

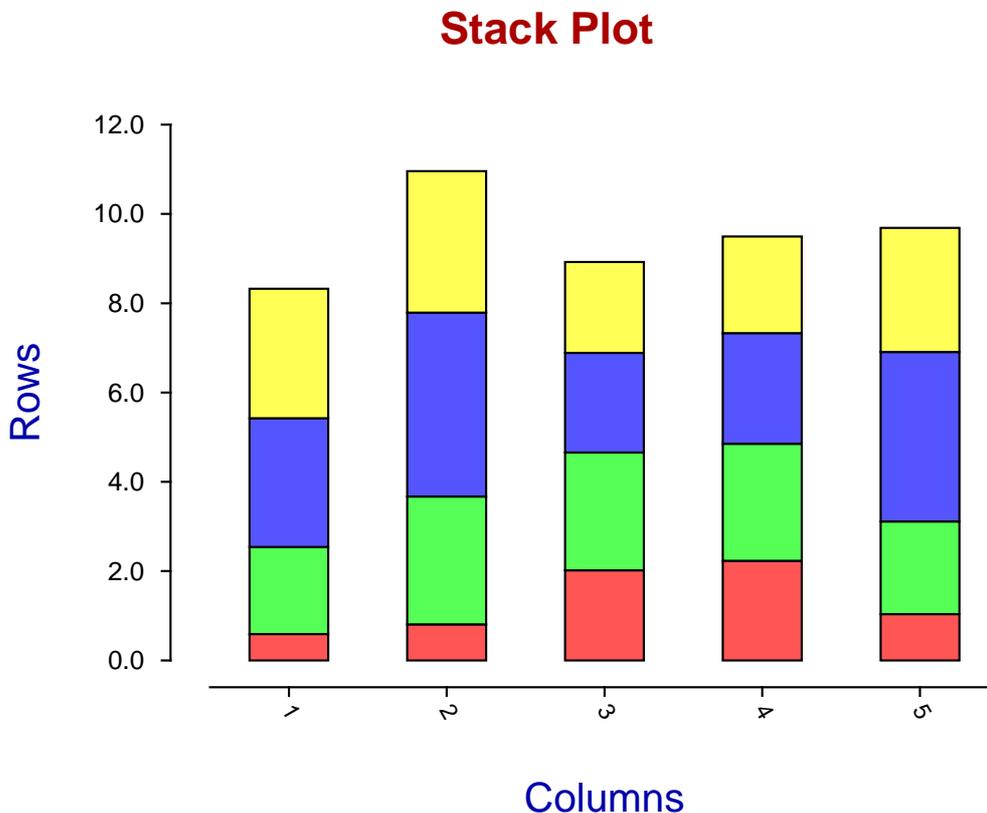


Tutorials and worked examples for simulation,
curve fitting, statistical analysis, and plotting.
<https://simfit.org.uk>

It is frequently convenient to edit SVG files retrospectively, usually in order to change sizes, titles, legends, labels, line-types, line-widths, and colors, etc. Fortunately, SVG files, like EPS files, are in ASCII text format so they can be edited using any text editor that supports UTF8 characters, such as the Windows program **notepad** or better **notepad++**. The actual format is in XML which is similar to HTML but, in addition, the SVG files created by SIMFIT have been designed with editing in mind. For instance, each individual markup code starts on a new line.

Titles and Legends

As a simple example consider the following stacked bar chart created by importing the SIMFIT metafile `stack_plot.metafile` into SIMFIT program **simplot** or SIMDEM program **simdem70**.



On searching for the string "Stack Plot" in the file `stack_plot.svg` using a text editor we find the following markup at line 34, where the tokens have been shown on separate lines for clarity.

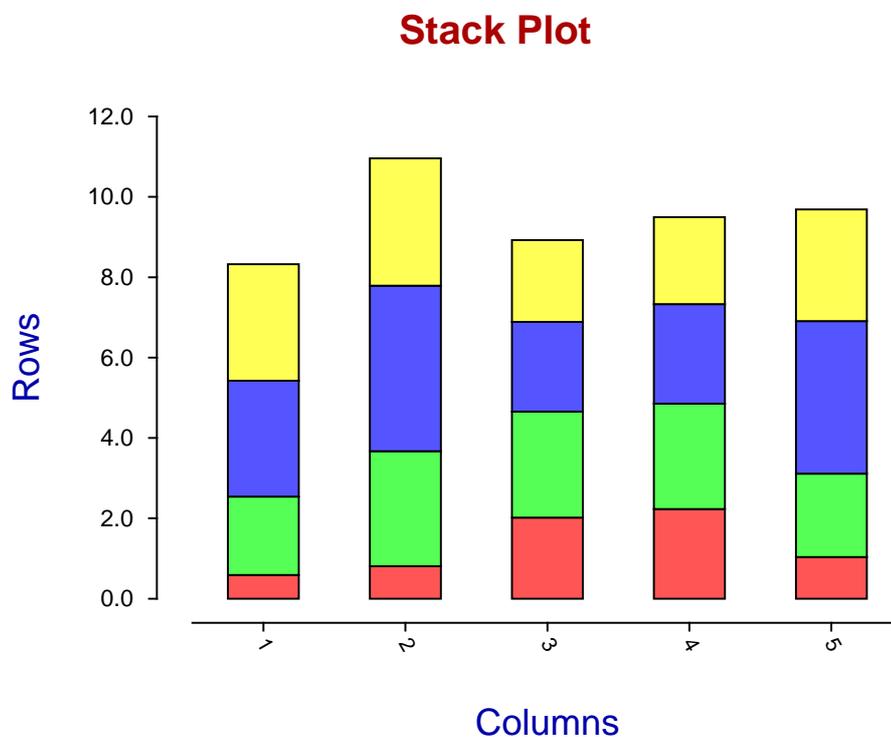
```
<text  
x="420.00" y="44.72"  
font-family="ARIAL"  
font-size="47px"  
font-weight="700"
```

```
fill="rgb(170,0,0)">
Stack Plot</text>
```

Note the key markup codes used in this line.

- `<text ... ></text>`
- `font-family=`
- `font-size=`
- `font-weight=`
- `fill=`

We might think the title could be more explanatory and would be better using a more subdued coloring given the amount of color already associated with the stack segments. So, after some editing in **notepad++**, re-displaying using **firefox** gives the next graph.



Lines 34, 35, and 36 now look like this (with line numbers added for clarity).

```
34 <text x="330.00" y="44.72" font-family="ARIAL" font-size="47px"
    font-weight="700" fill="rgb(0,0,170)">Bar Chart in Stacked Format</text>
35 <text x="504.00" y="788.92" font-family="ARIAL" font-size="42px"
    font-weight="400" fill="rgb(0,170,0)">Columns</text>
36 <text x="68.00" y="449.44" font-family="ARIAL" font-size="44px"
    font-weight="400" transform="rotate(270.00,68.00,460.00)"
    fill="rgb(0,170,0)">Rows</text>
```

The following editing will be apparent on inspection.

Line 34

The original short title "Stack Plot" has been changed to the longer title "Bar Chart in Stacked Format", the color has been changed from red `rgb(170,0,0)` to blue `rgb(0,0,170)`, and the x-coordinate has been changed from 420.0 to 330.00 (so that the title remains centralized over the plot).

Line 35

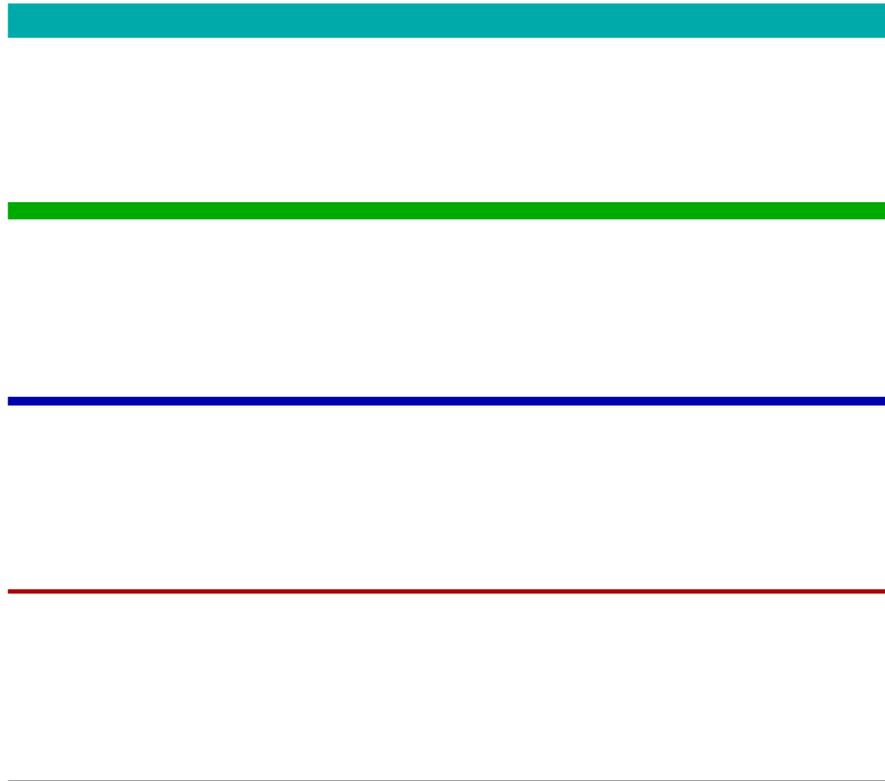
The x-legend color has been changed from blue `rgb(0,0,170)` to green `rgb(0,170,0)`

Line 36

The y-legend color has been changed from blue `rgb(0,0,170)` to green `rgb(0,170,0)`

Lines and Curves

Consider the next plot with five lines, each in a different color and line width.



This plot can easily be created by reading the `SIMFIT` metafile `lines.metafile` into `SIMFIT` program `simplot` or `SIMDEM` program `simdem70`.

Note that in this plot the lines were created by drawing a polyline between two points which could easily have been drawn as a simple line. However a polyline was drawn as it would be more usual to change the line type, thickness, and color for a smooth curve which would then have made it more difficult to comprehend with the presence of a large number of coordinates, whereas only the end points are needed for a straight line.

The point is that, in order to search for a graphical object in a SVG file, the the markup code would be used. For instance, searching for the text string `<polyline points=` in the file `lines.svg` locates the following code on a single line, but here broken up into separate tokens for clarity.

```
<polyline points="196.81,823.62 1095.69,823.62"  
style="fill:none;stroke:rgb(0,0,0) ;stroke-width:1.84"  
stroke-linecap="butt"  
stroke-linejoin="round" />
```

To understand this code the following summary is presented.

1. `<polyline points= ... />`
This markup section starting with `<` and ending with `>` defines all the properties of the polyline.
2. `style=`
Here all the details of the curve to be drawn are to be found.
3. `:rgb(0,0,0)`
This color convention is exactly as used for HTML, where red, blue, and green color components are defined on a scale from 0 to 255.
4. `stroke width:`
This defines the line width in pixels.
5. `stroke-linecap=`
This defines the way the ends of curves are finished off.
6. `stroke-linejoin=`
This is where the way that the sections of the polyline making up the curve are to be connected together is defined.

In fact the code for drawing all five lines is as follows , where dots ... are used to indicate the text omitted for clarity.

```
... style="fill:none;stroke:rgb(0,0,0) ;stroke-width:1.84" ...  
... style="fill:none;stroke:rgb(170,0,0) ;stroke-width:2.00" ...  
... style="fill:none;stroke:rgb(0,0,170) ;stroke-width:4.00" ...  
... style="fill:none;stroke:rgb(0,170,0) ;stroke-width:8.00" ...  
... style="fill:none;stroke:rgb(0,170,170) ;stroke-width:16.00" ...
```

Note that the order of these commands is in the direction from bottom to top, so that the first line (`stroke-width:1.84`) refers to the bottom line, while the last command (`stroke-width:16`) refers to the top line.

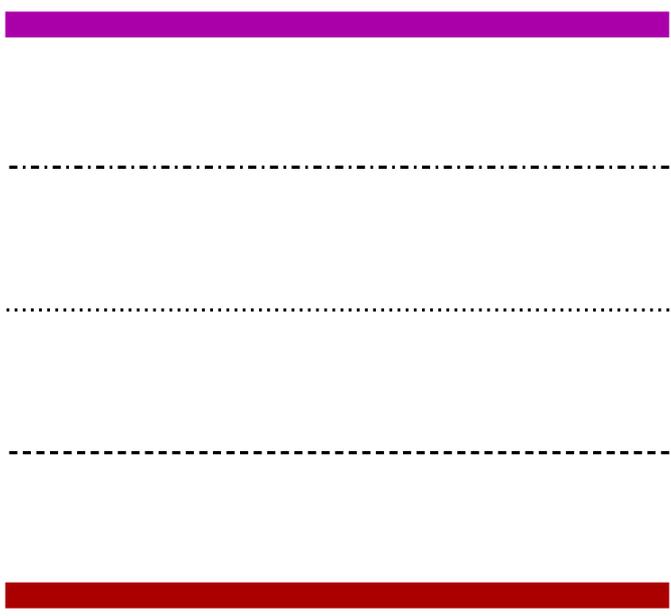
Another change that is often required is to swap between solid lines and dashed, dotted, or dash-dotted lines which requires the definition of a `stroke-dasharray` parameter as follows.

- `stroke-dasharray="18.00,12.00"` for dashed lines
- `stroke-dasharray="6.00,12.00"` for dotted lines
- `stroke-dasharray="18.00,12.00,6.00,12.00"` for dash-dotted lines

To demonstrate this technique, note that replacing this section of `lines.svg` by the next section

```
:rgb(170,0,0) ;stroke-width:16.00"  
:rgb(0,0,0) ;stroke-width:2.00" stroke-dasharray="18.00,12.00"  
:rgb(0,0,0) ;stroke-width:2.00" stroke-dasharray="6.00,12.00"  
:rgb(0,0,0) ;stroke-width:2.00" stroke-dasharray="18.00,12.00,6.00,12.00"  
:rgb(170,0,170) ;stroke-width:16.00"
```

generates the following graph with the dashed, dotted, and dash-dotted lines colored black.



Character Strings and Fonts

A common need is to reposition a character string, to edit the string, to change the font size, or to change the colour. Consider, for instance, the following diagram.

This is Arial/Helvetica size = 1.2

This is Arial/Helvetica size = 2.0

This is Arial/HelveticaBold size 2.0 in red

Arial BoldOblique rotated 90 degrees
Arial BoldOblique rotated -90 degrees

Times Roman rotated 45 degrees

Times Roman rotated 0 degrees

Times Roman rotated -45 degrees

αβχδεφγηιφκλμνοπθρστυϖωξψζ
ΑΒΧΔΕΦΓΗΙΘΚΛΜΝΟΠΘΡΣΤΥΖΩΞΨΖ

Here is the first character string with Arial/Helvetica at size 1.2, but broken into separate tokens for clarity.

```
<text x="289.00" y="62.48"  
font-family="ARIAL"  
font-size="23px"  
font-weight="400"  
fill="rgb(0,0,0)"  
>This is Arial/Helvetica size = 1.2  
</text>
```

If this is understood then editing such a SVG file will be simple. Here is what the rules are.

- `<text ...`
This indicates the start of a new character string which includes several self-evident definitions such as these.
 - font-family
 - font-size
 - font-weight
 - fill
- `This is Arial/Helvetica size = 1.2`
This is the actual string itself that is going to be displayed
- `</text>`
This indicates the end of the instructions to display the string

The command to rotate 90 degrees, again with tokens separated for clarity, is as follows.

```
<text x="232.00" y="705.00"  
font-family="ARIAL"  
font-size="25px"  
font-weight="700"  
transform="rotate(270.00,232.00,711.00)"  
font-style = "oblique"  
fill="rgb(0,140,0)">  
Arial BoldOblique rotated 90 degrees</text>
```

The following files are distributed with the SIMFYT package in order to understand the previous details.

1. lines.metafile, new_lines.metafile, fonts.metafile
These can be used to generate the figures using the SIMFYT program **simplot** or the SIMDEM program **simdem70**
2. lines.eps, new-lines.eps, fonts.eps
PostScript graphics files
3. lines.svg, new-lines.svg, fonts.svg
SVG graphics files

Note that, to edit text strings containing non-ASCII characters as in the lower strings using Symbol font, an editor supporting UTF8 must be used.