Week 5 Module 6: Histograms and Curves

CSCI E-5a: Introduction to R

Let's clear the global environment:

```
rm( list = ls() )
```

And now let's load in the Module 6 R objects:

```
load( "Module 6 R Objects.Rdata" )
```

Module Overview and Learning Objectives

Hello! And welcome to Module 6: Histograms and Curves.

In this module, we'll examine methods for superimposing curves on histograms.

- In section 1, we'll review the theory of the normal distribution.
- In section 2, we'll see how to superimpose a density curve over a histogram.
- In section 3, we'll learn how to determine the best-fitting density curve for a normally distributed set of data.
- In section 4, we'll explore non-parametric density curves.

When you've completed this module, you should be able to:

- Superimpose a normal density curve over a histogram.
- Determine the parameters for the best-fitting normal density curve.
- Superimpose a non-parametric density curve over a histogram.
- Draw non-parametric empirical density curves with normal density curves or other non-parametric density curves.

There are two new built-in R functions in this module:

- `dnorm()`
- `density()`

All right! Let's get started with a review of the normal distribution.

Section 1: Review of the Normal Distribution

Main Idea: *Let's review the normal distribution*

In this section, we'll explore the theory of the normal distribution.

Much of classical statistics is based on objects called *normal distributions*.

A *parameter* of a distribution is a value that must be specified in order to completely determine a normal distribution.

A normal distribution has two parameters:

- The first parameter, denoted by the Greek letter μ, is the *mean* of the distribution, which describes the central location of the density curve.
- The second parameter, denoted by the Greek letter σ, is the *standard deviation* of the distribution, which describes how spread out the distribution is.

Once we've specified the mean and standard deviation for a normal distribution, we've completely determined it.

Every normal distribution has an associated "density curve".

Normal density curves have a characteristic symmetric "mountain" shape, where the center of the mountain is determined by the value of the mean μ, and the width of the mountain is determined by the value of the standard deviation σ.

The built-in R function `dnorm()` can calculate normal density curves.

The `dnorm()` function takes three input arguments:

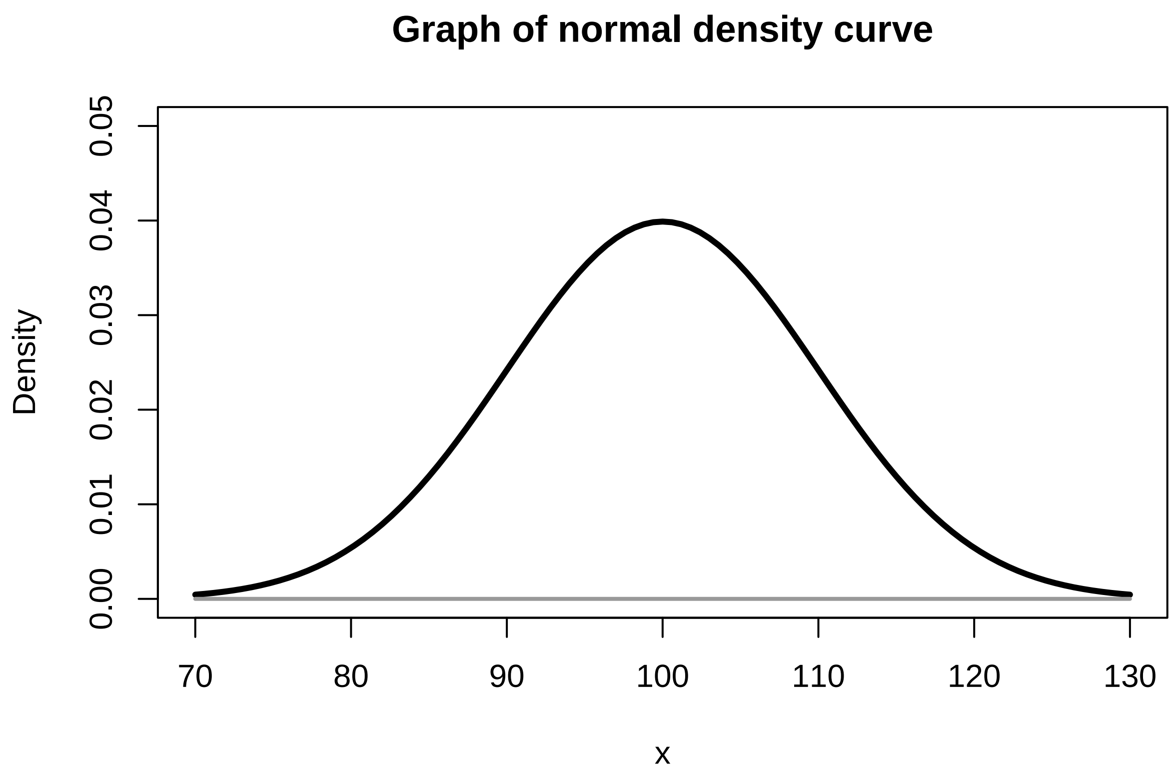
- The first input argument is `x`, and this is just the value at which the density function is to be evaluated.
- The second input argument is `mean`, and this is the mean of the normal distribution.
- The third input argument is `sd`, and this is the standard deviation of the normal distribution.

The `dnorm()` function returns the value of the normal density function at the specified value of `x`, for the normal distribution with population mean `mean` and population standard deviation `sd`.

Let's make a graph of the density function for the normal distribution, with mean $\mu = 100$ and standard deviation $\sigma = 10$:

```
curve(
    expr = dnorm(x, mean = 100, sd = 10),
    main = "Graph of normal density curve",
    xlim = c(70, 130),
    ylim = c(0, 0.05),
    xlab = "x",
    ylab = "Density",
    lty = "solid",
    lwd = 3,
    col = "black"
)
```

```
segments(
    x0 = 70,
    y0 = 0,
    x1 = 130,
    y1 = 0,
    lty = "solid",
    lwd = 2,
    col = "gray60"
)
```



So that's a quick overview of the theory of the normal distribution.

Now let's see how to superimpose a mathematical curve over a histogram.

Exercise 6.1: Drawing a normal density curve

Construct a graph of the density function for the normal distribution, with mean $\mu = 45$ and standard deviation $\sigma = 7$.

Solution

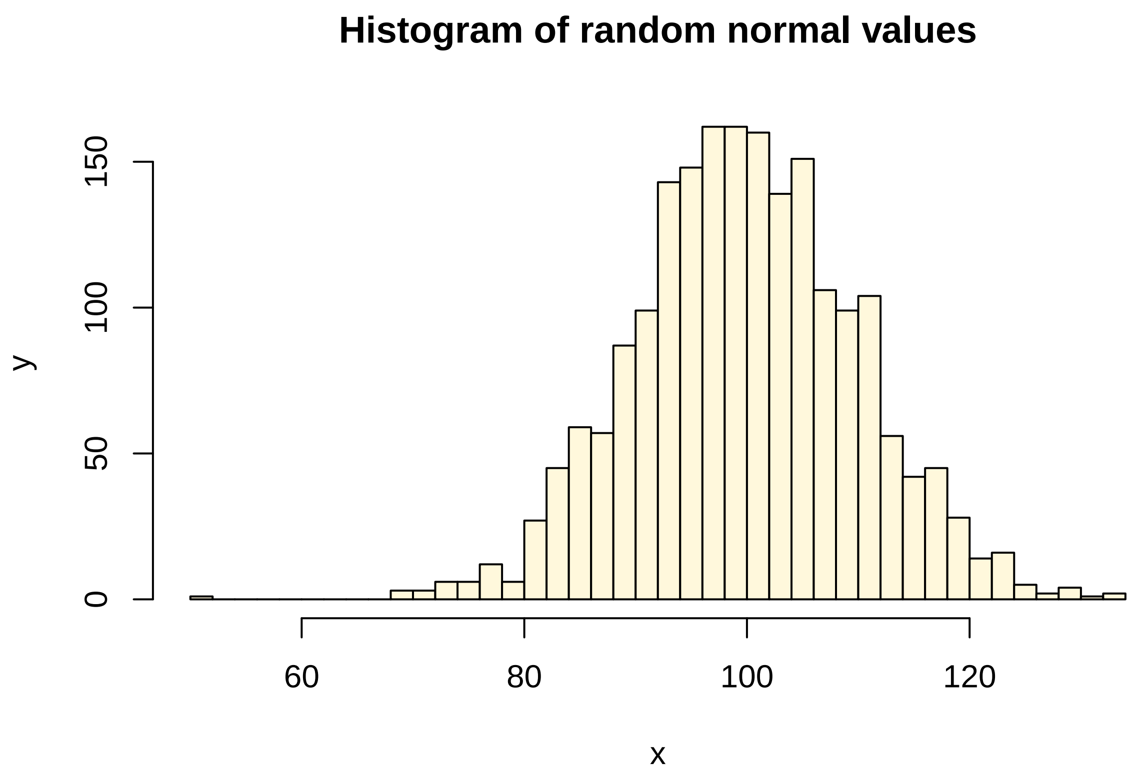
Section 2: Superimposing a Normal Density Curve

> **Main Idea:** *We can superimpose a density curve on a histogram*

In this section, we'll see how to superimpose a density curve over a histogram.

Let's make a histogram of the values in the `example.6.1.data` vector:

```
hist(
    x = example.6.1.data,
    main = "Histogram of random normal values",
    col = "cornsilk",
    xlab = "x",
    ylab = "y",
    breaks = 50
)
```



This data looks like it might come from a normal distribution.

To help us decide if this data comes from a normal distribution, we will *superimpose* the density curve of a particular normal distribution on top of the histogram of the observed data:

- First, we create the histogram using the `hist()` function.
- Next, we draw the density curve on top of the histogram using the `curve()` function.

In order to do this, we need to set two options:

- In the `hist()` function, we need to set `prob = TRUE`.
- In the `curve()` function, we need to set `add = TRUE`.

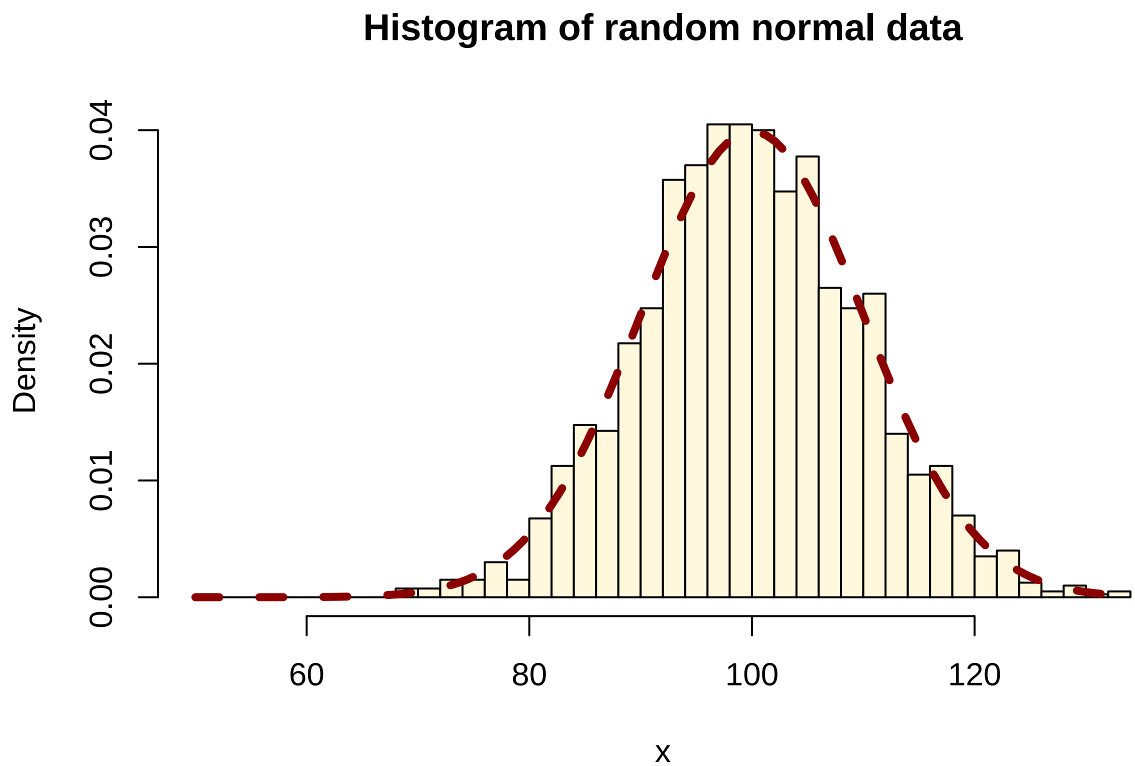
The `prob = TRUE` option in the `hist()` function means that the histogram will display the density on the y axis.

If you don't do this, the density curve will appear as a flat line along the x-axis.

The `add = TRUE` option in the `curve()` function means that the curve will be drawn on the existing plot instead of creating a new plot.

Let's redraw the histogram of `example.6.1.data`, and this time we'll superimpose a normal density curve with a mean of 100 and a standard deviation of 10:

```
hist(
    x = example.6.1.data,
    prob = TRUE,
    main = "Histogram of random normal data",
    xlab = "x",
    ylab = "Density",
    col = "cornsilk",
    breaks = 50
)

curve(
    dnorm(
        x,
        mean = 100,
        sd = 10
    ),
    lty = "dashed",
    lwd = 4,
    col = "darkred",
    add = TRUE
)
```



So that's how to superimpose a normal density curve over a histogram.

Now let's see how to fit a normal density curve to data.

Exercise 6.2: Superimposing a normal density curve

Draw the histogram of the data in `exercise.6.2.data`. Then superimpose a normal density curve with a mean of 45 and a standard deviation of 7:

Solution

Section 3: Fitting a Normal Density Curve

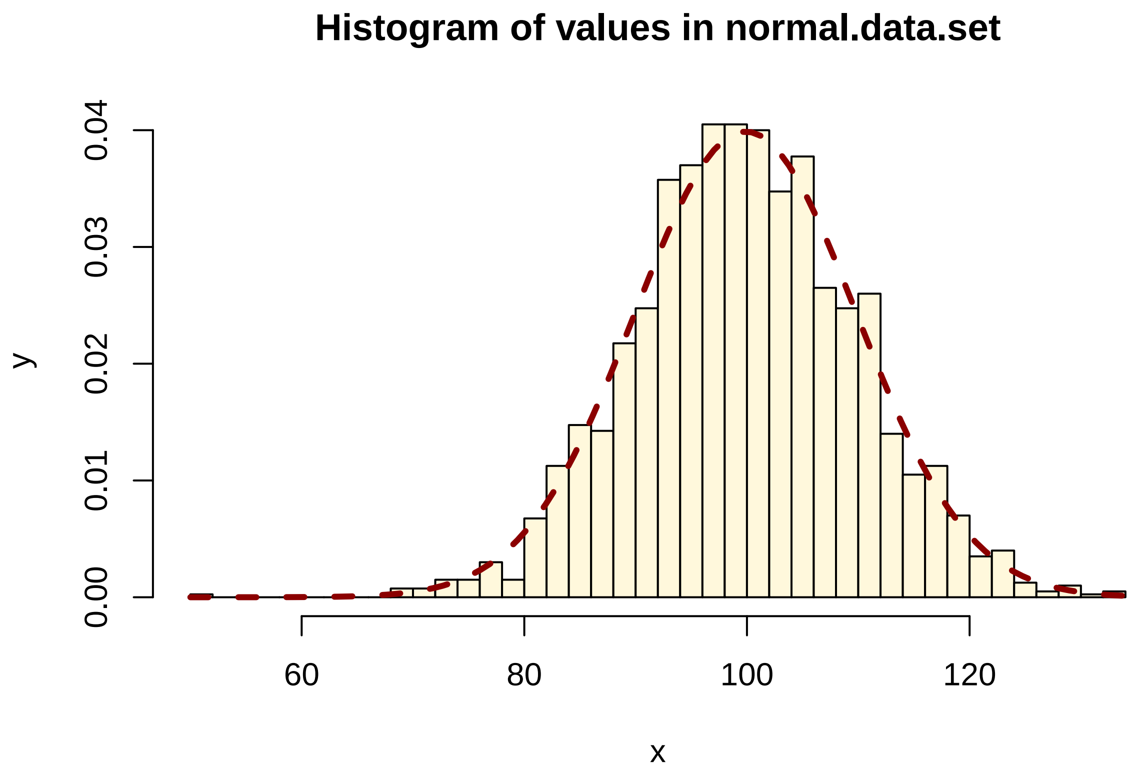
> **Main Idea:** *We can fit a normal density curve to data*

In this section, we'll learn how to determine the best-fitting density curve for a normally distributed set of data.

How do we choose the values of the `mean` and `sd` parameters when we fit a normal distribution to this data?

According to statistical theory, the best normal distribution to choose to fit this data is the normal distribution that has a mean equal to the observed sample mean, and a standard deviation equal to the observed sample standard deviation.

```
hist(
    example.6.1.data,
    main = "Histogram of values in normal.data.set",
    col = "cornsilk",
    xlab = "x",
    ylab = "y",
    prob = TRUE,
    breaks = 50
)


sample.mean <-
    mean( example.6.1.data )

sample.standard.deviation <-
    sd( example.6.1.data )

curve(
    dnorm(
        x,
        mean = sample.mean,
        sd = sample.standard.deviation
    ),
    lty = "dashed",
    lwd = 3,
    col = "darkred",
    add = TRUE
)
```

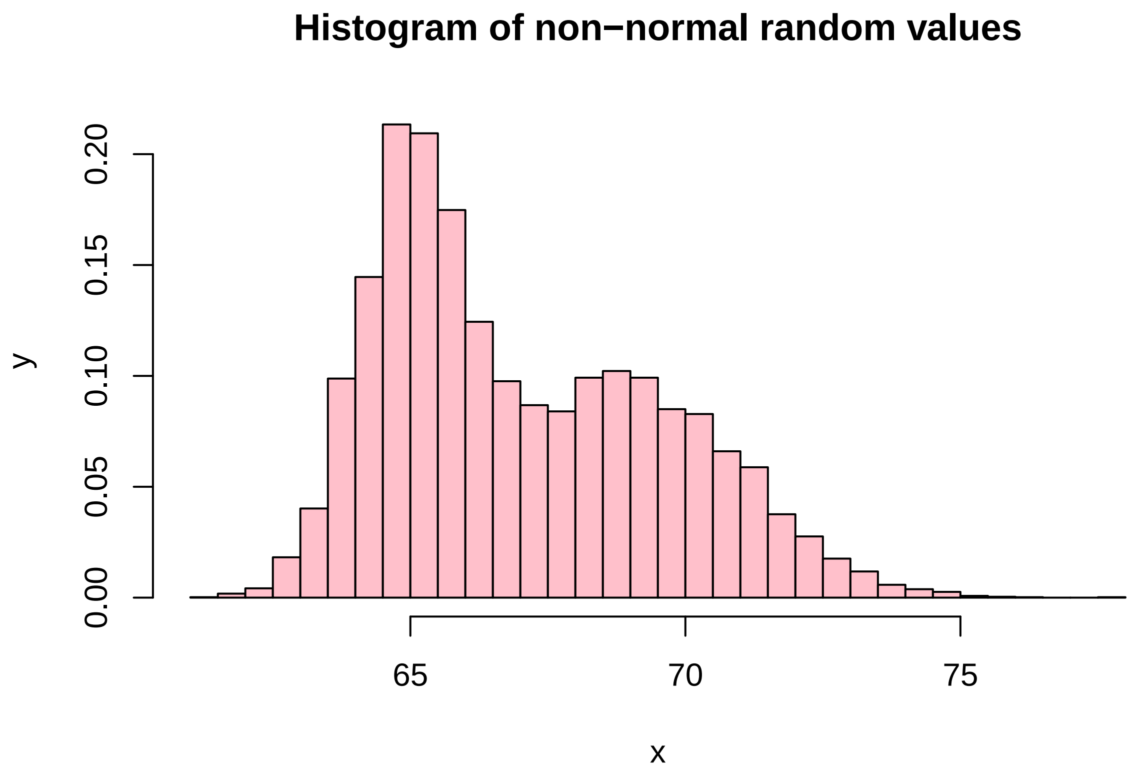


Notice that this method *requires* that the population is normally distributed; the only question is to determine the best-fitting curve from among all possible normal distributions.

If the population is not normally distributed, then this method won't make much sense.

To see an example of this, let's make a histogram of `example.6.2.data`:

```
hist(
    example.6.2.data,
    main = "Histogram of non-normal random values",
    col = "pink",
    xlab = "x",
    ylab = "y",
    prob = TRUE,
    breaks = 50
)
```

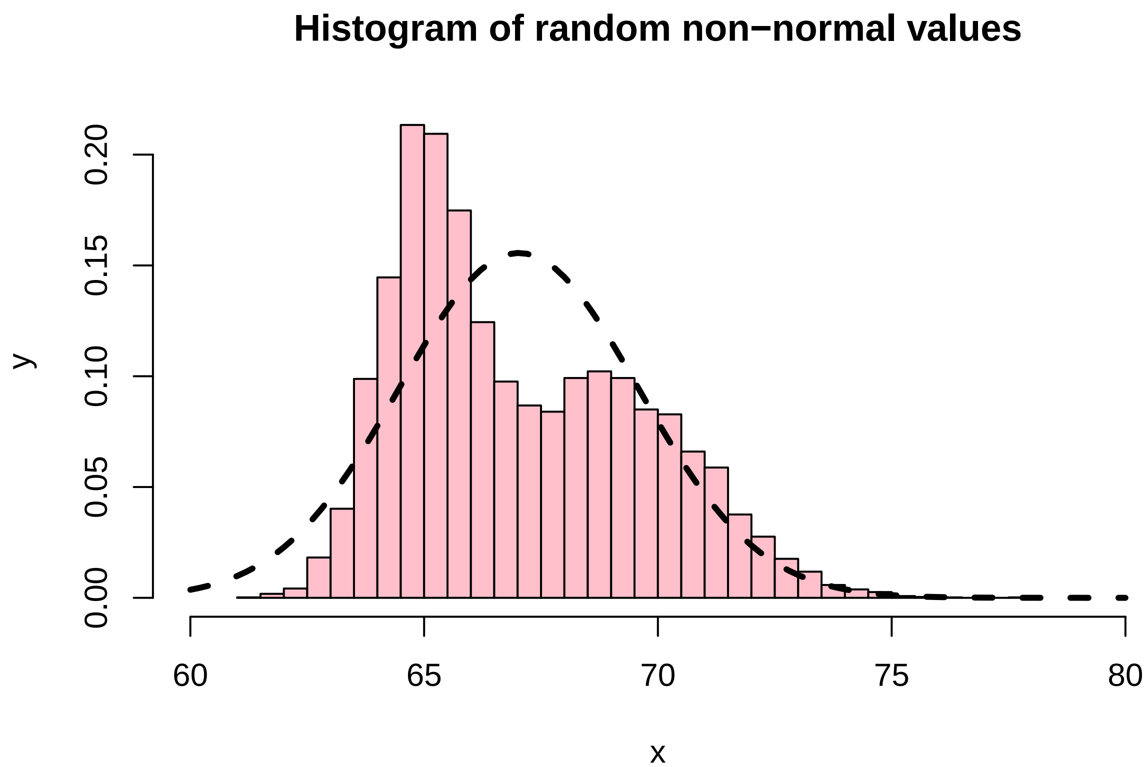


This doesn't have the familiar mountain shape of the normal distribution.

Now we'll attempt to fit a normal curve to this data by using the observed sample mean and sample variance values:

```
hist(
    example.6.2.data,
    xlim = c(60, 80),
    main = "Histogram of random non-normal values",
    col = "pink",
    xlab = "x",
    ylab = "y",
    prob = TRUE,
    breaks = 50
)

sample.mean <-
    mean( example.6.2.data )

sample.standard.deviation <-
    sd( example.6.2.data )

curve(
    dnorm(
        x,
        mean = sample.mean,
        sd = sample.standard.deviation
```

```
    ),
    lty = 2,
    lwd = 3,
    col = "black",
    add = TRUE
)
```



This didn't work very well at all.

The conclusion here is that you shouldn't try to fit a normal density curve to data that isn't normally distributed.

So that's how to fit a normal density curve to data.

Now let's see how to relax the assumption of normality, and construct density curves for non-normal data.

Exercise 6.3: Fitting a normal density curve

Calculate the sample mean and the sample standard deviation of the values in `exercise.6.2.data`. Redraw the histogram, and fit a normal density curve using the sample mean and sample standard deviation.

Solution

Section 4: Fitting a Kernel Density Curve

> **Main Idea:** *We can fit a non-parametric kernel density curve to data*

In this section, we'll explore non-parametric kernel density curves.

When we fit a normal density curve to data, we are assuming that the data follows a normal distribution, and we just have to determine the mean and standard deviation to use.

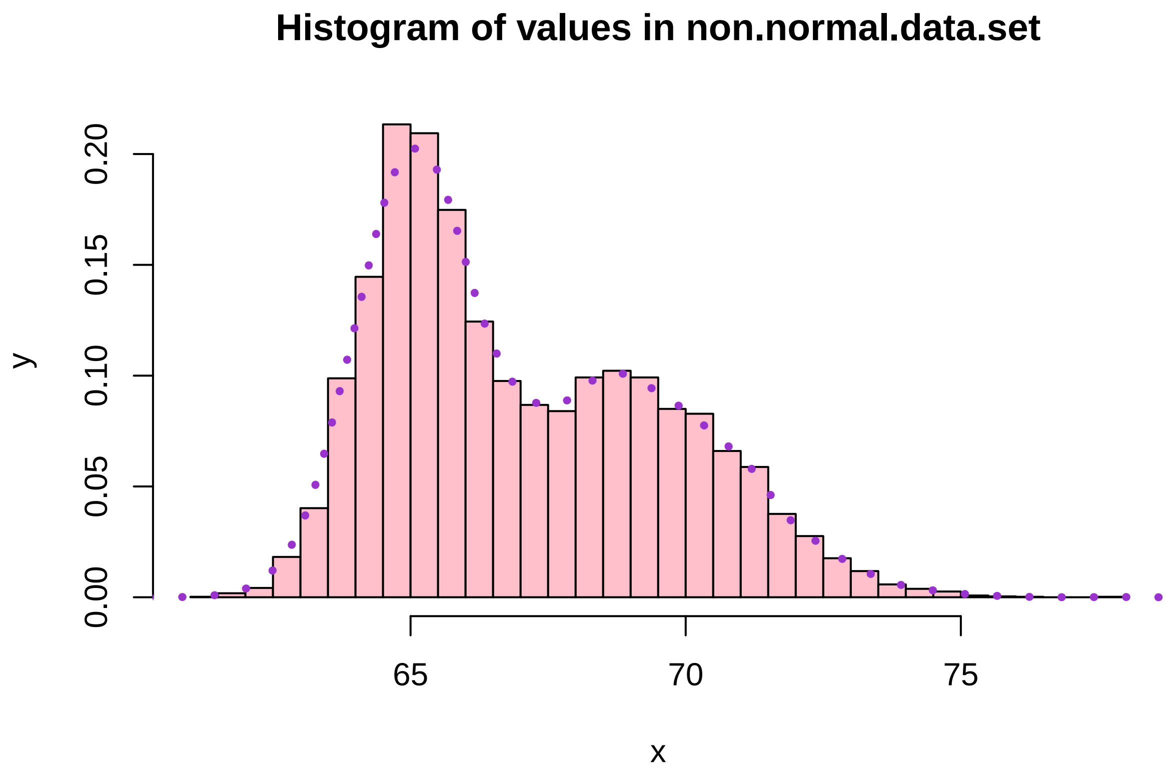
A *kernel density estimator* doesn't make any assumption of normality, or any other distribution – it just looks at the data and tries to create a curve that becomes large when there are many data points in a region, and becomes small when there are few data points in a region.

Sometimes I'll use the terms such as "empirical density curve" or "non-parametric density function", but this means the same thing as "kernel density curve".

We can fit a kernel density curve to the data in `example.6.2.data` using the `density()` function.

We can then plot this using the `lines()` function (no need to use `add = TRUE`).

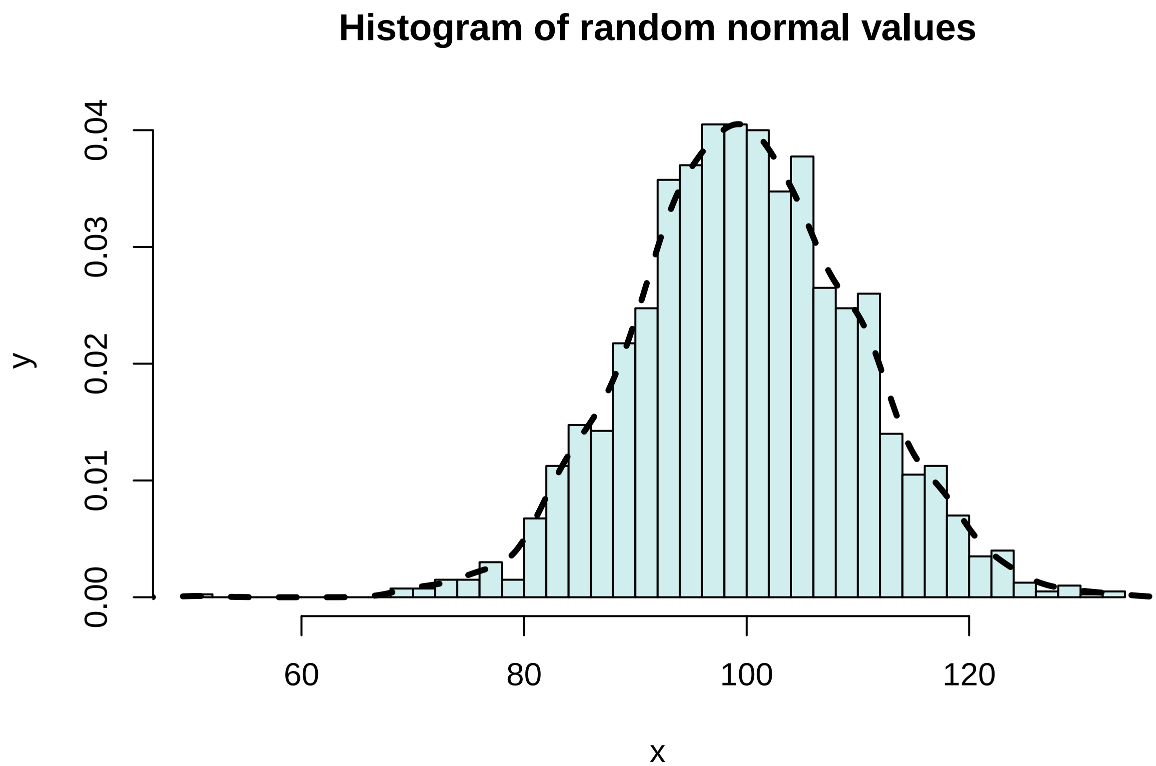
```
hist(
    example.6.2.data,
    main = "Histogram of values in non.normal.data.set",
    col = "pink",
    xlab = "x",
    ylab = "y",
    prob = TRUE,
    breaks = 50
)

lines(
    density( example.6.2.data ),
    lty = "dotted",
    lwd = 4,
    col = "darkorchid3"
)
```



That works well!

The kernel density estimator works really well with normal data:

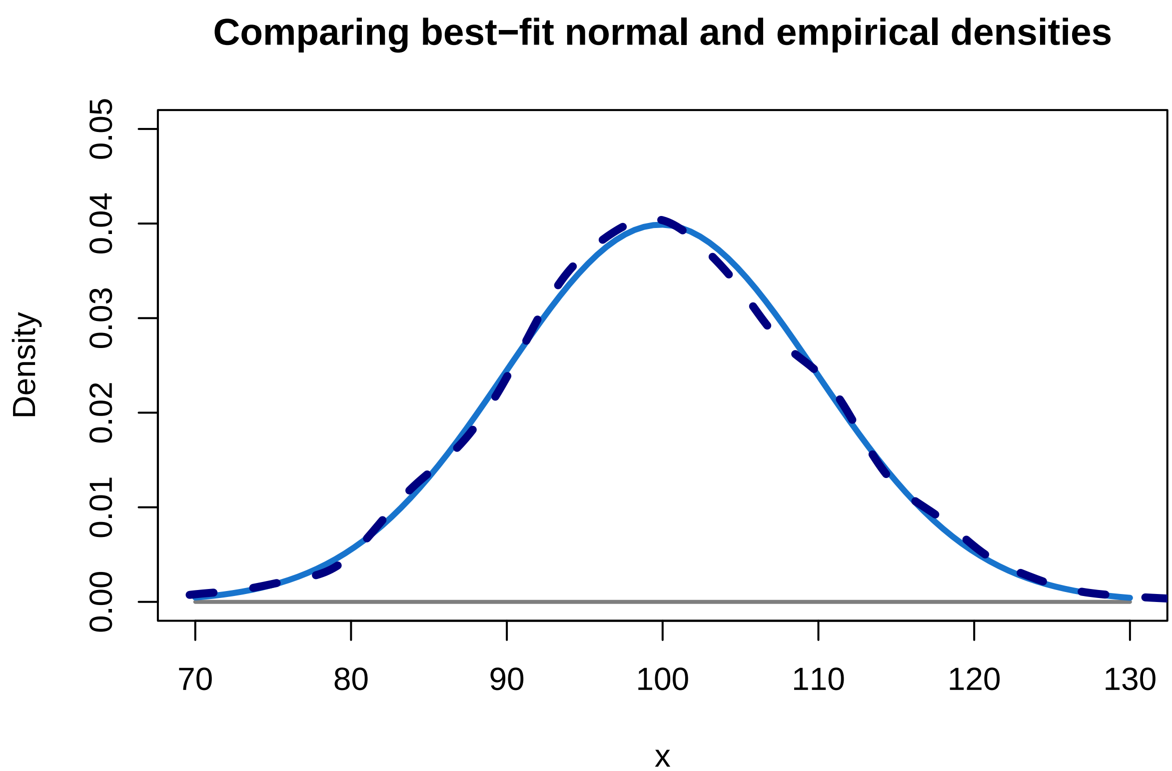
```
hist(
    example.6.1.data,
    main = "Histogram of random normal values",
    col = "lightcyan2",
    xlab = "x",
    ylab = "y",
    prob = TRUE,
    breaks = 50
)

lines(
    density( example.6.1.data ),
    lty = "dashed",
    lwd = 3,
    col = "black"
)
```



We can dispense with the histogram entirely, and just draw the best-fitting normal density along with the non-parametric empirical density:

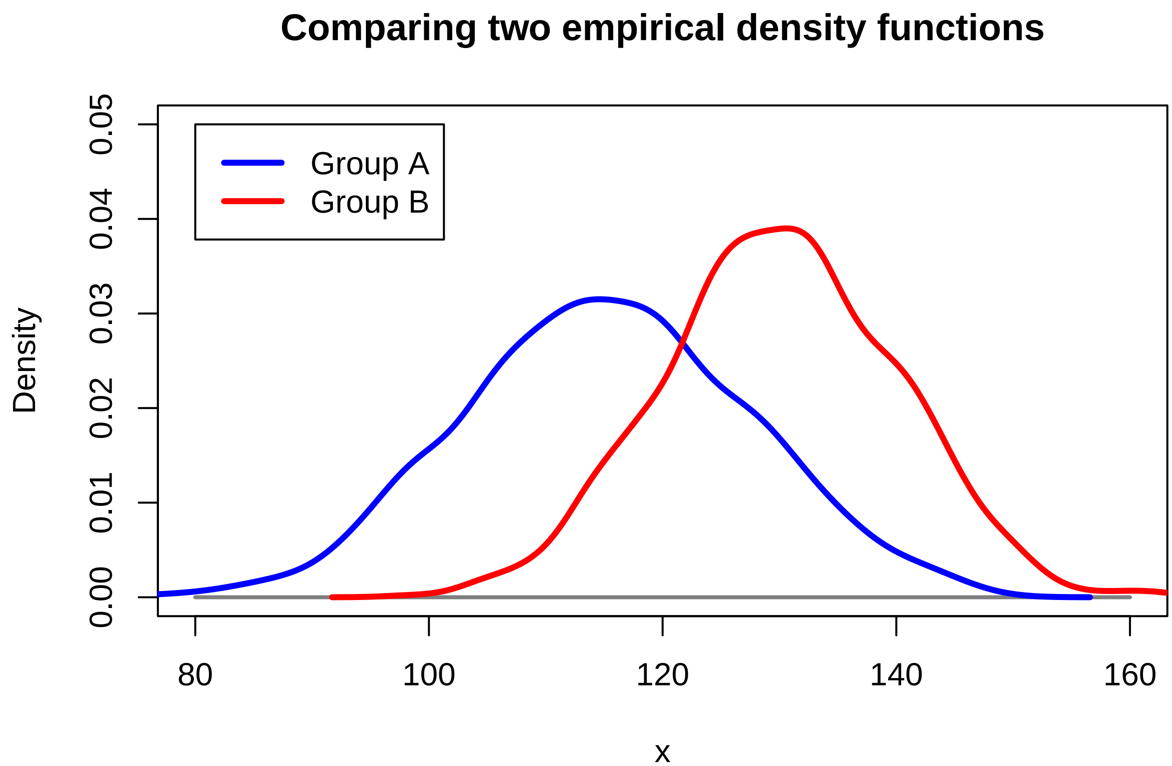
```
plot(
    x = NULL,
    xlim = c(70, 130),
    ylim = c(0, 0.05),
    main =
        "Comparing best-fit normal and empirical densities",
    xlab = "x",
    ylab = "Density"
)


segments(
    x0 = 70,
    y0 = 0,
    x1 = 130,
    y1 = 0,
    lty = "solid",
    lwd = 2,
    col = "gray50"
)

sample.mean <-
    mean( example.6.1.data )
```

```
sample.standard.deviation <-
    sd( example.6.1.data )

curve(
    dnorm(
        x,
        mean = sample.mean,
        sd = sample.standard.deviation
    ),
    lty = 1,
    lwd = 3,
    col = "dodgerblue3",
    add = TRUE
)

lines(
    density( example.6.1.data ),
    lty = 2,
    lwd = 4,
    col = "navy"
)
```



Empirical density curves can be very useful when you want to visualize the distribution of multiple variables in one display.

```
plot(
    x = NULL,
    xlim = c(80, 160),
    ylim = c(0, 0.05),
    main = "Comparing two empirical density functions",
    xlab = "x",
    ylab = "Density"
)


segments(
    x0 = 80,
    y0 = 0,
    x1 = 160,
    y1 = 0,
    lty = "solid",
    lwd = 2,
    col = "gray50"
)



lines(
    density( example.6.3.data ),
    lty = "solid",
    lwd = 3,
    col = "blue"
)

lines(
    density( example.6.4.data ),
    lty = "solid",
    lwd = 3,
    col = "red"
)


legend(
    x = 80,
    y = 0.05,
    legend =
        c( "Group A", "Group B" ),
    lty = "solid",
    lwd = 3,
    col =
        c( "blue", "red" )
)
```



So that's how to fit a non-parametric kernel density curve to data.

Now let's review what we've learned in this module.

Exercise 5.4: Superimposing a non-parametric density curve

Construct a histogram of the values in `exercise.6.4.data`, and then superimpose a normal density curve. Finally, superimpose a non-parameteric density curve over this display.

Solution

Module Review

In this module, we examined methods for superimposing curves on histograms.

- In section 1, we reviewed the theory of the normal distribution.
- In section 2, we saw how to superimpose a density curve over a histogram.
- In section 3, we learned how to determine the best-fitting density curve for a normally distributed set of data.
- In section 4, we explored non-parametric density curves.

Now that you've completed this module, you should be able to:

- Superimpose a normal density curve over a histogram.
- Determine the parameters for the best-fitting normal density curve.
- Superimpose a non-parametric density curve over a histogram.
- Draw non-parametric empirical density curves with normal density curves or other non-parametric density curves.

There were two new built-in R functions in this module:

- `dnorm()`
- `density()`

All right! That's it for Module 6: Histograms and Curves.

In fact, that's all the content for Week 5: Logical Values.

Now you can finish Problem Set 5.

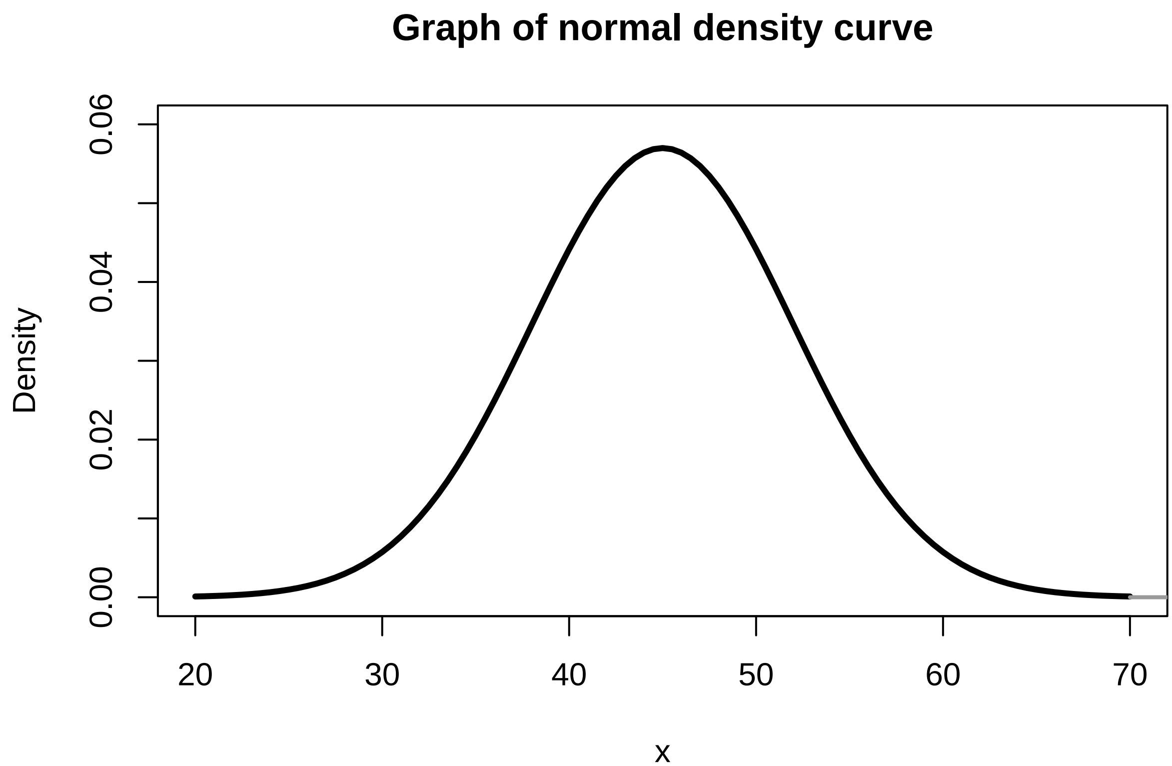
Solutions to the Exercises

Exercise 6.1: Drawing a normal density curve

Construct a graph of the density function for the normal distribution, with mean $\mu = 45$ and standard deviation $\sigma = 7$.

Solution

```
curve(
    dnorm(x, mean = 45, sd = 7),
    main = "Graph of normal density curve",
    xlim = c(20, 70),
    ylim = c(0, 0.06),
    xlab = "x",
    ylab = "Density",
    lty = "solid",
    lwd = 3,
    col = "black"
)


segments(
    x0 = 70,
    y0 = 0,
    x1 = 130,
    y1 = 0,
    lty = "solid",
    lwd = 2,
    col = "gray60"
)
```



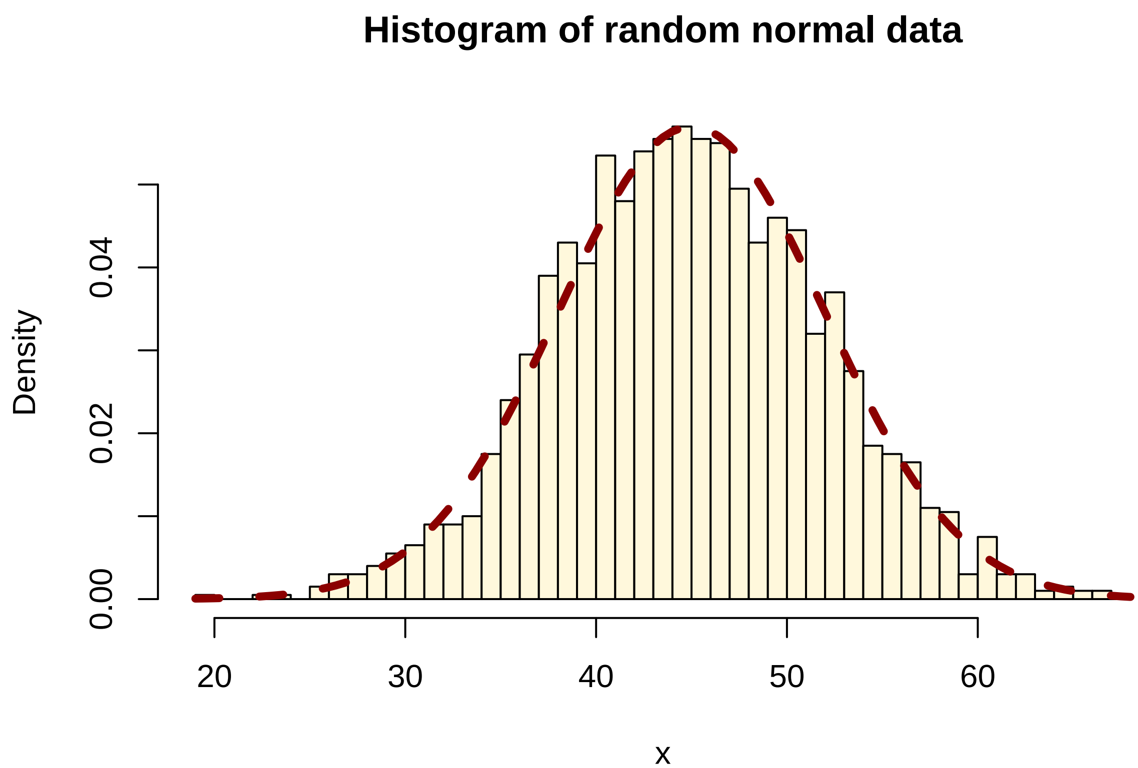
Exercise 6.2: Superimposing a normal density curve

Draw the histogram of the data in `exercise.6.2.data`. Then superimpose a normal density curve with a mean of 45 and a standard deviation of 7:

Solution

```
hist(
    exercise.6.2.data,
    prob = TRUE,
    main = "Histogram of random normal data",
    xlab = "x",
    ylab = "Density",
    col = "cornsilk",
    breaks = 50
)

curve(
    dnorm(
        x,
        mean = 45,
        sd = 7
    ),
    lty = "dashed",
    lwd = 4,
```

```
    col = "darkred",
    add = TRUE
)
```



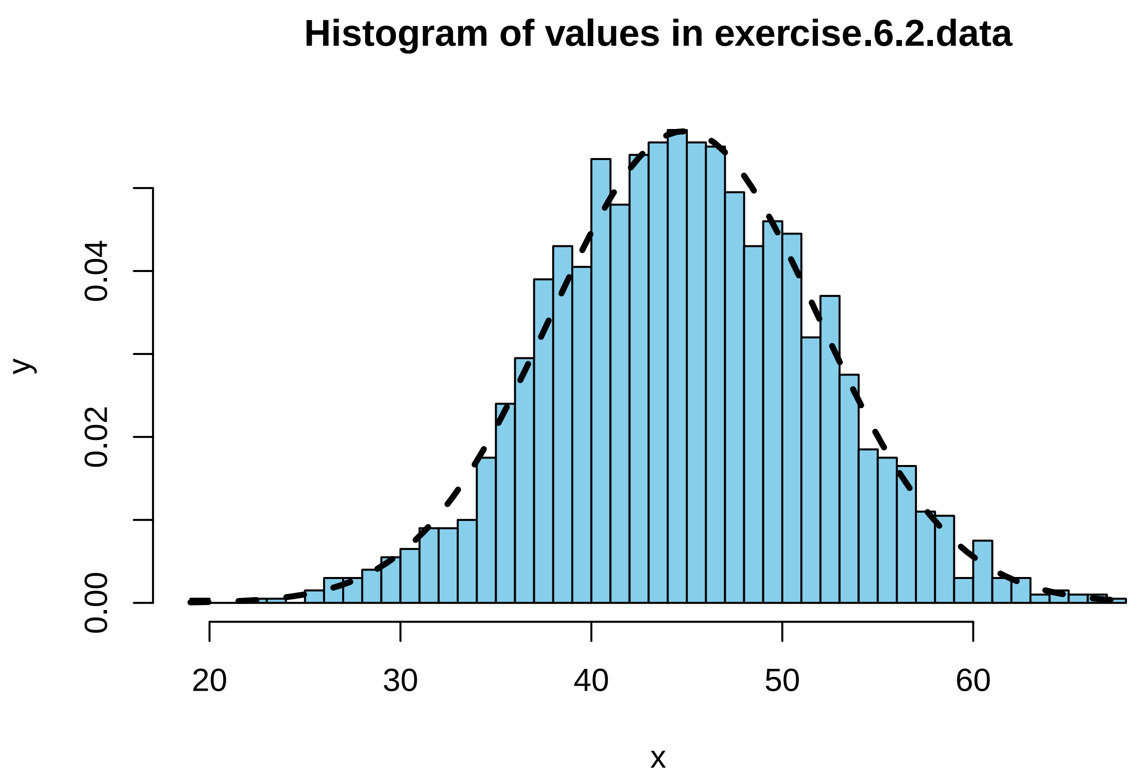
Exercise 6.3: Fitting a normal density curve

Calculate the sample mean and the sample standard deviation of the values in `exercise.6.2.data`. Redraw the histogram, and fit a normal density curve using the sample mean and sample standard deviation.

Solution

```
hist(
    exercise.6.2.data,
    main = "Histogram of values in exercise.6.2.data",
    col = "skyblue",
    xlab = "x",
    ylab = "y",
    prob = TRUE,
    breaks = 50
)


sample.mean <-
    mean( exercise.6.2.data )
```

```
sample.standard.deviation <-
    sd( exercise.6.2.data )

curve(
    dnorm(
        x,
        mean = sample.mean,
        sd = sample.standard.deviation
    ),
    lty = "dashed",
    lwd = 3,
    col = "black",
    add = TRUE
)
```



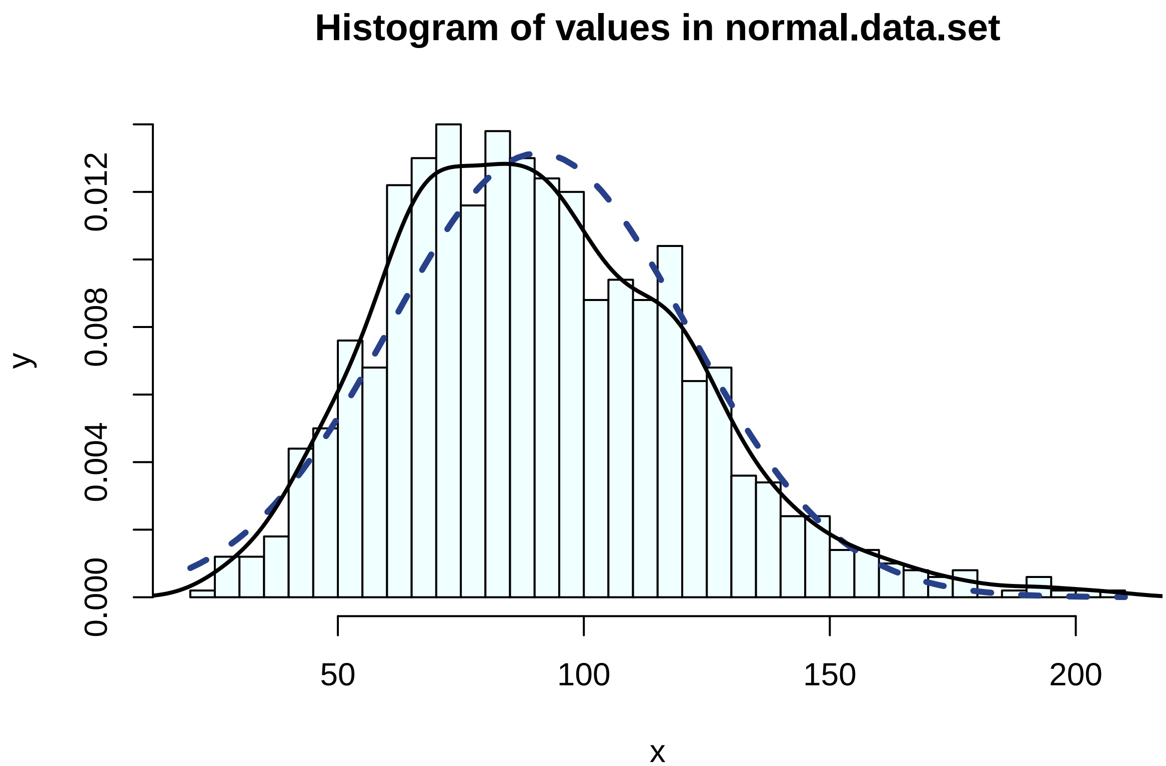
Exercise 6.4: Superimposing a non-parametric density curve

Construct a histogram of the values in `exercise.6.4.data`, and then superimpose a normal density curve. Finally, superimpose a non-parameteric density curve over this display.

Solution

```
hist(
    exercise.6.4.data,
    main = "Histogram of values in normal.data.set",
```

```
    col = "azure1",
    xlab = "x",
    ylab = "y",
    prob = TRUE,
    breaks = 50
)


sample.mean <-
    mean( exercise.6.4.data )

sample.standard.deviation <-
    sd( exercise.6.4.data )

curve(
    dnorm(
        x,
        mean = sample.mean,
        sd = sample.standard.deviation
    ),
    lty = "dashed",
    lwd = 3,
    col = "royalblue4",
    add = TRUE
)

lines(
    density( exercise.6.4.data ),
    lty = "solid",
    lwd = 2,
    col = "black"
)
```



Histogram of values in normal.data.set
y
0.000
0.004
0.008
0.012
50
100
150
200
x