Interp Solutions

Public MySpline Solutions

Q1.

```
% function pwc = MySpline(x, y)
%
function pwc = MySpline(x, y)

   n = length(x);
   h = zeros(n-1,1);
   b = zeros(n-1,1);
   c = zeros(n-1,1);

   M = zeros(n,n);
   r = zeros(n,1);

   % Set first and last row of matrix and RHS vector
   M(1,1) = 1;
   r(1) = 0; % Natural BCs
   M(n,n) = 1;
   r(n) = 0; % Natural BCs

   % Calculate h's
   for k = 1:(n-1)
      h(k) = x(k+1) - x(k);
   end

   for row = 2:(n-1)

      k = row - 1;

      % Set k-th row of matrix
      M(row,row-1) = h(k)/6;
      M(row,row) = (h(k) + h(k+1)) / 3;
      M(row,row+1) = h(k+1)/6;

      % Set k-th row of right-hand side vector
      r(row) = (y(k+2)-y(k+1))/h(k+1) - (y(k+1)-y(k))/h(k);

   end

   % Solve system for a
   a = M \ r;

   % Compute b's and c's
   for k = 1:(n-1)
     b(k) = y(k)/h(k) - a(k)*h(k)/6;
     c(k) = y(k+1)/h(k) - a(k+1)*h(k)/6;
   end

   pwc.a = a;
   pwc.b = b;
   pwc.c = c;
```

Public AltSpline Solution

$$S(x) = \begin{cases} a + bx + cx^2 + dx^3 & 0 \le x < 1 \\ e + fx + gx^2 + hx^3 & 1 \le x < 4 \end{cases}$$

Passes through $(0, 2.5)$, $(1, 1.8)$ & $(4, -2)$

a) Interpolation Constraints

$$S_1(0) = a = 2.5 \quad (1)$$

$$S_1(1) = a + b + c + d = 1.8 \quad (2)$$

$$S_2(1) = e + f + g + h = 1.8 \quad (3)$$

$$S_2(4) = e + 4f + 16g + 64h = -2 \quad (4)$$

b) $S'(x)$ continuous at $x = 1$

$$S'(x) = \begin{cases} b + 2cx + 3dx^2 & 0 \le x < 1 \\ f + 2gx + 3hx^2 & 1 \le x \le 4 \end{cases}$$

$$b + 2c + 3d = f + 2g + 3h \quad (5)$$

c) $S'(0) = -1 \Rightarrow b = -1 \quad (6)$

$$S'(4) = 1 \Rightarrow f + 8g + 48h = 1 \quad (7)$$

d) Still need $S''(1)$ continuous

$$S''(x) = \begin{cases} 2c + 6dx & 0 \le x < 1 \\ 2g + 6hx & 1 \le x \le 4 \end{cases}$$

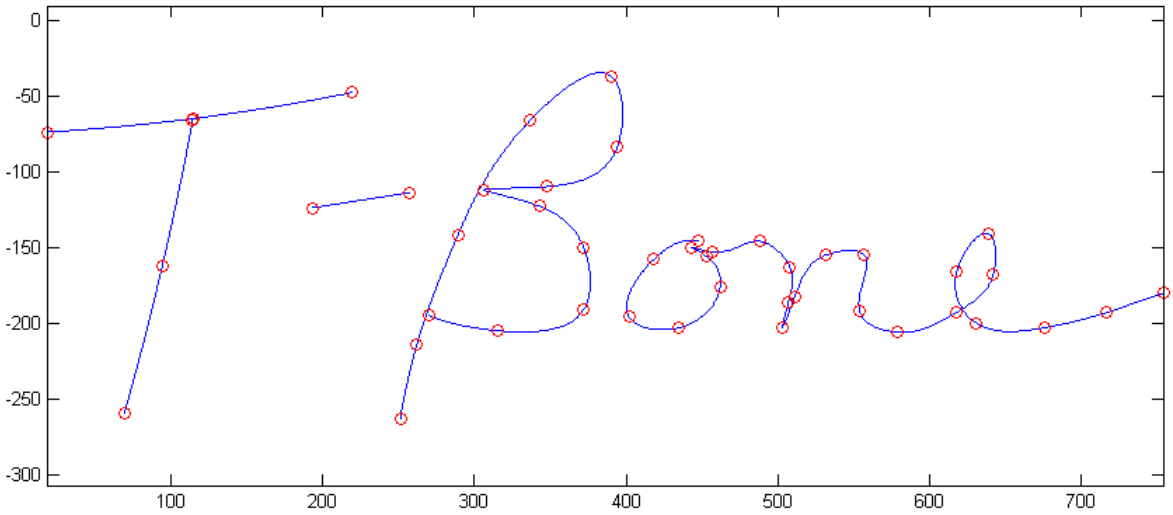
$$S_1''(1) = S_2''(1)$$

$$2c + 6d = 2g + 6h \quad (8)$$

Public Nickname Solutions

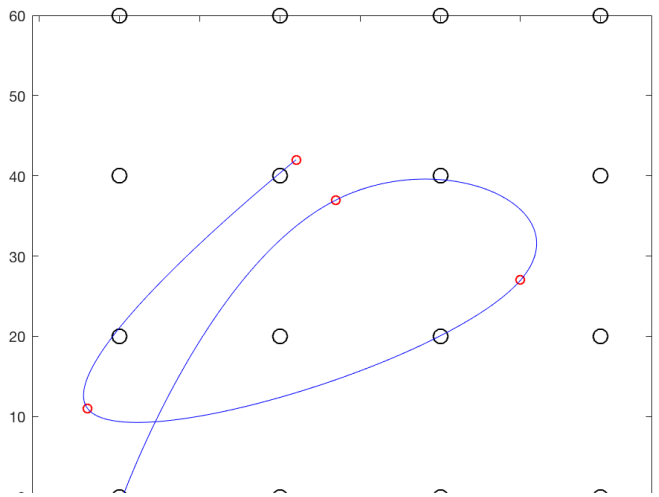
```
% Load data points

% Top of "T"
s1x = [18 115 219]';
s1y =-[74  65  48]';

% Trunk of "T"
s2x = [114  94  69]';
s2y =-[ 66 162 259]';

% "B" of "Bone"
s3x = [251 262 289 337 390 394 348 306 343 372 372 315 270]';
s3y =-[263 214 142  66  37  84 110 112 123 150 191 205 195]';

% "one" of "Bone"
s4x = [448 418 402 435 462 453 443 457 488 508 507 503 511 532 557 ...
     554 579 618 642 639 618 631 676 717 755]';
s4y =-[146 158 196 203 176 156 150 153 146 163 186 203 183 155 155 ...
     192 206 193 168 141 166 200 203 193 180]';

% Hyphen
s5x = [193 257]';
s5y =-[124 114]';


% Now get a cubic spline for each component (x and y)
% of each segment.
% Call ParametricSpline function
[x1_cs y1_cs t1] = ParametricSpline(s1x,s1y);
[x2_cs y2_cs t2] = ParametricSpline(s2x,s2y);
[x3_cs y3_cs t3] = ParametricSpline(s3x,s3y);
[x4_cs y4_cs t4] = ParametricSpline(s4x,s4y);
[x5_cs y5_cs t5] = ParametricSpline(s5x,s5y);


% Plot output
% Call ppval
% First I'll plot the splines
tt = linspace(t1(1),t1(length(t1)),1000);
xx = ppval(x1_cs,tt); yy = ppval(y1_cs,tt);
plot(xx,yy); hold on;
tt = linspace(t2(1),t2(length(t2)),1000);
xx = ppval(x2_cs,tt); yy = ppval(y2_cs,tt);
plot(xx,yy);
tt = linspace(t3(1),t3(length(t3)),1000);
xx = ppval(x3_cs,tt); yy = ppval(y3_cs,tt);
plot(xx,yy);
tt = linspace(t4(1),t4(length(t4)),1000);
xx = ppval(x4_cs,tt); yy = ppval(y4_cs,tt);
plot(xx,yy);
tt = linspace(t5(1),t5(length(t5)),1000);
xx = ppval(x5_cs,tt); yy = ppval(y5_cs,tt);
plot(xx,yy);


% Then I'll plot the points
plot(s1x,s1y,'ro');
plot(s2x,s2y,'ro');
plot(s3x,s3y,'ro');
plot(s4x,s4y,'ro');
plot(s5x,s5y,'ro');

axis equal; hold off;
```

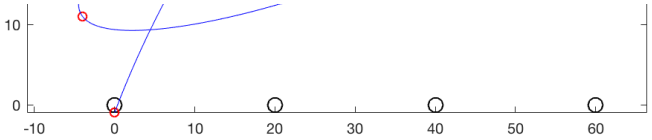
```
% function [x_cs, y_cs, t] = ParametricSpline(Sx,Sy)
%
%
function [x_cs, y_cs, t] = ParametricSpline(Sx,Sy)

   pts = length(Sx); % How many points?

   % Create parameter values
   % This code implements the pseudo-arclength (distance) formula
   % from equation (3.2) in the course notes.
   %{
   delta_Sx = -Sx + circshift(Sx,1); delta_Sx(1,1) = 0;
   delta_Sy = -Sy + circshift(Sy,1); delta_Sy(1,1) = 0;
   delta = sqrt(delta_Sx.^2 + delta_Sy.^2);
   t = toeplitz(ones(1,pts), [1 zeros(1,pts-1)]) * delta;
   %}

   % Pseudo-arclength could also be done using...
   t = zeros(1,pts);
   for n = 2:pts
      dist = sqrt( (Sx(n)-Sx(n-1))^2 + (Sy(n)-Sy(n-1))^2 );
      t(n) = t(n-1) + dist;
   end

   % Call spline functions on x and y
   x_cs = csape(t,Sx);  % or x_cs = spline(t,Sx);
   y_cs = csape(t,Sy);  % or y_cs = spline(t,Sy);
```


0
-50
-100
-150
-200
-250
-300
100
200
300
400
500
600
700

Public Unlock Pattern

```
% unlock.m script
% [4 marks total]

% Display grid of 16 circles
[gx, gy] = meshgrid([0 20 40 60], [0 20 40 60]);
plot(gx(:), gy(:),'ko', 'MarkerSize',10);
hold on; % don't erase plot on next plot command

% (a) -------------
% Load data
t = [0 795 1447 2172 2897];
x = [0 27 50 -4 22];
y = [-1 37 27 11 42];

plot(x, y, 'ro');

% Fit a cubic spline for each component (x and y).
% Call spline function
% Clamp slope to 0 at both ends
x_cs = spline(t, [0 x 0]);
y_cs = spline(t, [0 y 0]);


% (b) -------------
% Plot output
tt = linspace(t(1),t(end),1000);

xx = ppval(x_cs,tt);
yy = ppval(y_cs,tt);
plot(xx,yy,'b-');

axis equal; hold off;
```






```
axis equal; hold off;


% (c) ------------
% The unlock pattern is 1-11-7-5-10.
```