

CS 257: Numerical Methods
Spring 2006

Homework, Set 5

Due Thursday February 23, 2006

- (-) Start EACH PROBLEM on a SEPARATE piece of paper (This is important since we may assign each problem to a different grader).
 - (-) Put your NETID and HW NUMBER on top of EACH PAGE clearly, e.g. “netid: zamani hw2”.
 - (-) Write descriptive solutions. Comment your code!
 - (-) Include your curves/graphs (and other supporting materials) in your write-up.
 - (-) Don’t use handwritten code (unless you want to lose points), copy-paste your code into your write-up or attach a proper print of code to your papers.
 - (-) Please write everything in a “portrait” style (not landscape).
 - (-) Please number problems according to numbers presented in the homework write-up that appears on the course page, NOT according to the numbers in the textbook.
 - (-) Please type your homework or hand-write it legibly (but yet attach a print of your codes to your handwritten stuff).
 - (-) Show that your code works (even if the problem doesn’t explicitly asks to test your code!)
-

- (1) Develop a Newton function (newton.m) and a secant function (secant.m) with the following structure:
(note: you can use your Newton implementation from the previous week as a base)

```
1 function [x,errf,errx,ratio,erra] = newton(fun,x0,xtol,ftol,verbose,a)
2 % newton      Newton's method to find a root f(x) = 0
3 %
4 % Synopsis:
5 % [x,errf,errx,ratio] = newton(fun,x0)
6 % [x,errf,errx,ratio] = newton(fun,x0,xtol)
7 % [x,errf,errx,ratio] = newton(fun,x0,xtol,ftol)
8 % [x,errf,errx,ratio] = newton(fun,x0,xtol,ftol,verbose)
9 % [x,errf,errx,ratio,erra] = newton(fun,x0,xtol,ftol,verbose,a)
10 %
11 % Input:
12 % fun        = (string) name of mfile that returns f(x) and f'(x).
13 % x0         = initial guess
14 % xtol       = (optional) absolute tolerance on x.      Default: xtol=5*eps
15 % ftol       = (optional) absolute tolerance on f(x).  Default: ftol=5*eps
16 % verbose    = (optional) flag.      Default: verbose=1, print table
17 % a         = (optional) exact root
18 %
19 % Output:
20 %      x = vector of approximations to the root
21 %      errf = vector of relative errors in f
22 %      errx = vector of relative approximate errors in x
23 %      ratio = ratio of approximate errors in x
24 %      erra = vector of relative errors in x if a is given
```

- (2) Consider the function

$$f(x) = x^3 + x - 1 \tag{1}$$

(A) Find the exact real root, α , (to machine precision) using built-in function `roots`

(B) Use α and the functions developed above to fill in the table for both methods

k	x_k	$f(x_k)$	$x_k - x_{k-1}$	$\alpha - x_k$	$\frac{ \alpha - x_k }{ \alpha - x_{k-1} }$
0	?	?		?	
1	?	?	?	?	?
2	?	?	?	?	?
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots

- Run until $|\alpha - x_k| < 1e - 11$

- use $x_0 = -.07$ for Newton's Method and $x_0 = 0.0$ and $x_1 = 1.0$ for the secant method

(C) From your data, determine the rate of convergence p so that $\frac{|\alpha - x_k|}{|\alpha - x_{k-1}|^p}$ is constant in each case. Show all of your work and verification for your data. *Hint: Consider $p = 2$ for Newton and $p = \frac{1+\sqrt{5}}{2}$ for Secant.*

(3) Use your Newton function to find the smallest root of

$$f(x) = -4.68999 + x(9.1389 + x(-5.56 + x)) \tag{2}$$

(A) Fill in the table above with $x_0 = 1.22$

(B) What do you observe about convergence for Newton's method in this case? Interpret the result.

(4) (NMM Chap. 7 #7) When describing computation as an abstract algorithmic level, it is convenient to use matrix and vector operations. When the algorithm is translated into actual code, the matrix and vector operations may or may not be used. In some cases they are very efficient, but in other cases they are not. The following matrix-matrix multiplications are not usually carried out exactly as written.

(a) Row scaling: $B = DA$, where D is a diagonal matrix

(b) Column scaling: $B = AD$, where D is a diagonal matrix

(c) Row permutation: $B = PA$, where P is a permutation matrix

(d) Column permutation: $B = AP$, where P is a permutation matrix

Describe a more efficient procedure for each of these operations. Compare the flop count for the matrix-matrix multiplication with the alternative procedure you recommend.

(5) (NMM Chap. 7 #10) Compare the efficiency of norm calculations in MATLAB with the following statements (note the use of commas, not semicolons, between the `tic`, `norm(u,2)`, and `toc`):

```
>> u = linspace(1,2,32000);
>> tic, norm(u,2), toc
>> tic, norm(u,1), toc
>> tic, norm(u,inf), toc
```

You should repeat these calculations for three times and report the average time for the last two runs.

(6) (NMM Chap. 7 #39) Given the matrices

$$A = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}, \text{ and } B = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ \sin(\theta) & -\cos(\theta) \end{bmatrix}$$

- (a) Compute (by hand) $A^T A$ and $B^T B$. What kind of matrices are A and B ?
- (b) For $\theta = \pi/4$ and $u = [1, 0]$, $u = [1, 1]$, $u = [0, 1]$, compute Au and Bu . What is the geometrical significance of these products? (*Hint*: Draw these vectors in the (x, y) -plane with tails at $(0, 0)$ and tips at (u_1, u_2) . Add the line through $(0, 0)$ and $(x, y) = (\cos(\theta/2), \sin(\theta/2))$.) You may want to use **myArrow** function to automate the drawing, get it from:
<http://www.me.pdx.edu/~gerry/nmm/mfiles/utills/myArrow.m>.