Assigned: Tuesday Oct. 14, 2011
Due Date: Tuesday Oct. 21, 2011

MATH 174/274: Homework IV
"Linear Algebra (Part II)"
Fall 2011

1. Consider the following linear system,

$$A\vec{x} = \vec{b}$$

where A is the following matrix,

$$A = \begin{pmatrix} 0 & 1 & 4 & 5 \\ 2 & 0 & 2 & 4 \\ 2 & 4 & 0 & 1 \\ 1 & -3 & -5 & 0 \end{pmatrix}.$$

(a) Determine the P, L, U decomposition of the matrix A, such that $PA = LU$. (Show **EACH STEP** in the decomposition.)

(b) Use the P, L, U decomposition found in (a) to find the solution to $A\vec{x} = \begin{pmatrix} 2 \\ -2 \\ 0 \\ 1 \end{pmatrix}$

(Show **ALL** relevant steps).

(c) Use the P, L, U decomposition found in (a) to find the solution to $A\vec{x} = \begin{pmatrix} 0 \\ 1 \\ 5 \\ 2 \end{pmatrix}$

(Show **ALL** relevant steps).

2. **SOURCE CODE:**

Write separate MATLAB functions for **LU decomposition with partial pivoting**, **forward substitution**, and **backward substitution**. I recommend using the pseudocode provided in the lecture notes. These functions should be written so that they can be used to solve $A\vec{x} = \vec{b}$ in the following way:

- LU decomposition:
  `[L,U,P]=LUdecomp(A,m)`

- Solve $L\vec{y} = P\vec{b}$:
  `y = ForwardSubs(L,P*b,m)`
- Solve $U\vec{x} = \vec{y}$:
  `x = BackwardSubs(U,y,m)`

CAUTION: If you type something like:

```
for i=1:10
   x(i) = i;
end,
```

MATLAB will create a *row vector* of length 1×10. We always want a *column vector*, in which case you should type:

```
for i=1:10
   x(i,1) = i;
end.
```

This will create a vector of length 10×1.

3. **DISCUSSION QUESTION:**

For the linear system

$$\begin{aligned} -x_3 + 3x_4 &= 0 \\ -x_1 + 2x_2 - x_3 &= -1 \\ 3x_1 - x_2 &= 0 \\ -x_2 + 2x_3 - x_4 &= 1 \end{aligned}$$

do all of the following:

- Solve this system using the `x = A\b` command in MATLAB.
- Use your LU decomposition with forward/backward substitution to solve this problem.
- Make sure that the answers agree.

4. **APPLICATION PROBLEM:**

In mathematical biology, integral equations of the form shown below arise when modeling the population of an invading organisms into an ecosystem (Wang, Kot, & Neubert, 2002):

$$N(x-c) - \lambda \int_0^\infty K(x,y)N(y)\,dy = f(x)\,.$$

The organism being modeled by the above equation has both a growth and dispersal stage. $N(x)$ is the population of the invading species at spatial location x, $K(x,y)$

is the probability density function for dispersal from a source y, and $f(x)$ is related to the growth rate of the population.

We will attempt to approximately solve the above integral equation for the population $N(x)$ using numerical linear algebra. Consider an example where $c = 0$, $\lambda = 5$, and

$$K(x,y) = \frac{1}{\pi + \pi\big(x-y\big)^2}$$

$$f(x) = \frac{1}{2} - \frac{1}{\pi}\arctan(x)\,.$$

We define the following quantities:

$$\begin{aligned}
m &= \text{“The number of points in our approximation”}\\
D &= 70\\
h &= D/(m-1)\\
x_i &= -7 + (i-1)\cdot(h/5)\\
y_j &= 0 + (j-1)\cdot h\\
f_i &= 1/2 - (1/\pi)\arctan(x_i)\\
N_i &\approx N(x_i) = \text{“approximation to } N(x)\text{”}\\
K_{ij} &= K(x_i, y_j)\\
I &= \text{“the } m\times m \text{ identity matrix”}\,.
\end{aligned}$$

We replace the integral by a Riemann sum (we will talk about this in much more detail later in the semester):

$$\int_0^\infty K(x_i,y)N(y)\,dy \approx \int_0^D K(x_i,y)N(y)\,dy \approx h\left[\frac{1}{2}K_{i1}N_1 + \sum_{j=2}^{m-1} K_{ij}N_j + \frac{1}{2}K_{im}N_m\right].$$

The original integral equation can now be re-written as a system of linear equations:

$$A\vec{N} = \vec{f}\,,$$

where

$$\vec{N} = \begin{bmatrix} N_1\\ N_2\\ \vdots\\ N_m\end{bmatrix}, \quad \vec{f} = \begin{bmatrix} f_1\\ f_2\\ \vdots\\ f_m\end{bmatrix}, \quad A = I - h\lambda\begin{bmatrix} \frac{1}{2}K_{11} & K_{12} & K_{13} & \cdots & K_{1,m-1} & \frac{1}{2}K_{1m}\\ \frac{1}{2}K_{21} & K_{22} & K_{23} & \cdots & K_{2,m-1} & \frac{1}{2}K_{2m}\\ \vdots & \vdots & \vdots & & \vdots & \vdots\\ \frac{1}{2}K_{m1} & K_{m2} & K_{m3} & \cdots & K_{m,m-1} & \frac{1}{2}K_{mm}\end{bmatrix}.$$

Do all of the following:

- Write a MATLAB m-file that for a given m creates $\vec{x}$, $\vec{y}$, $\vec{f}$, matrix K, and matrix A.
- Inside this script make a call to your `LUdecomp.m`, `ForwardSubs.m`, and `BackwardSubs.m` functions in order to solve for $\vec{N}$.

- Immediately **before** your call to `LUdecomp.m`, place the following line into your script:

```
t0 = cputime;
```

- Immediately **after** your call to `BackwardSubs.m`, place the following line into your script:

```
tf = cputime - t0;
```

  The variable `tf` tells you how much CPU time in seconds was spent solving the linear system of equations.

Using your script, answer all of the following questions:

(a) Run your code with $m = 80$. Plot $\vec{N}$ vs. $\vec{x}$.

(b) Run your code with $m = 10, 20, 40, 80, 160$, and 320 and fill out the table:

m	**CPU Time (seconds)**	**CPU Time ratio (curr/prev)**
$m = 10$		XXXXXXXXXXXXXXXXXXXX
$m = 20$		
$m = 40$		
$m = 80$		
$m = 160$		
$m = 320$		

(c) Theoretically, what should this ratio be going to as $m \to \infty$? Explain your answer.

(d) Do the results in the above table agree with this prediction? Explain your answer.