CS 322: Assignment P4

Due: Wednesday, April 10, 2002 (In Lecture)

You may work in pairs. Follow the course rules for the submission of assignments. Do not submit work unless you have adhered to the principles of academic integrity as descibed on the course website. Points will be deducted for poorly commented code, redundant computation that seriously effects efficiency, and failure to use features of MATLAB that are part of the course syllabus.

Part A. Band Cholesky (8 Points) If $A \in \mathbb{R}^{n\times n}$ is symmetric and positive definite then there exists a lower triangular matrix $G \in \mathbb{R}^{n\times n}$ such that $A = GG^T$. Here is one algorithm for computing G:

```
  function G = CholSax(A)
% G = CholSax(A)
% Cholesky factorization of a symmetric and positive definite matrix A.
% G is lower triangular so A = G*G'.

[n,n] = size(A);
G = zeros(n,n);
s = zeros(n,1);
for j=1:n
   s(j:n) = A(j:n,j);
   for k=1:j-1
      s(j:n) = s(j:n) - G(j:n,k)*G(j,k);
   end
   G(j:n,j) = s(j:n)/sqrt(s(j));
end
```

If A has upper (and lower) bandwidth equal to p, then it turns out that G has lower bandwidth p. Modify `CholSax` so that it exploits this property. In particular, implement the following function

```
  function G = CholSaxBand(A,p)
% G = CholSaxBand(A,p)
% Cholesky factorization of a symmetric and positive definite matrix A
% which has bandwidth p, 1<=p<=n-1.
% G is lower triangular so A = G*G'.
```

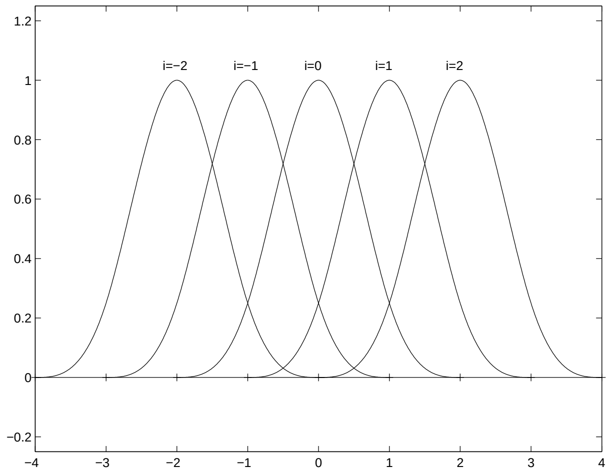
Whereas `CholSax` involves $n^3/3$ flops, `CholSaxBand` should involve just $O(p^2 n)$ flops if $p << n$. To obtain the reduced flopcount you need to "abbreviate" the vector operations that are part of `CholSax`. For example, when bandedness is exploited, the assignment `s(j:n) = A(j:n,j)` can be replaced by `s(j:r,n) = A(j:r,j)` where `r = min(j+p,n)`.

Test your implementation on the script `P4A` which is available on the website. Also on the website are `UTriSolBand` and `LTriSolBand`, the banded versions of `UTriSol` and `LTriSol` respectively. These are needed for Part B, but they may get you thinking the right way about band matrix computations.

Part B. Least Squares Fitting with Cubic Splines (12 Points) Suppose $h > 0$ and x_1 are given and define $x_j = x_1 + (j-1)h$ for all integers j. Define the piecewise cubic function $B_j(x)$ by

$$B_j(x) = \begin{cases} 0 & \text{if } x < x_{j-2} \\ \frac{1}{4}\left(\frac{x-x_{j-2}}{h}\right)^3 & \text{if } x_{j-2} \le x \le x_{j-1} \\ 1 - \frac{3}{2}\left(\frac{x-x_j}{h}\right)^2 - \frac{3}{4}\left(\frac{x-x_j}{h}\right)^3 & \text{if } x_{j-1} \le x \le x_j \\ 1 - \frac{3}{2}\left(\frac{x-x_j}{h}\right)^2 + \frac{3}{4}\left(\frac{x-x_j}{h}\right)^3 & \text{if } x_j \le x \le x_{j+1} \\ -\frac{1}{4}\left(\frac{x-x_{j+2}}{h}\right)^3 & \text{if } x_{j+1} \le x \le x_{j+2} \\ 0 & \text{if } x_{j+2} < x \end{cases}$$

Below is a depiction of $B_{-2}(x)$, $B_{-1}(x)$, $B_0(x)$, $B_1(x)$ and $B_2(x)$ for the case when $h = 1$ and $x_1 = 1$.



It is straightforward to show that these piecewise cubic functions are continuous and have a continuous first and second derivative. In other words, they are cubic splines. We refer to the B_j as *basis splines.* Here are some additional facts:

- Only four B_j are nonzero on the interval $[x_j, x_{j+1}]$, namely, B_{j-1}, B_j, B_{j+1}, and B_{j+2}.
- Any linear combination of the B_j is also a cubic spline. In particular,

$$\mathcal{B}(x) = \sum_{j=L}^{R} \alpha_j B_j(x)$$

  is continuous and has continuous first and second derivatives.

We are now set to show how to do least squares fitting with cubic splines. Suppose we are given data $(z_1, y_1), \ldots, (z_m, y_m)$ with

$$z_1 < z_2 < \cdots < z_m$$

Let n be an integer that satisfies $2 \le n \le m-2$ and set $h = (z_m - z_1)/(n-1)$ and $x_1 = z_1$. Our goal is to compute $\alpha_0, \alpha_1, \ldots, \alpha_n, \alpha_{n+1}$ so that

$$\phi(\alpha_0, \alpha_1, \ldots, \alpha_n, \alpha_{n+1}) \;=\; \sum_{i=1}^{m} \left(y_i - \sum_{j=0}^{n+1} \alpha_j B_j(z_i) \right)^2$$

is as small as possible. A standard manipulation shows that

$$\phi(\alpha_0, \alpha_1, \ldots, \alpha_n, \alpha_{n+1}) \;=\; \| \, F\alpha - y \, \|_2^2$$

where

$$F\alpha - y \;=\; \begin{bmatrix} B_0(z_1) & B_1(z_1) & B_2(z_1) & \cdots & B_{n+1}(z_1) \\ B_0(z_2) & B_1(z_2) & B_2(z_2) & \cdots & B_{n+1}(z_2) \\ B_0(z_3) & B_1(z_3) & B_2(z_3) & \cdots & B_{n+1}(z_3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ B_0(z_{m-1}) & B_1(z_{m-1}) & B_2(z_{m-1}) & \cdots & B_{n+1}(z_{m-1}) \\ B_0(z_m) & B_1(z_m) & B_2(z_m) & \cdots & B_{n+1}(z_m) \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{n+1} \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix}$$

The requirement $n \le m-2$ means that the F-matrix has at least as many rows as columns. Moreover, if z_i is in the interval $[x_j, x_{j+1}]$ then the only nonzero elements in $F(i,:)$ are $F_{i,j}$, $F_{i,j+1}$, $F_{i,j+2}$ and $F_{i,j+3}$.

Another observation is that $F(:,j_1)^T F(:,j_2) = 0$ if $|j_1 - j_2| > 4$. This is because the product of the functions B_{j_1-1} (associated with column j_1) and B_{j_2-1} (associated with column j_2) is the zero function. This means that if we solve the least squares problem via the normal equations $F^T F\alpha = F^T y$ then $A = F^T F$ has bandwidth $p = 4$.

Implement the function

```
  function alfa = LSSpline(z,y,n)
% z and y are column m-vectors and 2 <= n <= m-2.
% Assume z(1) < z(2) < ... < z(m).
% alfa is a column (n+2)-vector with the property that if
%
%                alfa' = [alpha(0),...,alpha(n+1)]
%
% then
%
%          alpha(0)B_{0}(x) + alpha_{1}B_{1}(x) + ... + alpha_{n+1}B_{n+1}(x)
%
% is the least squares fit of (z(1),y(1)),...,(z(m),y(m)). Here, B_{0},...,B_{n+1}
% are the basis splines associated with x(j) = a + (j-1)h, j=0:n+1, h = (z(m)-z(1))/(n-1).
```

Use the method of normal equations and make use of `CholSaxBand`, `LtriSolBand`, and `UTriSolBand`. (The latter two functions are available on the website.) Set up the F-matrix explicitly and to facilitate this write the following function

```
  function [f,j] = BEval(zVal,a,b,m,n)
% zVal is a scalar that satisfies a <= z <= b.
% m and n are integers that satisfy 2 <= n <= m-2.
% Let h = (b-a)/(n-1) and define x(k) = a + (k-1)h for any integer k.
% f is a row 4-vector and j is a positive integer such that
% x(j) <= zVal <= x(j+1) and f = [B_{j-1}(zVal) B_{j}(zVal)  B_{j+1}(zVal)  B_{j+2}(zVal) ],
% i.e., the values of the four nonzero basis splines at zVal.
```

A vector version of this (with `zVal` a vector) is possible but we'll skip that embellishment in this assignment.

Exploit the sparsity of F when you compute $A = F^TF$ and $c = F^Ty$ in `LSSpline`.

Download the file `SunSpotArea.txt` from the website (as in P2) and run the script `P4B`. It produces plots of various least squares fits of the sunspot data. Make sure `SunSpotArea.txt` is in whatever directory houses the `.m` files associated with this problem.