NCEI GHCN Processing Example

This R Notebook demonstrates reading a pre-processed CF-Metadata-compliant netCDF file into R for basic processing.

```
library("ncdf4")
library("ncdf.tools")
library("lubridate")
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library("units")
```

```
##
## Attaching package: 'units'
```

```
## The following object is masked from 'package:base':
##
##     %*%
```

```
library("ncdf4.helpers")
library("beanplot")
```

File Access

File Location: Users on Windows machines may need to download these files as OpenDAP is not included with R's -easyto-install NetCDF package

Technical Note Found here

According to one developer, this works but I have not tried it.

If this doesn't work you will need to download the netcdf file(s) from our university THREDDS Server

```
# NetCDF File Access

# replace this with your local storage directory if you need to download the data serparately.

netcdf_file_URL_or_directory = "http://kyrill.ias.sdsmt.edu:8080/thredds/dodsC/GHCN_POINT_DATA/"

netcdf_file = "GHCND-USW00014944__SIOUX_FALLS_FOSS_FIELD_SD.nc"
netcdf_file = "GHCND-USW00024090__RAPID_CITY_REGIONAL_AIRPORT_SD.nc"


netcdf_source_file = paste(netcdf_file_URL_or_directory,
                           netcdf_file,
                           sep = "")

# crack open the NetCDF File
ncf  = nc_open(filename = netcdf_source_file)
```

```
# tidy as we go...

remove(netcdf_file_URL_or_directory,
       netcdf_file)


# and let's view the inventory
```

Date-Time Management

The hardest part is to get the time coordinate data...

Here we use one of the ncdf4.helper functions, nc.get.time.series.

```
# Read the time coordinates netadata

netcdf_calendar = ncatt_get(nc      = ncf,
                            varid   = "time",
                            attname = "calendar")


netcdf_timeunits = ncatt_get(nc      = ncf,
                             varid   = "time",
                             attname = "units")


time = nc.get.time.series(f = ncf) # creates a specialized date coordiante

origin_time = netcdf_timeunits$value[1]
origin_time = unlist(strsplit(x     = netcdf_timeunits$value,
                              split = " "))

origin_time = paste(origin_time[3],
                    origin_time[4],
                    sep = " ")

time = as.POSIXct(x      = time,
                  origin = origin_time,
                  tz     = "UTC")


remove(netcdf_timeunits)
remove(netcdf_calendar)
remove(origin_time)
```

Let's also get the station name

```
station_name = ncatt_get(nc      = ncf,
                         varid   = 0,   # 0  is used for global attrubutes
                         attname = "Station_Name")

station_name = station_name$value
```

Importing and Organizing Our Data

Now we read in the basic fields.

We first crete a data frame and then populate it.

We begin by adding some climatology aggregators.

```
# creating climatology data frame

data = data.frame(time     = time,
                  row.name = time)

data$month  = month(x = time,    # R time value
                    label=TRUE)  # use text values

data$year   = year( x = time)   # R time value

data$decade = paste((data$year  %/% 10) * 10,
                    "'s",
                    sep = "")

data$season3mo = "EMPTY"

  monthdigit = month(x = time)

  data$season3mo[(monthdigit == 01)] =  "DJF"
  data$season3mo[(monthdigit == 02)] =  "DJF"
  data$season3mo[(monthdigit == 03)] =  "MAM"
  data$season3mo[(monthdigit == 04)] =  "MAM"
  data$season3mo[(monthdigit == 05)] =  "MAM"
  data$season3mo[(monthdigit == 06)] =  "JJA"
  data$season3mo[(monthdigit == 07)] =  "JJA"
  data$season3mo[(monthdigit == 08)] =  "JJA"
  data$season3mo[(monthdigit == 09)] =  "SON"
  data$season3mo[(monthdigit == 10)] =  "SON"
  data$season3mo[(monthdigit == 11)] =  "SON"
  data$season3mo[(monthdigit == 12)] =  "DJF"

data$season6mo = "EMPTY"

  data$season6mo[(monthdigit == 01)] =  "ONDJFM"
  data$season6mo[(monthdigit == 02)] =  "ONDJFM"
  data$season6mo[(monthdigit == 03)] =  "ONDJFM"
  data$season6mo[(monthdigit == 04)] =  "AMJAJS"
  data$season6mo[(monthdigit == 05)] =  "AMJAJS"
  data$season6mo[(monthdigit == 06)] =  "AMJAJS"
  data$season6mo[(monthdigit == 07)] =  "AMJAJS"
  data$season6mo[(monthdigit == 08)] =  "AMJAJS"
  data$season6mo[(monthdigit == 09)] =  "AMJAJS"
  data$season6mo[(monthdigit == 10)] =  "ONDJFM"
  data$season6mo[(monthdigit == 11)] =  "ONDJFM"
  data$season6mo[(monthdigit == 12)] =  "ONDJFM"

remove(monthdigit)
```

Then we import our GHCN data

Here we are going to do it all automatically.

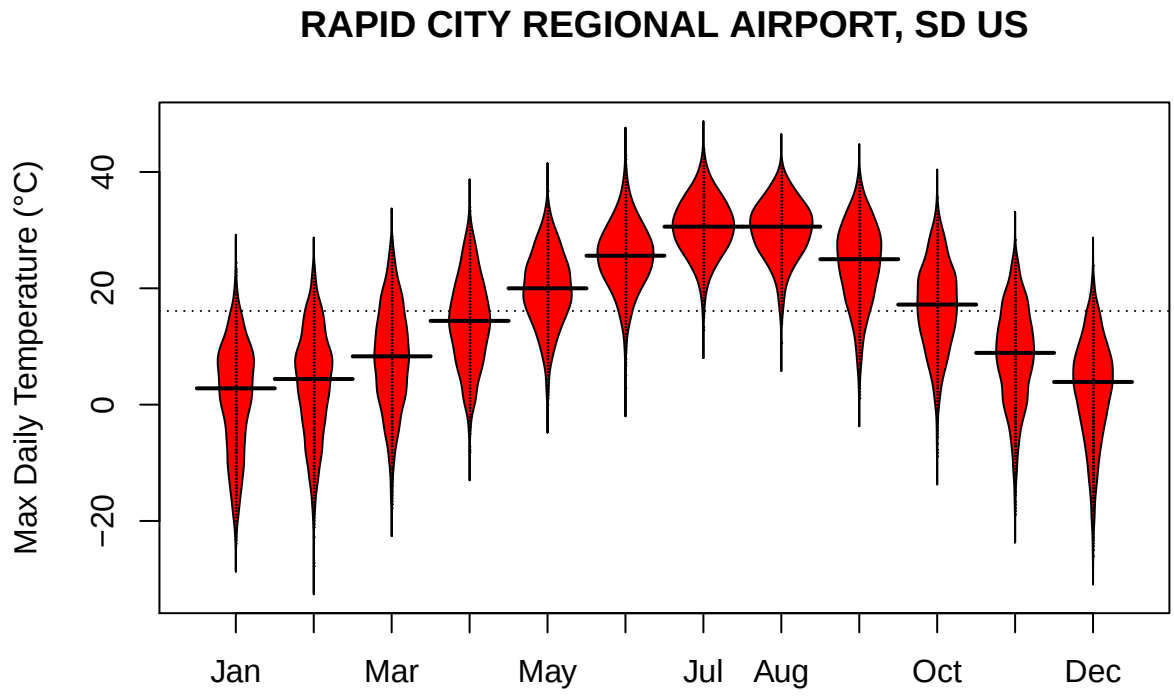
```
variable_list = nc.get.variable.list(f = ncf)

for (variable_name in variable_list) {
  temp_var = ncvar_get(nc    = ncf,
                       varid = variable_name)
  print(variable_name)
  data = cbind(data, temp_var)
  ncol(x = data)
  colnames(data)[ncol(x = data)] <- variable_name
  remove(temp_var)
}
```

```
## [1] "maximum_air_temperature"
## [1] "minimum_air_temperature"
## [1] "mean_air_temperature"
## [1] "precipitation_amount"
## [1] "thickness_of_snowfall_amount"
## [1] "surface_snow_thickness"
## [1] "liquid_water_content_of_surface_snow"
## [1] "mean_wind_speed"
```

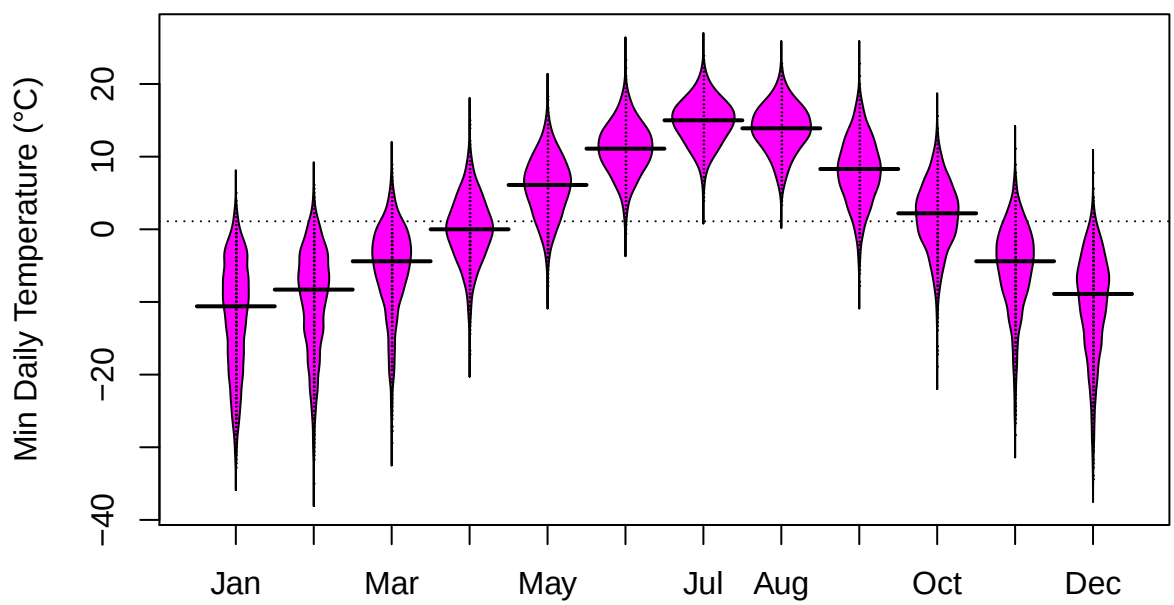
```
# SPECIAL QC NOTE
#  THERE IS A MISARCHIVE OF LIQUID SWE.  VALUES OF 1008.1 should be set to zero


data$liquid_water_content_of_surface_snow[data$liquid_water_content_of_surface_snow > 990] = NA

remove(variable_name)
```

Processing by Month

```
beanplot(data$maximum_air_temperature~data$month, # variable
         ll          = 0,                          # no crosticks per point
         overallline = "median",
         beanlines   = "median",
         main        = station_name,                    # main title
         ylab        = "Max Daily Temperature (°C)",    # y axis label
         col         = "red")                           # colour
```



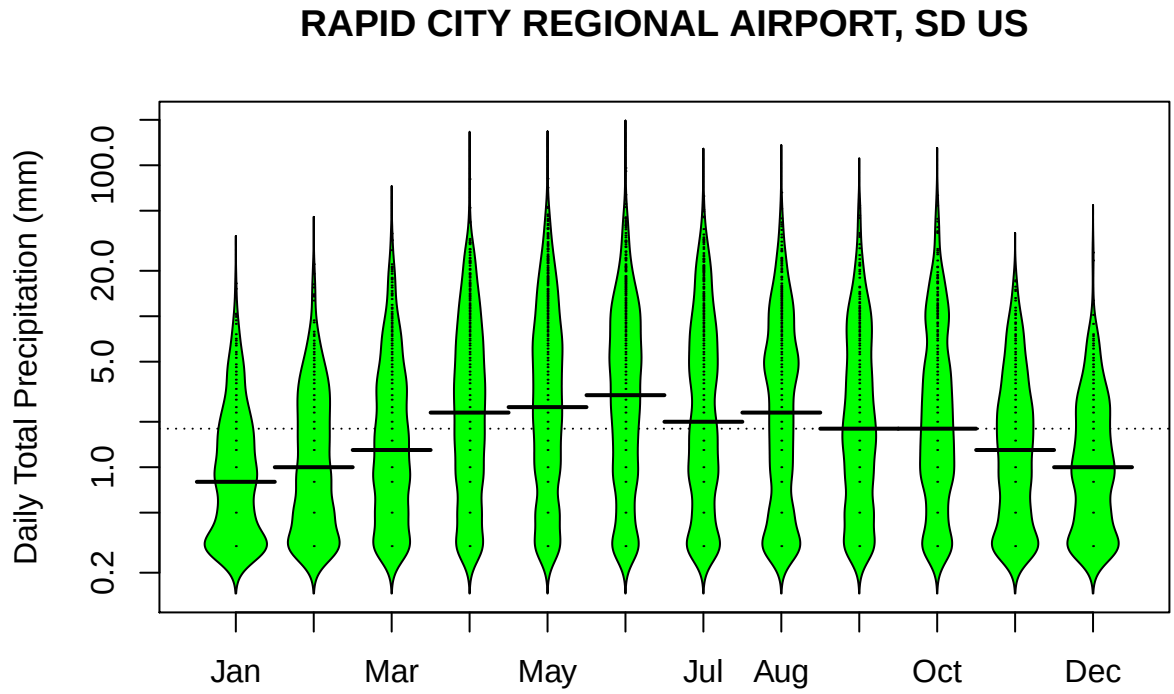

```
beanplot(data$minimum_air_temperature~data$month,
         ll          = 0,
         overallline = "median",
         beanlines   = "median",
         main        = station_name,
         ylab        = "Min Daily Temperature (°C)", # y axis label
         col         = "magenta")
```




```
beanplot(data$precipitation_amount[data$precipitation_amount>0]~data$month[data$precipitation_amount>0]
         ll          = 0,
```

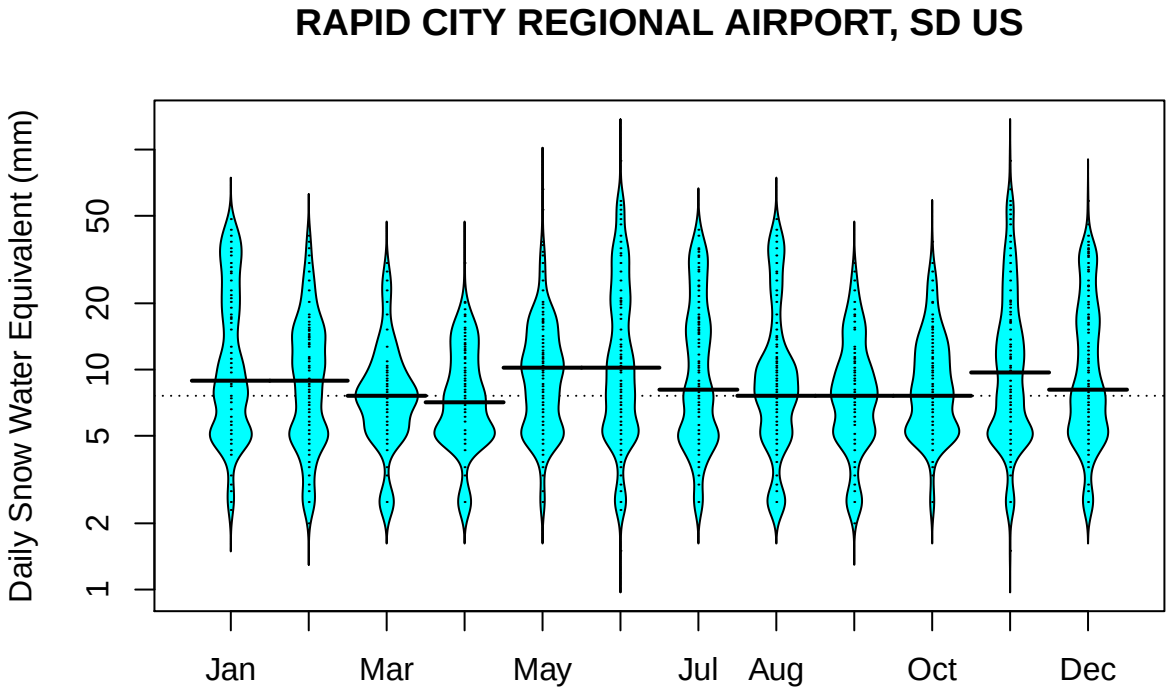
```
        overallline = "median",
        beanlines   = "median",
        main        = station_name,
        ylab        = "Daily Total Precipitation (mm)", # y axis label
        col         = "green")
```

```
## log="y" selected
```



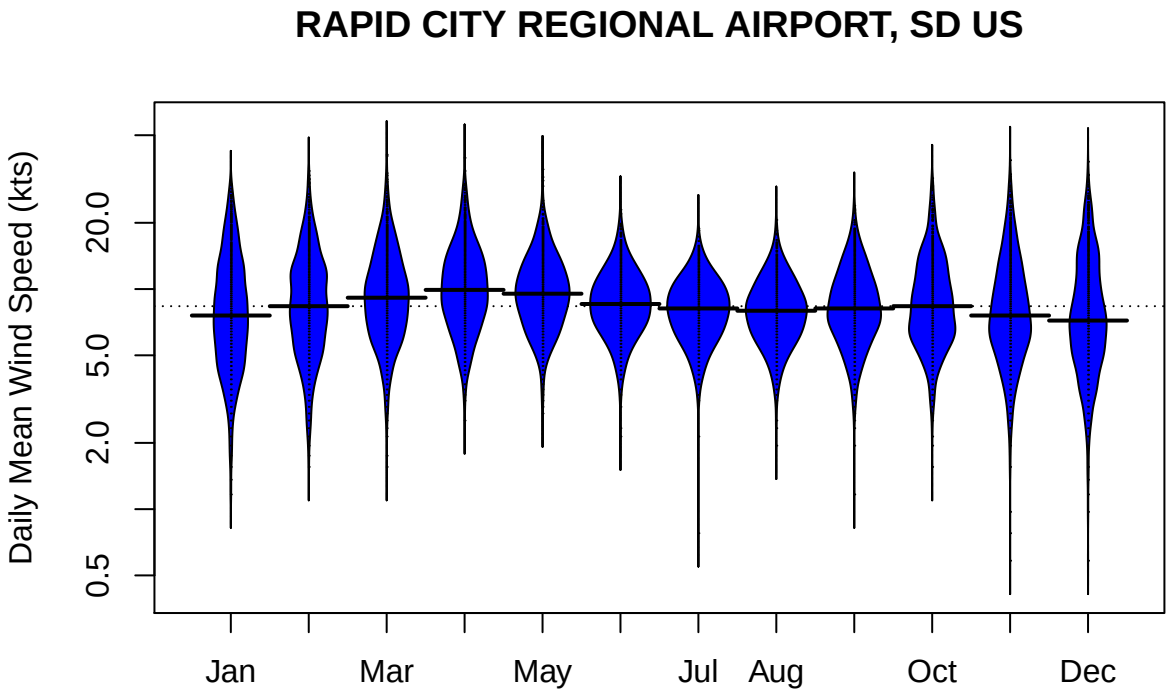

```
beanplot(data$liquid_water_content_of_surface_snow[data$liquid_water_content_of_surface_snow>0]~data$mon
        ll          = 0,
        main        = station_name,
        overallline = "median",
        beanlines   = "median",
        ylab        = "Daily Snow Water Equivalent (mm)", # y axis label
        col         = "cyan")
```

```
## log="y" selected
```



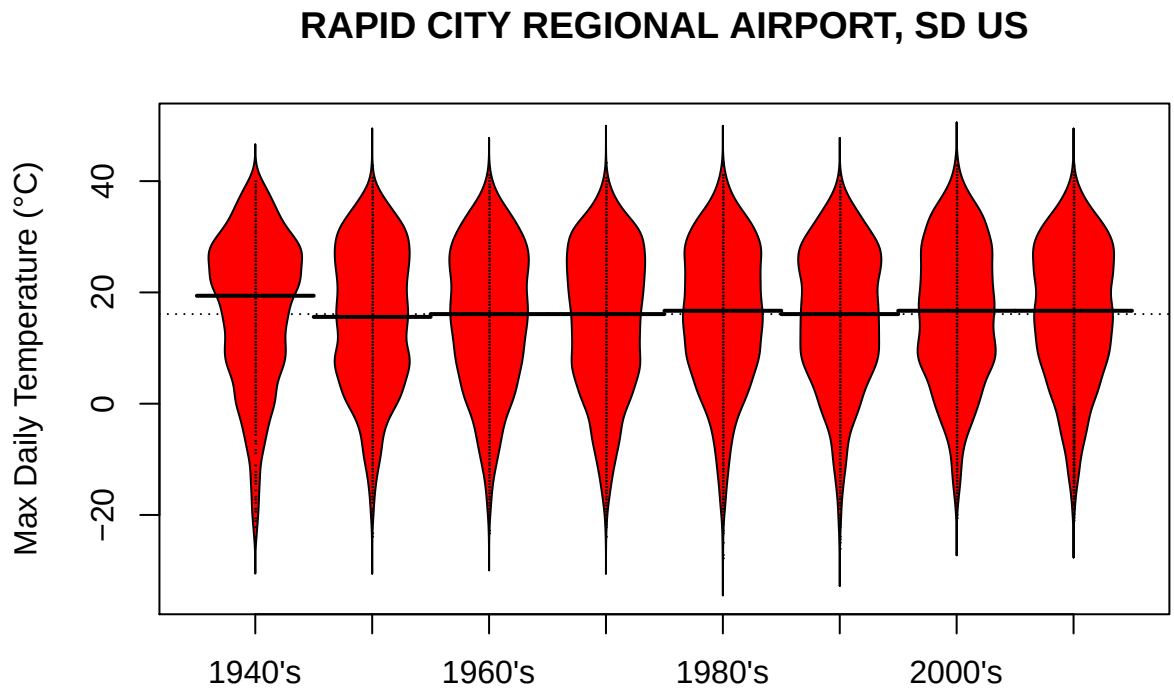

```
beanplot((data$mean_wind_speed*1.94384)~data$month,
         ll          = 0,
         overallline = "median",
         beanlines   = "median",
         main        = station_name,
         ylab        = "Daily Mean Wind Speed (kts)", # y axis label
         col         = "blue")
```

```
## log="y" selected
```

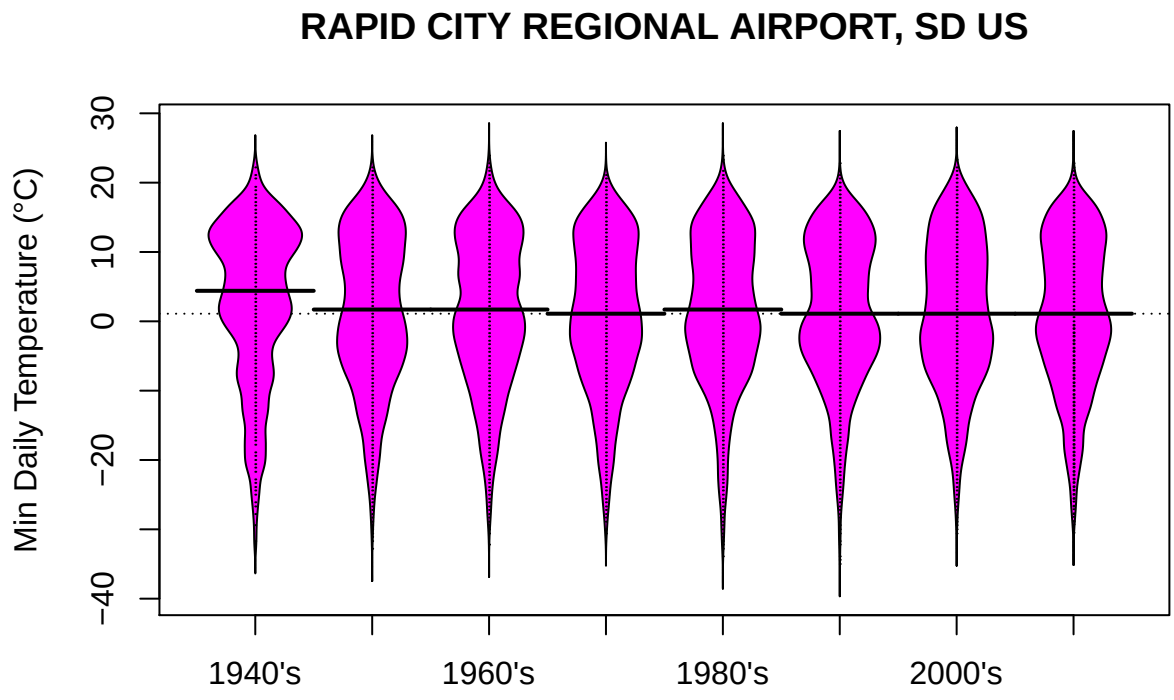


Processing by Decade

```
beanplot(data$maximum_air_temperature~data$decade, # variable
         ll          = 0,                                  # no crosticks per point
         overallline = "median",
         beanlines   = "median",
         main        = station_name,                      # main title
         ylab        = "Max Daily Temperature (°C)",      # y axis label
         col         = "red")                              # colour
```

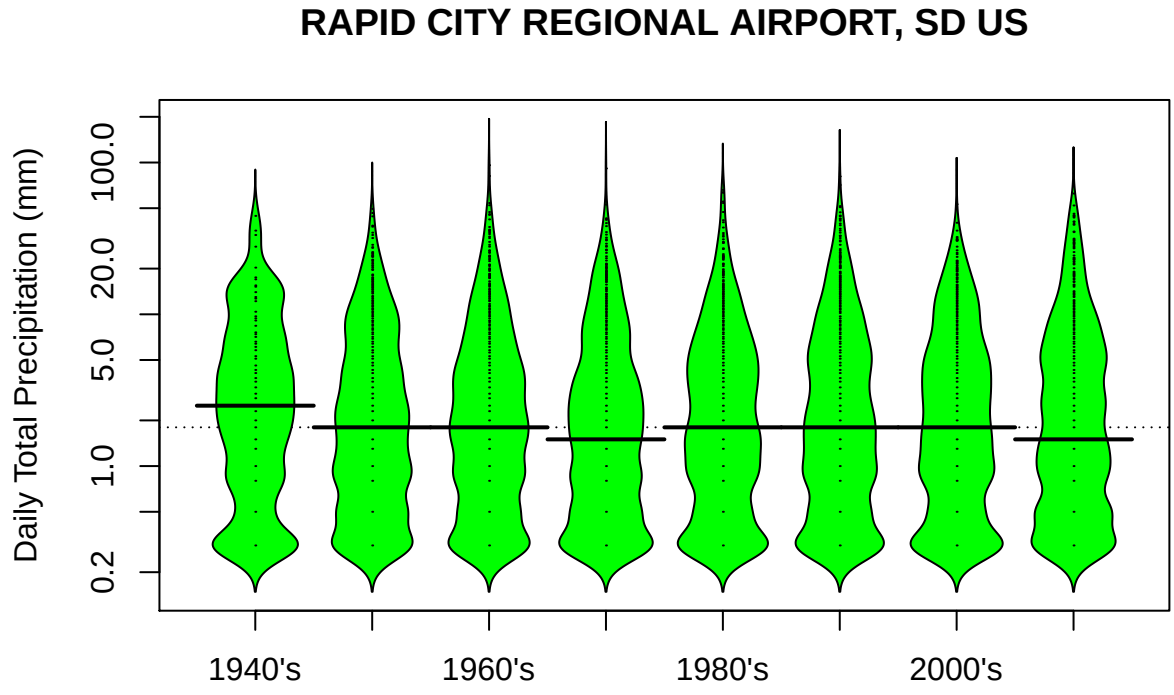



```
beanplot(data$minimum_air_temperature~data$decade,
         ll          = 0,
         overallline = "median",
         beanlines   = "median",
         main        = station_name,
         ylab        = "Min Daily Temperature (°C)", # y axis label
         col         = "magenta")
```



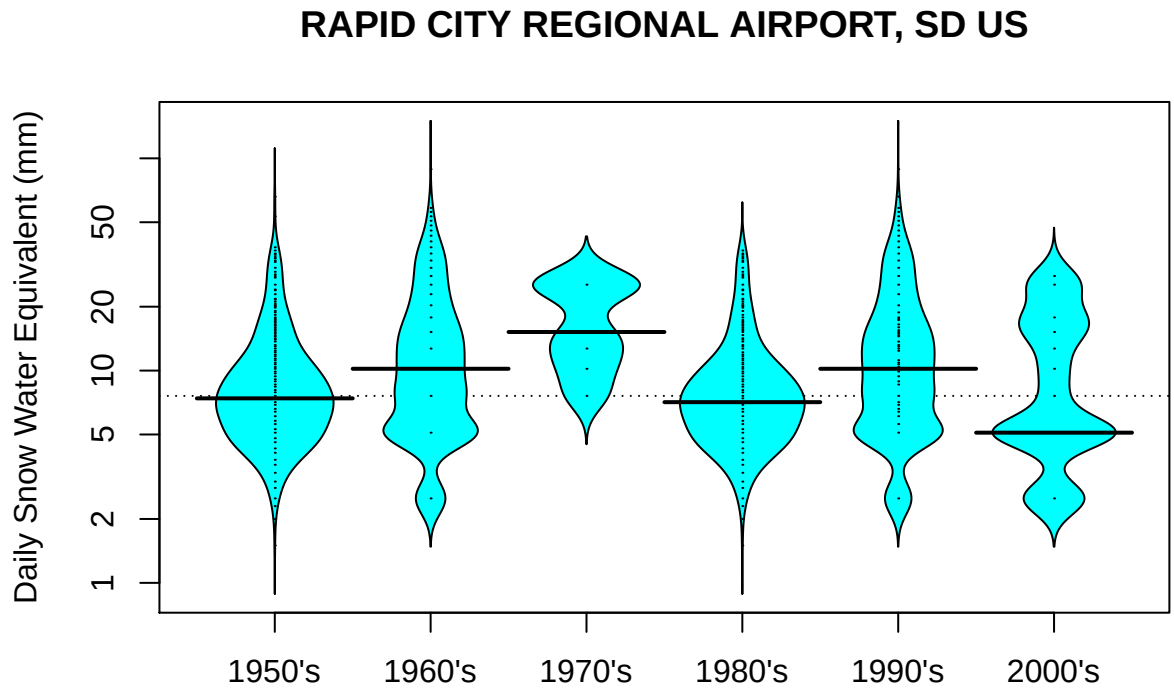

```
beanplot(data$precipitation_amount[data$precipitation_amount>0]~data$decade[data$precipitation_amount>0]
         ll          = 0,
         overallline = "median",
         beanlines   = "median",
         main        = station_name,
         ylab        = "Daily Total Precipitation (mm)", # y axis label
         col         = "green")
```

```
## log="y" selected
```



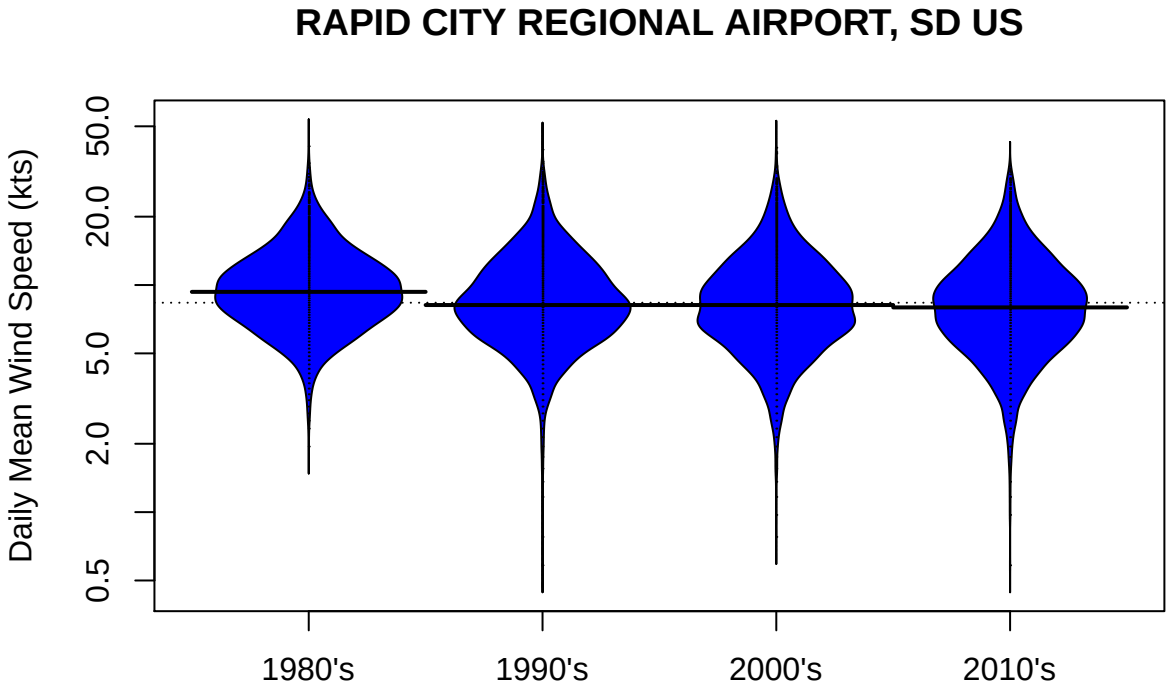
```
beanplot(data$liquid_water_content_of_surface_snow[data$liquid_water_content_of_surface_snow>0]~data$de
         ll          = 0,
         overallline = "median",
         beanlines   = "median",
         main        = station_name,
         ylab        = "Daily Snow Water Equivalent (mm)", # y axis label
         col         = "cyan")
```

```
## log="y" selected
```




```
beanplot((data$mean_wind_speed*1.94384)~data$decade,
         ll          = 0,
         main        = station_name,
         overallline = "median",
         beanlines   = "median",
         ylab        = "Daily Mean Wind Speed (kts)", # y axis label
         col         = "blue")
```

```
## log="y" selected
```


RAPID CITY REGIONAL AIRPORT, SD US
Daily Mean Wind Speed (kts)
0.5
2.0
5.0
20.0
50.0
1980's
1990's
2000's
2010's