Computational Statistics LAB1

GROUP 12

Question 1

q1: Check the result of the snippets. Comment what is going on

In the first test, we get $\frac{1}{3} - \frac{1}{4}! = \frac{1}{12}$. That's because computer can not store numbers like $\frac{1}{3}$ which have infinite digits. So, the actual number in the memory is just very closed to $\frac{1}{3}$. As for $\frac{1}{4}$, has the finite digits (within 32), computer can store it without losing accuracy. As a result, $\frac{1}{3} - \frac{1}{4}$ is not equal to $\frac{1}{12}$.

q2: do you have any problems, suggest improvement？

Suggest:

When doing the comparison, we can use a very small-scale number as a threshold. For example, we pick threshold $\alpha = 10^{-15}$. If $|(a - b) - c| < \alpha$, then we can assume that (a-b) equals c.

Question 2

q1: Write R function to calculate the derivative of $f(x) = x$ in this way with $ETA = 10^{-15}$.

The function can be seen in appendix.

q2: Evaluate your derivative function at $x = 1$ and $x = 100000$.

We get 1.110223 at x=1 and 0 at x=100000.

q3: What values did you obtain? What are the true values? Explain the reasons behind the discovered differences.

We get 1.110223 at x=1 and 0 at x=100000.

Assume that ETA = 10^{-15}. When doing this calculating, computer first calculates x+ETA and get an answer(Y), then Y-eta, finally, the division.

In this process, when x==1, x+ETA will lose some part of the mantissa, so Y is not 1+ETA but very closed. So, the result is biased. When x == 100000, because of the same reason, computer can not store any digits of the ETA part, so here Y == 100000, thus $\frac{Y-100000}{ETA}$ = 0.

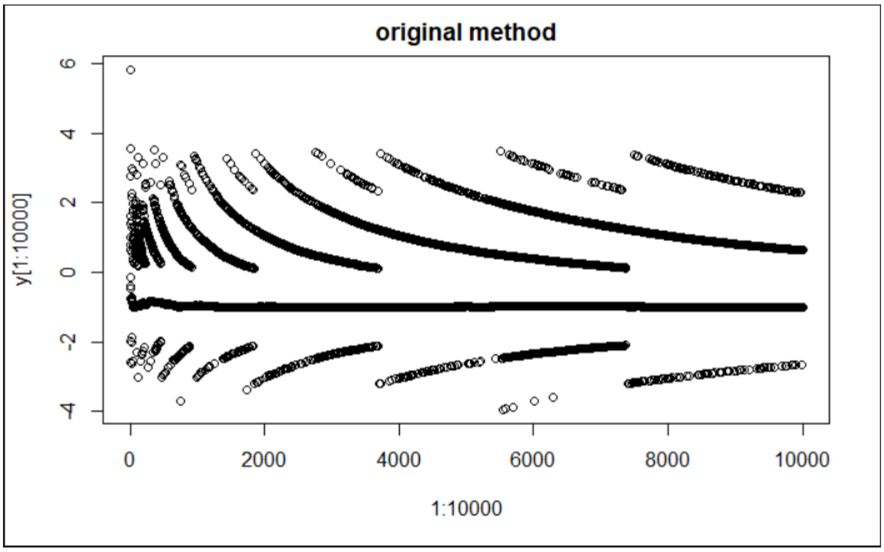
Question 3

q1,q2: Write your own R function, myvar, to estimate the variance in this way. Generate a vector $x = (x_1, \ldots, x_{10000})$ with 10000 random numbers with mean 10^8 and variance 1.

The code can be seen in the appendix part.

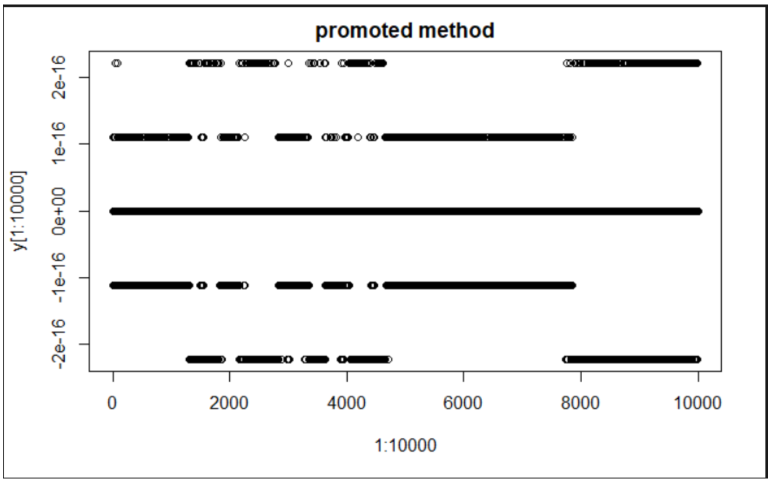
q3: How well does your function work? Can you explain the behaviour?

It doesn't work well. In the section sum(x)*sum(x), the integer overflow happens. So, the answer will lose accuracy. The graphic is below.



q4: How can you better implement a variance estimator? Find and implement a formula that will give the same results as var ()?

We can use another formula, $var(\vec{x}) = \frac{1}{n-1}(\sum_{i=0}^{n}(x_i - \bar{x})^2)$, the bias is very closed to 0.



Question 4

Consideration: which computes the product of all the elements of the vector passed to it?

When n and k are comparatively small, all these three methods will compute all the input vector. However, with n and k become greater and greater, method A will fail to compute first then method B will also lose its function. Finally, method C will also fail.

q1: Even if overflow and underflow would not occur these expressions will not work correctly for all values of n and k. Explain what the problem in A, B and C is respectively.

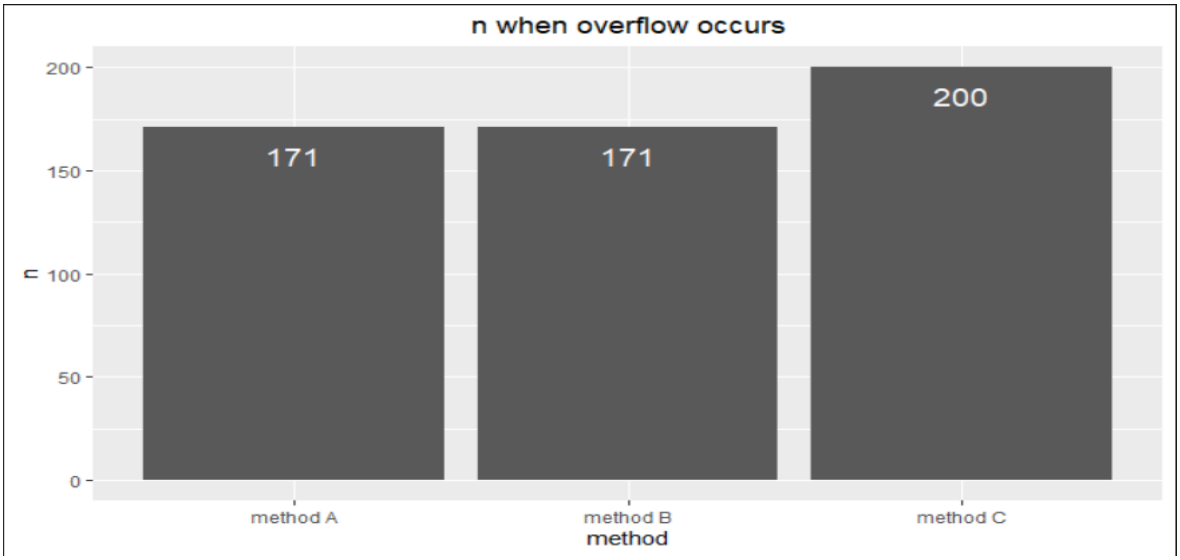
A: when k==0 or k==n, it will compute prod (1:0) and get a zero-divisor leading to an Inf result.

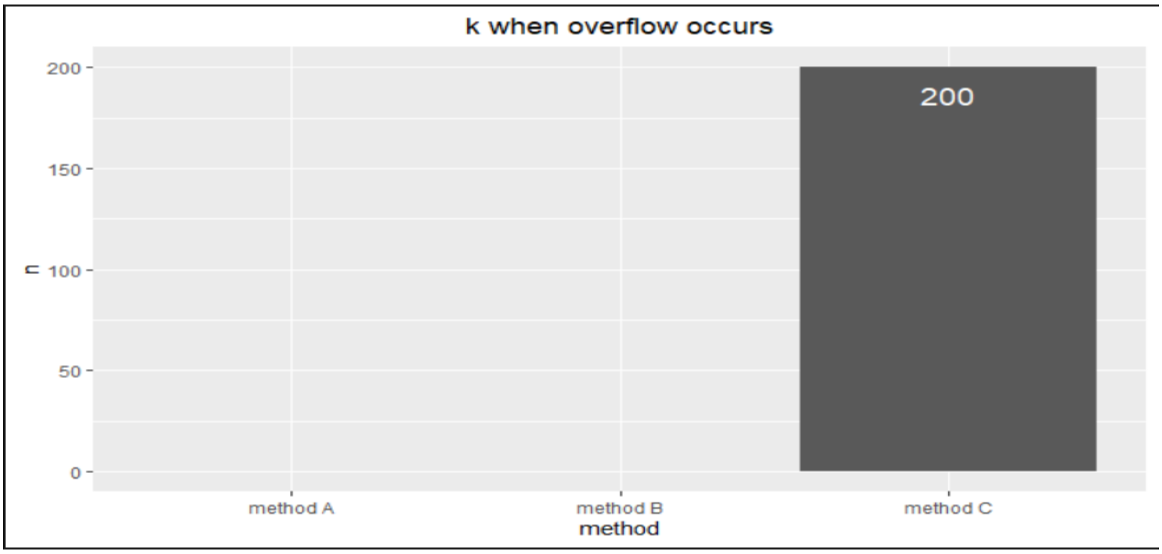
B: when n==k, it will compute prod (1:0) and get a zero-divisor leading to an Inf result.

C: when n==k, it will compute prod (1:0) and get a zero-divisor leading to an Inf result.

q2: In mathematical formula one should suspect overflow to occur when parameters, here n and k, are large. Experiment numerically with the code of A, B and C, for different values of n and k to see whether overflow occurs. Graphically present the results of your experiments.

(Hits: we assume that n grows from 1 and k grows from 0. In method A, zero divisor overflow will always occur when k==0, so for method A, we only consider n)





(The numbers of method A and B are both 0)

From the result we can see that:

In method A, when n = 171, it will overflow.

In method B, when n = 171, k = 0, it will overflow.

In method C, when n = 200, k = 200, it will overflow.

q3: Which of the three expressions have the overflow problem? Explain why.

Method A have most serious overflow problems. Since it will compute prod (1: n) first, then no matter the magnitude of k, it will overflow when n grows.

Method B is better than A, but when K == 0, it will face the same problem as method A.

Method C is the best in these three methods, since it computes ((k+1): n)/(1:(n-k)) first, which lower the scale and the risk of overflow. But it inevitably faces overflow problems when n and k grow in large scale.

APPENDIX

Q1:

```
x1<- 1/3
x2<- 1/4
if ( 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' )
}

```
```

```
[1] "Subtraction is wrong"
[1] "Subtraction is correct"
```

```
x1<- 1/3
x2<- 1/4
if ( 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' )
}
```
```

Q2:

```
dev <- function(x)
{
  eta <- 10**-15
  return ((x+eta-x)/eta)
}
print(dev(1))
print(dev(100000))
```

```
[1] 1.110223
[1] 0
```


```
dev <- function(x)
{
  eta <- 10**-15
  return ((x+eta-x)/eta)
}
print(dev(1))
print(dev(100000))
```

Q3:

```r
myvar <- function(x)
{
  len <- length(x)
  return((sum(x*x)-sum(x)*sum(x)/len)/(len-1))
}

myvar2 <- function(x)
{
  m <-mean(x)
  len <- length(x)
  return(sum((x-m)*(x-m))/(len-1))
}

test_myvar <-function(x,i)
{
  return(myvar(x[1:i])-var(x[1:i]))
}

test_myvar2 <-function(x,i)
{
  return(myvar2(x[1:i])-var(x[1:i]))

}
xq3 <- rnorm(10000,mean=10**8,sd=1)
y<-c()
for(i in 1:10000)
{
  y<-c(y,test_myvar(xq3,i))
}
plot(1:10000,y[1:10000],main = 'original method')

y<-c()
for(i in 1:10000)
{
  y<-c(y,test_myvar2(xq3,i))
}
plot(1:10000,y[1:10000],main = 'promoted method')
```

The print result is in the Question 3.

```
myvar <- function(x)
{
  len <- length(x)
  return((sum(x*x)-sum(x)*sum(x)/len)/(len-1))
}

myvar2 <- function(x)
{
  m <-mean(x)
  len <- length(x)
  return(sum((x-m)*(x-m))/(len-1))
}

test_myvar <-function(x,i)
{
  return(myvar(x[1:i])-var(x[1:i]))
}

test_myvar2 <-function(x,i)
{
  return(myvar2(x[1:i])-var(x[1:i]))

}
xq3 <- rnorm(10000,mean=10**8,sd=1)
y<-c()
for(i in 1:10000)
{
  y<-c(y,test_myvar(xq3,i))
}
plot(1:10000,y[1:10000],main = 'original method')

y<-c()
for(i in 1:10000)
{
  y<-c(y,test_myvar2(xq3,i))
}
plot(1:10000,y[1:10000],main = 'promoted method')
```

Q4

```r
plotQ4 <- function()
{
  max_for_An <- 0
  max_for_Bn <- 0
  max_for_Cn <- 0
  max_for_Bk <- 0
  max_for_Ck <- 0
  for(i in 1:200)
  {
    if(prod(1:i)==Inf)
    {
      max_for_An <- i
      break
    }
  }
  flagB <- 1
  for(i in 1:200)
  {
    for(j in 0:i)
    {
      if(flagB)
      if(prod((j+1):i)==Inf||prod(1:(i-j))==Inf)
      {
        max_for_Bn <- i
        max_for_Bk <- j
        flagB <- 0
      }
      if(prod(((j+1):i)/(1:(i-j)))==Inf)
        {max_for_Cn <- i;max_for_Ck <- j;break}
    }
  }
  res <- c(max_for_An,max_for_Bn,max_for_Cn)
  res2 <- c(0,max_for_Bk,max_for_Ck)

  df <-data.frame(method = c('method A','method B','method C'),n = res)
  p1<-ggplot2::ggplot(df, ggplot2::aes(x = method, y = n)) +
  ggplot2::geom_bar(stat = "identity")+
  geom_text(aes(label = n), vjust = 2, colour = "white", position = position_dodge(.9), size = 5)+
  ggplot2::ggtitle('n when overflow occurs')+
    ggplot2::theme(plot.title = ggplot2::element_text(hjust=0.5))

  df <-data.frame(method = c('method A','method B','method C'),n = res2)
  p2<-ggplot2::ggplot(df, ggplot2::aes(x = method, y = n)) +
  ggplot2::geom_bar(stat = "identity")+
  geom_text(aes(label = n), vjust = 2, colour = "white", position = position_dodge(.9), size = 5)+
  ggplot2::ggtitle('k when overflow occurs')+
    ggplot2::theme(plot.title = ggplot2::element_text(hjust=0.5))
  print(p1)
  print(p2)
}
plotQ4()
```

The printed graphic is in the Question4 part

```r
plotQ4 <- function()
{
  max_for_An <- 0
  max_for_Bn <- 0
  max_for_Cn <- 0
  max_for_Bk <- 0
  max_for_Ck <- 0
  for(i in 1:200)
  {
    if(prod(1:i)==Inf)
    {
      max_for_An <- i
      break
    }
  }
  flagB <- 1
  for(i in 1:200)
  {
    for(j in 0:i)
    {
      if(flagB)
      if(prod((j+1):i)==Inf||prod(1:(i-j))==Inf)
      {
        max_for_Bn <- i
        max_for_Bk <- j
        flagB <- 0
      }
      if(prod(((j+1):i)/(1:(i-j)))==Inf)
        {max_for_Cn <- i;max_for_Ck <- j;break}
    }
  }
  res <- c(max_for_An,max_for_Bn,max_for_Cn)
  res2 <- c(0,max_for_Bk,max_for_Ck)

  df <-data.frame(method = c('method A','method B','method C'),n = res)
  p1<-ggplot2::ggplot(df, ggplot2::aes(x = method, y = n)) +
  ggplot2::geom_bar(stat = "identity")+
  ggplot2::geom_text(ggplot2::aes(label = n), vjust = 2, colour = "white", position
= ggplot2::position_dodge(.9), size = 5)+
  ggplot2::ggtitle('n when overflow occurs')+
    ggplot2::theme(plot.title = ggplot2::element_text(hjust=0.5))

  df <-data.frame(method = c('method A','method B','method C'),n = res2)
  p2<-ggplot2::ggplot(df, ggplot2::aes(x = method, y = n)) +
  ggplot2::geom_bar(stat = "identity")+
  ggplot2::geom_text(ggplot2::aes(label = n), vjust = 2, colour = "white", position
= ggplot2::position_dodge(.9), size = 5)+
  ggplot2::ggtitle('k when overflow occurs')+
    ggplot2::theme(plot.title = ggplot2::element_text(hjust=0.5))
  print(p1)
  print(p2)
}
plotQ4()
```