3

CLASSICAL STATISTICS

“The difference between ‘significant’ and ‘not significant’ is not itself statistically significant.”
—Andrew Gelman

All hurricanes are different, but Statistics helps you characterize hurricanes from the typical to the extreme. In this chapter, we provide an introduction to classical (or frequentist) statistics. To get the most out of it, we again encourage you to open an R session and type in the code as you read along.

3.1 DESCRIPTIVE STATISTICS

Descriptive statistics are used to summarize your data. The mean and the variance are good examples. So is correlation. Data can be a set of weather records or output from a global climate model. Descriptive statistics provide answers to questions like does Jamaica experience more hurricanes than Puerto Rico?

In Chapter 2, you learned some functions for summarizing your data, let us review. Recall that the data set *H.txt* is a list of hurricane counts by year making landfall in the United States (excluding Hawaii). To input the data and save them as a data object, type

```
> H = read.table("H.txt", header=TRUE)
```

Make sure the data file is located in your working directory. To check your working directory, type `getwd()`.

3.1.1 Mean, Median, and Maximum

Sometimes all you need are a few summary statistics from your data. You can obtain the mean and variance by typing

```
> mean(H$All); var(H$All)
[1] 1.69375
[1] 2.10059
```

Recall that the semicolon acts as a return so you can place multiple function commands on the same text line. The sample mean is a measure of the central tendency and the sample variance is a measure of the spread. These are called the first- and second- moment statistics. Like all statistics, they are random variables. A random variable can be thought of as a quantity whose value is not fixed; it changes depending on the values in your sample.

If you consider the number of hurricanes over a different sample of years, the sample mean will almost certainly be different. Same with the variance. The sample mean provides an estimate of the population mean (the mean over all past and future years). Different samples drawn from the same population will result in different values, but as the sample size increases, the values will get closer to the population values.

The values printed on your screen have too many digits. Since there are only 160 years, the number of significant digits is 2 or 3. This can be changed using the `signif` function.

```
> signif(mean(H$All), digits=3)
[1] 1.69
```

Notice how the functions are nested. Here the `mean` function is nested in the `signif` function. You can also set the number of digits globally using the `options` function as mentioned in Chapter 2.

The median, standard deviation, maximum, and minimum are obtained by typing

```
> median(H$All); sd(H$All); max(H$All); min(H$All)
[1] 1
[1] 1.45
[1] 7
[1] 0
```

At least one year had seven hurricanes hit the U.S. coast. Higher-order moments like skewness and kurtosis are available in the `moments` package.

To determine which year has the maximum, you first test each year's count against the maximum using the logical operator ==. This provides a list of years for which the operation returns a TRUE. You then subset the hurricane year according to this logical test. For example, type

```
> maxyr = H$All == max(H$All)
> H$Year[maxyr]
[1] 1886
```

This tells you that 1886 is the single year with the most hurricanes.

The `which.max` function is similar but returns the row number index corresponding only to the *first* occurrence of the maximum. You can then subset the hurricane year on the index.

```
> H$Year[which.max(H$All)]
[1] 1886
```

Similarly, to find the years with no hurricanes, type

```
> H$Year[H$All == min(H$All)]
 [1] 1853 1862 1863 1864 1868 1872 1884 1889 1890
[10] 1892 1895 1902 1905 1907 1914 1922 1927 1930
[19] 1931 1937 1951 1958 1962 1973 1978 1981 1982
[28] 1990 1994 2000 2001 2006 2009 2010
```

And to determine how many years have no hurricanes, type

```
> sum(H$All == 0)
[1] 34
```

The `sum` function counts a TRUE as 1 and a FALSE as 0, so the result tells you how many years have a count of zero hurricanes.

You also might be interested in streaks. For instance, what is the longest streak of years without a hurricane? To answer this, first you create an ordered vector of years with at least one hurricane. Next you use the `diff` function to take differences between sequential years given in the ordered vector. Finally, you find the maximum of these differences minus one.

```
> st = H$Year[H$All > 0]
> max(diff(st) - 1)
[1] 3
```

Thus the longest streak without hurricanes is only 3 years.

Alternatively, you can use the `rle` function to compute the length and values of runs in a vector and then table the results.

```
> st = H$All == 0
> table(rle(st))
       values
lengths FALSE TRUE
      1     2   21
      2     6    5
      3     4    1
      4     4    0
      5     2    0
      6     3    0
      7     2    0
```

```
8       1    0
10      1    0
11      1    0
13      1    0
```

The results show five sets of two consecutive years without a hurricane and one set of three consecutive years without a hurricane.

3.1.2 Quantiles

Percentiles also help you describe your data. The *p*th percentile ($p/100$ quantile) is the value that cuts off the first *p* percent of the data when the data values are sorted in ascending order. The `quantile` function is used to obtain these values.

First import the NAO data file. Refer to Chapter 6 for a description of these data.

```
> NAO = read.table("NAO.txt", header=TRUE)
```

To obtain quantiles of the June NAO values, type

```
> nao = NAO$Jun
> quantile(nao)
    0%    25%    50%    75%   100%
-4.050 -1.405 -0.325  0.760  2.990
```

By default, you get the minimum, the maximum, and the three quartiles (the 0.25, 0.5, and 0.75 quantiles), so named because they correspond to a division of the values into four equal parts. The difference between the first and third quartiles is called the IQR, which is sometimes use as an alternative to the standard deviation because it is less affected by extremes.

To obtain other quantiles, you include the argument `prob` in the function. For example, to find the 19th, 58th, and 92nd percentiles of the June NAO values, type

```
> quantile(nao, prob=c(.19, .58, .92))
   19%    58%    92%
-1.620 -0.099  1.624
```

Be aware that there are different ways to compute quantiles. Details can be found in the documentation (type `help(quantile)`).

3.1.3 Missing Values

Things become a bit more complicated if your data contain missing values. R handles missing values in different ways depending on the context. With a vector of values, some of which are missing and marked with `NA`, the `summary` function computes statistics and returns the number of missing values. For example, read in the monthly sea-surface temperatures (*SST.txt*), create a vector of August values, and summarize them.

```
> SST = read.table("SST.txt", header=TRUE)
> sst = SST$Aug
> summary(sst)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   22.6    23.0    23.1    23.1    23.3    23.9
   NA's
    5.0
```

Here you see the summary statistics and note that there are 5 years with missing values during August. The summary statistics are computed by removing the missing values.

However, a function that computes a statistic, like the `mean`, applied to a vector with missing values returns an `NA`.

```
> mean(sst)
[1] NA
```

In this case, R does not remove the missing values unless requested to do so. The mean of a vector with an unknown value is unknown. If you wish to have the missing values removed, you need to include the argument `na.rm=TRUE`,

```
> mean(sst, na.rm=TRUE)
[1] 23.1
```

An exception is the `length` function, which does not understand the argument `na.rm`, so you cannot use it to count the number of missing values. Instead, you use the `is.na` function, which returns a TRUE for a missing value and FALSE otherwise. You then use the `sum` function to count the number of TRUEs. For your August SST data, type

```
> sum(is.na(sst))
[1] 5
```

The number of nonmissing data values are obtained by using the logical negation operator `!` (read as 'not'). For example, type

```
> sum(!is.na(sst))
[1] 155
```

This tells you there are 155 years with August SST values.

3.2 PROBABILITY AND DISTRIBUTIONS

Climate is the set of all weather patterns. You can think of weather as a data-generating machine. For example, with each season the number of hurricanes is recorded as a count. This count is a data value from the weather machine that gets collected alongside counts from other years. Other data are also available like the highest wind speed, and so on. This view of data arising from a generating process gives statistics a prominent role in understanding climate.

3.2.1 Random Samples

Statistics is an application of probability theory. Probability theory arose from studying games of chance like rolling dice and picking cards at random. Randomness and probability are central to statistics.

You can simulate games of chance with the `sample` function. For instance, to pick 4 years at random from the 1990s, you type

```
> sample(1990:1999, size=4)
[1] 1997 1995 1998 1996
```

The sequence function `:` is used to create a vector of 10 values representing the years in a decade. The `sample` function is then used to pick, at random without replacement, a set of four (`size=4`) values. This is called a *random sample*.

Notice that in deciphering R code, it is helpful to read from right to left and from inside to outside. That is, you start by noting a size of four from a sequence of numbers from 1990 through 1999 and then taking a sample from these numbers.

The default is "sample without replacement." The sample will not contain a repeat year. If you want to sample with replacement, use the argument `replace=TRUE`. Sampling with replacement is suitable for modeling the occurrence of El Niño (E) and La Niña (L) events. An El Niño event is characterized by a warm ocean along the equator near the coast of Peru. A La Niña event is just the opposite. The fluctuation between El Niño and La Niña events coincides with the fluctuation in hurricane activity.

To model the occurrence over 10 random seasons, type

```
> sample(c("E", "L"), size=10, replace=TRUE)
 [1] "L" "E" "L" "E" "E" "E" "L" "L" "E" "E"
```

Historically, the probability of an El Niño is about the same as the probability of a La Niña, but the idea of random events is not restricted to equal probabilities. For instance, suppose you are interested in the occurrence of hurricanes hitting Florida. Let the probability be 12 percent that a hurricane picked at random hits Florida. You simulate 24 random hurricanes by typing

```
> sample(c("hit", "miss"), size=24, replace=TRUE,
+   prob=c(.12, .88))
 [1] "miss" "miss" "miss" "miss" "hit"  "miss" "miss"
 [8] "miss" "hit"  "miss" "miss" "miss" "miss" "miss"
[15] "miss" "miss" "miss" "miss" "hit"  "miss" "miss"
[22] "miss" "miss" "miss"
```

The simulated frequency of hits will not be exactly 12 percent, but the variation about this percentage decreases as the sample size increases according to the law of large numbers.

3.2.2 Combinatorics

Returning to your set of years from a decade. Common sense tells you that the probability of getting each of the 10 years in a sample of size 10 with each year picked at random without replacement is one. But what is the probability of randomly drawing a set of three particular years?

This is worked out as follows. The probability of getting a particular year (say 1992) as the first one in the sample is $1/10$, the next one is $1/9$, and the next one is $1/8$. Thus the probability of a given sample is $1/(10 \times 9 \times 8)$. The `prod` function calculates the product of a set of numbers, so you get the probability by typing

```
> 1/prod(10:8)
[1] 0.00139
```

Note that this is the probability of getting a set of 3 years in a particular order (i.e., 1992, 1993, 1994). If you are not interested in the arrangement of years, then you need to include the cases that give the 3 years in a different order. The probability will be the same if the years are in any order, so you need to know the number of combinations and then multiply this number by the previous probability.

Given a sample of three numbers, there are three possibilities for the first number, two possibilities for the second, and only one possibility for the third. Thus the number of combinations of three numbers in any order is $3 \times 2 \times 1$ or 3!. You can use the `factorial` function and multiply this number by the probability to get

```
> factorial(3)/prod(10:8)
[1] 0.00833
```

Thus the probability of a given sample of 3 years in any order is 0.83 percent.

You get the same result using the `choose` function. For any set containing n elements, the number of distinct x-element samples of it that can be formed (the x combinations of its elements) is given by

$$\binom{n}{x} = \frac{n!}{x!(n-x)!}, \tag{3.1}$$

which is read as “n choose x.” The multiplicative inverse of this number is the probability. You can check the equality of these two ways of working out the probability by typing

```
> factorial(3)/prod(10:8) == 1/choose(10,3)
[1] TRUE
```

Recall that the double equal signs indicate a logical operator with two possible outcomes (TRUE and FALSE), so a return of TRUE indicates equality.

3.2.3 Discrete distributions

It is likely that you are more interested in some calculated value from a random sample. Instead of a set of hits and misses on Florida from a sample of hurricanes, you might want to know the number of hits. Since the set of hits is random so is the sum over all hits.

The number of hits is another example of a *random variable*. In this case, it is a nonnegative integer that can take on values in $0, 1, 2, \ldots, n$, where n is the total number of hurricanes. Said another way, given a set of n North Atlantic hurricanes, the number that hit Florida is a discrete random variable H.

A random variable H has a *probability distribution* that is described using $f(h) = P(H = h)$. The set of all possible Florida counts is the random variable denoted with a large H, while a particular count is denoted with a small h. This is standard statistical notation. Thus $f(h)$ is a function that assigns a probability to each possible count. It is written as

$$f(h|p,n) = \binom{n}{h} p^h (1-p)^{n-h}, \tag{3.2}$$

where the parameter p is the probability of a Florida hit given a North Atlantic hurricane. This is known as the *binomial distribution* and $\binom{n}{h}$ is known as the binomial coefficient.

Returning to your example above, in a set of 24 hurricanes with a probability $p =$ 12% that a hurricane picked at random hits Florida (hit rate), the probability that exactly three of them will strike Florida $[P(H = 3)]$ is found by typing

```
> choose(24, 3) * .12^3 * (1 - .12)^(24 - 3)
[1] 0.239
```

Thus there is a 23.9 percent chance of having 3 of the 24 hurricanes hit Florida. Note, there is a probability associated with all nonnegative integers from 0 to 24. For counts 0 to 15, the probabilities are plotted in Figure 3.1. The distribution is discrete with probabilities assigned only to counts. The distribution peaks at three and four hurricanes, and there are small but nonzero probabilities for the largest counts.

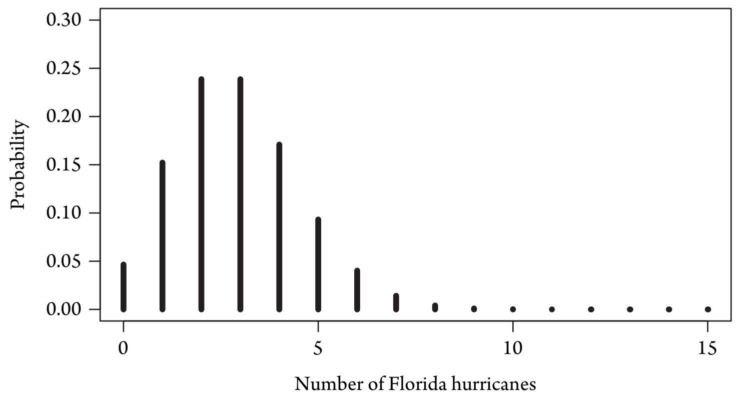

Figure 3.1 Probability of h Florida hurricanes, given a random sample of 24 North Atlantic hurricanes using a binomial model with a hit rate of 12%.

3.2.4 Continuous Distributions

A lot of climate data all available as observations on a continuous scale. For instance, the NAO index used in Chapter 2 is given in values of standard deviation. Values are recorded to a finite precision, but in practice this is generally not relevant. What is relevant is the fact that the values tend to cluster near a central value; values far from this central value are more rare than values farther from it.

A continuous value has no probability associated with it. This is because there are infinitely many values between any two values, so the probability at any particular one is zero. Instead we have the concept of *density*. The density of a continuous random variable is a function that describes the relative likelihood that the variable will have a particular value. The density is the probability associated with a small region around the value divided by the size of the region. This probability density function is nonnegative and its sum over the set of all possible values is equal to one.

The *cumulative distribution function* for continuous random variables is given by

$$F(x) = \int_{-\infty}^{x} f(u)du \tag{3.3}$$

If f is continuous at x, then

$$f(x) = \frac{\mathrm{d}}{\mathrm{d}x}F(x) \tag{3.4}$$

You can think of $f(x)\mathrm{d}x$ as the probability of X falling within the small interval $[x, x+\mathrm{d}x]$.

The most common continuous distribution is the normal (or Gaussian distribution). It has a density given by

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3.5}$$

where the parameter μ is the mean and the parameter σ^2 is the variance. We write $N(\mu,\sigma^2)$ as a shorthand for this distribution. The normal distribution has the characteristic bell shape with the mean located at the peak of the distribution.

Changing μ shifts the distribution left or right changing σ^2 widens and narrows the distribution, while the values remain symmetric about the peak (Fig. 3.2). The distance between the two inflection points, where the curves change from opening downward to opening upward, is two standard deviations. The normal distribution is a family of distributions, where the family members have different parameter values. The family member with $\mu = 0$ and $\sigma^2 = 1$ is called the standard normal distribution.

The normal distribution is the foundation for many statistical models. It is analytically tractable and arises as the outcome of the *central limit theorem*, which states that the sum of a large number of random variables, regardless of their distribution, is distributed approximately normally. Thus the normal distribution is commonly encountered in practice as a model for complex phenomena. In climatology, it is used as a model for observational error and for the propagation of uncertainty.

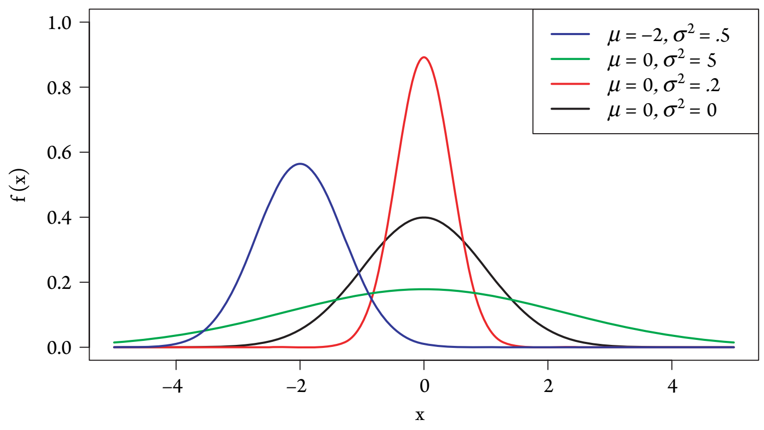

Figure 3.2 Probability density functions for a normal distribution.

3.2.5 Distributions

In R distributions are functions. This eliminates the need for lookup tables. Distributions come in families. Each family is described by a function with parameters. As noted earlier, the normal distribution is a family of distributions with each member having a different mean and variance.

The *uniform distribution* is a family of continuous distributions on the interval $[a, b]$ that assigns equal probability to equal-sized areas in the interval, where the parameters a and b are the endpoints of the interval. Here we look at the normal and Poisson distributions, but the others follow the same pattern.

Four items can be calculated for a statistical distribution:

- Density or point probability
- Cumulative distribution function
- Quantiles
- Random numbers

For each distribution, there is a function corresponding to each of the four items. The function has a prefix letter indicating which item. The prefix letter `d` is used for the probability density function, `p` is used for the cumulative distribution function, `q` is used for the quantiles, and `r` is used for random samples. The root name for the normal distribution in R is `norm`, so the probability density function for the normal distribution is `dnorm`, the cumulative distribution function is `pnorm`, the quantile function is `qnorm`, and the random sample function is `rnorm`.

3.2.6 Density

The density for a continuous distribution is a measure of the relative probability of getting a value close to x. With "close" defined as a small interval, this probability is the area under the curve between the endpoints of the interval. The density for a discrete distribution is the probability of getting exactly the value x. It is a density with respect to a counting measure.

You can use the density function to draw a picture of the distribution. First create a sequence of x values, then plot the sequence along with the corresponding densities from a distribution with values for the parameters.

As an example, the Weibull continuous distribution provides a good description for wind speeds. To draw its density, type

```
> w = seq(0, 80, .1)
> plot(w, dweibull(w, shape=2.5, scale=40), type="l",
+   ylab="Probability density")
```

The `seq` function generates equidistant values in the range from 0 to 80 in steps of 0.1. The distribution has two parameters: the shape and scale. Along with the vector of w values, you must specify the shape parameter. The default for the scale parameter is 1. The result is shown in Figure 3.3. Here the parameters are set to values that describe tropical cyclone wind speeds. The family of Weibull distributions includes members that are not symmetric. In this case, the density values in the right tail of the distribution are higher than they would be under the assumption that the winds are described by a normal distribution.

You create a similar plot using the `curve` function by typing

```
> curve(dweibull(x, shape=2.5, scale=40), from=0,
+   to=80)
```

Note that the first argument (here `dweibull`) must be a function or expression that contains `x`.

For discrete distributions, it is better to use pins as discrete symbols rather than a continuous curve. As an example, the Poisson distribution is a good description for

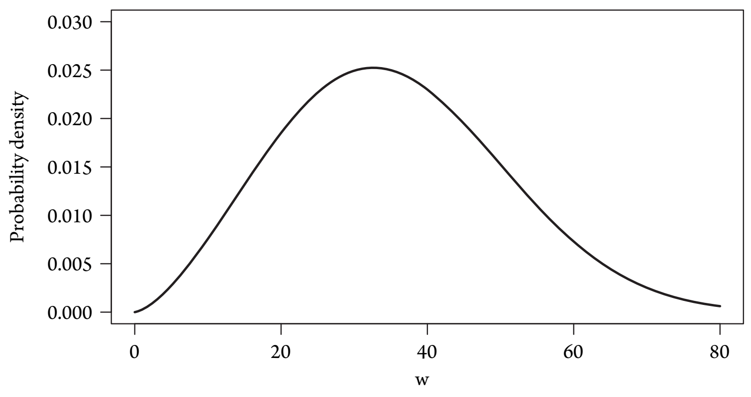

Figure 3.3 Probability density function for a Weibull distribution.

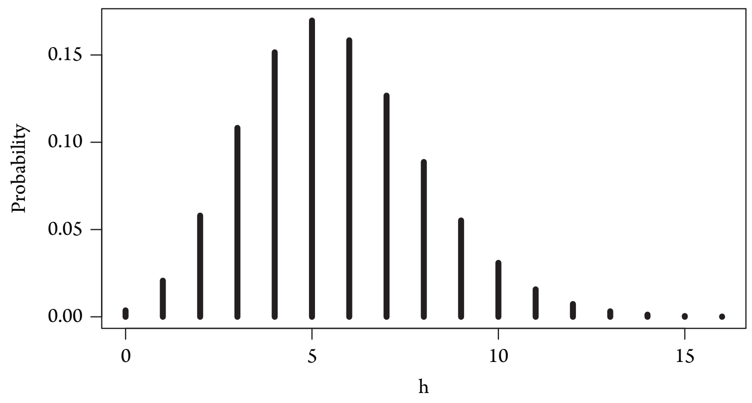

Figure 3.4 Probability mass function for a Poisson distribution.

hurricane counts. To draw its distribution, type

```
> h = 0:16
> plot(h, dpois(h, lambda=5.6), type="h", lwd=3,
+   ylab="Probability distribution")
```

The result is shown in Figure 3.4. The distribution corresponds to the probability of observing *h* number of hurricanes, given a value for the rate parameter. The argument `type="h"` causes pins to be drawn. The Poisson distribution is a limiting form of the binomial distribution with no upper bound on the number of occurrences. The rate parameter λ characterizes this process. For small values of λ, the distribution is positively skewed.

3.2.7 Cumulative Distribution Functions

The cumulative distribution function describes the probability less than or equal to a value x. It can be plotted, but it is often more informative to get probabilities for distinct values. The function `pnorm` returns the probability of getting a value equal to or smaller than its first argument in a normal distribution with a given mean and standard deviation.

For example, consider again the NAO data for June. Assuming these values are described by a normal distribution with a mean of -0.38 and standard deviation of 1.43, the chance that a June value is less than -1.5 is gotten by typing

```
> pnorm(-1.5, mean=mean(nao), sd=sd(nao))
[1] 0.217
```

or approximately 22 percent. That is, only about 22 percent of the June NAO values are less than or equal to -1.5.

Consider the hurricane data as another example. The annual rate of East Coast hurricanes is obtained by typing `mean(H$E)`. This is the rate (λ) parameter value for the

Poisson distribution. So the probability of next year having no East Coast hurricane is obtained by typing

```
> ppois(0, lambda=mean(H$E))
[1] 0.626
```

You express the probability as a percentage and round to three significant figures by typing

```
> round(ppois(0, lambda=mean(H$E)), digits=3) * 100
[1] 62.6
```

3.2.8 Quantile Functions

The quantile function is the inverse of the cumulative distribution function. The p-quantile is the value such that there is a p probability of getting a value less than or equal to it. The median value is, by definition, the .5 quantile.

In tests of statistical significance (see the next section), the p-quantile is usually set at $p = 0.05$ and is called the $\alpha = p \times 100\%$ significance level. You are interested in knowing the threshold that a test statistic must cross in order to be considered significant at that level. The p-value is the probability of obtaining a value as large, or larger, than the p-quantile.

Theoretical quantiles are also used to calculate confidence intervals. If you have n normally distributed observations from a population with mean μ and standard deviation σ, then the average $\bar{x}$ is normally distributed around μ with standard deviation $\sigma/\sqrt{n}$. A 95 percent confidence interval for μ is obtained as

$$\bar{x} + \sigma/\sqrt{n} \times N_{.025} \leq \mu \leq \bar{x} + \sigma/\sqrt{n} \times N_{.975} \tag{3.6}$$

where $N_{0.025}$ and $N_{0.975}$ are the 2.5 and 97.5 percentiles of the standard normal distribution, respectively.

You compute a 95 percent confidence interval about the population mean using the sample of June NAO values by typing

```
> xbar = mean(nao)
> sigma = sd(nao)
> n = length(nao)
> sem = sigma/sqrt(n)
> xbar + sem * qnorm(.025)
[1] -0.604
> xbar + sem * qnorm(.975)
[1] -0.161
```

This produces a 95% confidence interval for the population mean of $[-0.6, -0.16]$.

The normal distribution is symmetric so $-N_{0.025} = N_{0.975}$. You can verify this by typing

```
> -qnorm(.025); qnorm(.975)
[1] 1.96
[1] 1.96
```

Note that `qnorm(.025)` gives the left tail value of the normal distribution and that `qnorm(.975)` gives the right tail value.

Also the quantile for the standard normal is often written as $\Phi^{-1}(.975)$, where Φ is the notation for the cumulative distribution function of the standard normal. Thus it is common to write the confidence interval for the population mean as

$$\bar{x} \pm \sigma/\sqrt{n} \times \Phi^{-1}(.975) \tag{3.7}$$

Another application of the quantile function is to help assess the assumption of normality for a set of observed data. This is done by matching the empirical quantiles with the quantiles from the standard normal distribution. We give an example of this in Chapter 5.

3.2.9 Random Numbers

Random numbers are generated from algorithms. But the sequence of values appear as if they were drawn randomly. That is why they are sometimes referred to as "pseudo-random" numbers.

Random numbers are important in examining random variation. They are also important in simulating synthetic data that have the same statistical properties as your observations. This is useful when you want to know what effect your assumptions and approximations have on your results.

The distribution functions can generate random numbers (deviates) for you. The first argument specifies the number of random numbers to generate, and the subsequent arguments are the parameters of the distribution. For instance, to generate 10 random numbers from a standard normal distribution, type

```
> rnorm(10)
 [1] -0.1950 -1.0521 -0.5509  1.2252 -0.3236  1.6867
 [7]  0.3956 -0.2206 -0.0383 -1.3363
```

Your numbers will be different than those printed here since they are generated randomly. It is a good strategy to generate the same set of random numbers each time in an experimental setting. You do this by specifying a random number generator (RNG) and a seed value. If the RNG is the same and the seed value is the same, the set of random numbers will always be the same.

```
> set.seed(3042)
> rnorm(10)
 [1]  0.644 -0.461  1.400  1.123  0.908  0.320 -1.014
 [8] -0.241  0.523 -1.694
```

Here the `set.seed` function uses the default Mersenne Twister RNG with a seed value of 3042. Note that the commands must by typed in the same order; first set the seed, then generate the random numbers.

Specifying an RNG and a seed value allows your results to be replicated exactly. This is important when your results depend on a method that exploits random values as is the case with some of the Bayesian models you will consider in Chapters 4 and 12.

To simulate the next 20 years of Florida hurricane counts based on the counts over the historical record, you type

```
> rpois(20, lambda=mean(H$FL))
 [1] 1 1 1 1 3 1 0 0 0 0 0 1 1 0 1 0 1 0 0 0
```

To simulate the maximum wind speed (m s^{-1}) from the next 10 tropical cyclones occurring over the North Atlantic, you type

```
> rweibull(10, shape=2.5, scale=50)
 [1]  51.7  38.7  18.2  30.5  41.2  82.7  47.9 100.4
 [9]  26.1  21.3
```

Note that if your numbers are the same as those here, you continued with the same sequence of RNG initiated with the seed value provided earlier.

3.3 ONE-SAMPLE TEST

Inferential statistics refers to using your data to draw conclusions. Perhaps the simplest case involves testing whether your data support a particular mean value. The population mean is a model for your data and your interest is whether your single sample of values is consistent with this simple mean model.

The one-sample t (Student's t) test is based on the assumption that your data values $(x_i, \ldots, x_n)$ are independent and come from a normal distribution with a mean μ and variance σ^2. The shorthand notation is

$$x_i \sim \text{iid}\, N(\mu, \sigma^2) \tag{3.8}$$

where the abbreviation iid indicates "independent and identically distributed." You wish to test the *null hypothesis* that $\mu = \mu_0$.

You estimate the parameters of the normal distribution from your sample. The average $\bar{x}$ is an estimate of μ and the sample variance s^2 is an estimate of σ^2. It is important to keep in mind that you can never know the true parameter values. In statistical parlance, they are said to be fixed, but unknowable.

The key concept is that of the *standard error*. The standard error of the mean (or s.e.($\bar{x}$)) describes the variation in your average calculated from your n values. That is, suppose you had access to another set of n values (from the same set of observations or from the same experiment) and you again compute $\bar{x}$ from these values. This average will almost surely be different from the average calculated from the first set.

Statistics from different samples taken from the same population will vary. You can demonstrate this. First generate a population from a distribution with a fixed mean (3) and standard deviation (4).

```
> X = rnorm(2000, mean=3, sd=4)
```

Next take five samples each with a sample size of six and compute the average. This can be done using a `for` loop. The loop structure is `for(i in 1:n){cmds}`, where `i` is the loop index and `n` is the number of times the loop will execute the commands (`cmds`). Here your command is print the mean of a sample of six from the vector `X`.

```
> for(i in 1:5) print(mean(sample(X, size=6)))
[1] 1.68
[1] 1.34
[1] 3.05
[1] 4.35
[1] 1.01
```

The list of sample means are not all the same and not equal to three.

What happens to the variability in the list of means when you increase your sample size from 6 to 60? What happens when you increase the population standard deviation from 4 to 14? Try it.

The standard error of the mean is

$$\text{s.e.}(\bar{x}) = \frac{\sigma}{\sqrt{n}} \tag{3.9}$$

where σ is the population standard deviation and n is the sample size. Even with only a single sample, we can estimate s.e.$(\bar{x})$ by substituting the sample standard deviation s for σ.

The s.e.$(\bar{x})$ tells you how far the sample average may reasonably be from the population mean. With data that are normally distributed there is a 95 percent probability of the sample average staying within $\mu \pm 1.96\sigma$. Note how this is worded. It implies that if you take many samples from the population, computing the average for each sample, you will find that 95 percent of the samples have an average that falls within about two s.e.$(\bar{x})$s of the population mean.

The value of 1.96 comes from the fact that the difference in cumulative probability distribution from the standard normal between ± 1.96 is .95. To verify, type

```
> pnorm(1.96) - pnorm(-1.96)
[1] 0.95
```

With a bit more code, you can verify this for your sample of data saved in the object `X`. This time instead of printing five sample averages, you save 1,000 of them in an object called `Xb`. You then sum the number of TRUEs when logical operators are used to define the boundaries of the interval.

```
> set.seed(3042)
> X = rnorm(2000, mean=3, sd=4)
> Xb = numeric()
> for(i in 1:1000) Xb[i] = mean(sample(X, size=6))
> p = sum(Xb > 3 - 2 * 4/sqrt(6) &
+   Xb < 3 + 2 * 4/sqrt(6))/1000
> p
[1] 0.952
```

That is, for a given sample, there is a 95 percent chance that the interval defined by the s.e.($\bar{x}$) will cover the true (population) mean.

In the case of a one-sample test, you postulate a population mean and you have a single sample of data. For example, let μ_0 be a guess at the true mean, then you calculate the t statistic as

$$t = \frac{\bar{x} - \mu_0}{\text{s.e.}(\bar{x})} \tag{3.10}$$

With $(\bar{x} - \mu_0) = 2 \times \text{s.e.}(\bar{x})$, your t statistic is two. Your sample mean could be larger or smaller than μ_0 so t can be between -2 and $+2$ with $\bar{x}$ within 2 s.e.($\bar{x}$)s of μ_0.

If you have few data (less than about 30 cases), you need to correct for the fact that your estimate of s.e.($\bar{x}$) uses the sample standard deviation rather than σ. By using s instead of σ in Eq. 3.9, your chance of being farther from the population mean is larger. The correction is made by substituting the t-distribution (Student's t-distribution) for the standard normal distribution.

Like the standard normal, the t-distribution is continuous, symmetric about the origin, and bell-shaped. It has one parameter called the degrees of freedom (ν) that controls the relative "heaviness" of the tails. The degrees of freedom (d.f.) parameter $\nu = n - 1$, where n is the sample size. For small samples, the tails of the t-distribution are heavier than the tails of a standard normal distribution (see Fig. 3.5), meaning that it is more likely to produce values that fall far from the mean.

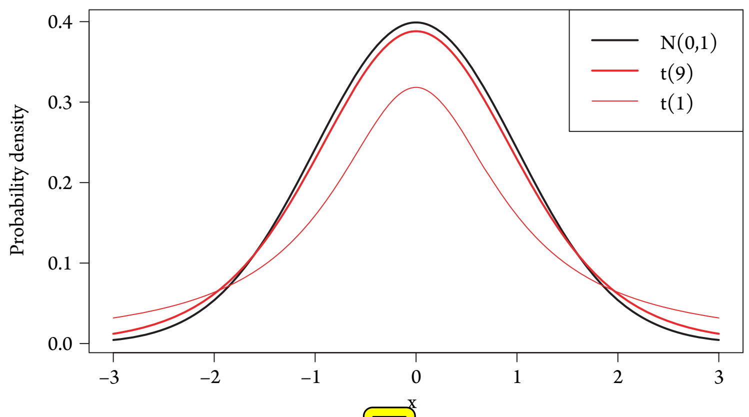

Figure 3.5 Probability density functions.

For instance, the difference in cumulative probabilities at the ±1.96 quantile values from a *t*-distribution with 9 d.f. is given by

```
> pt(q=1.96, df=9) - pt(q=-1.96, df=9)
[1] 0.918
```

This probability is smaller than that for the standard normal distribution. This indicates that only about 92 percent of the time the interval between ±1.96 will cover the true mean. Note in the figure how the *t*-distribution approximates the normal distribution as the sample size increases. For sample sizes of 30 or more, there is essentially no difference.

First, given a hypothesized population mean (μ_0), the *t* statistic is computed from your sample of data using Eq. 3.10. Next the cumulative probability of that value is determined from the *t*-distribution with d.f. equal to sample size minus one. Finally, the probability is multiplied by two to obtain the *p*-value. A small *p*-value leads to a rejection of the null hypothesis and a large *p*-value leads to a failure to reject the null hypothesis.

The *p*-value is an estimate of the probability that a particular result, or a result more extreme than the result observed, could have occurred by chance if the null hypothesis is true. In the present case, if the true mean is μ_0, what is the probability that your sample mean is as far or farther from μ_0 as it is? In short, the *p*-value is a measure of the credibility of the null hypothesis. The higher the *p*-value, the more credible the null hypothesis appears given your sample of data.

But the *p*-value, is best interpreted as evidence *against* the null hypothesis, thus a small value indicates evidence to reject the null. The interpretation is not black and white. A convenient way to express the evidence is given in Table 3.1.

To illustrate, consider the area-averaged North Atlantic SST values each August in units of degree cel as an example. Input the monthly data and save the values for August in a separate vector by typing

```
> SST = read.table("SST.txt", header=TRUE)
> sst = SST$Aug
```

Begin with a look at a summary table of these values.

Table 3.1 The *p*-Value as evidence against the null hypothesis.

p-*value Range*	*Evidence Against Null Hypothesis*
0–0.01	Convincing
0.01–0.05	Moderate
0.05–0.15	Suggestive, but inconclusive
>0.15	None

```
> summary(sst)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   22.6    23.0    23.1    23.1    23.3    23.9
   NA's
    5.0
```

The median temperature is 23.1°C with a maximum of maximum of 23.9°C over the 155 years. Note that the object `sst` has a length of 160 years, but the first 5 years have missing values.

You might be interested in the hypothesis that the values deviate significantly from 23.2°C. Although we spent some effort to explain the test statistic and procedure, the application is rather straightforward. The task for you is to test whether this distribution has a mean $\mu = 23.2$. Assuming the data come from a normal distribution, this is done using the `t.test` function as follows:

```
> t.test(sst, mu=23.2)
    One Sample t-test


data:  sst
t = -2.94, df = 154, p-value = 0.003759
alternative hypothesis: true mean is not equal to 23.2
95 percent confidence interval:
 23.1 23.2
sample estimates:
mean of x
     23.1
```

Here there are several lines of output. The output begins with a description of the test you asked for followed by the name of the data object used (here `sst`). The next line contains the value of the t-statistic (`t`) as defined in Eq. 3.10, degrees of freedom (`df`), and the p-value (`p-value`).

The degrees of freedom is the number of values in calculating the t-statistic that are free to vary. Here it is equal to the number of years (number of independent pieces of information) that go into calculating the t-statistic minus the number of statistics used in the intermediate steps of the calculation. Equation 3.10 shows only one statistic (the mean) is used, so the number of degrees of freedom is 155.

You do not need a table of the t-distribution to look at which quantiles the t-statistic belongs to. You can see that the p-value is 0.004 indicating conclusive evidence against the null hypothesis that the mean is 23.2. When the argument `mu=` is left off, the default is `mu=0`.

A sentence regarding the alternative hypothesis is printed next. It has two pieces of information: the value corresponding to your hypothesis and whether your test is one- or two-sided. Here it states `not equal to`, indicating a two-sided test.

You specify a one-sided test against the alternative of a larger μ by using the `alternative="greater"` argument. For instance, you might hypothesize the temperature to exceed a certain threshold value. Note that abbreviated argument names often work. For example, here it is okay to write `alt="g"` to get the one-sided, greater than, alternative.

The next output is the 95 percent confidence interval for the true mean. You can think of it as defining a set of hypothetical mean values, such that if they were used as values for your null hypothesis (instead of 23.2), they would lead to a *p*-value of 0.05 or greater (failure to reject the null). You can specify a different confidence level with the `conf.level` argument. For example, `conf.level=.99` will give you a 99 percent interval.

The final bit of output is the mean value from your sample. It is the best estimate for the true mean. Note that you are not testing the data. You are testing a hypothesis about some hypothetical value using your data.

To summarize, in classical statistical inference, you state a hypothesis that, for example, the population mean has a value equal to μ_0. You then use your data to see if there is evidence to reject it. The evidence is summarized as a *p*-value. A *p*-value less than 0.15 is taken as suggestive but inconclusive evidence that your hypothesis is wrong, while a *p*-value less than 0.01 is convincing evidence you are wrong. The larger the value of $|t|$, the smaller the *p*-value.

3.4 WILCOXON SIGNED-RANK TEST

Even if you cannot assume your data are sampled from a normal distribution, the *t*-test will provide a robust inference concerning the population mean. By robust we mean that the test results are not overly sensitive to departures from normality, especially in large samples. Keep in mind that the assumption of normality is about the distribution of the population of values, not just the sample you have.

The Wilcoxon signed-rank test does not require you to make the assumption of normality. First, the hypothesized value (μ_0) is subtracted from each observation. Next, absolute values of each difference is taken and sorted from smallest to largest. Ranks are assigned to each difference according to the sorting with the smallest difference given a rank of one. Then, the set of absolute magnitudes of the differences are ranked. Ranking is done by ordering from lowest to highest all magnitudes and counting the number of magnitudes with this value or lower. The lowest magnitude gets a rank of one.

The function `rank` is used to obtain ranks from a set of values. For example, type

```
> rank(c(2.1, 5.3, 1.7, 1.9))
[1] 3 4 1 2
```

The function returns the ranks with each value assigned a ranking from lowest to highest. Here the value of 2.1 in the first position of the data vector is ranked third and the value of 1.7 in the fourth position is ranked one.

Returning to your SST data, to see the ranks of the first 18 differences, type

```
> x = sst - 23.2
> r = rank(abs(x))
> r[1:18]
```

```
 [1] 156.0 157.0 158.0 159.0 160.0  34.0 112.0 124.0
 [9]  91.0 102.0  11.0 147.0 150.0  37.0  96.0   3.5
[17]  77.0  20.0
```

This says there are 156 years that have a difference (absolute value) less than or equal to the first year's SST value. By default ties are handled by averaging the ranks, so for an even number of ties, the rank are expressed as a fractional half, otherwise they are a whole number.

The test statistic (V) is the sum of the ranks corresponding to the values that are above the hypothesized mean

```
> sum(r[x > 0], na.rm=TRUE)
[1] 4224
```

Assuming only that the distribution is symmetric around μ_0, the test statistic corresponds to selecting each rank from 1 to n with a probability of 10.5 and calculating the sum. The distribution of the test statistic can be calculated exactly, but becomes computationally prohibitive for large samples. For large samples, the distribution is approximately normal.

The application of the nonparametric Wilcoxon signed-rank test in R is done in the same way as the *t*-test. You specify the data values in the first argument and the hypothesized population mean in the second argument.

```
> wilcox.test(sst, mu=23.2)
    Wilcoxon signed rank test with continuity
    correction

data:  sst
V = 4170, p-value = 0.001189
alternative hypothesis: true location is not equal
   to 23.2
```

The *p*-value of 0.0012 indicates moderate evidence against the null hypothesis, which is somewhat greater evidence than that with the *t*-test.

There is less output as there is no parameter estimate and no confidence limits, although it is possible under some assumptions to define a location measure and confidence intervals for it (Dalgaard, 2002). The continuity correction refers to a small adjustment to the test statistic when approximating the discrete ranks with a continuous (normal) distribution. Note that the test statistic is slightly different than what you calculated preiviously. See the help file for additional details.

Although a nonparametric alternative to a parametric test can be valuable, caution is advised. If the assumptions are met, then the *t*- test will be more efficient by about 5 percent relative to the nonparametric Wilcoxon test. That is, for a given sample size, the *t*-test better maximizes the probability that the test will reject the null hypothesis when it is false. That is, the *t*-test has more power than the Wilcoxon test. However, in the presence of outliers, the nonparametric Wilcoxon test will less likely indicate spurious significance compared with the parametric t-test.

The Wilcoxon test has problems when there are ties in the ranks for small samples. By default (if exact is not specified), an exact p-value is computed if the samples contain less than 50 values and there are no ties. Otherwise a normal approximation is used.

3.5 TWO-SAMPLE TEST

It is often useful to compare two samples of climate data. For instance, you might be interested in examining whether El Niño influences hurricane rainfall. Here you would create two samples with one sample containing hurricane rainfall during years denoted as El Niño, and the other sample containing hurricane rainfall during all other years.

The two-sample t-test is used to test the hypothesis that two samples come from distributions with the same population mean. The theory is the same as that employed in the one-sample test. Vectors are now doubly indexed ($x_{1,1},\ldots,x_{1,n_1}$ and $x_{2,1},\ldots,x_{2,n_2}$). The first index identifies the sample and the second identifies the case. The assumption is that the values follow normal distributions $N(\mu_1,\sigma_1^2)$ and $N(\mu_2,\sigma_2^2)$, and your interest is to test the null hypothesis $\mu_1 = \mu_2$. You calculate the t-statistic as

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\text{s.e.}(\bar{x}_1)^2 + \text{s.e.}(\bar{x}_2)^2}} \tag{3.11}$$

where the denominator is the standard error of the difference in means and $\text{s.e.}(\bar{x}_i) = s_i/\sqrt{n_i}$. If you assume the two samples have the same variance ($s_1^2 = s_2^2$), then you calculate the s.e.$(\bar{x})$s using a single value for s based on the standard deviation of all values over both samples. Under the null hypothesis that the population means are the same, the t-statistic will follow a t-distribution with $n_1 + n_2 - 2$ degrees of freedom.

If you do not assume equal variance, the t-statistic is approximated by a t-distribution after adjusting the degrees of freedom by the Welch procedure. By default, the function uses the Welch procedure resulting in a noninteger degrees of freedom. Regardless of the adjustment, the two-sample test will usually give about the same result unless the sample sizes and the standard deviations are quite different.

As an example, suppose you are interested in whether the June NAO values have mean values that are different depending on hurricane activity along the Gulf coast later in the year. First create two samples of NAO values. The first sample contains June NAO values in years with no Gulf hurricanes and the second sample contains June NAO values in years with at least two Gulf hurricanes. This is done using the `subset` function.

```
> nao.s1 = subset(nao, H$G == 0)
> nao.s2 = subset(nao, H$G >= 2)
```

You then summarize the two sets of values with the mean, standard deviation, and sample size.

```
> mean(nao.s1); sd(nao.s1); length(nao.s1)
[1] -0.277
[1] 1.51
[1] 80
> mean(nao.s2); sd(nao.s2); length(nao.s2)
[1] -1.06
[1] 1.03
[1] 22
```

The mean NAO value is larger during inactive years, but is it significantly larger? The standard deviation is also larger and so is the sample size.

Your null hypothesis is that the population mean of June NAO values during active Gulf years is equal to the population mean of June NAO values during inactive years. The test is performed with the `t.test` function with the two data vectors as the two arguments. By default, the function uses the Welch procedure.

```
> t.test(nao.s1, nao.s2)
    Welch Two Sample t-test

data:  nao.s1 and nao.s2
t = 2.82, df = 48.7, p-value = 0.007015
alternative hypothesis: true difference in means is not
   equal to 0
95 percent confidence interval:
 0.223 1.337
sample estimates:
mean of x mean of y
   -0.277    -1.057
```

The output is similar to that from the one-sample test provided earlier. The type of test and the data objects are given in the preamble. The value of the *t*-statistic, the degrees of freedom, and the *p*-value follow.

Here you find a *t*-statistic of 2.815. Assuming that the null hypothesis of no difference in population means is correct, this *t* value (or a value larger in magnitude) has a probability 0.007 of occurring by chance given a *t*-distribution with 48.67 degrees of freedom. Thus, there is compelling evidence that June NAO values are different between the two samples.

As with the one-sample test, the alternative hypothesis, which is that the true difference in means is not equal to zero, is stated as part of the output. This is the most common alternative in these situations.

The confidence interval (CI) refers to the difference in sample means (mean from sample 1 minus mean from sample 2). So you state that the difference in sample means is 0.78 [(0.22, 1.34), 95% CI]. The interval does not include zero consistent with the conclusion from the test statistic and the corresponding *p*-value of compelling evidence against the null hypothesis (less than the 5% significance level).

If you are willing to assume the variances are equal (for example, both samples come from the same population), you can specify the argument `var.equal=T`. In this case, the number of degrees of freedom is a whole number, the *p*-value is larger, and the confidence interval is wider.

3.6 STATISTICAL FORMULA

Instead of creating subsets of the object `nao` based on values in the object `H`, you can create a data frame with two parallel columns. Include all values for the NAO in one column and the result of a logical operation on Gulf hurricane activity in a separate column.

```
> gulf = H$G > 1
> nao.df = data.frame(nao, gulf)
> tail(nao.df)
       nao  gulf
155 -1.00  TRUE
156 -0.41 FALSE
157 -3.34 FALSE
158 -2.05  TRUE
159 -3.05 FALSE
160 -2.40 FALSE
```

This displays the NAO values and whether or not there was two or more Gulf hurricanes in corresponding years.

The goal is to see whether there is a shift in the level of the NAO between the two groups of hurricane activity years (`TRUE` and `FALSE`). Here the groups are years with two or more Gulf hurricanes (TRUE) and years with one or fewer hurricanes (FALSE).

With this setup, you specify a two-sample *t*-test using the tilde (~) operator as

```
> t.test(nao ~ gulf, data=nao.df)
    Welch Two Sample t-test

data:  nao by gulf
t = 3.1, df = 36, p-value = 0.00373
alternative hypothesis: true difference in means is not
   equal to 0
95 percent confidence interval:
 0.271 1.293
sample estimates:
mean in group FALSE  mean in group TRUE
             -0.275              -1.057
```

The object to the left of the tilde is the variable you want to test and the object to the right is the variable used for testing. The tilde is read as "described by" or "conditioned on." That is, the June NAO values are described by Gulf coast hurricane activity. This is how statistical models are specified in R. You will see this model structure throughout the book. Note that by using the `data=nao.df,` you can refer to the column vectors in the data frame by name in the model formula.

The conclusion is the same. Years of high and low Gulf hurricane activity appear to be presaged by June NAO values that are significantly different. The output is essentially the same although the group names are taken from the output of the logical operation. Here `FALSE` refers to inactive years.

3.7 TWO-SAMPLE WILCOXON TEST

If the normality assumption is suspect or sample sizes are small, you might prefer a nonparametric test for differences in the mean. As with the one-sample Wilcoxon test, the two-sample counterpart is based on replacing your data values by their corresponding rank. This is done without regard to group. The test statistic W is then computed as the sum of the ranks in one group.

The function is applied using the model structure as

```
        Wilcoxon rank sum test with continuity
        correction

data:  nao by gulf
W = 2064, p-value = 0.006874
alternative hypothesis: true location shift is not
     equal to 0
```

The results are similar to those found using the t-test and are interpreted as convincing evidence of a relationship between late spring NAO index values and hurricane activity along the Gulf coast of the United States.

3.8 COMPARE VARIANCES

It is not necessary to assume equal variances when testing for differences in means. Indeed, this is the default option with the `t.test` function. Yet your interest could be whether the variability is changing. For instance, you might speculate that the variability in hurricane activity will increase with global warming.

Note that the variance is strictly positive. Given two samples of data, the ratio of variances will be unity if the variances are equal. Under the assumption of equal population variance, the F-statistic, as the ratio of the sample variances, has an F-distribution with two parameters. The parameters are the two sample sizes minus one.

The F-distribution is positively skewed meaning the tail on the right is longer than the tail on the left. Figure 3.6 shows the probability density for two F-distributions. A larger sample size results in a density centered on one and more symmetric.

The function `var.test` is called in the same way as the function `t.test`, but performs an F test on the equating of the group variances.

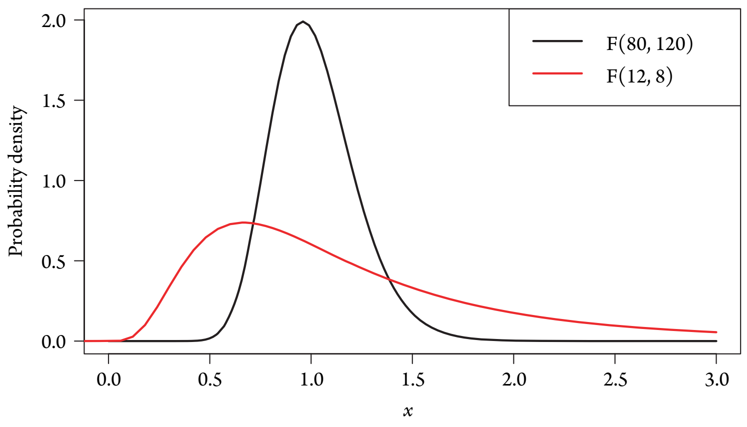

Figure 3.6 Probability density functions for an *F*-distribution.

```
> var.test(nao ~ gulf, data=nao.df)
    F test to compare two variances

data:  nao by gulf
F = 2, num df = 137, denom df = 21, p-value =
0.06711
alternative hypothesis: true ratio of variances is not
     equal to 1
95 percent confidence interval:
 0.95 3.58
sample estimates:
ratio of variances
                 2
```

Results show an *F*-statistic of 2 with degrees of freedom equal to 137 and 21 resulting in a *p*-value of 0.067 under the null hypothesis of equal variance. The magnitude of the *p*-value provides suggestive but inconclusive evidence of a difference in population variance. Note that the 95 percent confidence interval on the *F*-statistic includes the value of one as you would expect given the *p*-value.

The *F*-test is sensitive to departures from normality. Also, for small data sets, the confidence interval will be quite wide (see Fig. 3.6) often requiring you to take the assumption of equal variance as a matter of belief.

3.9 CORRELATION

Correlation extends the idea of comparing one variable in relation to another. Correlation indicates the amount and the direction of association between two variables. If hurricanes occur more often when the ocean is warmer, then you say that ocean temperature is positively correlated with hurricane incidence as one goes up, the other goes up. If hurricanes occur less often when sun spots are numerous, then you say that sun spots are inversely correlated with hurricane incidence. Meaning the two variables go in opposite direction as one goes up, the other goes down.

A correlation coefficient is a symmetric scale-invariant measure of the correlation between two variables. It is symmetric because the correlation between variables x and y is the same as the correlation between variables y and x. It is scale-invariant because the value does not depend on the units of either variable.

Correlation coefficients range from -1 to $+1$, where the extremes indicate perfect correlation and 0 means no correlation. The sign is negative when large values of one variable are associated with small values of the other and positive if both tend to be large or small together. Different metrics of correlation lead to different correlation coefficients. Most common is Pearson's product-moment correlation coefficient followed by Spearman's rank and Kendall's τ.

3.9.1 Pearson's Product-Moment Correlation

Pearson's product-moment correlation coefficient is derived from the bivariate normal distribution of two variables, where the theoretical correlation describes contour ellipses about the two-dimensional densities. It is the workhorse of climatological studies. If both variables are scaled to have unit variance, then a correlation of zero corresponds to circular contours and a correlation of one corresponds to a line segment.

Figure 3.7 shows two examples: one where the variables x and y have a small positive correlation and the other where they have a fairly large negative correlation. The points are generated from a sample of bivariate normal values with a Pearson product-moment correlation of 0.2 and -0.7. The contours enclose the 75 and 95 percent probability region for a bivariate normal distribution with mean of zero, unit variances, and corresponding correlations.

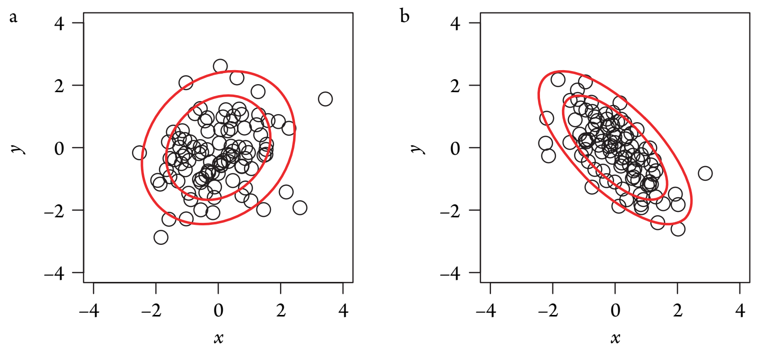

Figure 3.7 Scatter plots of correlated variables with (a) $r = 0.2$ and (b) $r = -0.7$.

The Pearson correlation coefficient between x and y is

$$r(x,y) = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}} \tag{3.12}$$

The Pearson correlation is often called the "linear" correlation since the absolute value of r will be one when there is a perfect linear relationship between x_i and y_i.

The function `cor` is used to compute the correlation between two or more vectors. For example, to get the linear correlation between the May and June values of the NAO, type

```
> cor(NAO$May, NAO$Jun)
[1] 0.0368
```

The value indicates weak positive correlation. Note that the order of the vectors in the function is irrelevant as $r(x,y) = r(y,x)$. You can verify in this case by typing

```
> cor(NAO$May, NAO$Jun) == cor(NAO$Jun, NAO$May)
[1] TRUE
```

If there are missing values, the function will return `NA`. The argument `na.rm=TRUE` works for one-vector functions like `mean`, `sd`, `max`, and others to indicate that missing values should be removed before computation. However, with `cor` function there are additional ways to handle the missing values, so you need the `use` argument. As an example, to handle the missing values in the SST data frame by case-wise deletion, type

```
> cor(SST$Aug, SST$Sep, use="complete.obs")
[1] 0.944
```

Here the value indicates strong positive correlation between August and September SST.

This value of r estimated from the data is a random variable and is thus subject to sampling variation. For instance, adding another year's worth of data will result in an r value that is somewhat different. Typically, your hypothesis is that the population correlation is zero. As might be guessed from the differences in r, your conclusions about this hypothesis will likely be different for the SST and NAO data.

You can ask the question differently. For example, in 1,000 samples of x and y each of size 30 from a population with zero correlation, what is the largest value of r? You answer this question using simulations by typing

```
> set.seed(3042)
> n = 30
> cc = numeric()
> for(i in 1:1000){
+   x = rnorm(n); y = rnorm(n)
+   cc[i] = cor(x, y)
+ }
> mean(cc); max(cc)
[1] -0.0148
[1] 0.569
```

The variable `n` sets the sample size and you simulate 1,000 different correlation coefficients from different independent samples of x and y. The average correlation is close to zero as expected, but the maximum correlation is quite large.

High correlation can arise by chance. Thus when you report a correlation coefficient, a CI on your estimate or a test of significance should be included. This is done with the `cor.test` function. The test is based on transforming r to a statistic that has a t-distribution using

$$t = \sqrt{\nu}\frac{r}{\sqrt{1-r^2}} \tag{3.13}$$

where $\nu = n - 2$ is the degrees of freedom and n is the sample size.

Returning to the NAO example, to obtain a confidence interval on the correlation between the May and June values of the NAO and a test of significance, type

```
> cor.test(NAO$May, NAO$Jun)
    Pearson's product-moment correlation

data:  NAO$May and NAO$Jun
t = 0.463, df = 158, p-value = 0.6439
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.119  0.191
sample estimates:
   cor
0.0368
```

The type of correlation and the data used in the test are output in the preamble. The correlation of 0.037 (given as the last bit of output) is transformed to a t value of 0.463 with 158 degrees of freedom providing a 95 percent CI of $[-0.119, 0.191]$. This means that if the procedure used to estimate the CI was repeated 100 times, 95 of the intervals would contain the true correlation coefficient. The output also gives a p-value of 0.644 as evidence in support of the null hypothesis of no correlation.

Repeat this example using the January and September values of SST. What is the CI on the correlation estimate? How would you describe the evidence against the null hypothesis of zero correlation in this case?

3.9.2 Spearman's Rank and Kendall's τ Correlation

Inferences based on the Pearson correlation assume the variables are adequately described by normal distributions. An alternative is Spearman's rank (ρ) correlation, which overcomes the effect of outliers and skewness by considering the rank of the data rather than the magnitude. The Spearman correlation coefficient is defined as the Pearson correlation coefficient between the ranked variables.

The Pearson correlation is the default in the `cor.test` function. You change this with the `method` argument. To obtain Spearman's rank correlation and the associated test of significance, type

```
> cor.test(H$G, H$FL, method="spearman")
    Spearman's rank correlation rho

data:  H$G and H$FL
S = 551867, p-value = 0.01524
alternative hypothesis: true rho is not equal to 0
sample estimates:
  rho
0.192
```

The correlation is 0.192 with a p-value of 0.015, providing suggestive evidence against the null hypothesis of no correlation.

Kendall's τ is another alternative to Pearson correlation. It is based on counting the number of concordant and discordant point pairs from your data. For two data vectors x and y each of length n, a point at location i is given in two-dimensional space as (x_i, y_i). A point pair is defined as $[(x_i, y_i); (x_j, y_j)]$ for $i \neq j$.

A point pair is concordant if the difference in the x values is of the same sign as the difference in the y values, otherwise it is discordant. The value of Kendall's τ is the number of concordant pairs minus the number of discordant pairs divided by the total number of unique point pairs, which is $n(n-1)/2$ where n is the sample size. For a perfect correlation, either all point pairs are concordant or all pairs are discordant. Under zero correlation, there are as many concordant pairs as discordant pairs.

Repeat the call to function `cor.test` on coastal hurricane activity, but now use the `kendall` method and save the resulting estimate.

```
> x = cor.test(H$G, H$FL, method="kendall")
> tau.all = x$estimate
```

The correlation is 0.174 with a p-value of 0.015, again providing suggestive evidence against the null hypothesis of no correlation.e

3.9.3 Bootstrap Confidence Intervals

Kendall's τ and Spearman's rank correlations are computed without confidence intervals. You should always report a CI. In this case, you use a procedure called *bootstrapping*, which is a resampling technique.

The idea is to sample the values from your data with replacement using the `sample` function. The sample size is the size of your data. The bootstrap sample is called a replicate. You compute the statistic of interest using the values of your replicate. The bootstrap value will be different than the value computed from your data because the replicate contains repeat and data values and not all data values are included. You repeat the procedure many times collecting all the bootstrap statistic values. You then use the `quantile` function to determine the lower and upper quantiles corresponding to the 0.025 and 0.975 probabilities.

Bootstrapping is widely used for assigning measures of accuracy to sample estimates (Efron and Tibshirani, 1986). The function `boot` from the package **boot** generates bootstrap replicates of any statistic applied to your data. It has options for parametric and nonparametric resampling.

For example, to implement a bootstrap procedure for Kendall's τ using data on Florida and Gulf coast hurricane frequencies, you first create a function as follows:

```
> mybootfun = function(x, i){
+   Gbs = x$G[i]
+   Fbs = x$FL[i]
+ return(cor.test(Gbs, Fbs,method="k")$est)
+ }
```

Your function has two variables: the data (`x`) and an index variable (`i`). Next you generate 1,000 bootstrap samples and calculate the CIs by typing

```
> require(boot)
> tau.bs = boot(data=H, statistic=mybootfun, R=1000)
> ci = boot.ci(tau.bs, conf=.95)
```

The `boot` function must be run prior to running the `boot.ci` function. The result is a 95 percent CI of (0.032, 0.314) about the estimated τ of 0.174.

3.9.4 Causation

If you compute the correlation between June SST and U.S. hurricane counts using Kendall's method, you find a positive value for τ of 0.151 with a p-value of 0.011 indicating suggestive, but inconclusive, evidence against the null hypothesis of no correlation.

A positive correlation between ocean warmth and hurricane activity does not prove causality. Moreover, since the association is symmetric, it does not say that x causes y any more than it says y causes x. This is why you frequently hear "correlation does not equal causation." The problem with this adage is that it ignores the fact that correlation is needed for causation. It is necessary, but insufficient. When correlation is properly interpreted, it is indispensable in the study of hurricanes and climate.

Your correlation results are more meaningful if you explain how the variables are physically related. Particularly if your explanation comes before you look at your data. Studies showing a consistent correlation between two variables using different time and space scales, and over different time periods and different regions, provide greater evidence of an association than a single study. However, if you want proof that a single factor causes hurricanes, then correlation is not enough.

3.10 LINEAR REGRESSION

Correlation is the most widely used statistic in climatology, but linear regression is arguably the most important statistical model. When you say that variable x has a linear relationship to variable y, you mean $y = a + bx$, where a is the intercept and b is the slope of the line. You call x the independent variable and y the dependent variable because the value of y depends on the value of x.

But in statistics, you do not assume these variables have a perfect linear relationship. Instead, in describing the relationship between two random vectors x_i and y_i, you add an error term (ε) to the equation such that

$$y_i = \alpha + \beta x_i + \varepsilon_i \tag{3.14}$$

You assume the values ε_i are iid $N(0, \sigma^2)$.

The slope of the line is the regression coefficient β, which is the increase in the *average* value of y per unit change in x. The line intersects the y-axis at the *intercept* α. The vector x is called the explanatory variable and the vector y is called the response variable.

Equation 3.14 describes a regression of the variable y *onto* the variable x. This is always the case. You regress your response variable onto your explanatory variable(s), where the word "regression" refers to a model for the mean of the response variable.

The model consists of three parameters α, β, and σ^2. For a set of explanatory and response values, the parameters are estimated using the *method of least squares*. The method finds a set of α and β values that minimize the sum of squared residuals given as

$$\text{SS}_{res} = \sum_i [y_i - (\alpha + \beta x_i)]^2 \tag{3.15}$$

The solution to this minimization is a set of equations given by

$$\hat{\beta} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2} \tag{3.16}$$

$$\hat{\alpha} = \bar{y} - \hat{\beta}\bar{x} \tag{3.17}$$

that define estimates for α and β. The residual variance is $\text{SS}_{res}/(n-2)$, where $\hat{\alpha}$ and $\hat{\beta}$ are used in Eq. 3.15.

The regression line is written as

$$\hat{y}_i = \hat{\alpha} + \hat{\beta} x_i \tag{3.18}$$

The method of least squares to determine the $\hat{\alpha}$ and $\hat{\beta}$ values is performed by the function `lm` (linear model). If you are interested in regressing the August values of Atlantic SST onto the preceding January SST values, you type

```
> lm(SST$Aug ~ SST$Jan)
Call:
lm(formula = SST$Aug ~ SST$Jan)
```

```
Coefficients:
(Intercept)       SST$Jan
       7.11          0.84
```

The argument to `lm` is a model formula where the tilde symbol ($\sim$) as we have seen is read as "described by." Or to be more complete in this particular case, we state that the August SST values are *described by* the January SST values *using a linear regression model*. In this case, you have a single explanatory variable so the formula is simply `y~x` and you call the model *simple* linear regression. In the next section, you will see that with two covariates the model is `y~x+z`.

The response variable is the variable you are interested in modeling. This is crucial. You must decide which variable is your response variable before hard. Unlike correlation, a regression of y onto x is not the same as a regression of x onto y. Your choice depends on the question you want to be answered. You get no guidance by examining your data nor will R tell you. Here you choose August SST as the response since it is natural to consider using information from an earlier month to predict what might happen in a later month.

The output from `lm` includes a preamble of the call that includes the model structure and the data used. Parameter estimates are given in the table of coefficients. The estimated intercept value ($\hat{\alpha}$) is given under `(Intercept)` and the estimated slope value ($\hat{\beta}$) under `SST$Jan`. The output allows you to state that the best-fitting straight line (regression line) is defined as August SST = 7.11 + 0.84 $\times$ January SST.

The units on the intercept parameter are the same as the units of the response variable, here $^\circ$C. The units on the slope parameter are the units of the response divided by the units of the explanatory variable, here $^\circ$C per $^\circ$C. Thus you interpret the slope value in this example as follows: for every 1°C increase in January SST, the August SST increases on average by 0.84°C.

The slope and intercept values will deviate somewhat from the true values due to sampling variation. One way to examine how much deviation is to take samples from the data and, with each sample, use the `lm` function to determine the parameter. The code below does this for the slope parameter using January SST values as the explanatory variable and August SST values as the response.

```
> sl = numeric()
> for (i in 1:1000) {
+   id = sample(1:length(SST$Aug), replace=TRUE)
+   sl[i] = lm(SST$Aug[id] ~ SST$Jan[id])$coef[2]
+ }
> round(quantile(sl), digits=2)
  0%  25%  50%  75% 100%
0.52 0.78 0.83 0.89 1.06
```

Note, you sample from the set of row indices and use the same index for the January and the August values. Results indicate that 50 percent of the slopes fall between the values 0.78 and 0.89°C per $^\circ$C.

Although illustrative, sampling is not really needed. Recall you calculated the s.e.($\bar{x}$) from a single sample to describe the variability of the sample mean. You do the same to calculate the standard error of the slope (and intercept) from a sample of x and y values. These standard errors, denoted s.e.($\hat{\beta}$) and s.e.($\hat{\alpha}$), are used for inference and to compute CI.

Typically the key inference is a test of the null hypothesis that the population value for the slope is zero. A zero slope implies that the line is horizontal and there is no relationship between the response and explanatory variables. The test statistic in this case is

$$t = \frac{\hat{\beta}}{\text{s.e.}(\hat{\beta})} \tag{3.19}$$

which follows a t distribution with $n-2$ degrees of freedom if the true slope is zero. Similarly you can test the null hypothesis that the intercept is zero, but this often has little physical meaning because it typically involves an extrapolation outside the range of your x values.

The value for the test statistic (t value) is not provided as part of the raw output from the `lm` function. The result of `lm` is a *model* object. This is a key concept. In Chapter 2, you encountered data objects. You created structured data vectors and input data frames from a spreadsheet. The saved objects are listed in your working session by typing `objects()`. Functions, like `table`, are applied to these objects to extract information.

In the same way, a model object contains a lot of information. This information is extracted using functions. An important extractor function is the `summary`. You saw previously that applied to a data frame object, the `summary` function extracts statistics about the values in each column. When applied to a model object, it extracts information about the model.

For instance, to obtain the information about the regression model of August SST onto January SST, type

```
> summary(lm(SST$Aug ~ SST$Jan))
Call:
lm(formula = SST$Aug ~ SST$Jan)

Residuals:
    Min      1Q  Median      3Q     Max
-0.4773 -0.1390 -0.0089  0.1401  0.4928

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    7.1117     1.4597    4.87  2.7e-06
SST$Jan        0.8400     0.0765   10.98  < 2e-16

Residual standard error: 0.197 on 153 degrees of freedom
  (5 observations deleted due to missingness)
```

```
Multiple R-squared: 0.441,  Adjusted R-squared: 0.437
F-statistic:  121 on 1 and 153 DF,  p-value: <2e-16
```

The output includes the call function, summary of residuals, table of coefficients, residual standard error, R^2 values, and the F-statistic and the associated p-value. The output starts with a repeat of the function call. This is useful if the object is saved and examined later.

Summary statistics on the set of model residuals follow. A residual is the difference between the response value at a particular explanatory value and the modeled value. Residuals are important in helping you diagnose how well your model fits the data. The average of the residuals is zero by definition of least squares, so the median should not be far from zero and the minimum and maximum should roughly be equal in absolute magnitude. This is true here for the first (`1Q`) and third (`3Q`) quartile values.

Next, the table of coefficients shows the intercept and slope values as in the raw output, but here the accompanying standard errors, t values, and p-values are also provided. The output is in tabular form with the each row corresponding to a separate parameter. The first row is the intercept and the second row is the slope associated with the explanatory variable (`SST$Jan`). The vector `SST$Jan` is the explanatory variable and the slope of 0.84 is the amount of change in the mean response associated with a unit change in the explanatory variable.

The first labeled column is the sample estimate (`Estimate`), the second is the standard error (`Std. Error`), the third is the t value (`t value`), and the fourth is the p-value (`Pr(>|t|)`). Note that according to Eq. 3.19, the t value is the ratio of the estimated value to its standard error. The p-value is the probability of finding a t value as large or larger (in absolute value) by chance assuming the slope is zero. Here the p-value on the January SST slope is less than 2×10^{-16}, which is output in exponential notation as `2e-16`. This indicates a near-zero chance of no relationship between January SST and August SST given your sample of data.

By default, symbols are placed to the right of the p-values as indicators of the level of significance, and a line below the table provides the definition. Here we turned them off using an argument in the `options` function. The p-value should always be reported rather than simply providing a cateorical signifiance level.

Note that the interpretation of a p-value as evidence in support of the null hypothesis is the same as the t-test you encountered earlier. Your job is to determine the null hypothesis. In the context of regression, the assumption of no relationship between the explanatory and response variables is typically your null hypothesis. Therefore, a low p-value indicates evidence of a relationship between your explanatory and response variables.

Continuing with the output, the residual standard error quantifies the variation of the observed values about the regression line. It is computed as the square root of the sum of the squared residuals divided by the square root of the degrees of freedom. The degrees of freedom is the sample size minus the number of coefficients. It provides an estimate of the model parameter σ.

Next are the R-squared values. The "multiple R squared," is the proportion of variation in the response variable that can be explained by the explanatory variable. So here you state that the model explains 44.1 percent of the variation in August SST values. With only a single explanatory variable (simple linear regression), the multiple R squared is equal to the square of the Pearson correlation coefficient, which you verify by typing

```
> cor(SST$Jan, SST$Aug, use="complete")^2
[1] 0.441
```

The adjusted R squared ($\bar{R}^2$) is a modification to the multiple R squared for the number of explanatory variables. The adjusted R squared increases only if the new variable improves the model. It can be negative and will always be less than or equal to the multiple R squared. It is defined as

$$\bar{R}^2 = 1 - (1 - R^2)\frac{n-1}{n-p-1} \tag{3.20}$$

where n is the sample size and p is the number of explanatory variables. In small samples with many explanatory variables, the difference between R^2 and $\bar{R}^2$ will be large.

The final bit of output is related to an F test, which is a test concerning the entire model. The output includes the F statistic, the degrees of freedom (in this case, two of them), and the corresponding p-value as evidence in support of the null hypothesis that the model has no explanatory power. In the case of simple regression, it is equivalent to the test on the slope so it is only interesting when there is more than one explanatory variable. Note that the F statistic is equal to the square of the t statistic, which is true of any linear regression model with one explanatory variable.

Other extractor functions provide useful information about your model. The function `resid` takes a model object and extracts the vector of residual values. For example, type

```
> lrm = lm(Aug ~ Jan, data=SST)
> resid(lrm)[1:10]
      6       7       8       9      10      11
-0.0629 -0.2690  0.0278  0.0892 -0.1426  0.2165
     12      13      14      15
-0.3297 -0.2614  0.4356 -0.2032
```

First the model object is saved with name `lrm`. Here only the column names are referenced in the model formula because you specify the data frame with the `data` argument. Then the extractor function `resid` lists the residuals. Here using the subset function, you list only the first 10 residuals.

Similarly, the function `fitted` computes the mean response value for each value of the explanatory variable. For example, type

```
> fitted(lrm)[1:10]
   6    7    8    9   10   11   12   13   14   15
23.2 23.2 22.8 22.9 23.1 23.0 23.0 22.9 22.8 23.2
```

These fitted values lie along the regression line and are obtained by solving for $\hat{y}$ in Eq. 3.18.

Note that the residuals and fitted values are labeled with the row numbers of the SST data frame. In particular, note that they do not contain rows 1 through 5, which are missing in the response and explanatory variable columns.

Your model can be used to make predictions. The `predict` function is similar to the `fitted` function but allows you to predict values of the response for arbitrary values of the explanatory variable. The caveat is that you need to specify the explanatory values as a data frame using the `newdata` argument. For example, to make an SST prediction for August given a January value of 19.4°C, type

```
> predict(lrm,newdata=data.frame(Jan=19.4))
   1
23.4
```

A note on terminology. The word "predictor" is the generic term for an explanatory variable in a statistical model. A further distinction is sometimes made between covariates, which are continuous-valued predictors and factors, which can take on only a few values that may or may not be ordered.

A prediction is not worth much without an estimate of uncertainty. Assuming the model is correct, a predicted value from a statistical model has two sources of uncertainty. One is the uncertainty about the mean of the response *conditional* on the value of the explanatory variable. It is the precision with which the conditional mean is known. It is known as a *confidence interval*. To obtain the CI on the predicted value, type

```
> predict(lrm, data.frame(Jan=19.4), int="c")
   fit  lwr  upr
1 23.4 23.3 23.5
```

The argument `int="c"` tells the function `predict` to provide a CI on the predicted value. The output includes the predicted value in the column labeled `fit` and the lower and upper confidence limits in the columns `lwr` and `upr`, respectively. By default, the limits define the 95 percent CI. This can be changed with the `level` argument.

The interpretation is the same as before. Given the data and the model, there is a 95 percent chance that the interval defined by the limits will cover the true (population) mean when the January SST value is 19.4°C. The other source of uncertainty arises from the distribution of a particular value given the conditional mean. That is, even if you know the conditional mean exactly, the distribution of particular values about the mean will have a spread.

The *prediction interval* provides a bound on a set of new values from the model that contains both sources of uncertainty. As a consequence, for a given confidence level, the prediction interval will always be wider than the CI. The prediction interval relies on the assumption of normally distributed errors with a constant variance across the values of the explanatory variable.

To obtain the prediction interval on the predicted value, type

```
> predict(lrm, data.frame(Jan=19.4), int="p")
   fit lwr  upr
1 23.4  23 23.8
```

Given the data and the model, there is a 95 percent chance that the interval defined by these limits will cover any future values of August SST given that the January SST value is 19.4°C.

3.11 MULTIPLE LINEAR REGRESSION

Multiple regression extends simple linear regression by allowing more than one explanatory variable. Everything from simple regression carries over. Each additional explanatory variable contributes a new term to the model. However, an issue now arises because of possible relationships between the explanatory variables.

As an illustration, we continue with a model for predicting August SST values over the North Atlantic using SST values from earlier months. Specifically, for this example, you are interested in making predictions with the model at the end of March. You have January, February, and March SST values plus Year as the set of explanatory variables.

The first step is to plot your response and explanatory variables. This is done with the `pairs` function. By including the `panel.smooth` function as the argument to `panel`, a local smoother is used on the set of points that allows you to more easily see possible relationships. Smoothing is discussed in more detail in Chapter 5. Here you specify the August values (column 9 in `SST`) to be plotted in the first row (and column) followed by year and then the January through March values.

```
> pairs(SST[, c(9,1:4)], panel=panel.smooth)
```

The scatter plots are arranged in a two-dimensional matrix (Fig. 3.8). The response variable is August SST and the four explanatory variables include Year, and the SST values during January, February, and March. A locally weighted polynomial smoother with a span of 67 percent of the points is used to draw the red lines.

The diagonal elements of the matrix are the variable labels. The plot in the first row and second column is the August SST values on the vertical axis and the year on

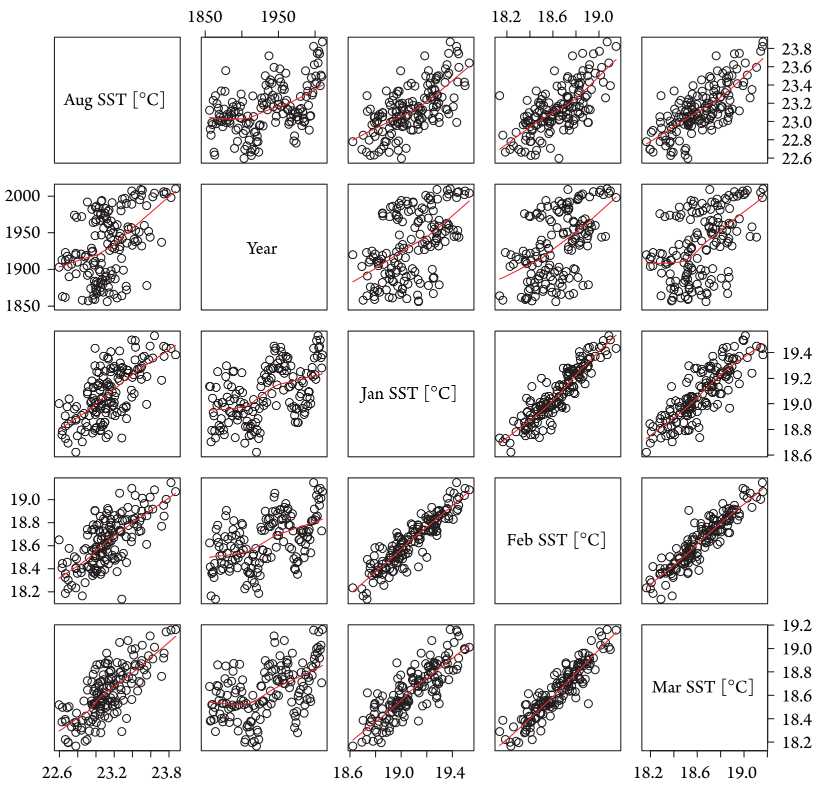

Figure 3.8 Scatter plot matrix of monthly SST values.

the horizontal axis. The plot in row [illegible]umn 3 is the August SST values on the vertical axis and January SST values on the horizontal axis, and so on. Note, in the lower left set of plots, the variables are the same except the axes are reversed, so the plot in column 1 is August SST values on the horizontal axis and the year is on the vertical axis.

The matrix of plots is useful in drawing attention to what explanatory variables might be important in your model of the response variable. Here you see all relationships are positive. Specifically, August SST values increase with increasing year and increasing January through March SST values. Based on these bivariate relationships, you might expect that all four explanatory variables, with the exception of perhaps Year, will be important in the model of August SST.

Importantly, the plots also reveal the relationships between the covariates. Here you see a tight linear relationship between each month's SST values. This warrants attention as a model that includes all three SST [illegible] contain a large amount of redundant information. Information contained in the February SST values is about the same as the information contained in the January SST values, and the information contained in the March SST values is about the same as the information contained in the February SST values.

Table 3.2 Coefficients of the multiple regression model.

	Estimate	*Std. Error*	*t value*	*Pr(> |t|)*
(Intercept)	6.1749	1.3154	4.69	0.0000
Year	0.0007	0.0004	1.90	0.0597
Jan	0.1856	0.1667	1.11	0.2673
Feb	−0.1174	0.2187	−0.54	0.5921
Mar	0.7649	0.1611	4.75	0.0000

To fit a multiple regression model to these data with August SST as the response variable, type

```
> m1 = lm(Aug ~ Year + Jan + Feb + Mar, data=SST)
```

Then to examine the model coefficients, type

```
> summary(m1)
```

The model coefficients and associated statistics are shown in Table 3.2. As expected, March SST is positively related to August SST, and significantly so. Year is also positively related to August SST. The Year term has a p value on its coefficient that is marginally significant (suggestive, but inconclusive) against the null hypothesis of a zero trend. However, you can see that the coefficient on January SST is positive, but not statistically significant, and the coefficient on February SST is negative.

From Figure 3.8 you can see that there is a *positive* relationship between February SST and August SST, so the fact that the relationship is negative in the context of multiple regression indicates a problem. The problem stems from the correlation between explanatory variables (multicollinearity). High correlation can result in an unstable model because the standard errors on the coefficients are not estimated with enough precision.

As long as two variables are not perfectly correlated estimation of the regression coefficients is possible, but the estimates and standard errors become sensitive to even the slightest change in the data when variables are correlated at levels above 0.6. Prior understanding of the partial correlation (here the correlation between February SST and August SST controlling for March SST) may help argue in favor of retaining two highly correlated explanatory variables, but in the usual case it is better to eliminate the variable whose relationship with the response variable is harder to explain physically or to eliminate the variable that has the smaller correlation with the response variable.

Here February SST has a smaller correlation with August SST, so you remove it and reestimate the coefficients. You create a new linear model object and summarize it by typing

```
> m2 = lm(Aug ~ Year + Jan + Mar, data=SST)
> summary(m2)
```

You see all the remaining explanatory variables have a positive relationship with the response variable, consistent with their bivariate plots, but the coefficient on January SST is not statistically significant.

Thus you need to try a third model with the insignificant term removed.

```
> m3 = lm(Aug ~ Year + Mar, data=SST)
> summary(m3)
```

The model makes sense. August SST values are higher when March SST values are higher. This relationship holds after accounting for the upward trend in August SST values.

Note that the order of the explanatory variables on the right side of the ~ does not matter. That is, you get the same output by typing

```
> summary(lm(Aug ~ Mar + Year, data=SST))
```

Note also that the multiple R-squared value is slightly lower in the final model with fewer variables. This is always the case and is the reason why the R squared should not be used for comparing models. Instead, use the adjusted R squred, which increases only when a statistically significant variable is added to the model.

The final model is checked for adequacy by examining the distribution of model residuals. The five-number summary of the residuals given as part of the summary output gives you no reason to suspect the assumption of normally distributed residuals. However, the residuals are likely to have some autocorrelation violating the assumption of independence. We will revisit this topic in Chapter 5.

3.11.1 Predictor Choice

Suppose H is your variable of interest and $X_1, \ldots, X_p$, a set of potential explanatory variables, are vectors of n observations. The problem of predictor selection arises when you want to model the relationship between H and a subset of the explanatory variables, but there is uncertainty about which subset to use. The situation is particularly interesting when p is large and the variables contain redundant information.

You want a model that fits the data well and has small variance. The problem is these two goals are in conflict. An additional predictor in a model will improve the fit (reduce the bias), but will increase the variance due to a loss in the number of degrees of freedom. This is known as the bias variance trade-off.

A commonly used statistic that helps with this trade-off is called the Akaike Information Criterion (AIC), given by

$$\text{AIC} = 2(p+1) + n[\log(\text{SSE}/n)], \tag{3.21}$$

where p is the number of predictors and SSE is the residual sum of squares. You can compare the AIC values when each predictor is added or removed from a given model. For example, if after adding a predictor, the AIC value for the model increases, then the trade-off is in favor of the extra degree of freedom and against retaining the predictor.

Returning to your original model for August SST, the model is saved in the object `m1`. The `drop1` function takes your regression model object and returns a table showing the effects of dropping, in turn, each variable from the model. To see this, type

```
> drop1(m1)
Single term deletions

Model:
Aug ~ Year + Jan + Feb + Mar
       Df Sum of Sq  RSS  AIC
<none>              4.61 -535
Year    1     0.111 4.72 -533
Jan     1     0.038 4.65 -536
Feb     1     0.009 4.62 -537
Mar     1     0.693 5.30 -515
```

Here the full model (all four covariates) has a residual sum of squares (RSS) of 4.61 (in the <none> row; none dropped). If you drop the Year variable, the RSS increases to 4.72 (you add 0.11 to 4.61) and you gain one degree of freedom. That is too much increase in RSS for the gain of only a single degree of freedom, thus the AIC increases to 4.72 from 4.61. You conclude that Year is too important to drop from the model. This is true of March SST, but not of January or February SST.

Therefore, to help you choose variables, you compare the AIC values for each variable against the AIC value for the full model. If the AIC value is less than the AIC for the full model, then the trade-off favors removing the variable from the model. If you repeat the procedure after removing the January and February SST variables, you will conclude that there is no statistical reason to make the model simpler.

Stepwise regression is a procedure for automating the `drop1` functionality. It is efficient in finding the best set of predictors. It can be done in three ways: forward selection, backward deletion, or both. Each uses the AIC as a criterion for choosing or deleting a variable and for stopping. To see how this works with your model, type

```
> step(m1)
```

The output is a series of tables showing the RSS and AIC values with successive variables removed. The default method is backward deletion, which amounts to a successive application of the `drop1` function. It is a good strategy to also try forward selection methods to see if the results are the same. Too have greater confidence in your final model if results are the same using forward selection and backward deletion.

3.11.2 Cross-validation

A cross-validation is needed if your statistical model will be used to make actual forecasts. Cross-validation is a procedure to assess how well your model will do in forecasting the unknown future. In the case of independent hurricane seasons, involves withholding a season's worth of data, developing the model on the remaining data, then using the model to predict data from the season that was withheld. Note that if your model-building algorithm involves stepwise regression or machine learning (Chapter 7), then the predictor selection component must be part of the cross-validation. That is, after removing a season's worth of data, you must run your selection procedure and then make a single-season prediction using the final model(s). And this needs to be done for each season removed.

The result of cross-validation is an estimate of out-of-sample error that accurately estimates the average forecast error when the model is used in predicting the future. The out-of-sample prediction error is then compared with the prediction error computed out of sample using a simpler model. If the error is larger with the simpler model, then your model is considered skillful. Note that the prediction error from the simpler model, even if it is long-run climatology, also needs to be obtained using cross-validation. More details on this important topic including some examples are given in Chapter 7.

In this chapter we reviewed classical statistics with examples from hurricane climatology. Topics included descriptive statistics, probability and distributions, one- and two-sample tests, statistical formula in R, correlation, and regression. Next, we give an introduction to Bayesian statistics.