A Handbook of Statistical Analyses Using R
— 2nd Edition

Brian S. Everitt and Torsten Hothorn
Density Estimation: Erupting Geysers and Star Clusters

8.1 Introduction

8.2 Density Estimation

The three kernel functions are implemented in R as shown in lines 1–3 of Figure 8.1. For some grid x, the kernel functions are plotted using the R statements in lines 5–11 (Figure 8.1).

The kernel estimator \( \hat{f} \) is a sum of ‘bumps’ placed at the observations. The kernel function determines the shape of the bumps while the window width \( h \) determines their width. Figure 8.2 (redrawn from a similar plot in Silverman [1986]) shows the individual bumps \( n^{-1}h^{-1}K((x - x_i)/h) \), as well as the estimate \( \hat{f} \) obtained by adding them up for an artificial set of data points:

```r
> x <- c(0, 1, 1.1, 1.5, 1.9, 2.8, 2.9, 3.5)
> n <- length(x)
```

For a grid:

```r
> xgrid <- seq(from = min(x) - 1, to = max(x) + 1, by = 0.01)
```

on the real line, we can compute the contribution of each measurement in \( x \) with \( h = 0.4 \) by the Gaussian kernel (defined in Figure 8.1, line 3) as follows:

```r
> h <- 0.4
> bump <- sapply(x, function(a) gauss((xgrid - a)/h)/(n * h))
```

A plot of the individual bumps and their sum, the kernel density estimate \( \hat{f} \), is shown in Figure 8.2.

8.3 Analysis Using R

8.3.1 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))
+ }
> startparam <- c(p = 0.5, mu1 = 50, sd1 = 3, mu2 = 80, sd2 = 3)
> opp <- optim(startparam, logL, x = faithful$waiting,
```
R> rec <- function(x) (abs(x) < 1) * 0.5
R> tri <- function(x) (abs(x) < 1) * (1 - abs(x))
R> gauss <- function(x) 1/sqrt(2*pi) * exp(-x^2/2)
R> x <- seq(from = -3, to = 3, by = 0.001)
R> plot(x, rec(x), type = "l", ylim = c(0,1), lty = 1,
R+ ylab = expression(K(x)))
R> lines(x, tri(x), lty = 2)
R> lines(x, gauss(x), lty = 3)
R> legend(-3, 0.8, legend = c("Rectangular", "Triangular",
R+ "Gaussian"), lty = 1:3, title = "kernel functions",
R+ bty = "n")

Figure 8.1 Three commonly used kernel functions.
ANALYSIS USING R

R> plot(xgrid, rowSums(bumps), ylab = expression(hat(f)(x)),
    + type = "l", xlab = "x", lwd = 2)
R> rug(x, lwd = 2)
R> out <- apply(bumps, 2, function(b) lines(xgrid, b))

Figure 8.2 Kernel estimate showing the contributions of Gaussian kernels evaluated for the individual observations with bandwidth $h = 0.4$.

+ method = "L-BFGS-B",
+ lower = c(0.01, rep(1, 4)),
+ upper = c(0.99, rep(200, 4))
R> opp

$\begin{array}{llll}
  p & \mu 1 & \sigma 1 & \mu 2 & \sigma 2 \\
  0.361 & 54.612 & 5.872 & 80.093 & 5.867 \\
\end{array}$

$value

[1] 1034

$counts$
R> epa <- function(x, y)
+   ((x^2 + y^2) < 1) * 2/pi * (1 - x^2 - y^2)
R> x <- seq(from = -1.1, to = 1.1, by = 0.05)
R> epavals <- sapply(x, function(a) epa(a, x))
R> persp(x = x, y = x, z = epavals, xlab = "x", ylab = "y",
+         zlab = expression(K(x, y)), theta = -35, axes = TRUE,
+         box = TRUE)

Figure 8.3  Epanechnikov kernel for a grid between \((-1.1, -1.1)\) and \((1.1, 1.1)\).
ANALYSIS USING R

```r
R> data("faithful", package = "datasets")
R> x <- faithful$waiting
R> layout(matrix(1:3, ncol = 3))
R> hist(x, xlab = "Waiting times (in min.)", ylab = "Frequency",
+     probability = TRUE, main = "Gaussian kernel",
+     border = "gray")
R> lines(density(x, width = 12), lwd = 2)
R> rug(x)
R> hist(x, xlab = "Waiting times (in min.)", ylab = "Frequency",
+     probability = TRUE, main = "Rectangular kernel",
+     border = "gray")
R> lines(density(x, width = 12, window = "rectangular"), lwd = 2)
R> rug(x)
R> hist(x, xlab = "Waiting times (in min.)", ylab = "Frequency",
+     probability = TRUE, main = "Triangular kernel",
+     border = "gray")
R> lines(density(x, width = 12, window = "triangular"), lwd = 2)
R> rug(x)
```

Figure 8.4  Density estimates of the geyser eruption data imposed on a histogram
of the data.
Figure 8.5 A contour plot of the bivariate density estimate of the CYGOB1 data, i.e., a two-dimensional graphical display for a three-dimensional problem.

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., 2012).
implementing the methodology described in [Fraley and Raftery (2002)]. Here, 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

'Mclust' model object: (E,2)

Available components:

Figure 8.6 The bivariate density estimate of the CYGOB1 data, here shown in a three-dimensional fashion using the persp function.
and the estimated means are

\begin{verbatim}
R> mc$parameters$mean
   1   2
54.6 80.1
\end{verbatim}

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

\begin{verbatim}
R> sqrt(mc$parameters$variance$sigmasq)
[1] 5.87
\end{verbatim}

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

\begin{verbatim}
R> library("flexmix")
R> fl <- flexmix(waiting ~ 1, data = faithful, k = 2)
\end{verbatim}

with \( \hat{p} = 0.52 \) and estimated parameters

\begin{verbatim}
R> parameters(fl, component = 1)
Comp.1
  coef.(Intercept) 70.8
  sigma 13.6
R> parameters(fl, component = 2)
Comp.2
  coef.(Intercept) 71.0
  sigma 13.6
\end{verbatim}

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 \text{with replacement} and those artificial replications of the original data are called \textit{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 \texttt{R} code based on the \texttt{Mclust} function follows. First, we define a function that, for a bootstrap sample \texttt{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 \( 1 - p \)):

\begin{verbatim}
R> library("boot")
R> fit <- function(x, indx) {
+   a <- Mclust(x[indx], minG = 2, maxG = 2,
+     modelNames="E")$parameters
+   if (a$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]))
+ }
\end{verbatim}
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, lty = 1:2, bty = "n",
+     legend = c("Fitted two-component mixture density",
+                 "Fitted single normal density"))

Figure 8.7 Fitted normal density and two-component normal mixture for geyser eruption data.
The function `fit` can now be fed into the `boot` function (Canty and Ripley 2012) for bootstrapping (here 1000 bootstrap samples are drawn)

```r
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
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%  (0.304, 0.423 )
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
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%  (53.4, 56.1 )
Calculations and Intervals on Original Scale
```

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

```r
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, 81 )
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
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 = "*")
```

ANALYSIS USING R

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

![Bootstrap distribution and confidence intervals for the mean estimates of a two-component mixture for the geyser data.](image)

Figure 8.8  Bootstrap distribution and confidence intervals for the mean estimates of a two-component mixture for the geyser data.

```r
+ points(ci[2], y, pch = ")")
+ points(est, y, pch = 19)
+ }
```

The element 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 8.8.
Bibliography


