1. (*35 points*) An Oceanographer does an experiment where temperature of a tide pool is measured every 5 minutes for 2 hours ($t = [0, 5, 10, \ldots, 115, 120]$). The data from the experiment is stored in **time.dat** and **temp.dat**.

please to go next page

(b) Let us think of our data as a set of points, (t_k, Y_k), where t measures time and Y measures temperature. Suppose we wanted to find an interpolating polynomial for this data. What is the lowest order polynomial which is guaranteed to go through all the points given in **time.dat** and **temp.dat**?

Since we have 25 data points, we can create a polynomial of degree 24

(c) Recall that a popular method of piecewise polynomial interpolation is to create a cubic spline $S_k(t)$ which is of the form

$$S_k(t) = S_{k,0} + S_{k,1}(t - t_k) + S_{k,2}(t - t_k)^2 + S_{k,3}(t - t_k)^3$$

for $t \in [t_k, t_{k+1}]$. To solve for the four coefficients $S_{k,j}$ we impose four constraints.

(i) What are these constraints?

$$\begin{aligned} S_k(t_k) &= Y_k \\ S_k(t_{k+1}) &= Y_{k+1} \\ S_k'(t_{k+1}) &= S_{k+1}'(t_{k+1}) \\ S_k''(t_{k+1}) &= S_{k+1}''(t_{k+1}) \end{aligned}$$

(ii) Why do we impose these particular constraints?

These constraints guarantee continuity of the function, as well as the first and second derivatives and results in a smooth function.

(d) Using **time.dat** and **temp.dat** write a matlab script that will fit, in the least squares sense, the function $Y(t) = A\sin(Bt) + Ct$ where Y is temperature which is a function of t, time (ie. solve for the coefficients A, B, and C).

You may need to write two m-files, please show both of them. (Don't forget to load your data!)

To solve for the coefficients we will call

```
coeff = fminsearch('E2',[0 0 0])
```

where we define the function E2 in a separate m-file which will look like:

```
function err = E2(C)

load time.dat
load temp.dat

err = sum(abs(C(1)*sin(C(2)*time) + C(3)*time - temp).^2);
```

2. (*35 points*) Consider the following set of equations:

$$\begin{aligned} x_1 + 2x_3 &= 2 \\ 2x_1 + 3x_2 + x_3 &= 7 \\ 5x_1 - 3x_2 + 2x_3 &= 29 \end{aligned}$$

(a) Write down the linear system that represents the equations above.

$$\begin{bmatrix} 1 & 0 & 3 \\ 2 & 3 & 1 \\ 5 & -3 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 7 \\ 29 \end{bmatrix}$$

(b) Write a matlab script that would solve the system using a Gauss-Seidel iteration method using a tolerance of 10^{-6}, a maximum number of iterations of 100, and an initial guess of $[1, 0, 1]^T$.

```
max_iter = 100;
tol = 10^(-6);
XO = [1 0 1]'; %old values

%loop through Gauss Seidel where XN is the vector of new values
for j = 1:max_iter
      XN(1) = 2 - 2*XO(3);
      XN(2) = (7 - 2*XN(1) - XO(3))/3;
      XN(3) = (29 - 5*XN(1) - 3*XN(2))/2;

      if norm(XN-XO,2) < tol
            break
      end
      XO = XN;

end
```

(c) Can we guarantee that if we used a Jacobi iteration method that our method would work? Why or why not? No. The matrix is not strictly diagonally dominant. For example in the third row

$$|2| \not> |5| + |-3|$$

In fact, none of the rows meet the criteria!

AMATH 301 - Summer 2008
Midterm Review

Midterm Review Solutions

Sample Problems

These are brief survey of the types of problems that may be on the midterm.

.

Exercise 1: Linear Systems

$$A = \begin{bmatrix} 1 & 5 & 3 \\ 2 & 1 & 0 \\ 0 & 3 & 4 \end{bmatrix} \qquad b_1 = \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix} b_2 = \begin{bmatrix} 10 \\ 5 \\ 7 \end{bmatrix}$$

(a) Perform a LU factorization on the matrix A by hand

Answer:

$$L_1 A = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 5 & 3 \\ 2 & 1 & 0 \\ 0 & 3 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 5 & 3 \\ 0 & -9 & -6 \\ 0 & 3 & 4 \end{bmatrix}$$

$$L_2(L_1 A) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \frac{1}{3} & 1 \end{bmatrix} \begin{bmatrix} 1 & 5 & 3 \\ 0 & -9 & -6 \\ 0 & 3 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 5 & 3 \\ 0 & -9 & -6 \\ 0 & 0 & 2 \end{bmatrix} \Rightarrow$$

$$A = (L_1^{-1} L_2^{-1}) U$$

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & -\frac{1}{3} & 1 \end{bmatrix} \begin{bmatrix} 1 & 5 & 3 \\ 0 & -9 & -6 \\ 0 & 0 & 2 \end{bmatrix}$$

(b) Solve the system $Ax = b_1$ and $Ax = b_2$ using your factorization

Answer: First solve $Ly = b$ and then $Ux = y$:

For b_1:

$$Ly = b_1 \Rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & -\frac{1}{3} & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix} \Rightarrow y = \begin{bmatrix} -1 \\ 3 \\ 2 \end{bmatrix}$$

$$Ux = y \Rightarrow \begin{bmatrix} 1 & 5 & 3 \\ 0 & -9 & -6 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ 3 \\ 2 \end{bmatrix} \Rightarrow x = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}$$

For b_2:

$$Ly = b_2 \Rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & -\frac{1}{3} & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 10 \\ 5 \\ 7 \end{bmatrix} \Rightarrow y = \begin{bmatrix} 10 \\ -15 \\ 2 \end{bmatrix}$$

$$Ux = y \Rightarrow \begin{bmatrix} 1 & 5 & 3 \\ 0 & -9 & -6 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 10 \\ -15 \\ 2 \end{bmatrix} \Rightarrow x = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}$$

(c) Write Matlab code that would solve the system $Ax = b_1$ using a Jacobi iteration technique and a tolerance of 10^{-6}

Answer: First we need to derive the iterative scheme, solve each of the equations for each of the unknowns:

$$\begin{aligned} x_1 + 5x_2 + 3x_3 &= -1 \\ 2x_1 + x_2 &= 1 \\ 3x_2 + 4x_3 &= 1 \end{aligned} \quad \Rightarrow \quad \begin{aligned} x_1 &= -5x_2 - 3x_3 - 1 \\ x_2 &= 1 - 2x_1 \\ x_3 &= \frac{1 - 3x_2}{4} \end{aligned}$$

```
clear all;
TOLERANCE = 10^(-6);

old = [0 0 0]';
for i=1:1000
        new(1) = -5*old(2) - 3 * old(3) - 1;
        new(2) = 1 - 2 *old(1);
        new(3) = (1-3*old(2))/4;

        if(norm(new-old)<TOLERANCE)
                break;
        end
        old = new;
end
```

(d) Write Matlab code that would solve the system $Ax = b_1$ using a Gauss-Seidel iteration technique and a tolerance of 10^{-6}

Answer: Again we use the same scheme but update slightly differently with the updated values

```
clear all;
TOLERANCE = 10^(-6);

old = [0 0 0]';
for i=1:1000
        new(1) = -5*old(2) - 3 * old(3) - 1;
        new(2) = 1 - 2 *new(1);
```

```
        new(3) = (1-3*new(2))/4;

        if(norm(new-old)<TOLERANCE)
                break;
        end
        old = new;
end
```

(e) How should we modify the matrix A to ensure that our Jacobi iteration code would converge?

Answer: Right now our matrix is not strictly diagonally dominant due to the first and second rows:

$$|5| + |3| > |1|$$
$$|2| > |1|$$

If we flip the first and second row however, we find that the rows are now strictly diagonally dominant.

$$A = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 5 & 3 \\ 0 & 3 & 4 \end{bmatrix} \Rightarrow$$

$$|1| < |2|$$
$$|1| + |3| < |5|$$

Exercise 2: Curve Fitting

Assume you have a file `depth.dat` that contains a cross-section of the depth of the puget sound. The file contains two columns of data, the first is the coordinate x, the second is the corresponding depth at that point.

(a) Setup the linear system that we would need to solve in order to fit the function $f(x) = Ax^2 + Bx + C$ to the data

Answer: We want to minimize the E_2 error $\sqrt{\frac{1}{N}\sum_{i=1}^{N}|f(x_i) - y_i|^2}$. In order to do this we need to take the derivative with respect to each coefficient A, B, and C. We can simplify our calculation by only considering the inner sum: $\sum_{i=1}^{N} |f(x_i) - y_i|^2$

$$\frac{\partial}{\partial A}\sum_{i=1}^{N} |Ax_i^2 + Bx_i + C - y_i|^2 = \sum_{i=1}^{N} 2(Ax_i^2 + Bx_i + C - y_i)x_i^2 = 0 \Rightarrow$$
$$\Rightarrow \sum_{i=1}^{N} Ax_i^4 + Bx_i^3 + Cx_i^2 = \sum_{i=1}^{N} y_i \cdot x_i^2$$
$$\frac{\partial}{\partial B}\sum_{i=1}^{N} |Ax_i^2 + Bx_i + C - y_i|^2 = \sum_{i=1}^{N} 2(Ax_i^2 + Bx_i + C - y_i)x_i = 0$$
$$\Rightarrow \sum_{i=1}^{N} Ax_i^3 + Bx_i^2 + Cx_i = \sum_{i=1}^{N} y_i \cdot x_i$$
$$\frac{\partial}{\partial C}\sum_{i=1}^{N} |Ax_i^2 + Bx_i + C - y_i|^2 = \sum_{i=1}^{N} 2(Ax_i^2 + Bx_i + C - y_i) = 0 \Rightarrow$$
$$\Rightarrow \sum_{i=1}^{N} Ax_i^2 + Bx_i + C = \sum_{i=1}^{N} y_i$$

We can then write this system of equations as a linear system with the unknowns the vector $[A, B, C]^T$

$$\begin{bmatrix} \sum_{i=1}^{N} x_i^4 & \sum_{i=1}^{N} x_i^3 & \sum_{i=1}^{N} x_i^2 \\ \sum_{i=1}^{N} x_i^3 & \sum_{i=1}^{N} x_i^2 & \sum_{i=1}^{N} x_i \\ \sum_{i=1}^{N} x_i^2 & \sum_{i=1}^{N} x_i & N \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{N} y_i \cdot x_i^2 \\ \sum_{i=1}^{N} y_i \cdot x_i \\ \sum_{i=1}^{N} y_i \end{bmatrix}$$

(b) Write a Matlab script that will read in the data and fit the function $f(x) = A\exp(-Bx^2)$ to the data using a least squares approach

Answer: We will use `fminsearch` to find the nonlinear minimum.

The call to `fminsearch` is:

```
C = fminsearch('exp_error',[0 0])
```

The function `exp_error` should look something like:

```
function E2 = exp_error(C)

load depth.dat
x = depth(:,1)
y = depth(:,2)

E2 = sum( abs( C(1) * exp(-C(2)*x.^2) - y).^2)
```