STA 141B HW 1

Vincent Barletta

2023-04-10

Introduction

This is an exercise in reading in unconventional data tables and learning how to transform the tables into a more readable, usable format to get immediate insights from weather data across multiple geographical areas.

.wea files First, we will begin with the .wea files. These are debatably the easiest to begin with, and provide us with some basic data.

```
path_sf = "C:/Users/HP/Documents/141BProject1Files/USA_CA_Fairfield-San.Francisco.Bay.Reserve.998011_TM
path_marin = "C:/Users/HP/Documents/141BProject1Files/USA_CA_Marin.County.AP-Gnoss.Field.720406_TMYx.20
path_napa = "C:/Users/HP/Documents/141BProject1Files/USA_CA_Napa.County.AP.724955_TMYx.2007-2021.wea"
path_davis = "C:/Users/HP/Documents/141BProject1Files/USA_CA_UC-Davis-University.AP.720576_TMYx.2007-20
path_reyes = "C:/Users/HP/Documents/141BProject1Files/USA_CA_Point.Reyes.Lighthouse.724959_TMYx.2007-20

#According to Piazza, the .wea and .pvsyst files should have two seperate dataframes and .stat will hav

# We will start by reading in the first 6 lines which provides the metadata for the table.

#Table 1 - Metadata


weather_metadata <- function(path) {

  table_name <- readLines(path, n = 6)
  place <- substring(table_name[1], nchar("place") + 2)
  latitude <- as.numeric(substring(table_name[2], nchar("laditude") + 2))
  longitude <- as.numeric(substring(table_name[3], nchar("longitude") + 2))
  time_zone <- as.numeric(substring(table_name[4], nchar("time_zone") + 2))
  site_elevation <- as.numeric(substring(table_name[5], nchar("site_elevation") + 2))
  weather_data_file_units <- as.numeric(substring(table_name[6], nchar("weather_data_file_units") + 2))

  table_name <- data.frame(place, latitude, longitude, time_zone, site_elevation, weather_data_file_uni
}

sf_wea_meta <- weather_metadata(path_sf)
marin_wea_meta <- weather_metadata(path_marin)
napa_wea_meta <- weather_metadata(path_napa)
davis_wea_meta <- weather_metadata(path_davis)
pr_wea_meta <- weather_metadata(path_reyes)
```

```
#After creating the last function, I realized table_name does not matter as the assigned name is create

weather_data <- function(path) {
  table <- read.table(path, header = FALSE, sep = " ", skip = 6)
  wea_column_names <- c("month", "day", "standard time", "direct normal irradiance", "diffuse horizontal
  colnames(table) <- wea_column_names
  return(table)
}

fairfield_wea <-weather_data(path_sf)
marin_wea_meta <- weather_metadata(path_marin)
napa_wea <- weather_metadata(path_napa)
davis_wea <- weather_metadata(path_davis)
pr_wea <- weather_metadata(path_reyes)
```

PVSyst

Let's start with the metadata. We can skip the first three lines as they provide no data.

```
path_sf = "C:/Users/HP/Documents/141BProject1Files/USA_CA_Fairfield-San.Francisco.Bay.Reserve.998011_TM
path_marin = "C:/Users/HP/Documents/141BProject1Files/USA_CA_Marin.County.AP-Gnoss.Field.720406_TMYx.20
path_napa = "C:/Users/HP/Documents/141BProject1Files/USA_CA_Napa.County.AP.724955_TMYx.2007-2021.pvsyst
path_davis = "C:/Users/HP/Documents/141BProject1Files/USA_CA_UC-Davis-University.AP.720576_TMYx.2007-20
path_reyes = "C:/Users/HP/Documents/141BProject1Files/USA_CA_Point.Reyes.Lighthouse.724959_TMYx.2007-20

 pv_metatable <- function(path) {
    file_lines <- readLines(path, n = 12)
    file_lines <- file_lines[4:12]
    file_lines <- file_lines[-4]
    split <- strsplit(file_lines, ",")

  #We want to get rid of all of the # in front of our data values
  for(i in seq_along(split)) {
    split[[i]][1] <- gsub("#", "", split[[i]][1])
  }

  #I couldn't find a good way to split the string directly, so we take two substrings and then split th

  WMO<- substring(split[[1]],1,10)
  DataSource<- substring(split[[1]],11)

  split[[9]] <- strsplit(DataSource, "=")
  split[[1]] <- strsplit(WMO, "=")

  split <- split[1:9]
  metadata_table_pv <- data.frame(split)
  #I tried numerous times to try and get the indexing correct so that the df_list would only take the s

  #Therefore, we will keep the names, and simply scrap the first row from the table.
  names(metadata_table_pv) <- metadata_table_pv[1,]
  metadata_table_pv <- metadata_table_pv[2,]
  metadata_table_pv
```

```
 }

sf_pvsyst_metadata <- pv_metatable(path_sf)
head(sf_pvsyst_metadata)
```

```
##      WMO                               Site Country Time step Latitude
## 2 998011 Fairfield-San.Francisco.Bay.Reserve     USA      Hour   38.200
##   Longitude Altitude Time Zone Data Source
## 2  -122.026        4     -8.00    SRC-TMYx
```

```
marin_pvsyst_metadata <- pv_metatable(path_marin)
head(marin_pvsyst_metadata)
```

```
##      WMO                         Site Country Time step Latitude Longitude
## 2 720406 Marin.County.AP-Gnoss.Field     USA      Hour   38.150  -122.550
##   Altitude Time Zone Data Source
## 2        1     -8.00    SRC-TMYx
```

```
davis_pvsyst_metadata <- pv_metatable(path_davis)
davis_pvsyst_metadata
```

```
##      WMO                   Site Country Time step Latitude Longitude Altitude
## 2 720576 UC-Davis-University.AP     USA      Hour   38.533  -121.783       21
##   Time Zone Data Source
## 2     -8.00    SRC-TMYx
```

```
pointreyes_pvsyst_metadata <- pv_metatable(path_reyes)
pointreyes_pvsyst_metadata
```

```
##      WMO                   Site Country Time step Latitude Longitude Altitude
## 2 724959 Point.Reyes.Lighthouse     USA      Hour   37.996  -123.023       87
##   Time Zone Data Source
## 2     -8.00    SRC-TMYx
```

```
napa_pvsyst_metadata <- pv_metatable(path_napa)
napa_pvsyst_metadata
```

```
##      WMO           Site Country Time step Latitude Longitude Altitude Time Zone
## 2 724955 Napa.County.AP     USA      Hour   38.210  -122.285       11     -8.00
##   Data Source
## 2    SRC-TMYx
```

```
#Now, for the actual data...

pvsyst_table <- function(path) {
  ll = readLines(path)[-c(1:12,14)]
  con = textConnection(ll)
  table <- read.table(con, header = TRUE, sep = ",")
  return(table)
```

```
}

sf_pvsyst_data <- pvsyst_table(path_sf)
head(sf_pvsyst_data)
```

```
##   Year Month Day Hour Minute GHI DHI DNI Tamb WindVel WindDir
## 1 2059     1   1    1     30   0   0   0    5       4      90
## 2 2059     1   1    2     30   0   0   0    4       4      40
## 3 2059     1   1    3     30   0   0   0    3       5      70
## 4 2059     1   1    4     30   0   0   0    1       2      40
## 5 2059     1   1    5     30   0   0   0    1       3      80
## 6 2059     1   1    6     30   0   0   0    1       3      70
```

```
marin_pvsyst_data <- pvsyst_table(path_marin)
head(marin_pvsyst_data)
```

```
##   Year Month Day Hour Minute GHI DHI DNI Tamb WindVel WindDir
## 1 2059     1   1    1     30   0   0   0    7       0     316
## 2 2059     1   1    2     30   0   0   0    5       2     170
## 3 2059     1   1    3     30   0   0   0    4       0     349
## 4 2059     1   1    4     30   0   0   0    5       2     190
## 5 2059     1   1    5     30   0   0   0    2       0      80
## 6 2059     1   1    6     30   0   0   0    2       0      98
```

```
davis_pvsyst_data <- pvsyst_table(path_davis)
head(davis_pvsyst_data)
```

```
##   Year Month Day Hour Minute GHI DHI DNI Tamb WindVel WindDir
## 1 2059     1   1    1     30   0   0   0    4       5     350
## 2 2059     1   1    2     30   0   0   0    4       5     340
## 3 2059     1   1    3     30   0   0   0    4       6     340
## 4 2059     1   1    4     30   0   0   0    5       7     340
## 5 2059     1   1    5     30   0   0   0    5       6     340
## 6 2059     1   1    6     30   0   0   0    5       8     340
```

```
pointreyes_pvsyst_data <- pvsyst_table(path_reyes)
head(pointreyes_pvsyst_data)
```

```
##   Year Month Day Hour Minute GHI DHI DNI Tamb WindVel WindDir
## 1 2059     1   1    1     30   0   0   0   12       3     350
## 2 2059     1   1    2     30   0   0   0   12       3      10
## 3 2059     1   1    3     30   0   0   0   12       2     280
## 4 2059     1   1    4     30   0   0   0   13       3      40
## 5 2059     1   1    5     30   0   0   0   14       6      80
## 6 2059     1   1    6     30   0   0   0   14       2      30
```

```
napa_pvsyst_data <- pvsyst_table(path_napa)
head(napa_pvsyst_data)
```

```
##   Year Month Day Hour Minute GHI DHI DNI Tamb WindVel WindDir
## 1 2059     1   1    1     30   0   0   0    9       4     240
## 2 2059     1   1    2     30   0   0   0   10       3     220
## 3 2059     1   1    3     30   0   0   0    7       3      20
## 4 2059     1   1    4     30   0   0   0    4       3     330
## 5 2059     1   1    5     30   0   0   0    3       0     236
## 6 2059     1   1    6     30   0   0   0    3       2     300
```

STAT Files

```
dir = "C:/Users/HP/Documents/141BProject1Files"
path = file.path(dir, "USA_CA_Fairfield-San.Francisco.Bay.Reserve.998011_TMYx.2007-2021.stat")

#Reads in all of the tables
#Next: needs to make adjustments to tables

stat_table <- function(start_name, end_name = "Daily Avg", colname = "Statistic", type = "monthly", pat

  master = readLines(path)

  start = grep(start_name, master)

  if(type == "hourly") {

    #We use grep to find all the occurences of "Min Hour" in our master document. We then see of all of

    end = grep("Min Hour", master)[min(which(grep("Min Hour", master) > start))]
    colname = "Hour"
  }

  else {
    end = grep(end_name, master)[min(which(grep(end_name, master) > start))]
  }

  # We read in the table up until the last line of the table we need, and then subset out lines 1-Start

  ll = readLines(path, n=end)[-(1:start)]
  con = textConnection(ll)
  table <- read.table(con, header = TRUE, sep = "\t")[-c(1,15)]
  colnames(table)[1] <- colname
  table

}

hourly_bulb <- stat_table(start_name = "Average Hourly Statistics for Dry Bulb temperatures", type = "h
hourly_dewpoint <- stat_table(start_name = "Average Hourly Statistics for Dew Point temperatures", type
hourly_humidity <- stat_table(start_name = "Average Hourly Relative Humidity", type = "hourly" , path =
hourly_solar <- stat_table(start_name = "Average Hourly Statistics for Direct Normal Solar Radiation",
hourly_windspeed <- stat_table(start_name = "Average Hourly Statistics for Wind Speed", type = "hourly"

monthly_wind <- stat_table(start_name = "Monthly Wind Direction", end_name = "NNW", colname = "Directio
```

```
monthly_drybulb <- stat_table(start_name = "Monthly Statistics for Dry Bulb temperatures", end_name = "
monthly_dewpoint <- stat_table(start_name = "Monthly Statistics for Dew Point temperatures" , path = pa
monthly_windspeed <- stat_table(start_name = "Monthly Statistics for Wind Speed" , path = path)
```

Table Generation

```
#Transposing Monthly List

monthly_transformations <- function(table) {

  df = data.frame(t(table))
  colnames(df) = df[1,]
  df = df[-1,]
  rownames <- rownames(df)

   #if argument suggested by ChatGPT; it takes the name of the passed in value, deparses it, and checks
   # This is done to scrape the table name passed in; I did not know a better way to do it. In this cas

  if (identical(deparse(substitute(table)), "monthly_wind")) {

    df <- lapply( df, as.numeric )
    #When you change the entire table to numerics using lapply
    df_names <- names( df )
    df <- data.frame( df, row.names = rownames )
    colnames(df) <- df_names
    return(df)

  }

  else {

  #Day:Hour Max
  df[ , 2 ] <- as.POSIXct(paste("2023", rownames(df), df[,2], sep = "-"), format = "%Y-%B-%d:%H")

  #Day:Hour Min
  df[ , 4 ] <- as.POSIXct(paste("2023", rownames(df), df[,4], sep = "-"), format = "%Y-%B-%d:%H")

  #Change the rest into Numerics
  df[, c(-2,-4)] <- lapply(df[, c(-2,-4)], as.numeric)
  df

  }
}

monthly_dewpoint <- monthly_transformations(monthly_dewpoint)
monthly_drybulb <- monthly_transformations(monthly_drybulb)
monthly_windspeed <- monthly_transformations(monthly_windspeed)
monthly_wind <- monthly_transformations(monthly_wind)
```

Monthly Table Transformations

Hourly Table Transformation Functions

Our central tasks here are to confirm that the listed max and min hours are indeed where the maximum and minimum values occur, and to convert all tables to numeric format. For whatever reason, the Solar table has some char columns, so we need to make sure that everything is a uniform data format as we will transform the table in the future and combine it with the other tables.

The Max-Min rows at the bottom correspond to the line numbers where the maximum occurs, not the hour. This means that it is actually usually the hour before it. Max Hour = 16 means the maximum temperature, for example, occurs from 15:00-16:00

```
#Checking Maxes

maxChecker <- function(table) { #This function scans through all of the columns, finds the max in each
# It then removes the Maximum Value row (n = 25) from the table
  mapply(function(col, maxIndex) col[maxIndex] == max(col), table[1:24, 2:13], table[25, 2:13])
  print("All max hour values match")
  table[-25,]
}

minChecker <- function(table) { # This function does the same as the one above, but with the min value
  mapply(function(col, minIndex) col[minIndex] == min(col), table[1:24, 2:13], table[26, 2:13])
  print("All min hour values match")
  table[-26,]
}

all_numeric <- function(table_name) {
  df = table_name
  rownames <- rownames(df)
  df[ , -1] <- lapply( df[, -1], as.numeric )
  df_names <- names( df )
  df <- data.frame( df, row.names = rownames )
  colnames(df) <- df_names
  df
}
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.2.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.2.3
```

```
hourly_transformation <- function(table_name) {

df = table_name

df <- minChecker(df)
df <- maxChecker(df)

df <- all_numeric(df)

#Here we call the minChecker and maxChecker

# Pivot_longer is a tidyr method suggested by ChatGPT. Essentially, it transforms our table from being

#We do not want to change the Hour column, so we leave it out. All of the values are put under the Valu

df <- df %>%
  pivot_longer(-Hour, names_to = "Month", values_to = "Value")

# Now, we simply scrape the first two characters of the Hour string to get the starting Hour value from
df <- df %>%
  mutate(Hour = substring(Hour,1,2) %>% as.integer())

# In order to properly arrange our data, we have to change our Month values into factors. They match th
df$Month <- factor(df$Month, levels = month.abb, ordered = TRUE)

# Lastly, we can arrange our data in order based on Month and Hour.

df <- df %>% arrange(Month, Hour)

df

}

hourly_bulb <- hourly_transformation(hourly_bulb)
```

```
## [1] "All min hour values match"
## [1] "All max hour values match"
```

```
hourly_windspeed <- hourly_transformation(hourly_windspeed)
```

```
## [1] "All min hour values match"
## [1] "All max hour values match"
```

```
hourly_dewpoint <- hourly_transformation(hourly_dewpoint)
```

```
## [1] "All min hour values match"
## [1] "All max hour values match"
```

```
hourly_solar <- hourly_transformation(hourly_solar)
```

```
## [1] "All min hour values match"
## [1] "All max hour values match"
```

```
hourly_humidity <- hourly_transformation(hourly_humidity)
```

```
## [1] "All min hour values match"
## [1] "All max hour values match"
```

Validating the Data It's important that we recheck the structure of everything at this point before we combine the tables to ensure that the joining happens as planned.

```
data_validation <- function(table_name) {
  print(dim(table_name))
  print(str(table_name))
  print(class(table_name))
}

data_validation(hourly_bulb)
```

```
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 7.6 7.3 6.9 6.9 6.7 6.4 6.3 6.2 6.3 7.4 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
data_validation(hourly_windspeed)
```

```
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 1.3 1.7 1.8 1.9 1.7 1.7 1.7 1.5 1.8 1.9 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
data_validation(hourly_dewpoint)
```

```
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 5.8 5.8 5.8 5.7 5.5 5.2 5.3 5.2 4.9 5.9 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
data_validation(hourly_solar)
```

```
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 0 0 0 0 0 0 0 137 327 404 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
data_validation(hourly_humidity)
```

```
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 90 92 93 93 93 93 94 94 93 92 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
join_tables <- function(table1, table2, table3, table4, table5) {

# Here we commit a lot of left joins to get all of the tables in line with the first table. Because the

combined_table <- table1 %>%
  left_join(table2, by = join_by(Hour, Month)) %>%
  left_join(table3, by = join_by(Hour, Month)) %>%
  left_join(table4,by = join_by(Hour, Month)) %>%
  left_join(table5, by = join_by(Hour, Month))

# Here, we change the names of each column because it simply uses default naming.
names <- c( Hourly.Bulb = "Value.x" , Hourly.Dewpoint = "Value.y", Hourly.Solar = "Value.x.x", Hourly.W

combined_table <- combined_table %>% rename(all_of(names))

combined_table
}

combined_table <- join_tables(hourly_bulb, hourly_dewpoint, hourly_solar, hourly_windspeed, hourly_humi
```

Joining the tables together

Creating Plots and Data Verification

This is a vital to not only visualizing our findings, but also to verify that our results are consistent with general common sense.

Our general assumptions would be that:

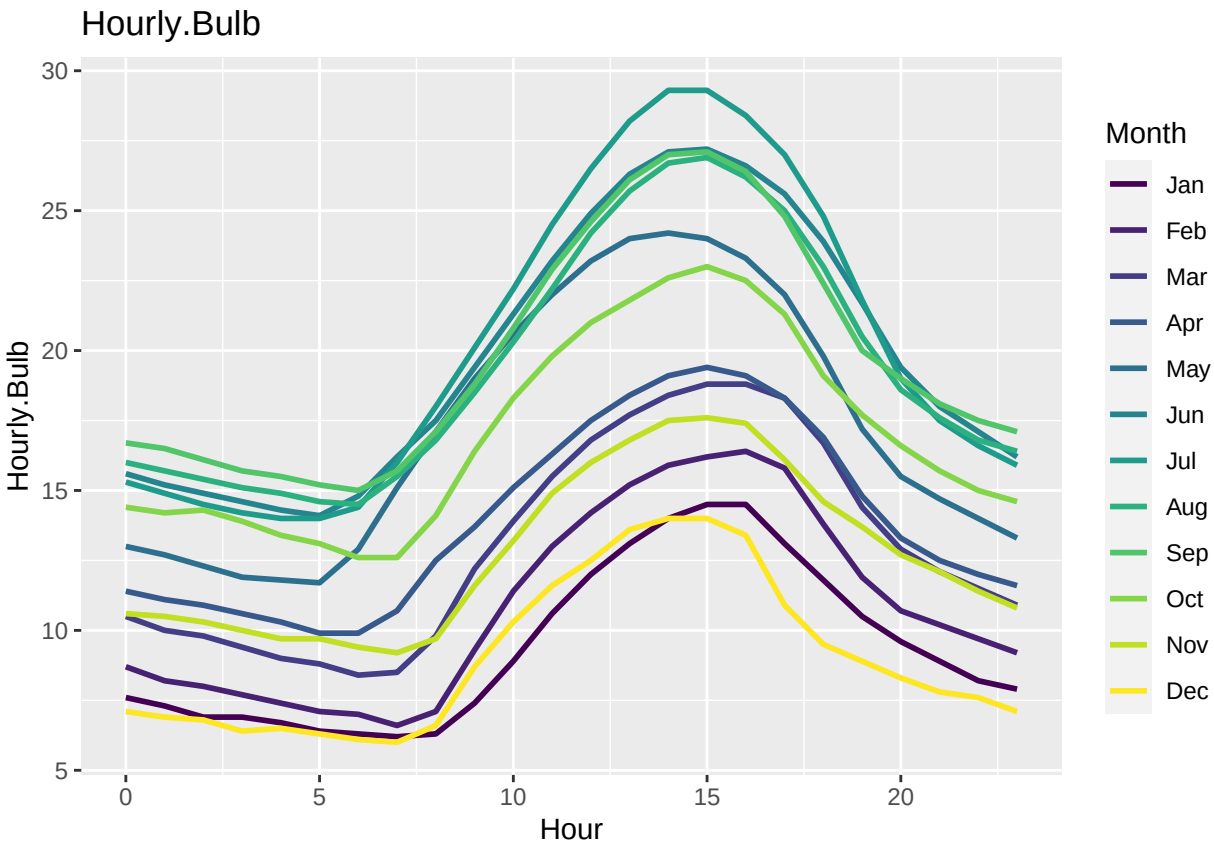
1. Drybulb and Dewpoint temperatures are higher during the summer months, and are highest when the sun is at its apex during the day (2-3 PM).
2. Solar values are higher during the summer, given that the sun is out for longer and the temperatures are higher.

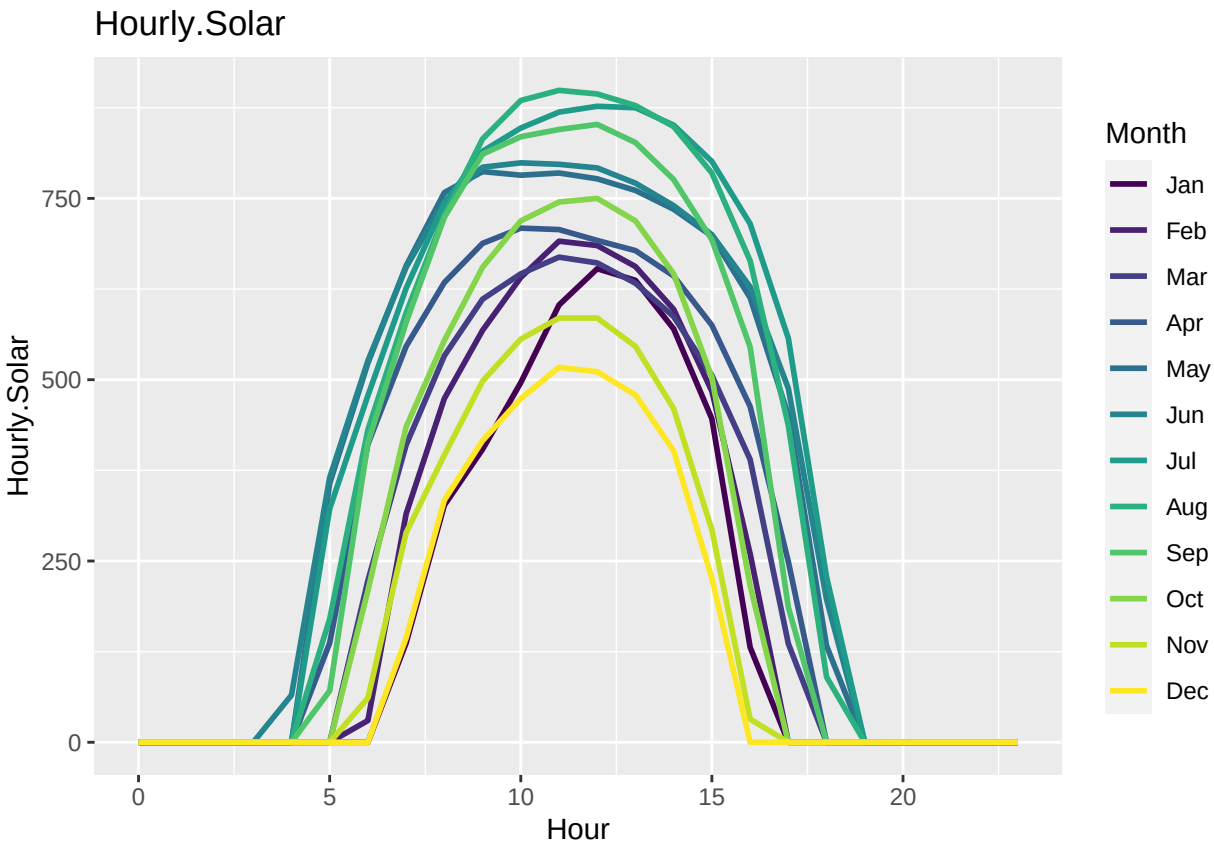
It is more difficult to predict windspeed and humidity, but we can confirm the validity of our results by comparing how the graphs look across the different datasets.

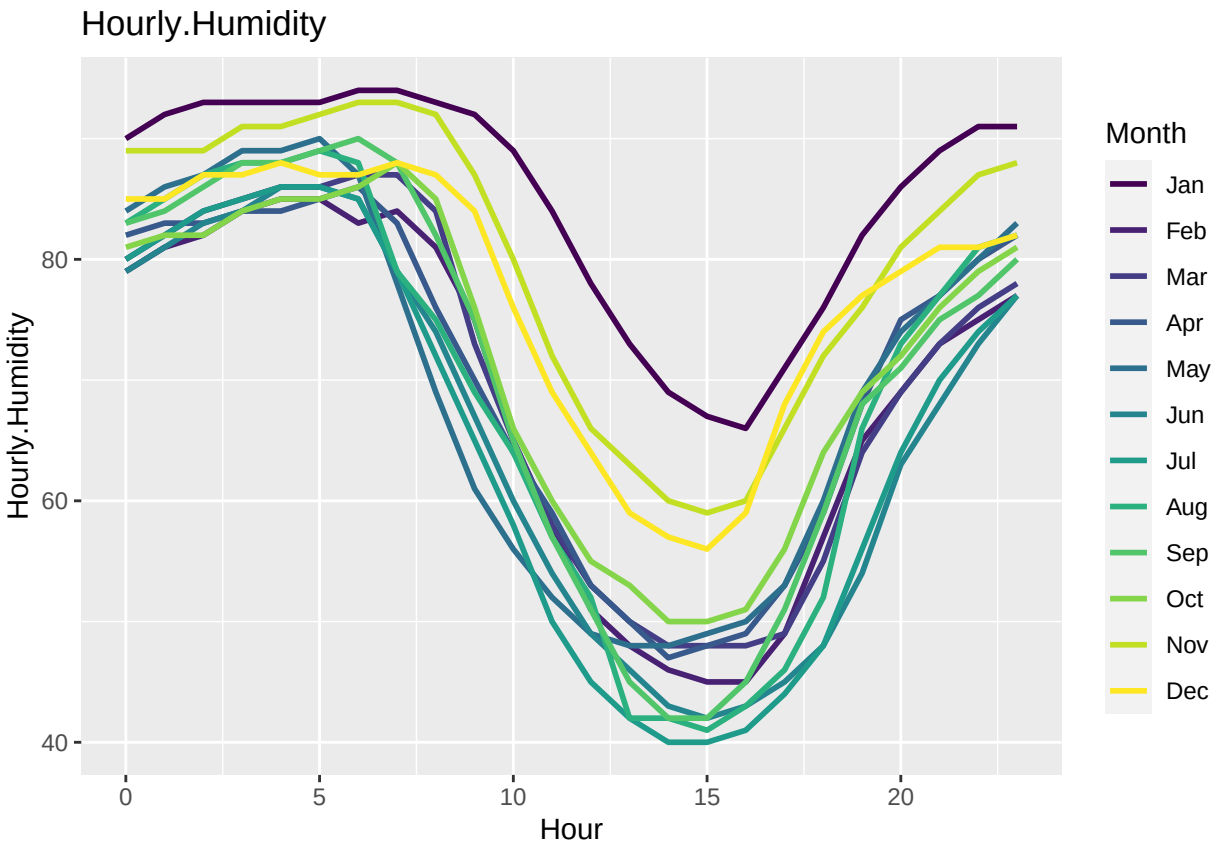
```
library(ggplot2)
```

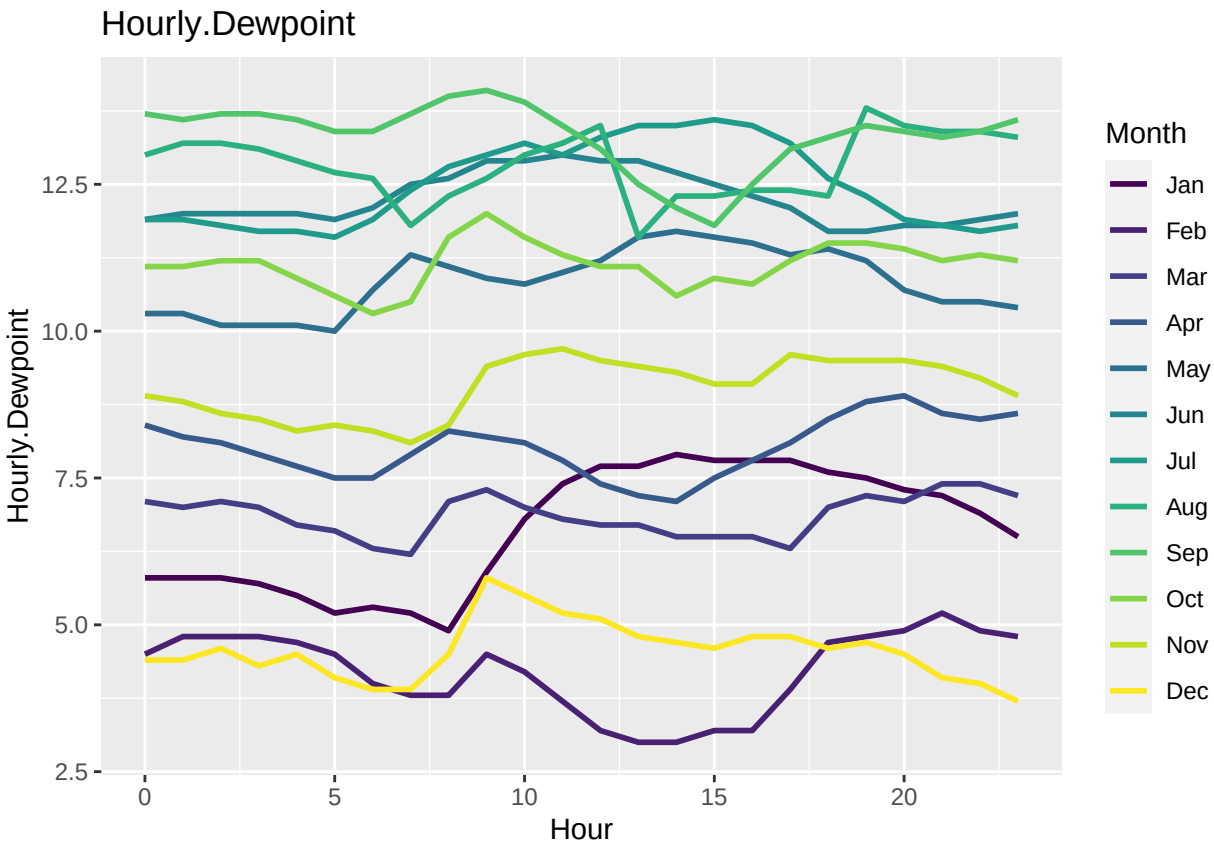
```
## Warning: package 'ggplot2' was built under R version 4.2.2
```

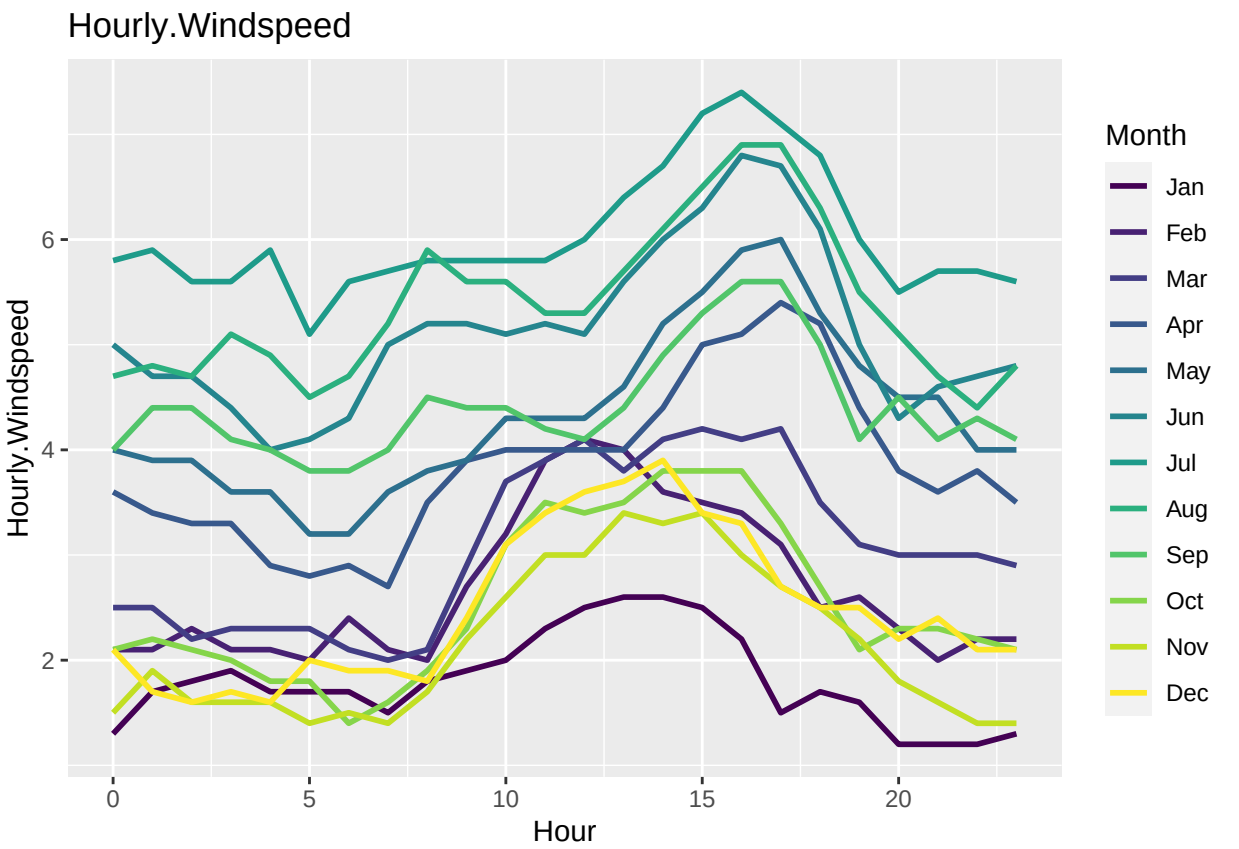
```
create_plots <- function(combined_table) {

#Each plot will be the same. We use the Hour as the x axis, the value for the y-axis, and color accordi
g1 <- ggplot(combined_table, aes(x = Hour, y = Hourly.Bulb, color = Month)) +
  geom_line(linewidth = 1) + ggtitle("Hourly.Bulb")

g2 <- ggplot(combined_table, aes(x = Hour, y = Hourly.Solar, color = Month)) +
  geom_line(linewidth = 1) + ggtitle("Hourly.Solar")

g3 <- ggplot(combined_table, aes(x = Hour, y = Hourly.Humidity, color = Month)) +
  geom_line(linewidth = 1) + ggtitle("Hourly.Humidity")

g4 <- ggplot(combined_table, aes(x = Hour, y = Hourly.Dewpoint, color = Month)) +
  geom_line(linewidth = 1) +   ggtitle("Hourly.Dewpoint")

g5 <- ggplot(combined_table, aes(x = Hour, y = Hourly.Windspeed, color = Month)) +
 geom_line(linewidth = 1) +   ggtitle("Hourly.Windspeed")

print(g1)
print(g2)
print(g3)
print(g4)
print(g5)
}


create_plots(combined_table)
```


Hourly.Bulb
Hourly.Bulb
30
25
20
15
10
5
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Solar
Hourly.Solar
750
500
250
0
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Humidity
Hourly.Humidity
80
60
40
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Dewpoint
Hourly.Dewpoint
12.5
10.0
7.5
5.0
2.5
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec



From this first dataset from the San Francisco-Fairfield region, we can see some general trends in the graphs that occur yearlong.

Firstly, looking at the Drybulb temperatures, they follow a very similar curve regardless of the time of year. While initially starting low, it steadily rises from 8 AM before peaking at 3 PM, and slowly decreasing as the sun sets into the night. Perhaps unsurprisingly, the highest temperatures occur in the summer months from May - August.

The Solar graph follows a parabolic curve. When you consider that this is time series data, it makes perfect sense! There is no sun out very early in the morning or late at night. It is interesting to see how much later the sun is out in the summer months compared to the winter months. Due to a larger amount of clouds in the Winter months, there is even less solar energy than normal, even at peak daytime hours.

The humidity graph shows us that it is lowest mid-day (3 PM) and higher during the nighttime. Humidity is also highest during the winter months, with January having a much higher amount than most other months.

The dewpoint graph is the first of our bunch that does not follow a general pattern of change throughout a day. Seemingly more determined by month rather than hour of the day, the dewpoint temperature level is relatively constant for the month. Summer months have higher dewpoint temperatures while winter months have noticeably lower temperatures.

Windspeeds tend to peak between 3 - 5 PM and are higher during the Summer months. This is surprising to me. I am curious as to why this is the case, but I figure more data and research would be needed to figure out this correlation.

Stat Table Building Complete Function

Here, we will combine most of our past functions into a mega-function that generates all of the hourly and monthly table data and applies the necessary transformations.

It will output the monthly tables, confirm the correct structure of the hourly tables, and output the plots.

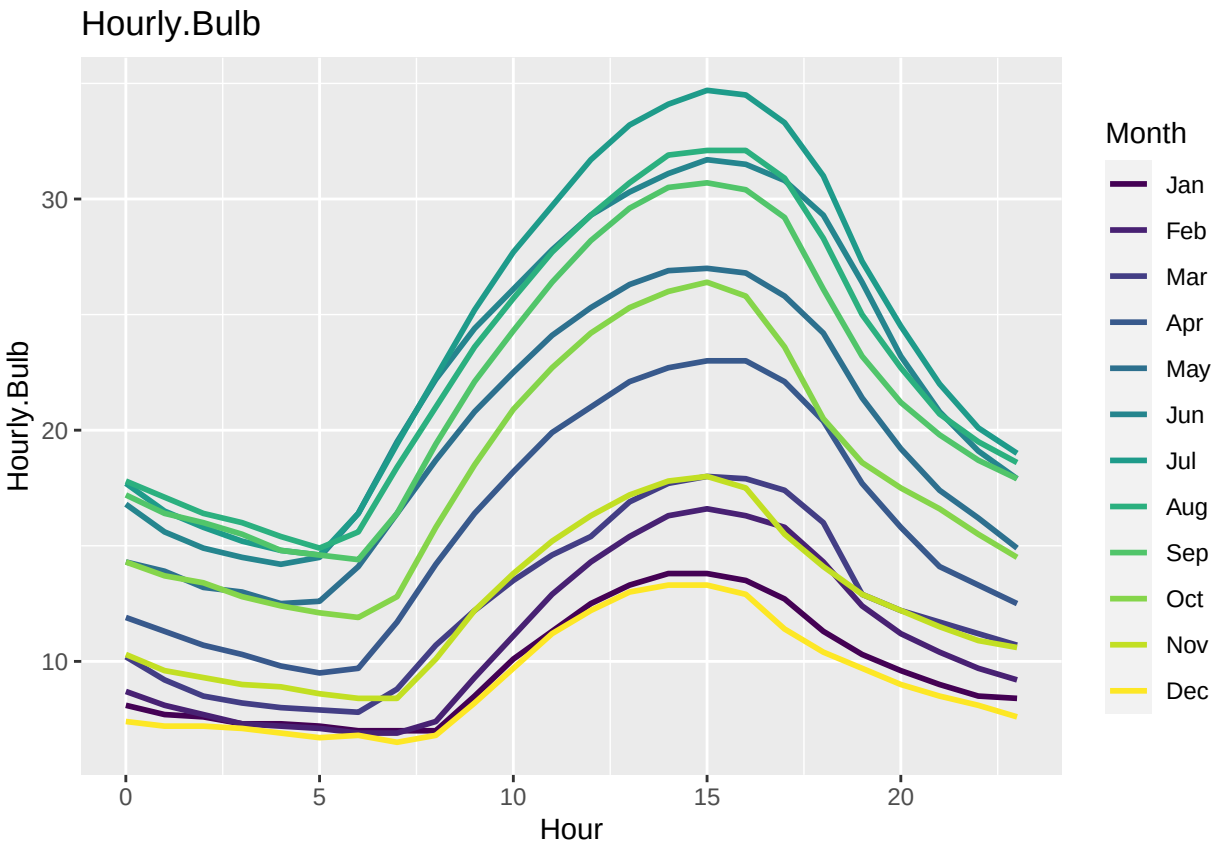
```
stat_transformation_process <- function(path) {

hourly_bulb <- stat_table(start_name = "Average Hourly Statistics for Dry Bulb temperatures", type = "h
hourly_dewpoint <- stat_table(start_name = "Average Hourly Statistics for Dew Point temperatures", type
hourly_humidity <- stat_table(start_name = "Average Hourly Relative Humidity", type = "hourly", path = p
hourly_solar <- stat_table(start_name = "Average Hourly Statistics for Direct Normal Solar Radiation",
hourly_windspeed <- stat_table(start_name = "Average Hourly Statistics for Wind Speed", type = "hourly"

monthly_wind <- stat_table(start_name = "Monthly Wind Direction", end_name = "NNW", colname = "Directio
monthly_drybulb <- stat_table(start_name = "Monthly Statistics for Dry Bulb temperatures", end_name = "
monthly_dewpoint <- stat_table(start_name = "Monthly Statistics for Dew Point temperatures", path = patl
monthly_windspeed <- stat_table(start_name = "Monthly Statistics for Wind Speed", path = path)


monthly_dewpoint <- monthly_transformations(monthly_dewpoint)
head(monthly_dewpoint)
monthly_drybulb <- monthly_transformations(monthly_drybulb)
head(monthly_drybulb)
monthly_windspeed <- monthly_transformations(monthly_windspeed)
head(monthly_windspeed)
monthly_wind <- monthly_transformations(monthly_wind)
head(monthly_wind)

hourly_bulb <- hourly_transformation(hourly_bulb)
hourly_windspeed <- hourly_transformation(hourly_windspeed)
hourly_dewpoint <- hourly_transformation(hourly_dewpoint)
hourly_solar <- hourly_transformation(hourly_solar)
hourly_humidity <- hourly_transformation(hourly_humidity)

data_validation(hourly_bulb)
data_validation(hourly_windspeed)
data_validation(hourly_dewpoint)
data_validation(hourly_solar)
data_validation(hourly_humidity)

combined_table <- join_tables(hourly_bulb, hourly_dewpoint, hourly_solar, hourly_windspeed, hourly_humi

create_plots(combined_table)
}
```

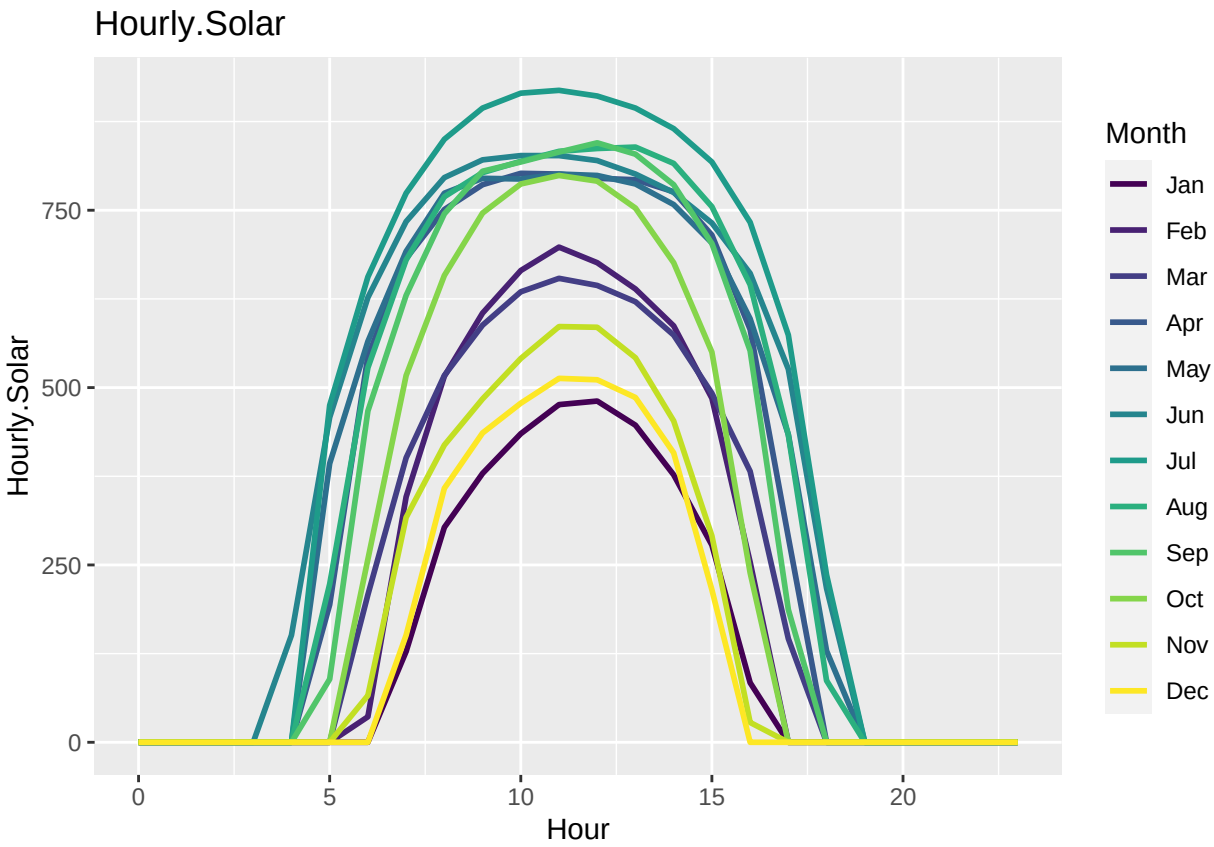
```
path_davis = "C:/Users/HP/Documents/141BProject1Files/USA_CA_UC-Davis-University.AP.720576_TMYx.2007-20
davis_stat <- stat_transformation_process(path_davis)
```

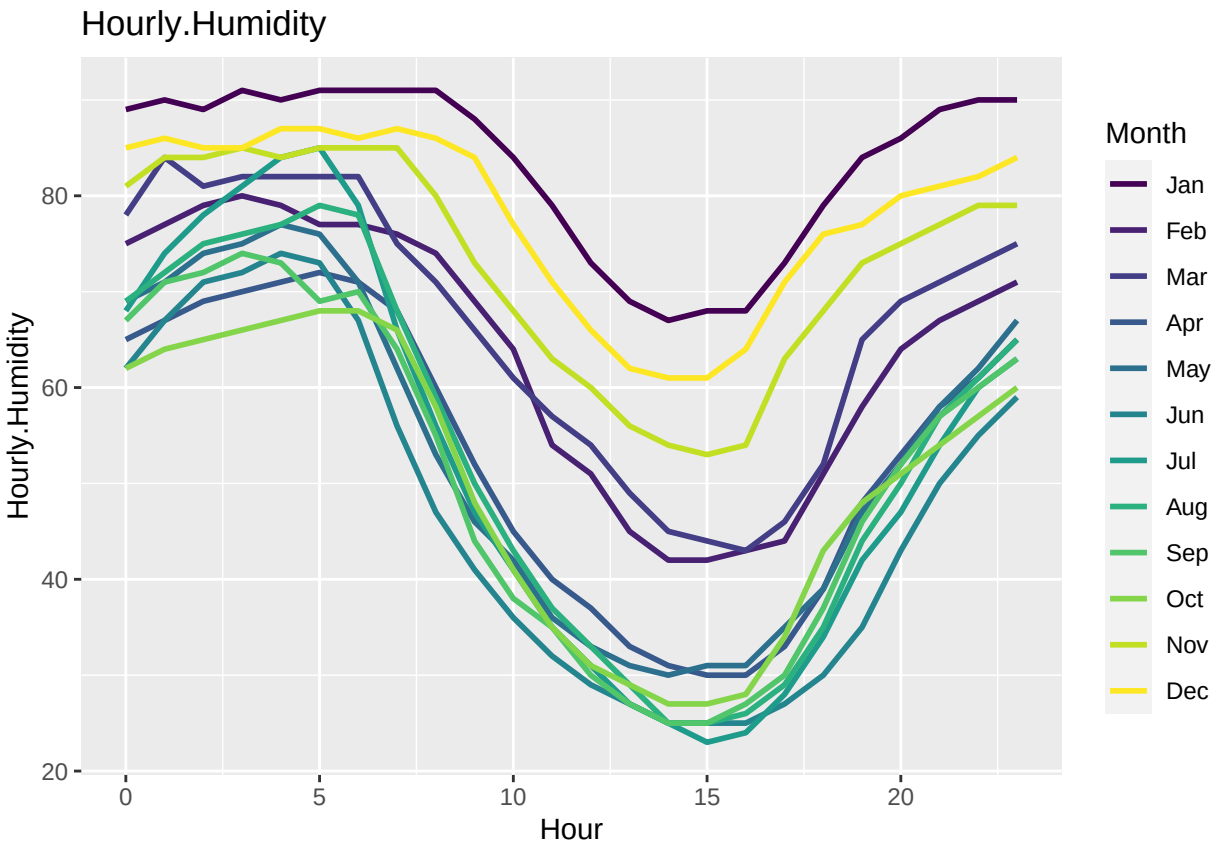
Davis Stat Tables

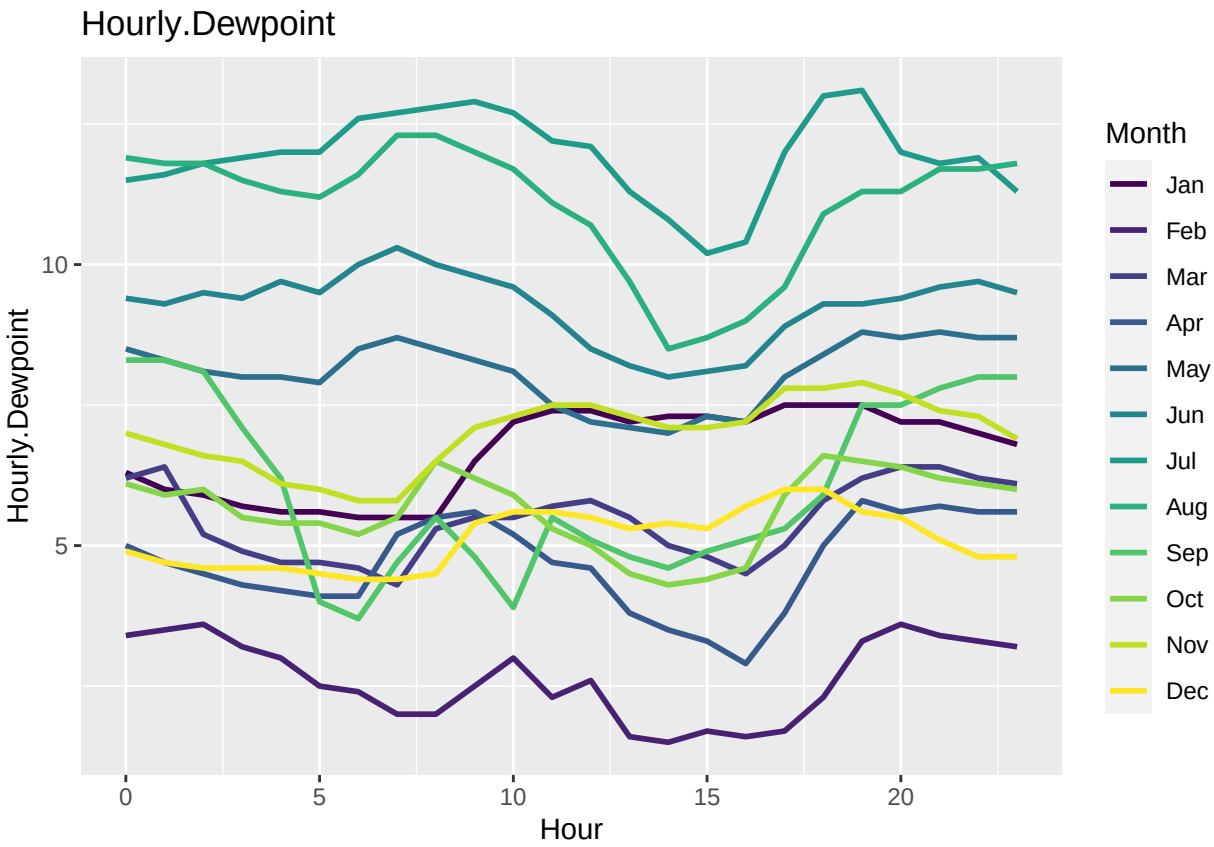
```
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
```

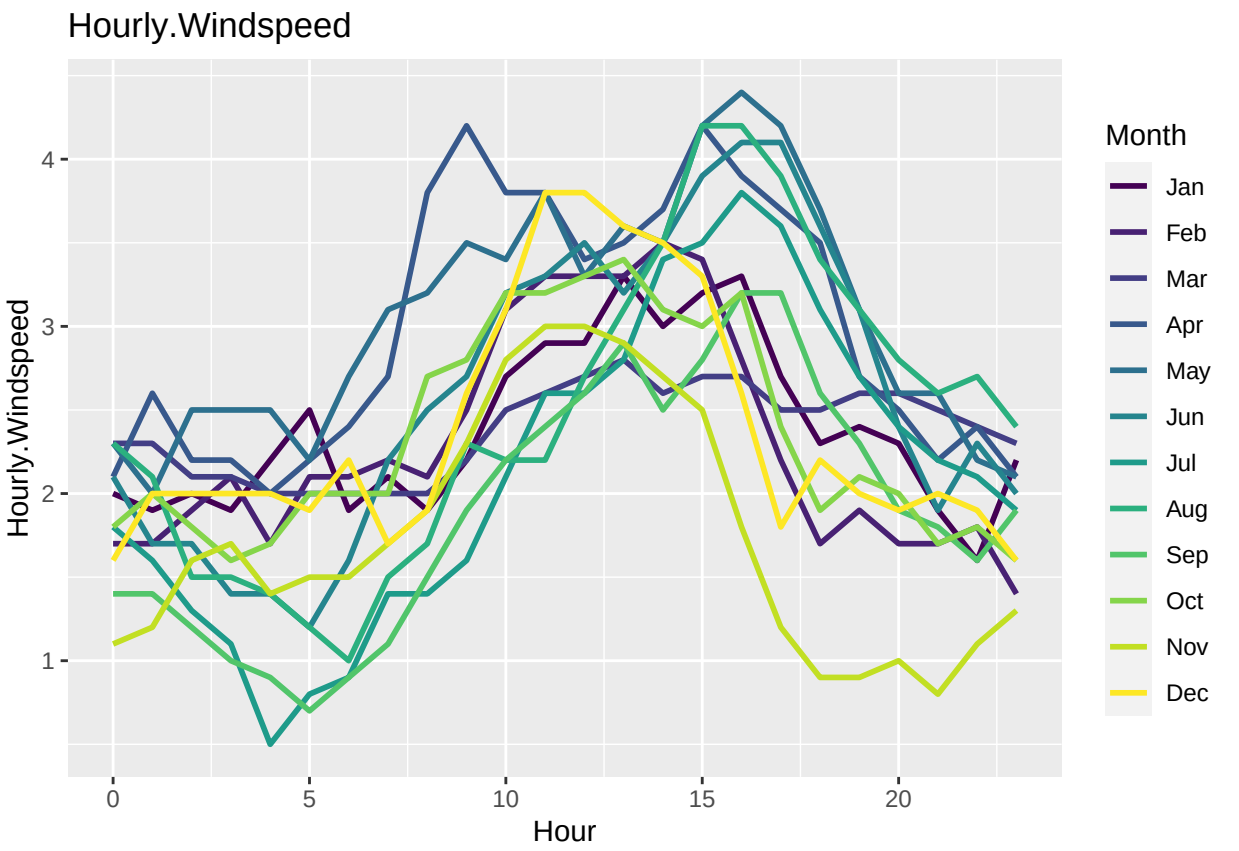
```
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 8.1 7.7 7.6 7.3 7.3 7.2 7 7 7 8.5 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 2 1.9 2 1.9 2.2 2.5 1.9 2.1 1.9 2.2 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 6.3 6 5.9 5.7 5.6 5.6 5.5 5.5 5.5 6.5 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 0 0 0 0 0 0 0 128 303 379 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 89 90 89 91 90 91 91 91 91 88 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
```


Hourly.Bulb
Hourly.Bulb
30
20
10
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Solar
Hourly.Solar
750
500
250
0
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Humidity
Hourly.Humidity
80
60
40
20
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Dewpoint
Hourly.Dewpoint
10
5
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec



Let's look at some general trends from Davis.

Firstly, looking at the Drybulb temperatures, they follow a very similar curve regardless of the time of year. While initially starting low, it steadily rises from 8 AM before peaking at 3 PM, and slowly decreasing as the sun sets into the night. Perhaps unsurprisingly, the highest temperatures occur in the summer months from May - August.

The Solar graph follows a parabolic curve. Since there is no sun out very early in the morning or late at night, it peaks at around 11am-Noon. Due to a larger amount of clouds in the Winter months, there is even less solar energy than normal, even at peak daytime hours. I'm not surprised by this given how cloudy and rainy it gets here in the winter time.

The humidity graph shows us that it is lowest mid-day (3 PM) and higher during the nighttime. Humidity is also highest during the winter months, with January having a much higher amount than most other months. Truthfully, it never feels all that humid in Davis.

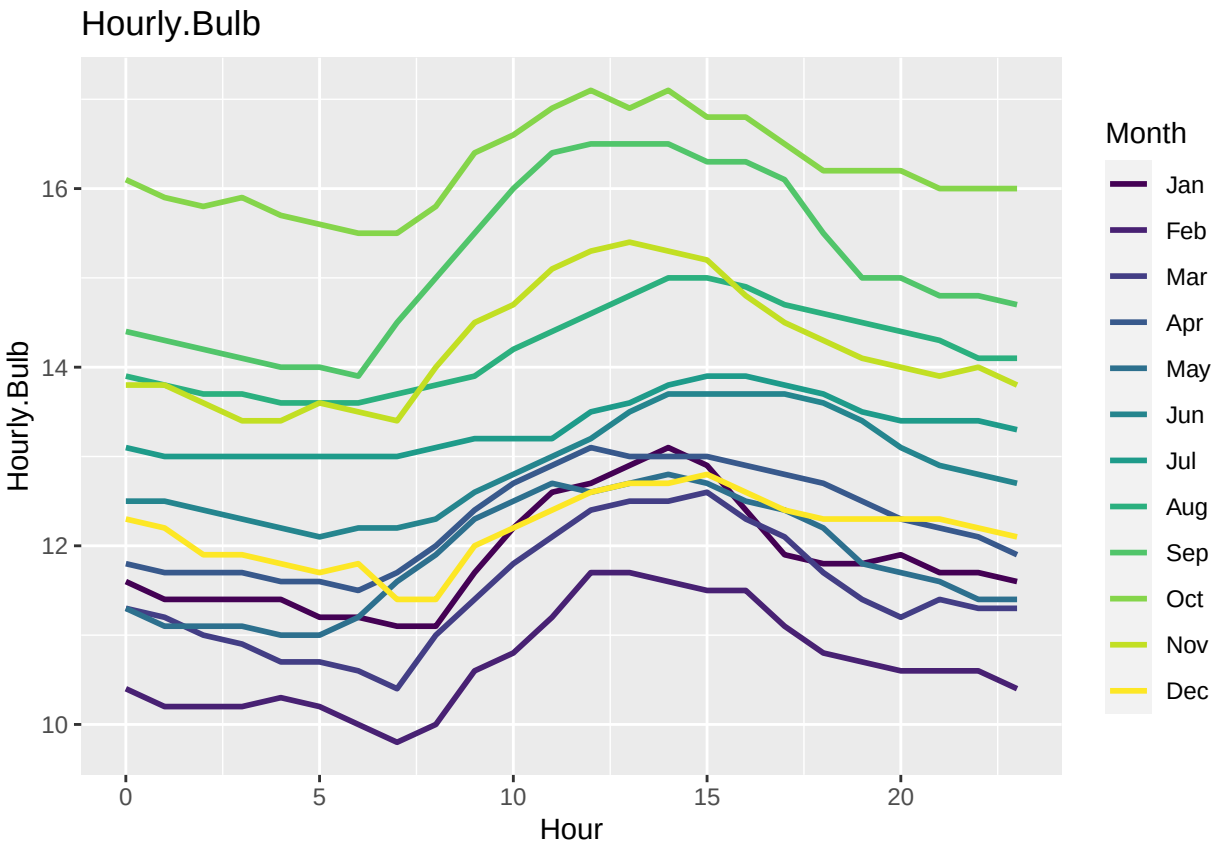
The dewpoint graph is the first of our bunch that does not follow a general pattern of change throughout a day. Seemingly more determined by month rather than hour of the day, the dewpoint temperature level is relatively constant for the month. Summer months have higher dewpoint temperatures while winter months have noticeably lower temperatures.

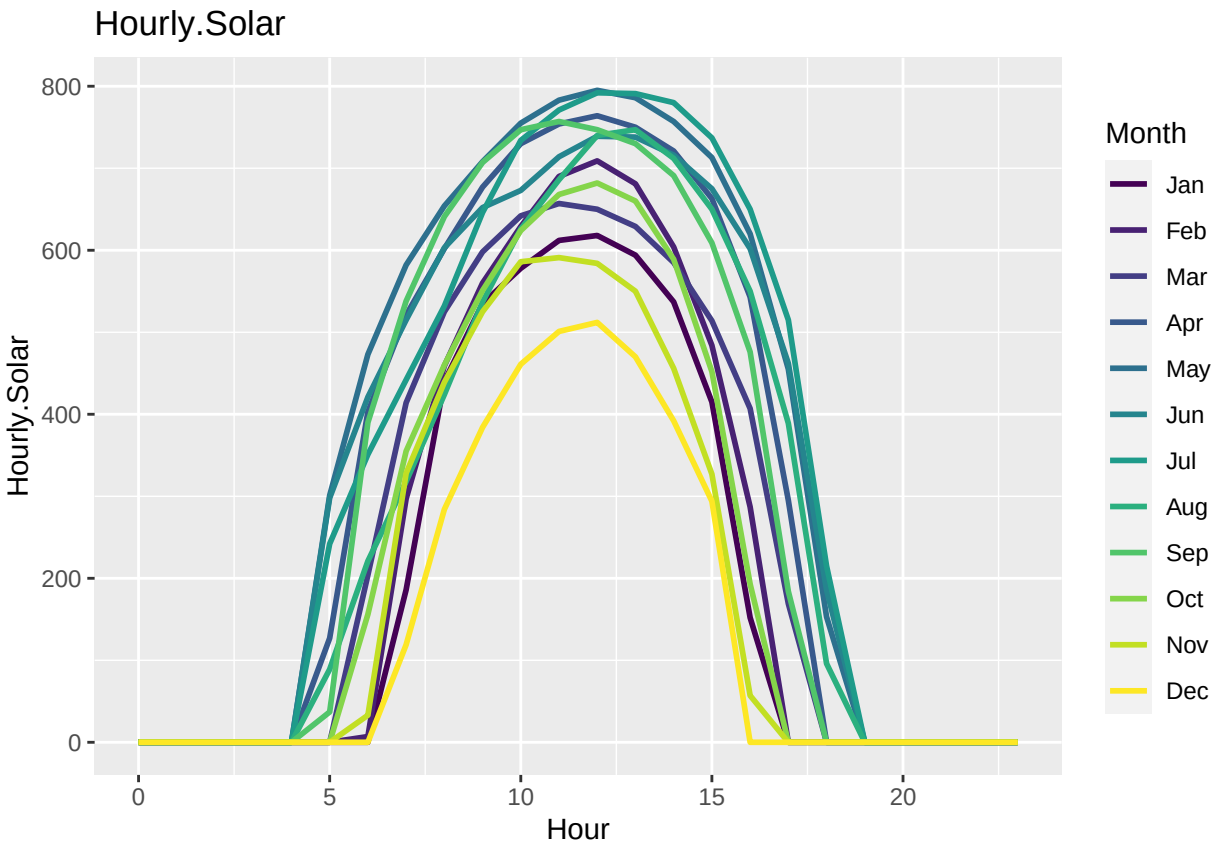
Windspeeds are very erratic in Davis, often peaking at different times during the day depending on the month. There is generally a pretty high windspeed level across all times and months.

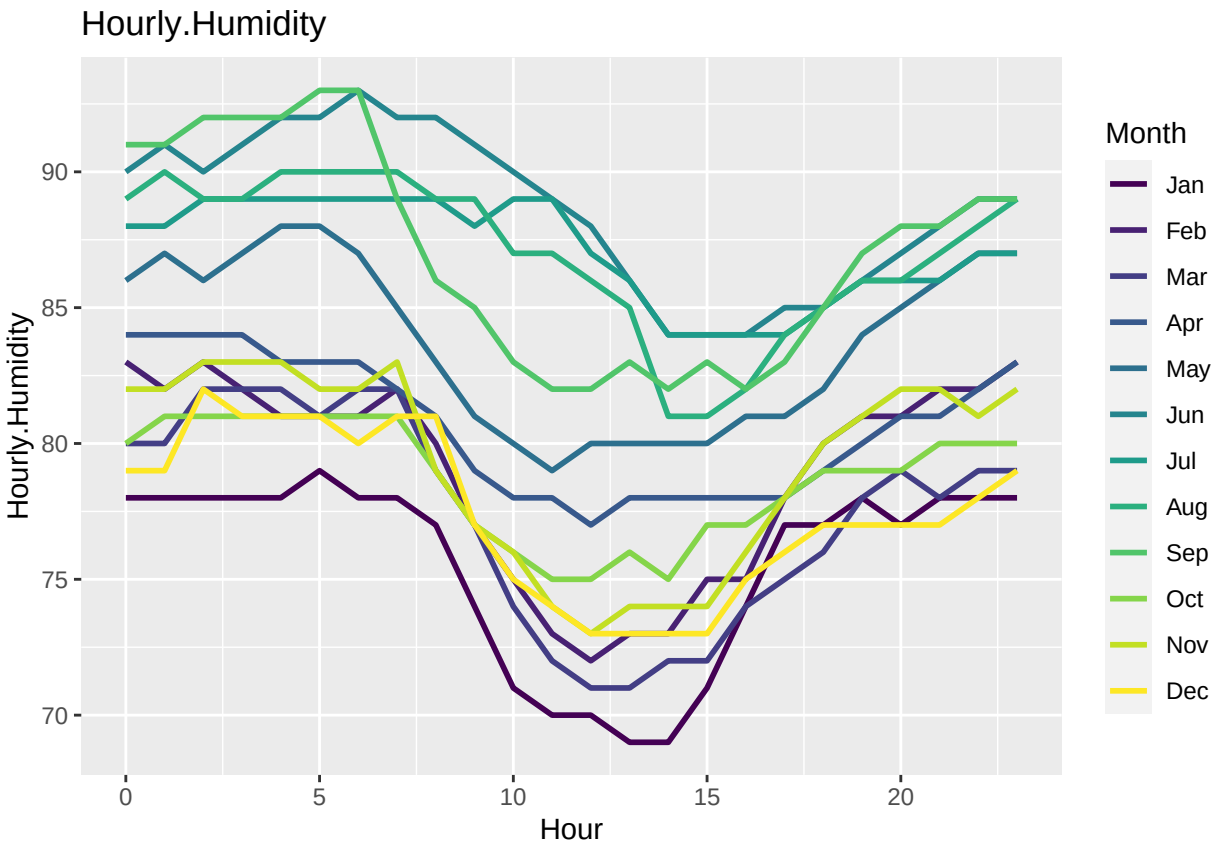
```
path_point_reyes = "C:/Users/HP/Documents/141BProject1Files/USA_CA_Point.Reyes.Lighthouse.724959_TMYx.2
point_reyes_stat <- stat_transformation_process(path_point_reyes)
```

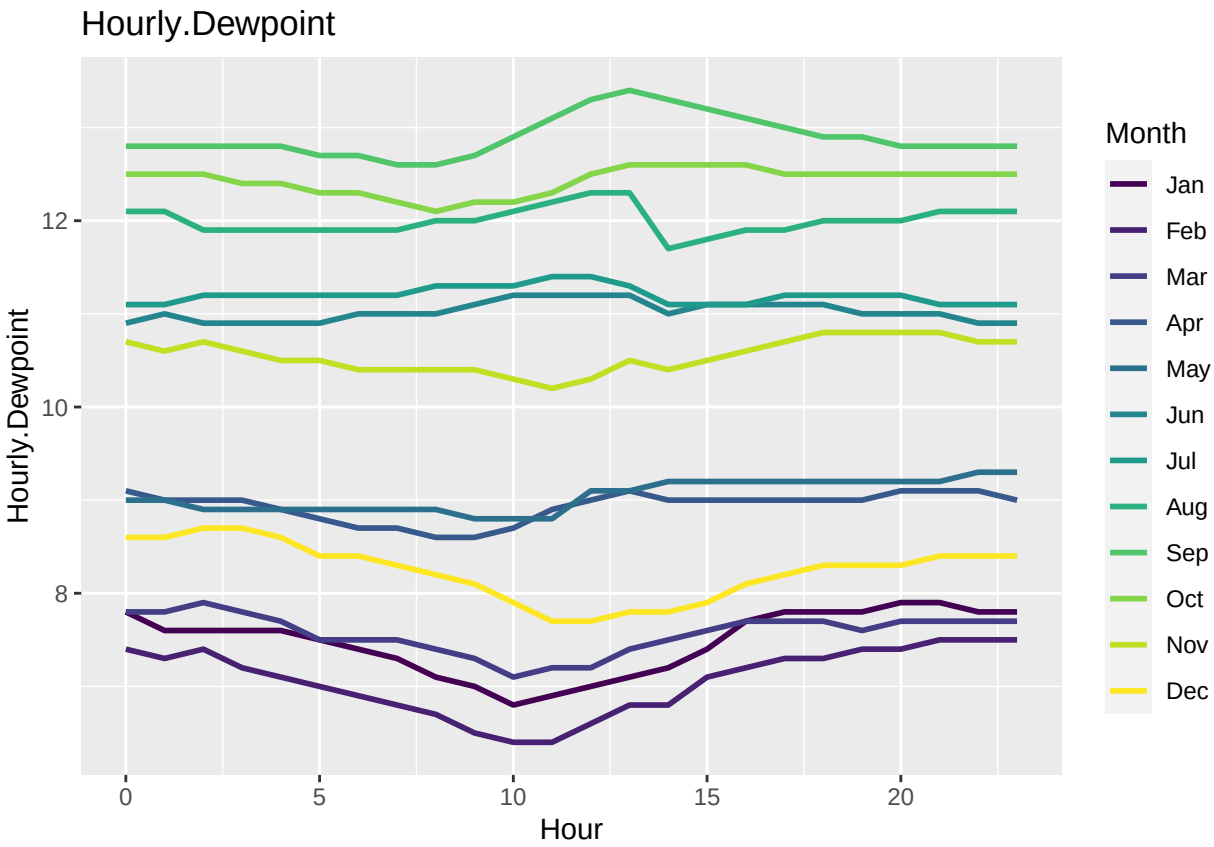
Point Reyes Stat Tables

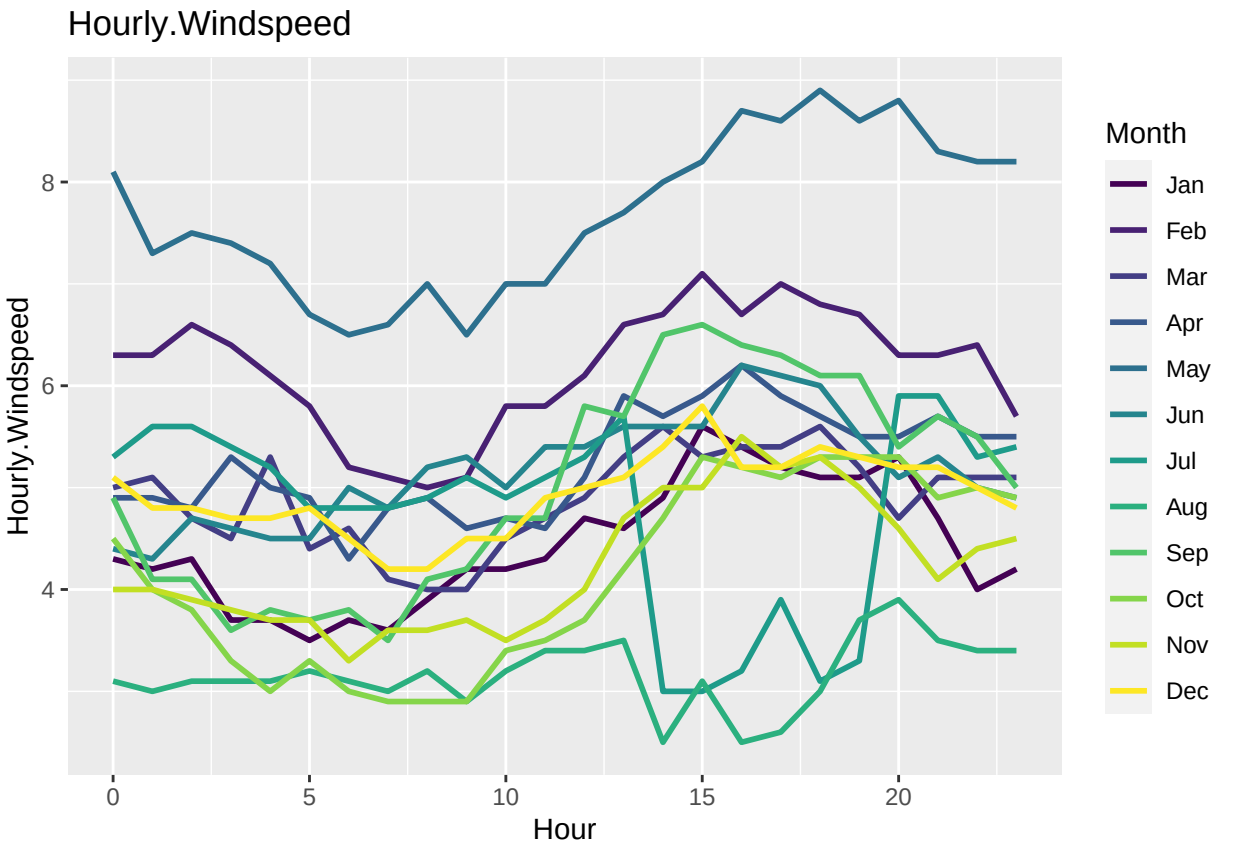
```
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 11.6 11.4 11.4 11.4 11.4 11.2 11.2 11.1 11.1 11.7 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 4.3 4.2 4.3 3.7 3.7 3.5 3.7 3.6 3.9 4.2 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 7.8 7.6 7.6 7.6 7.6 7.5 7.4 7.3 7.1 7 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 0 0 0 0 0 0 0 186 440 535 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 78 78 78 78 78 79 78 78 77 74 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
```


Hourly.Bulb
Hourly.Bulb
16
14
12
10
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Solar
800
600
400
200
0
Hourly.Solar
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Humidity
Hourly.Humidity
90
85
80
75
70
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Dewpoint
Hourly.Dewpoint
12
10
8
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec



Noticeably, the temperatures are much less in Point Reyes than in Davis. While Davis peaks at 30, Point Reyes bulb temperatures only get up to around 16.

Surprisingly, it's solar values do peak higher than Davis' values, though.

It's humidity values display much less consistency than Davis or SF-Fairfield. Rather than a relatively uniform curve, there are many more peaks and troughs throughout the plots. Its humidity is also much higher in the Summer than the Winter, which is inverse to Davis and SF-Fairfield. It also exhibits higher peaks overall.

Dewpoint values are much more consistent. They tend to follow very similar grouping bands and values stay consistent throughout the day. Summer reigns supreme over Winter here as well, which is the trend across all three graphs.

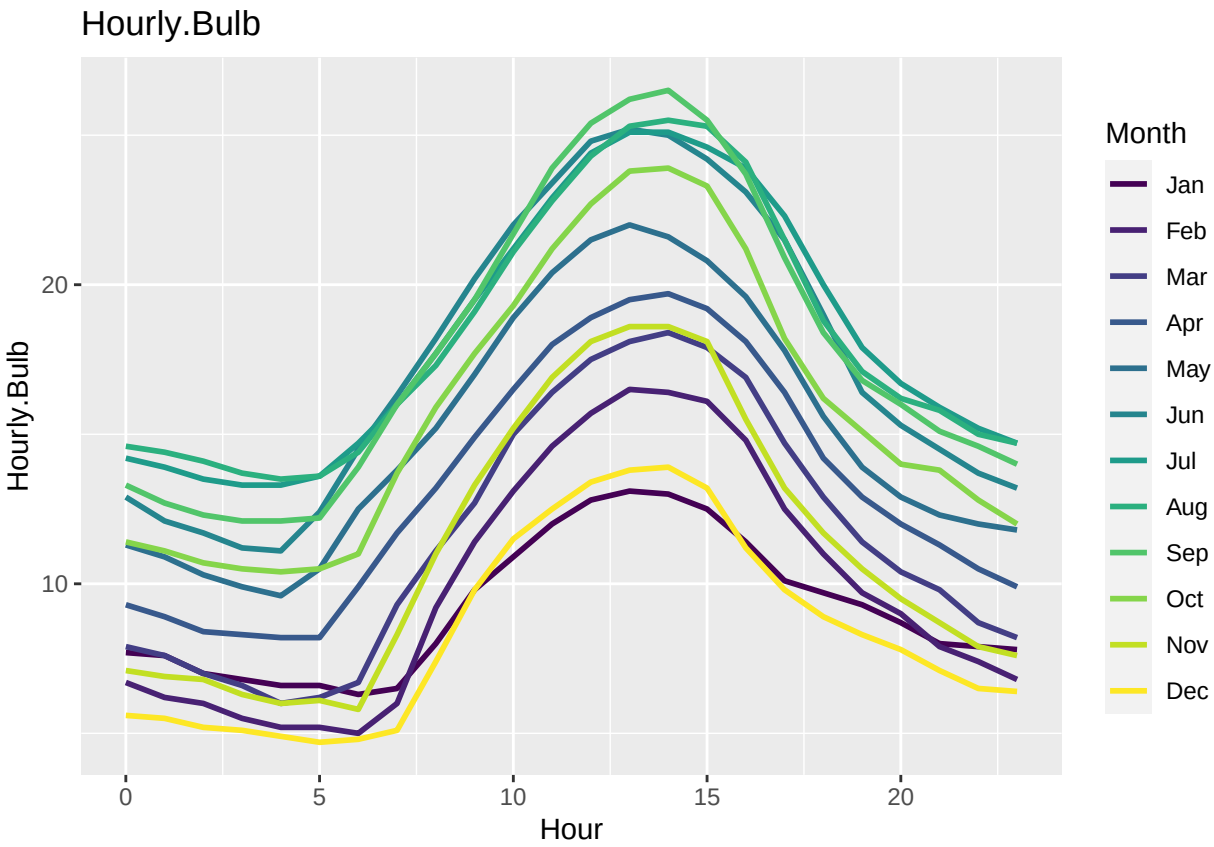
Lastly, Point Reyes' windspeed graph is much more erratic than Davis's, without much of a general pattern. In addition, it reaches up to 8 mph in June, nearly double the maximum of Davis.

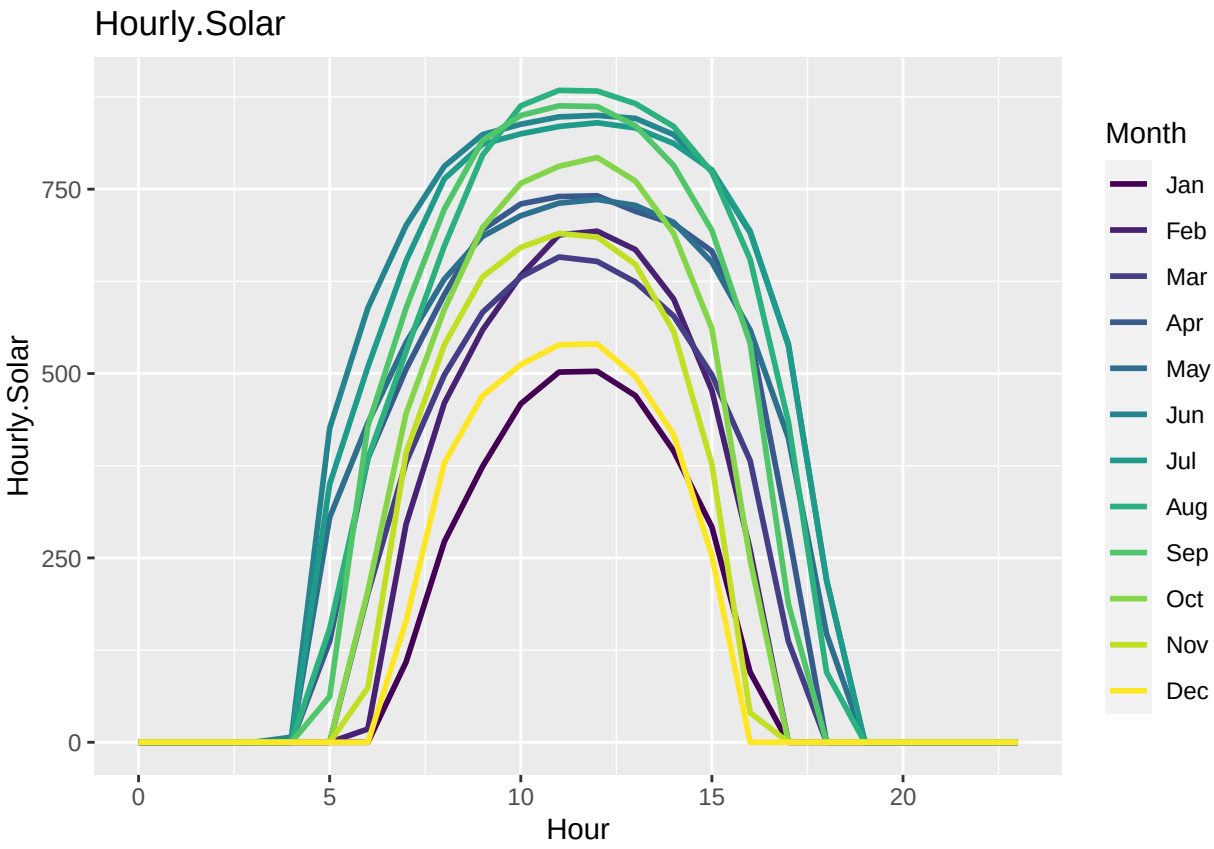
```
path_napa = "C:/Users/HP/Documents/141BProject1Files/USA_CA_Napa.County.AP.724955_TMYx.2007-2021.stat"
napa_stat <- stat_transformation_process(path_napa)
```

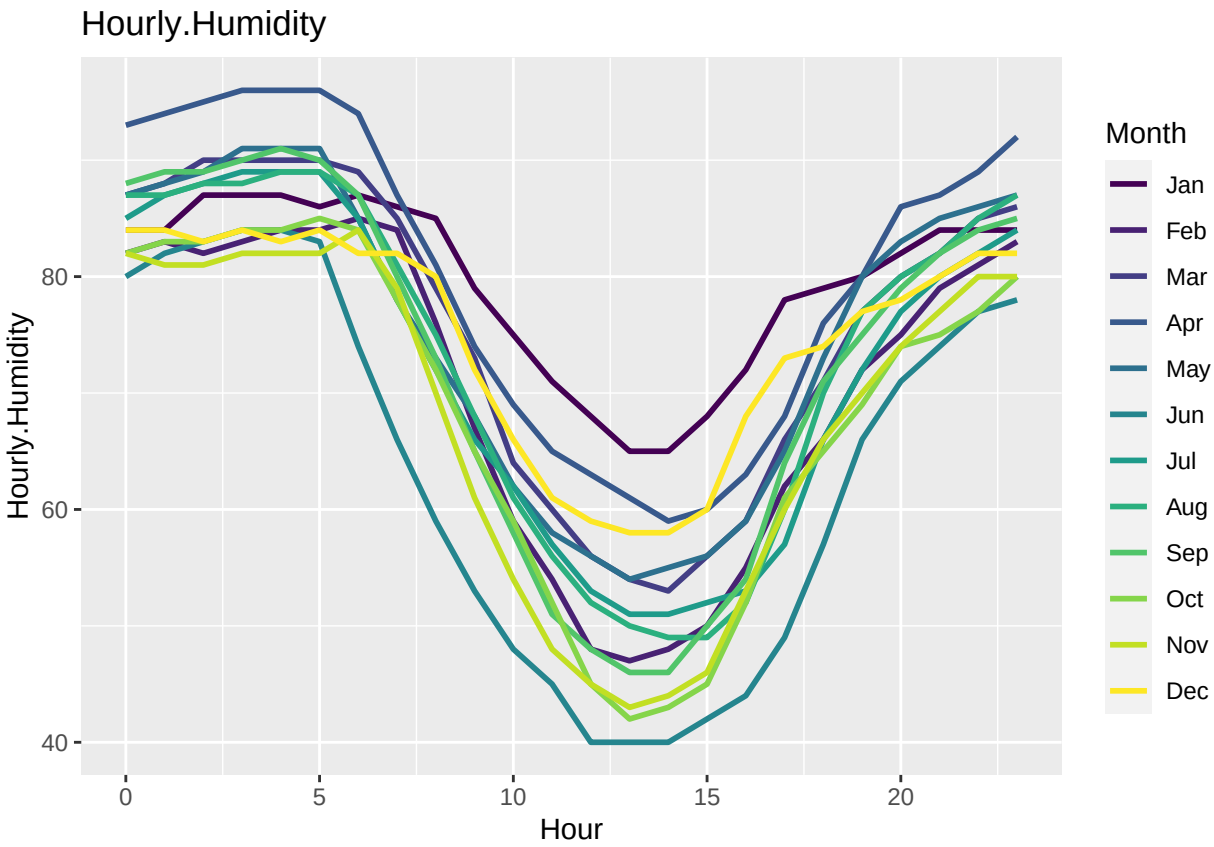
Napa Stat Tables

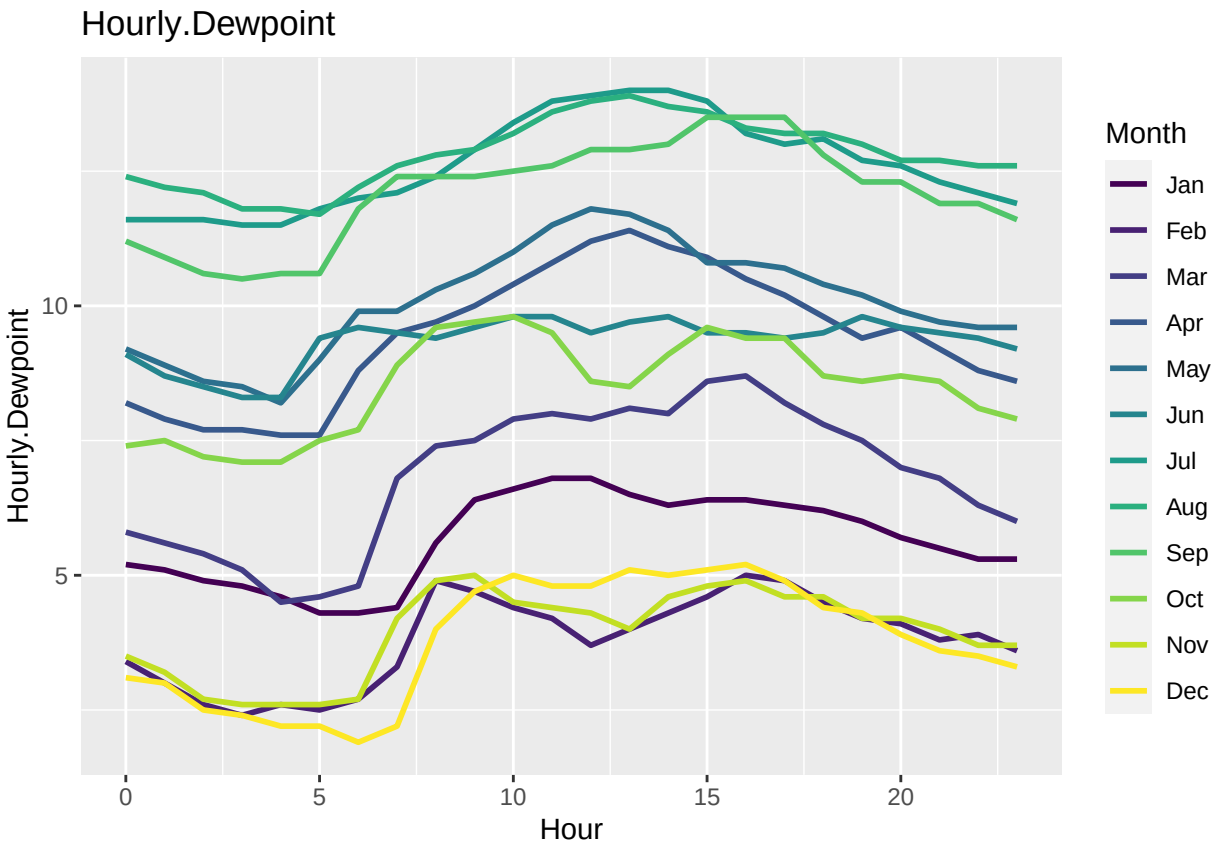
```
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
```

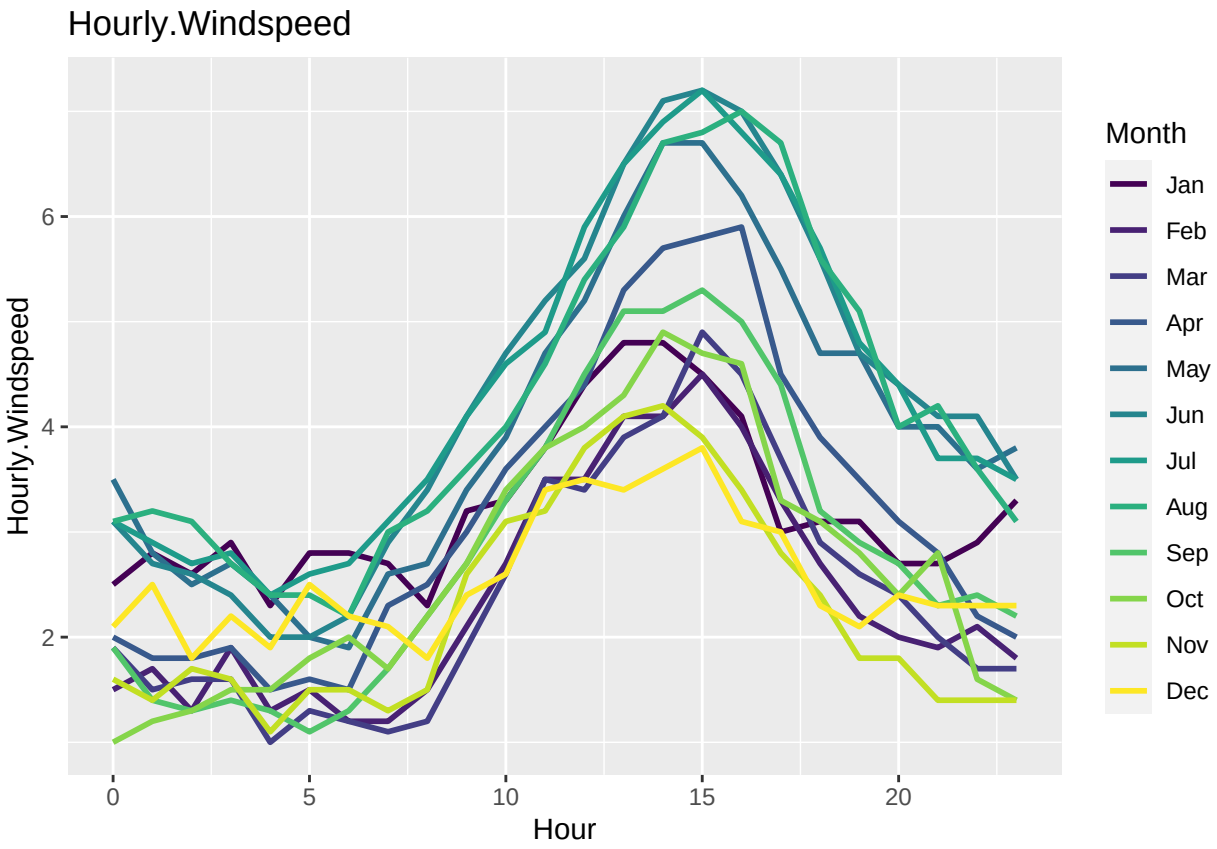
```
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 7.7 7.6 7 6.8 6.6 6.6 6.3 6.5 8 9.8 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 2.5 2.8 2.6 2.9 2.3 2.8 2.8 2.7 2.3 3.2 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 5.2 5.1 4.9 4.8 4.6 4.3 4.3 4.4 5.6 6.4 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 0 0 0 0 0 0 0 109 272 374 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 84 84 87 87 87 86 87 86 85 79 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
```


Hourly.Bulb
Hourly.Bulb
20
10
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Solar
Hourly.Solar
750
500
250
0
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Humidity
Hourly.Humidity
80
60
40
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Dewpoint
Hourly.Dewpoint
10
5
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec



Let's look at the Napa charts.

For the bulb, solar, and humidity graphs, it displays uniform curves, similar to Davis and Fairfield-SF.

Its dewpoint values display the consistent bands similar to Point Reyes, but more up-and-down movement like the first two data groups.

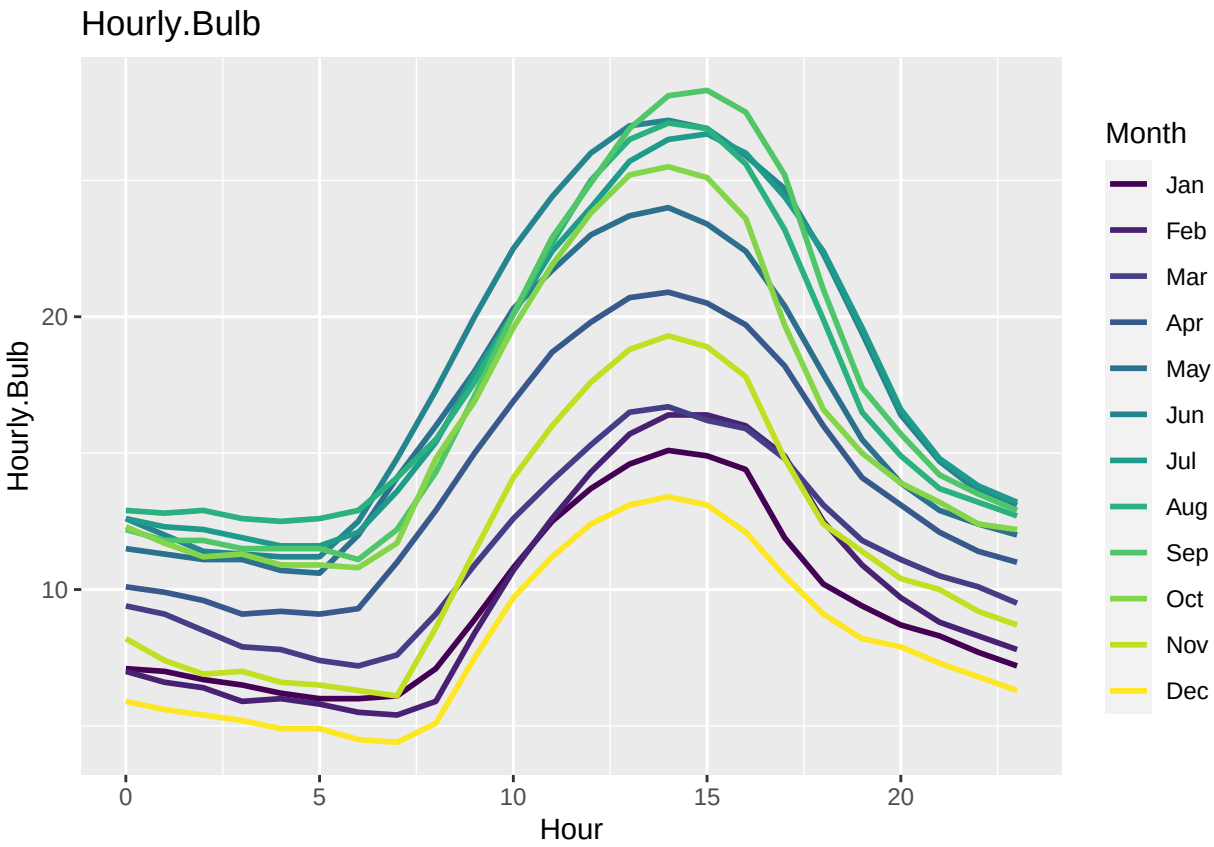
Its windspeed graph follows a very concise curve throughout all seasons, peaking at 3 PM each day. Napa also has very high windspeeds, reaching to 7 MPH in the summer months.

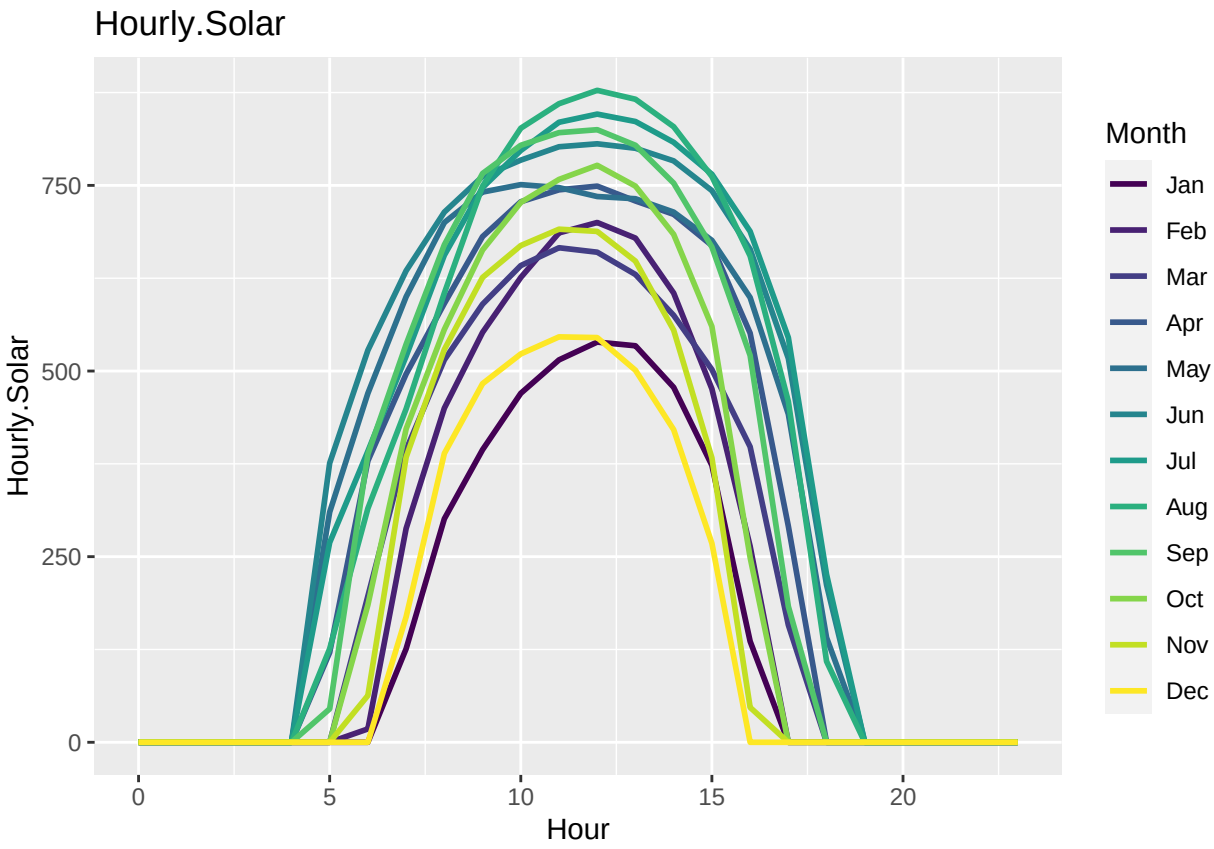
```
path_marin = "C:/Users/HP/Documents/141BProject1Files/USA_CA_Marin.County.AP-Gnoss.Field.720406_TMYx.20

marin_stat <- stat_transformation_process(path_marin)
```

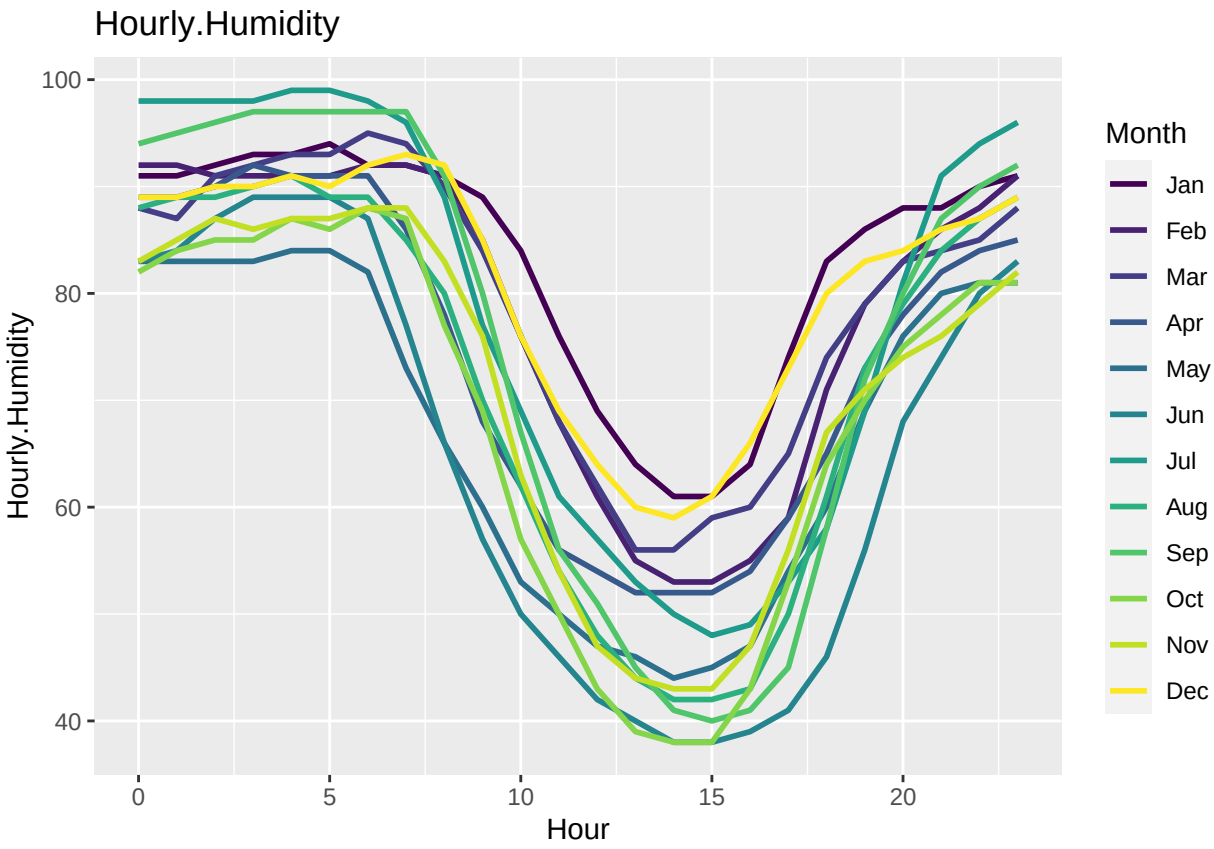
Marin Stat Tables

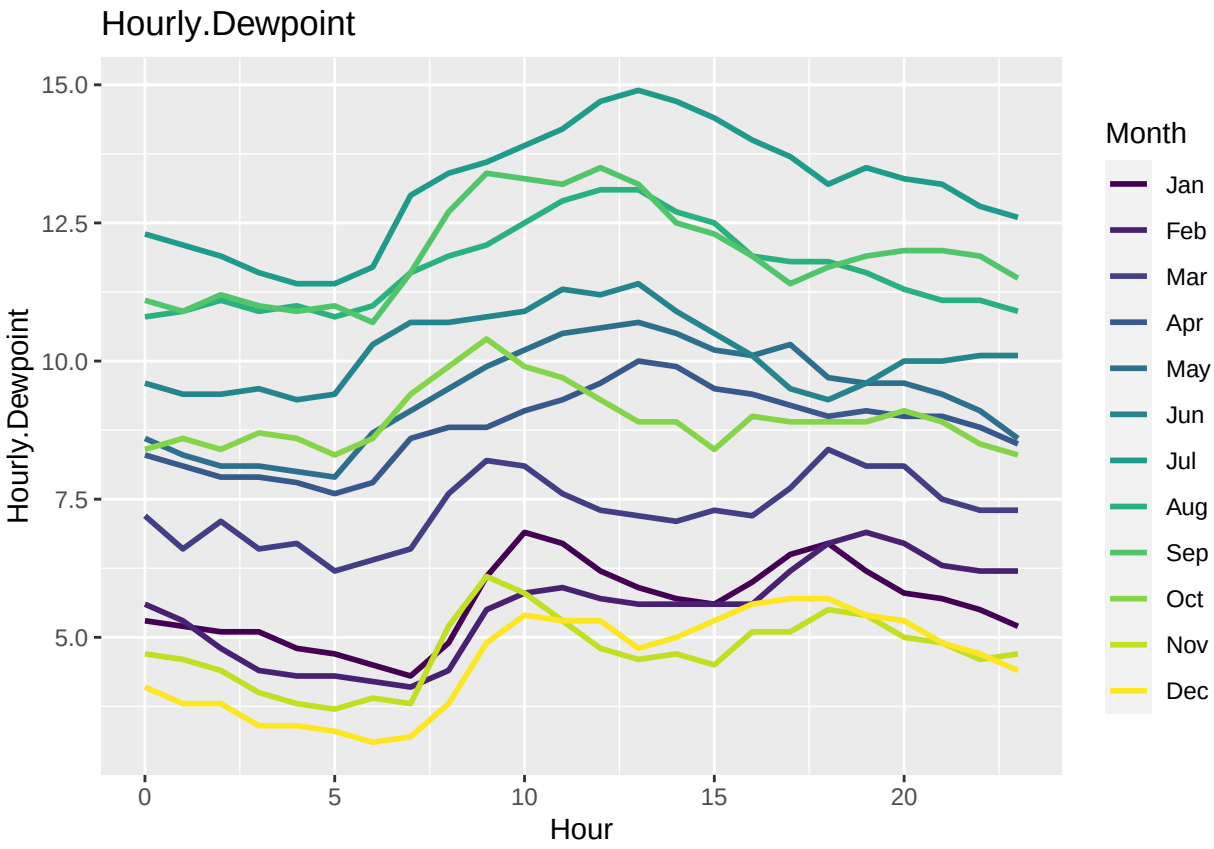
```
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
## [1] "All max hour values match"
## [1] "All min hour values match"
```

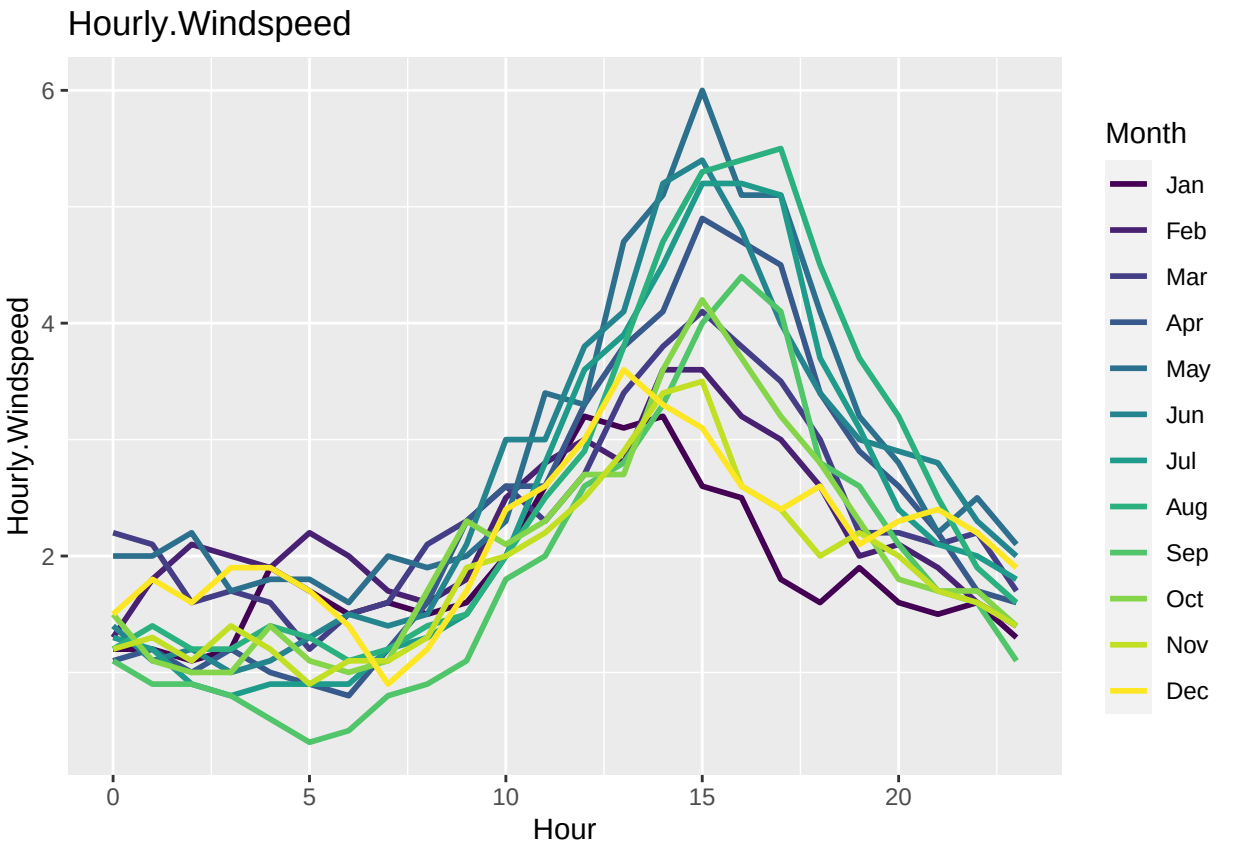
```
## [1] "All max hour values match"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 7.1 7 6.7 6.5 6.2 6 6 6.1 7.1 8.9 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 1.2 1.2 1.1 1.2 1.9 1.7 1.5 1.6 1.5 1.6 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 5.3 5.2 5.1 5.1 4.8 4.7 4.5 4.3 4.9 6.1 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 0 0 0 0 0 0 0 126 301 394 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 288   3
## tibble [288 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Hour : int [1:288] 0 1 2 3 4 5 6 7 8 9 ...
##  $ Month: Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Value: num [1:288] 91 91 92 93 93 94 92 92 91 89 ...
## NULL
## [1] "tbl_df"     "tbl"        "data.frame"
```


Hourly.Bulb
Hourly.Bulb
20
10
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Solar
Hourly.Solar
750
500
250
0
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Humidity
100
80
60
40
Hourly.Humidity
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Hourly.Dewpoint
15.0
12.5
10.0
7.5
5.0
Hourly.Dewpoint
0
5
10
15
20
Hour
Month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec



Marin's bulb graph is almost identical to Napa's. It peaks at around the same time and exhibits higher temperatures in the summer (as to be expected).

The solar graph has a very similar parabolic curve to all other datasets.

Humidity also has a very similar structure to all except Point Reyes. Noticeably, it does peak very high during the summer months, clocking in at around 100. Once the sun rises, it declines greatly in the summer. At 3 PM, the inversion point of the graph, the winter graphs are the highest.

Dewpoint also follows the trend of the other graphs, with summer months boasting a noticeably higher temperature than the winter months.

Windspeed has a very uniform curve, peaking around 3 PM. The summer months reach up to 6 MPH, which places it around the middle of the overall distribution.

Conclusion

After reading in many tables, validating, transforming, and plotting the data, we are able to get some basic insights about the general topography and climate of five northern california regions.

While many of them follow similar trends and curves across days and months, there are some very interesting differences and quirks for each region. For example, inverse to the other regions, Point Reyes' has the highest humidity levels during the summer.

This was a great exercise in order to familiarize oneself with the basics of R and how to take command of data, no matter how archaic the format is.