Week 3

gherardo varando

December 12, 2018

A trigonometric density

We now load the data

```
angles <- read.table("angles.txt")$x
```

Ex 1

1.0

This density is not normalized (that is, its integral is not 1), we need thus to compute the normalization constant $C_k = 1/\int_0^\pi sin(x)^k dx$ and then define the appropriate density function in R.

You can find an R file with all the functions for this density here

```
source("f_sink.R") ###now we have dsin, psin, qsin and rsin
```

We actually only need `dsin` but I had to generate the data so I needed also `rsin` (the random number generetor) and since I used the inverse transform sampling I needed also the quantile functions and thus the CDF.

1.1

The model is parametric with one parameter k,

$$f(x|k) = \frac{sin(x)^k}{\int_0^\pi sin(t)^k dt}$$

1.1

Analytically the minus log likelihood is,

$$-\ell(k) = -k \sum_{i=1}^{n} \log \sin(X_i) + n \log \left(\int_0^\pi sin(t)^k dt \right)$$

We define the minus log likelihood in R,

```
mll <- function(k, data){
  sum( -log(dsin(x = data, k = k )))
}
```

1.2

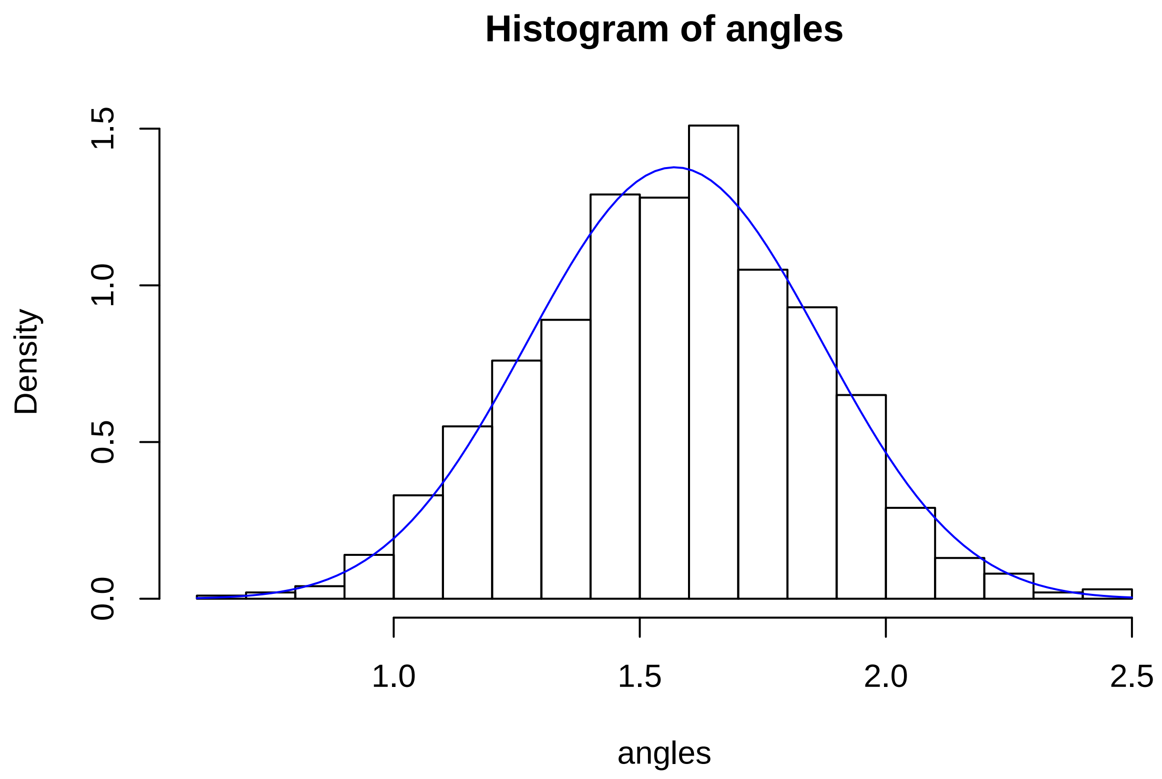
We use here numerical optimization to obtain the MLE of the parameter k.

```r
k_est <- optimize(mll, interval = c(0,100), data = angles)$minimum
k_est
```

```
## [1] 11.40039
```

1.3

```r
hist(angles, probability = TRUE, breaks = "FD")
curve( dsin(x, k_est), add = TRUE, col = "blue")
```



A case study of neuronal data

We continue the case study of the ISI data in the `neuronspikes.txt` file,

```r
isi_data <- read.table("neuronspikes.txt")
isi <- isi_data$V1
```

Ex 2

2.1

Exponential model $isi \sim exponential(\lambda)$

The maximum likelihood estimation of λ is $\hat{\lambda} = 1/\overline{X}$,

```r
rate_est <- 1/mean(isi)
rate_est
```

```
## [1] 1.146891
```

2.2

Gamma model,

$$isi \sim Gamma(\alpha, \beta)$$

α is called shape parameter and β is the rate.

The MLE can be found with optimization methods, we define the minus log-likelihood,

```r
mllg <- function(pars, data){
  -sum(dgamma(data, shape = pars[1], rate = pars[2], log = TRUE))
}
```

Now we can use numerical optimization, we use a very bad choice for the initial point:

```r
res_optim <- optim(par = c(10, 10), fn = mllg, data = isi)
pars_est <- res_optim$par
res_optim
```

```
## $par
## [1] 1.562647 1.792188
##
## $value
## [1] 252.7012
##
## $counts
## function gradient
##       79       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

2.3

We can find the method of moments estimators from the following equations,

$$\frac{\alpha}{\beta} = \overline{X} \tag{1}$$

$$\frac{\alpha}{\beta^2} = s_n^2 = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - \overline{X}\right)^2 \tag{2}$$

From equation (1) we obtain,

$$\alpha = \beta\overline{X}$$

We substitute α in equation (2) and we obtain,

$$\frac{\beta\overline{X}}{\beta^2} = s_n^2$$

and then,

$$\frac{\overline{X}}{\beta} = s_n^2$$

we can now solve it for β obtaining

$$\hat{\beta} = \frac{\overline{X}}{s_n^2}$$

Then we can obtain for α

$$\hat{\alpha} = \hat{\beta}\overline{X} = \frac{\overline{X}^2}{s_n^2}$$

We will use s^2 instead of s_n^2, the results will not vary a lot and we avoid to rescale the empirical variance.

```
shape_mmest <- mean(isi)^2 / var(isi)  ##we just use the var function
rate_mmest <-  mean(isi) / var(isi)
c(shape_mmest, rate_mmest)
```

```
## [1] 1.283954 1.472556
```

we can now use this estimates as initial points in the optimization method,

```
res_optim <- optim(par = c(shape_mmest,rate_mmest),
                   fn = mllg, data = isi)
pars_est <- res_optim$par
res_optim
```

```
## $par
## [1] 1.562585 1.792009
##
## $value
## [1] 252.7012
##
## $counts
## function gradient
##       63       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

Ex 3

We use now the Inverse Gaussian distribution to model the ISI data. The inverse Gaussian density is,

$$f(x|\mu,\lambda) = \left(\frac{\lambda}{2\pi x^3}\right)^{1/2} \exp\left(\frac{-\lambda(x-\mu)^2}{2\mu^2 x}\right)$$

3.1

The log likelihood is

$$\ell(\mu,\lambda) = \sum_{i=1}^{n} \log f(X_i|\mu,\lambda) = \frac{1}{2}\sum_{i=1}^{n}\log(\lambda) - \sum_{i=1}^{n}\log(2\pi X_i^3) - \sum_{i=1}^{n}\frac{\lambda(X_i-\mu)^2}{2\mu^2 X_i}$$

$$= \frac{n}{2}\log(\lambda) - n\log(2\pi) - 3\sum_{i=1}^{n}\log(X_i) - \sum_{i=1}^{n}\frac{\lambda(X_i-\mu)^2}{2\mu^2 X_i}$$

3.2

To obtain abalytically the MLE of λ and μ we need to compute the gradient of ℓ.

We start with the partial derivative with respect to μ,

$$\frac{\partial \ell}{\partial \mu} = -\lambda\sum_{i=1}^{n}\frac{-2(X_i-\mu)2\mu^2X_i - 4\mu X_i(X_i-\mu)^2}{4\mu^4X_i^2}$$

$$= -\lambda\sum_{i=1}^{n}\frac{-\mu X_i + \mu^2 - X_i^2 - \mu^2 + 2\mu X_i}{\mu^3 X_i}$$

$$= -\lambda\sum_{i=1}^{n}\frac{\mu X_i - X_i^2}{\mu^3 X_i} = -\lambda\sum_{i=1}^{n}\frac{\mu - X_i}{\mu^3}$$

So if we impose $\partial\ell/\partial\mu = 0$ we obtain,

$$-\lambda\sum_{i=1}^{n}\frac{\mu - X_i}{\mu^3} = 0$$

and hence (by multiplying by μ^3 and dividing by $-\lambda$),

$$\sum_{i=1}^{n}\mu - X_i = 0$$

$$\hat{\mu} = \overline{X}$$

To obtain $\hat{\lambda}$ we impose that $\partial\ell/\partial\lambda = 0$:

$$\frac{\partial \ell}{\partial \lambda} = \frac{n}{2\lambda} - \sum_{i=1}^{n}\frac{(X_i-\mu)^2}{2\mu^2X_i} = 0$$

$$\frac{1}{\hat{\lambda}} = \frac{1}{n}\sum_{i=1}^{n}\frac{X_i^2 + \overline{X}^2 - 2X_i\overline{X}}{\overline{X}^2 X_i} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{X_i} + \frac{1}{n\overline{X}^2}\sum_{i=1}^{n}X_i - 2\overline{X}$$

$$\frac{1}{\hat{\lambda}} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{X_i} - \frac{1}{\overline{X}}$$

3.3

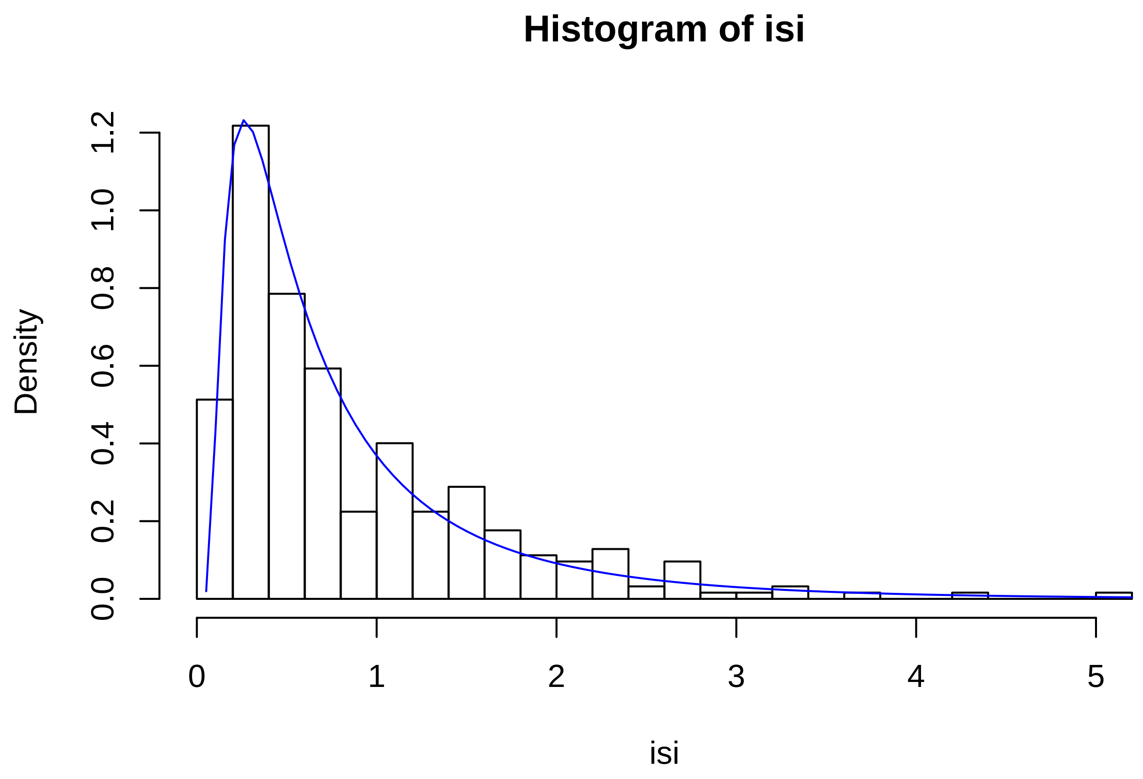
In R,

```r
mu_est <- mean(isi)
lambda_est <-1 /( mean(1/isi) - 1/mean(isi))
c(mu_est, lambda_est)
```

```
## [1] 0.8719221 0.8679884
```

Plotting,

```r
dinvgauss <- function(x, mu = 1, lambda = 1){
  sqrt(lambda/(2*pi*x^3))*exp( -(lambda* (x-mu)^2)/ (2*mu^2*x) )
}
hist(isi, probability = TRUE, breaks = "FD")
curve(dinvgauss(x, mu = mu_est, lambda = lambda_est), add = TRUE,
      col = "BLUE")
```



3.4

As usual we define the minus log-likelihood,

```r
mllig <- function(pars, data){
  -sum(log(dinvgauss(data, pars[1], pars[2])))
}
```

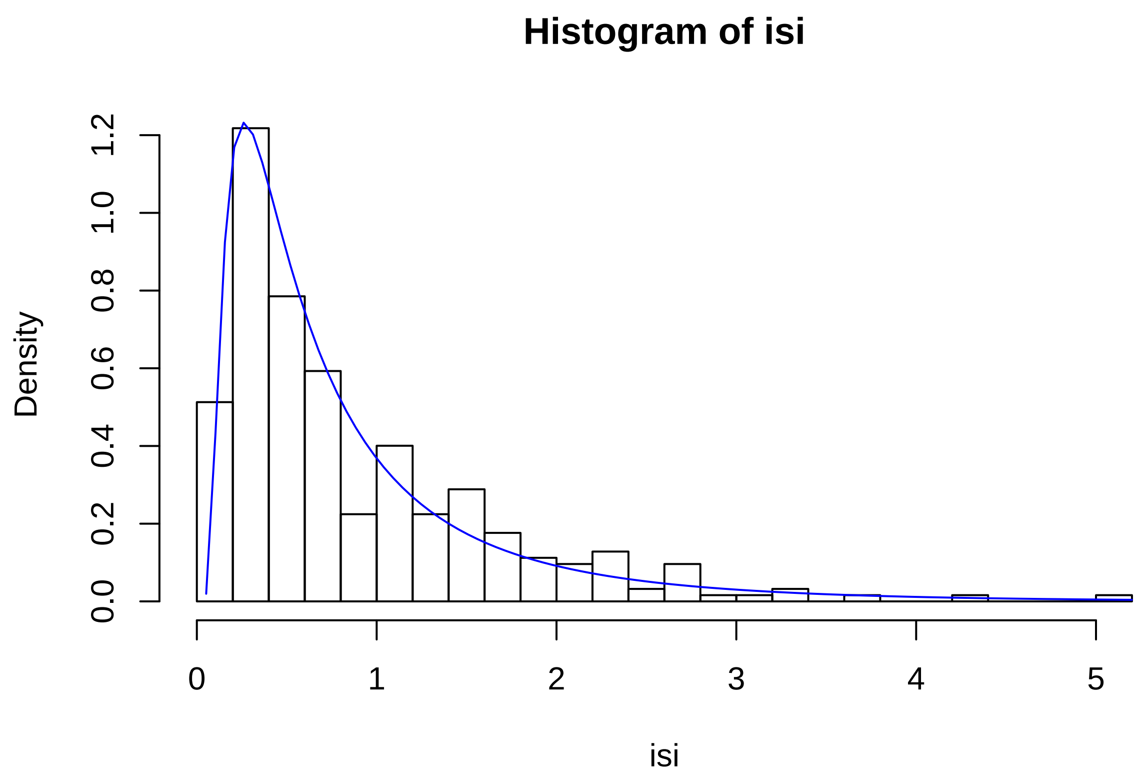
And we call the `optim` function,

```r
optim(par = c(1, 1), fn = mllig, data = isi)
```

```
## $par
## [1] 0.8718732 0.8679736
##
## $value
## [1] 235.4785
##
## $counts
## function gradient
##       51       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

3.5

```
hist(isi, probability = TRUE, breaks = "FD")
curve(dinvgauss(x, mu = mu_est, lambda = lambda_est), add = TRUE,
      col = "BLUE")
```



Brain cell dataset

We load the data,

```
cells <- read.csv("cell_types.csv", na.strings = "")
rampspiketime <- cells$ef__peak_t_ramp
```

Ex 4

4.1

We can find the maximum-likelihood easily with numerical optimization, as usual we first of all define the minus log-likelihood, we use the function `na.omit` so we are sure to clean the data from NA values before applying the density:

```
mllln <- function(pars, data){
  -sum(dlnorm(na.omit(data), pars[1], pars[2], log = TRUE))
}
```

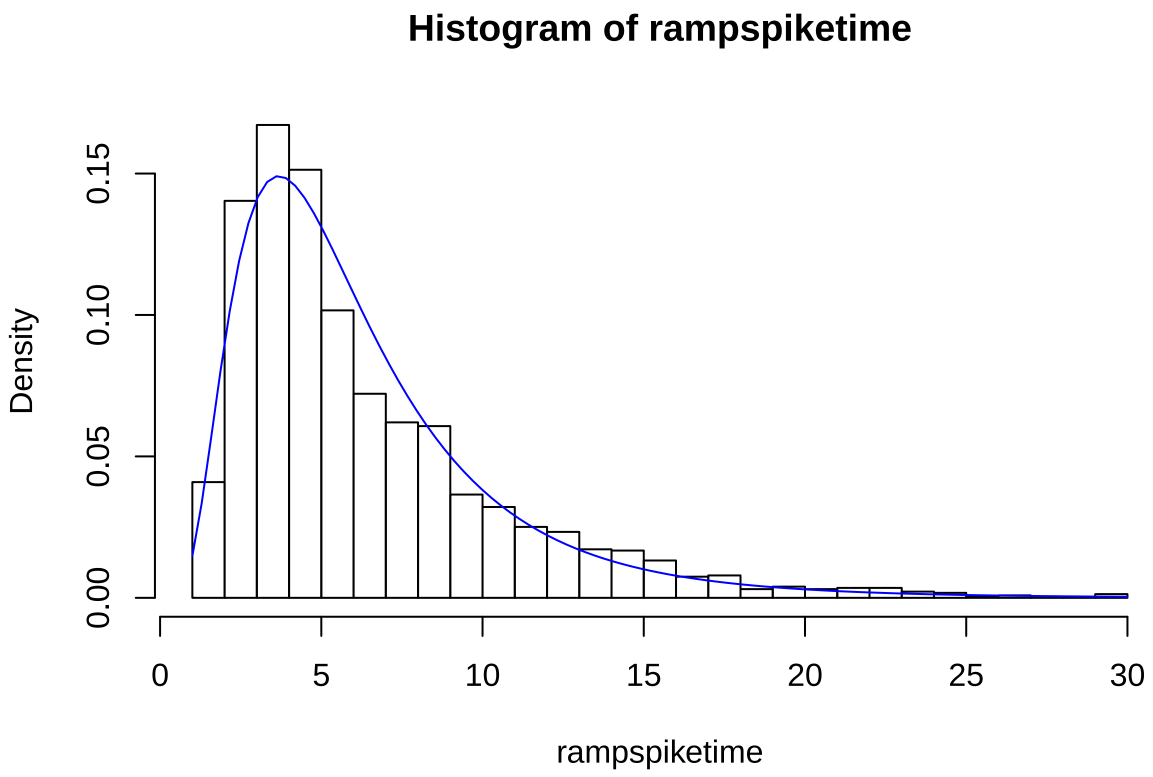
Now we can run `optim`

```
pars_est <- optim( c(0,1), fn = mllln, data = rampspiketime)$par
pars_est
```

```
## [1] 1.6689543 0.6056442
```

We plot the result,

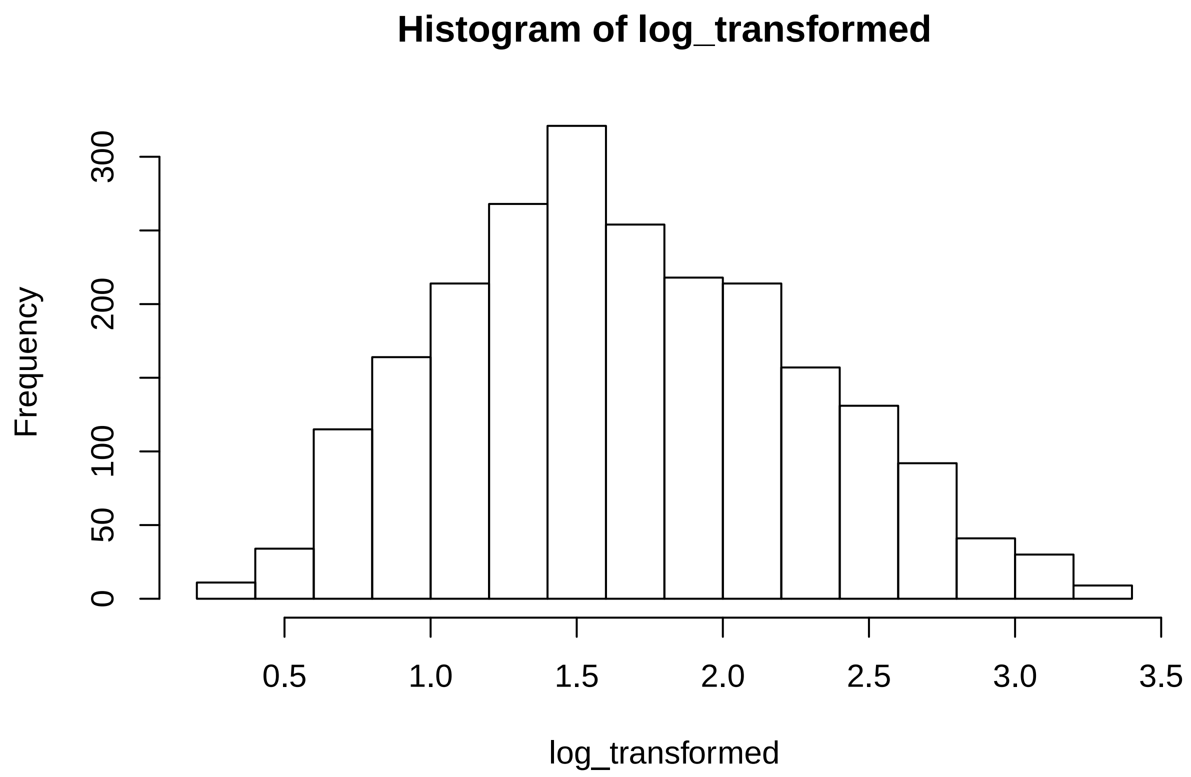
```
hist(rampspiketime, probability = TRUE, breaks = "FD")
curve(dlnorm(x, meanlog = pars_est[1], sdlog = pars_est[2]), add
      = TRUE, col = "blue")
```



4.2

We transform the ramp spike observations with the logarithm

```
log_transformed <- log(rampspiketime)
hist(log_transformed)
```



The MLE for the parameter of a Gaussian distribution can be obtained analytically,

$$\hat{\mu} = \overline{X} \quad \hat{\sigma} = \frac{(n-1)s}{n} \approx s$$

So we get in this case,

```r
n <- length(na.omit(log_transformed))
mu_est <- mean(log_transformed, na.rm = TRUE)
sigma_est <- sd(log_transformed, na.rm = TRUE)
c(mu_est, sigma_est)
```

```
## [1] 1.6688950 0.6057655
```

4.3

For the human male observations:

```r
male_obs_log <- log_transformed[cells$donor__sex == "Male"]
mu_male <- mean(male_obs_log, na.rm = TRUE)
sigma_male <- sd(male_obs_log, na.rm = TRUE)
c(mu_male, sigma_male)
```

```
## [1] 1.8357933 0.7271868
```

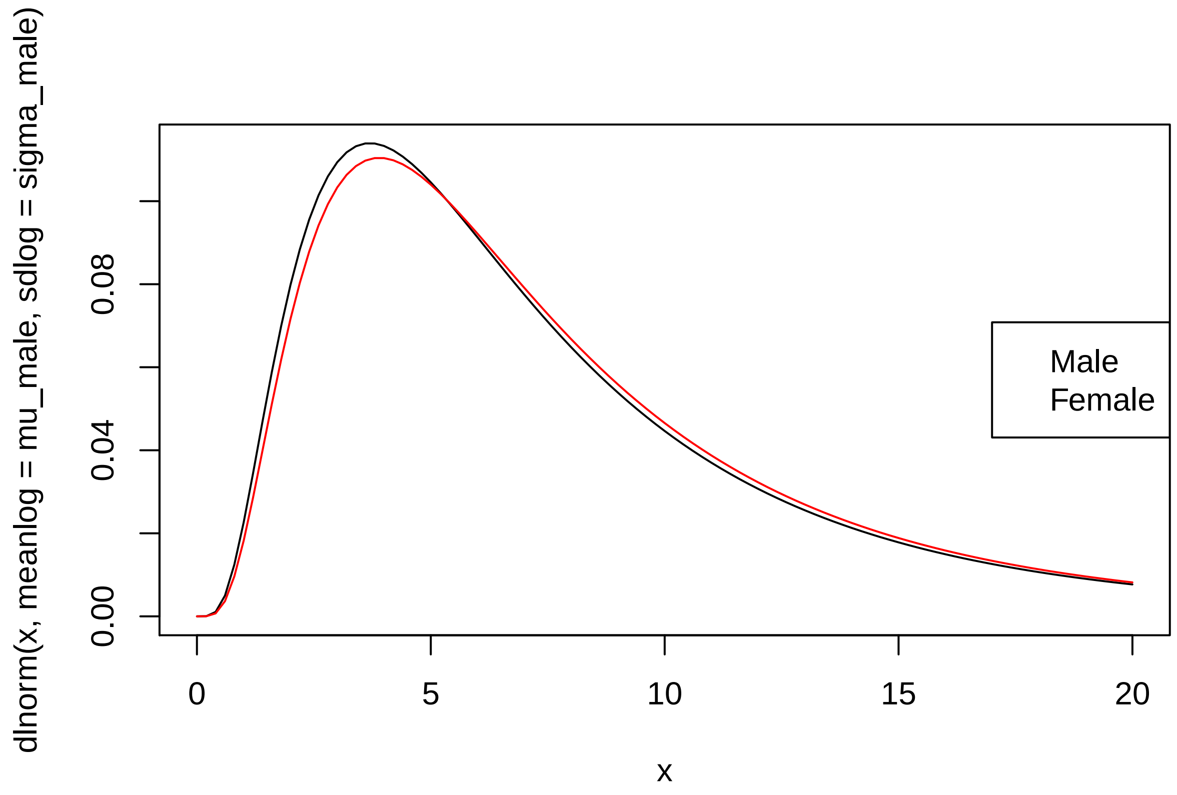
For the human female

```r
female_obs_log <- log_transformed[cells$donor__sex == "Female"]
mu_female <- mean(female_obs_log, na.rm = TRUE)
```

```
sigma_female <- sd(female_obs_log, na.rm = TRUE)
c(mu_female, sigma_female)
```

```
## [1] 1.8741756 0.7168038
```

```
curve(dlnorm(x, meanlog = mu_male, sdlog = sigma_male), from = 0, to = 20)
curve(dlnorm(x, meanlog = mu_female,
             sdlog = sigma_female), from = 0, to = 20, col = "red", add = TRUE)
legend("right", legend = c("Male", "Female"), col = c("black", "red"))
```



Jukes-Cantor model

Ex 5

5.1

```
f_jk <- function(x, y, t, a){
  if (x == y){
    0.25 + 0.75 * exp(-4*a*t)
  }else{
    0.25 - 0.25 * exp(-4*a*t)
  }
}
```

5.2

We can write the likelihood as:

$$\mathcal{L}(\alpha) = \prod_{i=1}^{n} P(X = x_i, Y = y_i) = \prod_{i=1}^{n} P(X = x_i)P(Y = y_i|X = x_i) =$$

$$\prod_{x_i=y_i} \frac{1}{4}\left(1/4 + 3e^{-4\alpha t}/4\right) \prod_{x_i\neq y_i} \frac{1}{4}\left(1/4 - e^{-4\alpha t}/4\right) =$$

$$= \frac{1}{4^{2n}}\left(\prod_{x_i=y_i} 1 + 3e^{-4\alpha t} \prod_{x_i\neq y_i} 1 - e^{-4\alpha t}\right)$$

so the log-likelihood is:

$$\ell(\alpha) = \log\left(\mathcal{L}(\alpha)\right) = -2n\log(4) + \sum_{x_i=y_i} \log(1 + 3e^{-4\alpha t}) + \sum_{x_i\neq y_i} \log(1 - e^{-4\alpha t})$$

If we introduce the statistics

$$n_1 = |\{i : X_i = Y_i\}|$$

and

$$n_2 = |\{i : X_i \neq Y_i\}| = n - n_1$$

we have that

$$\ell(\alpha) = -2n\log(4) + n_1\log(1 + 3e^{-4\alpha t}) + n_2\log(1 - e^{-4\alpha t})$$

5.3

To find the MLE of α we have to compute the derivative of ℓ.

$$\frac{d\ell(\alpha)}{d\alpha} = \frac{n_1}{1 + 3e^{-4\alpha t}}(-12e^{-4\alpha t}) + \frac{n_2}{1 - e^{-4\alpha t}}(4e^{-4\alpha t}) =$$

$$\frac{4n_2e^{-4\alpha t}(1 + 3e^{-4\alpha t}) - 12n_1e^{-4\alpha t}(1 - e^{-4\alpha t})}{\dots} =$$

We can divide by $4e^{-4\alpha t} > 0$ and obtain the equation for the critical point as:

$$n_2 + 3n_2e^{-4\alpha t} - 3n_1 + 3n_1e^{-4\alpha t} = 0$$

thus

$$e^{-4\alpha t} = \frac{3n_1 - n_2}{3(n_1 + n_2)}$$

Thus if $3n_1 \leq n_2$ there are no solution and $d\ell/d\alpha > 0$ thus the log-likelihood is unbounded. If $3n_1 > n_2$ then the only critical point is

$$\hat{\alpha} = -\frac{1}{4t}\log\left(\frac{3n_1 - n_2}{3n}\right)$$

which we can prove to be a maximum of ℓ and thus the MLE. We can also rewrite it as

$$\hat{\alpha} = \frac{\log(3n) - \log(3n_1 - n_2)}{4t}$$

5.4

```r
simulate_jk <- function(n = 1, t = 1, a = 1){
x <- sample(x = c("A", "C", "G", "T"), size = n,
            replace = TRUE)
y <- sapply(x, function(xx){
  probs  <- sapply(c("A", "C", "G", "T"),
                   function(yy){
                     f_jk(x = xx, y =yy, t = t, a = a)
                   })
  sample(c("A", "C", "G", "T"), size = 1, prob = probs)
})
return(data.frame(x = x, y = y))
}


simulate_jk(5, t = 2, a = 0.5)
```

```
##   x y
## 1 T G
## 2 G T
## 3 G A
## 4 C T
## 5 G A
```

We write e minus log-likelihood

```r
mll_jk <- function(a, data, t = 1){
  -sum(sapply(1:nrow(data), function(i){
    log(f_jk(x = data$x[i], y = data$y[i], t = t, a = a))
  }))
}
D <- simulate_jk(n = 100, t = 1, a = 5)
mll_jk(1, data = D)
```

```
## [1] 138.9665
```

we try to solve the MLE numerically,

```r
a_real <- 0.2
t <- 1
n <- 1000
D <- simulate_jk(n = n, t = t, a = a_real)
a_est <- optimize(f = mll_jk, interval = c(0,2), data = D, t = t)$minimum

a_est
```

```
## [1] 0.1905384
```

we can compare it now with the analytical solution of the MLE obtained before:

```r
n1 <- sum(apply(D, MARGIN = 1,
                 function(r) r[1] == r[2]))
n2 <- nrow(D) - n1
a_mle <- - log( (3*n1 - n2)/ (3*(n1+n2))) / (4*t)
a_mle
```

```
## [1] 0.190535
```