Pontificia Universidad Católica de Chile
Facultad de Economía y Administración
Instituto de Economía
Profesor: Alexandre Janiak

Teoría Macroeconómica I - EAE320B

Leonardo Montoya (lalms@uc.cl) - Ignacio Rojas (irojasking@gmail.com)

Ayudantia 1 - Iniciación de MATLAB

1. Descargar Matlab

Para poder realizar esto, uno tiene que seguir los siguientes pasos:

1. Ingresar a su https://portal.uc.cl/ e iniciar sesión en su cuenta.
2. Una vez dentro, dirigirse a la sección de **Herramientas** y en el apartado de convenios de software, suscribirse a **MathWorks/Matlab**. Luego, seguir las instrucciones proporcionadas en el documento PDF.

2. División de Matlab

Después de descargar Matlab, lo primero que vemos es lo siguiente:

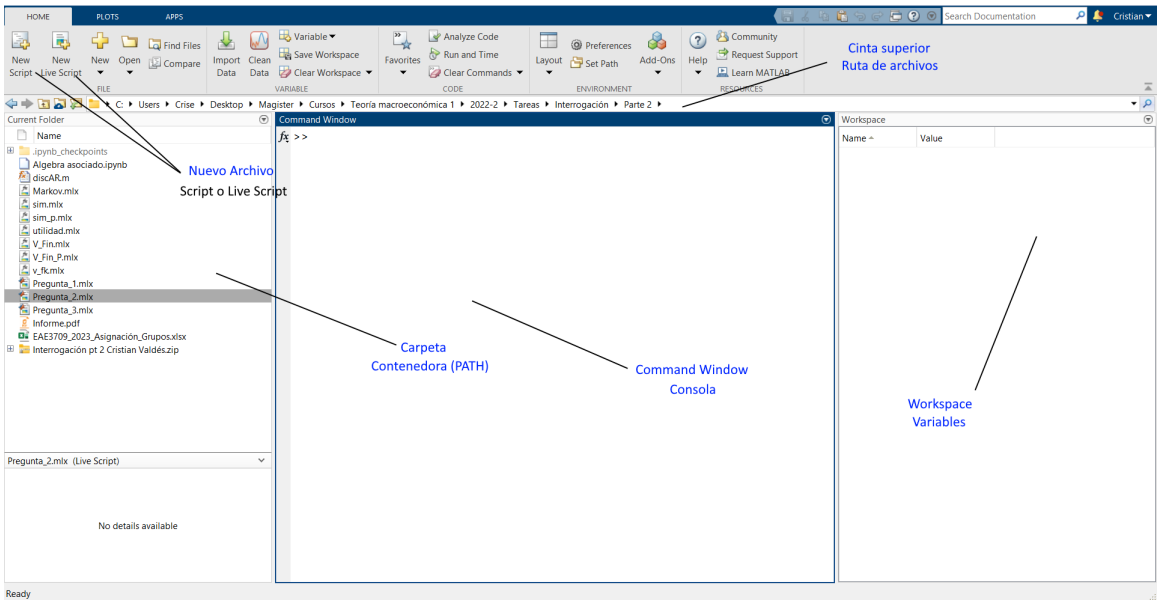

Figura 1: Lo primero que vemos al abrir Matlab (software).

Dentro de esta tenemos divisiones importantes que serían las siguientes:

- **Cinta superior:** Muestra la ruta donde se ubican los archivos dentro del directorio; todos los archivos que se extraigan quedarán guardados por defecto en la ruta señalada.
- **Current Folder (Carpeta contenedora):** Detalla los distintos archivos dentro de ese directorio.
- **Command Window:** Corresponde a la consola de trabajo, similar a la de otros lenguajes de programación. En ella se detallan los resultados obtenidos y se puede trabajar directamente en ella. Sin embargo, por comodidad, trabajaremos en archivos llamados **scripts**.
- **Script:** Archivo en el cual escribiremos nuestros programas y algoritmos. Los scripts guardados tendrán la terminación ".m".
- **Live Script:** Formato especial de script basado en el sistema de bloques. Es más amigable para el trabajo en equipo y presentaciones. La extensión de estos archivos es ".mlx".
- **Run codes:** A través del comando *Run*, se ejecutará todo el código escrito. Mediante la función *Run section*, solamente se ejecutará el código previamente seleccionado o delimitado (como dice su nombre, solo la sección).

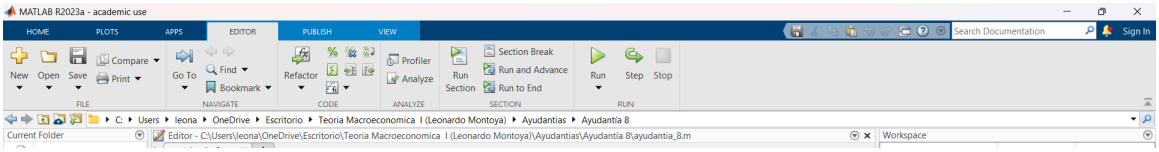

Figura 2: Cómo correr mi código con "Run".

En *Windows*, a través de *F9*, se pueden ejecutar líneas de códigos específicas (no se recomienda si tiene funciones definidas en el mismo script).

3. Comandos importantes

- **close all.** Cierra todos los gráficos que están abiertos en pantalla.
- **clear.** Borra todas las variables existentes en el workspace.
- **clc.** Limpia la consola (Borra los resultados).
- **Punto y coma (;).** Carácter utilizado al finalizar cada sentencia de nuestro script, su utilidad radica en silenciar la linea de código que se está ejecutando, impidiendo que esta aparezca en la consola.
- **Porcentaje (%).** Al escribirlo, convierte todo lo que está a la derecha de el en texto plano (no ejecutable). Sirve para escribir comentarios y describir el código.
- **Doble porcentaje (%%).** Genera una nueva sección (solo Script).

- **help.** Al anteponerlo a un comando, lo explica.
- **Dos puntos (:).** Todos los elementos de un conjunto.
- **rng(seed).** Establece la semilla del generador de números aleatorios utilizando el algoritmo del generador actual.

4. Generar variables, vectores y matrices

Para crear una variable (vector o matriz) nueva, introduzca el nombre de la variable (vector o matriz) en la ventana de comandos, seguido por un signo igual (=) y el valor que desea asignar a la variable (vector o matriz).

```
%% Variables
a = 32;
b = 621;
c = a + b;

%% Vectores
v_1 = [a,b,c]; %vector generado con las variables antes definidas (fila)
v_2 = v_1'; %vector anterior pero transpuesto (columna)
z_1 = 1:100; %vector que va desde el 1 hasta el 100, se veria de la forma
% [1,2,3,...,99,100]
z_2 = linspace(0,100,51); %vector COLUMNA de 51 puntos equisdistante desde
% el 0 hasta el 100
% Que hacer si quiero volver este vector columna un vector fila?

%% Matrices
M_1 = [1,2,3,7;2,4,6,3;3,6,9,5]; %matriz generada con "," y ";"
%M_1_v2 = [1 2 3 7; 2 4 6 3; 3 6 9 5]; %matriz generada con " " y ";"
valor_max_por_columna_M_1 = max(M_1);
valor_max_por_fila_M_1 = max(M_1,[],2);
vector_suma_por_columna_de_M_1 = sum(M_1);
vector_suma_por_fila_de_M_1 = sum(M_1,2);
%vector_suma_por_columna_de_M_1_v2 = sum(M_1,1);

%% Matrices populares o muy usadas
matriz_identidad = eye(5);
matriz_identidad_modificada = eye(3,5); %notar que 3 es la cantidad de
% filas y 5 en las cantidad de columnas
matriz_de_ceros = zeros(6,7);
matriz_de_unos = ones(3,5);
% Que hacer si quiero una matriz de 6 filas y 7 columnas de puros 20?
matriz_de_20 = 20.*ones(6,7);

%% Ubicacion en matrices
% Por ejemplo en nuestra matriz M_1 podemos hacer lo siguiente (3x4)
elemento_3_3_de_M_1 = M_1(3,3);
columna_4_de_M_1 = M_1(:,4);
fila_3_de_M_1 = M_1(3,:);
ultimo_elemento_de_las_primeras_dos_filas_de_M_1 = M_1(1:2,end);
vector_columna_con_todos_los_valores_mayor_a_4_de_M_1 = M_1(M_1>4);
% Como lo vuelvo fila?
% Como puedo reemplazar filas o columnas en la matriz?
M_1(:,1)=v_2; %en este caso estamos haciendo es reemplazar la primera columna
% Como saber que dimensiones tiene mi matriz?
cant_filas_de_M_1 = size(M_1,1);
cant_columnas_de_M_1 = size(M_1,2);
% Como reemplazar valores segun condiciones?
% Ejemplo de contar cuantos valores de M_1 son mayores o igual a 5 y luego
% reemplazar estos por 0
contador = 0;
for i = 1:cant_filas_de_M_1
    for j = 1:cant_columnas_de_M_1
```

```
        if M_1(i,j)>=5
            contador = contador +1;
            M_1(i,j)=0;
        end
    end
end

```

5. Distribuciones de probabilidad

```
clc; close all; clear;
rng(73)

n = 1000; % numero de filas
c = 2; % numero de columnas

%% Distribucion uniforme
x1 = rand(n,c); % U(0,1)
x1_2 = 55 + (65-55)*rand(n,c); % U(55,65), ie, distribucion unirforme en (55,65)

%% Distribucion normal
x2 = normrnd(0,10,n,c); %N(0,10^2)
x2_2 = randn(n,c)*6 + 70; %N(70,6), ie, distribucion normal con media 70 y desviacion estandar 6

%% Otras
x3 = chi2rnd(15,n,c); %distribucion chi-cuadrado con 15 grados de libertad
%x4 = 50 + trnd(2,n,c); %distribucion t student centrada en 0 con 2 grados de libertad.

%% Mixturas
% Crear la variable x4 donde esta es una mixtura de variables utilizando
% x2 y x3. La mixtura en este caso es una distribucion en donde un 50% de
% los casos se comporta como x2 y el 50 restante como x3.

tic
x4= zeros(n,c);
for i= 1:n
    for j = 1:c
        if rand(n,c)>0.5
            x4(i,j) = x2(i,j);
        else
            x4(i,j) = x3(i,j);
        end
    end
end
toc

%metodo alternativo
u = rand(n,c); %mixtura de variables utilizando X2 y X3, 50% de cada una.
tic
x4_v2= (u>0.5).*x2 + (1-(u>0.5)).*x3;
toc

```

6. Introducción a funciones y sentencias de control

1. La función de utilidad de tipo CRRA (Constant Relative Risk Aversion) es una representación de preferencias muy utilizada en macroeconomía, la cual está definida como:

$$u(c) = \begin{cases} \dfrac{c^{1-\sigma} - 1}{1-\sigma} & \text{si } \sigma > 0 \text{ y } \sigma \neq 1 \\ \ln c & \text{si } \sigma = 1 \end{cases}$$

2. Calcule la utilidad evaluada en c (una grilla de 500 puntos entre 0 y 10), usando $\sigma = [0\ 1/5\ 1/3\ 1/2\ 1\ 2]$.

```
%% Funciones distinta forma
clc; close all; clear;

% Parametros
sigma= [0, 1/5, 1/3, 1/2, 1, 2];
c= linspace(0,10,500)';

% Forma 1: Llamar al a funcion desde el final del archivo
utilidad = zeros(length(c),size(sigma,2));
for i = 1:size(sigma,2)
    utilidad(:,i) = util(c,sigma(i));
end

% Forma 2: Funcion anonima "u"
utilidad_v2= zeros(length(c),length(sigma));
for s= 1:length(sigma)
    % Especificacion Utilidad
    if sigma(s)~= 1
        util_v2= @(c) ( (c.^(1-sigma(s)) -1) /(1-sigma(s)) );
    else
        util_v2= @(c) log(c);
    end
    % Utilidad Evaluada
    utilidad_v2(:,s)= util_v2(c);
end

%% Funcion al final
function u = util(c, sigma)
    if sigma==1
        u= log(c);
    else
        u= (c.^(1-sigma)-1)/(1-sigma);
    end
end

```

Seguimiento 1

El decano, al reconocer tu potencial como estudiante prometedor, te contrata para analizar algunas estadísticas dentro de la universidad. Te asigna el estudio del comportamiento de los estudiantes de la facultad, específicamente los tiempos que utilizan durante la semana para cambiar de salas, estudiar y llegar a la universidad.

Después de unas rigurosas entrevistas, has obtenido los siguientes resultados (en minutos) para un total de 418 estudiantes que ingresaron el año pasado y fueron observados durante un mes (4 semanas):

- Tiempo en cambio de salas: $Salas \sim U(30, 100)$.
- Tiempo de estudio: $Estudio \sim N(0, 3600)$.
- Tiempo de viaje: $Viaje \sim U(1, 600)$.

En función de lo anterior, se le propone resolver los siguientes incisos:

1. Utilizando loops, cree la matriz `M_DECANO_LOOP` que contenga como fila a cada estudiante y como columna el tiempo total de compra del mismo cliente para cada semana.[1]
2. Obtenga la misma matriz del ítem anterior pero de forma matricial. Nombre a esta matriz `M_DECANO_MATRICIAL`.
3. Compruebe que ambas matrices son iguales.
4. Identifique al estudiante que más tiempo usó de su semana en:
   - Cambios de sala.
   - Estudio.
   - Viaje.
   - Total.

   Realice esto para cada semana y calcule, a su vez, el máximo del mes.
5. Realice lo mismo que en el inciso anterior pero con el mínimo, es decir, el que menos tiempo usó de su semana.
6. Grafique en 2 paneles (uno para el máximo y otro para el mínimo) los respectivos máximos y mínimos de cada semana.[2]

Nota. Use la semilla número 73.

[1] El tiempo total es la suma de los tiempos en cambios de sala, estudio y viaje.

[2] Para esto revise la documentación o use help para figure, plot, subplot, hold on, hold off, title, legend, xlabel, ylabel, xticks, yticks y saveas.