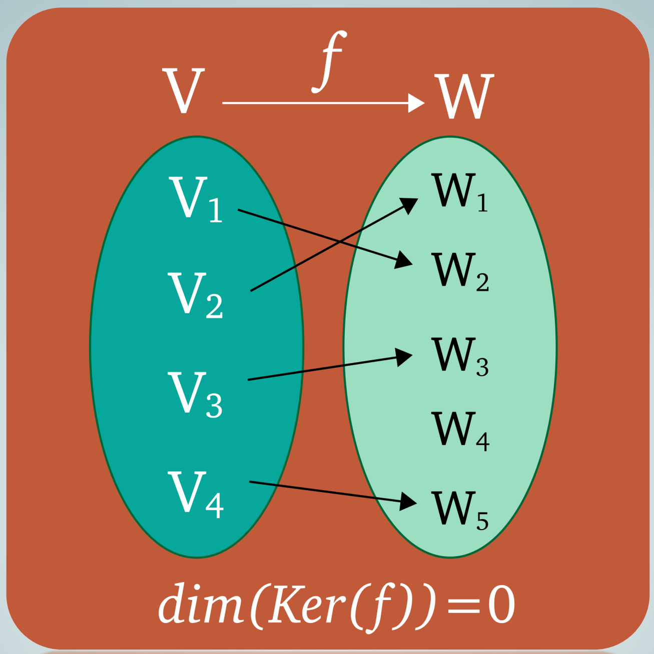
Álgebra

Práctica 1. Introducción a MATLAB



Rodrigo García Manzanas
Neila Campos González
Ana Casanueva Vicente

Departamento de Matemática Aplicada y Ciencias de la Computación

Grado en Ingeniería Química

Práctica 1: Introducción a MATLAB

Rodrigo García Manzanas (rodrigo.manzanas@unican.es)

Neila Campos González (neila.campos@unican.es)

Objetivos

- Familiarizarse con el entorno MATLAB
- Crear vectores y matrices
- Calcular determinantes e inversas

Introducción

El nombre de MATLAB es un acrónimo de MATrix LABoratory. Hoy en día MATLAB es un programa muy potente con un entorno agradable, que incluye herramientas de visualización gráfica, así como un lenguaje de alto nivel. La interfaz principal de MATLAB se divide en varias partes:

- Las órdenes a ejecutar se escriben en el *Command Window* (ventana de comandos).
- En el editor (ventana *Editor*) es donde escribimos código que guardaremos en scripts (con extensión *.m)* o live scripts (extensión *.mlx*), para su posterior ejecución desde el *Command Window*. El color que aparece en la barra lateral a la derecha del script nos indica si la sintáxis es correcta (verde), es correcta pero podría llegar a dar problemas (naranja) o es incorrecta (rojo). Los comentarios (texto no ejecutable) se introducen con el caracter "*%".*
- La ventana *Workspace* (espacio de trabajo) proporciona información sobre las variables que están almacenadas en memoria.

Hay ciertas reglas para definir variables en MATLAB:

- Su nombre puede tener como máximo 63 caracteres que pueden ser letras, números y el guión bajo.
- El primer carácter tiene que ser una letra: *matriz2* es un nombre válido, *2matriz* no lo es.
- Las mayúsculas y las minúsculas tienen valor distintivo: la variable *X* es distinta de la variable *x*.
- No puede haber espacios en blanco: *modulo1* es un nombre de variable válido, *modulo 1* no.
- Existen nombres que deben evitarse porque tienen significado propio en MATLAB: *ans*, *pi*, *Inf*, *i*, . . .

A tener en cuenta:

- Si se escribe el caracter ";" al final de una linea de código, dicha línea es ejecutada, pero el resultado de la ejecución no se muestra por pantalla.
- Los comandos *help, doc y lookfor* sirven para obtener ayuda sobre el resto de comandos/funciones. La documentación online de MATLAB (accesible mediante *doc*) es realmente buena.

```
help diag
```

```
diag - Create diagonal matrix or get diagonal elements of matrix

    This MATLAB function returns a square diagonal matrix with the elements of
    vector v on the main diagonal.

    D = diag(v)
    D = diag(v,k)
    x = diag(A)
    x = diag(A,k)

    See also blkdiag, isdiag, istril, istriu, spdiags, tril, triu

    Documentation for diag
    Other functions named diag
```

```
lookfor diagonal
```

```
spdiags                        - Sparse matrix formed from diagonals.
balance                        - Diagonal scaling to improve eigenvalue accuracy.
cdf2rdf                        - Complex diagonal form to real block diagonal form.
isdiag                         - Determine whether a matrix is diagonal.
rsf2csf                        - Real block diagonal form to complex diagonal form.
trace                          - Sum of diagonal elements.
sqrtm_tbt                      - Square root of 2x2 matrix from block diagonal of Schur form.
blkdiag                        - Block diagonal concatenation of matrix input arguments.
diag                           - Diagonal matrices and diagonals of a matrix.
lesp                           - Tridiagonal matrix with real, sensitive eigenvalues.
poisson                        - Block tridiagonal matrix from Poisson's equation (sparse).
toeppen                        - Pentadiagonal Toeplitz matrix (sparse).
tridiag                        - Tridiagonal matrix (sparse).
ssbal                          - Balancing of state-space model using diagonal similarity.
blkbuild                       - Builds a block-diagonal state-space structure from a block
diag                           - Create or extract symbolic diagonals.
getHistogramAxes               - This function adds a set of axes along the diagonals for the histograms.
bkbrk                          - Part(s) of an almost block-diagonal matrix.
slvblk                         - Solve almost block-diagonal linear system.
diag                           - Diagonal matrices or diagonals of matrix
trace                          - Sum of diagonal elements.
hFindDiagElementsInLocalPart   - Return the local linear indices of the diagonal
diag                           - Diagonal matrices and diagonals of a codistributed matrix
isdiag                         - Determine whether a codistributed matrix is diagonal.
spdiags                        - Sparse matrix formed from diagonals.
diagSolve                      - Diagonal solver for codistributed matrices
hFindDiagElementsInLocalPart   - Return the local linear indices of the diagonal
cdf2rdf                        - Complex diagonal form to real block diagonal form.
diag                           - Diagonal matrices and diagonals of a gpuArray matrix
isdiag                         - Determine whether a gpuArray matrix is diagonal.
spdiags                        - Sparse matrix formed from diagonals.
trace                          - Sum of diagonal elements.
diagrep                        - Replicates model along the diagonal.
diagrep                        - Replicates model along the diagonal.
diagrep                        - Replicates model along the diagonal.
```

```
diagrep                         - Replicates model along the diagonal.
diagrep                         - Replicates model along the diagonal.
blkdiag                         - Block-diagonal concatenation of input/output models.
bdschur                         - Block-diagonal Schur factorization.
```

Operaciones básicas

La forma de operar con MATLAB es igual que con una calculadora de bolsillo, usando los símbolos +, -, *, /, ^. Por ejemplo:

- Suma y resta:

```
2+4
```

```
ans = 6
```

```
2-4
```

```
ans = -2
```

- Multiplicación y división:

```
2*pi
```

```
ans = 6.2832
```

```
pi/10
```

```
ans = 0.3142
```

- Potenciación:

```
3^2
```

```
ans = 9
```

- Otros operadores importantes:

```
sqrt(4)  % raíz cuadrada
```

```
ans = 2
```

```
log(1)  % logaritmo neperiano
```

```
ans = 0
```

```
exp(1)  % exponencial
```

```
ans = 2.7183
```

```
sin(pi/2)  % seno
```

```
ans = 1
```

```
cos(pi/2)  % coseno
```

```
ans = 6.1232e-17
```

```
tan(pi/2)  % tangente
```

```
ans = 1.6331e+16
```

- Formatos:

```
format short  % o simplemente "format": 4 cifras decimales
1/3
```

```
ans = 0.3333
```

```
format long  % 15 cifras decimales
1/3
```

```
ans =
   0.333333333333333
```

```
format rat  % formato racional
1/3
```

```
ans =
       1/3
```

```
format  % volvemos al formato corto de 4 cifras decimales
```

Creación de vectores y matrices

Hay varias maneras de crear un vector fila:

```
a = [1 4 9]
```

```
a = 1×3
     1     4     9
```

```
a = [1, 4, 9]  % podemos dejar un espacio en blanco entre números o poner una coma
```

```
a = 1×3
     1     4     9
```

```
a = 1:2:10  % secuencia regular de elementos entre 1 y 10, escogidos de 2 en 2
```

```
a = 1×5
     1     3     5     7     9
```

Para crear un vector columna:

```
b = [-1; 2; 3]
```

```
b = 3×1
    -1
     2
     3
```

```
b = [1 2 3]'  % la comilla simple se utiliza para transponer
```

```
b = 3×1
     1
     2
     3
```

Para crear una matriz combinamos la definición de vector fila y vector columna. Por ejemplo:

```
M = [1 2 3; 4 5 6]  % matriz de 2 filas y 3 columnas
```

```
M = 2×3
     1     2     3
     4     5     6
```

```
A = [1 2 1; 2 4 3; 3 5 2]  % matriz de 3 filas y 3 columnas
```

```
A = 3×3
     1     2     1
     2     4     3
     3     5     2
```

Se pueden seleccionar elementos de una matriz indicando, entre paréntesis, la posición de la fila y la columna.

```
A(1, 3)  % elemento de la primera fila y tercera columna
```

```
ans = 1
```

Para extraer filas o columnas enteras se utiliza el símbolo ":"

```
A(2, :)  % fila 2
```

```
ans = 1×3
     2     4     3
```

```
A(:, 1)  % columna 1
```

```
ans = 3×1
     1
     2
     3
```

```
A(:, 1:3) % columnas 1, 2 y 3
```

```
ans = 3×3
     1     2     1
     2     4     3
     3     5     2
```

```
A(:, [1, 3]) % columnas 1 y 3
```

```
ans = 3×2
     1     1
     2     3
     3     2
```

Para crear una matriz de ceros:

```
mceros = zeros(2,3)  % matriz con 2 filas y 3 columnas de ceros
```

```
mceros = 2×3
     0     0     0
     0     0     0
```

Para crear una matriz de unos:

```
munos = ones(2,3)  % matriz con 2 filas y 3 columnas de unos
```

```
munos = 2×3
     1     1     1
     1     1     1
```

Para crear la matriz identidad:

```
I5 = eye(5)  % matriz identidad de orden 5
```

```
I5 = 5×5
     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
     0     0     0     1     0
     0     0     0     0     1
```

Para crear una matriz de números aleatorios:

```
maleatoria = rand(3,3)  % matriz 3x3 de números aleatorios (entre el 0 y el 1)
```

```
maleatoria = 3×3
    0.8003    0.9157    0.6557
    0.1419    0.7922    0.0357
    0.4218    0.9595    0.8491
```

Para extraer la diagonal principal de una matriz:

```
d = diag(maleatoria)
```

```
d = 3×1
    0.8003
    0.7922
    0.8491
```

Para transponer una matriz:

```
M'  % transpuesta de M
```

```
ans = 3×2
     1     4
     2     5
     3     6
```

Para saber cuál es el tamaño de una matriz:

```
size(M)  % tamaño (filas x columnas) de M
```

```
ans = 1×2
     2     3
```

En el caso de un vector, podemos comprobar su longitud con *length*:

```
v = [5 6 -5 1 0];
length(v)  % número de elementos en un vector fila
```

```
ans = 5
```

Sean A y B dos matrices tales que:

```
A = [3 -2 1; 4 -8 -1; -5 2 2]
```

```
A = 3×3
     3    -2     1
     4    -8    -1
    -5     2     2
```

```
B = [1 -1 0; -2 2 6; 7 -2 -1]
```

```
B = 3×3
     1    -1     0
    -2     2     6
     7    -2    -1
```

Es importante saber que $A * B$ no es lo mismo que $A. * B$:

```
A*B  % producto matricial
```

```
ans = 3×3
    14    -9   -13
    13   -18   -47
     5     5    10
```

```
A.*B  % producto elemento a elemento
```

```
ans = 3×3
     3     2     0
    -8   -16    -6
   -35    -4    -2
```

Cálculo del determinante de una matriz

```
det(A)   % determinante de la matriz A
```

```
ans = -68
```

Cálculo de la matriz inversa

```
inv(A)  % inversa de la matriz A
```

```
ans = 3×3
    0.2059   -0.0882   -0.1471
    0.0441   -0.1618   -0.1029
    0.4706   -0.0588    0.2353
```

Ejercicios propuestos

Ejercicio 1:

Sea la matriz $A = \begin{bmatrix} 1 & 3 & 2 & -1 \\ 0 & 0 & 2 & 5 \\ -2 & 0 & 0 & 3 \\ 1 & 1 & 1 & -1 \end{bmatrix}$

- Sustituye el elemento $a_{2,3}$ por un -3 y el $a_{1,2}$ por un 7, y llama B a la matriz resultante
- Extrae en un vector la primera fila de B, y en otro vector la segunda columna de B
- Extrae en una matriz C las filas 1 y 3 de B y en otra matriz D las columnas 2 y 4
- Añade la matriz identidad 4x4 ($I_{4\times4}$) a la derecha de B, y llama a la matriz resultante B^*. ¿Tienes algún problema?
- Añade la matriz identidad 4x4 ($I_{4\times4}$) debajo de B, y dale el nombre que quieras a la matriz resultante
- Crea un vector cuyos elementos sean los términos de la diagonal principal de A
- Crea una matriz 4x4 de números aleatorios entre 0 y 1 y súmasela a la matriz A

Ejercicio 2:

Crea las siguientes matrices:

- Matriz 4x4 con todos sus elementos cero
- Matriz 4x4 con todos sus elementos iguales a 0.5
- Matriz 3x5 con todos sus elementos iguales a π

Ejercicio 3:

Partiendo de la matriz $A = \begin{bmatrix} 1 & -1 & 0 & 3 \\ 4 & 6 & -2 & 3 \\ 5 & -3 & 1 & 0 \\ 0 & 0 & 0 & -9 \end{bmatrix}$, comprueba que se cumplen las siguientes propiedades de los determinates:

- *a)* Si todos los elementos de una fila o columna son nulos, el determinante es nulo (haz ceros en la fila o columna que prefieras)
- *b)* Si hay dos filas o columnas iguales, el determinante es nulo (haz iguales las dos filas o columnas que prefieras)
- *c)* Si intercambiamos dos filas o columnas, el valor del determinante cambia de signo (intercambia las filas o columnas que prefieras)
- *d)* $|A| = |A^t|$

Ejercicio 4:

Halla la inversa de la matriz $\begin{bmatrix} 1 & 2 \\ 4 & 8 \end{bmatrix}$. ¿Qué sucede? ¿Por qué?

Ejercicio 5:

Sean las matrices $A = \begin{bmatrix} 2 & 1 & 3 \\ -2 & -1 & 0 \\ -4 & 1 & 1 \end{bmatrix}$ y $B = \begin{bmatrix} 0 & 1 & -1 \\ 1 & 2 & -1 \\ 1 & 0 & 7 \end{bmatrix}$. Halla la matriz X que cumple que $AX = B$.

Ejercicio 6:

Sean las matrices $A = \begin{bmatrix} 2 & -3 & -5 \\ -1 & 4 & 1 \\ 1 & -3 & -4 \end{bmatrix}$ y $B = \begin{bmatrix} 2 & 1 & -4 \\ 0 & 1 & 4 \\ 1 & -2 & -3 \end{bmatrix}$. Sabiendo que $D = ABC = \begin{bmatrix} 11 & 17 \\ 17 & 1 \\ 4 & 10 \end{bmatrix}$, calcula la matriz C. Comprueba que el resultado que has obtenido es correcto.

Ejercicio 7:

Halla la matriz A sabiendo que $B^{-1} = \begin{bmatrix} 1 & -1 & 2 \\ 3 & 0 & 1 \\ 4 & -2 & -2 \end{bmatrix}$ y $(\mathrm{AB})^{-1} = \begin{bmatrix} 2 & -1 & -5 \\ 0 & 0 & 1 \\ -1 & -3 & 6 \end{bmatrix}$. Comprueba que el resultado que has obtenido es correcto.