

# Visualizing Probability Distributions

Nathaniel E. Helwig

University of Minnesota

## 1 Overview

In the previous chapter, we covered several commonly used probability distributions, such as the binomial, uniform, normal, chi-square, and  $F$  distribution. As a reminder, a random variable is characterized by its cumulative distribution function (CDF)  $F(x) = P(X \leq x)$ , which has a corresponding probability mass function (PMF) if  $X$  is discrete, or probability density function (PDF) if  $X$  is continuous. To get a better understanding of these distributions, we looked at visualizations of the PMFs/PDFs and CDFs for several different families of probability distributions. Such visualizations are useful for gaining theoretical insight into the forms of the probability distributions, but are not very helpful for understanding an observed sample of data. Note that for any observed sample of real data, we never know the true data generating probability distribution. So, in practice, we typically need some approach to help us visualize our sample's distribution. In this chapter, we will cover different graphical tools that are useful for visualizing distributions of real data.

## 2 Empirical Cumulative Distribution Function

The empirical cumulative distribution function (ECDF) is a simple and powerful approach for estimating the CDF. Given an independent and identically distributed (iid) sample of data  $x_1, \dots, x_n$  from some distribution  $F$ , the ECDF is defined as

$$\hat{F}_n(x) = \frac{1}{n} \sum_{i=1}^n I(x_i \leq x)$$

where  $I(\cdot)$  is an indicator function, i.e.,  $I(x_i \leq x) = 1$  if  $x_i \leq x$  and  $I(x_i \leq x) = 0$  otherwise. Note that the ECDF simply calculates the proportion of observations in the sample that are less than or equal to the input  $x$ . Since  $\hat{F}_n(x)$  is a proportion estimate, we have that

$$E\left(\hat{F}_n(x)\right) = F(x) \quad \text{and} \quad \text{Var}\left(\hat{F}_n(x)\right) = \frac{1}{n}F(x)(1 - F(x))$$

which implies that  $\hat{F}_n(x)$  is an unbiased estimate of the true proportion  $P(X \leq x)$ . Furthermore, as the sample size gets large, i.e., as  $n \rightarrow \infty$ , we have that  $\hat{F}_n(x) \xrightarrow{d} F(x)$ , which is known as the Glivenko-Cantelli theorem.

In R, you can calculate the ECDF using the `ecdf()` function, which was used to create the below plots (by using `plot(ecdf(x))` where  $x$  is the vector of data).

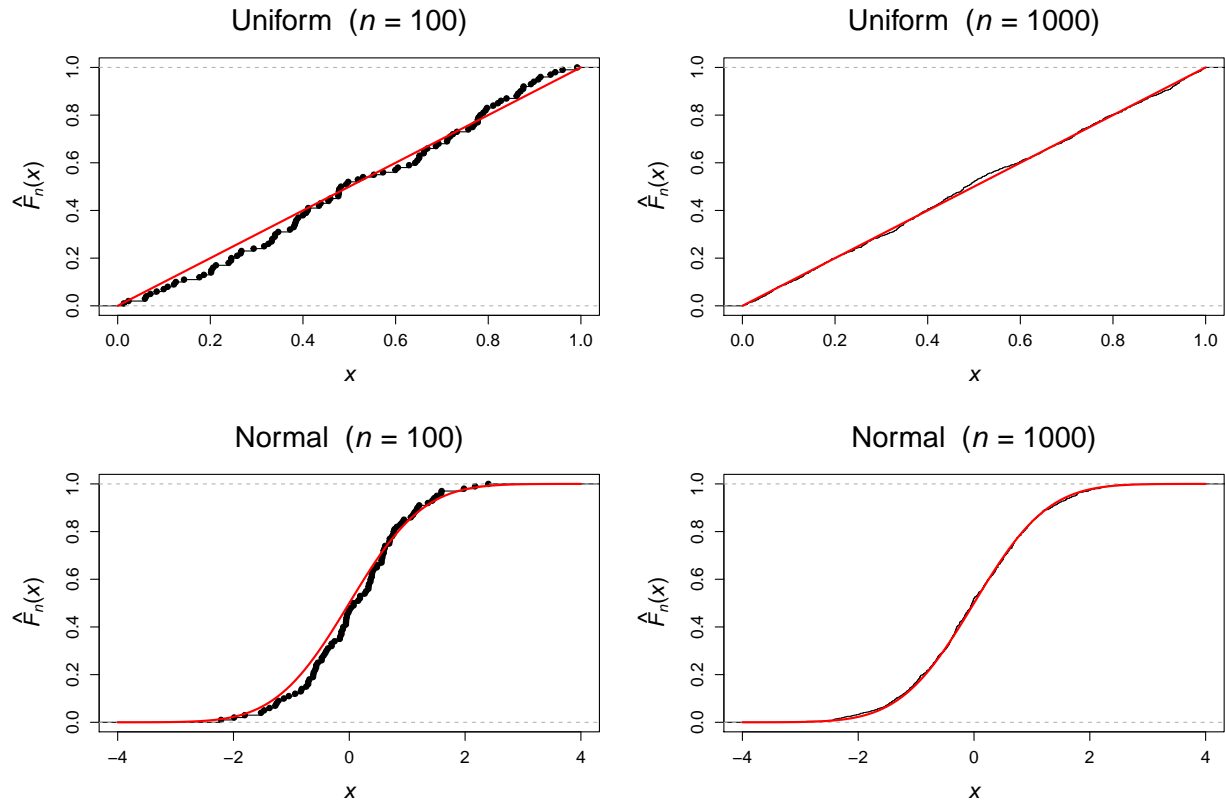


Figure 1: Empirical cumulative distribution function (ECDF) for  $n \in \{100, 1000\}$  samples drawn from a  $U[0, 1]$  distribution (top) and a  $N(0, 1)$  distribution (bottom). The black dots denote the ECDF and the red line denotes the true CDF for each distribution.

### 3 Quantile-Quantile (Q-Q) Plots

Q-Q plots are used to plot sample quantiles against one another or against population quantiles. Such plots can be useful for assessing whether (i) one sample of data follows a particular distribution, or (ii) two samples of data have a similar distribution. As a reminder, the population quantile function  $Q(p)$  is the inverse of the CDF function, such that it takes in a probability  $p \in [0, 1]$  and returns a value  $x \in S$  such that  $F(x) \geq p$ . Before looking at some Q-Q plots, we need to discuss how sample quantiles are estimated from a sample of data.

Given an iid sample of data  $x_1, \dots, x_n$  from some distribution  $F$ , the order statistics are

$$x_{(1)} \leq x_{(2)} \leq \dots \leq x_{(n-1)} \leq x_{(n)}$$

which is simply the sample of data sorted from smallest to largest. The first order statistic is  $x_{(1)} = \min(x_i)$  and the  $n$ -th order statistic is  $x_{(n)} = \max(x_i)$ . In general, the  $i$ -th order statistic  $x_{(i)}$  is the  $x$  value that is in the  $i$ -th position after sorting the sample of data.

For convenience of notation, let's assume that the observations are sorted from smallest to largest, so that  $x_i = x_{(i)}$  for  $i = 1, \dots, n$ . The sample quantiles are defined as

$$\hat{Q}_n(p) = x_{[h]} + (h - [h]) (x_{[h]+1} - x_{[h]})$$

where the value of  $h$  depends on what interpolation scheme is used to estimate the quantiles. Default use of R's `quantile()` function (i.e., `type = 7`) defines this value as  $h = (n-1)p + 1$ . However, the `qqnorm` function in R uses the `ppoints()` function to generate probability points, and this function uses the `type = 5` definition of  $h = np + 1/2$ .

Two ways in which Q-Q plots are typically used:

- If you have a single sample of data, it is typical to plot the theoretical quantiles  $Q(p)$  on the x-axis and the sample quantiles  $\hat{Q}_n(p)$  on the y-axis. If the theoretical and sample quantiles are similar to one another, the points in the plot will approximately fall on the 45-degree line.
- If you have two samples of data with sizes  $m$  and  $n$ , it is typical to plot the sample quantiles of the first sample  $\hat{Q}_m^1(p)$  on the x-axis and the sample quantiles  $\hat{Q}_n^2(p)$  on the y-axis. If the two sets of sample quantiles are similar to one another, the points in the plot will approximately fall on the 45-degree line.

Note that in both uses of Q-Q plots, having the points fall on the 45-degree line indicates that the two sets of plotted quantiles reasonably agree with one another. As we will see in the following examples, the nice thing about Q-Q plots is that any deviations from the 45-degree line can provide graphical insights into how the quantiles differ from one another. As a result, Q-Q plots can be used to understand (i) how well a particular sample of data approximates an assumed distribution, such as the normal distribution, and (ii) how similar two samples of data are to one another. Note that comparing sample quantiles to theoretical normal quantiles is so commonly used in statistical applications that R has a special function designed specifically for this purpose (see `?qqnorm`).

In the below example, note the following: (i) for left-skewed data, the Q-Q points fall below the 45-degree line, (ii) for right-skewed data, the Q-Q points fall above the 45-degree line, (iii) for leptokurtic data, the points fall below the 45-degree line for negative values and above the 45-degree line for positive values, and (iv) for platykurtic data, the points fall above the 45-degree line for negative values and below the 45-degree line for positive values.

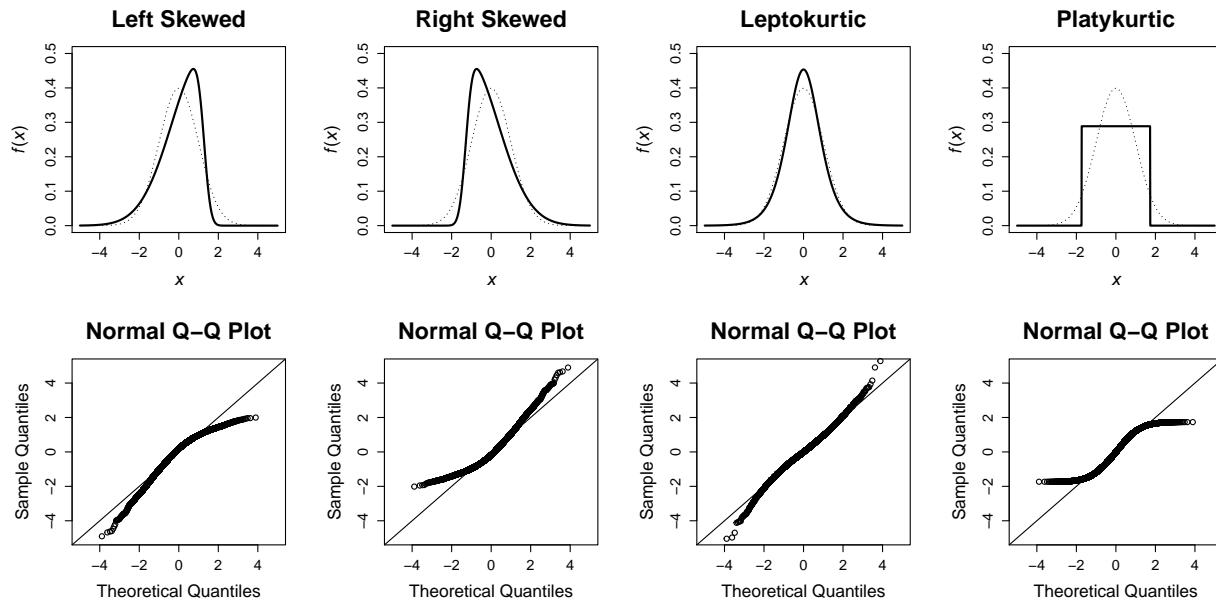


Figure 2: Top: probability density functions for distributions with different values of skewness and kurtosis (solid line), along with the standard normal density function (dotted line). Bottom: corresponding normal Q-Q plots with theoretical quantiles from a standard normal. The sample quantiles we calculated using 10,000 independent samples from each distribution.

## 4 Boxplots

A box plot is a simple graphic that can be used to visualize the distribution for a sample of independent and identically distributed (iid) data from some distribution  $F$ . A standard box plot consists of a few different components: (i) a rectangle to denote the interquartile range, i.e.,  $IQR = Q_3 - Q_1$ , (ii) a line for the median, i.e., the second quartile  $Q_2$ , and (iii) whiskers on each end of the box plot to denote the data range. See Figure 3 for a diagram.

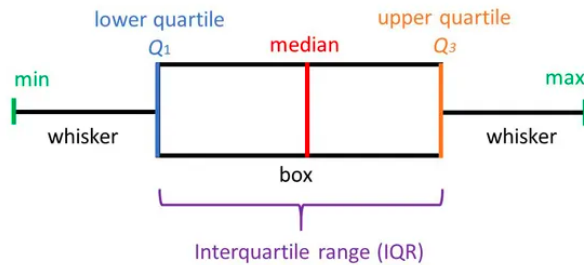


Figure 3: Properties of a box plot. From <https://www.simplypsychology.org/boxplots.html>

Note that R's `boxplot()` function draws the whiskers to extend to  $\pm 1.5IQR$ . The data that are beyond the range of the whiskers are denoted with points, which indicates that these data points are potential outliers. As the below example illustrates, box plots can be quite helpful for visualizing the location, spread, and shape of a sample of data.

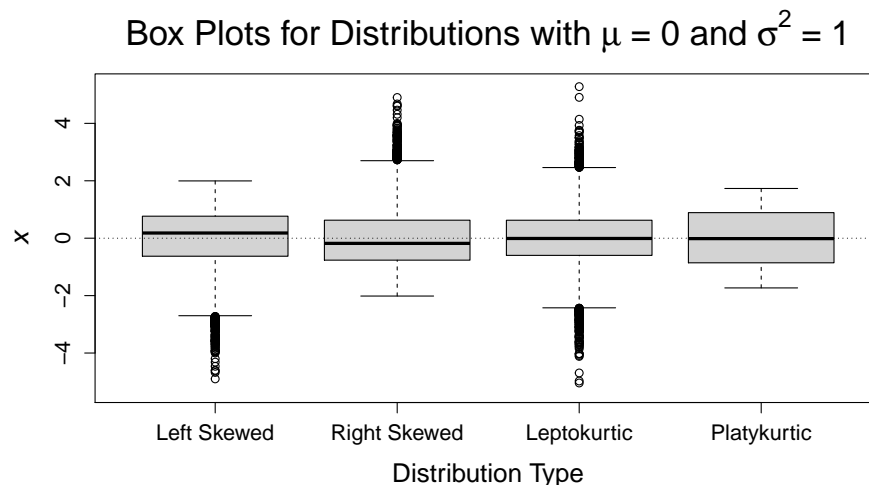


Figure 4: Box plots created with R's `boxplot()` function. The box plots we calculated using 10,000 independent samples from each distribution.

## 5 Histograms

A histogram is a simple way to estimate the probability density function (PDF) given a sample of independent and identically distributed (iid) data from some distribution  $F$ . Note that if the PDF  $f(x)$  is smooth, then we have that

$$\begin{aligned} P(x - h/2 < X < x + h/2) &= F(x + h/2) - F(x - h/2) \\ &= \int_{x-h/2}^{x+h/2} f(z) dz \approx hf(x) \end{aligned}$$

where  $h > 0$  is some small constant referred to as the “bin width”. If the CDF  $F(x)$  were known, then we could estimate the PDF using

$$\hat{f}(x) = \frac{F(x + h/2) - F(x - h/2)}{h}$$

but this isn’t practical because we never know the true CDF  $F(x)$  for real data.

In practice, we can plug the ECDF estimate  $\hat{F}_n(x)$  into the above equation, which gives

$$\begin{aligned} \hat{f}_n(x) &= \frac{\hat{F}_n(x + h/2) - \hat{F}_n(x - h/2)}{h} \\ &= \frac{\sum_{i=1}^n I(x_i \leq x + h/2) - \sum_{i=1}^n I(x_i \leq x - h/2)}{nh} \\ &= \frac{\sum_{i=1}^n I(x_i \in (x - h/2, x + h/2])}{nh} \end{aligned}$$

More generally, we could estimate the PDF  $f(x)$  in a window around  $x$  using

$$\hat{f}_n(x) = \frac{\sum_{i=1}^n I(x_i \in w_j)}{nh} = \frac{n_j}{nh}$$

for all  $x \in w_j = (b_j - h/2, b_j + h/2]$  where the  $b_1 < b_2 \dots < b_{m+1}$  are chosen constants known as “break points” that span the range of the observed data. Thus, to form a histogram you just need to (i) break the real number line into  $m$  mutually exclusive bins at break points spanning your data, and (ii) count the number of observations  $n_j$  that fall within each bin. Of course, different choices of the number of bins  $m$  will affect the estimate. So how should we choose the break points or bin widths to form our histogram estimate?

Various different methods exist to choose reasonable values of  $m$  and  $h$  for a histogram:

- Sturges (default in R's `hist()` function):  $m = \lceil \log_2(n) + 1 \rceil$  and  $h = (x_{(n)} - x_{(1)})/m$
- Freedman and Diaconis:  $h = 2IQR/n^{1/3}$  and  $m = \lceil (x_{(n)} - x_{(1)})/h \rceil$
- Scott:  $h = 3.5s/n^{1/3}$  and  $m = \lceil (x_{(n)} - x_{(1)})/h \rceil$  where  $s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$

However, these algorithms may not produce a value of  $m$  that is useful for your given sample of data. In practice, it may be helpful to set the break points using pre-specified values. This is what I did to create the histograms in the below figure. Note that this has the benefit of ensuring that the bin width is the same for each of the subplots.

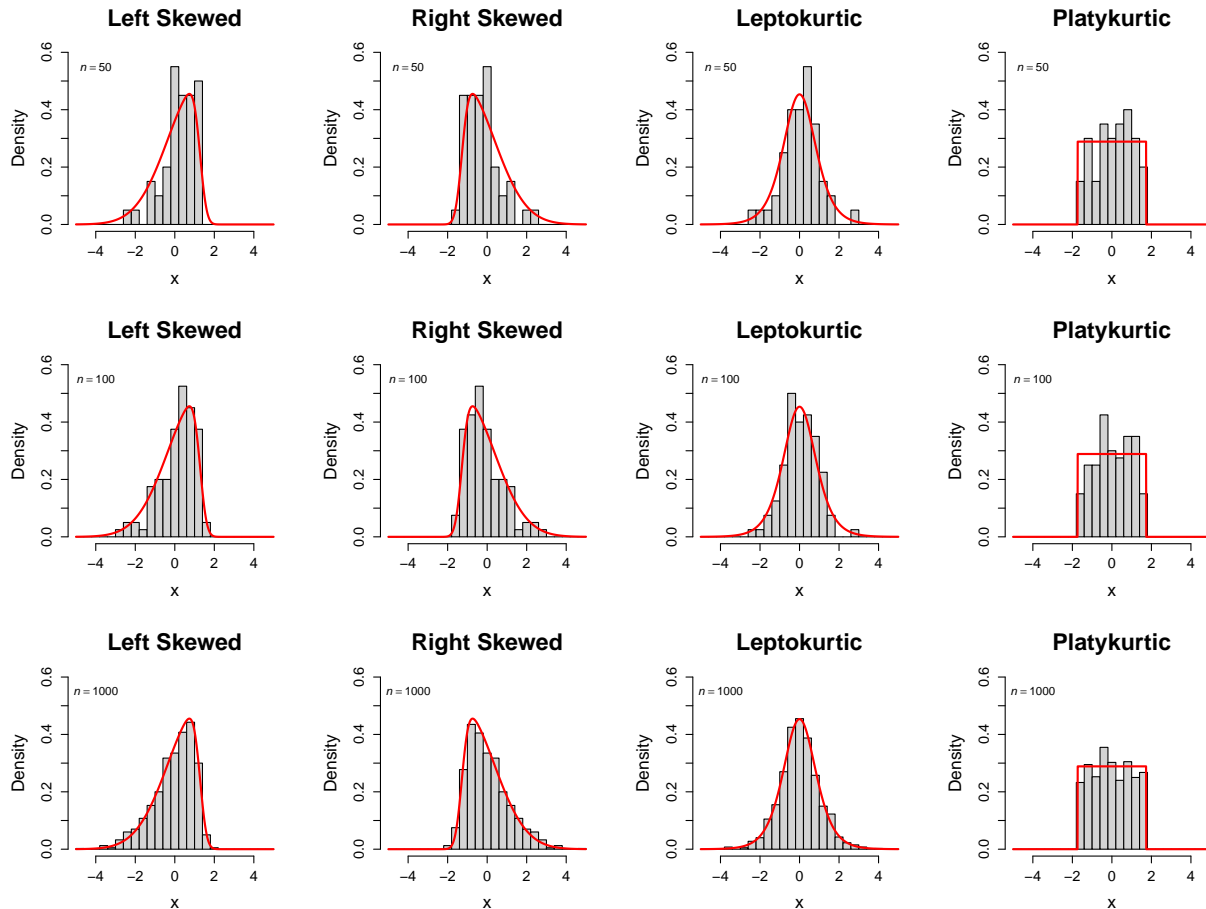


Figure 5: Histograms created with R's `hist()` function. Red line denotes the true density.

## 6 Kernel Density Estimates

Histograms are simple to understand and create, but they provide rather crude estimates of PDFs. As the previous example demonstrates, histograms tend to perform better as the number of observations  $n$  increases. However, no matter how large of a sample you obtain, histograms will have an obvious limitation: they produce jagged (i.e., unsmooth) estimates. Kernel density estimates (KDEs) are a powerful alternative to histograms that overcome this limitation by providing a smooth estimate of the PDF.

Given an iid sample of data  $x_1, \dots, x_n$  from some distribution  $F$ , a KDE has the form

$$\hat{f}_h(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right)$$

where  $K(\cdot)$  is a user-specified kernel function and  $h > 0$  is the chosen bandwidth. The kernel function  $K(\cdot)$  can be any function that satisfies

- $K(x) \geq 0$  for all  $x$  (non-negative)
- $K(x) = K(-x)$  for all  $x$  (symmetric)
- $\int_{-\infty}^{\infty} K(x) = 1$  (unit measure)

which implies that any PDF that is symmetric around zero can be used as a kernel function. Assuming that the chosen kernel function satisfies these three properties, the KDE  $\hat{f}_h(x)$  will be a valid PDF for any chosen bandwidth  $h > 0$ . In most cases, the standard normal PDF is used as the kernel function, but other choices of the kernel as possible (see Figure 6).

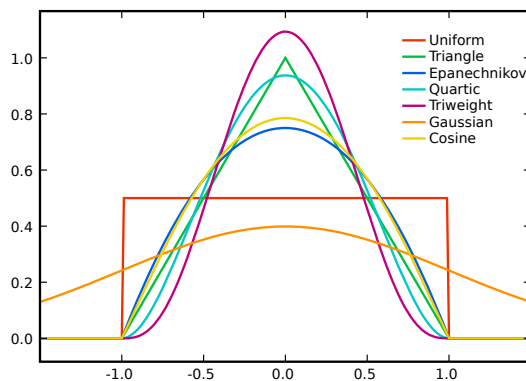


Figure 6: Different kernel functions.  
From <https://upload.wikimedia.org/wikipedia/commons/4/47/Kernels.svg>



The bandwidth parameter  $h$  is analogous to the bin width parameter  $h$  in a histogram, such that different values of  $h$  will produce different estimates. Note that the bandwidth parameter controls the compactness of the kernel function, such that larger values of  $h$  use wider kernels. If the standard normal PDF is used as the kernel function, then the bandwidth  $h$  corresponds to the standard deviation of the scaled kernel function  $\frac{1}{h}K\left(\frac{x-x_i}{h}\right)$ . If we choose a larger  $h$ , the scaled kernel function will borrow information from more nearby data points when forming the estimate  $\hat{f}_h(x)$ , which will produce a smoother estimate of the PDF. As the bandwidth parameter gets smaller, the scaled kernel functions will become more compact around the observed data points, which will produce a rougher estimate of the PDF. In practice, it is typical to use Silverman's rule of thumb to define  $h$ , which has the form  $h = 0.9n^{-1/5} \min(s, IQR/1.34)$  where  $s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$ . But using some sort of cross-validation method is likely a better idea, particularly if you are using the KDE for more than a simple visualization of your data.

As a simple example, suppose that we  $n = 6$  data points  $(-2.1, -1.3, -0.4, 1.9, 5.1, 6.2)$ , and we want to form a histogram and a KDE. Furthermore, suppose that we use a standard normal PDF as the kernel function and set the bandwidth at  $h = 1.5$ . Then the KDE is formed by centering a  $N(0, 1.5^2)$  density at each data point  $x_i$ , and then calculating the average of the  $N(x_i, 1.5^2)$  densities to form the KDE  $\hat{f}_h(x)$ , see Figure 7.

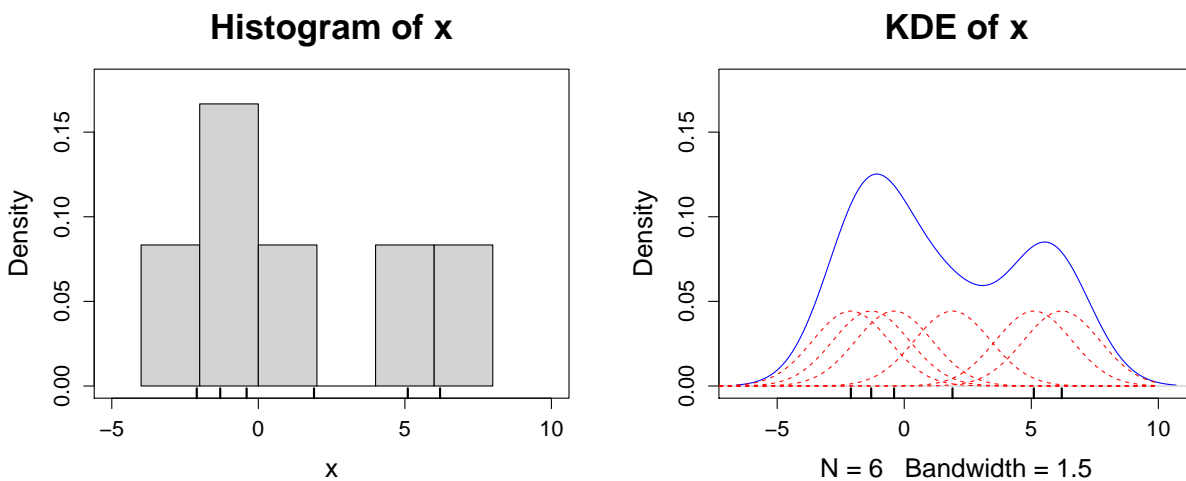


Figure 7: Histogram and KDE estimate for six data points. The red dashed lines are showing  $\frac{1}{nh}K\left(\frac{x-x_i}{h}\right)$ , which are summed together to obtain the blue line, which is the KDE.

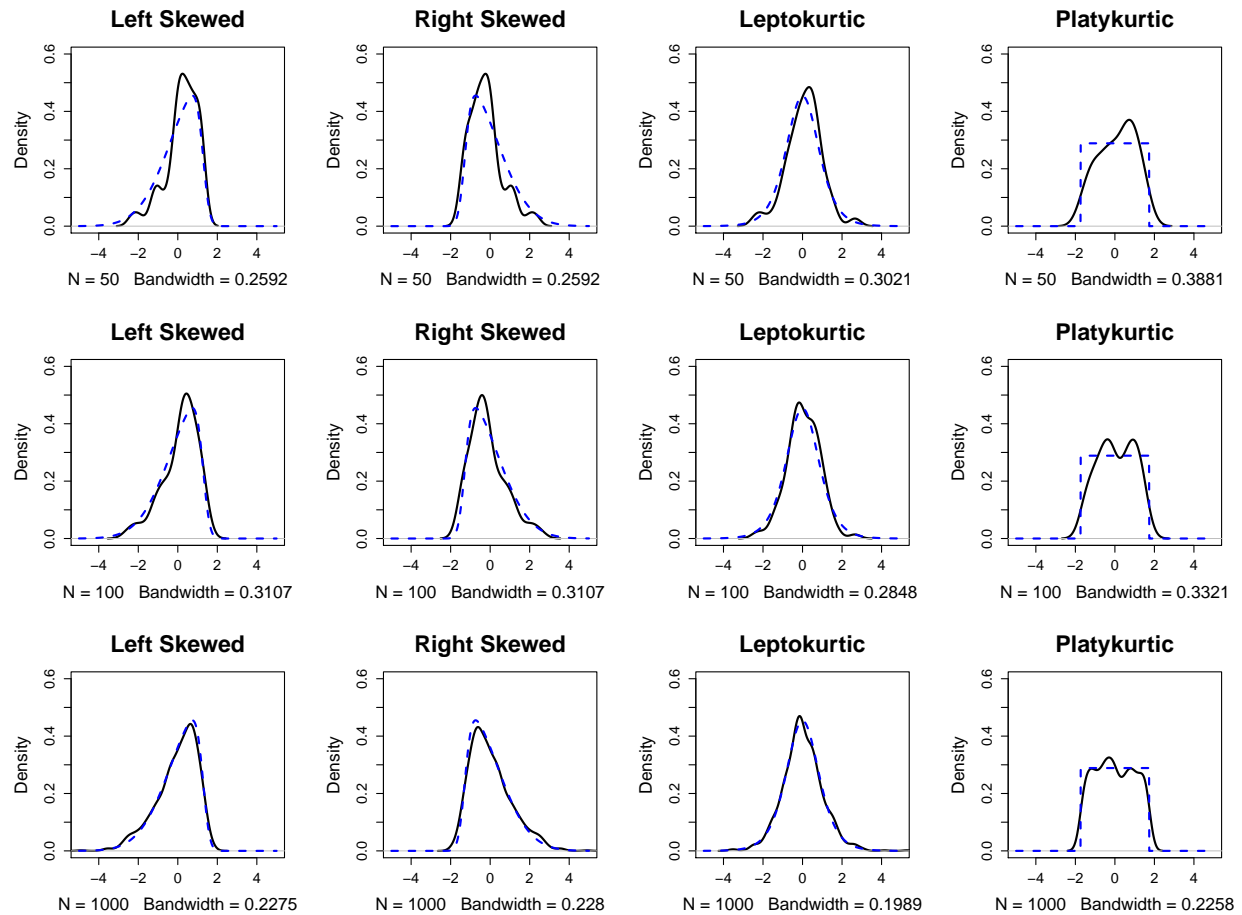


Figure 8: Kernel density estimates (KDEs) created with R's `density()` function. The blue dashed line denotes the true density.