ROBOT
- a graph plotting program and data analysis tool
Version X4.91

Robin H.D. Corbet

December 12, 1997

Laboratory for High Energy Astrophysics, Code 662
NASA/Goddard Space Flight Center, Greenbelt MD 20771

corbet@lheamail.gsfc.nasa.gov

Contents

1 Overview

Used in the simplest way "Robot" will produce good looking graphs from a data file with the minimum of user interaction. However, by using other commands more sophisticated plots may be produced. A number of data analysis utilities are also included, these include things such as fitting functions, folding data, period searching, and data manipulation (functions and arithmetic). Different text fonts, styles, line thicknesses, and colors can be used. Examples of the uses of Robot can be found in the "Demos" directory. The program stores commands you give it and these may be saved in a user specified file (the default name is robotlog.rob) and can store results (e.g. coefficients of fitted polynomials) in a file with the default name of robot.inf.

The resulting plot can be written as a PostScript file for subsequent printing on a laser printer, emailing to a friend etc. The quality of this plot is naturally better than is displayed on the screen. It includes the color information and the output can be either formatted for printing in landscape or portrait mode.

Robot is written in a mixture of 'C' and FORTRAN and presently uses the XView toolkit for its graphical user interface. The feasibility of replacing this with Java is being investigated. The presently maintained version runs on Sun workstations. LINUX systems, and Alphas. Installation instructions are found in the "README" file distributed with the source code.

If you wish to reference Robot you may cite
Corbet, R.H.D., Larkin, C., & Nousek, J.A., Proceedings of "Astronomical Data Analysis and Software Systems", Tucson, November, 1991. Astronomical Society of the Pacific Conference Series, 1992, Vol. 25, 106. (ISBN 0-937707-44-9)

2 Help

When using ROBOT you can get on-screen help by pressing the "Help" button while the cursor is positioned over a button or menu item. For this to work the file robot.info must be installed in the correct directory. This can be either the directory you are running ROBOT from or a directory specified by the environment variable HELPPATH. For example /usr/lib/help.

3 Getting Started

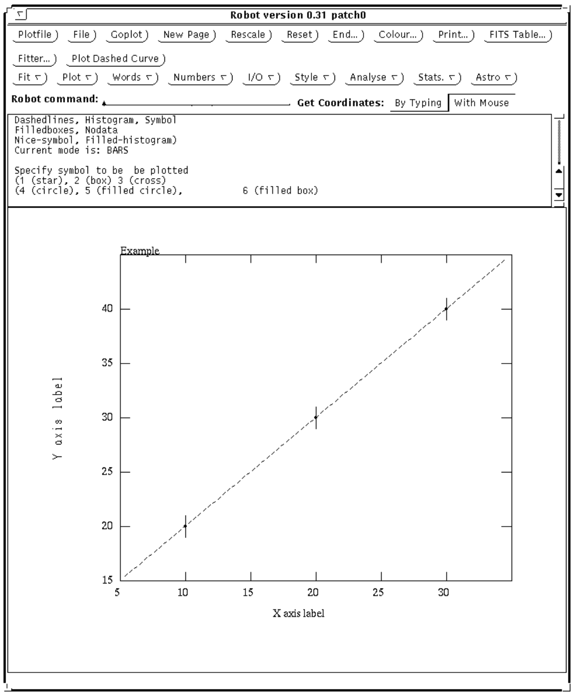
To run a demonstration script press the button marked "File..." and select the demo. script you want to run. (This requires that you copied the entire Robot distribution including the demonstration scripts).

To use the program in its simplest mode the commands:

```
DATAFILE
                 filename
GOPLOT
```

are required, where "filename" is the name of your data file. The basic format of the input file is:

```
Title of graph
```



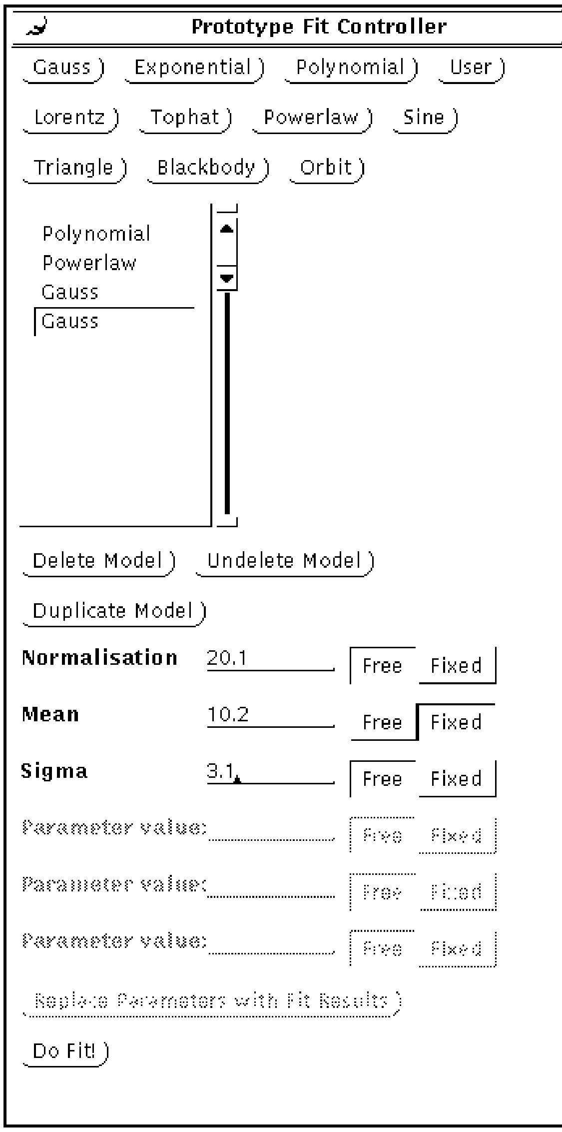


Figure 1: The main Robot control and display areas (left). The fit control panel (right).

```
x-axis title
y-axis title
x1, y1, error on y1, error on x1
             ....
xn, yn, error on yn, error on xn.
```

You need not use all columns and if your columns are not in this order you can tell ROBOT about this using the "TABLE" command (see next section). Data can be integer or real (but real*8 may give problems). Values may be separated using spaces, tabs and/or commas.

If the GOPLOT command is given ROBOT automatically plots the points using error bars, adds axes with labels and adds the titles for the axes specified in the input file. To tailor the graph to your specific requirements you can add extra commands.

The program makes calls a plotting package based on one called "ARK" which in turn is based on the MSSL plotting package (described in separate documentation). I have also made some use of algorithms based on Bevington and "Numerical Recipes" routines.

It is easy to add new commands to ROBOT so the author will welcome any suggestions for additions. At the end of this document are several examples of the use of ROBOT.

If you publish a paper which makes use of ROBOT I'd very much appreciate it if you could send me a reprint of your paper.

4 Commands currently recognized by ROBOT

Some commands need arguments which are entered on the following line or sometimes on the next two lines. For one line arguments these may now be placed on the same line as the command providing you use the "terse" form (e.g. "polfit 3" is acceptable but not "polynomial fit 3"). For a two or more line argument the first argument only may be specified on the command line. For several commands the graphics cursor can be used if you hit return in reply to the request for an argument (e.g. to position text). You should then click on the left hand mouse button in the desired position. If an abbreviation can be used instead of the full command this is given in brackets below. The menu items displayed on the screen will typically be rather less terse then are shown below. However, the command names written below are what is actually written to the output log file. In general, either the long or the short forms should be recognized by ROBOT. Some commands also have a couple of "aliases" (e.g. to allow for "British" and U.S. spellings) A number of the buttons on the main control panel cause secondary panels to be popped up. At the moment these are for controlling the color, printing, fitting and reading FITS binary table ("FITS 3D") files.

4.1 The Basic Commands

DATA/DATAFILE/PLOTFILE (P) - specifies data file to be plotted, sets up default plotting limits, axis intervals etc. Must be followed by: input file name. If the name given can't be found then ROBOT will also try adding a ".dat" to the end.
GOPLOT (G) - tells ROBOT to plot data.
END (E) - terminates the program.
FILE (F) - read commands from specified file instead of direct user interaction. Argument: file

name. Note that files can contain the FILE command themselves. This gives a limited subroutine like ability. If the file name specified can't be found as given, ROBOT will try adding a ".rob" to the name.
RETURN - return to reading ROBOT commands from the default input instead of the current file.
You may add a comment to your ROBOT instructions by typing # or ! before your comment after a ROBOT command (see examples). Comments may, in general, be added to arguments.

4.2 The Input Data File

Although ROBOT can read up to 4 columns (x, y, y error, x error) of input you need not use all columns. E.g. you can just have a file with x and y points and the errors will be taken as 0. If you have a file with only one column this will be taken as y values and the x values will be set to the sequence numbers of the data. If you want ROBOT to read different columns as the x, y values etc. use the command:
TABLE - the argument for this is four numbers which give the column numbers that the x, y, y error and x error values are found in. The default is 1 2 3 4. Note that using this command you can select up to four columns from an input file which contains more than four columns.
INFILENOTEXT - specifies that the input file doesn't have the three lines of text at the top.
INFILETEXT - specifies that the input file does have three lines of text header (default).
INFILENOTEXT and INFILETEXT must be used before DATAFILE.
OVERPLOT - read in new data file, similar to DATAFILE except that the plotting area etc. are not redefined.
TYPE - means that ROBOT will directly read data - i.e. from the terminal for interactive use. This command can also be used to include data directly in a command file.
Input format is the same as that for DATAFILE.
OVERTYPE - as TYPE but doesn't rescale the plotting area.
TYPEMORE/MORETYPE - like TYPE but reads in additional data and adds to existing data set.
PLOTMORE/MOREPLOT - like DATAFILE but reads in additional data and adds to existing data set.
FITS - reads a cross section of a FITS image. Argument: rows to add up to produce the cross section.
FITSX - as FITS but sums across columns.

4.3 General Plotting Commands

PLOTMODE - how data points are to be plotted, must be followed by either: BARS, BARS2, BOXES, FILLEDBOXES, CROSSES, DIAMONDS, ELLIPSES, LINES, HISTOGRAM, FILLED-HISTOGRAM, FILL, BAR-GRAPH, STACKED-BAR, PILLAR, SPLINE, DASHEDLINES, SYMBOL, NICE-SYMBOL or NODATA,. These specify either error bars (the default), errors bars with horizontal bars at the top and bottom of the vertical bar rather than at the middle, error boxes, error crosses, error diamonds, error ellipses, connect points with lines, plot using a histogram, connect points with dashed lines. SYMBOL plots the requested symbol very nearly centered on the data points (centering in the "Y" direction is not yet guaranteed). SYMBOL also requires the character string which is to be plotted as an argument. NICE-SYMBOL plots symbols such as filled boxes, open circles etc. precisely centered on the data value. Argument is the type of symbol to draw (star, box, cross, circle, filled-circle, or filled-box). NODATA means that the no plotting of the data is done. Useful if you just want to plot the axes.

SYMBOLSIZE changes the size of symbols plotted with the "NICE-SYMBOL" option of PLOT-MODE.

SPLINE - plots a cubic spline fit to your data. (Same effect as using the SPLINE option for PLOT-MODE).

TITLE - overwrites the graph title to be used. Must be followed by: graph title

LEFTTITLE - specifies that the graph title will be left justified. (This is the default)

CENTERTITLE/CENTRETITLE - specifies that the graph title will be centered.

RIGHTTITLE - specifies that the graph title will be right justified.

XLABEL - overwrites the x-axis title to be used. Must be followed by: new x-axis title

YLABEL - overwrites the y-axis title to be used. Must be followed by: y-axis title

TITLE2, TITLE3, TITLE4, XLABEL2, XLABEL3, XLABEL4, and YLABEL2 are extra positions for user defined titles (see example). Use like TITLE etc. For label locations run the file title_demo.rob in the Demos directory.

TEXT - add text to the graph, followed by 1) text, and then 2) the x and y coordinates of where the text string will start from. See the section on "Text Strings" for more details.

CENTRETEXT/CENTERTEXT - add text to the graph. The difference between this and the TEXT command is that CENTRETEXT plots the text centered on the X coordinate. Arguments: 1) text string, 2) x and y coordinates.

RIGHTTEXT - add text to the graph. The difference between this and the TEXT command is that RIGHTTEXT plots text right-justified to the specified X coordinate. Arguments: 1) text string, 2) x and y coordinates.

TEXTSIZE - gives size of characters, argument is: size of character in "points".

TENPOINT, TWELVEPOINT, FOURTEENPOINT, EIGHTEENPOINT - use text size as specified (units are "points").

WIDTH0, WIDTH1, WIDTH2, WIDTH4, WIDTH8 - change the width of lines. The default width is 0. This is supposed to give faster plotting under X Windows. On high resolution postscript devices it is possible that this might give some problems.

SOLIDLINES, DASHEDLINES, DOTTEDLINES, DOTDASHEDLINES - specify different line styles for drawing. The default is to use solid lines.

NORMAL - use normal font style. i.e. neither bold nor italic.

BOLD - use bold fonts. ITALIC - use an Italic (for Times-Roman) or oblique (for Helvetica or Courier) style fonts.

BOLDITALIC - use bold italic/oblique fonts.

TIMES - use the Times-Roman family of fonts.

HELVETICA - use the Helvetica family of fonts. This is a sans serif font.

COURIER - use the Courier font family. This is a mono-spaced (non-proportional) font and will probably look ugly for most applications.

TANGLE - angle at which text is to be plotted, argument is angle in degrees. The default is 0 degrees.

DATALIMITS - user defined constraints on minimum and maximum data values to be plotted (The DFLIMITS command described later is more general) DATALIMITS is useful if you want to zoom in on a region of interest. Argument: minimum x, minimum y, maximum x, maximum y.
PAGEUSE - specifies percentage fraction of plotting area to be used. Useful for constructing multi-panel graphs. Argument is: coordinates bottom left hand corner and top right hand corner in percentages. Default is: bottom left hand corner is 20.,20., top right corner is 90.,90.
LINE - draw a line. Argument: start coordinates and end coordinates.
DASHEDLINE - draw a dashed line. Arguments: 1) start and end coordinates, 2) no. of dashes, ratio of space to dash size.
MOVETO - set the current plot position without drawing a line. Argument: coordinates of position.
LINETO - draw a line from the present plot position to the specified point and set the present plot position to that point too. Argument: coordinates of position.
POLYLINE - draw a series of connected lines. Arguments: coordinates of points to be connected.
POLYGON - draw a series of connected lines with the last point connected to the first point. Arguments: coordinates of points to be connected.
POLYFILL - draw a filled polygon. Uses the “odd even” filling rule. Arguments: coordinates of points defining the polygon. BOX - draw a box. Argument: coordinates of bottom left hand corner and top right hand corner.
FILLEDBOX - draw a filled box. Argument: coordinates of bottom left hand corner and top right hand corner.
CIRCLE - draw a circle. Argument: coordinates of center and radius in units of the X axis.
FILLEDCIRCLE - draw a filled circle. Argument: coordinates of center and radius in units of the X axis.
ELLIPSE - draw an ellipse. Argument: coordinates of the center and a corner of a bounding box.
ARC - draw a circular arc. Argument: center, radius (X axis units), start angle and end angle. Angles are measured in degrees. The order in which the angles are given doesn’t matter. e.g. 180, 90 is the same as 90, 180. To draw the “other” part of the circle use e.g 90, -180.
ARROW - draw an arrow. Argument: start and end coordinates.
SINGLEARROW - arrow heads on one end of arrow (default).
DOUBLEARROW - arrow heads on both ends of arrow.
ARROWLINE - draw arrow head using two lines (default).
ARROWHOLLOW - draw hollow arrow head.
ARROWFILL - draw filled arrow head.
ARROWSIZE - specify size of arrow head. Argument: size of arrow head.
ARROWANGLE - specify shape of arrow head. Argument: opening angle of arrow head in degrees.
TITLES - plot titles to graph and axes (default).
NOTITLES - don’t put titles on graph and axes.
AXES - draw axes (default).
NOAXES - don’t draw axes.
LABELAXES - label axis at tick marks (default).
NOLABELAXES - don’t label axis tick marks.
LABELXAXIS - label the x-axis at tick marks (default).
NOLABELXAXIS - don’t label the x-axis at tick marks.

LABELYAXIS - label the y-axis at tick marks (default).
NOLABELYAXIS - don't label the y-axis at tick marks.
NTICKX - changes the approximate number of tick marks to use in the x-direction. The default is to use about 10 (the program adjusts the exact number to give good looking plots). Argument: approximate number of tick marks for the x direction.
NTICKY - as NTICKX but for the y direction. Argument: approximate number of tick marks in the y direction.
MINORAXES - also minor tick marks for both axes.
NOMINORAXES - don't draw any minor tick marks (default).
LOGAXES - Axes are to be logarithmic. Note, this doesn't do anything to the data, it is assumed that your data already is log values.
LINEARAXES - Use normal non-logarithmic axes (default).
LOGXAXIS - the x axis is log.
LINEARXAXIS - the x axis is linear.
LOGYAXIS - the y axis is log.
LINEARYAXIS - the y axis is linear.
USERAXIS - draw a user defined axis. Argument: id, ticksize, start x coordinate, start y coordinate, distance between tick marks, number of tick mark intervals (integer). id is an integer defining axis direction, 0 is x direction, 1 is y direction. To get tick marks to go on the opposite side of the axis make tick size a negative number.
USERAXISLABEL - user defined axis labels. Arguments, 1): start value for labels, increment value. 2) id, start x coordinate, start y coordinate, interval between labels, number (integer) of labels. id is as in USERAXIS above.
LOGAXIS - as for USERAXIS except that a logarithmic axis will be used.
LOGAXISLABEL - as for USERAXISLABEL except labels will be written as 10^x.
COLOR/COLOUR/PENCOLOUR - This alters the color used to plot your graph or portions of it. Argument: one of either white, blue, green, yellow, red, magenta, cyan or black.
TOPAXIS - draw an axis at the top of the plot .
NOTOPAXIS - don't draw an axis at the top of the plot.
BOTTOMAXIS - draw an axis at the bottom of the plot.
NOBOTTOMAXIS - don't draw an axis at the bottom of the plot.
RIGHTAXIS - draw an axis at the right hand side of the plot.
NORIGHTAXIS - don't draw an axis at the right hand side of the plot.
LEFTAXIS - draw an axis at the left hand side of the plot.
NOLEFTAXIS - don't draw an axis at the left hand side of the plot.
ALLAXES - draw all of the above axes (default). XTICKSIZE - specify the size of the tick marks used for the X axis. Units are the dimensions of the Y axis.
YTICKSIZE - specify the size of the tick marks used for the X axis. Units are the dimensions of the Y axis.
DEFAULTTICKSIZE - let Robot calculate tick sizes to be used (default).
EXTERNALTICKS - plot external ticks marks.
INTERNALTICKS - plot internal tick marks (default).
EXTERNALXTICKS - plot external tick marks on the X axis.

INTERNALXTICKS - plot internal tick marks on the X axis.
EXTERNALYTICKS - plot external tick marks on the Y axis.
INTERNALYTICKS - plot internal tick marks on the Y axis.
MINORTICKSIZE - specify the size of the minor tick marks. Argument is the ratio of minor tick mark size to major tick mark size.
MINORTICKFREQUENCY - specify the frequency of minor tick marks. Argument: approximate ratio of minor tick mark frequency to major tick mark frequency.
RGB - Specify the color using intensities for red, green and blue. Argument: R, G and B values. These must be scaled between 1 and 256.
RGBDATA - only specify the color used for plotting the data itself. Axes, text, and anything else is left in the current plot color.
RGBREST - specify the color for plotting everything apart from the data. The color used to plot the data is left in the current plot color.
TILE - select one of the predefined patterns for tiling. A value of 0 means don't tile but fill with a solid color.
NOTILE - stop tiling. The same as using a tile value of "0".
RESET - resets everything to default values, i.e. plot data as error bars, plot axes with labels, add titles to graphs, use from 20-90% of plot dimensions in both x and y directions, don't divide into sub-windows, use standard pen color/thickness, use approximately 10 tick mark intervals in x and y directions.
NXTPAG (N) - start a new page.
DIVWINDOW - divide the plotting area into several sub-windows. Argument: no. of windows in x-direction, no. of windows in y direction.
SELWINDOW - select a subwindow. Argument: no. of sub-window in x and y directions.
WINDOWGAP - specify spacing between sub-windows. Argument: percentage gap between windows. Default is to have no gap. XWINDOWGAP - as WINDOWGAP but specifies gap in just X direction.
YWINDOWGAP - as WINDOWGAP but specifies gap in just Y direction.
VALUES/SHOW - causes printing of values of various ROBOT parameters, e.g. data limits, plotting limits, version no. of ROBOT being used.
ECHO - causes extra output to be written to the screen (deprecated).
NOECHO - reduces output written to screen (deprecated).
CURSOR - calls up the graphics cursor. When RETURN is pressed the coordinates are stored as the variables XCURSOR and YCURSOR which can be used as parameters for the arguments to other commands (e.g. the arithmetic ones).
RULER - use a cursor to measure X and Y distances between points on a graph.

4.4 3D Data

Although ROBOT is mainly designed for two dimensional data some start has been made on dealing with data sets which consist of (x, y, z) values. It uses what is nominally the y error array as the z values. The addition of three D data has lead to some inconsistencies in Robot which may be changed at some date. One contradiction is that the Robot presently uses an array named "Z" as a temporary storage array. Sometime in the future this may be renamed and commands such as "ZARITH" will then work on the real Z (a.k.a. y error) array.

3D - tell Robot to do a "three dimensional" plot instead of the normal X/Y plot. The argument is the three rotation angles of the coordinate system.
2D - go back to regular two dimensional plotting.
3DCOLOURCODE/3DCOLORCODE - the color used to plot a data point will depend on the Z value. Arguments: 1) Color of minimum (RGB triplet), 2) color of maximum (RGB triplet).
3DNOCOLOURCODE/3DNOCOLORCODE - switch off color coding based on Z value.
3DFILL - specifies that the 3D plot will fill as much of the plot area as possible.
3DNOFILL - only one quarter of the possible plot area will be used for 3D plots. The corner of the plot is thus anchored at the corner. This is useful for rotating a data set and is the default.
CONTOUR - The x and y arrays store the grid coordinates. The grid must be evenly spaced. The default is to use ten contour levels evenly spaced between the maximum and the minimum data values. To change the contour levels use:
CONTOURVALUES - Argument: the values of the contour levels to be used. To go back to the default contouring give a blank line as the argument.

4.5 Data Analysis Commands

The data analysis commands which fold data or subtract a fitted function result in the plot area being rescaled. Folding data overwrites the internal ROBOT data arrays. Arithmetic operations and functions of the x and y arrays don't rescale the plot area. To rescale the area manually use the command RESCALE.
SORT - sorts the data into ascending x values. Sorts x , y and associated error arrays but not z array.
SORTY - as SORT but sorts by y value.
SMOOTH - smooth data by convolving with a triangle. Replaces each point with half its original value plus a quarter of the points on either side.
SMOOTHN - as smooth but repeat the smoothing several times. Argument: no. of times to smooth data.
REBIN - sum together bins to create new arrays. Argument: no. of old bins to make one new one. The SMOOTH and REBIN commands are obviously best suited to evenly spaced data with no gaps.
LOGX - take log10 of x values.
LOGY - take log10 of y values.
LOGXY - take log10 of both x and y values.
XFITLIMITS - restricts the data range to be used when making fits to the data. Argument: low x, high x fitting limits.
YFITLIMITS - restrict fitting limits based on y values. Argument: low y, high y fitting limits.
XYFITLIMITS - restrict fitting limits using both x and y values. Argument: low x, low y, high x, high y.
DFLIMITS - restrict both plotting area and fit limits. Equivalent to a combination of DATALIMITS

and XYFITLIMITS. Argument: low x, low y, high x, high y. DFLIMITS is the most frequently used method of selecting a portion of data for display/analysis. ROBOT will try and adjust your plotting limits to make them "nicer". If you don't want this enclose fixed values in (brackets). This method is also used to obtain "upside-down" plots. e.g. for plotting astronomical magnitudes.
MOMENTS - calculate sum, mean, standard deviation, standard error, skewness and kurtosis of x and y arrays.
MEDIAN - median value of y array.
MEDIANX - median value of x array.
SUMDATA - calculate sum of y values in the range specified by e.g. XFITLIMITS.
INTEGRATE - integrate the area under the curve defined by the data points.
XARITH - perform arithmetic on the x values. Arguments: 1) arithmetic operation, i.e. +, -, /, * or **; 2) operand; either a number or x, y, xerr, yerr or z to operate using the numbers in these arrays. N.B. only the * and / operations also change the error arrays if an operation is made on a data array.
YARITH, ZARITH, XERRARITH, YERRARITH - as XARITH but on the y, z, x error and y error arrays respectively.
XFUNCTION - replace the x array with a function of x. Argument: the type of function; e.g. SQUARE, SQRT, CUBERT, INVERSE, FACTORIAL, SINE, COS, TAN, ASINE, ACOS, ATAN, LOG (base 10), LOGE and LOG2 (base 2). The trigonometric functions all assume degrees.
YFUNCTION, ZFUNCTION - as XFUNCTION but for the y and z arrays respectively.
ZRESET - sets the z array to zero.
LINFIT - make a straight line fit to the data and calculate the linear correlation coefficient and its significance.
PLOTFIT - plot the straight line fit.
KENDALL - non-parametric test of correlation between x and y values.
POLFIT - fit a polynomial to the data. Argument: order of the polynomial to be fitted.
PLOTPOLY - plot the results of the polynomial fit.
SUBPOLY - subtract the polynomial fit from the data.
FOLD - fold the data on specified period. Argument: period and phase zero.
BINFOLD - fold the data into bins. Also calculates the χ^2 value for the hypothesis of source constancy. Argument: period, phase zero and the no. of bins to fold the data into.
WBINFOLD - Same as BINFOLD except that the errors on the data points are used to calculate weighted mean and standard deviations for each bin.
FINDPERIOD - period search using the BINFOLD routine. Argument: start period, end period, period step and the number of bins to fold the data into.
BINFOLDBEST - fold and bin the data on the best period found by FINDPERIOD.
FOLDBEST - fold the data on the best period found by FINDPERIOD but don't bin it.
PERIODOGRAM - like FINDPERIOD but plots the periodogram. Argument: as in FINDPERIOD. N.B. you can also search for periods by fitting sine waves to your data and looking for χ^2 minima. Use GRID followed by CURFIT and grid the period.
POWER - Calculates power spectrum of data using a Fourier transform. The power spectrum is plotted against frequency. The strongest peak is found and the "FOLDBEST" routine will take the corresponding period automatically. If "OVERWRITE" is true then the data is overwritten with

the power spectrum. If “WEIGHTFT” is set then data points are weighted by the inverse square of their errors.
FFT - for evenly spaced data the Fast Fourier Transform can be used. If the number of data points is not an integer power of two the default is to truncate the data. Alternatively the data can be padded with the mean of the data.
FFTMODE - takes a value of either TRUNCATE (the default), PAD, or NEAREST. NEAREST uses the logarithmically closest value to either pad or truncate.
OVERWRITE - enables the FFT and POWER routines to overwrite the data with the power spectrum. By default this is FALSE.
NOOVERWRITE - stops the FFT and POWER routines overwriting data. This is the default.
WEIGHTFT - For the POWER routine *only* the power spectrum is weighted by the inverse square of each data point. This is *not* used by the FFT routine. While this appears to be formally correct, it should be used with caution. Sometimes it appears to improve the results, other times it doesn't.
NOWEIGHTFT - don't weight points for the POWER routine. This is the default.
RANDOMISE/RANDOMIZE - replaces y values with Gaussian distributed random numbers having the same mean and standard deviation as the original data set. If MOMENTS has not already been called it calls this to evaluate the mean and standard deviation of the current data.
GRANDOMISE/GRANDOMIZE - same as the regular randomize command except that, instead of taking the standard deviation of the input data, the standard deviation is the square root of the mean of the input data.
FITMODE - determines the weighting of data points in the fit. Argument: 1 for weighting by errors in the y error array, 0 for no weighting and -1 for weighting by inverse of y array (e.g. Poisson errors). Default is weight by errors in y error array.
CURFIT - generalized fitting routine using the famous Bevington routine. You can add many models together. The following models are currently supported: POLYNOMIAL, TRIANGLE, LORENTZ, GAUSS, POWERLAW, EXPONENTIAL, SINE, BLACKBODY, TOPHAT, ORBIT and USER. End of models is signaled by ENDMODELS. After each function give initial guesses for each parameter. Polynomial: initial guesses for coefficients (all on one line). The order of the polynomial is automatically determined from the number of parameters. Triangle: height, start of function, half width at zero. Lorentz: normalization, mean and width. Gauss: normalization, mean and width. Powerlaw: normalization and power. Exponential: normalization, start time and decay time. Sine: amplitude, phase and period. Blackbody: normalization and kT in units of the x axis. Top hat: height, start of function and width. Orbit: amplitude, phase period, mean value, eccentricity and omega. The TOPHAT function is almost impossible to fit without CURFIT crashing! It is very easy to add new models so please tell me if there are any others you want included. Currently up to 10 models can be added together. The same model type may be specified as many times as you want (e.g. 5 Gaussians plus a polynomial is O.K.).

To write your own model to fit with CURFIT you need to write a FORTRAN subroutine called USER. An example of the type of function you can write is:

```
C Function User.
C An example of a user supplied function for use with the "CURFIT" command in
C ROBOT. This routine is for a straight line fit; much more complicated
C functions can be used as long as the basic format is retained.
```

```
FUNCTION USER(A,X)
REAL A(*), X
USER = A(1)+A(2)*X
END
```

The equivalent in C is:

```
float user_(a, x)
float *a, *x;
{
float ret_value;
--a;
ret_value = a[1] + a[2]* *x;
return ret_value;
}
```

For the X windows and Fujitsu versions you will need to relink ROBOT, for the Mac program you just need to compile your subroutine as the Mac FORTRAN compiler can link your subroutine during run time order that your subroutine uses them. You can combine your model together with the other models already available with ROBOT.

If you want to fix a parameter in the list of input parameters to any model enclose that parameter in (brackets). Any number of parameters (including all of them) may be fixed. If you want to produce a grid of χ^2 square against the value of a particular parameter enclose it in <this type of bracket>. The default is to produce the grid using values of the parameter ranging from + to - 3 times the error found by the curfit (before doing the grid a fit is done with the parameter free) with steps of one fifth the supposed error. If you don't want to use these defaults use the command:

GRID - to specify the grid range. Argument: start value for grid, end value, step size. In this case no fit with the parameter free is done. Currently only one parameter at a time can be gridded.

PLOTCUR - plots the fitted function.

SUBCUR - subtracts the fitted function from the data.

ADDCUR - adds the fitted function to the data.

CURMODELPLOT - plots individual fitted models. Argument: either ALL for all models, a single number (e.g. 2) to just plot that model number or e.g. 3:6 to plot individual model numbers 3 to 6. The model number is its sequence in the list you gave to CURFIT.

CONVERGENCE - changes convergence conditions. Argument: fractional change of χ^2 at which to stop and maximum number of times to call CURFIT. Default is fractional change in $\chi^2 = 0.01$, call CURFIT 10 times.

FITDELTA - changes step values used in calculating derivatives of parameters of fit. The steps are parameter value divided by the FITDELTA value.

PREDICT - once a fit has been made "PREDICT" gives the expected "y" value from the fit for a given "x" value.

WRITEDATA - write data to specified file. Argument: output file name. Only data inside the region defined for plotting (e.g. via DFLIMITS) is written to the output file.

CCF - calculates the cross correlation coefficient of the data already inside ROBOT with a specified input file as a function of shift in X. This input file must have a "DATAFILE" format with three lines of text at the top. This command can be used for doing pulse arrival time analysis where you fold data using the BINFOLD command and then use CCF to get the relative phase vs. a template. The CCF is calculated simply by shifting bins and multiplying, however it centroids the cross-correlation function and so can give results more accurate than one bin width.

4.6 Astronomical Stuff

RADECGRID - plot a sky grid suitable for plotting RA and declination.
RADECPLOT - plots points on the screen. Data should be in decimal degrees.
RADECTOGAL - converts data from RA and declination to galactic longitude and latitude.
GALTORADEC - converts data from galactic longitude and latitude to RA and declination.
FITS - extracts a cross section from a FITS image. Argument is the start and end columns to use.

4.7 Miscellaneous

$\uparrow\uparrow$ - repeat the previous command. Alternatively from the "Robot Command" line (*not* within a file) you can use "!!". Also, from the "Robot Command" line a command entered starting with a "!" or a "$\uparrow$" will attempt to match the command with a previously issued one similar to the way the 'C' shell works. From this line the previous command can also be edited using a sequence like "$\uparrow$old_string$\uparrow$new_string$\uparrow$" - this is again similar to the 'C' shell. Note that $\uparrow\uparrow$ differentiates between commands from a file and commands entered interactively. If you have $\uparrow\uparrow$ in a file then the previous non-interactive command is repeated. If $\uparrow\uparrow$ is entered interactively then the previous interactive command is repeated (e.g. if you execute a ".rob" file then the command repeated will be "file" and *not* the last command in the ".rob" file.
FLUSH - flushes graphics output.
WAIT - wait for a while. Argument is how long to wait. This is implemented by doing a calculation a certain number of times. The length of time waited thus depends on machine speed.
PAUSE - pop up a notice and halt execution until the button on the notice is pressed.
ALIAS - alias a command to another string. Arguments: 1) name of alias to use (no spaces), 2) what the command is to be aliased to.
UNALIAS - remove an alias. Argument: name of alias.
NOALIASES - remove all aliases.
CD - change working directory. Argument: directory to change to.
ARRAYSIZE - change the size of the data arrays inside Robot. i.e. this affects how big a data file you can use. Argument: new data array size. Arraysize can also be specified as a command line argument. (e.g. robot -arraysize 500)
PLOTSCALE - scales the plotting area from its "normal" size. e.g. if you have a small screen this can be useful. This can be specified as a command line argument. For example, invoke robot with "robot -plotscale 0.8 -scale small". The "scale" argument is a standard OpenWindows argument which will make the control panels smaller. Using a plotscale $\neq 1$ will make the display non-WYSIWYG as fonts are not reduced in proportion to the new scale.
FLIP/FLIPPAGE - alternates graphs between landscape and portrait orientations.

4.8 Text Strings

When using the TEXT command to plot text the input text string can contain various codes to change the font type and size within the string, and also to give you access to additional characters. These "control" sequences can also be used with any other commands which takes text input such as TITLE.

The sequences presently recognized are:
\sf Symbol Font
\if Italic Font
\bf Bold Font
\bi Bold Italic
\tf Times Font (default)
\cf Courier Font
\hf Helvetica Font
\nf normal Font (not bold or italic)
\up move up to a position 1/2 the font size higher (superscript)
\dn move down
\bs move back one character size (approx.)
\red change color to red. (Also green, blue, white, black)
Color changes are only applied to the text within the string being plotted. After the text is plotted the current drawing color reverts back to that previously being used.
A text size in points can be given such as
\08
\10
\12
\18 etc.

To decrease or increase the font size by two points use the commands \- and \+ respectively.

In order to obtain "special" characters use codes such as \{infinity}. This pattern can also be used to generate Greek characters. e.g. \{alpha} to get α or \{Gamma} to get Γ.

"Accented" characters can be accessed using two schemes. One is based on TEX usage. e.g. `'o`, (ó); `^o`, (ô); `"o`, (ö); `` `o ``, (ò); `~o`, (õ). Alternatively more explicit but longer names can be used: oacute, (ó); ocircumflex, (ô); odieresis, (ö); ograve, (ò); otilde, (õ). Note that only a limited number of characters have "accented" versions available.

Other character names are patterned after those used in TEX in general with some additional duplicate names (typically those used in PostScript). These include: -, –; ?, ¿; !, ¡; ae, æ; AE, Æ; aleph, $\aleph$; angle, $\angle$; approx, $\approx$; aring, å; aa, å; Angstrom, Å; Aring, Å; AA, Å; ast, $*$; asterisk, $*$; backslash, \; bullet, $\bullet$; bot, $\perp$; cap, $\cap$; circlemultiply, $\otimes$; circleplus, $\oplus$; club, $\clubsuit$; clubsuit, $\clubsuit$; copyright, ©; congruent, $\cong$; cong, $\cong$; cup, $\cup$; degree, °, diamond, $\diamond$; diamondsuit, $\diamondsuit$; divide, $\div$; div, $\div$; downarrow, $\downarrow$; Downarrow, $\Downarrow$; element, $\in$; emptyset, $\emptyset$; equiv, $\equiv$; equivalent, $\equiv$; exists, $\exists$; forall, $\forall$; geq, $\geq$; gradient, ∇; greaterequal, $\geq$; heart, $\heartsuit$; heartsuit, $\heartsuit$; Im, $\Im$; in, $\in$; infinity, ∞; infty, ∞; integral, $\int$; intersection, $\cap$; land, $\wedge$; langle, $\langle$; lceil, $\lceil$; leftarrow, $\leftarrow$; Leftarrow, $\Leftarrow$; lessequal, $\leq$; leq, $\leq$; lfloor, $\lfloor$; lnot, $\neg$; lor, $\vee$; leftrightarrow, $\leftrightarrow$; Leftrightarrow, $\Leftrightarrow$; minus, $-$, nabla, ∇; ne, $\neq$; neq, $\neq$; notequal, $\neq$; notsubset, $\not\subset$; not=, $\neq$; notin, $\notin$; o, ø; O, Ø; paragraph, ¶; partial, ∂; partialdiff, ∂; perpendicular, $\perp$; plusminus, $\pm$; pm, $\pm$; pounds, £; product, $\prod$; prod, $\prod$; proportional, $\propto$; propto,

$\propto$; rangle, $\rangle$; rceil, $\rceil$; Re, $\Re$; registered, (registered); rfloor, $\rfloor$; rightarrow, $\rightarrow$; Rightarrow, $\Rightarrow$; spade, $\spadesuit$; spadesuit, $\spadesuit$; ss, ß; subset, $\subset$; subseteq, $\subseteq$; summation, $\sum$; sum, $\sum$; superset, $\supset$; supset, $\supset$; supseteq, $\supseteq$; surd, $\surd$; therefore, (therefore); times, $\times$, trademark, TM; union, $\cup$; universal, $\forall$; uparrow, $\uparrow$; Uparrow, $\Uparrow$; section, §; sterling, £; weierstrass, $\wp$; wp, $\wp$; and yen, (yen).

4.8.1 Japanese Text

The command KANJI will load the (Japanese) Kanji font. Standard ASCII characters are printed as normal roman characters. To access additional symbols in the Kanji font there are two methods: Use a sequence such as
\[ka]
This will print the hiragana equivalent of the "ka" syllable. To use katakana instead use
\[Ka]

The small "tsu" characters are entered using \[tt] for hiragana and \[Tt] for katakana.

If you know the JIS code this can also be used. For example \[3b33] will print the symbol "yama" (mountain).

The escape sequences used for JIS are also recognized. As the Escape character is not accepted in the text prompt, to be able to print this type of text you'll probably need to include it in a ".rob" file and use the "FILE" command to access it.

PostScript output files use the Ryumin-Light font with EUC encoding. You therefore need to have this font installed on your printer to be able to print output files.

Japanese PostScript hardcopy has never been demonstrated to work. (I'd welcome someone in Japan to work with on this!)

4.9 Interactive Editing of Plots

By double clicking on a text string you are able to bring up a window which will enable that text string to be changed. The change is *not* recorded in the log file but if you create a PostScript output file the change will be present.

One you have double clicked on a string to pop up the text editing window, single clicking on a text string will then alter the text string that is being edited.

Clicking or dragging within the current plot area also displays the coordinates of the mouse position at the bottom of the plot frame.

4.10 Miscellaneous

POSITIONPLOT - specify absolute location of the current plot window on the computer screen. Argument: x and y coordinates of top left corner in units of pixels. (0, 0) is the top left of the screen.
NEWPLOTAREA - create a new plotting area. The present plotting area is retained.
PRINT - create a PostScript file or send the current plot to the printer. PRINT properties (e.g. whether it creates a file or prints, the name of any file created) are set by the Print GUI control panel.

4.11 Publication

When using a ROBOT plot in a publication I typically use a font size of 24 points and a line width of "2". This usually seems to work pretty well, even if a journal reduces your plots down to a small size.

I appreciate receiving preprints of papers which have used ROBOT to make the plots. Also, if appropriate, please consider referencing the Corbet, Larkin & Nousek paper. Even if you don't reference the paper I still like to get reprints!

4.12 Programming

There are a few commands in ROBOT which enable simple "programs" to be constructed. Warning: these are all prototype and their formats and functionality may change. They're also presently a little limited.

VARIABLE - define a name to be used as a "variable". Variable names are *not* case sensitive. All variables are floating point. Argument: variable name.

ASSIGN - assigns a value to one of several user accessible variables. Arguments: (i) variable name, (ii) value. The default variable names available are A1, A2, A3, A4 and A5 together with any assigned using the VARIABLE command. These are "global" in the sense that instructions inside a FILE know all variable values.

LOOP - enables a simple FORTRAN-like do loop. Argument: start value for loop, end value for loop and increment value for each iteration of the loop. This is presently rather limited - you can't nest loops and only approximately 100 commands may be contained inside each loop. When you are inside a loop the variable "LOOP" which is the value of the loop parameter may be used in arguments for other commands.

IF - the argument is evaluated and the following commands are executed if the argument is positive.

ELSEIF - argument is evaluated and following commands executed if argument is positive.

ELSE - following commands are executed if preceding IF or ELSEIF had a negative argument.

ENDIF - terminates an IF block. This *must* be used.

IF statements can be nested. The FILE command may be used to create simple subroutine type structures - albeit with no arguments or returned values apart from those set using ASSIGN. The degree to which FILE commands may be contained inside a file called by a FILE command depends on the number of open files allowed by your operating system.

4.13 Arguments: names, arithmetic and functions

For most arguments, instead of entering a number you can give one of the following names: XMAX, XMIN, YMAX, YMIN, XMAXP, XMINP, YMAXP or YMINP. XMAX, for example, is the maximum x value of the data (including the error bars). XMAXP is the x coordinate of the right hand y axis. Using the command DATALIMITS changes the values of these parameters. The following names are also recognized: NUMBERPOINTS (number of data values), XCURSOR and YCURSOR (the coordinates of the graphics cursor after the CURSOR command has been used), PI (3.142...), TWOPI (6.2...), C (speed of light), E (electronic charge), H (Planck's constant), K (Boltzmann's constant). All physical constants are SI! If you have used the command MOMENTS then the parameters MEAN and SIGMA can be used which are the mean and the standard deviation of the y array.

If you are inside a loop the variable LOOP is also available. The user accessible variable A1 through A5 can be given values using the ASSIGN command. Any variables specified using the VARIABLE command may also be used. These also have precedence over the built in constants. For example, a user defined variable named "C" can have the value set using ASSIGN and need not be the speed of light.

Very simple arithmetic operations are also possible in the input. Only two operands are allowed and there must be no spaces between these and the operator. Examples are:

`c**2`

evaluated as a single value: the speed of light squared.

`c^2`

this is equivalent to the expression above.

`1+2`

evaluated as a single value: 3.

`1 +2`

because of the space this will be evaluated as two numbers: 1 and 2.

`1/2,-1/2 a1*mean`

evaluated as three numbers: 0.5, -0.5 and the product of the mean of the data set and the user assigned variable A1.

Simple functions of arguments are also possible. For example, these are valid arguments:

```
sin{10}
sqrt{4.1}
inverse{3/2}
tan{a1}
acos{a1**2}
cos{a1} tan{a2} 3+2
```

As with XFUNCTION etc. trigonometric functions assume degrees. Functions presently available include:

sin, cos, tan, sqrt, square, cbrt, cube, log, loge, log2, exp, exp10, abs, factorial, inverse, asin, acos, atan, sinh, cosh, tanh, asinh, acosh, atanh, j0, j1, round (IEEE rounding), ceil and floor. In addition to these data values are also available as "functions" i.e. x, y, z, xerr and yerr. x{3} for example is the "X" value of the 3rd data point. The results of least squares fits are available via fitvalue and fiterror. e.g. fitvalue{2} is the result of the fit for parameter no. 2. It is advised not to directly use fitvalue and fiterror as input to the CURFIT command.

5 Appendices

5.1 History

ROBOT was originally written to run on a Fujitsu mainframe computer as a command line driven program. This progressed via a Macintosh port to the only currently maintained version which runs under X Windows. This uses an OpenLook style user interface. However, it is possible to run it under other flavors of X window managers such as twm and mwm. It is anticipated that the future will be a Java port (of at least the user interface code).

Please send bug reports, comments, enhancement requests etc. to corbet@lheamail.gsfc.nasa.gov

5.2 Example scripts

simple_demo.rob

```
# Almost the simplest Robot file possible
# In principle the first two lines are not needed
# but are there in case this isn't the first Robot script to be run
# Anything after a "#" or a "!" is a comment
reset   # back to default values
n       # new page
type    # input some data
The title
X axis label
Y axis label
1 2 1
2 3 1
3 4 1

g                   # plot the data
return              # and finish reading from this file
```

fit1_demo.rob

```
! a simple straight line fit
n
reset
type
The title
X axis label
Y axis label
10 20 1
20 30 1
30 40 1

g
curfit
polynomial
1 1
end
plotcur
return
```

style_demo.rob

```
! some different plotting styles
reset
N
nticky 3
type



10 20 1 3
20 30 3.2 2
40 50 1.2 1
50 10 2.1 4

divwindow
1 9
selwindow
1 1
g
selwindow
1 3
minoraxes
plotmode
filledhistogram
g
selwindow
1 5
plotmode
dashedlines
g
selwindow
1 7
plotmode
nice-symbol
5
g
selwindow
1 9
plotmode
bars2
title
Different Plotting Styles
g
```

```
return
```

5.3 Installing ROBOT on a UNIX machine

ROBOT has mainly tested on a Sun. Limited tests on DECstations and Alphas have been made and a LINUX version has also been tried out. A more complete description of the installation procedure is to be found in the main README file included in the source distribution. The distributed ROBOT package contains three main directories: (i) Robot - containing the main code, (ii) ARK - containing a heavily modified version of the ARK plotting package, (iii) Fitsio, this has copies of Bill Pence's FORTRAN FITS routines. If you want to use the FITS routines in your own code to ensure you have the latest versions of the FITS code it is advised to obtain them directly from Bill Pence (ftp to heasarc.gsfc.nasa.gov). (iv) Docs - several pieces of documentation including this file, a "man" page, and robot.info (for real time help).

To build ROBOT move to the Robot directory. There will be a Makefile there. If you are lucky you will just have to type "make" and everything will compile. However, it may be necessary to make some changes. If you are not running OpenWindows when you do make you will first have to set a couple of environment variables. e.g.
$ setenv LD_LIBRARY_PATH /usr/local/openwin3/lib
$ setenv OPENWINHOME /usr/local/openwin3
As ROBOT uses both 'C' and FORTRAN it is preferable to have compilers for both of these. The f2c program may also be used, however, to convert the FORTRAN to 'C' (see makefile.f2c). If you don't already have the XView package this can be obtained by anonymous ftp from ftp.x.org. Binaries and ported source for XView on DECstations can be found at media-lab.mit.edu. f2c can be obtained by ftp from research.att.com (log in as *netlib* instead of *anonymous*).

5.4 The Macintosh version

[Section deleted]

5.5 Adding your own code to ROBOT

Since changing from a FORTRAN program on a Fujitsu mainframe (a user hostile machine) to an X based workstation program the internal workings of Robot have changed considerably. Things are still somewhat in a state of flux and, if you want to add your own code to ROBOT, it is strongly recommended that you contact the author directly.

Internally the main routine for getting user input is called "getit". This obtains input from a file or from the workstation screen as appropriate and writes to the log file if required. The program returns a string. The routine "dcode" obtains numerical information/key-words from the text string, determines if any of these values were enclosed in brackets (this is what ROBOT uses to tell if it's allowed to alter user input values or not) and does simple arithmetic operations. Adding a new command to the GUI is simple providing: (i) your command doesn't require other commands to be activated/deactivated, (2) your command itself isn't activated/deactivated by other commands, or (3) the command displayed in a menu is the same as the one stored in the log file and used internally by ROBOT.

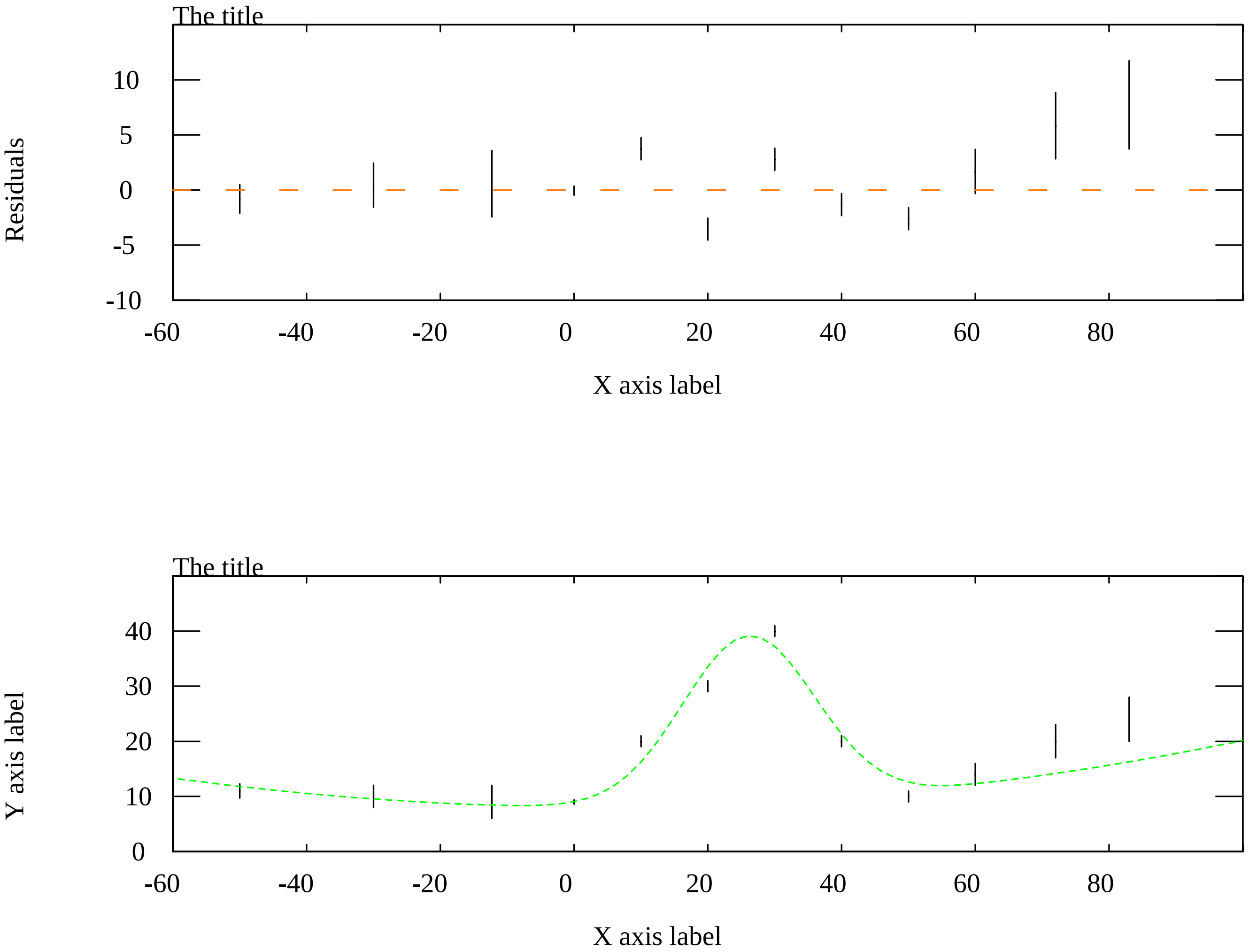


Figure 2: Example output from Robot (fit3_demo.rob)

5.6 Bugs

(i) Titles which contain control sequences are not perfectly centered.
(ii) Loops cannot be nested.
(iii) The icon is rather silly.
(iv) When an old plot window is made the current plot window the grayed out items in the menus don't correspond to reality. The workaround is to click on "Reset".
(v) The code does not yet use double precision.

6 Acknowledgments

No doubt people are missing from this list, but Robot has benefited from code and comments from at least: Alberto Meroni, Carlos Puchol, Jacques Steyaert, & Blair Zajac. The original ARK plotting package was written by Alan Smale, Robin Corbet, Koji Mukai, and Tim Naylor. This in turn was based on the MSSL (Mullard Space Science Laboratory) plotting package.

7 Bibliography

Adobe Systems Inc., 1990, *PostScript Language Reference Manual*, Addison Wesley, ISBN, 0-201-18127-4.

Bevington, P.R., 1969, *Data Reduction and Error Analysis for the Physical Sciences*, McGraw-Hill.

Corbet, R.H.D., Larkin, C. & Nousek, J.A., Proceedings of "Astronomical Data Analysis and Software Systems", Tucson, November, 1991. Astronomical Society of the Pacific Conference Series, 1992, **25**, 106. (ISBN 0-937707-44-9)

Knuth, D.E., 1986, *The TEXbook*, Addison Wesley.

Marquardt, D.W., 1963, *J. Soc. Ind. Appl. Math.*, **11**, 2.

Pence, W.D., 1992, BAAS, 180, 1401.

Pence, W.D., "Astronomical Data Analysis and Software Systems", ASP Conference Series, 1995, **77**, 245.

Press, W.H., Flannery, B.P., Teukolsky, S.A., & Vetterling, W.T., 1986, *Numerical Recipes*, Cambridge.