The University of British Columbia

Physics 210 Assignment # 4: PLOTTING

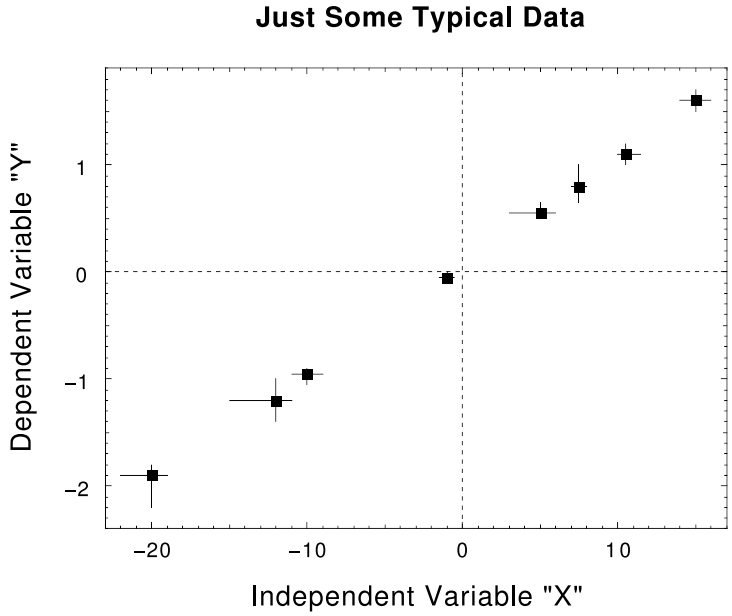
Tue. 28 Sep. 2010 — finish by Tue. 05 Oct.

In this assignment you will take the following data file (on the left)[1] and generate the plot on the right (or as close as you can manage) *using several different plotting applications*, namely `muview`, `gnuplot`, `extrema`, `matlab` and `octave`.

data.db

```
TITLE
Just Some Typical Data
LABELS
Data Set Number
Independent Variable "X"
Dependent Variable "Y"


DATA DSN X Y
1,,, -20,1,2, -1.9,0.1,0.3,
1,,, -12,1,3, -1.2,0.2,,
1,,, -10,1,, -0.95,0.05,0.1,
2,,, -1,0.5,, -0.05,0.05,,
3,,, 5,1,2, 0.55,0.1,0.05,
3,,, 7.5,0.5,, 0.8,0.2,0.15,
3,,, 10.5,1,0.5, 1.1,0.1,,
3,,, 15,1,, 1.6,0.1,,
```



Create your `/home2/phys210/<you>/a04/` directory and the subdirectories `muview/`, `gnuplot/`, `extrema/`, `matlab/` and `octave/`, where you should store any files used to make the same plot (or as close as you can get) with the respective applications.

With each application, for full credit, display the data with *asymmetric uncertainties* ("error bars") and plot them on a simple graph like that above. (*Hint:* to do this in `matlab` you will have to Google the *undocumented* `matlab` procedure **errorbarxy**.) If this proves too difficult, use the RMS (square root of the average of the squares) of the positive and negative uncertainties to plot symmetric "error bars".

You will need to edit `data.db` into differently formatted ASCII "data files" from which each program can read in the data; if so, give those files obvious mnemonic names like `matlab.dat`; if possible, each file some should include some comment lines explaining what it is and how it is meant to be used. (Most applications allow comments in some form.) In each case you should also be able to make a "command file" that tells the application to read in the data file, prepare and execute the plot, *etc.* It should also have some obvious mnemonic name like `plot.cmd` or `plot.m` (some applications may require a special extension for command files).

---

[1] The file **data.db** can be copied from the `/home/phys210/` directory. It is in a format originally designed to be read in by a program called `db` which I wrote several decades ago, and which is now adopted as one of the recognized input formats for the *Java* spreadsheet program *μView*, whose use will be demonstrated in class. Not surprisingly, that format seems self-explanatory to me by now; its one non-obvious aspect is that every "column" of data comes with *two* "errors" (uncertainties): a positive error and a negative error. Thus, in the first row, "`1,,,`" in the `DSN` column specifies a value of 1 for the Data Set Number, with no errors; "`-20,1,2,`" in the `X` column specifies a value of -20 for `X` with a positive uncertainty of $+1$ and a negative uncertainty of -2; that is, $\mathtt{X}_1 = -20^{+1}_{-2}$. When the first error is finite but the second one is missing, as in "`-1.2,0.2,,`" in the `Y` column (second row), it does *not* mean that the negative uncertainty is zero; rather that the uncertainties are *symmetric* — *i.e.* $\mathtt{Y}_2 = -1.2 \pm 0.2$, and so on. I hope this is clear. Lists of arbitrary length, like `LABELS` and `DATA`, are terminated by a blank line.

Make a `plot.ps` or `plot.eps` or `plot.pdf` file in each case (this is your "answer") and store it with the other files for that application.

Within reason,[2] you should learn how to control the æsthetics of your graph (things like limits, "tick marks", font sizes, symbol shapes and sizes, zero-axes, axis labels, title, captions and so on. There are certain standards for "publication quality" plots. See for instance `http://authors.aps.org/STYLE/` or any copy of PHYSICAL REVIEW LETTERS or another APS journal to get an idea of what is expected. (Most of it is "just common sense", but Einstein taught us how unreliable "common sense" can be. `:-)`

The "freeware" applications `muview` and `extrema` were developed at TRIUMF; their use will be demonstrated in class so that you should have no trouble producing the required plots with them. Both run on both Linux and Windows.

Because `muview` is a *Java* applet, your `$HOME` directory must contain the file `.java.policy`. (This file is also needed to use `muview` under Mac-OS/X or Windows — see `http://musr.org/muview/requirements.html`. On your Mac it goes in your `$HOME` directory, but in Windows its proper location is mysterious — it depends on which version of Windows you are running.)

The other freeware applications `gnuplot` and `octave` can be found on any Linux distribution and are also available free for Windows.

The very expensive proprietary application `matlab` is available on `hyper` for your use; it comes with a fancy GUI (Graphical User Interface) including voluminous `help` files, excellent tutorials (see the "Manuals" page on our Website) and slick graphics — unlike its freeware cousin `octave`, which has only simple `help` commands and uses `gnuplot` for its graphics. But `octave` is free and Open Source.[3]

[2] "Æsthetics" are the subject of endless and passionate debate because they involve subjective, idiosyncratic and undefinable values. In other words, graphics is an art form. It is also open-ended: you can invest an arbitrary amount of time in "perfecting" any single graph, only to discover that someone else doesn't like it. That is not the aim of this assignment.

[3] Another freeware "cousin" of `matlab` is `scilab`, a project supported by the government of France (presumably to free that country's engineers from the grip of *MathWorks*). Each of these handles scripts that are very similar to the basic `matlab` "`.m`" scripts, but each has its own idiosyncracies, especially with respect to plotting. There is also a `python` package called `pylab` which emulates `matlab` and makes nice plots, but these five are enough for one assignment. We'll get back to `matlab` soon. . . .