Convert XlispStat plots to Xfig 3.1

Bernhard Walter

May 31, 1995

1 Background

One of the major drawbacks of XlispStat is that graphics can only be saved as bitmaps. Therefore Jan De Leeuw and Frederic Udina wrote code to export plots derived from `graph-proto` to Gnuplot . This code can be found on `ftp.stat.ucla.edu` .
While this is an interesting way due to the availability of gnuplot on all platforms where XlispStat runs, it has, IMHO, some drawbacks:

- It produces no 1:1 image of a scatterplot (including colors, point symbols, selected points, labels, variable labels, etc)
- It produces one data file for every line, so you can easiliy get about 10 or more files for one plot (including the problem of a too long commandline for Gnuplot).
- The plots look much more like Gnuplot than XlispStat .
- The ratio between length of x-axis and y-axis of the original plot gets lost (circles were no circles any more) if you didn't change the Gnuplot size setting by trial and error.

I used the routine `:to-gnuplot` to convert my scatterplots to Gnuplot , convert them via Gnuplot to Xfig 3.1 , modify them in Xfig 3.1 and save them as Encapsulated Postscript.
It wasn't as easy as I wanted to have it. So I decided to write lisp code that converts scatterplots and other plots derived from `graph-proto` directly to Xfig 3.1 . It should have following features:

- Respect the ratio of the length x-axis to y-axis,
- Only use points that are really visible.
- Respect the color and symbol type of every point,

- Respect width, color and type of every line,
- Respect the setting of x-axis and y-axis, including ticks and labels,
- Lines constructed with `:add-lines` or `:abline` should get one polyline,
- Plot the variable names like XLispStat, if there are any.
- Include ability to change font (Postscript fonts) and fontsize,
- Provide the ability to select whether result should use colors or not,
- Provide a parameter to scale the plot as one likes it.

The result is a method for `graph-proto` called `:to-xfig`, and a bunch of auxiliary functions. These can be found in the file `xfig.lsp`. The code has following drawbacks:

- XFIG 3.1 only runs under UNIX (but this is no drawback at all :-)). Maybe the converting package of XFIG 3.1 (transfig) runs on Mac or under MSDOS, I don't know.
- It only works for objects in the plot that are drawn by `:add-points`, `:add-lines` or any function using these commands. (like `:add-function`, `:abline`, etc). Background items and strings besides axis and labels cannot be converted. This is due to the graphic system of XLISPSTAT , where one cannot find information about background items by using appropriate methods and slots. (this should read 'I cannont find', perhaps someone has an idea to detect these objects)

At this point I want to thank Frederic Udina for his useful tips during the development of this little program.

2 Example

If you load the file example.lsp, you'll get a scatterplot like figure 1, which object is named `plot`. Additionally three messages are sent to the object `plot`:

```
(send plot :to-xfig :filename "example.fig")
(send plot :to-xfig :bw t :filename "example_bw.fig")
(send plot :to-xfig :bw t :filename "example_bw_all.fig"
           :x-zero-axis t  :y-zero-axis t :box t :grid t)
```

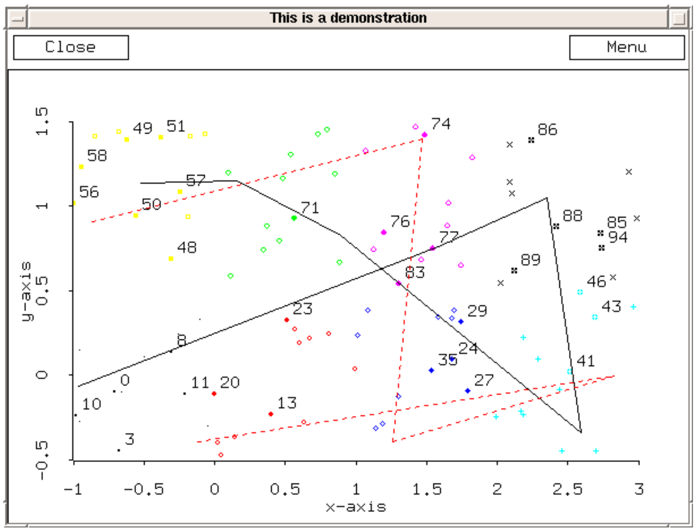

Figure 1: scatterplot grabbed from screen

The three commands create three files:

`example.fig`	colored XFIG 3.1 source
`example_bw.fig`	black/white XFIG 3.1 source
`example_bw_all.fig`	black/white XFIG 3.1 source with box, grid, zeroaxes

The colored version in XFIG 3.1 looks like figure 2. and after exporting it as encapsulated Postscript, you'll have a file that can easily be imported in e.g. TEX-texts. The results for the three files created above are in figures 3, 4 and 5.

3 Usage

To see how the method is used, simply load `xfig.lsp` into XLISPSTAT and say

```
(send scatterplot-proto :help :to-xfig)
```

or look at the file `example.lsp`.
The easiest method when you have a graph called `plot` is, to use

```
(send plot :to-xfig)
```

This creates a file with the name `plot.fig` that can be read into XFIG 3.1 .

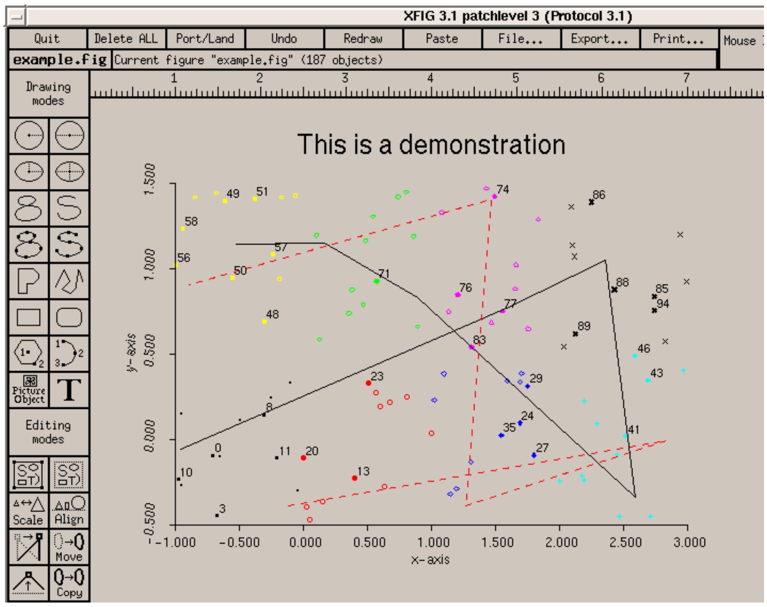

Figure 2: scatterplot in XFIG 3.1

Additionally some parameters exist:

keyword	default	description
:scale	12	factor to convert from 100 dpi screen to 1200 dpi XFIG 3.1 resolution
:bw	nil	Use black/white or not
:font	16	see below
:font-size	12	in pt (= 1/72.27 inch as usual)
:filename	`"plot.fig"`	should be clear
:box	nil	draw a box instead of XLISPSTAT axis
:x-zero-axis	nil	draw a horizontal line where $y = 0$
:y-zero-axis	nil	draw a vertical line where $x = 0$
:grid	nil	draw a grid according to the axes' ticks
:fig2dev	nil	T means, call fig2dev with scale factor 1.0 number s means, call with scale factor s
:fig2dev-par	`"-Lps -c -p xxx"`	Defaults to centered portrait EPS, to omit the bounding box use: `"-Lps -c -p xxx -P"`

As you can see, the last four parameters contradict the idea of creating a 1:1 image of a XLISPSTAT plot. But sometimes one can see more in a graph if there is a grid and if there are zero axes in it.
Of course these key arguments could be set by using a dialog-box like in the `:to-gnuplot` method of Udina/De Leeuw, but I personally don't miss one.

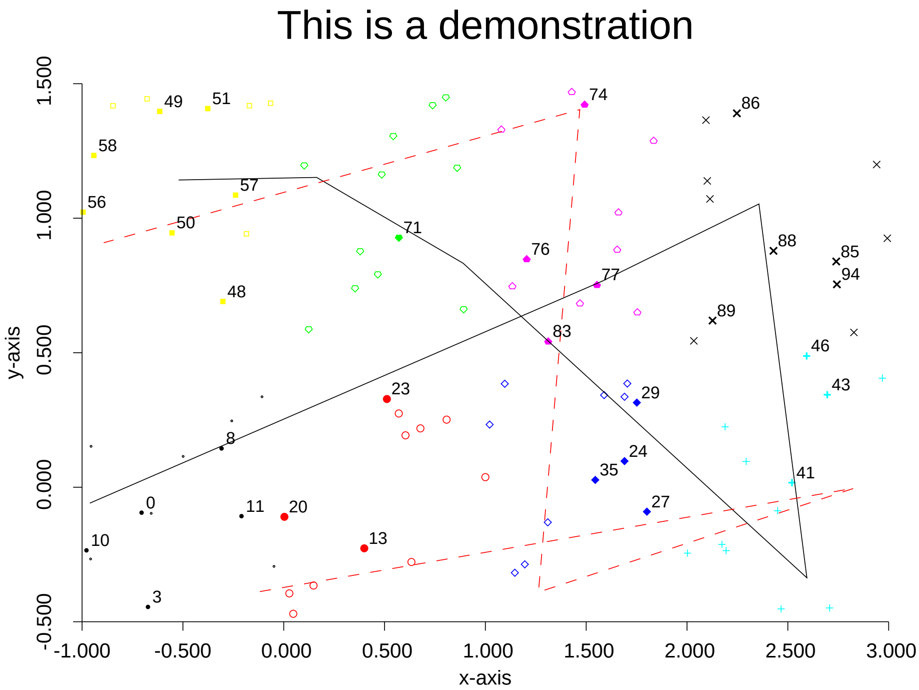

Figure 3: imported colored EPS file

The following list of fonts is derived from the description file `FORMAT3.1` that comes with XFIG 3.1 :

0	Times Roman	18	Helvetica Bold
1	Times Italic	19	Helvetica Bold Oblique
2	Times Bold	20	Helvetica Narrow
3	Times Bold Italic	21	Helvetica Narrow Oblique
4	AvantGarde Book	22	Helvetica Narrow Bold
5	AvantGarde Book Oblique	23	Helvetica Narrow Bold Oblique
6	AvantGarde Demi	24	New Century Schoolbook Roman
7	AvantGarde Demi Oblique	25	New Century Schoolbook Italic
8	Bookman Light	26	New Century Schoolbook Bold
9	Bookman Light Italic	27	New Century Schoolbook Bold Italic
10	Bookman Demi	28	Palatino Roman
11	Bookman Demi Italic	29	Palatino Italic
12	Courier	30	Palatino Bold
13	Courier Oblique	31	Palatino Bold Italic
14	Courier Bold	32	Symbol
15	Courier Bold Oblique	33	Zapf Chancery Medium Italic
16	Helvetica	34	Zapf Dingbats
17	Helvetica Oblique		

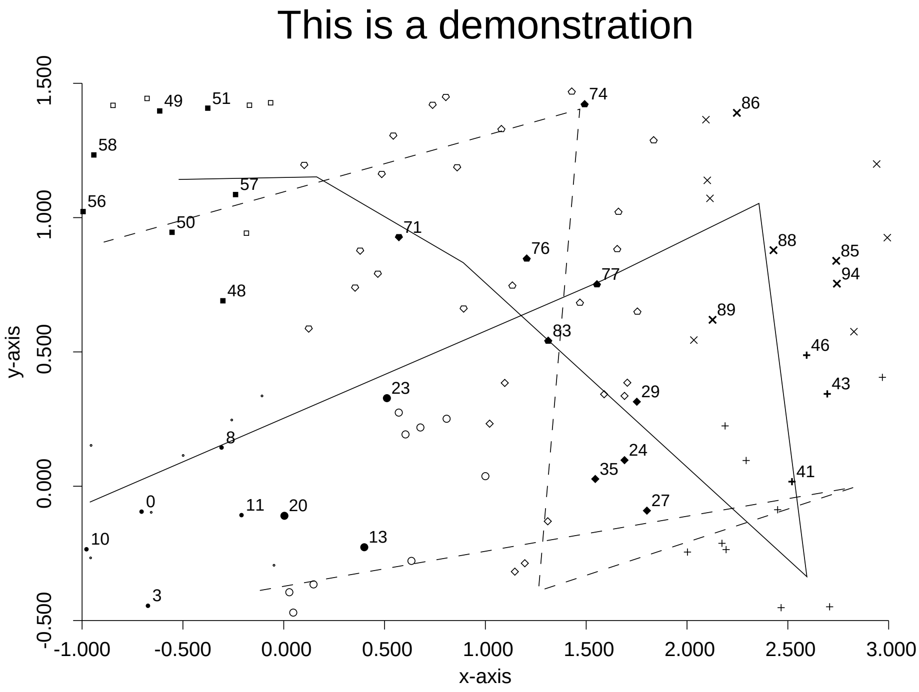

Figure 4: imported black/white EPS file

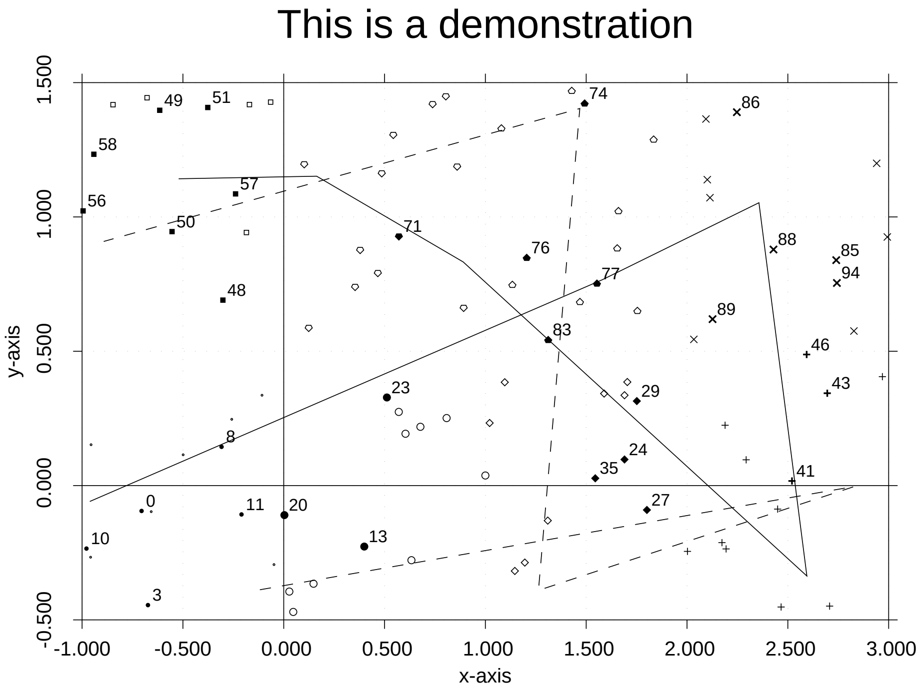

Figure 5: imported black/white EPS file (with box and grid)