Series 3

1. **a)** The artificial data set shown in Figure 1 is generated from a mixture of two normal distributions (for details, see b)). Imagine you would not know anything about the distribution, and you would want to estimate the density by use of a kernel density estimator with Gaussian kernel. Guess a good bandwidth.

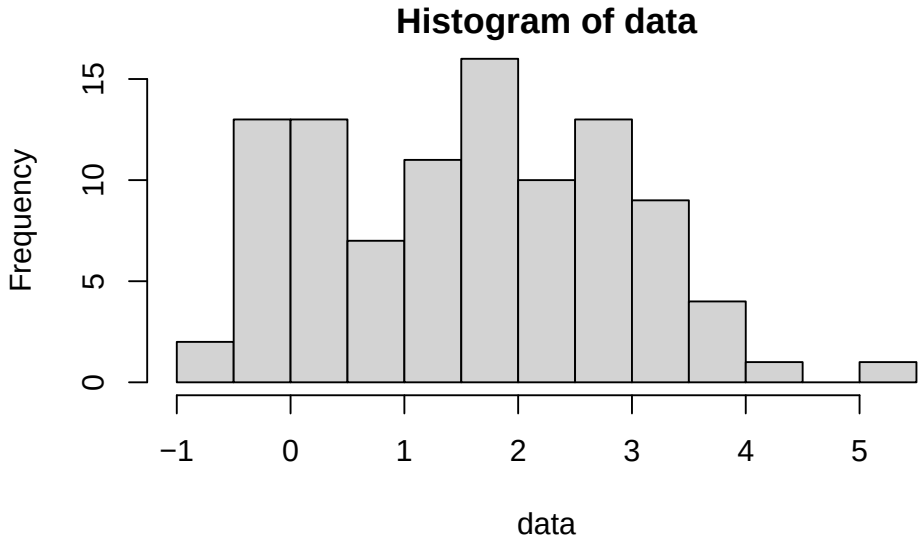

Figure 1: Artificial data from mixture of two normal distributions

b) Carry out a simulation to evaluate the quality of bandwidths for kernel density estimation for a mixture of two normal distributions where a point is generated by a $\mathcal{N}(0, 0.01)$-distribution with probability 0.2 and by a $\mathcal{N}(2, 1)$-distribution with probability 0.8. Repeat the following code 200 times for multiple bandwidth values: $h = 2^k$ where $k \in \{-5, -4.8, \ldots, 1\}$ as well as for a plug-in estimate of the optimal bandwidth (see below):

R-hints:

```
> ## You can create a list for the bandwidth values using the following command:
> bandwidths <- c(as.list(2^seq(-5,1,0.2)),"sj")
> ## Elements of a list can be accessed by using double square brackets.
> bandwidths[[5]]
```

1. Generate a data set of 100 points from the mixture distribution from above:

```
> set.seed(79)
> data <- numeric(100)
> for(i in 1:100){
   p <- runif(1, min = 0, max = 1)
   if (p < 0.2)
     data[i] <- rnorm(1, mean = 0, sd = sqrt(0.01))
   else
     data[i] <- rnorm(1, mean = 2, sd = 1)
 }
```

2. Compute the kernel density estimator with the function `density`. Consult `help(density)` to understand the syntax.

```
> ke <- density(data, bw = 0.1, n = 61, from = -1, to = 5)
```

An estimated plug-in bandwidth is obtained by using `bw ="sj"`.
In the lecture, we saw that the goodness of a kernel estimate can be measured using the integrated mean squared error (IMSE). In practice, the IMSE can be approximated by summing the squared difference between the kernel estimator and the true density values (up to a constant factor).

```
> ## Compute the true density at the given data points
> xpts <- seq(-1,5,0.1)
> dmix <- 0.2 * dnorm(xpts, mean = 0, sd = sqrt(0.01)) +
      0.8 * dnorm(xpts, mean = 2, sd = 1)
> ## Take the mean of the squared differences
> mean((ke$y - dmix)^2)
```

Note that the commands given above are suitable for a single execution but not necessarily for the full simulation. For example, you will need a vector of the qualities of all simulation runs. You may define a matrix for the qualities to store the results for all different bandwidths. Consider `help(matrix), help(apply)`. Compute and compare the averaged quality of the kernel estimators for the different bandwidths using appropriate plots, and find out what the optimal bandwidth is.

c) Repeat the whole project with the Epanechnikov kernel, which can be obtained by specifying `kernel="epanechnikov"` in `density`.

d) Now obtain the optimal bandwidth via a different route, this time by computing the log-likelihood function as a function of the bandwidth. Use the same bandwidths as before, except for "sj". Compare with your results from b) and c).

e) Judge your guess from a). Did you guess a good bandwidth?

Preliminary discussion: Friday, March 18.

Deadline: Friday, March 25.