

# AMS 527, Spring 2009, Homework 1

65 points. Due: Monday 02/09.

Electronic submission of homework assignments is strongly encouraged for the computer problems. For the written part, you are encouraged (but not required) to typeset using  $\text{\LaTeX}$  or  $\text{\LyX}$  (an easy-to-use front-end of  $\text{\LaTeX}$ ). For electronic submission, homework is due at 11:59pm on the due date. For paper submission, homework is due in class on the due date.

0. (0 points) This exercise is for you to get familiar with the computing environment. Unless you are proficient in setting up your own computer, you are urged to use the Linux (not Windows) computers at the Matlab SINC Site in Math Tower S-235 to do all your computing assignments. Before you can log-on to the computers at the SINC site, you may need to go to the Matlab SINC site in person to activate your account on any of the Linux system by following the on-screen instructions. After you log in, make sure you can run `emacs` (for editing), `gv` (for viewing eps files), and `ddd` (for debugging).

If you have a personal computer, after activating your Matlab account you can remotely log onto the computer “`compute.mathlab.sunysb.edu`” from your PC. You need an ssh client and XWindows server on your local computer (available on Linux, Mac OS X, and Cygwin on Microsoft Windows). You can remotely log onto the computer using command “`ssh -X -Y username@compute.mathlab.sunysb.edu`”, where `username` should be changed to your own username, and `-X` and `-Y` options would ensure X Windows applications can be displayed on your local computer. After you remotely log onto the computer, try to run `gv`, `ddd`, and `emacs` and make sure the windows are properly displayed on your local computer. Practice using `scp` to copy files between you local and remote computers.

Download [http://www.ams.sunysb.edu/~jiao/teaching/ams527\\_spring09/HW/hw1\\_template.tgz](http://www.ams.sunysb.edu/~jiao/teaching/ams527_spring09/HW/hw1_template.tgz), copy it to the Matlab computer, and run command “`tar zxvf hw1_template.tgz`”. You are now ready to start the actual homework.

1. (25 points)
  - (a) (5 points) Use the given C program `hw1_1a.c` to compute an approximate value for the derivative of a function using the finite difference formula

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}.$$

The program uses the test function  $\tan(x)$  for  $x = 1$ , and determines the error by comparing with  $\tan(x)^2 + 1$ . It prints out the magnitude of the error for  $h = 10^{-k}$ ,  $k = 0, \dots, 16$ , and plots the magnitude of error as a function of  $h$  using log scale.

Read the program and follow its instruction to compile and run it. What is the minimum value for the magnitude of the error? How does the corresponding value for  $h$  compare with the “rule of thumb”  $h \approx \sqrt{\epsilon_{\text{mach}}}$ ?

- (b) (10 points) The given program uses double-precision floating point numbers. Edit the program in an editor (such as `emacs`) and modify the program to use single-precision floating point numbers. Perform the same test, plot the results, and answer the two questions in part (a).

- (c) (10 points) Modify the program to use the centered difference approximation

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

and repeat the same test as in (a) using both double-precision and single-precision floating point arithmetic.

For each part of the problem, turn in the modified part of the program, the plots, and your explanations of the results.

2. (15 points) Starting from the code template `hw1_2.c`, complete the program to generate the first 60 terms in the sequence given by the difference equation

$$x_{k+1} = 2.25x_k - 0.5x_{k-1},$$

with starting values

$$x_1 = \frac{1}{3} \quad \text{and} \quad x_2 = \frac{1}{12}.$$

Make a semi-log plot of the values you obtain as a function of  $k$ . The exact solution of the difference equation is given by

$$x_k = \frac{4^{1-k}}{3},$$

which decreases monotonically as  $k$  increases. Does your graph confirm this theoretically expected behavior? Explain your results. Turn in the modified part of the program, the plots, and your explanations of the results.

3. (25 points) To solve the quadratic equation  $ax^2 + bx + c = 0$ , one can use the standard quadratic formula

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

or the alternative formula

$$x = \frac{2c}{-b \mp \sqrt{b^2 - 4ac}}.$$

- (a) Show that the alternative quadratic formula indeed gives the correct roots for the quadratic equation (assuming exact arithmetic). When the standard quadratic formula would be more accurate than the alternative formula and vice versa using floating point arithmetic?
- (b) Use the given sample C program `hw1_3b.c` for the standard quadratic formula to solve the quadratic formula for the following coefficients in the following table. Fill the solutions into the table.

coefficients			reference solutions		approximate solutions	
$a$	$b$	$c$	root 1	root 2	root 1	root 2
6	5	-4	1/2	-4/3		
6e154	5e154	-4e154	1/2	-4/3		
0	1	1	-1	-		
1	-1e5	1	9.999999999e4	1.0000000001e-5		
1	-4	3.999999	2.00100000000007	1.99899999999993		
1e-155	-1e155	1e155	1	-		

- (c) Enhance the robustness of the code to guard against
- division by zero,
  - unnecessary overflow and underflow (e.g., by scaling the coefficients), and
  - cancellation (e.g., by using alternative formula when appropriate).

You do not need to handle complex roots. Produce a similar table as in part (b). Turn in your modified part of the program, a description of your changes, and the tables.