

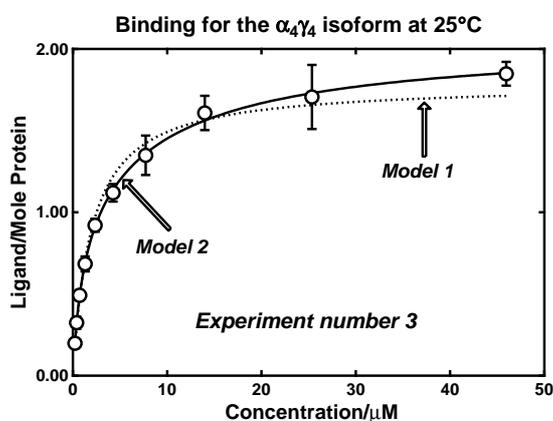
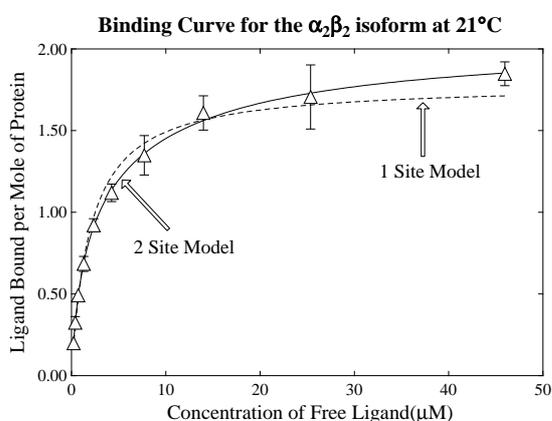


Tutorials and worked examples for simulation,
curve fitting, statistical analysis, and plotting.
<http://www.simfit.org.uk>

One of the unique features of SIMFIT PostScript files is that the format is designed to make retrospective editing easy. A typical example of when this could be useful would be when a graph needs to be changed for some reason. Typically an experimentalist might have many plots stored as .eps files and want to alter one for publication or presentation. SIMFIT users are strongly recommended to save all their plots as .eps files, so that they can be altered in the way to be described. Even if you do not have a PostScript printer it is still best to save as .eps, then use GSview/Ghostscript to print or transform into another graphics format.

Introductory example

Consider these next two figures, showing how a graph can be transformed by simple editing in a text editor, e.g., **notepad**.



This type of editing should always be done if you want to use one figure as a reduced size inset figure inside another, or when making a slide, otherwise the SIMFIT default line thickness will be too thin. Note that most of the editing to be described below can actually be done at the stage of creating the file, or by using program **editps**. In this hypothetical example, we shall suppose that the experimentalist had realized that the title referred to the wrong isoform and temperature, and also wanted to add extra detail, but simplify the graph in order to make a slide using thicker lines and a bolder font. In the following sections the editing required to transform the SIMFIT example file `simfig1.ps` will be discussed, following a preliminary warning.

Warning about editing PostScript files

In the first place the technique to be described can only be done with SIMFIT PostScript files, because the format was developed to facilitate the sort of editing that scientists frequently need to perform. Secondly, it must be realized that PostScript files must conform to a very strict set of rules. If you violate these rules, then GSview/Ghostscript will warn you and indicate the fault. Unfortunately, if you do not understand PostScript, the warning will be meaningless. So here are some rules that you must keep in mind when editing.

- Always keep a backup copy at each successful stage of the editing.
- All text after a single percentage sign % to the line end is ignored in PostScript.
- Parentheses must always be balanced as in (figure 1(a)) not as in (figure 1(a)).
- Fonts must be spelled correctly, e.g. Helvetica-Bold and not helveticabold.

- Character strings for displaying must have underneath them a vector index string of EXACTLY the same length.
- When introducing non-keyboard characters each octal code represents one byte.
- The meaning of symbols and line types depends on the function, e.g. da means dashed line while do means dotted line.

A review of the PostScript colours, fonts and conventions is also in the `w_readme` files. In the next sections it will be assumed that are running `SimFIG` and have a renamed copy of `simfig1.ps` in your text editor (e.g., **notepad**), and after each edit you will view the result using `GSview`/`Ghostscript`. Any errors reported when you try to view the edited file will be due to violation of a PostScript convention. The most usual one is to edit a text string without correctly altering the index below it to have exactly the same number of characters.

The percent-hash escape sequence

Later versions of `SimFIG` create PostScript files that can be edited by a stretch, clip, slide procedure, which relies on each line containing coordinates being identified by a comment line starting with `%#`. All text extending to the right from the first character of this sequence can safely be ignored and is suppressed for clarity in the following examples.

Changing line thickness and plot size

The following text will be observed in the original `simfig1.ps` file.

```
72.00 252.00 translate 0.07 0.07 scale 0.00 rotate
11.00 setlinewidth 0 setlinecap 0 setlinejoin [] 0 setdash
2.50 setmiterlimit
```

The postfix argument for `setlinewidth` alters the line width globally. In other words, altering this number by a factor will alter all the linewidths in the figure by this factor, irrespective on any changes in relative line thicknesses set when the file was created. The `translate`, `scale` and `rotate` are obvious, but perhaps best done by program **editps**. Here is the same text edited to increase the line thickness by a factor of two and a half.

```
72.00 252.00 translate 0.07 0.07 scale 0.00 rotate
27.50 setlinewidth 0 setlinecap 0 setlinejoin [] 0 setdash
2.50 setmiterlimit
```

Changing PostScript fonts

In general the Times-Roman fonts may be preferred for readability in diagrams to be included in books, while Helvetica may look better in scientific publications. For making slides it is usually preferable to use Helvetica-Bold. Of course any PostScript fonts can be used, but in the next example we see how to change the fonts in `simfig1.ps` to achieve the effect illustrated.

```
/ti-font /Times-Bold D%plot-title
/xl-font /Times-Roman D%x-legend
/yl-font /Times-Roman D%y-legend
/zl-font /Times-Roman D%z-legend
/tc-font /Times-Roman D%text centred
/td-font /Times-Roman D%text down
/tl-font /Times-Roman D%text left to right
/tr-font /Times-Roman D%text right to left
/ty-font /Times-Roman D%text right y-mid
/tz-font /Times-Roman D%text left y-mid
```



```

910 1581 958 1581 li
6118 1581 6070 1581 li
(0.50) 862 1581 ty
(0000) fx
910 2491 958 2491 li
6118 2491 6070 2491 li
(1.00) 862 2491 ty
(0000) fx
910 3401 958 3401 li
6118 3401 6070 3401 li
(1.50) 862 3401 ty
(0000) fx

```

This is the text, after suppressing the tick marks and notation for $y = 0.5$ and $y = 1.5$ by inserting a percentage sign. Note that the index must also be suppressed as well as the text string.

```

%910 1581 958 1581 li
%6118 1581 6070 1581 li
%(0.50) 862 1581 ty
%(0000) fx
910 2491 958 2491 li
6118 2491 6070 2491 li
(1.00) 862 2491 ty
(0000) fx
%910 3401 958 3401 li
%6118 3401 6070 3401 li
%(1.50) 862 3401 ty
%(0000) fx

```

Changing line and symbol types

This is simply a matter of substituting the desired line or plotting symbol key.

```

Lines      : li (normal) da (dashed) do (dotted) dd (dashed dotted) pl (polyline)
Circles    : ce (empty) ch (half)   cf (full)
Triangles  : te (empty) th (half)   tf (full)
Squares    : se (empty) sh (half)   sf (full)
Diamonds   : de (empty) dh (half)   df (full)
Signs      : ad (add)   mi (minus)  cr (cross)  as (asterisk)

```

Here is the original text for the dashed line and empty triangles.

```

5697 3788 120 da
933 1032 72 te
951 1261 72 te
984 1566 73 te
1045 1916 72 te
1155 2346 72 te
1353 2708 73 te
1714 3125 72 te
2367 3597 72 te
3551 3775 72 te
5697 4033 72 te

```

Here is the text edited for a dotted line and empty circles.

```
5697 3788 120 do
933 1032 72 ce
951 1261 72 ce
984 1566 73 ce
1045 1916 72 ce
1155 2346 72 ce
1353 2708 73 ce
1714 3125 72 ce
2367 3597 72 ce
3551 3775 72 ce
5697 4033 72 ce
```

Adding extra text

Here is the original extra text section.

```
/font /Times-Roman D /size 216 D
GS font F size S 4313 2874 M 0 rotate
(1 Site Model)
(000000000000) fx
/font /Times-Roman D /size 216 D
GS font F size S 1597 2035 M 0 rotate
(2 Site Model)
(000000000000) fx
```

Here is the above text after changing the font.

```
/font /Helvetica-BoldOblique D /size 216 D
GS font F size S 4313 2874 M 0 rotate
(Model 1)
(0000000) fx
/font /Helvetica-BoldOblique D /size 216 D
GS font F size S 1597 2035 M 0 rotate
(Model 2)
(0000000) fx
```

Here is the additional code required to add another label to the plot.

```
/font /Helvetica-BoldOblique D /size 240 D
GS font F size S 2250 1200 M 0 rotate
(Experiment number 3)
(00000000000000000000) fx
```

Changing colors

For instance, to have a red (c4) title instead of a black (c0) title, change

```
c4
(Survival Analysis) 3195 4467 ti##title
(000000000000000000) fx
```

into

```
c0
(Survival Analysis) 3195 4467 ti##title
(000000000000000000) fx
```