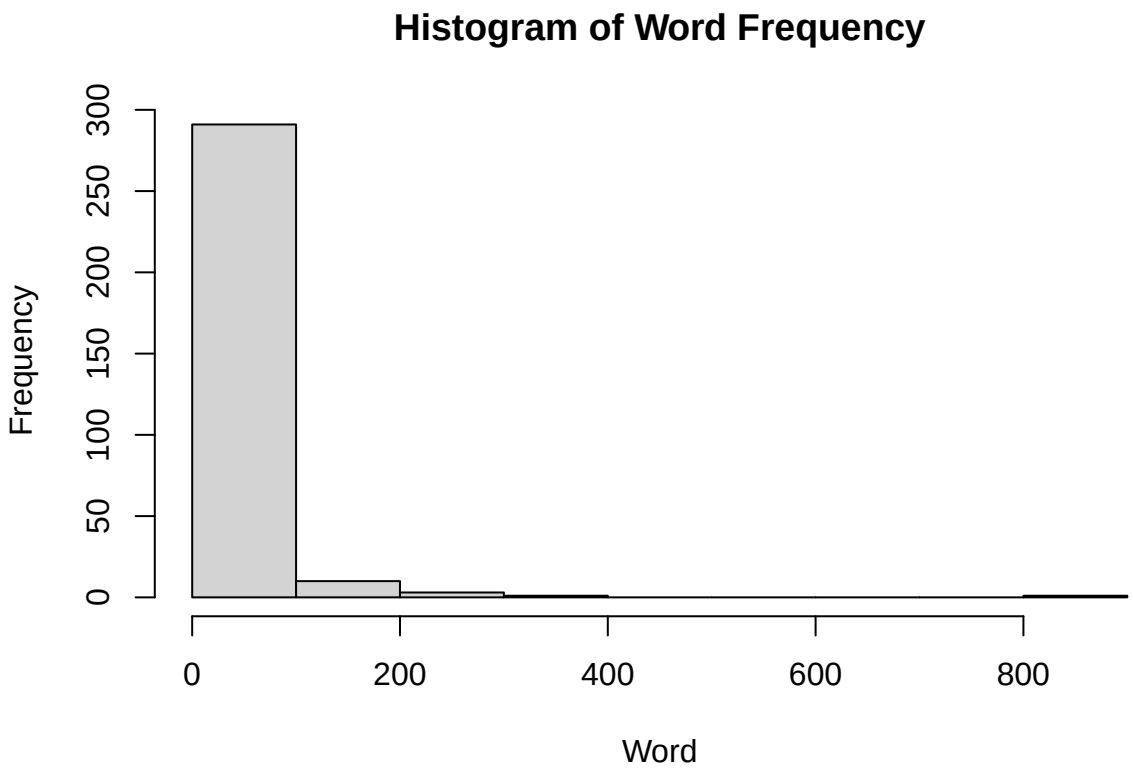
Fitting Power Law Distribution

This is an R Markdown Notebook. When you execute code within the notebook, the results appear beneath the code.

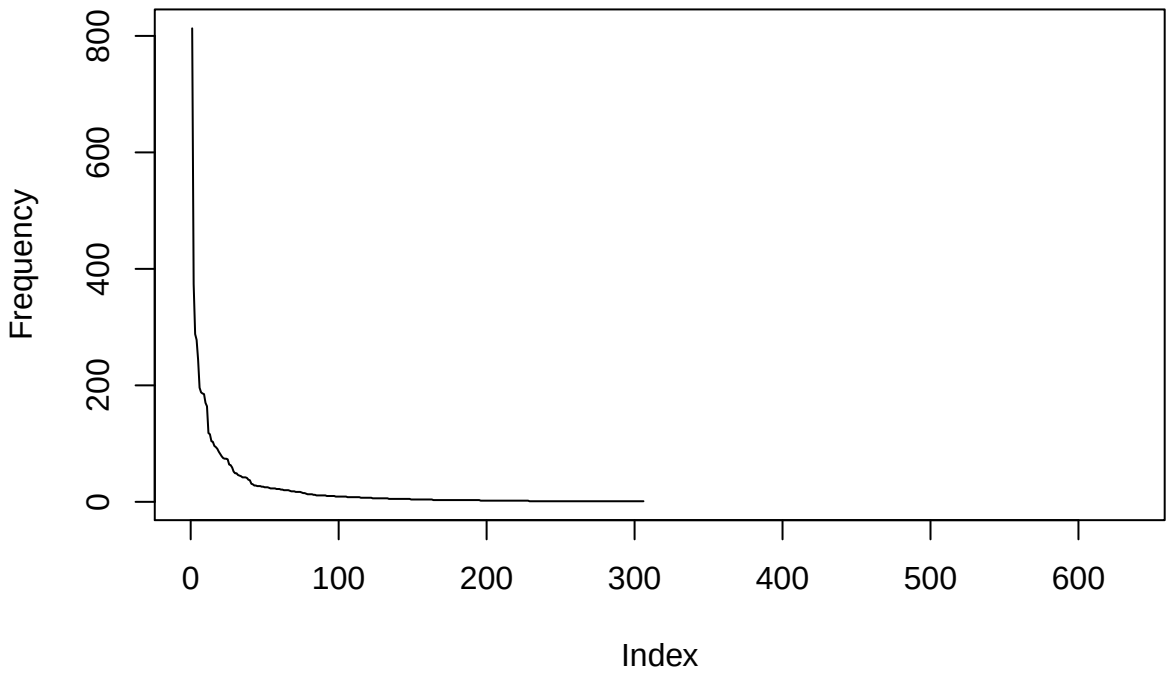
Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Ctrl+Shift+Enter*.

```
##Import Clean Data
data_url = "https://raw.githubusercontent.com/xiaomeng-ma/Tolerance-Principle/master/8x4.csv"
df <- read.csv(data_url, header = TRUE)
```

```
#Visualize Data
hist(df$Adam,xlab='Word',main='Histogram of Word Frequency')
```




```
plot(df$Adam,type='l',ylab='Frequency',main='Plot of Word Frequency')
```




```
#x is the frequency of each word
#y is the rank
x<-na.omit(df$Adam)
y<-seq.int(1:306)
```

```
library(poweRlaw)
m_pl <- displ$new(x)
est_pl <- estimate_xmin(m_pl)
est_pl$xmin #estimated x_min
```

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

```
est_pl$pars #estimated alpha
```

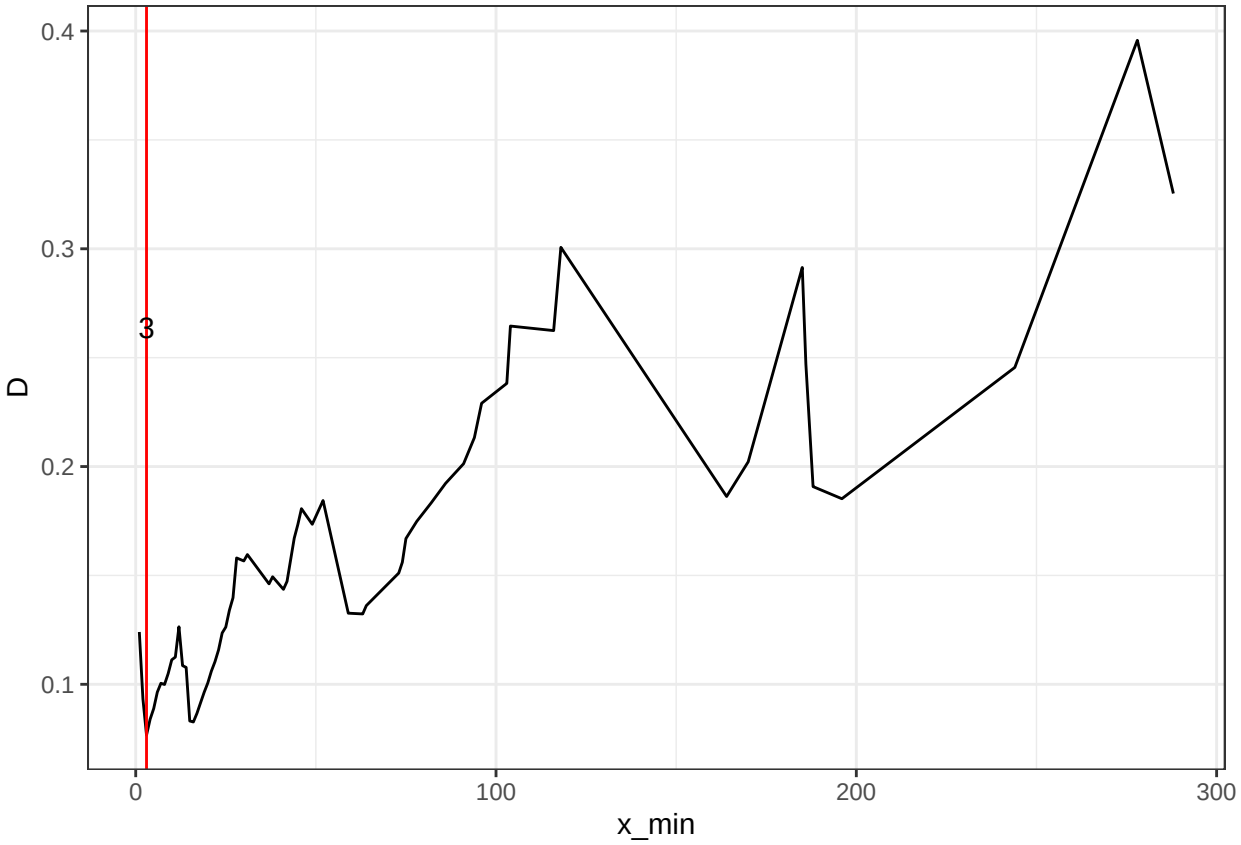
```
## [1] 1.621801
```

```
est_pl$gof #D(x_min), Kolmogorov-Smirnov statistics
```

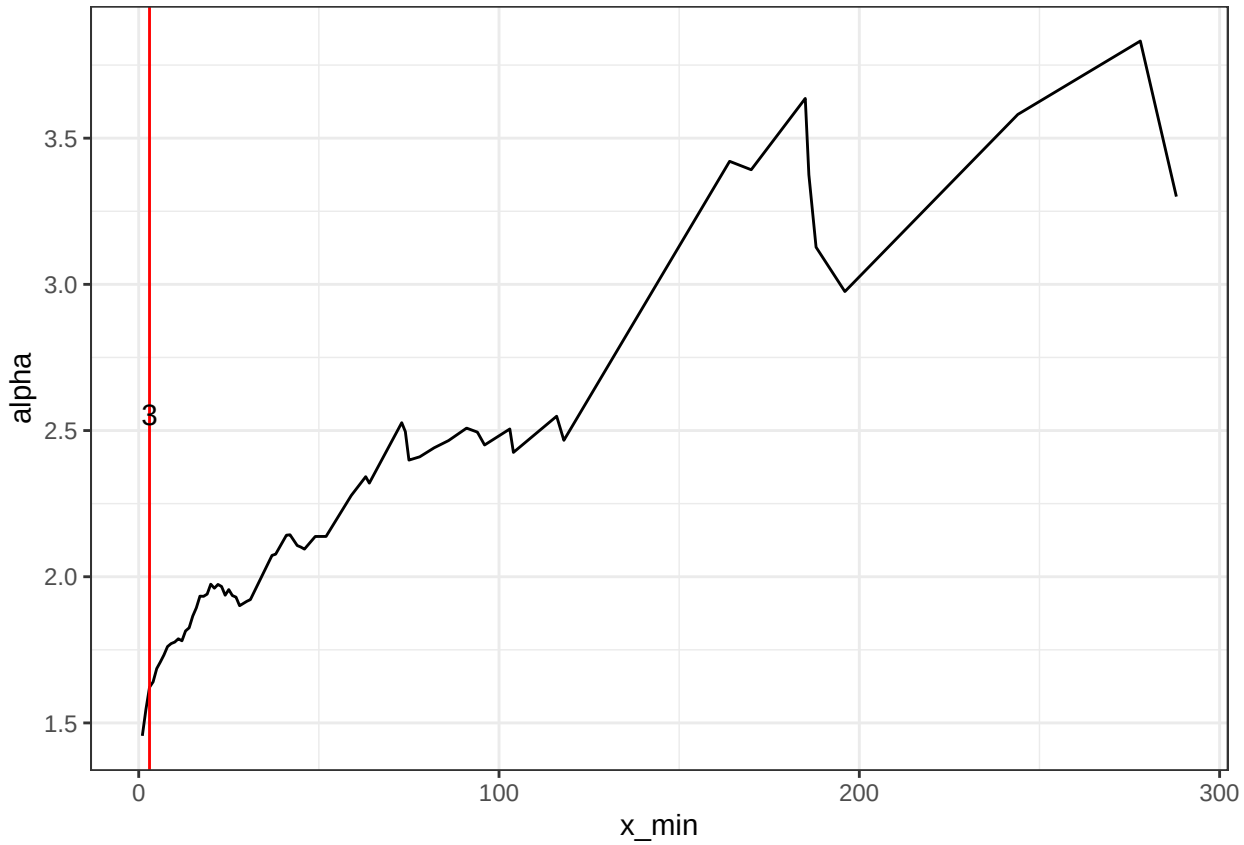
```
## [1] 0.07667388
```

```
##Scanning the whole range to make sure D is minimum
data.s <- unique(x)

d_est <- data.frame(x_min=sort(data.s)[1:(length(data.s)-2)], alpha=rep(0,length(data.s)-2), D=rep(0,le

for (i in d_est$x_min){
  d_est[which(d_est$x_min == i),2] <- estimate_xmin(m_pl, xmins = i)$pars
  d_est[which(d_est$x_min == i),3] <- estimate_xmin(m_pl, xmins = i)$gof
}

x.min_D.min <- d_est[which.min(d_est$D), 1]
```

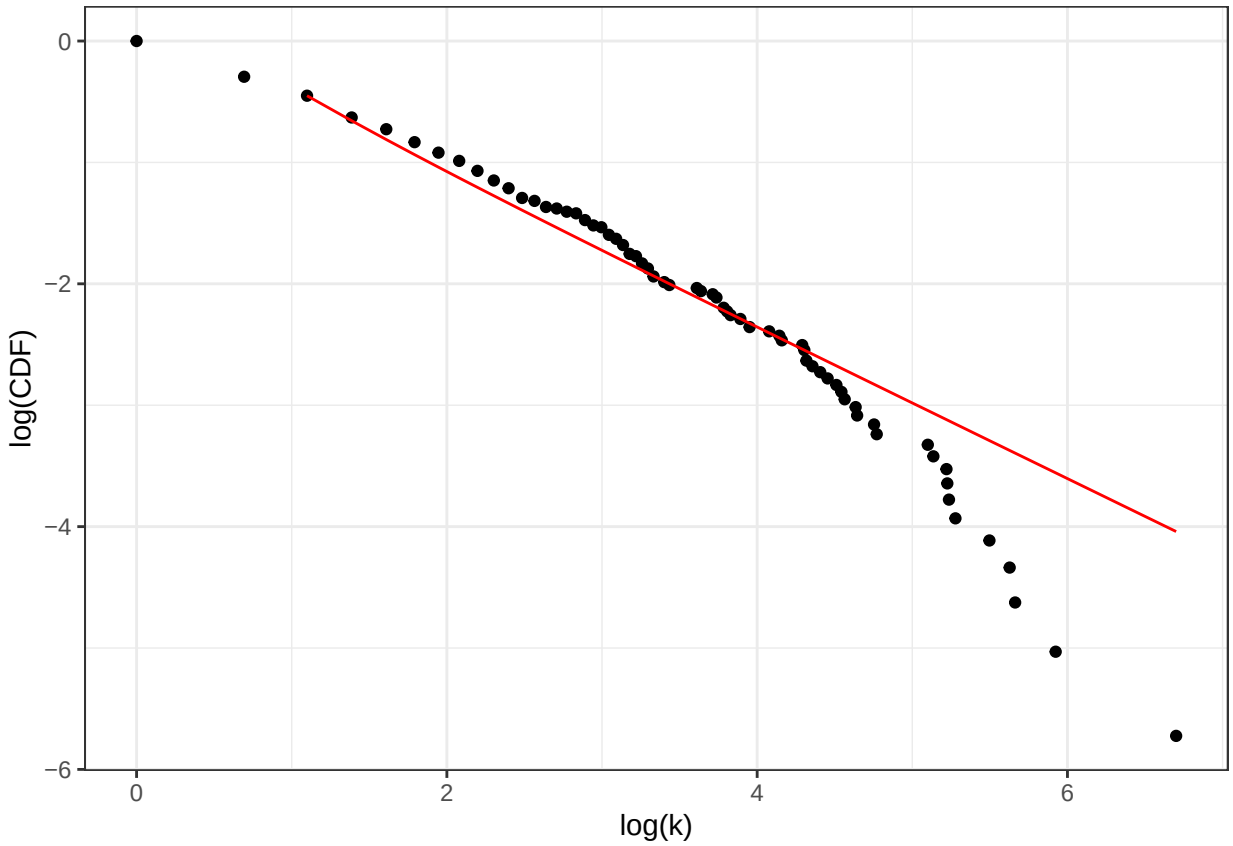
```
library(ggplot2)
ggplot(data=d_est, aes(x=x_min, y=D)) + geom_line() + theme_bw() +
  geom_vline(xintercept=x.min_D.min, colour="red") + annotate("text", x=x.min_D.min, y=max(d_est$D)/3*2
```




```
ggplot(data=d_est, aes(x=x_min, y=alpha)) + geom_line() + theme_bw() +
  geom_vline(xintercept=x.min_D.min, colour="red") + annotate("text", x=x.min_D.min, y=max(d_est$alpha),
```




```
##Fitting power-law on CDF curve
m_pl$setXmin(est_pl)
plot.data <-plot(m_pl, draw = F)
fit.data <-lines(m_pl, draw = F)
ggplot(plot.data) + geom_point(aes(x=log(x), y=log(y))) + labs(x="log(k)", y="log(CDF)") + theme_bw() +
  geom_line(data=fit.data, aes(x=log(x), y=log(y)), colour="red")
```




```
## get xmin and xmax pairs
pairs <- as.data.frame(t(combn(sort(data.s),2)))
pairs$D <- rep(0, length(pairs$V1))
pairs$alpha <- rep(0, length(pairs$V1))

## go through D for all xmin and xmax pairs
for (i in 1:length(pairs$D)){
  m_pl$setXmin((pairs[i,1]))
  pairs[i, 3]<-estimate_xmin(m_pl, xmin = pairs[i,1], xmax = pairs[i,2], distance = "ks")$gof
  pairs[i, 4]<-estimate_xmin(m_pl, xmin = pairs[i,1], xmax = pairs[i,2], distance = "ks")$pars
}
```

```
bs_pl_sat_cut <- bootstrap_p(m_pl, xmins = pairs[which.min(pairs$D), 1], xmax = pairs[which.min(pairs$D)
```

```
## Some of your data is larger than xmax. The xmax parameter is
##              the upper bound of the xmin search space. You could try increasing
##              it. If the estimated values are below xmax, it's probably OK not to
##              worry about this.
```

```
## Expected total run time for 20 sims, using 8 threads is 0.0196 seconds.
```

```
##get parameters
pairs[which.min(pairs$D), 1] #x_min
```

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

```
pairs[which.min(pairs$D), 2] #x_max
```

```
## [1] 196
```

```
pairs[which.min(pairs$D), 3] #D
```

```
## [1] 0.06671667
```

```
pairs[which.min(pairs$D), 4] #alpha
```

```
## [1] 1.621801
```

```
## p-value
bs_pl_sat_cut$p
```

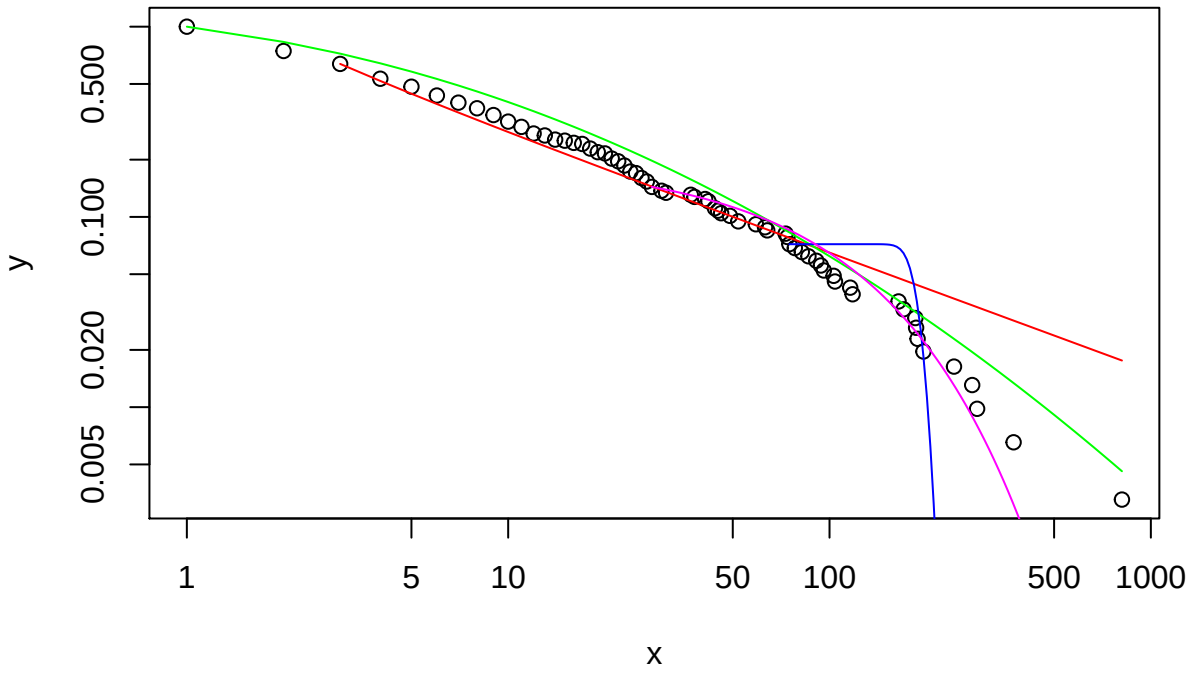
```
## [1] 0.85
```

```
## since the score is 1, based on the PoweRlaw package document, this means the power law model is a po
```

```
x_min <- 3
x_max <-196
alpha <-1.621801
```

```
##Compare 4 similar distributions

#powerlaw
m_pl = displ$new(x)
est_pl <- estimate_xmin(m_pl, xmins = x_min, xmax = x_max, distance = "ks")
m_pl$setXmin(est_pl)

#lognormal
m_ln = dislnorm$new(x)
est_ln <- estimate_xmin(m_ln)
m_ln$setXmin(est_ln)

#exponential
m_exp = disexp$new(x)
est_exp <- estimate_xmin(m_exp)
m_exp$setXmin(est_exp)


#poisson
m_poi = dispois$new(x)
est_poi <-estimate_xmin(m_poi)
m_poi$setXmin(est_poi)
```

```
plot(m_pl)
lines(m_pl, col="red")
lines(m_ln, col="green")
lines(m_poi, col="blue")
lines(m_exp, col="magenta")
```


y
0.500
0.100
0.020
0.005
1
5
10
50
100
500
1000
x