NUR A lecture 2

In this lecture:

Linear interpolation
Lagrange polynomial
Neville’s algorithm
Natural cubic spline
Akima sub-spline
Interpolation in log vs linear space
Bilinear interpolation
Other 2D and 3D interpolation

Numerical Recipes for Astrophysics A

Lecture 2

Questions about last week?

Interpolation (and extrapolation)

NR Ch 3

Interpolation in general

Sample points / knots $(y_0, y_1, \ldots)$

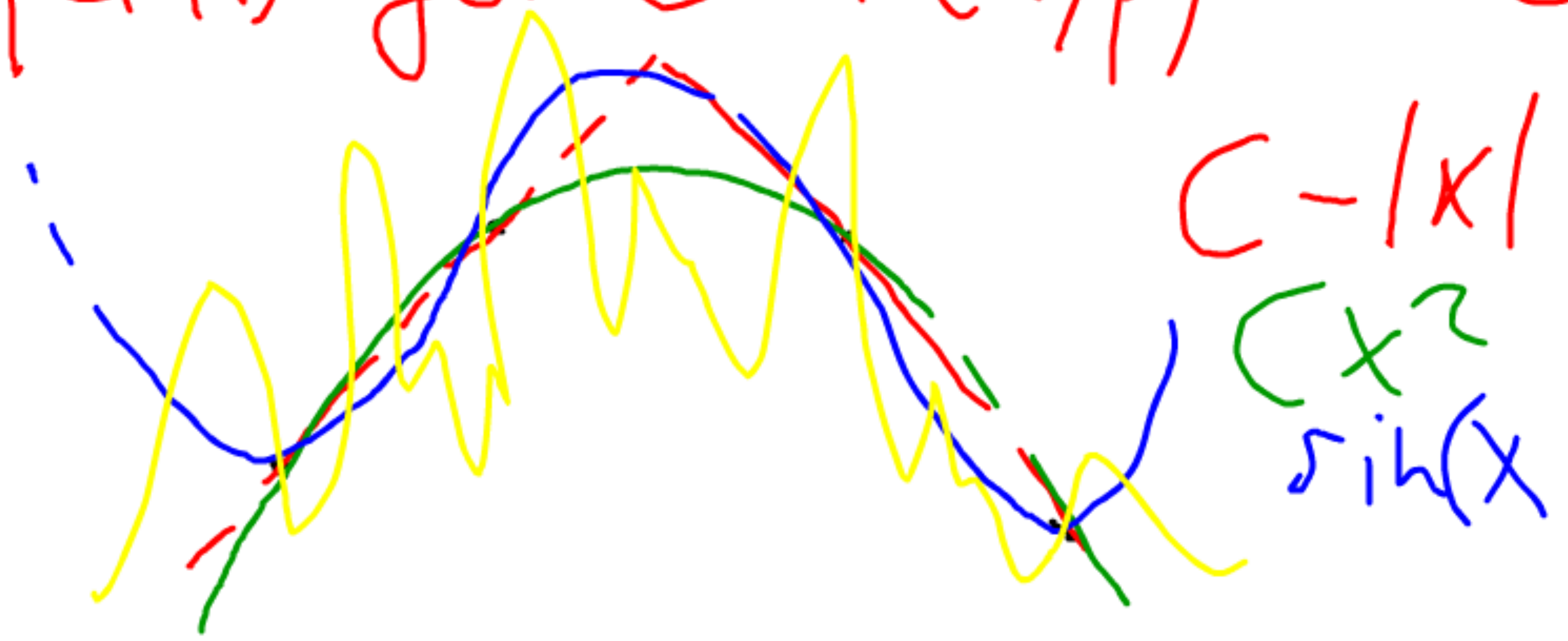
(Unknown) $f(x)$ known at N points x_i $(x_0, x_1, \ldots, x_{N-1})$

assume: smooth & well-behaved (e.g. polynomial)

x_i : strictly monotonic

Sample points = absolute truth (no noise, outliers)

Use M of N points for each (x, y) → order $M-1$



Building an interpolation object

- The bisection algorithm:

1. Start at the sample points at the edges, then split the sample points in two and see if the point you want interpolated falls in the right or left half (choose equality to mean either left or right consistently).

2. Update your edge points to the boundary of the relevant half.

3. Repeat steps 1 and 2 until the edge points are adjacent (or in the same position): you've now found the closest sample point(s).

4. Then identify the M sample points around the edge points (including the points themselves), taking care not to go outside of range, and return the index of the first of these. Let's call this index j_{low}.

Linear and beyond

piece-wise, $M=2$

- Linear interpolation $y = ax+b$

$$a = \frac{\delta y}{\delta x} \rightarrow y(x) = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}(x - x_i) + y_i$$

Where does this fail? $\rightarrow x_{i+1} = x_i$ (won't happen if x_i str. monot.)

- Lagrange polynomial $\rightarrow$ unique polynomial through M points

$$P(x) = \sum_{i=0}^{M-1} \prod_{j \neq i} \frac{(x - x_j)}{(x_i - x_j)} y_i$$

$\leftarrow$ NEVER use for $M \gtrsim 5$

$\leftarrow$ NEVER use like this

Neville's algorithm

Valid on $[x_i, x_j]$

E.g. $M=4$

Solution exact at x_i

Solution exact at x_j

$$H(x) = \frac{(x_j - x)F_i(x) + (x - x_i)G_j(x)}{x_j - x_i}$$

$y_0 =$

$y_1 =$

$$\begin{tikzcd}
x_0 : & P_0 \arrow[dr] & & & \\
& & P_{01} \arrow[dr] & & \\
x_1 : & P_1 \arrow[ur] \arrow[dr] & & P_{012} \arrow[dr] & \\
& & P_{12} \arrow[ur] \arrow[dr] & & P_{0123} \\
x_2 : & P_2 \arrow[ur] \arrow[dr] & & P_{123} \arrow[ur] & \\
& & P_{23} \arrow[ur] & & \\
x_3 : & P_3 \arrow[ur] & & &
\end{tikzcd}$$

$$P_{01} = \frac{(x_1 - x)P_0 + (x - x_0)P_1}{x_1 - x_0}$$

$$P_{i(i+1)(i+2)} = \frac{(x_{i+2} - x)P_{i(i+1)} + (x - x_i)P_{(i+1)(i+2)}}{x_{i+2} - x_i}$$

Neville's algorithm

e.g. bisection

1. Identify the M tabulated points around x to use in interpolation (see before – order is $M - 1$).

2. Set the initial P_i to the values at each of these points. $P_i = y_i$

3. Check which point is closest to x, and make the initial solution equal to this tabulated value.

4. Loop over orders k from 1 through $M - 1$: loop $\longrightarrow$

   loop

5. Loop over the current intervals $[x_i, x_j]$ (with $j = i + k$) with i from 0 through $M - 1 - k$:

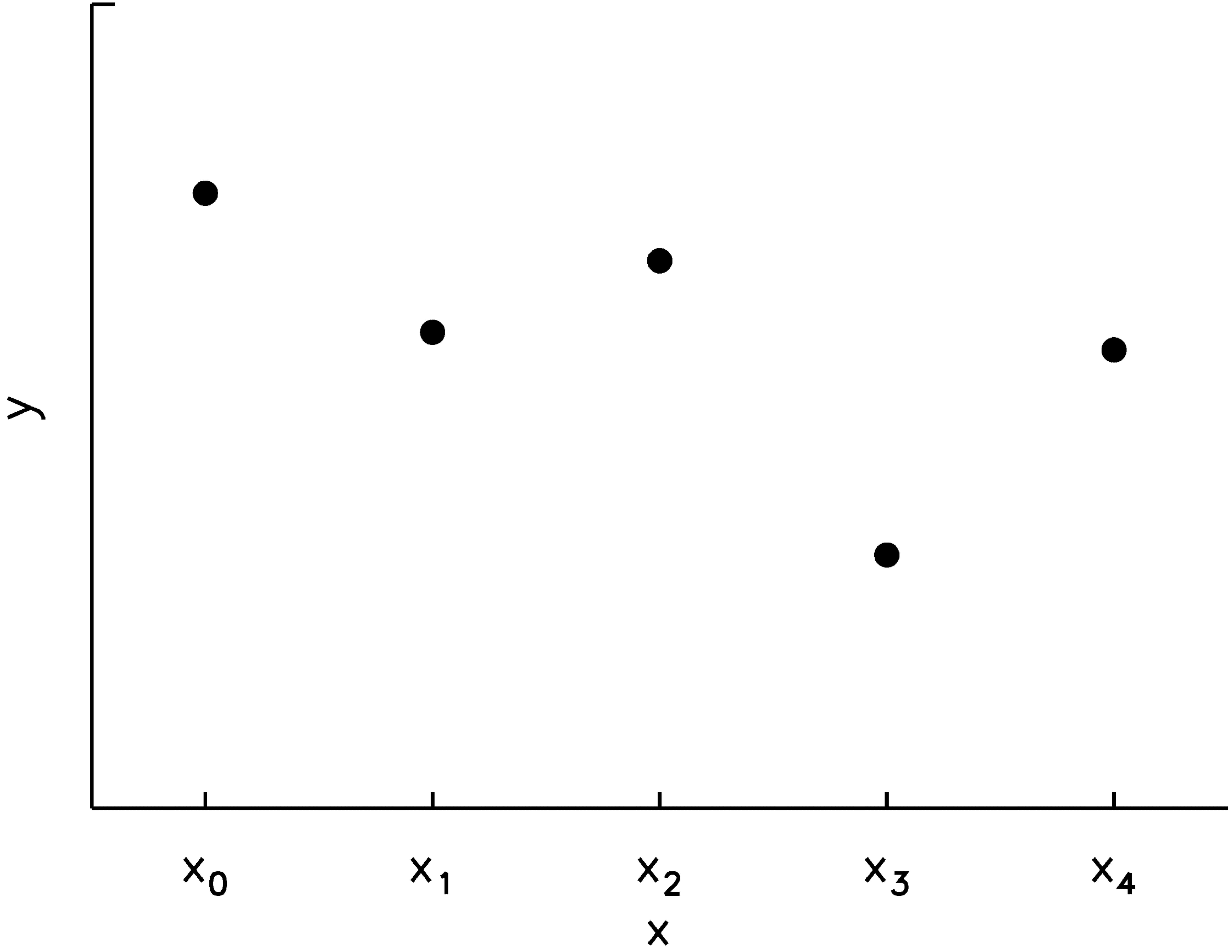
   $\leftarrow$ use $H(x)$

6. Update the P_i value for the interval, overwriting previous orders (P_0 by P_{01}, P_1 by P_{12}, etc).

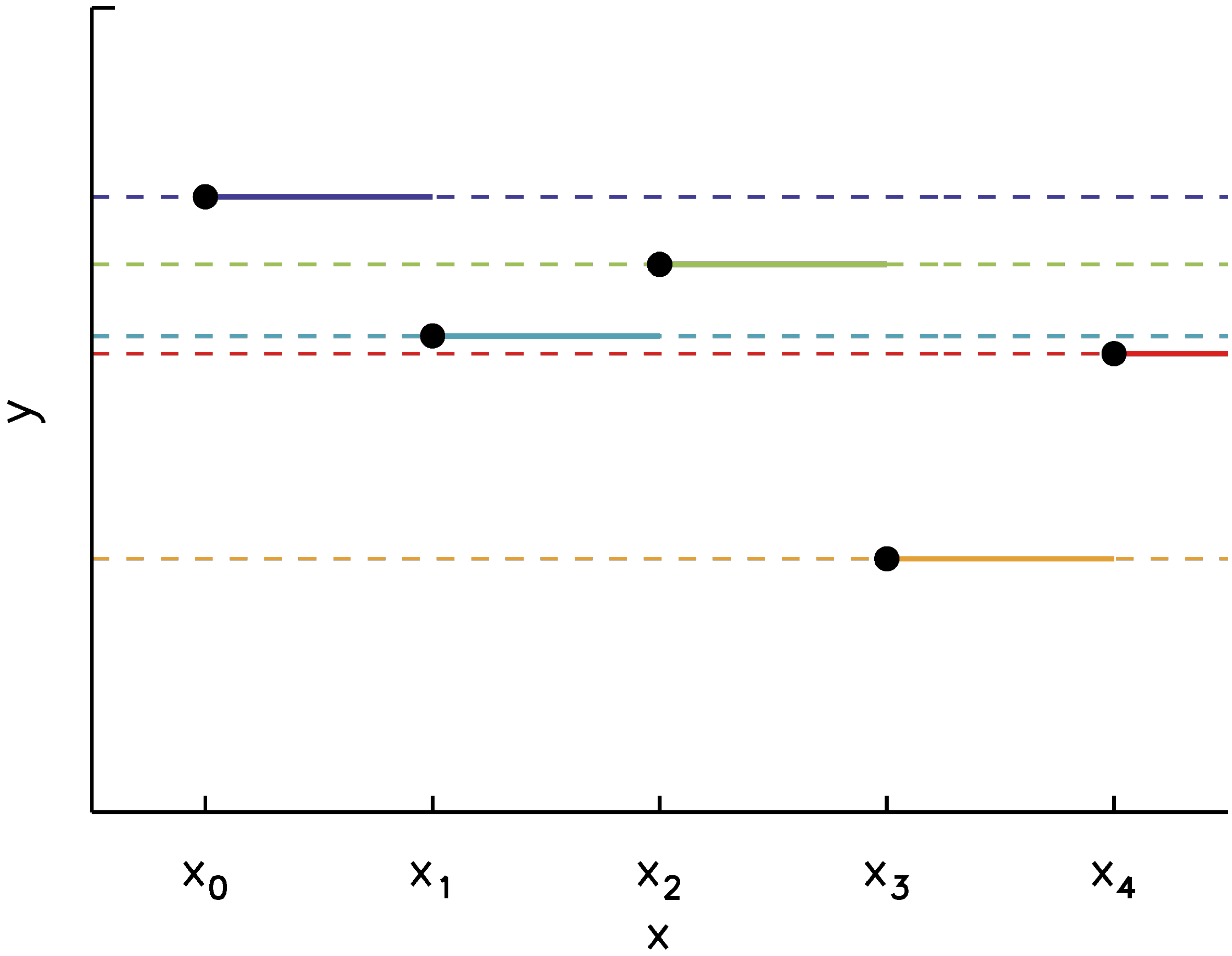
7. Close the loops, and save the last addition closest to x as the error estimate; P_0 holds the solution.

   e.g. $dy = |P_{0123} - P_{012}|$

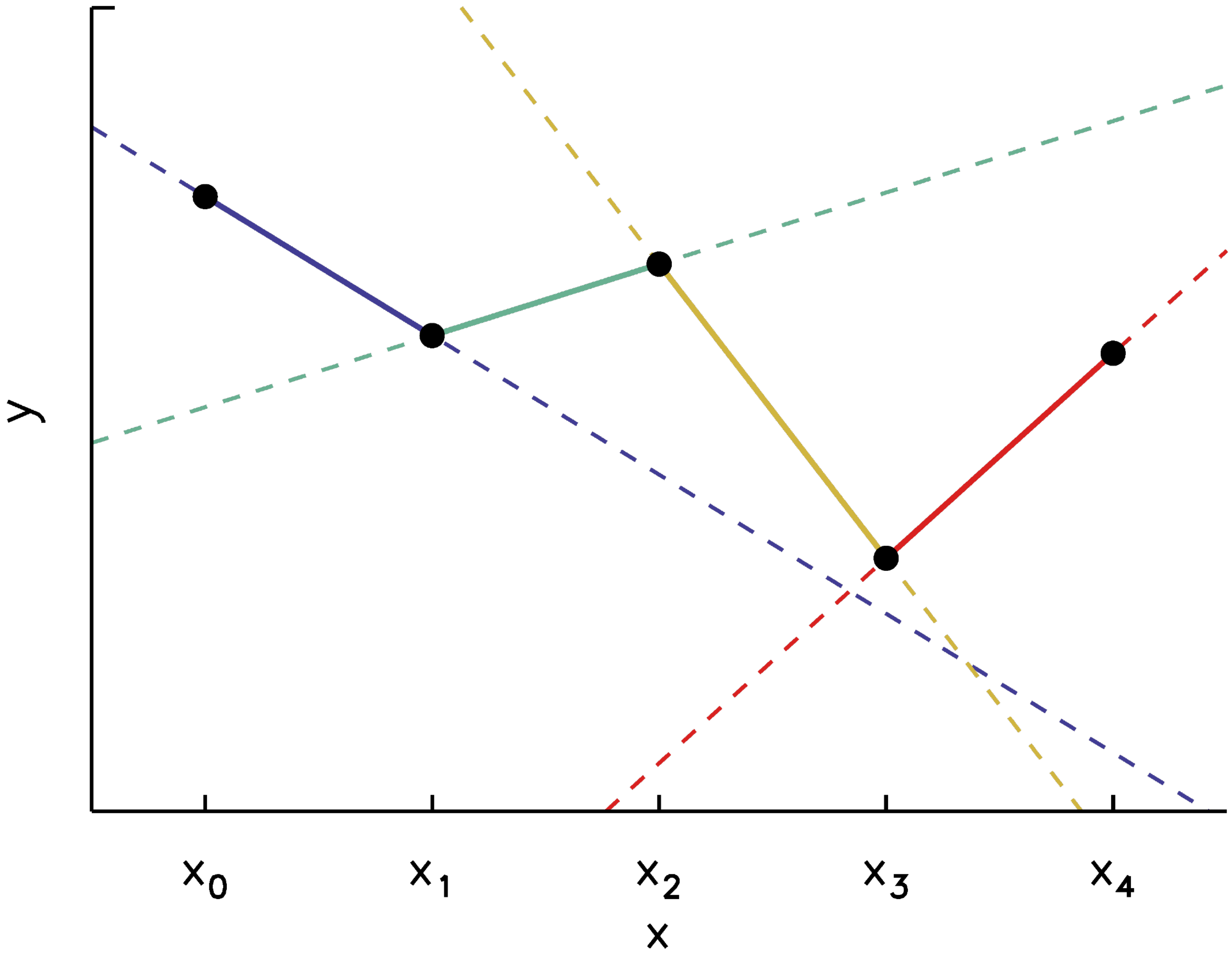
Neville's algorithm



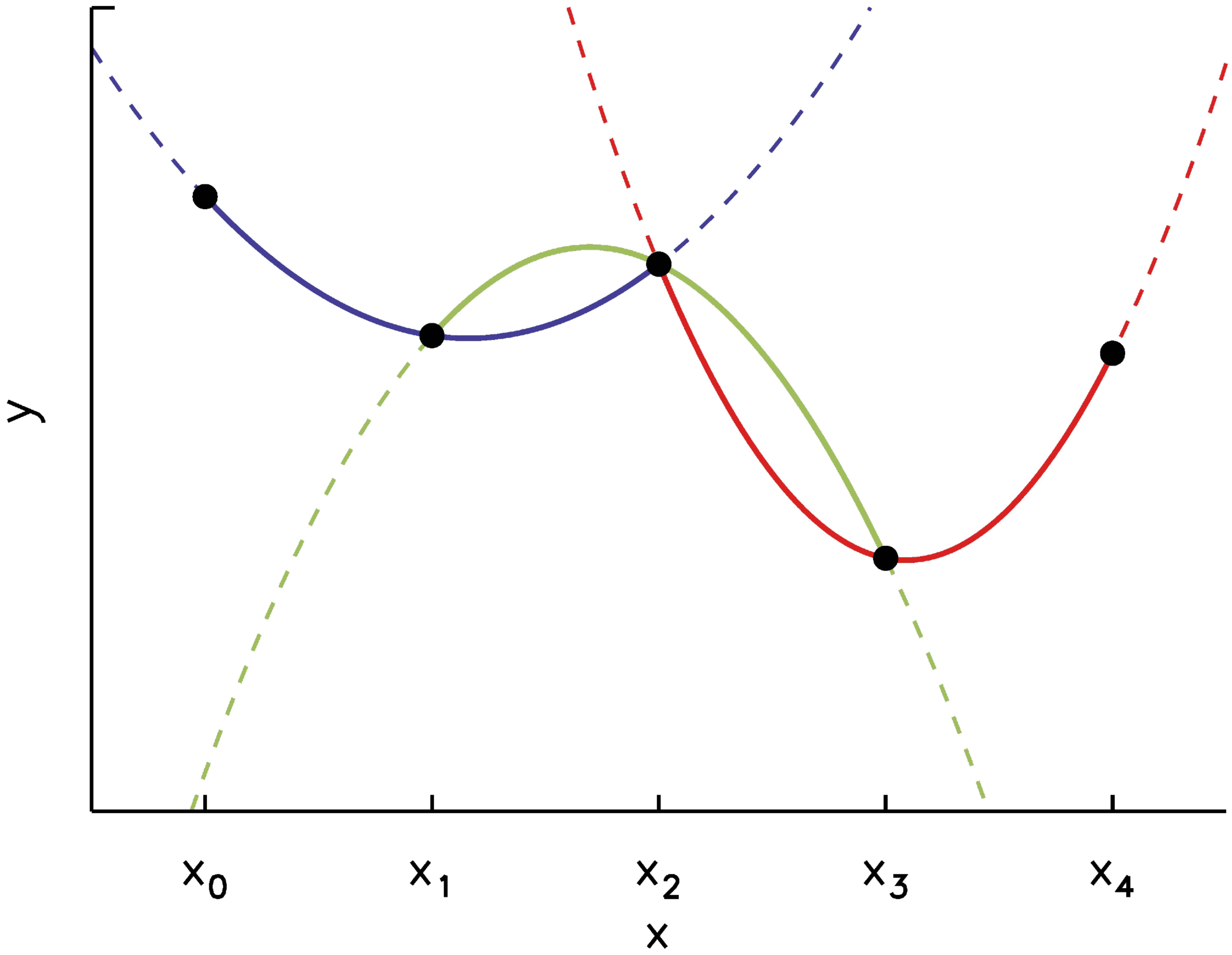
Neville's algorithm



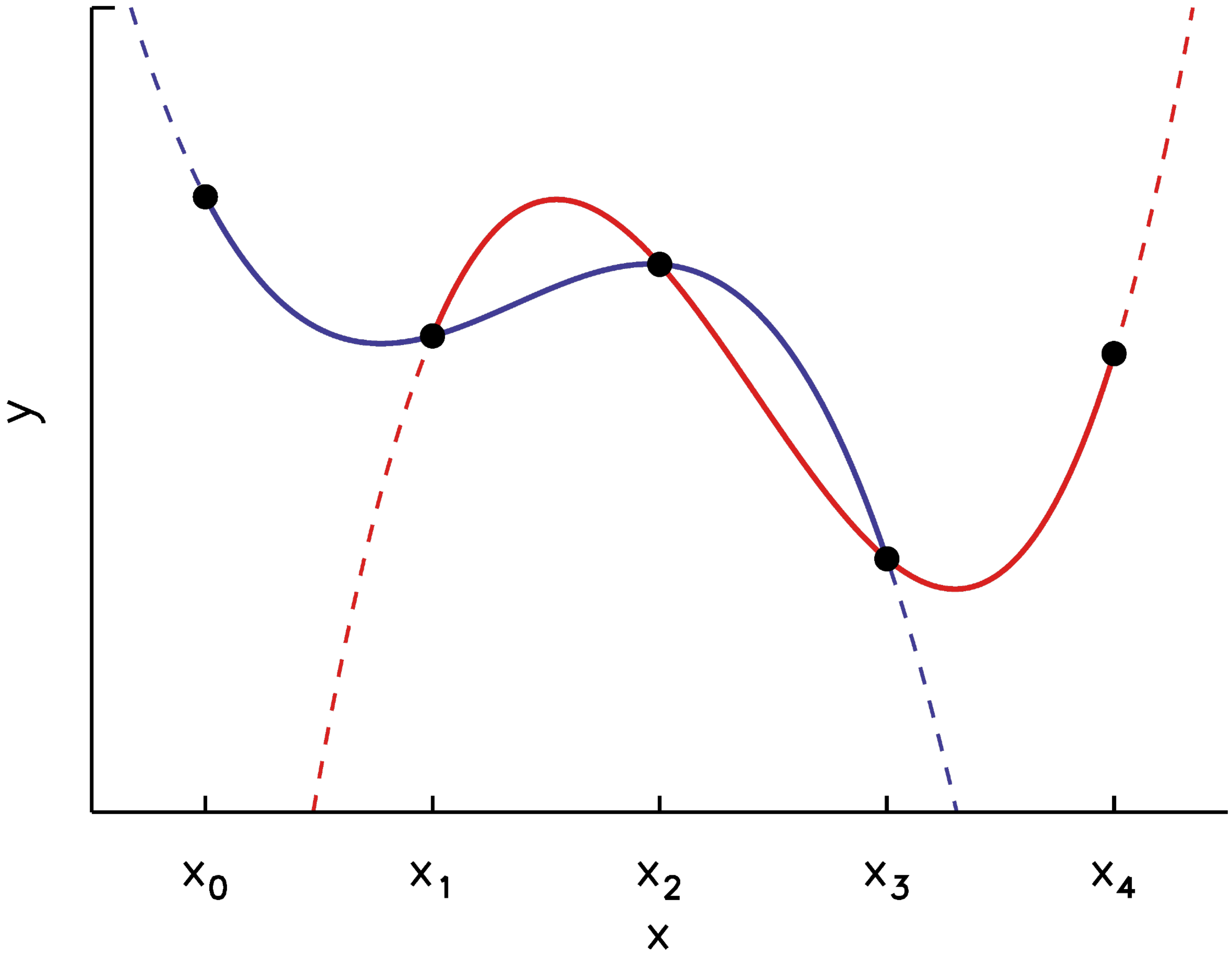
Neville's algorithm



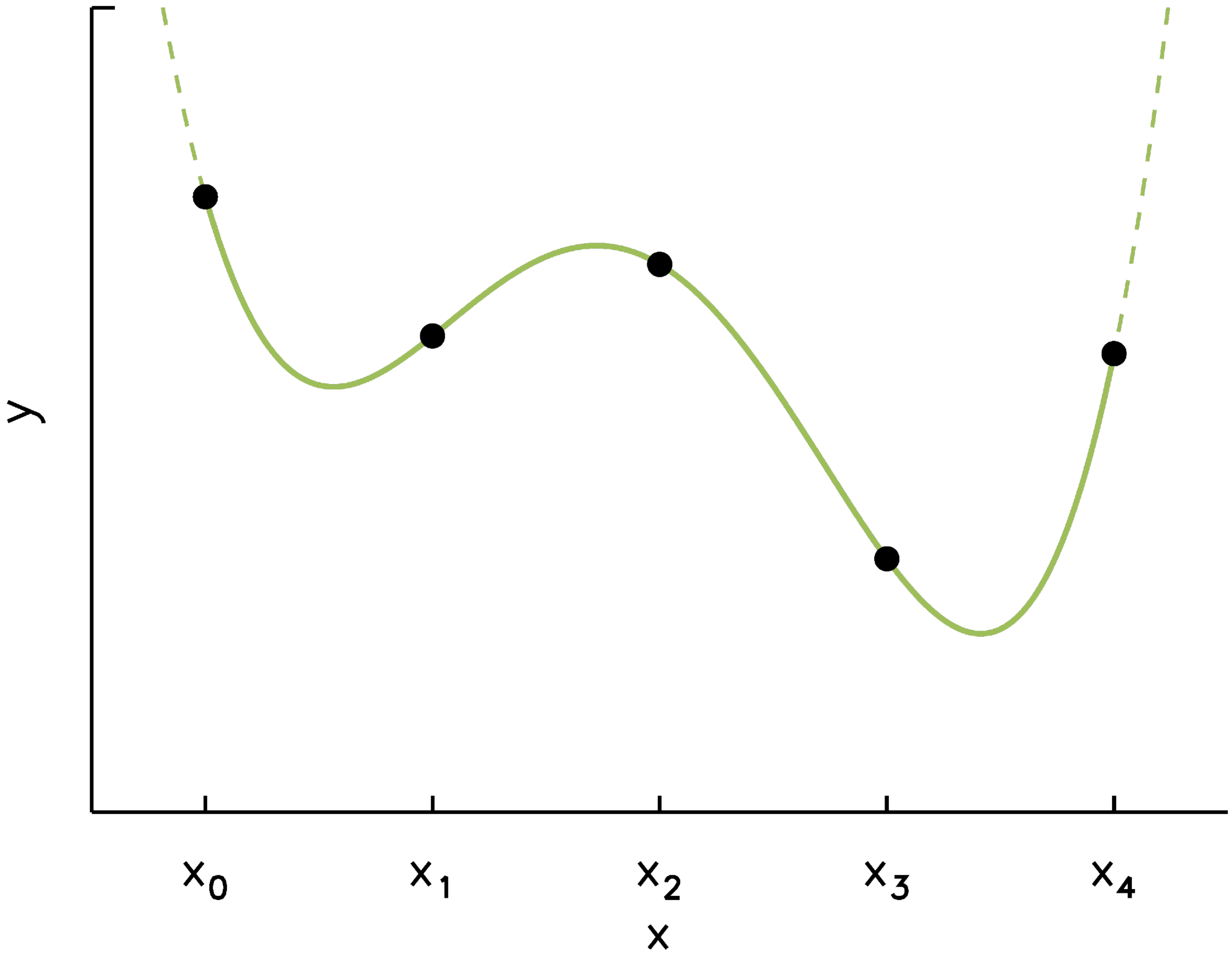
Neville's algorithm



Neville's algorithm



Neville's algorithm



Splines

piecewise polynomial with derivative constraints

- (Natural) cubic spline

$$[x_i, x_{i+1}]: \quad y(x) = a_i + b_i(x-x_i) + c_i(x-x_i)^2 + d_i(x-x_i)^3$$

$$y(x_i) = y_i = a_i; \quad y(x_{i+1}) = y_{i+1} \rightarrow N-1 \text{ constraints}$$

Unknowns: $3(N-1) = 3N-3$

$$[y'(x_{i+1})]_i = [y'(x_{i+1})]_{i+1} \rightarrow \text{cont. 1st deriv}$$

$$[y''(x_{i+1})]_i = [y''(x_{i+1})]_{i+1} \rightarrow \text{cont. 2nd deriv}$$

$$N-1+N-2+N-2 = 3N-5$$

Splines

$y'' = 0$ at endpoints

- (Natural) cubic spline

Boundary conditions → last 2 constr.
(others possible! e.g. set 1st deriv)

solve system of eqs.

$$
\left\{
\begin{aligned}
y_i + b_i(x_{i+1} - x_i) + c_i(x_{i+1} - x_i)^2 + d_i(x_{i+1} - x_i)^3 &= y_{i+1}, \quad i = 0, \ldots, N-2 \\
b_i + 2c_i(x_{i+1} - x_i) + 3d_i(x_{i+1} - x_i)^2 &= b_{i+1}, \quad i = 0, \ldots, N-3 \\
2c_i + 6d_i(x_{i+1} - x_i) &= 2c_{i+1}, i = 0, \ldots, N-3
\end{aligned}
\right.
$$

Splines

no cont. 2nd derivs

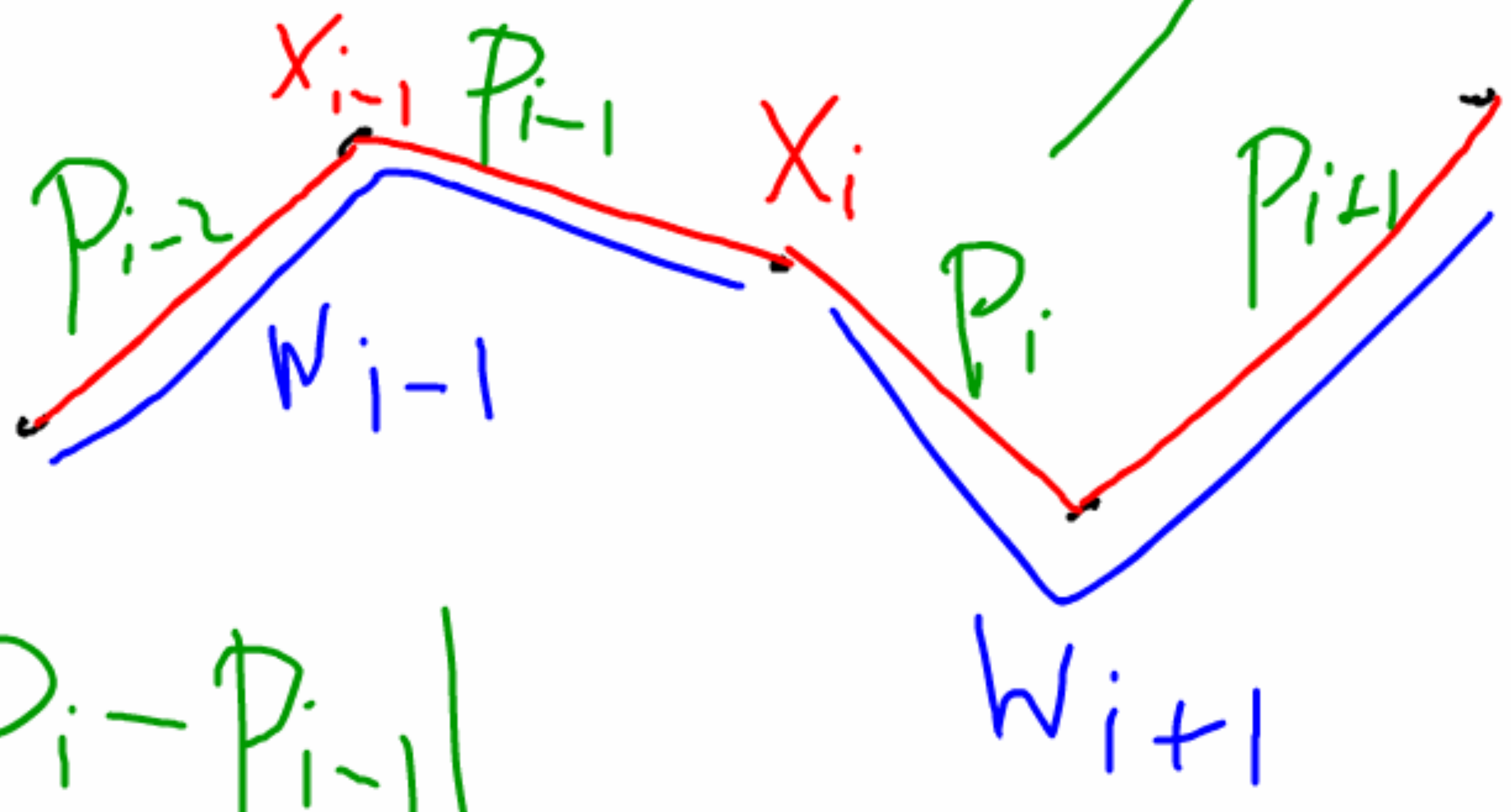
- Akima (sub-)spline (but it lets data lead more)

Same $y(x)$ per interval $\left(y(x) = a_i + b_i (x - x_i) + \ldots \right)$

$y(x_i) = y_i$, $y(x_{i+1}) = y_{i+1}$, $[y']_i = [y']_{i+1}$ as before

$$y'(x_i) = \frac{w_{i+1} p_{i-1} + w_{i-1} p_i}{w_{i+1} + w_{i-1}}$$

p_i : linear slope right of x_i



$w_i \equiv |p_i - p_{i-1}|$

Where does this fail? $w_{i+1} = w_{i-1} = 0$

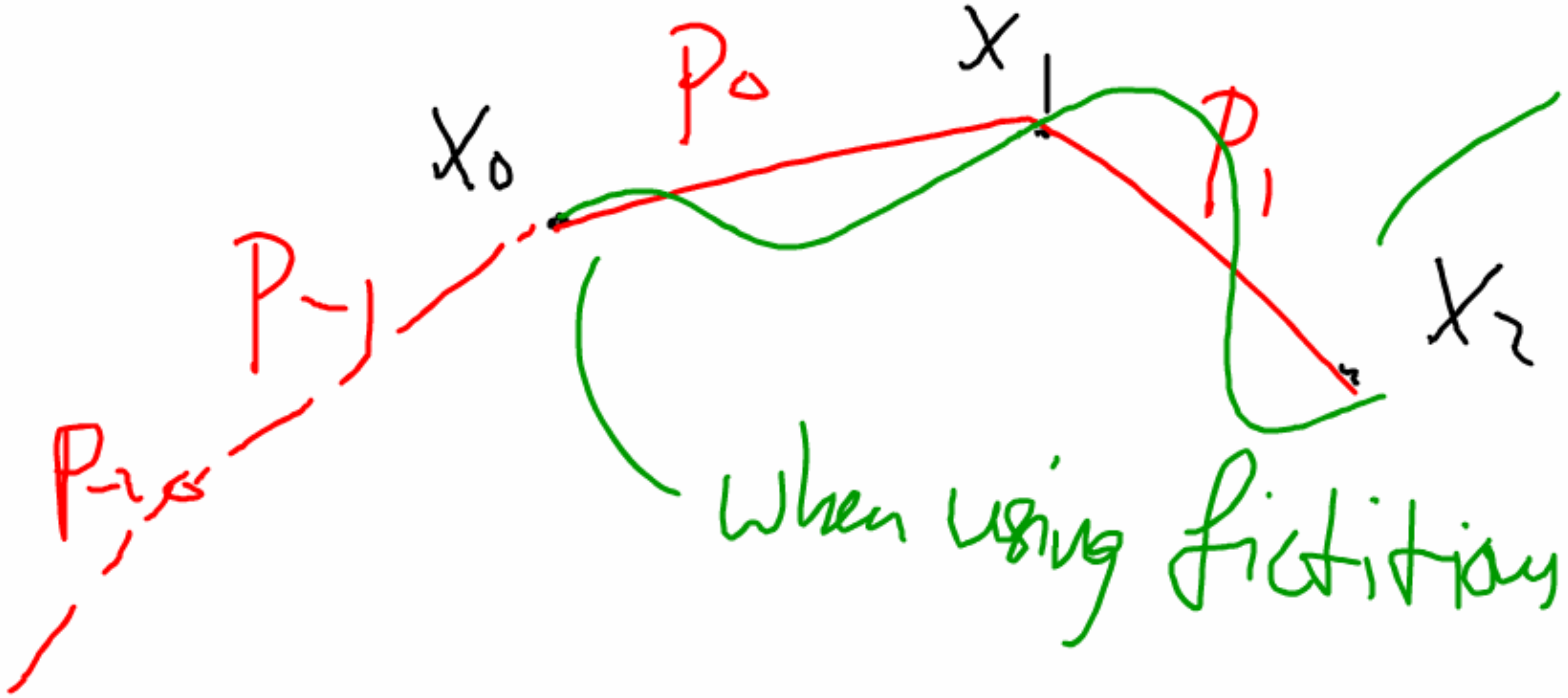
Splines

$W_{i-1} = W_{i+1} = 0$

$\|$

- Akima (sub-)spline

If 3 points in a line on both sides $\Rightarrow y'(x_i) = \frac{P_{i-1}+P_i}{2}$



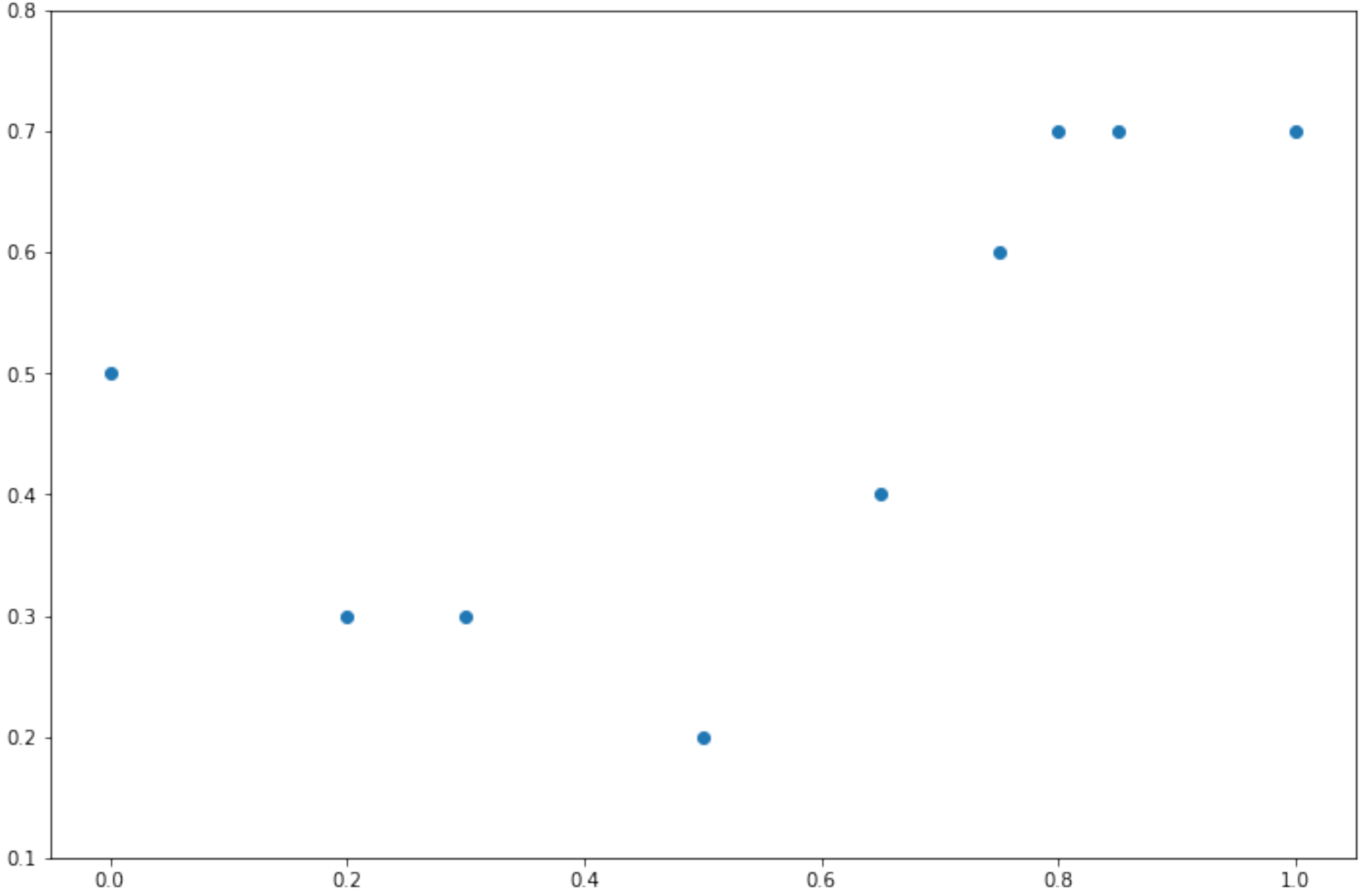
internal intervals: 3^{rd} order

when using fictitious slopes: 2^{nd} order

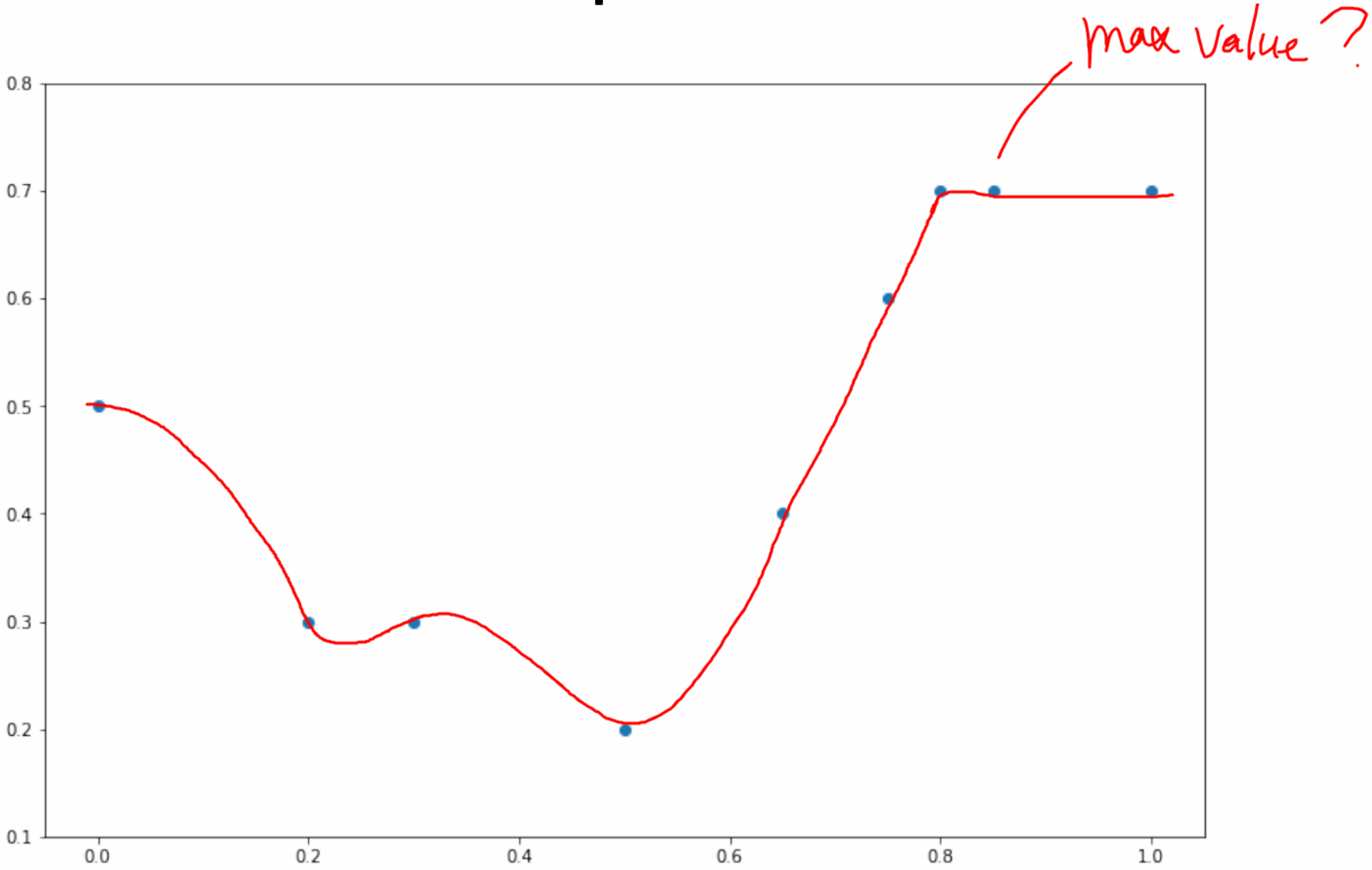
$$p_{-2} = 2p_{-1} - p_0 \qquad p_{N-1} = 2p_{N-2} - p_{N-3}$$

$$p_{-1} = 2p_0 - p_1 \qquad p_N = 2p_{N-1} - p_{N-2}$$

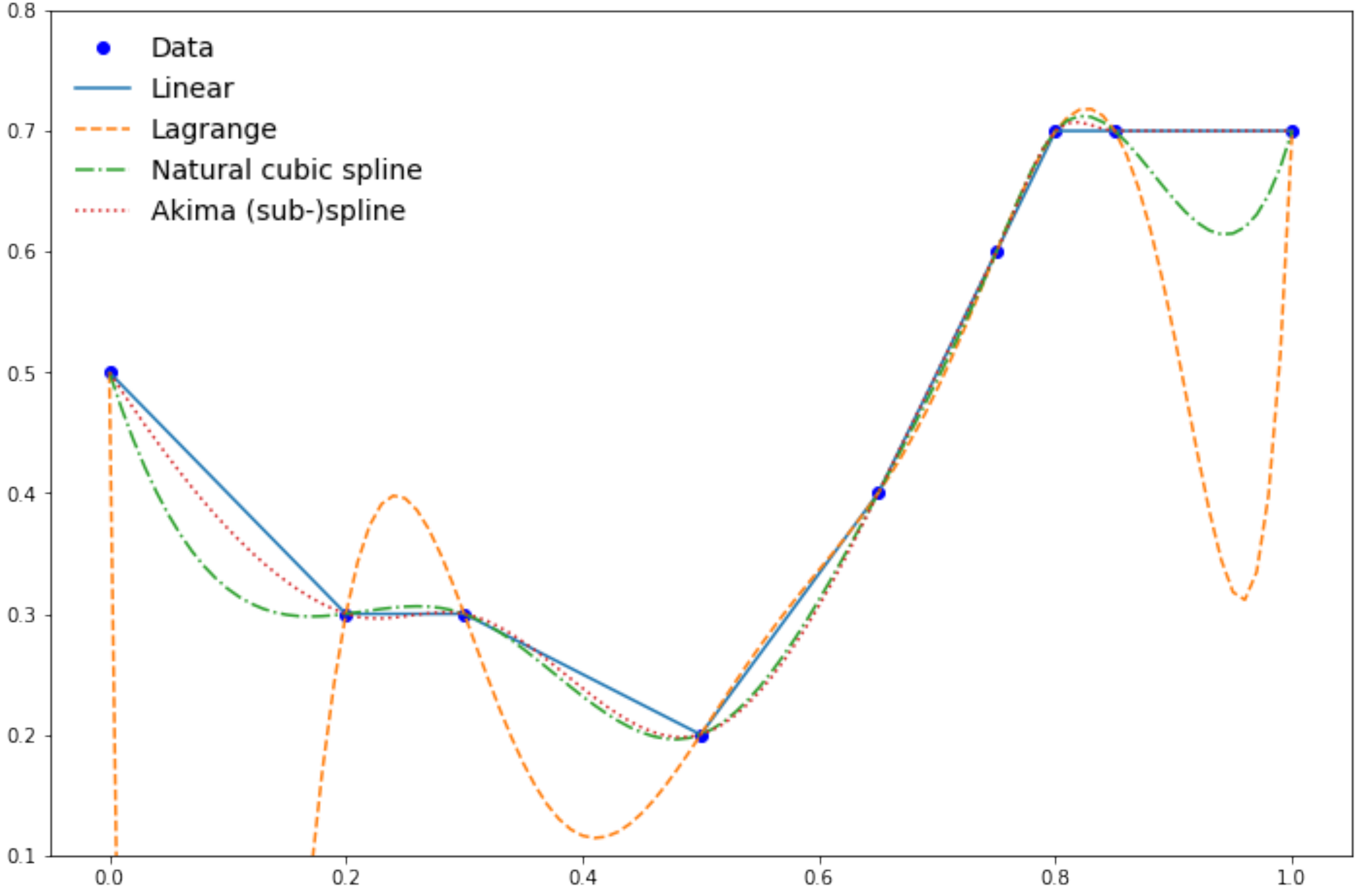
A comparison



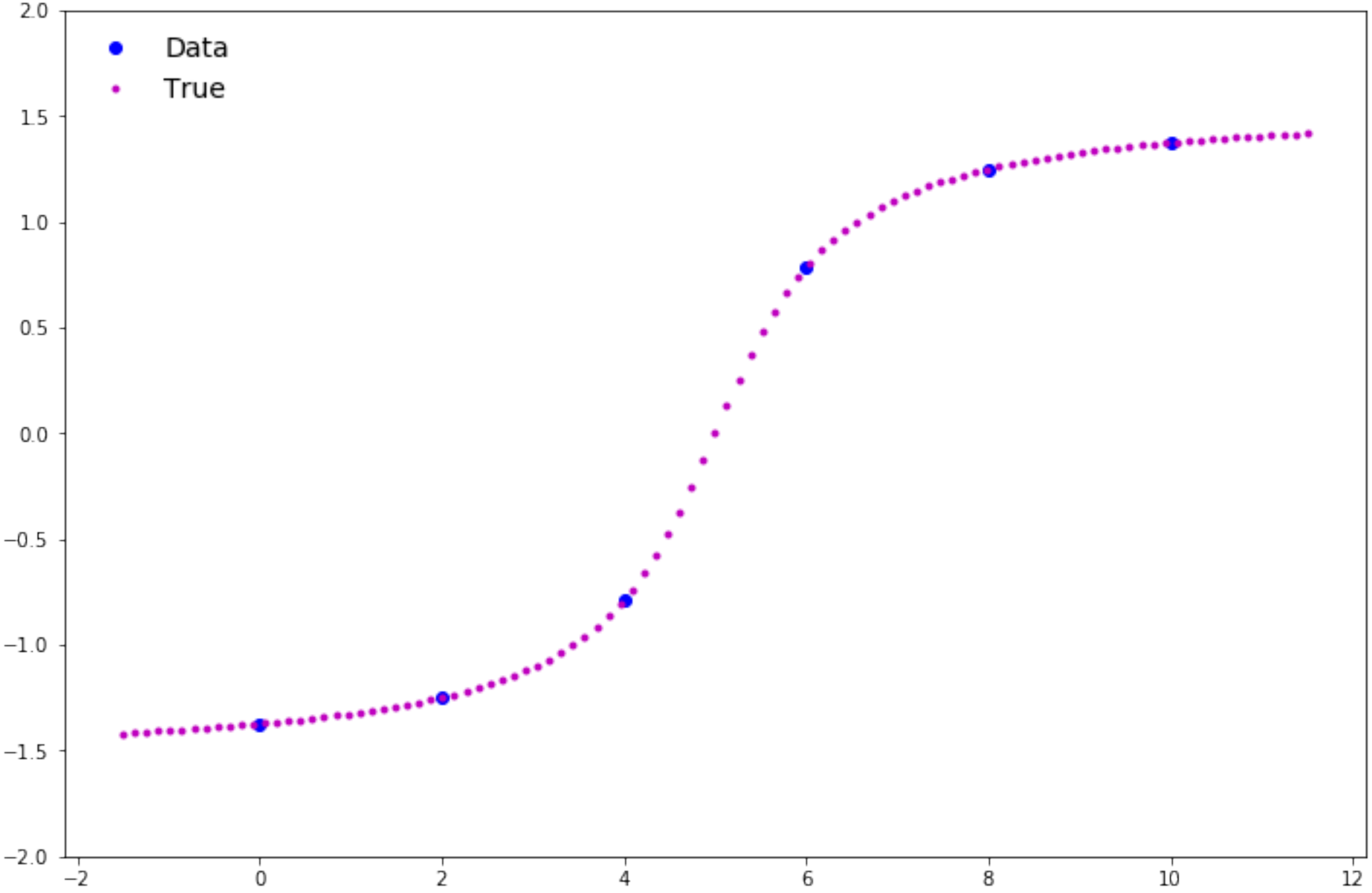
A comparison



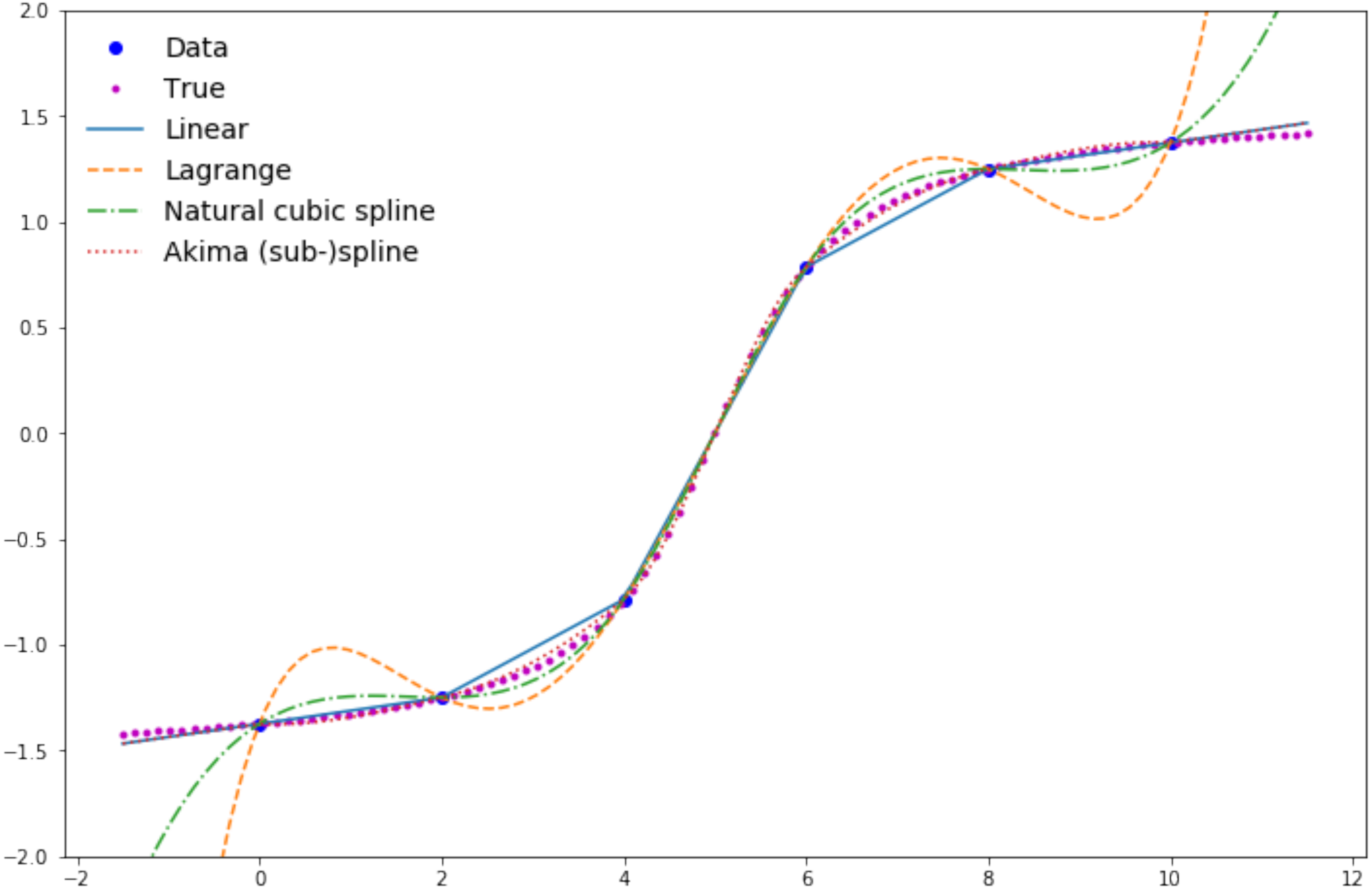
A comparison



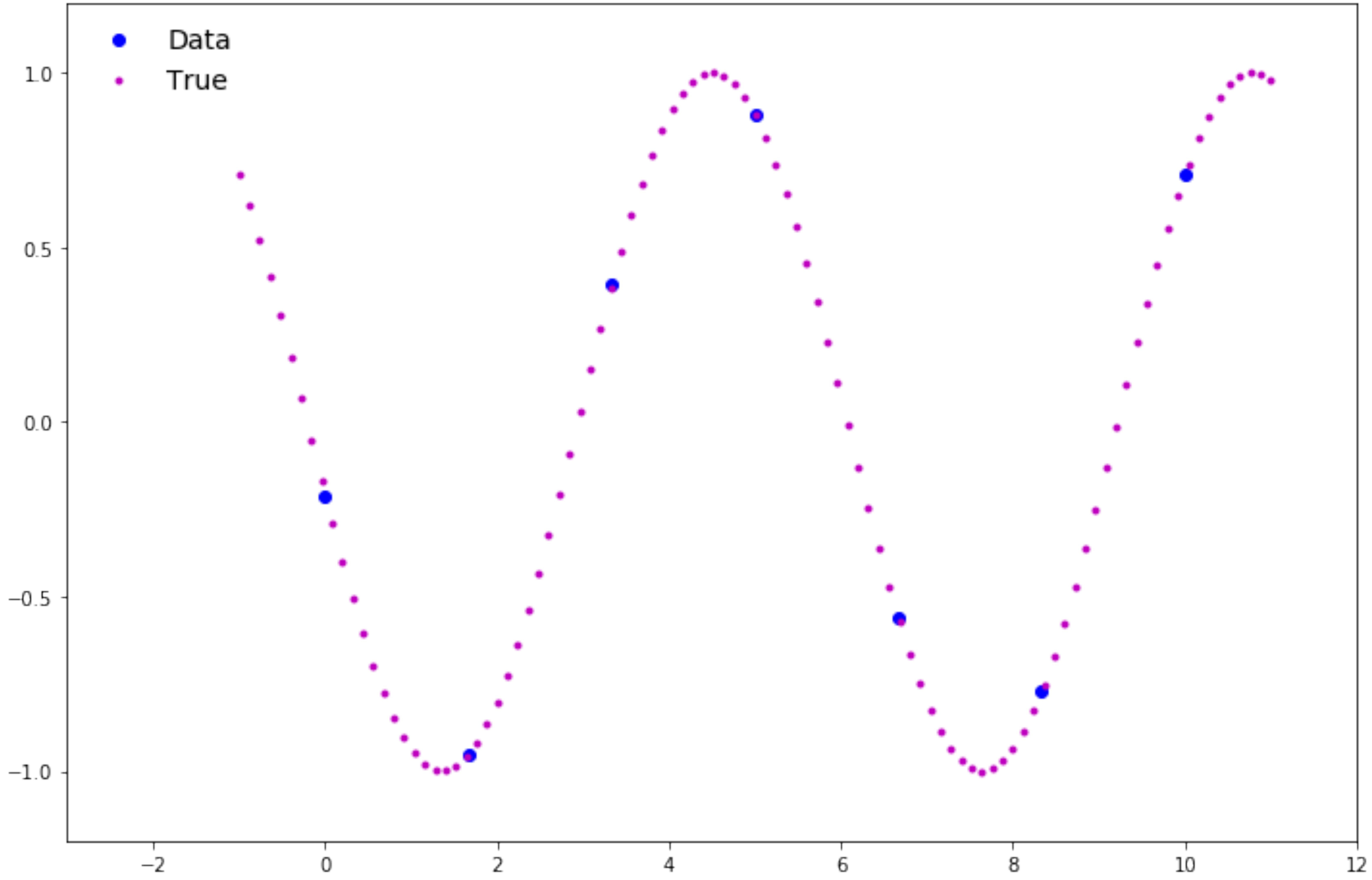
Inter- and extrapolation



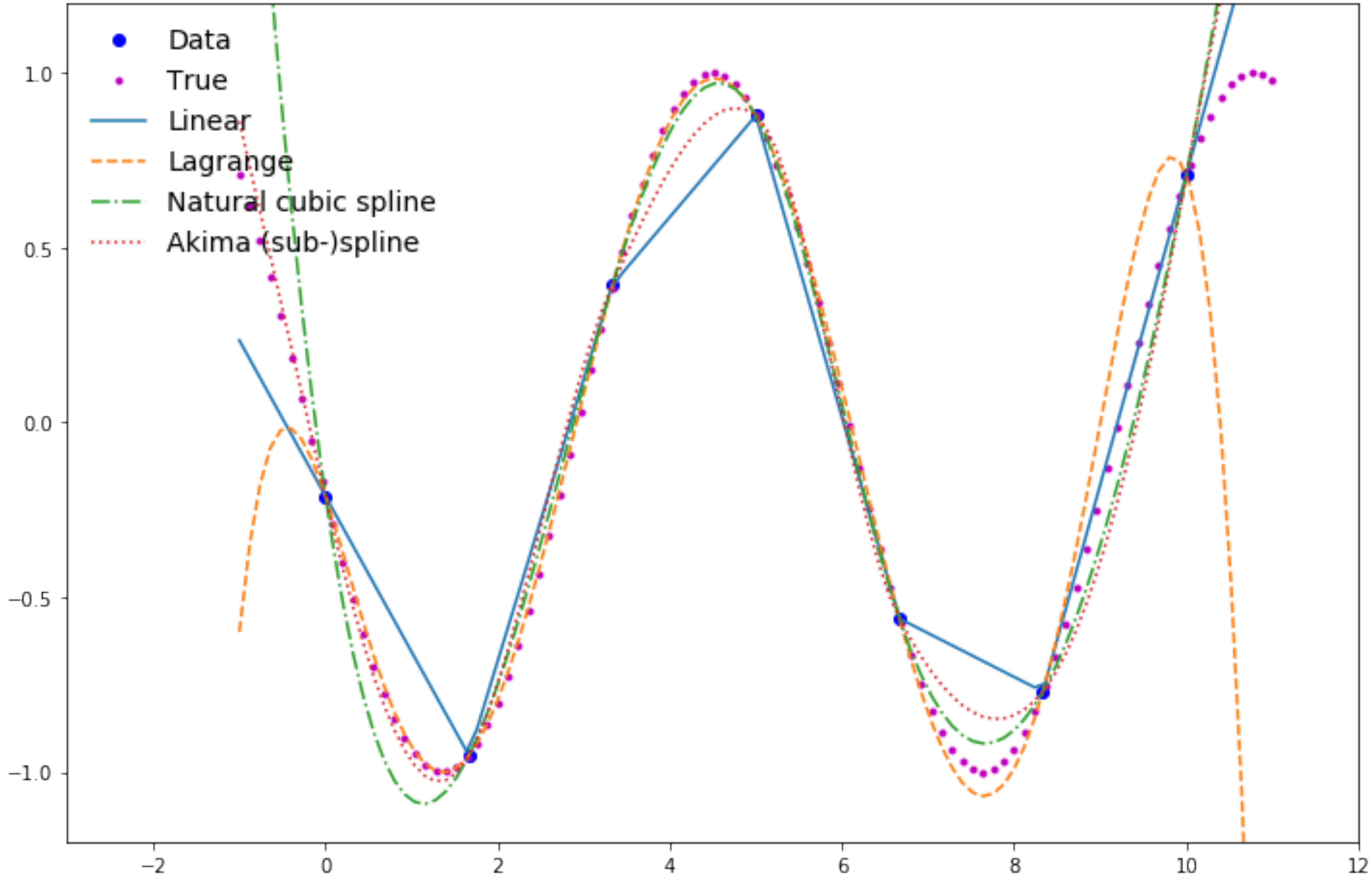
Inter- and extrapolation



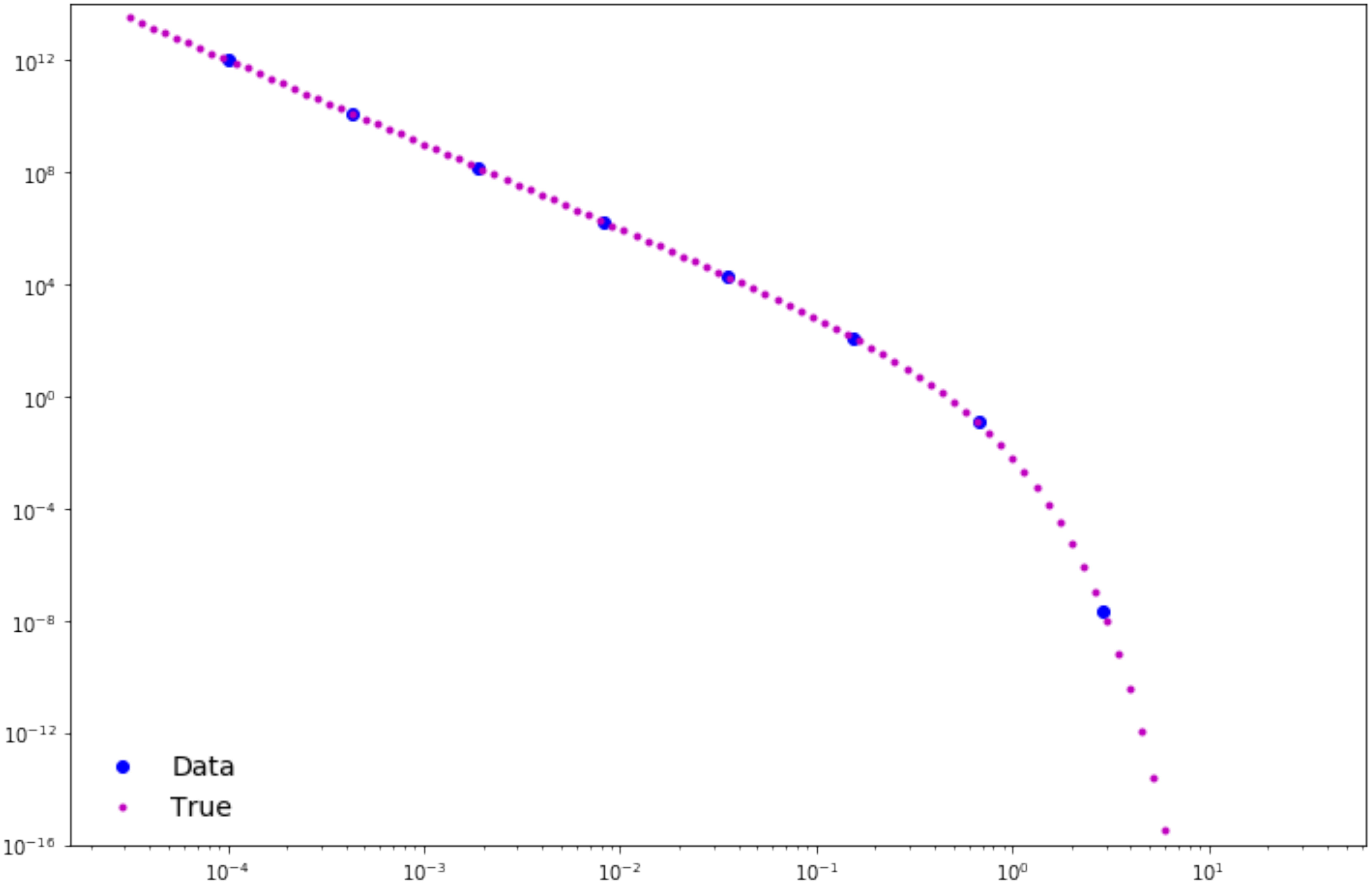
Inter- and extrapolation



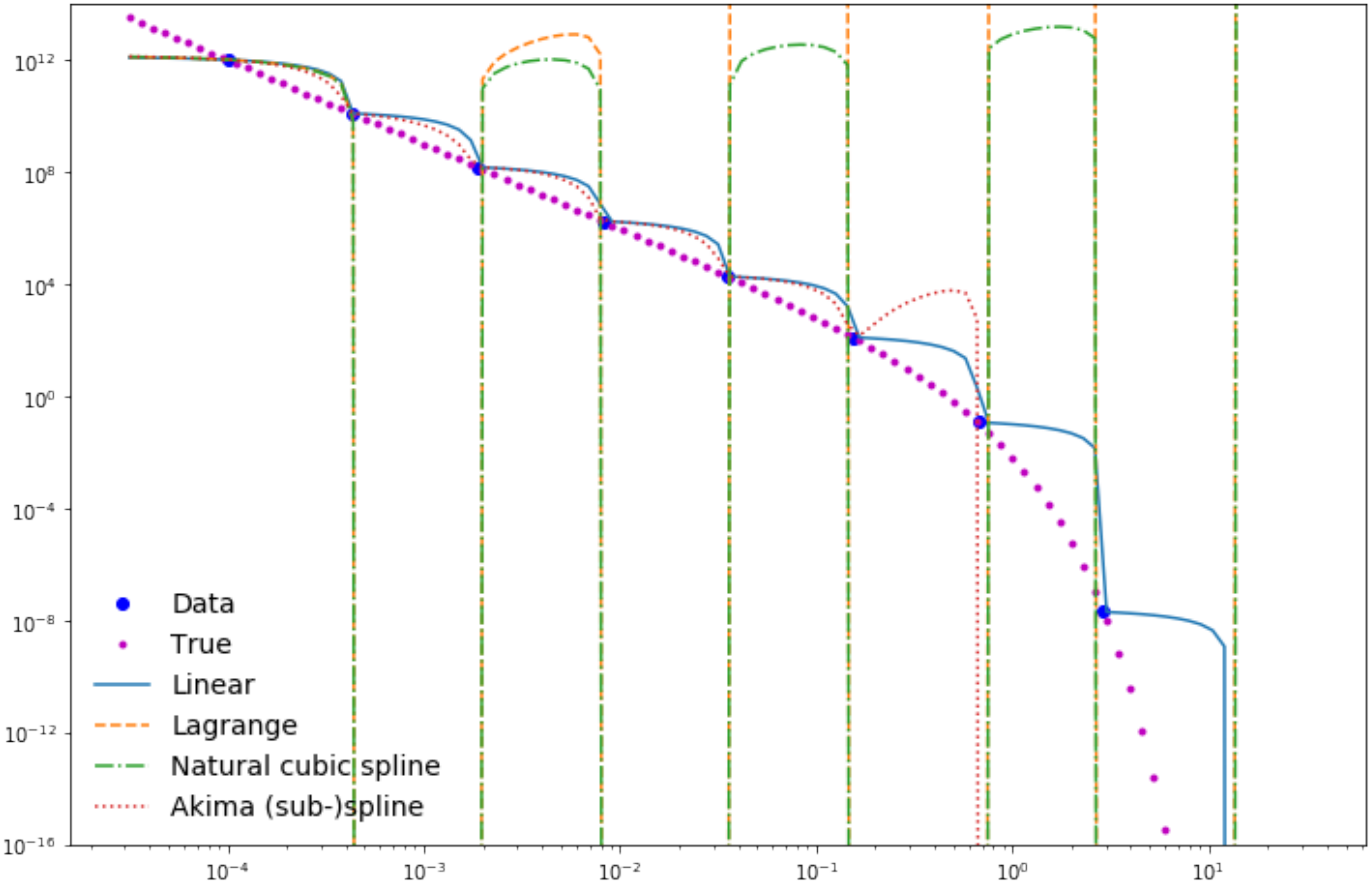
Inter- and extrapolation



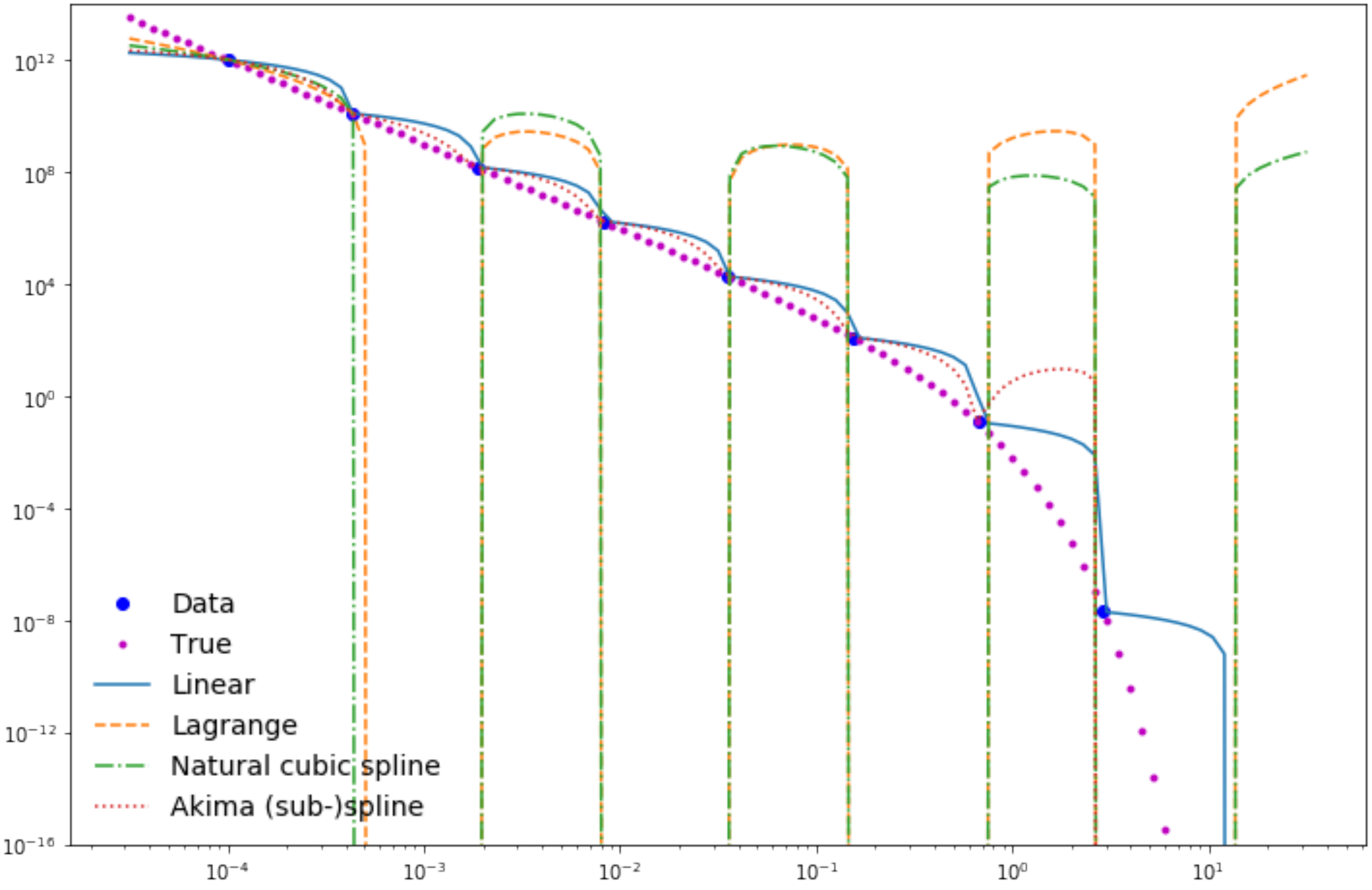
Log vs linear: a cautionary tale



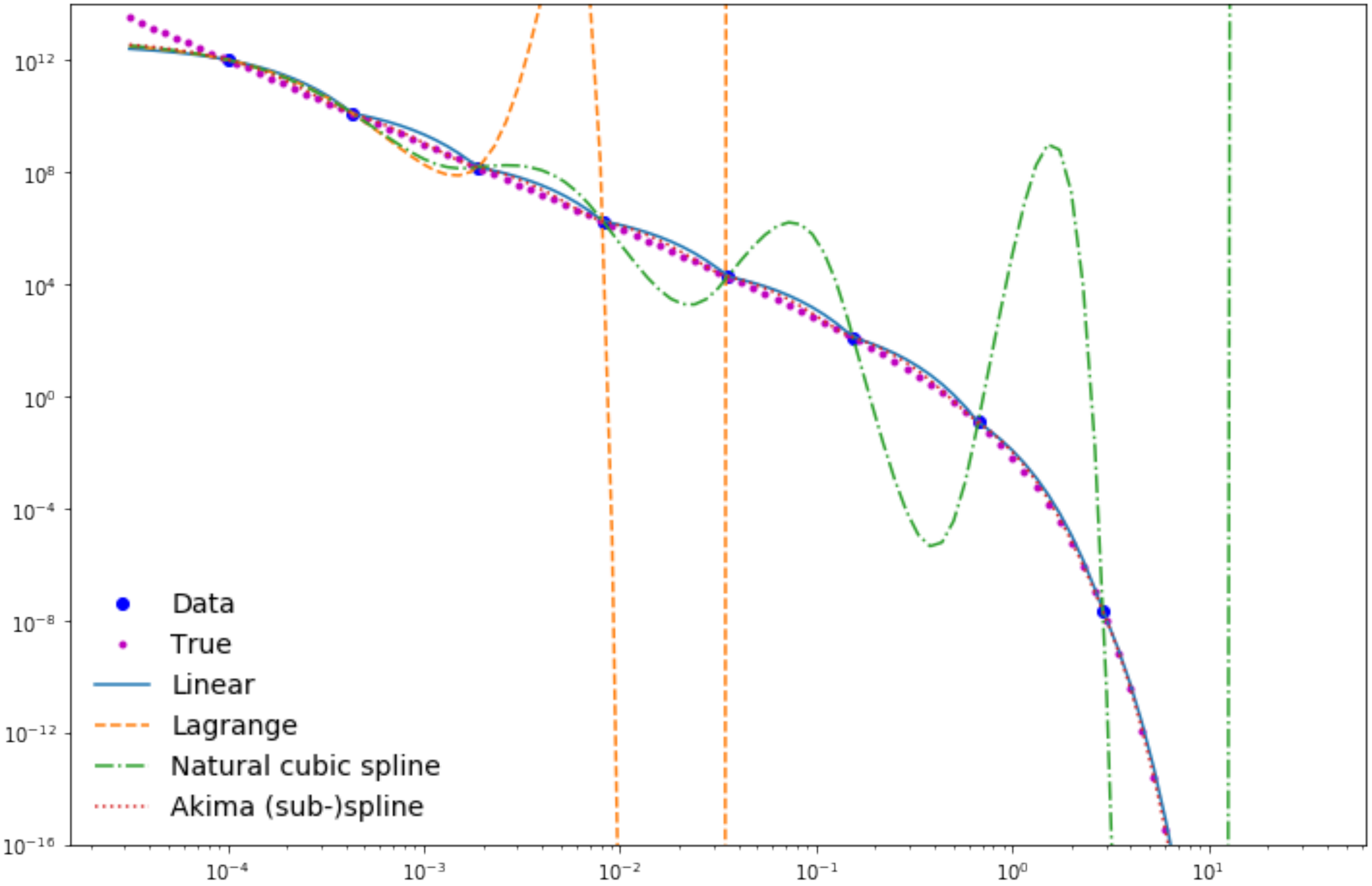
Log vs linear: lin y–lin x fit



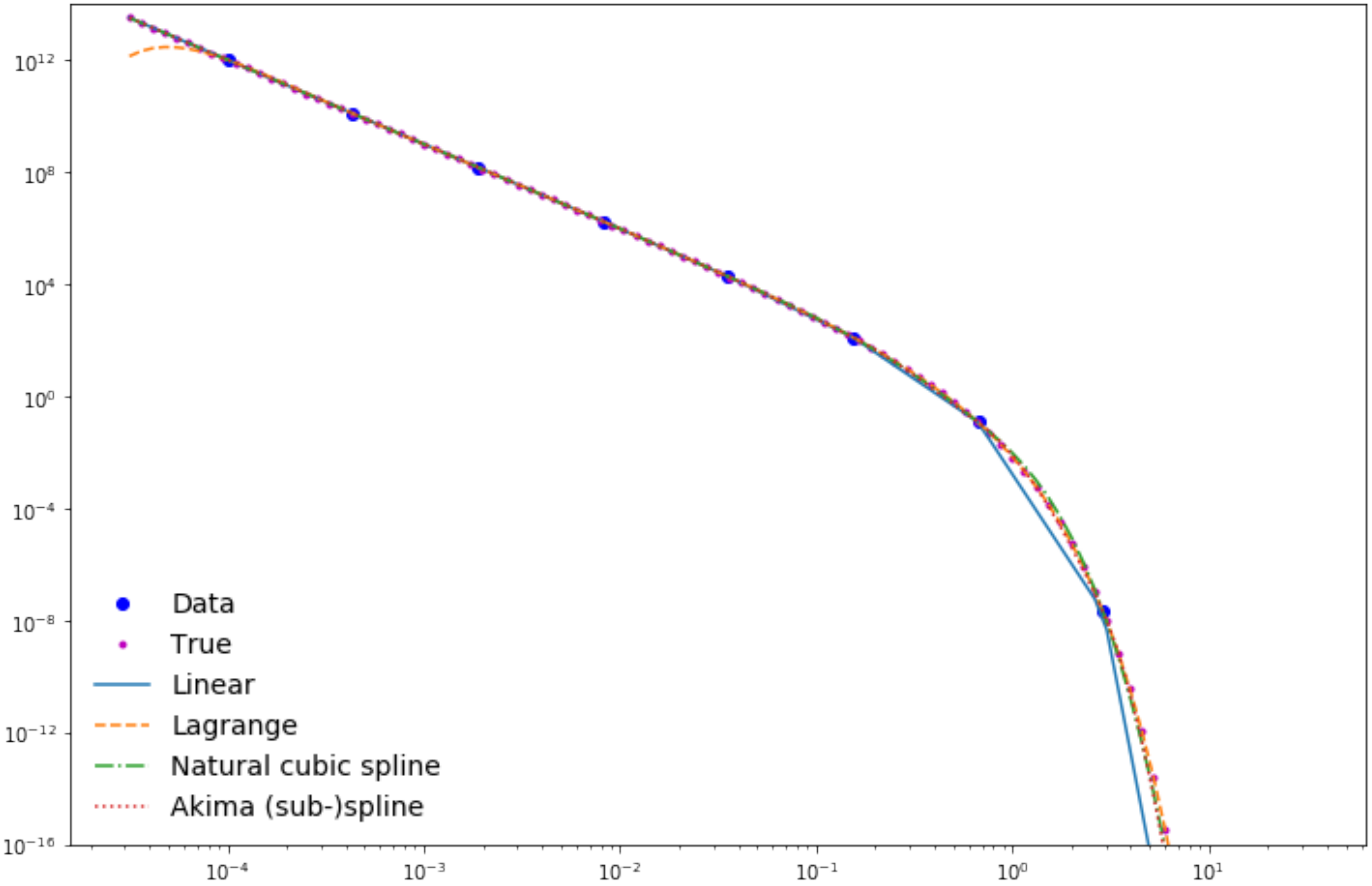
Log vs linear: lin y–log x fit



Log vs linear: log y–lin x fit



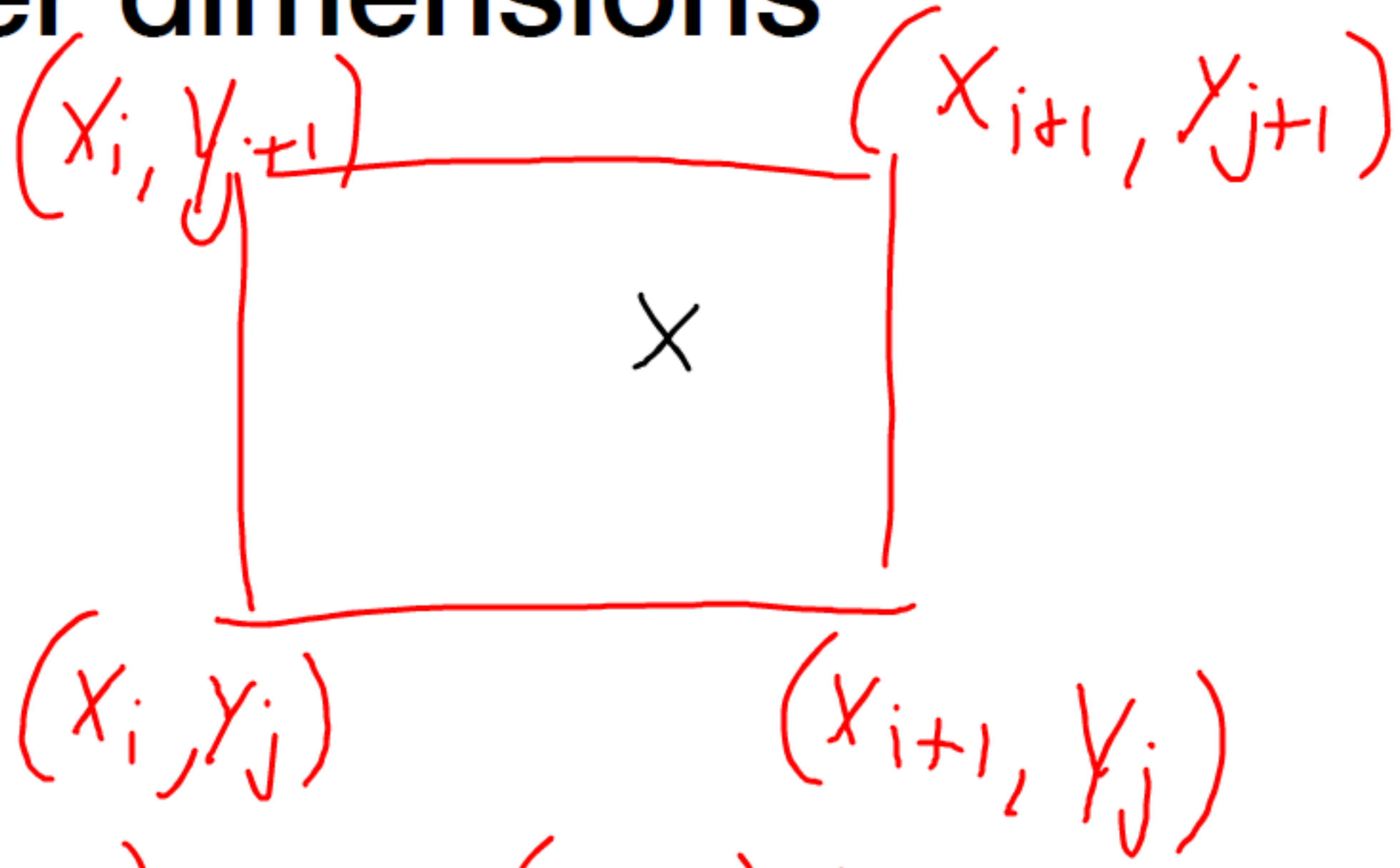
Log vs linear: log y–log x fit



Interpolation in higher dimensions

- Bilinear interpolation

Assume grid



$$z(x,y) = a_{ij} + b_{ij}(x - x_i) + c_{ij}(y - y_j) + d_{ij}(x - x_i)(y - y_j)$$

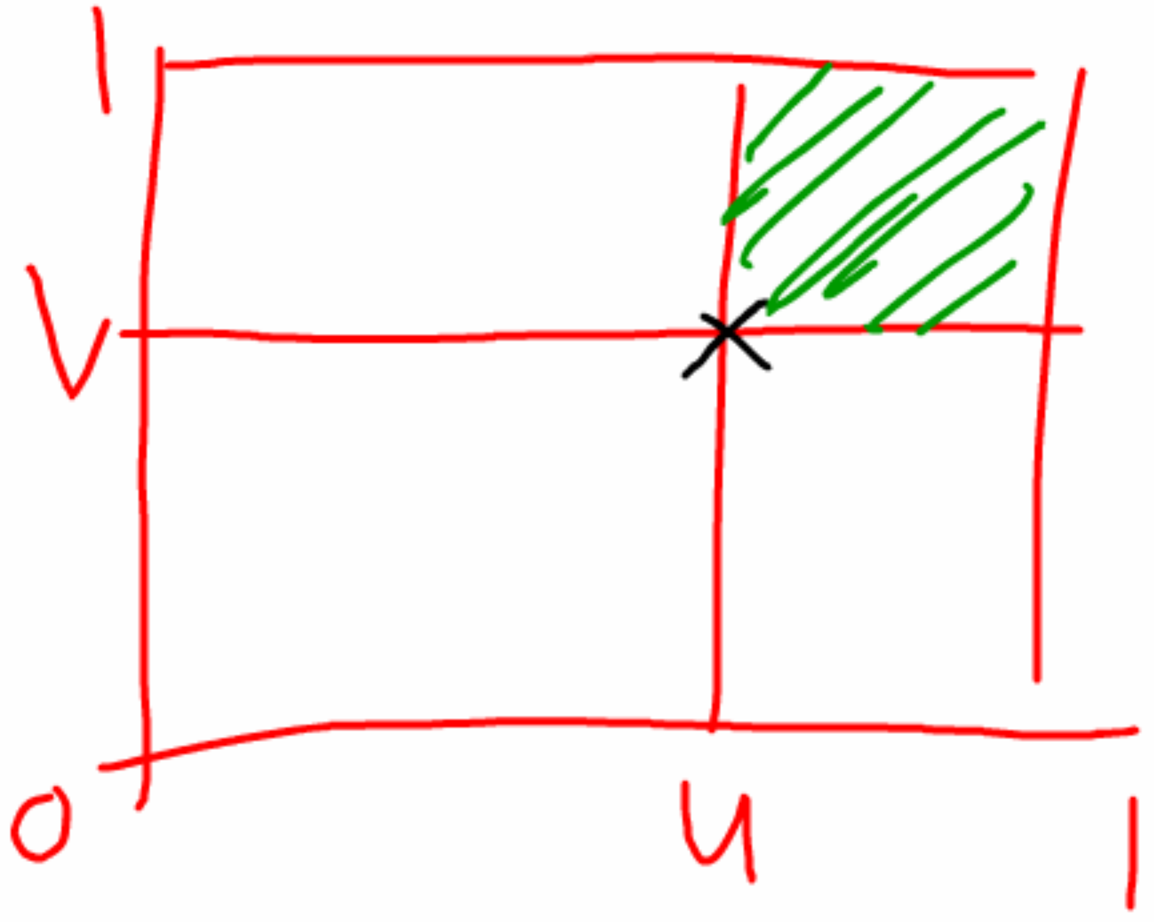
System of eqs ....

alternatives?

Interpolation in higher dimensions

- Bilinear alternative 1

$$u \equiv \frac{x - x_i}{x_{i+1} - x_i}, \quad v \equiv \frac{y - y_j}{y_{j+1} - y_j}$$



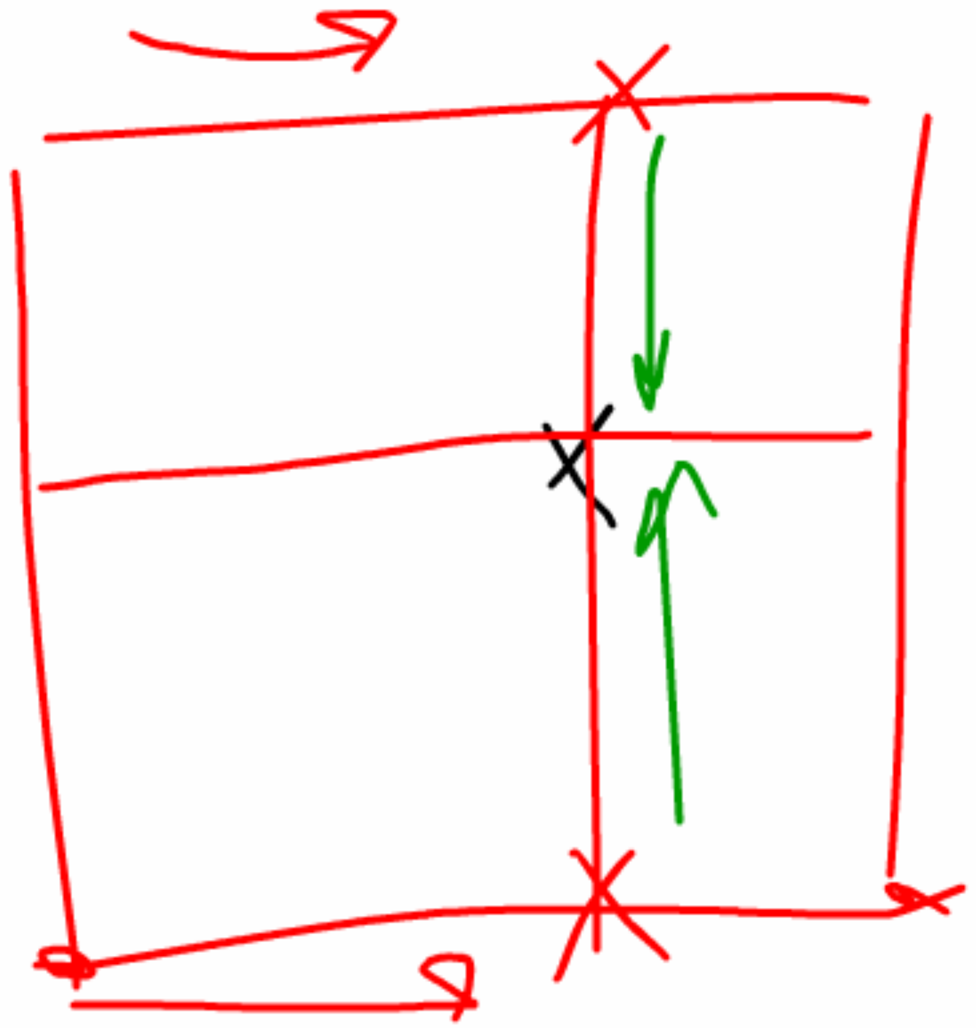
$$z(x,y) = (1-u)(1-v) z_{ij} + u(1-v) z_{(i+1)j} + (1-u) v z_{i(j+1)} + uv z_{(i+1)(j+1)}$$

- Bilinear alternative 2

$$z_{bottom} = (x_{i+1} - x) z_{ij} + (x - x_i) z_{(i+1)j}$$

$$z_{top} = \ldots$$

$$z(x,y) = \frac{(y_{j+1} - y) z_{bottom} + (y - y_j) z_{top}}{y_{j+1} - y_j}$$



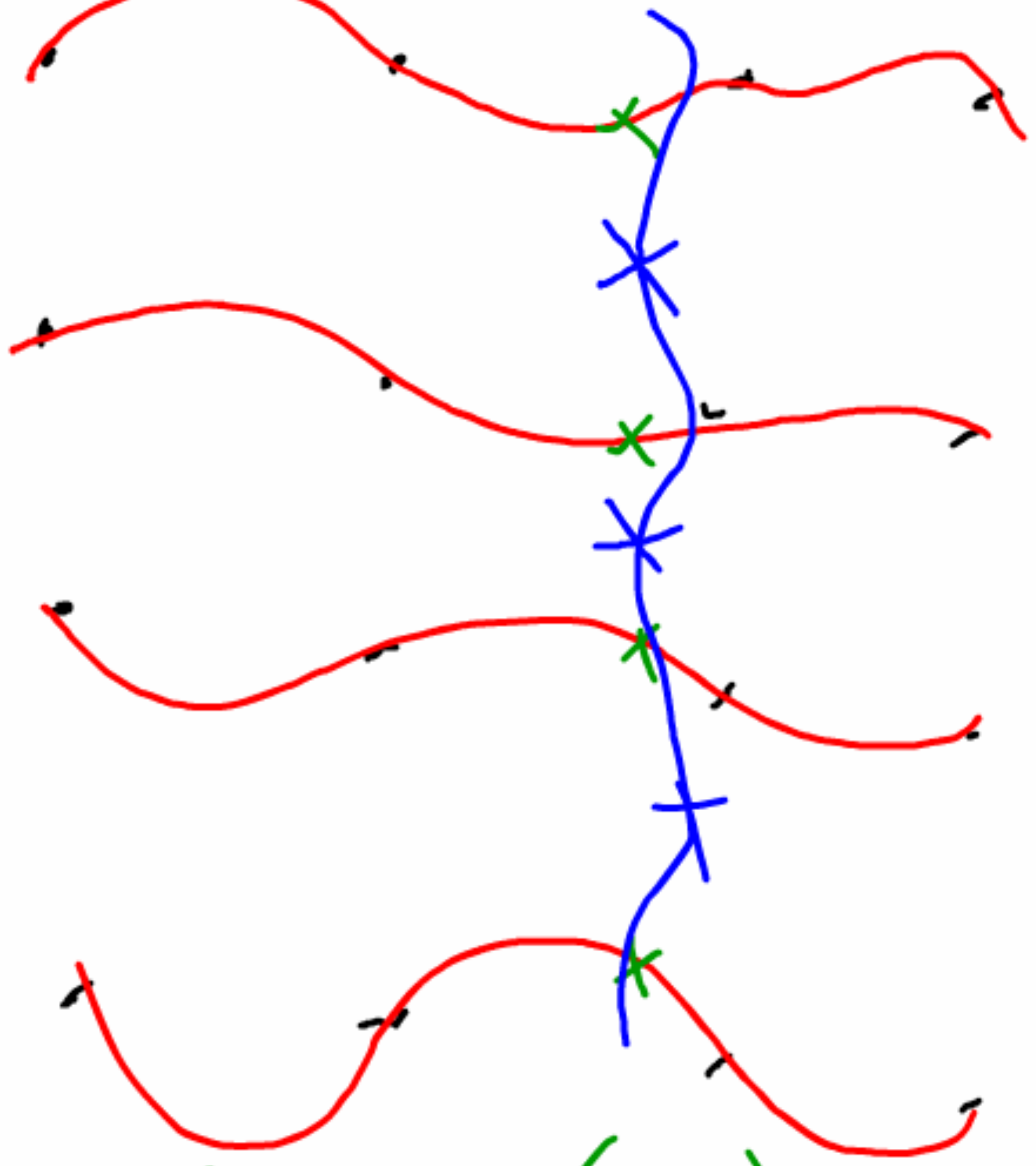
Interpolation in higher dimensions

- Beyond 2D/bilinear

E.g. 3D

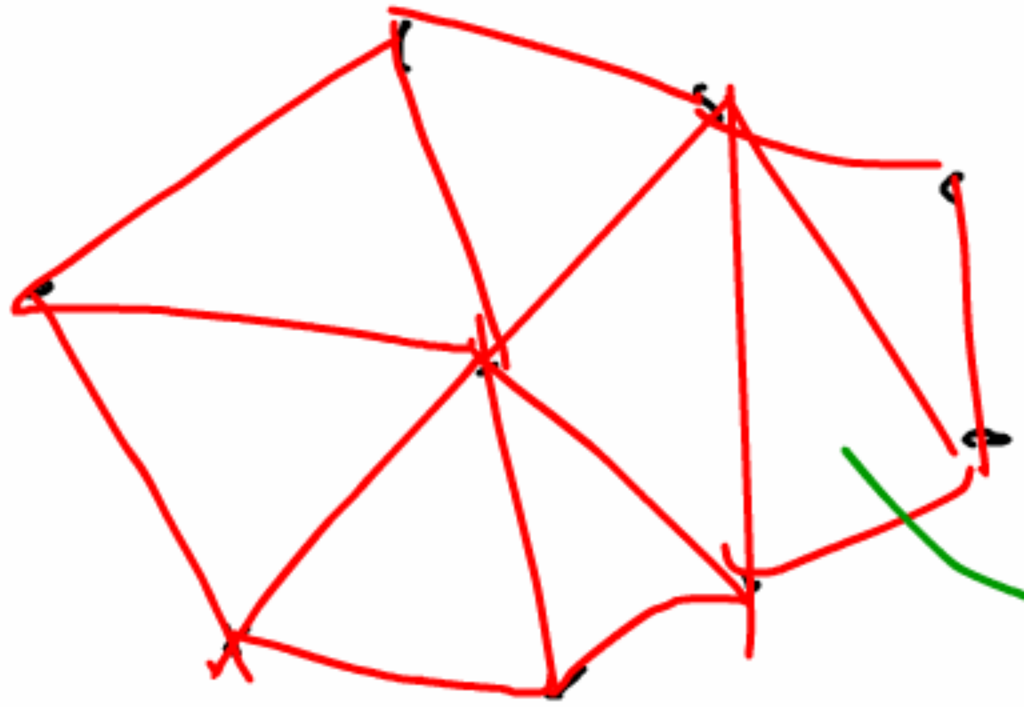
5 1D cubic interp.!

bicubic (cubic in 2D) →



(3D: 21 1D interp.)

- Gridless in 2D



3 points → 3 constr. ⇒ $z(x,y) = a_{ij} + b_{ij}(x - x_i) + c_{ij}(y - y_j)$

Next week:
Solving linear equations