```r
# Lecture 6: Computer Problem Solving with R commands

# If you wish to use any of the libraries noted below, then you will need to copy and paste the following commands in R first
install.packages("install.load") # install the install.load package
library(install.load) # load the install.load package (library)
install_load("psych", "ggplot2", "DescTools", "prob", "combinat") # install and load the named packages and all of their dependencies (this process may take a while depending on the number of dependencies)


samp <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1)
length(samp) # number of sample observations
mean(samp)


# Data Import
# clipboard
library(psych)

# Excerpt of MillCreekcsv (rows 112 - 125) found in 03431000_MillCreek_AntiochTN_revised.csv
"agency_cd","site_no","datetime","02_00060_00003","02_00060_00003_cd"
"USGS","03431000",1954-01-20,2670,"A"
"USGS","03431000",1954-01-21,1120,"A"
"USGS","03431000",1954-01-22,1300,"A"
"USGS","03431000",1954-01-23,360,"A"
"USGS","03431000",1954-01-24,225,"A"
"USGS","03431000",1954-01-25,156,"A"
"USGS","03431000",1954-01-26,127,"A"
"USGS","03431000",1954-01-27,510,"A"
"USGS","03431000",1954-01-28,183,"A"
"USGS","03431000",1954-01-29,131,"A"
"USGS","03431000",1954-01-30,98,"A"
"USGS","03431000",1954-01-31,78,"A"
"USGS","03431000",1954-02-01,66,"A"
"USGS","03431000",1954-02-02,59,"A"


MillCreekclip <- read.clipboard.csv() # copy the table above, including the blank line below the table and the table header (column names)
mean(MillCreekclip[, 4]) # 02_00060_00003 or column 4 with all rows


pop <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1, 89, 0, 34, 65, 98, 3)
length(pop)
source("varpop.R")
varpop(pop)


pop <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1, 89, 0, 34, 65, 98, 3)
length(pop)
source("varpop.R")
varpop(pop)


samp <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1)
length(samp)
var(samp)


samp <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1)
length(samp)
sd(samp)


samp <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1)
length(samp)
source("cv.R")
cv(samp)
```

```r
samp <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1)
length(samp)
source("sgm.R")
sgm(samp)


# histogram of samp data using ggplot2
# point for both arithmetic and geometric means
meansamp <- mean(samp)
sgmsamp <- sgm(samp)

sampdf <- data.frame(samp) # create a data.frame out of the numeric vector
colnames(sampdf) <- "samp" # change the column name to samp

library(ggplot2)
ggplot(sampdf, aes(x = samp)) + geom_histogram() + geom_point(data = sampdf, shape = 3, size = 3, colour =
"green", aes(x = meansamp, y = 1, show_guide = TRUE)) + geom_text(aes(label = "A M", x = meansamp, y = 1),
vjust = 2, colour = "red") + geom_point(data = sampdf, shape = 3, size = 3, colour = "yellow", aes(x =
sgmsamp, y = 1, show_guide = TRUE)) + geom_text(aes(label = "G M", x = sgmsamp, y = 1), vjust = 2, colour =
"blue") + ggtitle("Comparing Arithmetic (A M) and Geometric Means (G M) For Sample Data")


# histogram of samp data using base R graphics
hist(samp, main = "Histogram of Sample Data")



samp1 <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1, 89, 0, 34, 65, 98, 3)
length(samp1)
median(samp1)


samp2 <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1, 89, 0, 34, 65, 98, 3, 61)
length(samp2)
median(samp2)


samp <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1, 89, 0, 34, 65, 98, 3)
library(DescTools)
Mode(samp)


samp <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1, 89, 0, 34, 65, 98, 3)
length(samp)
range(samp)
source("ranges.R")
ranges(samp)


scores <- c(95, 81, 63, 84, 91, 70, 88, 97, 70, 76, 80, 77, 78, 91, 93, 60, 85, 79, 84, 90)
range(scores)
ranges(scores)
sd(scores)
var(scores)
mean(scores)
sgm(scores)
median(scores)


samp <- c(0.5, 100, 1000.25, 345, 0.0213, 0, 45, 99, 23, 11, 1, 89, 0, 34, 65, 98, 3)
length(samp)
source("rmssam.R")
rmssam(samp)


ctable <- c(3.4, 2.6, 1.6, 1.3, 1.0, 0.5)
```

```r
t <- seq(10, 60, by = 10)
cfunct <- 4.84 * exp(-0.034 * t)
source("relerror.R")
relerror(ctable, cfunct)


d_i <- c(96, 107, 95, 100, 108, 85, 116, 102, 90, 111)

# a) Write a R vector expression to calculate the error associated with each of the trials.

d_target <- 100 # m

err_i <- d_target - d_i
[1]   4  -7   5   0  -8  15 -16  -2  10 -11 # m

# b) Write a R vector expression to calculate the average error for each of the trials.

avg_error <- mean(err_i)
[1] -1 # m

# c) Write a R vector expression to calculate the average of the square of the error for each of the trials,
and then takes the square root of this quantity.

err_i_sq <- (d_target - d_i) ^ 2

avg_err_i_sq <- mean(err_i_sq)

avg_sq_error <- sqrt(avg_err_i_sq)
avg_sq_error
[1] 9.273618 # m

# d) Use the R sd function to find the standard deviation of the data.
sd(d_i)
[1] 9.718253 # m


v <- seq(10, 80, by = 10) # Generate x-values (v) for the data points from 10 to 80 by 10
F <- c(25, 70, 380, 550, 610, 1220, 830, 1450) # These are the y-values (F) for the data points (from the
table)
vequation <- c(0:80) # Generate x-values (vequation) for the linear regression line from 0 to 80 by 1
Fequation <- 19.47024 * vequation - 234.2857 # Generate y-values (Fequation) for the linear regression line

dfeq <- data.frame(vequation, Fequation)
dfp <- data.frame(v, F)
library(ggplot2)
ggplot() + geom_line(data = dfeq, aes(x = vequation, y = Fequation, label = Fequation)) + geom_text(aes(label
= "F = 19.47024 * v - 234.2857 (linear regression line)", x = 20, y = 1000), vjust = 2) + labs(list(title =
"Experimental Data From a Wind Tunnel", x = "Velocity (m/s)", y = "Force (N)")) + geom_point(data = dfp, size
= 3, aes(x = v, y = F, show_guide = TRUE))


v <- seq(10, 80, by = 10)
F <- c(25, 70, 380, 550, 610, 1220, 830, 1450)
lm.F <- lm(F ~ v) # linear model
coef(lm.F) # coefficients of linear model
lm.F
summary(lm.F) # summary of linear model

df <- data.frame(v, F) # create data.frame for use in ggplot2 plot
library(ggplot2)
source("ggplot_smooth_func2.R")
ggplot(data = df, aes(x = v, y = F, label = F)) + stat_smooth_func(geom = "text", method = "lm", hjust = 0,
parse = TRUE) + geom_smooth(method = "lm", se = FALSE) + geom_point() + labs(list(title = "Experimental Data
From a Wind Tunnel", x = "Velocity (m/s)", y = "Force (N)"))


Diameter <- c(7.4,
5.4,
```

```
7.5,
14.0,
7.0,
9.0,
12.0,
5.5,
6.0) # inches

Time <- c(16.41,
12.02,
20.21,
32.62,
17.84,
22.82,
29.48,
15.61,
13.25) # min

lm.T <- lm(Time ~ Diameter) # linear model
coef(lm.T) # coefficients of linear model
lm.T
summary(lm.T) # summary of linear model

df <- data.frame(Diameter, Time) # create data.frame for use in ggplot2 plot
library(ggplot2)
source("ggplot_smooth_func3.R")
ggplot(data = df, aes(x = Diameter, y = Time, label = Time)) + stat_smooth_func(geom = "text", method = "lm",
hjust = 0, parse = TRUE) + geom_smooth(method = "lm", se = FALSE) + geom_point() + labs(list(title =
"Polishing Times for Bowls and Their Diameters", x = "Diameter (inches)", y = "Time (min)"))

Diameter10in <- 10 # inches
Time10in <- coef(lm.T)[[2]] * Diameter10in + coef(lm.T)[[1]]
Time10in
[1] 24.21794 # minutes



library(prob)
cds <- cards(makespace = TRUE) # include the probability column in the cards function and create a data.frame
called cds of the cards function

# What is the probability that a diamond is drawn from the first deck, an ace from the second, and the ace of
hearts from the third?

Diamond <- subset(cds, suit == "Diamond") # subset cds with only the Diamond suit
Diamondprob <- Prob(Diamond) # Calculates the probability

Ace <- subset(cds, rank == "A") # subset cds with only Aces
Aceprob <- Prob(Ace) # Calculates the probability

AceHearts <- subset(cds, rank == "A" & suit == "Heart") # subset cds with only the Heart Ace
AceHeartsprob <- Prob(AceHearts) # Calculates the probability

probcards <- Diamondprob * Aceprob * AceHeartsprob
probcards
[1] 0.0003698225


source("probAB.R")
A <- 1/3
B <- 3/4
probAB(A, B)


library(prob)
bag <- rep(c("Orange", "Green", "White"), times = c(7, 8, 2)) # Create a bag of 7 orange, 8 green, and 2
white balls
bagsample <- urnsamples(bag, size = 2, replace = FALSE, ordered = FALSE) # Create a sample space of picking 2
```

```
balls where there is no replacement and the order does not matter
bagsampleProbspace <- probspace(bagsample) # Probability space of all events occurring
bagsampleProb <- Prob(bagsampleProbspace, isin(bagsampleProbspace, c("White", "Orange"))) # The probability
of picking a white or orange ball
bagsampleProb
[1] 0.1029412


library(prob)
ncol(permsn(5, 3)) # number of columns of permutation
permsn(5, 3)


library(combinat)
ncol(combn(5, 3)) # number of columns of combination
combn(5, 3)


failure <- 6
hours <- 7502
est_failure_rate <- failure / hours
[1] 0.0007997867
```