Tests

Raj Patel

2020-03-29

Question 1:

Consider a (univariate) log normal random variable: X $X \sim logN(\mu, \sigma^2)$.

(a) Write a R function that determines the value of parameters μ and σ^2 from the expectation E(x) and variance V(x).

```
param_logn <- function(e_x, var_x){

  # determining parameter sigma_2
  sigma2 <- log((var_x/((e_x)^2)) + 1)

  # determining parameter mu
  mu <- log(e_x) - (sigma2/2)
  return(c(mu,sigma2))

}
```

(b)

```
# Set seed for consistency
set.seed(10)

# extracting the parameters
mu_1 <- param_logn(3,5)[1]
sigma2_1 <- param_logn(3,5)[2]

cat("The parameter mu is:", mu_1)
```

```
## The parameter mu is: 0.8776959
```

```
cat("\nThe parameter sigma_2 is:", sigma2_1)
```

```
##
## The parameter sigma_2 is: 0.4418328
```

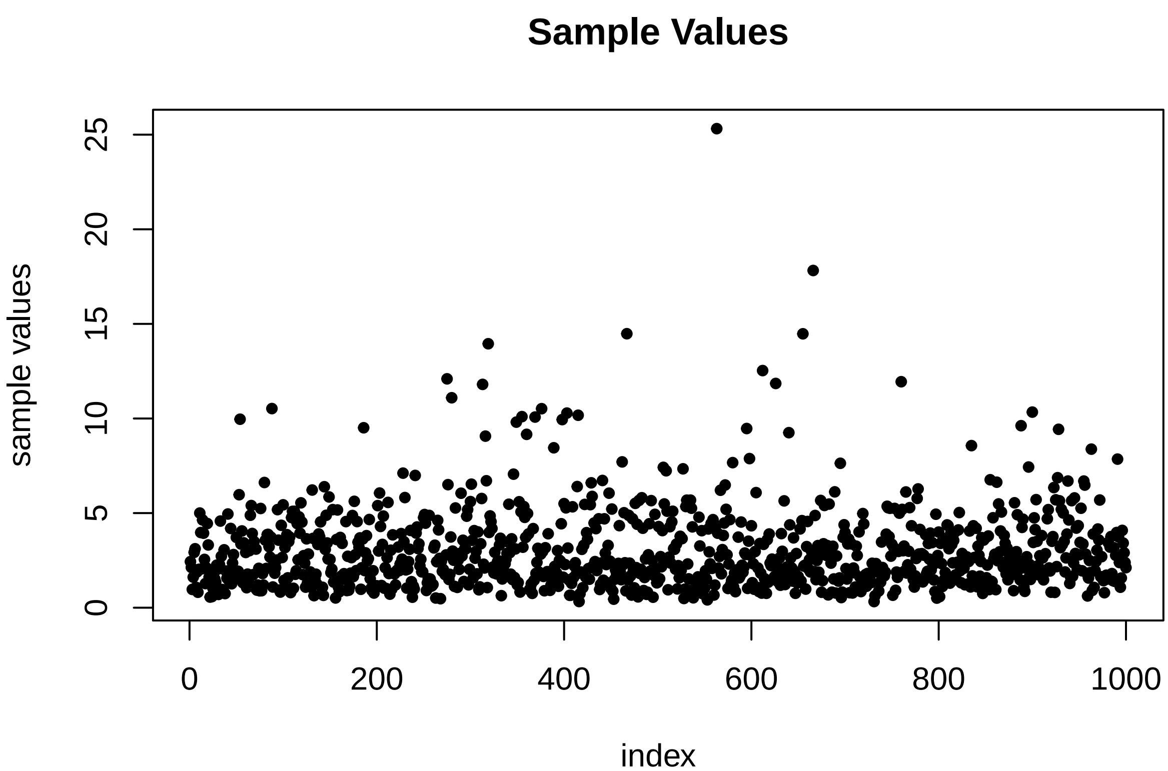
```
# Double-checking the parameters:
e_x <- exp(mu_1 + (sigma2_1/2))
v_x <- (exp(sigma2_1) - 1)*exp(2*mu_1 + (sigma2_1))

cat("\nThe mean for lognormal RV is:", e_x, ", which matches what
    was provided in the question i.e.,3")
```

```
##
## The mean for lognormal RV is: 3 , which matches what
##     was provided in the question i.e.,3
```

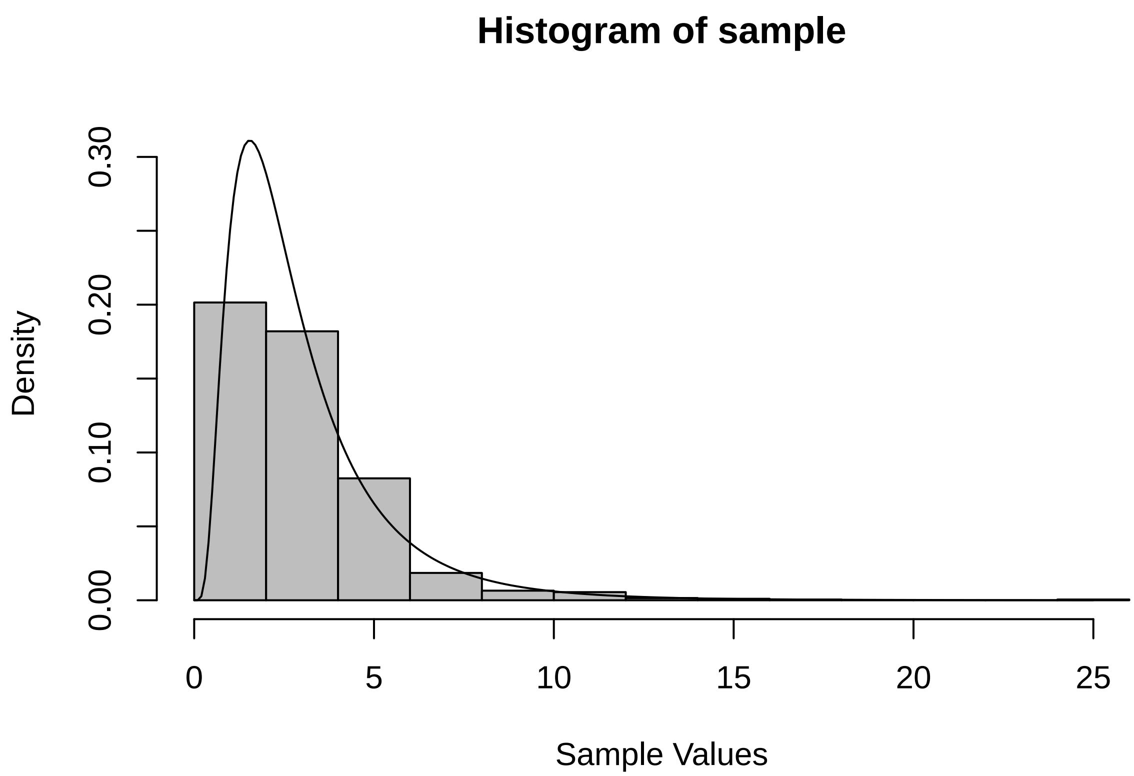
```
cat("\nThe variance for lognormal RV is:", v_x, ", which matches
    what was provided in the question i.e.,5")
```

```
##
## The variance for lognormal RV is: 5 , which matches
##     what was provided in the question i.e.,5
```

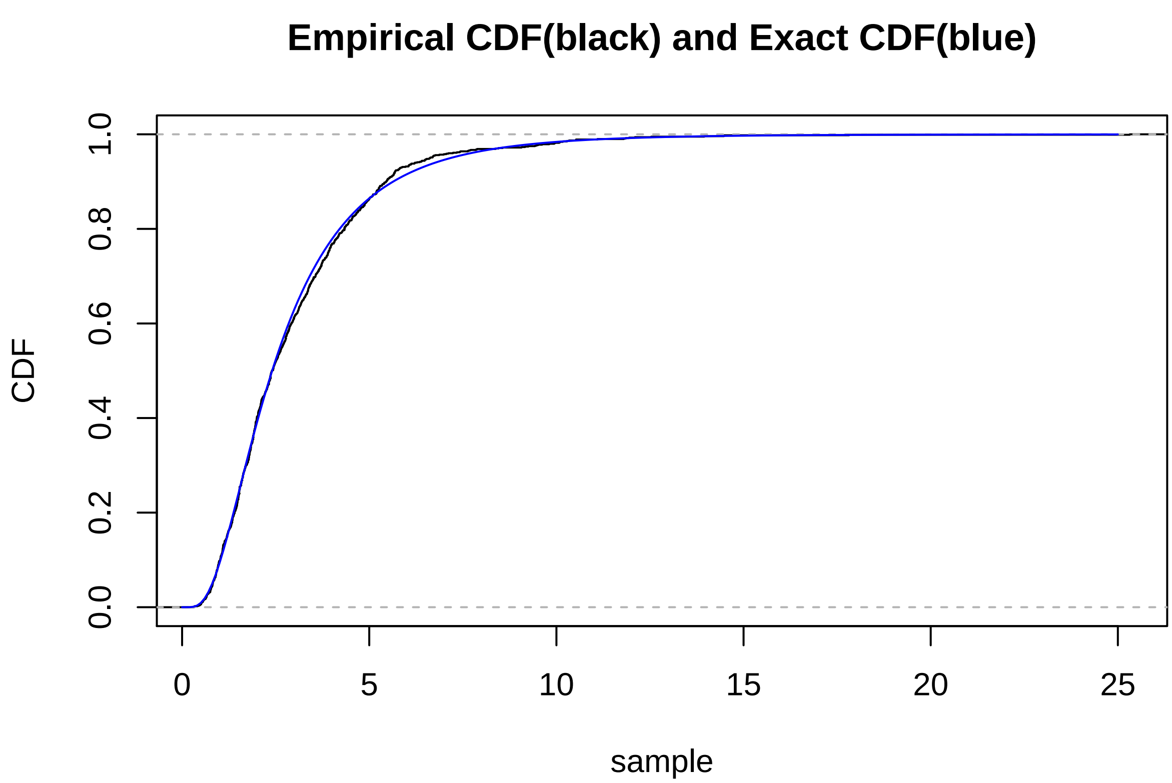
```
# generating sample
sample <- rlnorm(1000,mu_1,sqrt(sigma2_1))

# plotting the sample
plot(sample, ylab = "sample values", xlab = "index", pch = 20, main = "Sample Values")
```




```
# plotting histogram and superimposing it with exact pdf
x <- seq(0,25,by=0.1)
hist(sample, freq = FALSE, col = "grey", ylim = c(0,0.32), xlab = "Sample Values")
lines(x,dlnorm(x, mu_1,sqrt(sigma2_1)))
```




```r
# plotting the empirical cdf and superimposing it with exact cdf
plot.ecdf(sample,xlab="sample",ylab="CDF",xlim=c(min(sample),max(sample)),
          main="Empirical CDF(black) and Exact CDF(blue)")
lines(x,plnorm(x, mu_1,sqrt(sigma2_1)), col = "blue")
```



Question 2:

Part a:

```r
fit_locdisp_mlfp <- function(e,p,v,t){

  # Step 0: Initialize

  # Initial mu
  mu_init = c()
  for(i in 1:ncol(e)){
    mu_init[i] = sum(e[,i] * p[i])
  }

  # Initial covariance matrix
  var_init <- matrix(vector(mode = "numeric", length = (ncol(e)^2)),
                     nrow = ncol(e), ncol = ncol(e))
  for(i in 1:nrow(e)){
    a <- p[i]*(e[i,] - mu_init)%*%t(e[i,] - mu_init)
    var_init <- var_init + a
  }

  ifelse(v > 2, var_init <- ((v-2)/v)*var_init, var_init <- var_init)


  # Step 1: Update weights and FP
  wgts <- vector(mode = "numeric", length = nrow(e))
  qt <- vector(mode = "numeric", length = nrow(e))

  # weights
  for(i in 1:nrow(e)){
    wgts[i] <- (v + ncol(e))/(v + t(e[i,] - mu_init)%*%solve(var_init)%*%(e[i,] -
                                                                          mu_init))
  }

  # qt
  for(i in 1:nrow(e)){
    qt[i] <- p[i]*wgts[i]/sum(p*wgts)
  }


  # Step 2: Updating the parameters

  # Updating mu
  mu_updated <- c()
  for(i in 1:ncol(e)){
    mu_updated[i] <- sum(e[,i]*qt)
  }

  # Updating covariance matrix
  var_updated <- matrix(vector(mode = "numeric",
                               length = (ncol(e)^2)), nrow = ncol(e), ncol = ncol(e))
  for(i in 1:nrow(e)){
```

```r
    a <- qt[i]*(e[i,] - mu_updated)%*%t(e[i,] - mu_updated)
    var_updated <- var_updated + a
  }

  # Step 3: Checking convergence and continuing if not converged
  while ((norm(mu_updated - mu_init, type = "2") / norm(mu_init, type = "2")) > t
         || (norm(var_updated - var_init, type = "F") / norm(var_init, type = "F")) > t){

    mu_init <- mu_updated
    var_init <- var_updated

    # Re-Step 1: Update weights and FP
    wgts <- vector(mode = "numeric", length = nrow(e))
    qt <- vector(mode = "numeric", length = nrow(e))

    # Re-weights
    for(i in 1:nrow(e)){
      wgts[i] <- (v + ncol(e))/(v + t(e[i,] - mu_init)%*%solve(var_init)%*%(e[i,]
                                                                            - mu_init))
    }

    # Re-qt
    for(i in 1:nrow(e)){
      qt[i] <- p[i]*wgts[i]/sum(p*wgts)
    }


    # Re-Step 2: Updating the parameters
    mu_updated <- c()
    for(i in 1:ncol(e)){
      mu_updated[i] <- sum(e[,i]*qt)
    }

    var_updated <- matrix(vector(mode = "numeric", length = (ncol(e)^2)),
                          nrow = ncol(e), ncol = ncol(e))
    for(i in 1:nrow(e)){
      a <- qt[i]*(e[i,] - mu_updated)%*%t(e[i,] - mu_updated)
      var_updated <- var_updated + a
    }

  }

  # Returning converged location and dispersion parameters
  list_ret <- list(mu_updated, var_updated)

  return(list_ret)

}
```

Question 2:

Part b:

```r
# Set seed for consistency
set.seed(10)

# Covariance for generating samples
Sigma <- matrix(c(1,0,0,1),2,2)

# Samples from standard bivariate normal
e <- mvrnorm(n = 1000, rep(0, 2), Sigma)

# Probabilities
p <- rep(1/1000,1000)

# MLE of location and dispersion parameters
fit_locdisp_mlfp(e, p, 100, 10^(-9))
```

```
## [[1]]
## [1] -0.01707193  0.01203408
##
## [[2]]
##             [,1]        [,2]
## [1,]  1.06866964 -0.04570523
## [2,] -0.04570523  0.96490795
```