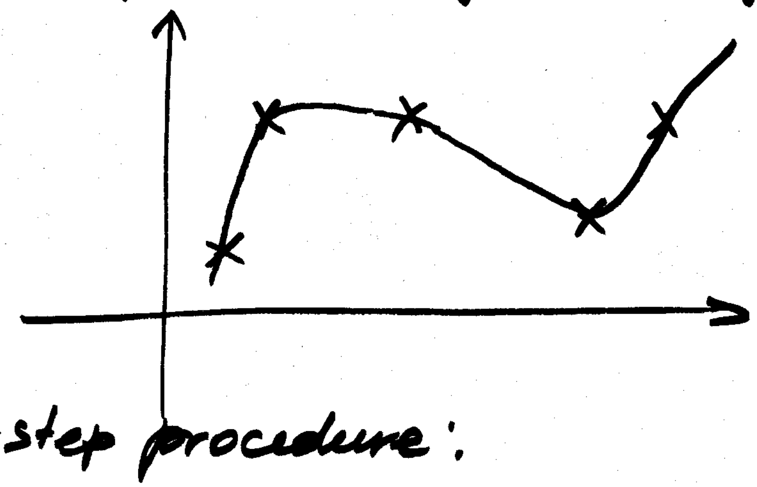
Math 685.
Lecture 6

Interpolation: (t_i, y_i) - data points

Interpolator: f s.t. $f(t_i) = y_i$



has to be easy to use

has to represent the data

2-step procedure:

1) Choose the space (type of interpolants)
   - → polynomial
   - → trig fcts
   - → exp, rational etc
2) Choose the basis

   Same space but different basis results in different numerical issues

Interpolation consists of:
- → construction of the interpolant
- → evaluation of interpolant at data pts

I. Monomial basis

$$A = \begin{bmatrix} 1 & t_1 & \cdots & t_1^{n-1} \\ \vdots & & & \\ 1 & t_n & \cdots & t_n^{n-1} \end{bmatrix}$$

Vandermonde

full matrix with bad conditioning

"+" easy to evaluate,
"+" easy to interpolate data, esp. if using Horner's rule to save on flops.

A is always nonsingular, since if $Az = 0$ then $z \equiv 0$

<u>Pf</u>: $p_z(t) = z_1 + z_2 t + \ldots + z_n t^{n-1}$

interpolant i.e. $p_z(t_i) = 0$ since $Az = 0$.

poly of deg $\leq n-1$ with n zeros $t_1 \ldots t_n$

<u>Fund. Thm of Algebra</u>:

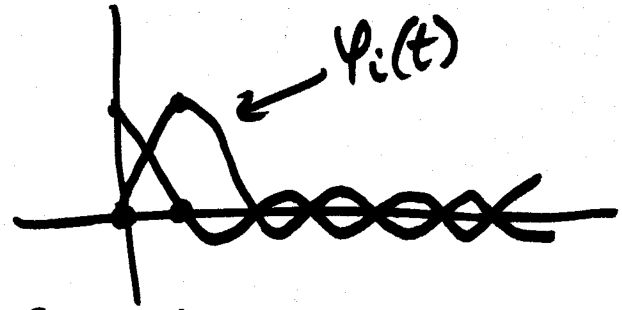
$\deg p = n \Rightarrow p$ has at most n zeros

So $z \equiv 0$, and $|A| \neq 0$.

II. Lagrange basis

$\varphi_i(t_i) = 1$

$\varphi_i(t_j) = 0, \ j \neq i$



$$A = \{\varphi_j(t_i)_{\substack{i=1\dots n \\ j=1\dots n}}\} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & & & \\ 0 & \dots & \dots & 1 \end{pmatrix}$$

<u>diagonal</u>

perfectly well conditioned

"−" comes at a high cost of calculating nodal basis,

"+" interpolating data with same $\{t_1 \dots t_n\}$ but different $\{y_1 \dots y_n\}$ is easy:

$$p(t) = \sum_{j=1}^{n} y_j \varphi_j(t)$$

Lagrange Basis fcts

"−" taking deriv. is hard

III. Newton Basis.

$p_1 = y_1$ ← interp. y_1 s.t. $p_1(t_1) = y_1$

$p_2 = p_1 + c(t - t_1)$ ← want to interp. y_1 & y_2 s.t.

$p_2(t_2) = p_1(t_2) + c(t_2 - t_1) = y_2$ $\quad$ $p(t_1) = y_1$ & $p(t_2) = y_2$

$c = \dfrac{y_2 - y_1}{t_2 - t_1}$ $\quad$ automatic

$p_3 = p_2 + c(t-t_1)(t-t_2)$

pick c s.t. $p_3(t_3) = y_3$

... etc

incremental Newton interpolation

$$A = \{\varphi_j(t_i)\} = \begin{bmatrix} 1 & 0 & \cdots & \cdots & 0 \\ 1 & t_2 - t_1 & & & \\ \vdots & & & & \\ 1 & t_n - t_1 & - & \cdots & (t_n-t_1)\cdots(t_n-t_{n-1}) \end{bmatrix}$$

lower triangular

"+" matrix $O(n^2)$ complexity to solve $Ax = y$.

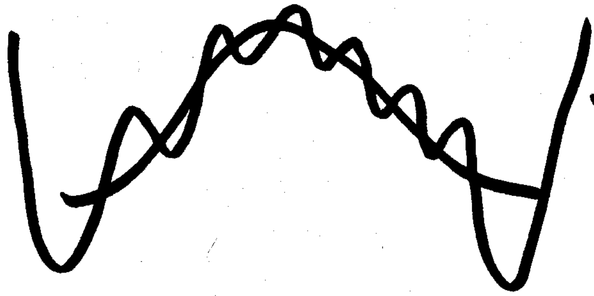
best of both worlds

"+" evaluating is not that hard

IV. Orthogonal Basis

→ Chebyshev → produces Chebyshev nodes: "optimal" placement for interp. error minimization

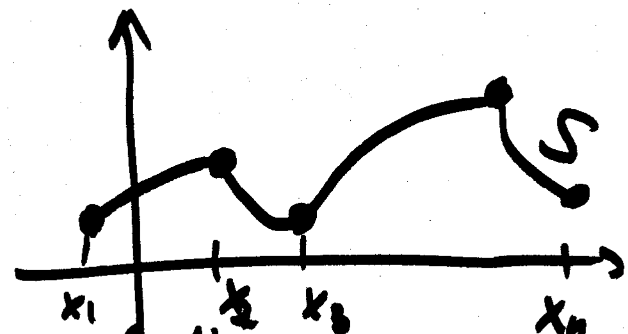
→ Hermite



← "wiggly" high-order phenomenon

Piecewise interpolation

1) Hermite polys
2) Cubic splines
3) B-splines



$S_+(x_i) = S_-(x_i)$ by interpolation

$$S_+(x_i) = \lim_{x \to x_i^+} S(x), \qquad S_-(x_i) = \lim_{x \to x_i^-} S(x)$$

$$S_+'(x_i) = S_-'(x_i)$$

$S_+''(x_i) = S_-''(x_i)$ – in addition to above

$$S(x) = \begin{cases} b_0^{(1)} + b_1^{(1)} x + b_2^{(1)} x^2 + b_3^{(1)} x^3, & [x_1, x_2] \\ \cdots\cdots & \\ b_0^{(n-1)} + b_1^{(n-1)} x + b_2^{(n-1)} x^2 + b_3^{(n-1)} x^3, & [x_{n-1}, x_n] \end{cases}$$

cubic spline or Hermite

$4(n-1)$ coeffs to determine

$2(n-1)$ interp. conditions

$+ \; n-2$ conds on 1st deriv. (interior pts)

$3n-4$ equations for Hermite (2 addit. constraints)

$+ \; n-2$ conds on 2nd deriv.

$4n-6$ eqns for cubic spline (2 addit. constraints)

natural spline $S''(x_1) = S''(x_n) = 0.$

B-spline: fix a different basis
in the space of splines