

Mathematics 550 MATLAB Project 1

Gaussian Elimination, LU Factorization, and Solving Linear Systems

Fall 2006

In this lab you will use MATLAB to study the following topics:

- Gaussian elimination and reduced row echelon form of a matrix.
- The LU (lower-upper triangular) factorization of a matrix
- Using the LU factorization to solve $A\mathbf{x} = \mathbf{b}$.
- The *null space* $\text{Null}(A)$ of a matrix A .
- Particular solutions and complete solution to an inhomogeneous linear equation $A\mathbf{x} = \mathbf{b}$.

Diary Files: Before doing this assignment, please read the document [Notes on Matlab Assignments](#). It describes how to record the results of your MATLAB session in a *diary file*, how to customize the random number generator, how to insert comments into your work, and then how to edit this file to create your Lab report.

Tcodes: A special set of MATLAB *m-files* called *Tcodes* has been written to accompany Strang's textbook. To obtain any of these files, use a web browser (such as Netscape) and go to the Math Department Home page <http://www.math.rutgers.edu>. Click on *course materials*, then on *Math 550A Linear Algebra and Applications*, and then on *MATLAB Teaching Codes*. You will see a directory of the *m-files*. Click on the particular *m-file* that you need. Then in the menu bar click on *Files* and *Save As*. Fill in the directory information that is requested.

For this lab you will need the Teaching Codes `nulbasis.m`, `partic.m`. Before beginning work on the Lab questions you should copy these codes to your workspace.

Script Files: When you need to write a script file for this lab and subsequent labs, use the following procedure: Start MATLAB and click on *File*, then *New*. Move the pointer to the right and click on *m-Files*. This will open the MATLAB Editor/Debugger Window. Type the script commands in this window, then click on *File* in the Editor/Debugger toolbar and save your script on in your directory.

After you have created and saved an *m-file*, you must set the *Path* so that MATLAB can find this file. Click on *File* and then *Set Path* and follow the directions to add your directory to the list of path names.

Script Files for this Lab: Use the text editor in MATLAB to create the following MATLAB *function* *m-files*:

(a) Create a *function* *m-file* with the commands

```
function v = rvect(m)
v = fix(10*rand(m,1));
```

(note the semicolon on the end of the second line). Save this file under the name `rvect.m` (be sure that you have set the *Path* as described above so that MATLAB can find this *m-file*). Test the file by clicking on the MATLAB window and typing `v = rvect(3)` at the MATLAB prompt. You should get a column vector $\mathbf{v} \in \mathbb{R}^3$ with entries that are (random) integers between 0 and 9. Now type `u = rvect(3)` You will get another random column vector $\mathbf{u} \in \mathbb{R}^3$. Type `v` at the prompt. You should get the *same* vector \mathbf{v} as before.

Note that the name \mathbf{v} in the `rvect` function file is a *local variable*; you can assign any name to the output. If you have already defined a vector \mathbf{v} in your work space, it is not changed when you generate \mathbf{u} by `rvect`.

(b) Create another function m-file with the commands

```
function A = rmat(m,n)
A = fix(10*rand(m,n));
```

(note the semicolon on the end of the second line). Save this file under the name `rmat.m`. Then test the file by clicking on the MATLAB window and typing `A = rmat(3, 5)` at the MATLAB prompt. You should get a 3×5 matrix A with entries that are (random) integers between 0 and 9.

Random Seed: When you start your MATLAB session, initialize the random number generator by typing

```
rand('seed', abcd)
```

where `abcd` are the last four digits of your Student ID number. This will ensure that you generate your own particular random vectors and matrices.

BE SURE TO INCLUDE THIS LINE IN YOUR LAB WRITE-UP

The lab report that you hand in must be your own work. The following problems all use randomly generated matrices and vectors, so the matrices and vectors in your lab report will not be the same as those of other students doing the lab. Sharing of lab report files is not allowed in this course.

Question 1. Gaussian Elimination and Reduced Row Echelon Form

The most basic algorithm in linear algebra is Gaussian elimination. In this question you will use MATLAB to carry out this algorithm and obtain the reduced row-echelon form of a matrix. Before starting work on this question, type `rrefmovie` at the MATLAB prompt. You will see a step-by-step example of the row operations that transform a matrix A into its reduced row echelon form $R = \text{rref}(A)$. In this demonstration, each pivot is chosen to be the *largest* in its column (for numerical stability), so extra row interchanges are used. Since $\text{rref}(A)$ is *uniquely* determined by A , this does not affect the final answer. (Do not include this part in your lab write-up.)

(a) Generate a 3×4 matrix $A = \text{rmat}(3, 4)$. Now use MATLAB to step through the transformation of A into its **row reduced echelon form** R . Start with $R = A$ and normalize the first row of R to get $R(1, 1) = 1$:

```
R = A; R(1, :) = R(1, :)/R(1, 1)
```

(note the use of the colon to operate on whole rows of the matrix). If your random matrix happens to have $A(1, 1) = 0$, then interchange rows to get a nonzero entry in the $(1, 1)$ position before doing the calculation above. Next subtract a multiple of the first row of R from the second row to make $R(2, 1) = 0$:

```
R(2, :) = R(2, :) - R(2, 1)*R(1, :)
```

Repeat this procedure to make $R(3, 1) = 0$:

```
R(3, :) = R(3, :) - R(3, 1)*R(1, :)
```

(you can minimize typing by using the up-arrow key and editing the command line). The first column of your matrix R should now be the same as the first column of the desired result.

(b) Operate on R by the same method as in (a) to obtain the second column of $\text{rref}(A)$. First normalize row 2 of R , then subtract multiples of row 2 from rows 1 and 3 to put zeros in the $(1, 2)$ and $(3, 2)$ positions (you can minimize typing by using the up-arrow key and editing the commands used in part (a)).

(c) Operate on R by the same method as in (b) to obtain the third column of `rref(A)`. First normalize row 3 of R , then subtract multiples of row 3 from rows 1 and 2 to put zeros in the (1, 3) and (2, 3) positions.

(d) Your matrix R should now have been transformed into the desired form. (since A is a random 3×4 matrix, the result is (almost) sure to have rank 3). MATLAB has a single command that produces this result. Give the command `rref(A)` and compare your result to R . If the MATLAB answer is not the same as the current value of your R , go back and redo your calculations.

Question 2. Row Operations and LU Factorization

In this problem you will use MATLAB to step through the elementary row operations to obtain the matrix factorization $A = LU$ for a square 3×3 matrix A . Note how the entries of L are obtained from the current columns and used to define the row operations to modify U .

(a) Generate a random 3×3 matrix :

```
A = rmat(3,3), L = eye(3), U = A
```

Here U has the initial value A , but at the end of the LU algorithm U will be upper triangular.

(b) Use the MATLAB editor to create an m-file called `col1.m` with the following MATLAB commands:

```
v = U(2,1)/U(1,1);
L(2,1) = v;
U(2,:) = U(2,:) - v*U(1,:);
v = U(3,1)/U(1,1);
L(3,1) = v;
U(3,:) = U(3,:) - v*U(1,:);
```

(notice the use of ; to suppress screen output of the intermediate results). Execute this file by typing `col1` at the MATLAB prompt. If you get a division by zero error message, go back to step (a) and generate another A and U .

Then, use the MATLAB editor to create an m-file called `col2.m` with the commands

```
v = U(3,2)/U(2,2);
L(3,2) = v;
U(3,:) = U(3,:) - v*U(2,:);
```

This will be used to continue the modification of U . Execute this file by typing `col2` at the MATLAB prompt. If you get a division by zero error message, go back to step (a) and start again (this is very unlikely to happen since A is completely random).

(c) Use MATLAB to verify that $A = L * U$ (where U has the value from (c)). Describe in words how the entries of L and U were obtained from A .

Question 3. Using the LU Factorization to Solve $Ax = b$

Although we often write $x = A^{-1}b$ for the solution of $Ax = b$, the inverse of A should **never** be used for this purpose. MATLAB has a **left division** operator, denoted `\` that is used to solve equations efficiently.

(a) Solving $Ax = b$ using L and U (See Equation (8) on page 35 of Strang's text): Generate a random vector $b = \text{rvech}(3)$. Calculate the solution

$$c = L \backslash b$$

to the triangular system $Lc = b$. Then calculate the solution

$$x = U \backslash c$$

to the triangular system $Ux = c$. Finally, calculate Ax and check that it is b (since the entries in b are integers, this should be obvious by inspection).

(b) Comparing the speed of LU versus rref

MATLAB has a command `lu` that produces the LU factorization without forcing the user to step through the calculation. This, together with `rref` will be used to compare the speed of these operations.

You can compare the speed of two methods of solving the equation $Ax = b$ when A is an invertible square matrix by using the MATLAB `tic` and `toc` commands.

Caution: Be sure to use the semicolon ";" as indicated in the following commands so that the matrices and vectors are **not** displayed or included in your diary file. Do not print or include these large matrices in your lab write-up.

Generate a random 100×100 matrix A , a vector $b \in \mathbb{R}^{100}$, and calculate the LU decomposition of A by

$$A = \text{rand}(100) ; \quad b = \text{rand}(100,1) ; \quad [L \ U] = \text{lu}(A) ;$$

First, solve $Ax = b$ by using RREF (Gaussian elimination). The last column y of the augmented matrix $R = \text{rref}([A \ b])$ satisfies $Ay = b$ because $\text{rref}(A)$ is the identity matrix if A is a random square matrix.

$$\text{tic} ; \quad R = \text{rref}([A \ b]) ; \quad y = R(:,101) ; \quad \text{toc}$$

Define the number `rref_time` to be the `elapsed_time` given by the MATLAB output.

Now solve $Ax = b$ by using the LU decomposition of A :

$$\text{tic} ; \quad c = L \backslash b ; \quad x = U \backslash c ; \quad \text{toc}$$

Define the number `lutime` to be the `elapsed_time` given by the MATLAB output.

Check that the two solutions are the same (up to round-off error) by calculating `norm(x - y)`.

For an $n \times n$ system, the number of floating point arithmetic operations (flops) needed for Gaussian elimination is approximately $2n^3/3$, while the the number of flops for the LU method (after the L , U factors are already calculated) is approximately $2n^2$ (solving two triangular systems). Compare the ratio `rref_time/lutime` that you observed with the theoretical flops ratio for the two methods when $n = 100$.

Question 4. Null Space

Generate a *partly random* 3×6 matrix A and its reduced row echelon form R by

$$B = \text{rmat}(3, 2) ; \quad C = \text{rmat}(3,2) ; \quad A = [B, 3*B, C], \quad R = \text{rref}(A)$$

(a) Let V be the subspace of \mathbb{R}^6 given by the homogeneous system of equations $Ax = 0$ (the *null space* of A).

(i) In the system of linear equations $Ax = 0$ (where $x \in \mathbb{R}^6$), what are the *free variables*?

(ii) What is $\dim V$?

(b) Use the MATLAB Teaching Code `nulbasis.m` to calculate the *special solutions* to the system of equations $A\mathbf{x} = 0$:

```
N = nulbasis(A)
```

The columns of N are the solutions to $A\mathbf{x} = 0$ obtained by setting one free variable to 1 and all the other free variables to 0. Define

```
v1 = N(:,1), v2 = N(:,2), v3 = N(:,3)
```

(Notice that each \mathbf{v}_i is a 6-component *vector*, not a scalar.)

(i) Which component of \mathbf{v}_1 *must be* 1, for *every* random B and C ?

(ii) Which components of \mathbf{v}_1 *must be* 0, for *every* random B and C ?

Answer questions (i) and (ii) also for \mathbf{v}_2 and \mathbf{v}_3 . Check by MATLAB that these three vectors are all in $\text{Null}(A)$.

(c) Now generate a random linear combination \mathbf{x} of the vectors \mathbf{v}_1 , \mathbf{v}_2 , and \mathbf{v}_3 by

```
x = rand(1)*v1 + rand(1)*v2 + rand(1)*v3
```

(Each occurrence of `rand(1)` generates a different random coefficient). Check by MATLAB that $A\mathbf{x} = 0$.

(i) Explain (without MATLAB) why \mathbf{x} also satisfies $R\mathbf{x} = 0$. Then confirm this by MATLAB.

Question 5. Solving a system

Let A be a matrix of size $m \times n$. The linear system $A\mathbf{x} = \mathbf{b}$ is called *underdetermined* if $m < n$ (more variables than equations). It is *overdetermined* if $m > n$ (more equations than variables). In both cases the matrix A is *not* square, so the system can never be solved by finding an inverse matrix for A .

(a) Particular Solution (underdetermined system): Generate a random 3×5 matrix A (the coefficient matrix for an *underdetermined* system of 3 equations in 5 unknowns) and its reduced row echelon form R by

```
A = rmat(3,5), R = rref(A)
```

Now generate a random 3×1 vector \mathbf{b} and use the Teaching Code `partic.m` to find a particular solution to $A\mathbf{x} = \mathbf{b}$ by

```
b = rmat(3,1), x = partic(A, b)
```

This is the solution with all the free variables set to zero. Check with MATLAB that $A\mathbf{x} = \mathbf{b}$. Repeat for another random vector \mathbf{b} , using the same matrix A .

(i) What entries in \mathbf{x} are zero both times?

(ii) Which columns of R correspond to free variables?

Calculate `rank(A)` and `rank([A, b])` using MATLAB.

(iii) Does the equation $A\mathbf{x} = \mathbf{b}$ have a solution for *every* vector \mathbf{b} in this case? Why?

(b) Particular Solution (overdetermined system): Generate a random 5×3 matrix $A = \text{rmat}(5, 3)$ (the coefficient matrix for an *overdetermined* system of 5 equations in 3 variables). The following MATLAB command will generate a random 5×1 vector \mathbf{b} and try to find a particular solution to $A\mathbf{x} = \mathbf{b}$:

```
b = rmat(5,1), x = partic(A, b)
```

Calculate `rank(A)` and `rank([A, b])` using MATLAB.

(i) Why is there no solution to $A\mathbf{x} = \mathbf{b}$ for a completely random choice of \mathbf{b} ?

Now use the (partly) random vector

```
b = rand(1)*A(:,1) + rand(1)*A(:,2) + rand(1)*A(:,3)
```

and calculate $\mathbf{x} = \text{partic}(A, \mathbf{b})$. Write comments to answer the following:

(ii) Explain why the special form of the vector \mathbf{b} guarantees that there always is a solution \mathbf{x} for any choice of the random coefficients.

(iii) Since there is a solution to $A\mathbf{x} = \mathbf{b}$ for this \mathbf{b} , what is the rank of the augmented matrix $[A, \mathbf{b}]$? Check your answer to (iii) using MATLAB.

(c) General Solution Execute the commands

```
A = rmat(3,5), b = rmat(3,1), N = nulbasis(A), xp = partic(A,b)
```

Set $\mathbf{v}_1 = N(:,1)$, $\mathbf{v}_2 = N(:,2)$ and form a random *general solution*

```
x = xp + rand(1)*v1 + rand(1)*v2
```

to $A\mathbf{x} = \mathbf{b}$. Check by MATLAB that $A\mathbf{x} - \mathbf{b} = 0$.

(d) Now solve the equation $A\mathbf{x} = \mathbf{b}$ with the extra condition that \mathbf{x} should be of the form

$$\mathbf{x} = [x_1, x_2, x_3, -9, 8]^T$$

For this, you must choose particular scalars c_1 and c_2 so that $\mathbf{x} = \mathbf{x}_p + c_1\mathbf{v}_1 + c_2\mathbf{v}_2$. (**Hint:** Look at the free variables in \mathbf{x} , \mathbf{x}_p , \mathbf{v}_1 , and \mathbf{v}_2). Then check your answer by calculating $A\mathbf{x} - \mathbf{b}$ with MATLAB.

Final Editing of Lab Write-up:

After you have worked through all the parts of the lab assignment, edit your diary file. Include the MATLAB calculations, but remove errors that you made in entering commands and remove other material that is not directly related to the questions.

End of Lab 1