MA615-HW4

Quan Zhou

October 14th, 2016

1. Warm-Up

```
#1
library(ggplot2)
#(a)
xstat <- function(x){
  if(is.numeric(x)==F)stop("Invalid input. xstat only takes numeric vectors.")
  return(list(mean(x),median(x),var(x)))
}
z <- c(seq(1:10))
z1 <- c("Alex","Bob","Catherine","Daniel")

#(b)
summ <- function(x, n){
  return(sum(exp(-x)*x^(seq(0, n, by=1))/factorial(seq(0, n, by=1))))
}

# x = log 2
# n = 10
summ(log(2), 10)
```

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

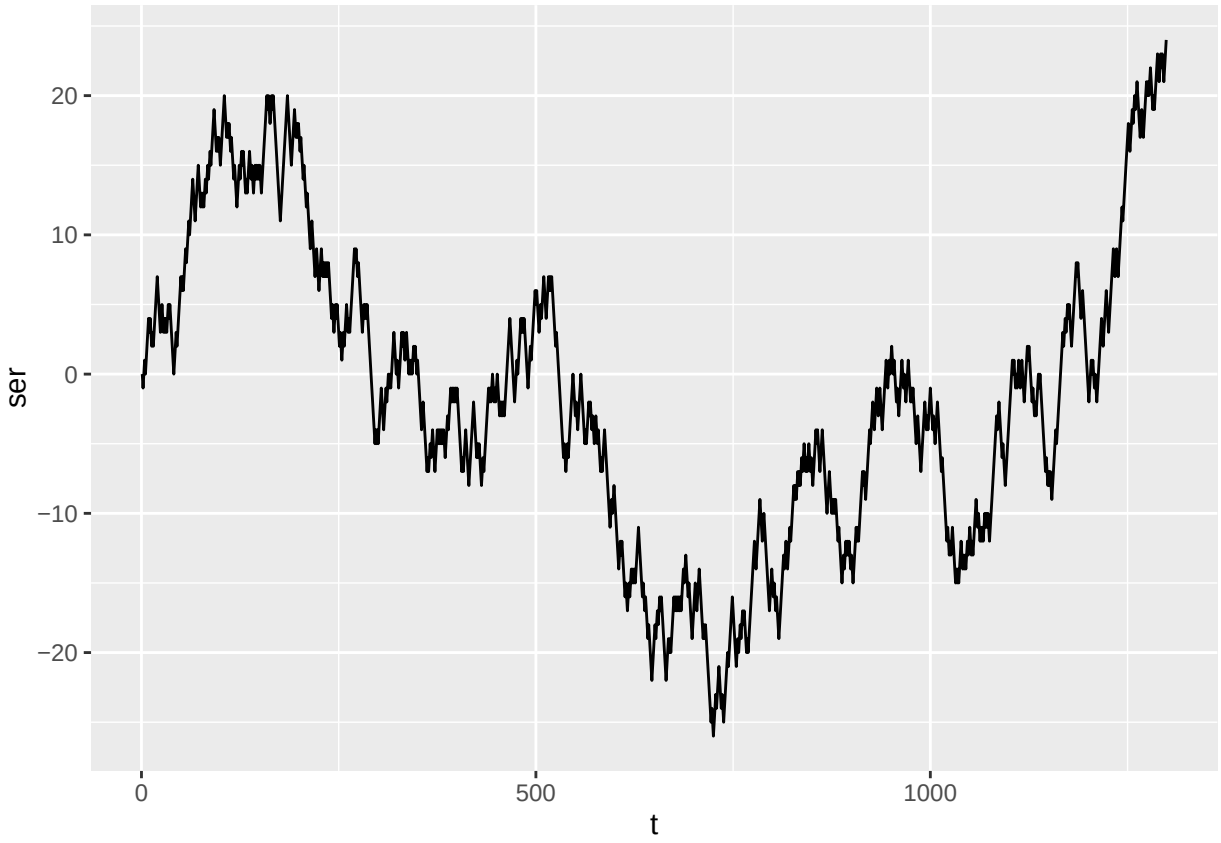
```
#(c) goes through every entry in a list, checks whether it is a character vector
m<-list("abc", 31, "John", "z", 7, -5)
c <- unlist(m[sapply(m, function(x) is.character(x))])
cat(c)
```

```
## abc John z
```

```
#(d) random walk function
rands <- function(k){
  # check to make sure input is an integer.
  k1 = as.integer(k)
  if(k1 != k)stop('rands() requires an integer value to start.')

  # initialize i and x
  i=1
  x=0
  while(x[i] != k){
    if(runif(1)<.5)D = 1
    else D = -1
    i = i+1
    x[i] = x[i-1]+ D
```

```
  }
  return(data.frame(x))
}

## try it with 24
set.seed(10)
ser <- data.frame(rands(24))
t <- 1:length(ser$x)
qplot(t,ser, data=ser, geom="line")
```

```
## Don't know how to automatically pick scale for object of type data.frame. Defaulting to continuous.
```



2. Moving Average

```
# (a)
set.seed(10)

rand <- rnorm(20)

ma3 <- function(x)
{
  #check for size and type
  r=length(x)
```

```
  x=c(x,0,0,0,x,0,0,0,x)
  x=matrix(x, ncol = r+2, nrow = 3, byrow = TRUE)

  return(as.vector(colMeans(x)))

}

mov <- ma3(rand)

# (b)
ma3_2 <- function(x,k){filter(x,rep(1/k,k), sides=2)}
# TEST: when length(x) = 10 and k = 3
ma3_2(c(seq(1:10)), 3)
```

```
## Time Series:
## Start = 1
## End = 10
## Frequency = 1
##  [1] NA  2  3  4  5  6  7  8  9 NA
```

```
# (c)
# TEST: when length(x) = 10 and k = 12
# ma3_2(c(seq(1:10)), 12)

print("Error in filter(x, rep(1/k, k), sides = 2) : ")
```

```
## [1] "Error in filter(x, rep(1/k, k), sides = 2) : "
```

```
print(" filter is longer than time series.")
```

```
## [1] " filter is longer than time series."
```

Error occurs when k is greater than length of vector x. This makes sense because it can not average over a domain larger than the given domain defined in x.

```
# (d)
ma3_3 <- function(x,k){
  if(length(x)<=k)stop("Error in ma3_3: sequence cannot be longer than length of vector .")
  else return(filter(x,rep(1/k,k), sides=2))
}
# TEST: when length(x) = 10 and k = 12
# ma3_2(c(seq(1:10)), 12)
#
# (e)
ma3_3(c(seq(1:10)), 1)
```

```
## Time Series:
## Start = 1
## End = 10
## Frequency = 1
##  [1]  1  2  3  4  5  6  7  8  9 10
```

It returns the vector again. We can write a case for k = 1 such that it does not have to do the computation at all.

```
ma3_4 <- function(x,k){
  if(length(x)<k)stop("Error in ma3_3: sequence cannot be longer than length of vector .")
  else if(k == length(x))return(mean(x))
  else if(k == 1)return(x)
  else return(filter(x,rep(1/k,k), sides=2))
}

ma3_4(c(seq(1:10)), 1) # k = 1
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
ma3_4(c(seq(1:10)), 10) # k = length(x)
```

```
## [1] 5.5
```

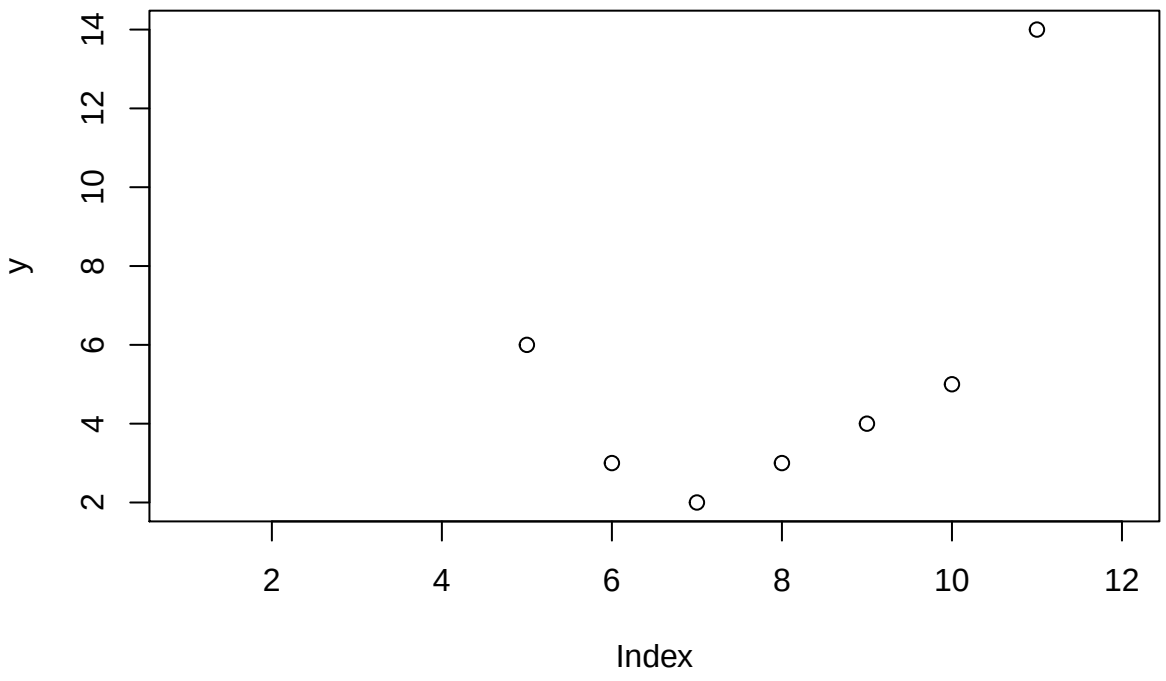
3. Optional Plot

```
left <- function (x) {
  return(x ^ 2 + 2 * x + 3)
}

mid <- function (x) {
  return(x + 3)
}

right <- function (x) {
  return(x ^ 2 + 4 * x - 7)
}

combo <- function(x){
  if(x<0){return(left(x))}
  else if(x >=0 & x<2) {return(mid(x))}
  else if(x>=2) {return(right(x))}
  else(stop("undefined domain x!"))
}

four24 <- function(x, plot = FALSE){
  loops = length(x)
  out = rep(0, loops)
  for (i in 1:loops) {
    if(x[i] <= -4 | x[i] >= 4){
      warning("input values outside the domain -4<x<4 are skipped")
      out[i] = NA
      next
    }
  out[i] = combo(x[i])
  }
  if(plot == TRUE){
```

```
    plot(out)
  }
  return(out)
}

y <- four24(c(-7:4))
```

```
## Warning in four24(c(-7:4)): input values outside the domain -4<x<4 are
## skipped

## Warning in four24(c(-7:4)): input values outside the domain -4<x<4 are
## skipped

## Warning in four24(c(-7:4)): input values outside the domain -4<x<4 are
## skipped

## Warning in four24(c(-7:4)): input values outside the domain -4<x<4 are
## skipped

## Warning in four24(c(-7:4)): input values outside the domain -4<x<4 are
## skipped
```

```
plot(y)
```



4. Matrix Input

```
dblodd = function(mx) {
  ifelse(mx %% 2 == 0, mx, 2*mx)
}
```

```r
mx <- matrix(c(1,1,3,5,2,6,-2,-1,-3), nrow = 3, byrow = TRUE)
mx
```

```
##      [,1] [,2] [,3]
## [1,]    1    1    3
## [2,]    5    2    6
## [3,]   -2   -1   -3
```

```r
res <- dblodd(mx)
res
```

```
##      [,1] [,2] [,3]
## [1,]    2    2    6
## [2,]   10    2    6
## [3,]   -2   -2   -6
```

5. Poisson Process

```r
library(ggplot2)
library(qualityTools)
```

```
## Loading required package: Rsolnp
```

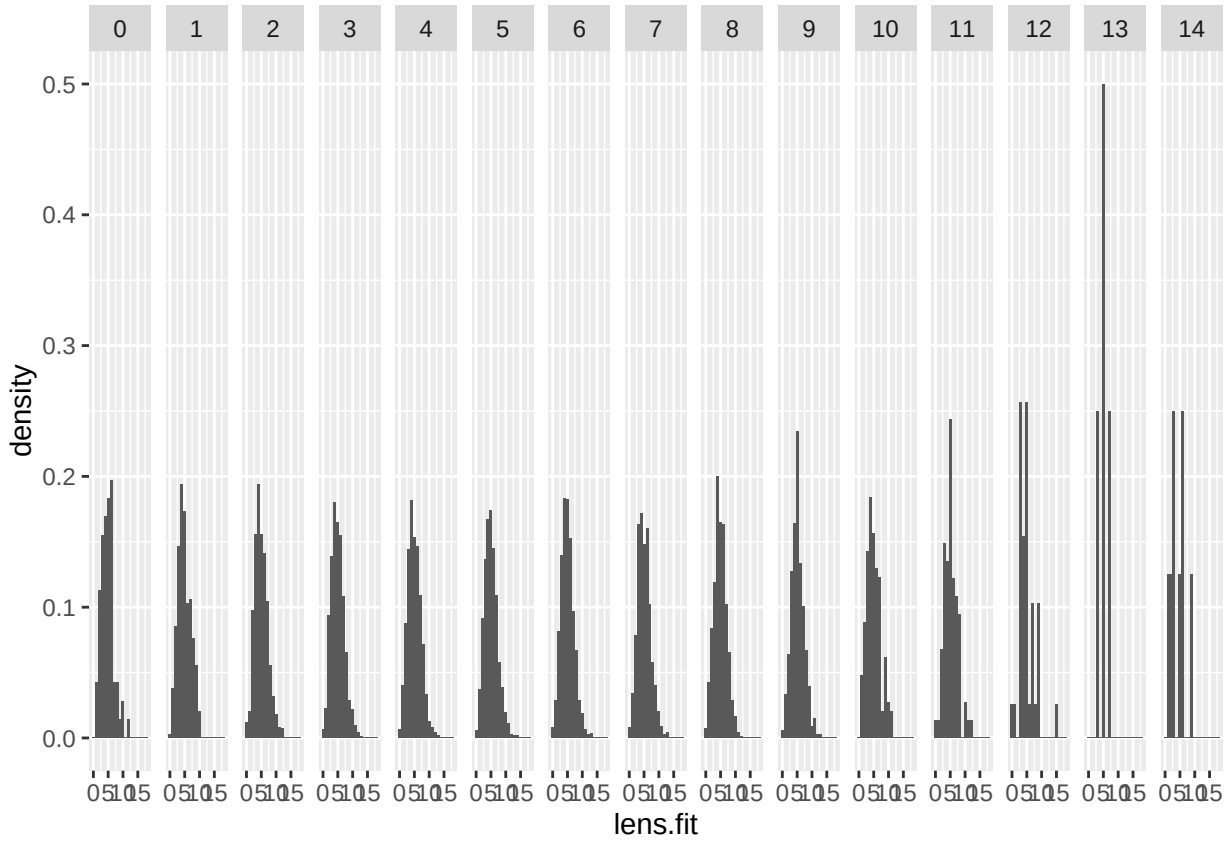
```
## Loading required package: MASS
```

```
##
## Attaching package: 'qualityTools'
```

```
## The following object is masked from 'package:stats':
##
##     sigma
```

```r
# (a)
poi.test = function(lambda, M) {
  res = rexp(1, lambda)
  len = 1
  while (res[len] < M){
# keep adding values until one exceeds M
    res = c(res, res[len] + rexp(1, lambda))
    len = len + 1
  }
  # return everything except the value > M
  return(res[-len])
}

#(b)
lens = numeric(10000)
for (i in 1:10000)
  lens[i] = length(poi.test(5, 1))
```

```
lens.fit <- c(rpois(10000,5))
data <- data.frame(lens, lens.fit)
ggplot(data, aes(lens.fit)) +
  geom_histogram(aes(y=..density..), binwidth=1,position="identity")+
  facet_grid(.~lens)
```




```
#qqplot(qpois(ppoints(10000), lambda = 5), ppp.main="qqplot")

poi.mean <- mean(lens)
poi.vars <- var(lens)
```

The lengths are distributed in a Poisson distribution: as the Poisson function fits well.