Bogdan Chwaliński
grupa Poniedziałek

Opis:
Wprowadzenie na podstawie książki
Numerical Mathematics and Computing
Autorzy E. Cheney,David Kincaid

Splajn kubiczny to funkcja, która jest klasy C^2 na odcinka [a,b] i dla danego podziału tego odcinka a = x0 < x1 … < xn = b na podocinki. Te funkcje obciete do każdego pododcinka [xi, xi+1] są wielomianami kubicznymi.
Zadanie interpolacji splajnami kubicznymi polega na znalezieniu splajnu kubicznego S spełniającego:

$$S(x) = \begin{cases} S_0(x) & (t_0 \le x \le t_1) \\ S_1(x) & (t_1 \le x \le t_2) \\ \vdots & \vdots \\ S_{n-1}(x) & (t_{n-1} \le x \le t_n) \end{cases}$$

gdzie t(n) to nasze punkty x, dla zadanych wartości y(n).

Dodatkowo trzeba dodać dwa warunki na s – warunki brzegowe, tzn. związane z wartościami s, pierwszych lub drugich pochodnych s w końcach odcinka.

W naszym programie funkcja Spline3_coef to algorytm który rozwiązuje trójdiagonalną macierz. Bierze wartości z tabeli (ti,yi) z wygenerowanych tablic i oblicza wektor zi który przechowuje wartości drugiej pochodnej.

Następnie funkcja Spline3_eval oblicza równianie

$$S_i(x) = y_i + (x - t_i)\left(B_i + (x - t_i)\left(\frac{z_i}{2} + \frac{1}{6h_i}(x - t_i)(z_{i+1} - z_i)\right)\right)$$

dla zadanych punktów.

Tabela wygenerowanych danych: (wygenerowany wektor y znajduje się poniżej w kodzie)

x	0	1	2	3	4	5	6	7	8	9
y	0.2760	0.6797	0.6551	0.1626	0.1190	0.4984	0.9597	0.3404	0.5853	0.2238

Kod programu przygotowany w Matlabie:

```
Funkcja Spline3_coef
function [z] = spline3_coef(n,t,y)

if (n <=2 )
   fprintf(1,'*** N musi być większe od 2! *** \n');
   return;
end;
nm1=n-1;
nm2=n-2;
h=zeros(nm1,1);
b=zeros(nm1,1);
```

```
u=zeros(nm2,1);
v=zeros(nm2,1);
z=zeros(n  ,1);
%
for i = 1:nm1,
    h(i)=t(i+1)-t(i);
    b(i)=(y(i+1)-y(i))/h(i);
end;
%
u(1)=2*(h(1)+h(2));
v(1)=6*(b(2)-b(1));
%
for i = 2:nm2,
    u(i)=2*( h(i+1)+h(i) )-( h(i)*h(i)/u(i-1) );
    v(i)=6*( b(i+1)-b(i) )-( h(i)*v(i-1)/u(i-1) );
end;
%
z(n)=0;
%
for i = nm1:-1:2,
    z(i) = ( v(i-1)-h(i)*z(i+1) )/u(i-1);
end;
z(1)=0;
%
return;

Funkcja Spline3_eval
function [val] = spline3_eval(n,t,y,z,x)
%
i=n-1;
while ( x < t(i) ),
   i=i-1;
end;
h= t(i+1)-t(i);
diff=x-t(i);
tmp = (z(i)/2) + diff*( z(i+1)-z(i) )/(6*h);
tmp = -(h/6)*( z(i+1)+2*z(i) ) + ( y(i+1)-y(i) )/h + diff*tmp;
val = y(i)+diff*tmp;
```

>> n=10; (10 punktów)
>> t = [0.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0]; // **nasze punkty x (10 punktów od 0 do 9)**
>> t=t';
>> ymin=0.0;
>> ymax=1.0;
>> y = ymin+rand(1,n)*(ymax-ymin) //**nasze wygenerowane wartości z zakresu [0,1]**
>> y=y';
>> y
y =
   0.2760
   0.6797
   0.6551
   0.1626

```
    0.1190
    0.4984
    0.9597
    0.3404
    0.5853
    0.2238
```
>> z=zeros(n,1); // **wektor pomocniczy który będzie przechowywał wartości drugich pochodnych**
>> x=zeros(n*2+1,1);
>> d=zeros(n*2+1,1);
>> z=spline3_coef(n,t,y); // **algorytm do znalezienia wartości drugich pochodnych**
>> h=0.1;
>> for i=1:n*10+1,
x(i)=0.0+(i-1)*h;
d(i)=spline3_eval(n,t,y,z,x(i));
end;
>> xmarkers = t;
>> ymarkers = y;
>> xlabel('x');
>> ylabel('y');
>> title('Splajn kubiczny');
>> plot(x,d,'b',xmarkers,ymarkers,'b*')
>> xlabel('x');
>> ylabel('y');
>> title('Splajn kubiczny');

Wygenerowany splajn kubiczny: