CompStatsLab1

Omkar Bhutra

29 February 2019

Question 1:

Be careful when comparing

```
## [1] "subtraction is wrong"
```

Due to underflow the subtraction is displaying the same number although when the digits are increased using options we can see that the number is actually different. Underflow is the loss of significant digits.

```
## [1] "subtraction is correct"
```

```
## [1] "subtraction is correct"
```

Evaluating the results of the 2 snippets we see that in the first occasion we get the wrong print of the if-else statement.The problem lies to the fact that float numbers that have infinite numbers of decimals can't be represented exactly in the binary system in computers due to memory storage limitation.Using print(x1-x2,digits=16) and print(1/12,digits=16) we will see that the resulting floats are (0.08333333333333315,0.08333333333333329) respectfully and they are not the same causing the condition of unerflow which leads to the failure of the if statement and evaluation of else. We can adress this problem using the "all.equals()" in the if statement instead of "==" to compare the numbers and we will see that the if statement will be executed and the correctly print message will be outputed. The second statement is evaluated correctly and we get the correct print output because 1/2 has finite numbers of decimals so we don't have the occurence of underflow here.

Question 2:

Derivative

```
## [1] 1.1102230246251565
```

```
## [1] 0
```

The value of the derivative for when x=1 is 1.11022 while the value obtained for the derivative when x=100000 is 0. Looking at the equation algebraically it seems that the answer should be 0. But in the first case, when x=1 the addition of a small value epsilon retains its effect compared to the 2nd case and hence produces a different result than the expected value of 0.

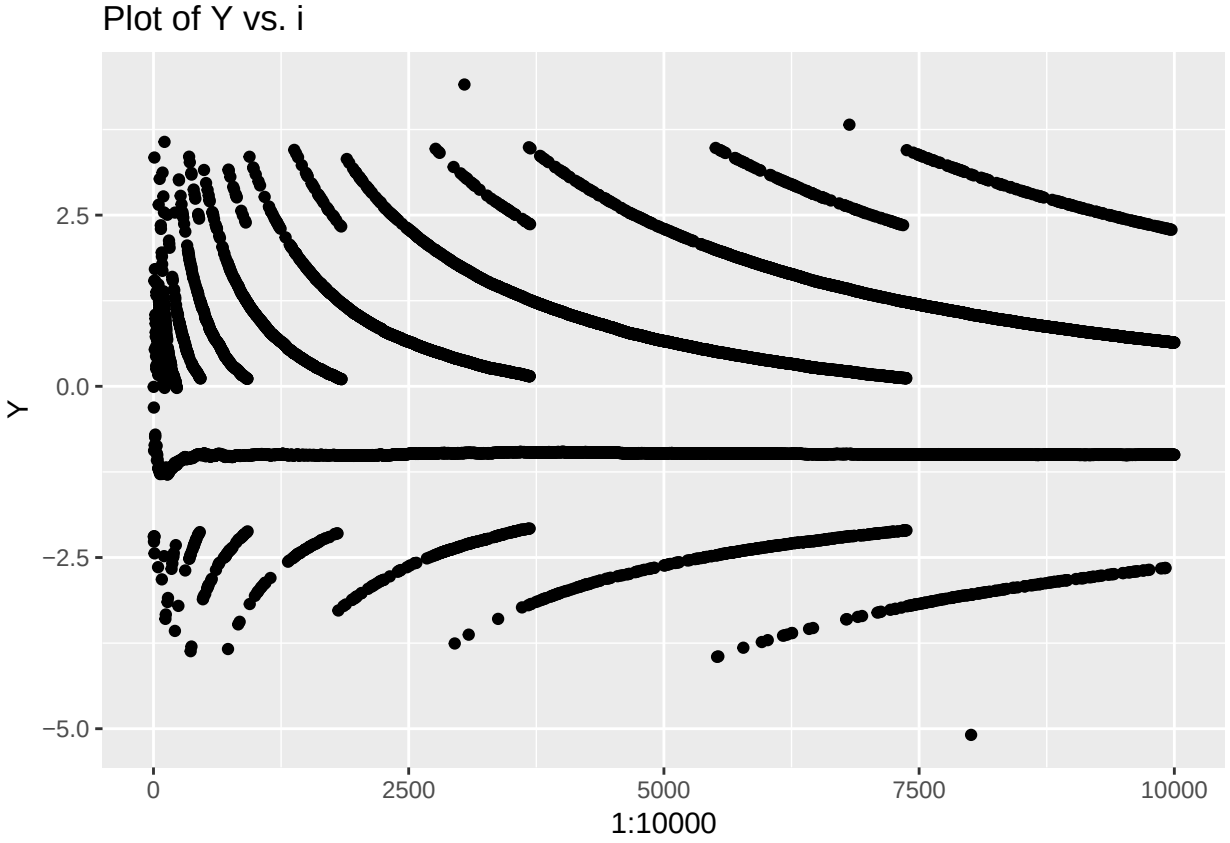
The true value for the function using the function $f(x) = x$ is $f'(x) = \frac{f(x+\epsilon)-f(x)}{\epsilon} = \frac{(x+\epsilon)-x}{\epsilon} = 1$ is always constant with value 1.Regarding the result of the derivative function we see that for $x = 100000$ R doesn't take into account the decimals after a specific number of x and rounds the number to the nearest integer which is 100000 due to underflow occurance so the numeretor of the derivative formula becomes 0 leading finally to 0.When instead $x = 1$ the numerator evaluated is 1.1102230246251565e-15 and the devision with epsilon 10^-15 is just discards the last 15 decimals resulting 1.1102230246251565.

Question 3:

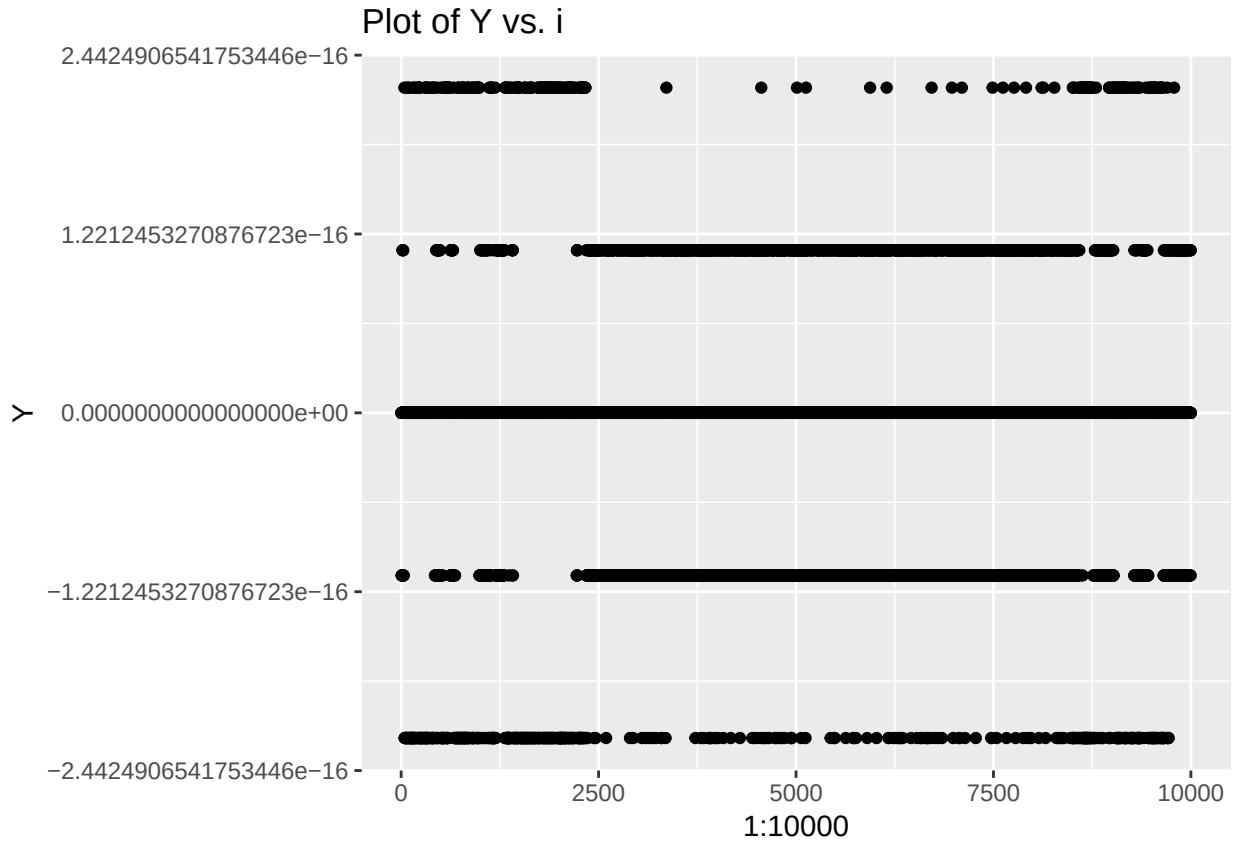
Variance

```
## [1] 1.6385638563856386
```

The plot above shows the dependence Y_i on i with the formula $Var(x) = \frac{\sum_{i=1}^{n}(x_i-\mu)^2}{n-1}$ where μ is the mean. Using the new formula where we center the points arround the mean we see that we have an improvement in the range of the errors and the deviation of the errors is steady and we can see an upper and a lower band with few errors lie beyond these linear bands represented with red in the plot.Also we can observe that the range of the errors is much smaller with means the formula used almost as good as the var() basic function in R.




```
## [1] 0.99969989923081803
```



The function does not perform as well as expected due to the numerical precision of the expression myvar(Xi)-var(Xi) which shows us negative values on most occurances.

Question 4:

Linear Algebra

```
## [1] 1157834236871692.2
```

Error in solve.default(A, b) : system is computationally singular: reciprocal condition number = 7.13971e-17

Printing the number of kappa for the value of A matrix we see that is very big and that implies that the matrix is said to be ill-conditioned a very small change in matrix A will cause a large error in b and makes the solution unstable.

```
##                              [,1]
## Channel1    -110.64200663177356887
## Channel2    -221.18999213628123357
## Channel3     378.00670489934344687
## Channel4    -129.70382900682116656
## Channel5     413.41802724878726849
## Channel6     -79.75199462292130193
## Channel7    -203.00600103836293897
## Channel8      82.79496481288938980
## Channel9    -132.38107625535101874
## Channel10    255.82331130982933587
```

```
## Channel11   -328.60850197387361504
## Channel12   -304.14980672129559025
## Channel13    624.12672247994066765
## Channel14   -298.89902984730480284
## Channel15     40.74336715520891516
## Channel16   -257.53594209754874100
## Channel17    169.23891908084701186
## Channel18    296.66293885907538197
## Channel19   -325.06601012322818178
## Channel20     -3.00774341513279264
## Channel21    554.56043514869395494
## Channel22  -1366.02819464972321839
## Channel23   1860.35645048260312251
## Channel24  -1416.13210988432069826
## Channel25    631.83966192630055048
## Channel26   -112.04395491771877857
## Channel27     17.01671931621323708
## Channel28   -228.93063252299120336
## Channel29    444.27242587767221949
## Channel30   -597.38053691993104621
## Channel31    438.14844153976412144
## Channel32    315.03940879430871291
## Channel33   -349.81437683657878779
## Channel34   -285.91097574921798241
## Channel35    418.58105049764253636
## Channel36    -79.10682360375271571
## Channel37   -305.94133876342527856
## Channel38    284.25434730174231390
## Channel39   -435.56578629077682763
## Channel40    819.74862823138187196
## Channel41   -885.00733142452941138
## Channel42    324.58934112061319865
## Channel43    524.58956696461768843
## Channel44   -583.44189030616257696
## Channel45   -140.17097449229814288
## Channel46    577.23617438118196787
## Channel47   -294.26844675559453890
## Channel48    -68.07512752921009280
## Channel49    -90.49228410711393167
## Channel50    404.14395235715244326
## Channel51   -698.99905269889643478
## Channel52   1258.88337827146801828
## Channel53  -1672.73084007547663532
## Channel54   1486.22991720257186898
## Channel55   -812.36134431546076939
## Channel56    192.49547990397724107
## Channel57    -32.91204662098924416
## Channel58      7.37525455331103430
## Channel59    -88.69071417608546426
## Channel60    344.87690207084136773
## Channel61   -454.35186074457743644
## Channel62    447.62052960726805395
## Channel63   -197.41868472109501909
## Channel64    222.33707920152659199
```

```
## Channel65   -399.25583177909453525
## Channel66    364.86655737276095124
## Channel67   -367.16148297452463112
## Channel68    243.92206215756519327
## Channel69    -76.29483445491032967
## Channel70   -318.19160711413701392
## Channel71    327.66533461201169075
## Channel72   -178.52315275400727046
## Channel73    119.18564986588171450
## Channel74    445.11500447600673169
## Channel75    -20.01273187169612910
## Channel76   -642.75099614710870810
## Channel77    369.48095078598106511
## Channel78    -74.90113656891863059
## Channel79    -23.48543216535520983
## Channel80   -676.86153344610886506
## Channel81   1013.45380290573928050
## Channel82   -889.76231887539347554
## Channel83    403.00656326222281223
## Channel84    424.08483028785201441
## Channel85   -801.09561546783777430
## Channel86    655.01342015748275571
## Channel87    659.18297852899979716
## Channel88  -2150.83256466095554060
## Channel89   1671.80888532636413402
## Channel90    298.69770682030031139
## Channel91   -332.17277624942408920
## Channel92   -487.36897587493234596
## Channel93    278.62773677083617940
## Channel94    201.66273526775202640
## Channel95   -609.50814557014871298
## Channel96    565.28517886262272896
## Channel97   -133.34075951392054549
## Channel98   -368.00872501373430623
## Channel99    238.20159678039942719
## Channel100    24.64181878308056639
## Fat           -1.66664028405493303
## Moisture      -0.93410994774249811
```

This happens because the tolerence returned is larger than the default threshold set by the function solve (argument tolerence) so an error returned and we cannot get a solution.The torrelance is related to conditon number by the function $tolerance = \frac{1}{conditionnumber}$ so in our case $tolerance = \frac{1}{kappa(A)} = 7.425326e-16$ and it is bigger that the threshold of $7.425326e-17$ that is set by solve function as we see in the printed error resulting the end of execution of the function. Using the scaled data we where able to solve the linear system and get coefficients for every feature value. Printing the number of kappa again we can see that is still high but much less that the previous used with the unscaled data and we where able to solve the linear system and get coefficient values.

When we scale the data we see that the linear system did not get any better or worse the linear dependences of the column features are still present but we manage to make the value of condition number smaller with scaling.This is happening because If we look at the definition of the condition number $k(A) = ||A|| * ||A^{-}1||$ and just by making the range of the columns smaller the magnitude got smaller leading to a smaller value of condition number which is below threshold value of solve function and we manage to get the solution.The tolerence now is $tolerance = \frac{1}{kappa(A1)} = \frac{1}{490471518993} = 2.038854e-12$ which is smaller than the default

$7.425326e-17$ set by solve so now we are able to get a solution.

Apendix

```
knitr::opts_chunk$set(echo = TRUE)
library(dplyr)
library(plotly)
library(ggplot2)
library(xlsx)
library(readxl)
library(boot)
library(kableExtra)
library(knitr)
library(testthat)
options(digits=22)
x1<-1/3;x2<-1/4
if(x1-x2==1/12){
  print("subtraction is correct")
}else{
  print("subtraction is wrong")
}
x1<-1/3;x2<-1/4
if(all.equal((x1-x2),(1/12))){
  print("subtraction is correct")
}else{
  print("subtraction is wrong")
}
x1<-1;x2<-1/2
if(x1-x2==1/2){
  print("subtraction is correct")
}else{
  print("subtraction is wrong")
}
derivative<-function(x){
  f<-function(x){
    return(x)
  }
  epsilion<-10^-15
  x<-(f(x+epsilion)-f(x))/epsilion
  return(x)
}

derivative(x=1)

derivative(x=100000)

set.seed(12345)
myvar<-function(x){
n=length(x)
var<-(1/(n-1))*(sum(x^2)-((1/n)*(sum(x)^2)))
return(var)
}
```

```r
x<-rnorm(n=10000,mean=10^8,sd=sqrt(1))
myvar(x)
Y <- c()
for (i in 1:10000){
options(digits = 22)
Y[i] <- myvar(x[1:i])-var(x[1:i])
}

p1<-ggplot()+ geom_point(aes(1:10000,Y))+ labs(title="Plot of Y vs. i")
p1
set.seed(12345)
varfun <- function(x){
vari <- (1/(length(x) - 1)) * sum((x - mean(x))^2)
return(vari)
}

x<-rnorm(n=10000,mean=10^8,sd=sqrt(1))
varfun(x)

Y <- c()
for (i in 1:10000){
Y[i] <- varfun(x[1:i])-var(x[1:i])
}

p2<-ggplot() + geom_point(aes(1:10000,Y))+ labs(title="Plot of Y vs. i")
p2
tecator = read_excel("tecator.xls",sheet ="data" )
X<-as.matrix(tecator[,c(2:102,104)])
Y<-as.matrix(tecator[,c(103)])
A<-t(X)%*%X
b<-t(X)%*%Y
kappa(A)
#solve(A,b)
tecatorscale <- as.matrix(scale(tecator))
Xscale <- as.matrix(tecatorscale[,-c(1,103)])
yscale <- as.matrix(tecatorscale[,103])
Ascale <- t(Xscale)%*%Xscale
bscale <- t(Xscale)%*%yscale
solve(Ascale,bscale)
```