



# **A Handbook of Statistical Analyses Using R**

---

Brian S. Everitt and Torsten Hothorn



---

## CHAPTER 7

# Density Estimation: Erupting Geysers and Star Clusters

---

### 7.1 Introduction

### 7.2 Density Estimation

### 7.3 Analysis Using R

#### *7.3.1 A Parametric Density Estimate for the Old Faithful Data*

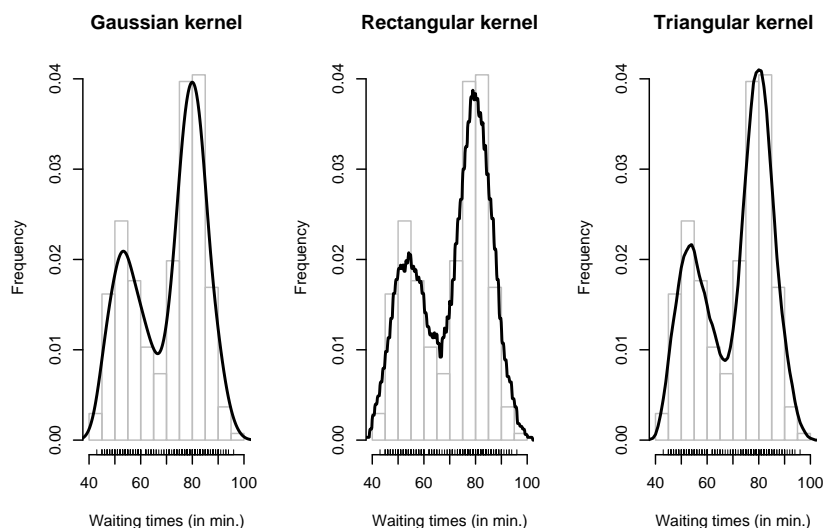
```
R> logL <- function(param, x) {  
+   d1 <- dnorm(x, mean = param[2], sd = param[3])  
+   d2 <- dnorm(x, mean = param[4], sd = param[5])  
+   -sum(log(param[1] * d1 + (1 - param[1]) * d2))  
+ }  
R> startparam <- c(p = 0.5, mu1 = 50, sd1 = 3, mu2 = 80, sd2 = 3)  
R> opp <- optim(startparam, logL, x = faithful$waiting, method = "L-BFGS-B",  
+   lower = c(0.01, rep(1, 4)),  
+   upper = c(0.99, rep(200, 4)))  
R> opp  
  
$par  
      p      mu1      sd1      mu2      sd2  
0.360891 54.612122 5.872379 80.093415 5.867289  
  
$value  
[1] 1034.002  
  
$counts  
function gradient  
      55      55  
  
$convergence  
[1] 0
```

Of course, optimising the appropriate likelihood ‘by hand’ is not very convenient. In fact, (at least) two packages offer high-level functionality for estimating mixture models. The first one is package *mclust* (Fraley et al., 2006) implementing the methodology described in Fraley and Raftery (2002). Here,

```

1 R> data("faithful", package = "datasets")
2 R> x <- faithful$waiting
3 R> layout(matrix(1:3, ncol = 3))
4 R> hist(x, xlab = "Waiting times (in min.)", ylab = "Frequency",
5 +       probability = TRUE, main = "Gaussian kernel", border = "gray")
6 R> lines(density(x, width = 12), lwd = 2)
7 R> rug(x)
8 R> hist(x, xlab = "Waiting times (in min.)", ylab = "Frequency",
9 +       probability = TRUE, main = "Rectangular kernel", border = "gray")
10 R> lines(density(x, width = 12, window = "rectangular"), lwd = 2)
11 R> rug(x)
12 R> hist(x, xlab = "Waiting times (in min.)", ylab = "Frequency",
13 +       probability = TRUE, main = "Triangular kernel", border = "gray")
14 R> lines(density(x, width = 12, window = "triangular"), lwd = 2)
15 R> rug(x)

```



**Figure 7.1** Density estimates of the geyser eruption data imposed on a histogram of the data.

a Bayesian information criterion (BIC) is applied to choose the form of the mixture model:

```

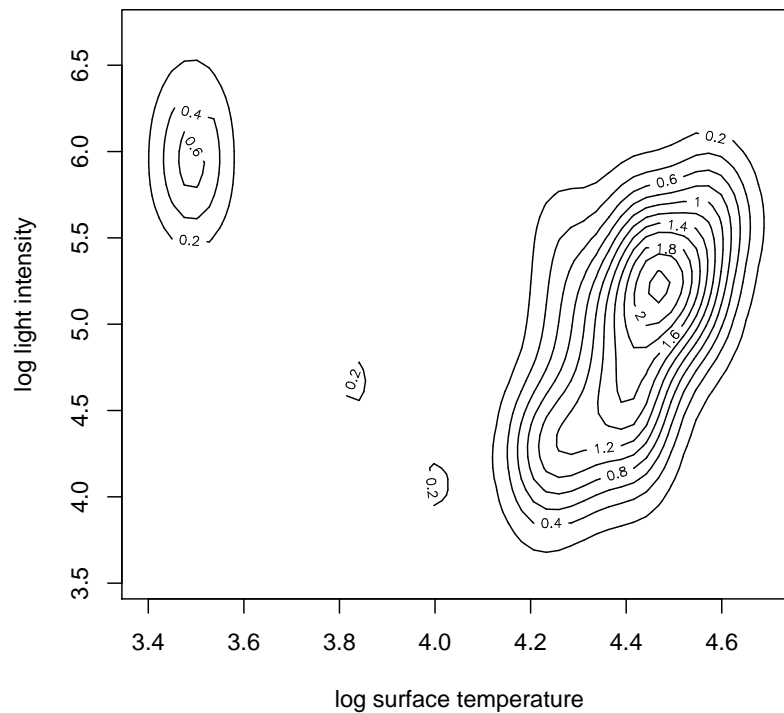
R> library("mclust")
R> mc <- Mclust(faithful$waiting)
R> mc

```

*best model: E with 2 components*

and the estimated means are

```
R> library("KernSmooth")
R> data("CYGOB1", package = "HSAUR")
R> CYGOB1d <- bkde2D(CYGOB1, bandwidth = sapply(CYGOB1, dpik))
R> contour(x = CYGOB1d$x1, y = CYGOB1d$x2, z = CYGOB1d$fhat,
+         xlab = "log surface temperature", ylab = "log light intensity")
```



**Figure 7.2** A contour plot of the bivariate density estimate of the CYGOB1 data, i.e., a two-dimensional graphical display for a three-dimensional problem.

```
R> mc$parameters$mean
```

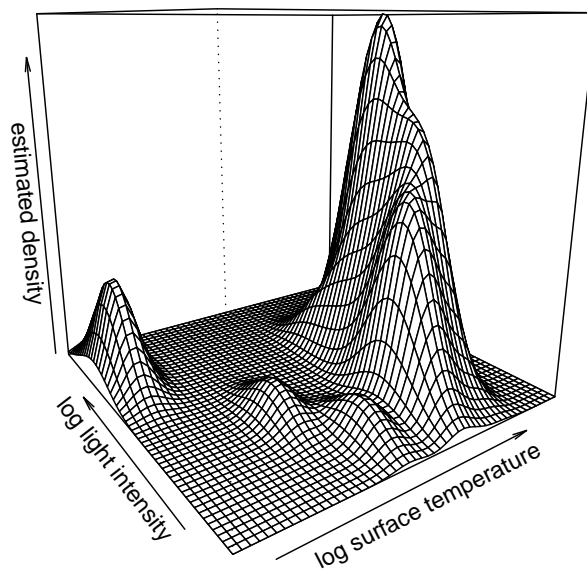
```
      1      2
54.61911 80.09384
```

with estimated standard deviation (found to be equal within both groups)

```
R> sqrt(mc$parameters$variance$sigma$sd)
```

```
[1] 5.86848
```

```
R> persp(x = CYGOB1d$x1, y = CYGOB1d$x2, z = CYGOB1d$fhat,
+        xlab = "log surface temperature", ylab = "log light intensity",
+        zlab = "estimated density", theta = -35, axes = TRUE, box = TRUE)
```



**Figure 7.3** The bivariate density estimate of the CYGOB1 data, here shown in a three-dimensional fashion using the `persp` function.

The proportion is  $\hat{p} = 0.36$ . The second package is called *flexmix* whose functionality is described by [Leisch \(2004\)](#). A mixture of two normals can be fitted using

```
R> library("flexmix")
R> fl <- flexmix(waiting ~ 1, data = faithful, k = 2)
```

with  $\hat{p} = 0.36$  and estimated parameters

```
R> parameters(fl, component = 1)
```

```
$coef
(Intercept)
  54.6287
```

```

$sigma
[1] 5.895234
R> parameters(f1, component = 2)
$coef
(Intercept)
  80.09858

```

```

$sigma
[1] 5.871749

```

We can get standard errors for the five parameter estimates by using a bootstrap approach (see [Efron and Tibshirani, 1993](#)). The original data are slightly perturbed by drawing  $n$  out of  $n$  observations *with replacement* and those artificial replications of the original data are called *bootstrap samples*. Now, we can fit the mixture for each bootstrap sample and assess the variability of the estimates, for example using confidence intervals. Some suitable R code based on the `Mclust` function follows. First, we define a function that, for a bootstrap sample `indx`, fits a two-component mixture model and returns  $\hat{p}$  and the estimated means (note that we need to make sure that we always get an estimate of  $p$ , not  $1 - p$ ):

```

R> library("boot")
R> fit <- function(x, indx) {
+   a <- Mclust(x[indx], minG = 2, maxG = 2)$parameters
+   if (a$pro[1] < 0.5)
+     return(c(p = a$pro[1], mu1 = a$mean[1], mu2 = a$mean[2]))
+   return(c(p = 1 - a$pro[1], mu1 = a$mean[2], mu2 = a$mean[1]))
+ }

```

The function `fit` can now be fed into the `boot` function ([Canty and Ripley, 2006](#)) for bootstrapping (here 1000 bootstrap samples are drawn)

```

R> bootpara <- boot(faithful$waiting, fit, R = 1000)

```

We assess the variability of our estimates  $\hat{p}$  by means of adjusted bootstrap percentile (BCa) confidence intervals, which for  $\hat{p}$  can be obtained from

```

R> boot.ci(bootpara, type = "bca", index = 1)

```

```

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

```

```

CALL :
boot.ci(boot.out = bootpara, type = "bca", index = 1)

```

```

Intervals :
Level      BCa
95%      ( 0.3041,  0.4233 )
Calculations and Intervals on Original Scale

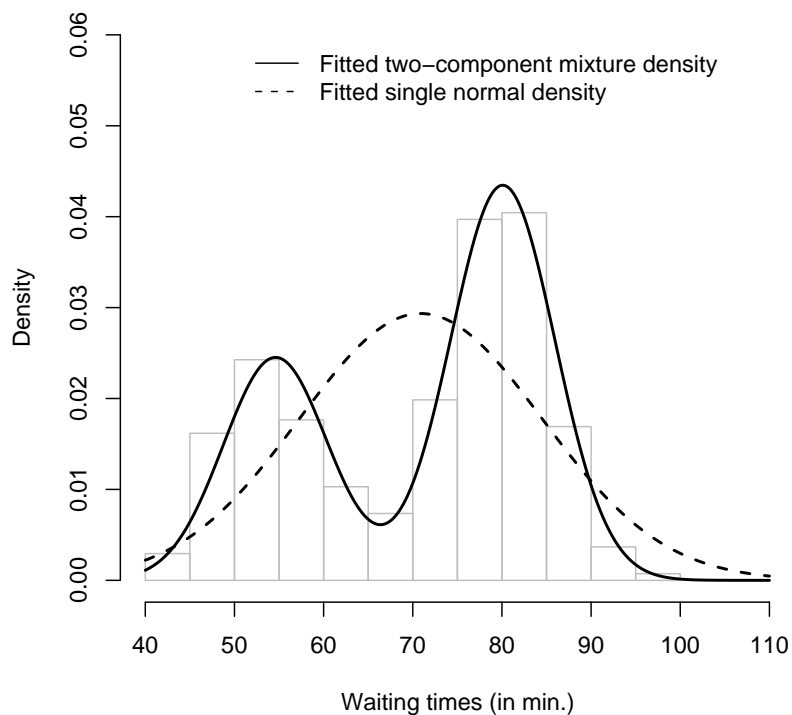
```

We see that there is a reasonable variability in the mixture model, however, the means in the two components are rather stable, as can be seen from

```

R> opar <- as.list(opp$par)
R> rx <- seq(from = 40, to = 110, by = 0.1)
R> d1 <- dnorm(rx, mean = opar$mu1, sd = opar$sd1)
R> d2 <- dnorm(rx, mean = opar$mu2, sd = opar$sd2)
R> f <- opar$p * d1 + (1 - opar$p) * d2
R> hist(x, probability = TRUE, xlab = "Waiting times (in min.)",
+       border = "gray", xlim = range(rx), ylim = c(0, 0.06), main = "")
R> lines(rx, f, lwd = 2)
R> lines(rx, dnorm(rx, mean = mean(x), sd = sd(x)), lty = 2, lwd = 2)
R> legend(50, 0.06, legend = c("Fitted two-component mixture density",
+                               "Fitted single normal density"),
+        lty = 1:2, bty = "n")

```



**Figure 7.4** Fitted normal density and two-component normal mixture for geyser eruption data.



```
R> boot.ci(bootpara, type = "bca", index = 2)
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates
```

```
CALL :
boot.ci(boot.out = bootpara, type = "bca", index = 2)
```

```
Intervals :
Level      BCa
95%      (53.42, 56.07 )
Calculations and Intervals on Original Scale
```

for  $\hat{\mu}_1$  and for  $\hat{\mu}_2$  from

```
R> boot.ci(bootpara, type = "bca", index = 3)
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates
```

```
CALL :
boot.ci(boot.out = bootpara, type = "bca", index = 3)
```

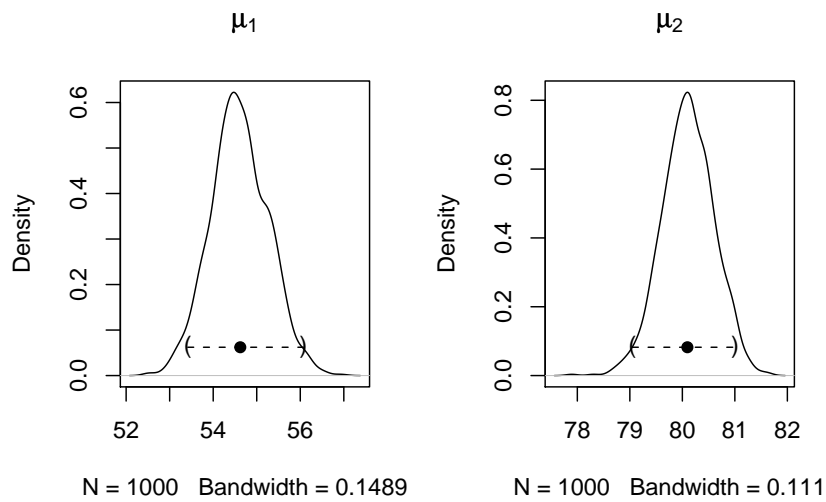
```
Intervals :
Level      BCa
95%      (79.05, 81.01 )
Calculations and Intervals on Original Scale
```

Finally, we show a graphical representation of both the bootstrap distribution of the mean estimates *and* the corresponding confidence intervals. For convenience, we define a function for plotting, namely

```
R> bootplot <- function(b, index, main = "") {
+   dens <- density(b$t[,index])
+   ci <- boot.ci(b, type = "bca", index = index)$bca[4:5]
+   est <- b$t0[index]
+   plot(dens, main = main)
+   y <- max(dens$y) / 10
+   segments(ci[1], y, ci[2], y, lty = 2)
+   points(ci[1], y, pch = "(")
+   points(ci[2], y, pch = ")")
+   points(est, y, pch = 19)
+ }
```

The element `t` of an object created by `boot` contains the bootstrap replications of our estimates, i.e., the values computed by `fit` for each of the 1000 bootstrap samples of the `geyser` data. First, we plot a simple density estimate and then construct a line representing the confidence interval. We apply this function to the bootstrap distributions of our estimates  $\hat{\mu}_1$  and  $\hat{\mu}_2$  in Figure 7.5.

```
R> layout(matrix(1:2, ncol = 2))  
R> bootplot(bootpara, 2, main = expression(mu[1]))  
R> bootplot(bootpara, 3, main = expression(mu[2]))
```



**Figure 7.5** Bootstrap distribution and confidence intervals for the mean estimates of a two-component mixture for the geyser data.

---

## Bibliography

---

- Canty, A. and Ripley, B. D. (2006), *boot: Bootstrap R (S-PLUS) Functions (Canty)*, URL <http://CRAN.R-project.org>, R package version 1.2-26.
- Efron, B. and Tibshirani, R. J. (1993), *An Introduction to the Bootstrap*, London, UK: Chapman & Hall/CRC.
- Fraley, C. and Raftery, A. E. (2002), “Model-based clustering, discriminant analysis, and density estimation,” *Journal of the American Statistical Association*, 97, 611–631.
- Fraley, C., Raftery, A. E., and Wehrens, R. (2006), *mclust: Model-based Cluster Analysis*, URL <http://www.stat.washington.edu/mclust>, R package version 3.0-0.
- Leisch, F. (2004), “FlexMix: A general framework for finite mixture models and latent class regression in R,” *Journal of Statistical Software*, 11, URL <http://www.jstatsoft.org/v11/i08/>.