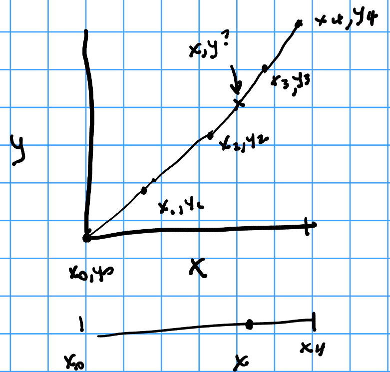
Lecture 18 – Interpolation

I. Concept of Interpolation

II. Linear interpolation by hand / Excel

III. Interpolation in Python

I. Concept of Interpolation



$\{x_i, y_i\}$ data

$y(x)$ not in our set.

interpolation, but $x \in [\min, \max]$

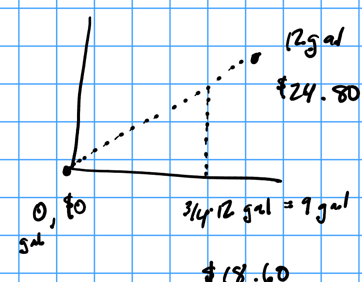
extrapolation $x \notin [\min, \max]$

Example: \$24.80 to fill up car w/ gas

12 gal tank.

• suppose 1/4 full

↳ fill 3/4



In interpolation, you want smooth functions that go through all of the data. ⇒ piecewise functions

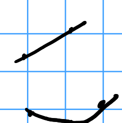
Piecewise polynomials

- why polynomials?
  - simple : lines, quadratic, ...
  - Polynomial Uniqueness Theorem

    2 points → 1 line

    3 points → 1 quadratic

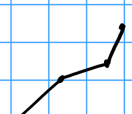
    4 pts → 1 cubic



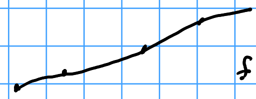
- why piecewise?
  - wild oscillations – Runge's Phenomena

* A piecewise polynomial that exactly fits every data point is called a spline.

- linear spline



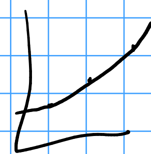
- quadratic spline (2^{nd}, quadratics)
- cubic spline (3^{rd}, cubic)



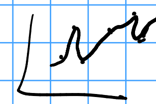
f, f', f'' continuous

Fitting vs. Interpolation

single EQ/model	piecewise polynomials
Physically informed	purely empirical
Noisy Data	Smooth data
Approx. fit (only some/no pts)	Exact fit (every point)

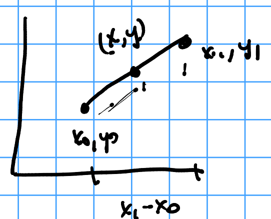


$x = vt$



II. Interpolation by Hand & Excel

* Linear Interpolation (Excel no interpolation functions ¨̮)



$x_0 \;\; y_0$

$x_1 \;\; y_1$

$x \rightarrow y$

$$\frac{y-y_0}{x-x_0} = \frac{y_1-y_0}{x_1-x_0} \quad \text{(equal slopes)}$$

$$y - y_0 = (x-x_0)\frac{y_1-y_0}{x_1-x_0}$$

$$y = y_0 + (x-x_0)\frac{(y_1-y_0)}{x_1-x_0} \quad ✓ ¨̮$$

$$\frac{(x_1-x_0)}{x_1-x_0}$$

$$= y_0\frac{x_1-x_0}{x_1-x_0} + y_1\frac{x-x_0}{x_1-x_0} - y_0\frac{x-x_0}{x_1-x_0}$$

$$\boxed{y = y_0\frac{x_1-x}{x_1-x_0} + y_1\frac{x-x_0}{x_1-x_0}}$$

lever rule (use again in thermo)

Example from page 1:

$x_0 = 0 \qquad x_1 = 12\text{gal}$

$y_0 = \$0 \qquad y_1 = \$24.80$

$x = \frac{3}{4}\cdot 12\text{gal} = 9\text{gal}$

$$y = \$0\cdot\cancel{\frac{12-9}{12}} + \$24.80\cdot\frac{9-0}{12}$$

$$+ \frac{3}{4}\cdot\$24.80 = \$18.60$$

III. Interpolation in Python

scipy.optimize → root, minimize, curve_fit

scipy.interpolate