Homework 0 - SVN and IDL

Don't forget to `svn up` *before you issue any other commands in SVN—this is to guard against you changing a document that someone else is working on in the same directory*[1].

Don't forget to `svn ci` *(with* `-m` *comments) frequently as you work. This allows others to see how your work progressed, and it automatically backs your work up as you produce it so that you're less likely to lose any of your work and/or so that you can revert to earlier versions of your work.*

Don't forget to log in using **your** *username and password, not the username and password that we use in class. Remember to comment your code carefully with your initials beside every comment (as in* `; ADM I just wrote an IDL comment`*). Remember to provide an informative header for* **every** *function and procedure that you write.*

Homework

1. Write an IDL function that is passed the values m and c in the expression for a straight line $y = mx + c$ and; **(i)** generates 10 floating point numbers in the range 0 – 10, at random, along the x-axis (*hint: look at the RANDOMU function in IDL*); **(ii)** uses $y = mx + c$ to recover the appropriate y values; and **(iii)** scatters each of the 10 points in the $\pm y$ direction according to a random offset drawn from a Gaussian of standard deviation 0.5 centered on a given y value (*hint: look at the RANDOMN function in IDL*). Your function should return an array of the x values, an array of the y values, and an array of the error on y (this $y_{\rm error}$ is always 0.5).

2. Use MPFITFUN (see the Week 1 links in the syllabus for documentation of MPFITFUN) to fit a straight line to your generated $x, y, y_{\rm error}$ data and recover the best fitting values of m_2 and c_2 (which may, of course differ from your original m and c).

3. Plot your data, the original line ($y = mx+c$) that your data was drawn from, and the best-fitting line ($y = m_2x + c_2$) that MPFITFUN derived, to the screen.

4. Use PS_START and PS_END (see the link from the Week 1 links in the syllabus for documentation of PS_START and PS_END) to create a hardcopy of your plot in PNG format (PNG is suitable for displaying in webpages). Try to make your plot look as professional as possible. Consider whether the x and y ranges displayed in the plot are appropriate. Use color to distinguish lines and points. Use an appropriate symbol to display your points and don't forget to display the errorbar. Consider adding labels. Make sure your characters and lines are of appropriate thickness to display on a webpage (*hint: look at IDL functions like plot, oplot, ploterror, oploterror, errplot*).

5. Finally, create a procedure that hooks the steps 1. – 4. listed above together to create a single IDL procedure that is passed a value of m and a value of c, generates some mock x, y, y_{error} data, fits a line to the mock data, plots everything to the screen and makes a hardcopy in PNG format. Use the SYSTIME(1) function to add a timer to your code, and print useful comments and times to the screen after each of the steps (1. – 4. listed above) in this master procedure.

[1] this shouldn't be a big deal until you start to work on the team project, but you should get into the habit *now*