

## MS554 Assignment 4: Solving Non-linear and Linear Systems of Equations

- Assigned Oct. 11, 2001
- Due Sept. 18, 2001; Late work penalized 50%

### Solving Non-linear Equations

1. Think of a non-linear problem in one variable ( $f(x) = 0$ ) that you encounter in oceanography, sediment transport, or life in general. Graph the function for a range of  $x$ . Solve the non-linear problem using three of the following five methods: the bisection method, newton's method, secant method, false position, and fixed-point iteration. Use code that you write yourself to find the solutions; do not use a canned matlab or fortran routine. Solve the problem to within a relative error of  $1 \times 10^{-6}$ . Examine the rate at which each method converges.
2. Write your own routines (i.e. don't use canned matlab or fortran routines) to solve the problem  $e^x = 2 - \sin(2x)$ :
  - (a) Use bisection to find the smallest positive root. Use a graph to find good starting values. Continue the algorithm until the relative accuracy is  $1 \times 10^{-6}$ .
  - (b) Repeat, using the secant method. How many iterations were needed to reach a relative accuracy of  $1 \times 10^{-6}$ ?
  - (c) Repeat, using the method of false position. How many iterations were needed to reach a relative accuracy of  $1 \times 10^{-6}$ ?
  - (d) Repeat, using Newton's method. How many iterations were needed to reach a relative accuracy of  $1 \times 10^{-6}$ ?

### Solving sets of Linear Equations

3. Given the matrices:

$$A = \begin{pmatrix} 1 & -2 & 2 \\ 3 & 1 & 1 \\ 2 & 0 & 1 \end{pmatrix};$$

$$B = \begin{pmatrix} -1 & -2 & 4 \\ 1 & 3 & -5 \\ 2 & 4 & -7 \end{pmatrix};$$

$$C = \begin{pmatrix} 1 & 0 & 2 \\ 4 & 2 & -1 \\ 2 & 3 & 1 \end{pmatrix};$$

Do the following using a calculator (i.e. don't use canned matlab routines).

- (a) Verify that  $AI = A$ .
  - (b) Show that  $AB = BA = I$ .
  - (c) Show that  $AC \neq CA$ .
  - (d) Find lower- and upper-triangular matrices  $L$  and  $U$ , s.t.  $LU = A$ .
4. Verify the following operation counts for Gaussian elimination:
- (a) Number of additions (or multiplications) to compute new right hand side =  $n(n-1)/2$ .
  - (b) Number of divisions to compute the multipliers ( $l_{ik}$ ) =  $n(n-1)/2$ .
  - (c) Number of divisions in back substitution =  $n$ .
  - (d) Number of additions (or multiplications) in back substitution =  $n(n-1)/2$ .
5. Set up a tridiagonal square matrix that has 2's along the main diagonal and  $-1$  along the super- and subdiagonal. Start with a  $2 \times 2$  matrix. Use both matlab's Gaussian elimination routine, and it's inverse routine to solve  $Ax = b$ ; where  $b = [1 \ 0 \ 0 \ \dots \ 0 \ 0 \ 1]'$ . Keep doubling the size of the matrices  $A$  and  $B$  (to  $4 \times 4$ ,  $8 \times 8$ , etc.), and repeat. Keep track of the time that it takes to do the Gaussian elimination and the inverse (hint, use matlab's tic / toc routine:

```
>> tic;
>> x1=inv(A)*b;
>> time_inv(ii)=toc;
>> tic;
>> x2=A\b;
>> time_ge(ii)=toc;
```

Now, write a routine that does Gaussian elimination for a tri-diagonal matrix. Repeat the process, measuring the rate at which matlab can solve for  $x$ . Make a plot of computation speed vs. matrix dimension for the three methods.