spline_1

October 30, 2020

In order to form our polynomial function properly i.e. to make sure that, it's applicable for Natural Cubic spline, we have to make sure that the 2nd derivatives of the function at breakpoints are zero. To do so, we do the following,

$$\int\int (x-1)(x-5)dx = (x^4)/12 - x^3 + 5x^2/2 + 2x + 2;$$

Where, we have chosen the arbirtary constant C=2.

[1]:
```
%%file cubic_main.m
function cubic_main(N)
%
% this is the main driver that calls out different functions to plot the
% spline
%
x = linspace(1,5,N);
y = prob(x);
n = length(x);
[a,b,c,d] = cbicv5(x',y');
M = 100; % for plotting points
xx = linspace(1,5,M);
yy = prob(xx);
figure(1);
plot(xx,yy,'g*');
hold on;
plot_cubic_spline(x,a,b,c,d,n);
title('f')
legend('True','Approx');
pos1 = get(gcf,'Position'); % get position of Figure(1)
set(gcf,'Position', pos1 - [pos1(3)/2,0,0,0]) % Shift position of Figure(1)
figure(2);
yy1 = proderi(xx);
plot(xx,yy1,'r*');
hold on;
plot_der_cubic_spline(x,b,c,d,N);
legend('True','Approx');
title('f-prime')
set(gcf,'Position', get(gcf,'Position'));
pos2 = get(gcf,'Position');  % get position of Figure(2)
```

```
set(gcf,'Position', pos2 + [pos1(3)/2,0,0,0]) % Shift position of Figure(2)
```

```
Created file 'C:\Users\aBr\Desktop\New folder\cubic_main.m'.
```

[2]:
```
%%file prob.m
function yy = prob(x)
%
% this script contains our function
%
    yy = (x.^4)/12 -x.^3 +5*x.^2/2 + 2*x +2;
end
```

```
Created file 'C:\Users\aBr\Desktop\New folder\prob.m'.
```

[3]:
```
%%file proderi.m
function yy = proderi(x)
%
% this is the f-prime
%
    yy=(x.^3)/3 -3*x.^2 +5*x + 2 ;
end
```

```
Created file 'C:\Users\aBr\Desktop\New folder\proderi.m'.
```

[4]:
```
%%file cbicv5
function [a,b,c,d]=cbicv5(x,y)
%
% this script generates the co-efficient of the cubic spline
%
n = length(x);

h = x(2:n) - x(1:n-1);
d = (y(2:n) - y(1:n-1))./h;

lower = h(1:end-1);
main  = 2*(h(1:end-1) + h(2:end));
upper = h(2:end);

T = spdiags([lower main upper], [-1 0 1], n-2, n-2);
rhs = 6*(d(2:end)-d(1:end-1));

r = T\rhs;

r = [ 0; r; 0];

a = y;
b = d - h.*(2*r(1:end-1) + r(2:end))/6;
```

```
c = r/2;
d =(r(2:end)-r(1:end-1))./(6*h);
```

```
Created file 'C:\Users\aBr\Desktop\New folder\cbicv5'.
```

[5]:
```
%%file plot_cubic_spline.m
function plot_cubic_spline(x,a,b,c,d,N)
%
% this script forms the spline using the co-efficient above for f
%
n = length(x);

for i=1:n-1
   xx = linspace(x(i),x(i+1),N);
   xi = repmat(x(i),1,N);
   yy = a(i) + b(i)*(xx-xi) + ...
     c(i)*(xx-xi).^2 + d(i)*(xx - xi).^3;
   plot(xx,yy,'b')
   plot(x(i),0,'r');
      Y= prob(xx);
end

error = norm(yy-Y,inf)
```

```
Created file 'C:\Users\aBr\Desktop\New folder\plot_cubic_spline.m'.
```

[6]:
```
%%file plot_dericubic_spline.m
function plot_dericubic_spline(x,b,c,d,N)
%
% this script forms the spline using the co-efficient above for f'
%
n = length(x);

for i=1:n-1
   xx = linspace(x(i),x(i+1),N);
   xi = repmat(x(i),1,N);
   yy = b(i) + 2*c(i)*(xx-xi) + 3*d(i)*(xx - xi).^2;
   plot(xx,yy,'m')
   plot(x(i),0,'r');
      Y= proderi(xx);
end

error = norm(yy-Y,inf)
```

```
Created file 'C:\Users\aBr\Desktop\New folder\plot_dericubic_spline.m'.
```

N	a) $\|f - s\|$	Convergence of Error of a	b) $\|f' - s'\|$	Convergence of Error of b
5	0.0126		0.0476	
10	5.1096e-04	4.6241	0.0042	3.5025
20	2.5665e-05	4.3153	4.4893e-04	3.2258
40	1.4489e-06	4.1468	5.1909e-05	3.1124

Choosing N=5