Final project 677_Illinois_precipitations

Jinyu

5/13/2022

#Introduction

To start with, I will solve the exercises mentioned in the book In All Likelihood

Answer the exercise 4.25

To solve this problem, we need to build a function to come up with the order statistics

```
# reference: https://stackoverflow.com/questions/24211595/order-statistics-in-r?msclkid=fd6683dac56711e

f <- function(x, a=0, b=1) dunif(x, a,b) #pdf function
F <- function(x, a=0, b=1) punif(x, a,b, lower.tail=FALSE) #cdf function

# a function of distribution of the order statistics
integrand <- function(x,r,n) {
  x * (1 - F(x))^(r-1) * F(x)^(n-r) * f(x)
}
```

Then, we get the approximation function

```
#obtain the expectation
E <- function(r,n) {
  (1/beta(r,n-r+1)) * integrate(integrand,-Inf,Inf, r, n)$value
}

# approx function
medianprrox<-function(k,n){
  m<-(k-1/3)/(n+1/3)
  return(m)
}
```

And we try to calculate the median when n=5 and when n=10

```
# get the value when n=5, i=3
print("Get the median value When n=5")
```

```
## [1] "Get the median value When n=5"
```

```r
E(3,5)
```

```
## [1] 0.5
```

```r
medianprrox(3,5)
```

```
## [1] 0.5
```

```r
# get the value when n=10, i=5.5
print("Get the median value When n=10")
```

```
## [1] "Get the median value When n=10"
```

```r
(E(5,10) + E(6,10))/2
```

```
## [1] 0.5
```

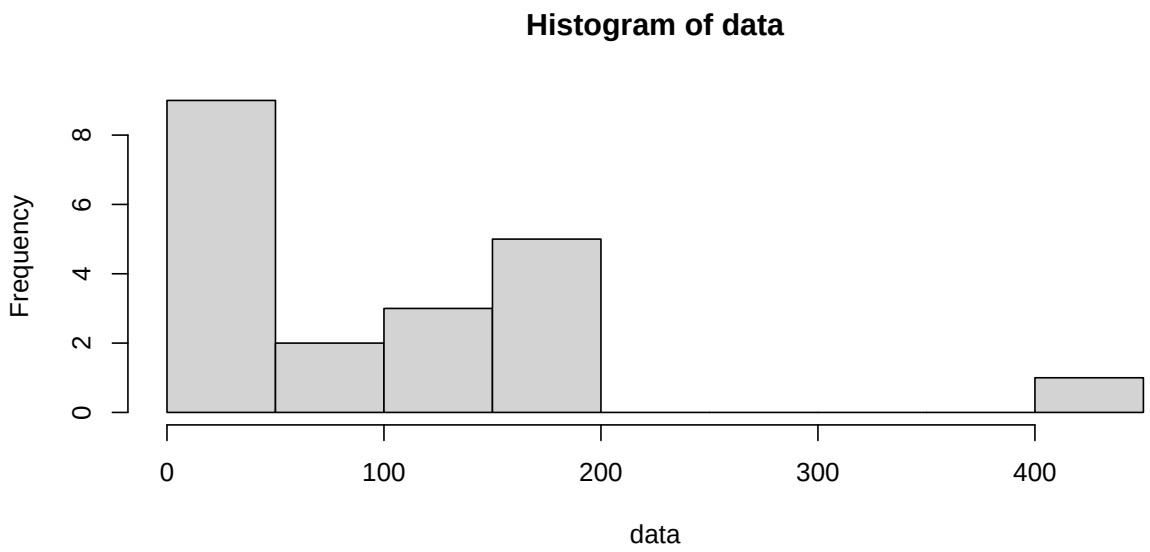
```r
(medianprrox(5,10) + medianprrox(6,10))/2
```

```
## [1] 0.5
```

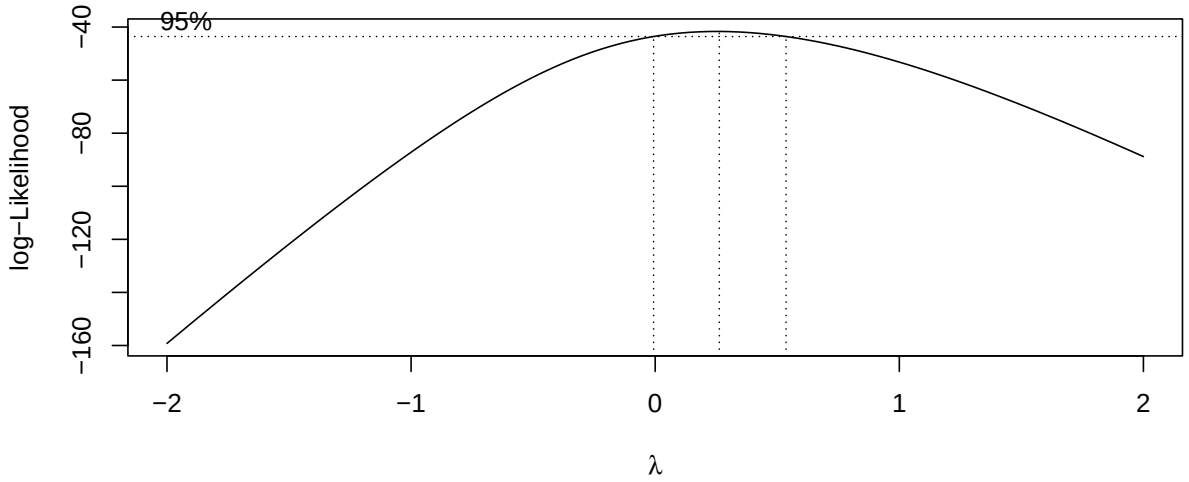
As we can see, the approximation for n=5 and n=10 are similar or exactly the same.

Answer the exercise 4.39

```r
data<-c(0.4,1.0,1.9,3.0,5.5,8.1,12.1,25.6,50.0,56.0,70.0,115.0,115.0,119.5,154.5,157.0,175.0,179.0,180.
hist(data)
```



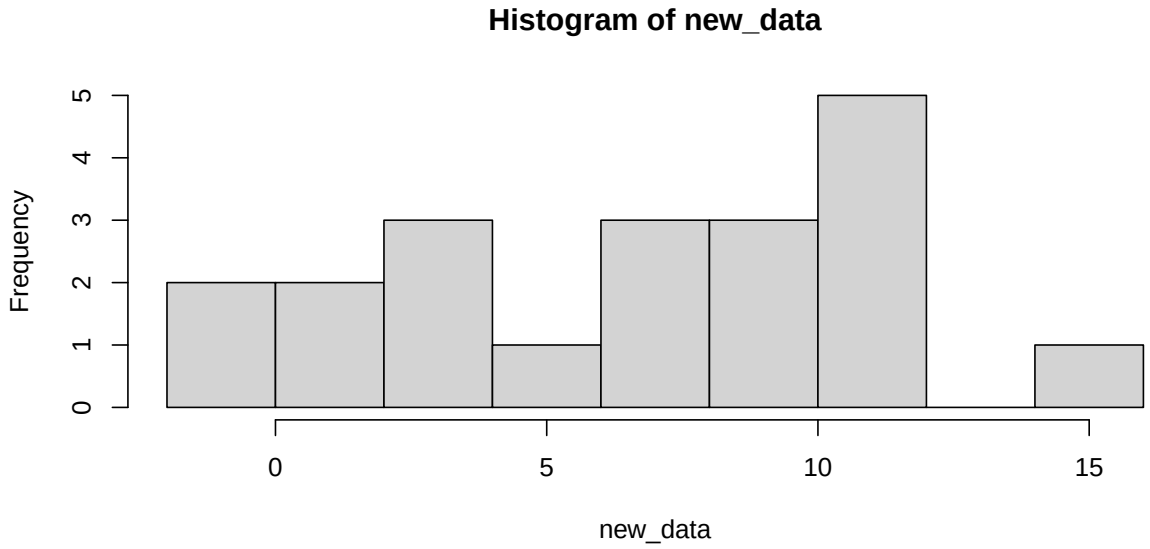
```r
# Conduct boxcox transformation
b <- boxcox(lm(data ~ 1))
```




```r
# Exact lambda
lambda <- b$x[which.max(b$y)]
lambda #lambda=0.2626263
```

```
## [1] 0.2626263
```

```r
new_data <- (data ^ lambda - 1) / lambda
hist(new_data)
```



Answer the exercise 4.27

```
Jan<-c(0.15,0.25,0.10,0.20,1.85,1.97,0.80,0.20,0.10,0.50,0.82,0.40,1.80,0.20,1.12,1.83,
       0.45,3.17,0.89,0.31,0.59,0.10,0.10,0.90,0.10,0.25,0.10,0.90)
Jul<-c(0.30,0.22,0.10,0.12,0.20,0.10,0.10,0.10,0.10,0.10,0.10,0.17,0.20,2.80,0.85,0.10,
       0.10,1.23,0.45,0.30,0.20,1.20,0.10,0.15,0.10,0.20,0.10,0.20,0.35,0.62,0.20,1.22,
       0.30,0.80,0.15,1.53,0.10,0.20,0.30,0.40,0.23,0.20,0.10,0.10,0.60,0.20,0.50,0.15,
      0.60,0.30,0.80,1.10,
      0.2,0.1,0.1,0.1,0.42,0.85,1.6,0.1,0.25,0.1,0.2,0.1)
```

(a)

```
summary(Jan)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1000  0.1875  0.4250  0.7196  0.9000  3.1700
```

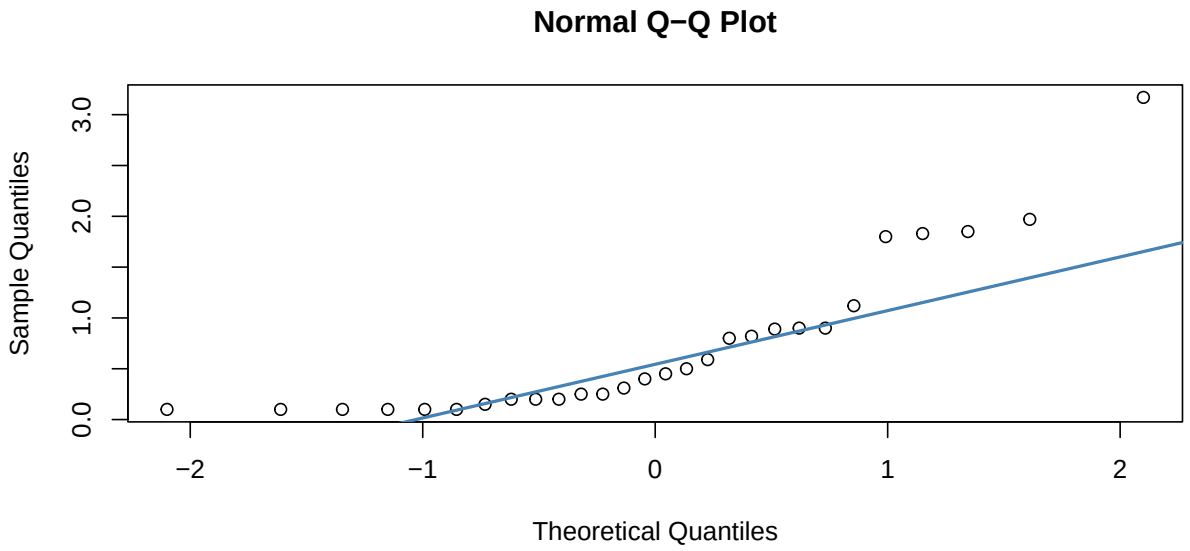
```
summary(Jul)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1000  0.1000  0.2000  0.3931  0.4275  2.8000
```

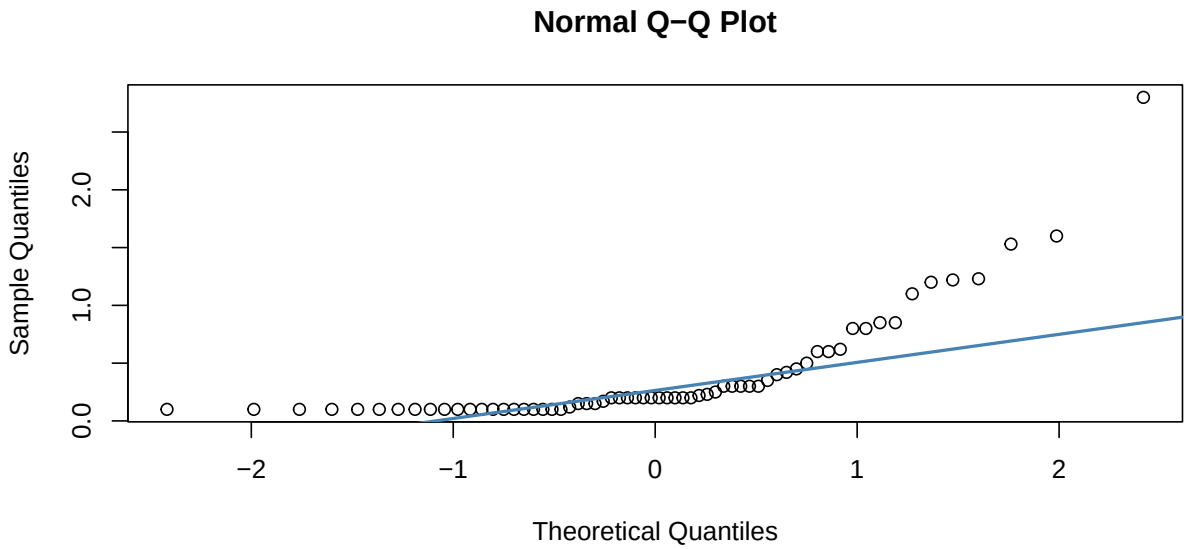
Jan's 1st, Median, Mean 3rd Max are higher than the one in Jul. Also, Jan's IQR is higher than the one in Jul.

(b)

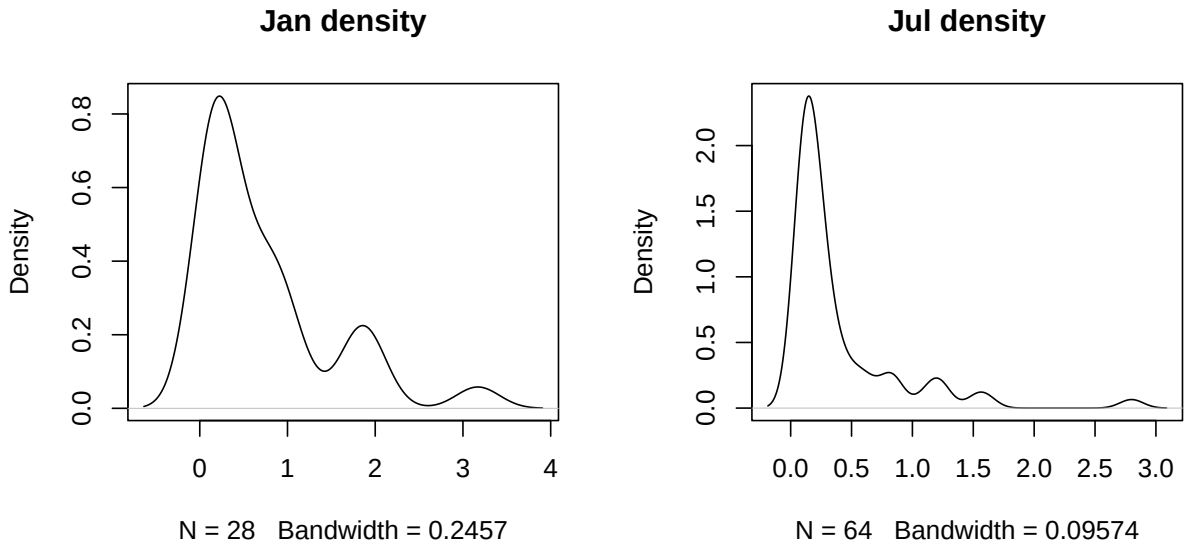
```
qqnorm(Jan, pch = 1)
qqline(Jan, col = "steelblue", lwd = 2)
```



```
qqnorm(Jul, pch = 1)
qqline(Jul, col = "steelblue", lwd = 2)
```




```
par(mfrow = c(1, 2))
plot(density(Jan),main='Jan density')
plot(density(Jul),main='Jul density')
```



The qqplots show that the sample doesn't follow normal distribution.
From the density plot, these data looks like gamma distribution. Therefore, gamma distribution can be considered to fit the model.

(c)

There are many ways to solve the problem. I listed three methods here. The first one is to use fitdist:

```
Jan.fit1=fitdist(Jan,'gamma','mle')
Jan.fit1
```

```
## Fitting of the distribution ' gamma ' by maximum likelihood
## Parameters:
##       estimate Std. Error
## shape 1.056222  0.2497495
## rate  1.467650  0.4396202
```

```
Jul.fit1=fitdist(Jul,'gamma','mle')
Jul.fit1
```

```
## Fitting of the distribution ' gamma ' by maximum likelihood
## Parameters:
##       estimate Std. Error
## shape 1.196419  0.1891196
## rate  3.043403  0.5936302
```

The second method is to nlm:

```
#https://stackoverflow.com/questions/59435824/nlm-with-multiple-variables-in-r
data<-Jan
neg_likelihood<-function(param){
  alpha<-param[1]
  beta<-param[2]
  p<-dgamma(data,shape=alpha,scale=1/beta)
  re<--1*sum(log(p))
  return(re)
}
#neg_likelihood(c(0.5,1))


p <- array(c(0.4, 0.4), dim = c(2, 1))
ans_jan <- nlm(f = neg_likelihood,p,hessian=T)
ans_jan$estimate
```

```
## [1] 1.056259 1.467754
```

```
data<-Jul
ans_jul <- nlm(f = neg_likelihood,p,hessian=T)
ans_jul$estimate
```

```
## [1] 1.196403 3.043315
```

Here is the std

```
#https://stats.stackexchange.com/questions/81542/standard-error-of-mle#:~:text=How%20are%20you%20obtain
sqrt(diag(solve(ans_jan$hessian))) #use hessian matrix to get std
```

```
## [1] 0.2498280 0.4397828
```

```
sqrt(diag(solve(ans_jul$hessian)))
```

```
## [1] 0.1891739 0.5938104
```

For MLE, do some stransformation loglikelihood into MLE:

```
exp(Jan.fit1$loglik)
```

```
## [1] 7.11117e-09
```

```
exp(Jul.fit1$loglik)
```

```
## [1] 0.02638693
```

```
exp(-ans_jan$minimum)
```

```
## [1] 7.11117e-09
```

```
exp(-ans_jul$minimum)
```

```
## [1] 0.02638693
```

From MLE, Jul's MLE is higher than the one of Jan. Jul's model is better than Jan's.
Parameter comparison: Jan's alpha is lower than Jul's alpha. Jan's beta is lower than Jul's beta.

The third way is to use optim. I will use this method to conduct profile likelihood.

```
#https://www.r-bloggers.com/2015/11/profile-likelihood/
# optim is similar to nlm
# Jan
x=Jan
prof_log_lik=function(a){
   b=(optim(1,function(z) -sum(log(dgamma(x,a,z)))))$par
   return(-sum(log(dgamma(x,a,b))))
 }

vx=seq(.1,3,length=50)
vl=-Vectorize(prof_log_lik)(vx)
```

```
## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly
```

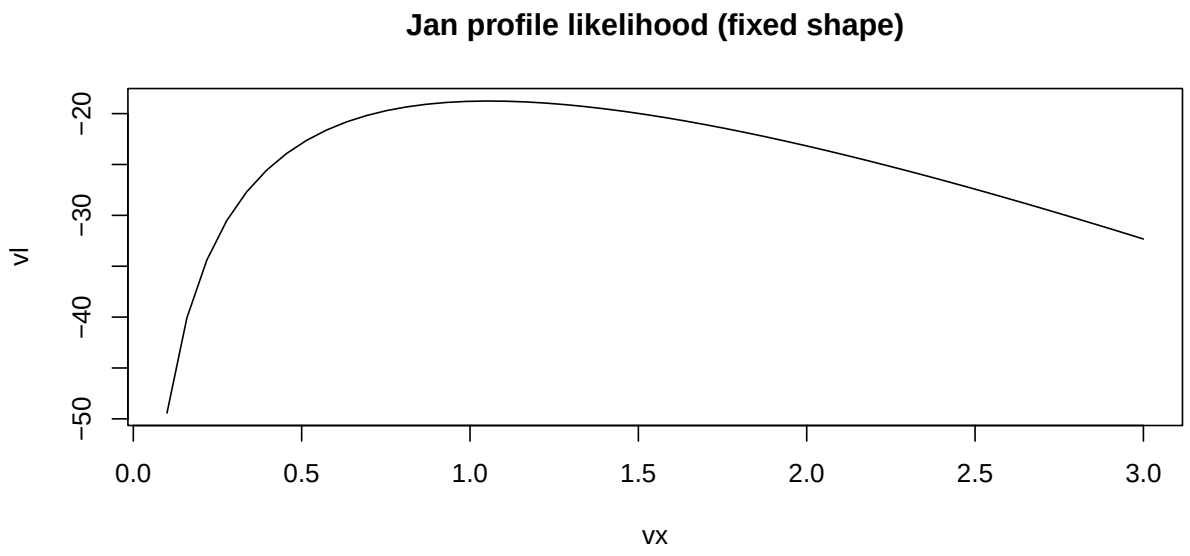
```
## Warning in dgamma(x, a, z): NaNs produced

## Warning in dgamma(x, a, z): NaNs produced

## Warning in dgamma(x, a, z): NaNs produced

## Warning in dgamma(x, a, z): NaNs produced

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in dgamma(x, a, z): NaNs produced

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly
```

```
## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly
```

```
## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly
```

```
## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly
```

```
plot(vx,vl,type="l",main='Jan profile likelihood (fixed shape)')
```




```
x=Jul
vx=seq(.1,3,length=50)
vl=-Vectorize(prof_log_lik)(vx)
```

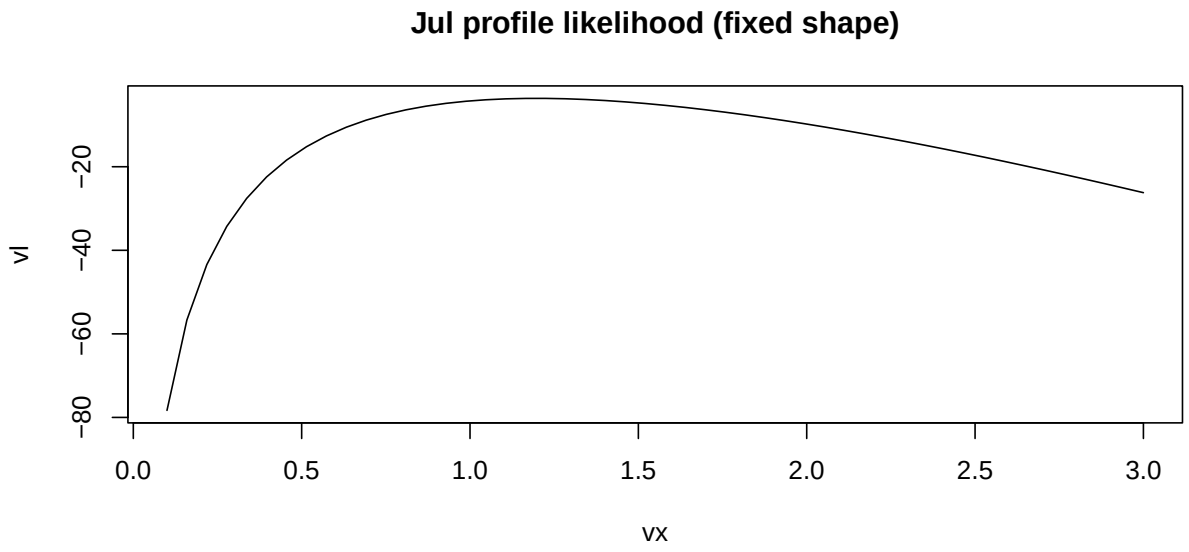
```
## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in dgamma(x, a, z): NaNs produced

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly
```

```
## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly
```

```
## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly
```

```
## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(z) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly
```

```
plot(vx,vl,type="l",main='Jul profile likelihood (fixed shape)')
```



For fixed rate, we can use the same method to get the profile likelihood.

```
x=Jan
prof_log_lik=function(z){
   a=(optim(1,function(a) -sum(log(dgamma(x,a,z)))))$par
   return(-sum(log(dgamma(x,a,z))))
 }

vx=seq(.1,3,length=50)
vl=-Vectorize(prof_log_lik)(vx)
```

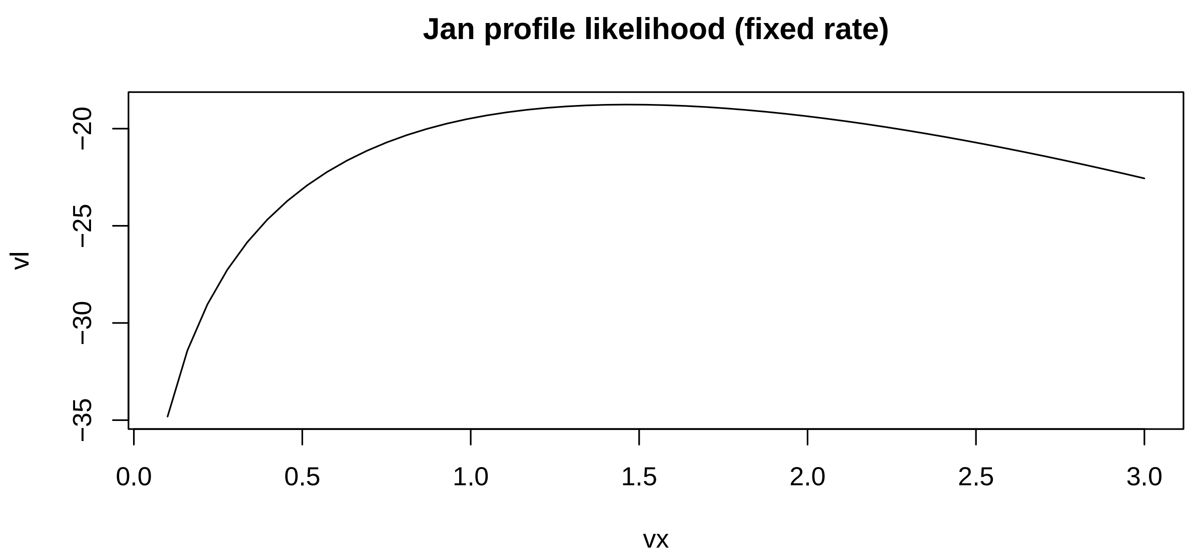
```
## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced
```

```
## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly
```

```
## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly
```

```
## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly
```

```r
plot(vx,vl,type="l",main='Jan profile likelihood (fixed rate)')
```



```
x=Jul

vx=seq(.1,5,length=50)
vl=-Vectorize(prof_log_lik)(vx)
```

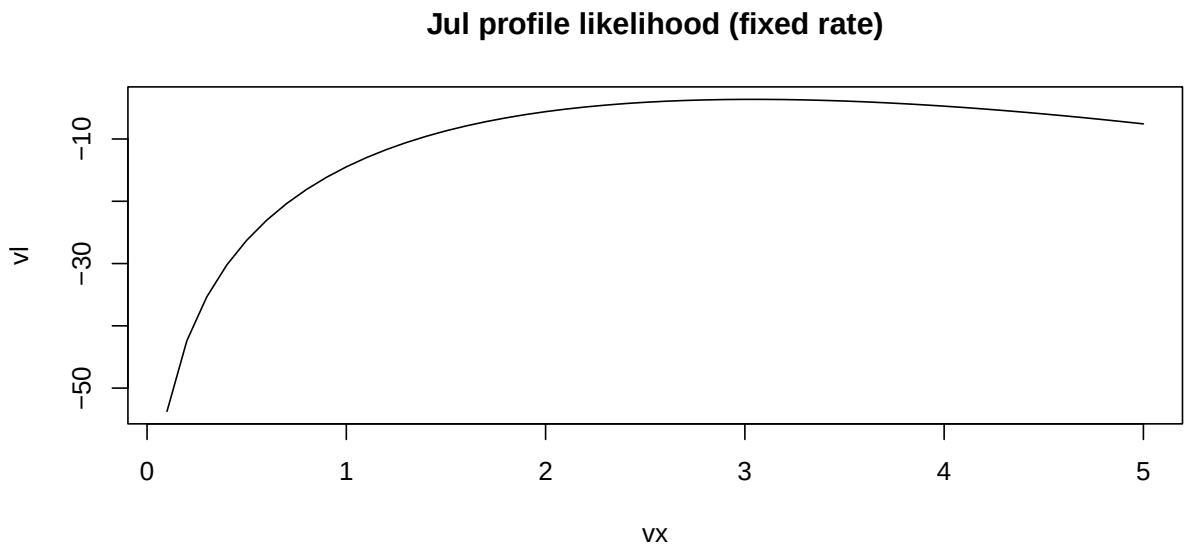
```
## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in dgamma(x, a, z): NaNs produced

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in dgamma(x, a, z): NaNs produced

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
```

```
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
```

```
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-
```

```
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly

## Warning in optim(1, function(a) -sum(log(dgamma(x, a, z)))): one-dimensional optimization by Nelder-M
## use "Brent" or optimize() directly
```

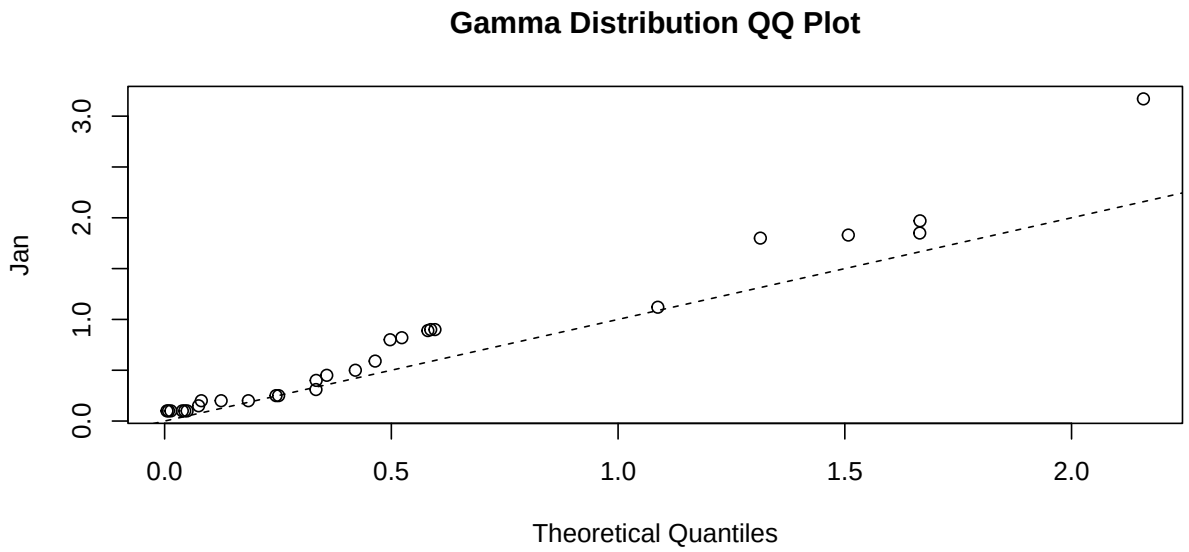
```
plot(vx,vl,type="l",main='Jul profile likelihood (fixed rate)')
```



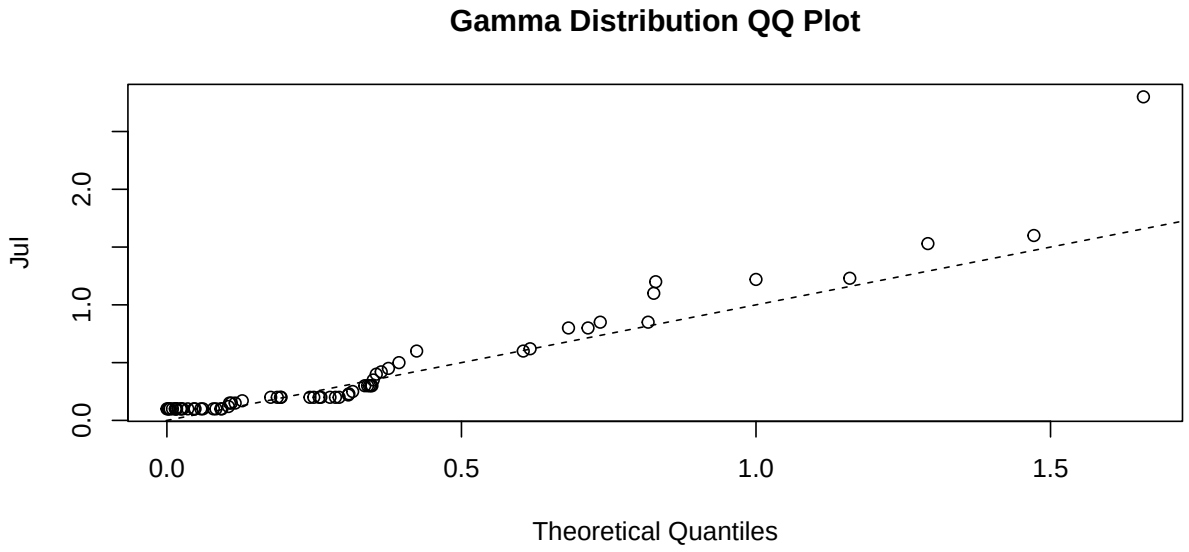
(d)

```
# library(qpToolkit)
# qqGamma(resid(Jan.fit))
# reference:qpToolkit
# https://github.com/qPharmetra/qpToolkit/blob/master/R/qqGamma.r
qqGamma <- function(x
                    , ylab = deparse(substitute(x))
                    , xlab = "Theoretical Quantiles"
                    , main = "Gamma Distribution QQ Plot",...)
{
    # Plot qq-plot for gamma distributed variable

    xx = x[!is.na(x)]
    aa = (mean(xx))^2 / var(xx)
    ss = var(xx) / mean(xx)
    test = rgamma(length(xx), shape = aa, scale = ss)
```

```
    qqplot(test, xx, xlab = xlab, ylab = ylab, main = main,...)
    abline(0,1, lty = 2)
}


qqGamma(Jan)
```




```
qqGamma(Jul)
```



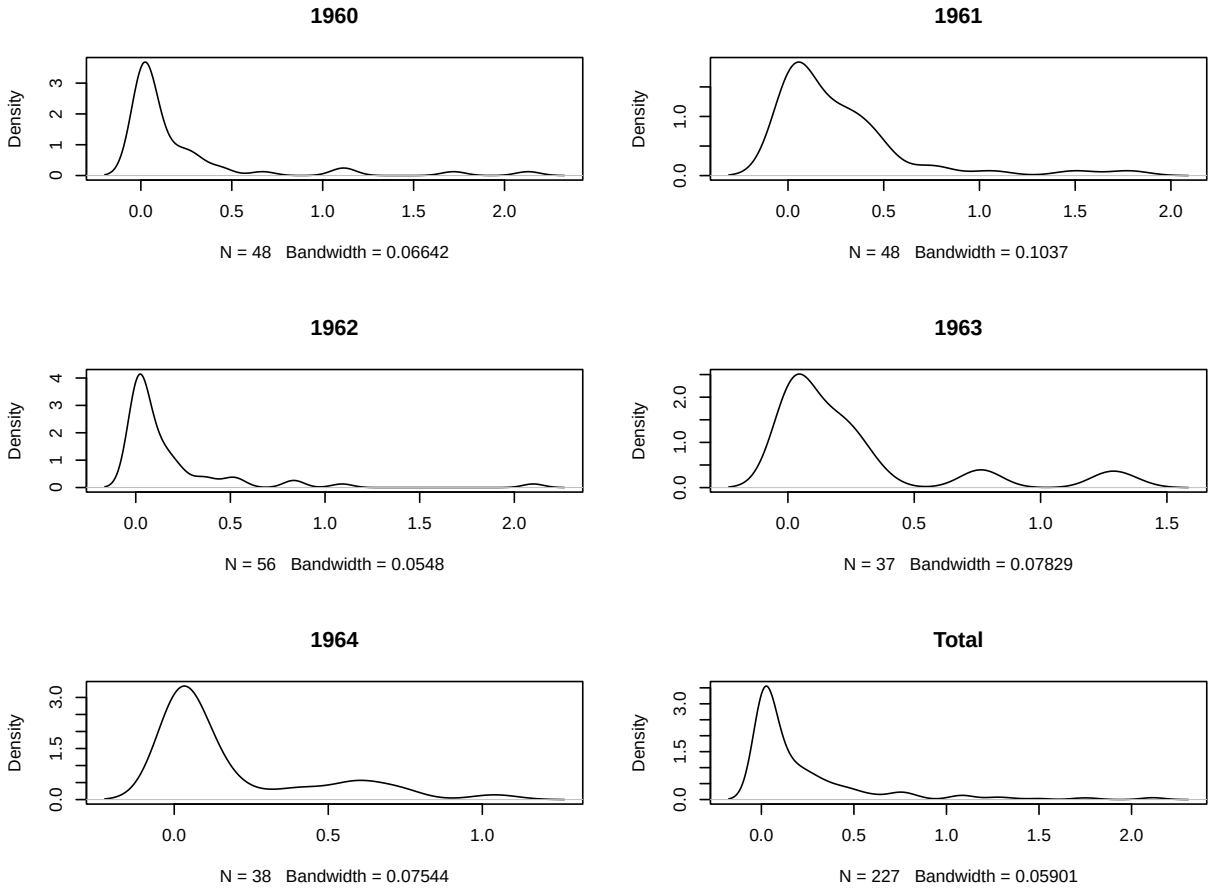
It seems that Jul is better.

Illinois rain

Question 3

Use the data to identify the distribution of rainfall produced by the storms in southern Illinois. Estimate the parameters of the distribution using MLE. Prepare a discussion of your estimation, including how confident you are about your identification of the distribution and the accuracy of your parameter estimates.

```
rain=read.xlsx('Illinois_rain_1960-1964.xlsx')
par(mfrow = c(3, 2))
density(rain$`1960` %>% na.omit()) %>% plot(main='1960')
density(rain$`1961` %>% na.omit()) %>% plot(main='1961')
density(rain$`1962` %>% na.omit()) %>% plot(main='1962')
density(rain$`1963` %>% na.omit()) %>% plot(main='1963')
density(rain$`1964` %>% na.omit()) %>% plot(main='1964')
density(unlist(rain) %>%  na.omit()) %>% plot(main='Total')
```



First I use the whole dataset to conduct fitdist. For method, MLE and MSE are selected to compare which method is better.

```
fit1<-fitdist(unlist(rain) %>%  na.omit() %>% c(),'gamma',method='mle') #MLE estimation
fit2<-fitdist(unlist(rain) %>%  na.omit() %>% c(),'gamma',method='mse') #MSE estimation
```

```
summary(bootdist(fit1)) #boot get confidence interval
summary(bootdist(fit2)) #boot get confidence interval
```

Table 1: MLE fit of Rain

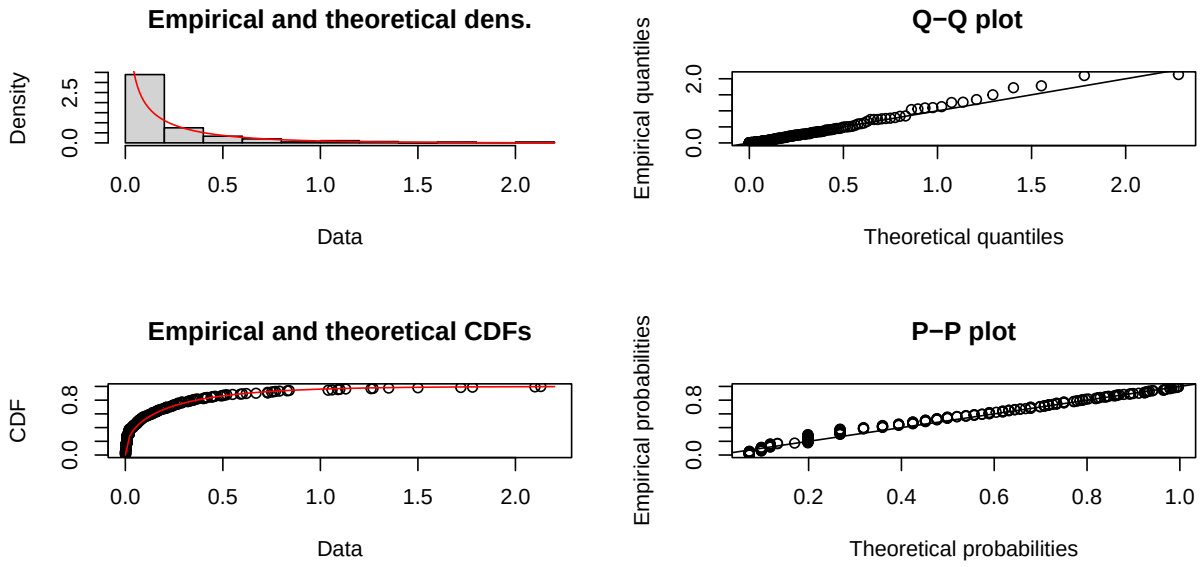
	Median	2.5%	97.5%
shape	0.4447568	0.3867703	0.521934
rate	1.9825172	1.5781062	2.567854

Table 2: MSE fit of Rain

	Median	2.5%	97.5%
shape	0.3917468	0.2726087	0.5300762
rate	1.7586151	1.1622105	2.5227109

The median and 95% confidence interval is shown in the above table. Compared to MSE, MSE's confidence interval is much narrower. Therefore, MLE fits the rain data better. The confidence interval indicates that the estimation is reliable.

```
plot(fit1)
```



Question 2

Using this distribution, identify wet years and dry years. Are the wet years wet because there were more storms, because individual storms produced more rain, or for both of these reasons?

```
rain_mean=fit1$estimate[1]/fit1$estimate[2] #get mean for whole dataset
re=apply(rain,2,mean,na.rm =TRUE) # get mean for each year

out<-c(re,rain_mean %>% as.numeric() %>% round(4))
names(out)[6]='mean'
#out

num_storm<-c(nrow(rain)-apply(is.na(rain),2,sum),'/')

knitr::kable(rbind(out,num_storm)) # show the result
```

	1960	1961	1962	1963	1964	mean
out	0.220291666666667	0.2749375	0.18475	0.262432432432432	0.187105263157895	0.2244
num_storm	48	48	56	37	38	/

Compared to mean, 1962, 1964 are dryer years, 1961 and 1963 are wetter years. 1960 is the normal year. We can also conclude that more storms don't necessarily result in wet year and more rain in individual storm don't necessarily result in wet year.
Therefore, both of these reasons have influence on amount of rainfall.

I also conduct fitdist on each individual year. The results are shown below:

	1960	1961	1962	1963	1964
mean	0.22032	0.27494	0.18475	0.26245	0.18713
num_storm	48	48	56	37	38

Question 3

To what extent do you believe the results of your analysis are generalizable? What do you think the next steps would be after the analysis? An article by Floyd Huff, one of the authors of the 1967 report is included.

Data is not enough, which is to say that 5 years of observations are not enough for the distribution verification. We can try to enlarge the data based on some API online or applied bootstrap or MCMN.