Fitting mixture models to the `faithful` dataset

Problem 3

Problem solver: Rohan Karthikeyan

Reviewer: Adarsha Mondal

Let's attach the dataset and sort the variable of interest, `waiting`.

```
attach(faithful)
waiting = sort(waiting)
```

Model 1: Gamma-Normal mixture

```
NLL_Model1 <- function(theta, data)
{
  shape = theta[1]
  scale = theta[2]
  mu = theta[3]
  sigma = exp(theta[4])
  p = exp(theta[5])/(1+exp(theta[5]))
  n = length(data)
  l = 0

  for(i in 1:n)
  {
    l = l + log(p * dgamma(data[i], shape = shape, scale = scale)+
                  (1-p) * dnorm(data[i], mean = mu, sd = sigma))
  }
  return(-l)
}
```

Let's now fit this model to the `faithful` dataset.

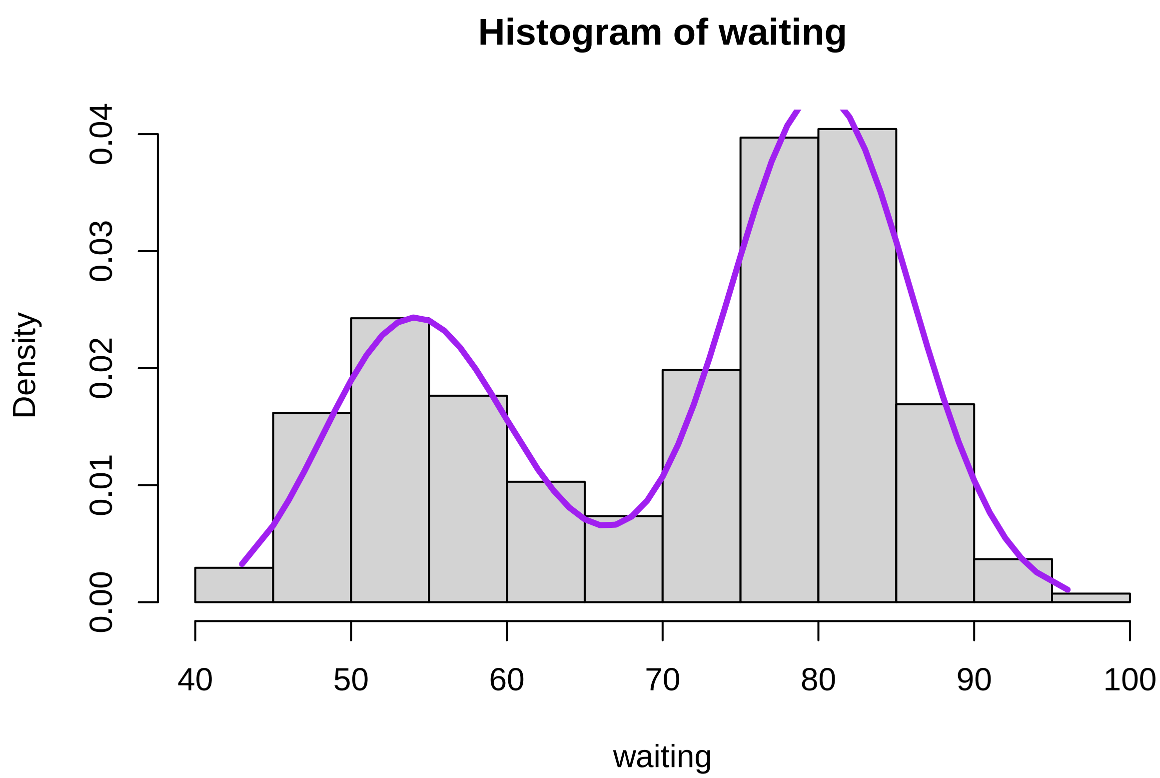
```
p_hat = length(waiting[waiting<65])/length(waiting)
theta_initial1 = c(11, 5, 80, 6, 0.35)

fit_1 = optim(theta_initial1, NLL_Model1,
              data=waiting,
              control=list(maxit=1500))
theta_hat = fit_1$par

shape_hat = theta_hat[1]
scale_hat = theta_hat[2]
mu_hat = theta_hat[3]
sigma_hat = exp(theta_hat[4])
p_hat1 = exp(theta_hat[5])/(1+exp(theta_hat[5]))
```

Additionally, we can plot:

```
d_mle1 = p_hat1*dgamma(waiting, shape = shape_hat, scale = scale_hat) +
  (1-p_hat1)*dnorm(waiting, mean=mu_hat, sd=sigma_hat)
hist(waiting, probability = T)
lines(waiting, d_mle1, lwd=3, col='purple')
```



Model 2: Gamma-Gamma mixture

```
NLL_Model2 <- function(theta, data)
{
  shape1 = theta[1]
  scale1 = theta[2]
  shape2 = theta[3]
  scale2 = theta[4]

  p = exp(theta[5])/(1+exp(theta[5]))
  n = length(data)
  l = 0

  for(i in 1:n)
  {
    l = l + log(p * dgamma(data[i], shape = shape1, scale = scale1)+
```

```r
                    (1-p) * dgamma(data[i], shape = shape2, scale = scale2))
  }
  return(-l)
}
```

Fitting this model to the dataset:

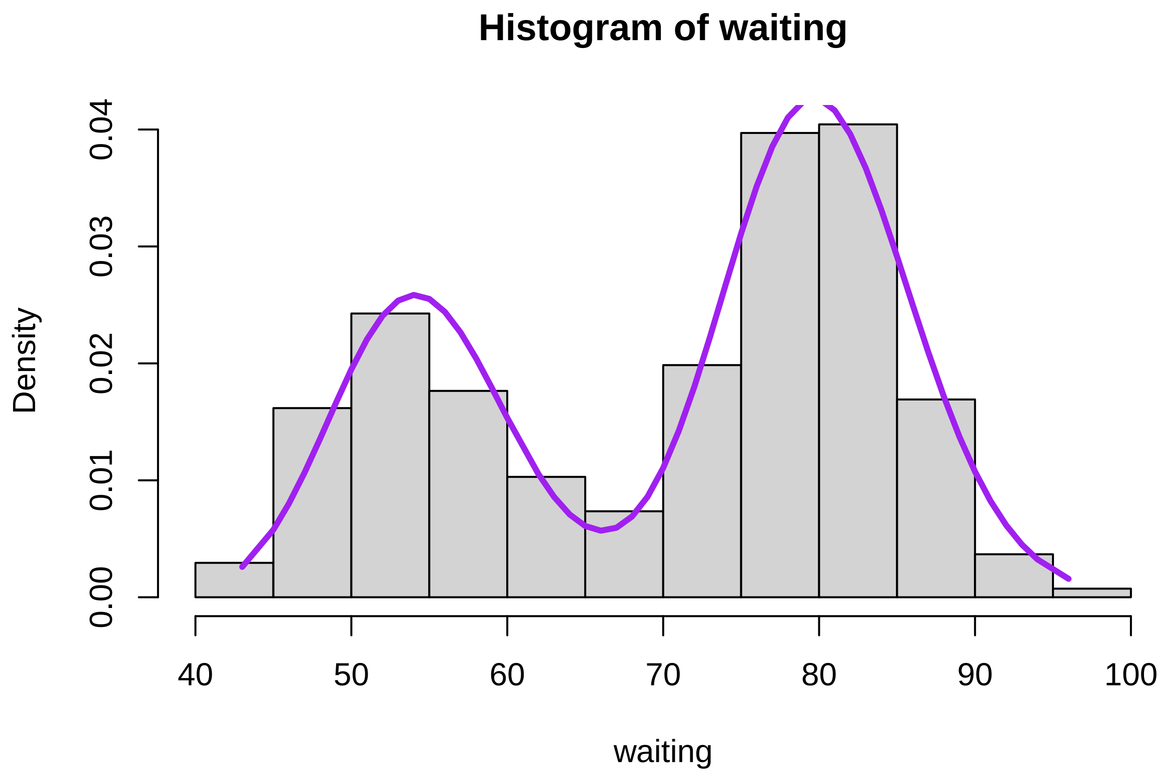
```r
p_hat = length(waiting[waiting<65])/length(waiting)
theta_initial2 = c(11.5, 5, 24.5, 3, 0.35)

fit_2 = optim(theta_initial2, NLL_Model2,
              data=waiting,
              control=list(maxit=1500))
theta_hat = fit_2$par

shape1_hat = theta_hat[1]
scale1_hat = theta_hat[2]
shape2_hat = theta_hat[3]
scale2_hat = theta_hat[4]
p_hat2 = exp(theta_hat[5])/(1+exp(theta_hat[5]))
```

A plot:

```r
d_mle2 = p_hat2*dgamma(waiting, shape = shape1_hat, scale = scale1_hat) +
  (1-p_hat2)*dgamma(waiting, shape = shape2_hat, scale = scale2_hat)
hist(waiting, probability = T)
lines(waiting, d_mle2, lwd=3, col='purple')
```



Model 3: Lognormal-lognormal mixture

```
NLL_Model3 <- function(theta, data)
{
  mu1 = theta[1]
  sigma1 = exp(theta[2])
  mu2 = theta[3]
  sigma2 = exp(theta[4])

  p = exp(theta[5])/(1+exp(theta[5]))
  n = length(data)
  l = 0

  for(i in 1:n)
  {
    l = l + log(p * dlnorm(data[i], meanlog = mu1, sdlog = sigma1)+
                  (1-p) * dlnorm(data[i], meanlog = mu2, sdlog = sigma2))
  }
  return(-l)
}
```

Let's fit this model to the dataset:

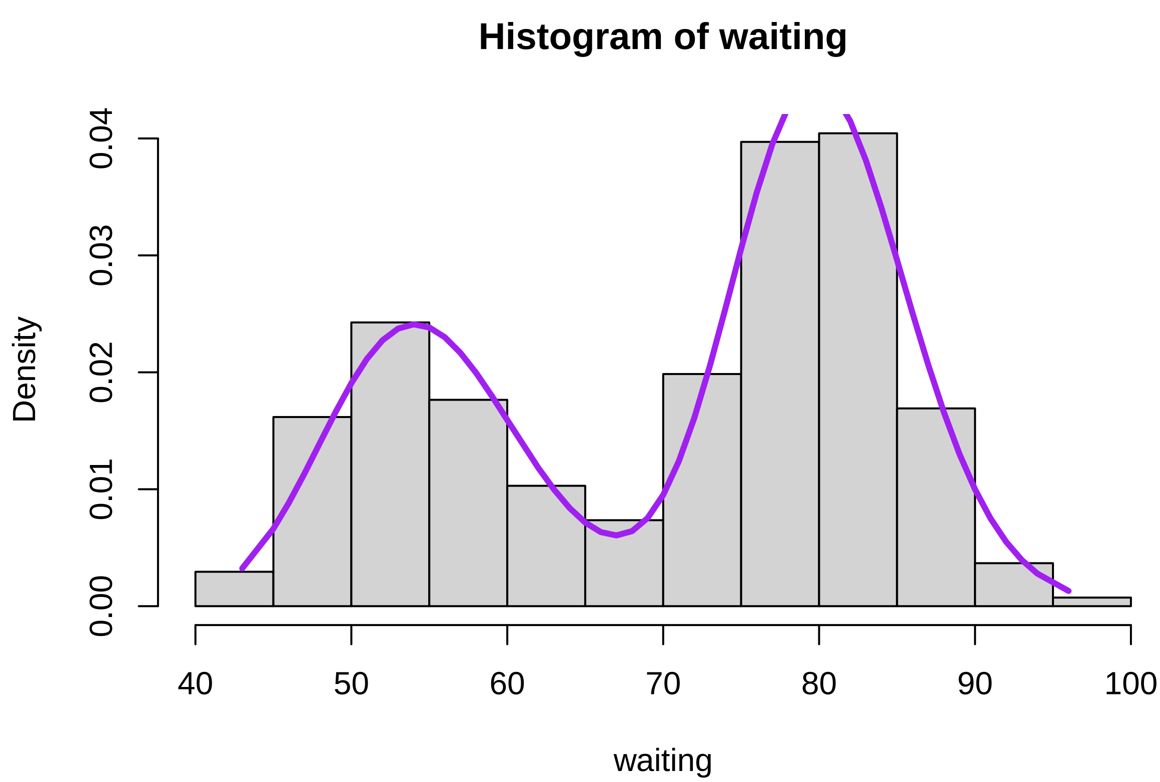
```
p_hat = length(waiting[waiting<65])/length(waiting)
#theta_initial3 = c(3.905, 0.12, 4.38, 0.075, 0.35)
theta_initial3 = c(4, 0.1, 4.85, 0.1, 0.35)

fit_3 = optim(theta_initial3, NLL_Model3,
              data=waiting,
              control=list(maxit=1500))
theta_hat = fit_3$par

mu1_hat = theta_hat[1]
sigma1_hat = exp(theta_hat[2])
mu2_hat = theta_hat[3]
sigma2_hat = exp(theta_hat[4])
p_hat3 = exp(theta_hat[5])/(1+exp(theta_hat[5]))
```

A plot:

```
d_mle3 = p_hat3*dlnorm(waiting, meanlog = mu1_hat, sdlog = sigma1_hat)+
  (1-p_hat3)*dlnorm(waiting, meanlog = mu2_hat, sdlog = sigma2_hat)
hist(waiting, probability = T)
lines(waiting, d_mle3, lwd=3, col='purple')
```



AIC analysis

The Akaike information criterion (AIC) is an estimator of prediction error and thereby relative quality of statistical models for a given set of data.

Suppose that we have a statistical model of some data. Let k be the number of estimated parameters in the model. Let L be the maximized value of the likelihood function for the model. Then the AIC value of the model is the following:

$$\mathrm{AIC} = 2k - 2\ln L$$

Given a set of candidate models for the data, the preferred model is the one with the **minimum** AIC value.

Let's define the AIC function:

```
get_AIC <- function(optim_fit) {
  2 * length(optim_fit$par) + 2 * optim_fit$value
}
```

Let's calculate AIC:

```
library(dplyr)
AIC_scores <- tibble(
  AIC_model1 = get_AIC(fit_1),
  AIC_model2 = get_AIC(fit_2),
  AIC_model3 = get_AIC(fit_3)
)

AIC_scores
```

```
## # A tibble: 1 x 3
##   AIC_model1 AIC_model2 AIC_model3
##        <dbl>      <dbl>      <dbl>
## 1      2076.      2077.      2075.
```

From the AIC scores, we can see that Model 3 is just marginally better than the other models.

Probability calculation

Let's use Model 3 to calculate $\mathbb{P}(60 < \texttt{waiting} < 70)$.

```
dMix<-function(x,theta){
  mu1 = theta[1]
  sigma1 = theta[2]
  mu2 = theta[3]
  sigma2 = theta[4]
  p = theta[5]

  f = p*dlnorm(x, meanlog = mu1, sdlog = sigma1) +
    (1-p)*dlnorm(x, meanlog = mu2, sdlog = sigma2)
  return(f)
}
```

```r
integrate(dMix, 60, 70,
          c(mu1_hat, sigma1_hat,
            mu2_hat, sigma2_hat, p_hat3))
```

```
## 0.08981431 with absolute error < 1e-15
```