Homework 4 Arrays and Lists 1,2,3,4

Andrew Marshall

6/19/2018

# Data

In part, this will be an exercise in “programming to the data”. To complete these, you should need only 0.05, 0, 1, 2 and 3 as numeric constants; all other numeric values should come from the data. All the values you will need are defined below:

CaloriesPerRecipeMean <- c(2123.8, 2122.3, 2089.9, 2250.0, 2234.2, 2249.6, 3051.9)  
CaloriesPerRecipeSD <- c(1050.0, 1002.3, 1009.6, 1078.6, 1089.2, 1094.8, 1496.2)  
CaloriesPerServingMean <- c(268.1, 271.1, 280.9, 294.7, 285.6, 288.6, 384.4)  
CaloriesPerServingSD <- c(124.8, 124.2, 116.2, 117.7, 118.3, 122.0, 168.3)  
ServingsPerRecipeMean <- c(12.9, 12.9, 13.0, 12.7, 12.4, 12.4, 12.7)  
ServingsPerRecipeSD <- c(13.3, 13.3, 14.5, 14.6, 14.3, 14.3, 13.0)  
sample.size <- 18  
tenth.increment <- 0.10  
idx.1936 <- 1  
idx.2006 <- length(CaloriesPerRecipeMean)  
idxs36\_07 <- c(idx.1936,idx.2006)  
alpha=0.05

If you find yourself at a point where you feel you must write in your code a numeric value other than c(0.05,0,1,2,3), say, 7, you can do that, but before turning in your code, review this list to see if that value might be in the list, or where such a value might come from in the data.

You will lose 0.5 points for each exercise that requires a numeric constant in your code, other than c(0.05,0,1,2,3), unless you can make a strong argument otherwise.

# Instructions.

The instructions for this homework are different from the previous two:

* There are 6 exercises. Choose 4 to be graded.
* One of the exercises must be completed in both SAS and R. Make sure you document this in the output.
* The other 3 exercises are to be complete in either R or SAS. Make sure you document this in the output.
* You may choose to work the other exercises. If you do, put these *after* the exercises you want graded. Otherwise, we’ll grade the first four exercises and stop grading there. Time permitting, we’ll provide feedback on the additional exercises.
* Exercises 1 and 2 have unit tests; put your code where specified and do not change the unit test chunks.
* This homework is an exercise in working with arrays, array operations and array functions. We haven’t covered iteration in lecture and you should be able to complete this exercise without iteration (for loops, etc). You will not get credit for solutions that require iteration. Similarly, do not use packages for this homework; work in base R.
* You are not required to write additional (other than previous Homework) functions for this exercise, but you may. You will be expected to clearly document additional functions - identify the expected inputs and outputs.

Some of the exercises have detailed instructions and are divided into parts You might find it easier use separate code chunks for each part and divide the work into phases. I find it helpful, in SAS IML, to print("Part a"), etc, preceding phases.

Review your typeset code to ensure that it is readable and appropriately formatted and that all your comments are legible and clearly distinquished from the instructions. It might be a good idea to put your comments in *italicized text*. Don’t use # in literate text,

# it looks like shouting.

This exercise has a lot of equations that don’t always typeset well in Word, so you may wish to refer to the typeset version of this file uploaded to D2L.

You will be asked to reuse functions from the last homework. Copy them into place, directly, without modification, for this exercise. If you need to modify these functions to work with these data, comment out the portions that don’t work but leave the code in place. Add new code that works, then include a comment on your change. What did you need to do as the data changed between this assignment and the last assignment?

# Exercise 1

For this exercise, we will use matrices to test significance of the difference among all the means in Wansink Table 1. This is frequently done using iteration, but we will use array operations for this exercises.

### Part a.

Create a square () matrix M\_1 () with the means for Calories per Recipe in rows, duplicated in columns:

#Generating matrix with requested parameters and assigned it to M\_1  
m\_1 <- matrix(CaloriesPerRecipeMean,nrow = 7,ncol = 7)  
m\_1

## [,1] [,2] [,3] [,4] [,5] [,6] [,7]  
## [1,] 2123.8 2123.8 2123.8 2123.8 2123.8 2123.8 2123.8  
## [2,] 2122.3 2122.3 2122.3 2122.3 2122.3 2122.3 2122.3  
## [3,] 2089.9 2089.9 2089.9 2089.9 2089.9 2089.9 2089.9  
## [4,] 2250.0 2250.0 2250.0 2250.0 2250.0 2250.0 2250.0  
## [5,] 2234.2 2234.2 2234.2 2234.2 2234.2 2234.2 2234.2  
## [6,] 2249.6 2249.6 2249.6 2249.6 2249.6 2249.6 2249.6  
## [7,] 3051.9 3051.9 3051.9 3051.9 3051.9 3051.9 3051.9

You should be able to create this from `CaloriesPerRecipeMean.

Create matrix M\_2 as the transpose of .

#Transposing matrix M\_1 and assigning the result to M\_2  
m\_2 <- t(m\_1)  
m\_2

## [,1] [,2] [,3] [,4] [,5] [,6] [,7]  
## [1,] 2123.8 2122.3 2089.9 2250 2234.2 2249.6 3051.9  
## [2,] 2123.8 2122.3 2089.9 2250 2234.2 2249.6 3051.9  
## [3,] 2123.8 2122.3 2089.9 2250 2234.2 2249.6 3051.9  
## [4,] 2123.8 2122.3 2089.9 2250 2234.2 2249.6 3051.9  
## [5,] 2123.8 2122.3 2089.9 2250 2234.2 2249.6 3051.9  
## [6,] 2123.8 2122.3 2089.9 2250 2234.2 2249.6 3051.9  
## [7,] 2123.8 2122.3 2089.9 2250 2234.2 2249.6 3051.9

Repeat this process with the corresponding standard deviations (matrices S\_1 and S\_2). These values have been defined in **Data**.

Copy your Cohen’s function from Homework 3 into this section and use the four matrices as parameters, assigning the result to d.mat. Print this matrix, then use R/IML functions to determine and print the maximum effect size among all treatment pairs.

Assign to d\_12 the element from this matrix corresponding to the effect size for years 1936 and 2006 to pass the unit test. You should be able to use idx.1936 and idx.2006 as indexes.

d\_12 <- 0  
d\_12CPR <- 0  
  
#Generating 7X7 matrix for using values from CaloriesPerRecipeSD vector  
s\_1 <- matrix(CaloriesPerRecipeSD,nrow = 7, ncol = 7)  
  
#Transposing matrix s\_1  
s\_2 <- t(s\_1)  
  
#Cohen's d function for use with Calories Per Recipe matrix  
d\_12CPR <- function(m\_1,m\_2,s\_1,s\_2) {  
d\_12\_var <- (abs(m\_1-m\_2))/(sqrt((s\_1^2 + s\_2^2)/2))  
return(d\_12\_var)  
}  
d.mat <- d\_12CPR(m\_1,m\_2,s\_1,s\_2)  
  
#Assigning element from matrix d.mat using idx.1936 and idx.2006 to determine effect size.  
d\_12 <- d.mat[max(idxs36\_07)]  
d\_12

## [1] 0.718065555553

## [1] 2

# Exercise 2.

### Part a.

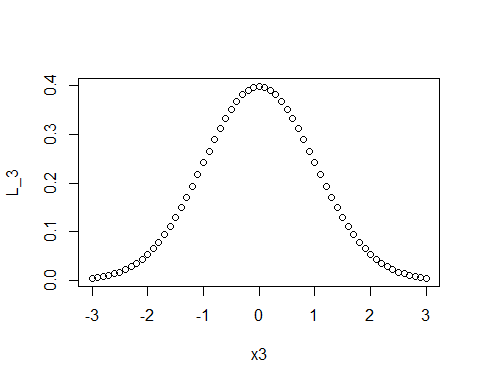
Given and , create three sequences of values, x1 = , x2 = and x3 = , using increments defined by tenth.increment. Use your likelihood function from Homework 2 to calculate for each sequence.

Plot vs for x3.

Calculate and print the sum of each of these sequences, multiplied by the increment value.

Assign appropriate values from one of the sequences to l\_1 and l\_2 to pass the unit tests for this exercise (hint - should be at the midpoint of the sequence, index relative to this element).

mu <- 0  
sigma <- 1  
  
# Generating sequences for x1,x2 and x3  
x1 <- seq(from=(mu-sigma),to=(mu+sigma),by=tenth.increment)  
x2 <- seq(from=(mu-2\*sigma),to=(mu+2\*sigma),by=tenth.increment)  
x3 <- seq(from=(mu-3\*sigma),to=(mu+3\*sigma),by=tenth.increment)  
  
L\_1 <- 0  
L\_2 <- 0  
L\_3 <- 0  
  
# Assigning values to l\_1 and l\_2 for unit testing  
L\_1 <- (1/(sigma \* sqrt(2 \* pi))) \* (exp(-((x1-mu)^2)/(2\*sigma^2)))  
l\_1 <- L\_1[(length(x1)/2)]  
L\_2 <- (1/(sigma \* sqrt(2 \* pi))) \* (exp(-((x2-mu)^2)/(2\*sigma^2)))  
l\_2 <- L\_2[(length(x2)/2)-1]  
L\_3 <- (1/(sigma \* sqrt(2 \* pi))) \* (exp(-((x3-mu)^2)/(2\*sigma^2)))  
  
  
  
# Plotting L vs x  
plot(x3,L\_3)



# Calculating sequence sums  
sum(x1)\*tenth.increment

## [1] 1.11022302463e-16

sum(x2)\*tenth.increment

## [1] 4.66293670343e-16

sum(x3)\*tenth.increment

## [1] 1.06581410364e-15

### Part b.

Create a sequence from , incremented by tenth.increment, given and for Servings per Recipe, 1936, from Wansink Table 1 (use idx.1936), and use your likelihood function to calculate , given and for Servings per Recipe, 1936.

Plot vs . Comment on the similarities or differences between the plots in parts a and b.

Calculate and print the sum of this series, multiplied by tenth.increment. Compare this sum to the sum in part a.

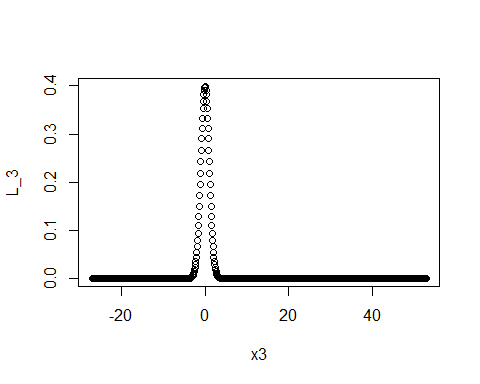
mu <- ServingsPerRecipeMean[idx.1936]  
sigma <- ServingsPerRecipeSD[idx.1936]  
  
# Generating sequence and assigning to x3  
x3 <- seq(from=(mu-3\*sigma),to=(mu+3\*sigma),by=tenth.increment)   
x3

## [1] -2.7000000000e+01 -2.6900000000e+01 -2.6800000000e+01  
## [4] -2.6700000000e+01 -2.6600000000e+01 -2.6500000000e+01  
## [7] -2.6400000000e+01 -2.6300000000e+01 -2.6200000000e+01  
## [10] -2.6100000000e+01 -2.6000000000e+01 -2.5900000000e+01  
## [13] -2.5800000000e+01 -2.5700000000e+01 -2.5600000000e+01  
## [16] -2.5500000000e+01 -2.5400000000e+01 -2.5300000000e+01  
## [19] -2.5200000000e+01 -2.5100000000e+01 -2.5000000000e+01  
## [22] -2.4900000000e+01 -2.4800000000e+01 -2.4700000000e+01  
## [25] -2.4600000000e+01 -2.4500000000e+01 -2.4400000000e+01  
## [28] -2.4300000000e+01 -2.4200000000e+01 -2.4100000000e+01  
## [31] -2.4000000000e+01 -2.3900000000e+01 -2.3800000000e+01  
## [34] -2.3700000000e+01 -2.3600000000e+01 -2.3500000000e+01  
## [37] -2.3400000000e+01 -2.3300000000e+01 -2.3200000000e+01  
## [40] -2.3100000000e+01 -2.3000000000e+01 -2.2900000000e+01  
## [43] -2.2800000000e+01 -2.2700000000e+01 -2.2600000000e+01  
## [46] -2.2500000000e+01 -2.2400000000e+01 -2.2300000000e+01  
## [49] -2.2200000000e+01 -2.2100000000e+01 -2.2000000000e+01  
## [52] -2.1900000000e+01 -2.1800000000e+01 -2.1700000000e+01  
## [55] -2.1600000000e+01 -2.1500000000e+01 -2.1400000000e+01  
## [58] -2.1300000000e+01 -2.1200000000e+01 -2.1100000000e+01  
## [61] -2.1000000000e+01 -2.0900000000e+01 -2.0800000000e+01  
## [64] -2.0700000000e+01 -2.0600000000e+01 -2.0500000000e+01  
## [67] -2.0400000000e+01 -2.0300000000e+01 -2.0200000000e+01  
## [70] -2.0100000000e+01 -2.0000000000e+01 -1.9900000000e+01  
## [73] -1.9800000000e+01 -1.9700000000e+01 -1.9600000000e+01  
## [76] -1.9500000000e+01 -1.9400000000e+01 -1.9300000000e+01  
## [79] -1.9200000000e+01 -1.9100000000e+01 -1.9000000000e+01  
## [82] -1.8900000000e+01 -1.8800000000e+01 -1.8700000000e+01  
## [85] -1.8600000000e+01 -1.8500000000e+01 -1.8400000000e+01  
## [88] -1.8300000000e+01 -1.8200000000e+01 -1.8100000000e+01  
## [91] -1.8000000000e+01 -1.7900000000e+01 -1.7800000000e+01  
## [94] -1.7700000000e+01 -1.7600000000e+01 -1.7500000000e+01  
## [97] -1.7400000000e+01 -1.7300000000e+01 -1.7200000000e+01  
## [100] -1.7100000000e+01 -1.7000000000e+01 -1.6900000000e+01  
## [103] -1.6800000000e+01 -1.6700000000e+01 -1.6600000000e+01  
## [106] -1.6500000000e+01 -1.6400000000e+01 -1.6300000000e+01  
## [109] -1.6200000000e+01 -1.6100000000e+01 -1.6000000000e+01  
## [112] -1.5900000000e+01 -1.5800000000e+01 -1.5700000000e+01  
## [115] -1.5600000000e+01 -1.5500000000e+01 -1.5400000000e+01  
## [118] -1.5300000000e+01 -1.5200000000e+01 -1.5100000000e+01  
## [121] -1.5000000000e+01 -1.4900000000e+01 -1.4800000000e+01  
## [124] -1.4700000000e+01 -1.4600000000e+01 -1.4500000000e+01  
## [127] -1.4400000000e+01 -1.4300000000e+01 -1.4200000000e+01  
## [130] -1.4100000000e+01 -1.4000000000e+01 -1.3900000000e+01  
## [133] -1.3800000000e+01 -1.3700000000e+01 -1.3600000000e+01  
## [136] -1.3500000000e+01 -1.3400000000e+01 -1.3300000000e+01  
## [139] -1.3200000000e+01 -1.3100000000e+01 -1.3000000000e+01  
## [142] -1.2900000000e+01 -1.2800000000e+01 -1.2700000000e+01  
## [145] -1.2600000000e+01 -1.2500000000e+01 -1.2400000000e+01  
## [148] -1.2300000000e+01 -1.2200000000e+01 -1.2100000000e+01  
## [151] -1.2000000000e+01 -1.1900000000e+01 -1.1800000000e+01  
## [154] -1.1700000000e+01 -1.1600000000e+01 -1.1500000000e+01  
## [157] -1.1400000000e+01 -1.1300000000e+01 -1.1200000000e+01  
## [160] -1.1100000000e+01 -1.1000000000e+01 -1.0900000000e+01  
## [163] -1.0800000000e+01 -1.0700000000e+01 -1.0600000000e+01  
## [166] -1.0500000000e+01 -1.0400000000e+01 -1.0300000000e+01  
## [169] -1.0200000000e+01 -1.0100000000e+01 -1.0000000000e+01  
## [172] -9.9000000000e+00 -9.8000000000e+00 -9.7000000000e+00  
## [175] -9.6000000000e+00 -9.5000000000e+00 -9.4000000000e+00  
## [178] -9.3000000000e+00 -9.2000000000e+00 -9.1000000000e+00  
## [181] -9.0000000000e+00 -8.9000000000e+00 -8.8000000000e+00  
## [184] -8.7000000000e+00 -8.6000000000e+00 -8.5000000000e+00  
## [187] -8.4000000000e+00 -8.3000000000e+00 -8.2000000000e+00  
## [190] -8.1000000000e+00 -8.0000000000e+00 -7.9000000000e+00  
## [193] -7.8000000000e+00 -7.7000000000e+00 -7.6000000000e+00  
## [196] -7.5000000000e+00 -7.4000000000e+00 -7.3000000000e+00  
## [199] -7.2000000000e+00 -7.1000000000e+00 -7.0000000000e+00  
## [202] -6.9000000000e+00 -6.8000000000e+00 -6.7000000000e+00  
## [205] -6.6000000000e+00 -6.5000000000e+00 -6.4000000000e+00  
## [208] -6.3000000000e+00 -6.2000000000e+00 -6.1000000000e+00  
## [211] -6.0000000000e+00 -5.9000000000e+00 -5.8000000000e+00  
## [214] -5.7000000000e+00 -5.6000000000e+00 -5.5000000000e+00  
## [217] -5.4000000000e+00 -5.3000000000e+00 -5.2000000000e+00  
## [220] -5.1000000000e+00 -5.0000000000e+00 -4.9000000000e+00  
## [223] -4.8000000000e+00 -4.7000000000e+00 -4.6000000000e+00  
## [226] -4.5000000000e+00 -4.4000000000e+00 -4.3000000000e+00  
## [229] -4.2000000000e+00 -4.1000000000e+00 -4.0000000000e+00  
## [232] -3.9000000000e+00 -3.8000000000e+00 -3.7000000000e+00  
## [235] -3.6000000000e+00 -3.5000000000e+00 -3.4000000000e+00  
## [238] -3.3000000000e+00 -3.2000000000e+00 -3.1000000000e+00  
## [241] -3.0000000000e+00 -2.9000000000e+00 -2.8000000000e+00  
## [244] -2.7000000000e+00 -2.6000000000e+00 -2.5000000000e+00  
## [247] -2.4000000000e+00 -2.3000000000e+00 -2.2000000000e+00  
## [250] -2.1000000000e+00 -2.0000000000e+00 -1.9000000000e+00  
## [253] -1.8000000000e+00 -1.7000000000e+00 -1.6000000000e+00  
## [256] -1.5000000000e+00 -1.4000000000e+00 -1.3000000000e+00  
## [259] -1.2000000000e+00 -1.1000000000e+00 -1.0000000000e+00  
## [262] -9.0000000000e-01 -8.0000000000e-01 -7.0000000000e-01  
## [265] -6.0000000000e-01 -5.0000000000e-01 -4.0000000000e-01  
## [268] -3.0000000000e-01 -2.0000000000e-01 -1.0000000000e-01  
## [271] -7.1054273576e-15 1.0000000000e-01 2.0000000000e-01  
## [274] 3.0000000000e-01 4.0000000000e-01 5.0000000000e-01  
## [277] 6.0000000000e-01 7.0000000000e-01 8.0000000000e-01  
## [280] 9.0000000000e-01 1.0000000000e+00 1.1000000000e+00  
## [283] 1.2000000000e+00 1.3000000000e+00 1.4000000000e+00  
## [286] 1.5000000000e+00 1.6000000000e+00 1.7000000000e+00  
## [289] 1.8000000000e+00 1.9000000000e+00 2.0000000000e+00  
## [292] 2.1000000000e+00 2.2000000000e+00 2.3000000000e+00  
## [295] 2.4000000000e+00 2.5000000000e+00 2.6000000000e+00  
## [298] 2.7000000000e+00 2.8000000000e+00 2.9000000000e+00  
## [301] 3.0000000000e+00 3.1000000000e+00 3.2000000000e+00  
## [304] 3.3000000000e+00 3.4000000000e+00 3.5000000000e+00  
## [307] 3.6000000000e+00 3.7000000000e+00 3.8000000000e+00  
## [310] 3.9000000000e+00 4.0000000000e+00 4.1000000000e+00  
## [313] 4.2000000000e+00 4.3000000000e+00 4.4000000000e+00  
## [316] 4.5000000000e+00 4.6000000000e+00 4.7000000000e+00  
## [319] 4.8000000000e+00 4.9000000000e+00 5.0000000000e+00  
## [322] 5.1000000000e+00 5.2000000000e+00 5.3000000000e+00  
## [325] 5.4000000000e+00 5.5000000000e+00 5.6000000000e+00  
## [328] 5.7000000000e+00 5.8000000000e+00 5.9000000000e+00  
## [331] 6.0000000000e+00 6.1000000000e+00 6.2000000000e+00  
## [334] 6.3000000000e+00 6.4000000000e+00 6.5000000000e+00  
## [337] 6.6000000000e+00 6.7000000000e+00 6.8000000000e+00  
## [340] 6.9000000000e+00 7.0000000000e+00 7.1000000000e+00  
## [343] 7.2000000000e+00 7.3000000000e+00 7.4000000000e+00  
## [346] 7.5000000000e+00 7.6000000000e+00 7.7000000000e+00  
## [349] 7.8000000000e+00 7.9000000000e+00 8.0000000000e+00  
## [352] 8.1000000000e+00 8.2000000000e+00 8.3000000000e+00  
## [355] 8.4000000000e+00 8.5000000000e+00 8.6000000000e+00  
## [358] 8.7000000000e+00 8.8000000000e+00 8.9000000000e+00  
## [361] 9.0000000000e+00 9.1000000000e+00 9.2000000000e+00  
## [364] 9.3000000000e+00 9.4000000000e+00 9.5000000000e+00  
## [367] 9.6000000000e+00 9.7000000000e+00 9.8000000000e+00  
## [370] 9.9000000000e+00 1.0000000000e+01 1.0100000000e+01  
## [373] 1.0200000000e+01 1.0300000000e+01 1.0400000000e+01  
## [376] 1.0500000000e+01 1.0600000000e+01 1.0700000000e+01  
## [379] 1.0800000000e+01 1.0900000000e+01 1.1000000000e+01  
## [382] 1.1100000000e+01 1.1200000000e+01 1.1300000000e+01  
## [385] 1.1400000000e+01 1.1500000000e+01 1.1600000000e+01  
## [388] 1.1700000000e+01 1.1800000000e+01 1.1900000000e+01  
## [391] 1.2000000000e+01 1.2100000000e+01 1.2200000000e+01  
## [394] 1.2300000000e+01 1.2400000000e+01 1.2500000000e+01  
## [397] 1.2600000000e+01 1.2700000000e+01 1.2800000000e+01  
## [400] 1.2900000000e+01 1.3000000000e+01 1.3100000000e+01  
## [403] 1.3200000000e+01 1.3300000000e+01 1.3400000000e+01  
## [406] 1.3500000000e+01 1.3600000000e+01 1.3700000000e+01  
## [409] 1.3800000000e+01 1.3900000000e+01 1.4000000000e+01  
## [412] 1.4100000000e+01 1.4200000000e+01 1.4300000000e+01  
## [415] 1.4400000000e+01 1.4500000000e+01 1.4600000000e+01  
## [418] 1.4700000000e+01 1.4800000000e+01 1.4900000000e+01  
## [421] 1.5000000000e+01 1.5100000000e+01 1.5200000000e+01  
## [424] 1.5300000000e+01 1.5400000000e+01 1.5500000000e+01  
## [427] 1.5600000000e+01 1.5700000000e+01 1.5800000000e+01  
## [430] 1.5900000000e+01 1.6000000000e+01 1.6100000000e+01  
## [433] 1.6200000000e+01 1.6300000000e+01 1.6400000000e+01  
## [436] 1.6500000000e+01 1.6600000000e+01 1.6700000000e+01  
## [439] 1.6800000000e+01 1.6900000000e+01 1.7000000000e+01  
## [442] 1.7100000000e+01 1.7200000000e+01 1.7300000000e+01  
## [445] 1.7400000000e+01 1.7500000000e+01 1.7600000000e+01  
## [448] 1.7700000000e+01 1.7800000000e+01 1.7900000000e+01  
## [451] 1.8000000000e+01 1.8100000000e+01 1.8200000000e+01  
## [454] 1.8300000000e+01 1.8400000000e+01 1.8500000000e+01  
## [457] 1.8600000000e+01 1.8700000000e+01 1.8800000000e+01  
## [460] 1.8900000000e+01 1.9000000000e+01 1.9100000000e+01  
## [463] 1.9200000000e+01 1.9300000000e+01 1.9400000000e+01  
## [466] 1.9500000000e+01 1.9600000000e+01 1.9700000000e+01  
## [469] 1.9800000000e+01 1.9900000000e+01 2.0000000000e+01  
## [472] 2.0100000000e+01 2.0200000000e+01 2.0300000000e+01  
## [475] 2.0400000000e+01 2.0500000000e+01 2.0600000000e+01  
## [478] 2.0700000000e+01 2.0800000000e+01 2.0900000000e+01  
## [481] 2.1000000000e+01 2.1100000000e+01 2.1200000000e+01  
## [484] 2.1300000000e+01 2.1400000000e+01 2.1500000000e+01  
## [487] 2.1600000000e+01 2.1700000000e+01 2.1800000000e+01  
## [490] 2.1900000000e+01 2.2000000000e+01 2.2100000000e+01  
## [493] 2.2200000000e+01 2.2300000000e+01 2.2400000000e+01  
## [496] 2.2500000000e+01 2.2600000000e+01 2.2700000000e+01  
## [499] 2.2800000000e+01 2.2900000000e+01 2.3000000000e+01  
## [502] 2.3100000000e+01 2.3200000000e+01 2.3300000000e+01  
## [505] 2.3400000000e+01 2.3500000000e+01 2.3600000000e+01  
## [508] 2.3700000000e+01 2.3800000000e+01 2.3900000000e+01  
## [511] 2.4000000000e+01 2.4100000000e+01 2.4200000000e+01  
## [514] 2.4300000000e+01 2.4400000000e+01 2.4500000000e+01  
## [517] 2.4600000000e+01 2.4700000000e+01 2.4800000000e+01  
## [520] 2.4900000000e+01 2.5000000000e+01 2.5100000000e+01  
## [523] 2.5200000000e+01 2.5300000000e+01 2.5400000000e+01  
## [526] 2.5500000000e+01 2.5600000000e+01 2.5700000000e+01  
## [529] 2.5800000000e+01 2.5900000000e+01 2.6000000000e+01  
## [532] 2.6100000000e+01 2.6200000000e+01 2.6300000000e+01  
## [535] 2.6400000000e+01 2.6500000000e+01 2.6600000000e+01  
## [538] 2.6700000000e+01 2.6800000000e+01 2.6900000000e+01  
## [541] 2.7000000000e+01 2.7100000000e+01 2.7200000000e+01  
## [544] 2.7300000000e+01 2.7400000000e+01 2.7500000000e+01  
## [547] 2.7600000000e+01 2.7700000000e+01 2.7800000000e+01  
## [550] 2.7900000000e+01 2.8000000000e+01 2.8100000000e+01  
## [553] 2.8200000000e+01 2.8300000000e+01 2.8400000000e+01  
## [556] 2.8500000000e+01 2.8600000000e+01 2.8700000000e+01  
## [559] 2.8800000000e+01 2.8900000000e+01 2.9000000000e+01  
## [562] 2.9100000000e+01 2.9200000000e+01 2.9300000000e+01  
## [565] 2.9400000000e+01 2.9500000000e+01 2.9600000000e+01  
## [568] 2.9700000000e+01 2.9800000000e+01 2.9900000000e+01  
## [571] 3.0000000000e+01 3.0100000000e+01 3.0200000000e+01  
## [574] 3.0300000000e+01 3.0400000000e+01 3.0500000000e+01  
## [577] 3.0600000000e+01 3.0700000000e+01 3.0800000000e+01  
## [580] 3.0900000000e+01 3.1000000000e+01 3.1100000000e+01  
## [583] 3.1200000000e+01 3.1300000000e+01 3.1400000000e+01  
## [586] 3.1500000000e+01 3.1600000000e+01 3.1700000000e+01  
## [589] 3.1800000000e+01 3.1900000000e+01 3.2000000000e+01  
## [592] 3.2100000000e+01 3.2200000000e+01 3.2300000000e+01  
## [595] 3.2400000000e+01 3.2500000000e+01 3.2600000000e+01  
## [598] 3.2700000000e+01 3.2800000000e+01 3.2900000000e+01  
## [601] 3.3000000000e+01 3.3100000000e+01 3.3200000000e+01  
## [604] 3.3300000000e+01 3.3400000000e+01 3.3500000000e+01  
## [607] 3.3600000000e+01 3.3700000000e+01 3.3800000000e+01  
## [610] 3.3900000000e+01 3.4000000000e+01 3.4100000000e+01  
## [613] 3.4200000000e+01 3.4300000000e+01 3.4400000000e+01  
## [616] 3.4500000000e+01 3.4600000000e+01 3.4700000000e+01  
## [619] 3.4800000000e+01 3.4900000000e+01 3.5000000000e+01  
## [622] 3.5100000000e+01 3.5200000000e+01 3.5300000000e+01  
## [625] 3.5400000000e+01 3.5500000000e+01 3.5600000000e+01  
## [628] 3.5700000000e+01 3.5800000000e+01 3.5900000000e+01  
## [631] 3.6000000000e+01 3.6100000000e+01 3.6200000000e+01  
## [634] 3.6300000000e+01 3.6400000000e+01 3.6500000000e+01  
## [637] 3.6600000000e+01 3.6700000000e+01 3.6800000000e+01  
## [640] 3.6900000000e+01 3.7000000000e+01 3.7100000000e+01  
## [643] 3.7200000000e+01 3.7300000000e+01 3.7400000000e+01  
## [646] 3.7500000000e+01 3.7600000000e+01 3.7700000000e+01  
## [649] 3.7800000000e+01 3.7900000000e+01 3.8000000000e+01  
## [652] 3.8100000000e+01 3.8200000000e+01 3.8300000000e+01  
## [655] 3.8400000000e+01 3.8500000000e+01 3.8600000000e+01  
## [658] 3.8700000000e+01 3.8800000000e+01 3.8900000000e+01  
## [661] 3.9000000000e+01 3.9100000000e+01 3.9200000000e+01  
## [664] 3.9300000000e+01 3.9400000000e+01 3.9500000000e+01  
## [667] 3.9600000000e+01 3.9700000000e+01 3.9800000000e+01  
## [670] 3.9900000000e+01 4.0000000000e+01 4.0100000000e+01  
## [673] 4.0200000000e+01 4.0300000000e+01 4.0400000000e+01  
## [676] 4.0500000000e+01 4.0600000000e+01 4.0700000000e+01  
## [679] 4.0800000000e+01 4.0900000000e+01 4.1000000000e+01  
## [682] 4.1100000000e+01 4.1200000000e+01 4.1300000000e+01  
## [685] 4.1400000000e+01 4.1500000000e+01 4.1600000000e+01  
## [688] 4.1700000000e+01 4.1800000000e+01 4.1900000000e+01  
## [691] 4.2000000000e+01 4.2100000000e+01 4.2200000000e+01  
## [694] 4.2300000000e+01 4.2400000000e+01 4.2500000000e+01  
## [697] 4.2600000000e+01 4.2700000000e+01 4.2800000000e+01  
## [700] 4.2900000000e+01 4.3000000000e+01 4.3100000000e+01  
## [703] 4.3200000000e+01 4.3300000000e+01 4.3400000000e+01  
## [706] 4.3500000000e+01 4.3600000000e+01 4.3700000000e+01  
## [709] 4.3800000000e+01 4.3900000000e+01 4.4000000000e+01  
## [712] 4.4100000000e+01 4.4200000000e+01 4.4300000000e+01  
## [715] 4.4400000000e+01 4.4500000000e+01 4.4600000000e+01  
## [718] 4.4700000000e+01 4.4800000000e+01 4.4900000000e+01  
## [721] 4.5000000000e+01 4.5100000000e+01 4.5200000000e+01  
## [724] 4.5300000000e+01 4.5400000000e+01 4.5500000000e+01  
## [727] 4.5600000000e+01 4.5700000000e+01 4.5800000000e+01  
## [730] 4.5900000000e+01 4.6000000000e+01 4.6100000000e+01  
## [733] 4.6200000000e+01 4.6300000000e+01 4.6400000000e+01  
## [736] 4.6500000000e+01 4.6600000000e+01 4.6700000000e+01  
## [739] 4.6800000000e+01 4.6900000000e+01 4.7000000000e+01  
## [742] 4.7100000000e+01 4.7200000000e+01 4.7300000000e+01  
## [745] 4.7400000000e+01 4.7500000000e+01 4.7600000000e+01  
## [748] 4.7700000000e+01 4.7800000000e+01 4.7900000000e+01  
## [751] 4.8000000000e+01 4.8100000000e+01 4.8200000000e+01  
## [754] 4.8300000000e+01 4.8400000000e+01 4.8500000000e+01  
## [757] 4.8600000000e+01 4.8700000000e+01 4.8800000000e+01  
## [760] 4.8900000000e+01 4.9000000000e+01 4.9100000000e+01  
## [763] 4.9200000000e+01 4.9300000000e+01 4.9400000000e+01  
## [766] 4.9500000000e+01 4.9600000000e+01 4.9700000000e+01  
## [769] 4.9800000000e+01 4.9900000000e+01 5.0000000000e+01  
## [772] 5.0100000000e+01 5.0200000000e+01 5.0300000000e+01  
## [775] 5.0400000000e+01 5.0500000000e+01 5.0600000000e+01  
## [778] 5.0700000000e+01 5.0800000000e+01 5.0900000000e+01  
## [781] 5.1000000000e+01 5.1100000000e+01 5.1200000000e+01  
## [784] 5.1300000000e+01 5.1400000000e+01 5.1500000000e+01  
## [787] 5.1600000000e+01 5.1700000000e+01 5.1800000000e+01  
## [790] 5.1900000000e+01 5.2000000000e+01 5.2100000000e+01  
## [793] 5.2200000000e+01 5.2300000000e+01 5.2400000000e+01  
## [796] 5.2500000000e+01 5.2600000000e+01 5.2700000000e+01  
## [799] 5.2800000000e+01

mu <- 0  
sigma <- 1  
  
# Using likelihood function to calculate L as requested  
L\_3 <- (1/(sigma \* sqrt(2 \* pi))) \* exp(-((x3-mu)^2)/(2\*sigma^2))  
L\_3

## [1] 1.99788925917e-159 2.95797868936e-158 4.33586473072e-157  
## [4] 6.29235858293e-156 9.04082906449e-155 1.28605667407e-153  
## [7] 1.81121085431e-152 2.52542788783e-151 3.48624566291e-150  
## [10] 4.76472738214e-149 6.44725997140e-148 8.63712814485e-147  
## [13] 1.14556727492e-145 1.50428072134e-144 1.95566397502e-143  
## [16] 2.51719370519e-142 3.20771735221e-141 4.04699441542e-140  
## [19] 5.05505833754e-139 6.25139288159e-138 7.65392973642e-137  
## [22] 9.27788966349e-136 1.11345069563e-134 1.32296950503e-133  
## [25] 1.55627281600e-132 1.81250279355e-131 2.08991533591e-130  
## [28] 2.38580946010e-129 2.69649658687e-128 3.01731775770e-127  
## [31] 3.34271444179e-126 3.66635540525e-125 3.98131830975e-124  
## [34] 4.28032056461e-123 4.55598982411e-122 4.80116078591e-121  
## [37] 5.00918198882e-120 5.17421446324e-119 5.29150360108e-118  
## [40] 5.35760660852e-117 5.37056036502e-116 5.32997826928e-115  
## [43] 5.23706940547e-114 5.09457870396e-113 4.90665222880e-112  
## [46] 4.67863681725e-111 4.41682757926e-110 4.12817988138e-109  
## [49] 3.82000415438e-108 3.49966208663e-107 3.17428155282e-106  
## [52] 2.85050516990e-105 2.53428396652e-104 2.23072366255e-103  
## [55] 1.94398687056e-102 1.67725052285e-101 1.43271431315e-100  
## [58] 1.21165315773e-99 1.01450476809e-98 8.40982428520e-98  
## [61] 6.90202942013e-97 5.60820326279e-96 4.51157041955e-95  
## [64] 3.59326122097e-94 2.83339351514e-93 2.21198438021e-92  
## [67] 1.70967779580e-91 1.30828855468e-90 9.91173923786e-90  
## [70] 7.43452538968e-89 5.52094836216e-88 4.05911332651e-87  
## [73] 2.95464782466e-86 2.12930230833e-85 1.51923858480e-84  
## [76] 1.07317783407e-83 7.50541068645e-83 5.19677942467e-82  
## [79] 3.56246955400e-81 2.41782628292e-80 1.62463603677e-79  
## [82] 1.08079698745e-78 7.11851203924e-78 4.64185294138e-77  
## [85] 2.99675049817e-76 1.91543249167e-75 1.21210479491e-74  
## [88] 7.59399913842e-74 4.71040200309e-73 2.89269375274e-72  
## [91] 1.75874954259e-71 1.05867484136e-70 6.30925735560e-70  
## [94] 3.72263921594e-69 2.17460663430e-68 1.25767238288e-67  
## [97] 7.20130815272e-67 4.08236960107e-66 2.29123852370e-65  
## [100] 1.27316689976e-64 7.00418213432e-64 3.81493003615e-63  
## [103] 2.05718230303e-62 1.09828749002e-61 5.80518880653e-61  
## [106] 3.03790169879e-60 1.57393987978e-59 8.07345850316e-59  
## [109] 4.10004053583e-58 2.06145442958e-57 1.02616307279e-56  
## [112] 5.05726930437e-56 2.46758905157e-55 1.19202851278e-54  
## [115] 5.70108489094e-54 2.69951302459e-53 1.26552404660e-52  
## [118] 5.87370906628e-52 2.69905364439e-51 1.22791316723e-50  
## [121] 5.53070954984e-50 2.46632952588e-49 1.08887595533e-48  
## [124] 4.75951575302e-48 2.05970102241e-47 8.82475497459e-47  
## [127] 3.74333057988e-46 1.57206595861e-45 6.53642677532e-45  
## [130] 2.69071123564e-44 1.09660655939e-43 4.42477958332e-43  
## [133] 1.76762241025e-42 6.99108224971e-42 2.73751419236e-41  
## [136] 1.06126881392e-40 4.07334767753e-40 1.54787046630e-39  
## [139] 5.82337559974e-39 2.16906240026e-38 7.99882775701e-38  
## [142] 2.92036879387e-37 1.05561635025e-36 3.77773572115e-36  
## [145] 1.33848679925e-35 4.69519535797e-35 1.63061073484e-34  
## [148] 5.60665692630e-34 1.90859913464e-33 6.43254033464e-33  
## [151] 2.14638373566e-32 7.09070266843e-32 2.31914677726e-31  
## [154] 7.50972877250e-31 2.40756113184e-30 7.64165541159e-30  
## [157] 2.40134540001e-29 7.47100227588e-29 2.30123070885e-28  
## [160] 7.01775994266e-28 2.11881925351e-27 6.33353782183e-27  
## [163] 1.87437240234e-26 5.49189783182e-26 1.59311113270e-25  
## [166] 4.57537559052e-25 1.30096161992e-24 3.66234516856e-24  
## [169] 1.02073055943e-23 2.81656654428e-23 7.69459862671e-23  
## [172] 2.08117682020e-22 5.57300002272e-22 1.47749549270e-21  
## [175] 3.87811193175e-21 1.00779353943e-20 2.59286470110e-20  
## [178] 6.60457986074e-20 1.66558803238e-19 4.15859897911e-19  
## [181] 1.02797735717e-18 2.51580577695e-18 6.09575812956e-18  
## [184] 1.46229635750e-17 3.47296274857e-17 8.16623563167e-17  
## [187] 1.90108153791e-16 4.38163943551e-16 9.99837874850e-16  
## [190] 2.25880940315e-15 5.05227108354e-15 1.11879562144e-14  
## [193] 2.45285528570e-14 5.32414837225e-14 1.14415649018e-13  
## [196] 2.43432053303e-13 5.12775363680e-13 1.06938378715e-12  
## [199] 2.20798996314e-12 4.51354367721e-12 9.13472040836e-12  
## [202] 1.83033221702e-11 3.63096150179e-11 7.13132812400e-11  
## [205] 1.38667999417e-10 2.66955661476e-10 5.08814028164e-10  
## [208] 9.60143337031e-10 1.79378390796e-09 3.31788424355e-09  
## [211] 6.07588284982e-09 1.10157636247e-08 1.97731964062e-08  
## [214] 3.51395509482e-08 6.18262050017e-08 1.07697600425e-07  
## [217] 1.85736184456e-07 3.17134921672e-07 5.36103534470e-07  
## [220] 8.97243516238e-07 1.48671951473e-06 2.43896074589e-06  
## [223] 3.96129909103e-06 6.36982517887e-06 1.01408520655e-05  
## [226] 1.59837411069e-05 2.49424712901e-05 3.85351967421e-05  
## [229] 5.89430677565e-05 8.92616571771e-05 1.33830225765e-04  
## [232] 1.98655471393e-04 2.91946925791e-04 4.24780270551e-04  
## [235] 6.11901930114e-04 8.72682695046e-04 1.23221916847e-03  
## [238] 1.72256893905e-03 2.38408820146e-03 3.26681905620e-03  
## [241] 4.43184841194e-03 5.95253241978e-03 7.91545158298e-03  
## [244] 1.04209348144e-02 1.35829692337e-02 1.75283004936e-02  
## [247] 2.23945302948e-02 2.83270377416e-02 3.54745928462e-02  
## [250] 4.39835959804e-02 5.39909665132e-02 6.56158147747e-02  
## [253] 7.89501583009e-02 9.40490773769e-02 1.10920834679e-01  
## [256] 1.29517595666e-01 1.49727465636e-01 1.71368592048e-01  
## [259] 1.94186054983e-01 2.17852177033e-01 2.41970724519e-01  
## [262] 2.66085249899e-01 2.89691552761e-01 3.12253933367e-01  
## [265] 3.33224602892e-01 3.52065326764e-01 3.68270140303e-01  
## [268] 3.81387815461e-01 3.91042693975e-01 3.96952547477e-01  
## [271] 3.98942280401e-01 3.96952547477e-01 3.91042693975e-01  
## [274] 3.81387815461e-01 3.68270140303e-01 3.52065326764e-01  
## [277] 3.33224602892e-01 3.12253933367e-01 2.89691552761e-01  
## [280] 2.66085249899e-01 2.41970724519e-01 2.17852177033e-01  
## [283] 1.94186054983e-01 1.71368592048e-01 1.49727465636e-01  
## [286] 1.29517595666e-01 1.10920834679e-01 9.40490773769e-02  
## [289] 7.89501583009e-02 6.56158147747e-02 5.39909665132e-02  
## [292] 4.39835959804e-02 3.54745928462e-02 2.83270377416e-02  
## [295] 2.23945302948e-02 1.75283004936e-02 1.35829692337e-02  
## [298] 1.04209348144e-02 7.91545158298e-03 5.95253241978e-03  
## [301] 4.43184841194e-03 3.26681905620e-03 2.38408820146e-03  
## [304] 1.72256893905e-03 1.23221916847e-03 8.72682695046e-04  
## [307] 6.11901930114e-04 4.24780270551e-04 2.91946925791e-04  
## [310] 1.98655471393e-04 1.33830225765e-04 8.92616571771e-05  
## [313] 5.89430677565e-05 3.85351967421e-05 2.49424712901e-05  
## [316] 1.59837411069e-05 1.01408520655e-05 6.36982517887e-06  
## [319] 3.96129909103e-06 2.43896074589e-06 1.48671951473e-06  
## [322] 8.97243516238e-07 5.36103534470e-07 3.17134921672e-07  
## [325] 1.85736184456e-07 1.07697600425e-07 6.18262050017e-08  
## [328] 3.51395509482e-08 1.97731964062e-08 1.10157636247e-08  
## [331] 6.07588284982e-09 3.31788424355e-09 1.79378390796e-09  
## [334] 9.60143337031e-10 5.08814028165e-10 2.66955661476e-10  
## [337] 1.38667999417e-10 7.13132812400e-11 3.63096150179e-11  
## [340] 1.83033221702e-11 9.13472040837e-12 4.51354367721e-12  
## [343] 2.20798996314e-12 1.06938378715e-12 5.12775363680e-13  
## [346] 2.43432053303e-13 1.14415649018e-13 5.32414837225e-14  
## [349] 2.45285528570e-14 1.11879562144e-14 5.05227108354e-15  
## [352] 2.25880940315e-15 9.99837874850e-16 4.38163943551e-16  
## [355] 1.90108153791e-16 8.16623563167e-17 3.47296274857e-17  
## [358] 1.46229635750e-17 6.09575812956e-18 2.51580577695e-18  
## [361] 1.02797735717e-18 4.15859897912e-19 1.66558803238e-19  
## [364] 6.60457986074e-20 2.59286470110e-20 1.00779353943e-20  
## [367] 3.87811193175e-21 1.47749549270e-21 5.57300002272e-22  
## [370] 2.08117682020e-22 7.69459862671e-23 2.81656654428e-23  
## [373] 1.02073055943e-23 3.66234516856e-24 1.30096161992e-24  
## [376] 4.57537559052e-25 1.59311113270e-25 5.49189783182e-26  
## [379] 1.87437240234e-26 6.33353782183e-27 2.11881925351e-27  
## [382] 7.01775994266e-28 2.30123070885e-28 7.47100227588e-29  
## [385] 2.40134540001e-29 7.64165541159e-30 2.40756113184e-30  
## [388] 7.50972877250e-31 2.31914677726e-31 7.09070266843e-32  
## [391] 2.14638373566e-32 6.43254033464e-33 1.90859913464e-33  
## [394] 5.60665692630e-34 1.63061073484e-34 4.69519535798e-35  
## [397] 1.33848679925e-35 3.77773572115e-36 1.05561635025e-36  
## [400] 2.92036879387e-37 7.99882775701e-38 2.16906240026e-38  
## [403] 5.82337559974e-39 1.54787046630e-39 4.07334767753e-40  
## [406] 1.06126881392e-40 2.73751419236e-41 6.99108224971e-42  
## [409] 1.76762241025e-42 4.42477958332e-43 1.09660655939e-43  
## [412] 2.69071123564e-44 6.53642677532e-45 1.57206595861e-45  
## [415] 3.74333057989e-46 8.82475497460e-47 2.05970102241e-47  
## [418] 4.75951575302e-48 1.08887595533e-48 2.46632952588e-49  
## [421] 5.53070954984e-50 1.22791316723e-50 2.69905364439e-51  
## [424] 5.87370906628e-52 1.26552404660e-52 2.69951302459e-53  
## [427] 5.70108489094e-54 1.19202851278e-54 2.46758905157e-55  
## [430] 5.05726930438e-56 1.02616307279e-56 2.06145442959e-57  
## [433] 4.10004053583e-58 8.07345850316e-59 1.57393987978e-59  
## [436] 3.03790169879e-60 5.80518880653e-61 1.09828749002e-61  
## [439] 2.05718230303e-62 3.81493003615e-63 7.00418213432e-64  
## [442] 1.27316689976e-64 2.29123852370e-65 4.08236960107e-66  
## [445] 7.20130815272e-67 1.25767238288e-67 2.17460663430e-68  
## [448] 3.72263921595e-69 6.30925735560e-70 1.05867484136e-70  
## [451] 1.75874954260e-71 2.89269375274e-72 4.71040200309e-73  
## [454] 7.59399913842e-74 1.21210479491e-74 1.91543249167e-75  
## [457] 2.99675049817e-76 4.64185294138e-77 7.11851203924e-78  
## [460] 1.08079698745e-78 1.62463603677e-79 2.41782628292e-80  
## [463] 3.56246955400e-81 5.19677942467e-82 7.50541068646e-83  
## [466] 1.07317783407e-83 1.51923858480e-84 2.12930230833e-85  
## [469] 2.95464782466e-86 4.05911332651e-87 5.52094836216e-88  
## [472] 7.43452538968e-89 9.91173923787e-90 1.30828855468e-90  
## [475] 1.70967779580e-91 2.21198438021e-92 2.83339351514e-93  
## [478] 3.59326122097e-94 4.51157041955e-95 5.60820326279e-96  
## [481] 6.90202942013e-97 8.40982428521e-98 1.01450476809e-98  
## [484] 1.21165315774e-99 1.43271431315e-100 1.67725052285e-101  
## [487] 1.94398687056e-102 2.23072366255e-103 2.53428396652e-104  
## [490] 2.85050516990e-105 3.17428155283e-106 3.49966208663e-107  
## [493] 3.82000415438e-108 4.12817988138e-109 4.41682757926e-110  
## [496] 4.67863681725e-111 4.90665222880e-112 5.09457870396e-113  
## [499] 5.23706940547e-114 5.32997826928e-115 5.37056036502e-116  
## [502] 5.35760660852e-117 5.29150360108e-118 5.17421446324e-119  
## [505] 5.00918198882e-120 4.80116078591e-121 4.55598982411e-122  
## [508] 4.28032056461e-123 3.98131830975e-124 3.66635540526e-125  
## [511] 3.34271444180e-126 3.01731775771e-127 2.69649658688e-128  
## [514] 2.38580946010e-129 2.08991533591e-130 1.81250279355e-131  
## [517] 1.55627281600e-132 1.32296950503e-133 1.11345069563e-134  
## [520] 9.27788966349e-136 7.65392973642e-137 6.25139288160e-138  
## [523] 5.05505833754e-139 4.04699441542e-140 3.20771735221e-141  
## [526] 2.51719370519e-142 1.95566397502e-143 1.50428072134e-144  
## [529] 1.14556727492e-145 8.63712814485e-147 6.44725997140e-148  
## [532] 4.76472738215e-149 3.48624566291e-150 2.52542788783e-151  
## [535] 1.81121085431e-152 1.28605667407e-153 9.04082906449e-155  
## [538] 6.29235858293e-156 4.33586473072e-157 2.95797868936e-158  
## [541] 1.99788925917e-159 1.33599501770e-160 8.84494873647e-162  
## [544] 5.79752802911e-163 3.76224825106e-164 2.41718057862e-165  
## [547] 1.53754486968e-166 9.68285784653e-168 6.03721119591e-169  
## [550] 3.72671547411e-170 2.27757747874e-171 1.37808859334e-172  
## [553] 8.25540095828e-174 4.89616742424e-175 2.87495733387e-176  
## [556] 1.67133536513e-177 9.61950793085e-179 5.48149681849e-180  
## [559] 3.09244889201e-181 1.72728082016e-182 9.55169454195e-184  
## [562] 5.22943724367e-185 2.83456578641e-186 1.52116099634e-187  
## [565] 8.08203779163e-189 4.25131835353e-190 2.21402965534e-191  
## [568] 1.14156407952e-192 5.82739170107e-194 2.94513483558e-195  
## [571] 1.47364613488e-196 7.30025938428e-198 3.58047320774e-199  
## [574] 1.73859978084e-200 8.35825983392e-202 3.97822342728e-203  
## [577] 1.87464711726e-204 8.74594901602e-206 4.03972134895e-207  
## [580] 1.84736594235e-208 8.36395160586e-210 3.74910103986e-211  
## [583] 1.66379507294e-212 7.31020536827e-214 3.17992133961e-215  
## [586] 1.36949430808e-216 5.83930474651e-218 2.46501238665e-219  
## [589] 1.03022984258e-220 4.26291036578e-222 1.74636625676e-223  
## [592] 7.08306972316e-225 2.84423029229e-226 1.13074598253e-227  
## [595] 4.45063936075e-229 1.73435026361e-230 6.69126564877e-232  
## [598] 2.55585839614e-233 9.66545627354e-235 3.61880338076e-236  
## [601] 1.34141966735e-237 4.92290445336e-239 1.78869049550e-240  
## [604] 6.43437023934e-242 2.29157433605e-243 8.08014041387e-245  
## [607] 2.82072612610e-246 9.74899818349e-248 3.33592373263e-249  
## [610] 1.13013231106e-250 3.79052640009e-252 1.25871328184e-253  
## [613] 4.13819697743e-255 1.34695336207e-256 4.34061246559e-258  
## [616] 1.38486207834e-259 4.37440599597e-261 1.36800822449e-262  
## [619] 4.23560433154e-264 1.29837191981e-265 3.94039627714e-267  
## [622] 1.18396193825e-268 3.52202664443e-270 1.03730050281e-271  
## [625] 3.02463932863e-273 8.73171710062e-275 2.49564480116e-276  
## [628] 7.06192247120e-278 1.97842764815e-279 5.48749901072e-281  
## [631] 1.50690471762e-282 4.09688888488e-284 1.10275651738e-285  
## [634] 2.93874657699e-287 7.75356968685e-289 2.02534171621e-290  
## [637] 5.23783696964e-292 1.34110474927e-293 3.39962081376e-295  
## [640] 8.53208775564e-297 2.12000655153e-298 5.21526219883e-300  
## [643] 1.27020011383e-301 3.06284629070e-303 7.31198534550e-305  
## [646] 1.72823373228e-306 4.04414480935e-308 9.36931787664e-310  
## [649] 2.14904893390e-311 4.88024573402e-313 1.09722105199e-314  
## [652] 2.44232597178e-316 5.38232644251e-318 1.17434463360e-319  
## [655] 2.53455676317e-321 5.43472210425e-323 0.00000000000e+00  
## [658] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [661] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [664] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [667] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [670] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [673] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [676] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [679] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [682] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [685] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [688] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [691] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [694] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [697] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [700] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [703] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [706] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [709] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [712] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [715] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [718] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [721] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [724] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [727] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [730] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [733] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [736] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [739] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [742] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [745] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [748] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [751] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [754] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [757] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [760] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [763] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [766] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [769] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [772] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [775] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [778] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [781] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [784] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [787] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [790] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [793] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [796] 0.00000000000e+00 0.00000000000e+00 0.00000000000e+00  
## [799] 0.00000000000e+00

# Plotting L vs x  
plot(x3,L\_3)



# Sum of x3 sequence multiplied by tenth.increment  
sum(x3)\*tenth.increment

## [1] 1030.71

# Commentary: The top of both curves in part a and part b peak at 0 point on the x-axis

### Unit Test (R)

## [1] 4

## [1] 6

# Exercise 3 (R or SAS, optional)

Suppose we wish to determine the linear relationship be Calories per Serving and Calories per Recipe. We can determine this by solving a system of linear equations.

We would write this in matrix notation as

We would generally write this as , and find a solution by computing

.

However, the simplest function for the inverse, require square matrices, so commonly we use the *normal* equations,

(where is the transpose of ). We then find

Define appropriate X and y matrices (y can be a vector in R). Multiply the transpose of X by X, then use solve (R) or inv (IML) to find the inverse. Multiply this by the product of transpose X and y to find hat.beta.

Print your hat.beta.

# Moving Calories Per Serving data into matrix  
CPRM <- matrix(CaloriesPerServingMean,nrow = 7,ncol = 1)  
  
# Building matrix X for linear equation operation  
X <- matrix(0,nrow = 7,ncol = 2)  
X[1,1] <- 1  
X[2,1] <- 1  
X[3,1] <- 1  
X[4,1] <- 1  
X[5,1] <- 1  
X[6,1] <- 1  
X[7,1] <- 1  
X[1,2] <- CPRM[1,1]  
X[2,2] <- CPRM[2,1]  
X[3,2] <- CPRM[3,1]  
X[4,2] <- CPRM[4,1]  
X[5,2] <- CPRM[5,1]  
X[6,2] <- CPRM[6,1]  
X[7,2] <- CPRM[7,1]  
  
# Transposed matrix X  
X\_t <- t(X)  
  
# Loading CaloriesPerRecipe data into matrix  
y <- matrix(CaloriesPerRecipeMean,nrow = 7, ncol = 1)  
  
# multiply X\_t by x  
X\_m <- X\_t %\*% X  
  
#Multiply X\_t by y  
y\_m <- X\_t %\*% y  
  
# Inverse of matrix X  
X\_i <- solve(X\_m)  
  
# Calculating hat.beta for comparison with next section  
hat.beta <- X\_i %\*% y\_m  
hat.beta

## [,1]  
## [1,] -166.92279403500  
## [2,] 8.33903711693

Compare your hat.beta to

summary(lm(CaloriesPerRecipeMean~CaloriesPerServingMean))

##   
## Call:  
## lm(formula = CaloriesPerRecipeMean ~ CaloriesPerServingMean)  
##   
## Residuals:  
## 1 2 3 4 5   
## 55.02694299 28.50983163 -85.61273211 -40.59144432 19.49379344   
## 6 7   
## 9.87668209 13.29692629   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -166.922794035 158.101389707 -1.05580 0.33938  
## CaloriesPerServingMean 8.339037117 0.529639883 15.74473 1.8795e-05  
##   
## (Intercept)   
## CaloriesPerServingMean \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 51.9082754 on 5 degrees of freedom  
## Multiple R-squared: 0.980229069, Adjusted R-squared: 0.976274882   
## F-statistic: 247.896532 on 1 and 5 DF, p-value: 1.87948158e-05

If your estimate for does not match the R result, you can sill get partial credit by printing your , , and .

# Exercise 4

The probability mass function for value from Poisson data with a mean of is given by

### Part a

Create a sequence of given the mean and standard deviation for Servings per Recipe 1936, and calculate the , given for Servings per Recipe, 1936.

# mu and sigma variables for use with sequence and function below   
mu <- ServingsPerRecipeMean[idx.1936]  
sigma <- ServingsPerRecipeSD[idx.1936]  
mu

## [1] 12.9

sigma

## [1] 13.3

# Creating requested sequence and assigning to y  
y <- seq(from=floor(mu-sigma),to=ceiling(mu+sigma))  
y

## [1] -1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21  
## [24] 22 23 24 25 26 27

# Function to calculate f(y,mu) for Servings Per Recipe (SPR), 1936  
Poissonfunc <- function(y,mu) {   
Poissonvar <- (exp(-mu))\*(1/factorial(y))\*(exp(y))\*(log(mu))  
return(Poissonvar)  
}   
  
Poissonfunc(y,mu)

## Warning in gamma(x + 1): NaNs produced

## [1] NaN 6.38808251848e-06 1.73646086287e-05  
## [4] 2.36009500468e-05 2.13846778822e-05 1.45323953237e-05  
## [7] 7.90062922646e-06 3.57935614328e-06 1.38995696598e-06  
## [10] 4.72286845370e-07 1.42645416621e-07 3.87750443914e-08  
## [13] 9.58195441518e-09 2.17053771399e-09 4.53856401994e-10  
## [16] 8.81221150194e-11 1.59693829295e-11 2.71308021431e-12  
## [19] 4.33818626218e-13 6.55134049165e-14 9.37283674236e-15  
## [22] 1.27390058989e-15 1.64896229751e-16 2.03742920415e-17  
## [25] 2.40795947062e-18 2.72729686360e-19 2.96542460206e-20  
## [28] 3.10033069594e-21 3.12132318259e-22

# NaN value was generated and may affect future calculations if this sequence is used again.

### Part b

Next, compute the likelihood for each using your likelihood function from Exercise 2, with the same mean and standard deviation as in Part a of this exercise.

Plot vs from Part a, then vs from this part. Use different colors, symbols or line types (or combination of these) for the two curves. (If you do this in SAS, research the group option).

Comment on the differences and similarities of the curves, noting in your comments the curve type (color, symbol, etc).

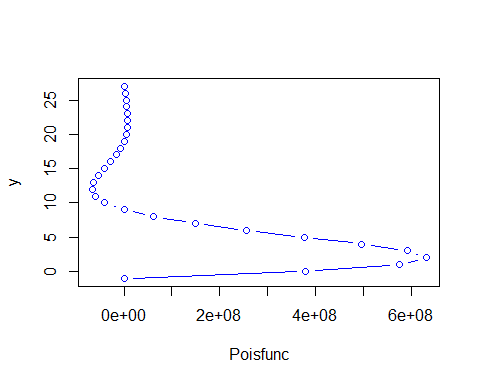
mu <- ServingsPerRecipeMean[idx.1936]  
sigma <- ServingsPerRecipeSD[idx.1936]  
  
# x3 from Exercise 2, part a as x  
x <- seq(from=(mu-3\*sigma),to=(mu+3\*sigma),by=tenth.increment)  
  
#Calculating likelihood of y  
yL <- (1/(sigma \* sqrt(2 \* pi))) \* exp(-((y-mu)^2)/(2\*sigma^2))  
yL

## [1] 0.0173730880837 0.0187402879094 0.0201011233636 0.0214392324060  
## [5] 0.0227375134220 0.0239784738239 0.0251446120947 0.0262188220277  
## [9] 0.0271848063757 0.0280274860842 0.0287333908331 0.0292910167938  
## [13] 0.0296911383406 0.0299270619222 0.0299948123306 0.0298932441260  
## [17] 0.0296240738520 0.0291918317763 0.0286037350512 0.0278694872587  
## [21] 0.0270010121170 0.0260121315609 0.0249182003310 0.0237357105508  
## [25] 0.0224818804740 0.0211742416364 0.0198302380752 0.0184668501308  
## [29] 0.0171002537187

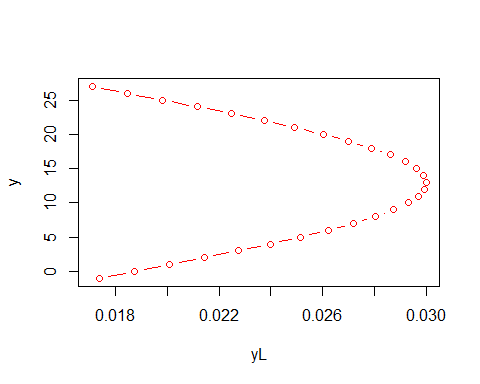
xPlot <- Poissonfunc(x,mu)

## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'

Poisfunc <- xPlot[1:29]  
  
#f(x,mu) vs y plot  
plot(Poisfunc,y,col="Blue",type = "b")



# L(y,mu,sigma) vs y plot  
plot(yL,y,col= "Red",type = "b")



# Commentary: The bottom half of the curve of the first plot resembles the curve of the second plot.  
# Also, the curve in the second plot covers whole range of y, while the comparable curve portion in the first plot  
# only covers approximately half that range.

### Part c

Calculate and print the sum of each sequence and , multiplied by the increment.

sum(Poissonfunc(x,mu))\*tenth.increment

## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'  
  
## Warning in factorial(y): full precision may not have been achieved in  
## 'gammafn'

## [1] 317885110.393

sum(yL)\*tenth.increment

## [1] 0.072448575647

# Exercise 5

Given a vector of mean estimates , a vector of standard deviations and a vector of sample sizes , we can calculate a one-way analysis of variance by

and

where is the mean of and . The test statistic is which is distributed as

### Part a

Calculate MSW and MSB for Calories per Serving from Wansink Table 1. You can use the variables CaloriesPerServingMean and CaloriesPerServingSD. Use array functions and arithmetic for your calculations (hint - use pairwise operations). Do not hard code values for and , calculate these from the appropriate variables.

Print both MSB and MSW.

### Part b

Calculate an F-ratio and a for this , using the distribution with and degrees of freedom. Use . Compare these values to the corresponding values reported in Wansink Table 1.

You can check results by entering appropriate values in an online calculator like <http://statpages.info/anova1sm.html> .

# Exercise 6 (R or SAS)

You will refer to the matrices from Exercise 1 in this exercise. If you want to use IML for this exercise, put this exercise in the same PROC IML block.

### Part a

Compute the absolute value of the differences between pairs of treatments after subtracting and . Name this matrix TreatmentDifferences and print this matrix, then print the maximum value of this matrix (using builtin funtions).

### Part b

Use the fisher.lsd function to compute the LSD for each pair of treatments. Pass as parameters S\_1 and S\_2 from Exercise 1. Name this matrix PairwiseLSD. Print this matrix, then perform a comparison between TreatmentDifferences and PairwiseLSD to determine which treatment pairs are significantly different. Compute and print the number of pairs that are significantly different.

### Part c

Now, modify your Tukey HSD function from Homework 3 to accept three and only three paramaters - a vector of standard deviations , a vector of sample sizes and an optional .

Inside your modifed HSD, use the formula for from Exercise 5 to compute a pooled . Use as the degrees of freedom and determine the number of comparisons from the data. For the divisor of the standard error term, use the harmonic mean of , given by

Call your new HSD function with the standard deviations used in Exercise 1 to compute a single HSD value; name this value PairwiseHSD and print this value. You will need to create a vector of .

Repeating the steps from Part b, compare this value to TreatmentDifferences and compute the number of treatment means that are signficantly different. Comment on the number of different treatment pairs found in Part b and and this part.

# Total points from unit tests

unit.test.points

## [1] 6