Tema 3
Interpolación

Dra. Paula Triguero Navarro

Máster en Ingeniería Matemática y Computación
Escuela Superior en Ingeniería y Tecnología

LA UNIVERSIDAD
EN INTERNET

Contenido

1. Introducción
2. Interpolación de Newton
   - Diferencias divididas de Newton
   - Implementación computacional
3. Interpolación de Lagrange
   - Implementación computacional
4. Interpolación de Hermite
   - Implementación computacional
5. Splines
   - Splines cúbicos naturales

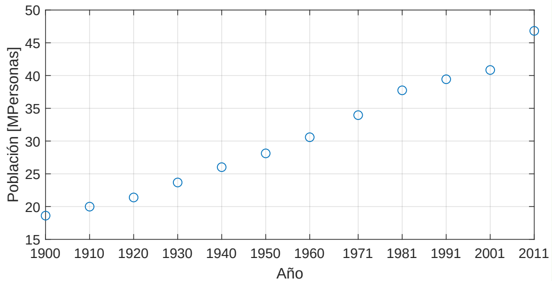
1

Introducción

Introducción

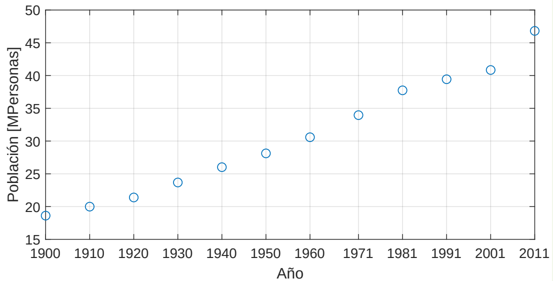
Ejemplo 1. Evolución demográfica en España

Año	Población [MPersonas]
1900	18.617
1910	19.991
1920	21.389
1930	23.677
1940	26.014
1950	28.118
1960	30.583
1971	33.956
1981	37.743
1991	39.434
2001	40.847
2011	46.816



Ejemplo 1. Evolución demográfica en España

Año	Población [MPersonas]
1900	18.617
1910	19.991
1920	21.389
1930	23.677
1940	26.014
1950	28.118
1960	30.583
1971	33.956
1981	37.743
1991	39.434
2001	40.847
2011	46.816



¿Qué población había en 1983?

Interpolación

Para obtener los puntos intermedios en los que no conocemos los datos, utilizaremos polinomios de la forma:

$$p_n(x) = \sum_{i=0}^{n} a_i x^i, \quad n \geq 0, \quad a_i \in \mathbb{R}.$$

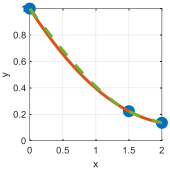
Teorema 1 (Teorema de aproximación de Weierstrass)

Sea $f \in \mathcal{C}[a,b]$. Entonces, para todo $\epsilon > 0$ existe un polinomio $p(x)$ tal que

$$|f(x) - p(x)| < \epsilon, \quad \forall x \in [a,b].$$

Dada cualquier función definida en un intervalo cerrado, existe un polinomio que se ajusta cuanto se quiera a la función

INTERPOLACIÓN



$$\boldsymbol{n+1} \text{ puntos}$$
$$\{(x_0, f(x_0)), (x_1, f(x_1)), \dots, (x_n, f(x_n))\}$$

Polinomios de grado $\boldsymbol{n}$

- Newton
- Lagrange
- Hermite

Funciones a trozos

- Splines
  - Naturales
  - Restringidos

Objetivos

- Conocer los polinomios de interpolación de Newton, Lagrange y Hermite
- Implementar computacionalmente los polinomios de interpolación de Newton, Lagrange y Hermite
- Conocer los splines cúbicos
- Implementar computacionalmente los splines cúbicos naturales

2

Interpolación de Newton

Interpolación de Newton

- El polinomio de grado n de interpolación de Newton pasa por los $n+1$ puntos

$$(x_0, f(x_0)), (x_1, f(x_1)), \ldots (x_n, f(x_n)).$$

- Expresión general:

$$p_n(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) + \cdots + b_n(x - x_0)(x - x_1)\cdots(x - x_{n-1}).$$

- Para la obtención de los coeficientes b_i, $i = 0, 1, \ldots, n$, se fuerza a que el polinomio p_n pase por todos los puntos

Interpolación lineal

- Puntos: $(x_0, f(x_0))$, $(x_1, f(x_1))$
- Expresión general: $p_1(x) = b_0 + b_1(x - x_0)$
- Coeficientes b_0, b_1:

$$\begin{cases} p_1(x_0) = f(x_0) = b_0 \\ p_1(x_1) = f(x_1) = b_0 + b_1(x_1 - x_0) \end{cases} \quad \Leftrightarrow \quad \begin{cases} b_0 = f(x_0) \\ b_1 = \dfrac{f(x_1) - f(x_0)}{x_1 - x_0} \end{cases}$$

- Polinomio de interpolación de Newton de grado uno:

$$p_1(x) = f(x_0) + f[x_1, x_0](x - x_0),$$

donde $f[x_1, x_0] = \dfrac{f(x_1) - f(x_0)}{x_1 - x_0}$.

Interpolación cuadrática

- Puntos: $(x_0, f(x_0))$, $(x_1, f(x_1))$, $(x_2, f(x_2))$
- Expresión general: $p_2(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1)$
- Coeficientes b_0, b_1, b_2:

$$\begin{cases} p_2(x_0) = f(x_0) = b_0 \\ p_2(x_1) = f(x_1) = b_0 + b_1(x_1 - x_0) \\ p_2(x_2) = f(x_2) = b_0 + b_1(x_2 - x_0) + b_2(x_2 - x_0)(x_2 - x_1) \end{cases} \Leftrightarrow$$

$$\Leftrightarrow \quad b_0 = f(x_0), \quad b_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0}, \quad b_2 = \frac{\frac{f(x_2)-f(x_1)}{x_2-x_1} - \frac{f(x_1)-f(x_0)}{x_1-x_0}}{x_2 - x_0}$$

- Polinomio de interpolación de Newton de grado dos:

$$p_2(x) = f(x_0) + f[x_1, x_0](x - x_0) + f[x_2, x_1, x_0](x - x_0)(x - x_1),$$

donde

$$f[x_1, x_0] = \frac{f(x_1) - f(x_0)}{x_1 - x_0}, \qquad f[x_2, x_1, x_0] = \frac{f[x_2, x_1] - f[x_1, x_0]}{x_2 - x_0}.$$

Contenidos

1 Introducción

2 Interpolación de Newton
- Diferencias divididas de Newton
- Implementación computacional

3 Interpolación de Lagrange

4 Interpolación de Hermite

5 Splines

Interpolación de Newton

Diferencias divididas de Newton

Las diferencias divididas de orden n se definen a partir de las diferencias divididas de orden $n-1$ como:

$$f[x_n, x_{n-1}, x_{n-2}, \cdots, x_1, x_0] = \frac{f[x_n, x_{n-1}, \cdots, x_1] - f[x_{n-1}, x_{n-2}, \cdots, x_1, x_0]}{x_n - x_0}.$$

$f(x_0)$	$f[x_1, x_0]$	$f[x_2, x_1, x_0]$	$f[x_3, x_2, x_1, x_0]$
$f(x_1)$	$f[x_2, x_1]$	$f[x_3, x_2, x_1]$	
$f(x_2)$	$f[x_3, x_2]$		
$f(x_3)$			

Ejemplo 2. Diferencia dividida de orden 3

$$f[x_3, x_2, x_1, x_0] = \frac{f[x_3, x_2, x_1] - f[x_2, x_1, x_0]}{x_3 - x_0} = \frac{\frac{f[x_3,x_2]-f[x_2,x_1]}{x_3-x_1} - \frac{f[x_2,x_1]-f[x_1,x_0]}{x_2-x_0}}{x_3 - x_0}$$

$$= \frac{\dfrac{\frac{f(x_3)-f(x_2)}{x_3-x_2} - \frac{f(x_2)-f(x_1)}{x_2-x_1}}{x_3 - x_1} - \dfrac{\frac{f(x_2)-f(x_1)}{x_2-x_1} - \frac{f(x_1)-f(x_0)}{x_1-x_0}}{x_2 - x_0}}{x_3 - x_0}$$

Interpolación general (grado n)

- Puntos: $(x_0, f(x_0))$, $(x_1, f(x_1)), \ldots, (x_n, f(x_n))$
- Expresión general:
  $p_n(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) + \cdots + b_n(x - x_0)(x - x_1) \cdots (x - x_{n-1})$
- Coeficientes:

$$
\begin{cases}
b_0 = f(x_0) \\
b_1 = f[x_1, x_0] \\
b_2 = f[x_2, x_1, x_0] \\
\vdots \\
b_i = f[x_i, \cdots, x_1, x_0] \\
\vdots \\
b_n = f[x_n, x_{n-1}, \cdots, x_1, x_0]
\end{cases}
$$

- Polinomio de interpolación de Newton de grado n:

$$
\begin{aligned}
p_n(x) = f(x_0) + f[x_1, x_0](x - x_0) + f[x_2, x_1, x_0](x - x_0)(x - x_1) + \cdots \\
\cdots + f[x_n, x_{n-1}, \cdots, x_1, x_0](x - x_{n-1})(x - x_{n-2}) \cdots (x - x_1)(x - x_0).
\end{aligned}
$$

Teorema 2 (Error en el polinomio de interpolación de Newton)

Sean $a = x_0 < x_1 < \ldots < x_n = b$ *y sea* $f \in \mathcal{C}^{n+1}[a,b]$. *Entonces,* $\forall x \in [a,b]$ *existe un* $\xi(x) \in (a,b)$ *tal que*

$$f(x) = p_n(x) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x-x_0)\cdots(x-x_n),$$

donde p_n *es el polinomio de interpolación de Newton de grado* n *dado por:*

$$\begin{aligned} p_n(x) = {} & f(x_0) + f[x_1,x_0](x-x_0) + f[x_2,x_1,x_0](x-x_0)(x-x_1) + \cdots \\ & \cdots + f[x_n,x_{n-1},\cdots,x_1,x_0](x-x_{n-1})(x-x_{n-2})\cdots(x-x_1)(x-x_0). \end{aligned}$$

Ejemplo 3. Dados los datos del censo de población de España desde 1971, calcula el polinomio de interpolación de Newton de mayor grado posible para estimar la población en el año 2005.

Año	1971	1981	1991	2001	2011
Población [MPersonas]	33.956	37.743	39.434	40.847	46.816

- 5 datos ➔ $n = 4$
- Polinomio de interpolación de grado 4:

$$\begin{aligned} p_4(x) = {} & f(x_0) + f[x_1, x_0](x - x_0) + f[x_2, x_1, x_0](x - x_0)(x - x_1) \\ & + f[x_3, x_2, x_1, x_0](x - x_0)(x - x_1)(x - x_2) \\ & + f[x_4, x_3, x_2, x_1, x_0](x - x_0)(x - x_1)(x - x_2)(x - x_3) \end{aligned}$$

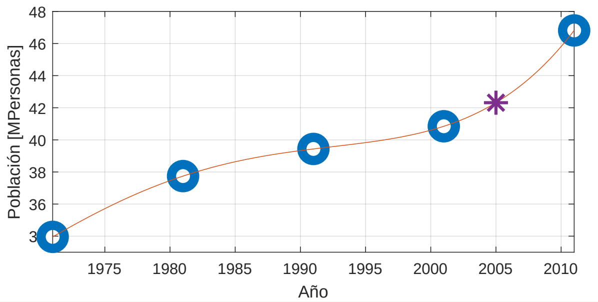
donde

$$\begin{aligned} & f(x_0) = f(1971) = 33.956, \\ & f[x_1, x_0] = f[1981, 1971] = 0.3787, \\ & f[x_2, x_1, x_0] = f[1991, 1981, 1971] = -0.01048, \\ & f[x_3, x_2, x_1, x_0] = f[2001, 1991, 1981, 1971] = 0.000303, \\ & f[x_4, x_3, x_2, x_1, x_0] = f[2011, 2001, 1991, 1981, 1971] = 0.0000127. \end{aligned}$$

Ejemplo 3. Dados los datos del censo de población de España desde 1971, calcula el polinomio de interpolación de Newton de mayor grado posible para estimar la población en el año 2005.

$$\begin{aligned} p_4(x) = 33.956 + 0.3787(x-1971) - 0.01048(x-1971)(x-1981) + \\ + 0.000303(x-1971)(x-1981)(x-1991) \\ + 0.0000127(x-1971)(x-1981)(x-1991)(x-2001). \end{aligned}$$

- Población en el año 2005: $p(2005) = 42.315$ MPersonas.



Contenidos

1 Introducción

2 Interpolación de Newton
- Diferencias divididas de Newton
- Implementación computacional

3 Interpolación de Lagrange

4 Interpolación de Hermite

5 Splines

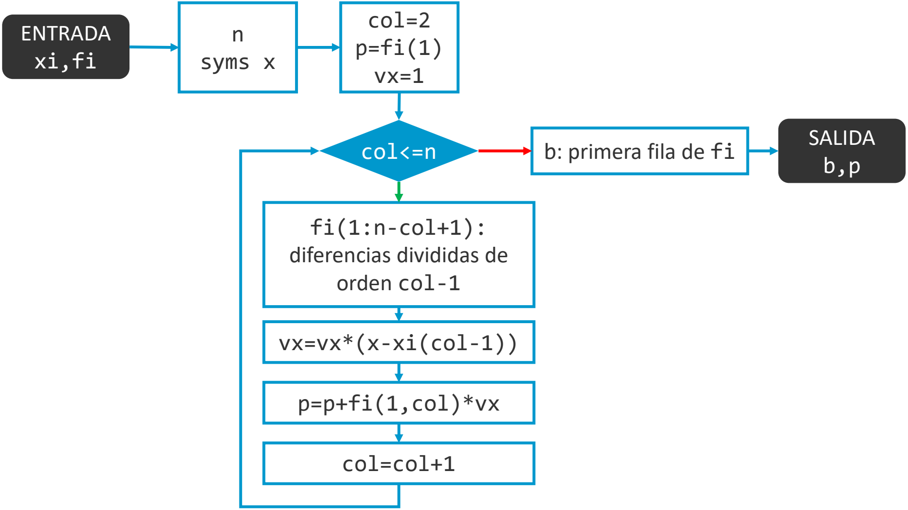
$$
\begin{aligned}
p_n(x) = f(x_0) &+ f[x_1, x_0](x - x_0) + f[x_2, x_1, x_0](x - x_0)(x - x_1) \\
&+ f[x_3, x_2, x_1, x_0](x - x_0)(x - x_1)(x - x_2) + \cdots \\
&+ f[x_n, x_{n-1}, \ldots, x_1, x_0](x - x_0)(x - x_1)(x - x_2)\cdots(x - x_{n-1})
\end{aligned}
$$

$f(x_0)$	$f[x_1,x_0]$	$f[x_2,x_1,x_0]$	$f[x_3,x_2,x_1,x_0]$
$f(x_1)$	$f[x_2,x_1]$	$f[x_3,x_2,x_1]$	
$f(x_2)$	$f[x_3,x_2]$		
$f(x_3)$			

polinomioNewton.m

- **Entrada:** Los $n+1$ puntos conocidos $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$. Las coordenadas x_i de los puntos irán en un vector, y las coordenadas $f(x_i)$ en otro vector.
- **Salida:** Los $n+1$ coeficientes $b_0, \ldots, b_n$ y el polinomio $p(x)$.

```
function [b,p]=polinomioNewton(xi,fi)
...
end
```


ENTRADA
xi,fi
n
syms x
col=2
p=fi(1)
vx=1
col<=n
b: primera fila de fi
SALIDA
b,p
fi(1:n-col+1):
diferencias divididas de orden col-1
vx=vx*(x-xi(col-1))
p=p+fi(1,col)*vx
col=col+1

3

Interpolación de Lagrange

- El polinomio de grado n de interpolación de Lagrange pasa por los $n+1$ puntos

$$(x_0, f(x_0)), (x_1, f(x_1)), \ldots (x_n, f(x_n)).$$

- Expresión general:

$$\begin{aligned} p_n(x) &= L_{n,0}(x)f(x_0) + L_{n,1}(x)f(x_1) + \cdots + L_{n,n}(x)f(x_n) \\ &= \sum_{i=0}^{n} L_{n,i}(x)f(x_i). \end{aligned}$$

- Las funciones $L_{n,i}(x)$, $i = 0, 1, \ldots, n$, se construyen de forma que

$$L_{n,i}(x_j) = \begin{cases} 1, & i = j, \\ 0, & i \neq j, \end{cases}$$

Interpolación lineal

- Puntos: $(x_0, f(x_0))$, $(x_1, f(x_1))$
- Expresión general: $p_1(x) = L_0(x)f(x_0) + L_1(x)f(x_1)$
- Se definen las funciones $L_0(x)$ y $L_1(x)$ tales que

$$\left\{ \begin{array}{l} L_0(x_0) = 1, \\ L_0(x_1) = 0, \end{array} \right. \left\{ \begin{array}{l} L_1(x_0) = 0, \\ L_1(x_1) = 1, \end{array} \right. \quad \Leftrightarrow \quad L_0(x) = \frac{x - x_1}{x_0 - x_1}, \quad L_1(x) = \frac{x - x_0}{x_1 - x_0}$$

- Polinomio de interpolación de Lagrange de primer orden:

$$p_1(x) = \frac{x - x_1}{x_0 - x_1} f(x_0) + \frac{x - x_0}{x_1 - x_0} f(x_1).$$

Interpolación general (grado n)

- Puntos: $(x_0, f(x_0))$, $(x_1, f(x_1)), \dots, (x_n, f(x_n))$
- Expresión general: $p_n(x) = \sum_{i=0}^{n} L_{n,i}(x) f(x_i)$
- Las funciones $L_{n,i}(x)$ satisfacen:

$$\left\{\begin{array}{l} L_{n,0}(x_0) = 1, \\ L_{n,0}(x_1) = 0, \\ \vdots \\ L_{n,0}(x_{n-1}) = 0, \\ L_{n,0}(x_n) = 0, \end{array}\right. \quad \left\{\begin{array}{l} L_{n,1}(x_0) = 0, \\ L_{n,1}(x_1) = 1, \\ \vdots \\ L_{n,1}(x_{n-1}) = 0, \\ L_{n,1}(x_n) = 0, \end{array}\right. \quad \dots \left\{\begin{array}{l} L_{n,n}(x_0) = 0, \\ L_{n,n}(x_1) = 0, \\ \vdots \\ L_{n,n}(x_{n-1}) = 0, \\ L_{n,n}(x_n) = 1, \end{array}\right.$$

es decir,

$$L_{n,i}(x_j) = \left\{\begin{array}{ll} 1, & i = j, \\ 0, & i \neq j, \end{array}\right.$$

Por tanto:

$$L_{n,i}(x) = \frac{(x - x_0)\cdots(x - x_{i-1})(x - x_{i+1})\cdots(x - x_n)}{(x_i - x_0)\cdots(x_i - x_{i-1})(x_i - x_{i+1})\cdots(x_i - x_n)} = \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)}$$

Ejemplo 4. Polinomio de interpolación de Lagrange de grado 2

Puntos: $(x_0, f(x_0))$, $(x_1, f(x_1))$, $(x_2, f(x_2))$

$$p_2(x) = L_0(x)f(x_0) + L_1(x)f(x_1) + L_2(x)f(x_2),$$

donde
$L_0(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}$, $\quad L_1(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}$, $\quad L_2(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$

Teorema 3 (Error en el polinomio de interpolación de Lagrange)

Sean $a = x_0 < x_1 < \ldots < x_n = b$ y sea $f \in \mathcal{C}^{n+1}[a,b]$. Entonces, $\forall x \in [a,b]$ existe un $\xi(x) \in (a,b)$ tal que

$$f(x) = p_n(x) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x-x_0)\cdots(x-x_n),$$

donde p_n es el polinomio de interpolación de Lagrange de grado n dado por:

$$p_n(x) = \sum_{i=0}^{n} L_{n,i}(x)f(x_i) = \sum_{i=0}^{n} \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{(x-x_j)}{(x_i-x_j)} f(x_i).$$

Ejemplo 5. Dados los datos del censo de población de España desde 1971, calcula el polinomio de interpolación de Lagrange de mayor grado posible para estimar la población en el año 2005.

Año	1971	1981	1991	2001	2011
Población [MPersonas]	33.956	37.743	39.434	40.847	46.816

- 5 datos ➔ $n = 4$
- Polinomio de interpolación de grado 4:

$$p_4(x) = L_0(x)f(x_0) + L_1(x)f(x_1) + L_2(x)f(x_2) + L_3(x)f(x_3) + L_4(x)f(x_4)$$

donde

$$L_0(x) = \frac{(x-1981)(x-1991)(x-2001)(x-2011)}{(1971-1981)(1971-1991)(1971-2001)(1971-2011)} = \frac{(x-1981)(x-1991)(x-2001)(x-2011)}{240000}$$

$$L_1(x) = \frac{(x-1971)(x-1991)(x-2001)(x-2011)}{(1981-1971)(1981-1991)(1981-2001)(1981-2011)} = \frac{-(x-1971)(x-1991)(x-2001)(x-2011)}{60000}$$

$$L_2(x) = \frac{(x-1971)(x-1981)(x-2001)(x-2011)}{(1991-1971)(1991-1981)(1991-2001)(1991-2011)} = \frac{(x-1971)(x-1991)(x-2001)(x-2011)}{40000}$$

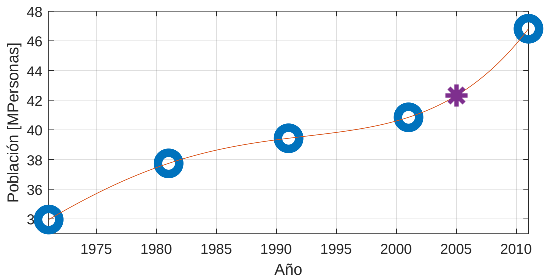
$$L_3(x) = \frac{(x-1971)(x-1981)(x-1991)(x-2011)}{(2001-1971)(2001-1981)(2001-1991)(2001-2011)} = \frac{-(x-1971)(x-1981)(x-1991)(x-2011)}{60000}$$

$$L_4(x) = \frac{(x-1971)(x-1981)(x-1991)(x-2001)}{(2011-1971)(2011-1981)(2011-1991)(2011-2001)} = \frac{(x-1971)(x-1981)(x-1991)(x-2011)}{240000}.$$

Ejemplo 5. Dados los datos del censo de población de España desde 1971, calcula el polinomio de interpolación de Lagrange de mayor grado posible para estimar la población en el año 2005.

$$\begin{aligned} p_4(x) \quad = \quad & \frac{33.956}{240000}(x-1981)(x-1991)(x-2001)(x-2011) \\ & + \frac{37.743}{60000}(x-1971)(x-1991)(x-2001)(x-2011) \\ & + \frac{39.434}{40000}(x-1971)(x-1981)(x-2001)(x-2011) \\ & + \frac{40.847}{60000}(x-1971)(x-1981)(x-1991)(x-2011) \\ & + \frac{46.816}{240000}(x-1971)(x-1981)(x-1991)(x-2001). \end{aligned}$$

- Población en el año 2005: $p_4(2005) = 42.316$ MPersonas.



Contenidos

1. Introducción
2. Interpolación de Newton
3. Interpolación de Lagrange
   - Implementación computacional
4. Interpolación de Hermite
5. Splines

$$p_n(x) = L_{n,0}(x)f(x_0) + L_{n,1}(x)f(x_1) + L_{n,2}(x)f(x_2) + \cdots + L_{n,n}(x)f(x_n),$$

$$L_{n,i}(x) = \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)}$$

polinomioLagrange.m

- **Entrada:** Los $n+1$ puntos conocidos $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$. Las coordenadas x_i de los puntos irán en un vector, y las coordenadas $f(x_i)$ en otro vector.
- **Salida:** El polinomio $p_n(x)$.

```
function p = polinomioLagrange(xi,fi)
...
end
```

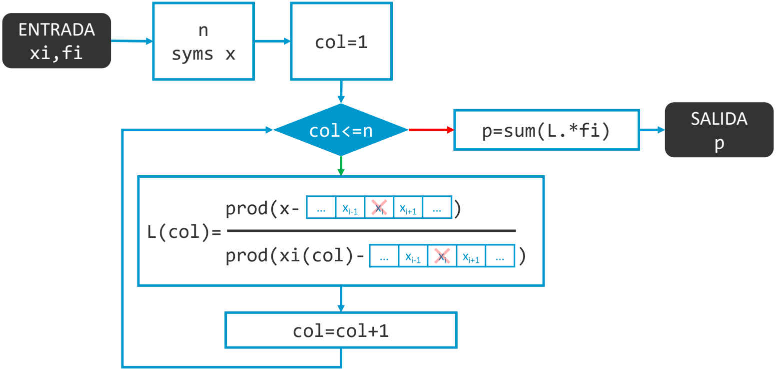
Interpolación de Lagrange
Implementación computacional

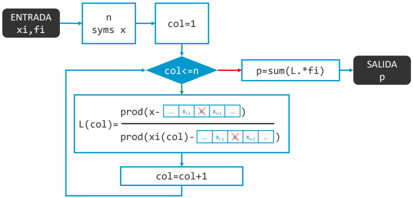
- Obtención de las funciones $L_{n,i}(x)$, $i = 0, 1, \ldots, n$:
  - Vector para el numerador:

$$x - \begin{bmatrix} x_0 & x_1 & \cdots & x_{i-1} & x_{i+1} & \cdots & x_{n-1} & x_n \end{bmatrix}$$

  - Vector para el denominador:

$$x_i - \begin{bmatrix} x_0 & x_1 & \cdots & x_{i-1} & x_{i+1} & \cdots & x_{n-1} & x_n \end{bmatrix}$$





polinomioLagrange.m

```
function p = polinomioLagrange(xi,fi)
xi=xi(:); fi=fi(:);
n=length(xi);
syms x
col=1;
  while col<=n
    ...
    num=...
    den=...
    L(col)=num/den;
    col=col+1;
  end
end
```

4

Interpolación de Hermite

Teorema 4 (Polinomio de Hermite)

Sean $f \in \mathcal{C}^1[a,b]$ *y* $x_0, \ldots, x_n \in [a,b]$. *Entonces, el único polinomio que coincide con* f *y* f' *en los puntos* $x_0, \ldots, x_n$ *es el polinomio de Hermite de grado menor o igual a* $2n+1$ *dado por*

$$H_{2n+1}(x) = \sum_{i=0}^{n} f(x_i) H_{n,i}(x) + f'(x_i) \hat{H}_{n,i}(x),$$

donde

$$H_{n,i} = \left[1 - 2(x - x_i) L'_{n,i}(x_i)\right] L^2_{n,i}(x), \qquad \hat{H}_{n,i} = (x - x_i) L^2_{n,i}(x),$$

siendo $L_{n,i}(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)}$ *las funciones de Lagrange.*

Teorema 4 (Polinomio de Hermite)

Sean $f \in \mathcal{C}^1[a,b]$ y $x_0, \ldots, x_n \in [a,b]$. Entonces, el único polinomio que coincide con f y f' en los puntos $x_0, \ldots, x_n$ es el polinomio de Hermite de grado menor o igual a $2n+1$ dado por

$$H_{2n+1}(x) = \sum_{i=0}^{n} f(x_i) H_{n,i}(x) + f'(x_i) \hat{H}_{n,i}(x),$$

donde

$$H_{n,i} = \left[1 - 2(x - x_i) L'_{n,i}(x_i)\right] L^2_{n,i}(x), \qquad \hat{H}_{n,i} = (x - x_i) L^2_{n,i}(x),$$

siendo $L_{n,i}(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)}$ las funciones de Lagrange.

Polinomio de Hermite de grado $2n+1$

- Además de conocer el valor de la función en los $n+1$ puntos, requiere conocer el valor de la derivada de la función en éstos.
- Requiere conocer:

$$(x_0, f(x_0), f'(x_0)), \quad (x_1, f(x_1), f'(x_1)), \quad \ldots, \quad (x_n, f(x_n), f'(x_n))$$

Teorema 4 (Polinomio de Hermite)

Sean $f \in \mathcal{C}^1[a,b]$ y $x_0, \dots, x_n \in [a,b]$. Entonces, el único polinomio que coincide con f y f' en los puntos $x_0, \dots, x_n$ es el polinomio de Hermite de grado menor o igual a $2n+1$ dado por

$$H_{2n+1}(x) = \sum_{i=0}^{n} f(x_i) H_{n,i}(x) + f'(x_i) \hat{H}_{n,i}(x),$$

donde

$$H_{n,i} = \left[1 - 2(x - x_i) L'_{n,i}(x_i)\right] L^2_{n,i}(x), \qquad \hat{H}_{n,i} = (x - x_i) L^2_{n,i}(x),$$

siendo $L_{n,i}(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)}$ las funciones de Lagrange.

Pasos para la obtención del polinomio de Hermite

1	2	3	4	5
$L_{n,i}(x)$	$L'_{n,i}(x)$	$H_{n,i}(x)$	$\widehat{H}_{n,i}(x)$	$H_{2n+1}(x)$

Teorema 5 (Error en el polinomio de interpolación de Hermite)

Sean $a = x_0 < x_1 < \ldots < x_n = b$ *y sea* $f \in \mathcal{C}^{n+1}[a, b]$. *Entonces,* $\forall x \in [a, b]$ *existe un* $\xi(x) \in (a, b)$ *tal que*

$$f(x) = H_{2n+1}(x) + \frac{f^{(2n+2)}(\xi(x))}{(2n+2)!}(x - x_0)^2 \cdots (x - x_n)^2,$$

donde H_{2n+1} *es el polinomio de interpolación de Hermite de grado menor o igual a* $2n + 1$.

Ejemplo 6. Funciones de Bessel

Las funciones de Bessel de primera especie y orden k, $J_k(x)$, son solución de la ecuación diferencial de Bessel:

$$x^2 \frac{d^2y}{dx^2} + x\frac{dy}{dx} + (x^2 - k^2)y = 0.$$

Como caso particular, se tiene que $J_0'(x) = -J_1(x)$.
Conociendo los datos

$\downarrow x, k \rightarrow$	0	1
0.0000	1.0000	0.0000
0.5000	0.9385	0.2423
1.0000	0.7652	0.4401

calcula el polinomio de interpolación de Hermite para obtener $J_0(0.75)$.

- Los valores de la tabla son los de $J_0(x)$ y $J_0'(x)$ en $x_0 = 0$, $x_1 = 0.5$ y $x_2 = 1$
- Calcularemos el polinomio de interpolación de Hermite para $n = 2$:

$$\begin{aligned} H_5(x) = {} & J_0(x_0)H_0(x) + J_0(x_1)H_1(x) + J_0(x_2)H_2(x) + J_0'(x_0)\hat{H}_0(x) \\ & + J_0'(x_1)\hat{H}_1(x) + J_0'(x_2)\hat{H}_2(x) \end{aligned}$$

Ejemplo 6. Funciones de Bessel

1. Cálculo de $L_i(x)$, $i = 0, 1, 2$.

$$L_0(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} = 2x^2 - 3x + 1,$$

$$L_1(x) = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} = -4x^2 + 4x,$$

$$L_2(x) = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} = 2x^2 - x.$$

2. Cálculo de $L_i'(x)$, $i = 0, 1, 2$.

$$L_0'(x) = 4x - 3, \quad L_1'(x) = -8x + 4, \quad L_2(x) = 4x - 1.$$

3. Cálculo de $H_i(x)$, $i = 0, 1, 2$.

$$H_0(x) = [1 - 2(x - x_0)L_0'(x_0)]\, L_0^2(x) = 24x^5 - 68x^4 + 66x^3 - 23x^2 + 1,$$
$$H_1(x) = [1 - 2(x - x_1)L_1'(x_1)]\, L_1^2(x) = 16x^4 - 32x^3 + 16x^2,$$
$$H_2(x) = [1 - 2(x - x_2)L_2'(x_2)]\, L_2^2(x) = -24x^5 + 52x^4 - 34x^3 + 7x^2.$$

Ejemplo 6. Funciones de Bessel

4. Cálculo de $\hat{H}_i(x)$, $i = 0, 1, 2$.

$$\begin{aligned}
\hat{H}_0(x) &= (x - x_0)L_0^2(x) = 4x^5 - 12x^4 + 13x^3 - 6x^2 + x,\\
\hat{H}_1(x) &= (x - x_1)L_1^2(x) = 16x^5 - 40x^4 + 32x^3 - 8x^2,\\
\hat{H}_2(x) &= (x - x_2)L_2^2(x) = 4x^5 - 8x^4 + 5x^3 - x^2.
\end{aligned}$$

5. Obtención de $H_5(x)$

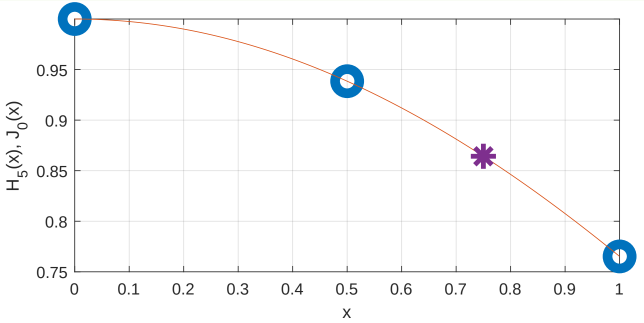
$$\begin{aligned}
H_5(x) = {} & J_0(x_0)H_0(x) + J_0(x_1)H_1(x) + J_0(x_2)H_2(x) + J_0'(x_0)\hat{H}_0(x)\\
& + J_0'(x_1)\hat{H}_1(x) + J_0'(x_2)\hat{H}_2(x)\\
= {} & -0.002x^5 + 0.0192x^4 - 0.0029x^3 - 0.2491x^2 + 1.
\end{aligned}$$

Ejemplo 6. Funciones de Bessel

$$H_5(x) = -0.002x^5 + 0.0192x^4 - 0.0029x^3 - 0.2491x^2 + 1.$$

Por tanto, el valor de $J_0(0.75)$ lo aproximamos por

$$J_0(0.75) \approx H_5(0.75) = 0.901461.$$



Contenidos

1 Introducción

2 Interpolación de Newton

3 Interpolación de Lagrange

4 Interpolación de Hermite
- Implementación computacional

5 Splines

Expresión en diferencias divididas del polinomio de Hermite de grado $2n+1$

$$H_{2n+1}(x) = f(z_0) + \sum_{k=1}^{2n+1} f[z_k, \ldots, z_0](x - z_0)(x - z_1) \cdots (x - z_{k-1}),$$

donde

$$z_{2i} = z_{2i+1} = x_i, \qquad f[z_{2i+1}, z_{2i}] = f'(x_i), \qquad i = 0, 1, \ldots, n.$$

Ejemplo 7. Polinomio de Hermite de grado 5 ($n = 2$)

- Puntos: x_0, x_1, x_2
- Datos conocidos: $f(x_0)$, $f'(x_0)$, $f(x_1)$, $f'(x_1)$, $f(x_2)$, $f'(x_2)$
- Expresión general: $H_5(x) = f(z_0) + \sum_{k=1}^{5} f[z_k, \ldots, z_0](x - z_0) \cdots (x - z_{k-1})$

$$\begin{aligned}
H_5(x) = f(z_0) &+ f[z_1, z_0](x - z_0) + f[z_2, z_1, z_0](x - z_0)(x - z_1) \\
&+ f[z_3, z_2, z_1, z_0](x - z_0)(x - z_1)(x - z_2) \\
&+ f[z_4, z_3, z_2, z_1, z_0](x - z_0)(x - z_1)(x - z_2)(x - z_3) \\
&+ f[z_5, z_4, z_3, z_2, z_1, z_0](x - z_0)(x - z_1)(x - z_2)(x - z_3)(x - z_4)
\end{aligned}$$

¿Puntos $z_0, z_1, \ldots, z_5$?

Ejemplo 7. Polinomio de Hermite de grado 5 ($n = 2$)

- Puntos: x_0, x_1, x_2
- Datos conocidos: $f(x_0)$, $f'(x_0)$, $f(x_1)$, $f'(x_1)$, $f(x_2)$, $f'(x_2)$
- $z_{2i+1} = z_{2i} = x_i$, $i = 0, 1, 2$:

$i = 0$: $z_1 = z_0 = x_0$
$i = 1$: $z_3 = z_2 = x_1$
$i = 2$: $z_5 = z_4 = x_2$

$$\underbrace{z_0, z_1,}_{x_0} \quad \underbrace{z_2, z_3,}_{x_1} \quad \underbrace{z_4, z_5}_{x_2} \quad \Rightarrow \quad \underbrace{f(z_0), f(z_1),}_{f(x_0)} \quad \underbrace{f(z_2), f(z_3),}_{f(x_1)} \quad \underbrace{f(z_4), f(z_5)}_{f(x_2)}$$

- $f[z_{2i+1}, z_{2i}] = f'(x_i)$, $i = 0, 1, 2$:

$$f[z_1, z_0] = f'(x_0), \qquad f[z_3, z_2] = f'(x_1), \qquad f[z_5, z_4] = f'(x_2)$$

$f(z_0) = f(x_0)$	$f[z_1, z_0] = f'(x_0)$	$f[z_2, z_1, z_0]$	$f[z_3, z_2, z_1, z_0]$
$f(z_1) = f(x_0)$	$f[z_2, z_1]$	$f[z_3, z_2, z_1]$	
$f(z_2) = f(x_1)$	$f[z_3, z_2] = f'(x_1)$		
$f(z_3) = f(x_1)$			

Interpolación de Hermite

Implementación computacional

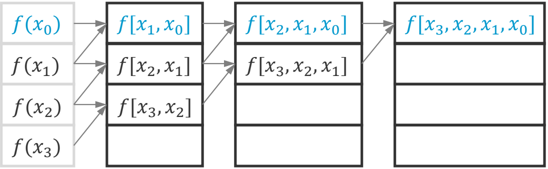


Figura: Generación de las diferencias divididas para el polinomio de **Newton**

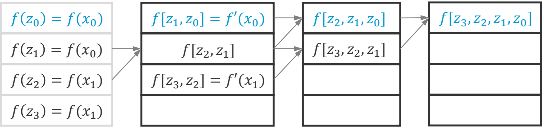
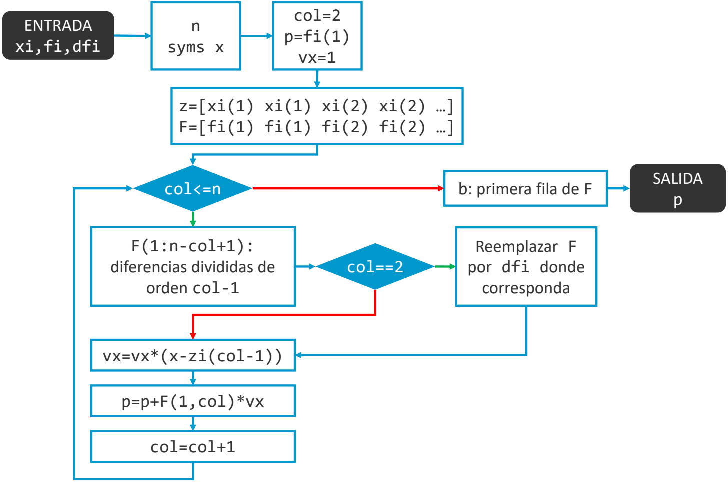


Figura: Generación de las diferencias divididas para el polinomio de **Hermite**



ENTRADA
xi,fi,dfi
n
syms x
col=2
p=fi(1)
vx=1
z=[xi(1) xi(1) xi(2) xi(2) …]
F=[fi(1) fi(1) fi(2) fi(2) …]
col<=n
b: primera fila de F
SALIDA
p
F(1:n-col+1):
diferencias divididas de
orden col-1
col==2
Reemplazar F
por dfi donde
corresponda
vx=vx*(x-zi(col-1))
p=p+F(1,col)*vx
col=col+1

polinomioHermite.m

```
function p = polinomioHermite(xi,fi,dfi)
xi=xi(:); fi=fi(:); dfi=dfi(:);
n=2*length(xi);
syms x
col=2; p=fi(1); vx=1
% generar vectores z y F duplicando xi y fi
...
  while col<=n
    % diferencias divididas de orden col-1
    ...
    if col==2
      ...
    end
    vx=...
    p=...
    col=col+1;
  end
end
```

5

Splines

Splines

- Hasta hora: polinomios que se ajustan a una serie de puntos o a una función
- Splines:
  - Utilizar un polinomio diferente para cada pareja de puntos consecutivos
  - → función a trozos en la que en cada trozo hay un polinomio

Splines

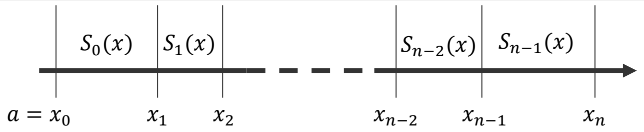
- Los splines más habituales son los cúbicos
- → Splines cúbicos naturales

Definición 1 (Splines cúbicos)

Sea f una función definida en $a = x_0 < x_1 < \cdots < x_{n-1} < x_n = b$, el **spline cúbico** $S(x)$ *es una función que cumple las siguientes condiciones:*

1. *Coincide con la función f en los puntos x_i, es decir, $S(x_i) = f(x_i)$ y $S(x_{i+1}) = f(x_{i+1})$, con $i = 0, 1, \ldots, n-1$.*
2. *Se define por intervalos, de modo que $S(x) = S_i(x)$, $x_i \leq x \leq x_{i+1}$, es decir:*

$$S(x) = \begin{cases} S_0(x), & x \in [x_0, x_1], \\ S_1(x), & x \in [x_1, x_2], \\ \vdots & \\ S_{n-1}(x), & x \in [x_{n-1}, x_n]. \end{cases}$$



Definición 1 (Splines cúbicos)

Sea f una función definida en $a = x_0 < x_1 < \cdots < x_{n-1} < x_n = b$, el **spline cúbico** *$S(x)$ es una función que cumple las siguientes condiciones:*

3. *Los splines en un intervalo coinciden en los puntos que comparten:*

$$S_{i+1}(x_{i+1}) = S_i(x_{i+1}), \quad i = 0, 1, \ldots, n-2.$$

4. *Las primeras derivadas de los splines en un intervalo coinciden en los puntos que comparten:*

$$S'_{i+1}(x_{i+1}) = S'_i(x_{i+1}), \quad i = 0, 1, \ldots, n-2.$$

5. *Las segundas derivadas de los splines en un intervalo coinciden en los puntos que comparten:*

$$S''_{i+1}(x_{i+1}) = S''_i(x_{i+1}), \quad i = 0, 1, \ldots, n-2.$$

6. *Las condiciones de contorno pueden ser*
   - *naturales, si $S''(x_0) = S''(x_n) = 0$, o*
   - *restringidas, por ejemplo $S'(x_0) = f'(x_0)$ y $S'(x_n) = f'(x_n)$.*

Expresión general de los splines cúbicos

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \quad i = 0, 1, \ldots, n - 1.$$

Cálculo de los coeficientes a_i, b_i, c_i, d_i $(i = 0, 1, \ldots, n - 1)$

Denotamos $h_i = x_{i+1} - x_i$. Aplicamos las condiciones de la Definición 1:

- $S(x_i) = f(x_i) \Leftrightarrow a_i = f(x_i)$
- $S_{i+1}(x_{i+1}) = S_i(x_{i+1}) \Leftrightarrow a_{i+1} = a_i + b_i h_i + c_i h_i^2 + d_i h_i^3$ **(1)**
- $S'_{i+1}(x_{i+1}) = S'_i(x_{i+1}) \Leftrightarrow b_{i+1} = b_i + 2c_i h_i + 3d_i h_i^2 \Leftrightarrow$ **(2)**
- $S''_{i+1}(x_{i+1}) = S''_i(x_{i+1}) \Leftrightarrow d_i = \frac{c_{i+1} - c_i}{3h_i}$ **(3)**

Reemplazando **(3)** en **(1)** y en **(2)** y operando obtenemos el sistema de ecuaciones:

$$h_{i-1}c_{i-1} + 2(h_{i-1} + h_i)c_i + h_i c_{i+1} = \frac{3}{h_i}(a_{i+1} - a_i) - \frac{3}{h_{i-1}}(a_i - a_{i-1}), \quad i = 1, 2, \ldots, n-1,$$

Contenidos

1. Introducción
2. Interpolación de Newton
3. Interpolación de Lagrange
4. Interpolación de Hermite
5. Splines
   - Splines cúbicos naturales

Expresión general y cálculo de coeficientes

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \qquad i = 0, 1, \dots, n-1,$$

con las condiciones de contorno naturales $S''(x_0) = S''(x_n) = 0$ y conococidos x_i y $f(x_i)$, $i = 0, 1, \dots, n$.

➔ Pasos:

1. $h_i = x_{i+1} - x_i$
2. $a_i = f(x_i)$
3. Cálculo de c_i para $i = 1, 2, \dots, n-1$:

$$h_{i-1}c_{i-1} + 2(h_{i-1} + h_i)c_i + h_i c_{i+1} = \frac{3}{h_i}(a_{i+1} - a_i) - \frac{3}{h_{i-1}}(a_i - a_{i-1}) \qquad (1)$$

4. De las condiciones naturales $S''(x_0) = S''(x_n) = 0 \Rightarrow c_0 = c_n = 0$
5. $b_i = \frac{1}{h_i}(a_{i+1} - a_i) - \frac{h_i}{3}(2c_i + c_{i+1})$
6. $d_i = \frac{c_{i+1} - c_i}{3h_i}$

Expresión matricial del sistema (1)

Desarrollando (1) para cada uno de los valores de i, junto con las condiciones naturales $S''(x_0) = S''(x_n) = 0$, obtenemos el sistema $Ax = b$, donde:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ h_0 & 2(h_0+h_1) & h_1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & h_1 & 2(h_1+h_2) & h_2 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix},$$

$$x = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix} \quad \text{y} \quad b = 3 \begin{bmatrix} 0 \\ \frac{a_2-a_1}{h_1} - \frac{a_1-a_0}{h_0} \\ \frac{a_3-a_2}{h_2} - \frac{a_2-a_1}{h_1} \\ \vdots \\ \frac{a_n-a_{n-1}}{h_{n-1}} - \frac{a_{n-1}-a_{n-2}}{h_{n-2}} \\ 0 \end{bmatrix}.$$

- Permite resolver sistemas lineales $Ax = b$, con A matriz tridiagonal, de manera óptima en cuanto al número de operaciones
- Se basa en la factorización de la matriz:

$$A = \begin{pmatrix} a_{11} & a_{12} & 0 & \cdots & 0 & 0 \\ a_{21} & a_{22} & a_{23} & \cdots & 0 & 0 \\ 0 & a_{32} & a_{33} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & a_{n-1n-1} & a_{n-1n} \\ 0 & 0 & 0 & \cdots & a_{nn-1} & a_{nn} \end{pmatrix}$$

en $A = LU$, donde

$$L = \begin{pmatrix} l_{11} & 0 & 0 & \cdots & 0 & 0 \\ l_{21} & l_{22} & 0 & \cdots & 0 & 0 \\ 0 & l_{32} & l_{33} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & l_{n-1n-1} & 0 \\ 0 & 0 & 0 & \cdots & l_{nn-1} & l_{nn} \end{pmatrix}, U = \begin{pmatrix} 1 & u_{12} & 0 & \cdots & 0 & 0 \\ 0 & 1 & u_{23} & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & u_{n-1n} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

- Igualando término a término los elementos de A y los de LU, obtenemos:
  - $i = 1$:

$$l_{11} = a_{11}, \qquad u_{12} = a_{12}/l_{11}$$

  - Para $i = 2, 3, \ldots, n-1$:

$$\begin{aligned} l_{ii-1} &= a_{ii-1}, \\ l_{ii} &= a_{ii} - l_{ii-1}u_{i-1i}, \\ u_{ii+1} &= a_{ii+1}/l_{ii}, \end{aligned}$$

  - $i = n$:

$$l_{nn-1} = a_{nn-1}, \qquad l_{nn} = a_{nn} - l_{nn-1}u_{n-1n}$$

Transformación del sistema

$$Ax = b \;\Leftrightarrow\; LUx = d \;\Leftrightarrow\; \left.\begin{array}{c} Lz = b \\ Ux = z \end{array}\right\}$$

$Lz = b$ ← Sustitución directa

$Ux = z$ ← Sustitución inversa

Crout.m

```
function x = Crout(dP,dS,dI,b)
n=length(dP);
% 1) l(1)=dP(1);
u(1)=dS(1)/l(1);
  for i=2:n-1
    l(i)=dP(i)-dI(i-1)*u(i-1);
    u(i)=dS(i)/l(i);
  end
l(n)=dP(n)-dI(n-1)*u(n-1);
% 2) z(1)=b(1)/l(1);
  for i=2:n
    z(i)=(1/l(i))*(b(i)-dI(i-1)*z(i-1));
  end
% 3) x(n)=z(n);
  for i=n-1:-1:1
    x(i)=z(i)-u(i)*x(i+1);
  end
x=x(:);
end
```

1) Obtención de L y U a partir de $A = LU$

2) Solución del sistema $Lz = b$

3) Solución del sistema $Ux = z$

Splines cúbicos naturales

Teorema 6

Sea f una función definida en los nodos $a = x_0 < x_1 < \cdots < x_{n-1} < x_n = b$, entonces f tiene un único spline S en los nodos, es decir, un spline que cumple $S''(a) = S''(b) = 0$.

Teorema 7 (Error en splines cúbicos naturales)

Sean $a = x_0 < x_1 < \ldots < x_n = b$, $f \in \mathcal{C}^2[a, b]$ y $h = \max_i\{h_i\}$. Entonces,

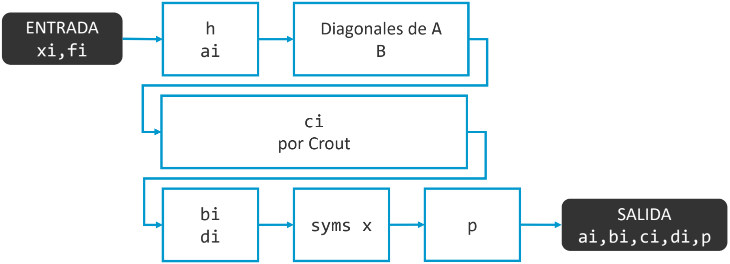
$$|\epsilon(x)| \leq h^{3/2}\sqrt{\left(\int_a^b [f''(x)]^2\, dx\right)}.$$

splineCubicoNatural.m

- **Entrada:** Los $n+1$ puntos conocidos $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$. Las coordenadas x_i de los puntos irán en un vector, y las coordenadas $f(x_i)$ en otro vector.
- **Salida:** El polinomio $p(x) = p_0 + p_1 x + p_2 x^2 + p_3 x^3$, los coeficientes a_i, b_i, c_i, d_i de

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \qquad i = 0, 1, \ldots, n-1,$$

```
function [ai,bi,ci,di,p] = splineCubicoNatural(xi,fi)
...
end
```



Ejemplo 8. Dados los datos del censo de población de España desde 1971, obtén los splines cúbicos naturales para estimar la población en el año 2005.

Ejecutamos sobre la consola:

```
>> xi = 1971:10:2011;
>> fi = [33.956 37.743 39.434 40.847 46.816];
>> [ai,bi,ci,di,p] = splineCubicoNatural(xi,fi);
```

El polinomio resultante será

$$S(x) = \begin{cases} S_0(x), & x \in [1971, 1981], \\ S_1(x), & x \in [1981, 1991], \\ S_2(x), & x \in [1991, 2001], \\ S_3(x), & x \in [2001, 2011], \end{cases}$$

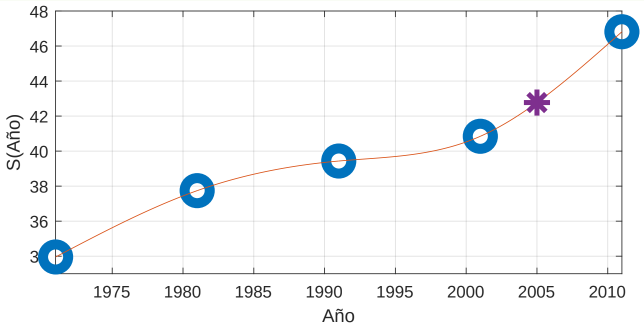
donde

$$\begin{aligned} S_0(x) &= 33.9560 + 0.4247(x-1971) - 0.0005(x-1971)^3, \\ S_1(x) &= 37.7430 + 0.2867(x-1981) - 0.0138(x-1981)^2 + \\ &\quad +0.0002(x-1981)^3, \\ S_2(x) &= 39.4340 + 0.0720(x-1991) - 0.0077(x-1991)^2 + \\ &\quad +0.0015(x-1991)^3, \\ S_3(x) &= 40.8470 + 0.3563(x-2001) + 0.0361(x-2001)^2 - \\ &\quad -0.0012(x-2001)^3. \end{aligned}$$

Ejemplo 8. Dados los datos del censo de población de España desde 1971, obtén los splines cúbicos naturales para estimar la población en el año 2005.

- Para conocer el valor del polinomio en el año 2005, buscamos el valor $S(2005) = S_3(2005)$, y ejecutamos:

```
>> p2005 = double(subs(p(4),x,2005))
p2005 = 42.7727
```



- Lecciones magistrales
- Material complementario: A fondo
- Bibliografía recomendada

...Y por supuesto:

TEST DE APRENDIZAJE!!

LA UNIVERSIDAD
EN INTERNET