Programación: B-splines lineales

Objetivos. Programar el cálculo de B-splines lineales.

Requisitos. Función lineal (afín), fórmulas recursivas para B-splines.

1. Datos iniciales. En las siguientes funciones vamos a suponer que t es el vector de los nodos:

$$t = [t_1, \ldots, t_n]^\top, \qquad t_1 \le t_2 \le \ldots \le t_n,$$

y u es el vector de los puntos en los cuales queremos evaluar las funciones:

$$p = [u_1, \ldots, u_m]^\top.$$

2. Fórmulas para B-splines de grado 0. Para cada $j \in \{1, \ldots, n-1\}$,

$$B_{0,k}(u) = \begin{cases} 1, & u \in [t_k, t_{k+1}); \\ 0, & \text{en otro caso} \end{cases} = \chi_{[t_k,+\infty)}(u) - \chi_{[t_{k+1},+\infty)}(u).$$

3. Supongamos que

$$t = \begin{bmatrix} -2 \\ 3 \\ 4 \end{bmatrix}, \qquad u = \begin{bmatrix} 1 \\ 3 \end{bmatrix}.$$

Para cada j calculamos los valores de las funciones $\chi_{[t_j,+\infty)}$ en los puntos u_1 y u_2:

$$C_1(u) = \chi_{[t_1,+\infty)}(u) = \begin{bmatrix} 1 \ge -2? \\ 3 \ge -2? \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

$$C_2(u) = \chi_{[t_2,+\infty)}(u) = \begin{bmatrix} 1 \ge 3? \\ 3 \ge 3? \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

$$C_3(u) = \chi_{[t_3,+\infty)}(u) = \begin{bmatrix} 1 \ge 4? \\ 3 \ge 4? \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Restando estas columnas obtenemos los valores de las funciones $B_{0,1}$ y $B_{0,2}$ en los puntos u_1 y u_2:

$$B_{0,1}(u) = C_1(u) - C_2(u) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad B_{0,2}(u) = C_2(u) - C_3(u) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Notamos que la respuesta es correcta:

$$1 \in [-2, 3), \quad 3 \notin [-2, 3), \qquad 1 \notin [3, 4), \quad 3 \in [3, 4).$$

4. Comparación de un vector con un número. Ejecute los siguiente comandos en Matlab o GNU Octave:

```
u = [1; 3]
u >= -2
u >= 3
u >= 4
```

5. Cálculo de B-splines de grado 0. La función calcula y devuelve una matriz $V \in \mathcal{M}_{m\times(n-1)}(\mathbb{R})$, tal que la k-ésima columna de V contiene los valores de la función $B_{0,k}$ en los puntos $u_1, \ldots, u_m$.

```
function [V] = bsplines0(t, u),
   m = length(u); n = length(t);
   C = zeros(m, n);
   for k = 1 : n,
      C(:, k) = (u >= ???)
   end
   V = zeros(???, ???);
   for k = 1 : ???,
      V(:, k) = ??? - ???;
   end
end
```

6. Tarea adicional: eliminar los ciclos en el programa anterior. Usando operaciones matriciales adecuados elimine los ciclos `for` en el programa anterior. Pruebe el comando `uext >= text`, donde

$$\texttt{uext} = \begin{bmatrix} 1 & 1 & 1 \\ 3 & 3 & 3 \end{bmatrix}, \qquad \texttt{text} = \begin{bmatrix} -2 & 3 & 4 \\ -2 & 3 & 4 \end{bmatrix}.$$

Las matrices `uext` y `text` se puede construir de los vectores `p` y `t` usando la transposición y la función `matrep`. Luego con ejemplos descubra qué hace la función `diff` con una matriz dada.

7. Fórmulas para B-splines de grado 1. Para cada j pongamos

$$q_{1,j}(u) = \frac{u - t_j}{t_{j+1} - t_j},$$

$$B_{1,j}(u) = q_{1,j}(u)B_{0,j}(u) + (1 - q_{1,j+1}(u))B_{0,j+1}(u).$$

8. Cálculo de B-splines de grado 1.

```
function [V] = bsplines1(t, u),
   m = length(u); n = length(t);
   B0 = bsplines0(t, u);
   Q = zeros(???, ???);
   for j = 1 : ???,
      Q(:, j) = ???;
   end
   V = zeros(???, ???);
   for j = 1 : ???,
      V(:, j) = ???;
   end
end
```

9. Tarea adicional. Escribir la función anterior sin ciclos for, usando operaciones matriciales de nivel más alto.

10. Prueba.

```
function [] = testsplines1(),
   t = [-2; 1; 2; 6; 7];
   u = linspace(min(t), max(t), 40);
   V = bsplines1(t, u);
   plot(u, V, '*');
end
```