1. Manipulación de Vectores y Matrices

- Matrix

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \end{pmatrix}$$

`A=[1 2 3 4 5; 6 7 8 9 10; 11 12 13 14 15; 16 17 18 19 30]`

or you can replace the spaces between elements in a row by commas

`A=[1,2,3,4,5;6,7,8,9,10;11,12,13,14,15;16,17,18,19,20]`

or you can replace the semicolon by a line break

```
A=[1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20]
```

The brackets at the beginning and end of the matrix are *rectangular*.

An upper case `A` is not the same as as lower case `a`. Matlab treats them as different variables.

- Matrix dimensions: The matrix A is 4×5

`size(A)` $= \begin{bmatrix} 4 & 5 \end{bmatrix}$
Number of rows: `size(A,1)` $= 4$
Number of columns: `size(A,2)` $= 5$

Larger of the two dimensions: `length(A)` $= 5$

This is helpful for counting the number of elements in a row vector or a column vector.

- Matrix elements

Element $(1,2)$: `A(1,2)` $= 2$
Element $(2,5)$: `A(2,5)` or `A(2,end)`
Element $(4,3)$: `A(4,3)` or `A(end,3)`

Note the *round* parentheses.

- Rows and columns

Row 1 of A: `A(1,:)` $= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \end{pmatrix}$
Row 4 of A: `A(4,:)` or `A(end,:)`

Column 2 of A: `A(:,2)` $= \begin{pmatrix} 2 & 7 & 12 & 17 \end{pmatrix}^T$
Column 5 of A: `A(:,5)` or `A(:.end)`

- Submatrices

Rows 1, 2, 3 and columns 2, 3, 4: `A(1:3,2:4)` $= \begin{pmatrix} 2 & 3 & 4 \\ 7 & 8 & 9 \\ 12 & 13 & 14 \end{pmatrix}$

Rows 1, 2 and column 1, 3, 5: `A([1,2],[1,3,5])` $= \begin{pmatrix} 1 & 3 & 5 \\ 6 & 8 & 10 \end{pmatrix}$

Rows 3 and 4: `A(3:4,:)`

Columns 1, 2, 3, 5: `A(:, [1:3,5])`

- Matrix operations:

Addition: `A+B`
Scalar multiplication: `3*A`
Multiplication: `A*B`

- Special matrices:

`eye(5)` 5×5 identity matrix
`zeros(3,2)` 3×2 zero matrix
`zeros(size(A))` zero matrix of same dimension as matrix A

- Diagonal elements

`diag(A)` $= \begin{pmatrix} 1 & 7 & 13 & 19 \end{pmatrix}^T$

When the input is a matrix, `diag` extracts the diagonal elements (i.e. all elements in position (j,j)) and puts them into a *column* vector.

`diag([1:3,50])` $= \begin{pmatrix} 1 & & & \\ & 2 & & \\ & & 3 & \\ & & & 50 \end{pmatrix}$

When the input is a vector, `diag` puts the elements of the vector onto the diagonal of a matrix.

`diag(diag(A))` $= \begin{pmatrix} 1 & & & \\ & 7 & & \\ & & 13 & \\ & & & 19 \end{pmatrix}$

extracts the diagonal of a matrix and removes the offdiagonal elements

- Permutations

  `A(:,[2 1 3:end])` permute the two leading columns of A

  `A(end:-1:1,:)` permute the rows of A in reverse order

- Conjugate transpose: `A'`

- Piecing together matrices:

`[eye(3) zeros(3,4); [1 30:10:80]]` $= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 30 & 40 & 50 & 60 & 70 & 80 \end{pmatrix}$

Creación automática de matrices especiales:

función	resultado
`eye(n)`	matriz identidad de tamaño $n \times n$
`diag(vector)`	matriz diagonal con los elementos de `vector`
`zeros(m,n)`	matriz de ceros de tamaño $m \times n$
`ones(m,n)`	matriz de unos de tamaño $m \times n$
`rand(m,n)`	matriz de aleatorios (dist. $\mathrm{U}(0,1)$) de tamaño $m \times n$
`randn(m,n)`	matriz de aleatorios (dist. $\mathrm{N}(0,1)$) de tamaño $m \times n$

2. Análisis de Sistemas Lineales

You can specify LTI models as:

- Transfer functions (TF), for example,

$$P(s) = \frac{s+2}{s^2+s+10}$$

- Zero-pole-gain models (ZPK), for example,

$$H(z) = \left[\frac{2(z-0.5)}{z(z+0.1)} \quad \frac{(z^2+z+1)}{(z+0.2)(z+0.1)} \right]$$

- State-space models (SS), for example,

$$\frac{dx}{dt} = Ax + Bu$$
$$y = Cx + Du$$

where A, B, C, and D are matrices of appropriate dimensions, x is the state vector, and u and y are the input and output vectors.

Creación y conversión entre modelos:

Command	Description
`set`	Set LTI model properties.
`size`	Get output/input/array dimensions or model order.
`ss`	Create a state-space model.
`ssdata`, `dssdata`	Retrieve state-space data (respectively, descriptor state-space data) or convert it to cell array format.
`tf`	Create a transfer function.
`tfdata`	Retrieve transfer function data.
`zpk`	Create a zero-pole-gain model.
`zpkdata`	Retrieve zero-pole-gain data.

Table 2-2: Converting LTI Models

Command	Description
`c2d`	Continuous- to discrete-time conversion.
`d2c`	Discrete- to continuous-time conversion.
`d2d`	Resampling of discrete-time models.
`frd`	Conversion to an FRD model.
`pade`	Padé approximation of input delays.
`ss`	Conversion to state space.
`tf`	Conversion to transfer function.
`zpk`	Conversion to zero-pole-gain.

General Model Characteristics Commands	
class	Display model type ('tf', 'zpk', 'ss', or 'frd').
hasdelay	Test true if LTI model has any type of delay.
isa	Test true if LTI model is of specified class.
isct	Test true for continuous-time models.
isdt	Test true for discrete-time models.
isempty	Test true for empty LTI models.
isproper	Test true for proper LTI models.
issiso	Test true for SISO models.
ndims	Display the number of model/array dimensions.
reshape	Change the shape of an LTI array.
size	Output/input/array dimensions. Used with special syntax, size also returns the number of state dimensions for state-space models, and the number of frequencies in an FRD model.

Time Response	
impulse	Impulse response.
initial	Initial condition response.
gensig	Input signal generator.
lsim	Simulation of response to arbitrary inputs.
step	Step response.

State-Space Realizations	
canon	Canonical state-space realizations.
ctrb	Controllability matrix.
ctrbf	Controllability staircase form.
gram	Controllability and observability gramians.
obsv	Observability matrix.
obsvf	Observability staircase form.
ss2ss	State coordinate transformation.
ssbal	Diagonal balancing of state-space realizations.

Model Dynamics	
covar	Covariance of response to white noise.
damp	Natural frequency and damping of system poles.
dcgain	Low-frequency (DC) gain.
dsort	Sort discrete-time poles by magnitude.
esort	Sort continuous-time poles by real part.
norm	Norms of LTI systems (H_2 and L_∞).
pole, eig	System poles.
pzmap	Pole/zero map.
zero	System transmission zeros.