Matlab in Math 461, part three

Some special matrices

You can generate a 4×8 matrix of all zeroes by `zeros(4,8)`. A matrix of all ones is `ones(4,5)`. A 5×5 identity matrix is `eye(5)`. If x is a vector, then `diag(x)` is a diagonal matrix with x down the diagonal. If A is a matrix, then `diag(A)` is a vector consisting of the diagonal entries of A. See if you can guess what `diag(diag(A))` is and verify your guess by experiment. If A is a matrix, then `triu(A)` is the upper triangular part of A and `tril(A)` is the lower triangular part of A. Also useful are `triu(A,1)` and `tril(A,-1)` which give the parts of A respectively above and below the diagonal.

Partitioned matrices in Matlab

Matlab can easily work with partitioned matrices. For example, suppose you have already defined matrices A and B and want to define the matrix $C = \begin{bmatrix} A & B \\ 0 & I_3 \end{bmatrix}$ where 0 is the 3×7 matrix of zeroes. You could type:

```
>> C = [A B; zeros(3,7) eye(3)];
```

You can also change a block within a matrix. For example, here is another way to construct the matrix C. Let us suppose that A is 5×7 and B is 5×3. You could type:

```
>> C = zeros(8,10);
```

to give you an 8×10 zero matrix. Now type in the commands below to set various blocks of C to the proper values.

```
>> C(1:5,1:7) = A;
>> C(1:5,8:10) = B;
>> C(6:8,8:10) = eye(3);
```

You can get real fancy if you wanted. For example suppose you wanted to set the nine entries in the first, third and seventh rows and second fourth and sixth columns to 7.

```
>> C([1 3 7],[2 4 6]) = 7*ones(3,3)
```

would do the trick. Don't worry about this too much though.

You can extract blocks from a partitioned matrix as well. Suppose you wished to extract from A the 3×4 matrix consisting of the third through fifth rows and seventh through tenth columns. Just type

```
>> C = A(3:5,7:10)
```

LU decomposition

To find the LU decomposition of a square matrix A type:

```
>> [L U] = lu(A)
```

Note that L might not be lower triangular, but instead will be a lower triangular matrix with perhaps some rows switched. Recall that we only got L to be lower triangular if it was not necessary to do any row switches when reducing A to echelon form. If row switches must be done (or in Matlab's case are deemed advisable for accuracy reasons) the resulting L will have corresponding rows switched.

Problems due Thursday March 11

As usual you may work in groups of two or three. Remember to set up the random number generator by beginning your session with `rand('state',sum(100*clock));`. Also remember to use the methods you learned in matlab #2 to compare large matrices. I do not want them printed out unless absolutely necessary.

Problem 1: Generate a random 7×7 matrix A (but don't print it out!). Find its LU decomposition.

a) Check that $A = LU$, (but don't print out A and LU).

b) Let $\mathbf{b} = A([1\ 1\ 1\ 1\ 1\ 1\ 1]')$. Use the LU decomposition to solve $A\mathbf{x} = \mathbf{b}$ by solving $L\mathbf{y} = \mathbf{b}$ and then $U\mathbf{x} = \mathbf{y}$. Make sure you get the expected solution.

c) Note that $A^T = U^T L^T$ and the transpose of an upper triangular matrix is lower triangular and vice versa. Find the LU decomposition of A^T. Is it $U^T L^T$? Why or why not? Explain. (Note your results may vary, I want you to explain your results.)

Problem 2: Generate two random 8×8 matrices A and B and check whether or not each of the following identities holds. Warning: some of these identities are false, be sure to say which. Don't print out matrices unnecessarily.

a) $det(AB) = det(A)det(B)$.

b) $det(A^{-1}) = 1/det(A)$.

c) $det(A + B) = det(A) + det(B)$.

d) $\begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} & 0 \\ 0 & B^{-1} \end{bmatrix}$.

e) $det(\begin{bmatrix} A & I_8 \\ 0 & B \end{bmatrix} = det(A)det(B)$.

f) $det(A) = prod(diag(A))$.

g) $det(diag(diag(A))) = prod(diag(A))$.

Problem 3: Let `V = rand(10,10)` be a random matrix and set `U = eye(10) + 1000*triu(V,1);`. Note that U is upper triangular with all ones on the diagonal so its determinent is 1. Verify that $det(U) = 1$. In theory, $det(U^T) = det(U) = 1$ and $det(UU^T) = det(U)det(U^T) = 1$. Use Matlab to calculate these quantities. Probably they will be quite different from what they should be. This is one reason numerical linear algebra is so exasperating. You do so many arithmetic operations that tiny roundoff errors can in some cases accumulate to become extremely significant. Always have a healthy disrespect for the computer's answer to a linear algebra (or any other) problem. Usually the answer given is accurate, or you will be warned if it is of doubtful accuracy, but there is always that chance of being completely misled. So if people's lives or livelyhood depend on the answer, do it again a different way.