Piecewise Polynomial Interpolation/Approximation

A. Donev, Spring 2021

What we did before is
global polynomial approx.
One $p_n(x)$ on $[a,b]$

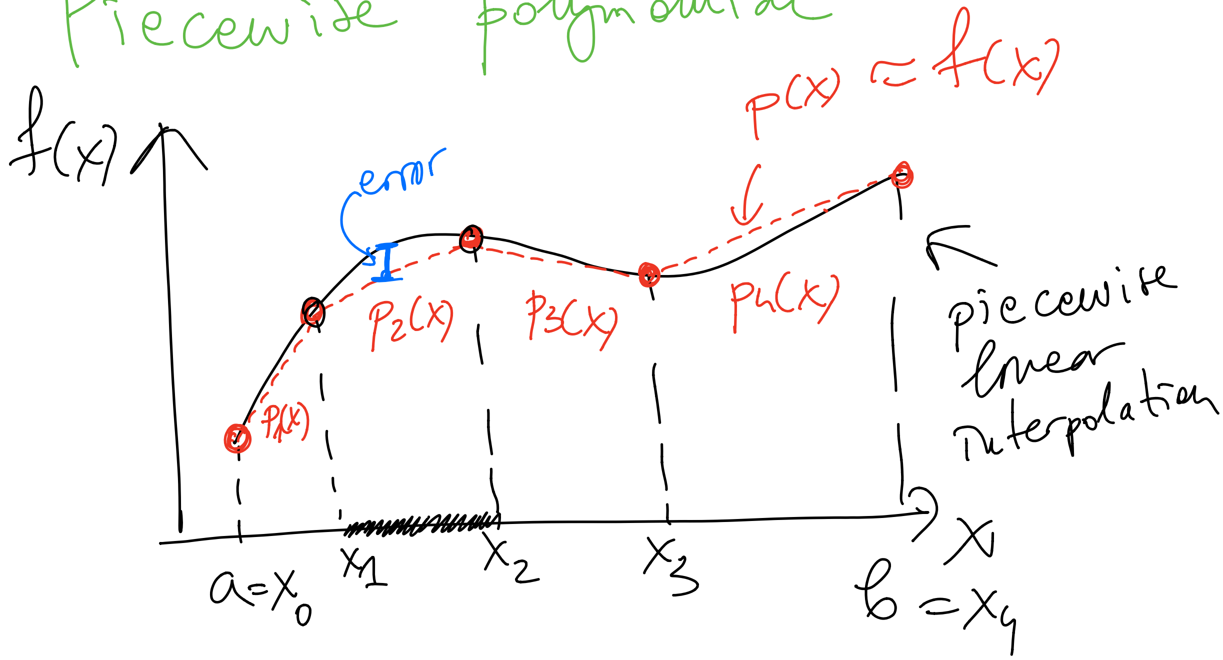
Doesn't work if:

1) We need to choose nodes flexibly

2) If the function is not sufficiently smooth

Z → True Type font

Piecewise polynomial



$f(x) \simeq p_k(x)$ on $[x_{k-1}, x_k]$

different poly in each interval

E.g. Lagrange form

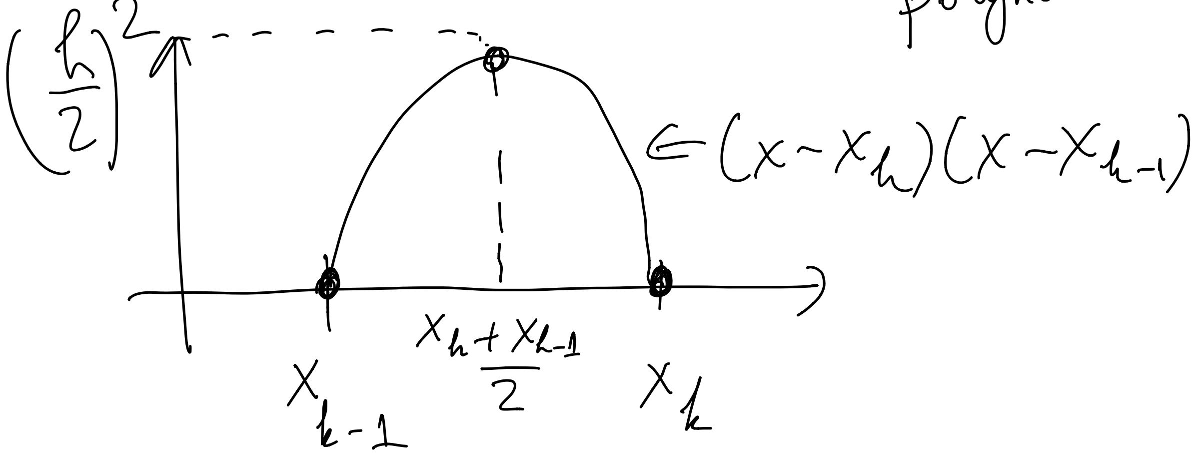
$$p_k(x) = f(x_{k-1}) \frac{x - x_k}{x_{k-1} - x_k} + f(x_k) \frac{x - x_{k-1}}{x_k - x_{k-1}}$$

Accuracy of pp approx?

$x \in [x_{k-1}, x_k]$

$$f(x) - p_k(x) = \frac{f''(\xi)}{2} \underbrace{\overbrace{(x-x_k)(x-x_{k-1})}^{\le \frac{h^2}{4}}}_{\text{Nodal polynomial}}$$

$\xi \in [x_{k-1}, x_k]$



h = grid spacing $= x_k - x_{k-1}$

$$|f(x) - p_k(x)| \le \frac{\max_{x_{k-1} \le x \le x_k} |f''(x)|}{8} h^2$$

$x \in [x_{k-1}, x_k]$

$$\text{Error} = O(h^2)$$ ← second-order accurate approximation

m points → $2m$ points

$h \to h/2$

error → error / 4 < always decreases

Guaranteed convergence

$$\text{error} \xrightarrow[m\to\infty]{} 0$$

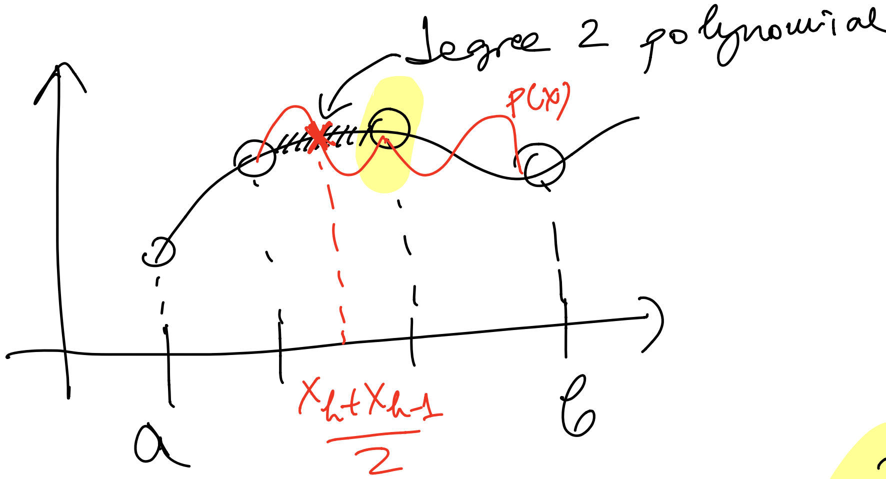
Piecewise approx is not very accurate (like global approx)

but it is robust

(works for any nodes & even less-smooth functions)

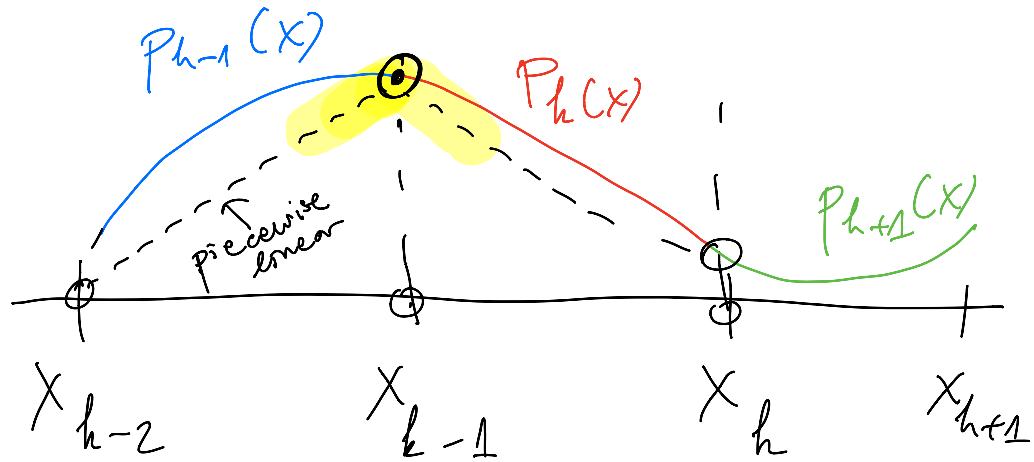
To improve accuracy, increase degree of poly

Piece wise quadratic



Then error $\sim \max|f'''(x)|\, h^3$

[We really want once, twice continuously differentiable approximations



We want

$\rightarrow$ $p'_{k+1}(x_k) = p'_k(x_k)$ ... (*)

(continuity of derivative)

$$\begin{cases} p_{k+1}(x_k) = f(x_k) \\ p_k(x_k) = f(x_k) \end{cases} \quad \dots (\square)$$

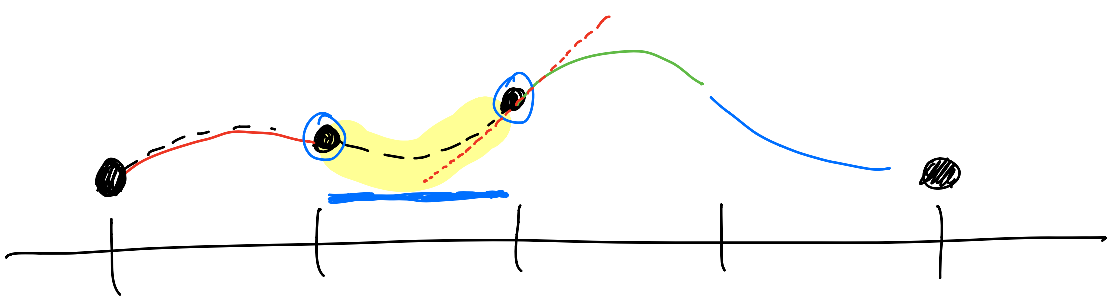
(interpolating condition)

cont. $p_k(x_k) = p_{k+1}(x_k) = f(x_k)$

We could also ask

$$z_k = p_k''(x_k) = p_{k+1}''(x_k) \quad \ldots (**)$$

Let's try piecewise quadratic



3n coefficients

We need 3n equations

From ($\square$) we get 2n equations

For (*) we have 1 eq. per interior node, $n+1-2 = n-1$ equations.

$3n$ coeff but $3n-1$ equations

In practice, we use piecewise cubic = spline interpolation

$4n$ coeff.

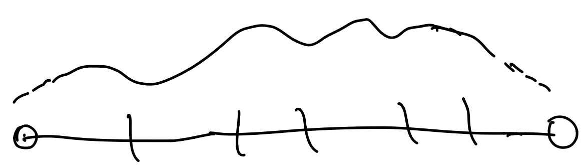
$D = 2n$

$* = n-1$

$(**) = n-1$

$4n-2$ equations

$$+ \quad P_1''(x_0) = P_n''(x_n) = 0$$

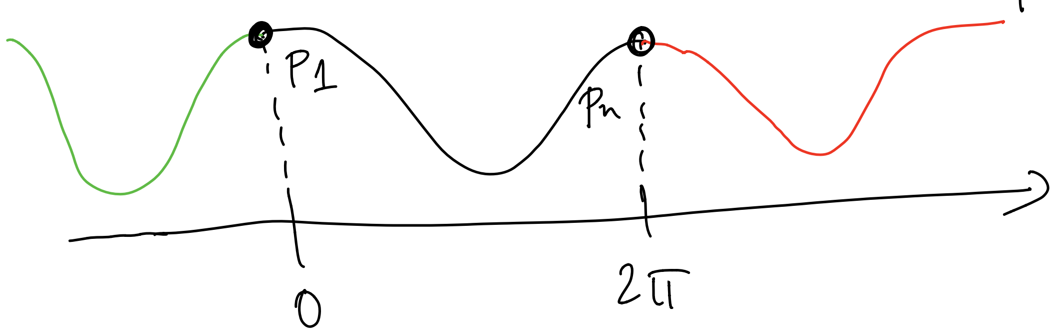


This uniquely defines a minimum curvature or natural spline interpolant

which is the C^2 (twice continuously differentiable) interpolant that has minimum total (approximate) curvature.

unique

$f(x)$ is periodic



$x_0 = x_n$

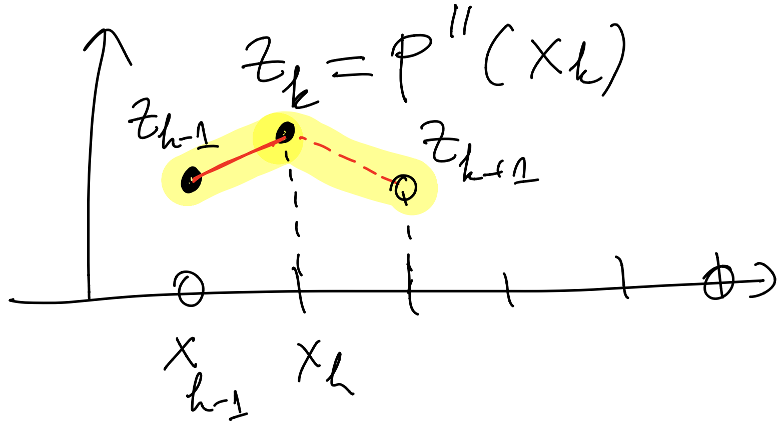
$$\left\{\begin{array}{l} P_1'(x_0) = P_n'(x_n) \\ P_1''(x_0) = P_n''(x_n) \end{array}\right\} \text{periodic Spline}$$

We want faster than n^3 FLOPS, and we want controlled round-off error.

Best case scenario is $O(n)$ operations ✓

$P_k(x)$, $P_k'(x)$ is quadratic, $P_k''(x)$ is linear and continuous

$p''(x)$



If we know $z_k = p''(x_k)$

then we know $p_k''(x)$

$$p_k''(x) = \underset{\substack{\| \\ p''(x_{k-1})}}{z_{k-1}} \frac{x - x_k}{x_{k-1} - x_k}$$

$x_{k-1} \le x \le x$

$$+ z_k \frac{x - x_{k-1}}{x_k - x_{k-1}}$$

Integrate twice

(assume $x_k - x_{k-1} = h$ = const)

$$p_k(x) = \frac{1}{h} z_{k-1} \frac{(x_k - x)^3}{6}$$

$$+ \frac{1}{h} z_k \frac{(x - x_{k-1})^3}{6}$$

$$+ C_k (x - x_{k-1}) + D_k$$

unknown integration constants

$(\square)$ at x_{k-1}:

$$p_k(x = x_{k-1}) = f(x_{k-1}) = f_{k-1}$$

$$\Rightarrow \frac{1}{h} \frac{h^3}{6} z_{k-1} + D_k = f_{k-1}$$

Determines D_k

(D) at $x = x_k$

$$\frac{h^3}{6} z_k + C_k h + D_k = f_k$$

Determines C_k

$$\begin{cases} C_k = \dfrac{1}{h}\left[(f_k - f_{k-1}) + \dfrac{h^2}{6}(z_{k-1} - z_k)\right] \\[2ex] D_k = f_{k-1} - \dfrac{h^2}{6} z_{k-1} \end{cases}$$

Now go back to (*)

& natural spline condition

$$z_0 = z_n = 0$$

$$\begin{bmatrix} 2/3 & 1/6 & & \emptyset \\ 1/6 & \ddots & \ddots & \\ & \ddots & \ddots & 1/6 \\ \emptyset & & 1/6 & 2/3 \end{bmatrix} \begin{bmatrix} z_1 \\ \vdots \\ z_{n-1} \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_{n-2} \end{bmatrix}$$

Tridiagonal system

Solve in $O(n)$ operations

$$b_k = \frac{1}{h^2}\left(f_{k+1} - 2f_k + f_{k-1}\right)$$

will appear in the worksheets

In Matlab

spline to get $p(x)$

ppval to evaluate it