Ampliación de interpolación con Splines

Miguel Anguita Ruiz Pablo Baeyens Fernández Pablo David Medina Sánchez
Ruben Morales Pérez Francisco Javier Morales Piqueras

Splines cuadráticos

El espacio de splines de clase 2 se nota $S_2(x_1, x_2, ..., x_n)$. Una base sería $\{1, x, x^2, (x - x_1)_+, (x - x_2)_+, ..., (x - x_n)_+\}$

Descripción del espacio de splines cuadráticos

El espacio de splines de clase 2 con n nodos se denota $S_2(x_1, x_2, ..., x_n)$. Los splines de clase 2 están constituidos por parábolas, de forma que además de tener una función continua, su derivada también lo es. Por lo tanto, para $i = 1, ..., n - 1$ tenemos la siguiente condición:

$$\boxed{\begin{array}{l} s_i(x_i) = s_{i+1}(x_{i+1}) \\ s_i'(x_i) = s_{i+1}'(x_{i+1}) \end{array}}$$

Proposición 1 *El conjunto $S_2(x_1, x_2, ..., x_n)$ satisface las propiedades siguienes:*

1. *Es un espacio vectorial con* $\dim(S_2(x_1, x_2, ..., x_n))$

Ejemplos

Splines cúbicos

Construcción a partir de los valores de $s''(x)$ en los nodos $\{x_i\}$

Propiedades de minimización

Ejemplos

Implementación en ordenador: Octave

Hemos implementado las siguientes funciones en Octave:

0. `SplineLineal`: Calcula spline **lineal**. *(Usado en los splines cúbicos)*
1. `Spline31` : Calcula spline de **clase 1**.

2. `SplineNat`: Calcula spline **natural**.
3. `SplinePer`: Calcula spline **periódico**.
4. `SplineSuj`: Calcula spline **sujeto**.
5. `SplineCuad`: Calcula spline **cuadrático** de clase 1.

Spline Lineal

La función que nos permite calcular un spline lineal es muy

```
function s = SplineLineal(x,y)
  p = diff(y)./diff(x);
  A = [p' y(1:end-1)'];
  s = mkpp(x,A);
end
```

Splines cuadráticos

Utilizando el sistema que vimos anteriormente, podemos definir fácilmente una función que calcule los coeficientes de un spline cuadrático de clase 1:

```
function s = coefsSplineCuad(x, y, d_k, k)
  # Número de intervalos
  n = length(x) - 1;

  # 1, x, x²
  A(:,1) = [ones(n+1,1); 0];
  A(:,2) = [x'         ; 1];
  A(:,3) = [x'.^2      ; 2.*x(k+1)];

  # Potencias truncadas
  for j = 4 : n + 2
    pot    =  @(t) (t > x(j-2)) .* (t - x(j-2));
    A(:,j) = [pot(x').^2; 2.*pot(x(k+1))];
  end

  # Resolución del sistema
  s = A \ [y' ; d_k];

end
```

Definimos una función `pot` correspondiente a la potencia truncada en el valor correspondiente: `pot = @(t) (t > x(j-2)) .* (t - x(j-2))`. Como *Octave* tiene tipos dinámicos convertirá `(t > x(j-2))` a 1 o 0. De esta forma, $pot(x) = (x - x_{j-1})_+$.