StatR 301: Spring 2013

Homework 04

Rod Doe

Sunday, June 2, 2013

Blowing in the Wind…

Package Building

In theory, to build a package, one must:

- Start with a clean R environment.
- Create all the functions and data required for the package.
- Come up with a name for the project (like, say, “Manahattan”, except, that one’s been used already, with pretty scary results. Let’s move downscale a bit and name it after the Bronx.)
- Set your working directory to somewhere you recognize.
- Invoke the function package.skeleton(name=”Bronx”).
- At the completion of the package.skeleton(name=”Bronx”) execution, there will be a directory in the working directory with the same name as the project (Bronx). At this point, it is necessary to run:
    - R CMD check, to determine what needs to be added/fixed.
        - There are some documentation items that must be addressed.
    - R CMD build, to assemble the package into a tarball for distribution.

I created a clear R environment, and added these functions to it:

- plotMap <- function(minLat, maxLat, minLong, maxLong, text = TRUE, pos = 1, cex = 0.8, font = 4)
- loadBuoyData <- function(buoyID)

I also added the Stations data frame to the package. Here is the code to accomplish all this wizardry. I especially liked the use of the intersect function to select applicable buoys in the selected region.:

```
# HW04.r
setwd("C:/Users/Rod/SkyDrive/R/301/Week10")

# Get station data.
require(RCurl)
require(mapdata)
url <- "http://www.ndbc.noaa.gov/stndesc.shtml"

a <- getURL(url)  # Returns an HTML document.
b <- sub(".*?<pre>(.*?)(</pre>.*|$)", "\\1", a)
```

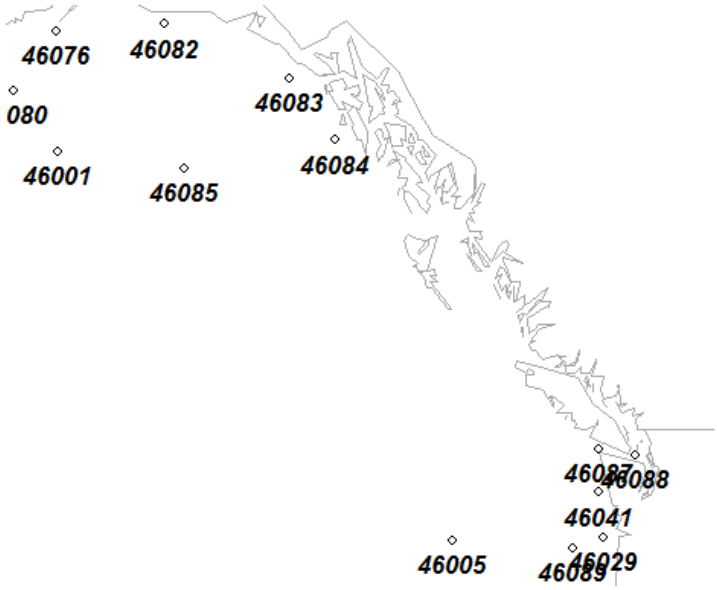
```
rm(url) # remove to exclude from package

# Messy conversion.  I might have used PowerShell to screen scrape here...
write(b, file = "StationData.txt", ncolumns = 12)
Stations <- read.table("StationData.txt", skip = 4, header = TRUE)
rm(a) # remove to exclude from package
rm(b) # remove to exclude from package

# Need to convert longitude from deg min sec E-W to a single signed value.
# Longitude is positive when it is east?
# Stations 85, 86 are in east longitude.
# 46069        3D89           33  40  13  N    120  12  0   W      1020.6
1799
# 46070        6N49           55  5   0   N    175  16  12  E      3804
3351
Latitude <- Stations$DEG + Stations$MIN/60 + Stations$SEC/60/60
Longitude <- (Stations$DEG.1 + Stations$MIN.1/60 + Stations$SEC.1/60/60) *
ifelse(Stations$E.W == "E", 1, -1)
Station <- Stations$STATION
# Simplify Stations dataframe.
Stations = data.frame(Station, Latitude, Longitude)

# Remove clutter for package generation.
rm(Station)
rm(Latitude)
rm(Longitude)

#write(as.matrix(df.buoys, ncols=length(df.buoys[1,])), file="NOAABuoys.txt")
#write(as.matrix(df.buoys), file="NOAABuoys.txt",
ncolumns=length(df.buoys[1,]) )

plotMap <- function(minLat, maxLat, minLong, maxLong, text = TRUE, pos = 1,
cex = 0.8, font = 4) {
      suppressWarnings(data(Stations))

      # Select indices of stations with valid latitude.
      idx.LE.max.lat = which(Stations$Latitude <= maxLat)
      idx.GE.min.lat = which(Stations$Latitude >= minLat)
      idx.valid.lat = intersect(idx.LE.max.lat, idx.GE.min.lat)

      # Select indices of stations with valid longitude.
      idx.LE.max.long = which(Stations$Longitude <= maxLong)
      idx.GE.min.long = which(Stations$Longitude >= minLong)
      idx.valid.long = intersect(idx.LE.max.long, idx.GE.min.long)

      # Select indices of stations with valid latitude and longitude.
      idx.valid.stations = intersect(idx.valid.lat, idx.valid.long)
      selected.stations = Stations[idx.valid.stations,]
```

```
      latitude = selected.stations$Latitude
      longitude = selected.stations$Longitude

      map("world", xlim = range(c(minLong, maxLong)), ylim = range(c(minLat,
maxLat)), col = "darkgrey")
      points(longitude, latitude, cex=0.8, font=4)
      text(longitude, latitude, labels=selected.stations$Station, pos=1,
font=4)
}


# My selected buoys are in the Gulf of Alaska.
# I spent my formative years in some rough weather on a tug out there.
# I've actually seen one of these buoys.  The one I saw was yellow.
# It got my attention because I thought it was a liferaft.
# A yellow thing in a gray sea gets one's attention.
# 46076  59.5N  148W
# 46082  60N 143W
# 46410  57N 144W

# buoys <- c('46067', '46082', '46410')
# buoyID <- 46076

# function to load buoy data
# More properly uses sprintf in lieu of paste.
# Paste is for elementary school kids.
loadBuoyData <- function(buoyID)
{
      # Load data from NOAA site.
      url <-
sprintf("http://www.ndbc.noaa.gov/view_text_file.php?filename=%dc2012.txt.gz&
dir=data/historical/cwind/", buoyID)
      df <- read.table(url, skip=1, header=TRUE, comment="*")

      # Convert data to acceptable format.
      Time <- as.POSIXlt(sprintf("%04d-%02d-%02d %02d:%02d:00", df$X.yr,
df$mo, df$dy, df$hr, df$mn))
    Speed <- df$m.s
    Direction <- df$degT

      # Construct a refined dataframe and return.
    df.tidy <- data.frame(Time, Speed, Direction)
      df.tidy
}
```

The plotMap function is quite cool. When invoked with these arguments:

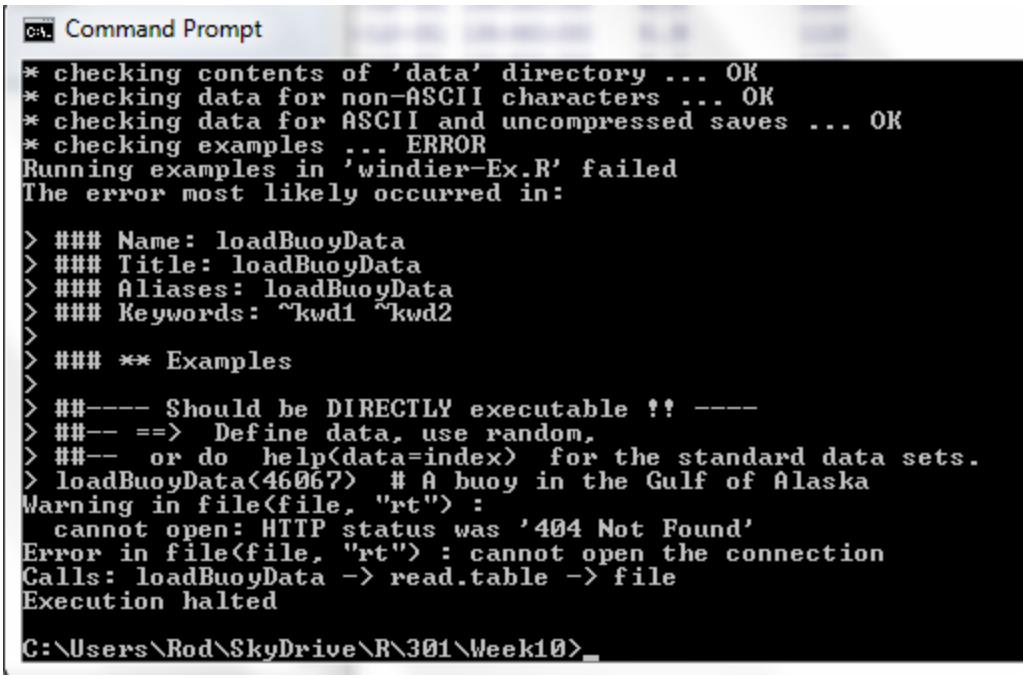
plotMap(45, 60, -150, -120)

It plots name Buoy ID and position of all NOAA weather stations between 45N, 120W and 60N 150W. Here is a screenshot:



NOTE: In my seafaring days (I went from seafaring to C-faring), I actually saw one of these buoys in the Bering Sea. It was yellow in a place where there wasn't supposed to be anything yellow. It was noted on the chart.

So far, so good. I then tried to create the tar ball (R CMB build), and that failed. I did a check (R CMD check), and saw that there were some documentation issues left. I fixed those to get the errors down to warnings, and tried to build again. The build failed with an HTTP 404 error (file not found).



Apparently, it failed because it was trying to execute the loadBuoyData function and it could not find data for station 46076, which was weird, because that was one of the buoys in the screen shot above. Oh well. Because of this, I could not get the build step to produce a tar ball. However, I used a tar utility and to cheat and create a tar ball anyways.

Time series analysis of wind speeds

I used as much of the lab as I could, and actually found a bug in the code for the lab. I noticed that when we converted wind direction in degrees to compass points (NESW), we occasionally got NA values for NESW. Those NA values occurred when the original wind direction was out of the north (Direction = 0).

```
# This leaves NA for direction 0.
nesw = cut(Direction, breaks=c(0,45,135,225,315,360), labels=c('N',
'E', 'S', 'W', 'N'))
```

The cut function was giving a warning about duplicate levels in factors:

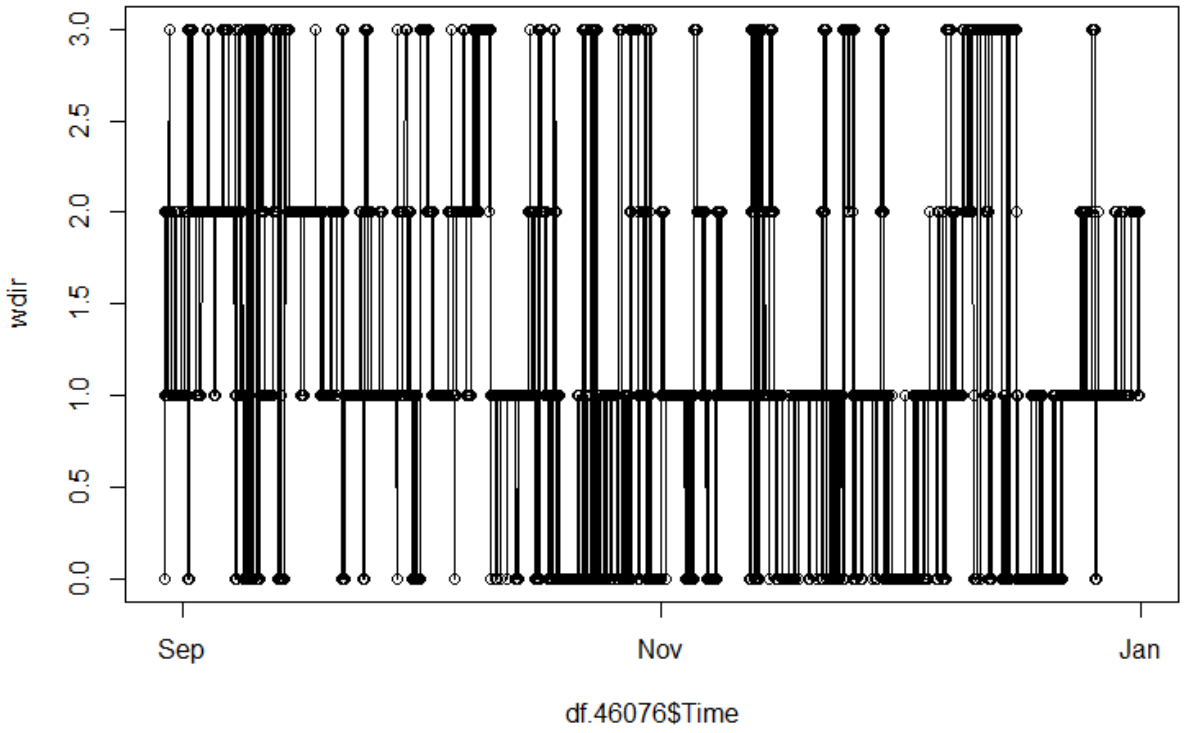
```
Warning message:
In `levels<-`(`*tmp*`, value = if (nl == nL) as.character(labels) else
paste0(labels,  :
  duplicated levels will not be allowed in factors anymore
```

Both of these techniques fixed the problem:

```
degToNESW <- function(deg) {
        nesw = 'N'
```

```
        if ((deg >= 45) && (deg < 135)) { nesw = 'E' }
        else if ((deg >= 135) && (deg < 225)) { nesw = 'S'}
        else if ((deg >= 225) && (deg < 315)) { nesw = 'W'}
        nesw
}

nesw = apply(as.matrix(df.46076$Direction), 1, degToNESW)

degToDir <- function(deg) {
        nesw = 0
        if ((deg >= 45) && (deg < 135)) { nesw = 1 }
        else if ((deg >= 135) && (deg < 225)) { nesw = 2}
        else if ((deg >= 225) && (deg < 315)) { nesw = 3}
        nesw
}

wdir = apply(as.matrix(df.46076$Direction), 1, degToDir)
```

This produced a fundamentally correct graph:



However, I ran out of time. My son is graduating from high school (on to Cal Poly for Electrical Engineering), and we have grandparents visiting from out of town.

Markov-chain analysis of wind direction

Ran out of time here. Too many graduation activities…