1 AJUSTE DE CURVAS

Básicamente el ajuste de curvas se utiliza cuando se tiene una serie de datos calculados y se desea conocer valores intermedios no conocidos, o también en aquellos casos en que se desee una versión simplificada de una función que se ajuste a un número de valores concretos, y posteriormente usar la función simplificada para derivar nuevos valores.

"Ajustar una curva implica ajustar una función g(x) a un conjunto de datos (xi,yi), *i=1,2,...,L.* g(x) puede ser un polinomio, una función lineal o combinación de funciones conocidas"[1]

Hay básicamente dos métodos para lograr ajuste de curvas:

1. Si los datos no son muy exactos o tienen asociado un error (ruido) entonces la mejor manera es establecer una sola curva que represente la tendencia general de los datos observados. Se conoce como **REGRESIÓN LINEAL**, cuyo método más sencillo es la **REGRESIÓN POR MÍNIMOS CUADRADOS**.

2. Si los datos que se tienen son precisos se trazan una o varias curvas que pasan por cada uno de los puntos de datos. A esto se le llama **INTERPOLACIÓN**, la cual puede ser **lineal** o **curvilínea**.

El ajuste de curvas en ingeniería tiene como aplicación principal, a partir de una serie de datos experimentales, realizar:

- ✓ ANÁLISIS DE TENDENCIA: realizar predicciones de la variable dependiente ya sea para buscar valores fuera del límite observado (EXTRAPOLACIÓN) o dentro del rango de datos observado (INTERPOLACIÓN).

- ✓ PRUEBA DE HIPÓTESIS: cuando se tiene un modelo matemático que se puede usar para comparar los datos que produce con los datos medidos experimentalmente. Si desconocemos los coeficientes del modelo se calculan a partir de los datos y luego se prueba que tan adecuado es al evaluar los resultados que produce.

1.1 REGRESIÓN POR MÍNIMOS CUADRADOS

En este método se pretende trazar la recta que más se acerque al conjunto de datos dado, a la cual se le llama "línea (recta) de regresión", expresada matemáticamente como:

$$\boxed{Y = C_1x + C_2 + Error \qquad (2.1)}$$

[1] Análisis numérico y Visualización gráfica con MathLab. Soichiro Nakamura. Pág. 284.

Los valores de C_1, C_2 y el *Error,* se pueden calcular de las siguientes maneras:

1.1.1 Usando las fórmulas matemáticas.

$$C_1 = \frac{n\sum XiYi - \sum Xi \sum Yi}{n\sum Xi^2 - \left(\sum Xi\right)^2}, C_2 = \frac{\sum Yi}{n} - C_1 \frac{\sum Xi}{n} \text{ y } Error = \sum (Yi - C_2 - C_1 Xi)^2 \qquad \textbf{(2.2)}$$

El error se da en términos de la suma de los cuadrados de la diferencia entre el valor muestral y el valor calculado con la recta de regresión.

Ejemplo: encontrar la línea de regresión que se ajusta mejor a los siguientes datos.

x	1.0	1.5	2.0	2.5	3.0
y	2.0	3.2	4.1	4.9	5.9

La siguiente tabla muestra los valores calculados necesarios para hallar los valores de los coeficientes de la recta buscada:

n	***Xi***	***Yi***	***Xi*Yi***	***Xi²***	***Error***
1	1,00	2,00	2,00	1,00	0,0144
2	1,50	3,20	4,80	2,25	0,0169
3	2,00	4,10	8,20	4,00	0,0064
4	2,50	4,90	12,25	6,25	0,0049
5	3,00	5,90	17,70	9,00	0,0004
TOTAL	10,00	20,10	44,95	22,50	0,043

Reemplazando en las ecuaciones **(2.2)** tenemos:

$$C_1 = \frac{5*44.95 - 10.00*20.10}{5*22.50 - 10^2} = 1.90, \quad C_2 = \frac{20.10}{5} - 1.90\frac{10.0}{5} = 0.22, \quad Error = 0.043$$

Con estos valores al reemplazarlos en la ecuación **(2.1)** tenemos: ***Y = 1.90X + 0.22***.

1.1.2 Usando MatLab.

En MatLab, podemos calcular la recta de regresión usando el comando ***polyfit***, cuya sintaxis es: **C = polyfit(x,y,1)**, donde *x, y* son vectores que contienen los datos puntuales que se desean ajustar.

Una vez ejecutado el comando ***C*** contendrá los coeficientes C_1 y C_2 buscados, siendo C_1= C(1) y C_2 = C(2). Las siguientes instrucciones tecleadas desde la línea de comandos de Matlab efectúan el proceso:

```
x= [1 1.5 2 2.5 3];
```

```
y=[2 3.2 4.1 4.9 5.9];
C=polyfit(x,y,1)
```

Que dan como resultado: **C = [1.900000 0.22000].**

Por lo tanto los datos se ajustan a la recta: ***y(x) = 1.90X + 0.22***. Esta ecuación se puede utilizar luego para estimar nuevos valores de ***y*** a partir de un valor da do de ***x***.

El error lo calculamos como la suma del cuadrado de la diferencia entre los datos muestrales y el valor calculado con la recta de regresión, así:

*Error = (2-1.9*1-0.22)²+(3.2-1.9*1.5-0.22)²+(4.1-1.9*-0.22)²+(4.9–1.9*2.5-0.22)² + (5.9-1.90*3-0.22)² =* ***0.043***

Este cálculo se puede realizar en Matlab con el comando: **sum((y-polyval(C,x)).^2)** .

La solución también se puede hallar usando matrices, planteando el sistema de ecuaciones sobredeterminado:

$$AC = y \qquad (2.3)$$

Siendo ***A*** la matriz de ***nx2***, formada por: la primera columna son los valores de la variable independiente, la segunda columna el valor **1**; ***C*** es el vector columna con los coeficientes de la recta de regresión, e ***y*** es un vector columna formado con los valores de la variable dependiente, como se muestra a continuación:

$$A=\begin{bmatrix} x1 & 1 \\ x2 & 1 \\ \vdots & \vdots \\ xn & 1 \end{bmatrix}, \quad y=\begin{bmatrix} y1 \\ y2 \\ \vdots \\ yn \end{bmatrix}, \quad C=\begin{bmatrix} C1 \\ C2 \end{bmatrix}$$

Esto nos da un sistema de ecuaciones sobredeterminado (más ecuaciones que incógnitas) el cual puede resolverse así: aplicando matriz transpuesta en la ecuación (2.3) obtenemos: $A^tAC = A^ty$, de donde $C=A^ty$, *que en MatLab se puede resolver así:*

C = (A'*A)|(A'*y) o su equivalente ***C = A|y.***

Para el ejemplo anterior los comandos a ejecutar sería:[2]

```
A=x';        %Convertimos en columna el vector con la variable independiente
A(:,2)=1;    %Agregamos una segunda columna con el valor 1
y=y';        %Convertimos en columna el vector con la variable dependiente
C=A\y        %Efectuamos la división matricial.
r=sum((y-polyval(C,x)).^2)  %Evalúa el error
```

[2] En este caso los datos se han digitado a manera de vectores fila.

El resultado es: **C = [1.900000 0.22000]** y **r=0.043.**

Las dos primeras instrucciones las podemos reemplazar por: **A=[x', ones(size(x'))];** Los comandos anteriores sirven en el caso de que hayamos digitado los datos como vectores fila.

Podemos digitar los datos en la forma matricial requerida, en ese caso los comandos a digitar serán:

```
A=[1 1; 1.5 1; 2 1; 2.5 1; 3 1];    %Crea la matriz cuya columna 2 es 1.
y=[2; 3.2; 4.1; 4.9; 5.9];
C=A\y
r=sum(y – polyval(C,x)).^2)         %Evalúa el valor del error
```

Lo cual da los mismos resultados obtenidos anteriormente.

1.1.3 Usando la hoja electrónica Excel.

La hoja electrónica trae incorporada la función **ESTIMACION.LINEAL**, con la cual se puede realizar el cálculo de la recta de regresión.

Se procede así: en una hoja de cálculo se crea una tabla con los valores de las variables independiente y dependiente. Se resalta un bloque de al menos dos celdas, en la misma fila, en donde van a quedar los resultados de los coeficientes C_1 y C_2 buscados, luego se digita la fórmula: ***ESTIMACION.LINEAL(Rango_Y;Rango_X).*** **Rango_Y** nos indica el rango de celdas en donde se encuentran los valores de la variable dependiente; **Rango_X** contiene los valores de la variable independiente. Luego se presionan simultáneamente las teclas: ***Shift + Ctrl + Enter***[3]. En la primera celda queda el valor de C_1 y en la segunda el valor de C_2, con los cuales se forma la recta de regresión.[4]

1.1.4 Modelo gráfico de la recta de la regresión lineal.

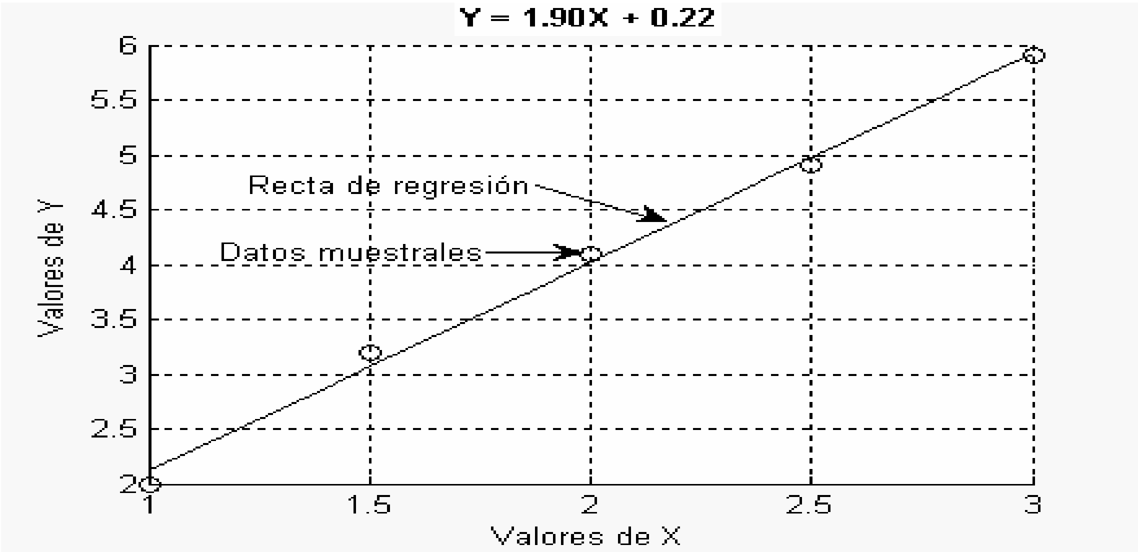
Para trazar la recta de regresión es necesario recalcular unos nuevos puntos de datos con espaciado más fino que los datos originales y luego evaluar la recta de regresión en estos puntos. Los comandos en MatLab que hacen esta tarea son:

```
x1=[x(1):0.1:x(length(x))]  %Se generan puntos de datos en el intervalo.5
y1=polyval(C,x1)            %Se evalúa la recta de regresión en los nuevos puntos.6
```

[3] Esto permite efectuar fórmulas matriciales en la hoja electrónica.
[4] Para más información consultar la ayuda de la hoja electrónica Excel.
[5] La función ***length(x)*** calcula la longitud de un vector. Note que ***x(1)*** es el valor del extremo inferior del intervalo y ***x(length(x))*** da el valor del último elemento del vector.

```
plot(x1,y1)          %Dibuja la recta de regresión
hold on              %Impide borrar la ventana gráfica
plot(x,y,'o')        %Dibuja los datos muestrales
```

Los tres últimos comando se pueden reemplazar por el comando: plot(x,y,'o',x1,y1). Gráficamente los datos se observarían como se muestra en la figura siguiente:



Ejemplo 2: En una muestra se obtuvieron los siguientes resultados:

x	0.1	0.4	0.5	0.7	0.7	0.9
y	0.61	0.92	0.99	1.52	1.47	2.03

Hallar la recta de regresión y estimar el valor para x=1.3?. Los cálculos a realizar se muestran en la siguiente tabla:

n	x	y	*Xi*Yi*	*Xi²*	*Error*
1	0,1	0,61	0,061	0,01	0,021722034
2	0,4	0,92	0,368	0,16	0,00518157
3	0,5	0,99	0,495	0,25	0,031840412
4	0,7	1,52	1,064	0,49	1,82307E-06
5	0,7	1,47	1,029	0,49	0,002636844
6	0,9	2,03	1,827	0,81	0,024254448
Total	3,3	7,54	4,844	2,21	0,085637131

[6] La función ***polyval(C,x1)*** permite evaluar un polinomio en los puntos contenidos en el vector *x*. C contiene los coeficientes de la recta de regresión, previamente calculado con ***polyfit(x,y,1)***.

Evaluar	1,3	***2,58008***
	C1	***C2***
	1,764557	0,28616

Los valores de C1 y C2 se obtuvieron utilizando las ecuaciones (2.3).

Podemos plantear la recta de regresión como ***Y = 1.7645X + 0.2862***, y al evaluarla en **X=1.3** da como resultado: **2.58008**.

De la misma manera usando MatLab el proceso sería:

```
x=[0.1 0.4 0.5 0.7 0.7 0.9];
y=[0.61 0.92 0.99 1.52 1.47 2.03];
C=polyfit(x,y,1);
```

Que dan como resultado: C = [1.76455696202532   0.28616033755274] lo que nos permite construir la recta de regresión: Y=1.76455696202532X+ 0.28616033755274. Al evaluarla en X=1.3 digitamos el comando:

polyval(C,1.3)

Dando como respuesta: **2.58008438818565**, que es el valor buscado.

1.2 REGRESIÓN MÚLTIPLE

Cuando se tienen datos de *n* variables independientes y una dependiente, ya no podemos ajustar los datos por medio de una recta, es necesario usar un plano n-dimensional para hacer dicha aproximación.

Para el caso de dos variables se tendría:

$$Y = C_0 + C_1X_1 + C_2X_2 + Error. \qquad \textbf{(2.4)}$$

El error se calcula restando a los valores de la variable dependiente (y) el valor estimado usando la ecuación Si tenemos una serie de *n* datos puntuales:

X_1=	X_{11}	X_{12}	...	X_{1n}
X_2=	X_{21}	X_{22}	...	X_{2n}
Y =	Y_1	Y_2	...	Y_n

Matricialmente podemos expresar el sistema de ecuaciones como:

$$\underbrace{\begin{bmatrix} n & \sum X_1 i & \sum X_{2i} \\ \sum X_{1i} & \sum X_{1i}^{\;2} & \sum X_{1i}X_{2i} \\ \sum X_{2i} & \sum X_{1i}X_{2i} & \sum X_{2i}^{\;2} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} C_0 \\ C_1 \\ C_2 \end{bmatrix}}_{\mathbf{C}} = \underbrace{\begin{bmatrix} \sum Y_i \\ \sum X_{1i}Y_i \\ \sum X_{21}Y_i \end{bmatrix}}_{\mathbf{y}}$$

Lo cual se resuelve para **C** así: ***C= inv(A)*y*** ó usando MatLab: ***C = A|y***.

Veamos un ejemplo: se tienen los siguientes valores muestrales para las variables independientes X_1 *y* X_2 y para la variable dependiente *Y*.

X_1=	0.0	2.0	2.5	1.0	4.0	7.0
X_2=	0	1	2	3	6	2
Y =	5	10	9	0	3	27

En la siguiente tabla vemos los cálculos necesarios para hallar el sistema de ecuaciones:

n	Y	X_1	X_2	X_1^2	X_2^2	X_1X_2	X_1Y	X_2Y
1	5,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
2	10,0	2,0	1,0	4,0	1,0	2,0	20,0	10,0
3	9,0	2,5	2,0	6,3	4,0	5,0	22,5	18,0
4	0,0	1,0	3,0	1,0	9,0	3,0	0,0	0,0
5	3,0	4,0	6,0	16,0	36,0	24,0	12,0	18,0
6	27,0	7,0	2,0	49,0	4,0	14,0	189,0	54,0
SUMAS	**54,0**	**16,5**	**14,0**	**76,3**	**54,0**	**48,0**	**243,5**	**100,0**

A= $\begin{bmatrix} 6 & 16.5 & 14.0 \\ 16.5 & 76.3 & 48.0 \\ 14.0 & 48.0 & 54.0 \end{bmatrix}$, y= $\begin{bmatrix} 54.0 \\ 243.5 \\ 100.0 \end{bmatrix}$, C= $\begin{bmatrix} C_0 \\ C_1 \\ C_2 \end{bmatrix}$ para hallar *C= A|y* = $\begin{bmatrix} 5.01282 \\ 3.99251 \\ -2.9967 \end{bmatrix}$

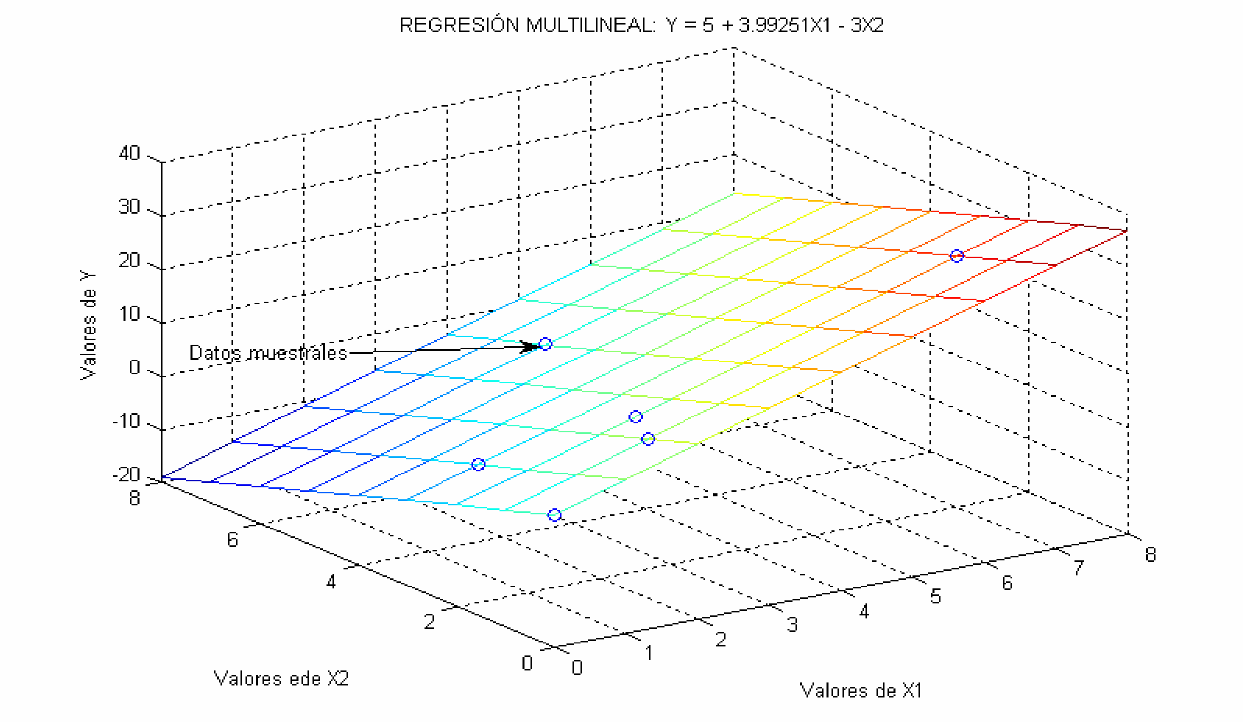
El plano de regresión esta dado por: ***y= 5.01282 + 3.99251X₁ – 2.9967X₂***.

Gráficamente podemos ver el resultado, usando los siguientes comandos de MatLab:

```
x=0:0.1:8;                  % Se generan puntos de datos para el eje x
y=x;                        %Se toman los mismos puntos para el eje y
[X,Y]=meshgrid(x,y);        %Creo el espacio tridimensional para graficar
z=5+3.99251.*X -3.*Y;       %Calculo los valores de Z para trazar el plano
mesh(X,Y,z)                 %Elaboro la imagen del plano en 3D
hold on                     %Evito borrar la gráfica ya trazada
X1=[0 2 2.5 1 4 7];         %Datos originales de X1
X2=[0 1 2 3 6 2];           %Datos originales de X2
```

```
Yi=[5 10 9 0 3 27];         %Datos originales de Y
scatter(X1,X2,Yi);          %Trazo la gráfica de los datos originales
r=max(y - A*C)              %Error con respecto a los datos muestrales
```

Con lo cual obtenemos la siguiente gráfica:



En general para varias variables podemos expresar el sistema como:

$$Y = C_0 + C_1X_1 + C_2X_2 + \ldots + C_nX_{1n} + \text{Error} \qquad \textbf{(2.5)}$$

Conocemos los muestrales:

X1= $[X_{11} \quad X_{12} \quad \ldots \quad X_{1n}]$;
X2= $[X_{21} \quad X_{22} \quad \ldots \quad X_{2n}]$;
...
Xn= $[X_{n1} \quad X_{n2} \quad \ldots \quad X_{nn}]$;
Y = $[Y_1 \quad Y_2 \quad \ldots \quad Y_n]$;

Planteamos matricialmente el sistema de ecuaciones sobredeterminado AC = y, donde:
A = [ones(size(X1')) X1' X2' ... Xn'];[7]

La solución se obtiene con el comando:

C=A\y'

[7] Los puntos suspensivos indican que pueden ir más columnas de datos.

Y el resultado es: C=[5.0; 4.0; -3.0]. Valores muy similares a los obtenidos anteriormente.

1.3 INTERPOLACIÓN

Si vemos el diccionario de la Real Academia de la Lengua española la definición de interpolación es:

> **interpolar**. (Del lat. *interpolāre*, alterar, mezclar, cambiar). tr. Poner algo entre otras cosas. || 2. Intercalar palabras o frases en el texto de un manuscrito antiguo, o en obras y escritos ajenos. || 3. Interrumpir o hacer una breve intermisión en la continuación de algo, y volver luego a proseguirlo. || 4. Mat. Calcular el valor aproximado de una magnitud en un intervalo cuando se conocen algunos de los valores que toma a uno y otro lado de dicho intervalo.[8]

Este método se utiliza cuando se desea conocer valores intermedios en una serie de datos precisos dados.

El modelo más popular se denomina diferencia dividida de Newton y sus formas más simples son las interpolación lineal (interpolación de primer orden) y la interpolación cuadrática (interpolación parabólica) y el caso más general sería la interpolación polinómica o interpolación en series de potencia.

1.3.1 Interpolación lineal.

En este caso se aproximan los datos por medio de una línea recta, para lo cual se necesita conocer dos puntos ***(x_0, $f(x_0)$) y (x_1, $f(x_1)$)***. La ecuación de la línea que pasa por dichos puntos es**:**

$$f(x) - f(x_0) = m(x-x_0) \qquad \textbf{(2.6)}$$

Siendo m la pendiente de la línea, calculada como: $m = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$ y al reemplazar en (2.4) tenemos:

$$f_1(x) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0) \qquad \textbf{(2.7)}$$

Ejemplo: Hallar el valor del Logaritmo natural de 2 si conocemos los valores en cero y en 4. La siguiente tabla muestra lo datos conocidos.

x	**1**	**2**	**4**
Ln(x)	0	¿ ?	1.386294

[8] *Diccionario de la Real Academia de la Lengua Española. Biblioteca de Consulta Microsoft® Encarta® 2005.*

En este caso contamos con dos puntos de datos conocidos y queremos calcular un valor que se encuentra en dicho intervalo, por lo tanto podemos hacerlo por interpolación lineal. El primer punto es **(x_0, f(x_0))=(1,0)** y el segundo: **(x_1, f(x_1)=(4,1.386294)**, reemplazando en la ecuación (2.5) obtenemos:

$$f_1(x) = 0 + \frac{1.386294 - 0}{4-1}(x-1)$$

simplificando términos obtenemos: $f_1(x) = 0.462098x$. Con esta ecuación podemos estimar el valor de *$f_1(x)$* para valores de ***x*** en el rango ***(0,4)***, de manera que podemos evaluarla en ***x=2***, para hallar *Ln(2)* que es el valor buscado, obteniendo: **f_1(2) = Ln(2) = 0.462098*(2-1) = 0.462098**. Si se compara este valor con aquel que se obtiene en una calculadora, que es **0,693147**, vemos que el valor calculado por interpolación lineal no es una buena aproximación, ya que tiene un error relativo alto (aproximadamente del 33%).

1.3.2 **Interpolación cuadrática.**

En este caso se utiliza un polinomio de grado 2 para realizar el cálculo, el cual estaría representado por la siguiente ecuación:

$$\boxed{f_2(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) \quad (2.8)}$$

En donde los coeficientes *b_0, b_1 y b_2* se calculan de la siguiente manera:

$$b_0 = f(x_0),\ b1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0} \text{ y } b2 = \frac{f(x_2) - f(x_1)}{x_2 - x_1} - \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

Ejemplo: hallar el valor de Ln(2) utilizando interpolación cuadrática, conociendo los valores Ln(x) cuando x vale 1, 4 y 6, como lo muestra la siguiente tabla:

x	**1**	**2**	**4**	**6**
Ln(x)	0	**¿ ?**	1.386294	1.791759

Tomemos los puntos de datos así: **(x_0, f(x_0)) = (1, 0), (x_1, f(x_1) = (4, 1.386294)** y **(x_2, f(x_2))=(6,1.791759)** con los que calculamos:

$$b_0 = 0,\ b1 = \frac{1.386294 - 0)}{4-1} = 0.462098 \text{ y } b2 = \frac{\frac{1.791759 - 1.366294}{6-4} - \frac{1.386294 - 0}{4-1}}{6-1} = -0.0518736$$

Reemplazando estos valores en la ecuación **(2.6)** obtenemos:

$$f_2(x) = 0 + 0.462098(x - 1) - 0.0518736(x - 1)(x - 4).$$

Si evaluamos esta ecuación en x=2 obtenemos el valor buscado:

$$f_2(x) = 0.462098\,(2 - 1) - 0.0518736\,(2 - 1)(2 - 4) = 0.5658442.$$

Si comparamos con el valor obtenido en la calculadora, obtenemos un error relativo de vemos que es una mejor aproximación, con un error relativo del 22,49%.

1.3.3 **Interpolación polinomial**.

(Interpolación en series de potencias). El caso más general es usar polinomios de *n*-esimo grado para aproximar la función deseada. La expresión general para un polinomio de grado *n* es:

$$f(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^n \qquad \textbf{(2.9)}$$

Para *n+1* puntos conocidos, solo existe un polinomio que pasa exactamente por todos los puntos y es precisamente el polinomio de grado n.

Utilizando MatLab es sencillo calcular el polinomio que aproxima una serie de puntos así como también utilizar dicho polinomio para calcular nuevos valores.

El proceso a seguir es:

- Crear dos vectores fila, uno con los datos de la variable independiente (X) y otro con los de la variable dependiente (y).
- Usar el comando **C=polyfit(X,y,n)** para hallar los coeficientes a_i (i=0,1,..n), donde **n** representa el grado del polinomio. Si queremos calcular el polinomio de máximo grado podemos hallar **n** con el comando **length(X)-1**.
- Utilizar el polinomio para evaluar nuevos puntos de datos, ya sea al interior del rango objeto de análisis (nuevos valores entre **Min(X)** y **Max(X)** - interpolación)[9] o fuera de dicho intervalo (extrapolación). Para esto se usa el comando polyval(C,xi), en donde C es el vector de coeficientes del polinomio evaluado anteriormente.
- Evaluamos el error utilizando la ecuación **(2.2)**.

Veamos el proceso por medio de un ejemplo.

Ejemplo: Aproximar los siguientes datos por medio de un polinomio:

x	94	205	371
f(x)	929	902	860

Se tienen 3 puntos por lo tanto se pueden aproximar por un polinomio de grado menor o igual a 2.

[9] Min(X) y Max(X) son funciones de Matlab que devuelven el valor mínimo y máximo del vector X respectivamente.

Usando MatLab desde la ventana de comandos, creamos dos vectores uno para x y otro para y.

```
x = [94   205   371];
y = [929  902  860];
```

Luego calculamos los coeficientes del polinomio:

C = polyfit(x,y,length(x)-1); % Se genera el polinomio de grado n-1.

La función *polyfit* recibe como argumentos los vectores x e y que contienen los puntos de datos y el grado del polinomio equivalente al número de datos menos 1.

Lo cual genera como resultado: **C = [0.000 –0.2327 951.1853]** de donde podemos construir el polinomio **P(x) = -0.2327X + 951.1853.**

Por último, evaluamos el error:

r=sum(y – polyval(C,x)).^2) %Evalúa el valor del error

Si se desea se puede crear la gráfica tanto de los datos originales como del polinomio calculado podemos usar estos comandos:

```
plot(x,y,'o');              %Dibuja la gráfica de los datos originales.
z = x(1):0.1:x(length(x));  % Genera un vector con incrementos de 0.1 desde el primer valor
                            en x de los datos originales hasta el último valor.
w = polyval(C,z);           % Evalúa el polinomio calculado en cada punto calculado en z.
hold on                     %evita que se borre la gráfica anterior.
plot(z, w,'r')              % Dibuja el polinomio en color rojo.
```

Los polinomios de grado superior a 3 tienen el inconveniente de que pueden introducir oscilaciones grandes en los datos, por lo tanto el error puede aumentar.

Ejemplo: Ajustar los siguientes datos a un polinomio.

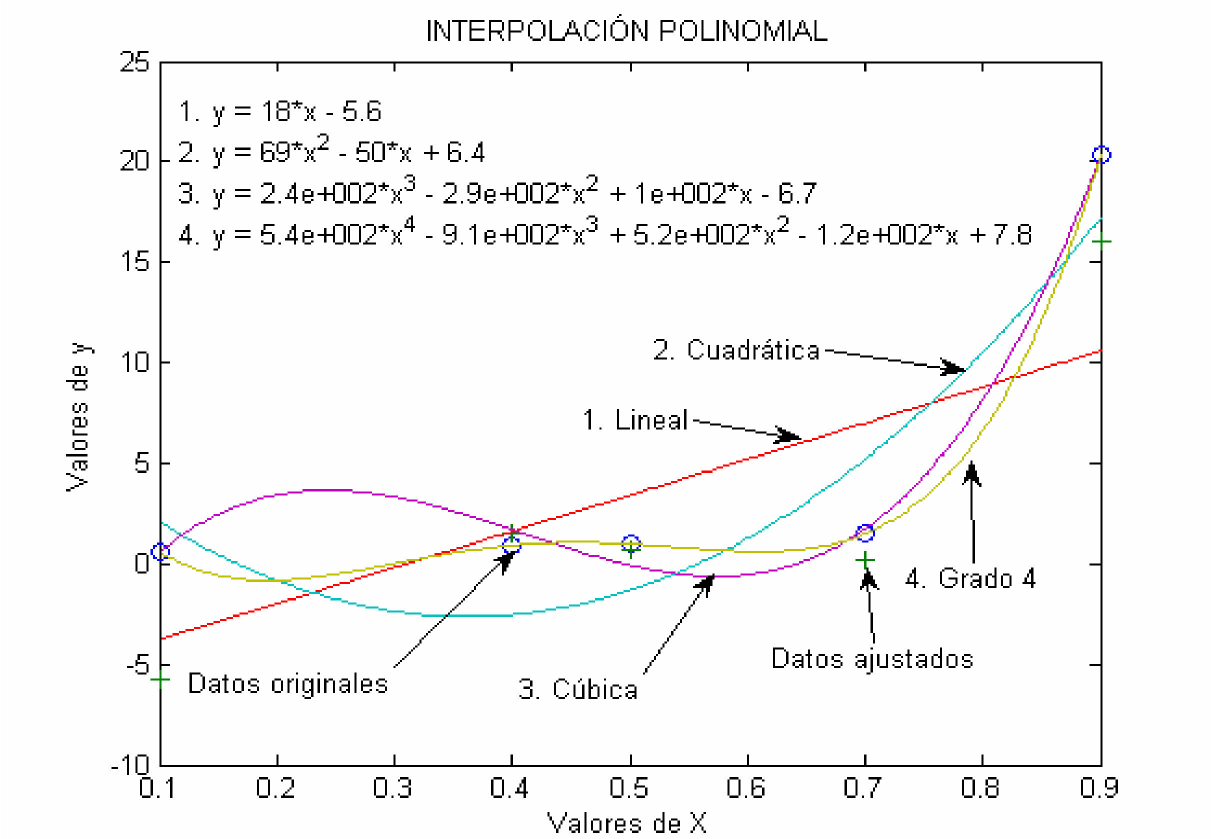
x	0.1	0.4	0.5	0.7	0.7	0.9
f(x)	0.61	0.92	0.99	1.52	1.47	2.03

En este caso tenemos 6 puntos por lo tanto podemos ajustarlos a un polinomio de grado 5 o menor. Vamos entonces a intentas comparar los resultados con distintos polinomios. Los siguientes comandos de MatLab permiten realizar la tarea.

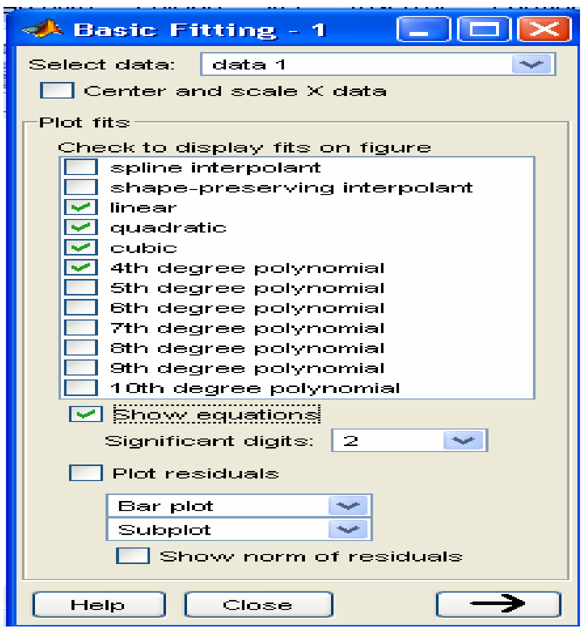
```
clear all;
x=[0.1 0.4 0.5 0.7 0.7 0.9];
y=[0.61 0.92 0.99 1.52 1.47 20.3];
```

```
C=polyfit(x,y,5);
yi=polyval(C,x);        %Recalculamos los datos originales usando el polinomio
plot(x,y,'o',x,yi,'+')  %graficamos los puntos de datos originales y los ajustados.
```

La siguiente gráfica nos muestra los resultados obtenidos.



Para hallar los polinomios restantes, se utilizo la opción de ajuste de curvas que provee MatLab. Para ello, una vez se hayan graficado algunos datos, procedemos a activar la opción ***Basic fitting***, eligiéndola en el menú ***Tools*** en la barra de opciones de la ventana grafica del MatLab. Con ello aparece una ventana como la mostrada en la siguiente figura. Al activar las respectivas opciones obtenemos los resultados mostrados en la gráfica siguiente. Como podemos observar las herramientas computacionales actuales incluyen una amplia gama de opciones para el ajuste de curvas lo que facilita el tratamiento de grandes volúmenes de datos de manera rápida y confiable.



1.3.4 **Ajuste de curvas no lineales con una función de potencia**. Cuando los datos observados no siguen comportamientos lineales, algunas veces conviene ajustarlos por medio de funciones como:

$$g(x) = \beta X^{\alpha} \qquad (2.10)$$

donde β, α son valores no determinados, pero que pueden resolverse así:

Aplicando logaritmos a ambos lados tenemos:

$$\ln(g(x)) = \ln(\beta x^{\alpha})$$
$$\ln(g(x)) = \ln(\beta) + \ln(x^{\alpha})$$
$$\ln(g(x)) = \ln(\beta) + \alpha \ln(x)$$
$$\downarrow \qquad \downarrow \qquad \downarrow$$
$$G \;=\; A + \; Bx$$

Si defino: ***G(x) = log(g(x))***, ***A =*** α y ***B = log(***β***)*** obtenemos una ecuación con las mismas características que la recta de regresión y el problema se resuelve por el método de los mínimos cuadrados convirtiendo el conjunto de datos **(x,y)** en **(log(x), log(y)).** Para calcular los coeficientes podemos usar: **C=polyfit(log(x),log(y),1).** Para luego calcular: **α = C(1)** y **β = exp(C(2))** y de esta manera definir completamente la ecuación **(2.10)**.

Ejemplo: Ajustar los siguientes datos a una función de potencia.

X	0.1	0.5	1.1	1.16	2.1	2.7	3.1	3.5	4.1	4.6
y	1.81	2.5	2.79	2.99	3.14	3.26	3.37	3.46	3.54	3.61

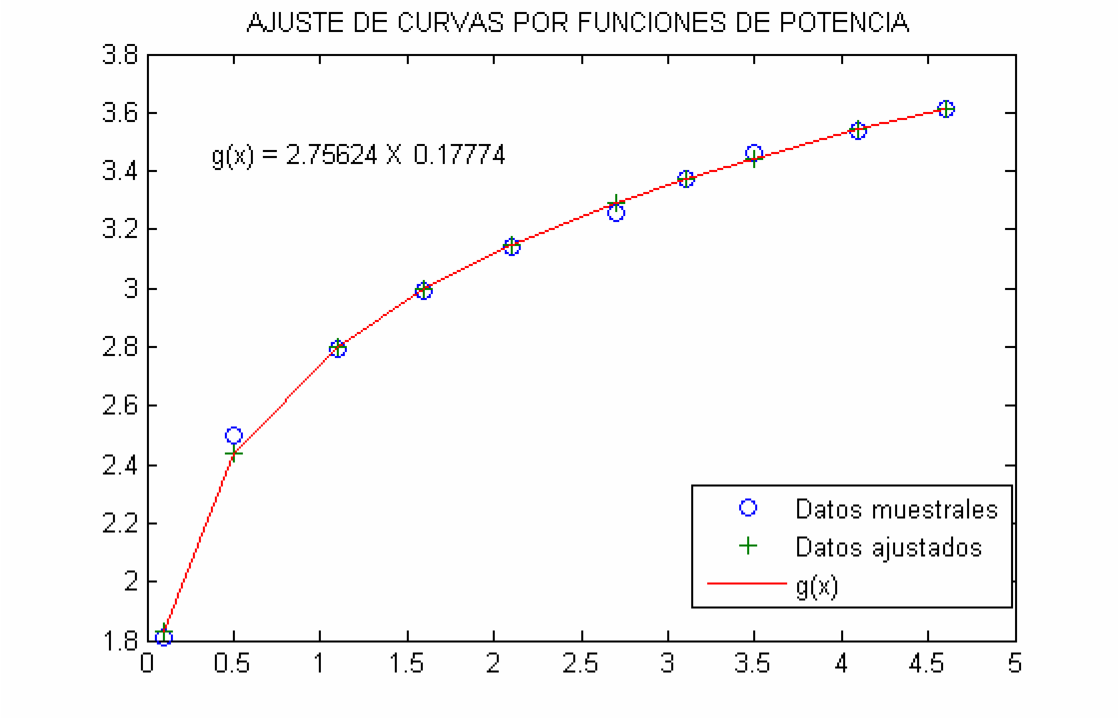
El siguiente guión de MatLab, permite realizar el trabajo:

```
clear all;
clc;
x=[0.1  0.5  1.1  1.6  2.1  2.7  3.1  3.5  4.1  4.6];
y=[1.81  2.50  2.79  2.99  3.14  3.26  3.37  3.46  3.54  3.61];
C=polyfit(log(x),log(y),1);
yi=polyval(C,x);      %Recalculamos los datos originales usando el polinomio
plot(x,y,'o',x,yi,'+')  %graficamos los puntos de datos originales y los ajustados.
```

El resultado obtenido para C = [0.17773963887162   1.01386785498698]. De manera que C(1) = 0.17774 y C(2) = exp(1.013867) = 2.75623880906059. Con estos valores la función de potencia sería:

g(x) = 2.75624 X $^{\mathbf{0.17774}}$**.** Si evaluamos el error tenemos:

r = sum((y – 2.75624 .* X.^0.17774).^2) = 0.00576665548140, el cual es un valor significativamente bajo, lo cual indica que los datos ajustados se acercan suficientemente a los datos muestrales. La siguiente figura confirma este hecho.



1.3.5 **Interpolación por medio de polinomios de lagrange.** Podemos formular los polinomios de Lagrange, a partir de los datos conocidos utilizando las siguientes ecuaciones:

$\boxed{f_n(x) = \sum_{i=0}^{n} L_i(x) f(x_i)}$, donde: $\boxed{L_i(x) = \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x - x_j}{x_i - x_j}}$

1.3.6 **Interpolación segmentaria.** Utilizando polinomios de *n*-esimo grado es factible llegar a cálculos erróneos debido a errores de redondeo y puntos alejados. Una alternativa es usar polinomios de orden inferior (cuadráticos y cúbicos) y aplicarlos a subconjuntos de datos.

1.3.6.1 Trazador cúbico. Es uno de los métodos de interpolación más usados en ingeniería.

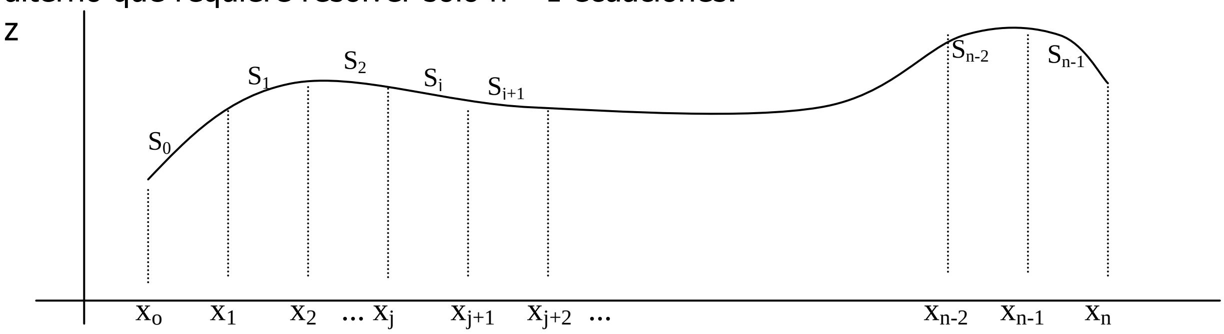
Cuando se trata de aproximar muchos puntos por medio de un único polinomio, el error crece rápidamente, por lo tanto se intenta aproximar cada intervalo de puntos dados por medio de un polinomio de tercer orden así:

$\boxed{f_i(x_i) = a_i x^3 + b_i x^2 + c_i x + d_i}$; *i=0, 1, 2, ..., n.*

Si se tienen n+1 puntos de datos se pueden crear n intervalos de modo que se desconocen 4n incógnitas y se requiere de 4n ecuaciones para poderlas evaluar. Estas ecuaciones se obtienen así:

1. Los valores de la función deben ser iguales en los puntos interiores. Esto produce 2n – 2 ecuaciones.
2. La primera y última función deben pasar por los puntos extremos, esto produce 2 ecuaciones adicionales.
3. Las primeras derivadas en los nodos interiores deben ser iguales. Esto genera n – 1 ecuaciones.
4. Las segundas derivadas en los nodos interiores deben ser iguales. Esto genera n – ecuaciones.
5. Las segundas derivadas en los nodos extremos deben ser cero. Esto genera 2 ecuaciones.

Como se ve es necesario conocer las primeras y segundas derivadas de los polinomios cúbicos, las cuales deben ser continuas en cada punto de datos. La condición 5 gráficamente se puede interpretar como si la función se volviera una línea recta en los nodos extremos. A esto se denomina segmentaria natural, ya que se asemeja a la curva que se puede trazar con una varilla flexible que recorre los puntos dados, si el valor de la segunda derivada en los nodos extremos no es cero, o si existe curvatura, esta información sirve para suministrar las dos condiciones finales. Existe un método alterno que requiere resolver solo n – 1 ecuaciones.



Según la figura, dada un función ***f*** definida en **[a,b]** y un conjunto de puntos de la forma: $\mathbf{a = x_0 < x_1 < \ldots < x_n = b}$, un interpolante de trazador cúbico es una función ***S*** para ***f*** que cumple las condiciones antes mencionadas equivalentes a:

a) S(x) es un polinomio cúbico denotado por $S_j(x)$ en subintervalo $[x_j, x_{j+1}]$, para cada *j=0,1,2,...,n-1.*

b) $S(x_j) = f(x_j)$ para cada *j=0,1,2,...,n.*

c) $S_{j+1}(x_{j+1}) = S_j(x_{j+1})$ para cada *j=0,1,2,...,n-2.*

d) $S'_{j+1}(x_{j+1}) = S'_j(x_{j+1})$ para cada *j=0,1,2,...,n-2*
e) $S''_{j+1}(x_{j+1}) = S''_j(x_{j+1})$ para cada *j=0,1,2,...,n-2*
f) En la frontera se cumple alguna de estas condiciones:
   i. $S''(x_0) = S''(x_n) = 0$ (Frontera natural o libre)
   ii. $S'(x_0) = f'(x_0)$ y $S'(x_n) = f'(x_n)$ (Frontera sujeta)

S(j) es un polinomio cúbico de la forma:

$$S_j(x)=a_j+b_j(x-x_j)+c_j(x-x_j)^2+d_j(x-x_j)^3$$ para **j**=0,1,2,...,n-1.

Sabemos que: $S_j(x) = a_j = f(x_j)$ y si consideramos la condición **c)** obtenemos:

$$a_{j+1}=S_{j+1}(x_{j+1})=a_j+b_j(x_{j+1}-x_j)+c_j(x_{j+1}-x_j)^2+d_j(x_{j+1}-x_j)^3$$,**j**=0,1,2,...,n-1.

Reemplacemos $h_j = x_{j+1} - x_j$ para **j**=0,1,2,...,n-1.

Si tomamos $a_n = f(x_n)$ la ecuación: $a_{j+1} = a_j + b_j h_j + c_j h_j^2 + d_j h_j^3$ **(1)**
la cual es válida para **j**=0,1,2,...,n-1.

Si definimos: $b_n = S'(x_n)$ obtenemos: $S'_j(x)=b_j+2c_j(x-x_j)+3d_j(x-x_j)^2$
lo cual significa que: $S'_j(x)=b_j$, y al aplicar la condición **d)** obtenemos

$$b_{j+1}=b_j+2c_jh_j+3d_jh_j^2$$ para **j**=0,1,2,...,n-1. **(2)**

Definamos: $c_n = \dfrac{S''(x_n)}{2}$

Con ***Cn*** y la condición **e)** obtenemos: $c_{j+1}=c_j+3d_jh_j$ para **j**=0,1,2,...n-1.

Resolviendo ***dj*** en **(3)** y sustituir en **(1)** y **(2)** se obtienen las ecuaciones para j=0,1,2,...,n-1:

$$a_{j+1} = a_j + b_j h_j + \frac{h_j^2}{3}(2c_j + c_{j+1}) \qquad \textbf{(4)}$$

$$b_{j+1} = b_j + h_j(c_j + c_{j+1}) \qquad \textbf{(5)}$$

Despejando $\mathbf{b_j}$ de la ecuación (4) obtenemos:

$$b_j = \frac{1}{h_j}(a_{j+1} - a_j) - \frac{h_j}{3}(2c_j + c_{j+1}) \qquad \textbf{(6)}$$

Reduciendo el índice para b_{j-1} se obtiene:

$$b_{j-1} = \frac{1}{h_{j-1}}(a_j - a_{j-1}) - \frac{h_{j-1}}{3}(2c_{j-1} + c_j) \qquad \textbf{(7)}$$

Con el índice reducido en 1 se obtiene el sistema de ecuaciones lineales:

$$\boxed{h_{j-1}c_{j-1} + 2c_j(h_{j-1} + h_j) + h_j c_j = \frac{3}{h_j}(a_{j+1} - a_j) - \frac{3}{h_{j-1}}(a_j + a_{j-1})}\ , \mathbf{j}=0,1,2,\ldots,n-1. \qquad \textbf{(8)}$$

Este sistema de ecuaciones contiene solo a $\{c_j\}_{j=0}^{n-1}$ como incógnitas ya que los valores de $\{h_j\}_{j=0}^{n-1}$ y $\{a_j\}_{j=0}^{n-1}$ están dados por el espaciado de los puntos en $\{x_j\}_{j=0}^{n-1}$ y los respectivos valores de f en cada uno de ellos.

Este sistema de ecuaciones contiene solo a $\{c_j\}_{j=0}^{n-1}$ como incógnitas ya que los valores de $\{h_j\}_{j=0}^{n-1}$ y $\{a_j\}_{j=0}^{n-1}$ están dados por el espaciado de los puntos en $\{x_j\}_{j=0}^{n-1}$ y los respectivos valores de f en cada uno de ellos.

Una vez conocidos los valores de $\{c_j\}_{j=0}^{n-1}$ s pueden hallar el resto de las constantes $\{b_j\}_{j=0}^{n-1}$ usando la ecuación (5) y $\{d_j\}_{j=0}^{n-1}$ usando la ecuación (3) para construir los polinomios cúbicos $\{S_j(x)\}_{j=0}^{n-1}$.

Si se cumplen las condiciones de frontera junto con la ecuación **(8)** se obtiene un sistema de ecuaciones que se puede expresar en forma matricial como: **Ax = b**, donde **A** es una matriz de (n+1), x > (n+1) cuya solución es única.

$$A=\begin{bmatrix} 1 & 0 & 0 & & & \\ h_0 & 2(h_0+h_1) & h_1 & & & \\ 0 & h_1 & 2(h_1+h_2) & h_2 & & \\ & & & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ & & & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{b=}\begin{bmatrix} 0 \\ \frac{3}{h_1}(a_2-a_1)-\frac{3}{h_0}(a_1-a_0) \\ \\ \frac{3}{h_{n-1}}(a_n-a_{n-1})-\frac{3}{h_{n-2}}(a_{n-1}-a_{n-2}) \\ 0 \end{bmatrix} \quad \mathbf{y\ x=}\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \\ c_n \end{bmatrix}$$

Este sistema se puede resolver matricialmente como **x = A^{-1}b**.

1.3.7 Interpolación inversa.

Hasta ahora se han planteado métodos para hallar valores de la variable dependiente basándonos en valores conocidos de la variable independiente.

En el caso de la interpolación inversa, se desea hallar el valor de la variable independiente que produce un determinado valor de la variable dependiente (hallar ***x*** conocido ***f(x)***).

La solución al problema se reduce a calcular la raíz de la ecuación que surge de ajustar los puntos de datos conocidos a una ecuación, ya sea usando regresión o interpolación.

Ejemplo: Hallar el valor de **x**, para el cual **f(x) = 0.3** , dados los siguientes datos:

X	1	2	3		5	6	7
y	1	0.5	0.33	0.25	0.2	0.1667	0.1429

Vamos a intentar hallar el polinomio que mejor se ajuste a este conjunto de datos, evaluando el error en cada uno de ellos.

Con el comando **polyfit(x,y,n)**, donde **n** es el grado del polinomio y **sum((y-polyval(cu,x)).^2)** para calcular el error, obtenemos los resultados mostrados en la tabla:

Grado	Polinomio	Error	X
Lineal	*f(x)* = -0.120282 X + 0.851071	0.146507	4.581482
Cuadrático	*f(x)* = $0.037196X^2 - 0.441785X + 1.297429$	0.030287	3.031433
Cúbico	*f(x)* = $-0.010939X^3 + 0.16846309\ X^2 - 0.866348\ X + 1.69122$	0.0044404	3.089783
4	*f(x)* = $0.002999X^4 - 0.058927X^3 + 0.427683X^2 - 1.404498X + 2.030571$	3.693145E-4	3.190552
5	*f(x)* = $-0.000725X^5 + 0.017491X^4 - 0.166407X^3 + 0.789975X^2 - 1.944278X + 2.303843$	9.300097E-6	3.311523
6	*f(x)* = $0.000126X^6 - 0.003815X^5 + 0.046694X^4 - 0.304053X^3 + 1.126528X^2 -2.341483X + 2.476$	2.456716E-29	3.273438

Como era de esperarse, el polinomio que mejor se ajusta al conjunto de datos es el de grado 6.

Ahora podemos calcular el valor de **x** para el cual **f(x) = 0.3**, que equivale a encontrar la raíz de la ecuación: f(x) = 0.3. Usando el polinomio de grado 6 tenemos:

$0.000126X^6-0.003815X^5+0.046694X^4-0.304053X^3+1.126528X^2-2.341483X+2.476 = 0.3$

Usando el método de bisección en el intervalo [3, 4] con 5 cifras significativas obtenemos el valor de: **x = 3.273438.**