Plotting with openair: Exercises

These exercises accompany the Plotting with openair tutorial: http://rpubs.com/NateByers/Openair. These exercises use data frames from the region5air package. Run the following code to clean out your global environment and load the data you need:

rm(list = ls())  
library(region5air)  
library(openair)  
library(dplyr)  
library(tidyr)  
data(chicago\_air)  
data(chicago\_wind)  
data(airdata)

## Exercises

1. Create a properly formatted "date" column in the chicago\_wind dataset. Use the as.POSIXct() function to make it a POSIXct class, and use the rename() function to rename the "datetime" column to "date". Once you have created a properly formatted "date" column, run this filter on the data frame to remove one row with an NAin the "date" column:

chicago\_wind <- filter(chicago\_wind, !is.na(date))

**Note:** One hour was not formatted as a POSIXct class because of the switch to daylight savings time.

[Solution 1](#ex1)

1. Use the summaryPlot() function to visualize the chicago\_wind dataset.

[Solution 2](#ex2)

1. Use the windRose() function on the chicago\_wind dataset and split the data into different panels by season. Remember to rename the "wind\_speed" and "wind\_dir" columns as "ws" and "wd" respectively.

[Solution 3](#ex3)

1. Use the pollutionRose() function on the ozone data in the chicago\_wind data frame and change the statistic parameter to "prop.mean".

[Solution 4](#ex4)

#### Advanced Exercise

1. Use the filter() function to subset the airdata data frame down to the site "840180890022". Use the group\_by() function to group by the "datetime" and "parameter" columns. Use summarize() to replace the "value" column with the mean for multiple values per hour/parameter (i.e., for sites with more than one poc). Usetidyr to reshape the data to a wide format. Make time series plots of the parameters using the timePlot() function in openair. Be sure to rename the columns and format the date column properly. **Hints:** Remember to remove rows that have an NA in the date-time column. Also, the rename() function will not work on this data frame, so use names() <- instead.

[Solution 5](#ex5)

## Solutions

#### Solution 1

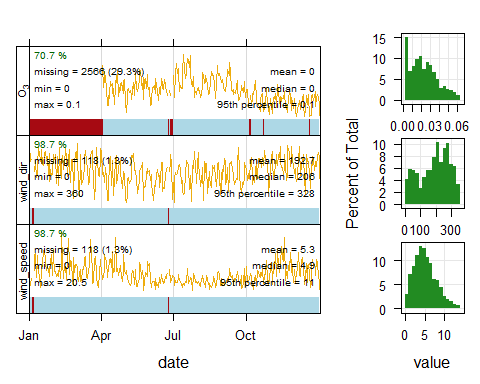
chicago\_wind$datetime <- as.POSIXct(chicago\_wind$datetime, format = "%Y%m%dT%H%M",  
 tz = "America/Chicago")  
chicago\_wind <- rename(chicago\_wind, date = datetime)  
chicago\_wind <- filter(chicago\_wind, !is.na(date))

[Back to exercises](#ex)

#### Solution 2

summaryPlot(chicago\_wind)

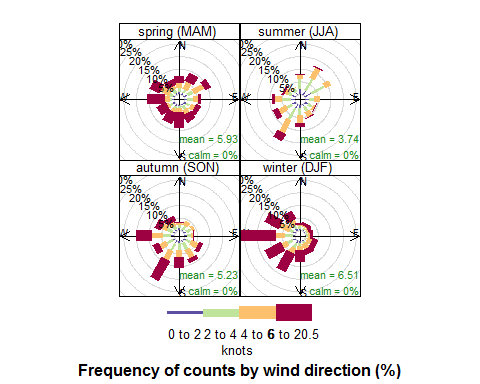
## date1 date2 wind\_speed wind\_dir ozone   
## "POSIXct" "POSIXt" "numeric" "numeric" "numeric"



[Back to exercises](#ex)

#### Solution 3

chicago\_wind <- rename(chicago\_wind, ws = wind\_speed, wd = wind\_dir)  
windRose(chicago\_wind, type = "season", key.footer = "knots")

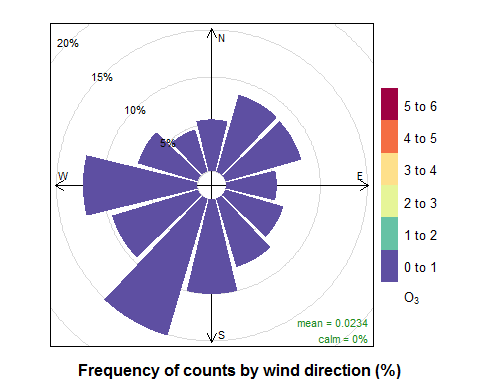


[Back to exercises](#ex)

#### Solution 4

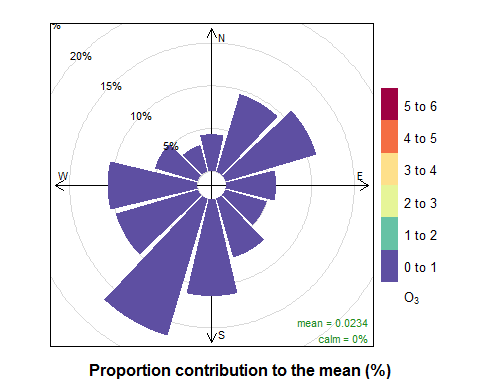
First let's plot with the default statistic of "prop.count".

pollutionRose(chicago\_wind, pollutant = "ozone", statistic = "prop.count")



Now we'll change it to "prop.mean".

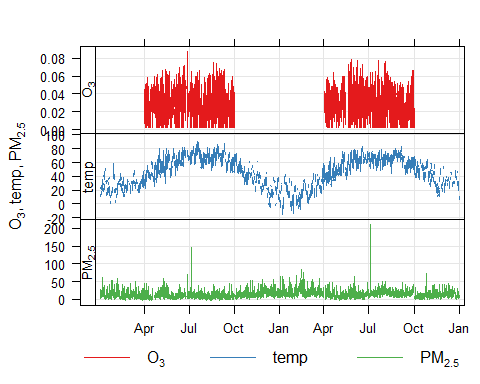
pollutionRose(chicago\_wind, pollutant = "ozone", statistic = "prop.mean")



[Back to exercises](#ex)

#### Solution 5

# filter down to the right monitor and get the mean for multiple pocs  
site22 <- filter(airdata, site == "840180890022")  
site22 <- group\_by(site22, datetime, parameter)  
site22 <- summarize(site22, value = mean(value))  
  
# reshape the data  
site22\_wide <- spread(site22, parameter, value)  
  
# format the date column properly  
site22\_wide$datetime <- as.POSIXct(site22\_wide$datetime, format = "%Y%m%dT%H%M",  
 tz = "America/Chicago")  
  
# some dates weren't converted--remove those  
site22\_wide <- filter(site22\_wide, !is.na(datetime))  
  
# we can't use rename() because the column names are numbers  
# so we'll use names() <-   
names(site22\_wide) <- c("date", "ozone", "temp", "pm2.5")  
  
timePlot(site22\_wide, pollutant = c("ozone", "temp", "pm2.5"))



[Back to exercises](#ex)