Laboratorio Introducción a MATLAB

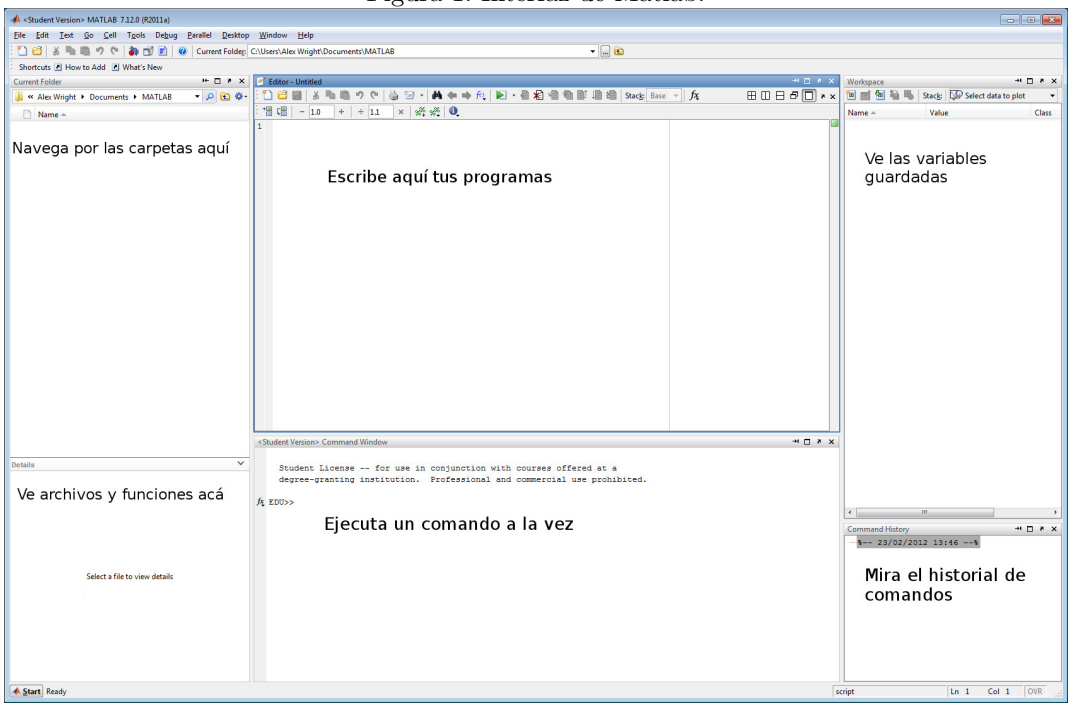
Nociones Básicas

Matlab es el nombre abreviado de *MATriz LABoratory*. Éste es un software y un lenguaje de programación que permite realizar cómputos científicos y técnicos en general.

1. 1. Interfaz

Al iniciar *Matlab*, se verán ventanas con distinta información, entre ellas:

Figura 1: Interfaz de Matlab.



- **Editor**: Aquí se escriben las líneas de código en un script o function (detalle en sección 4). Es aquí donde se realiza la mayoría del trabajo.
- **Ventana de comando**: Se ven los resultados y/o operaciones que indicadas desde el editor. Además, aquí se puede escribir una **sóla línea de instrucción**, ya sea para ejecutar un script o para realizar operaciones (menos utilizado).
- **Área de Trabajo**: Aquí se almacenan las variables utilizadas junto a su valor númerico.
- **Historial de comando**: Aquí se almacenan las variables utilizadas junto con su valor númerico.
- **Historial de comando**: Se ven las líneas de comando ejecutadas. Esto incluye los programas ejecutados.
- **Carpeta**: Muestra los programas en la carpeta actual. Es importante indicar que para un *script* utilice una función, esta debe estar incluida en la carpeta.

Las ventanas más importantes son el **editor** y la **ventana de comandos**, por lo que el resto de las ventanas puede minimizarse. Para realizar esto se presiona el ícono que está junto a la X en la esquina superior derecha

de cada ventana, y se muestran varias opciones. Además de minimizar ventanas, también se pueden acomodar. Normalmente, la ventana de comandos aparece como na aplicación aparte, pero usando **dock**, se puede llevar a la aplicación principal de *Matlab* y tener todo en el mismo lugar, Para mover las ventanas a distintas posiciones, basta con seleccionar la barra superior y moverla a la posición deseada.

2. 2. Operaciones

Al introducir una variable u operación en *Matlab* y ejecutarla ,esta aparecerá en la ventana de comando. Si la variable u operación termina con ";", esta no aparecerá en pantalla. Esto es útil cuando se quiere observar sólo el resultado de la operación en la pantalla y no todas las variables u operaciones. A continuación daremos una serie de comandos que muestran como trabajar con escalares, vectores y matrices.

```
a = 1; %Comentario en MATLAB
b = 2;
c = a^2+b^2 %Este resultado si se muestra en pantalla.
```

Cada vez que se utiliza una variable, esta es almacenada y puede ser utilizada para posteriores cálculos. *Matlab* puede diferenciar mayúsculas y minúsculas, por lo tanto la variable a es distinta a la variable A.

```
whos %Comando para revisar las variables almacenadas
clear a,b %Comando para eliminar una o mas variables
clear all %Comando para eliminar todas las variables
clc %Comando para limpiar la pantalla de comando
```

Si se realiza una operación sin almacenarla en una variable, ésta será representada como *ans*. El comando *format* permite cambiar la forma en la que se representan los números en la pantalla.

```
format long %devuelve 0.14285714285714
format short e %devuelve 1.4286 e-01
format long e %devuelve 1.428571428571428e-01
format short g %devuelve 0.14286
format long g %devuelve 0.14285714285713
```

Las operaciones realizadas tienen orden de importancia, por lo que utilizar paréntesis permite cambiar la secuencia. En la tabla 1 se muestran operaciones entre escalares.

```
a*b^2 %Primero obtiene el cuadrado de b
(a*b)^2 %Primero multiplica a y b
```

+	Adición
-	Resta
*	Multiplicación
/	División
^	Potencia

Tabla 1: Operaciones aritméticas

También existen funciones y constantes dentro de *Matlab*.

```
pi
abs(-1/7) %Valor absoluto de un escalar
cos(pi/2)coseno
sin(pi/2) %seno
sqrt(4) %raiz
exp(7) %funcion potencial
log(10) %Logaritmo natural
```

3. 3. Vectores

Se pueden almacenar vectores dentro de variables. Por ejemplo:

$$Vector\ fila\ v = (1\ 2\ 3\ 4)$$

```
v = [1 2 3 4] Componentes separado por un espacio
```

$$Vector\ columnaw = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}$$

```
v = [1;2;3;4] Componentes separados por un punto y coma
```

También es posible crear vectores cuando se indica un incremento.

```
i=1:100 %vector fila del 1 al 100 con espaciado 1
j=2:4:156 %vector del 2 al 256 con espaciado 4
```

+	Adición
-	Resta
*	Producto Matricial
\	Inversa por la izquierda ($A^{-1}b$)

Tabla 2: Operaciones aritméticas en matrices

A continuación se muestran algunos ejemplos. Realizar uno a la vez.

```
u=[1 2 3 4 5 6 7 8]
v=8:-1:1
u+v v' %Vector Transpuesto
u*v'%u por v traspuesto (producto interior)
sqrt(u*u') %Norma del vector u
sin(u) %seno de cada elemento de u.
```

.*	Producto de elemento por elemento
./	División Elemento por elemento
.^	Potencia elemento por elemento

Tabla 3: Operaciones aritméticas elemento por elemento

Por último, existen operaciones sobre cada elemento de la matriz.

```
u.*v % Entrega un vector cuyas componentes son el producto entre las componentes u y v
u./v
u. ^3
```

Para ingresar una matriz de dos dimensiones $A_{(3,3)} = \begin{pmatrix} 4 & 8 & 9 \\ 2 & 0 & -2 \\ 4 & 1 & 3 \end{pmatrix}$.

```
A= [ 4 8 9; 2 0 -1; 4 1 3]
inv(A)  % inversa de la matriz
A'  % traspuesta de la matriz
det(A)  determinante de la matriz
eig(A)  %Valores propios de la matriz
```

Funciones útiles para manejar matrices son el comando *size* y *length*

```
length(A)  %longitud de A
size(A)  %El primer valor corresponde al número de filas y el segundo al número de columnas
```

Notar que un escalar es una matriz $1X1$ y que un vector columna es una matriz de $nX1$.
A continuación, se muestran algunas operaciones que se pueden realizar con matrices.

```
A = [1 2 5 5;2 -1 6 0;3 0 -1 4;-1 2 4 8;1 2 3 6] % Se ingresa la matriz A
A(2,3)  % Muestra el elemento e la posicióo (2,3)
A(:,4)  %Muestra toda la columna del cuarto elemento
A(2,:)  %Muestra toda la fila del segundo elemento
A(1:3,2)  %Muestra el elemento 1 al 3 de la columna 2
[m,n]=size(A)  %Almacena la dimensión de la matriz en las variables m y n
```

En la tabla 4 se muestra comandos para generar matrices con distintos valores

eye	Matriz Identidad
zeros	Matriz de Ceros
diag	Si x es un vector, diag(x) crea una matriz diagonal cuya diagonal son las componentes de x. Si Aes una matriz cuadrada, diag(A) es un vector formado por la diagonal de A.
triu	Parte triangular superior de la matriz
tril	Parte triangular inferior de la matriz
rand	Matriz generada aleatoriamente con valores entre 0 y 1
hilb	Matriz de Hilbert

Tabla 4: Comandos para generar matrices

Los comandos de la tabla 4 combinados son muy útiles para la construcción de matrices.

```
A=[1 2; 5 -2]
B=[10 30; A]
C=[eye(2) zeros(2,2); zeros(2,2) A]
D=diag(diag(c))
```

4. 4. Funciones Lógicas

Para realizar operaciones dentro de *Matlab*, a veces es necesario repetirlo una cierta cantidad de veces, o hasta que se cumpla una condición. Es por eso que existen funciones lógicas que ejecutaran una operación según el usuario lo desee.

- **for**:

```
for i=vi:in:vf
    instrucciones
end
```

donde vi es el valor inicial, in es el incremento, y vf es el valor final. Esto significa que la instrucción se realizará $\frac{vf-vi}{in}$ veces. Es muy útil para recorrer matrices.

```
for i=1:length(A)
    instrucciones
end
```

- **while**:

```
while relacion
    instrucciones
end
```

La instrucción se realizará hasta que la relación se cumpla. Por ejemplo:

```
i==4 % mientras i sea igual a 4
i<n % mientras i sea menor que n
i>=4 % mientras i sea mayor p igual a n
i =4 % mientras i sea distinto de n
```

- **if**:

```
if relacion
    instrucciones
end
```

La instrucción se realiza si se cumple cierta condición. Además, existe también el *else*:

```
if relacion
    instruccione 1
end
else
    instrucción 2
end
```

Si la relación es verdadera, se realizará la instrucción 1. De no ser así, se realizará la instrucción 2. Para una o más condiciones, se pueden usar las siguientes conexiones:

```
a & b % a y b
a | b % a o b
a xor b % a o excluyente b
```

5. 5. Gráficar en Matlab

5.1. 5.1 Gráficos de 2 variables

Se pueden realizar gráficos simples con el comando *plot*:

```
x= 0:0.01:10;
y=sin(x);
plot(x,y)
plot(x,y),'r') %averiguar diferencia
plot(x,y,'*') %averiguar diferencia
plot(x,y,'*y') %averiguar diferencia
z=sin(x).^2;
plot(x,y,'r',x,z,'b') %2 curvas en el mismo gráfico
```

También pueden hacerse varios gráficos a la vez agregando el comando *subplot*:

```
x=1:0.01:10;
y=sin(4*x);
subplot(2,2,1) %Se divide la pantalla de gráficos en 2 filas y 2 columnas, se utiliza la primera
pantalla
plot(x,y)
subplot(2,2,2) %Se utiliza la segunda pantalla
plot(x,y,'r')
subplot(2,2,3) %Se utiliza la tercera pantalla
plot(x,y,'*')
```

```
subplot(2,2,4) %Se utiliza la cuarta pantalla
plot(x,y,'*y')
```

5.2. 5.2 Gráficos en 2D

Para hacer gráficos en 3D, se utiliza el comando *surf*, el cual se puede utilizar de distintas maneras. *surf(Z)* crea una superficie sombrada 3D a partir de la componente z de la matriz Z, usando $x = 1 : n$ y $y = 1 : m$, donde $[m, n] = size(Z)$. La altura Z es una función de valor único que se define sobre un grid rectangular. Z especifica tanto el color como la altura, por tanto, la altura es proporcional al color. *surf(X,Y,Z)* utiliza Z para el color y la altura, mientras que X e Y son vectores o matrices que definen las componentes x e y de la superficie. Por ejemplo:

```
x=0:0.01:1;
y=0:0.01:1;
z=zeros(length(x),length(y));
for i=1:length(x)
      for j=1:length(x)
          z(i,j)=sin(2*pi*x(1,i))+sin(2*pi*y(1,j));
      end
end
surf(x,y,z)      %notar diferencia entre ejes
pause() %detiene el script hasta que se apriete una tecla
surf(z)      %notar diferencia entre ejes
```

Por otro lado, podemos utilizar el comando *contourf(Z)* el cual crea muestra isolíneas calculadas a partir de la matriz Z y llena el área entre las isolíneas, los valores de Z se interpretan como la altura respecto al plano *(x,y)*:

```
contourf(x,y,z)      %notar diferencia entre ejes
pause(5) %detiene el script hasta que pasen 5 segundos
contourf(z)      %notar diferencia entre ejes
pause(5)
z=peaks(20)    %función de ejemplo
contourf(z)
colorbar
```

Finalmente, si se quiere saber algo más sobre algún comando, se puede utilizar la función *help*:

```
help surf
help plot
help contourf
help colorbar
```