x=sqrt(x)` is a way of replacing the previous values stored in `x` with new values that are the square root of the old ones.

be used to generate a set of 10000 choices. The `sample` operator can be used to select some of these choices at random.

One way to connect the computations is by using objects to store the intermediate outputs.

```
> choices = seq(1,10000)
> sample( choices, 20 )
[1] 5970 8476 9340 8266 6909
[6] 3692 8979 1640 4266 5580
[11] 1208 6141 4973 5575 8498
[16] 1001  923 3246 4194 2126
```

You can also pass the output of an operator *directly* as an argument to another operator. Here’s another way to accomplish exactly the same thing as the above.

```
> sample( seq(1,10000), 20 )
```

Numbers and Arithmetic

The language has a concise notation for arithmetic that looks very much like the traditional one:

```
> 7+2
[1] 9
> 3*4
[1] 12
> 5/2
[1] 2.5
> 3-8
[1] -5
> -3
[1] -3
> 5^2
[1] 25
```

Arithmetic operators, like any other operators, can be connected to form more complicated computations. For instance,

```
> 8+4/2
[1] 10
```

To a human reader, the command $8+4/2$ might seem ambiguous. Is it intended to be $(8+4)/2$ or $8+(4/2)$? The computer uses unambiguous rules to interpret the expression, but it's a good idea for you to use parenthesis so that you can make sure that what you intend is what the computer carries out:

```
> (8+4)/2
[1] 6
```

Traditional mathematical notation uses superscripts and radicals to indicate exponentials and roots, e.g., 3^2 or $\sqrt{3}$ or $\sqrt[3]{8}$. This special typography doesn't work well with an ordinary keyboard, so R and most other computer languages uses a different notation:

```
> 3^2
[1] 9
> sqrt(3)
[1] 1.73
> 8^(1/3)
[1] 2
```

There is a large set of mathematical functions: exponentials, logs, trigonometric and inverse trigonometric functions: Some examples:

Traditional	Computer
e^2	<code>exp(2)</code>
$\log_e(100)$	<code>log(100)</code>
$\log_{10}(100)$	<code>log10(100)</code>
$\log_2(100)$	<code>log2(100)</code>
$\cos(\frac{\pi}{2})$	<code>cos(pi/2)</code>
$\sin(\frac{\pi}{2})$	<code>sin(pi/2)</code>
$\tan(\frac{\pi}{2})$	<code>tan(pi/2)</code>
$\cos^{-1}(-1)$	<code>acos(-1)</code>

Numbers can be written in **scientific notation**. For example, the “universal gravitational constant” that describes the gravitational attraction between masses is 6.67428×10^{-11} (with units meters-cubed per kilogram per second squared). In the computer notation, this would be written `G=6.67428e-11`. The Avogadro constant, which gives the number of atoms in a mole, is $6.02214179 \times 10^{23}$ per mole, or `6.02214178e23`.

The computer language does not directly support the recording of units. This is unfortunate, since in the real world numbers often have units and the units matter. For example, in 1999 the Mars Climate Orbiter crashed into Mars because the design engineers specified the engine's thrust in units of pounds, while the guidance engineers thought the units were newtons.

Computer arithmetic is accurate and reliable, but it often involves very slight rounding of numbers. Ordinarily, this is not noticeable. However, it can become

apparent in some calculations that produce results that are zero. For example, mathematically $\sin(\pi) = 0$, however the computer does not duplicate this mathematical relationship exactly:

```
> sin(pi)
[1] 1.22e-16
```

Whether a number like this is properly interpreted as “close to zero,” depends on the context and, for quantities that have units, on the units themselves. For instance, the unit “parsec” is used in astronomy in reporting distances between stars. The closest star to the sun is Proxima, at a distance of 1.3 parsecs. A distance of 1.22×10^{-16} parsecs is tiny in astronomy but translates to about 2.5 meters — not so small on the human scale.

In statistics, many calculations relate to probabilities which are always in the range 0 to 1. On this scale, 1.22e-16 is very close to zero.

There are two “special” numbers. Inf stands for ∞ , as in

```
> 1/0
[1] Inf
```

NaN stands for “not a number,” and is the result when a numerical operation isn’t defined, for instance

```
> 0/0
[1] NaN
> sqrt(-9)
[1] NaN
```

Aside. 1.2 Complex Numbers

Mathematically oriented readers will wonder why R should have any trouble with a computation like $\sqrt{-9}$; the result is the imaginary number $3i$. R works with complex numbers, but you have to tell the system that this is what you want to do. To calculate $\sqrt{-9}$, use `sqrt(-9+0i)`.

Types of Objects

Most of the examples used so far have dealt with numbers. But computers work with other kinds of information as well: text, photographs, sounds, sets of data, and so on. The word **type** is used to refer to the kind of information.

Modern computer languages support a great variety of types. There are four types that will be most important here:

numeric The numbers of the sort already encountered.

character Text data.

logical Answers to yes/no questions.

data frames Collections of data more or less in the form of a spreadsheet table.

It's important to know about the types of data because operators expect their input arguments to be of specific types. When you use the wrong type of input, the computer might not be able process your command.

Character Data

You indicate character data to the computer by enclosing the text in double quotation marks. For example:

```
> filename = "swimmers.csv"
```

There is something a bit subtle going on in the above command, so look at it carefully. The purpose of the command is to create an object, named `filename`, that stores a little bit of text data. Notice that the name of the object is not put in quotes, but the text characters are.

Whenever you refer to an object name, make sure that you don't use quotes, for example:

```
> filename  
[1] "swimmers.csv"
```

If you make a command with the object name in quotes, it won't be treated as referring to an object. Instead, it will merely mean the text itself:

```
> "filename"  
[1] "filename"
```

Similarly, if you omit the quotation marks from around text, the computer will treat it as if it were an object name and will look for the object of that name. For instance, the following command directs the computer to look up the value contained in an object named `swimmers.csv` and insert that value into the object `filename`.

```
> filename = swimmers.csv  
Error: object "swimmers.csv" not found
```

As it happens, there was no object named `swimmers.csv` because it had not been defined by any previous assignment command. So, the computer generated an error.

For the most part, you will not need to use very many operators on text data; you just need to remember to include text, such as file names, in quotation marks. Sometimes, you will want to convert non-text items to text in order to display them in graphs. There is a special operator, `as.character` for doing this:

```
> as.character(3)  
[1] "3"
```


The quotes in the output show that it is character type rather than numeric type. This isn't terribly important to the human reader, but the computer regards "3" as a different thing than 3. For instance, you can do arithmetic on numbers but not on characters.

```
> 3 + 2
[1] 5
> as.character(3) + 2
Error in as.character(3) + 2 :
  non-numeric argument to binary operator
```

Data Frames

A data frame is a collection of values arranged as a table. For example, here is part of a data frame that records an experiment on the uptake of carbon dioxide by the grass species *Echinochloa crus-galli*.

Plant	Type	Treatment	conc	uptake
Mc1	Mississippi	chilled	350	18.9
Mc3	Mississippi	chilled	250	17.9
Mn3	Mississippi	nonchilled	95	11.3
Mn3	Mississippi	nonchilled	350	27.9
Qn1	Quebec	nonchilled	500	35.3
Qc2	Quebec	chilled	500	38.6
Qc3	Quebec	chilled	175	21.0
Qn3	Quebec	nonchilled	500	42.9
Qn3	Quebec	nonchilled	175	32.4

... and so on for 84 lines altogether

A data frame is a kind of tabular organization of data. In this example, it records several variables: the geographic origin of the plant (Mississippi or Quebec), and whether the plant had been chilled overnight before the uptake measurement was made, the ambient atmospheric CO₂ level and the uptake of CO₂ by the plant.

Each of the components of the data frame could be stored by an object of character type or of numerical type, for instance, *Treatment* is character and *conc* is numerical. The data frame brings the various components together in one place, facilitating processing and analysis of the data.

The information for a data frame is often stored in a spreadsheet file and read into R for analysis. Until you learn how to read in such files, you can use some of the built-in data frames intended for example purposes. If you want to follow along the examples with the CO₂ data frame, use this command to create an object named *C02* that contains the data frame:

```
> data(C02)
```

Columns of Data Frames

Perhaps the most common operation on a data frame is to refer to the values in a single column. This can be done using a special syntax involving the \$ sign. To refer to the `conc` column in the `C02` data frame, you would use the command `C02$conc`. To refer to the `Treatment` column, use `C02$Treatment`. Think of this style of reference as analogous to naming a person with a first name and a last name: the name of the data frame object comes first and the variable name second, separated by the \$, something like `Einstein$Albert`.

Each component is just like an ordinary object and can be used in any way you would use an object:

```
> length(C02$conc)
[1] 84
> mean(C02$conc)
[1] 435
> max(C02$uptake)
[1] 45.5
> table( C02$Treatment)
nonchilled    chilled
          42         42
```

Logical Data and Logical Operators

Many computations involve selections of subsets of data that meet some criterion. For example, in studying the health of newborn babies, you might want to focus only on those below a certain birthweight or perhaps those babies whose mother smoked during pregnancy. The question of whether the case satisfies the criterion boils down to a yes-or-no answer.

Logic is the study of valid inference; it is intimately tied up with the idea of truth versus falsehood. In computer languages, **logical data** refers to a type of data that can represent whether something is true or false. To illustrate, consider the simple sequence 1 to 10

```
> x = seq(1,7)
```

Now a simple question about the values in `x`: Are any of them less than π ? Here's how you can ask that question:

```
> x < pi
[1] FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE
```

A computer command like `x<pi` is not the same as an algebraic statement like $x < \pi$. The algebraic statement describes the relationship between x and π , namely that x is less than π . The computer command asks a question: Is the value of `x` less than the value of `pi`? Asking this question invokes a computation; the returned value is the answer to the question, either `TRUE` or `FALSE`.

Here are some of the operators for asking such questions:

<code>x<y</code>	Is x less than y?
<code>x<=y</code>	Is x less than or equal to y?
<code>x>y</code>	Is x greater than y?
<code>x>=y</code>	Is x greater than or equal to y?
<code>x==y</code>	Is x equal to y?
<code>x!=y</code>	Is x unequal to y?

Notice the double equal signs in `x==y`. A single equal sign would be the assignment operator.

Mostly, these comparison operators apply to numbers. The `==` and `!=` operators also apply to character strings. To illustrate, I'll define two objects `v` and `w`:

```
> v = "hello"
> w = seq(1,15,by=3)
> w
 1  4  7 10 13
> w < 12
[1] TRUE TRUE TRUE TRUE FALSE
> w > 7
[1] FALSE FALSE FALSE TRUE TRUE
> w != 4
[1] TRUE FALSE TRUE TRUE TRUE
> v == "goodbye"
[1] FALSE
> v != "hi"
[1] TRUE
> v == "hello"
[1] TRUE
> v == "Hello"
[1] FALSE
```

The FALSE in the last line results from taking into account the difference between upper-case and lower-case letters.

Sometimes you need to combine more than one logical result. For example, to ask, "Is `w` between 7 and 12?" involves combining two separate questions: "Is `w` greater than 7 AND is it less than 12?" In the computer language, this question would be stated `w>7&w<12`.

```
> w > 7 & w < 12
[1] FALSE FALSE FALSE TRUE FALSE
```

There are also logical operators for "or" and "not". For instance, you might ask whether `w` is less than 5 OR greater than 9:

```
> w < 5 | w > 9
[1] TRUE TRUE FALSE TRUE TRUE
```

The "not" operator just flips TRUE and FALSE:

```
> !(w < 5 | w > 9)
[1] FALSE FALSE TRUE FALSE FALSE
```

As this example illustrates, you can group logical operations in just the same way as arithmetic operations.

Missing Data

When recording data from an experiment or an observational study, it sometimes happens that a particular measurement can't be made or is lost or is otherwise unavailable. In R, such **missing data** can be recorded with the special code `NA`. As you might expect, arithmetic and other operations on missing data can't be sensibly performed: giving `NA` as an input produces `NA` as an output.

```
> 7 == NA
[1] NA
> NA == 'Hello'
[1] NA
> NA == NA
[1] NA
```

In order to test whether there is missing data, a special operator `is.na` can be used:

```
> is.na(NA)
[1] TRUE
```

Collections

R can work with collections of numbers and character strings. Some operators work on each item in the collection, while others combine the items together in some way. To illustrate, I'll define three small collections, `x`, `y`, and `fruits`:

```
> x = seq(1,7)
> x
[1] 1 2 3 4 5 6 7

> y = c(7,8,9)
> y
[1] 7 8 9

> fruits = c("apple","berry","cherry")
> fruits
[1] "apple" "berry" "cherry"
```

The `c` operator used in defining `y` and `fruits` is useful for creating a small collection “by hand.” Often, however, collections will be created by reading in data from a file or using some other operator. I'll introduce these as needed for specific tasks.

Arithmetic and comparison operators often work item-by-item on the collection. For example:

```
> x + 100
[1] 101 102 103 104 105 106 107
> sqrt(y)
[1] 2.645751 2.828427 3.000000
> fruits == "cherry"
[1] FALSE FALSE TRUE
```

If the operator involves two collections, they have to be the same size, or R will reuse the smaller collection to match the size of the larger one:

```
> x == y
[1] FALSE FALSE FALSE FALSE FALSE FALSE TRUE
Warning message:
In x == y : longer object length is not a
multiple of shorter object length
```

The warning message is displayed when some aspect of the computation is deemed suspect or odd. Pay attention to such messages since they may signal that the computation the interpreter carried out is not the one you intended.

It's usually obvious what sorts of operators will combine the items of the collection rather than working on them item by item. Here are some examples:

```
> x
[1] 1 2 3 4 5 6 7
> y
[1] 7 8 9
> mean(x)
[1] 4
> median(y)
[1] 8
> min(x)
[1] 1
> max(x)
[1] 7
> sum(x)
[1] 28
> any( fruits == "cherry")
[1] TRUE
> all( fruits == "cherry")
[1] FALSE
```

The length operator tells how many items there are in the collection:

```
> length(x)
[1] 7
> length(y)
```

```
[1] 3
> length(fruits)
[1] 3
```

When there are too many items in a collection to display conveniently in one line, the R interpreter will break up the display over multiple lines.

```
> seq(3, 19)
[1] 3 4 5 6 7 8 9 10 11
[10] 12 13 14 15 16 17 18 19
```

At the start of each line, the number in brackets tells the index of the item that starts that line. In the above, for instance, the item 3 is displayed following a [1] because 3 is the first item in the collection. Similarly, the [10] indicates that the item 12 is the tenth item in the collection. The brackets are just for display purposes; they are not part of the collection itself.

Defining your own operators

Occasionally, you may need to define your own operators. This is convenient if you need to repeat an operation many times or if you need to define a mathematical function.

It's important to keep in mind the difference between an operator and a command. A command is an instruction to perform a particular computation on a particular input argument or set of input arguments. The input arguments always have to be values, though of course you can refer to the value by giving the name of an object that has already been assigned a value.

In contrast, in defining an operator, you can treat the arguments abstractly; just a name without a value having been assigned. To illustrate, here is a command that creates the mathematical function $f(x) = 3x^2 + 2$ and stores it in an operator named `f`:

```
> f = function(x) { 3*x^2 + 2 }
```

Once you have defined the function, you can invoke it in the standard way. For example:

```
> f(3)
[1] 29
> f(10)
[1] 302
```

There are some novel features to the syntax used to define a new operator. First, the arguments to `function` aren't treated as values but as pure names. Second, the contents of the curly braces `{` and `}` — the function contents — are the commands that will be evaluated when the function is invoked.

It doesn't matter what names you use in the function contents so long as they match the names used in the arguments to `function`. For example, here is another operator, called `g`, that will perform exactly the same computation as `f` when invoked:

```
> g = function(marge) { 3*marge^2 + 2 }
```

When you invoke an operator, the interpreter carries out several steps. Consider the invocation

```
> g(7)
```

In carrying out this command, the interpreter will:

1. Temporarily define or redefine an object `marge` that has the value 7.
2. Execute the function contents.
3. Return the value of these contents as the return value of the command.
4. Discard the definition or redefinition in (1).

Operators can have more than one argument. For instance, here is an operator `hypotenuse` that computes the length of the hypotenuse of a right triangle given the lengths of the legs

```
> hypotenuse = function(a,b) { sqrt( a^2 + b^2 ) }
```

When programmers create new operators that they expect to use on many different occasions, they put the commands to define the operators into a text file called a **source file**. This file can be read into R using a special operator, called `source` that causes the commands to be executed, thereby defining the new operators.

Saving and Documenting Your Work

Up to now, the commands used as examples have been simple one-line statements. As you work further, you will build more elaborate computations by combining simpler ones. It will become important to be able to document what you have done, providing a record so that others can confirm your results and so that you and others can modify your work as needed.

One way in which a record is created of your interaction with the computer is the dialog in the interpreter console itself. In some ways this is analogous to a document created by a word processor: for example, you can copy the contents and paste it into another document.

But the idea of dialog-as-document is flawed. For example, in a word-processor, when you correct a mistake the old version is erased. But in the R dialog, to fix a mistake you give a new command — the old, mistaken command is still there in the dialog.

You should keep in mind that there are several different components, some of which are more appropriate than others for your documentation.

Your Commands The commands that you execute are what defines the computation being performed. These commands themselves are a valuable form of documentation.

The objects you create These objects, and the values that are stored in them, reflect the **state** of the computation. If you want to pick up on your work where you left off, you can save these objects. This is called “saving the **workspace**.”

Side effects This refers to the output printed by the interpreter and plots. Sometimes you will want to include this in your documentation, but usually just select elements.

1.4.4 Using Customizations to R

One of the features that makes R so powerful is that new commands can easily be added to the system. This makes it possible, for instance, to customize the software to make routine tasks easier. Such customizations have been written specifically for the people following this book. To use them — and you will need them in later chapters — you should download a file from the web:

`www.macalester.edu/~kaplan/ISM/ISM.Rdata`

This file contains various data sets that you be using.

Once you have downloaded the file, called `ISM.Rdata`, double-click on it in the ordinary way to start R or to load the data into an already started session of R.

Chapter 2

Data: Cases, Variables, Samples

The tendency of the casual mind is to pick out or stumble upon a sample which supports or defies its prejudices, and then to make it the representative of a whole class. — Walter Lippmann (1889-1974)

The word “data” is plural. This is appropriate. Statistics is, at its heart, about variability: how things differ from one another.

Figure 2.1 shows a small collection of sea shells collected during an idle quarter hour sitting at one spot on the beach on Malololailai island in Fiji. All the

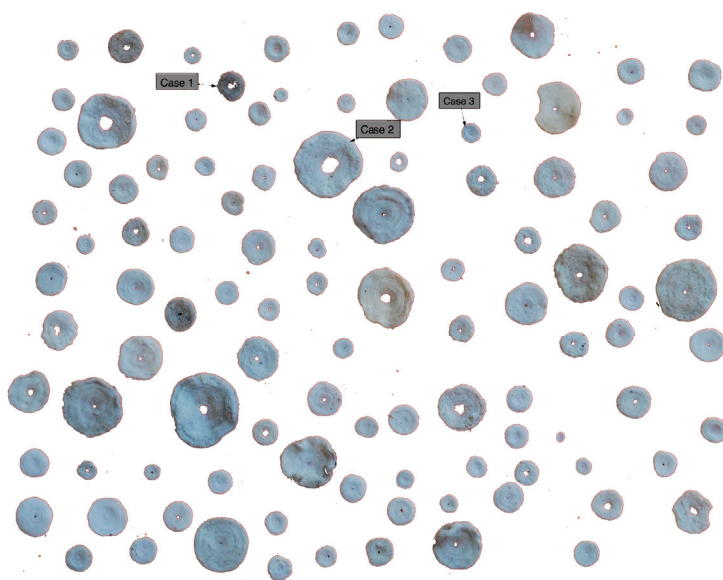


Figure 2.1: A collection of 103 sea shells. (The photo is printed to $3/4$ scale.)

shells in the collection are similar; the collector picked up only the small disk-shaped shells with a hole in the center. But the shells also differ from one another: in overall size and weight, in color, in smoothness, in the size of the hole, etc.

Any data set is something like the shell collection. It consists of **cases**: the objects in the collection. Each case has one or more attributes or qualities, called **variables**. This word “variable” emphasizes that it is differences or variation that is often of primary interest.

Usually, there are many possible variables. The researcher chooses those that are of interest, often drawing on detailed knowledge of the system that is under study. The researcher measures or observes the value of each variable for each case. The result is a **table**, also known as a **data frame**: a sort of spreadsheet. Within the data frame, each row refers to one case, each column to one variable.

A data frame for the sea shell collection might look like this:

Case	diameter	weight	color	hole
1	4.3 mm	0.010 mg	dark	medium
2	12.0 mm	0.050 mg	light	very large
3	3.8 mm	0.005 mg	light	none
and so on — there are 103 cases altogether				

Each individual shell has a case number that identifies it; three of these are shown in the figure. These case numbers are arbitrary; the position of each case in the table — first, second, tenth, and so on — is of no significance. The point of assigning an identifying case number is just to make it easier to refer to individual cases later on.

If the data frame had been a collection of people rather than shells, the person’s name or another identifying label could be used to identify the case.

There are many sorts of data that are recorded in different formats. For example, photographic images are stored as arrays of pixels without any reference to cases and variables. But the data frame format is very general and can be used to arrange all sorts of data, even if it is not the format always used in practice. Even data that seems at first not to be suitable to arrange as a data frame is often stored this way, for example the title/artist/genre/album organization found in music MP3 players or the geographic features (place locations and names, boundaries, rivers, etc.) found in geographic information systems (GIS).

2.1 Kinds of Variables

Most people tend to think of data as numeric, but variables can also be descriptions, as the sea shell collection illustrates. The two basic types of data are:

Quantitative: Naturally represented by a number, for instance diameter, weight, temperature, age, and so on.

Categorical: A description that can be put simply into words or categories, for instance male versus female or red vs green vs yellow, and so on. The

value for each case is selected from a fixed set of possibilities. This set of possibilities are the **levels** of the categorical variable.

Categorical variables show up in many different guises. For example, a data frame holding information about courses at a college might have a variable subject with levels biology, chemistry, dance, economics, and so on. The variable semester could have levels Fall2008, Spring2009, Fall2009, and so on. Even the instructor is a categorical variable. The instructor's name or ID number could be used as the level of the variable.

Quantitative variables are numerical but they often have units attached to them. In the shell data frame, the variable diameter has been recorded in millimeters, while weight is given in milligrams. The usual practice is to treat quantitative variables as a pure number, without units being given explicitly. The information about the units — for instance that diameter is specified in millimeters — is kept in a separate place called a **code book**.

The code book contains a short description of each variable. For instance, the code book for the shell data might look like this:

Code book for shells from Malololailai island, collected on January 12, 2008.

diameter: the diameter of the disk in millimeters.

weight: the shell weight in milligrams.

color: a subjective description of the color. Levels: light, medium, and dark.

hole: the size of the inner hole. Levels: none, small, large, very large.

On the computer, a data frame is usually stored as a spreadsheet file, while the corresponding code book can be stored separately as a text file.

Sometimes quantitative information is represented by categories as in the hole variable for the sea shells. For instance, a data frame holding a variable income might naturally be stored as a number in, say, dollars per year. Alternatively, the variable might be treated categorically with levels of, say, "less than \$10,000 per year," "between \$10,000 and \$20,000," and so on. Almost always, the genuinely quantitative form is to be preferred. It conveys more information even if it not correct to the last digit.

The distinction between quantitative and categorical data is essential, but there are other distinctions that can be helpful even if they are less important.

Some categorical variables have levels that have a *natural* order. For example, a categorical variable for temperature might have levels such as "cold," "warm," "hot," "very hot." Variables like this are called **ordinal**. Opinion surveys often ask for a choice from an ordered set such as this: strongly disagree, disagree, no opinion, agree, strongly agree.

For the most part, this book will treat ordinal variables like any other form of categorical variable. But it's worthwhile to pay attention to the natural ordering of an ordinal variable. Indeed, sometimes it can be appropriate to treat an ordinal variable as if it were quantitative.

2.2 Data Frames and the Unit of Analysis

When collecting and organizing data, it's important to be clear about what is a case. For the sea shells, this is pretty obvious; each individual shell is an individual case. But in many situations, it's not so clear.

A key idea is the **unit of analysis**. Suppose, for instance, that you want to study the link between teacher pay, class size, and the performance of school children. There are all sorts of possibilities for how to analyze the data you collect. You might decide to compare different schools, looking at the average class size, average teacher pay, and average student performance in each school. Here, the unit of analysis is the school.

Or perhaps rather than averaging over all the classes in one school, you want to compare different classes, looking at the performance of the students in each class separately. The unit of analysis here is the class.

You might even decide to look at the performance of individual students, investigating how their performance is linked to the individual student's family income or the education of the student's parents. Perhaps even include the salary of student's teacher, the size of the student's class, and so on. The unit of analysis here is the individual student.

What's the difference between a unit of analysis and a case? A case is a row in a data frame. In many studies, you need to bring together different data frames, each of which may have a different notion of case. Returning to the teacher's pay example, one can easily imagine at least three different data frames being involved, with each frame storing data at a different level:

1. A frame with each class being a case and variables such as the size of the class, the school in which the class is taught, etc.
2. A frame with each teacher being a case and variables such as the teacher's salary, years of experience, advanced training, etc.
3. A frame with each student being a case and variables such as the student's test scores, the class that the student is in, and the student's family income and parent education.

Once you choose the unit of analysis, you combine information from the different data frames to carry out the data analysis, generating a single data frame in which cases are your chosen unit of analysis. The choice of the unit of analysis can be determined by many things, such as the availability of data. As a general rule, it's best to make the unit of analysis as small as possible. But there can be obstacles to doing this. You might find, for instance, that for privacy reasons (or less legitimate reasons of secrecy) the school district is unwilling to release the data at the individual student level, or even to release data on individual classes.

In the past, limitations in data analysis techniques and computational power provided a reason to use a coarse unit of analysis. Only a small amount of data could be handled effectively, so the unit of analysis was made large. For example, rather than using tens of thousand of individual students as the unit of

analysis, a few dozen schools might be used instead. Nowadays these reasons are obsolete. The methods that will be covered in this book allow for a very fine unit of analysis. Standard personal computers have plenty of power to perform the required calculations.

2.3 Populations and Samples

A data frame is a collection, but a collection of what? Two important statistical terms are “population” and “sample.” A **population** is the set of all the possible objects or units which might have been included in the collection. The root of the word “population” refers to people, and often one works with data frames in which the cases are indeed individual people. The statistical concept is broader; one might have a population of sea shells, a population of houses, a population of events such as earthquakes or coin flips.

A **sample** is a selection of cases from the population. The **sample size** is the number of cases in the sample. For the shells in Figure 2.1, the sample size is $n = 103$.

A **census** is a sample that contains the entire population. The most familiar sort of census is the kind to count the people living in a country. The United States and the United Kingdom have a census every ten years. Countries such as Canada, Australia, and New Zealand hold a census every five years.

Almost always, the sample is just a small fraction of the population. There are good reasons for this. It can be expensive or damaging to take a sample: Imagine a biologist who tried to use all the laboratory rats in the world for his or her work! Still, when you draw a sample, it is generally because you are interested in finding out something about the population rather than just the sample at hand. That is, you want the sample to be genuinely representative of the population. (In some fields, the ability to draw conclusions from a sample that can be generalized is referred to as **external validity** or **transferability**.)

The process by which the sample is taken is important because it controls what sorts of conclusions can legitimately be drawn from the sample. One of the most important ideas of statistics is that a sample will be representative of the population if the sample is collected at random. In a **simple random sample**, each member of the population is equally likely to be included in the sample.

Ironically, taking a random sample, even from a single spot on the beach, requires organization and planning. The sea shells were collected haphazardly, but this is not a genuinely random sample. The bigger shells are much easier to see and pick up than the very small ones, so there is reason to think that the small shells are under-represented: the collection doesn’t have as big a proportion of them as in the population. To make the sample genuinely random, you need to have access in some way to the entire population so that you can pick any member with equal probability. For instance, if you want a sample of students at a particular university, you can get a list of all the students from the university registrar and use a computer to pick randomly from the list. Such a list of the entire set of possible cases is called a **sampling frame**.

In a sense, the sampling frame is the *definition* of the population for the purpose of drawing conclusions from the sample. For instance, a researcher studying cancer treatments might take the sampling frame to be the list of all the patients who visit a particular clinic in a specified month. A random sample from that sampling frame can reasonably be assumed to represent that particular population, but not necessarily the population of all cancer patients.

You should always be careful to define your sampling frame precisely. If you decide to sample university students by picking randomly from those who enter the front door of the library, you will get a sample that might not be typical for *all* university students. There's nothing wrong with using the library students for your sample, but you need to be aware that your sample will be representative of just the library students, not necessarily all students.

When sampling at random, use formal random processes. For example, if you are sampling students who walk into the library, you can flip a coin to decide whether to include that student in your sample. When your sampling frame is in the form of a list, it's wise to use a computer random number generator to select the cases to include in the sample.

A **convenience sample** is one where the sampling frame is defined mainly in a way that makes it easy for the researcher. For example, during lectures I often sample from the set of students in my class. These students — the ones who take statistics courses from me — are not necessarily representative of all university students. It might be fine to take a convenience sample in a quick, informal, preliminary study. But don't make the mistake of assuming that the convenience sample is representative of the population. Even if you believe it yourself, how will you convince the people who are skeptical about your results?

When cases are selected in an informal way, it's possible for the researcher to introduce a non-representativeness or **sampling!bias**. For example, in deciding which students to interview who walk into the library, you might consciously or subconsciously select those who seem most approachable or who don't seem to be in a hurry.

There are many possible sources of sampling bias. In surveys, sampling bias can come from non-response or self-selection. Perhaps some of the students who you selected randomly from the people entering the library have declined to participate in your survey. This **non-response** can make your sample non-representative. Or, perhaps some people who you didn't pick at random have walked up to you to see what you are up to and want to be surveyed themselves. Such **self-selected** people are often different from people who you would pick at random.

In a famous example of self-selection bias, the newspaper columnist Ann Landers asked her readers, "If you had it to do over again, would you have children?" Over 70% of the respondents who wrote in said "No." This result is utterly different from what was found in surveys on the same question done with a random sampling methodology: more than 90% said "Yes." Presumably the people who bothered to write were people who had had a particularly bad experience as parents whereas the randomly selected parents are representative of the whole population. Or, as Ann Landers wrote, "[T]he hurt, angry, and

disenchanted tend to write more readily than the contented” (See [7].)

Non-response is often a factor in political polls, where people don’t like to express views that they think will be unpopular.

It’s hard to take a genuinely random sample. But if you don’t, you have no guarantee that your sample is representative. Do what you can to define your sampling frame precisely and to make your selections as randomly as possible from that frame. By using formal selection procedures (e.g., coin flips, computer random number generators) you have the opportunity to convince skeptics who might otherwise wonder what hidden biases were introduced by informal selections. If you believe that your imperfect selection may have introduced biases — for example the suspected under-representation of small shells in my collection — be up-front and honest about it. In surveys, you should keep track of the non-response rate and include that in whatever report you make of your study.

Example 2.1: Struggling for a Random Sample Good researchers take great effort to secure a random sample. One evening I received a phone call at home from the state health department. They were conducting a survey of access to health care, in particular how often people have illnesses for which they don’t get treatment. The person on the other end of the phone told me that they were dialing numbers randomly, checked to make sure that I live in the area of interest, and asked how many adults over age 18 live in the household. “Two,” I replied, “Me and my wife.” The researcher asked me to hold a minute while she generated a random number. Then the researcher asked to speak to my wife. “She isn’t home right now, but I’ll be happy to help you,” I offered. No deal.

The sampling frame was adults over age 18 who live in a particular area. Once the researcher had made a random selection, as she did after asking how many adults are in my household, she wasn’t going to accept any substitutes. It took three follow-up phone calls over a few days — at least that’s how many I answered, who knows how many I wasn’t home for — before the researcher was able to contact my wife. The researcher declined to interview me in order to avoid self-selection bias and worked hard to contact my wife — the randomly selected member of our household — in order to avoid non-response bias.

2.4 Longitudinal and Cross-Sectional Samples

Data are often collected to study the links between different traits. For example, the data in the following table are a small part of a larger data set of the speeds of runners in a ten-mile race held in Washington, D.C. in 2004. The variable net gives the time from the start line to the finish line, in seconds. Such data might be used to study the link between age and speed, for example to find out at what age people run the fastest and how much they slow down as they age beyond that.



state	net	age	sex
DC	6382	23	F
VA	5080	26	F
DC	4742	27	M
Kenya	2962	27	M
DC	6291	29	F
MD	6405	32	M
VA	6608	34	F
DC	5921	37	M
MD	5549	41	F
MD	5486	46	F
VA	8374	53	F
PA	6026	53	M
VA	5526	60	M
VA	5585	61	M
VA	5931	65	M

... and so on.

This sample is a **cross section**, a snapshot of the population that includes people of different ages. Each person is included only once.

Another type of sample is **longitudinal**, where the cases are tracked over time, each person being included more than once in the data frame. A longitudinal data set for the runners might look like this:

state	net	age	sex	year
DC	6382	23	F	2004
DC	6516	24	F	2005
DC	6493	25	F	2006
DC	6526	26	F	2007
DC	6571	27	F	2008
MD	6405	32	M	2004
MD	6819	34	M	2006
MD	6753	35	M	2007

... and so on.

If your concern is to understand how individual change as they age, it's best to collect data that show such change in individuals. Using cross-sectional data to study a longitudinal problem is risky. Suppose, as seems likely, that younger runners who are slow tend to drop out of racing as they age, so the older runners who do participate are those who tend to be faster. This could bias your estimate of how running speed changes with age.

2.5 Computational Technique

2.5.1 Reading and Writing Data

Data used in statistical modeling are usually organized into tables, often created using spreadsheet software. Most people presume that the same software used to create a table of data should be used to display and analyze it. This is part of the reason for the popularity of spreadsheet programs such as Excel.

For statistical modeling, it's helpful to take another approach that strictly separates the processes of data collection and of data analysis: use one program to create data files and another program to analyze the data stored in those files. By doing this, one guarantees that the original data are not modified accidentally in the process of analyzing them. This also makes it possible to perform many different analyses of the data; modelers often create and compare many different models of the same data.

The spreadsheet programs that can create data files use a variety of different formats. Many of these formats are proprietary and include various features that make it difficult for any other software to read the file. A good, simple, general purpose format supported by spreadsheet software and by statistical software is called the **comma separated value** or **CSV** format.

The next sections describe how to read data from a CSV file into R and how to use a spreadsheet program to create new data.

Reading CSV Files into R

For the data sets associated with this book and its exercises, an easy way to import data into R is with the `ISMdata` operator that's included in the extensions to R found in the `ISM.Rdata` workspace file. (See Section 1.4.4 on page 30. You must load the workspace file before you can use `ISMdata`.)

`ISMdata` lets you refer to a file by a short name in quotes without worrying about where it is located. It knows where to locate the data files used with this book, whether they be on your computer or on the web. For example, the data set `hdd-minneapolis.csv` is stored on the Internet. You can read it into an object named `hdd` with this statement:

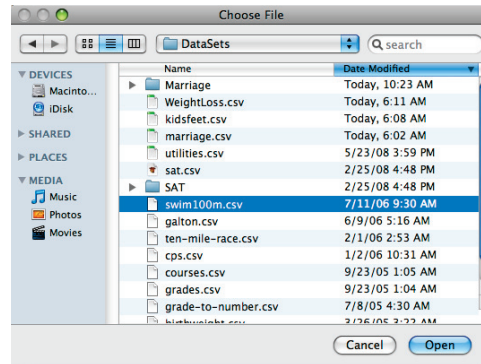
```
> hdd = ISMdata("hdd-minneapolis.csv")
Not in library. Trying to find it on the web ...
File was read from the web.
```

If you do not have an Internet connection, then `ISMdata` will be able to locate only files on your computer.

It is possible to use `ISMdata` to read in your own data that you have created and stored in CSV files. To do this, you need to tell `ISMdata` where the data is located. The easiest way to do this is to use a mouse-based file navigator. To do this, invoke `ISMdata` with no input argument:

```
> swim = ISMdata()
```

Since there was no character string file name given as an argument, `ISMdata` brings up a file navigator to let you select the file interactively:



Selecting a file interactively.

Selecting and opening the desired file will cause it to be read into R as a data frame. Make sure to choose the correct CSV file. If you choose some other sort of file by accident, `ISMdata` will struggle to read it in, displaying various junk on your screen.

The statement above will cause the resulting data frame to be called `swim`. But be careful. If you accidentally choose a different file (e.g., `kidsfeet.csv`) the data from that file will be read in and stored under the name `swim`, even though they have nothing to do with swimming.

[Optional] Although `ISMdata` will work well for most purposes, some users might prefer the flexibility offered by the built-in R operators for reading in files. These include `read.csv`, `scan`, and a variety of operators for importing files from other packages, such as the `read.spss` operator in the “foreign” library package.

For obvious reasons, the most of the examples used in this book are based on data that has already been collected and stored in CSV files. In your own investigations, you will generally need to create your own data sets. Instructions and suggestions for creating CSV files and codebooks are given in the exercises.



2.5.2 Simple Operations with Data Frames

To illustrate, here are data on world-record swimming times:

```
> swim = ISMdata("swim100m.csv")
```

This data set, stored in the object named `swim`, has three variables:

```
> names(swim)
[1] "year" "time" "sex"
```

Year refers to the calendar year in which the record was set; time is the record time itself, in seconds; sex records whether the record is for men or women.

```
> head(swim)
  year time sex
1 1905 65.8  M
2 1908 65.6  M
3 1910 62.8  M
... and so on.
```

Numerical Operations

There are two basic kinds of variables: quantitative and categorical. R treats these variables differently, as it should since operations that make perfect sense for a quantitative variable (such as the sum or the mean or median) make little sense for categorical variables. For instance, it makes sense to compute

```
> max(swim$year)
[1] 2004
```

But it does not make sense to compute the mean sex:

```
> max(swim$sex)
Error in Summary.factor
  max not meaningful for factors
```

The word **factor** is how R refers to categorical variables. Once you know that, the second line of the error message makes more sense.

Adding a New Variable

Sometimes you will compute a new quantity from the variables and you want to treat this as a new variable. You can do this by assignment to the data frame, using the \$ and giving a new name for the variable. As a trivial example, here is how to convert the swim time from seconds to minutes:

```
> swim$minutes = swim$time / 60
```

Once this has been done, the new variable appears just like the old ones:

```
> names(swim)
[1] "year"    "time"    "sex"     "minutes"
```

You could also, if you want, redefine an existing variable, for instance:

```
> swim$time = swim$time / 60
```

Such assignment operations do not change the original file from which the data were read, only the data frame in the current session of R.

Extracting Subsets of Data

Selecting a subset of cases is done with the `subset` operator. For instance, here's how to create a data frame with just the women's records:

```
> women = subset( swim, sex=='F' )
```

The `subset` operator takes two arguments: the first is a data frame from which to extract the subset. The second is a logical (TRUE/FALSE) criterion for each case, saying whether to include it.

Notice that in this example, the name `sex` was used, rather than the full `swim$sex`. The `subset` operator allows the shorthand since the first argument sets a context for evaluating any names in the second argument. Other operators also allow this sort of shorthand.

The `subset` operator creates a new data frame which you can assign to a name; it does not modify the original data frame. You can have as many data frames as you want in an R session, so there is little reason to modify the original. But if you want to, you can do it by re-assignment:

```
> swim = subset( swim, sex=='F' )
```

After this command, the male records are no longer in `swim`. If you want them back, you have to re-read the original data file.

2.5.3 Sampling

Suppose you have a sampling frame with 1000 cases, arranged as a spreadsheet with one row for each case. You want to select a random set of 20 of these. The `shuffle` operator lets you pick random members of a set.

A common use for `shuffle` is to pick random cases from a data frame, just as you might do when sampling randomly from a sampling frame.

```
> shuffle(swim, 5)
  year time sex
55 1976 55.65  F
39 1924 72.20  F
12 1944 55.90  M
18 1964 52.90  M
46 1936 64.60  F
```

The results returned by `shuffle` will never contain the same case more than once, just as if you were dealing cards from a shuffled deck. In contrast, `resample` replaces each case after it is dealt so that it can appear more than once in the result. This is called **sampling with replacement**. This will be useful later on when studying the statistical properties of the sampling process. For example, `resample` allows you to generate a sample of any size, even one that is bigger than the data set from which the sample is being drawn.

Chapter 3

Describing Variation

Variation itself is nature's only irreducible essence. Variation is the hard reality, not a set of imperfect measures for a central tendency. Means and medians are the abstractions. — Stephen Jay Gould

A statistical model partitions variation into parts. People describe the partitioning in different ways depending on their purposes and the conventions of the field in which they work: explained variation versus unexplained variation; described variation versus undescribed; predicted variation versus unpredicted; signal versus noise; common versus individual. This chapter describes ways to quantify variation in a single variable. Once you can quantify variation, you can describe how models divide it up.

To start, consider a familiar situation: the variation in human heights. Everyone is familiar with height and how heights vary from person to person, so variation in height provides a nice example to compare intuition with the formal descriptions of statistics. Perhaps for this reason, height was an important topic for early statisticians. In the 1880s, Francis Galton, one of the pioneers of statistics, collected data on the heights of about 900 adult children and their parents in London. Figure 3.1 shows part of his notebook.

Galton was interested in studying the relationship between a full-grown child's height and his or her mother's and father's height. One way to study this relationship is to build a model that accounts for the child's height — the **response variable** — by one or more **explanatory variables** such as mother's height or father's height or the child's sex. In later chapters you will construct such models. For now, though, the objective is to describe and quantify how the response variable varies across children without taking into consideration any explanatory variables.

Galton's height measurements are from about 200 more-or-less normal families in the city of London. In itself, this is an interesting choice. One might think that the place to start would be with exceptionally short or tall people, looking at what factors are associated with these extremes.

	Father	Mother	Sons in order of height	Daughters in order of height
1	18.5	7.0	13.2	9.2, 9.0, 9.0
2	15.5	6.5	13.5, 12.5	5.5, 5.5
3	15.0	about 4.0	11.0	8.0
4	15.0	4.0	10.5, 8.5	7.0, 4.5, 3.0
5	15.0	1.5	12.0, 9.0, 8.0	6.5, 2.5, 2.5

Figure 3.1: Part of Francis Galton's notebook recording the heights of parents and their adult children. [8]

Adults range in height from a couple of feet to about nine feet. (See Figure 3.2.) One way to describe variation is by the **range of extremes**: an interval that includes every case from the smallest to the largest.

What's nice about describing variation through the extremes is that the range includes every case. But there are disadvantages. First, you usually don't always have a **census**, measurements on the entire population. Instead of a census, you typically have only a **sample**, a subset of the population. Usually only a small proportion of the population is included in a sample. For example, of the millions of people in London, Galton's sample included only 900 people. From such a sample, Galton would have had no reason to believe the either the tallest or shortest person in London, let alone the world, happened to be included.

A second disadvantage of using the extremes is that it can give a picture that is untypical. The vast majority of adults are between $4\frac{1}{2}$ feet and 7 feet tall. Indeed, an even narrower range — say 5 to $6\frac{1}{2}$ feet — would give you a very good idea of typical variation in heights. Giving a comprehensive range — 2 to 9 feet — would be misleading in important ways, even if it were literally correct.

A third disadvantage of using the extremes is that even a single case can have a strong influence on your description. There are about six billion people on earth. The discovery of even a single, exceptional 12-foot person would cause you substantially to alter your description of heights even though the population — minus the one new case — remains unchanged.

It's natural for people to think about variation in terms of records and extremes. People are used to drawing conclusions from stories that are newsworthy and exceptional; indeed, such anecdotes are mostly what you read and hear about. In statistics, however, the focus is usually on the unexceptional cases, the typical cases. With such a focus, it's important to think about **typical variation** rather than extreme variation.



Figure 3.2: Some extremes of height. Angus McAskill (1825-1863) and Charles Sherwood Stratton (1838-1883). McAskill was 7 feet 9 inches tall. Stratton, also known as Tom Thumb, was 2 feet 6 inches in height.

3.1 Coverage Intervals

One way to describe typical variation is to specify a fraction of the cases that are regarded as typical and then give the **coverage interval** or range that includes that fraction of the cases.

Imagine arranging all of the people in Galton's sample of 900 into order, the way a school-teacher might, from shortest to tallest. Now walk down the line, starting at the shortest, counting heads. At some point you will reach the person at position 225. This position is special because one quarter of the 900 people in line are shorter and three quarters are taller. The height of the person at position 225 — 64.0 inches — is the 25th **percentile** of the height variable in the sample.

Continue down the line until you reach the person at position 675. The height of this person — 69.7 inches — is the 75th percentile of height in the sample.

The range from 25th to 75th percentile is the **50-percent coverage interval**: 64.0 to 69.7 inches in Galton's data. Within this interval is 50% of the cases.

For most purposes, a 50% coverage interval excludes too much; half the cases are outside of the interval.

Scientific practice has established a convention, the **95-percent coverage interval** that is more inclusive than the 50% interval but not so tied to the extremes as the 100% interval. The 95% coverage interval includes all but 5% of the cases, excluding the shortest 2.5% and the tallest 2.5%.

To calculate the 95% coverage interval, find the 2.5 percentile and the 97.5

Position in list k	Height (inches)	# of previous cases $k - 1$	Percentile
1	59.0	0	0
2	61.5	1	10
3	62.0	2	20
4	63.0	3	30
5	64.7	4	40
6	65.5	5	50
7	68.0	6	60
8	69.0	7	70
9	72.0	8	80
10	72.0	9	90
11	72.0	10	100

Table 3.1: Finding sample percentiles by sorting and counting.

percentile. In Galton's height data, these are 60 inches and 73 inches, respectively.

There is nothing magical about using the coverage fractions 50% or 95%. They are just conventions that make it easier to communicate clearly. But they are important and widely used conventions.

To illustrate in more detail the process of finding coverage intervals, let's look at Galton's data. Looking through all 900 cases would be tedious, so the example involves just a few cases, the heights of 11 randomly selected people:

72.0 62.0 68.0 65.5 63.0 64.7 72.0 69.0 61.5 72.0 59.0

Table 3.1 puts these 11 cases into sorted order and gives for each position in the sorted list the corresponding percentile. For any given height in the table, you can look up the percentile of that height, effectively the fraction of cases that came previously in the list. In translating the position in the sorted list to a percentile, convention puts the smallest case at the 0th percentile and the largest case at the 100th percentile. For the k th position in the sorted list of n cases, the percentile is taken to be $(k - 1)/(n - 1)$.

With $n = 11$ cases in the sample, the sorted cases themselves stand for the 0th, 10th, 20th, ..., 90th, and 100th percentiles. If you want to calculate a percentile that falls in between these values, you (or the software) interpolate between the samples. For instance, the 75th percentile would, for $n = 11$, be taken as half-way between the values of the 70th and 80th percentile cases.

Coverage intervals are found from the tabulated percentiles. The 50% coverage interval runs from the 25th to the 75th percentile. Those exact percentiles happen not be in the table, but you can estimate them. Take the 25th percentile to be half way between the 20th and 30th percentiles: 62.5 inches. Similarly, take the 75th percentile to be half way between the 70th and 80th percentiles: 70.5 inches.

Thus, the 50% coverage interval for this small subset of $N = 11$ cases is 62.5

to 70.5 inches. For the complete set of $N = 900$ cases, the interval is 64 to 69 inches — not exactly the same, but not too much different. In general you will find that the larger the sample, the closer the estimated values will be to what you would have found from a census of the entire population.

This small $N = 11$ subset of Galton's data illustrates a potential difficulty of a 95% coverage interval: The values of the 2.5th and 97.5th percentiles in a small data set depend heavily on the extreme cases, since interpolation is needed to find the percentiles. In larger data sets, this is not so much of a problem.

A reasonable person might object that the 0th percentile of the sample is probably not the 0th percentile of the population; a small sample almost certainly does not contain the shortest person in the population. There is no good way to know whether the population extremes will be close to the sample extremes and therefore you cannot demonstrate that the estimates of the extremes based on the sample are valid for the population. The ability to draw demonstrably valid inferences from sample to population is one of the reasons to use a 50% or 95% coverage interval rather than the range of extremes.

Different fields have varying conventions for dividing groups into parts. In the various literatures, one will read about **quintiles** (division into 5 equally sized groups, common in giving economic data), **stanines** (division into 9 unevenly sized groups, common in education testing), and so on.

In general-purpose statistics, it's conventional to divide into four groups: **quartiles**. The dividing point between the first and second quartiles is the 25th percentile. For this reason, the 25th percentile is often called the "first quartile." Similarly, the 75th percentile is called the "third quartile."

The most famous percentile is the 50th: the **median**, which is the value that half of the cases are above and half below. The median gives a good representation of a typical value. In this sense, it is much like the **mean**: the average of all the values.

Neither the median nor the mean describes the variation. For example, knowing that the median of Galton's sample of heights is 66 inches does not give you any indication of what is a typical range of heights. In the next section, however, you'll see how the mean can be used to set up an important way of measuring variation: the typical distance of cases from the mean.

3.2 The Variance and Standard Deviation

The 95% and 50% coverage intervals are important descriptions of variation. For constructing models, however, another format for describing variation is more widely used: the variance and standard deviation.

To set the background, imagine that you have been asked to describe a variable in terms of a *single* number that typifies the group. Think of this as a very simple model, one that treats all the cases as exactly the same. For human heights, for instance, a reasonable model is that people are about 5 feet 8 inches tall (68 inches).