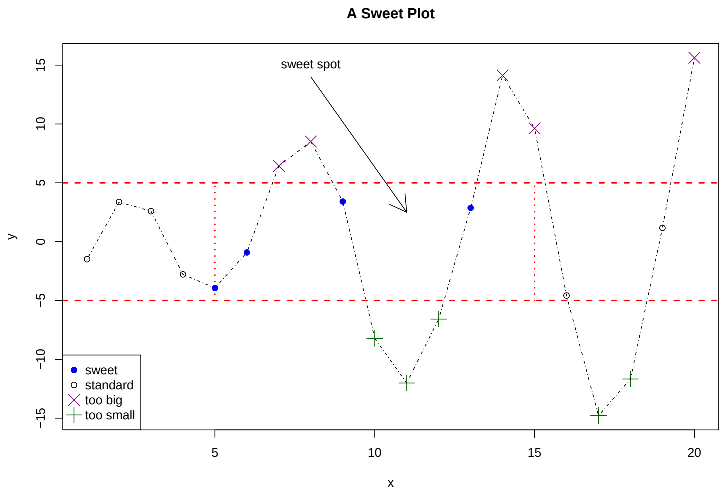
PSTAT 10 Worksheet 4

In this worksheet, we will go through the steps to make the following plot in base R:



The idea is to plot 20 data points and graphically mark them depending on where they fall within provided bounds. E.g. points within the "sweet spot" are marked as such with blue solid points.

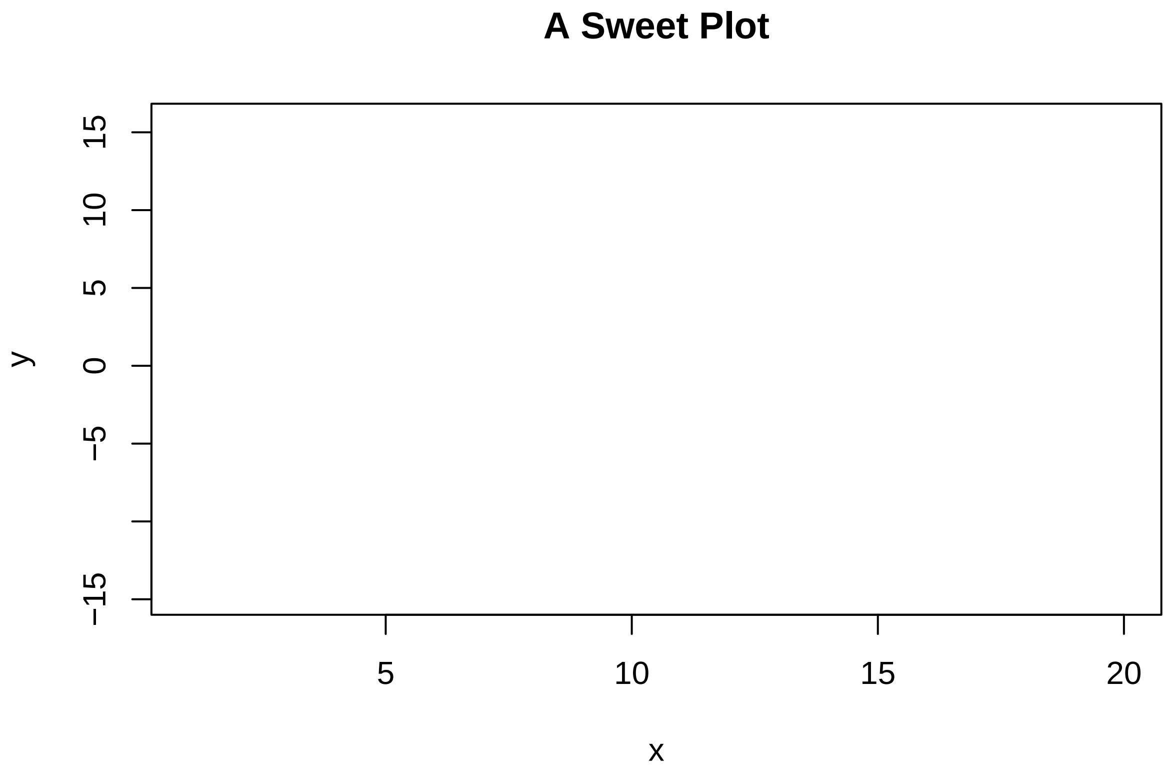
In what follows, you may need to adjust the figure output from R Markdown. I used the following settings within a code chunk. `fig.dim` sets the width and height of a figure in inches `out.width` and `out.height` scale the figure.

```
{r, fig.align = "center", fig.dim=c(10, 7), out.width="60%", out.height="60%"}
```

Step 0: Generate the data

```
x <- 1:20
y <- c(-1.49, 3.37, 2.59, -2.78, -3.94, -0.92, 6.43, 8.51, 3.41, -8.23,
  -12.01, -6.58, 2.87, 14.12, 9.63, -4.58, -14.78, -11.67, 1.17, 15.62)
```

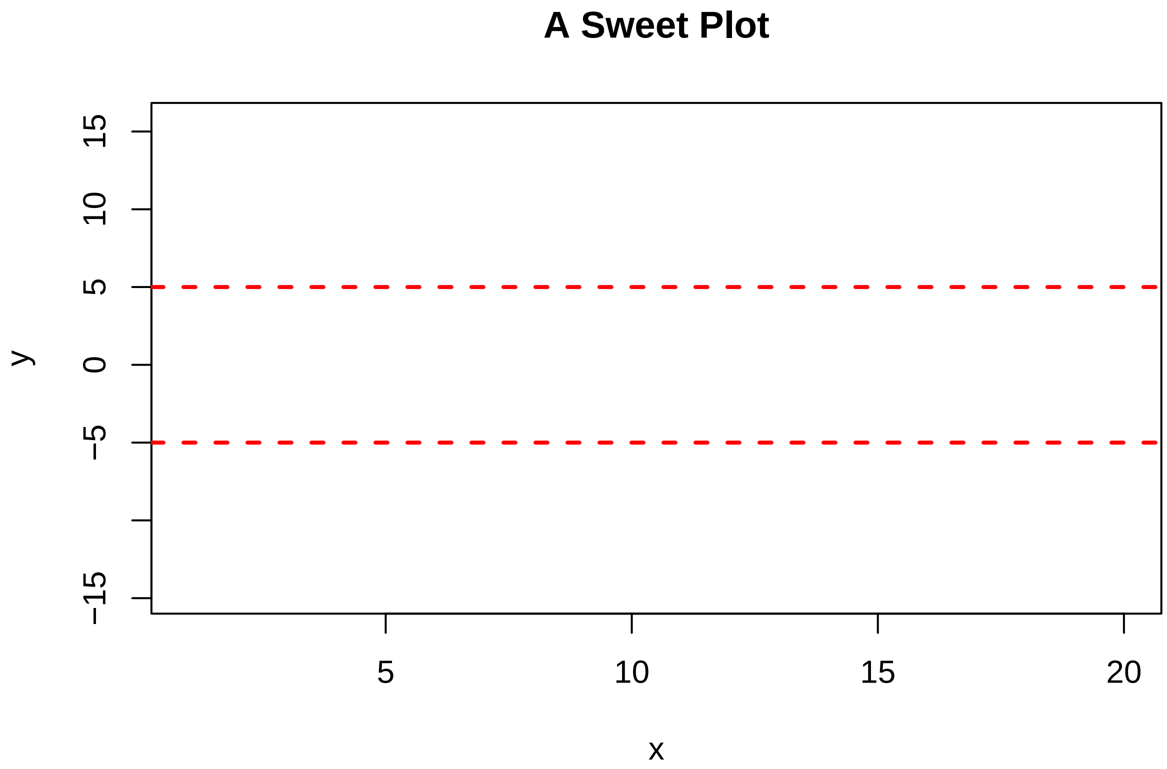
Step 1: Create an empty plot with a title.



Step 2: y limits

The abline function adds straight lines to an existing plot: `abline(b, m)` plots a line with y-intercept `b` and slope `m`. Alternatively, the `abline` has named arguments `h` and `v` that make it easy to plot horizontal and vertical lines: check out the help with `?abline`.

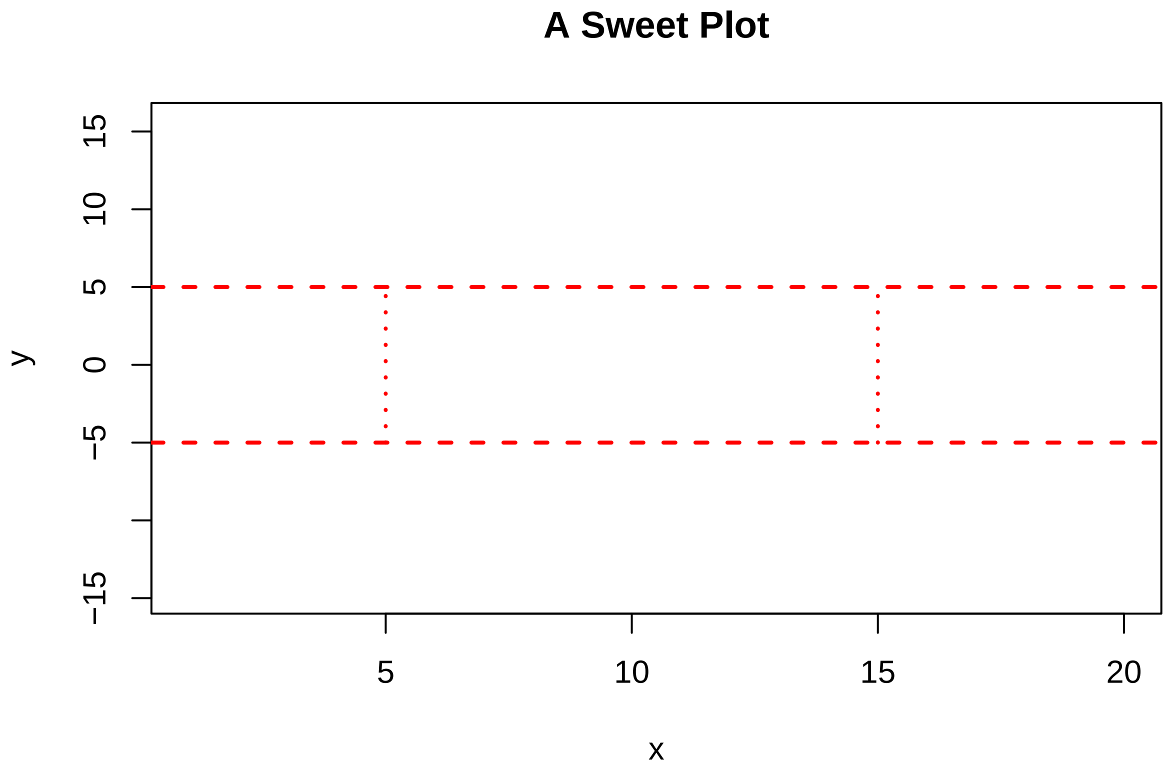
Update the plot with two horizontal lines. Play around with `col`, `lty`, and `lwd` to get the line right.



Step 3: x limits

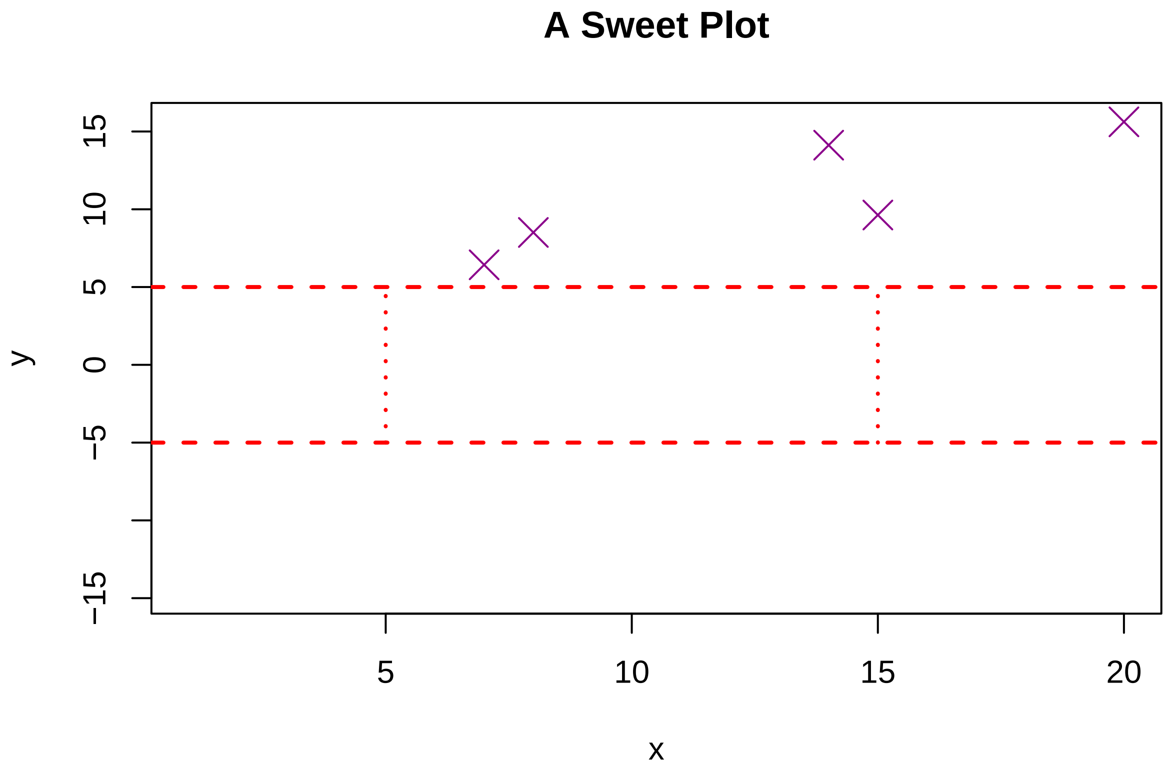
The segments function adds line segments to an existing plot. `segments(x0, y0, x1, y1)` draws a line segment connecting the point `(x0, y0)` to `(x1, y1)`. Remember to check out the help: `?segments`.

Add two vertical line segments connecting (5,−5) to (5,5) and connecting (15,−5) to (15,5). Remember to adjust the line type as needed.



Step 4: Add “too big” points

Using our vectors x and y, plot the pairs (x,y) such that y 5. *Hint:* Use filtering to identify the points; filter both `x` and `y` vectors with some logical vector. I’ve use the R color “darkmagenta” and `cex=2` to enlarge the symbol. Find the correct × symbol for pch.


A Sweet Plot
y
15
10
5
0
−5
−15
5
10
15
20
x

x, y = coordinate vectors of points to plot.

type = character indicating the type of plotting; actually any of the types as in plot.default.

cex = size of pointsar

pch = 0: Square

pch = 1: Circle

pch = 2: Triangle (pointing up)

pch = 3: Plus sign

pch = 4: Cross

pch = 5: Diamond

pch = 6: Triangle (pointing down)

pch = 7: Square cross

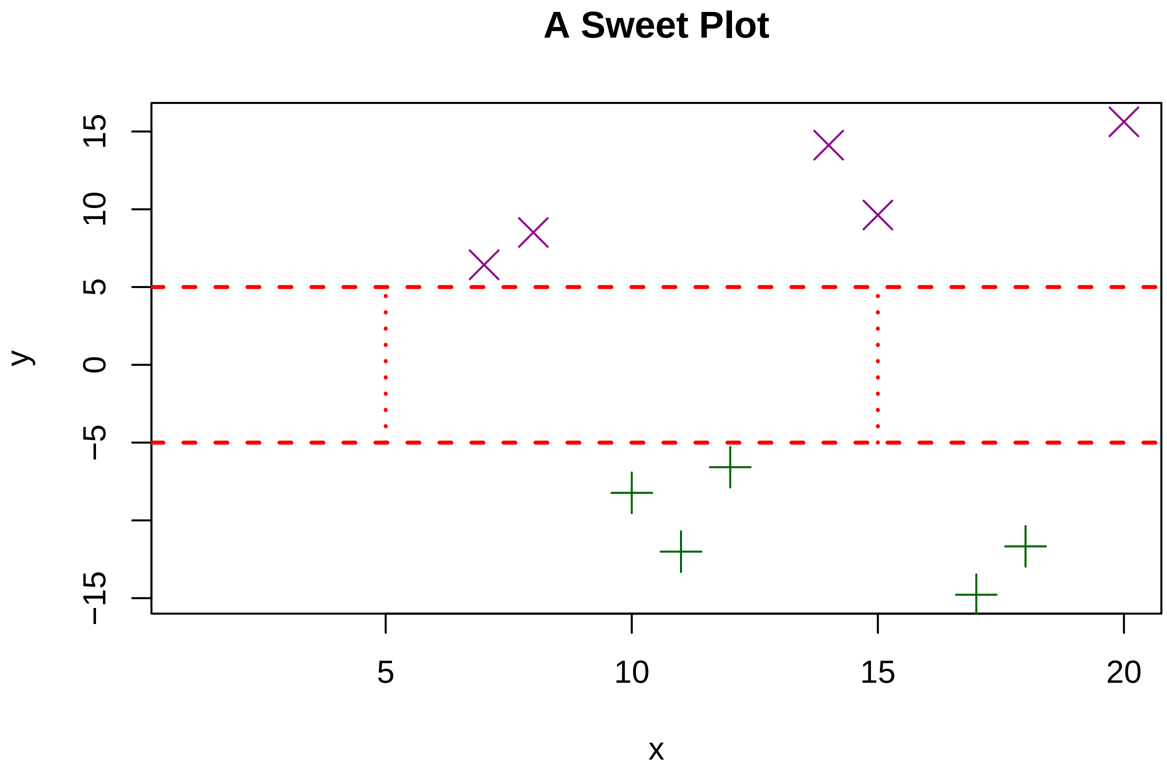
pch = 8: Star

pch = 9: Diamond plus

pch = 10: Circle plus

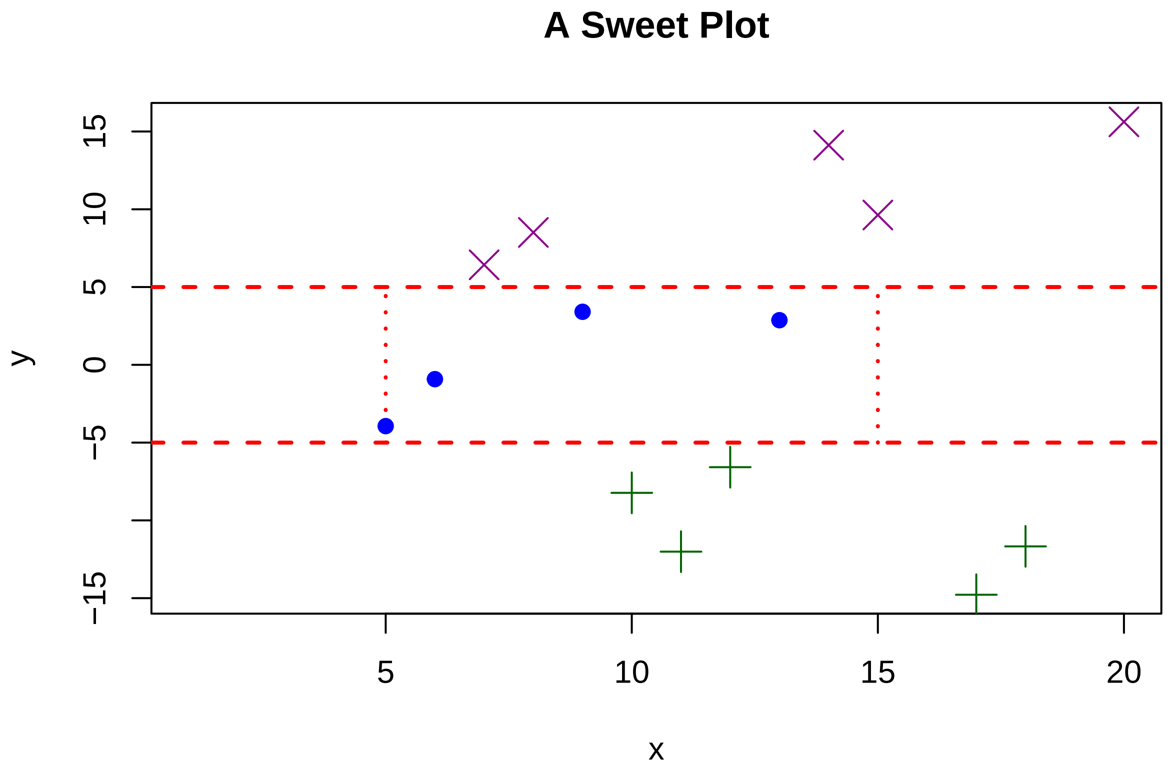
Step 5: Add “too small” points

Plot the points *(x,y)* such that y −5, using “darkgreen” + signs.



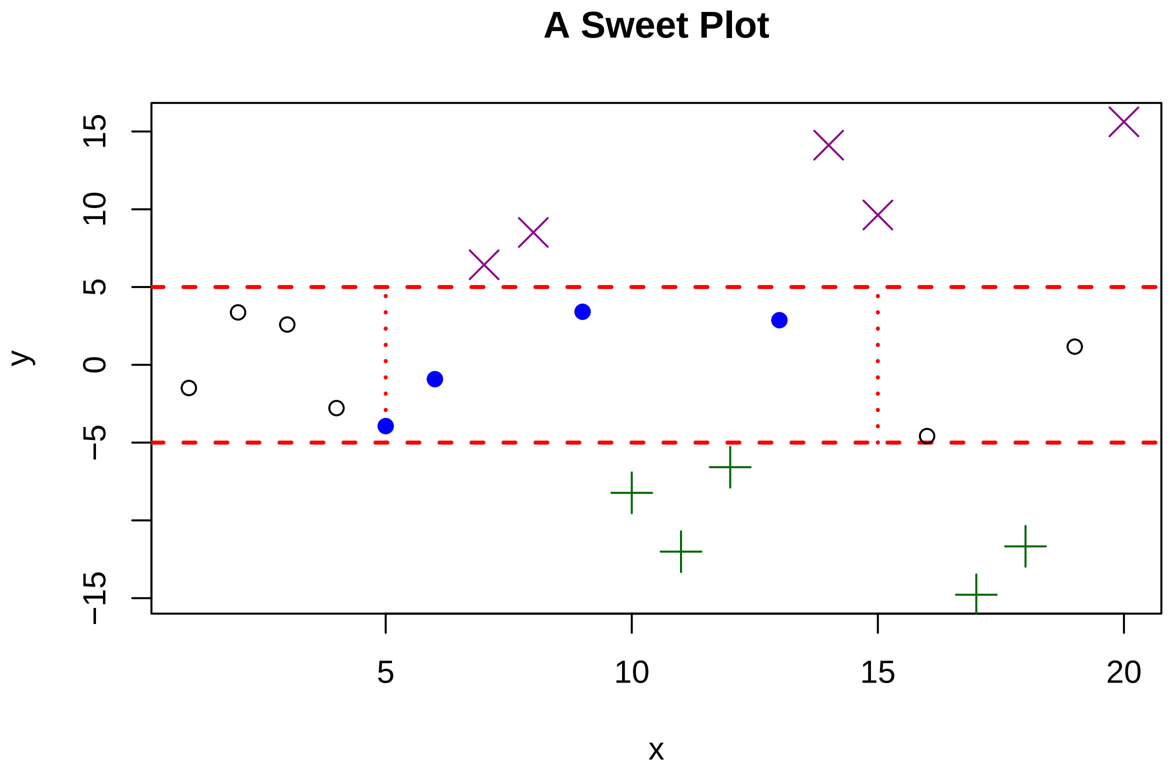
Step 6: Add “sweet spot” points

Plot the points satisfying all of x 5, x 15, y>−5, and y<5 using blue solid dots. Remember that & is a vectorized logical AND operator.



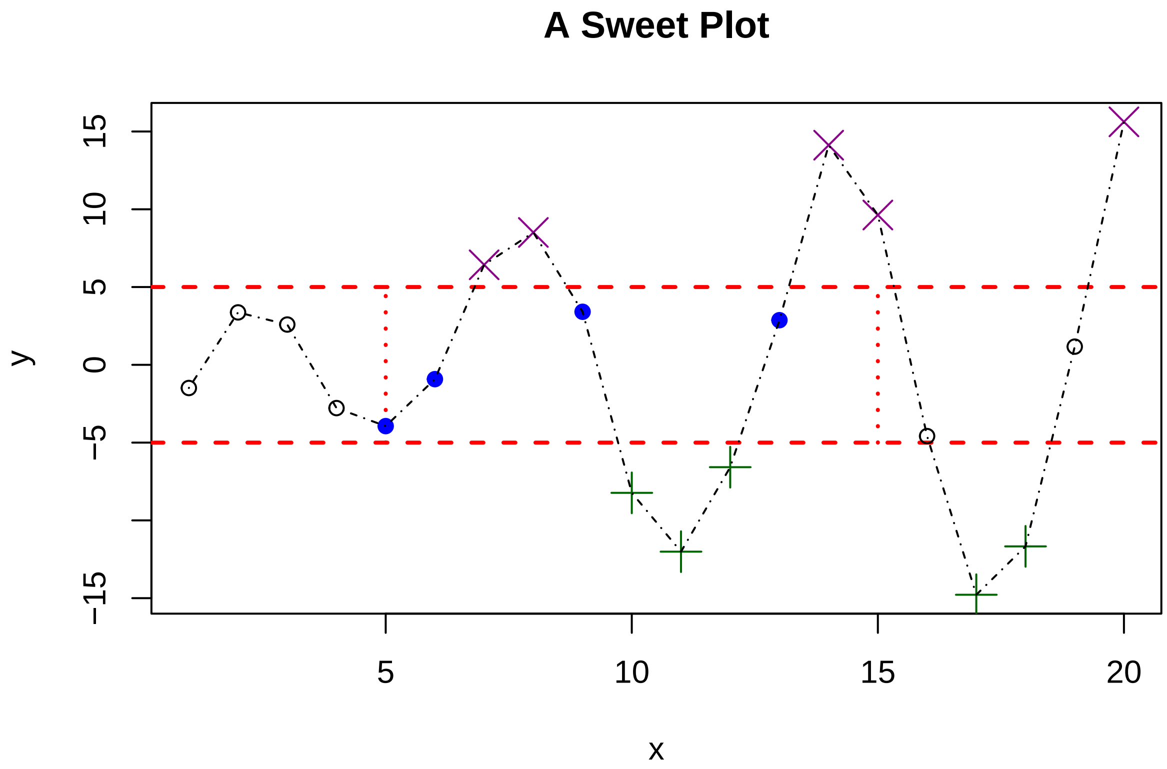
Step 7: Add the rest of the points

Finally, we must account for the rest of the points. These points satisfy (x<5 OR x>15) AND (y>−5 AND y<5). Plot them with no graphical parameters (so they are black empty circles by default).



Step 8: Connect the dots

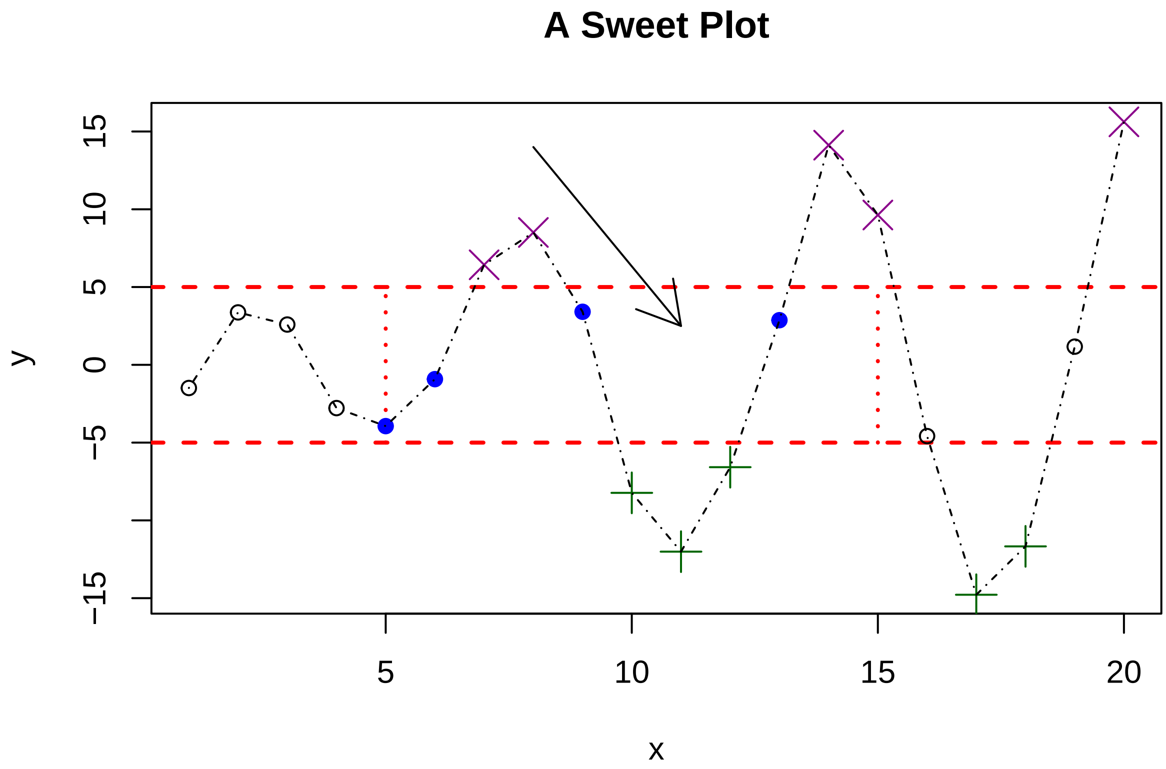
Use `lines` to connect the dots as follows:



# x, y = coordinate vectors of points to join. # type = character indicating the type of plotting; actually any of the types as in plot.default.

Step 9: Add sweet spot arrow

Use `arrows` to add an arrow. This function is a lot like `segments` from Step 3, except there is an arrow head at one end. Add an arrow pointing from (8,14) to (11,2.5)

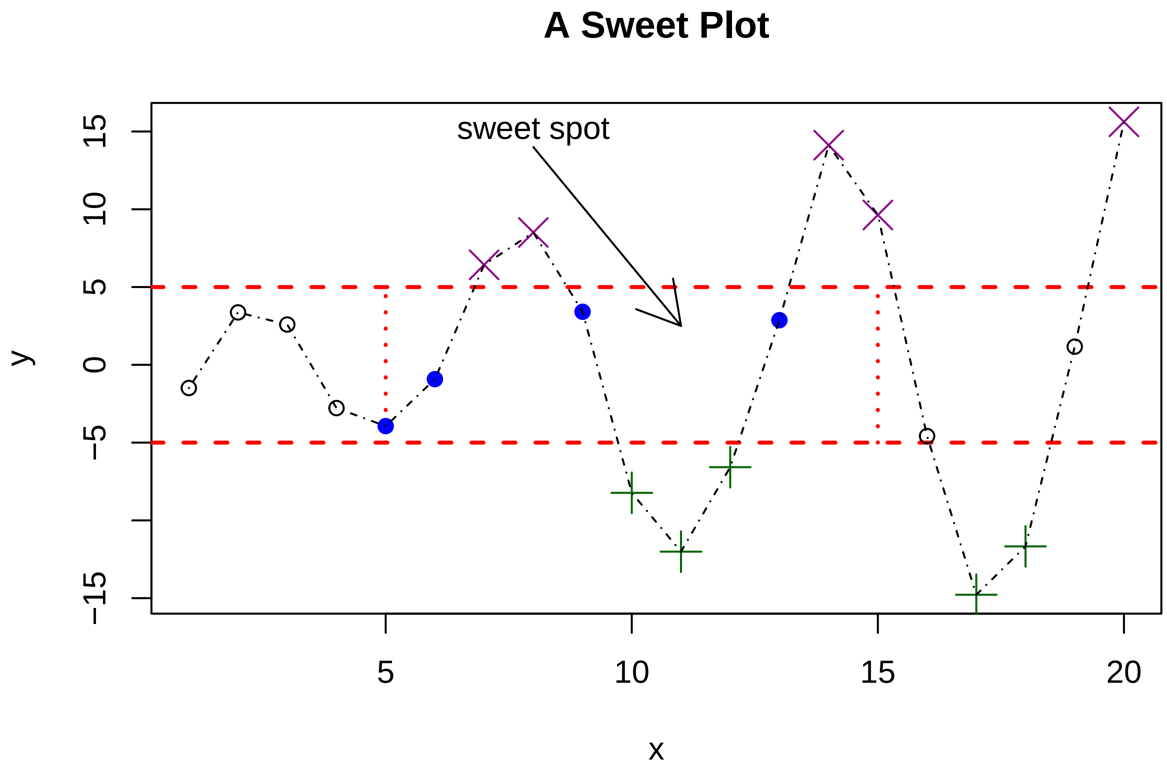


# x0, y0 = coordinates of points from which to draw. # x1, y1 = coordinates of points to which to draw. At least one must the supplied # length = length of the edges of the arrow head (in inches). # angle = angle from the shaft of the arrow to the edge of the arrow head. # code = integer code, determining kind of arrows to be drawn. # col, lty, lwd = graphical parameters, possible vectors. NA values in col cause the arrow to be omitted.

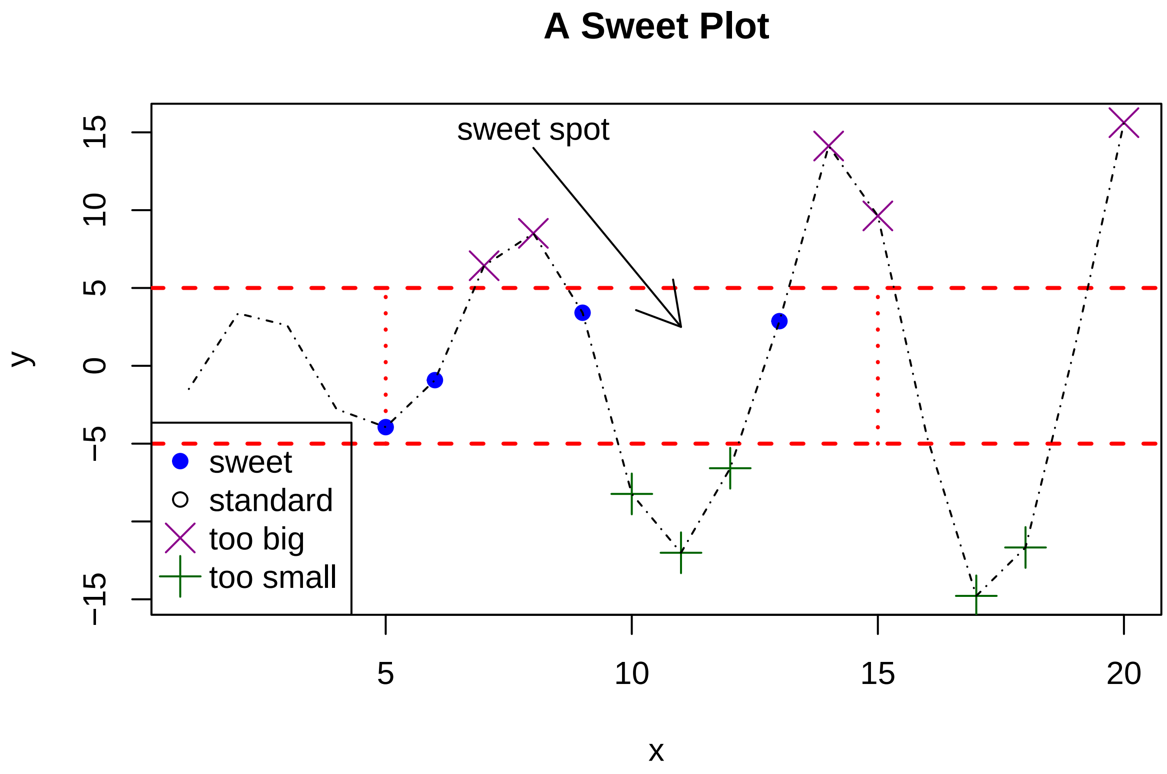
Step 10: Add a “sweet spot” label to the arrow

Text is added to an existing plot with the `text` function. Add the text “sweet spot” at the point (8,15) as follows:

```
text(8, 15, labels = "sweet spot")
```


A Sweet Plot
sweet spot
y
15
10
5
0
−5
−15
x
5
10
15
20

Step 11: Add a legend

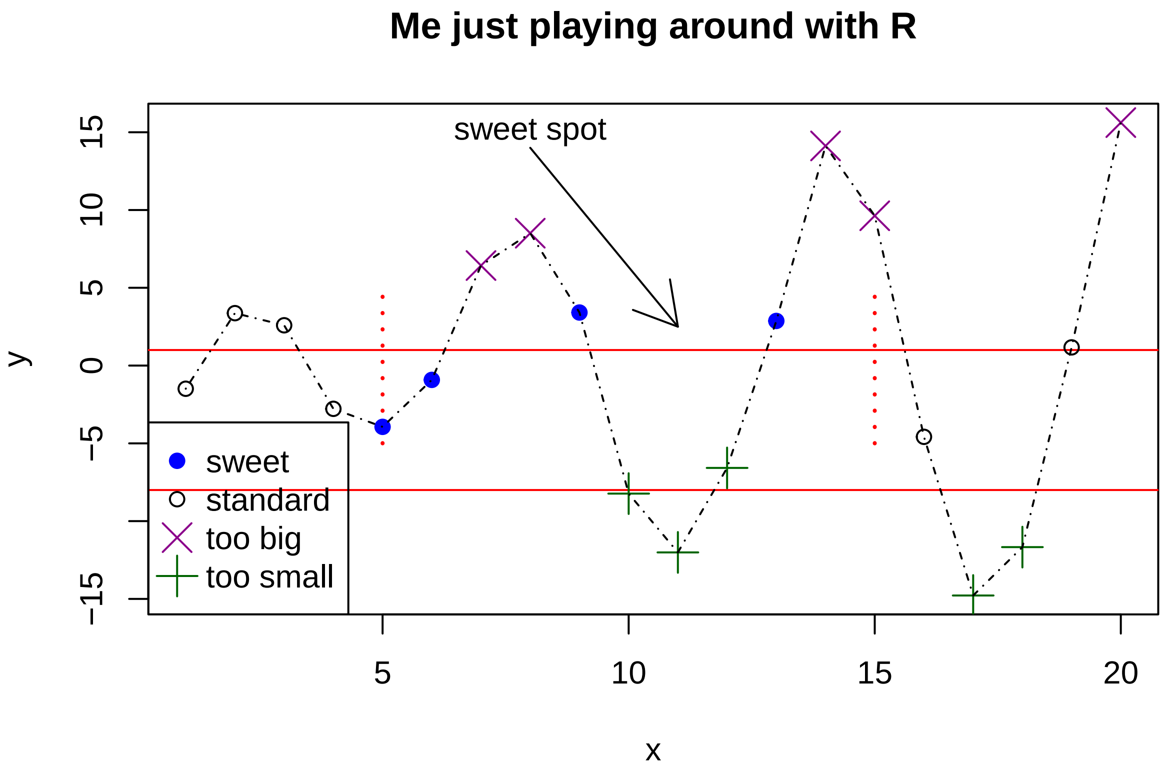


legend = a character or expression vector of length to appear in the legend

pch = the plotting symbols appearing in the legend (vector)

col = " the color of points or lines appearing in the legend

pt.cex = expansion factors for the points



Play around!!!