Homework Four

Yi Chen(yc3356)

November 19, 2017

Homework Four

Goals: Practice with simulating distributions via the Inverse Transform Method. Summarizing data using distributions and estimating parameters.

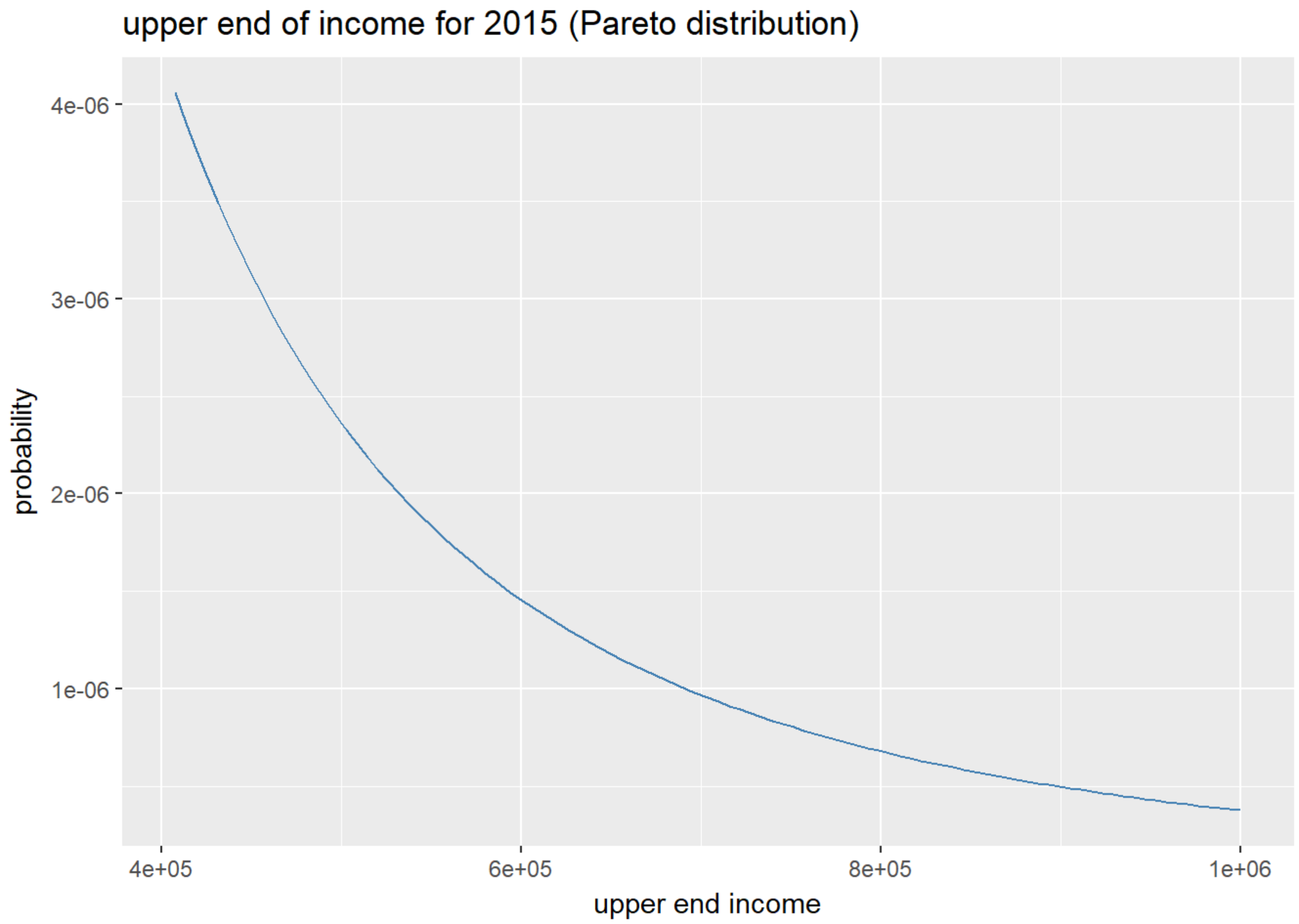
Part 1

i.

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.2
```

```
# define fuction f
f <- function(x,a=2.654,x_min=407760){
        stopifnot(x >= x_min)
        return(((a-1)/x_min)*((x/x_min)^(-a)))
}


ggplot(data.frame(x=c(407760,1000000)))+
        stat_function(mapping = aes(x=x),fun=f,col='steelblue')+
        labs(title='upper end of income for 2015 (Pareto distribution)',x='upper end income',
y='probability')
```



ii

$F(x) = 1 - (\frac{x}{x_{min}})^{-a+1}$

Thus, we can calcuate the inverse function is:

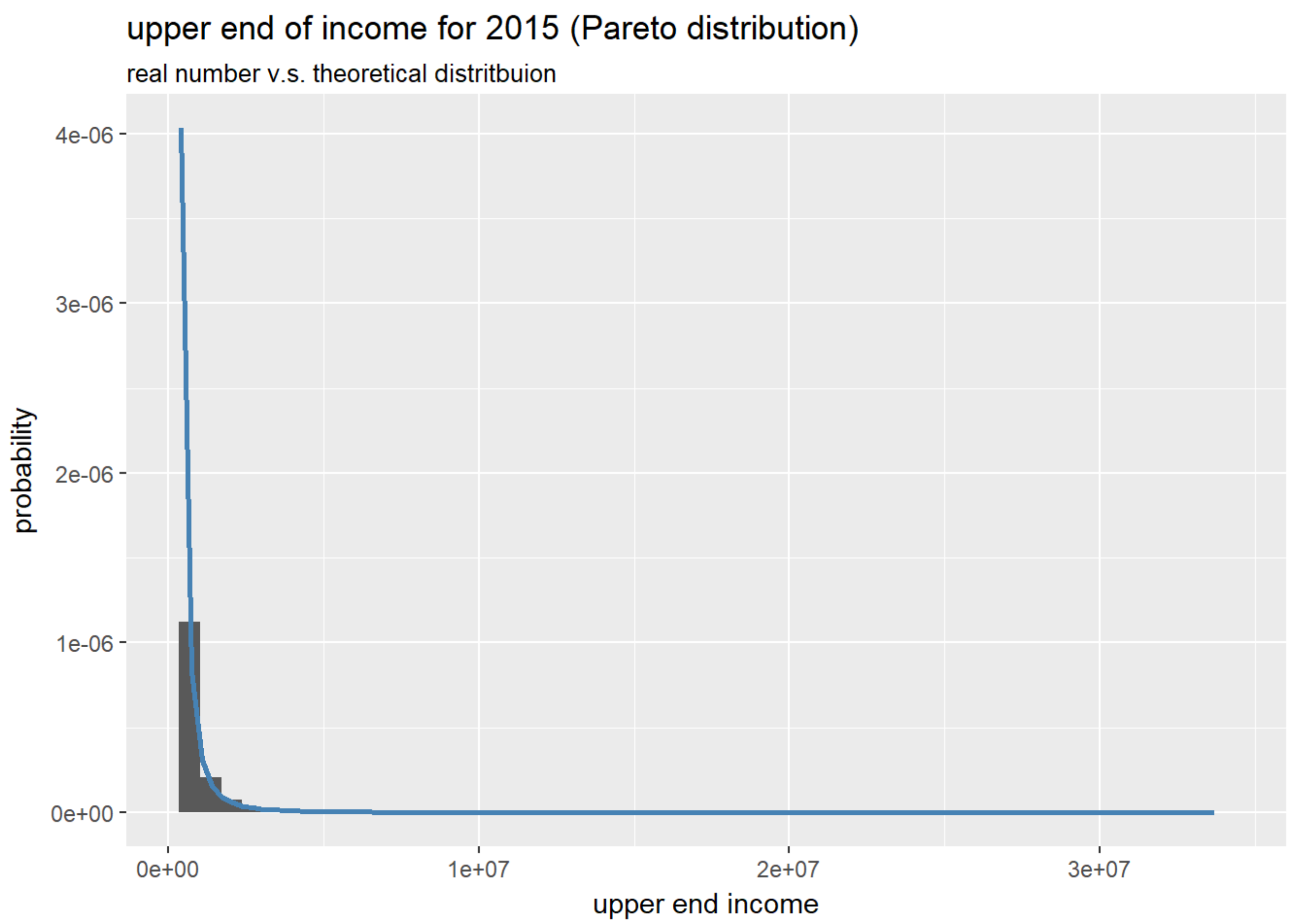
$F^{-1}(u) = x_{min}(1-u)^{\frac{1}{-a+1}}$

```
# define the function
upper.income <- function(u,a=2.654,x_min=407760){
        stopifnot(u>=0 & u<=1)
        return(x_min*(1-u)^(1/(-a+1)))
}

# test
upper.income(0.5)
```

```
## [1] 620020.2
```

iii

```
# inverse transfrom method
n <- 1000 # number of draws
u <- runif(n)
y <- upper.income(u)

ggplot(data = data.frame(y))+
        geom_histogram(mapping = aes(x=y,y=..density..),bins=50)+
        stat_function(mapping = aes(x=y),fun = f, args = list(a=2.654,x_min=407760),col='stee
lblue',lwd=1)+
        labs(title='upper end of income for 2015 (Pareto distribution)',x='upper end income',
y='probability',subtitle="real number v.s. theoretical distritbuion")
```

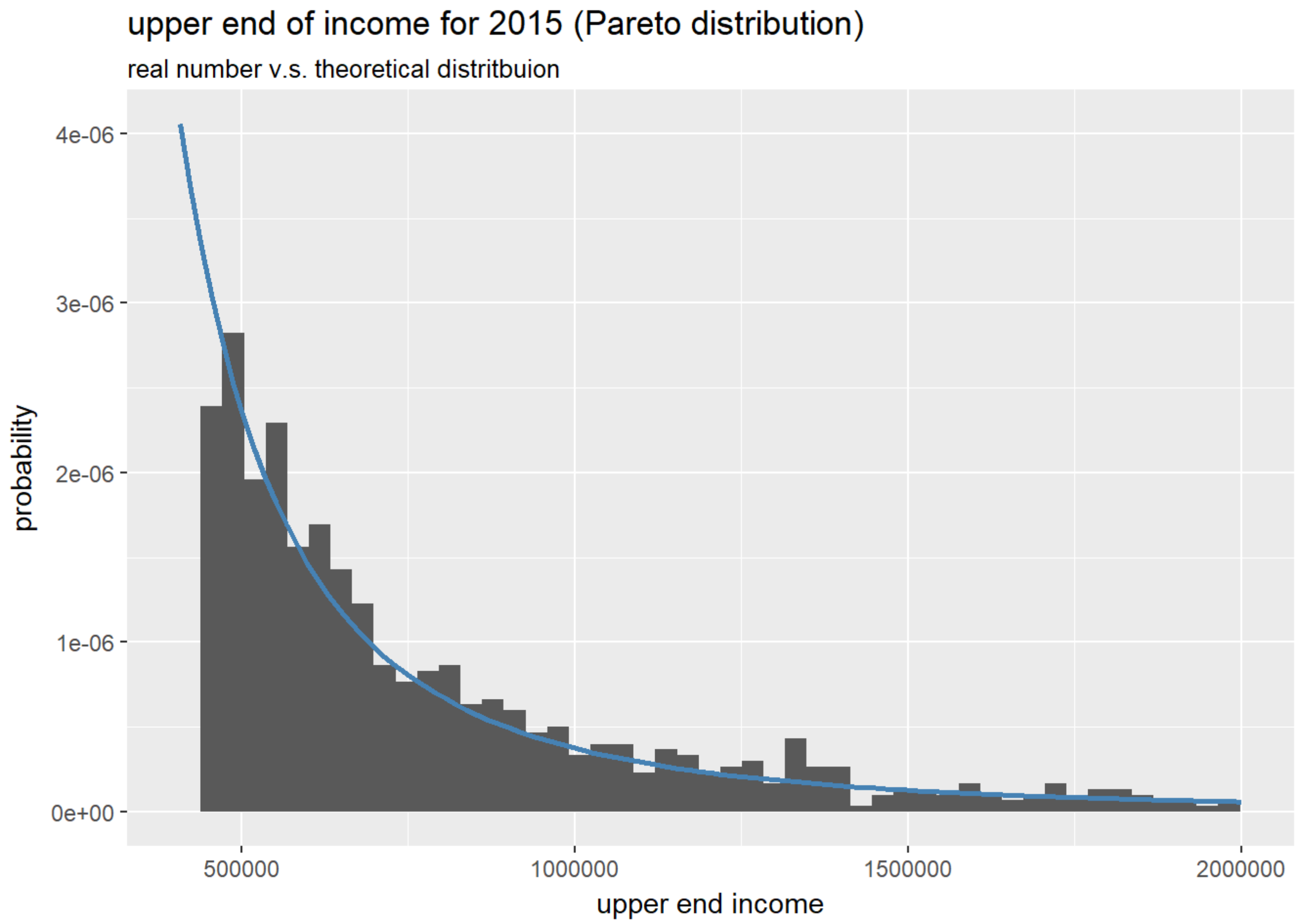


The picture above is hard to compare, we rerange the value of upper end income.

```
ggplot(data = data.frame(y))+
        geom_histogram(mapping = aes(x=y,y=..density..),bins=50)+
        stat_function(mapping = aes(x=y),fun = f, args = list(a=2.654,x_min=407760),col='stee
lblue',lwd=1)+
        labs(title='upper end of income for 2015 (Pareto distribution)',x='upper end income',
y='probability',subtitle="real number v.s. theoretical distritbuion")+
        xlim(407760.5, 2e06)
```

```
## Warning: Removed 74 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 1 rows containing missing values (geom_bar).
```



iv

$Pr(X > w) = (\frac{w}{407760})^{-a+1}$

Thus we can calculate that

$\frac{1}{2} = (\frac{w}{x_{min}})^{-a+1}$

Thus, we get:

$w = x_{min}(\frac{1}{2})^{\frac{1}{-a+1}}$

```
# the actual median of pareto distribution

w1 <- 407760*(1/2)^(1/(-2.654+1))

# median income of simuated set
w2 <- median(y)

cat("The actual median is:",w1,'\n','The median of simuated set is:',w2,'\n',"Thus, they are
 close to each other.")
```

```
## The actual median is: 620020.2
##  The median of simuated set is: 642346.5
##  Thus, they are close to each other.
```

Part 2

```
genres <- read.csv('moretti.csv',header = TRUE)
```

i

```
# define poisLoglik fuction
poisLoglik <- function(data,lambda){
        return(sum(log(((lambda^data)*(exp(-lambda)))/(factorial(data)))))
}

poisLoglik(data=c(1,0,0,1,1),lambda = 1)
```

```
## [1] -5
```

ii

```
# define the function
count_new_genres <- function(year){
        return(sum(genres$Begin==year))
}

# calculate the value of 1803 and 1850
count_new_genres(1803);
```

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

```
count_new_genres(1850)
```

```
## [1] 3
```

iii

```
# creat the vector
new_genres <- rep(NA,length(1740:1900))
for(i in 1740:1900){
        new_genres_this_year <- count_new_genres(i)
        new_genres[i-1739] <- new_genres_this_year
}

names(new_genres) <- 1740:1900

index_1803 <- which(names(new_genres)==1803)
index_1850 <- which(names(new_genres)==1850)

new_genres[index_1803] == count_new_genres(1803)
```

```
## 1803
## TRUE
```

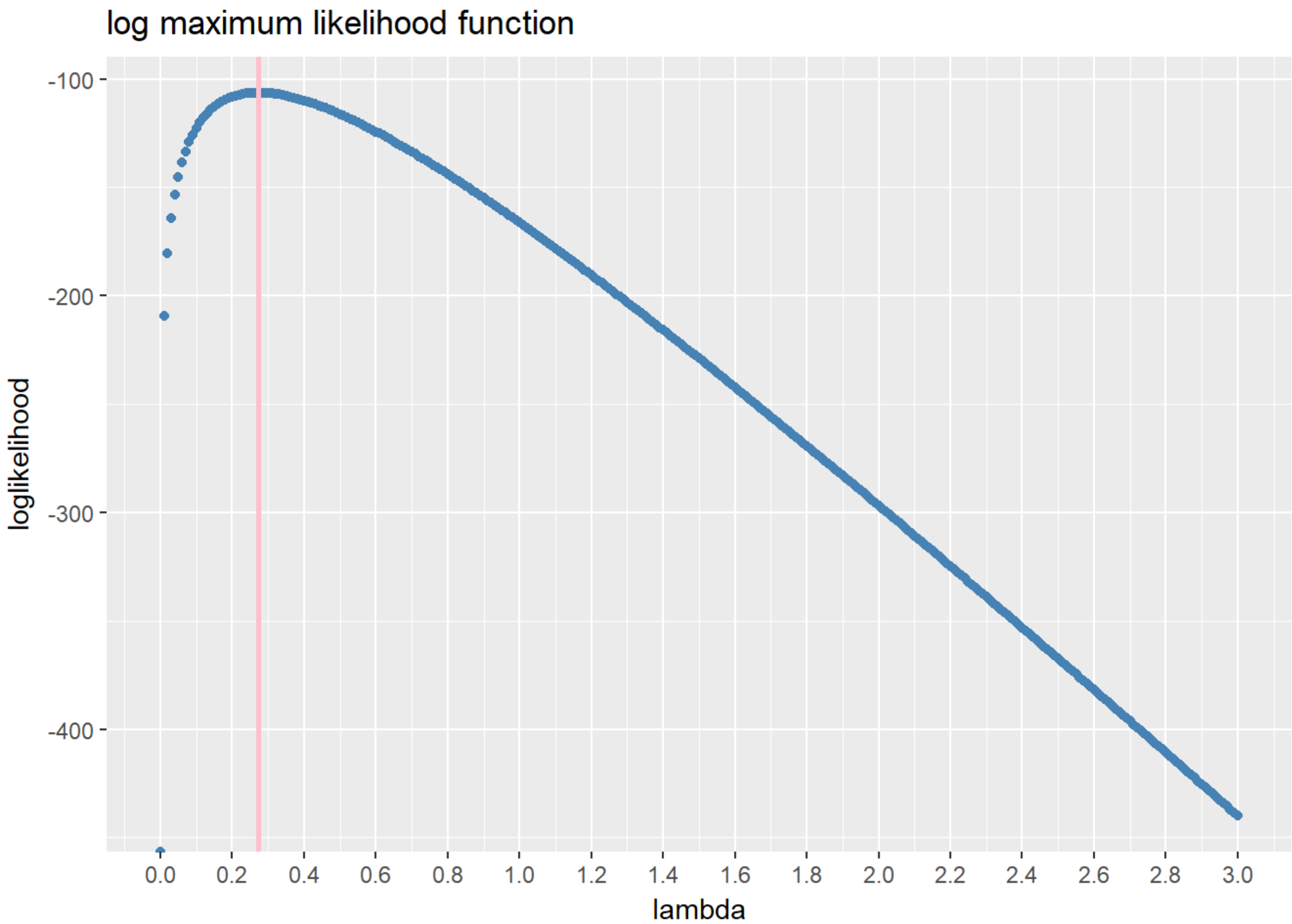
```
new_genres[index_1850] == count_new_genres(1850)
```

```
## 1850
## TRUE
```

1803 and 1850 is 64th and 111st data in the vector. The value for 1803 should be 0 and the value for 1850 should be 3. The result is same in both way.

iv

```
y <- c()
for(i in seq(0,3,0.01)){
        y_this <- poisLoglik(data=new_genres,lambda = i)
        y <- c(y,y_this)
}

ggplot()+
        geom_point(mapping = aes(x=seq(0,3,0.01),y=y),col='steelblue')+
        geom_vline(xintercept=0.273,lwd=1,col='pink')+
        labs(title='log maximum likelihood function',x='lambda',y='loglikelihood')+
        scale_x_continuous(breaks=seq(0 , 3, 0.2))
```



v

```
poisLoglik_new <- function(lambda){
        return(-sum(log(((lambda^new_genres)*(exp(-lambda)))/(factorial(new_genres)))))
        # add a negative so that we can calculate the maximum
}

nlm(poisLoglik_new,0.5)
```

```
## $minimum
## [1] 106.3349
##
## $estimate
## [1] 0.2732914
##
## $gradient
## [1] 1.278977e-07
##
## $code
## [1] 1
##
## $iterations
## [1] 7
```

vi

```
intervals <- diff(genres$Begin)

mean <- mean(intervals)

stantdard_deviation <- sd(intervals)

coefficient_of_variation <- stantdard_deviation/mean

cat('mean is:',mean,'\n',
    'standard deviation is:',stantdard_deviation,'\n',
    'coefficient of variation:',coefficient_of_variation)
```

```
## mean is: 3.44186
##  standard deviation is: 3.705224
##  coefficient of variation: 1.076518
```

vii

a.

```
random_draws <- rpois(161,0.273)
head(random_draws)
```

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

b.

```
interval_function <- function(data){
        calculate <- c()
        for(i in 1:length(data)){
                calculate <- c(calculate,rep(i,data[i]))
        }
        diff(calculate)
}

all(interval_function(data=new_genres) == intervals)
```

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

c.

```
simulation <- function(num.years,mean.genres){
        simulated_number <- rpois(num.years,lambda = mean.genres)
        inter_appearance <- interval_function(simulated_number)
        coefficient_of_variation <- sd(inter_appearance )/mean(inter_appearance )
        return(list("inter_appearance"=inter_appearance,"coefficient_of_variation"=coefficien
t_of_variation))

}

simulation(num.years = 161,mean.genres = 0.273)
```

```
## $inter_appearance
##  [1]  9  4  1  6  2  3  2  3  0  3  7  6  6  1  1  7  2  5  0  4  2  3  9
## [24]  2  4  0  1  3  4  5  4  0  7  8 14  0  0  2  0  4  2  1  3  2  0  3
##
## $coefficient_of_variation
## [1] 0.8856028
```

viii

```
cof <- c()
for(i in 1:10000){
        this_value <- simulation(num.year = 161,mean.genres = 0.273)[[2]]
        cof <- c(cof,this_value)
}

mean(cof>coefficient_of_variation)
```

```
## [1] 0.2326
```