Instructor: Emre Mengi

Math 304 (Spring 2010)

Study Guide for Weeks 1-2
Before attempting the homework please make sure that you have understood the topics listed below. Don't hesitate to ask for help if any of these topics is unclear. The related sections from textbooks are provided in parenthesis. Fausett refers to the main textbook "Applied Numerical Analysis Using Matlab by Laurene V. Fausett". Watkins refers to the supplementary book "Fundamentals of Matrix Computations by David S. Watkins". The relevant chapters of both books can be accessed through library. Visit the website `http://www.libunix.ku.edu.tr`, set the search criterion to "reserves by course" and type "Math 304".

- Some applications where systems of linear equations arise; in particular in class we talked about mass-spring systems and electrical circuits so far. (Watkins - section 1.2, pages 13-18)
- The inverse of a square matrix; you should know the definition and the following characterizations of the invertible matrices. (Below A is an $n \times n$ matrix.)
  - A is invertible $\iff$ the columns of A are linearly independent.
  - A is invertible $\iff$ $Ax \neq 0$ for each nonzero $x \in \mathbb{R}^n$.
  - A is invertible $\iff$ $\det(A) \neq 0$.

  (Watkins - section 1.2, pages 12-13)
- Let A be an $n \times n$ matrix and $b \in \mathbb{R}^n$. The solution to the linear system $Ax = b$ is unique $\iff$ A is invertible. (Watkins - section 1.2, pages 12-13)
- Definitions of a lower and an upper triangular matrix (Watkins - section 1.3, page 23)
- A square lower or upper triangular matrix is invertible if and only if all of its diagonal entries are nonzero. (Watkins - section 1.3, page 23)
- Forward substitution to solve a lower triangular system (Watkins - section 1.3, pages 24-26)
- Back substitution to solve an upper triangular system (Watkins - section 1.3, page 29)
- Basic Gaussian elimination (Fausett - Section 3.1.1, pages 98-106)

Homework 1 (due on March 5th, Friday by 4pm)

In Matlab questions (questions **4** and **5**) attach the Matlab outputs. Also in question **5.(c)** include a print-out of your Matlab code for back substitution.

1. Consider the following matrices

$$A_1 = \begin{bmatrix} 3 & -1 & 2 \\ 0 & 1 & -1 \\ 0 & 0 & 2 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 & -1 & 0 \\ 1 & 1 & 2 \\ 1 & -1 & 0 \end{bmatrix}, \quad A_3 = \begin{bmatrix} 3 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 2 & 2 \end{bmatrix}, \quad \text{and} \quad A_4 = \begin{bmatrix} 3 & 0 & 0 \\ 2 & 1 & 0 \\ 1 & 2 & 2 \end{bmatrix}.$$

(a) Which of the matrices A_1, A_2, A_3 and A_4 are invertible? Which of them are singular? Justify your answer.

(b) Given $b = \begin{bmatrix} -6 & -3 & 2 \end{bmatrix}^T$. For each of the linear systems $A_1x = b$, $A_2x = b$, $A_3x = b$ and $A_4x = b$ indicate whether the solution to the linear system is unique or not. If the solution is unique, calculate the solution by hand as well.

2. Suppose A is an $n \times n$ matrix and $b \in \mathbb{R}^n$. Show that if $Ax = b$ has a unique solution, then the matrix A is invertible. (Hint : Suppose $Ax = b$ has a unique solution and consider a $y \in \mathbb{R}^n$ such that $Ay = 0$. Deduce that $y = 0$.)

3. Solve the following linear systems using basic Gaussian elimination. (Second example below was taken from Fausett - P3.2 on page 127.)

$$\begin{bmatrix} 1 & -1 & 0 \\ 2 & 1 & -1 \\ 0 & 2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ -1 \\ 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 2 & 5 & 3 \\ 8 & 8 & 6 \\ 4 & 6 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 14 \\ 3 \end{bmatrix}$$

4. Solve either **Exercise 1.2.19** or **Exercise 1.2.20** in Watkins' book whichever one you prefer. You should use Matlab to solve the linear system in part **(b)**. If you don't know how you can enter matrices and/or vectors in Matlab, please see **5.(d)** below. To solve the linear system $Ax = b$ using Matlab's built-in routine you need to type `x = A\b`.

5. Recall that back substitution (see the subsection **Upper-Triangular Systems** on page 29 in Watkins' book) is the standard approach to solve the upper triangular linear system

$$\underbrace{\begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1(n-1)} & u_{1n} \\ 0 & u_{22} & & u_{2(n-1)} & u_{2n} \\ \vdots & & \ddots & & \vdots \\ 0 & & & u_{(n-1)(n-1)} & u_{(n-1)n} \\ 0 & & & 0 & u_{nn} \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{(n-1)} \\ x_n \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{(n-1)} \\ b_n \end{bmatrix}}_{b}.$$

(a) Write down a pseudocode for back substitution in the spirit of the pseudocode provided for forward substitution in class (or see the pseudocode (1.3.5) on page 25 in Watkins' book).

(b) Count the number of *flops* required by your pseudocode in part **(a)**. Your answer must be in terms of n, which is the size of the system $Ux = b$.

(c) Write a Matlab function that is an implementation of your pseudocode for back substitution in part **(a)**. Your matlab function must take two input parameters; an $n \times n$ (upper triangular) matrix U and a right-hand vector $b \in \mathbb{R}^n$. It should return one output parameter $x \in \mathbb{R}^n$, which is the solution of $Ux = b$. Your Matlab code should look like

```
function x = backsubstitute(U,b)

n = length(b);

....

return;
```

In the above code you need to fill in the part in between the statements `n = length(b);` and `return;`. Name your file as `backsubstitute.m`.

(d) Test your implementation in part **(c)** with the following upper triangular system

$$\begin{bmatrix} -1 & 3 & 1 & 2 \\ 0 & -1 & 3 & 1 \\ 0 & 0 & -1 & 3 \\ 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} -9 \\ -3 \\ 5 \\ -1 \end{bmatrix}.$$

You can enter the coefficient matrix U above in Matlab by typing

```
U = [-1 3 1 2; 0 -1 3 1; 0 0 -1 3; 0 0 0 -1]
```

and the right-hand vector by typing

```
b = [-9; -3; 5; -1]
```

Compare the answer computed by your implementation with the answer returned by the built-in linear system solver in Matlab. You can call your implementation by typing `x = backsubstitute(U,b)` after entering U and b as suggested above. To use the built-in linear system solver in Matlab type `x = U\b`.