Fórmulas para calcular los coeficientes de la interpolación segmentaria cúbica (splines cúbicos)

Requisitos. Una función `solvetridiag` que resuelve sistemas de ecuaciones tridiagonales, una función `countleq` que calcula el número de las componentes del arreglo dado que son menores o iguales que el número dado, una función `poleval` que calcula el valor del polinomio en el punto dado (o en un arreglo de puntos dados).

1. Notación para los datos iniciales. Sean $x_1 < x_2 < \ldots < x_n$ las abscisas dadas y sean $y_1, \ldots, y_n$ las ordenadas dadas. Luego denotemos por $h_1, \ldots, h_{n-1}$ las diferencias sucesivas de $x_1, \ldots, x_n$:

$$h_j = x_{j+1} - x_j.$$

Consideramos funciones segmentarias cúbicas de la forma

$$S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3 \qquad \forall x \in [x_j, x_{j+1}],$$

Se buscan listas de coeficientes $[a_j]_{j=1}^n$, $[b_j]_{j=1}^n$, $[c_j]_{j=1}^n$ y $[d_j]_{j=1}^n$, tales que la función S sea de clase $C^2([x_1, x_n])$ y tome los valores $y_1, \ldots, y_n$ en los puntos $x_1, \ldots, x_n$.

2. Sistema de ecuaciones lineales para $c_2, \ldots, c_{n-1}$. Para $n = 6$, se tiene $c_1 = c_6 = 0$, y los números $c_2, \ldots, c_5$ satisfacen el sistema

$$\begin{bmatrix} 2h_1 + 2h_2 & h_2 & 0 & 0 \\ h_2 & 2h_2 + 2h_3 & h_3 & 0 \\ 0 & h_3 & 2h_3 + 2h_4 & h_4 \\ 0 & 0 & h_4 & 2h_4 + 2h_5 \end{bmatrix} \begin{bmatrix} c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix} = \begin{bmatrix} \frac{3(y_3-y_2)}{h_2} - \frac{3(y_2-y_1)}{h_1} \\ \frac{3(y_4-y_3)}{h_3} - \frac{3(y_3-y_2)}{h_2} \\ \frac{3(y_5-y_4)}{h_4} - \frac{3(y_4-y_3)}{h_3} \\ \frac{3(y_6-y_5)}{h_5} - \frac{3(y_5-y_4)}{h_4} \end{bmatrix}. \tag{1}$$

3. Fórmulas para a_j, b_j y d_j. Se tiene $b_n = d_n = 0$, y para cada $j \in \{1, \ldots, n-1\}$,

$$b_j = \frac{1}{h_j}(y_{j+1} - y_j) - \frac{h_j}{3}(2c_j + c_{j+1}), \tag{2}$$

$$d_j = \frac{1}{3h_j}(c_{j+1} - c_j). \tag{3}$$

4. Diferencias sucesivas de los elementos de un arreglo. Dado un vector v, se pueden calcular las diferencias sucesivas de sus elementos de la siguiente manera:

```
v = [3; -5; 7; 1; 4]
diff(v)
v(2 : end) - v(1 : end - 1)
```

Si la longitud del arreglo v es n, entonces la longitud del arreglo `diff(v)` es $\underbrace{\qquad\qquad}_{?}$.

5. Programación. Escriba una función `constructcubicspline` con argumentos x, y que calcule los vectores de coeficientes $a, b, c, d \in \mathbb{R}^n$ y regrese los vectores x, a, b, c, d como una matriz o como una lista de listas. Use la función `solvetridiag` que resuelve sistemas de ecuaciones lineales tridiagonales.

```
function [S] = constructcubicspline(x, y),
   n = length(x); h = ???; ydiff = diff(y);
   diag0 = 2 * (h(??? : ???) + h(??? : ???));
   diag1 = h(??? : ???);
   q = 3 * ydiff ./ h;
   r = q(??? : ???) - q(??? : ???);
   c = solvetridiag(diag0, diag1, diag1, r);
   c = [0; c; 0];
   b = ??? ./ ??? - (??? / 3) .* (2 * c(1 : n - 1) + c(??? : ???));
   d = ???;
   b = [b; 0]; d = [d; 0];
   S = [x, y, b, c, d];
end
```

6. Programación. Escriba una función `evalcubicspline` con argumentos S, u que calcule los valores del interpolante segmentario cúbico natural en los puntos dados $u_1, \ldots, u_m$. Aquí S es una matriz o una lista de listas que consiste de los vectores x, a, b, c, d.

```
function [v] = evalcubicspline(S, u),
   m = length(u); v = zeros(???, 1); j = 1;
   for k = 1 : ???,
      j = countleq(S(:, 1), u(k), j);
      v(k) = poleval(S(???), ???);
   end
end
```

7. Prueba. Haga pruebas de las funciones programadas. Primero puede usar algún ejemplo pequeño:

```
function [] = testcubicspline(),
   x = [-4; -3; -1; 3; 5]; y = [2; 3; -1; -1; 2];
   S = constructcubicspline(x, y);
   u = linspace(min(x), max(x), 201);
   v = evalcubicspline(S, u);
   plot(u, v, 'k', x, y, 'o');
end
```

Luego puede inventar datos más interesantes:

```
   x = -5 + cumsum(0.5 + 2 * rand(7, 1)); y = mifuncionfavorita(x);
```

Por ejemplo, utilice la función y los nodos de la tarea del ajuste polinomial.