FF505

Computational Science

Lecture 2

Linear Algebra with Matlab
Linear Systems

Marco Chiarandini (marco@imada.sdu.dk)

Department of Mathematics and Computer Science (IMADA)
University of Southern Denmark

Outline

Resume

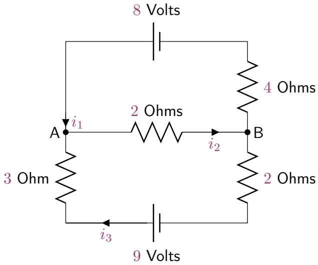
- MATLAB, numerical computing vs symbolic computing
- MATLAB Desktop
- Script files
- 1D and 2D arrays
- Plot
- Interacting with matlab

- matrices and vectors
- solving linear systems
- determinants
- linear transformation
- eigenvalues and eigenvectors
- diagonalization?
- spectral theorem?

- projectile trajectory?
- car market assignment?

Outline

Electrical Networks



We want to determine the amount of current present in each branch.

Kirchoff's Laws

- At every node, the sum of the incoming currents equals the sum of the outgoing currents
- Around every closed loop, the algebraic sum of the voltage gains must equal the algebraic sum of the voltage drops.

Voltage drops E (by Ohm's law)

$$E = iR$$

$i_1 - i_2 + i_3 = 0$	node A
$-i_1 + i_2 - i_3 = 0$	node B
$4i_1 + 2i_2 = 8$	top loop
$2i_2 + 5i_3 = 9$	bottom loop

Chemical Equations

$$x_1 CO_2 + x_2 H_2O \to x_3 O_2 + x_4 C_6 H_{12} O_6$$

To balance the equation, we must choose x_1, x_2, x_3, x_4 so that the numbers of carbon, hydrogen, and oxygen atoms are the same on each side of the equation.

$$x_1 = 6x_4 \qquad \text{carbon atoms}$$
$$2x_1 + x_2 = 2x_3 + 6x_4 \qquad \text{oxygen}$$
$$2x_2 = 12x_4 \qquad \text{hydrogen}$$

Outline

1. An Example: Electrical Networks

2. Matrices and Vectors in MatLab

3. Solving Linear Systems

Matrix Multiplication

$$
\begin{aligned}
i_1 - i_2 + i_3 &= 0 && \text{node A} \\
-i_1 + i_2 - i_3 &= 0 && \text{node B} \\
4i_1 + 2i_2 &= 8 && \text{top loop} \\
2i_2 + 5i_3 &= 9 && \text{bottom loop}
\end{aligned}
$$

$$
\begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 4 & 2 & 0 \\ 0 & 2 & 5 \end{bmatrix} \begin{bmatrix} i_1 \\ i_2 \\ i_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 8 \\ 9 \end{bmatrix} \qquad A\mathbf{x} = \mathbf{b}
$$

Matrix Multiplication

$$x_1 = 6x_4$$ carbon atoms

$$2x_1 + x_2 = 2x_3 + 6x_4$$ oxygen

$$2x_2 = 12x_4$$ hydrogen

$$\begin{bmatrix} 1 & 0 & 0 & -6 \\ 2 & 1 & 2 & 6 \\ 0 & 2 & 0 & 12 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \qquad A\mathbf{x} = \mathbf{0}$$

Creating Matrices

```
eye(4)  % identity matrix
zeros(4)  % matrix of zero elements
ones(4)  % matrix of one elements
```

```
A=rand(8)
triu(A)  % upper triangular matrix
tril(A)
diag(A)  % diagonal
```

```
>> [ eye(2), ones(2,3); zeros(2),
     [1:3;3:-1:1] ]

ans =

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

Can you create this matrix in one line of code?

```
 -5     0     0     0     0     0     0     1     1     1     1
  0    -4     0     0     0     0     0     0     1     1     1
  0     0    -3     0     0     0     0     0     0     1     1
  0     0     0    -2     0     0     0     0     0     0     1
  0     0     0     0    -1     0     0     0     0     0     0
  0     0     0     0     0     0     0     0     0     0     0
  0     0     0     0     0     0     1     0     0     0     0
  1     0     0     0     0     0     0     2     0     0     0
  1     1     0     0     0     0     0     0     3     0     0
  1     1     1     0     0     0     0     0     0     4     0
  1     1     1     1     0     0     0     0     0     0     5
```

Matrix-Matrix Multiplication

In the product of two matrices A * B,
the number of columns in A must equal the number of rows in B.

The product AB has the same number of rows as A and the same number of columns as B. For example

```
>> A=randi(10,3,2) % returns a 3-by-2 matrix containing pseudorandom integer values
      drawn from the discrete uniform distribution on 1:10
A =
     6 10
    10 4
     5 8

>> C=randi(10,2,3)*100
C =
        1000 900 400
         200 700 200
>> A*C % matrix multiplication
ans =
        8000 12400 4400
       10800 11800 4800
        6600 10100 3600
```

Exercise: create a small example to show that in general, $AB \neq BA$.

Matrix Operations

```
%% matrix operations
A * C % matrix multiplication
B = [5 6; 7 8; 9 10] * 100 % same dims as A
A .* B % element-wise multiplcation
% A .* C or A * B gives error - wrong dimensions
A .^ 2
1./B
log(B) % functions like this operate element-wise on vecs or matrices
exp(B) % overflow
abs(B)
v = [-3:3] % = [-3 -2 -1 0 1 2 3]
-v % -1*v

v + ones(1,length(v))
% v + 1 % same

A' % (conjuate) transpose
```

Array Operations

- **Addition/Subtraction**: trivial
- **Multiplication**:
  - of an array by a scalar is easily defined and easily carried out.
  - of two arrays is not so straightforward:
    MATLAB uses two definitions of multiplication:
    - array multiplication (also called element-by-element multiplication)
    - matrix multiplication
- **Division and exponentiation** MATLAB has two forms on arrays.
  - element-by-element operations
  - matrix operations

  ⇝ Remark:
  the operation division by a matrix is not defined. In MatLab it is defined but it has other meanings.

Element-by-Element Operations

Symbol	Operation	Form	Examples
+	Scalar-array addition	A + b	[6,3]+2=[8,5]
-	Scalar-array subtraction	A - b	[8,3]-5=[3,-2]
+	Array addition	A + B	[6,5]+[4,8]=[10,13]
-	Array subtraction	A - B	[6,5]-[4,8]=[2,-3]
.*	Array multiplication	A.*B	[3,5].*[4,8]=[12,40]
./	Array right division	A./B	[2,5]./[4,8]=[2/4,5/8]
.\	Array left division	A.\B	[2,5].\[4,8]=[2\4,5\8]
.^	Array exponentiation	A.^B	[3,5].^2=[3^2,5^2]
			2.^[3,5]=[2^3,2^5]
			[3,5].^[2,4]=[3^2,5^4]

Backslash or Matrix Left Division

`A\B` is roughly like `INV(A)*B` except that it is computed in a different way:
`X = A\B` is the solution to the equation `A*X = B` computed by Gaussian elimination.

Slash or right matrix division:
`A/B` is the matrix division of `B` into `A`, which is roughly the same as `A*INV(B)`, except it is computed in a different way. More precisely, `A/B = (B'\A')'`.

Dot and Cross Products

dot(A,B) inner or scalar product: computes the projection of a vector on the other. eg. dot(Fr,r) computes component of force $\mathbf{F}$ along direction $\mathbf{r}$
//Inner product, generalization of dot product

```
v=1:10
u=11:20
u*v' % inner or scalar product
ui=u+i
ui'
v*ui' % inner product of C^n
norm(v,2)
sqrt(v*v')
```

cross(A,B) cross product: eg: moment $\mathbf{M} = \mathbf{r} \times \mathbf{F}$

Exercise: Projectile trajectory

$\boldsymbol{p}$ position vector

$$\boldsymbol{p}_t = \boldsymbol{p}_0 + \boldsymbol{u}_t s_m t + \frac{\boldsymbol{g} t^2}{2}$$

s_m muzzle velocity (speed at which the projectile left the weapon)
u_t is the direction the weapon was fired
$g = -9.81\mathrm{ms}^{-1}$

Predict the landing spot

$$t_i = \frac{-u_i s_m \pm \sqrt{u_y^2 s_m^2 - 2 g_y (p_{y0} - p_{yt})}}{g_y} \qquad \boldsymbol{p}_E = \begin{bmatrix} p_{x0} + u_x s_m t_i \\ p_{y0} \\ p_{z0} + u_z s_m t_i \end{bmatrix}$$

Plot the trajectory in 2D.

Exercise: Projectile trajectory

Given a firing point $\boldsymbol{S}$ and s_m and a target point $\boldsymbol{E}$, we want to know the firing direction $\boldsymbol{u}$, $|\boldsymbol{u}| = 1$.

$$\begin{aligned} E_x &= S_x + u_x s_m t_i + \frac{1}{2} g_x t_i^2 \\ E_y &= S_y + u_y s_m t_i + \frac{1}{2} g_y t_i^2 \\ E_z &= S_z + u_z s_m t_i + \frac{1}{2} g_z t_i^2 \\ 1 &= u_x^2 + u_y^2 + u_z^2 \end{aligned}$$

four eq. in four unknowns, leads to:

$$|\boldsymbol{g}|^2 t_i^4 - 4(\boldsymbol{g} \cdot \boldsymbol{\Delta} + s_m^2) t_i^2 + 4|\boldsymbol{\Delta}|^2 = 0, \qquad \boldsymbol{\Delta} = \boldsymbol{E} - \boldsymbol{S}$$

solve in t, and interpret the solution.

Reshaping

```
%% reshape and replication
A = magic(3) % magic square
A = [A [0;1;2]]
reshape(A,[4 3]) % columnwise
reshape(A,[2 6])
v = [100;0;0]
A+v
A + repmat(v,[1 4])
```

Matrix Functions

Eigenvalues and eigenvectors:

```
A = ones(6)
trace(A)
A = A - tril(A)-triu(A,2)
eig(A)

diag(ones(3,1),-1)
[V,D]=eig(diag(1:4))

rank(A) % rank of A
orth(A) % orthonormal basis
```

Visualizing Eigenvalues

```
A=[5/4,0;0,3/4];
eigshow(A) %effect of operator A on unit
      verctor
```

Outline

1. An Example: Electrical Networks

2. Matrices and Vectors in MatLab

3. Solving Linear Systems

Systems of Linear Equations

How many solutions have these linear systems? Find it out using the graphical approach.

$$\begin{aligned} 6x - 10y &= 2 \\ 3x - 4y &= 5 \end{aligned}$$

```
% plot functions in implicit form
ezplot
```

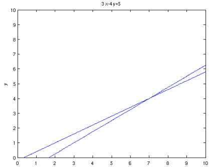
$$\begin{aligned} 3x - 4y &= 5 \\ 6x - 8y &= 10 \end{aligned}$$

$$\begin{aligned} 3x - 4y &= 5 \\ 6x - 8y &= 3 \end{aligned}$$

Systems of Linear Equations

$$\begin{aligned} 6x- \quad 10y &= \quad 2 \\ 3x- \quad 4y &= \quad 5 \end{aligned}$$

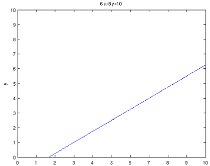
```
% plot functions in implicit form
ezplot('6*x-10*y=2',[0 10 0 10]),
hold,
ezplot('3*x-4*y=5',[0 10 0 10])
```



has one single solution

$$\begin{aligned} 3x- \quad 4y &= \quad 5 \\ 6x- \quad 8y &= \quad 10 \end{aligned}$$

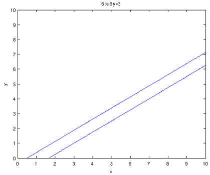
```
ezplot('3*x-4*y=5',[0 10 0 10]),
hold,
ezplot('6*x-8*y=10',[0 10 0 10])
```



has infinite solutions

$$\begin{aligned} 3x- \quad 4y &= \quad 5 \\ 6x- \quad 8y &= \quad 3 \end{aligned}$$

```
ezplot('3*x-4*y=5',[0 10 0 10]),
hold,
ezplot('6*x-8*y=3',[0 10 0 10])
```



has no solution

Matrix Form

The linear system:

$$\begin{aligned} 2x_1 + \quad 9x_2 &= \quad 5 \\ 3x_1 - \quad 4x_2 &= \quad 7 \end{aligned}$$

can be expressed in vector-matrix form as:

$$\begin{bmatrix} 2 & 9 \\ 3 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 7 \end{bmatrix}$$

In general, a set of m equations in n unknowns can be expressed in the form $A\mathbf{x} = \mathbf{b}$, where A is $m \times n$, $\mathbf{x}$ is $n \times 1$ and $\mathbf{b}$ is $m \times 1$.

The inverse of A is denoted A^{-1} and has property that

$$A^{-1}A = AA^{-1} = I$$

Hence the solution to our system is:

$$\mathbf{x} = A^{-1}\mathbf{b}$$

Inverse and Determinant

Compute the inverse and the determinant of this matrix in Matlab:

```
>> A=[3 -4; 6 -8];
```

Has the system solutions?
What about the system in the previous slide? What are its solutions?

A matrix is singular if $\det(A) = |A| = 0$

Inverse of a square matrix A is defined only if A is nonsingular.

If A is singular, the system has no solution

```
>> A=[3 -4; 6 -8];
>> det(A)
ans =
     0
>> inv(A)
Warning: Matrix is singular to working precision.
ans =
   Inf  Inf
   Inf  Inf
```

For a 2×2 matrix

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

the matrix inverse is

$$A^{-1} = \frac{1}{|A|} \begin{bmatrix} d & -c \\ -b & a \end{bmatrix}^T = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

For a 3×3 matrix

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

the matrix inverse is

$$A^{-1} = \frac{1}{|A|} \begin{bmatrix} +\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} & -\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} & +\begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix} \\ -\begin{vmatrix} a_{12} & a_{13} \\ a_{32} & a_{33} \end{vmatrix} & +\begin{vmatrix} a_{11} & a_{13} \\ a_{31} & a_{33} \end{vmatrix} & -\begin{vmatrix} a_{11} & a_{12} \\ a_{31} & a_{32} \end{vmatrix} \\ +\begin{vmatrix} a_{12} & a_{13} \\ a_{22} & a_{23} \end{vmatrix} & -\begin{vmatrix} a_{11} & a_{13} \\ a_{21} & a_{23} \end{vmatrix} & +\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} \end{bmatrix}^T$$

Calculating the inverse

$$A^{-1} = \frac{1}{|A|}\mathrm{adj}(A)$$

$\mathrm{adj(A)}$ is the adjugate matrix of A:

1. Calculate the (i, j) minor of A, denoted M_{ij}, as the determinant of the $(n-1) \times (n-1)$ matrix that results from deleting row i and column j of A.
2. Calculate the cofactor matrix of A, as the $n \times n$ matrix C whose (i, j) entry is the (i, j) cofactor of A

$$C_{ij} = (-1)^{i+j} M_{ij}$$

3. set $\mathrm{adj(A)}_{ij} = C_{ji}$

Left Division Method

- $\mathbf{x} = A^{-1}\mathbf{b}$ rarely applied in practice because calculation is likely to introduce numerical inaccuracy
- The inverse is calculated by LU decomposition, the matrix form of Gaussian elimination.

```
% left division method
x = A\b
```

$$A = LU$$
$$PA = LU$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}$$

- for a matrix A, $n \times n$, $\det(A) \neq 0 \Leftrightarrow$ rank of A is n
  - for a system $A\mathbf{x} = \mathbf{b}$ with m equations and n unknowns a solution exists iff $rank(A) = rank([A\mathbf{b}]) = r$
    - if $r = n \rightsquigarrow$ unique
    - if $r < n \rightsquigarrow$ infinite sol.
  - for a homogeneous system $A\mathbf{x} = \mathbf{0}$ it is always $rank(A) = rank([A\mathbf{b}])$ and there is a nonzero solution iff $rank(A) < n$

• `A\b` works for square and nonsquare matrices. If nonsquare ($m < n$) then the system is undetermined (infinite solutions). `A\b` returns one variable to zero

• `A\b` does not work when $\det(A) = 0$.

```
>> A=[2, -4,5;-4,-2,3;2,6,-8];
>> b=[-4;4;0];
>> rank(A)
ans =
     2
>> rank([A,b])
ans =
     2
>> x=A\b
Warning: Matrix is singular to working
     precision.
x =
   NaN
   NaN
   NaN
```

However since

$$rank(A) = rank([A\mathbf{b}])$$

an infinite number of solutions exist (undetermined system).
`x=pinv(A)b` solves with pseudoinverse and `rref([A,b])` finds the reduced row echelon form

Overdetermined Systems

An overdetermined system is a set of equations that has more independent equations than unknowns ($m > n$).

For such a system the matrix inverse method will not work because the A matrix is not square.

However, some overdetermined systems have exact solutions, and they can be obtained with the left division method `x = A \ b`

For other overdetermined systems, no exact solution exists. We need to check the ranks of A and $[A\mathbf{b}]$ to know whether the answer is the exact solution. If a solution does not exist, the left-division answer is the least squares solution.

Flowchart for Linear System Solver