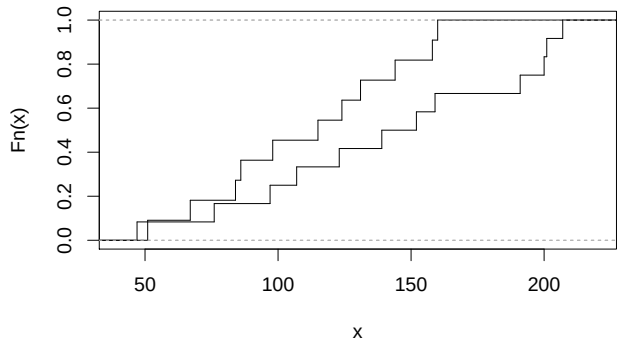
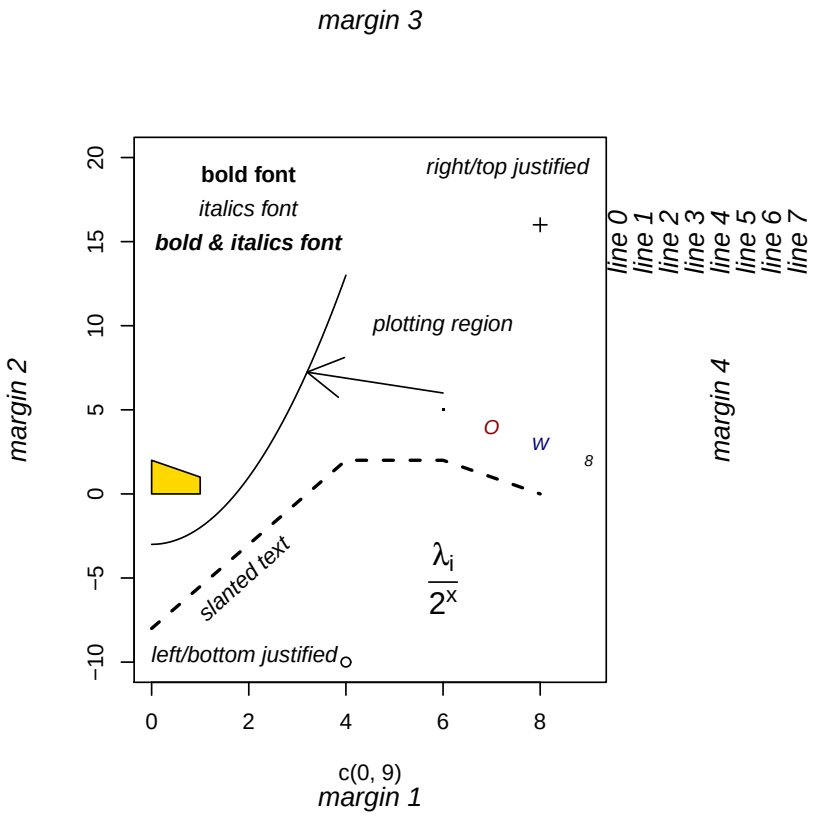


The first R command below calls the `par` function with the `mar` (margin) argument, which controls the sizes of the four margins, bottom, left, top, and right, in that order, measured in number of lines of text. This ordering of the margins is used by other R graphical functions. The second R command calls the `plot` function, which would ordinarily plot the points $(0, -10)$ and $(9, 20)$, but setting the `type` argument to the string `"n"` means that no points will be plotted. In addition, the y-axis will have no label because the `ylab` argument is set to an empty string. The `text` function is used to plot the string `"plotting region"` centered at the point $(6, 10)$. The `points` function is first called to plot the default style point at $(4, -10)$. Then `points` is called with the `pch` (plotting character) argument set to `3`. It is called a third time to print four different characters (`pch`), in four different colors (`col`), in four different sizes (`cex` for character expand). The `polygon` function plots a shaded polygon with the points associated with the x-values given in the first argument and the y-values given in the second argument. The `lines` function plots a line by connecting the points associated with the x-values given in the first argument and the y-values given in the second argument, with the `lwd` (line width) argument specifying the thickness of the line and the `lty` (line type) argument specifying the type of line (1 for solid, 2 for dashed, 3 for dotted, etc.). The `mtext` (margin text) function places text in the margins. The `srt` (string rotate) argument in the `text` function allows for slanted text (the argument is in degrees). The `font` argument in the `text` function can be used for bold text (`font = 2`), italics text (`font = 3`), bold italix text (`font = 4`), and Greek text (`font = 5`). The `adj` (adjust) argument in the `text` function specifies whether the text should be left justified (`adj = 0`), centered (`adj = 0.5`, the default), or right justified (`adj = 1`). The `adj` argument can assume any value on the interval $[0, 1]$. The `expression` function can be used for plotting mathematical expressions like $\lambda_i / 2^x$. The `lines` function plots the parabola $y = x^2 - 3$. Finally, the `arrows` function draws an arrow from $(6, 6)$ to a point on the parabola.

```
> par(mar = c(5, 8, 5, 8))
> plot(c(0, 9), c(-10, 20), type = "n", ylab = "")
> text(6, 10, "plotting region")
> points(4, -10)
> points(8, 16, pch = 3)
> points(6:9, 5:2, pch = c(".", "O", "W", "8"),
+        col = c("black", "red4", "navy", "gray2"),
+        cex = seq(1.0, 0.7, by = -0.1))
> polygon(c(0, 0, 1, 1), c(0, 2, 1, 0), col = "gold")
> lines(c(0, 4, 6, 8), c(-8, 2, 2, 0), lwd = 2, lty = 2)
> mtext(paste("margin", 1:4), side = 1:4, line = 4)
> mtext(paste("line", 0:7), side = 4, line = 0:7, at = 15)
> text(1.9, -5, "slanted text", srt = 42)
```

```
> text(2, 19, "bold font", font = 2)
> text(2, 17, "italics font", font = 3)
> text(2, 15, "bold & italics font", font = 4)
> text(2, 13, "abeG", font = 5)
> text(0, -10, "left/bottom justified", adj = c(0, 0))
> text(9, 20, "right/top justified", adj = c(1, 1))
> text(6, -5, expression(frac(lambda[i], 2 ^ x)), cex = 1.5)
> x = seq(0, 4, by = 0.1)
> y = x ^ 2 - 3
> lines(x, y)
> arrows(6, 6, x[33], y[33])
```



The `plot` function can also be used for plotting univariate data. The R command

```
> plot(precip)           # plot precipitation data
```

for example, plots the $n = 70$ average precipitation values contained in the built-in data set `precip` on the vertical axis and `1:70` on the horizontal axis.

It is more often the case that the `plot` function is used to graph data pairs. Recall that the built-in `cars` data set contains two variables: `speed`, which is the speed of the vehicle in miles per hour, and `dist`, which is the stopping distance in feet. The $n = 50$ data pairs were collected in the 1920s,