Sistemas de Control Inteligente

Práctica 0
Introducción a Matlab (I)

Jorge Revenga Martín de Vidales
Ángel Salgado

Grado en Ingeniería Informática
Universidad de Alcalá

15 de octubre de 2023

Índice

1. Ejercicio 1. Matrices y vectores.

1.1. Código

```
% Paso 1: Crear la matriz A y el vector v
A = [1 2; 3 4; 5 6; 7 8];
v = [14; 16; 18; 20];

% Paso 2: Obtener y visualizar la matriz B concatenando A y v
B = [A v];

% Paso 3: Obtener y visualizar un vector fila concatenando las filas de B
vector_fila = [B(1,:) B(2,:) B(3,:) B(4,:)]
% Opción 2:
vector_fila = reshape(B', 1, []);

% Paso 4: Obtener y visualizar un vector columna concatenando las columnas de B
vector_columna = [B(:,1); B(:,2); B(:,3)]
% Opción 2:
vector_columna = reshape(B, [], 1);

% Visualizar los resultados
disp('Matriz B:');
disp(B);
disp('Vector fila resultante de concatenar filas de B:');
disp(vector_fila);
disp('Vector columna resultante de concatenar columnas de B:');
disp(vector_columna);
```

1.2. Ejecución

```
Matriz B:
     1     2    14
     3     4    16
     5     6    18
     7     8    20

Vector fila resultante de concatenar filas de B:
     1     2    14     3     4    16     5     6    18     7     8    20

Vector columna resultante de concatenar columnas de B:
     1
     3
     5
     7
     2
     4
     6
     8
    14
    16
    18
    20
```

Figura 1: Ejecución ejercicio 1.

2. Ejercicio 2. Matrices y vectores.

2.1. Código

```
% Paso 1: Solicitar al usuario el tamaño de la matriz cuadrada
n = input('Indique el tamaño de la matriz: ');
% Paso 2: Generar una matriz aleatoria de tamaño n x n
matriz = rand(n);
% a) Mostrar la matriz generada
disp('Matriz generada:');
disp(matriz);
% b) Obtener una segunda matriz con las columnas impares de la matriz inicial
matriz_impares = matriz(:, 1:2:end);
disp('Segunda matriz con columnas impares:');
disp(matriz_impares);
% c) Obtener y mostrar los elementos de la diagonal de la matriz generada
diagonal = diag(matriz);
disp('Elementos de la diagonal:');
disp(diagonal);
% d) Calcular y graficar el máximo, mínimo, medio y varianza de cada fila
maximos = max(matriz, [], 2);
minimos = min(matriz, [], 2);
medios = mean(matriz, 2);
varianzas = var(matriz, 0, 2);
% Crear un gráfico de barras para los valores calculados
figure;
bar([maximos, minimos, medios, varianzas]);
title('Valores por fila');
xlabel('Número de fila');
ylabel('Valor');
legend('Máximo', 'Mínimo', 'Medio', 'Varianza');
```

2.2. Ejecución

```
Indique el tamaño de la matriz: 5
Matriz generada:
    0.9797    0.5949    0.1174    0.0855    0.7303
    0.4389    0.2622    0.2967    0.2625    0.4886
    0.1111    0.6028    0.3188    0.8010    0.5785
    0.2581    0.7112    0.4242    0.0292    0.2373
    0.4087    0.2217    0.5079    0.9289    0.4588

Segunda matriz con columnas impares:
    0.9797    0.1174    0.7303
    0.4389    0.2967    0.4886
    0.1111    0.3188    0.5785
    0.2581    0.4242    0.2373
    0.4087    0.5079    0.4588

Elementos de la diagonal:
    0.9797
    0.2622
    0.3188
    0.0292
    0.4588
```

Figura 2: Ejecución ejercicio 2.

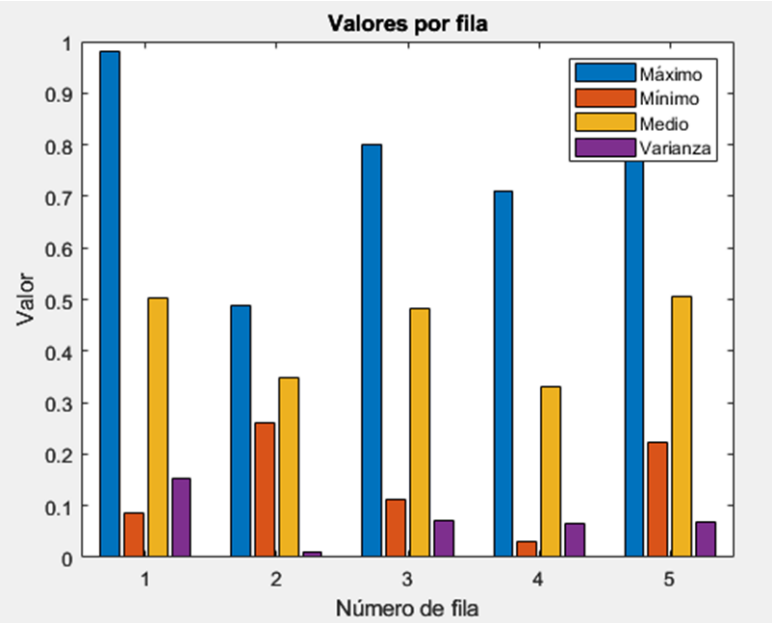


Figura 3: Gráfico ejecución ejercicio 2.

3. Ejercicio 3. Matrices y vectores.

3.1. Código

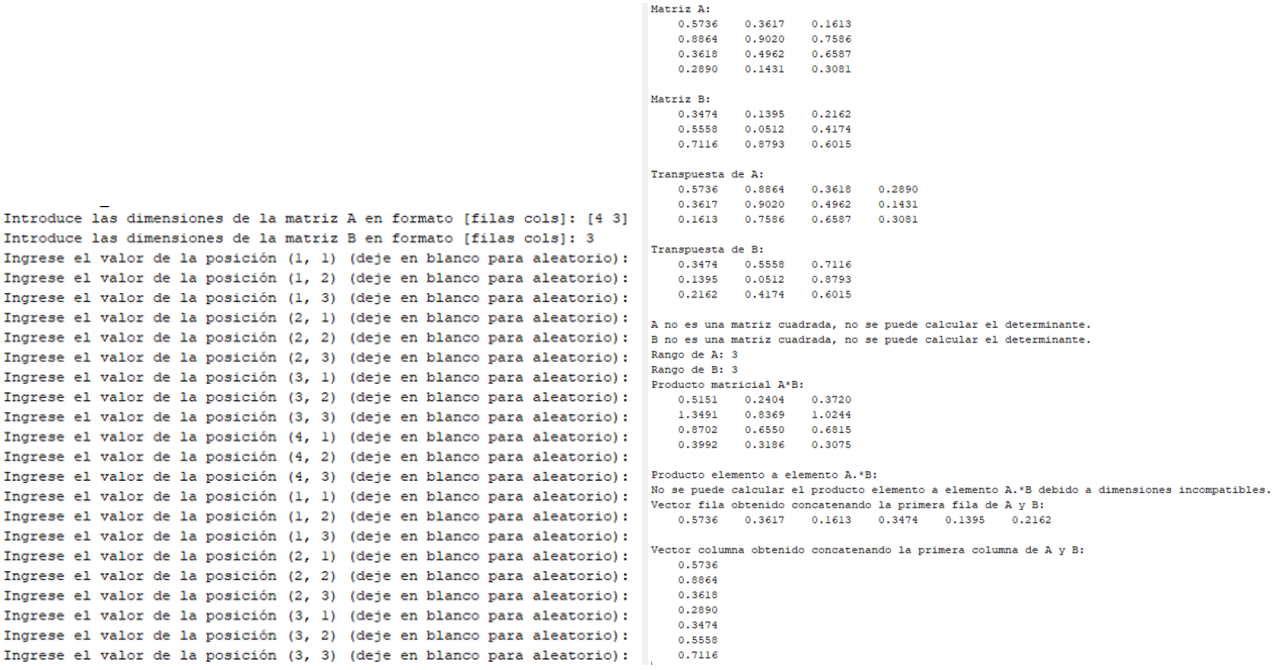
IntroducirMatriz.m

```
% Función para generar y rellenar una matriz
function Matriz = IntroducirMatriz(Dimensiones)
    if numel(Dimensiones) == 1  % Si solo se proporciona un número, asumimos una matriz cuadrada
        filas = Dimensiones;
        cols = Dimensiones;
    elseif numel(Dimensiones) == 2
        filas = Dimensiones(1);
        cols = Dimensiones(2);
    else
        error('Formato de dimensiones incorrecto. Debe ser [filas cols].');
    end

    Matriz = zeros(filas, cols);

    for i = 1:filas
        for j = 1:cols
            fprintf('Ingrese el valor de la posición (%d, %d) (deje en blanco para aleatorio): ', i, j);
            valor = input('','s');  % Leer como cadena de caracteres

            if isempty(valor) || isempty(strtrim(valor))  % Si el usuario ingresa un valor vacío
                Matriz(i, j) = rand();  % Rellenar con valor aleatorio
            else
                Matriz(i, j) = str2double(valor);  % Convertir entrada a número
            end
        end
    end
end
```

```
% Paso 1: Solicitar las dimensiones de las matrices al usuario
dimensiones_A = input('Introduce las dimensiones de la matriz A en formato [filas cols]: ');
dimensiones_B = input('Introduce las dimensiones de la matriz B en formato [filas cols]: ');

% Paso 2: Generar las matrices A y B
A = IntroducirMatriz(dimensiones_A);
```

```
B = IntroducirMatriz(dimensiones_B);

% Paso 4: Realizar los cálculos y mostrar los resultados
disp('Matriz A:');
disp(A);
disp('Matriz B:');
disp(B);
disp('Transpuesta de A:');
disp(A');
disp('Transpuesta de B:');
disp(B');

if isequal(size(A), size(B)) && size(A,1) == size(A,2)
    disp('Determinante de A: ');
    disp(det(A));
else
    disp('A no es una matriz cuadrada, no se puede calcular el determinante.');
end

if isequal(size(B), size(A)) && size(B,1) == size(B,2)
    disp('Determinante de B: ');
    disp(det(B));
else
    disp('B no es una matriz cuadrada, no se puede calcular el determinante.');
end

disp(['Rango de A: ' num2str(rank(A))]);
disp(['Rango de B: ' num2str(rank(B))]);

disp('Producto matricial A*B:');
if size(A,2) == size(B,1)
    disp(A * B);
else
    disp('No se puede calcular el producto matricial A*B debido a dimensiones incompatibles.');
end

disp('Producto elemento a elemento A.*B:');
if isequal(size(A), size(B))
    disp(A .* B);
else
    disp('No se puede calcular el producto elemento a elemento A.*B debido a dimensiones incompatibles.');
end

vector_fila = [A(1, :), B(1, :)];
disp('Vector fila obtenido concatenando la primera fila de A y B:');
disp(vector_fila);

vector_columna = [A(:, 1); B(:, 1)];
disp('Vector columna obtenido concatenando la primera columna de A y B:');
disp(vector_columna);
```

3.2. Ejecución



```
Introduce las dimensiones de la matriz A en formato [filas cols]: [4 3]
Introduce las dimensiones de la matriz B en formato [filas cols]: 3
Ingrese el valor de la posición (1, 1) (deje en blanco para aleatorio):
Ingrese el valor de la posición (1, 2) (deje en blanco para aleatorio):
Ingrese el valor de la posición (1, 3) (deje en blanco para aleatorio):
Ingrese el valor de la posición (2, 1) (deje en blanco para aleatorio):
Ingrese el valor de la posición (2, 2) (deje en blanco para aleatorio):
Ingrese el valor de la posición (2, 3) (deje en blanco para aleatorio):
Ingrese el valor de la posición (3, 1) (deje en blanco para aleatorio):
Ingrese el valor de la posición (3, 2) (deje en blanco para aleatorio):
Ingrese el valor de la posición (3, 3) (deje en blanco para aleatorio):
Ingrese el valor de la posición (4, 1) (deje en blanco para aleatorio):
Ingrese el valor de la posición (4, 2) (deje en blanco para aleatorio):
Ingrese el valor de la posición (4, 3) (deje en blanco para aleatorio):
Ingrese el valor de la posición (1, 1) (deje en blanco para aleatorio):
Ingrese el valor de la posición (1, 2) (deje en blanco para aleatorio):
Ingrese el valor de la posición (1, 3) (deje en blanco para aleatorio):
Ingrese el valor de la posición (2, 1) (deje en blanco para aleatorio):
Ingrese el valor de la posición (2, 2) (deje en blanco para aleatorio):
Ingrese el valor de la posición (2, 3) (deje en blanco para aleatorio):
Ingrese el valor de la posición (3, 1) (deje en blanco para aleatorio):
Ingrese el valor de la posición (3, 2) (deje en blanco para aleatorio):
Ingrese el valor de la posición (3, 3) (deje en blanco para aleatorio):
Matriz A:
    0.5736    0.3617    0.1613
    0.8864    0.9020    0.7586
    0.3618    0.4962    0.6587
    0.2890    0.1431    0.3081

Matriz B:
    0.3474    0.1395    0.2162
    0.5558    0.0512    0.4174
    0.7116    0.8793    0.6015

Transpuesta de A:
    0.5736    0.8864    0.3618    0.2890
    0.3617    0.9020    0.4962    0.1431
    0.1613    0.7586    0.6587    0.3081

Transpuesta de B:
    0.3474    0.5558    0.7116
    0.1395    0.0512    0.8793
    0.2162    0.4174    0.6015

A no es una matriz cuadrada, no se puede calcular el determinante.
B no es una matriz cuadrada, no se puede calcular el determinante.
Rango de A: 3
Rango de B: 3
Producto matricial A*B:
    0.5151    0.2404    0.3720
    1.3491    0.8369    1.0244
    0.8702    0.6550    0.6815
    0.3992    0.3186    0.3075

Producto elemento a elemento A.*B:
No se puede calcular el producto elemento a elemento A.*B debido a dimensiones incompatibles.
Vector fila obtenido concatenando la primera fila de A y B:
    0.5736    0.3617    0.1613    0.3474    0.1395    0.2162

Vector columna obtenido concatenando la primera columna de A y B:
    0.5736
    0.8864
    0.3618
    0.2890
    0.3474
    0.5558
    0.7116
```

Figura 4: Ejecución ejercicio 3.

4. Ejercicio 4. Tiempo de cómputo y representación gráfica.

4.1. Código

```
% Inicializar matrices para almacenar los tiempos de procesamiento
tiempo_rango = [];
tiempo_determinante = [];

% Bucle para calcular los tiempos
for n = 1:25
    % Generar una matriz aleatoria de tamaño n x n
    matriz = rand(n, n);

    % Calcular el tiempo para el cálculo del rango
    tic;
    rango = rank(matriz);
    tiempo_rango(n) = toc;

    % Calcular el tiempo para el cálculo del determinante
    tic;
    determinante = det(matriz);
    tiempo_determinante(n) = toc;
end

% Crear una figura para representar los tiempos
figure;
plot(tiempo_rango);
hold on;
plot(tiempo_determinante);
xlabel('Tamaño de la matriz');
ylabel('Tiempo (segundos)');
title('Tiempo de cálculo del rango y determinante en función del tamaño de la matriz');
legend('Tiempo de cálculo del rango', 'Tiempo de cálculo del determinante');

hold off;
```

4.2. Ejecución

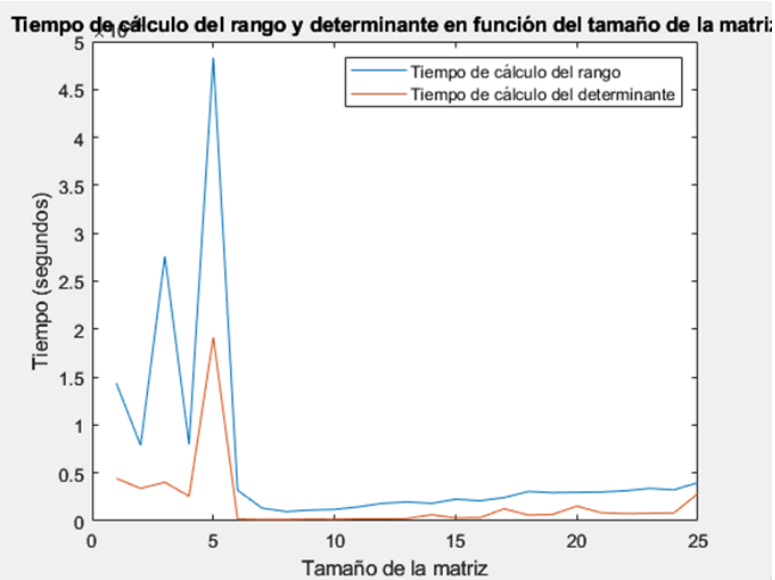


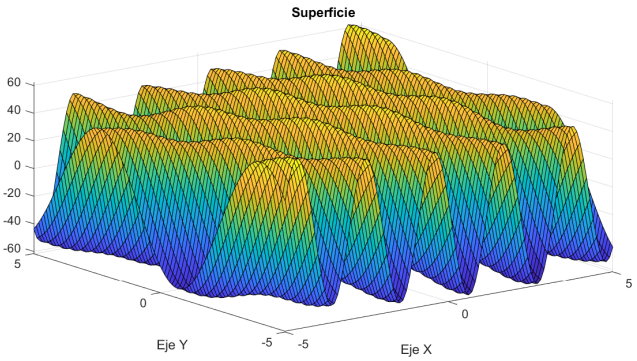
Figura 5: Gráfico ejecución ejercicio 4.

5. Ejercicio 5. Representación gráfica en 3D.

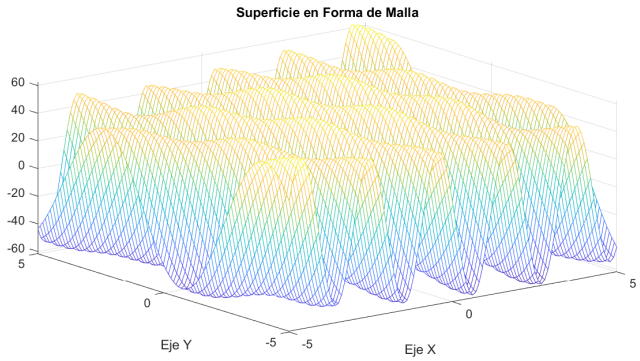
5.1. Código

```
% Definir el rango de valores para x y y
x = -5:0.1:5;
y = -5:0.1:5;

% Crear una malla de valores para x y y
[X, Y] = meshgrid(x, y);

% Calcular la función z en función de x y y
Z = Y .* sin(pi * X / 10) + 5 * cos((X.^2 + Y.^2) / 8) + cos(X + Y) * cos(3 * X - Y);

figure;

subplot(2, 2, 1);
surf(X, Y, Z);
title('Superficie');
xlabel('Eje X');
ylabel('Eje Y');

subplot(2, 2, 2);
mesh(X, Y, Z);
title('Superficie en Forma de Malla');
xlabel('Eje X');
ylabel('Eje Y');

subplot(2, 2, 3);
contourf(X, Y, Z);
colorbar;
title('Contorno con Barra de Color');
xlabel('Eje X');
ylabel('Eje Y');

% Ajustar tamaño a la pantalla
set(gcf, 'Position', get(0, 'Screensize'));

```

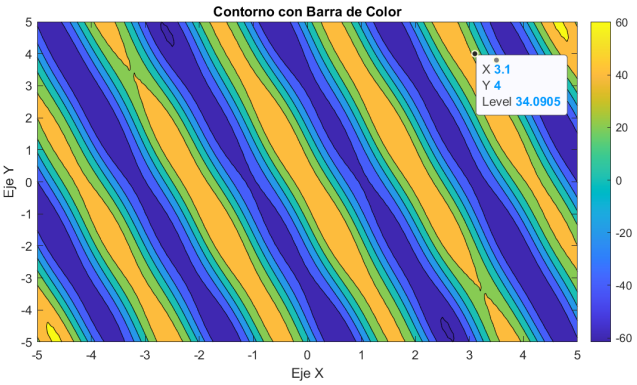
5.2. Ejecución



(a) Gráfico de superficie.



(b) Gráfico de superficie en forma de malla.



(c) Gráfico de contorno con barra de color.

Figura 6: Gráficos ejecución ejercicio 5.

6. Ejercicio 6. Sistemas lineales.

6.1. Código

```
% Definir las matrices A y b
A = [0 2 10 7; 2 7 7 1; 1 9 0 5; 4 0 0 6; 2 8 4 1; 10 5 0 3; 2 6 4 0; 1 1 9 3; 6 4 8 2; 0 3 0 9];
As = [90; 59; 15; 10; 80; 17; 93; 51; 41; 76];

b = [0.110 0 1 0; 0 3.260 0 1; 0.425 0 1 0; 0 3.574 0 1; 0.739 0 1 0; 0 3.888 0 1; 1.054 0 1 0; 0 4.202 0 1;
↪  1.368 0 1 0; 0 4.516 0 1];
bs = [317; 237; 319; 239; 321; 241; 323; 243; 325; 245];

% 1. Obtener el número de condición de la matriz A
c = cond(A);
disp(['Número de condición de A: ' num2str(c)]);

% 2. Resolver los sistemas de ecuaciones A = As y b = bs
X = linsolve(A, As);
disp('Solución del sistema de ecuaciones A:');
disp(X);
Y = linsolve(b,bs);
disp('Solución del sistema de ecuaciones b (sin ruido):');
disp(Y)

% 3. Añadir ruido a la matriz b y resolver el sistema
r = normrnd(0, 1, 10, 1);
b2 = bs + r;

```

```
% 4. Resolver el sistema de ecuaciones b = bs
B = b;
Y = linsolve(B, b2);
disp('Solución del sistema de ecuaciones b (con ruido):');
disp(Y);
```

6.2. Ejecución

```
Número de condición de A: 2.7257
Solución del sistema de ecuaciones A:
   -2.2571
    4.9336
    5.2986
    3.5649

Solución del sistema de ecuaciones b (sin ruido):
    6.3593
    6.3694
  316.2992
  216.2357

Solución del sistema de ecuaciones b (con ruido):
    4.3270
    6.0054
  317.5485
  217.5285
```

Figura 7: Gráfico ejecución ejercicio 6.

7. Ejercicio 7. Polinomios.

7.1. Código

raices.m

```
function [solucion, reales, complejas] = raices(poli_1, poli_2)
    % Recoge los arrays con los que se crean los polinomios
    p1 = poly2sym(poli_1);
    p2 = poly2sym(poli_2);

    % Solicita al usuario a cuál de los polinomios o al producto se aplicará la solución
    fprintf('Elija a cuál de los polinomios o al producto desea aplicar la solución:\n');
    fprintf('1. Polinomio 1\n');
    fprintf('2. Polinomio 2\n');
    fprintf('3. Producto de los polinomios\n');

    choice = input('Ingrese el número correspondiente: ');

    switch choice
        case 1
            % Calcular raíces del primer polinomio
            solucion = roots(poli_1);
        case 2
            % Calcular raíces del segundo polinomio
            solucion = roots(poli_2);
        case 3
            % Calcular raíces del producto de los polinomios
```

```
            prod_poli = conv(poli_1, poli_2);
            solucion = roots(prod_poli);
        otherwise
            error('Opción no válida. Debe elegir 1, 2 o 3.');
    end

    % Clasificar las raíces
    reales = sum(isreal(solucion));
    complejas = length(solucion) - reales;

    % Representar las raíces en el plano complejo
    figure;
    hold on;
    plot(real(solucion), imag(solucion), 'ro');
    xlabel('Parte Real');
    ylabel('Parte Imaginaria');
    title('Ubicación de las raíces en el plano complejo');
    hold off;

end
```

```
[solucion, reales, complejas] = raices([1 2 2], [1 3]);
fprintf('Raíces: ');
disp(solucion);
fprintf('Raíces reales: %d\n', reales);
fprintf('Raíces complejas: %d\n', complejas);
```

7.2. Ejecución

```
Elija a cuál de los polinomios o al producto desea aplicar la solución:
1. Polinomio 1
2. Polinomio 2
3. Producto de los polinomios
Ingrese el número correspondiente: 3
Raíces:   -3.0000 + 0.0000i
  -1.0000 + 1.0000i
  -1.0000 - 1.0000i

Raíces reales: 0
Raíces complejas: 3
```

Figura 8: Ejecución ejercicio 7.

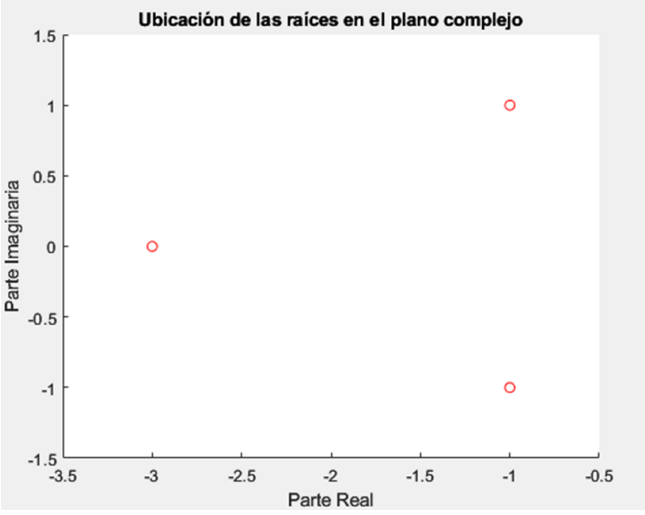


Figura 9: Gráfico ejecución ejercicio 7.