Lab exercises on matrices and Gauss elimination

Course on Mechanical Engineering, AY 2015-16

Prof. S. De Marchi

Padova, May 16, 2016

We start by introducing some useful matrices, commands and functions

1 Special matrices

```
A = zeros(2,3);
```

is a matrix 2×3 of all zeros.

```
A = eye(5);
```

is the identity matrix of order 5.

Suppose `A=[1 2 -2 4; 0 -1 6 4; 0.5 5 5 3; 0 0 3 10];`

```
d = diag(A);
```

gives the column vector of the diagonal elements, while

```
U = triu(A); L = tril(A);
```

gives the upper triangular part and the lower triangular part of `A`, respectively

2 Operation on rows or columns of matrices

- Given a matrix A of order n, the Matlab lines

  ```
  for i = 1:n
    A(i,j) = A(i,j)+1;
  end
  ```

  can be substituted by

  ```
  A(1:n,j) = A(1:n,j)+1;
  ```

  or, by using the operator **:**

  ```
  A(:,j) = A(:,j)+1;
  ```

- It is possible to **exchange rows or columns**. For example, by typing the line

  ```
  A = B([1 3 2],:);
  ```

we create a matrix A having as the first row the first row of B, the second row is the third row of B and the third row is the second row of B. Similarly,

```
A = B(:,[1:3 5:6]);
```

create a matrix A whose columns are the first 3 columns of B then the fifth and the six of B.

- It is possible to **concatenate matrices**. For example,

```
U = [A b];
```

create a matrix U which is the concatenation of A and the column vector b (dimensions of A and b must be compatible).

- It is possible to **assign the same value to a submatrix**. For example,

```
A(1:3,5:7) = 0;
```

set to zero the submatrix formed by the first 3 rows and the columns from 5 to 7 of the matrix `A`.

- Another useful command for matrix manipulation is `max`. For example, `max` in the form

```
[M, i] = max(A(2:7,j));
```

returns the biggest element M of the j-th column of A (among the second and the seventh rows) and the position `i` of such element in the vector $[a_{2,j}, a_{3,j}, \ldots, a_{7,j}]^{\mathrm{T}}$.

Conclusion. All vectorial instructions that substitute `for` loops, should be preferable for the sake of *Matlab efficiency*!

3 Matrices with special structure

- *Toeplitz* matrix. `toeplitz`

$$\begin{bmatrix} c_1 & r_2 & r_3 & \ldots & r_n \\ c_2 & c_1 & r_2 & \ddots & r_{n-1} \\ \vdots & c_2 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & r_2 \\ c_n & c_{n-1} & \ldots & c_2 & c_1 \end{bmatrix}$$

This matrix can be defined by the command `toeplitz`. As an example

```
>> toeplitz([0,1,2,3],[0,-1,-2,-3])

ans =

     0    -1    -2    -3
     1     0    -1    -2
     2     1     0    -1
     3     2     1     0
```

- *Hankel* matrix. hankel

$$\begin{bmatrix} c_1 & c_2 & c_3 & \dots & c_n \\ c_2 & c_3 & c_4 & \iddots & r_2 \\ \vdots & c_4 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & r_{n-1} \\ r_1 & r_2 & \dots & r_{n-1} & r_n \end{bmatrix}$$

This matrix can be defined by the command `hankel(c,r)`, where `c,r` are vectors of length n. Hankel matrices are symmetric, constant across the anti-diagonals, and have elements `H(i,j) = p(i+j-1)` where `p = [c r(2:end)]` completely determines the Hankel matrix. As an example

```
>> hankel([1,1/2,1/3,1/4],[1/4,1/5,1/6,1/7])

ans =

    1.0000    0.5000    0.3333    0.2500
    0.5000    0.3333    0.2500    0.2000
    0.3333    0.2500    0.2000    0.1667
    0.2500    0.2000    0.1667    0.1429
```

which corresponds to the *Hilbert matrix* of order 4, `hilb(4)`. hilb

- *Vandermonde* matrix. vander

$$\begin{bmatrix} c_1^{n-1} & \dots & c_1^2 & c_1 & 1 \\ c_2^{n-1} & \dots & c_2^2 & c_2 & 1 \\ \vdots & & \ddots & \ddots & \vdots \\ & & & & \\ c_n^{n-1} & \dots & c_n^2 & c_n & 1 \end{bmatrix}$$

```
>> vander([1 2 3 4])

ans =
```

```
1     1     1     1
8     4     2     1
27    9     3     1
64   16     4     1
```

- More special matrices can be found in

  `http://www.maths.manchester.ac.uk/~higham/mctoolbox/`

4 Fundamental functions

`det, eig, eye, diag, triu, tril, inv, cond,...`

5 The command `find`

One of the most useful command in Matlab is `find`. `find`

If we want to know which components of the vector `v=10:1:19` are ≥ 15, it suffices to type

```
>> find(v>=15)

ans =

     6     7     8     9    10
```

Now, we can do operations only on the specified elements

```
>> index=find(v>=15)

index =

     6     7     8     9    10

>> v(index)

ans =

    15    16    17    18    19

>> v(index)=v(index)-15

v =

    10    11    12    13    14     0     1     2     3     4
```

In the matrix case

```
>> A=[10,11;12,13]

A =

    10    11
    12    13

>> index=find(A<13)

index =

     1
     2
     3
```

The result of the command **find** is not a matrix, as one could expect, instead it is column vector. No problem arises in doing operations with the specified elements

```
>> A(index)=0

A =

     0     0
     0    13
```

For example, if we want to construct a matrix having elements corresponding only to the specified positions, it is necessary FIRST to initialize it with the proper dimensions

```
>> B=zeros(2)

B =

     0     0
     0     0
```

then to assign the values

```
>> B(index)=1

B =

     1     1
     1     0
```

where `index` was containing the values 1,2,3, as above.

6 Solution of linear systems by gaussian elimination

6.1 Elimination

Given the linear system $Ax = b$, the Matlab code for the **elimination step** of Gauss elimination method is as follows

```
for i=1:n-1,
  for j=i+1:n,
     m=A(j,i)/A(i,i);
     for k=i:n,
       A(j,k)=A(j,k)-m*A(i,k);
     end
   b(j)=b(j)-m*b(i);
   end
end
```

the result is an upper triangular matrix, stored again on matrix `A` and a new right hand side (r.h.s.) vector, again stored on `b`.

6.2 Back substitution

The algorithm for back substitution of a linear system $Ux = b$, with U upper triangular matrix, can be written in Matlab as in Table 1.

```
function x = BS(U,b)
% x = BS(U,b)
n = length(b);
x = b;
x(n) = x(n)/U(n,n);
for i = n-1:-1:1
 for j =i+1:n
  x(i) = x(i)-U(i,j)*x(j);
 end
x(i) = x(i)/U(i,i);
end
```

Table 1: Back substitution

Notice that the lines

```
for j = i+1:n
  x(i) = x(i)-U(i,j)*x(j);
```

```
end
x(i) = x(i)/U(i,i);
```

using matrix operators, can be substituted by the following ones

```
x(i) = (x(i)-U(i,i+1:n)*x(i+1:n))/U(i,i);
```

where, in this case, the operator `*` is the scalar product of vectors.

◇◇◇

Solve the following problems in Matlab

1. Take the `Hilbert matrix` of order n, in Matlab is `H=hilb(n)`, with n chosen by the user.

   Is this matrix symmetric? Compute its determinant for $n = 3 : 30$, what do you see? Compute also the corresponding eigenvalues by using the command *ee=eig* and the norms $\|\mathtt{det(H)} - \mathtt{prod(ee)}\|_2$ and $\|\mathtt{trace(H)} - \mathtt{sum(ee)}\|_2$.

2. Consider the vector **v=[4 1 zeros(1,n)]** and the matrix `A = toeplitz(v)`. Use the command `find` to see how many elements are different from zeros with $n = 4 : 10$. Can you derive a formula for the non-zero element of such a matrix? Can the matrix be considered *sparse*?

3. For $n = 2 : 50$, take the vectors

   - `x1=0:n;`
   - `x2=0:1/n:1;`
   - `x3=-0.5:1/n:0.5`

   Make a comparative plot, using the command `semilogy`, of the behavior of the condition numbers of the Vandermonde matrices based on the vectors `x1`, `x2` and `x3`, respectively. What do you see?

4. Take the `Hilbert matrix` of order n, in Matlab is `H=hilb(n)`, with n chosen by the user, and the linear system $Hx = b$. The vector `b` is taken so that the solution is $x = (1, ..., 1)^T$

   - Find the diagonal vector `d`, the upper triangular `U` and lower triangular `L` matrices of H. What do you see?
   - Using the vector `d`, do the command `D=diag(d)`: what do you see?
   - Solve the system $H\,x = b$ by using Matlab elementary functions.
   - Perturb the vector b by the vector $\delta b = (0, \ldots, 0, 1.e-4)^T$. Solve the new system $H\hat{x} = b + \delta\, b$ whose solution is $\hat{(x)}$.
     Estimate the `2-norm condition number`, $\kappa_2(H)$ of H by means of the relation
     $$\frac{\|x - \hat{x}\|}{\|x\|} \leq \kappa_2(H)\frac{\|\delta b\|}{\|b\|}\,.$$
     Compare κ_2 with `cond(H)`.

Notice: given a vector `v`, $\|v\|$ is computed by the command `norm(v)` which gives by default the 2-norm of `v`.

5. Consider the linear system $Ax = b$ with `A = toeplitz([4 1 0 0 0 0])` and b chosen so that the exact solution is $x = [2, 2, 2, 2, 2, 2]^T$. Solve it by the *gaussian elimination*. In the gaussian elimination, do you need to apply the pivoting?

6. Solve by gaussian elimination, the system $Ax = b$ with

$$A = \begin{bmatrix} 8 & 1 & 2 & 0.5 & 2 \\ 1 & 0.5 & 0 & 0 & 0 \\ 2 & 0 & 4 & 0 & 0 \\ 0.5 & 0 & 0 & 7 & 0 \\ 2 & 0 & 0 & 0 & 16 \end{bmatrix}.$$

   and b chosen so that `x=[0 0 1 1 1]'`.