Numerical Differentiation

HW 1: Wednesday, Aug. 31, 2016
DUE: Wednesday, Sep. 7, 2016 (no class Labor Day, Mon. Sep. 5)
READ: Numerical Recipes, Introduction, pages 1-24
Numerical Diff., section 5.7 pages 229-232
(you may skip the routine dfridr() on page 231)
OPTIONAL: Landau et al, Intro. to Roundoff Error and Prog., chp. 1,2
Numerical Diff., section 7.0 to 7.6
OPTIONAL: Yang, Intro. to C/C++, chap. 1,2, pgs. 1-47
OPTIONAL: Ammeraal, C/C++ for Prog., start reading the first 4 chapters

You should start reading the first few chapters of the programming book that you have chosen. In the future, explicit reading assignments on programming will not be given. You should read them on your own as needed.

PROBLEM:

Most of the homework assignments for this course will involve writing a program to calculate something and then graph the data you calculate. You will need to have a compiler (to compile your program), a text editor (to prepare the program) and some form of graphing program (see links on course web site and notes below). The main purpose of this homework is to become familiar with the computer, compiler and graphics package you plan to use. The numerical portion of this homework is not very complicated, so *you do not have to hand in a written homework for this assignment*. You are **strongly** advised to complete this assignment anyway. The homework assignment next week may be significantly more difficult if you do not do this one.

Given:

$$f(x) = \sin(x)\exp(-0.04x^2) \tag{1}$$

Write a program to numerically calculate the first derivative of $f(x)$, $f'(x)$ using the forward difference, backward difference and central difference methods with the difference step h=0.5 and h=0.05. Compare the three methods for h=0.5 and compare the results with h=0.5 to the results with h=0.05. You should write a short program (in C/C++) that prints out the numbers in a file. It is easiest to write a new program by starting from an old program. A sample listing (example1.cpp) of a similar program is shown below. You can start by typing it in and getting it to run. Then modify it to do the extra tasks listed above. Then run a graphing program (such as octave, python, scilab, matlab, gnuplot, etc.) to plot the data (UNIX/Linux users might look at the Grace/mgr data plotting program). You should produce a plot with the three first derivatives and the function in the range -7 to +7 with approx. 200 points in each of the four curves. Be sure to get a hardcopy of your program and the graph from the printer.

Next, numerically calculate the second derivative of $f(x)$ with h=0.05. Produce a graph of $f(x)$ and $f''(x)$ (on the same graph). Use the same range of x as above. Compare this to what you should expect. Be sure to get a hardcopy of the graph and the source program from the printer.

Once you have your program running and you have finished your project, copy your programs onto a removable disk for safe keeping. Remember that the hard disk files in a public computer facility may or may not be available the next time you come to the computer room. Also it is best to keep all of your files in a folder or sub-directory with your name on it and do not scatter them all over the hard disk.

COMPILING

There are many choices of compilers that you can use. Several are listed below.

gcc

gcc/g++ is a good, standard compiler available on a variety of different platforms. You will probably see it again later in your career so it's worth your time to learn something about it. There are free versions available for MS-Windows (see links to MinGW= 32 bit, cygwin, and tdm-gcc = 32/64 bit on the course web site) as well as several other types of computers. It is also the default compiler under Linux. The Mac-OS uses clang which has a gcc mode and gcc is available through macports. gcc is actually a collection of compilers and g++ is a form of gcc to compile C++ programs (when executed as gcc it compiles plain C programs). First create a new file with the C/C++ source code using a plain text editor such as gedit or emacs under Linux or the "Crimson Editor" or notepad++ (see course links) under Windows. Xcode is the default IDE front end for clang under the Mac-OS. Any plain text editor will due but the gcc/g++ error messages will refer to a specific line number so you need an editor that gives you the line numbers (i.e. notepad under MS-Windows will not give the line number although it works as a plain text editor). The code-blocks package is a user friendly GUI front ends for gcc under Windows (code-blocks is cross platform). Your files should be in your normal default directory (for example /home/students/yourname under Linux). Once the source code file is completed, compile it using gcc/g++. For example to compile the file called hw1.cpp execute the following command:

```
g++ -O -o hw1 hw1.cpp
```

The command options are case sensitive. -O (upper case) means to optimize and -o (lower case) means to output the executable with the name that follows. The rest of the line is just a list of the files to compile. There may be more than one file to compile (for example, the main program plus one or more files with separate subroutines, etc.) On-line help is available for gcc (under Linux or cygwin) using the 'man gcc' command (man is short for manual) and also at the GNU web site (www.gnu.org).

After compiling successfully, you can then run the file called hw1 by typing 'hw1', (some version of Linux may require you to type './hw1' instead) followed by a return (the Mac-OS is really UNIX runnig the bash shell so works this way too). You should arrange for your program to write data into a file which can then be plotted using a graphing program (see below).

Integrated Developement Environment, IDE

There are a variety of IDE's (integrated development environments) for most computer platforms. These typically include some form of graphical user front end with an editor and easy menus to compile and run your program. Most work in a similar manner and a few are discussed below. Most of these include a debug mode and a release mode. Use debug mode when writing your program. It will run a lot slower but gives you more information when something goes wrong. When your program is running switch to release mode which usually runs much faster. It's not always easy to find the right control to change modes (Xcode on the mac is hidden strangely). A few IDE's are described below. Many use gcc as their principle compiler.

Xcode

Xcode is the IDE for use with the Mac OS. You can download it from the Apple web site. It includes a graphical user front end with editor and compiles using clang. It is sometimes helpful to also have the command line tools for clang which are not installed by default (see link on course web site). It may leave code and data files in directories that can be very hard to find. One possible mode of operation is to use the Xcode IDE as an editor and then compile and run from the command line so that files are left is easy to find locations. Xcode also defaults to run in debug mode (slow execution speed but more error checking). Its good to start programming in debug mode but once your program is running OK then its better to switch to release mode. This option is hidden under the project name (upper left) and you must click on it and chose "edit scheme". A new selection window should appear and let you select debug or release mode.

Code::Blocks

This is a free integrated development environment (IDE) that is cross platform (windows, linux, MacOSX, etc.) and is a front end for a variety of different compilers (such as gcc). This discussion is for windows but the procedure should be similar under other operating systems. First open CodeBlocks, then choose "Create a new project" and select "console application" and "C++". You will then need to select a directory and project name. It will make an empty "main.cpp" file. You can start to edit this one or add an existing file with your programs (choose "project/add files", and then delete main.cpp). You can compile and run your new program from the "Build" menu item. There are a variety of menu items to control compiler options etc. It is a nice front end for programming in gcc/g++ (from MinGW). You can go back to your project by opening the .cbp file.

Orwell Dev-c under Windows

This is a free windows only IDE, which is interesting in that it produces very fast 64 bit executables using its own version of 64 bit gcc. It generally follows the same strategy as code-blocks but the details are a little different.

MS-Visual C under Windows

The computers in the AEP computer room running Win/7/Vista have MS-Visual Studio express (as well as gcc). The express version of MS Visual C++ can be downloaded for free from microsoft

(beware that it may take a long time to download and requires registration). This is a complete integrated development environment (IDE) that includes an editor, a compiler and a linker. It is relatively easy to type in a program and run it from within the Visual C/C++ IDE. First launch Visual C++ from the START button (in the lower left hand corner). A project is a description of the executable that you want to build. Pull down the 'File/New' menu and choose 'Project'. It also requests you to select a directory and project (i.e. executable program) name at this stage. Note that you may only store files in the directory C:\users\Students\StudentFolders directory on the AEP computers. Do NOT store files anywhere else. Then select 'Win32-Console-Application', give it a name and chose 'empty project' under 'Application options' to start a new project. To enter the actual C/C++ program choose 'Project/Add-New-Item', select 'C++ file' and give it a name ending in '.cpp'. When you are done typing save the file. It should be inserted into your project automatically. If not, then insert it into the project using the 'Project/Add-Existing-Items' menu item. To compile your program, pull down the "Build/Rebuild-Solution" menu to compile the program. MSVC++ should add the appropriate libraries etc. into your project as well as the source code file. You can execute your program while in the Visual Studio using 'Debug/Start...' menu or invoke the command prompt and execute it manually (it will be inside the Debug or Release sub- directory). Matlab, Scilab, octave, gnuplot or other programs may be used to graph your results. Note that MSVC defaults to running in Debug mode which is helpful in debugging a program but makes it a lot slower in most cases. When your program is working you may rebuild it in Release mode to speed things up (for long programs).

GRAPHING

There are also a large variety of scientific/engineering graphing programs available. A few are listed below. Note that MS-excel can do some simple graphs but in general is a poor choice for scientific/engineering graphics, and you should try to learn something better.

Matlab and Octave

Matlab is a general purpose numerical calculation package. It also has some useful graphing abilities (2D and 3D) which you may wish to use. **octave** is a free alternative to Matlab (see links on course web site), has a very similar syntax, and runs under MS-Windows, Linux/Unix and Mac-OS. The following Matlab/Octave commands plot the ASCII data in a file called example1.dat (with three columns of numbers, x plus two values of y).

```
%
%  Matlab/Octave script to plot data
%
load 'example1.dat'
x = example1(  :, 1 );
y1 = example1( :, 2 );
y2 = example1( :, 3 );
plot ( x, y1, x, y2, '--' );
xlabel( 'x' );
ylabel( 'y' );
print -deps example1m.eps
```

The result is shown in figure 1. The last line 'print -deps graph.eps' generates a postscript output into a file that can be inserted into MS-word, LaTeX or some other word processing package. It is in general useful to enter these commands into a file which can be called from Matlab. Then if you just need to make a few small changes you don't have to type them all in again, but just execute this script file.

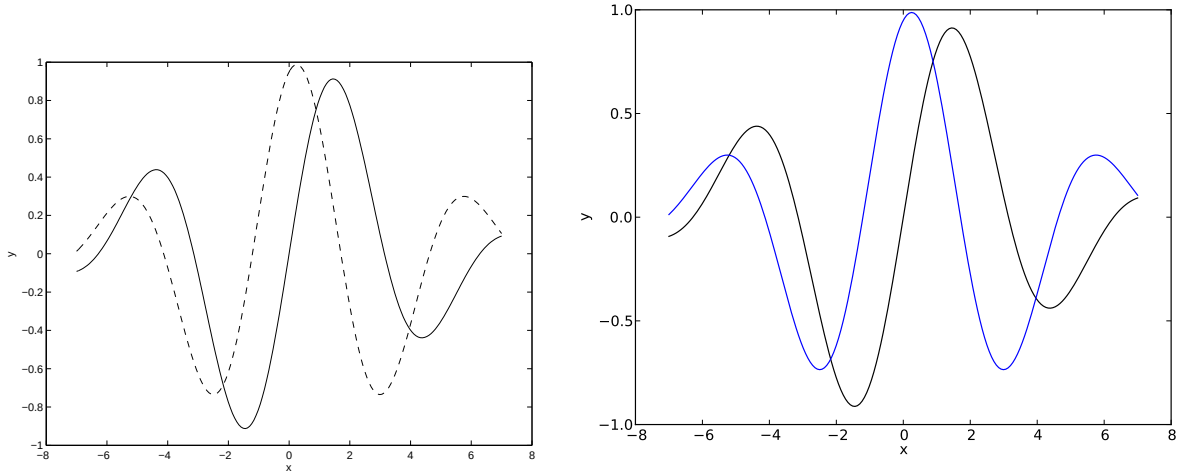

Figure 1: Sample graph from Matlab (left) and python with mathplotlib (right).

Python/Matplotlib

Python is a high level scripting language that also has a nice graphing package called matplotlib. There are several packages including python and the extra components for graphing etc. such as python(x,y) (32 bit windows) and anaconda (32 and 64 bit windows and mac-os) which are easier than installing python and several separate packages. It is free and available for most platforms

```
#
# uses numpy, scipy, matplotlib packages
from pylab import *

#  read three-col data file x and two y values
data = loadtxt( "example1.dat", 'float' )
plot( data[:,0], data[:,1], 'k-',  data[:,0], data[:,2])
xlabel( "x" )
ylabel( "y" )
savefig('example1py.eps')
#savefig( 'example1py.pdf' )
show()
```

The result is shown in figure 1. The last line 'savefig('example1py.eps')' generates a postscript output (other formats available) into a file that can be inserted into MS-word, LaTeX or some other word processing package. It is in general useful to enter these commands into a file which can be called from python. Then if you just need to make a few small changes you don't have to type them all in again, but just execute this script file.

Scilab

Scilab is very similar to Matlab, except that it can be downloaded for free. Within Scilab you can change directories using the file/ChangeDirectory menu item or from the Scilab command line type `cd('D:\students');` to move to the directory `D:\students`. A large variety of 2D and 3D graphing commands are available. Below is a list of commands to plot a data file (in 2D) with three columns of numbers (x plus two values of y).

```
//  scilab script to plot example1.dat
//
data = read('example1.dat', -1,3);
x=data(:,1);
y1=data(:,2);
y2=data(:,3);
plot( x, y1, 'k-', x, y2, 'b--' );
xtitle('example1','x','fp(x)');
//xs2eps(0,'example1s.eps');  // output graph as postscript file - broken in current version
xs2png(0,'example1s.png');  // output graph as png file
```

The result is shown in figure 2. As with Matlab it is helpful to enter these commands in a script file. It is then easy to make small changes and call it again. To call a script file mygraph.sci from within Scilab type exec('mygraph.sci'); or pull done the file/exec menu item. You can generate a postscript output file from the menu item file/export in the graphics window or with the command shown in the example. There is a good built in help facility and a lot of documentation on the scilab web site (www.scilab.org). Please read the documentation for an explanation of the above commands.

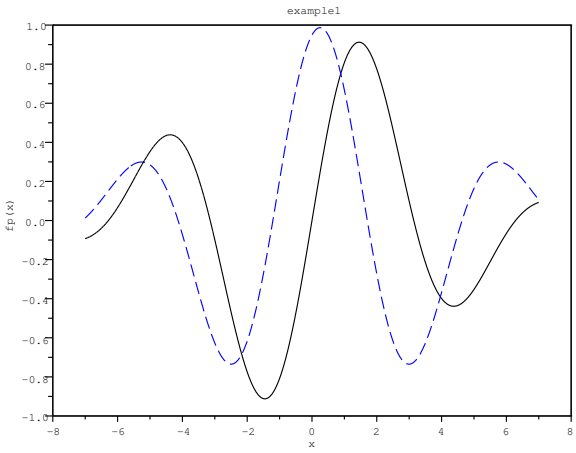

Figure 2: Sample graph from Scilab.

Gnuplot

Gnuplot is a simple data plotting software package. It is a little crude but it is free and can also do reasonable contour and 3D surface plots which may be needed later. There are Mac, Windows and

Linux/UNIX versions available over the Internet starting from the link given at www.gnuplot.info. GnuPlot is the main scientific graphing program under Linux. If the file example1.dat contains three columns of numbers (x, y1, y2) then the following gnuplot command will plot both sets of y values as a function of the common x value (some keywords may be abbreviated):

```
plot "example1.dat" using 1:2 with line, "example1.dat" us 1:3 wi dot
```

Once it looks OK on the screen you can create a postscript file that has a higher resolution (i.e. looks better) for printing (or cut and paste into word if running MS-Windows). To create a postscript file (in Linux) run the following after the above sequence:

```
set output "myfile.ps"
set term post
replot
set output "STDOUT"
set term x11
```

This creates a postscript file called "myfile.ps". Under Linux use the *lpr* command to spool the postscript file to the printer. You can also view a postscript file on the screen using ghostview (type "gv myfile.ps" in linux) and then print it from the ghostview menu. In MS-Windows you just double click on the eps file to display it if Ghostview is installed.

prestoplot

This is a free downloadable Windows program modeled after the UNIX/Linux program grace/mgr written by Cornell people (see course web site for a URL link). It is very easy to use. Launch the program and pull down the file/import menu item. Select your data file and type (2 col. or many col. data etc.) and it will plot it for you. You can annotate the graph by clicking on the axes etc. or using the appropriate menus. Print the graph using the file/print menu (you can produce an eps file by printing to a postscript printer and selecting 'print to file', some print drivers work better than others though).

Grace

Grace (plasma-gate.weizmann.ac.il/Grace/) is a free GUI based data plotting and analysis program manly intended for UNIX/Linux systems. It also runs under windows via the cygwin package. It looks nice but I don't have much experience with it.

Sample Source Code Listing

```
/*  AEP 438 Example # 1

    Test numerical derivatives
    Only the backward difference method is used here
    Run on a core i7 with g++ 4.8.1 and/or MS Visual C/C++

    E. Kirkland 23-aug-2016
*/

#include <cstdlib>  // plain C
#include <cmath>

#include <iostream>  //  stream IO
#include <fstream>   //  stream file IO
#include <iomanip>   //  to format the output

using namespace std;

int main()
{
    int i, n=200;
    double h=0.5, xmin=-7.0, xmax=+7.0, x, dx, f1, fpbd;
    double feval( double );

    ofstream fp;       // output file using streams

    fp.open( "example1.dat" );  // open new file for output
    if( fp.fail() ) { // or fp.bad()
        cout << "cannot open file" << endl;
        return( EXIT_SUCCESS );
    }

    dx = ( xmax - xmin ) / (n-1);
    for( i=0; i<n; i++){
       x = xmin + i * dx;
       f1 = feval(x);
       fpbd = ( f1 - feval(x-h) )/h;

       // data file for python, Matlab;  need to separate into col.
       fp << setw(15) << x << setw(15) <<  f1 << setw(15) << fpbd << endl;

       // print to screen
       cout << setw(15) << x << setw(15) <<  f1 << setw(15) << fpbd << endl;
    }

    fp.close();

    return( EXIT_SUCCESS );

}   // end main()

//------------------- function feval ----------------
double feval( double x )
{ return(  sin(x) * exp( -0.04*x*x ) ); }
```