UNIVERSIDAD DE MURCIA

Departamento de Matemáticas
Cálculo Numérico en una Variable
Curso 2023/24

Prof. S. Sánchez-Pedreño

Práctica 4

6 y 10 de noviembre de 2023

Para realizar esta práctica necesitas descargar algunos script de Octave que he subido al Aula Virtual. Si no lo has hecho ya, descarga el archivo `Practica4_AnadirBiblioteca.zip`, situado en la carpeta `CodigoPracticas` de los Recursos del AV. Descomprímelo y guarda su contenido en la carpeta `Biblioteca` que debe estar contenida en tu carpeta de trabajo con Octave de tu ordenador personal.

El objetivo de esta práctica es experimentar con la interpolación por splines. En el tercer ejercicio se aborda una caso particular de interpolación de Hermite a trozos y se estudian algunos detalles sobre splines y otros métodos de interpolación mediante curvas suaves que no hemos estudiado en el curso.

Recordemos que un spline cúbico es una función definida a trozos que interpola unos datos que conocemos: $\{(x_i, y_i) : i = 0, \ldots, n\}$. Para comprender bien las funciones de cálculo de los splines, en particular sus variables de entrada y salida, debes tener en cuenta que para estas funciones construidas en Octave, un spline se identifica con los nodos en los que interpola, $\{x_0, \ldots, x_n\}$, y los coeficientes de los distintos polinomios cúbicos que lo componen:

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

donde $p_i(x)$ es la pieza polinomial que forma el spline definida en $[x_i, x_{i+1}]$. Los coeficientes a_i, para $i = 0, \ldots, n-1$ coinciden con las ordenadas de los datos y_i, $i = 0, \ldots, n-1$, por lo que sólo se necesita calcular los b_i, c_i y d_i. Si el spline es sujeto necesitas proporcionar además los valores de la derivada en los puntos extremos x_0 y x_n. Ten en cuenta, finalmente, que para evaluar un spline en una abscisa se deben conocer los nodos, pues la evaluación necesita conocer en qué intervalo $[x_i, x_{i+1}]$ se encuentra dicha abscisa.

- `splineNatural.m`, devuelve los coeficientes b_i, c_i, d_i (en la forma `[b,c,d]`) de los polinomios cúbicos que componen el spline natural. Los argumentos que espera son: el vector de nodos(`x`) y el vector de ordenadas (`y`).
- `splineSujeto.m`, devuelve los coeficientes b_i, c_i, d_i (en la forma `[b,c,d]`) de los polinomios cúbicos que componen el spline sujeto. Se le suministran como variables, naturalmente, además de las coordenadas de los puntos que se interpolan (`x` e `y`), la derivada en los puntos inicial y final (`tan_ini` y `tan_fin`).
- `splineEval.m`, devuelve la evaluación de un spline en una abscisa (que puede ser un vector de abscisas). Necesita como argumentos: los nodos (`x`) y ordenadas (`y`), así como los vectores de coeficientes (`b`, `c`, `d`) y el vector de abscisas donde se evalúa (`t`). **Observación:** hemos elegido evaluar la función spline como el valor del polinomio $p_0(x)$ o $p_n(x)$ si $x < x_0$ o $x > x_n$ respectivamente. Otra opción podría ser definir el valor como cero en estos casos.
- `splineTabla.m`, es una función auxiliar, en general innecesaria, que únicamente produce una salida impresa de los polinomios que componen el spline: necesita por tanto todos los datos que lo definen (`x`, `y`, `b`, `c` y `d`).

Antes de utilizar cualquiera de estas funciones, asegúrate de entender qué argumentos espera, su tipo y significado, así como qué salidas produce, su significado, orden y formato. Importante: ¡los nodos deben pasarse ordenados!

1. *Ejemplos sencillos*

   a) Considera los puntos $\{(0,0),(1,2),(2,4),(5,10)\}$. ¿Qué función spline natural debes encontrar que interpole dichos puntos? Responde a la cuestión antes de verificarlo con un pequeño programa... Puedes utilizar la función `splineTabla.m`. Es una forma de verificar que el programa de cálculo de splines funciona correctamente.

   b) Responde a la misma pregunta anterior para un spline sujeto que interpola la colección de puntos $\{(0,0),(1,1),(2,8),(3,27)\}$ y tiende derivada 0 en $x=0$ y 27 en $x=3$.

   c) Considera la lista de puntos $\{(0,1),(3,0),(2,2),(1,4)\}$. Representa en un panel de dibujo el spline cúbico natural que interpola los puntos de la lista. ¿Serías capaz de dibujar de distinto color cada una de las tres piezas que componen el spline?

   d) Representa en otra ventana gráfica los splines cúbicos sujetos $S(x)$ que interpolan los mismos puntos y que tienen derivadas en los extremos $S'(0)=a, S'(3)=b$, para distintos valores de $(a,b)=(0,-1),(1,5),(-2,-5)$ y $(-5,-1)$.

2. *Curvas "suaves" en el plano obtenidas mediante el cálculo de splines: aproximar una espiral de Arquímedes mediante splines cúbicos*

   a) Consideramos una partición $\{\theta_i\}_{i=0}^{9}$ de 10 puntos equidistribuidos en el intervalo $[0,4\pi]$. Calculamos los 10 puntos correspondientes en la espiral arquimediana, que tiene por ecuación en coordenadas polares $\rho=2\theta$, con $\theta\in[0,4\pi]$. Así, las ecuaciones cartesianas de la espiral son:

   $$x(\theta)=2\theta\cos\theta,\quad y(\theta)=2\theta\,\mathrm{sen}\,\theta.$$

   b) Con las listas de puntos $\{(\theta_i,x(\theta_i))\}$ y $\{(\theta_i,y(\theta_i)\}$ calculamos sendos splines cúbicos naturales, que llamaremos $x(\theta)$, $y(\theta)$.

   c) Ahora representaremos en un panel gráfico la propia espiral arquimediana anterior y su aproximación por splines $(x(\theta),y(\theta))$.

   d) Realiza ahora la misma tarea con splines sujetos, añadiendo derivadas en los extremos: $x'(0)=2$, $x'(4\pi)=2$, $y'(0)=0$, $y'(4\pi)=8\pi$.

3. En primer lugar queremos programar una función que cree una función polinomial a trozos, en el que cada pieza sea un polinomio de Hermite. Concretamente, dada una sucesión de puntos $\{x_0,\ldots,x_m\}\subset[a,b]$, con $a\le x_0<x_1<\ldots<x_m\le b$, y una función $f:[a,b]\longrightarrow\mathbb{R}$ derivable, queremos definir una familia de polinomios $(p_i(x))_{i=0}^{m-1}$, de grado 3, de modo que el polinomio p_i, que consideramos definido en $[x_i,x_{i+1}]$ interpola los valores $\{f(x_i),f'(x_i),f(x_{i+1}),f'(x_{i+1})\}$. Así la función construida a trozos será de clase $\mathscr{C}^1$, aunque en general no será de clase $\mathscr{C}^2$. Para ello daremos los siguientes pasos.

   a) Construir en `/Biblioteca` una función `interpolTrozosHermite3.m` que reciba como variables tres vectores: `x`, `y`, `dy`. Donde `x` será el vector de los x_i, ¡ordenados de menor a mayor!, `y` será el vector de las imágenes $f(x_i)$, y finalmente `dy` será el vector de componentes $f'(x_i)$. La función debe devolver una matriz `Coef` de tamaño $m\times 4$, donde cada fila contiene los coeficientes del polinomio p_i en la forma de Newton. Para conseguir esto lo que haremos será llamar a la función `interpolHermite.m` desde dentro de nuestra función `interpolTrozosHermite3.m`.

   b) El siguiente paso será programar una función que evalúe la función polinomial a trozos generada por `interpolTrozosHermite3.m` en un vector de abscisas. Como se trata de una función a trozos, para evaluarla en un punto tendremos que ubicar el punto en uno de los trozos, para lo que se necesitan los nodos x_i. Así, programa una función `interpolTrozosHermite_eval.m` en `/Biblioteca` que reciba como variables la matriz de coeficientes de los polinomios `Coef`, el vector `x` de nodos y el vector `t` de abscisas donde queremos evaluar y devuelva el vector de evaluaciones. Podemos programarlo con un bucle o inspirarnos en la forma vectorial programada en `splineEval.m`.

Departamento de Matemáticas. Campus de Espinardo. 30100 Espinardo Murcia
e-mail: pedreno@um.es

c) Para experimentar las funciones anteriores creamos el script `Ejercicio3.m` en el que dibujaremos en una ventana la gráfica de la función $f(x) = \frac{1}{(1+e^{-8x})}$ sobre el intervalo $[-2, 2]$, el spline que interpola dicha función en 9 puntos equiespaciados y la interpolación de Hermite a trozos en los mismos puntos. Compara los resultados.

Ampliando horizontes... Observa el spline de la figura anterior: contiene algunas oscilaciones imprevistas y que pueden resultar indeseables. Nuestra función polinomial a trozos no tiene ese comportamiento, pero es imposible calcularla si no tenemos datos de las derivadas, por tanto, resulta imposible a partir de una colección de datos (x_i, y_i). Octave (y Matlab) contienen una función, denominada `pchip`, que, a partir de una colección de datos de ese tipo, calcula una función de clase $\mathscr{C}^1$ que evita las oscilaciones como las que pueden presentar los splines: son funciones que interpolan los datos respetando la monotonía, es decir, si de un dato al siguiente hay crecimiento, la función será creciente en el intervalo entre ambos nodos, ¡y no necesitan las derivadas! El algoritmo que implementa esta función fue publicado por F.N. Fritsch y R.E. Carlson en 1980 en un artículo que lleva por título *Monotone piecewise cubic interpolation*.

Crea en el script `Ejercicio3.m` anterior una nueva ventana donde representar la gráfica de la misma función $f(x)$, nuestra función polinomial a trozos y el resultado de la función `pchip` en los mismos datos que dicha función a trozos. La sintaxis `pchip(x,y,t)` devuelve el vector de evaluaciones en las abscisas `t` de la función que engendra el algoritmo a partir de los datos `(x,y)`.

Aún hay otras formas de conseguir curvas suaves y sin oscilaciones o protuberancias indeseadas a partir de datos (x_i, y_i). Uno de ellos está basado en la teoría de los *splines de tensión* o *tensor splines*. La idea es conseguir la forma de una hipotética cuerda que pasara por unos agujeros colocados en los puntos (x_i, y_i) y sometida a una cierta tensión... Pero esto es otra historia.