Summary, R lectures, Part II

Chapter 5: Graphics with R

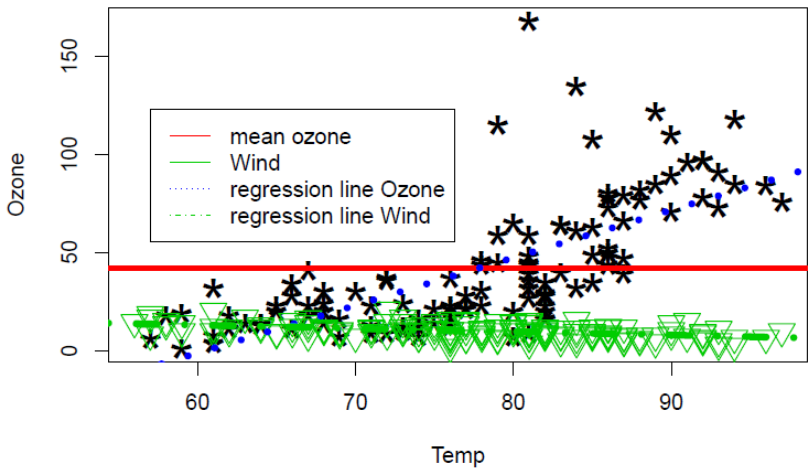
5.1. Simple scatterplot

```
plot(Ozone~Temp, data = airquality, pch = "*", cex = 3)
mean_ozone <- mean(airquality$Ozone, na.rm = T)
abline(h = mean_ozone, lty = 1, col = 2, lwd = 3)

# Adding extra wind measurements
lines(Wind~Temp, data = airquality, pch = 6, col = 3, cex = 2, type = 'p')

# Adding a regression line for Ozone versus Temp
result.lm <- lm(Ozone~Temp, data = airquality)
abline(result.lm, lty = 3, col = 4, lwd = 5)

# Adding a regression line for Wind versus Temp
result2.lm <- lm(Wind~Temp, data = airquality)
abline(result2.lm, lty = 4, col = 3, lwd = 5)
```

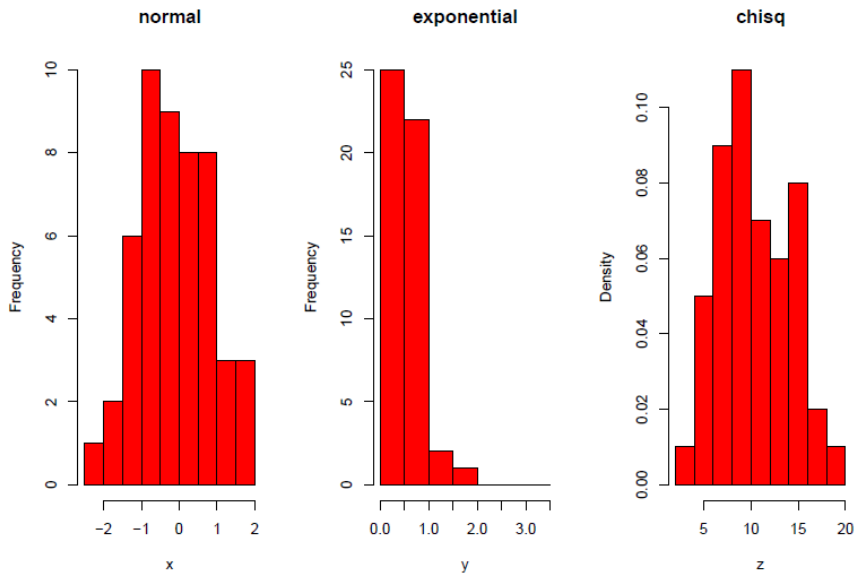


5.2. Histogram

```
hist(x, nclass = 10, main = 'normal', col = 2)

hist(y, breaks = seq(0.0, 3.5, 0.5), main = 'exponential', col = 2)
```

```
hist(z, nclass = 7, probability = T, main = 'chisq', col = 2)
```



5.3. Boxplot

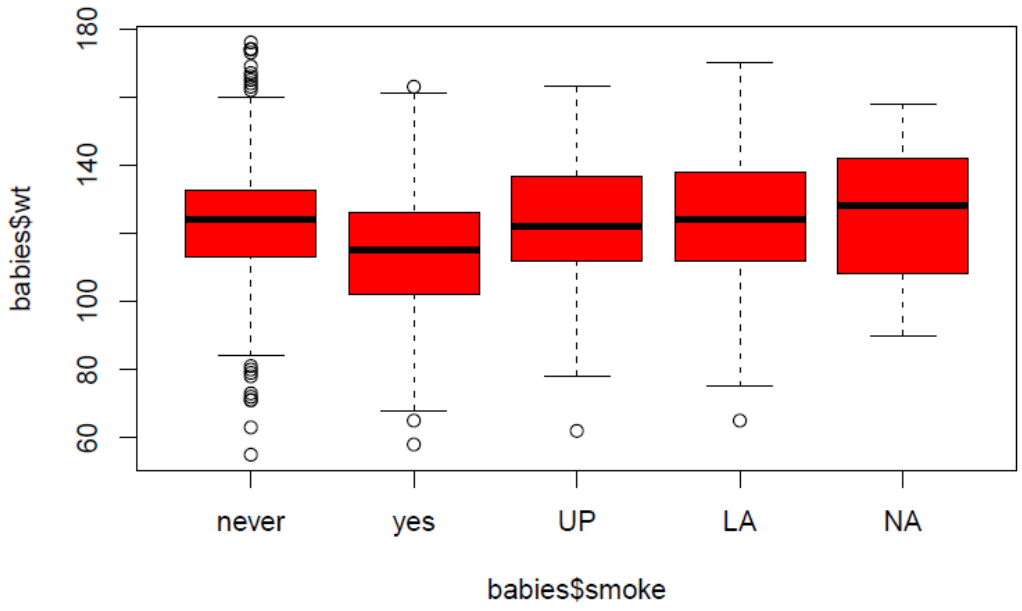
Simple box plot

```
boxplot(babies$wt, ylab = 'birth weight', col = 4)
```

Grouped box plot

```
smoke.names <- c("never", "yes", "UP", "LA", "NA")

boxplot(babies$wt ~ babies$smoke, col = 2, data = babies, names = smoke.names)
```



Chapter 6: Some concepts of dplyr package

2 Basic functions in `dplyr` package

Action	Function
Select observations	`filter()`
Select variables	`select()`
Sort data frames	`arrange()`
Create new variables	`mutate()`, `transmute`
Aggregate	`summarise()`
Grouping	`group_by()`
Merging	`inner_join()`, `left_join()`, `right_join()`, `full_join()`

Always same structure:

```
function(data frame, arguments)
```

Combining multiple operations with the pipe

Example 1 **without** the use of pipe

```
fl_group <- group_by(flights, dest)
fl_descr <- summarise(fl_group,
              avg_delay = mean(arr_delay, na.rm = TRUE),
              sd_delay = sd(arr_delay, na.rm = TRUE),
              max_delay = max(arr_delay, na.rm = TRUE),
              count = sum(!is.na(arr_delay)))
sort_example1 <- arrange(fl_descr, desc(count), avg_delay)
```

Example 1 **with** the use of pipe

```
sort_example1 <- flights %>%
          group_by(dest) %>%
          summarise(
              avg_delay = mean(arr_delay, na.rm = T),
              sd_delay = sd(arr_delay, na.rm = T),
              count = sum(!is.na(arr_delay))
                        ) %>%
          arrange(desc(count), avg_delay)
```

dplyr	merge
`inner_join(x, y)`	`merge(x, y)`
`left_join(x, y)`	`merge(x, y, all.x = TRUE)`
`right_join(x, y)`	`merge(x, y, all.y = TRUE)`
`full_join(x, y)`	`merge(x, y, all.x = TRUE, all.y = TRUE)`

Chapter 7: More on programming with R

Use of the apply functions + loops

4.2.1 Description of the CLT illustration

- Step 1: Generate 30 (= `n`) data points from an *exponential distribution* with rate 3. (hint: use the function `rexp`).
- Step 2: Do this now 5 times. Consider every sample as one new line in a data matrix `mat`. Hence `mat` will be a 5×30 matrix.
- Step 3: Compute a vector `all.samples.means` which has the averages for every sample. This vector has a length of 5.
- Step 4: Make two histograms next to each other (in one and the same graphical window).
    a. The first histogram is a frequency histogram of the first sample (the data from step 1).
    b. The second histogram is a relative frequency histogram of the sample averages, overlayed with the corresponding density curve.

```
# Step 1
rexp(30, rate=3)

# Step 2
mat <- matrix(rep(0,150), nrow=5)
for (i in (1:5))
{
mat[i,] <- rexp(30, rate=3)
}

# Step 3
# compute the average for every sample
all.sample.means <- apply(mat,1,mean)

# Step 4
# create a histogram with the original data of 1st row
#and another histogram with the averages
par(mfrow=c(1,2))
hist(mat[1,],col="blue", main="Distribution of One Sample")
hist(all.sample.means, col="green", main="Sampling Distribution of
     the Mean", prob=T)
lines(density(all.sample.means))
```

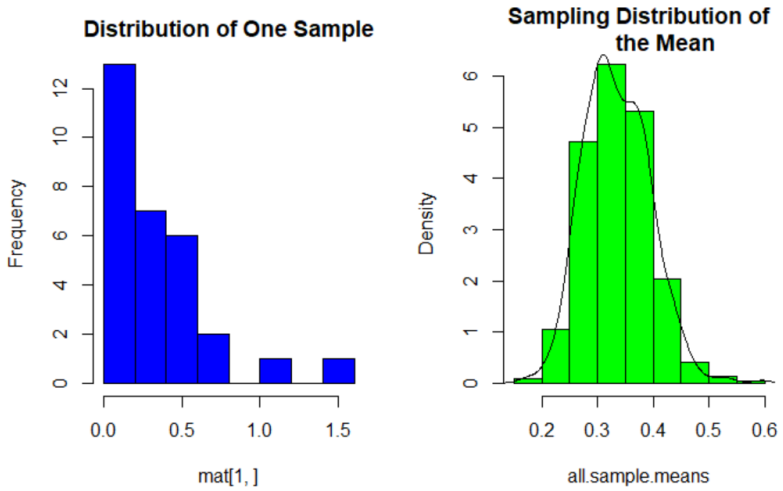
- Step 5: Create now a function which is producing the previous steps (2-4) and has as parameters: `n` (number of data points, default 30), `rpt` (number of samples to take, default 5).
- Step 6: Apply this function when `n = 30` and `rpt = 500`.

```
# Step 5
# create a function out of this
clt_fun <- function(rpt = 5, n=30)
{
  mat <- matrix(rep(0,n*rpt), nrow=rpt)
    for (i in (1:rpt))
        {
      mat[i,] <- rexp(n, rate=3)
        }

  # compute the average for every sample
  all.sample.means <- apply(mat,1,mean)

  # create a histogram with the original data for 1st row
  #and another histogram with the averages
  par(mfrow=c(1,2))
  hist(mat[1,],col="blue", main="Distribution of One Sample")
  hist(all.sample.means, col="green", main="Sampling Distribution of
         the Mean", prob=T)
  lines(density(all.sample.means))
}

# Step 6
clt_fun(rpt=500,n=30)
```



Dates

- 5.1 Create date from strings
- 5.2 Create date from individual components
- 5.3 How to compare to a fixed date?
- 5.4 Once you have a date, you can get components

Spreading and gathering

- 6.1 Gathering: make a long table
- 6.2 Spreading: make a wide table