Numerical Analysis II: approximation

Victor Eijkhout, Karl Schulz

Problem statement

Given a set of points (x_i, f_i) where $f_i = f(x_i)$ for some function f, we want to approximate f in some range.

Approximate by known functions: $F(x) = \Sigma_{j=1}^{m} c_j \phi_j(x)$
objective: $F(x)$ should be "close to $f(x)$"
(note: m can be different from n)

Pointwise approximation

From

$$f_i = F(x_i) = \Sigma_{j=1}^{n} c_j \phi_j(x_i) \qquad i = 1, \ldots, n$$

n equations with m unknowns:

$$M\underset{\sim}{c} = \underset{\sim}{f}, \quad M_{ij} = \phi_j(x_i)$$

- $n = m$: square matrix, can be solved if the ϕ_j linearly independent
- $n \neq m$: over or underdetermined system; for instance solve

$$M^t M\underset{\sim}{c} = M^t \underset{\sim}{f}$$

Approximation in norm

Minimize the error between f and F:

$$\min_{\underset{\sim}{c}} \frac{1}{2}\Sigma_{i=1}^{n}(F_i - f_i)^2 \quad \Rightarrow \quad \forall_{k=1,\ldots,m}\colon \frac{\partial}{\partial c_k}E = 0$$

$$\begin{aligned}
\frac{\partial}{\partial c_k}E &= \frac{\partial}{\partial c_k}\left(\frac{1}{2}\Sigma_{i=1}^{n}(F_i - f_i)^2\right) = \frac{1}{2}\Sigma_{i=1}^{n}\frac{\partial}{\partial c_k}(F_i - f_i)^2 = \frac{1}{2}\Sigma_{i=1}^{n}\left[2(\right. \\
&= \Sigma_{i=1}^{n}\left[(F_i - f_i)\frac{\partial}{\partial c_k}F_i\right] = \Sigma_{i=1}^{n}\left[(F_i - f_i)\left(\frac{\partial}{\partial c_k}\sigma_{j=1}^{m}c_j\phi_j(x_i)\right)\right. \\
&= \Sigma_{i=1}^{n}[(F_i - f_i)\phi_k(x_i)] = \Sigma_{i=1}^{n}\left[\left(\Sigma_{j=1}^{m}c_j\phi_j(x_i) - f_i\right)\phi_k(x_i)\right] \\
&= \Sigma_{i=1}^{n}\Sigma_{j=1}^{m}\phi_j(x_i)\phi_k(x_i) - \Sigma_{i=1}^{m}f_i\phi_k(x_i) = 0 \\
\Leftrightarrow\ & M^tM\underset{\sim}{c} = M^t\underset{\sim}{f}
\end{aligned}$$

Choice of ϕ_i functions

The ϕ_i functions need to be independent,
desirable to span ever larger polynomial spaces

Easiest choice $\phi_i(x) = x^i - 1$ gives matrix

$$\begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{m-1} \\ 1 & & & & x_2^{m-1} \\ \vdots & & & & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{m-1} \end{pmatrix}$$

Vandermonde matrix: very badly conditioned

Lagrange interpolation

Match basis functions to individual interpolation points: let $m = n$ and

$$\ell_j(x_i) = \begin{cases} 0 & j \neq i \\ 1 & j = i \end{cases}$$

Then $F(x) = \Sigma_{j=1}^{m} f_j \phi_j(x)$

Implementation:

$$\ell_j(x) = \frac{\Pi_{k\neq j}(x - x_k)}{\Pi_{k\neq j}(x_j - x_k)}$$

Equation $M\utilde{c} = \utilde{f}$ now diagonal matrix

Error of Lagrange interpolation

- If f (the function to be approximated) is a polynomial of degree $\leq m$, $F \equiv f$.
- If f is of higher degree:

$$(F - f)(x) = \frac{f^{(n)}(\xi)}{n!}(x - x_1)\cdots(x - x_n)$$

derivatives bounded $\Rightarrow$ error goes down

Newton's method for interpolation

Instead of direct construction, use induction

- Zeroth degree polynomial through one point (x_1, y_1):

$$p_1(x) = y_1$$

- First degree polymomial through two points:

$$p_2(x) = p_1(x) + c_2(x - x_1)$$

then automatically $p_2(x_1) = p_1(x_1) = y_1$. Furthermore

$$p_2(x_2) = p_1(x_2) + c_2(x_2 - x_1) \quad \Rightarrow \quad c_2 = \ldots$$

- General: suppose $p_n(x_i) = y_i$ for $i \leq n$; define

$$p_{n+1}(x) = p_n(x) + c_{n+1}(x - x_1) \cdots (x - x_n)$$

then $p_{n+1}(x_i) = p_n(x_i)$ for $i \leq n$, and from $p_{n+1}(x_{n+1}) = y_{n+1}$ we get c_{n+1}.

How does this relate to Lagrange interpolation?

Evaluation of newton polynomial

Naive evaluation: $\mathrm{cost}(p_{n+1}(x)) = 2n + \mathrm{cost}(p_n(x))$: quadratic in n; this should of course be linear.

- Write

$$
\begin{aligned}
p_3 &= p_2 + c_3(x-x_1)(x-x_2) \\
&= p_1 + c_2(x-x_1) + c_3(x-x_1)(x-x_2) \\
&= y_1 + (x-x_1)\big(c_2 + c_3(x-x_2)\big)
\end{aligned}
$$

- Similar:

$$
p_4 = y_1 + (x-x_1)\left(c_2 + (x-x_2)\big(c_3 + c_4(x-x_3)\big)\right)
$$

et cetera. Now the cost is linear.

Hermite interpolation

Same idea as Lagrange, but now also matching derivatives:

$$\forall_{i,j}\colon h_j(x_i) = 0; \qquad h'_j(x_i) = \begin{cases} 0 & j \neq i \\ 1 & j = i \end{cases}$$

Implementation:

$$h_j(x) = \alpha_j (x - x_j) \Pi_{k \neq j} (x - x_k)^2$$

Combination of Lagrange and Hermite interpolation for values and derivates

More interpolation

Certain choices of x_i lead to higher accuracy: "Gaus quadrature points"

Other norms lead to other basis functions, for instance

$$\min_F \int_0^1 w(x)(F(x) - f(x))^2 \, dx$$

Orthogonal polynomials: $\langle \phi_i, \phi_j \rangle = \delta_{ij}$.