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

Sometimes it is useful to collect a numbers of graphics files together in order to create a single graphics file. This is particularly easy and valuable to do using SimFIT *.eps files since the manipulations are facilitated by the presence of BoundingBox coordinates in the header sections, also the composite graph will be compact and device–independent with full resolution for display or printing. The options available are provided by program **editps** and are as now listed.

1. **Strict collages**

   Here there must be a set of graphics files, say *n*, that all have the same size, orientation, and aspect ratio. For instance, the SimFIT default portrait setting. Because all files have the same dimensions then all that is required is to decide on the number of columns, say *m*, required for the collage. Clearly, for best results *n* should be a multiple of *m*, but this is purely for appearance. Once a collage has been composed there are three other options that can be used.

   (a) A label can be added to facilitate reference to the individual graphs.

   (b) Such labels would usually be a single number or letter but short labels can also be added, exercising care to view the resultant collage before finally saving.

   (c) A section of text can be added at the bottom of the collage to describe the items. This should be in simple text and will be transformed into PostScript before adding to the graph. In order to use subscripts, superscripts, font changes, or mathematical symbols, there is a special syntax described in the SimFIT reference manual `w_manual.pdf`, or the tutorial document `pscodes.pdf`.

2. **Freestyle collages**

   In this type of collage the graphs can be of any size, orientation, or aspect ratio, because the user is going to resize the graphs and then move them about individually. After selecting the files to be used, a display is created where the graphs are represented as numbered and colored rectangles. These can be selected at will, moved to a new positions, then enlarged or diminished as required. The collage can be viewed repeatedly during this procedure until a correct assembly has been obtained before saving the final composite file.

3. **Insets**

   This procedure is used when it is wished to insert one or more child graphs into a parent graph, often the same data plotted in alternative coordinates. There are three vital things to bear in mind when carrying out this procedure.

   (a) Font size and line thickness.

   Because inserted graphs are going to be reduced in size, it is important to consider saving the original child graphs graphs to be inserted using increased font and symbol size and line thickness. Otherwise the inserted graphs could be hard to read.

   (b) Clipping

   It is useful to consider clipping the graphs to be inserted to remove surrounding background which, by default, is added by SimFIT to create a margin. Alternatively, program **GSview** can be used to create a new BoundingBox. Actually it is very easy to edit the header of SimFIT PostScript files to alter the clipping and BoundingBox coordinates. Try it.

   (c) Background opacity.

   When a SimFIT *.eps file is created there is an option to make white backgrounds transparent or opaque. It is recommended to consider whether the inset graph is going to obliterate the section of the master graph it overlies, or if the master graph should remain visible.

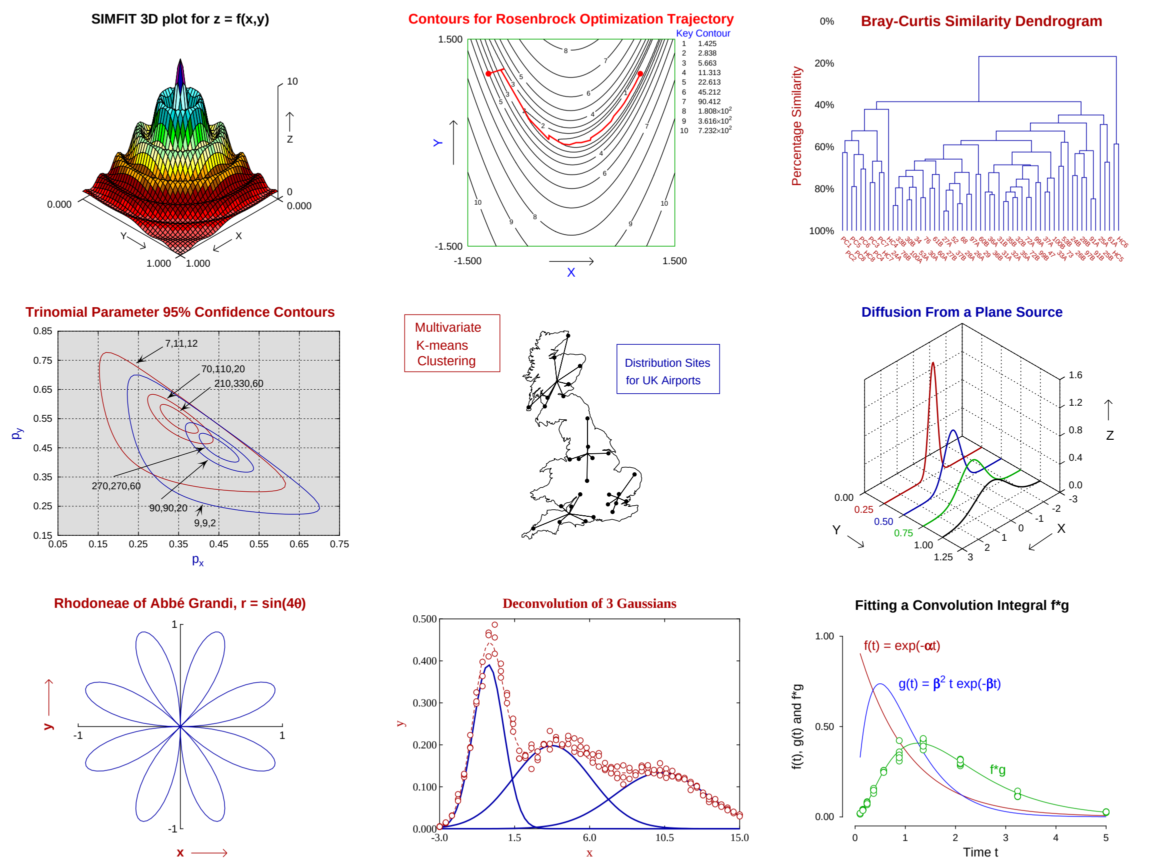
Example 1

The best way to input a collection of files is to create a library file for *.eps files using program **maklib**, and this was done to make the test library file `images.tfl` which contains this information.

```
Example of a EPS type library file
waves.eps
rosenbrock.eps
dendrogram.eps
trinom.eps
ukmap.eps
diffusion.eps
rose.eps
gauss3.eps
convolution.eps
```

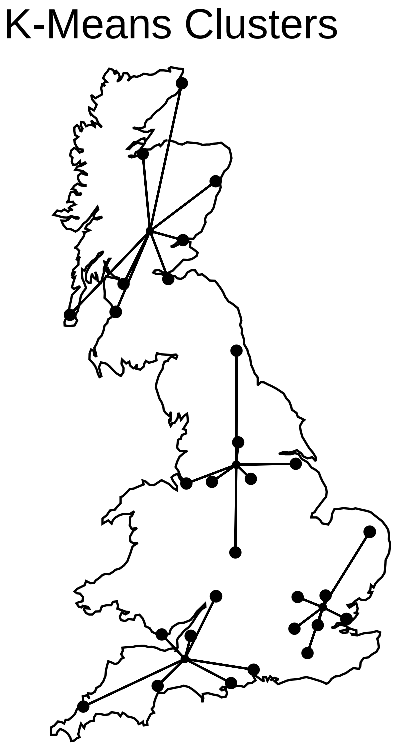
Note that the first line of a library file is an arbitrary title, while the next lines list the files in the order they will be used to make the collage. Normally these would be full paths, not local files names, and local file names are only used in this example because SIMFIT recognizes these short file names and can load the files.

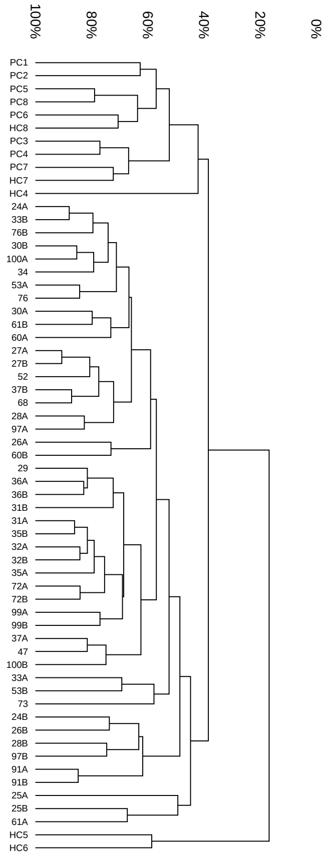
Opening program **editps**, choosing a structured, i.e. strict collage using the [Demo] button on the file opening control, then requesting three columns gives the collage illustrated below.

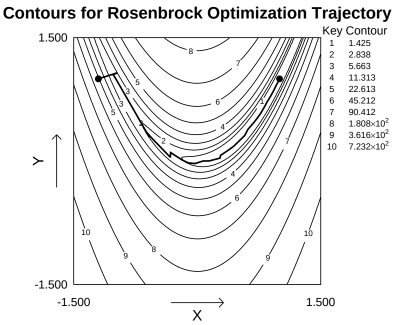


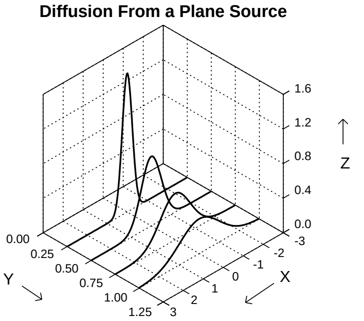
Example 2

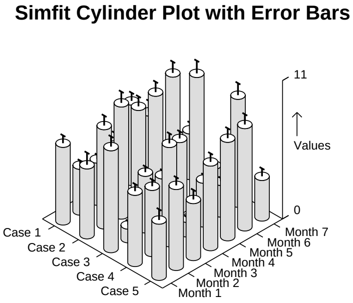
This illustrates a freestyle collage with different file sizes, aspect ratios and orientation.

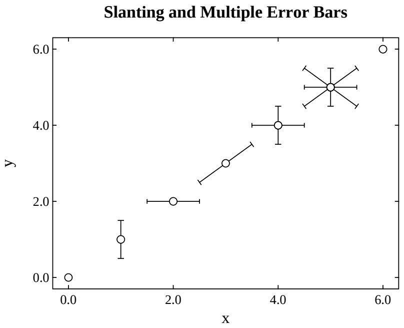

K-Means Clusters




Contours for Rosenbrock Optimization Trajectory


Diffusion From a Plane Source


Simfit Cylinder Plot with Error Bars


Slanting and Multiple Error Bars

Example 3

Figure 1 illustrates a special type of freestyle collage where a sub-graph is placed inside a parent graph. Sometimes it is best to enlarge the fonts and increase the line thicknesses when a sub-graph is going to be reduced in size in this way, and it is always important to remember the effects of opaque and transparent backgrounds that SIMFIT allows.

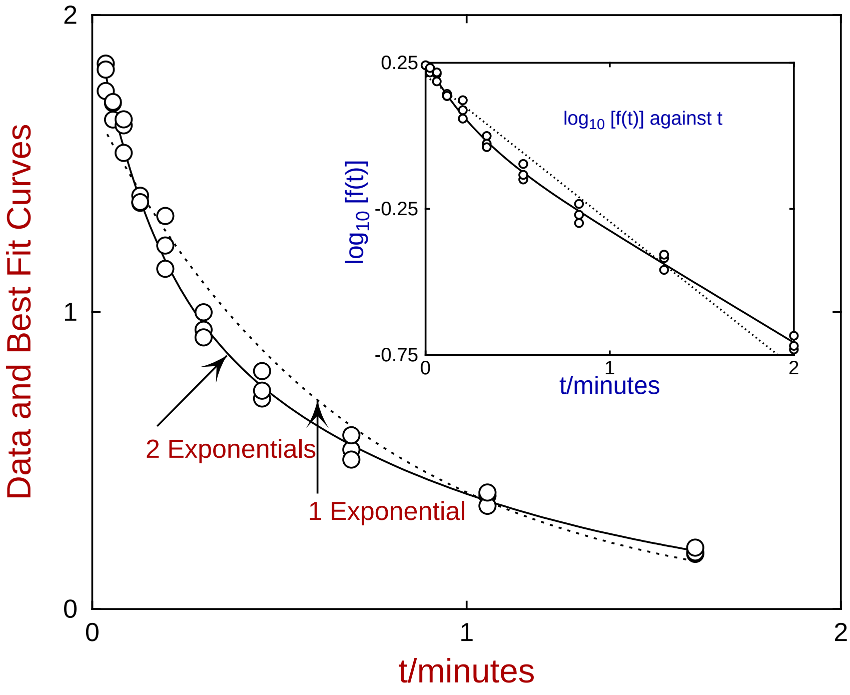


Figure 1: Subsidiary figures as insets

First of all the plots in figure 2 were created using **exfit** with test file `exfit.tf4` after fitting 1 exponential then 2 exponentials. Note that the line thickness and font size have been increased in the transformed plot as it is going to be reduced in the inset.

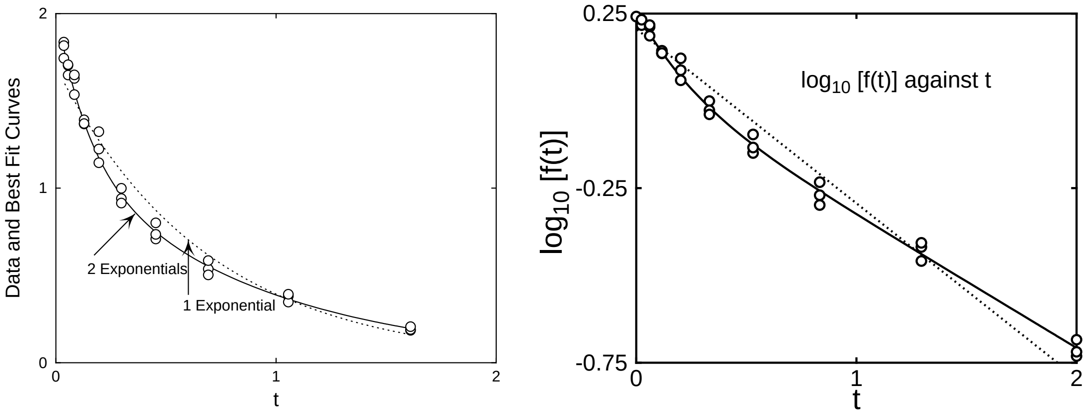


Figure 2: Insets 1: Exponential fitting and semilog transforms

Figure 3 was then created by **editps** using the option to create a freestyle collage. Note how, in the left hand plot the option to plot an opaque background even when white was selected and the transformed plot obscures the underlying main plot. In the right hand plot the option for a transparent background was used so that the main plot was not obscured. Both techniques are valuable when creating insets, and all that is now necessary to create figure 1 is to shrink the transformed plot and translate it to a more convenient location. A further point to note is that SimFIT plots have a border, which is obscuring more of the left hand main figure in figure 3 than seems necessary. When subsidiary figures are going to be used in this way it is often advisable to use the option to clip the plot to trim away extra white space, or else use program **GSview** to calculate a new BoundingBox in a transparent subsidiary plot by renaming *.eps as *.ps then transforming *.ps into *.eps.

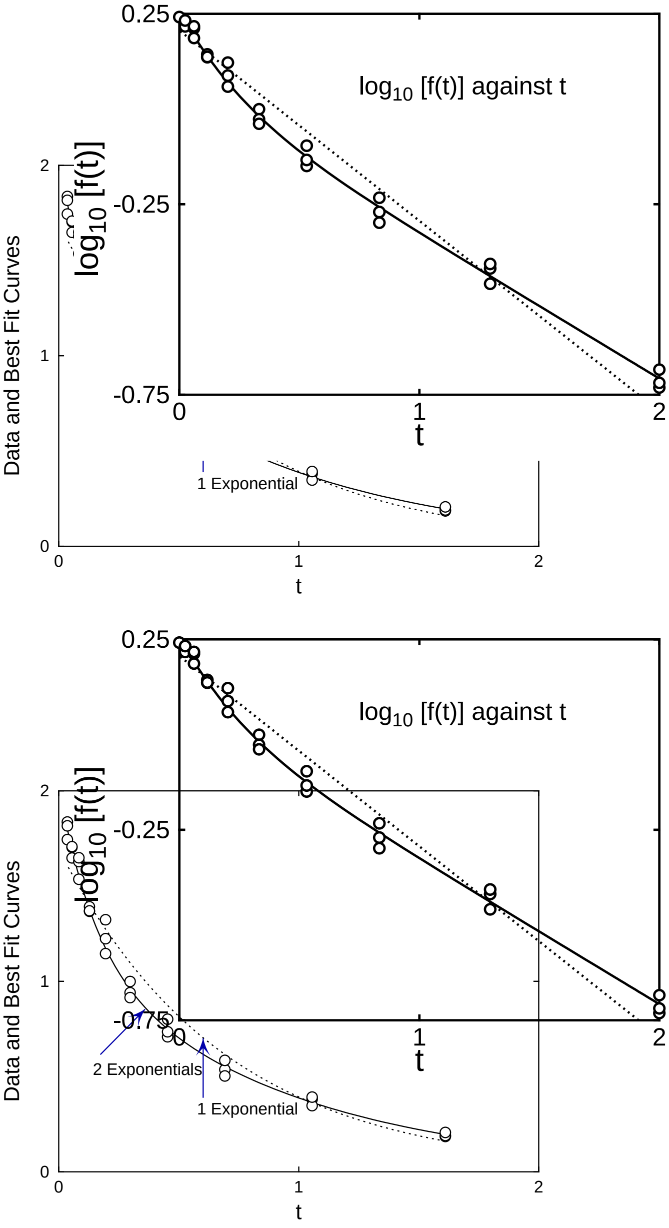


Figure 3: Insets 2: Opaque and transparent backgrounds in insets

Example 4

Sometimes it is useful to add labels to identify sub-graphs in a collage or even to add extra text. Consider the effect of reading test file `editps.tfl` into program **editps** to create Figure 4.

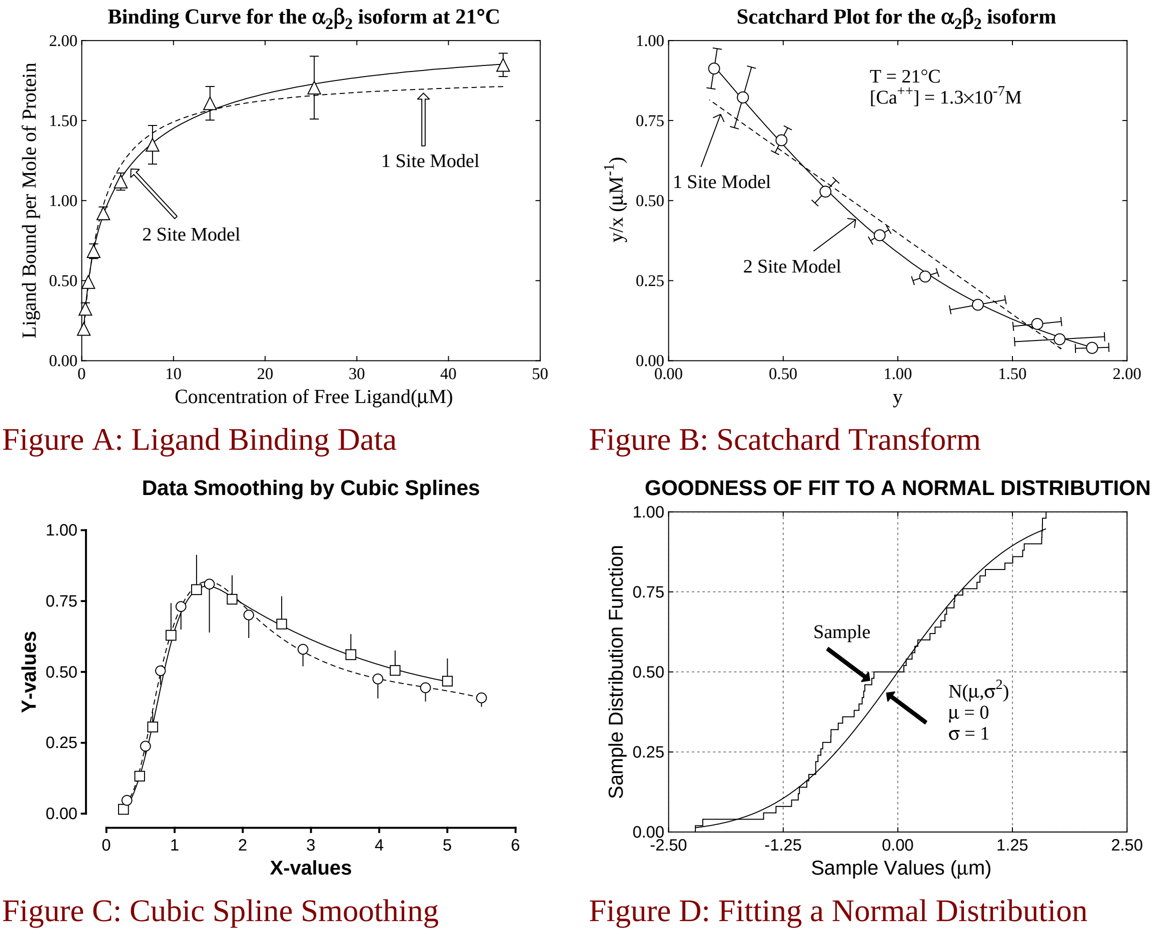


Figure A: Ligand Binding Data

Figure B: Scatchard Transform

Figure C: Cubic Spline Smoothing

Figure D: Fitting a Normal Distribution

Demonstrating Labels and Additional Text

Figure A. Note maths subscripts in α2 β2
Figure B. Note the slanting error bars
Figure C. Note the confidence region indicators with no top or bottom bars
Figure D. Note maths superscript in N(µ, σ2)

Figure 4: Adding labels and extra text

Once the collage has been created there are several options for adding the labels and extra text. Of course, as the composite file is to be created in PostScript then any commands added will have to be in PostScript. As a matter of fact raw Postscript can be added directly to the output file, but SIMFIT also provides an interface to

define labels, titles, and text in any language except that special commands must be used to change fonts and introduce features such as maths, superscripts and subscripts. This feature which is demonstrated in Figure 4 will now be explained.
Once the collage has been assembled and the number of columns chosen, the following options are provided.

1. Defining labels as numbers, letters, or user–defined text.
2. Adding a title
3. Adding a text section

For instance, read in the test file `editps.tfl` into program **editps** then choose two columns, request to edit the text and cut and paste the next lines into the SimFIT editor that has been opened.

```
{newline}
{newline}
{helveticabold}Demonstrating Labels and Additional Text{helvetica}{newline}
{newline}
Figure A. Note maths subscripts in {roman}{alpha}{lower}2{raise}{beta}{lower}2{raise}{newline}
{helvetica}Figure B. Note the slanting error bars{newline}
Figure C. Note the confidence region indicators with no top or bottom bars{newline}
Figure D. Note maths superscript in {roman}N\({mu},{sigma}{raise}2{lower}\)
```

This text will generate the title and additional text displayed in Figure 4. It uses the following commands that will be translated into Postscript.

- `{newline}` new line
- `{helveticabold}` bold font
- `{helvetica}` normal font
- `{roman}` roman font
- `{alpha}` α
- `{lower}` subscript
- `{raise}` superscript
- `{beta}` β
- `{mu}` μ
- `{sigma}` σ
- `{\(}` (
- `{\)}`)

Further details of the options available follows.

Program editps text formatting commands

Program **editps** uses the SimFIT convention for text formatting characters within included SimFIT .eps files but, because this is rather cumbersome, a simplified set of formatting commands is available within **editps** whenever you want to add text, or even create PostScript files containing text only. The idea of these formatting commands is to allow you to introduce superscripts, subscripts, accented letters, maths, dashed lines or plotting symbols into PostScript text files, or into collage titles, captions, or legends, using only ASCII text controls. To use a formatting command you simply introduce the command into the text enclosed in curly brackets as in: {raise}, {lower}, {newline}, and so on. If {anything} is a recognized command then it will be executed when the .eps file is created. Otherwise the literal string argument, i.e. anything, will be printed with no inter-word space. Note that no {commands} add interword spaces, so this provides a mechanism to build up long character strings and also control spacing; use {anything} to print anything with no trailing inter-word space, or use { } to introduce an inter-word space character. To introduce spaces for tabbing, for instance, just use {newline}{   }start-of-tabbing, with the number of spaces required inside the { }. Note that the commands are both spelling and case sensitive, so, for instance, {21}{degree}{C} will indicate the temperature intended, but {21}{degrees}{C} will print as 21degreesC while {21}{Degree}{C} will produce 21DegreeC.

Special text formatting commands, e.g. left

{left} ... use {left} to print a {
{right} ... use {right} to print a }
{%!command} ... use {%!command} to issue command as raw PostScript

The construction {%!command} should only be used if you understand PostScript. It provides PostScript programmers with the power to create special effects. For example {%!1 0 0 setrgbcolor}, will change the font colour to red, and {%!0 0 1 setrgbcolor} will make it blue, while {%!2 setlinewidth} will double line thickness. In fact, with this feature, it is possible to add almost any conceivable textual or graphical objects to an existing .eps file.

Coordinate text formatting commands, e.g. raise

{raise} ... use {raise} to create a superscript or restore after {lower}
{lower} ... use {lower} to create a subscript or restore after {raise}
{increase} ... use {increase} to increase font size by 1 point
{decrease} ... use {decrease} to decrease font size by 1 point
{expand} ... use {expand} to expand inter-line spacing by 1 point
{contract} ... use {contract} to contract inter-line spacing by 1 point

Currency text formatting commands, e.g. dollar

{dollar} $ {sterling} £   {yen} Y

Maths text formatting commands, e.g. divide

{divide} ÷   {multiply} ×   {plusminus} ±

Scientific units text formatting commands, e.g. Angstrom

{Angstrom} Å   {degree} ∘   {micron} μ

Font text formatting commands, e.g. roman

{roman}   {bold}   {italic}   {helvetica}
{helveticabold}   {helveticaoblique}   {symbol}   {zapfchancery}
{zapfdingbats}   {isolatin1}

Note that you can use octal codes to get extra-keyboard characters, and the character selected will depend on whether the StandardEncoding or IOSLatin1Encoding is current. For instance, \ 361 will locate an {ae} character if the StandardEncoding Encoding Vector is current, but it will locate a {ñ} character if the ISOLatin1Encoding Encoding Vector is current, i.e. the command {isolatin1} has been used previously. The command {isolatin1} will install the ISOLatin1Encoding Vector as the current Encoding Vector until it is canceled by any font command, such as {roman}, or by any shortcut command such as {ntilde} or {alpha}. For this reason, {isolatin1} should only be used for characters where shortcuts like {ntilde} are not available.

Poor man's bold text formatting command, e.g. pmb?

The command {pmb?} will use the same technique of overprinting as used by the Knuth TeX macro to render the argument, that is ? in this case, in bold face font, where ? can be a letter or an octal code. This is most useful when printing a boldface character from a font that only exists in standard typeface. For example, {pmbb} will print a boldface letter b in the current font then restore the current font, while {symbol}{pmbb}{roman} will print a boldface beta then restore roman font. Again, {pmb\ 243} will print a boldface pound sign.

Punctuation text formatting commands, e.g. dagger

{dagger} †	{daggerdbl} ‡	{paragraph} ¶	{subsection} §
{questiondown} ¿			

Letters and accents text formatting commands, e.g. Aacute

{Aacute} Á	{agrave} à	{aacute} á	{acircumflex} â
{atilde} ã	{adieresis} ä	{aring} å	{ae} æ
{ccedilla} ç	{egrave} è	{eacute} é	{ecircumflex} ê
{edieresis} ë	{igrave} ì	{iacute} í	{icircumflex} î
{idieresis} ï	{ntilde} ñ	{ograve} ò	{oacute} ó
{ocircumflex} ô	{otilde} õ	{odieresis} ö	{ugrave} ù
{uacute} ú	{ucircumflex} û	{udieresis} ü	

All the other special letters can be printed using {isolatin1} (say just once at the start of the text) then using the octal codes, for instance {isolatin1}{\ 303} will print an upper case ntilde.

Greek text formatting commands, e.g. alpha

{alpha} α	{beta} β	{chi} χ	{delta} δ
{epsilon} ϵ	{phi} ϕ	{gamma} γ	{eta} η
{kappa} κ	{lambda} λ	{mu} μ	{nu} ν
{pi} π	{theta} θ	{rho} ρ	{sigma} σ
{tau} τ	{omega} ω	{psi} ψ	

All the other characters in the Symbol font can be printed by installing Symbol font, supplying the octal code, then restoring the font, as in {symbol}{\ 245}{roman} which will print infinity, then restore Times Roman font.

Line and Symbol text formatting commands, e.g. ce

{li} = line
{da} = dashed line
{do} = dotted line

{dd} = dashed dotted line
{ce}, {ch}, {cf} = circle (empty, half filled, filled)
{te}, {th}, {tf} = triangle (empty, half filled, filled)
{se}, {sh}, {sf} = square (empty, half filled, filled)
{de}, {dh}, {df} = diamond (empty, half filled, filled)

These line and symbol formatting commands can be used to add information panels to legends, titles, etc. to identify plotting symbols.

Examples of text formatting commands

{TGF}{beta}{lower}{1}{raise} is involved
TGFβ_1 is involved

y = {x}{raise}{2}{lower} + 2
$y = x^2 + 2$

The temperature was {21}{degree}{C}
The temperature was 21°C

{pi}{r}{raise}{decrease}{2}{increase}{lower} is the area of a circle
πr^2 is the area of a circle

The {alpha}{lower}{2}{raise}{beta}{lower}{2}{raise} isoform
The $\alpha_2\beta_2$ isoform

{[Ca}{raise}{decrease}{++}{increase}{lower}{]} = {2}{mu}{M}
$[Ca^{++}] = 2\mu M$