Classes 3. MATLAB: Linear algebra. Solving linear systems

Please write diary command to file named: name_surname

1. Defining the matrix and to refer to elements:

The definition of the matrix can be made for several ways:
- by listing elements,
- by generating elements,
- by building from other matrices,
- by using two or more the above mentioned techniques together.

- Elements in the array row must be separated by a space or comma.
- A semicolon or a newline character ends the row of the matrix and moves to the next.

It is recommended to first generate a matrix by reserving memory when its size is known. To do this, you can use the instructions:

zeros	Create array of all zeros
ones	Create array of all ones
rand	Uniformly distributed random numbers
eye	Identity matrix
diag	Create diagonal matrix or get diagonal elements of matrix
blkdiag	Construct block diagonal matrix from input arguments
cat	Concatenate arrays along specified dimension
repelem	Repeat copies of array elements
repmat	Repeat copies of array
magic	Magic square
pascal	Pascal matrix
tril	Lower triangular part of matrix

Reference to matrix elements:
Specify a range of rows and columns (from, to) using colon operator, for example:

A(i,:)	i-th row of the matrix A
A(:,j)	j-th column of the matrix A
A(:)	Matrix in the form of a column vector
A(:,:)	The whole matrix (two-dimensional)
A(i,j:k)	The elements of the i-th row of the matrix A with numbers from j to k (columns from j to k)
A(i:j,k:l)	Elements from i-th and j-th row and from k-th to l-th column
A(X,i:j)	All elements in columns from i to j and rows of matrix A with numbers specified by vector elements X

2. Matrix Operations: length, diagonal, maximum, minimum, addition, multiplication, scalar product, matrix exponential, array operations

cumprod(A)	returns the cumulative product of A starting at the beginning of the first array dimension in A whose size does not equal 1
cumsum(A)	returns the cumulative sum of A starting at the beginning of the first array dimension in A whose size does not equal 1
[n m]=size(A)	assigns a variable n to the number of rows, and the variable m to the number of columns
n=size(A,1)	assigns the variable n the number of rows of matrix A
m=size(A,2)	assigns the variable m the number of columns of matrix A
length(x)	returns the length of the vector x or the length of the matrix
diag(A)	returns the vector of elements on a diagonal matrix
max(A)	returns the largest elements of A
min(A)	returns the smallest elements of A
mean(A)	returns the mean of the elements of A along the first array dimension whose size does not equal 1
median(A)	returns the median value of A
std(A)	returns the standard deviation of the elements of A along the first array dimension whose size does not equal 1
sum(A)	returns the sum of the elements of A along the first array dimension whose size does not equal 1
prod(A)	returns the product of the array elements of A
linspace(x1,x2)	returns a row vector of 100 evenly spaced points between x1 and x2.
inspace(x1,x2,n)	generates n points. The spacing between the points is (x2-x1)/(n-1).

Matrix operations performed on each array element. The symbol of the operation is preceded by point.

- The matrices must be of the same dimensions;
- Addition and subtraction of arrays and matrixes are the same.

$$B.\backslash A == A./B$$

$$A.\backslash B == B./A$$

Array power	Matrix power
$A.\hat{}k = \begin{bmatrix} a_{11}^k & a_{12}^k \\ a_{21}^k & a_{22}^k \end{bmatrix}$	$A\hat{}k = \underbrace{A * A * \ldots * A}_{k}$
Right array division	**Matrix right division**
$A./B = \begin{bmatrix} a_{11}/b_{11} & a_{12}/b_{12} \\ a_{21}/b_{21} & a_{22}/b_{22} \end{bmatrix}$	$A/B = A \cdot B^{-1}$

Left array division	Matrix left division
$A.\backslash B = \begin{bmatrix} b_{11}/a_{11} & b_{12}/a_{12} \\ b_{21}/a_{21} & b_{22}/a_{22} \end{bmatrix}$	$A\backslash B = A^{-1} \cdot B$
Array multiplication	Matrix multiplication
$A.* B = \begin{bmatrix} a_{11} * b_{11} & a_{12} * b_{12} \\ a_{21} * b_{21} & a_{22} * b_{22} \end{bmatrix}$	$A * B = \begin{bmatrix} a_{11} \cdot b_{11} + a_{12} \cdot b_{21} & a_{11} \cdot b_{12} + a_{12} \cdot b_{22} \\ a_{21} \cdot b_{11} + a_{22} \cdot b_{21} & a_{21} \cdot b_{21} + a_{22} \cdot b_{22} \end{bmatrix}$

3. Determinant, matrix transposed, inverse matrix, sum of elements in matrix

- Determinant: $\det(A)$
- Inverse matrix: $\mathrm{inv}(A)$
- Matrix transposed: A'
- $\mathrm{sum}(A)$ If A is a matrix, then sum(A) returns a row vector containing the sum of each column
- $\mathrm{sum}(A')'$ or $\mathrm{sum}(A, 2)$ is a column vector containing the sum of each row
- $\mathrm{trace(A)} = \mathrm{sum(diag(A))}$ is the sum of the diagonal elements of the matrix A
- $\mathrm{sum(diag(A))}$ the result is the sum of the elements on the second diagonal

4. Solving linear systems

The system of m linear equations with n unknowns x_1, x_2, ... , x_n, where m, n ∈N we call the system in the form

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2, \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m, \end{cases} \tag{1}$$

where $a_{ij} \in \mathbb{R},\ b_i \in \mathbb{R}$ for $1 \le i \le m$ and $1 \le j \le n$.

The system of linear equations (1) can be written in matrix form

$$A\,X = B, \tag{2}$$

where $A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdot\cdot\cdot & \cdot\cdot\cdot & \cdot\cdot\cdot & \cdot\cdot\cdot \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, X = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}, B = \begin{bmatrix} b_1 \\ b_2 \\ \cdots \\ b_m \end{bmatrix}.$

We can distinguish three cases of solutions of the system (2)[1]:

[1] http://mathworld.wolfram.com/LinearSystemofEquations.html

- If $m = n$ and the matrix A is nonsingular, i.e. $Det\ A \neq 0$, then the system (2) has a unique solution in the n variables. In particular, as shown by Cramer's rule, there is a unique solution if A has a matrix inverse A^{-1}. In this case the solution is given by the formula

$$X = A^{-1} \cdot B;$$

- If $m < n$, then the system (2) is (in general) overdetermined and there is no solution;
- if $m > n$, then the system is underdetermined. In this case, elementary row and column operations can be used to solve the system as far as possible, then the first $(m - n)$ components can be solved in terms of the last n components to find the solution space.

X = linsolve(A,B) solves the linear system $A*X = B$ using LU factorization with partial pivoting when A is square and QR factorization with column pivoting otherwise. The number of rows of A must equal the number of rows of B. If A is m-by-n and B is m-by-k, then X is n-by-k. linsolve returns a warning if A is square and ill conditioned or if it is not square and rank deficient[2].

[2] https://www.mathworks.com/help/matlab

4.1. System with unique solution

The system of equations is

$$\begin{cases} x_1 - 2x_2 = 1, \\ 2x_1 + 3x_2 = 9, \end{cases}$$

Matrix form of this system is

```
A =
        1 −2
        3  3
B =
        1
        9
```

The solution is

```
>> X=A\B
X =
3.0000
1.0000
```

Alternatively, you can use also: linsolve (A, B).

4.2. Underdetermined system

System of linear equations is considered underdetermined if there are fewer equations than unknowns.

This is a linear programming problem. To find parametric solutions we use QR method.

For example, the data are randomly-defined arrays:

```
R=fix(10*rand(2,4));
b=fix(10*rand(2,1));
```

We convert the format of display data to the fractional format

```
format rat
```

and we designate

```
p=R\b
Z=null(R,'r')
```

Then the multiplication $R * Z$ will give a zero matrix or close to zero matrix. Hence each vector q satisfies of equation $R * x = b$ and is solution of the initial equation.

5. Exercises

Exercise 5.1. Perform the following operations on matrices:

a) $\begin{bmatrix} 3 & -1 \\ -1 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 5 \\ 3 & -1 \end{bmatrix} =$

b) $\begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix} * \begin{bmatrix} 0 & 1 \\ 6 & 7 \end{bmatrix} =$

c) $4\begin{bmatrix} -4 & 1 \\ 0 & 2 \end{bmatrix} + 2\begin{bmatrix} -2 & 3 \\ 5 & -1 \end{bmatrix} =$

d) Perform the above operations using array operations. See what's the difference.

e) Create your matrix and calculate its second power for both exponential operators.

f) Calculate the inverse matrix and the determinant for matrices:

$$\begin{bmatrix} 1 & 2 & 3 & -1 \\ 2 & 1 & 1 & 2 \\ 4 & 3 & 2 & 0 \\ 1 & -2 & -3 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 3 & 0 & -1 \\ 2 & 4 & -2 & 4 \\ 3 & 1 & -1 & 7 \\ -1 & 3 & 8 & 3 \end{bmatrix}, \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 9 & 7 & 8 \\ 10 & 1 & 2 & 3 \\ 4 & 5 & 6 & 7 \end{bmatrix}.$$

Exercise 5.2. This matrix $\begin{bmatrix} 16 & 3 & 2 & 13 \\ 5 & 10 & 11 & 8 \\ 9 & 6 & 7 & 12 \\ 4 & 15 & 14 & 1 \end{bmatrix}$ is Dürer's matrix[1]. If you add all numbers from any row or column, or from one of the two major diagonals, you will always get the same result.

Check this properties using the sum and transpose functions in Matlab.

Exercise 5.3. From the given 4x4 matrix $\begin{bmatrix} 1 & 2 & 3 & -1 \\ 4 & 8 & 6 & 0 \\ 7 & 8 & 9 & -2 \\ 0 & 1 & 2 & 3 \end{bmatrix}$ write three matrixes of dimensions 3x3 and 2x2.

Exercise 5.4. For 100 elemental vector A, calculate:
a) the sum of the elements;
b) average elements;
c) median;

d) standard deviation;
e) the number of elements.

Exercise 5.5. For a 5x5 matrix calculate the sum of all elements.

Exercise 5.6. Create a vector of 9 numbers in the range of 5 up to 15, which are sorted and arranged evenly.

Exercise 5.7. In the Quota vector = [30, 25, 20, 34, 32] monetary sums are displayed for individual days of the week. Determine the cumulative vector of sums in which the sums of the values of the previous days are stored.

Exercise 5.8. Create a 5x5 size matrix with integer random numbers from 1 to 9 (e.g. random evenly).