Tabla

Descripción generada automáticamente

Calculo los coeficientes

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| X | f [xi | ∆f [xi] | ∆f 2[xi] | ∆f3 [xi] | ∆f4 [xi] | ∆f 5[xi] | ∆f6 [xi] |
| 1 | 4,5 | 6,98 | -7,67 | 2,1 | 8,2 | -15,92 | 22,89 |
| 1,25 | 2,48 | -0,69 | -5,57 | 10,3 | -7,72 | 6,97 |  |
| 1,5 | 1,79 | -6,26 | 4,73 | 2,58 | -0,75 |  |  |
| 1,75 | -4,47 | -1,53 | 7,31 | 1,83 |  |  |  |
| 2 | -6 | 5,78 | 9,14 |  |  |  |  |
| 2,25 | -0,22 | 14,92 |  |  |  |  |  |
| 2,5 | 14,7 |  |  |  |  |  |  |

Ahora, construyo los polinomios con x=1.6

Pn(x) = f [x0] + s∆f [x0] + s(s − 1) 2! ∆ 2 f [x0] + · · · + s(s − 1). . .(s − (n − 1) n! ∆ n f [x0]

s = (x − x0)/ h = (1, 6 − 1, 5)/ 0,25 = 0,4

1° = f [x0] + s∆f [x0]

1° = 1,79 + 0,4 \* -6,26

1°= -0,71

s = (x − x0)/ h = (1, 6 − 1, 25)/ 0,25 = 1,4

2° = f [x0] + s∆f [x0] + s(s − 1) 2! ∆ 2 f [x0]

2° = 2,48 + 1,4 \* -0,69+ [1.4(1.4-1)/2] \* -5,57

2° =−0, 045

3° = P2 + [S(S-1)(S-2)]/6 \*10,3

3° =-0,62

Tabla

Descripción generada automáticamente con confianza media

b)

Tabla

Descripción generada automáticamente

c)

Tabla

Descripción generada automáticamente

Imagen que contiene Texto

Descripción generada automáticamente

clear

x = [1.00 1.25 1.50 1.75 2.00 2.25 2.50];

y = [-4.50 2.48 1.79 -4.47 -6.00 -0.22 14.70];

% (a) Graficar los puntos

plot(x, y, 'o') % Grafica solamente los puntos

grid on % Grafica una grilla

hold on % Mantiene los gráficos anteriores

% (b) Polinomio de interpolación de orden 3

x\_values = [1:0.1:2.50]; % Define el vector x entre 1 y 2.50 con intervalos de 0.1

% Coeficientes del polinomio de tercer orden

a3 = ((y(4)-y(3))/(x(4)-x(3)) - (y(3)-y(2))/(x(3)-x(2))) / (x(4)-x(2));

a2 = ((y(3)-y(2))/(x(3)-x(2)) - (y(2)-y(1))/(x(2)-x(1))) / (x(3)-x(1));

a1 = (y(2)-y(1))/(x(2)-x(1)) - a2\*(x(1)+x(2)) - a3\*x(1)^2;

a0 = y(1) - a1\*x(1) - a2\*x(1)^2 - a3\*x(1)^3;

P3 = a0\*x\_values.^3 + a1\*x\_values.^2 + a2\*x\_values + a3; % Polinomio de tercer orden

plot(x\_values, P3, 'r') % Grafica el polinomio en línea roja

% (c) Coeficientes del polinomio utilizando polyfit

coef = polyfit(x, y, 3); % Calcula los coeficientes del polinomio de interpolación de orden 3

% (d) Evaluar el polinomio hallado y graficarlo

P3\_polyfit = polyval(coef, x\_values); % Evalúa el polinomio con los coeficientes hallados

plot(x\_values, P3\_polyfit, 'g') % Grafica el polinomio en línea verde

legend('Puntos', 'Interpolación manual', 'Interpolación con polyfit') % Añade leyenda

xlabel('x') % Etiqueta del eje x

ylabel('y') % Etiqueta del eje y

Tabla

Descripción generada automáticamentetitle('Interpolación de polinomios') % Título del gráfico

Lagrange

Diagrama, Texto

Descripción generada automáticamente

a)

Lo = ---------- Lo = -------------- Lo =

P(1,8) = Lo(1,8)\*4 +L1(1,8)\*2 +L2(1,8)\*0,6666

P(1,8) = 0,1867

Si lo comparo con Newton grado 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | x | y | ∆f [xi] | ∆f [xi] 2 |
| 0 | 0,50 | 4,00 | -4,00 | 1,33 |
| 1 | 1,00 | 2,00 | -0,67 |  |
| 2 | 3,00 | 0,67 |  |  |

Hacia adelante

𝑃𝑛(𝑥) = 𝑎0 + ∆f 1 (𝑥 − 𝑥0) + ∆f 2 (𝑥 − 𝑥0) (𝑥 − 𝑥1)

𝑃𝑛(𝑥) = 4 -4(x-0,5) + 1,33(x-0,5)(x-1)

Hacia atrás

𝑃𝑛(𝑥) = 𝑎2 + ∆f 1 (𝑥 – 𝑥2) + ∆f 2 (𝑥 – 𝑥2) (𝑥 − 𝑥1)

𝑃𝑛(𝑥) = 0,67 -0,67(x-3) +1,33(x-3)(x-1)

𝑆𝑖 𝑒𝑣𝑎𝑙𝑢𝑎𝑚𝑜𝑠 𝑎𝑚𝑏𝑜𝑠 𝑟𝑒𝑠𝑢𝑙𝑡𝑎𝑑𝑜𝑠 𝑒𝑛 𝑥 = 1.8 𝑠𝑒 𝑜𝑏𝑡𝑖𝑒𝑛𝑒 𝑃𝑛(1.8) = 𝑃𝑛 1.8 = 0.1867

Tabla

Descripción generada automáticamente

clearvars

|  |  |  |
| --- | --- | --- |
| i | X | Y |
| 0 | 0,5 | 0,3513 |
| 1 | 1 | 1,2817 |
| 2 | 1,3 | 1,5307 |
| 3 | 1,5 | 1,5183 |
| 4 | 1,9 | 0,9141 |

% Datos dados

X = [0.5000, 1.0000, 1.3000, 1.5000, 1.9000];

Y = [0.3513, 1.2817, 1.5307, 1.5183, 0.9141];

x\_interp = 1.1;

% Calcular el polinomio de Lagrange

n = length(X);

L = ones(1, n);

for i = 1:n

for j = 1:n

if j ~= i

L(i) = L(i) \* (x\_interp - X(j)) / (X(i) - X(j));

end

end

end

% Calcular el valor interpolado

y\_interp = sum(Y .\* L);

% Calcular el error absoluto

f = @(x) 4\*x - exp(x);

actual\_value = f(x\_interp);

error = abs(actual\_value - y\_interp);

disp(['El valor interpolado en x = ', num2str(x\_interp), ' es ', num2str(y\_interp)]);

disp(['El error absoluto de la interpolación es ', num2str(error)]);

% Grafica los puntos y el polinomio interpolante

fplot(f, [0.4, 2], 'DisplayName', 'f(x) = 4x - e^x');

hold on;

scatter(X, Y, 'r', 'filled', 'DisplayName', 'Datos dados');

% Evaluar el polinomio interpolante en un rango de valores de x para graficarlo

x\_range = linspace(min(X), max(X), 100);

L\_values = zeros(size(x\_range));

for i = 1:length(x\_range)

L\_i = ones(1, n);

for j = 1:n

for k = 1:n

if k ~= j

L\_i(j) = L\_i(j) \* (x\_range(i) - X(k)) / (X(j) - X(k));

end

end

end

L\_values(i) = sum(Y .\* L\_i);

end

plot(x\_range, L\_values, 'g', 'DisplayName', 'Polinomio interpolante');

legend;

xlabel('x');

ylabel('y');

title('Interpolación de puntos dados y comparación con f(x)');

grid on;

hold off;

F(X)= 4X -e\*(x)

Error: | F(1,1) – Lagrange(1,1) | = 1,3958 – 1,4 = 4x10-3

Error relativo : (Error\*100)/F(1,1) =0,3%

Tabla

Descripción generada automáticamente

% Datos dados

X = [0.5000, 1.0000, 1.3000, 1.5000, 1.9000];

Y = [3.5636 1.3424 -0.4972 -1.8872 -4.9594];

x\_interp = 1.75;

(Se borro en copiado, por lo tanto quedo incompleto)

Interpolación

Para un conjunto de 𝑛 + 1 puntos 𝑥𝑖 , 𝑦𝑖 𝑖=0 𝑛 , se puede obtener el polinomio único de grado menor o igual a 𝑛 usando:

▪ Método de Lagrange: Conveniente para interpolar varias veces con los mismos valores de abscisas. ▪ Método de Newton: Conveniente para interpolar varias veces agregando o sacando puntos del conjunto.

▪ Con diferencias divididas (hacia atrás o hacia adelante): para cualquier conjunto de puntos.

▪ Con diferencias finitas (hacia atrás o hacia adelante): solo para puntos equiespaciados. 𝑥𝑖 𝑥0 𝑥1 ⋯  
▪Para un conjunto de 𝑛 + 1 puntos 𝑥𝑖 , 𝑦𝑖 , 𝑦𝑖 ′ 𝑖=0 𝑛 , se puede obtener el polinomio único de grado menor o igual a 2𝑛 + 1 usando el método de Hermite. (Con derivadas)

Tabla

Descripción generada automáticamente

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| X | 𝑓[𝑥𝑖 ] | 𝑓[𝑥𝑖 , 𝑥𝑖+1] | 𝑓[𝑥𝑖 , 𝑥𝑖+1, 𝑥𝑖+2] | 𝑓[𝑥𝑖 , 𝑥𝑖+1, 𝑥𝑖+2, 𝑥𝑖+3] | |  | | --- | |  | |  |
| X0=1 | 3 | 0,2 | -1,7 | 2,2 |  | Tabla  Descripción generada automáticamente |
| X0'=1 | 3 | -1,5 | 0,5 |  |  |  |
| X1=2 | 1,5 | -1 |  |  |  |  |
| X1'=2 | 1,5 |  |  |  | i= | 0 1 |

Reemplazando los coeficientes en la fórmula de cociente de newton, queda

F(x) = 3 + 0,2(x-1) -1,7\*(x-1)^2 +2,2\*(x-1)^2 .(x-2)

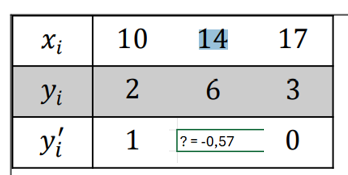
Texto

Descripción generada automáticamente

Diagrama

Descripción generada automáticamente con confianza baja

A partir del grafico, se obtienen los siguientes valores



Que me generan la siguiente tabla

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| X | 𝑓[𝑥𝑖 ] | 𝑓[𝑥𝑖 , 𝑥𝑖+1] | 𝑓[𝑥𝑖 , 𝑥𝑖+1, 𝑥𝑖+2] | 𝑓[𝑥𝑖 , 𝑥𝑖+1, 𝑥𝑖+2, 𝑥𝑖+3] | 𝑓[𝑥𝑖 , 𝑥𝑖+1, 𝑥𝑖+2, 𝑥𝑖+3] |  |
| X0=10 | 2,00 | 1,00 | 0,00 | -0,10 | 0,02 | 0,00 |
| X0'=10 | 2,00 | 1,00 | -0,39 | 0,04 | 0,02 |  |
| X1=14 | 6,00 | -0,57 | -0,11 | 0,15 |  |  |
| X1'=14 | 6,00 | -1,00 | 0,33 |  |  |  |
| X2=17 | 3,00 | 0,00 |  |  |  |  |
| X2'=17 | 3,00 |  |  |  |  |  |

Por lo tanto, el polinomio de Hermite es

P(x) = 2 + (x-10) + 0 – 0,1\*(x-10)^2(x-14) +0,020,1\*(x-10)^2(x-14)^2

Texto

Descripción generada automáticamente

Gráfico, Gráfico de líneas

Descripción generada automáticamente

S e aproxima mejor el de octavo grado

% Definir la función f(x)

f = @(x) 2./(1 + x.^2);

% Definir los puntos de interpolación para cada parte del ejercicio

x\_a = [-5, 0, 5]; % Para el polinomio de segundo grado

x\_b = [-5, -2.5, 0, 2.5, 5]; % Para el polinomio de cuarto grado

x\_c = [-6.5 -6 x\_b 6 6.5]; % Para el polinomio de octavo grado

% Definir el rango de valores de x sobre el cual evaluar la función interpolada

x\_range = linspace(-6.5, 6.5, 1000); % Ajustar el rango para enfocar la región de interés

% Calcular los polinomios de interpolación

p1 = lagrange\_interp(x\_a, f, x\_range);

p2 = lagrange\_interp(x\_b, f, x\_range);

p3 = lagrange\_interp(x\_c, f, x\_range);

% Graficar la función original y los polinomios de interpolación

plot(x\_range, f(x\_range), 'k-', 'LineWidth', 2); hold on;

plot(x\_range, p1, 'r--', 'LineWidth', 1.5); % Cambiar el color de la línea

plot(x\_range, p2, 'g--', 'LineWidth', 1.5); % Cambiar el color de la línea

plot(x\_range, p3, 'b--', 'LineWidth', 1.5); % Cambiar el color de la línea

% Graficar los puntos de interpolación

fxc = f(x\_c); % Calcular los valores de la función en los puntos de interpolación

plot(x\_c, fxc, 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');

grid on;

legend('f(x)', 'Polinomio de segundo grado', 'Polinomio de cuarto grado', 'Polinomio de octavo grado', 'Puntos de interpolación');

xlabel('x');

ylabel('f(x)');

title('Aproximación de f(x) por polinomios de Lagrange');

hold off;

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente9 skip

% Definir la función original f(x)

f = @(x) 2 ./ (1 + x.^2);

% Datos de interpolación

x1 = [-5, 0, 5];

y1 = f(x1);

x2 = [-5, -2.5, 0, 2.5, 5];

y2 = f(x2);

x3 = [-5, -2.5, 0, 2.5, 5];

y3 = f(x3);

% Interpolación en x=4 mediante spline cúbico

valor1 = interp1(x1, y1, 4, 'spline');

valor2 = interp1(x2, y2, 4, 'spline');

valor3 = interp1(x3, y3, 4, 'spline');

% Calcular polinomios de interpolación con spline cúbico

coef1 = spline(x1, [0, y1, 0]);

coef2 = spline(x2, [0, y2, 0]);

coef3 = spline(x3, [0, y3, 0]);

P1 = ppval(coef1, x1); % Evaluar el polinomio en los puntos x1

P2 = ppval(coef2, x2); % Evaluar el polinomio en los puntos x2

P3 = ppval(coef3, x3); % Evaluar el polinomio en los puntos x3

% Graficar los polinomios de interpolación con spline cúbico y la función original

x\_values = linspace(-5, 5, 1000);

y\_values = f(x\_values);

figure;

plot(x\_values, y\_values, 'k-', 'LineWidth', 2); % Función original

hold on

plot(x1, y1, 'bo', x\_values, ppval(coef1, x\_values), 'g-'); % Puntos y polinomio 1

plot(x2, y2, 'ro', x\_values, ppval(coef2, x\_values), 'r-'); % Puntos y polinomio 2

plot(x3, y3, 'mo', x\_values, ppval(coef3, x\_values), 'b-'); % Puntos y polinomio 3

legend('f(x)', 'Puntos y Polinomio 1', 'Puntos y Polinomio 2', 'Puntos y Polinomio 3');

xlabel('x');

ylabel('f(x)');

title('Polinomios de Interpolación con Spline Cúbico y Función Original');

grid on;

Gráfico

Descripción generada automáticamente

Codigo original

clear

x = [ -5.000 0.000 5.000]

y= [ 0.077 2.000 0.077]

valor=interp1(x,y,4,’spline’) %interpola x=4 mediante spline cúbico

Graficar los polinomios de interpolación con spline cúbico.

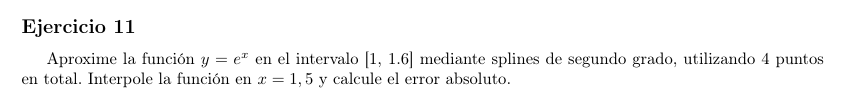
x1 = [-5.00:0.1:5.00]

P3=interp1(x,y,x1,’spline’) %interpola con spline para los puntos x1

plot(x,y,x1,P3,’-o’) %grafica superpuestos puntos y polinomio

P6= polyval(coef,x) %evalúa el polinomio con los coeficientes hallados

plot(x,P6,’g’) %grafica el polinomio en línea verde



Gráfico, Gráfico de líneas

Descripción generada automáticamente

clear; % Limpiar variables del entorno de trabajo

x = [1, 1.2, 1.4, 1.6]; % Definir puntos de interpolación correctamente

y = exp(x); % Calcular los valores de la función en los puntos de interpolación

x\_interp = 1.5; % Punto de interpolación

% Interpolar en x = 1.5 mediante spline cúbico

valor\_interp = interp1(x, y, x\_interp, 'spline');

% Graficar los polinomios de interpolación con spline cúbico

x\_graf = linspace(1, 1.6, 100); % Definir puntos para graficar el polinomio

P3 = interp1(x, y, x\_graf, 'spline'); % Interpolar con spline cúbico para los puntos x\_graf

% Graficar puntos de interpolación y polinomio de interpolación

plot(x\_graf, P3, '-'); % Graficar polinomio de interpolación

hold on;

plot(x, y, 'o'); % Graficar puntos de interpolación

plot(x\_interp, valor\_interp, 'xr', 'MarkerSize', 10); % Marcar punto interpolado

hold off;

xlabel('x');

ylabel('f(x)');

title('Interpolación de la función e^x con Spline Cúbico');

legend('Polinomio de Interpolación', 'Puntos de Interpolación', 'Punto Interpolado');

grid on;

Calculos Manuales

N=3 ---) 3\*3 = 9 Incognitas

3 FUNCIONES PRINCIPALES DE LA FORMA

Aix2+bix+ci

Tabla

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X | 1 | 1,2 | 1,4 | 1,6 |
| Y | 2,71 | 3,32 | 4,05 | 4,9 |

Inicio

Planteo condiciones de borde

Po(x) = a0x2 + b0x + c0 = 2,71 ----) Po(x) = a0+ b0 + c0 = 2,71

P2(x) = a2x2 + b2x + c2 = 4,9 ----) P2(x) = 2,56a2+ 1,6b2+ c2 = 4,9

Planteo condiciones intermedias

Po(x=1,2) = a0x2 + b0x + c0 = 3,32 ----) Po(x) = 1,44a0+ 1,2b0 + c0 = 3,32

P3(x=1,2) = a1x2 + b1x + c1 = 3,32 ----) P2(x) = 1,44a1+ 1,2b1 + c1 = 3,32

Po(x=1,4) = a1x2 + b1x + c1 = 4,05 ----) Po(x) = 1,96a1+1,4 b1 + c1 = 4,05

P3(x=1,4) = a2x2 + b2x + c2 = 4,05 ----) P2(x) = 1,96a2+ 1,4b2 + c2 = 4,05

Planteo condiciones derivadas

2ao\*(1,2) + bo = 2a1\*(1,2) +b1 (x=1,2)

2a1\*(1,4) + b1 = 2a2\*(1,4) +b2 (x=1,4)

Falta 1 condicion, que elijo por convenciencia a0 = 0

Planteando todo en una tabla, me queda

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| a0 | a1 | a2 | bo | b1 | b2 | c0 | c1 | c2 | res |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 2,71 |
| 0 | 0 | 2,56 | 0 | 0 | 1,6 | 0 | 0 | 1 | 4,9 |
| 1,44 | 0 | 0 | 1,2 | 0 | 0 | 1 | 0 | 0 | 3,32 |
| 0 | 1,44 | 0 | 0 | 1,2 | 0 | 0 | 1 | 0 | 3,32 |
| 0 | 1,96 | 0 | 0 | 1,4 | 0 | 0 | 1 | 0 | 4,05 |
| 0 | 0 | 1,96 | 0 | 0 | 1,4 | 0 | 0 | 1 | 4,05 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2,4 | -2,4 | 0 | 1 | -1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 2,8 | -2,8 | 0 | 1 | -1 | 0 | 0 | 0 | 0 |

Pantalla de un computador

Descripción generada automáticamente con confianza baja

Resolviendo, por software me da que

P(x) = 3,34x -0,66 [1—x—1,2]

P(x) = 3,34x - 0,73 [1,2—x—1,4]

P(x) = 0,7x2 + 1,37x +0,88 [1,4—x—1,6]

El error absoluto es F(1,6)-P(1,6 ) = 4,48 – 4,51 = 0,03

El error relativo es 0,03/4,48 = 0,006

Ajuste

Tabla

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente(Consultar)

Texto

Descripción generada automáticamente

Gráfico

Descripción generada automáticamente

Modificaciones

x = [0, 1, 2.2, 3.2, 4.2, 5.4, 6.5, 7.7, 8.9] \* 1e-4; % Multiplicar por 1e-4 para convertir a unidades correctas

y = [248.75, 497.51, 746.25, 995, 1243.75, 1492.5, 1741.25, 1990, 2238.75];

coef = polyfit(x, y, 1); % Calcula los coeficientes de un polinomio de grado 1

P1 = polyval(coef, x); % Evalúa el polinomio con los coeficientes hallados

plot(x, y, 'o', x, P1, 'g'); % Grafica los datos originales y la línea ajustada en verde

xlabel('X');

ylabel('Y');

title('Ajuste lineal');

legend('Datos originales', 'Ajuste lineal');

Tabla

Descripción generada automáticamente

Gráfico, Gráfico de líneas

Descripción generada automáticamente

% Datos

X = [0.1 0.2 0.3 0.4 0.5];

Y = [2 2.9 5 6.7 12];

% Ajuste lineal

A = [X' ones(length(X), 1)];

coef\_lineal = A\Y';

fprintf('Coeficientes del ajuste lineal: a = %f, b = %f\n', coef\_lineal(1), coef\_lineal(2));

% Gráfico del ajuste lineal

x\_vals = linspace(min(X), max(X), 100);

y\_vals = coef\_lineal(1) \* x\_vals + coef\_lineal(2);

figure;

subplot(2, 2, 1);

plot(X, Y, 'o', x\_vals, y\_vals);

xlabel('X');

ylabel('Y');

title('Ajuste lineal');

% Ajuste polinómico de 2do grado

A = [X'.^2 X' ones(length(X), 1)];

coef\_polinomial = A\Y';

fprintf('Coeficientes del ajuste polinomial de 2do grado: a = %f, b = %f, c = %f\n', coef\_polinomial(1), coef\_polinomial(2), coef\_polinomial(3));

% Gráfico del ajuste polinomial de 2do grado

y\_vals = coef\_polinomial(1) \* x\_vals.^2 + coef\_polinomial(2) \* x\_vals + coef\_polinomial(3);

subplot(2, 2, 2);

plot(X, Y, 'o', x\_vals, y\_vals);

xlabel('X');

ylabel('Y');

title('Ajuste polinomial de 2do grado');

% Transformación para ajuste potencial: ln(Y) = ln(a) + b \* ln(X)

ln\_X = log(X);

ln\_Y = log(Y);

% Ajuste lineal de ln(Y) vs ln(X)

A = [ln\_X' ones(length(ln\_X), 1)];

coef\_potencial = A\ln\_Y';

a = exp(coef\_potencial(2));

b = coef\_potencial(1);

fprintf('Coeficientes del ajuste potencial: a = %f, b = %f\n', a, b);

% Gráfico del ajuste potencial

y\_vals = a \* x\_vals.^b;

subplot(2, 2, 3);

plot(X, Y, 'o', x\_vals, exp(y\_vals));

xlabel('X');

ylabel('Y');

title('Ajuste potencial');

% Transformación para ajuste exponencial: ln(Y) = ln(a) + b \* X

% Ajuste lineal de ln(Y) vs X

A = [X' ones(length(X), 1)];

coef\_exponencial = A\ln\_Y';

a\_exp = exp(coef\_exponencial(2));

b\_exp = coef\_exponencial(1);

fprintf('Coeficientes del ajuste exponencial: a = %f, b = %f\n', a\_exp, b\_exp);

% Gráfico del ajuste exponencial

y\_vals = a\_exp \* exp(b\_exp \* x\_vals);

subplot(2, 2, 4);

plot(X, Y, 'o', x\_vals, y\_vals);

xlabel('X');

ylabel('Y');

title('Ajuste exponencial');

Tabla

Descripción generada automáticamente

Gráfico, Gráfico de líneas

Descripción generada automáticamente

% Datos

delta\_L = [0 3 6 9 12 15 20 20 25 30 35 40 45] \* 1e-2; % Convertir a metros

P = [500 1900 3250 4300 5450 6600 8100 9000 9350 9500 9700 9850 10000];

% Graficar los puntos

figure;

plot(delta\_L, P, 'o');

xlabel('Alargamiento (\DeltaL) [m]');

ylabel('Carga (P) [kg]');

title('Relación carga-alargamiento');

% Ajuste de curva

% Aquí puedes seleccionar un ajuste adecuado para tus datos, por ejemplo, una curva cuadrática:

% Ajuste polinómico de 2do grado

A = [delta\_L'.^2 delta\_L' ones(length(delta\_L), 1)];

coef\_polinomial = A\P';

fprintf('Coeficientes del ajuste polinomial de 2do grado: a = %f, b = %f, c = %f\n', coef\_polinomial(1), coef\_polinomial(2), coef\_polinomial(3));

% Graficar la curva ajustada

x\_vals = linspace(min(delta\_L), max(delta\_L), 100);

y\_vals = coef\_polinomial(1) \* x\_vals.^2 + coef\_polinomial(2) \* x\_vals + coef\_polinomial(3);

hold on;

plot(x\_vals, y\_vals, 'r');

legend('Datos', 'Curva ajustada');

hold off;

% Predicción del alargamiento para una carga de 7 toneladas

carga\_deseada = 7 \* 1000; % Convertir toneladas a kg

deltaL\_pred = polyval(coef\_polinomial, carga\_deseada);

fprintf('El alargamiento correspondiente a una carga de 7 toneladas es: %f m\n', deltaL\_pred);

Tabla

Descripción generada automáticamente

% Datos

R = [10.5 5.25 3.5 2.63 2.1 1.75 1.5 1.31 1.17 1.05];

I = [500 1900 3250 4300 5450 6600 8100 9000 9350 9500];

% Graficar los puntos

figure;

plot(R, I, 'o');

xlabel('Recurrencia (R) [años]');

ylabel('Intensidad de lluvia (I) [mm/h]');

title('Intensidad de lluvia en función de la recurrencia');

% Ajuste de curva

% Aquí puedes seleccionar un ajuste adecuado para tus datos, por ejemplo, una curva exponencial:

% Transformación para ajuste exponencial: ln(I) = ln(a) + b \* R

ln\_I = log(I);

A = [R' ones(length(R), 1)];

coef\_exponencial = A\ln\_I';

a\_exp = exp(coef\_exponencial(2));

b\_exp = coef\_exponencial(1);

fprintf('Coeficientes del ajuste exponencial: a = %f, b = %f\n', a\_exp, b\_exp);

% Graficar la curva ajustada en coordenadas semi-logarítmicas

x\_vals = linspace(min(R), max(R), 100);

y\_vals = a\_exp \* exp(b\_exp \* x\_vals);

figure;

semilogy(R, I, 'o', x\_vals, y\_vals);

xlabel('Recurrencia (R) [años]');

ylabel('Intensidad de lluvia (I) [mm/h]');

title('Intensidad de lluvia en función de la recurrencia (escala semi-logarítmica)');

% Predicción de la intensidad para una recurrencia de 3 años

recurrencia\_deseada = 3;

intensidad\_pred = a\_exp \* exp(b\_exp \* recurrencia\_deseada);

fprintf('La intensidad esperada de la lluvia para una recurrencia de 3 años es: %f mm/h\n', intensidad\_pred);

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

Tabla

Descripción generada automáticamente

Gráfico

Descripción generada automáticamente

% Datos

X = [1, 1.8, 2.3, 4.1];

Y = [0.5, 0.9, 1.1, 1.4];

% Transformación de datos

X\_log = log(X);

Y\_log = log(Y);

% Ajuste de polinomio

degree = 2; % Grado del polinomio

coefficients = polyfit(X\_log, Y\_log, degree);

% Parámetros del ajuste

a = exp(coefficients(1));

b = exp(coefficients(2));

disp(['Parámetros ajustados: a = ', num2str(a), ', b = ', num2str(b)]);

% Evaluación del polinomio ajustado

x\_vals = linspace(min(X\_log), max(X\_log), 100);

y\_vals\_log = polyval(coefficients, x\_vals);

y\_vals = exp(y\_vals\_log);

% Gráfico de los datos y la curva ajustada

figure;

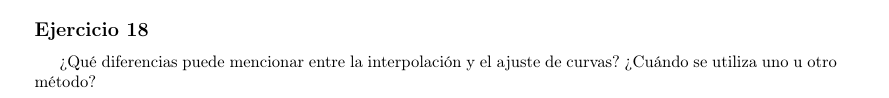
plot(X, Y, 'o', exp(x\_vals), y\_vals);

xlabel('X');

ylabel('Y');

title('Ajuste de datos a curva Y = a\*x / (b + x)');

legend('Datos', 'Curva ajustada');



La interpolación y el ajuste de curvas son dos técnicas utilizadas en el análisis de datos, pero difieren en sus enfoques y propósitos principales:

1. **Interpolación**:
   * La interpolación se refiere al proceso de estimar valores dentro del rango de datos conocidos o experimentales.
   * El objetivo principal de la interpolación es encontrar una función que pase exactamente a través de todos los puntos de datos dados.
   * La interpolación generalmente se utiliza cuando se tienen datos discretos y se necesita encontrar valores intermedios dentro de ese rango con precisión.
2. **Ajuste de curvas**:
   * El ajuste de curvas, por otro lado, implica encontrar una función matemática que se ajuste mejor a un conjunto de datos, pero no necesariamente pasa a través de todos los puntos de datos.
   * El ajuste de curvas busca modelar la tendencia general de los datos y puede ser útil para hacer predicciones o extrapolar más allá del rango de datos conocidos.
   * Se utiliza cuando los datos tienen cierto nivel de error o variabilidad, y el objetivo es encontrar una función que represente adecuadamente la relación subyacente entre las variables.

En resumen, la principal diferencia entre la interpolación y el ajuste de curvas radica en la precisión deseada y la flexibilidad en la representación de los datos. La interpolación se emplea cuando se necesita precisión absoluta dentro del rango de datos conocidos, mientras que el ajuste de curvas se utiliza cuando se busca modelar la tendencia general de los datos, incluso si esto implica no pasar exactamente por cada punto de datos.

Tabla

Descripción generada automáticamente

Tabla

Descripción generada automáticamente

Texto, Carta

Descripción generada automáticamente

(a) Falso. En el método de interpolación de Newton, el orden del polinomio se incrementa por cada punto adicional dentro del intervalo de interpolación, ya que cada nuevo punto introduce un término adicional al polinomio. En los polinomios de interpolación de Lagrange, el grado del polinomio también se incrementa por cada punto adicional agregado al intervalo de interpolación.

(b) Verdadero. El método de interpolación de Newton tiene la ventaja de no requerir el cálculo completo del polinomio al agregar un nuevo punto al inicio o al final del intervalo, a diferencia de los polinomios de Lagrange.

(c) Verdadero. Agregar nuevos puntos en los polinomios de interpolación de Newton o Lagrange puede provocar serpenteo. Los polinomios de interpolación de Hermite pueden ofrecer una mejor aproximación ya que también consideran las derivadas de la función, provocando que la pendiente sea la solicitada y reduciendo el error.

(d) Verdadero. La interpolación segmentaria mediante splines cuadráticos y cúbicos resuelve el problema de "serpenteo" que pueden tener los polinomios de grado elevado, especialmente cerca de los bordes del intervalo, aunque requiere resolver un sistema de ecuaciones.

(e) Verdadero. Para hallar un polinomio de interpolación de grado 𝑛*n* en base a 𝑛+1*n*+1 puntos, se puede plantear un sistema de ecuaciones, pero este puede volverse mal condicionado a medida que 𝑛*n* aumenta, especialmente si los puntos están cercanos o mal condicionados.