PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA DE CIÊNCIAS EXATAS E DA COMPUTAÇÃO

VITOR DE ALMEIDA SILVA

AED 4- Spline cúbico

CLARIMAR JOSE COELHO

GOIÂNIA,
2018

VITOR DE ALMEIDA SILVA

AED 4- Spline cúbico

Relatório apresentado como requisito parcial para obtenção de nota na disciplina Fundamentos 4 (quatro) no Curso de Engenharia da computação, na Pontifícia Universidade Católica de Goiás.

Clarimar Jose Coelho

GOIÂNIA,
2018

Introdução

O método consiste em se fazer uma aproximação das curvas de uma função delimitadas em intervalos, para tanto se usa algumas teorias de derivadas.

As curvas de terceiro grau usadas para conectar cada par de pontos dados são chamadas de splines cúbicos. Essas funções podem ser construídas de modo que as conexões entre equações cúbicas adjacentes sejam visualmente lisas.

Apresentação rápida do método de splines

Para se realizar os splines cúbicos em uma função é preciso saber, claro, os pontos da mesma que serão examinados. Para os splines também é preciso saber os valores das derivadas segundas aplicadas em alguns pontos, tais valores podem ser obtidos através de operações matriciais baseadas nas formulas que serão apresentadas.

O objetivo é determinar um polinômio de grau 3 para cada intervalo entre os nós, como em:

$$f_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i$$

Logo, para n + 1 pontos dados (i = 0, 1, 2, . . . , n), existem n intervalos e, consequentemente, 4n constantes indeterminadas para calcular. No fim, 4n condições são necessárias para calcular as incógnitas, São elas:

1. Os valores da função e dos polinômios adjacentes devem ser iguais nos nós interiores (2n − 2 condições).

2. A primeira e a última função devem passar pelos pontos extremos (2 condições).

3. As primeiras derivadas nos nós interiores devem ser iguais (n − 1 condições).

4. As segundas derivadas nos nós interiores devem ser iguais (n − 1 condições).

5. As segundas derivadas nos nós extremos são nulas (2 condições).

Após se realizar algumas análises matemáticas baseadas nas condições anteriormente listada, se pode chegar nas seguintes formulas decisivas para os splines cúbicos:

$$f_i(x) = \frac{f_i''(x_{i-1})}{6(x_i - x_{i-1})}(x_i - x)^3 + \frac{f_i''(x_i)}{6(x_i - x_{i-1})}(x - x_{i-1})^3 + \left[\frac{f(x_{i-1})}{x_i - x_{i-1}} - \frac{f''(x_{i-1})(x_i - x_{i-1})}{6}\right](x_i - x) + \left[\frac{f(x_i)}{x_i - x_{i-1}} - \frac{f''(x_i)(x_i - x_{i-1})}{6}\right](x - x_{i-1})$$

EQ.1

Está é a forma padrão do spline, repare que se pode separar em coeficientes as operações, é oque foi feito no código.

$$(x_i - x_{i-1})f''(x_{i-1}) + 2(x_{i+1} - x_{i-1})f''(x_i) + (x_{i+1} - x_i)f''(x_{i+1}) = \frac{6}{x_{i+1} - x_i}[f(x_{i+1}) - f(x_i)] + \frac{6}{x_i - x_{i-1}}[f(x_{i-1}) - f(x_i)]$$

EQ.2

Essa segunda formula é usada para se determinar os valores das segundas derivadas e parte do princípio de que as mesmas são contínuas, ou seja, S1``(x1)=S2``(x2). No código também foi aplicado o princípio 5, as segundas derivadas nos extremos são nulas (s(0)=0 e s(n)=0). Por fim olhando para porção que fica após a igualdade, vemos que o mesmo pode ser determinado, assim como os coeficientes ao lado das derivadas, isso resulta em um tipo de sistema, que pode ser resolvido para se obter as derivadas segundas.

Resolvendo essas equações para os pontos dados, e substituindo seus valores, pode-se obter os splines cúbicos.

Características gerais do spline cubico

- Computacionalmente falando, tau método é rápido para ser executado, ainda mais por se tratar de operações com matrizes e vetores onde muitos elementos podem ser considerados nulos.
- Os splines cúbicos retornam uma boa aproximação dos intervalos dos pontos examinados, ele também é mais suave visualmente que os quadráticos ou lineares.
- Pode ser usado para aplicações do mundo real, e ter seus erros relativos e absolutos calculados, dando bons resultados para serem usados em pesquisas ou estudos de casos.

Aplicação do método para a resolução do problema proposto

- Seja a função

$$f(x) = \begin{cases} e^x & -2 \leq x \leq 0, \\ x\sin(5x) + 1, & 0 \leq x \leq 4 \end{cases}$$

- Definida por n pontos distintos $(-2, f(-2))$, $(-2+h, f(-2+h))$, $(-2+2h, f(-2+2h)), \ldots, (4, f(4))$, com $h = (4-(-2))/(n-1)$
- Construa a curva de $y = f(x)$ usando *splines* cúbicos, com $m = 121$ número de pontos a interpolar e $n = 7$ e $n = 13$

Para a solução do problema, que consiste da aproximação da função pelos splines cúbicos, foi desenvolvido um código em octave. O código baseia-se nas equações 1 e 2 dadas anteriormente, cada elemento foi dividido e tratado separadamente.

Primeiro foi feito os passos para se obter as segundas derivadas doo sistema:

```
h=0;
k=0;
  for i=2: n-1
        h(i-1,i-1)=(x(i)-x(i-1));
        h(i-1,i)=(x(i+1)-x(i-1));
        h(i-1,i+1)=(x(i+1)-x(i));
        k(i)=(6/(x(i+1)-x(i)))*(y(i+1)-y(i))+(6/(x(i)-x(i-1)))*(y(i-1)-y(i));
  endfor
  k(1)=0;
  k(n)=0;
  s=0;
  s=k/h;  %matriz que contem o valor das segundas derivadas
```

A função 1 foi separada em coeficientes, e os mesmos determinados pelas contas equivalentes:

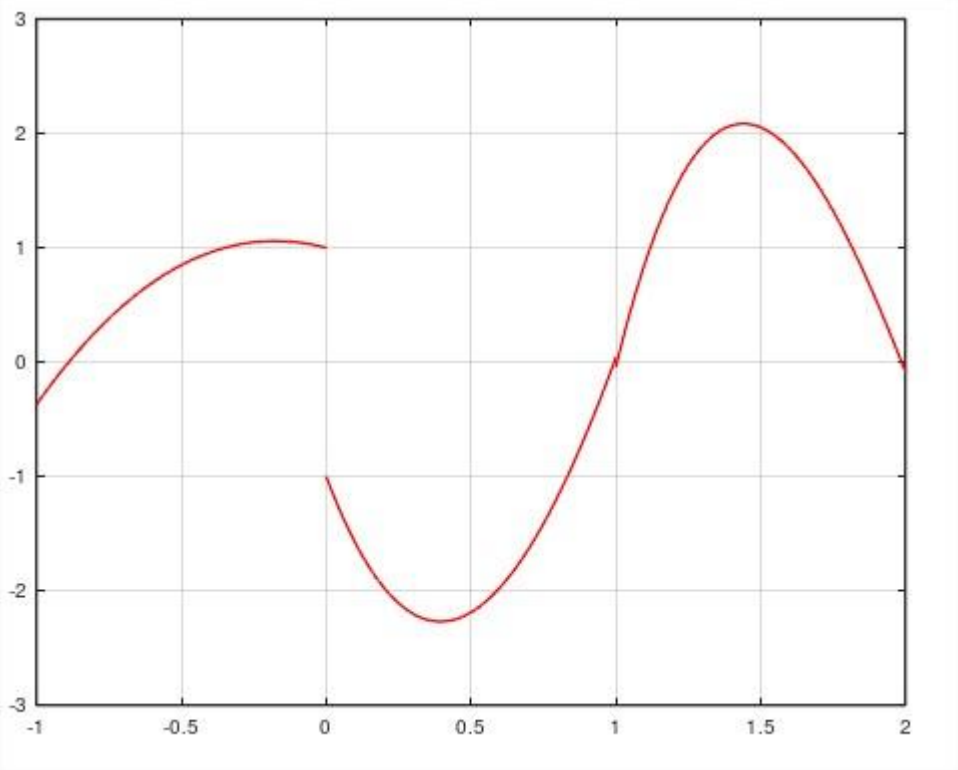
```
    a=b=c=d=0;
    a(n-1)=b(n-1)=c(n-1)=d(n-1)=0;
  for i=2: n-2
    a(i)=s(i-1)/(6*(x(i)-x(i-1)));
    b(i)=s(i)/(6*(x(i)-x(i-1)));
    c(i)=(y(i-1)/(x(i)-x(i-1)))-((s(i-1)*(x(i)-x(i-1)))/6);
    d(i)=(y(i)/(x(i)-x(i-1)))-((s(i)*(x(i)-x(i-1)))/6);
  endfor
```

Por fim, foi resolvido o sistema da equação 1, e plotado o gráfico:

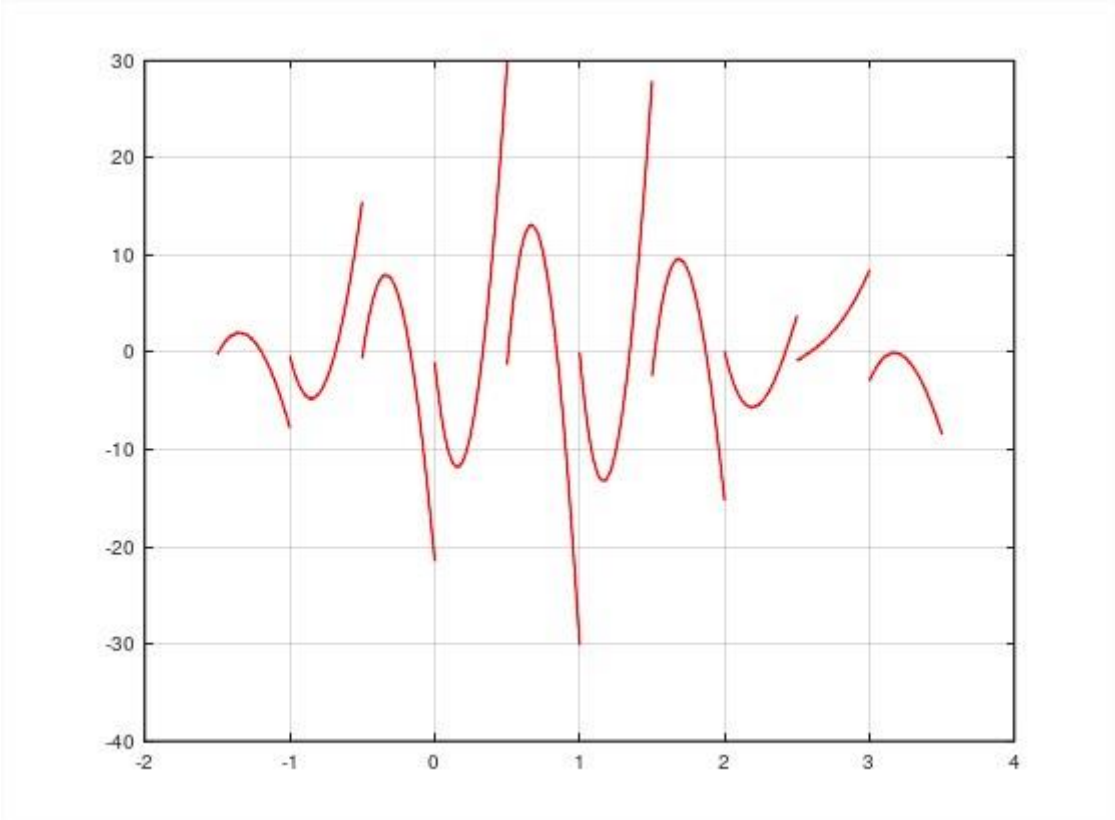
```
z=0;
t=0;
L=0;
    for i=2: n-2
      t= x(i-1): 0.01: x(i);

      z=a(i)*(t-x(i)).^3+b(i)*(t-x(i-1)).^2+c(i)*(t-x(i))+d(i)*(t-x(i-1));
      L=F2a0(x(i));
      ploty_h= plot(t,z,'-r');
      grid on
      hold on
      ploty_h= plot(t,L,'-b');
      grid on
      hold on
    endfor
     x(n)=0;
     z(n)=0;
```

De acordo com o exercício foi pedido dois gráficos, são eles:

- Para N=7:



- Para N=13:



Ocorreu alguns problemas na hora de se realizar o experimento e o gráfico não resultou em uma aproximação muito fiel, mas dá para se ter uma ideia do que se trata os splines cúbico.

Anexo

Código em octave:

```
    %% Inserção de dados

    n=input("Insira o indice do ultimo elemento(ex: x0,x1,x2...,xN):\n"); %grau do spline

    x=0; %vetor de pontos x
    y=0; %vetor de pontos y
    h=0; %passo da spline

    PontH=6/(n-1);
    %preenchendo os vetores de pontos
    for i=1: n
            x(i)=-2+i*PontH;
            y(i)=F2a0(x(i));
    endfor

% determinando os coeficientes
mem=1;
h=0;
k=0;
  for i=2: n-1
        h(i-1,i-1)=(x(i)-x(i-1));
        h(i-1,i)=(x(i+1)-x(i-1));
        h(i-1,i+1)=(x(i+1)-x(i));
        k(i)=(6/(x(i+1)-x(i)))*(y(i+1)-y(i))+(6/(x(i)-x(i-1)))*(y(i-1)-y(i));
  endfor
  k(1)=0;
  k(n)=0;
  s=0;
  s=k/h;  %matriz que contem o valor das segundas derivadas

    a=b=c=d=0;
    a(n-1)=b(n-1)=c(n-1)=d(n-1)=0;
  for i=2: n-2
    a(i)=s(i-1)/(6*(x(i)-x(i-1)));
    b(i)=s(i)/(6*(x(i)-x(i-1)));
    c(i)=(y(i-1)/(x(i)-x(i-1)))-((s(i-1)*(x(i)-x(i-1)))/6);
    d(i)=(y(i)/(x(i)-x(i-1)))-((s(i)*(x(i)-x(i-1)))/6);
  endfor

z=0;
t=0;
L=0;
    for i=2: n-2
      t= x(i-1): 0.01: x(i);

      z=a(i)*(t-x(i)).^3+b(i)*(t-x(i-1)).^2+c(i)*(t-x(i))+d(i)*(t-x(i-1));
```

```
    L=F2a0(x(i));
    ploty_h= plot(t,z,'-r');
    grid on
    hold on
    ploty_h= plot(t,L,'-b');
    grid on
    hold on
  endfor
   x(n)=0;
   z(n)=0;
```

Referências Bibliográficas

[1] CHAPRA, Steven C.; CANALE, Raymond P.. **Métodos Numéricos para Engenharia.** 5. ed. São Paulo: Amgh Editora Ltda, 2008. 825 p.