Downloading and Analyzing Weather Forecast Data with rNOMADS using grib

Daniel Bowman

March 22, 2017

Contents

1 Motivation

I designed the rNOMADS package to provide a quick and easy interface to the NOAA Operational Model Archive and Distribution System. It can retrieve data in ascii format (this works for all operating systems) or in a binary format called grib. This document describes downloading, reading, and displaying data in grib format. Please refer to the companion document for information on ascii data download via GrADS-DODS.

The grib portion of the rNOMADS package consists of two parts: a downloader that gathers data from the NOMADS website, and a reader that loads the model data into R. The reader requires the user to install **wgrib2**. wgrib2 is a set of routines for processing grib files and can be downloaded here. It

is simple to compile and install on Linux systems, and users have gotten it to work on Mac OS and Windows as well.

The rNOMADS package provides access to grib files for 58 models. This vignette shows how to download data from the Global Forecast System 0.5 x 0.5 degree model. This model describes the state of the atmosphere across the entire planet in 0.5 by 0.5 degree cells. The model divides the atmosphere into 64 layers. It produces forecasts for every 3 hours out to 192 hours.

The following examples demonstrate some useful things that can be accomplished with rNOMADS.

2 Making Global Weather Maps

Here, we make five maps. The first four are for the analysis forecast - the state of the weather at the time the model was run. Figure 1 is the temperature at 2 meters above the ground. High altitude regions such as Tibet and the Andes show up quite well. Figure 2 is the temperature at the 300 mb pressure level - approximately the height of the jet stream. Figure 3 is the relative humidity at 2 meters above the ground. Note how desert regions of the world are clearly depicted. Figure 4 is the wind speed at 300 mb. This is generally where the jet stream shows up. The last figure (Figure 5) is the surface wind gust forecast 6 hours from now. Hurricanes and tropical storms appear as small circular regions in this map.

2.1 Code

```
> #Get model data:
> #Planetary temperature, relative humidity, winds
> #at 2 m above ground and at 300 mb (jet stream level)
>
> library(GEOmap)
> library(rNOMADS)
> library(fields)
> library(aqfig)
> #Get the latest 2 model instances
> urls.out <- CrawlModels(abbrev = "gfs_0p50",
+     depth = 2, verbose = FALSE)
> #Get the available predictions.
```

```
> #If this throws an error, try urls.out[2]
> #because sometimes the web page appears before the data does
>
> model.parameters <- ParseModelPage(urls.out[2])
> latest.pred <- model.parameters$pred[1]
> levels <- c("2 m above ground", "300 mb")
> variables <- c("TMP", "RH", "UGRD", "VGRD")
> resolution <- c(0.5, 0.5)
> #Download model file
> grib.info <- GribGrab(urls.out[2], latest.pred,
+     levels, variables, verbose = FALSE)
> #Read the model file
> grb.data <- ReadGrib(grib.info[[1]]$file.name, levels, variables)
> #Make an array for quick indexing
> atmos <- ModelGrid(grb.data, resolution)

> #FIGURE 1
> #Temperature at ground level
>
>
> #Get variable and level indices
> li <- which(atmos$levels == "2 m above ground")
> vi <- which(atmos$variables == "TMP")
> #Set up color scale
> colormap <- rev(rainbow(100, start = 0 , end = 5/6))
> #Save image to PNG file
> #Omit this line if you want to display image
> #rather than save to file
> png(file = "fig_ground_temp.png", width = 1000, height = 750)
> #Make forecast image
> image(atmos$x + 180, sort(atmos$y), atmos$z[li,vi,,], col = colormap,
+     xlab = "Longitude", ylab = "Latitude",
+     main = paste("World Temperature at Ground Level:", atmos$fcst.date))
> #Plot coastlines
> plotGEOmap(coastmap, border = "black", add = TRUE,
+     MAPcol = NA, shiftlon = 180)
> vertical.image.legend(col=colormap,
+      zlim = range(atmos$z[li,vi,,]) - 272.15)
```

```
> #Turn of PNG device
> #Omit this line if you want to display image
> #rather than save to file
> dev.off()

null device
          1

> #FIGURE 2
> #Temperature at 300 mb
>
> li <- which(atmos$levels == "300 mb")
> vi <- which(atmos$variables == "TMP")
> colormap <- rev(rainbow(100, start = 0 , end = 5/6))
> png(file = "fig_300mb_temp.png", width = 1000, height = 750)
> image(atmos$x + 180, atmos$y, atmos$z[li,vi,,], col = colormap,
+     xlab = "Longitude", ylab = "Latitude",
+     main = paste("World Temperature at 300 mb:", atmos$fcst.date))
> plotGEOmap(coastmap, border = "black", add = TRUE,
+     MAPcol = NA, shiftlon = 180)
> vertical.image.legend(col=colormap,
+      zlim = range(atmos$z[li,vi,,]) - 272.15)
> dev.off()

null device
          1

> #FIGURE 3
> #Relative humidity at ground level
>
> li <- which(atmos$levels == "2 m above ground")
> vi <- which(atmos$variables == "RH")
> colormap <- rev(cm.colors(100))
> png(file = "fig_ground_rh.png", width = 1000, height = 750)
> image(atmos$x + 180, atmos$y, atmos$z[li,vi,,], col = colormap,
+     xlab = "Longitude", ylab = "Latitude",
+     main = paste("World Relative Humidity at Ground Level:",
+     atmos$fcst.date))
> plotGEOmap(coastmap, border = "black", add = TRUE,
```

```
+     MAPcol = NA, shiftlon = 180)
> vertical.image.legend(col=colormap,
+     zlim = range(atmos$z[li,vi,,]))
> dev.off()

null device
          1

> #FIGURE 4
> #Winds at 300 mb (around jet stream level)
>
> li <- which(atmos$levels == "300 mb")
> vi <- which(atmos$variables == "UGRD")
> ew.winds <- atmos$z[li,vi,,]
> vi <- which(atmos$variables == "VGRD")
> ns.winds <- atmos$z[li,vi,,]
> winds.vel <- sqrt(ew.winds^2 + ns.winds^2)
> colormap <-  terrain.colors(100)
> png(file = "fig_300mb_winds.png", width = 1000, height = 750)
> image(atmos$x + 180, atmos$y, winds.vel, col = colormap,
+     xlab = "Longitude", ylab = "Latitude",
+     main = paste("World Wind Velocity at 300 mb:", atmos$fcst.date))
> plotGEOmap(coastmap, border = "black", add = TRUE,
+     MAPcol = NA, shiftlon = 180)
> vertical.image.legend(col=colormap,
+     zlim = range(winds.vel))
> dev.off()

null device
          1

> #Get model data
> #and plot surface wind gust for the 6 hour forecast
>
> variables <- c("GUST")
> levels <- c("surface")
> resolution <- c(0.5, 0.5)
> #Get the latest 2 model instances
> urls.out <- CrawlModels(abbrev = "gfs_0p50",
```

```
+     depth = 2, verbose = FALSE)
> #Get the available predictions.
> #If this throws an error, try urls.out[2]
> #because sometimes the web page appears before the data does
>
> model.parameters <- ParseModelPage(urls.out[2])
> #Get 6 hr prediction
> #This will be 6 to 12 hours from now
> #depending on when the model was run
>
> pred.6hr <- model.parameters$pred[grep("06$", model.parameters$pred)]
> grib.info <- GribGrab(urls.out[2], pred.6hr,
+     levels, variables, verbose = FALSE)
> grb.data <- ReadGrib(grib.info[[1]]$file.name, levels, variables)
> #Make an array for quick indexing
> atmos <- ModelGrid(grb.data, resolution)
> li <- which(atmos$levels == "surface")
> vi <- which(atmos$variables == "GUST")
> #FIGURE 5
> png(file = "fig_ground_gust.png", width = 1000, height = 750)
> image(atmos$x + 180, atmos$y, atmos$z[li, vi, ,], col = colormap,
+     xlab = "Longitude", ylab = "Latitude",
+     main = paste("World Wind Gust at Ground Surface:", atmos$fcst.date))
> plotGEOmap(coastmap, border = "black", add = TRUE,
+     MAPcol = NA, shiftlon = 180)
> vertical.image.legend(col=colormap,
+     zlim = range(atmos$z[li,vi,,]))
> dev.off()

null device
          1

>
```

2.2 Figures

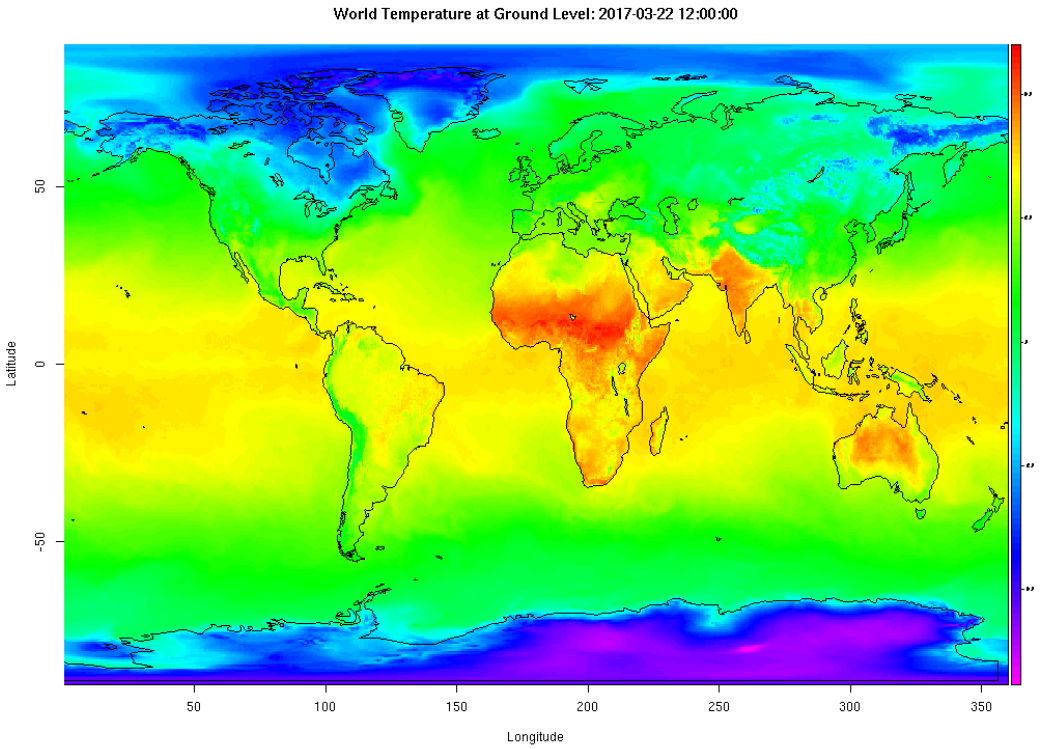

Figure 1: Temperature at 2 m above ground. Note how high altitude regions show up as cold spots.

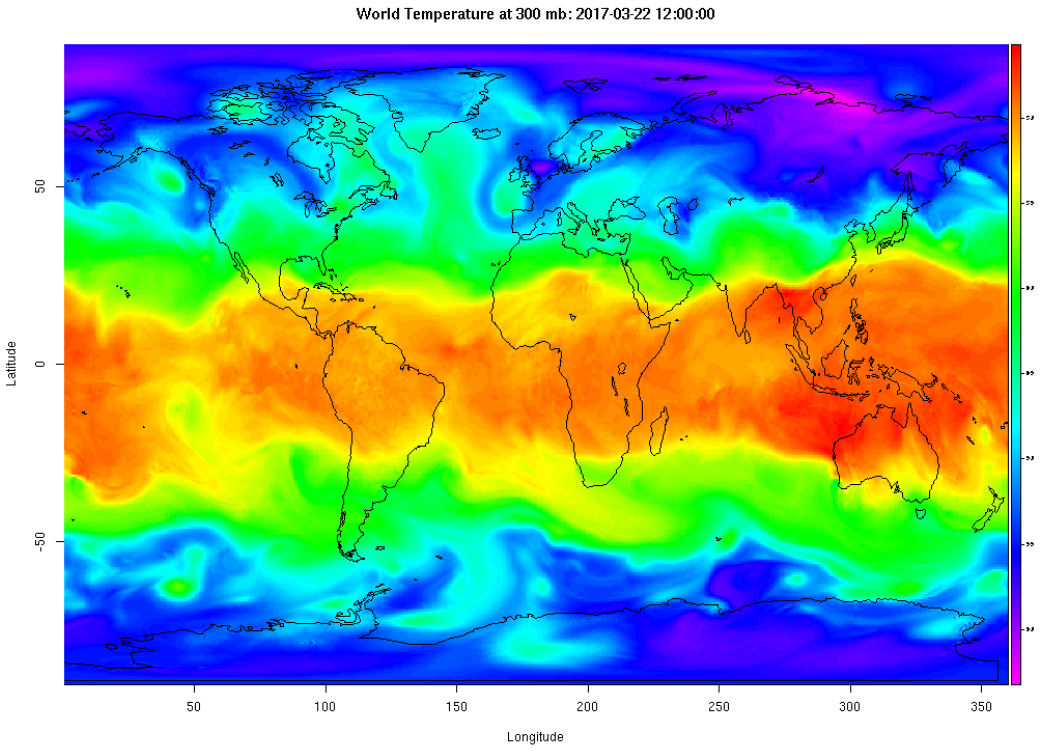

Figure 2: Temperature of the upper troposphere. Note how there is little relation between the location of continents and temperature.

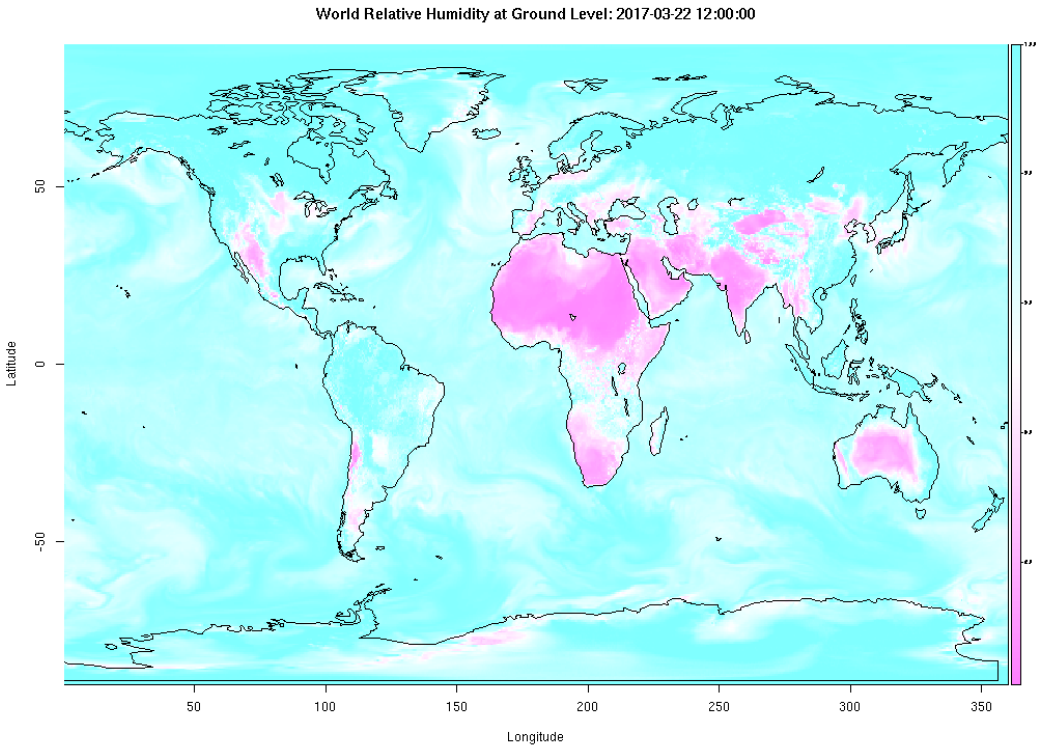

Figure 3: Relative humidity at 2 m above the ground. Desert regions show up nicely.

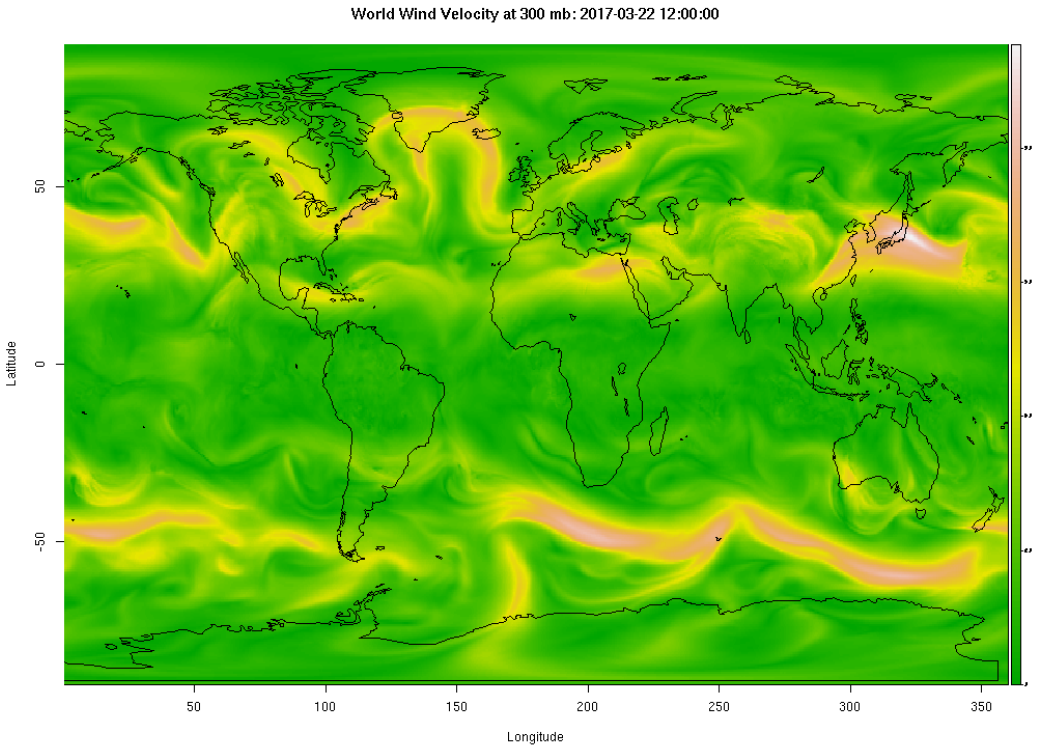

Figure 4: Wind velocity in the upper troposphere. Jet streams may show up here as patchy regions of intense wind.

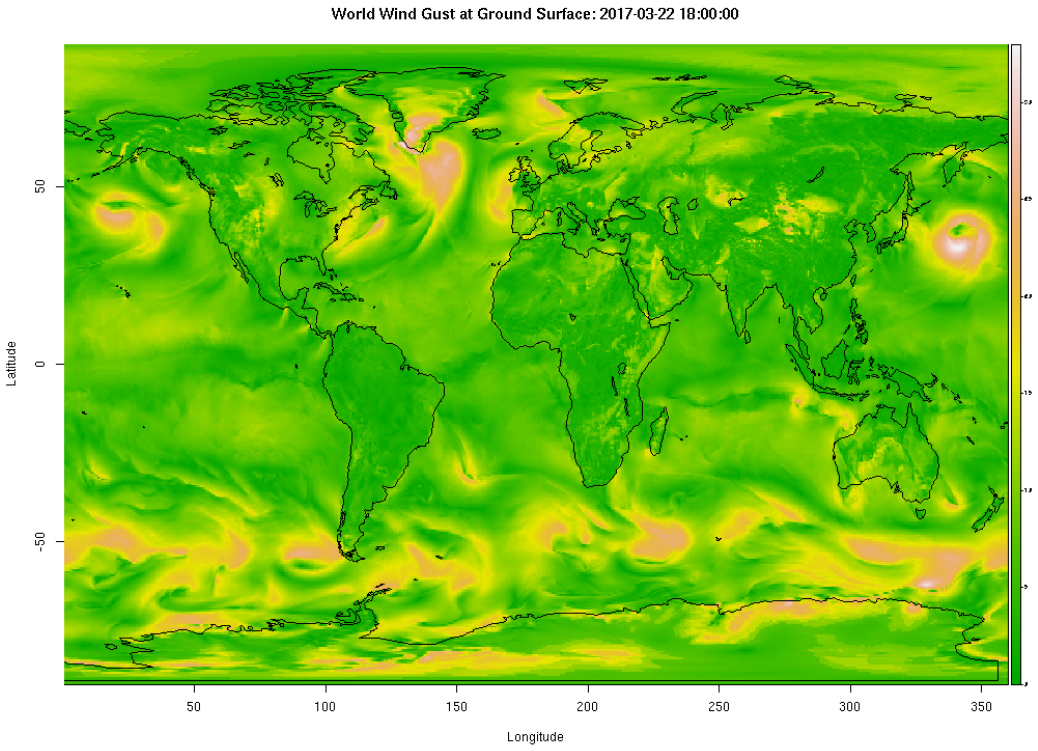

Figure 5: 6 hour forecast of wind gust speed at the ground surface. Hurricanes and tropical storms show up as small white dots.

3 Creating Atmospheric Profiles

We can generate data for specific points very quickly with rNOMADS. In this case, we will investigate the temperature and wind speed versus height directly above the University of North Carolina at Chapel Hill. The tropopause is clearly visible in the temperature profile (Figure 6). Wind speeds can be surprisingly high at times (Figure 7). We then show a quick calculation of sound velocity versus height (Figure 8) using the formula:

$$c = \sqrt{\gamma R_a T} \tag{1}$$

where c is the speed of sound, γ is the adiabatic index of air, R_a is the specific gas constant of air, and T is temperature. This does not take into account wind speed (a significant omission!).

Finally, we end with a calculation of atmospheric density versus height (Figure 9) via

$$\rho = \frac{p}{R_a T} \tag{2}$$

where ρ is density and p is pressure.

3.1 Code

```
> library(rNOMADS)
> #Get the latest 2 model instances
> urls.out <- CrawlModels(abbrev = "gfs_0p50",
+     depth = 2, verbose = FALSE)
> #Get the available predictions.
> #If this throws an error, try urls.out[2]
> #because sometimes the web page appears before the data does
>
> model.parameters <- ParseModelPage(urls.out[2])
> #Get latest prediction
>
> pred.now <- model.parameters$pred[1]
> #Get temperatures, winds, and elevations for whole atmosphere
>
> pressure <- c(1, 2, 3, 5, 7,
+    10, 20, 30, 50, 70,
```

```
+    seq(100, 1000, by = 25))
> levels <- paste(pressure, " mb", sep = "")
> #Height, temp, e-w wind, n-s wind
> variables <- c("HGT", "TMP", "UGRD", "VGRD")
> #Location to examine
> lon <- -79.052104
> lat <- 35.907553
> #Get grib file
> grib.info <- GribGrab(urls.out[2], pred.now, levels, variables,
+    model.domain = c(lon - 2, lon + 2, lat + 2, lat - 2))
> model.data <- ReadGrib(grib.info[[1]]$file.name, levels, variables)
> #Build atmospheric profile
> profile <- BuildProfile(model.data, lon, lat,
+     spatial.average = TRUE, points = 8)
> #Get height in meters
> hgt <- profile[[1]]$profile.data[, which(variables == "HGT"),]

> #FIGURE 6 - temperature
>
> tmp <- profile[[1]]$profile.data[, which(variables == "TMP"),]
> #Let's make a spline
>
> tmp.spline <- splinefun(hgt, tmp, method = "natural")
> synth.hgt <- seq(min(hgt), max(hgt), length.out = 1000)
> synth.tmp <- tmp.spline(synth.hgt)
> png(file = "fig_tmp_profile.png", width = 500, height = 500)
> plot(tmp - 272.15, hgt, pch = 19, col = "red",
+    xlab = "Temperature (C)", ylab = "Height (m)",
+    main = paste("Temperature versus Geopotential Height:",
+    profile[[1]]$forecast.date))
> lines(synth.tmp - 272.15, synth.hgt, col = "blue")
> legend("topright", col = c("red", "blue"), pch = c(19, NA),
+    lty = c(NA, 1), legend = c("Model Values", "Spline Fit"),
+    bg = "white")
> dev.off()

null device
          1
```

```
>

> #FIGURE 7 - Wind Wheel
>
> meridional <- profile[[1]]$profile.data[, which(variables == "VGRD"),]
> zonal <- profile[[1]]$profile.data[, which(variables == "UGRD"),]
> #Let's make a spline
>
> merid.spline <- splinefun(hgt, meridional, method = "natural")
> zone.spline  <- splinefun(hgt, zonal, method = "natural")
> synth.hgt <- seq(min(hgt), max(hgt), length.out = 1000)
> synth.merid <- merid.spline(synth.hgt)
> synth.zone  <- zone.spline(synth.hgt)
> png(file = "fig_wind_profile.png", width = 500, height = 500)
> PlotWindProfile(synth.zone, synth.merid, synth.hgt, lines = TRUE,
+     points = FALSE, elev.circles = c(0, 25000, 50000),
+     elev.labels = c(0, 25, 50),
+     radial.lines = seq(45, 360, by = 45), colorbar = TRUE, invert = FALSE,
+     point.cex = 2, pch = 19, lty = 1, lwd = 3,
+     height.range = c(0, 50000), colorbar.label = "Wind Speed (m/s)")
> dev.off()

null device
          1

>

> #FIGURE 8
> #NAIVE SOUND SPEED
>
> #Assuming it only depends on temperature
> #Let's make a spline
>
> R.sp  <- 287.058 #Specific gas constant of air
> gamma <- 1.4 #Adiabatic index
> tmp.spline <- splinefun(hgt, tmp, method = "natural")
> synth.hgt <- seq(min(hgt), max(hgt), length.out = 1000)
> synth.tmp <- tmp.spline(synth.hgt)
> c       <- sqrt(gamma * R.sp * tmp)
```

```
> synth.c <- sqrt(gamma * R.sp * synth.tmp)
> png(file = "fig_sound_profile.png", width = 500, height = 500)
> plot(c, hgt, pch = 19, col = "red",
+    xlab = "Speed of Sound (m/s)", ylab = "Height (m)",
+    main = paste("Speed of Sound versus Geopotential Height:",
+    atmos$fcst.date))
> lines(synth.c, synth.hgt, col = "blue")
> legend("topright", col = c("red", "blue"), pch = c(19, NA),
+    lty = c(NA, 1), legend = c("Model Values", "Spline Fit"),
+    bg = "white")
> dev.off()

null device
          1

>

> #FIGURE 8
> #ATMOSPHERIC DENSITY (ASSUMING DRY AIR)
>
> p <- as.numeric(gsub("\\D", "", profile[[1]]$levels)) * 100
> rho <- p / (tmp  * R.sp) #Air density
> rho.spline <- splinefun(hgt, rho, method = "natural")
> synth.hgt <- seq(min(hgt), max(hgt), length.out = 1000)
> synth.rho <- rho.spline(synth.hgt)
> png(file = "fig_density_profile.png", width = 500, height = 500)
> plot(rho, hgt, pch = 19, col = "red",
+    xlab = "Density (kg/m3)", ylab = "Height (m)",
+    main = paste("Dry Air Density versus Geopotential Height:",
+    atmos$fcst.date))
> lines(synth.rho, synth.hgt, col = "blue")
> legend("topright", col = c("red", "blue"), pch = c(19, NA),
+    lty = c(NA, 1), legend = c("Model Values", "Spline Fit"),
+    bg = "white")
> dev.off()

null device
          1

>
```

3.2 Figures

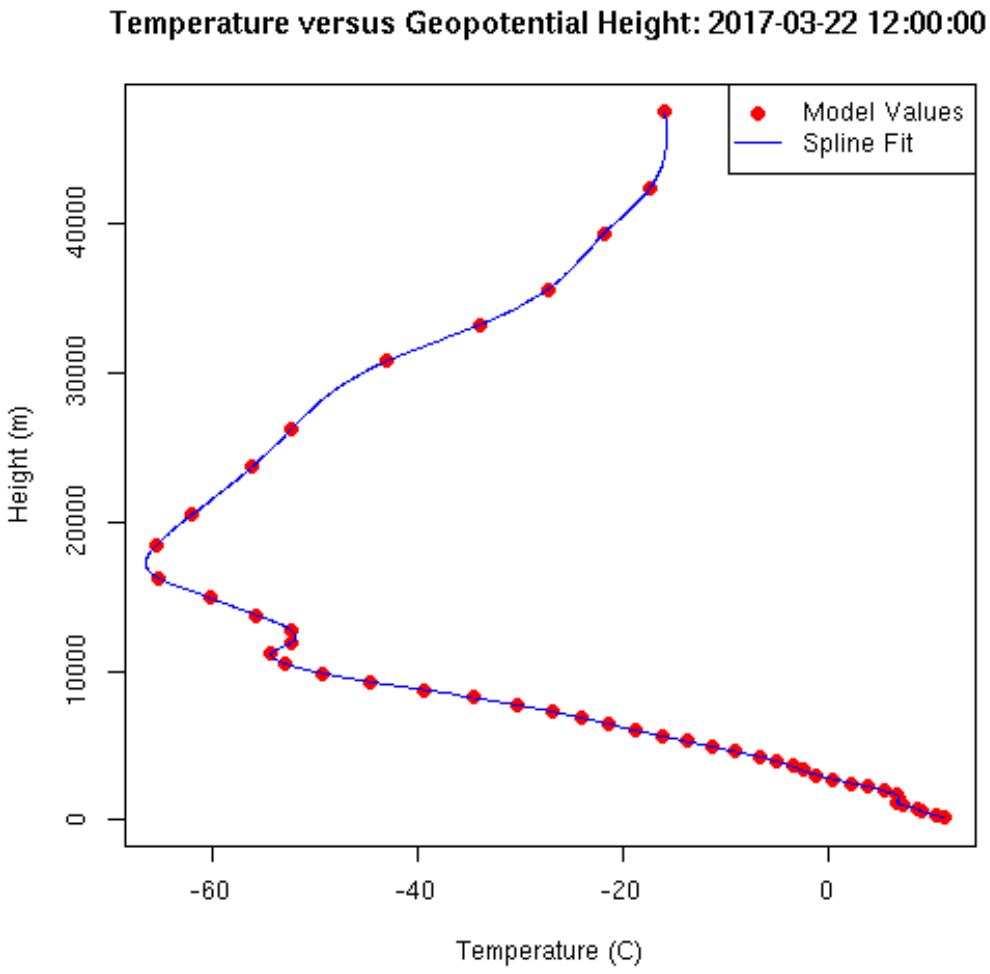

Figure 6: Temperature profile of the atmosphere near Chapel Hill, North Carolina.

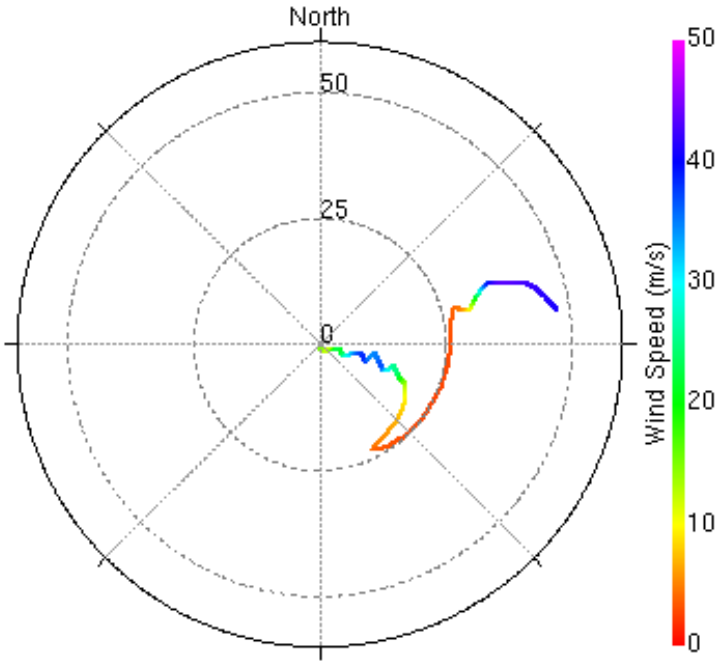

Figure 7: Wind profile of the atmosphere near Chapel Hill, North Carolina.

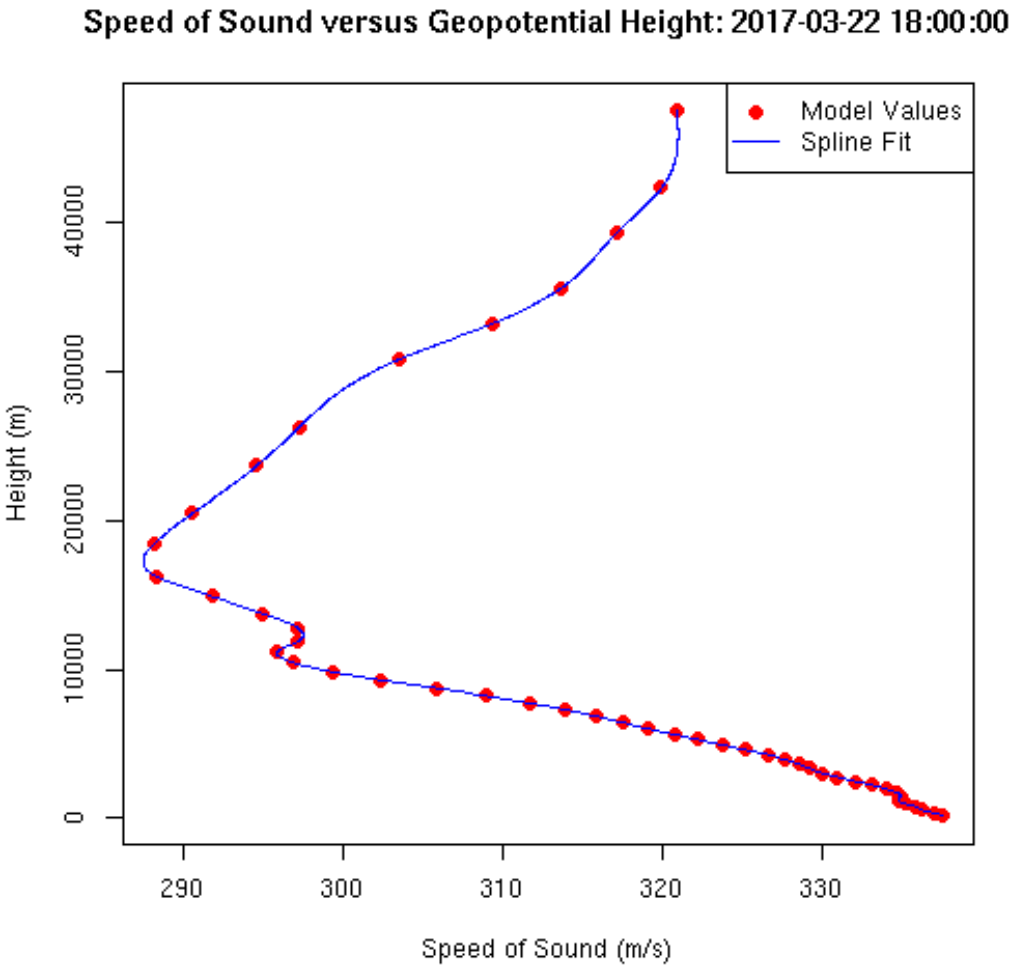

Figure 8: Sound speed profile of the atmosphere near Chapel Hill, North Carolina. This calculation assumes temperature dependence only.

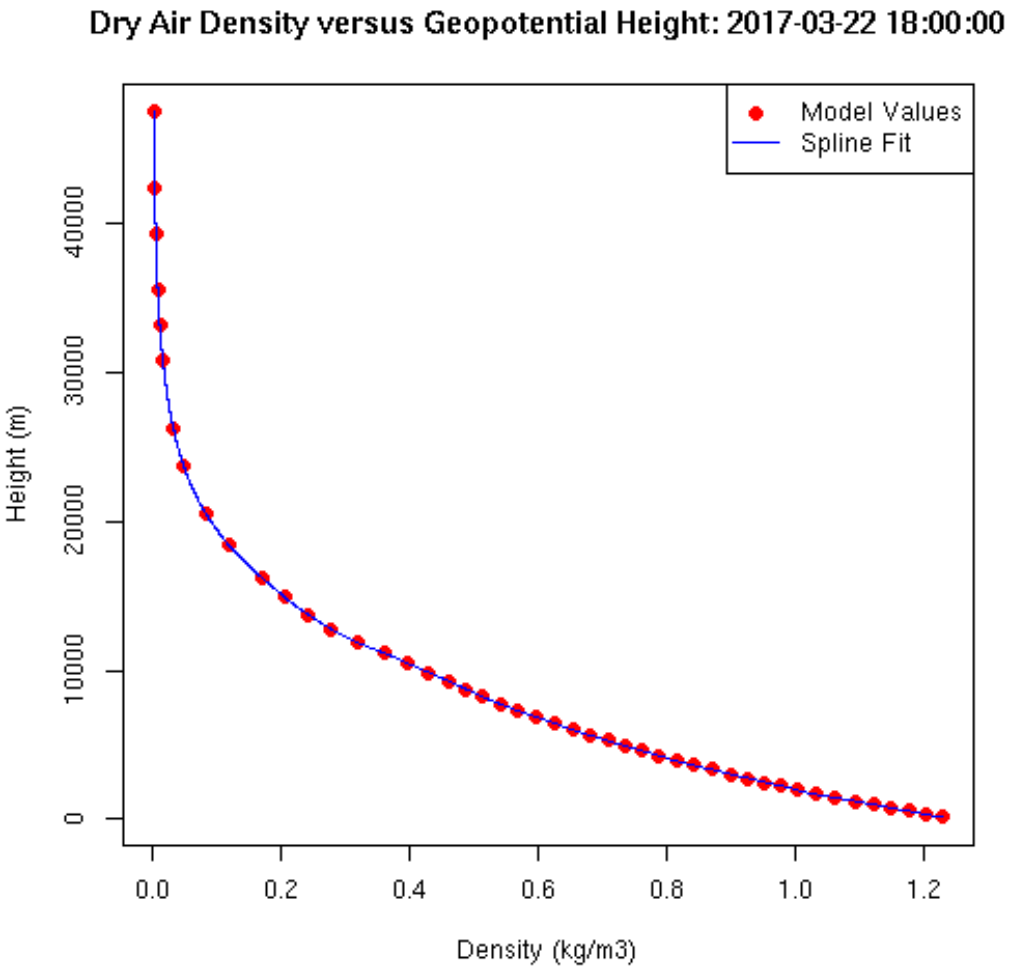

Figure 9: Air profile of the atmosphere near Chapel Hill, North Carolina.