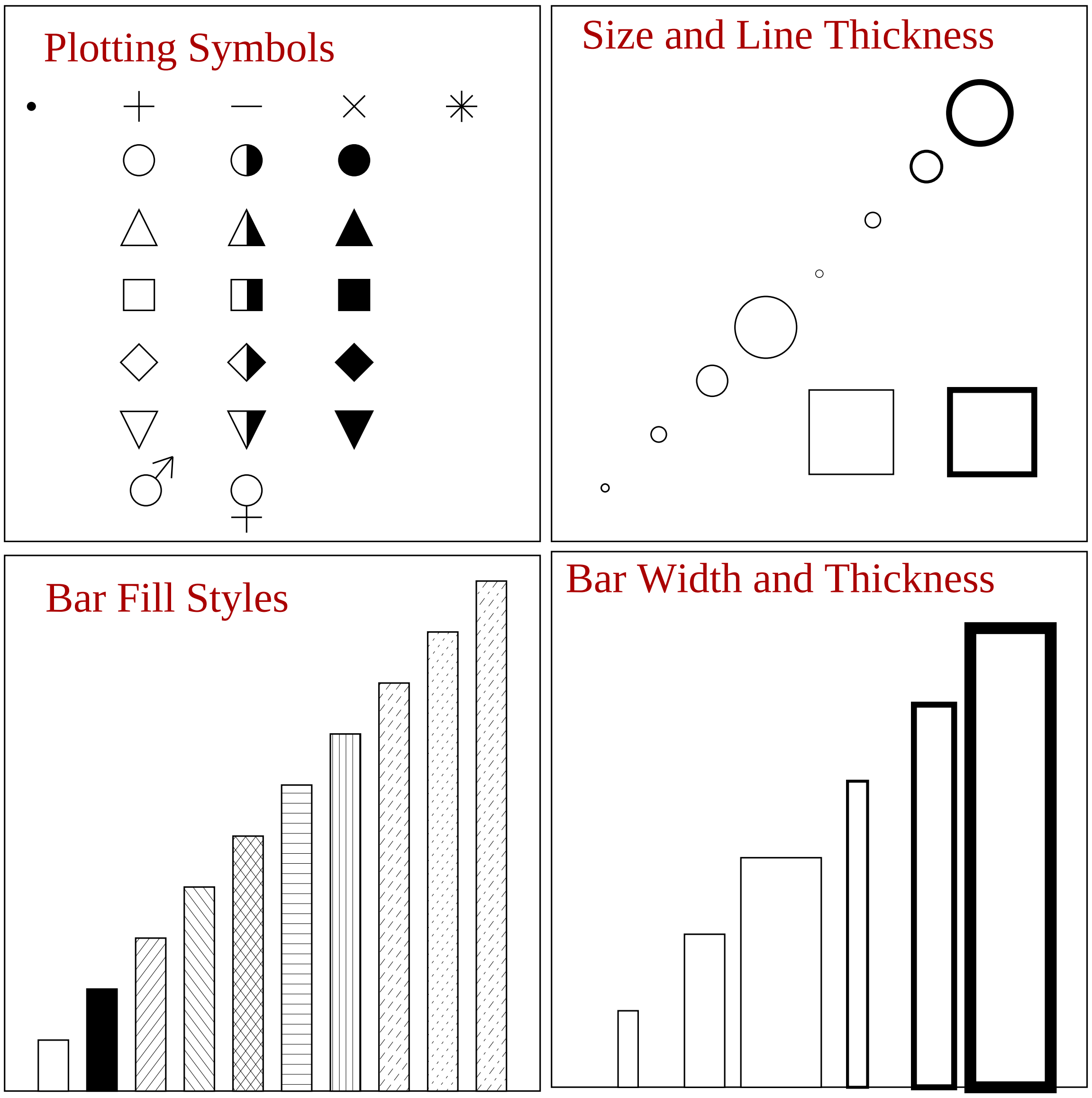
Tutorials and worked examples for simulation, curve fitting, statistical analysis, and plotting.
http://www.simfit.org.uk

Symbols

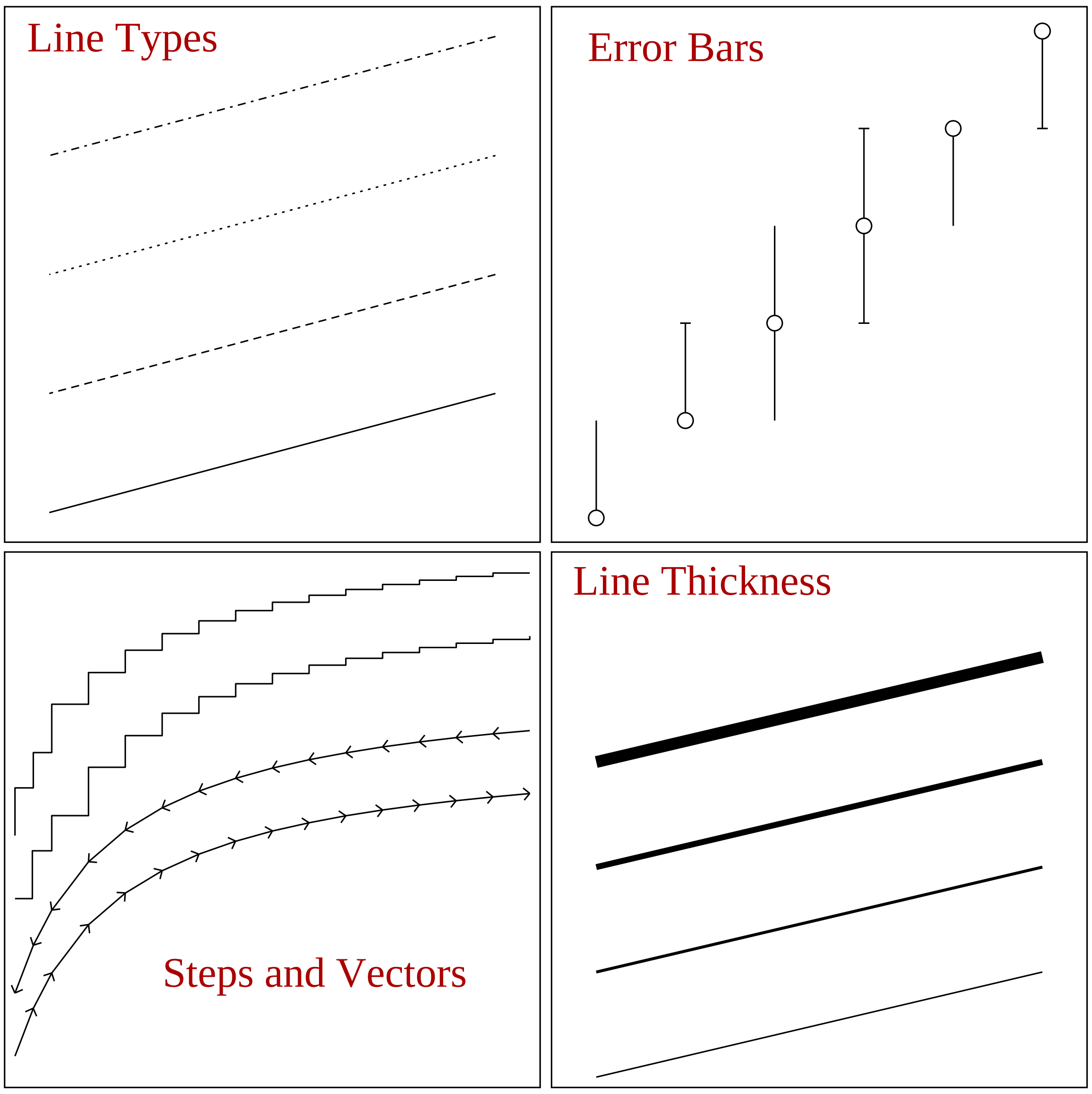
The collage below illustrates the following features of SimFIT graphics.

- There are 22 plotting symbols plus some outline-only symbols.
- The size and line thickness used to plot symbols can be varied.
- There are 10 alternative fill styles for bars in addition to colors.
- The bar width and line thickness used to plot bars can be varied.



Lines: standard types

There are four standard SIMFIT line types, normal, dashed, dotted and dot-dashed, and error bars can terminate with or without end caps if required, as shown in the collage below.

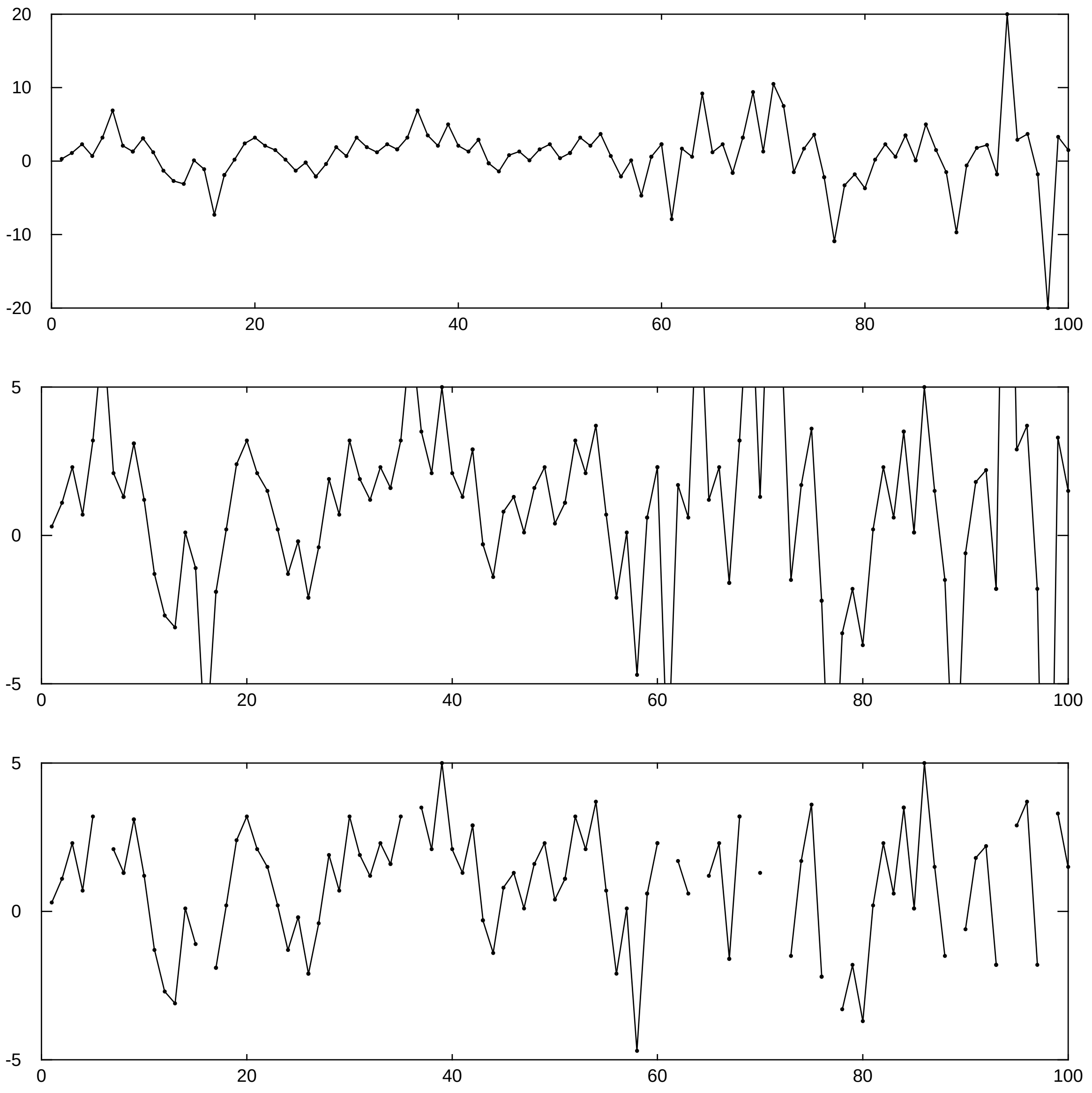


Special effects can be created using stair step lines, which can be used to plot *cdfs* for statistical distributions, or survival curves from survivor functions, and vector type lines, which can be used to plot orbits of differential equations. Note that steps can be first y then x, or first x then y, while vector arrows can point in the direction of increasing or decreasing t, and lines can have variable thickness.

Program **simplot** reads in default options for the sequence of line types, symbol types, colors, barchart styles, piechart styles and labels which will then correspond to the sequence of data files. Changes can be made interactively and stored as graphics configuration templates if required. However, to make permanent changes to the defaults, you configure the defaults from the main SIMFIT configuration option, or from program **simplot**.

Lines: extending to boundaries

The collage below illustrates the alternative techniques available in SimFIT when the data to be plotted are clipped to boundaries so as to eliminate points that are identified by symbols and also joined by lines.

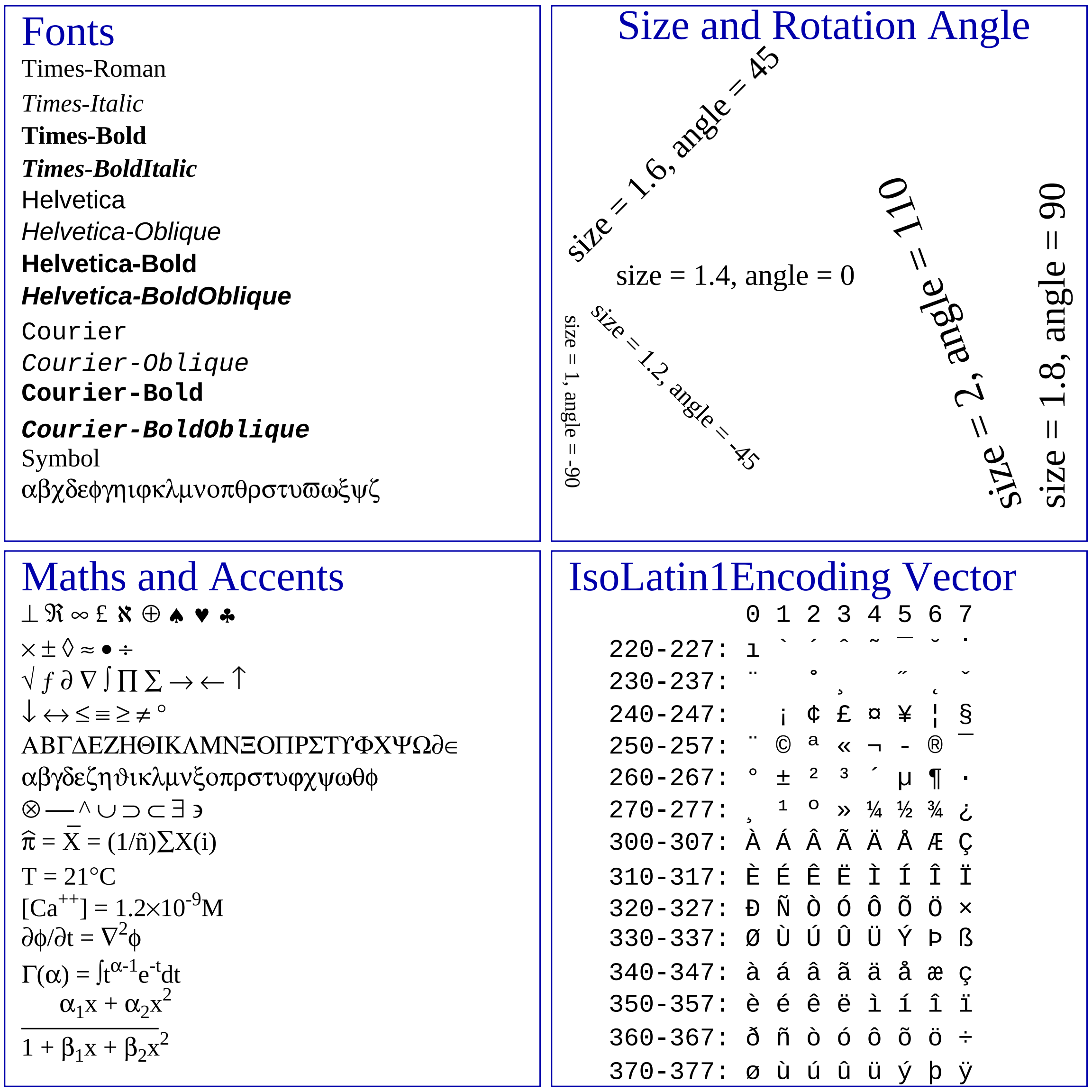


The first graph shows what happens when the test file `zigzag.tf1` was plotted with dots for symbols and lines connecting the points, but with all the data within the boundaries. The second graph illustrates how the lines can be extended to the clipping boundary to indicate the direction in which the next undisplayed symbol is located, while the third figure shows what happens when the facility to extend lines to boundaries is suppressed.

Note that these plots were first generated as .ps files using the flat-shape plotting option, then a PostScript *x* stretch factor of 2 was selected, followed by the use of GSview to transform to .eps and so recalculate the BoundingBox parameters.

Text

The next collage shows how fonts can be used in any size or rotation and with many nonstandard accents, e.g., $\hat{\theta}$.



Special effects can be created using graphics fonts such as ZapfDingbats, or user-supplied dedicated special effect functions, as described elsewhere. Scientific symbols and simple mathematical equations can be generated, but the best way to get complicated equations, chemical formulas, photographs or other bitmaps into SimFIT graphs is to use PSfrag or **editps**.

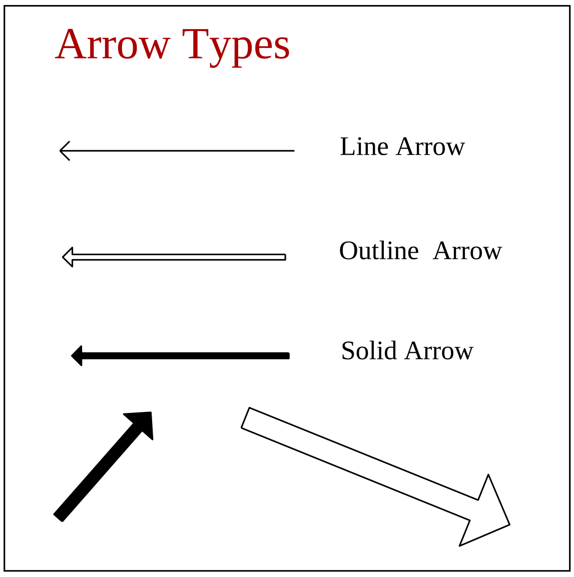
The collage also demonstrates several possibilities for displaying mathematical formulae directly in SimFIT graphs, and it also lists the octal codes for some commonly required characters from the IsoLatin1 encoding. Actually, octal codes can be typed in directly (e.g., `\361` instead of ñ), but note that text strings in SimFIT plots can be edited at two levels: at the simple level only standard characters can be typed in, but at the advanced level nonstandard symbols and maths characters can be selected from a font table. Note that, while accents can be added individually to any standard character, they will not be placed so accurately as when using the corresponding hard-wired characters e.g., from the IsoLatin1 encoding.

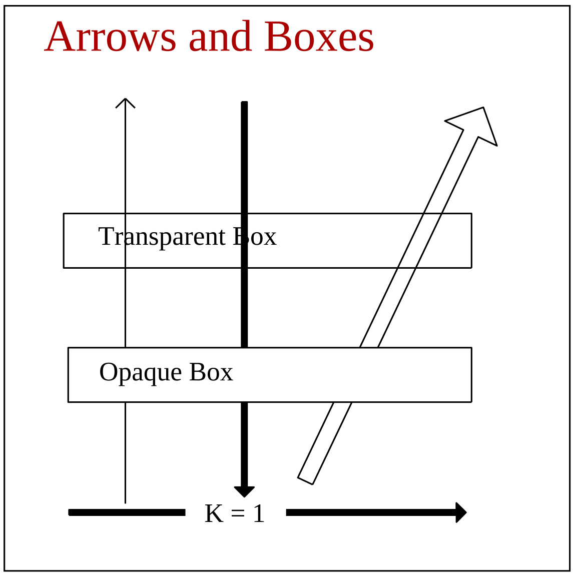
Fonts, character sizes and line thicknesses

The fonts, letter sizes, and line thicknesses used in SimFIT graphics are those chosen from the PostScript menu, so, whenever a font or line thickness is changed, the new details are written to the PostScript configuration file `w_ps.cfg`. If the size or thickness selected is not too extreme, it will then be stored as the default to be used next time. However, it should be noted that, when the default sizes are changed, the titles, legends, labels, etc. may not be positioned correctly. You can, of course, always make a title, legend, or label fit correctly by moving it about, but, if this is necessary, you may find that the defaults are restored next time you use SimFIT graphics. If you insist on using an extremely small or extremely large font size or line thickness and SimFIT keeps restoring the defaults, then you can overcome this by editing the PostScript configuration file `w_ps.cfg` and making it read-only. Users who know PostScript will prefer to use the advanced PostScript option, whereby the users own header file can be automatically added to the PostScript file after the SimFIT dictionary has been defined, in order to re-define the fonts, line thicknesses or introduce new definitions, logos plotting symbols, etc.

Arrows

The next figures show that arrows can be of three types: line, hollow or solid and these can be of any size.





However use can be made of headless arrows to create special effects. From this point of view a headless line arrow is simply a line which can be solid, dashed, dotted or dash-dotted. These are useful for adding arbitrary lines. A headless outline arrow is essentially a box which can be of two types: transparent or opaque. Note that the order of priority in plotting is

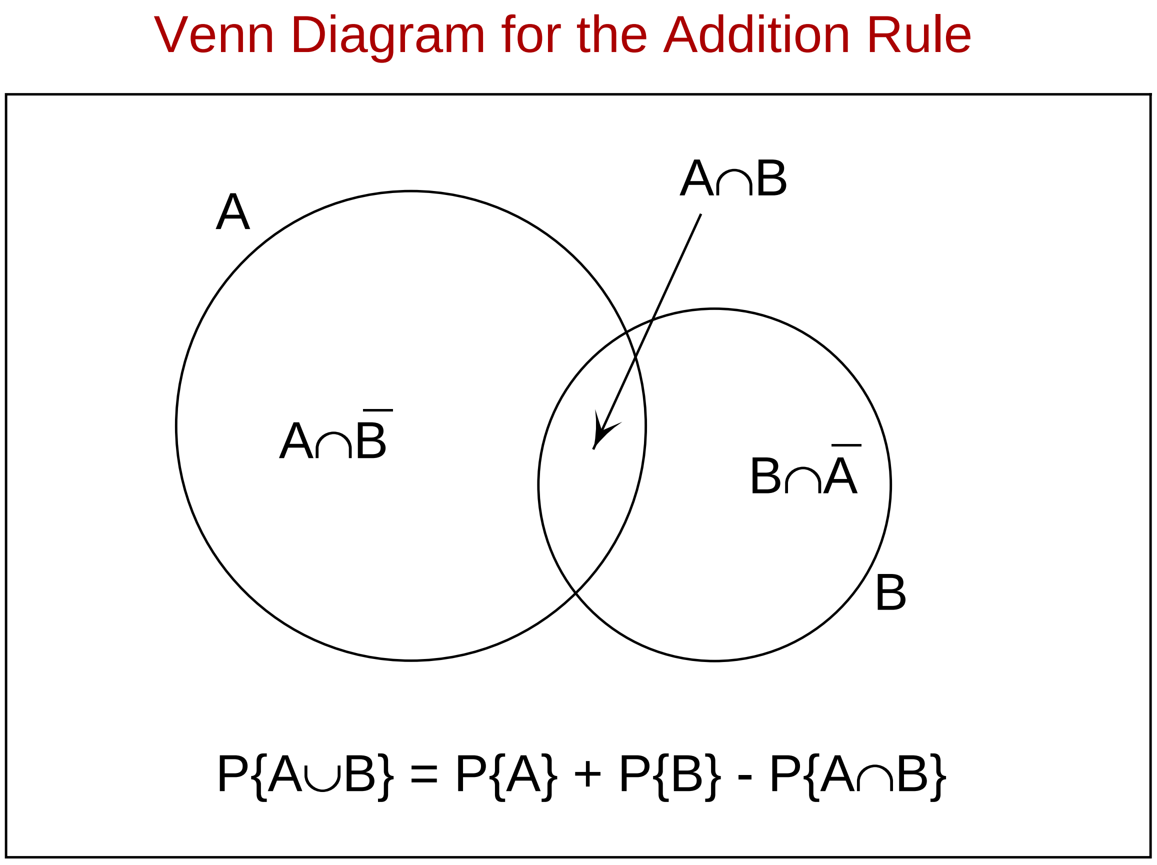
$$\boxed{\text{Extra Text} > \text{Graphical Objects} > \text{Data plotted, titles and legends}}$$

and this allows boxes to be used to simply obliterate plotted data or to surround extra text allowing the background to show through.

Transparent boxes, are useful for surrounding information panels, opaque boxes are required for chemical formulae or mathematical equations, while background colored solid boxes can be used to blank out features as shown in the above figures. To surround a text string by a rectangular box for emphasis, position the string, generate a transparent rectangular box, then drag the opposing corners to the required coordinates.

Example of plotting without data: Venn diagram

It is possible to use program **simplot** as a generalized diagram drawing program without any data points, as illustrated in the next figure.

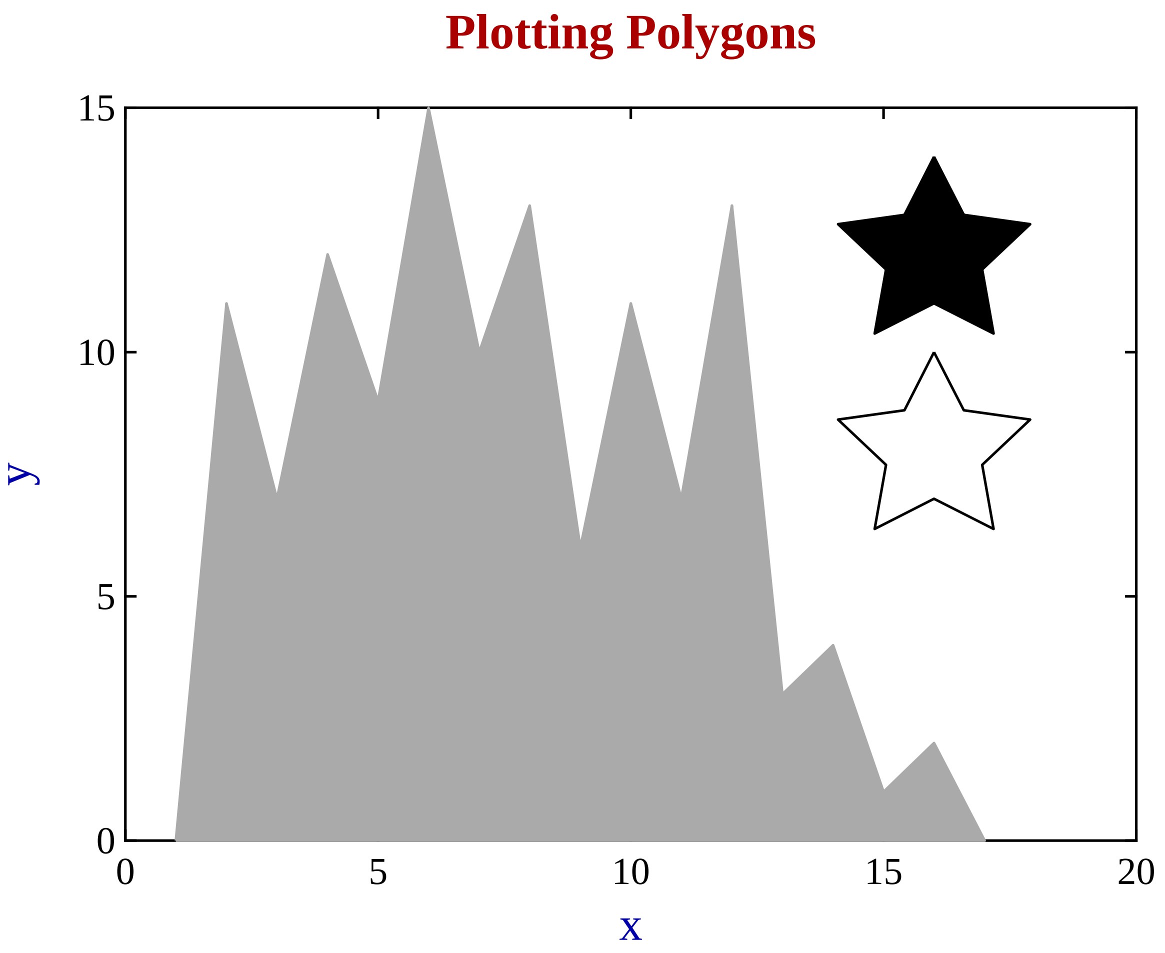

Venn Diagram for the Addition Rule
A
$A\cap B$
$A\cap\bar{B}$
$B\cap\bar{A}$
B
$P\{A\cup B\} = P\{A\} + P\{B\} - P\{A\cap B\}$

The procedure used to create such graphs using any of the SIMFIT graphical objects will be clear from the details now given for this particular Venn diagram.

- ❍ Program **simplot** was opened using an arbitrary dummy graphics coordinate file.
- ❍ The [Data] option was selected and it was made sure that the dummy data would not be plotted as lines or symbols, by suppressing the lines and symbols for this dummy data set.
- ❍ This transformed program **simplot** into an arbitrary diagram creation mode and the display became completely blank, with no title, legends, or axes.
- ❍ The circles were chosen (as objects) to be outline circle symbols, the box was selected (as an arrow-line-box) to be a horizontal transparent box, the text strings were composed (as text objects), and finally the arrow was chosen (as an arrow-line-box) to be a solid script arrow.
- ❍ The diagram was then completed by editing the text strings (in the expert mode) to introduce the mathematical symbols.

Polygons

Program **simplot** allows filled polygons as an optional linetype. So this means that any set of n coordinates (x_i, y_i) can be joined up sequentially to form a polygon, which can be empty if a normal line is selected, or filled with a chosen color if the filled polygon option is selected. If the last (x_n, y_n) coordinate pair is not the same as the first (x_1, y_1), the polygon will be closed automatically. This technique allows the creation of arbitrary plotting objects of any shape, as will be evident from the sawtooth plot and stars in the next figure.



The sawtooth graph above was generated from a set of (x, y) points in the usual way, by suppressing the plotting symbol but then requesting a filled polygon linetype, colored light gray. The open star was generated from coordinates that formed a closed set, but then suppressing the plotting symbol and requesting a normal, i.e. solid linetype. The filled star was created from a similar set, but selecting a filled polygon linetype, colored black.

If you create a set of ASCII text plotting coordinates files containing arbitrary polygons, such as logos or special plotting symbols, these can be added to any graph. However, since the files will simply be sets of coordinates, the position and aspect ratio of the resulting objects plotted on your graph will be determined by the ranges you have chosen for the x and y axes, and the aspect ratio chosen for the plot. Clearly, objects created in this way cannot be dragged and dropped or re-scaled interactively. The general rule is that the axes, title, plot legends, and displayed data exist in a space that is determined by the range of data selected for the coordinate axes. However, extra text, symbols, arrows, information panels, etc. occupy a fixed space that does not depend on the magnitude of data plotted. So, selecting an interactive data transformation will alter the position of data dependent structures, but will not move any extra text, lines, or symbols.