5.1 Program Your Own KDE Function

5.1.1 Write an R function to perform KDE.

It should be able to perform KDE for a vector of any length (>=1). It should be able to take on different specified bandwidths, `h` (which are numeric). The kernel should be triangular. No calls to other KDE functions can be made – you need to write an additional function to plot the kernel. Remember that the triangular kernel is

$$K(x) = 1 - |x|, \text{ on } -1 \leq x < 1.$$

```
# helper function to find the triangular kernel (vectorized)
.kernel <- function(x) {
  ifelse(-1 <= x & x < 1, 1 - abs(x), 0)
}
```

```
# helper function to find the KDE of a single value
.kde <- function(x, data, n, bandwidth) {
  return((1 / (n * bandwidth)) * sum(.kernel((x - data) / bandwidth)))
}
```

```
# function to perform triangular kernel density estimation
tkde <- function(data, bandwidth) {
  data <- data[!is.na(data)]
  x <- seq(min(data) - 2 * bandwidth, max(data) + 2 * bandwidth, length(data)/1000)
  y <- sapply(x, .kde, data = data, n = length(data), bandwidth = bandwidth)
  return(list(x = x, y = y))
}
```

```
# function to plot a triangular KDE
plotkde <- function(kde) {
  ggplot(data.frame(x = kde$x, y = kde$y), aes(x = x, y = y)) +
    geom_line() +
    labs(x = "X", y = "F(X)")
}
```

5.1.2 Use your function

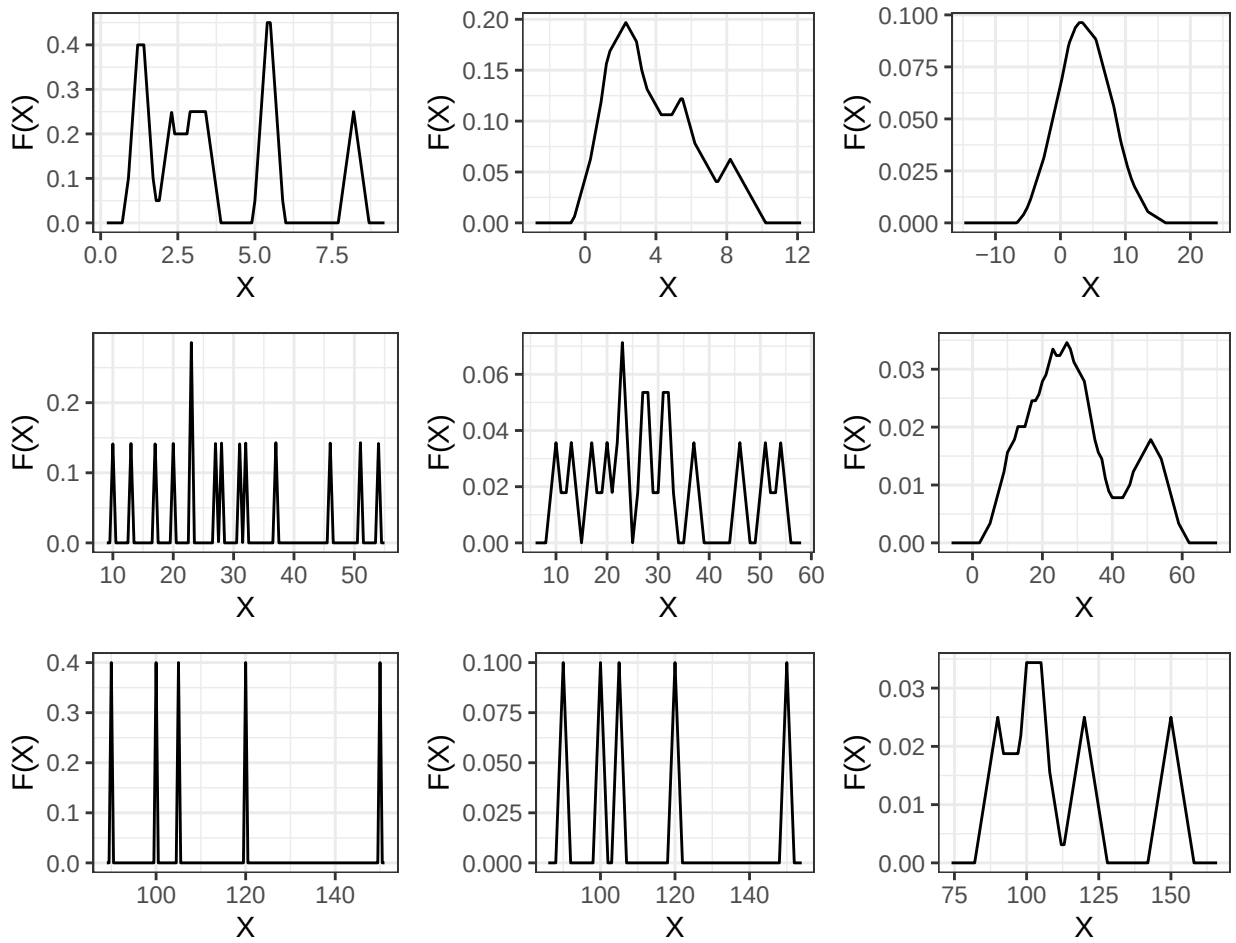
Estimate the densities of each of these vectors. Use three different `h` values: `0.5, 2, 8`. Use `cowplot::plotgrid` to plot a 3x3 matrix of plots, expanding the output to page width. Describe which of the three KDEs is the best for each vector.

```r
#The three vectors
vector1 = c(1.4, 5.5, 8.2, 2.3, 5.4, 1.2, 3.4, 2.9)
vector2 = c(10, 20, 31, 23, 54, 23, 13, 17, 27, 37, 28, 51, 32, 46)
vector3 = c(100, 105, 120, 150, 90)

densplots <- expand_grid(data = list(vector1, vector2, vector3), bandwidth = c(0.5, 2, 8)) %>%
  {mapply(tkde, data = .$data, bandwidth = .$bandwidth)} %>%
  apply(2, plotkde)

densplots <- lapply(1:9, function(i) {
  densplots[[i]]
})

cowplot::plot_grid(plotlist = densplots, nrow = 3)
```



For the first vector, the kernel density estimation with a bandwidth of 2 would be the best fit. The bandwidth of 2 produces a density plot that is neither too jagged nor too smooth. The bandwidth of 2 provides us enough information to assess the plot without over correction.

For the second vector, the kernel density estimation with a bandwidth of 8 would be the best fit. The bandwidths of 0.5 and 2 both produces density plots that are far too jagged and hard to assess

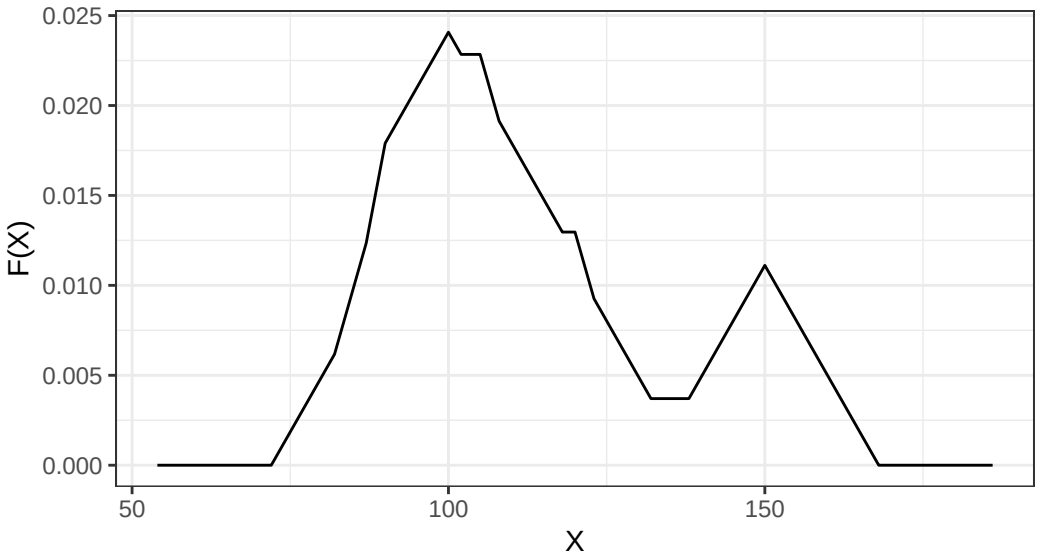
and interpret. The bandwidth of 8 still has some jaggedness, but not to too great of an extent. We are still able to accurately interpret the density plot produced by this bandwidth.

For the third vector, the kernel density estimation with a bandwidth of 8 would also be the best fit, but none of the three bandwidth estimations seem to be that great of a fit for this vector. All three of the density plots are far too jagged, with the density plot with a bandwidth of 8 being the least jagged of the three. Although we are able to interpret the density plot with a bandwidth of 8, a higher bandwidth may be more appropriate for this vector.

5.1.3 Use your function, pt II

If there is a vector which is not well represented with any of your KDEs, manually pick an h that works best for that vector. Explain why the given h values were not sufficient for the vector(s) they failed to represent well.

```
plotkde(tkde(vector3, 18))
```



Compared to vectors 1 and 2, vector 3 has much greater values with much greater ranges between the values. Since the h values given were smaller, they were not as appropriate for the larger values of vector 3. Smaller h values in turn lead to smaller neighborhoods, which lead us to have too many jagged peaks in our density plot. By choosing a larger h value, such as 18, we are able to create a larger neighborhood, smoothing out the peaks and providing us a more accurate and more easily-interpretable kernel density estimation plot.