

R Code: ON A MODIFIED WEIGHTED EXPONENTIAL DISTRIBUTION WITH AN APPLICATION

1 R codes

1.1 Figure S1: Graphics of the pdf of the MWE distribution

```
al=0.1 # parameter of the distribution
lm= 0.5 # parameter of the distribution
x  <- seq(0,3, length=1000)
y1 <- (lm*exp(-lm*x))*(((al+2)/(al+1))-exp(-lm*al*x))
al=0.2
y2 <- (lm*exp(-lm*x))*(((al+2)/(al+1))-exp(-lm*al*x))
al=0.4
y3 <- (lm*exp(-lm*x))*(((al+2)/(al+1))-exp(-lm*al*x))
al=0.6
y4 <- (lm*exp(-lm*x))*(((al+2)/(al+1))-exp(-lm*al*x))
colors <- rainbow(4)
linetype <- c(1:4)
plotchar <- seq(18,18+4,1)
plot(x,y1, type="l", lwd=1.5,lty=linetype[1], col=colors[1], pch=plotchar[1],
xlim=c(0,3), ylim=c(0.15,0.45), xlab="x", ylab="pdf")#, main="", font.main=2,
cex.main=1)
lines(x,y2, type="l", lwd=1.5, lty=linetype[2], col=colors[2], pch=plotchar[2])
lines(x,y3, type="l", lwd=1.5, lty=linetype[3], col=colors[3], pch=plotchar[3])
lines(x,y4, type="l", lwd=1.5,lty=linetype[4], col=colors[4], pch=plotchar[4])
# First density graph
pdf1=legend(1,0.45, c("\u03B1=0.1,\u03BB=0.5","\u03B1=0.2,\u03BB=0.5",
"\u03B1=0.4,\u03BB=0.5","\u03B1=0.6,\u03BB=0.5"), col=c("#FF0000", "#80FF00",
"#00FFFF", "#8000FF"),lty=1:4, cex=0.8, bty="n",text.width=0.1)

# Similarly, we can draw for other PDFs.
```

1.2 Figure S2: Graphics of the hrf of the MWE distribution

```
al=0.1 # parameter of the distribution
lm= 0.5 # parameter of the distribution
x  <- seq(0,3, length=1000)
y1 <- lm-lm*al*((exp(-lm*al*x))/(al+2-exp(-lm*al*x))) al=0.2
y2 <- lm-lm*al*((exp(-lm*al*x))/(al+2-exp(-lm*al*x))) al=0.4
y3 <- lm-lm*al*((exp(-lm*al*x))/(al+2-exp(-lm*al*x))) al=0.6
```

```

y4 <- lm-lm*al*((exp(-lm*al*x))/(al+2-exp(-lm*al*x)))
colors <- rainbow(4)
linetype <- c(1:4)
plotchar <- seq(18,18+4,1)
plot(x,y1, type="l", lwd=1.5, lty=linetype[1], col=colors[1], pch=plotchar[1],
xlim=c(0,3), ylim=c(0.30,0.48), xlab="x", ylab="hrf")#, main="", font.main=2,
cex.main=1)
lines(x,y2, type="l", lwd=1.5, lty=linetype[2], col=colors[2], pch=plotchar[2])
lines(x,y3, type="l", lwd=1.5, lty=linetype[3], col=colors[3], pch=plotchar[3])
lines(x,y4,type="l", lwd=1.5, lty=linetype[4], col=colors[4], pch=plotchar[4])
# First hazard function
hrf1=legend(1.2,0.375, c("\u03B1=0.1,\u03BB=0.5","\u03B1=0.2,\u03BB=0.5",
"\u03B1=0.4,\u03BB=0.5","\u03B1=0.6,\u03BB=0.5"), col=c("#FF0000", "#80FF00",
"#00FFFF", "#8000FF"),lty=1:4, cex=0.7, bty="n",text.width=0.1)

# Similarly, we can draw for other hrfs.

```

1.3 Table S1: Numerical values of the mean, variance, skewness and kurtosis of the MWE distribution

```

al1=seq(0.2,1.4,0.2) # parameter of the distribution
lm=rep(0.5,7) # parameter of the distribution
# m1, m2, m3, m4 represent the first four raw moments.
m1=m2=m3=m4=array()
for (i in 1:length(al1)) {
  al=al1[i]
  m1[i]=(al*(al+3)+1)/(lm*((al+1)^2))
  m2[i]=(2*(al^3+4*al^2+5*al+1))/((lm^2)*((al+1)^3))
  m3[i]=(6*(al^4+5*al^3+9*al^2+7*al+1))/((lm^3)*((al+1)^4))
  m4[i]=(24*(al^5+6*al^4+14*al^3+16*al^2+9*al+1))/((lm^4)*((al+1)^5))}
var=m2-(m1^2)
gamma1=((m3-3*m2*m1+2*(m1^3))/(var^(3/2)))
beta2=((m4-4*m3*m1+6*m2*(m1^2)-3*(m1^4))/(var^2))
result1=cbind(lm,al1,m1,var,gamma1,beta2)
lm1=seq(0.4,1.6,0.2)
al=rep(1.2,7)
m1=m2=m3=m4=array()
for (i in 1:length(lm1)) {
  lm=lm1[i]
  m1[i]=(al*(al+3)+1)/(lm*((al+1)^2))
  m2[i]=(2*(al^3+4*al^2+5*al+1))/((lm^2)*((al+1)^3))
  m3[i]=(6*(al^4+5*al^3+9*al^2+7*al+1))/((lm^3)*((al+1)^4))
  m4[i]=(24*(al^5+6*al^4+14*al^3+16*al^2+9*al+1))/((lm^4)*((al+1)^5))}
var=m2-(m1^2)
gamma1=((m3-3*m2*m1+2*(m1^3))/(var^(3/2)))
beta2=((m4-4*m3*m1+6*m2*(m1^2)-3*(m1^4))/(var^2))
result2=cbind(al,lm1,m1,var,gamma1,beta2)
final_result=rbind(result1, result2)
final_result

```

1.4 Figure S3: Graphics of the (a) skewness and (b) kurtosis of the MWE distribution

```

# Install all the packages which are given below
install.packages(c("plot3D", "reshape2", "ggplot2", "ggpubr",
"colorspace", "RColorBrewer"))
library(plot3D)
library(reshape2)
library(ggplot2)
library(ggpubr)
library(colorspace)
library(RColorBrewer)
par(mar=c(2,2, 2, 2), mfrow=c(1,2), oma = c(1,1,1,1))
al1=seq(0.2,1.4,0.3) # parameter of the distribution
lm1=seq(0.2,1.4,0.3) # parameter of the distribution
# m1, m2, m3, m4 represent the first four raw moments respectively.
m1=m2=m3=m4 =var=skewness=kurtosis= matrix(data=NA, nrow=length(al1),
ncol=length(lm1))
for (i in 1:length(al1)) { for(j in 1:length(lm1)){
al=al1[i]
lm=lm1[j]
m1[i,j]=(al*(al+3)+1)/(lm*((al+1)^2))
m2[i,j]=(2*(al^3+4*al^2+5*al+1))/((lm^2)*((al+1)^3))
m3[i,j]=(6*(al^4+5*al^3+9*al^2+7*al+1))/((lm^3)*((al+1)^4))
m4[i,j]=(24*(al^5+6*al^4+14*al^3+16*al^2+9*al+1))/((lm^4)*((al+1)^5))
var[i,j]=m2[i,j]-(m1[i,j]^2)
skewness[i,j]=((m3[i,j]-3*m2[i,j]*m1[i,j]+2*(m1[i,j]^3))/(var[i,j]^(3/2)))
kurtosis[i,j]=((m4[i,j]-4*m3[i,j]*m1[i,j]+6*m2[i,j]*m1[i,j]^2
-3*(m1[i,j]^4))/(var[i,j]^2))}
# Skewness graph
skewness=persp3D(al1,lm1,skewness,theta=40, phi=0, axes=TRUE,scale=2,
box=TRUE, nticks=5, ticktype="detailed", xlab="\n\u03B1", ylab="\n\u03BB",
zlab="\nSkewness", main="", font.main=1,cex.main=1)
#col=brewer.pal(n = 5, name = "Set2")
# Kurtosis graph
abc=persp3D(al1,lm1,kurtosis,theta=40, phi=0, axes=TRUE,scale=2, box=TRUE,
nticks=5, ticktype="detailed", xlab="\n\u03B1", ylab="\n\u03BB",
zlab="\nKurtosis", main="", font.main=1, cex.main=1),
# col=brewer.pal(n = 5, name = "Set2")

```

1.5 Table S2: Biases and MSEs of the estimates for $\alpha = 0.5$ and $\lambda = 1$

```

rm(list=ls())# for clear previous data stored
install.packages("nleqslv")
library(nleqslv)
set.seed(1595) # for fixing the population
x=array() #population
#Store the values of estimated parametrs, bias and mse
al1=array() # estimated parameters
bt1=array() # estimated parameters
bias_al1=array()
bias_bt1=array()
mse_al1=array()
mse_bt1=array()
y=array()
N=10000 # population size

```

```

al=0.5 # parameter of the distribution wor
bt= 1# parameter of the distribution
lm=bt
z=zz=NULL # sample from he distribution main
f <- function(x) {
(lm*exp(-lm*x))*(((al+2)/(al+1))-exp(-lm*al*x))
}
nn=6000 # number of replications
# MH algorithm
x <- numeric(nn)
x[1] <- rchisq(1, df=1)
k <- 0
u <- runif(nn)
for (i in 2:nn) {
xt <- x[i-1]
y <- rchisq(1, df = xt)
num <- f(y) * dchisq(xt, df = y)
den <- f(xt) * dchisq(y, df = xt)
if (u[i] <= num/den) x[i] <- y else {
x[i] <- xt
k <- k+1
}
}
plot(x,type="l")
print(k)
prob=k/nn
prob
burnin=.20*nn
y=x[(burnin+1):nn]
n=50 # sample size
# storing all the samples
z=matrix(nrow=n,ncol=nn)
for(l in 1:nn){
z[,l]=sample(y,n)
}
####MOM####
fn <- function(tt) {
al=tt[1]
lm=tt[2]
rate <- (al*(al+3)+1)/(lm*((al+1)^2)) - mean(x)
shape <- (2*(al^3+4*al^2+5*al+1))/((lm^2)*((al+1)^3))- mean(x^2)
return(c(rate, shape))
}
x=array()
for(j in 1:nn){
x=z[,j] #sample from pop (x)
k=nleqslv(c(0.2,1), fn, method = "Newton")$x
al1[j]=k[1] # estimates of alpha
bt1[j]=k[2] # estimates of beta
}
# avg (final) estimates
al1=al1[al1>0]
length(al1)
al1=sample(al1,5000)

```

```

mean(al1)
mean(bt1)
bt1=bt1[bt1>0]
length(bt1)
bl1=sample(bt1,5000)
bias_al1=(al1-al)
bias_bt1=(bt1-bt)
# mse of all estimators
mse_al1=(al1-al)^2
mse_bt1=(bt1-bt)^2
# avg (final) bias
mean(bias_al1)
mean(bias_bt1)
# avg (final) e mse
mean(mse_al1)
mean(mse_bt1)
# for Tabular form of all estimates , bias and mse
s1=cbind(mean(al1), mean(bt1)) # for estimates
s2=cbind(mean(bias_al1),mean(bias_bt1)) # for bias
s3=cbind(mean(mse_al1), mean(mse_bt1)) # for mse
s4=cbind(al,bt) # for true values of par
s_l1l1=cbind(s2,s3) # for final results
s_l1lp=cbind(s4,s4)
s_mme=rbind(s_l1l1) # for final results
s_mme
####MLE####
fn <- function(tp) {
al=tp[1]
lm=tp[2]
rate=(n/lm)-sum(x)+sum((1/((((al+2)/(al+1))-exp(-lm*al*x))))*(exp(-lm*al*x)*al*x))
shape=sum((1/((((al+2)/(al+1))-exp(-lm*al*x))))*(-(1/(al+1)^2)+exp(-lm*al*x)*lm*x))
return(c(rate, shape))
}
x=array()
for(j in 1:nn){
x=z[,j] #sample from pop (x)
k=nleqslv(c(0.5,1), fn)$x
k # all estimates
al1[j]=k[1] # estimates of alpha
bt1[j]=k[2] # estimates of beta
# bias of all estimators
bias_al1[j]=(al1[j]-al)
bias_bt1[j]=(bt1[j]-bt)
# mse of all estimators
mse_al1[j]=(al1[j]-al)^2
mse_bt1[j]=(bt1[j]-bt)^2
}
# avg (final) estimates
al1=al1[al1>0]
length(al1)
al1=sample(al1,5000)
mean(al1)
mean(bt1)
bt1=bt1[bt1>0]

```

```

length(bt1)
bl1=sample(bt1,5000)
bias_al1=(al1-al)
bias_bt1=(bt1-bt)
# mse of all estimators
mse_al1=(al1-al)^2
mse_bt1=(bt1-bt)^2
# avg (final) bias
mean(bias_al1)
mean(bias_bt1)
# avg (final) e mse
mean(mse_al1)
mean(mse_bt1)
# for Tabular form of all estimates , bias and mse
s1=cbind(mean(al1), mean(bt1)) # for estimates
s2=cbind(mean(bias_al1),mean(bias_bt1)) # for bias
s3=cbind(mean(mse_al1), mean(mse_bt1)) # for mse
s4=cbind(al,bt) # for true values of par
s_lll=cbind(s2,s3) # for final results
s_llp=cbind(s4,s4)
s_ml=rbind(s_lll) # for final results
s_ml
####OLS####
par=c(1,1)
TRTGLE=function(par){
al=par[1]
bt=par[2]
lm=bt
i=seq(1:n)
l=sum((((1-((1/(al+1)))*exp(-lm*x)*(al+2-exp(-lm*al*x))))-(i/(n+1))))^2) ####56##OLS##
}
x=array()
for(j in 1:nn){
x=z[,j]
x=sort(x) # order statistics
k=optim(par,TRTGLE, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf, Inf))$par
k
al1[j]=k[1]
bt1[j]=k[2]
bias_al1[j]=(al1[j]-al)
bias_bt1[j]=(bt1[j]-bt)
mse_al1[j]=(al1[j]-al)^2
mse_bt1[j]=(bt1[j]-bt)^2
}
mean(al1)
mean(bt1)
mean(bias_al1)
mean(bias_bt1)
mean(mse_al1)
mean(mse_bt1)
s1=cbind(mean(al1), mean(bt1))
s2=cbind(mean(bias_al1),mean(bias_bt1))
s3=cbind(mean(mse_al1), mean(mse_bt1))

```

```

s4=cbind(al,bt)
s_ols=cbind(s2,s3)
s_ols
#####WLS#####
TRTGLE=function(par){
al=par[1]
bt=par[2]
lm=bt
i=seq(1:n)
l=sum((((n+1)^2*(n+2))/(i*(n-i+1)))*(((1-((1/(al+1))*exp(-lm*x)*(al+2-exp(-lm*al*x))))))
}
x=array()
for(j in 1:nn){
x=z[,j]
x=sort(x)
k=optim(par,TRTGLE, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf, Inf))$par
k
al1[j]=k[1]
bt1[j]=k[2]
bias_al1[j]=(al1[j]-al)
bias_bt1[j]=(bt1[j]-bt)
mse_al1[j]=(al1[j]-al)^2
mse_bt1[j]=(bt1[j]-bt)^2
}
mean(al1)
mean(bt1)
mean(bias_al1)
mean(bias_bt1)
mean(mse_al1)
mean(mse_bt1)
s1=cbind(mean(al1), mean(bt1))
s2=cbind(mean(bias_al1),mean(bias_bt1))
s3=cbind(mean(mse_al1), mean(mse_bt1))
s4=cbind(al,bt)
s_wls=cbind(s2,s3)
s_wls
#####CME#####
TRTGLE=function(par){
al=par[1]
bt=par[2]
lm=bt
i=seq(1:n)
l=(1/(12*n))+sum((((1-((1/(al+1))*exp(-lm*x)*(al+2-exp(-lm*al*x)))))-((2*i-1)/(2*n))))^2
}
x=array()
for(j in 1:nn){
x=z[,j]
x=sort(x)
k=optim(par,TRTGLE, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf, Inf))$par
k
al1[j]=k[1]
bt1[j]=k[2]
bias_al1[j]=(al1[j]-al)
bias_bt1[j]=(bt1[j]-bt)

```

```

mse_al1[j]=(al1[j]-al)^2
mse_bt1[j]=(bt1[j]-bt)^2
}
mean(al1)
mean(bt1)
mean(bias_al1)
mean(bias_bt1)
mean(mse_al1)
mean(mse_bt1)
s1=cbind(mean(al1), mean(bt1))
s2=cbind(mean(bias_al1),mean(bias_bt1))
s3=cbind(mean(mse_al1), mean(mse_bt1))
s4=cbind(al,bt)
s_cme=cbind(s2,s3)
s_cme
s_name=c("bias(al)", "bias(lm)", "mse(al)", "mse(lm)")
s_true=c(al,lm,al,lm)
#all the answer in tabular form
simu=rbind(s_ml,s_mme,s_ols,s_wls,s_cme)
simu
# Similarly, we can calculate Table 4 and Table 5

```

1.6 Table S6: MLEs (standard errors) for the data set 1

```

install.packages("nleqslv")
library(nleqslv)
# Data set 1
x=c(0.12,0.43,0.92,1.14,1.24,1.61,1.93,2.38,4.51,5.09,6.79,7.64,8.45,11.9,11.94,
13.01,13.25,14.32,17.47,18.1,18.66,19.23,24.39,25.01,26.41,26.8,27.75,29.69,29.84,
31.65,32.64,35,40.7,42.34,43.05,43.4,44.36,45.4,48.14,49.1,49.44,51.17,58.62,60.29,
72.13,72.22,72.25,72.29,85.2,89.52)
n=length(x)
###MWE
TRTGLE=function(par){
al=par[1]
lm=par[2]
l=-(n*log(lm)-lm*sum(x)+sum(log((((al+2)/(al+1))-exp(-lm*al*x))))))
# likelihood function
par=c(.01,.01) # initial guess
k=optim(par,TRTGLE, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
s_e = sqrt(abs(diag(solve(-k$hessian)))) # standard error
k=k$par
al=k[1]
lm=k[2]
# Parameter values and their standard errors
s_tr=cbind(k[1],s_e[1],k[2],s_e[2])
####WE
TRTW=function(par){
al=par[1]
lm=par[2]
l=-(-n*log(al)+n*log(1+al)+n*log(lm)-lm*sum(x)+sum(log(1-exp(-lm*al*x))))
par=c(.01,.01)
kk_on=optim(par,TRTW, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)

```

```

s_eon = sqrt(abs(diag(solve(-kk_on$hessian))))
kk_on=kk_on$par
al1=kk_on[1]
lm1=kk_on[2]
#Parameter values and their standard errors
s_on=cbind(kk_on[1],s_eon[1],kk_on[2],s_eon[2])
### Weibull(W) distribution
W=function(par){
mu=par[1]
s=par[2]
l=-(n*log(mu)-n*log(s)+(mu-1)*sum(log((x/s)))-sum((x/s)^(mu)))}
par=c(1.5,1.5)
kk_wl=optim(par,W, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
s_ewl = sqrt(abs(diag(solve(-kk_wl$hessian))))
kk_wl=kk_wl$par
mu1=kk_wl[1]
s1=kk_wl[2]
# Parameter values and their standard errors
s_w=cbind(kk_wl[1],s_ewl[1],kk_wl[2],s_ewl[2])
###Gamma
Wg=function(par){
al=par[1]
bt=par[2]
l=-(n*al*log(bt)-n*log(factorial(al-1))+(al-1)*sum(log(x))-(1/bt)*sum((x)))}
par=c(.1,.9)
kk_g=optim(par,Wg, method = "L-BFGS-B",lower=c(0,0), upper=c(Inf, Inf),hessian=T)
s_g = sqrt(abs(diag(solve(-kk_g$hessian))))
kk_g=kk_g$par
al1=kk_g[1]
bt1=kk_g[2]
#Parameter values and their standard errors
s_gg=cbind(kk_g[1],s_g[1],kk_g[2],s_g[2])
###GE distribution
W_ge=function(par){
al=par[1]
bt=par[2]
l=-(n*log(al)+n*log(bt)+(al-1)*sum(log(1-exp(-bt*x)))-bt*sum(x))}
par=c(1.2,.01)
kk_ge=optim(par,W_ge, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
s_ge = sqrt(abs(diag(solve(-kk_ge$hessian))))
kk_ge=kk_ge$par
al1=kk_ge[1]
bt1=kk_ge[2]
#Parameter values and their standard errors
s_ge=cbind(kk_ge[1],s_ge[1],kk_ge[2],s_ge[2])
mle_par=cbind(s_tr,s_on,s_w,s_gg,s_ge)
mle_par

```

1.7 Table S7: The parameter estimates for the data set 1

```

install.packages("nleqslv")
library(nleqslv)
x=c(0.12,0.43,0.92,1.14,1.24,1.61,1.93,2.38,4.51,5.09,6.79,7.64,8.45,11.9,11.94,

```

```

13.01,13.25,14.32,17.47,18.1,18.66,19.23,24.39,25.01,26.41,26.8,27.75,29.69,29.84,
31.65,32.64,35,40.7,42.34,43.05,43.4,44.36,45.4,48.14,49.1,49.44,51.17,58.62,60.29,
72.13,72.22,72.25,72.29,85.2,89.52)
n=length(x)
####MOM####
fn <- function(tt) {
  al=tt[1]
  lm=tt[2]
  rate <- (al*(al+3)+1)/(lm*((al+1)^2)) - mean(x)
  shape <- (2*(al^3+4*al^2+5*al+1))/((lm^2)*((al+1)^3))- mean(x^2)
  return(c(rate, shape))}
k1=nleqslv(c(10,.1), fn, method = "Newton")$x
k1 # all estimates
#if (any(k < 0)) return (j=j)
al1=k1[1] # estimates of alpha
bt1=k1[2] # estimates of beta
####OLS####
par=c(1,1)
TRTGLE3=function(par){
  al=par[1]
  bt=par[2]
  lm=bt
  i=seq(1:n)
  l=sum((((1-((1/(al+1))*exp(-lm*x)*(al+2-exp(-lm*al*x)))))-(i/(n+1)))^2))
  x=sort(x)
  k3=optim(par,TRTGLE3, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf, Inf))$par
####WLS####
TRTGLE4=function(par){
  al=par[1]
  bt=par[2]
  lm=bt
  i=seq(1:n)
  l=sum((((n+1)^2*(n+2))/(i*(n-i+1)))*(((1-((1/(al+1))*exp(-lm*x)*(al+2-exp(-lm*al*x)))))-(i/(n+1)))^2))
  x=sort(x)
  k4=optim(par,TRTGLE4, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf, Inf))$par
####CME####
TRTGLE5=function(par){
  al=par[1]
  bt=par[2]
  lm=bt
  i=seq(1:n)
  l=(1/(12*n))+sum((((1-((1/(al+1))*exp(-lm*x)*(al+2-exp(-lm*al*x)))))-(2*i-1)/(2*n)))^2))
  k5=optim(par,TRTGLE5, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf, Inf))$par
  result=cbind(k1[1],k1[2],k3[1],k3[2],k4[1],k4[2],k5[1],k5[2])
  result

```

1.8 Table S5: Selection criteria statistics of the models for the data set 1

```

install.packages(c("nleqslv","gofstest"))
library(nleqslv)
library(gofstest)
x=c(0.12,0.43,0.92,1.14,1.24,1.61,1.93,2.38,4.51,5.09,6.79,7.64,8.45,11.9,11.94,
13.01,13.25,14.32,17.47,18.1,18.66,19.23,24.39,25.01,26.41,26.8,27.75,29.69,29.84,

```

```

31.65,32.64,35,40.7,42.34,43.05,43.4,44.36,45.4,48.14,49.1,49.44,51.17,58.62,60.29,
72.13,72.22,72.25,72.29,85.2,89.52)
n=length(x)
#####MWE#####
TRTGLE=function(par){
  al=par[1]
  lm=par[2]
  l=-(n*log(lm)-lm*sum(x)+sum(log((((al+2)/(al+1))-exp(-lm*al*x))))))
  par=c(.01,.01)
  k=optim(par,TRTGLE, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
  s_e = sqrt(abs(diag(solve(-k$hessian))))
  k=k$par
  al=k[1]
  lm=k[2]
  like=-TRTGLE(k)
  ll=-2*like
  AIC=ll+2*2
  s_tr=cbind(k[1],s_e[1],k[2],s_e[2])
  F=function(x) (1-((1/(al+1))*exp(-lm*x)*(al+2-exp(-lm*al*x))))
  ztr_ks=ks.test(x, F)
  ztr_cvm=cvm.test(x, F)
  ztr_ad=ad.test(x, F)
  z_tr=cbind(ll,AIC,ztr_ks$statistic,ztr_ks$p.value,ztr_cvm$statistic
,ztr_cvm$p.value,ztr_ad$statistic,ztr_ad$p.value)
#####WE#####
TRTW=function(par){
  al=par[1]
  lm=par[2]
  l=-(-n*log(al)+n*log(1+al)+n*log(lm)-lm*sum(x)+sum(log(1-exp(-lm*al*x))))
  par=c(.01,.01)
  kk_on=optim(par,TRTW, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
  s_eon = sqrt(abs(diag(solve(-kk_on$hessian))))
  kk_on=kk_on$par
  al1=kk_on[1]
  lm1=kk_on[2]
  like_on=-TRTW(kk_on)
  ll_on=-2*like_on
  AIC_on=ll_on+2*2
  s_on=cbind(kk_on[1],s_eon[1],kk_on[2],s_eon[2])
  F=function(x) (((al1+1)/al1)*(1-exp(-lm1*x))-(1/(al1+1))*(1-exp(-lm1*x*(1+al1))))
  zon_ks=ks.test(x, F)
  zon_cvm=cvm.test(x, F)
  zon_ad=ad.test(x, F)
  z_on=cbind(ll_on,AIC_on,zon_ks$statistic,zon_ks$p.value,zon_cvm$statistic
,zon_cvm$p.value,zon_ad$statistic,zon_ad$p.value)
#####W#####
W=function(par){
  mu=par[1]
  s=par[2]
  l=-(n*log(mu)-n*log(s)+(mu-1)*sum(log((x/s)))-sum((x/s)^(mu)))
  par=c(1.5,1.5)
  kk_wl=optim(par,W, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
  s_ewl = sqrt(abs(diag(solve(-kk_wl$hessian))))
  kk_wl=kk_wl$par

```

```

mu1=kk_wl[1]
s1=kk_wl[2]
like_w=-W(kk_wl)
ll_w=-2*like_w
AIC_w=ll_w+2*2
s_w=cbind(kk_wl[1],s_ewl[1],kk_wl[2],s_ewl[2])
F=function(x) (1-exp(-(x/s1)^(mu1)))
zwl_ks=ks.test(x, F)
zwl_cvm=cvm.test(x, F)
zwl_ad=ad.test(x, F)
z_wl=cbind(ll_w,AIC_w,zwl_ks$statistic,zwl_ks$p.value,zwl_cvm$statistic
,zwl_cvm$p.value,zwl_ad$statistic,zwl_ad$p.value)
#####Gamma#####
Wg=function(par){
al=par[1]
bt=par[2]
l=-(-n*al*log(bt)-n*log(factorial(al-1))+(al-1)*sum(log(x))-(1/bt)*sum((x)))}
par=c(.1,.9)
kk_g=optim(par,Wg, method = "L-BFGS-B",lower=c(0,0), upper=c(Inf, Inf),hessian=T)
s_g = sqrt(abs(diag(solve(-kk_g$hessian))))
kk_g=kk_g$par
al1=kk_g[1]
bt1=kk_g[2]
like_g=-Wg(kk_g)
ll_g=-2*like_g
AIC_g=ll_g+2*2
s_gg=cbind(kk_g[1],s_g[1],kk_g[2],s_g[2])
bt2=1/bt1
F=function(x) (pgamma(x,al1,bt2))
zg_ks=ks.test(x, F)
zg_cvm=cvm.test(x, F)
zg_ad=ad.test(x, F)
z_g=cbind(ll_g,AIC_g,zg_ks$statistic,zg_ks$p.value,zg_cvm$statistic
,zg_cvm$p.value,zg_ad$statistic,zg_ad$p.value)
#####GE#####
W_ge=function(par){
al=par[1]
bt=par[2]
l=-((n*log(al)+n*log(bt)+(al-1)*sum(log(1-exp(-bt*x)))-bt*sum(x)))}
par=c(1.2,.01)
kk_ge=optim(par,W_ge, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
s_ge = sqrt(abs(diag(solve(-kk_ge$hessian))))
kk_ge=kk_ge$par
al1=kk_ge[1]
bt1=kk_ge[2]
like_ge=-W_ge(kk_ge)
ll_ge=-2*like_ge
AIC_ge=ll_ge+2*2
s_ge=cbind(kk_ge[1],s_ge[1],kk_ge[2],s_ge[2])
F=function(x) ((1-exp(-bt1*x))^al1)
zge_ks=ks.test(x, F)
zge_cvm=cvm.test(x, F)
zge_ad=ad.test(x, F)
z_ge=cbind(ll_ge,AIC_ge,zge_ks$statistic,zge_ks$p.value,zge_cvm$statistic

```

```
,zge_cvm$p.value,zge_ad$statistic,zge_ad$p.value)
overall=rbind(z_tr,z_on,z_wl, z_g, z_ge) #z_trp1,z_wlp,
final=round(overall,6)
final
```

1.9 Figure S5: The histogram with the fitted pdfs for the data set 1

```
x=c(0.12,0.43,0.92,1.14,1.24,1.61,1.93,2.38,4.51,5.09,6.79,7.64,8.45,11.9,11.94,
13.01,13.25,14.32,17.47,18.1,18.66,19.23,24.39,25.01,26.41,26.8,27.75,29.69,29.84,
31.65,32.64,35,40.7,42.34,43.05,43.4,44.36,45.4,48.14,49.1,49.44,51.17,58.62,60.29,
72.13,72.22,72.25,72.29,85.2,89.52)
hist(x, prob = T, xlab = "x",xlim = c(min(x),65),ylim = c(0,0.05),ylab="pdf",
main = "", breaks = 25, font.main=6) #main = "Fitted pdfs", font.main=1
n=length(x)
####MWE####
TRTGLE=function(par){
al=par[1]
lm=par[2]
l=-(n*log(lm)-lm*sum(x)+sum(log((((al+2)/(al+1))-exp(-lm*al*x))))))
par=c(.01,.01)
k=optim(par,TRTGLE, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
s_e = sqrt(abs(diag(solve(-k$hessian))))
k=k$par
al=k[1]
lm=k[2]
like=-TRTGLE(k)
ll=-2*like
AIC=ll+2*2
s_tr=cbind(k[1],s_e[1],k[2],s_e[2])
F=function(x) (lm*exp(-lm*x))*((((al+2)/(al+1))-exp(-lm*al*x)))
curve(F(x), lwd = 1,col = "Brown", add=TRUE)
####WE####
TRTW=function(par){
al=par[1]
lm=par[2]
l=-(n*log(al)+n*log(1+al)+n*log(lm)-lm*sum(x)+sum(log(1-exp(-lm*al*x))))
par=c(.01,.01)
kk_on=optim(par,TRTW, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
s_eon = sqrt(abs(diag(solve(-kk_on$hessian))))
kk_on=kk_on$par
al=kk_on[1]
lm=kk_on[2]
like_on=-TRTW(kk_on)
ll_on=-2*like_on
AIC_on=ll_on+2*2
s_on=cbind(kk_on[1],s_eon[1],kk_on[2],s_eon[2])
F=function(x) (((al+1)/al)*lm*exp(-lm*x)*(1-exp(-lm*x*al)))
curve(F(x), lwd = 1, col = "orange", add=TRUE)
####W####
W=function(par){
mu=par[1]
s=par[2]
l=-(n*log(mu)-n*log(s)+(mu-1)*sum(log((x/s)))-sum((x/s)^(mu)))}
```

```

par=c(.8,6.9)
kk_wl=optim(par,W, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
s_ewl = sqrt(abs(diag(solve(-kk_wl$hessian))))
kk_wl=kk_wl$par
mu1=kk_wl[1]
s1=kk_wl[2]
like_w=-W(kk_wl)
ll_w=-2*like_w
AIC_w=ll_w+2*2
s_w=cbind(kk_wl[1],s_ewl[1],kk_wl[2],s_ewl[2])
F=function(x) (mu1/s1)*((x/s1)^(mu1-1))*exp(-(x/s1)^(mu1))
curve(F(x),lwd = 1, col = "orchid1", add=TRUE)
####Gamma####
Wg=function(par){
al=par[1]
bt=par[2]
l=-(-n*al*log(bt)-n*log(factorial(al-1))+(al-1)*sum(log(x))-(1/bt)*sum((x)))}
par=c(.1,.9)
kk_g=optim(par,Wg, method = "L-BFGS-B",lower=c(0,0), upper=c(Inf, Inf),hessian=T)
s_g = sqrt(abs(diag(solve(-kk_g$hessian))))
kk_g=kk_g$par
al1=kk_g[1]
bt1=kk_g[2]
like_g=-Wg(kk_g)
ll_g=-2*like_g
AIC_g=ll_g+2*2
s_gg=cbind(kk_g[1],s_g[1],kk_g[2],s_g[2])
F=function(x) (1/(bt1^al1*(1/factorial(al1-1))))*x^(al1-1)*exp(-(x/bt1)))
curve(F(x), lwd = 1, col = "blue", add=TRUE)
####GE####
W_ge=function(par){
al=par[1]
bt=par[2]
l=-((n*log(al)+n*log(bt)+(al-1)*sum(log(1-exp(-bt*x)))-bt*sum(x)))}
par=c(.1,.01)
kk_ge=optim(par,W_ge, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
s_ge = sqrt(abs(diag(solve(-kk_ge$hessian))))
kk_ge=kk_ge$par
al=kk_ge[1]
bt=kk_ge[2]
like_ge=-W_ge(kk_ge)
ll_ge=-2*like_ge
AIC_ge=ll_ge+2*2
s_ge=cbind(kk_ge[1],s_ge[1],kk_ge[2],s_ge[2])
F=function(x) (al*bt*((1-exp(-bt*x))^(al-1))*exp(-bt*x))
curve(F(x), lwd = 1, col = "red", add=TRUE)
legend('topright',c("MWE","W","G","GE","WE"), xpd=TRUE,hORIZ=FALSE,
col=c("Brown","red","orchid1","blue","orange"),lty=c(1,1,1,1,1),cex=0.8)
#bty = "n"
box()

```

1.10 Figure S6: The empirical cdf with the fitted cdf for the data set 1

```

x=c(0.12,0.43,0.92,1.14,1.24,1.61,1.93,2.38,4.51,5.09,6.79,7.64,8.45,11.9,11.94,
13.01,13.25,14.32,17.47,18.1,18.66,19.23,24.39,25.01,26.41,26.8,27.75,29.69,29.84,
31.65,32.64,35,40.7,42.34,43.05,43.4,44.36,45.4,48.14,49.1,49.44,51.17,58.62,60.29,
72.13,72.22,72.25,72.29,85.2,89.52)
n=length(x)
Fn <- ecdf(x)
plot(Fn,verticals = TRUE, do.points = FALSE, col="brown", xlab = "x",
xlim = c(min(x),60),ylim = c(0,1),ylab="cdf", main = "", font.main=6,
col.01line = NULL)
####MWE####
TRTGLE=function(par){
al=par[1]
lm=par[2]
l=-(n*log(lm)-lm*sum(x)+sum(log((((al+2)/(al+1))-exp(-lm*al*x))))))
par=c(.01,.01)
k=optim(par,TRTGLE, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
s_e = sqrt(abs(diag(solve(-k$hessian))))
k=k$par
al=k[1]
lm=k[2]
like=-TRTGLE(k)
ll=-2*like
AIC=ll+2*2
s_tr=cbind(k[1],s_e[1],k[2],s_e[2])
F=function(x) (1-((1/(al+1))*exp(-lm*x)*(al+2-exp(-lm*al*x))))
curve(F(x),lwd = 1, col = "chocolate3", add=TRUE)
####WE####
TRTW=function(par){
al=par[1]
lm=par[2]
l=-(-n*log(al)+n*log(1+al)+n*log(lm)-lm*sum(x)+sum(log(1-exp(-lm*al*x))))
par=c(.01,0.01)
kk_on=optim(par,TRTW, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
s_eon = sqrt(abs(diag(solve(-kk_on$hessian))))
kk_on=kk_on$par
al1=kk_on[1]
lm1=kk_on[2]
like_on=-TRTW(kk_on)
ll_on=-2*like_on
AIC_on=ll_on+2*2
s_on=cbind(kk_on[1],s_eon[1],kk_on[2],s_eon[2])
F=function(x) (((al1+1)/al1)*(1-exp(-lm1*x))-(1/(al1+1))*(1-exp(-lm1*x*(1+al1))))
curve(F(x), lwd = 1, col = "red", add=TRUE)
####W####
W=function(par){
mu=par[1]
s=par[2]
l=-(n*log(mu)-n*log(s)+(mu-1)*sum(log((x/s)))-sum((x/s)^(mu)))
par=c(.8,6.9)
kk_wl=optim(par,W, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
s_ewl = sqrt(abs(diag(solve(-kk_wl$hessian))))
kk_wl=kk_wl$par
mu1=kk_wl[1]
s1=kk_wl[2]

```

```

like_w=-W(kk_wl)
ll_w=-2*like_w
AIC_w=ll_w+2*2
s_w=cbind(kk_wl[1],s_ewl[1],kk_wl[2],s_ewl[2])
F=function(x) (1-exp(-(x/s1)^(mu1)))
curve(F(x), lwd = 1, col = "green", add=TRUE)
####Gamma####
Wg=function(par){
al=par[1]
bt=par[2]
l=-(-n*al*log(bt)-n*log(factorial(al-1))+(al-1)*sum(log(x))-sum((x/bt))))}
par=c(.1,.9)
kk_g=optim(par,Wg, method = "L-BFGS-B",lower=c(0,0), upper=c(Inf, Inf),hessian=T)
s_g = sqrt(abs(diag(solve(-kk_g$hessian))))
kk_g=kk_g$par
al1=kk_g[1]
bt1=kk_g[2]
like_g=-Wg(kk_g)
ll_g=-2*like_g
AIC_g=ll_g+2*2
s_gg=cbind(kk_g[1],s_g[1],kk_g[2],s_g[2])
bt2=1/bt1
F=function(x) (pgamma(x,al1,bt2))
curve(F(x),lwd = 1, col = "blue", add=TRUE)
####GE####
W_ge=function(par){
al=par[1]
bt=par[2]
l=-((n*log(al)+n*log(bt)+(al-1)*sum(log(1-exp(-bt*x)))-bt*sum(x)))}
par=c(.1,.01)
kk_ge=optim(par,W_ge, method = "L-BFGS-B",lower=c(0, 0), upper=c(Inf,Inf),hessian=T)
s_ge = sqrt(abs(diag(solve(-kk_ge$hessian))))
kk_ge=kk_ge$par
al1=kk_ge[1]
bt1=kk_ge[2]
like_ge=-W_ge(kk_ge)
ll_ge=-2*like_ge
AIC_ge=ll_ge+2*2
s_ge=cbind(kk_ge[1],s_ge[1],kk_ge[2],s_ge[2])
F=function(x) ((1-exp(-bt1*x))^(al1))
curve(F(x), lwd = 1, col = "yellow", add=TRUE)
legend("bottomright", c("Empirical_CDF","MWE","W","G","GE","WE"), xpd=TRUE,
horiz=FALSE, col=c("brown","chocolate3","green","blue","yellow","red"),
lty=c(1,1,1,1,1,1),cex=0.8)
#Similarly, we can calculate all values and graphs for data set 2.

```