CS4210 Assignment 3 Due: 10/3/14 (Fri) at 6pm

You must work either on your own or with one partner. You may discuss background issues and general solution strategies with others, but the solutions you submit must be the work of just you (and your partner). If you work with a partner, you and your partner must first register as a group in CMS and then submit your work as a group. Problem 1 is worth 10 points and problems 2 and 3 are worth 5 points. Some test scripts will be placed on website by Monday Sept 29. Points may be deducted for poor style and reckless inefficiency.

Topics: Interpolation and approximation with piecewise cubic polynomials.

Introduction to B-Splines

This assignment is about cubic *B-splines* (a.k.a. *cubic basis splines.*) Just as it is possible to formulate a polynomial approximation problem using some chosen basis of polynomials, so too is it possible to formulate various cubic spline approximation/interpolation problems using a basis of cubic splines. Download `ShowBsplines` and `Bstar` and run the former to get an idea about what such a basis looks like when the breakpoints are equally spaced. Everything we say and do below can be carried out with arbitrary breakpoint placement and higher-order polynomial splines, but the messy details conceal the main idea.

The starting point is to define the piecewise cubic function $B_*(z)$ as follows:

$$B_*(z) \;=\; \begin{cases} 0 & z \le -2 \\ (2+z)^3/6 & -2 \le z \le -1 \\ (-3(1+z)^3 + 3(1+z)^2 + 3(1+z) + 1)/6 & -1 \le z \le 0 \\ (-3(1-z)^3 + 3(1-z)^2 + 3(1-z) + 1)/6 & 0 \le z \le 1 \\ (2-z)^3/6 & 1 \le z \le 2 \\ 0 & 2 \le z \end{cases}$$

It is easy to verify that this function is continuous, just check that things are cool at the key breakpoints: $B_*([\,-2\;\;-1\;\;0\;\;1\;\;2\,] = [\,0\;\;1\;\;4\;\;1\;\;0\,]/6$.

Likewise, the first derivative

$$B'_*(z) \;=\; \begin{cases} 0 & z \le -2 \\ (2+z)^2/2 & -2 \le z \le -1 \\ (-9(1+z)^2 + 6(1+z) + 3)/6 & -1 \le z \le 0 \\ (9(1-z)^2 - 6(1-z) - 3)/6 & 0 \le z \le 1 \\ -(2-z)^2/2 & 1 \le z \le 2 \\ 0 & 2 \le z \end{cases}$$

is continuous. Check this: $B'_*([\,-2\;\;-1\;\;0\;\;1\;\;2\,] = [\,0\;\;3\;\;0\;\;-3\;\;0\,]/6$. Finally, the second derivative

$$B''_*(z) \;=\; \begin{cases} 0 & z \le -2 \\ 2+z & -2 < z \le -1 \\ (-12-18z)/6 & -1 \le z \le 0 \\ (-12+18z)/6 & 0 \le z \le 1 \\ 2-z & 1 \le z \le 2 \\ 0 & 2 \le z \end{cases}$$

is continuous: $B''_*([\,-2\;\;-1\;\;0\;\;1\;\;2\,] = [\,0\;\;1\;\;-2\;\;1\;\;0\,]$. Bottom line: $B_*(\cdot)$ *is a cubic spline.*

Next, assume that $h > 0$ and $a \in \mathbb{R}$ are given and define $x_k = a + (k-1)h$ where k is *any* integer. Consider the function $B_k(\cdot)$ defined by

$$B_k(z) \;=\; B_*\left(\frac{z - x_k}{h}\right) \tag{1}$$

Convince yourself that this function is a cubic spline that is zero everywhere except on the interval $[x_{k-2}, x_{k+2}] = [x_k - 2h, x_k + 2h]$. Again, run the demo script `ShowBsplines` to lock in your intuition.

Continuing with $x_k = a + (k-1)h$, assume that we want to interpolate the given data points

$$(x_1, y_1), \ldots, (x_n, y_n)$$

with a cubic spline $s(z)$. Think along these lines:

```
x = linspace(a,b,n)';
h = (b-a)/(n-1);
y = MyF(x)}
```

Here is a big fact: the interpolating spline $s(z)$ can be represented as a sum of the basis splines

$$\{B_0(z), B_1(z), \ldots, B_n(z), B_{n+1}(z)\}$$

i.e.,

$$s(z) \;=\; \sum_{k=0}^{n+1} \alpha_k B_k(z). \tag{2}$$

To determine the α's we must augment the n linear equations

$$s(x_k) \;=\; y_k \qquad k = 1{:}n$$

with two more constraints so as to obtain a (nonsingular) $(n+2)$-by-$(n+2)$ linear system $A\alpha = b$. Canonical examples:

(i) Specify $s'(z)$ at the endpoints:

$$s'(x_1) \;=\; e_L \qquad s'(x_n) = e_R$$

(i) Specify $s''(z)$ at the endpoints:

$$s''(x_1) \;=\; e_L \qquad s''(x_n) = e_R$$

Setting $e_L = e_R = 0$ gives the *natural spline*.

(iii) Specify third derivative continuity at x_2 and x_{n-1}:

$$\lim_{z \uparrow x_2} s'''(z) \;=\; \lim_{z \downarrow x_2} s'''(z) \qquad \lim_{z \uparrow x_{n-1}} s'''(z) \;=\; \lim_{z \downarrow x_{n-1}} s'''(z)$$

This is the not-a-knot spline.

(iv) Specify that s looks the same at the endpoints:

$$s'(x_1) \;=\; s'(x_n) \qquad s''(x_1) = s''(x_n)$$

This is the periodic spline assuming that $y_1 = y_n$.

1 Interpolation with B-splines

Complete the following function so that it performs as specified

```
  function alpha = MySpline(a,b,n,y,type,e)
% a and b are scalars with a<b.
% n is a positive integer >=4.
% y is a column n-vector
% alpha is a column (n+2)-vector with the property that if
%
%      s(z) = alpha(1)B_{0}(z) + ... + alpha_{n+2}B_{n+1}(z)
%
% s the B-spline representation, then s(x(k)) = y(k) k=1:n.
%
% If type==1 then e must be a 2-vector and s'(x(1))  = e(1) and s'(x(n))  = e(2).
% If type==2 then e must be a 2-vector and s''(x(1)) = e(1) and s''(x(n)) = e(2).
% If type==3 then s'''(z) is continuous at z = x(2) and z = x(n-1).
% If type==4 then s'(x(1)) = s'(x(n)) and s''(x(1)) = s''(x(n)).
```

Use \ to solve the underlying linear system. Submit `MySpline` to CMS. You are expected to exploit structure in the set-up of the linear system.

2 Evaluation of the B-spline Representation

Complete the following function so that it performs as specified

```
  function yVals = BsplineEval(a,b,n,alpha,zVals)
% a and b are scalars with a<b.
% n is a positive integer >=4.
% alpha is a column (n+2)-vector,
% zVals is a column m-vector with a <= zVals(1) <= ... <= zVals(m) <= b
% yVals is a column m-vector with the property that if
%
%      s(z) = alpha(1)B_{0}(z) + ... + alpha_{n+2}B_{n+1}(z)
%
% then yVals(k) = s(zVals(k)), k=1:m
```

Score on this problem will depend upon the `tic-toc` efficiency of your implementation. You may assume that $m >> n$. Submit `BsplineEval` to CMS.

3 Least Squares fitting with B-Splines

Assume that we are given data points $(z_1, y_1), \ldots, (z_m, y_m)$ with

$$a = z_1 \leq \cdots \leq z_m = b$$

Let $n \geq 4$ be a positive integer and set $x_k = a + (k-1)h$ where $h = (b-a)/(n-1)$. Assume that $m >> n$. We wish to approximate the data with a cubic spline of the form (2) with the B_k defined by (1). In particular, our goal is to minimize the following function $\phi\colon \mathbb{R}^{n+2} \rightarrow \mathbb{R}$:

$$\phi(\alpha) \;=\; \sum_{i=1}^{m} \left[y_i - s(z_i)\right]^2 \;=\; \sum_{i=1}^{m} \left[y_i - \sum_{j=0}^{n+1} \alpha_j B_j(z_i) \right]^2$$

Setting the gradient of ϕ to zero we obtain a linear system $M\alpha = d$ that specifies the optimal $\alpha \in \mathbb{R}^{n+2}$. Subscripting matrix and vector entries from zero, here are the recipes for $M \in \mathbb{R}^{(n+2)\times(n+2)}$ and $d \in \mathbb{R}^{n+2}$:

$$M_{pq} = \sum_{k=1}^{m} B_p(z_k)B_q(z_k), \qquad p = 0{:}n+1,\ q = 0{:}n+1$$

$$d_p = \sum_{k=1}^{m} B_p(z_k)y_k, \qquad p = 0{:}n+1$$

Complete the following function so that it performs as specified:

```
  function alpha = BsplineLS(zVals,yVals,n)
% zVals and yVals are column m-vectors with a = zVals(1) < ... < zVals(m) = b
% n is a positive integer that satisfies 4<=n<<m. Let h = (b-a)/(n-1).
% alpha is a column (n+2)-vector with the property that if x(k) = a + (k-1)h
% and
%              s(z) = alpha(1)B_{0}(z) + ... + alpha_{n+2}B_{n+1}(z)
%
% with B_{k}(z) = Bstar((z-x(k))/h), then
%
%                 (s(zVals(1)) - yVals(1))^2 + ... + (s(zVals(m)) - yVals(m))^2
% is minimized.
```

Submit your implementation of `BsplinesLS` to CMS. To receive full credit, you must include a comment on the zero-nonzero structure of the M matrix and you must exploit that structure in its formation. Vectorize when possible. You may use \ to solve linear systems. It is not necessary to use the MATLAB `sparse` function although you are welcome to do so.