Vectors and Matrices

To define a vector $u = (1, 2)$ or $v = \begin{pmatrix} 3 \\ 4 \end{pmatrix}$, type

> >> $u = [1, 2]$ or $u = [1\ \ 2]$
> >> $v = [3; 4]$

Defining a matrix is similar. A comma separates terms in a row and a semicolon separates rows. For instance, in order to define the matrix $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$, type

> >> $A = [1, 2; 3, 4]$ or >> $A = [1\ \ 2; 3\ \ 4]$ (omit the commas using only spaces)

Operations with Matrices

> >> *transpose*(A), >> *inv*(A) (gives the matrix inverse)

We can multiply matrices in MATLAB with simple commands like

> >> A*x+7*b

For example, find $Ax + 7b$ when

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \qquad x = \begin{bmatrix} -5 \\ 1 \end{bmatrix}, \qquad b = \begin{bmatrix} 6 \\ 7 \end{bmatrix}$$

Sometimes you will need or want to do operations on vectors or matrices componentwise

> >> C=A.*B this command gives a new matrix C, where the each entry in C is obtained from the product of the entries in A and B.

For example, if

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \qquad B = \begin{bmatrix} -2 & 3 \\ 9 & 0 \end{bmatrix} \xrightarrow[\text{the command}]{A.*B} C = \begin{bmatrix} -2 & 6 \\ 27 & 0 \end{bmatrix}$$

Getting Help Matlab

To get info on how to use a function/command type in

> >> Help command

Q1. 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.

! >> R=rref(A)
the row operations that transform a matrix A into its Reduced Row Echelon Form

1 Generate a 3×4 integer matrix $A = rmat(3,4)$. Now use Matlab to transform A into $R = rref(A)$. Start with $R = A$ and normalize the first row of R to get $R(1,1) = 1$:

! >> $R = A; \quad R(1,:) = R(1,:)/R(1,1)$

(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 $R(3,1) = 0$:

! >> $R(3,:) = R(3,:) - R(3,1) * R(1,:)$

2 Operate on R by the same method as in (1) to obtain the second column of $rref(A)$. 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.

3 Operate on R by the same method as in (2) 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.

4 Your matrix R should now be transformed into rref(A) (since A is a random 3×4 matrix, rref(A) is (almost) sure to have rank 3). Check your answer by Matlab with the command rref(A). If the Matlab answer is not the same as the current value of your R, go back and redo your calculations.

Q2. Row Operations and LU Factorization

In this problem you will use Matlab to carry out elementary row operations and to obtain the matrix factorization $A = LU$ for a square 3×3 matrix A.

1 Generate a random 3×3 matrix :

```
>> A = rand(3),    U = A
```

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

2 Use the Matlab editor to create an m-file called col1.m with the following Matlab commands:

```
>> L1 = eye(3);
>> L1(2,:) = L1(2,:) + (U(2,1)/U(1,1)) * L1(1,:);
>> L1(3,:) = L1(3,:) + (U(3,1)/U(1,1)) * L1(1,:);
>> L1
```

Execute this file by typing col1 at the Matlab prompt. If you get a division by zero error message, go back to step (1) and generate another A and U . What row operations are applied to the identity matrix to obtain the matrix L_1 ?

Use Matlab to calculate L_1^{-1} (the Matlab command $inv(X)$ will calculate the inverse matrix X^{-1}).

How are the matrix entries of L_1^{-1} related to the those of L_1? Now use Matlab to replace the current value of the matrix U by the new value $L^{-1} * U$ (remember that the command $X = Y$ in Matlab means to replace the current value of the variable X by the current value of the variable Y). How is the new value of U obtained from the old value of U by row operations?

3 Use the Matlab editor to create an m-file called col2.m with the commands

```
>> L2 = eye(3);
>> L2(3,:) = L2(3,:) + (U(3,2)/U(2,2)) * L2(2,:);
>> L2
```

This will be used with the matrix U modified as in (2). Execute this file by typing col2 at the Matlab prompt. If you get a division by zero error message, go back to step (1) and start again. Use Matlab to calculate L_2^{-1}. How are the matrix entries of L_2^{-1} obtained

from those of L_2?

Now use Matlab to replace the current value of the matrix U by the new value $L_2^{-1} * U$. How is the new value of U obtained from the old value of U by row operations?

4 Use Matlab to get $L = L_1 * L_2$. Verify by Matlab that $A = L * U$. How are the entries of L obtained from those of L_1 and L_2?

Q3. Using the LU Factorization to Solve $Ax = b$

1 Invert A, L and U

2 Solving $Ax = b$ using L and U.

Generate a random vector $b = rand(3,1)$. Calculate the solution

! >> $c = inv(L) * b$ to the triangular system $Lc = b$.

Then calculate the solution

! >> $x = inv(U) * c$ to the triangular system $Ux = c$.

Finally, calculate Ax and check that it is b.

3 Comparing the speed of LU versus RREF.

4 Generate a random 16×16 matrix A, a vector b.

Solve $Ax = b$

Calculate the LU decomposition of A

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

```
>> A = rand(16);
>> b = rand(16,1);
>> R = rref([A  b]);
>> y = R(:,17);
>> [L  U] = lu(A);
>> c = inv(L) * b;
>> x = inv(U) * c;
```

Iterative Method for Solving Linear Systems

The splittings for the methods discussed in this section all share the following notation. When A has no zeros on its diagonal, we write

$$A = D - \tilde{L} - \tilde{U} = D\big(I - L - U\big),$$

The matrix D is the diagonal of A,
The Matrix $\left(-\tilde{L}\right)$ is the strictly lower triangular part of A and $DL = \tilde{L}$,
The matrix $\left(-\tilde{U}\right)$ is the strictly upper triangular part of A and $DU = \tilde{U}$.

Jacobi Iterative Method

In matrix notation for the above splitting $A = D - E$, with $E = \tilde{L} + \tilde{U}$:

$$x^{k+1} = D^{-1}\left(b + Ex^{k}\right)$$

Component-wise notation for $i = 1, \ldots, n$

$$x^{k+1} = \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^k - \sum_{j=i+1}^{n} a_{ij}x_j^k\right)/a_{ii}$$

Gauss Seidel Iterative Method

In matrix notation for the above splitting $A = E - \tilde{U}$, with $E = D - \tilde{L}$:

$$x^{k+1} = E^{-1}\left(b + \tilde{U}x^{k}\right)$$

Component-wise notation for $i = 1, \ldots, n$

$$x^{k+1} = \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{k+1} - \sum_{j=i+1}^{n} a_{ij}x_j^k\right)/a_{ii}$$

Basic Iterative Solver

For a suitable splitting $A = E + F$, an arbitrary initial vector $x^{(0)}$ and some given accuracy threshold $\epsilon > 0$ the following algorithm approximately solves $Ax = b$

start with initial vector $x^{(0)}$
set $r^{(0)} = b - Ax^{(0)}$
for $k = 0, 1, 2, ...$
    solve $Ed^{(k+1)} = r^{(k)}$
    set $x^{(k+1)} = x^{(k)} + d^{(k+1)}$
    set $r^{(k+1)} = r^{(k)} - Ad^{(k+1)}$
until $||r^{(k+1)}|| < \epsilon$
result: approximate solution $x^{(k+1)}$

Q4. Iterative Method Algorithms

Starting with some initial $x^{(0)}$, apply the latter algorithm to solve your Q3. matrix problem using now both the Jacob and Gauss-Seidel iterative methods. Compare the four solutions (Direct, *LU*, Jacobi and Gauss-Seidel).