# 6

# DATA SETS

"Data, data, data, I cannot make bricks without clay."
—Sherlock Holmes

Hurricane data originate from careful analysis of past storms by operational meteorologists. The data include estimates of the hurricane position and intensity at 6-hourly intervals. Information related to landfall time, local wind speeds, damages, and deaths, as well as cyclone size, are included. The data are archived by season.

Some effort is needed to make the data useful for hurricane climate studies. In this chapter, we describe the data sets used throughout this book. We show you a work flow that includes importing, interpolating, smoothing, and adding attributes. We also show you how to create subsets of the data. Code in this chapter is more complicated and it can take longer to run. You can skip this material on first reading and continue with model building in Chapter 7. You can return here when you have an updated version of the data that includes the most recent years.

## 6.1 BEST-TRACKS DATA

Most statistical models in this book use the best-track data. Here we describe these data and provide original source material. We also explain how to smooth and interpolate them. Interpolations are needed for regional hurricane analyses.

### 6.1.1 Description

The *best-track* data set contains the 6-hourly center locations and intensities of all known tropical cyclones across the North Atlantic basin, including the Gulf of Mexico and Caribbean Sea. The data set is called HURDAT for HURricane DATa. It is maintained by the U.S. National Oceanic and Atmospheric Administration (NOAA) at the National Hurricane Center (NHC).

Center locations are given in geographic coordinates (in tenths of degrees) and the intensities, representing the one-minute near-surface ($\sim$ 10 m) wind speeds, are given in knots (1 kt $=$ .5144 m s$^{-1}$) and the minimum central pressures are given in millibars (1 mb $=$ 1 hPa). The data are provided in 6-hourly intervals starting at 00 UTC (Universal Time Coordinate). The version of HURDAT file used here contains cyclones over the period 1851 through 2010 inclusive.[1] Information on the history and origin of these data is found in Jarvinen et al (1984).

The file has a logical structure that makes it easy to read with a FORTRAN program. Each cyclone contains a header record, a series of data records, and a trailer record. Original best-track data for the first cyclone in the file is shown here.

```
00005 06/25/1851 M= 4  1 SNBR=   1 NOT NAMED   XING=1 SSS=1
00010 06/25*280 948  80    0*280 954  80    0*280 960  80    0*281 965  80    0*
00015 06/26*282 970  70    0*283 976  60    0*284 983  60    0*286 989  50    0*
00020 06/27*290 994  50    0*295 998  40    0*3001000  40    0*3051001  40    0*
00025 06/28*3101002  40    0*  0   0   0    0*  0   0   0    0*  0   0   0    0*
00030 HRBTX1
```

The header (beginning with 00005) and trailer (beginning with 00030) records are single rows. The header has eight fields. The first field is the line number in intervals of five and padded with leading zeros. The second is the start day for the cyclone in MM/DD/YYYY format. The third is `M= 4`, indicating four data records to follow before the trailer record. The fourth field is a number indicating the cyclone sequence for the season; here, 1 indicates the first cyclone of 1851. The fifth field, beginning with `SNBR=`, is the cyclone number over all cyclones and all seasons. The sixth field is the cyclone name. Cyclones were named beginning in 1950. The seventh field indicates whether the cyclone hit the United States, with `XING=1` indicating that it did and `XING=0` indicating that it did not. A hit is defined as the center of the cyclone crossed the coast on the continental United States as a tropical storm or hurricane. The final field indicates the Saffir–Simpson hurricane scale (1–5) impact in the United States based on the estimated maximum sustained winds at the coast. The value 0 was used to indicate U.S. tropical storm landfalls but has been deprecated.

The next four rows contain the data records. Each row has the same format. The first field is again the line number. The second field is the cyclone day in MM/DD format. The next 16 fields are divided into four blocks of four fields each. The first block is the 00 UTC record, and the next three blocks are in 6-hour increments (6, 12, and 18 UTC). Each block is the same and begins with a code indicating the stage of the cyclone, tropical cyclone `*`, subtropical cyclone `S`, extratropical low `E`, wave `W`, and remanent low `L`. The three digits immediately to the right of the stage code is the latitude of the center position in tenths of degree north (`280` is 28.0°N) and the next four digits are the longitude in tenths of a degree west (`948` is 94.8°W) followed by a space. The third set of three digits is the maximum sustained (1 minute) surface

[1] From `www.nhc.noaa.gov/pastall.shtml#hurdat`, August 2011.

(10 m) wind speed in knots. These are estimated to the nearest 10 kt for cyclones before to 1886 and to 5 kt afterward. The final four digits after another space is the central surface pressure of the cyclone in millibars if available. If not, the field is given a zero. Central pressures are available for all cyclones after 1978.

The trailer has at least two fields. The first field is the line number. The second field is the maximum intensity of the cyclone as a code using `HR` for hurricane, `TS` for tropical storm, and `SS` for subtropical storm. If there are additional fields, they relate to landfall in the United States. The fields are given in groups of four with the first three indicating location by state and the last indicating the Saffir–Simpson scale based on wind speeds in the state. Two-letter state abbreviations are used, with the exception of Texas and Florida, which are further subdivided as follows: `ATX`, `BTX`, `CTX` for south, central, and north Texas, respectively, and `AFL`, `BFL`, `CFL`, and `DFL` for northwest, southwest, southeast, and northeast Florida, respectively. An `I` is used as a prefix in cases where a cyclone had hurricane impact is in a noncoastal state.

### 6.1.2 Import

The HURDAT file (e.g., *tracks.txt*) is appended each year with the set of cyclones from the previous season. The latest version is available usually by late spring or early summer from `www.nhc.noaa.gov/pastall.shtml`. Additional modifications to older cyclones are made when newer information becomes available. After downloading the HURDAT file, we use a FORTRAN executable file for the Windows platform (**BT2flat.exe**) to create a flat file (*BTflat.csv*) listing the data records. The file is created by typing

```
BT2flat.exe tracks.txt > BTflat.csv
```

The resulting comma-separated flat file is read into R and the lines between the separate cyclone records removed by typing

```
> best = read.csv("BTflat.csv")
> best = best[!is.na(best[, 1]),]
```

Further adjustment are made to change the hours to ones, the longitude to degrees east, and the column name for the type of cyclone.

```
> best$hr = best$hr/100
> best$lon = -best$lon
> east = best$lon < -180
> best$lon[east] = 360 + best$lon[east]
> names(best)[12] = "Type"
```

The first six lines of the data frame are shown here (`head(best)`).

```
  SYear Sn        name   Yr Mo Da hr  lat   lon Wmax pmin Type
1  1851  1 NOT NAMED  1851  6 25  0 28.0 -94.8   80    0    *
2  1851  1 NOT NAMED  1851  6 25  6 28.0 -95.4   80    0    *
3  1851  1 NOT NAMED  1851  6 25 12 28.0 -96.0   80    0    *
```

```
4   1851   1 NOT NAMED   1851   6 25 18 28.1 -96.5   80    0    *
5   1851   1 NOT NAMED   1851   6 26  0 28.2 -97.0   70    0    *
6   1851   1 NOT NAMED   1851   6 26  6 28.3 -97.6   60    0    *
```

Note the 10-kt precision on the Wmax column. This is reduced to 5 kt from 1886 onward.

Cyclones in the data frame are identified by SYear and Sn. To make it easier to subset by cyclone you add a unique cyclone identifier as follows. First, use the paste function to create a character id string that combines the SYear and Sn columns. Second, table the number of cyclone records with each character id and save these as an integer vector (nrs). Third, create a structured vector indexing the number of cyclones beginning with the first one. Fourth, repeat the index by the number of records in each cyclone and save the result in a Sid vector.

```
> id = paste(best$SYear, format(best$Sn), sep = ":")
> nrs = as.vector(table(id))
> cycn = 1:length(nrs)
> Sid = rep(cycn, nrs[cycn])
```

Next create a column identifying s, This is needed to perform time interpolations. Begin by creating a character vector with strings identifying the year, month, day, and hour. Note that first you need to take care of years when cyclones crossed into a new calendar year. In the best-track file, the year remains the year of the season. The character vector is turned into a POSIXlt object with the strptime function (see Chapter 5) and the time zone argument set to GMT (UTC).

```
> yrs = best$Yr
> mos = best$Mo
> yrs[mos==1] = yrs[mos==1]+1
> dtc = paste(yrs, "-", mos, "-", best$Da, " ",
+   best$hr, ":00:00", sep="")
> dt = strptime(dtc, format="%Y-%m-%d %H:%M:%S",
+   tz="GMT")
```

Each cyclone record begins at 0, 6, 12, or 18 UTC. Retrieve those hours for each cyclone using the cumsum function and the number of cyclone records as an index. Offsets are needed for the first and last cyclones. Then subsample the time vector obtained here at the corresponding values of the index and populate those times for all cyclone records. Then, the cyclone hour is the time difference between the two vectors in units of hours and is saved as Shour.

```
> i0 = c(1, cumsum(nrs[-length(nrs)]) + 1)
> dt0 = dt[i0]
> dt1 = rep(dt0, nrs[cycn])
> Shour = as.vector(difftime(dt, dt1, units="hours"))
```

Finally, include the two new columns in the best data frame.

```
> best$Sid = Sid
> best$Shour = Shour
> dim(best)
[1] 41192    14
```

The best-track data provide information on 1,442 individual tropical cyclones over the period 1851–2010, inclusive. The data frame you created contains these data in 41,192 separate 6-hourly records each having 14 columns. You can output the data as a spreadsheet using the `write.table` function.

If you want to send the file to someone that uses R or load it into another R session, use the `save` function. This exports a binary file that is imported back using the `load` function.

```
> save(best, file="best.RData")
> load("best.RData")
```

Alternatively, you might be interested in the functions available in the **RNetCDF** and **ncdf** packages for exporting data in Network Common Data Form.

### 6.1.3 Intensification

You can add value to these data by computing intensification (and decay) rates. The rate of change is estimated with as a derivative. Here you use the Savitzky–Golay smoothing filter (Savitzky and Golay, 1964) specifically designed for calculating derivatives. The filter preserves the maximum and minimum cyclone intensities. Moving averages dampen the extremes and derivatives estimated using finite differencing have larger errors.

The smoothed value of wind speed at a particular location is estimated using a local polynomial regression of degree three on a window of six values (including three locations before and two after). This gives a window width of 1.5 days. The daily intensification rate is the coefficient on the linear term of the regression divided by 0.25, since the data are given in quarter-day increments. A third-degree polynomial captures most of the fluctuation in cyclone intensity without overfitting and ensures consistency with the 5-kt precision of the raw wind speed.

The functions are available in **savgol.R**. Download the file from the book web site and source it. Then use the function `savgol.best` on your `best` data frame saving the results back in `best`.

```
> source("savgol.R")
> best = savgol.best(best)
```

The result is an appended data frame with two new columns, `WmaxS` and `DWmaxDt` giving the filtered estimates of wind speed and intensification, respectively. The filtered speeds have units of knots to be consistent with the best-track winds and the intensification rates are in knots per hour.

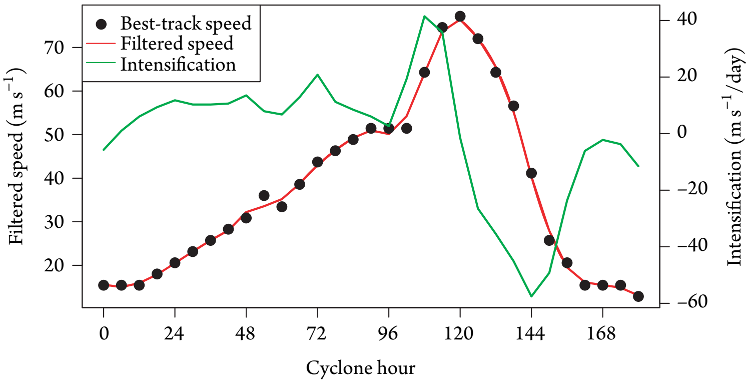

**Figure 6.1** Hurricane Katrina data.

As a comparison of the filtered and raw wind speeds, you look at the results from Hurricane Katrina of 2005. To make the code easier to follow, first save the rows corresponding to this cyclone in a separate object.

```
> Kt = subset(best, SYear == 2005 &
+   name == "KATRINA   ")
```

Next, plot the raw wind speeds as points and then overlay the filtered winds as a red line.

```
> plot(Kt$Shour, Kt$Wmax, pch=16, xlab="Cyclone Hour",
+   ylab="Wind Speed (m/s)")
> lines(Kt$Shour, Kt$WmaxS, lwd=2, col="red")
```

The spaces in the name after KATRINA are important as the variable name is a fixed-length character vector. On average, the difference between the filtered and raw wind speeds is 0.75 m s$^{-1}$, which is below the roundoff of 2.5 m s$^{-1}$ (15 kt) used in the best track. Over the entire set of data, the difference is less, averaging 0.49 m s$^{-1}$. Importantly, for estimating rates of change, the filtered wind speeds capture the maximum cyclone intensity.

Figure 6.1 shows filtered (red) and raw (circles) wind speeds and intensification rates (green) for Hurricane Katrina of 2005. Cyclogenesis occurred at 1800 UTC on Tuesday, August 23, 2005. The cyclone lasted for 180 h (7.5 days) dissipating at 600 UTC on Wednesday, August 31, 2005. The maximum best-track wind speed of 77.2 m s$^{-1}$ occurs at cyclone hour 120. This value equals the smoothed wind speed at that same hour.

### 6.1.4 Interpolation

For each cyclone, the observations are 6 h apart. For spatial analysis and modeling, this can be too coarse as the average forward motion of hurricanes is about 6 m s$^{-1}$

(12 kt). You therefore fill in the data using interpolation to 1 h. You also add an indicator variable for whether the cyclone is over land.

The interpolation is done with splines. The spline preserves values at the regular 6-hour times and uses a piecewise polynomial to obtain values between these times. For the cyclone positions, the splines are done using spherical geometry. The functions are available in **interpolate.R**.

```
> source("interpolate.R")
> load("landGrid.RData")
> bi = Sys.time()
> best.interp = interpolate.tracks(best,
+   get.land=TRUE, createindex=TRUE)
> ei = Sys.time()
```

Additionally, a land mask is used to determine whether the location is over land or water. Be patient, as the interpolation takes time to run. To see how long it takes, save the time (`Sys.time()`) before and after the interpolation and then take the difference in seconds.

```
> round(difftime(ei, bi, units="secs"), 1)
Time difference of 108 secs
```

The interpolation output is saved in the object `best.interp` as two lists: the data frame in a list called `data` and the index in a list called `index`. The index list is the record number by cyclone. The data frame has 239,948 rows and 30 columns. Additional columns include information related to the spherical coordinates and whether the position is over land as well as the cyclone's forward velocity (magnitude and direction). For instance, forward speed is in the column labeled `maguv` in units of kt. To obtain the average speed in meter per second over all cyclones, type

```
> mean(best.interp$data$maguv) * .5144
```

Finally, you add a day of year (`jd`) column giving the number of days since January 1 of each year. This is useful for examining intraseasonal activity (see Chapter 10). You use the function `ISOdate` from the **chron** package on the `ISOtime` column in the `best.interp$data` data frame. You first create a POSIXct object for the origin.

```
> x = best.interp$data
> start = ISOdate(x$Yr, 1, 1, 0)
> current = ISOdate(x$Yr, x$Mo, x$Da, x$hr)
> jd = as.numeric(current - start, unit="days")
> best.interp$data$jd = jd
> rm(x)
```

The hourly North Atlantic cyclone data prepared in the above manner are available in the file *best.use.RData*. The data include the best-track 6-hourly values plus the smoothed and interpolated values using the methods described here. The file is created by selecting specific columns from the interpolated data frame above. For example, type

```
> best.use = best.interp$data[, c("Sid", "Sn",
+   "SYear", "name", "Yr", "Mo", "Da", "hr", "lon",
+   "lat", "Wmax", "WmaxS", "DWmaxDt", "Type",
+   "Shour", "maguv", "diruv", "jd", "M")]
> save(best.use, file="best.use.RData")
```

You input these data as a data frame and list the first six lines by typing

```
> load("best.use.RData")
> head(best.use)
```

The `load` function imports an object saved as a compressed file with the `save` function. The object name in your workspace is the file name without the `.RData`.

Once the hurricane data are prepared in the manner described above, you can use functions to extract subsets of the data for particular applications. Here we consider a function to add regional information to the cyclone locations and another function to obtain the lifetime maximum intensity of each cyclone. These data sets are used throughout the book.

### 6.1.5 Regional Activity

Information about a cyclone's absolute location is available through the geographic coordinates (latitude and longitude). It is convenient to also have relative location information specifying, for instance, whether the cyclone is within a predefined area. Here your interest is near-coastal cyclones, so you consider three U.S. regions including the Gulf coast, Florida, and the East coast. The regions are shown in Figure 6.2. Boundaries are whole number parallels and meridians. The areas are large enough to capture enough cyclones, but not too large as to include many noncoastal strikes.

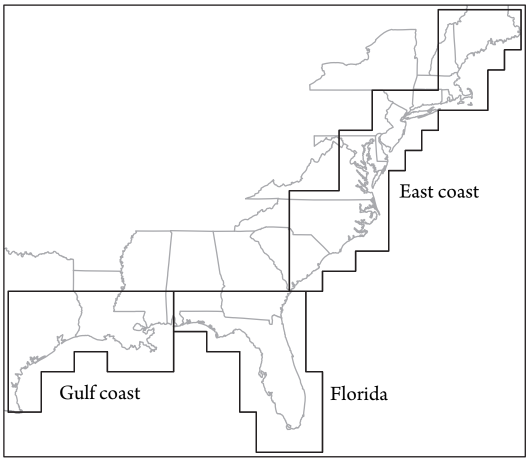

**Figure 6.2** Coastal regions.

Relative location is coded as a logical vector indicating whether or not the cyclone is inside the region. The three near-coastal regions are nonoverlapping, and you create one vector for each region. But it is also of interest to know whether the cyclone was in either of these areas or none of them.

The functions are in **datasupport.R** package. They include `import.grid`, which inputs a text file defining the parallels and meridians of the near-coastal regions and adds a regional name with the Gulf coast defined as region one, Florida defined as region two, the East coast defined as region three, and the entire coast as region four.

```
> source("datasupport.R")
> grid = import.grid("gridboxes.txt")
> best.use = add.grid(data=best.use, grid=grid)
```

### 6.1.6 Lifetime Maximum Intensity

An important variable for understanding hurricane climate is lifetime maximum intensity. Lifetime refers to the time from hurricane genesis to dissipation and lifetime maximum refers to the highest wind speed during this lifetime. The intensity value and the location where the lifetime maximum occurs are of general interest.

Here you use the `get.max` function in the **getmax.R** package. To make it accessible to your workspace, type

```
> source("getmax.R")
```

To apply the function on the `best.use` data frame using the default options `idfield="Sid"` and `maxfield="Wmax"`, type

```
> LMI.df = get.max(best.use)
```

List the values in 10 columns of the first 6 rows of the data frame rounding numeric variables to one decimal place.

```
> round(head(LMI.df)[c(1, 5:9, 12, 16)], 1)
      Sid   Yr Mo Da hr   lon WmaxS maguv
3.4     1 1851  6 25 16 -96.3  79.6   4.4
15      2 1851  7  5 12 -97.6  80.0   0.0
17      3 1851  7 10 12 -60.0  50.0   0.0
47.2    4 1851  8 23  2 -86.5  98.9   6.5
76.2    5 1851  9 15  2 -73.5  50.0   0.0
89.2    6 1851 10 17  2 -76.5  59.0   5.7
```

The data frame `LMI.df` contains the same information as `best.use` except here there is only one row per cyclone. Each row contains the lifetime maximum intensity and the corresponding location and other attribute information for the cyclone at the time the maximum was *first* achieved. If a cyclone is at its lifetime maximum intensity for more than 1 h, only the first hour information is saved. To subset the data frame of lifetime maximum intensities for cyclones of tropical storm intensity or stronger since 1967 exporting the data frame as a text file, type

```
> LMI.df = subset(LMI.df, Yr >= 1967 & WmaxS >= 34)
> write.table(LMI.df, file="LMI.txt")
```

### 6.1.7 Regional Maximum Intensity

Here your interest is the cyclone's maximum intensity only when it is within a specified region (e.g., near the coast). You create a set of data frames arranged as a list that includes the cyclone maximum within each of the regions defined in §6.1.5. You start by defining the first and last years of interest and create a structured list of those years, inclusive.

```
> firstYear = 1851
> lastYear = 2010
> sehur = firstYear:lastYear
```

These definitions make it easy for you to add additional of data as they become available or to focus your analysis on data only over the most recent years.

Next define a vector of region names and use the function `get.max.flags` (**datasupport.R**) to generate the set of data frames saved in the list object `max.regions`.

```
> Regions = c("Basin", "Gulf", "Florida", "East", "US")
> max.regions = get.max.flags(se=sehur, field="Wmax",
+   rnames=Regions)
```

You view the structure of the resulting list with the `str` function. Here you specify only the highest level of the list by setting the the argument `max.level` to one.

```
> str(max.regions, max.level=1)
List of 5
 $ Basin  :'data.frame': 1442 obs. of  23 variables:
 $ Gulf   :'data.frame': 246 obs. of  23 variables:
 $ Florida:'data.frame': 330 obs. of  23 variables:
 $ East   :'data.frame': 280 obs. of  23 variables:
 $ US     :'data.frame': 606 obs. of  23 variables:
```

The object contains a list of five data frames with names corresponding to the regions defined earlier. Each data frame has the same 1,442 columns of data defined in `best.use`, but the number of rows depends on the number of cyclones passing through the region. Note, the `Basin` data frame contains all cyclones.

To list the first six rows and several of the columns in the `Gulf` data frame, type

```
> head(max.regions$Gulf[c(1:7, 11)])
        Sid Sn SYear       name   Yr Mo Da  Wmax
3.4       1  1  1851 NOT NAMED  1851  6 25  80.8
130.4     7  1  1852 NOT NAMED  1852  8 26 100.6
353.4    20  1  1854 NOT NAMED  1854  6 26  71.8
389.4    23  4  1854 NOT NAMED  1854  9 18  90.9
443.3    29  5  1855 NOT NAMED  1855  9 16 111.1
456.3    30  1  1856 NOT NAMED  1856  8 10 132.1
```

You treat `max.regions$Gulf` as a regular data frame although it is part of a list. The output indicates that the sixth Gulf cyclone in the record is the 30th cyclone in the best-track record (`Sid` column) and the 1st cyclone of the 1856 season. It has a maximum intensity of 68 m s$^{-1}$ while in the region. You export the data using the `save` function as before.

```
> save("max.regions", file="max.regions.Rdata")
```

### 6.1.8 Tracks by Location

Suppose you want to know only about hurricanes that have affected a particular location. Or those that have affected several locations (e.g., San Juan, Miami, and Kingston). Hurricanes specific to a location can be extracted with functions in the **getTracks** package. To see how this works, load the *best.use.RData* data and install the source code.

```
> load("best.use.RData")
> source("getTracks.R")
```

The function is `get.tracks`. It takes as input the longitude and latitude of your location along with the search radius (nautical miles) and the number of cyclones and searches for tracks that are within this distance. It computes a score for each track with closer cyclones getting a higher score.

Here you use the function to find the five cyclones of at least tropical storm strength that have come closest to the Norfolk Naval Air Station (NGU) (76.28°W longitude and 36.93°N latitude) during the period 1900 through 2010. You save the location and a search radius of 100 nmi in a data frame. You also set the start and end years of your search and the number of cyclones before calling `get.tracks`.

```
> loc = data.frame(lon=-76.28, lat=36.93, R=100)
> se = c(1900, 2010); Ns = 5
> ngu = get.tracks(x=best.use, locations=loc, N=Ns,
+   se=se)
> names(ngu)
[1] "tracks"    "SidDist"   "N"         "locations"
```

The output contains a list with four objects. The objects N and locations are the input parameters. The object SidDist is the cyclone identifier for each cyclone captured by the input criteria listed from closest to farthest from NGU. The corresponding track attributes are given in the list object tracks with each component a data frame containing the individual cyclone attributes from best.use. The tracks are listed in order by increasing distance. For example, ngu$SidDist[1] is the distance of the closest track and ngu$tracks[[1]] is the data frame corresponding to this track.

You plot the tracks on a map reusing the code from Chapter 5. Here you use a gray scale on the track lines corresponding to a closeness ranking with darker lines indicating closer tracks.

```
> map("world", ylim=c(12, 60), xlim=c(-90, -50))
> points(ngu$location[1, 1], ngu$location[1, 2],
+   col="red", pch=19)
> for(i in Ns:1){
+ clr = gray((i - 1)/Ns)
+ Lo = ngu$tracks[[i]]$lon
+ La = ngu$tracks[[i]]$lat
+ n = length(Lo)
+ lines(Lo, La, lwd=2, col=clr)
+ arrows(Lo[n - 1], La[n - 1], Lo[n], La[n], lwd=2,
+    length=.1, col=clr)
+ }
> box()
```

The results are shown in Figure 6.3 for two locations: NGU only and for two locations: NGU and Roosevelt Naval Air Station in Puerto Rico (NRR). Darker tracks indicate closer cyclones. This application is useful for cyclone-relative hurricane climatologies (see Scheitlin et al. [2010]).

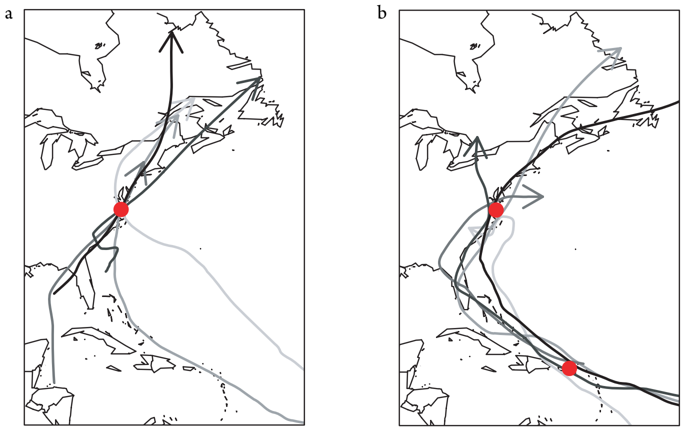

**Figure 6.3** Five closest cyclones. (a) NGU and (b) NRR and NGU.

### 6.1.9 Attributes by Location

Location-specific hurricane attributes are needed for local surge and wind models. To extract these data, first determine the cyclone observations within a grid box centered on your location of interest. This is done using the `inside.lonlat` utility function (**getTracks.R**). Here your location is NGU from above.

```
> ins = inside.lonlat(best.use, lon=loc[1, 1],
+   lat = loc[1, 2], r = 100)
> length(ins)
[1] 239948
```

Your grid box size is determined by twice the value of argument `r` in units of nautical miles. The box is square as the distances are computed on a great-circle. The function returns a logical vector with length equal to the number of cyclone hours in `best.use`.

Next you subset the rows in `best.use` for values of `TRUE` in `ins`.

```
> ngu.use = best.use[ins, ]
```

Since your interest is cyclones of hurricane intensity, further subset using `WmaxS`.

```
> ngu.use = subset(ngu.use, WmaxS >= 64)
> length(unique(ngu.use$Sid))
[1] 54
```

There are 54 hurricanes passing through your grid box over the period of record. A similar subset is obtained using a latitude/longitude grid by typing

```
> d = 1.5
> lni = loc[1, 1]
> lti = loc[1, 2]
> ngu.use = subset(best.use, lat <= lti + d &
+   lat >= lti - d & lon <= lni + d &
+   lni >= lni - d & WmaxS >= 64)
```

Finally use the `get.max` function to select cyclone-specific attributes. For example, to determine the distribution of *minimum* translation speeds for all hurricanes in the grid and to put plot them as a histogram, type

```
> source("getmax.R")
> ngu.use$maguv = -ngu.use$maguv
> ngu.use1 = get.max(ngu.use, maxfield="maguv")
> speed = -ngu.use1$maguv * .5144
> hist(speed, las=1, xlab="Forward Speed (m/s)",
+      main="")
```

The results is shown in Figure 6.4. Notice that you take the additive inverse of the speed since your interest is in the minimum. A parametric distribution is fit or

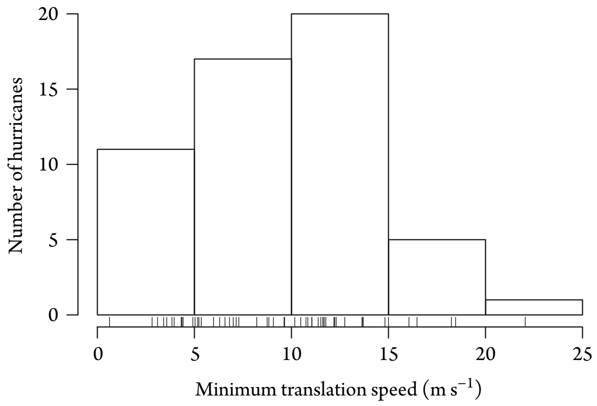

**Figure 6.4** Minimum per hurricane translation speed near NGU.

resampling is used to generate inputs for hurricane surge and wind models. (see Chapter 13).

## 6.2 ANNUAL AGGREGATION

It is also useful to have hurricane data aggregated in time. Aggregation is often done annually since hurricane occurrence has a strong seasonal cycle (see Chapter 10). Annual aggregation makes it convenient to merge hurricane data with monthly climate variables.

### 6.2.1 Annual Cyclone Counts

Annual counts are the most frequently analyzed hurricane climate data. Here you aggregate counts by year for the entire basin and the near-coastal regions defined in §6.1.5. First, simplify the region names to a single letter using the `substring` function making an exception to the U.S. region by changing it back to `US`.

```
> load("max.regions.RData")
> names(max.regions) = substring(names(max.regions),
+   1, 1)
> names(max.regions)[names(max.regions)=="U"] = "US"
```

This allows you to add the Saffir–Simpson category as a suffix to the names.

The `make.counts.basin` function (**datasupport.R**) performs the annual aggregation of counts by category and region with the `max.regions` list of data frames as the input and a list of years specified with the `se` argument.

```
> source("datasupport.R")
> sehur = 1851:2010
> counts = make.counts.basin(max.regions, se=sehur,
+   single=TRUE)
> str(counts, list.len=5, vec.len=2)
```

```
'data.frame':   160 obs. of  31 variables:
 $ Year: int  1851 1852 1853 1854 1855 ...
 $ B.0 : int  6 5 8 5 5 ...
 $ B.1 : int  3 5 4 3 4 ...
 $ B.2 : int  1 2 3 2 3 ...
 $ B.3 : int  1 1 2 1 1 ...
  [list output truncated]
```

The result is a data frame with columns labeled $X.n$ for $n = 0,1,\ldots,5$, where $X$ indicates the region. For example, the annual count of hurricanes affecting Florida is given in the column labeled `F.1`. The start year is 1851 and the end year is 2010.

Here you create a two-by-two matrix of plots showing hurricane counts by year for the basin, and the U.S., Gulf Coast, and Florida regions. The `with` function allows you to use the column names with the `plot` method.

```
> par(mfrow=c(2, 2))
> with(counts, plot(Year, B.1, type="h", xlab="Year",
+   ylab="Basin count"))
> with(counts, plot(Year, US.1, type="h", xlab="Year",
+   ylab="U.S. count"))
> with(counts, plot(Year, G.1, type="h", xlab="Year",
+   ylab="Gulf coast count"))
> with(counts, plot(Year, F.1, type="h", xlab="Year",
+   ylab="Florida count"))
```

The plots are shown in Figure  6.5. Regional hurricane counts indicate no long-term trend, but the basinwide counts show an active period beginning late in the twentieth century. Some of this variation is related to fluctuations in climate as examined in Chapter 7. Next the annually and regionally aggregated counts are merged with monthly and seasonal climate variables.

### 6.2.2 Environmental variables

The choice of climate variables is large. You narrow it down by considering what is known about hurricane climate. For example, it is well understood that ocean heat provides the fuel, a calm atmosphere provides a favorable setting, and the location and strength of the subtropical ridge provide the steering currents. Thus statistical models of hurricane counts should include covariates that index these climate variables including sea-surface temperature (SST), as an indicator of oceanic heat content, El Niño-Southern Oscillation (ENSO) as an indicator of vertical wind shear, and the North Atlantic Oscillation (NAO) as an indicator of steering flow. Variations in solar activity might also influence hurricane activity. We speculate that an increase in solar ultraviolet (UV) radiation during periods of strong solar activity might suppress tropical cyclone intensity as the temperature near the tropopause will warm

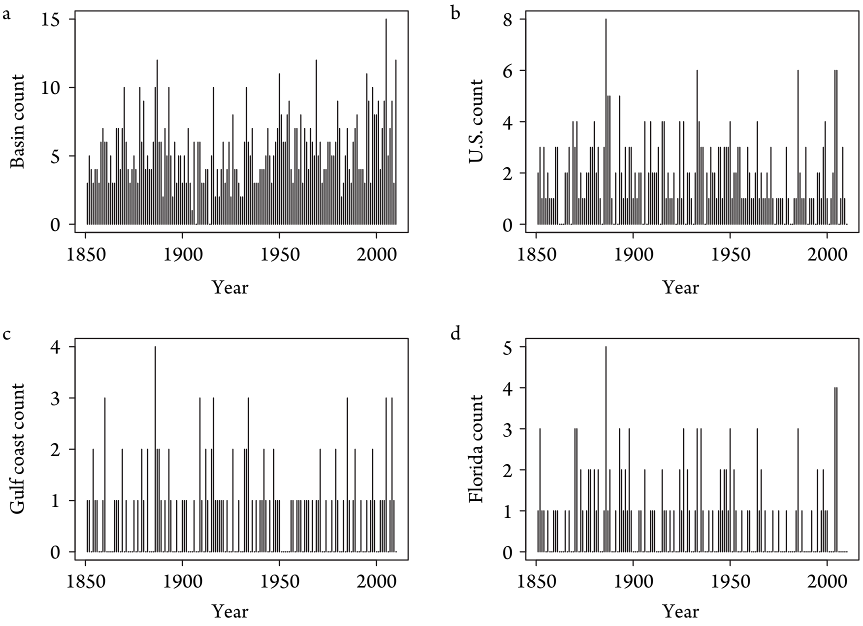

**Figure 6.5** Hurricane counts. (a) Basin, (b) U.S., (c) Gulf Coast, and (d) Florida.

through absorption of radiation by ozone and modulated by dynamic effects in the stratosphere (Elsner and Jagger, 2008).

Thus you choose four climate variables including North Atlantic Ocean SST, the Southern Oscillation Index (SOI) as an indicator of ENSO, an index for the NAO, and sunspot numbers (SSN). Monthly values for these variables are obtained from the following sources.

- **SST**: The SST variable is an area-weighted average (°C) using values in 5° latitude–longitude grid boxes from the equator north to 70°N latitude and spanning the North Atlantic Ocean (Enfield et al. 2001).[2] Values in the grid boxes are from a global SST data set derived from the UK Met Office (Kaplan et al. 1998).
- **SOI**: The SOI is the contemporaneous difference in monthly sea-level pressures between Tahiti ($T$) in the South Pacific Ocean and Darwin ($D$) in Australia ($T - D$) (Trenberth, 1984).[3] The SOI is inversely correlated with equatorial eastern and central Pacific SST, so an El Niño warm event is associated with negative values of the SOI.
- **NAO**: The NAO is the fluctuation in contemporaneous sea-level pressure differences between the Azores and Iceland. An index value for the NAO is calculated as the difference in monthly normalized pressures at Gibraltar and over Iceland

[2] From www.esrl.noaa.gov/psd/data/correlation/amon.us.long.data, November 2011.

[3] From www.cgd.ucar.edu/cas/catalog/climind/SOI.signal.annstd.ascii, November 2011.

(Jones et al, 1997).[4] The NAO index indicates the strength and the position of the subtropical Azores/Bermuda High.

- **SSN**: The SSN variable are the Wolf sunspot numbers measuring the number of sunspots present on the surface of the sun. They are produced by the Solar Influences Data Analysis Center (SIDC) of World Data Center for the Sunspot Index at the Royal Observatory of Belgium and available from NOAA's National Geophysical Data Center.[5]

You combine these above climate and solar variables by month (May through October) and season with the aggregate hurricane counts by year. You use the `useCov` (**datasupport.R**) function to input the data. The file must have a column indicating the year and 12 or 13 additional columns indicating the months and perhaps an annual average. The argument `miss` inputs the missing value code that is used in the file. The argument `ma` is for centering and scaling the values. The default is none; `"c"` centers, `"cs"` centers and scales, and `"l"` subtracts the values in the last row from values in each column. To accommodate using previous year's data for modeling current year's cyclone counts, the resulting data frame is augmented with columns corresponding to 1-year shift of all months using the argument `last=TRUE`. Column names for the previous year's months are appended with a `.last`.

You input and organize all climate variables at once with the `readClimate` function. Copy and paste the code to your R session.

```
> readClimate = function(){
+   sst = readCov("data/amon.us.long.mean.txt",
+     header=FALSE, last=TRUE, miss=-99.990, ma="l",
+     extrayear=TRUE)
+   soi = readCov("data/soi_ncar.txt", last=TRUE,
+     miss=-99.9, extrayear=TRUE)
+   nao = readCov("data/nao_jones.txt", last=TRUE,
+     miss=c(-99.99, 9999), extrayear=TRUE)
+   ssn = readCov("data/sunspots.txt", header=TRUE,
+     last=TRUE, extrayear=TRUE)
+ return(list(sst=sst, soi=soi, nao=nao, ssn=ssn))
+ }
```

The list of data frames, one for each climate variable, is created by typing

```
> climate = readClimate()
> str(climate, max.level=1)
List of 4
 $ sst:'data.frame':    157 obs. of  25 variables:
 $ soi:'data.frame':    147 obs. of  25 variables:
```

[4] From `www.cru.uea.ac.uk/~timo/datapages/naoi.htm`, November 2011.
[5] From `ftp.ngdc.noaa.gov/STP/SOLAR_DATA/SUNSPOT_NUMBERS/INTERNATIONAL/monthly/MONTHLY`, November 2011.

```
 $ nao:'data.frame':    192 obs. of  25 variables:
 $ ssn:'data.frame':    162 obs. of  25 variables:
```

Each data frame has 25 columns (variables) corresponding to two sets of monthly values (current and previous year) plus a column of years. The number of rows (observations) in the data frames varies with the NAO being the longest, starting with the year 1821, although not all months in the earliest years have values. To list the first six rows and several of the columns in the `nao` data frame, type

```
> head(climate$nao[c(1:2, 21:23)])
    Yr Jan.last   Aug   Sep   Oct
1 1821       NA -0.14    NA    NA
2 1822       NA -0.19 -1.09 -2.00
3 1823       NA  2.90  0.67 -1.39
4 1824    -3.39 -0.08  0.19    NA
5 1825    -0.16  1.43 -0.95  1.98
6 1826    -0.23  2.72 -0.76  0.18
```

Note how `climate$nao` is treated as a data frame although it is part of a list.

The final step is to merge the climate data with the cyclone counts organized in §6.2.1. This is done by creating a single data frame of your climate variables. First, create a list of month names by climate variable. Here you consider only the months from May through October. You use August through October as a group for the SOI and SST variables, May and June as a group for the NAO variable, and September for the SSN variable.

```
> months = list(
+   soi=c("May", "Jun", "Jul", "Aug", "Sep", "Oct"),
+   sst=c("May", "Jun", "Jul", "Aug", "Sep", "Oct"),
+   ssn=c("May", "Jun", "Jul", "Aug", "Sep", "Oct"),
+   nao=c("May", "Jun", "Jul", "Aug", "Sep", "Oct"))
> monthsglm = list(
+   soi=c("Aug", "Sep", "Oct"),
+   sst=c("Aug", "Sep", "Oct"),
+   ssn="Sep",
+   nao=c("May","Jun"))
```

Next, use the `make.cov` (**datasupport.R**) function on the `climate` data frame, specifying the month list and the start and end years. Here you use the word "covariate" in the statistical sense to indicate a variable this is predictive of cyclone activity. In statistics, a covariate is also called an explanatory variable, an independent variable, or a predictor.

```
> covariates = cbind(make.cov(data=climate,
+   month=months, separate=TRUE, se=sehur),
+   make.cov(data=climate, month=monthsglm,
+   separate=FALSE, se=sehur)[-1])
```

The cbind function brings together the columns into a single data frame. The last six rows and a sequence of columns from the data frame are listed by typing,

```
> tail(covariates[seq(from=1, to=29, by=5)])
    Year soi.Sep sst.Aug ssn.Jul nao.Jun    soi
155 2005     1.4   0.622    40.1   -1.00  0.800
156 2006    -1.9   0.594    12.2   -0.41 -3.867
157 2007     0.4   0.245     9.7   -3.34  0.833
158 2008     4.6   0.361     0.8   -2.05  3.767
159 2009     1.0   0.345     3.2   -3.05 -2.033
160 2010     8.0   0.725    16.1   -2.40  6.200
```

The columns are labeled $X.m$, where $X$ indicates the covariate (soi, sst, sun, and nao) and $m$ indicates the month using a three-letter abbreviation with the first letter capitalized. Thus, for example, June values of the NAO index are in the column labeled nao.Jun. The hurricane-season-averaged covariate is also given in a column labeled without the month suffix. Season averages use August through October for SST and SOI, May and June for NAO, and September only for SSN.

As you did with the counts, here you create a two-by-two plot matrix showing the seasonal-averaged climate and solar variables by year (Fig. 6.6).

```
> par(mfrow=c(2, 2))
> with(covariates, plot(Year, sst, type="l",
+   xlab="Year", ylab="SST [C]"))
> with(covariates, plot(Year, nao, type="l",
+   xlab="Year", ylab="NAO [s.d.]"))
> with(covariates, plot(Year, soi, type="l",
+   xlab="Year", ylab="SOI [s.d.]"))
> with(covariates, plot(Year, ssn, type="l",
+   xlab="Year", ylab="Sunspot Count"))
```

The long-term warming trend in SST is quite pronounced as is the cooling trend during the 1960s and 1970s. The NAO values show large year-to-year variations and a tendency for negative values during the early part of the twenty-first century. The SOI values also show large interannual variations. Sunspot numbers show a pronounced periodicity near 11 years (solar cycle) related to changes in solar dynamics.

Finally, you use the merge function to combine the counts and covariates data frames, merging on the variable Year which appears in both.

```
> annual = merge(counts, covariates, by="Year")
> save(annual, file="annual.RData")
```

The result is a single data frame with 160 rows and 59 columns. The rows correspond to separate years and the columns include the cyclone counts by Saffir–Simpson scale and the monthly covariates defined here. The data frame is exported to the file *annual.RData*.

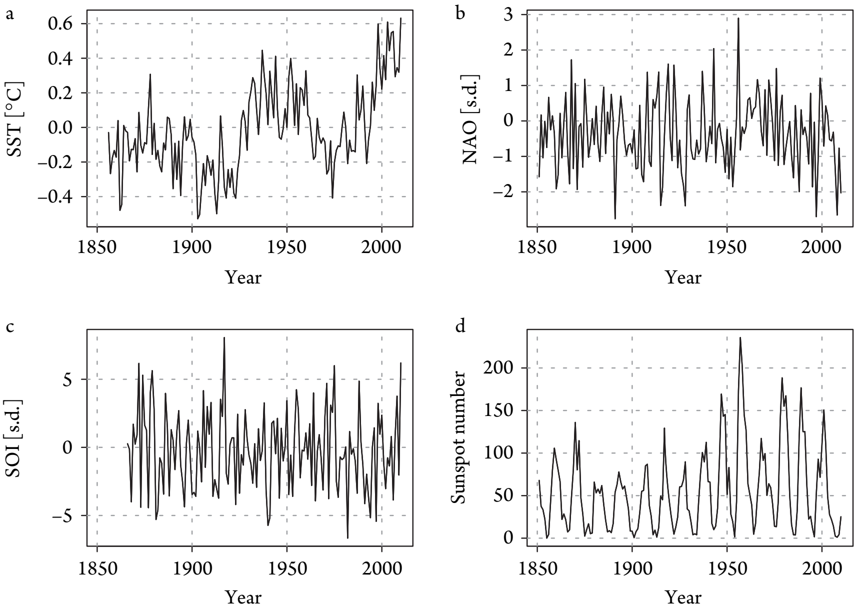

**Figure 6.6** Climate variables. (a) SST, (b) NAO, (c) SOI, and (d) sunspots.

## 6.3 COASTAL COUNTY WINDS

### 6.3.1 Description

Other hurricane data sets besides the best-track are available. County wind data are compiled in Jarrell et al. (1992) from reports on hurricane experience levels for coastal counties from Texas to Maine. The data are coded by Saffir–Simpson category and are available as an Excel$^{TM}$spreadsheet.[6]

The file consists of one row for each year and one column for each county. A cell contains a number if a tropical cyclone affected the county in a given year, otherwise, it is left blank. The number is the Saffir–Simpson intensity scale. For example, a county with a value of 2 indicates that category-two scale wind speeds were likely experienced at least somewhere in the county. If the number is inside parentheses, then the county received an indirect hit and the highest winds were likely at least one category weaker. Cells having multiple entries, separated by commas, indicate that the county was affected by more than one hurricane during that year.

The data set is originally constructed as follows. First, a Saffir–Simpson category is assigned to the hurricane at landfall based on central pressure and wind intensity estimates in the best-track data set. Some subjectivity enters the assignment particularly with hurricanes during earlier years of the twentieth century and with storms moving inland over a sparsely populated area. Thus there is some ambiguity about the category for hurricanes with intensities near the category cutoff. The category is

[6] From `www.aoml.noaa.gov/hrd/hurdat/Data_Storm.html`, November 2011.

**Table 6.1** Data symbols and interpretation. The symbol is from Appendix C of Jarrell et al. (1992).

| *Symbol* | *Saffir–Simpson Range* | *Wind Speed Range ($m\,s^{-1}$)* |
|---|---|---|
| (1) | [0,1) | 33–42 |
| 1 | [1,2) | 33–42 |
| (2) | [1,2) | 33–42 |
| 2 | [2,3) | 42–50 |
| (3) | [1,3) | 33–50 |
| 3 | [3,4) | 50–58 |
| (4) | [1,4) | 33–58 |
| 4 | [4,5) | 58–69 |
| (5) | [1,5) | 33–69 |
| 5 | [5,∞) | 69–1000 |

sometimes adjusted based on storm surge estimates, in which case the central pressure may not agree with the scale assignment. Beginning with the 1996 hurricane season, scale assignments are based solely on maximum winds.

Second, a determination is made about which coastal counties received direct and indirect hits. A direct hit is defined as the innermost core regions, or "eye," moving over the county. Each hurricane is judged individually based on the available data, but a general rule of thumb is applied in cases of greater uncertainty. That is, a county is regarded as receiving a direct hit when all or part of a county falls within a distance d to the left of a storm's landfall and a distance 2d to the right (with respect to an observer at sea looking toward shore), where d is the radius to maximum winds defined as the distance from the cyclone's center to the circumference of maximum winds around the center.

The determination of an indirect hit is based on a hurricane's strength and size and on the configuration of the coastline. In general, it is determined that the counties on either side of the direct-hit zone that received hurricane force winds or tides of at least 1–2 m above normal are considered an indirect hit. Subjectivity is also necessary here because of coastline geography and uncertainty about the hurricane's exact path.

Table 6.1 lists the possible cell entries for a given hurricane and our interpretations of the symbol in terms of the Saffir–Simpson category and wind speed range. The first column is the symbol used in Appendix C of Jarrell et al. (1992). The second column is the corresponding Saffir–Simpson scale range likely experienced somewhere in the county. The third column is the interpreted maximum sustained (1 min) near-surface (10 m) wind speed range (m $s^{-1}$).

The data are incomplete in the sense that you have a range of wind speeds rather than a single estimate. In statistics, the data are called "interval censored." Note that (1) is the same as 1, because they both indicate a cyclone with at least hurricane-force winds.

### 6.3.2 Counts and Magnitudes

The raw data need to be organized. First, remove characters that code for information not used. This includes codes such as 'W','E','*' and '_'. Second, convert all combinations of multiple-hit years, say (1, 2), to (1), (2) for parsing. Once parsed, all table cells consist of character strings that are either blank or contain cyclone hit information separated by commas. Finally, input the cleaned data to R with the first column used for row names by typing

```
> cd = read.csv("HS.csv", row.names=1, header=TRUE)
```

Remove the state row and save as a separate vector by typing

```
> states = cd[1,]
> cdf = cd[-1,]
> cdf[c(1, 11), 1:4]
     CAMERON WILLACY KENEDY KLEBERG
1900
1910     (2)     (2)      2       2
```

Rows 1 and 11 are printed so that you can see the data frame structure. In 1900, these four southern Texas counties were not affected by a hurricane (blank row), but a year later, Kenedy and Kleberg counties had a direct hit by a category-two hurricane that was also felt indirectly in Cameron and Willacy counties.

Next, you convert this data frame into a matrix object containing event counts and a list object for event magnitudes. First set up a definition table to convert the category data to wind speeds. Note the order is (1), . . ., (5), 1, . . ., 5. The column names `time` and `time2` are required for use with `Surv` function to create a censored data type.

```
> wt = data.frame(
+   time = c(rep(33, 6), 42, 50, 58, 69),
+   time2 = c(42, 42, 50, 58, 69, 42, 50, 58, 69,
+   1000))
> rownames(wt) = c(paste("(", 1:5, ")", sep=""),
+   paste(1:5))
```

Next, expand the data frame into a matrix. Each entry of the matrix is a character string vector. The character string is a zero vector for counties without a hurricane for a given year. For counties with a hurricane, the string contains symbols as shown in Table 6.1, one for each hurricane. This is done using `apply` and the `strsplit` function as follows:

```
> pd = apply(cdf, c(1, 2), function(x)
+   unlist(strsplit(gsub(" ", "", x), ",")))
```

Next, extract a matrix of counts and generate a list of events one for each county along with a list of years required for matching with the covariate data. Note that the year is extracted from the names of the elements.

```
> counts = apply(pd, c(1, 2), function(x)
+   length(x[[1]]))
> events = lapply(apply(pd, 2, unlist),
+   function(x)
+   data.frame(Year=as.numeric(substr(names(x), 1, 4)),
+   Events=x, stringsAsFactors=FALSE))
```

Finally, convert events to wind speed categories. You do this using the `Surv` function from the **survival** package (Therneau, 2012) as follows:

```
> require(survival)
> winds = lapply(events, function(x)
+   data.frame(Year=x$Year,
+   W = do.call("Surv", c(wt[x$Events, ],
+   list(type="interval2")))))
```

The object `winds` is a list of the counties with each list containing a data frame. To extract the data frame from county 57 corresponding to Miami-Dade county, type

```
> miami = winds[[57]]
> class(miami)
[1] "data.frame"
> head(miami)
  Year        W
1 1904 [33, 42]
2 1906 [33, 42]
3 1906 [50, 58]
4 1909 [50, 58]
5 1926 [58, 69]
6 1926 [33, 50]
```

The data frame contains a numerical year variable and a categorical survival variable. The survival variable has three components indicating the minimum and maximum Saffir–Simpson category.

You will use the `winds` and `counts` objects in Chapter 8 to create a probability model for winds exceeding threshold intensity levels. Here you export the objects as separate files using the `save` function so you can read them back using the `load` function.

```
> save(winds, file="catwinds.RData")
> save(counts, file="catcounts.RData")
```

The saved files are binary (8-bit characters) to ensure that they transfer without converting end-of-line markers.

## 6.4 NETCDF FILES

Climate data, such as monthly SST grids, are organized as arrays and stored in netCDF files. NetCDF (Network Common Data Form) is a set of software libraries and data formats from the Unidata community that support the creation, access, and sharing of data arrays. The National Center for Atmospheric Research (NCAR) uses netCDF files to store large data sets. The **ncdf** package (Pierce, 2011) provides functions for working with netCDF files in R. Install the package by typing

```
> require(ncdf)
```

You also might want to check out the functions available in **RNetCDF** for processing netCDF files.

Here your interest is with NOAA's extended reconstructed SST version 3b data set for the North Atlantic Ocean.[7] The data are provided by the NOAA/OAR/ESRL PSD in Boulder, Colorado. The data are available in file *sstna.nc* for the domain bounded by the equator and 70°N latitude and 100°W and 10°E longitude for the set of months starting with January 1854 through November 2009.

First, use the function `open.ncdf` to input the SST data.

```
> nc = open.ncdf("sstna.nc")
```

Next, convert the `nc` object of class ncdf into a three-dimension array and print the array's dimensions.

```
> sstvar = nc$var$sst
> ncdata = get.var.ncdf(nc, sstvar)
> dim(ncdata)
[1]   56   36 1871
> object.size(ncdata)
30175696 bytes
```

The file contains 3,771,936 monthly SST values distributed across 56 longitudes, 36 latitudes, and 1,871 months.

Additional work is needed before analysis can begin. First, extract the array dimensions as vector coordinates of longitudes, latitudes, and time. Then change the longitudes to negative west of the prime meridian and reverse the latitudes to increase from south to north. Also convert the time coordinate to a POSIX time (see Chapter 5) using January 1, 1[illegible], as the origin.

```
> vals = lapply(nc$var$sst$dim, function(x)
+   as.vector(x$vals))
> vals[[1]] = (vals[[1]] + 180) %% 360 - 180
> vals[[2]] = rev(vals[[2]])
> timedate = as.POSIXlt(86400 * vals[[3]],
```

[7] From `www.esrl.noaa.gov/psd/data/gridded/data.noaa.ersst.html`, November 2011.

```
+   origin=ISOdatetime(1800, 1, 1, 0, 0, 0, tz="GMT"),
+   tz="GMT")
> timecolumn = paste("Y", 1900 + timedate$year, "M",
+   formatC(as.integer(timedate$mo + 1), 1, flag="0"),
+   sep="")
> names(vals) = sapply(nc$var$sst$dim, "[", "name")
> vals = vals[1:2]
```

Note that the double percent symbol is the modulo operator, which finds the remainder of a division of the number to the left of the symbol by the number to the right.

Next, coerce the array into a data frame with one column per time period and assign column names.

```
> ncdata1 = ncdata[, (dim(ncdata)[2]:1), ]
> dims = dim(ncdata1)
> dim(ncdata1) = c(dims[1] * dims[2], dims[3])
> colnames(ncdata1) = timecolumn
> ncdata1 = as.data.frame(ncdata1)
> ncdataframe = cbind(expand.grid(vals), ncdata1)
```

Then find missing values at nonland locations and save the resulting data frame.

```
> misbyrow = apply(ncdataframe, 1, function(x)
+   sum(is.na(x)))
> ncdataframe = ncdataframe[misbyrow==0, ]
> save("ncdataframe", file="ncdataframe.RData")
```

Finally, to create a subset data frame with only July 2005 SST values on your latitude–longitude grid by typing

```
> sst = ncdataframe[paste("Y2005", "M",
+   formatC(7, 1, flag="0"), sep="")]
> names(sst) = "SST"
> sst$lon = ncdataframe$lon
> sst$lat = ncdataframe$lat
> write.table(sst, file="sstJuly2005.txt")
```

These data are used in Chapters 7 and 9.

This chapter showed how to extract hurricane data sets from raw data files. We began by showing how to create a spreadsheet-friendly flat file from the available best-tracks. We showed how to add value to these data by smoothing, interpolating, and computing derivative variables. We also showed how to parse the data regionally and locally and to create a subset based on lifetime maximum intensity. Next, we demonstrated how to aggregate the data annually and insert relevant environmental variables. We then examined a coastal county wind data set and showed how to work with NetCDF files.

Part II focuses on using these data to analyze and model hurricane activity. We begin, in Chapter 7 with models for hurricane frequency.