cs 514, lecture 05may–: spline interpolation (cont)

With $\tau = (\tau_1 < \cdots < \tau_n)$ in the interval $I = [a \,.\,.\, b]$ arbitrary, the data map

$$\Lambda'_\tau : f \mapsto (f(\tau_i) : i = 1{:}n)$$

is invertible on $F = \Pi_{<n}$ for any such τ. This cannot be true for $F = \$_{k,\mathbf{t}}$. For example, if some knot interval contains more than k of the τ_i, then surely we cannot match arbitrary values there since, on such an interval, each element of F is only a polynomial of degree $< k$.

The coefficient vector a of the interpolant $Pg = \sum_j a_j B_{jk}$, if any, must satisfy the linear system

$$A_\tau a = \Lambda'_\tau g,$$

with

$$A_\tau := \Lambda'_\tau [B_{j,k,\mathbf{t}} : j = 1{:}n] = (B_{j<k,\mathbf{t}}(\tau_i) : i, j = 1{:}n).$$

Hence it all depends on whether or not this Gramian matrix is invertible. Remarkably, there is a simple characterization for its invertibility.

Schoenberg-Whitney Theorem. *Let $\tau = (\tau_1 < \cdots < \tau_n)$ and $\mathbf{t} = (t_1 \le \cdots \le t_{n+k})$. Then the Gramian matrix A_τ is invertible if and only if all its diagonal entries are nonzero, i.e.,*

$$B_{i,k,\mathbf{t}}(\tau_i) \ne 0, \quad i = 1{:}n.$$

Moreover, when A_τ is invertible, then it is a banded matrix, with at most $2k + 1$ nontrivial bands (consecutive), hence its numerical solution is relatively cheap. More than that, the matrix is *totally positive* (whatever that may mean; look it up in the book), and this has as a consequence that the linear system can be solved by Gauss elimination *without any pivoting*, another savings in work and storage requirements. I hope you begin to appreciate just why B-splines are so wonderful.

There may be some doubt in the Schoenberg-Whitney Theorem in case τ_i equals some knot of multiplicity k, in which case $f \in \$_{k,\mathbf{t}}$ may be discontinuous at τ_i, hence has, in effect, *two* values at τ_i, namely the limit from the left and the limit from the right. In particular, by our way of choosing the knot sequence to work for a particular interval, the first and the last knot has that multiplicity. In such a case, you need to specify, more precisely, whether you want to match the limit from the left or the limit from the right. In particular, you would take the limit from the right at the first knot and the limit from the left at the last knot, and this is done automatically by the command `spapi`, which always takes the limit from the right except at the last knot where it takes the limit from the left.

So, how does the command `spapi(k,tau,g(tau))` choose the knot sequence for the interpolating spline of order k it outputs? It obtains the knot sequence as `aptknt(tau,k)`, and this, for $k > 1$, is simply the output from

```
augknt([tau(1),aveknt(tau,k),tau(end)] ,k)
```

meaning that it constructs the vector

$$\tau^* := (\tau^*_{jk} : j = 1{:}n{-}k),$$

and appends to it both the first and the last data site with multiplicity k. *Do verify that the resulting knot sequence does satisfy the above Schoenberg-Whitney conditions wrto τ.*

For example, if $k = 2$, then $\tau^* = (\tau_i : i = 2{:}n{-}1)$, hence the knot sequence used is

$$\mathbf{t} = (\tau_1, \tau_1, \tau_2, \tau_3, \ldots, \tau_{n-1}, \tau_n, \tau_n),$$

and the corresponding B-spline sequence is quite simply the sequence in $\$_{k,\mathbf{t}}$ dual to Λ'_τ, i.e., $B_{j,2,\mathbf{t}}(\tau_i) = \delta_{ij}$.

What happens when $k = 3$? Then

$$\tau^*_{j3} = (\tau_{j+1} + \tau_{j+2})/2, \quad j = 1{:}, n - 2,$$

hence the knots will lie between the data sites. In particular, the data sites will be at the midpoints between the knots if τ is uniformly spaced.

As another example, if $k = 4$ and τ is uniformly spaced, then $\tau^* = (\tau_3, \ldots, \tau_{n-2})$, hence the knot sequence used is

$$\mathbf{t} = (\tau_1, \tau_1, \tau_2, \tau_3, \ldots, \tau_{n-1}, \tau_n, \tau_n).$$

In particular, the resulting cubic spline interpolant has a break or knot at every data site except the second and the second-last and so satisfies what is known as the **not-a-knot end condition**.

Finally, if $k = n$, then τ^* is empty and, correspondingly, the knot sequence has both τ_1 and τ_n in it k times and nothing else. Correspondingly, $\$_{k,\mathbf{t}} = \Pi_{<n}$ (on the interval $[\tau_1 \,..\, \tau_n]$).

Next, we discuss **Hermite** or **osculatory** interpolation, as follows. You know that, for any continuously differentiable function,

$$\lim_{y \to x} \frac{g(y) - g(x)}{y - x} \; = \; Dg(x).$$

Hence, if all else is ok, we would expect our interpolant Pg to satisfy not only $(Pg)(\tau_i) = g(\tau_i)$ but also, in the limit as $\tau_{i-1} \to \tau_i$,

$$DPg(\tau_i) = Dg(\tau_i).$$

If also $\tau_{i-2} \to \tau_i$, then we would also expect, in the limit, to have

$$D^2 Pg(\tau_i) = D^2 g(\tau_i).$$

For this reason, it is reasonable to define the data map Λ'_τ even in the limiting case that τ is merely nondecreasing, i.e.,

$$\tau = (\tau_1 \le \cdots \le \tau_n),$$

in the following way:

$$\Lambda'_\tau : g \mapsto (D^r g(\tau_i) : \tau_{i-r-1} < \tau_{i-r} = \cdots = \tau_i; i = 1{:}n).$$

The command `spapi(k,tau,y)` interprets multiplicities in the data site sequence $\tau :=$ `tau` in exactly this way, i.e., interprets `y` as the information $\Lambda'_\tau g$ to be matched, with the following convenient feature. If the input `tau` is not already ordered, it will be re-ordered to make it nondecreasing and the input `y` will be re-ordered in the same way. Hence, if you would like to match the values of g at all points of a some sequence `x` without any multiplicities and also would like to match the first derivative at some subsequence `x1` of `x`, then the command

```
spapi(k, [x x1], [g(x) dg(x1)]);
```

would accomplish this (provided the desired order `k` is at least 3 (to make certain that the splines are continuously differentiable), and provided that `dg` provides values of the first derivative of g.

