ENGR 21:
Computer Engineering Fundamentals

Instructor: Emad Masroor

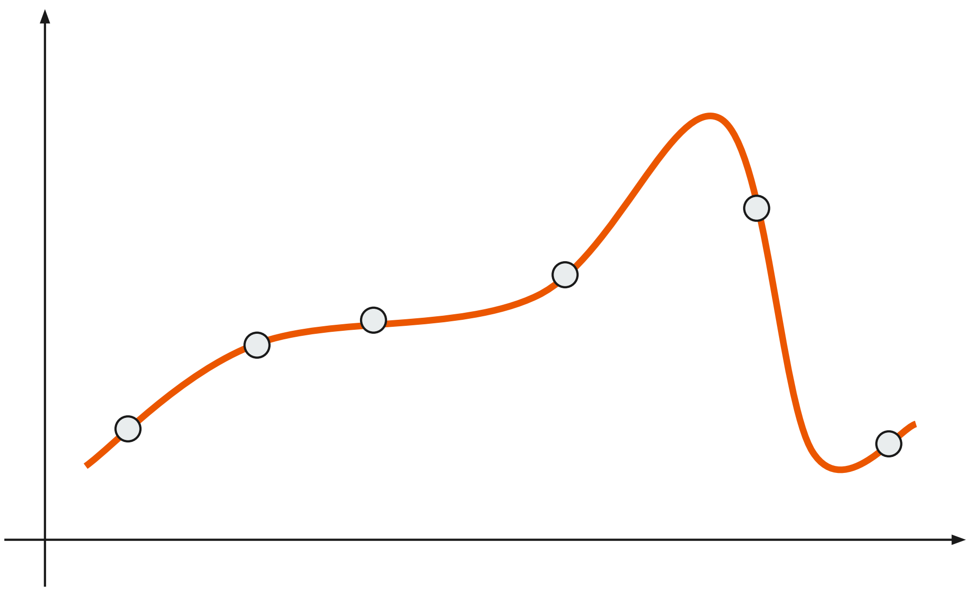
Lecture 22
Thursday, November 20, 2025

Curve Fitting

Curve Fitting vs Interpolation

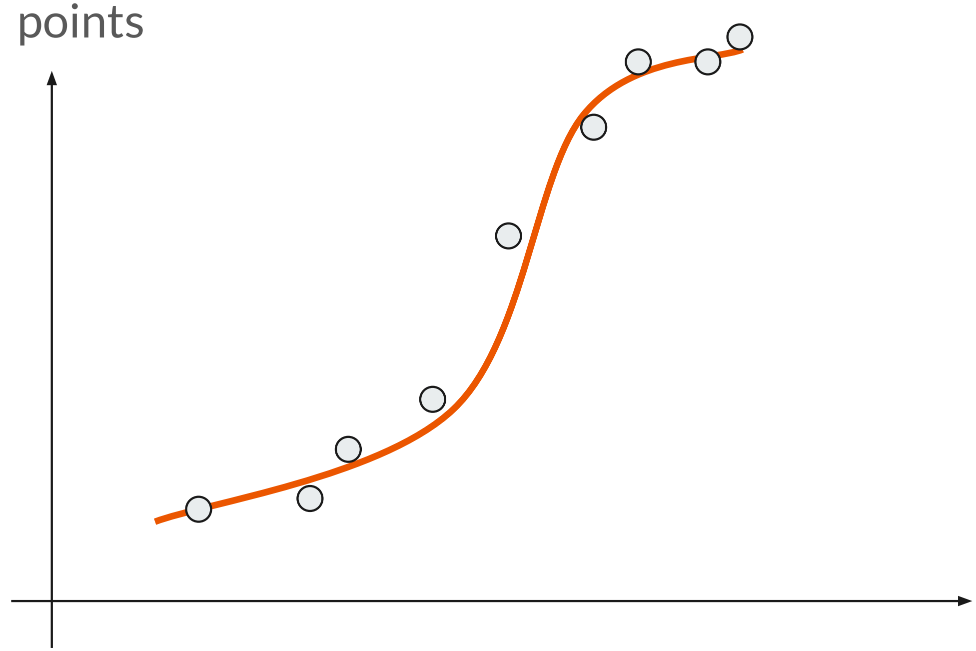
Interpolation

Finding a curve (i.e., a function) that connects all your data points



Curve Fitting

Finding a curve (i.e., a function) that **best represents** the trend shown by your data points



Curve Fitting, Interpolation, and Polynomials

- In general, we could fit ~~a curve~~ data to many kinds of functions:
  $f(x) = a_0 f_0(x) + a_1 f_1(x) + a_2 f_2(x) + \ldots$ where $f_1(x)$, $f_2(x)$, ... are some functions, like $\sin x$, e^x etc
- Polynomials are a special case of the above, where
  $f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \ldots$ where $f_0 = x^0$, $f_1 = x^1$, $f_2 = x^2$ ......
- Interpolation and curve-fitting can both be done using polynomial functions

e.g.

Find a cubic polynomial that passes through the points (0,0), (1,3), (3,7), (5,2)

If you have 4 points, then the only type of polynomial that will work is a cubic polynomial.

e.g.

Find a quadratic curve that ~~passes through the points~~ "best represents" (0,0), (1,3), (3,7), (5,2)

If you have 4 points, you can 'fit' to a straight line, a quadratic, or a cubic polynomial

Curve Fitting & Interpolation with non-polynomial functions

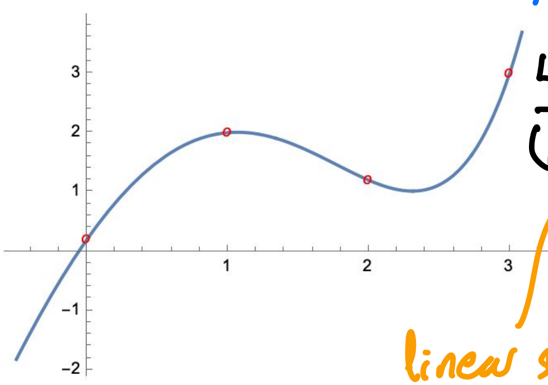
- Polynomials are just one type of function among many
  $$f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \ldots$$
- Interpolation and curve-fitting can both be done using other kinds of functions

free parameters

e.g.

Find the parameters {a,b,c,d} such that
$f(x) = a + be^x + cx^2 + d\sin x$
Passes through (0,0.2), (1,2), (2,1.2), (3,3)

f happens to be linear in {a,b,c,d}



4 points
$\hookrightarrow$ 4 eqns for
4 unknowns
{a,b,c,d}

linear system

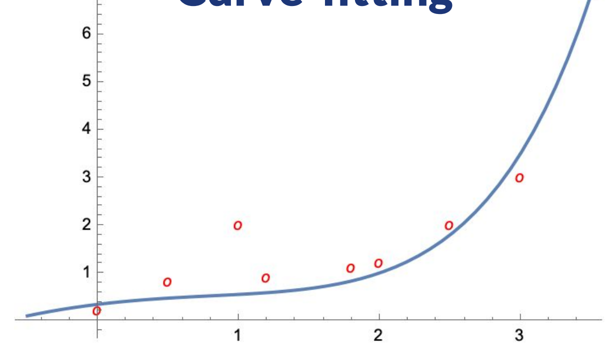
Interpolation

e.g.

Find the parameters {a,b} such that
$f(x) = ae^x + bx^2$
best captures the trend of the points shown

happens to be linear in {a,b}

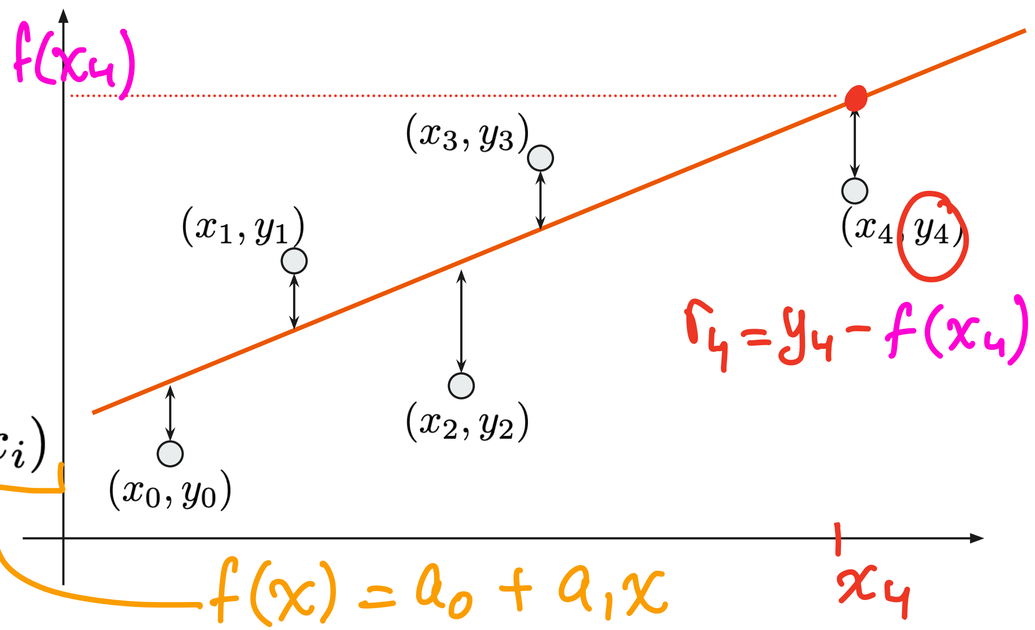
Curve-fitting



'Least squares'

The simplest kind of curve fitting: straight line

- Find a straight line that is as close as possible to the data points
- Minimize the vertical distance between the data points and the line
- Define a residual $r_i = y_i - f(x_i)$
- Find $f(x)$ such that $\sum_{i=0}^{n-1} r_i^2$ is minimized



$f(x) = a_0 + a_1 x$

Find a straight line $y = a_0 + a_1 x$ that best represents the data shown

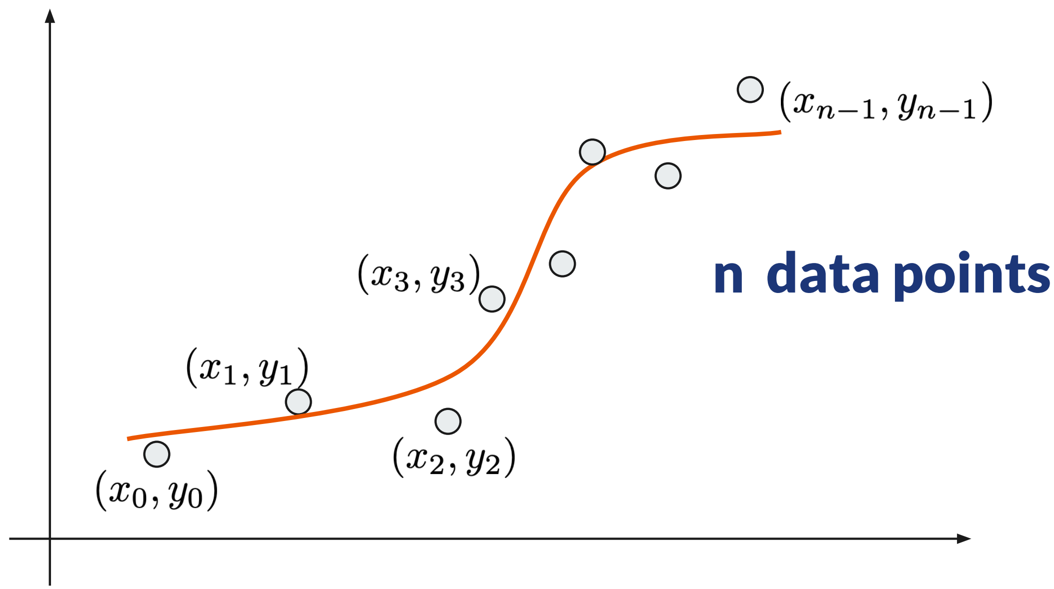
Minimize some OBJECTIVE FUNCTION over the parameters $\{a_0, a_1\}$

Fitting data to an arbitrary curve

- Find the parameters $\{a_0, a_1, a_2, \ldots, a_m\}$ that minimize the objective function

$$\boxed{S = \sum_{i=0}^{n-1} \left[y_i - f(x_i)\right]^2}$$

objective function for 'least squares fitting' to any function f(x).



$$\boxed{f(x) = a_0 + a_1 f_1(x) + a_2 f_2(x) + \ldots + a_m f_m(x)}$$

m+1 free parameters in this **f(x)**. For a straight line function, m = 1 and there are 1+1 free parameters.

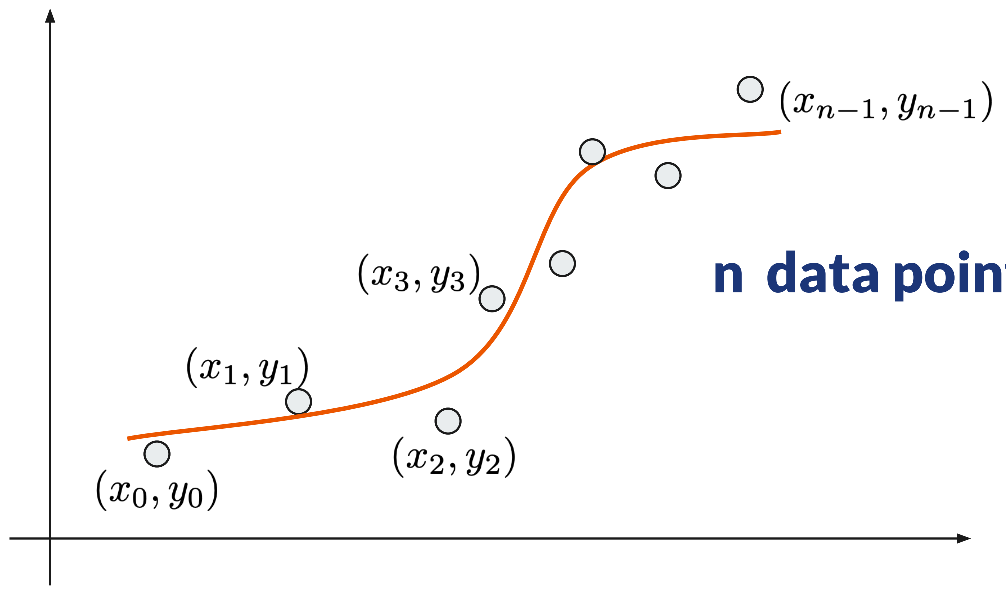
Fitting data to an arbitrary polynomial curve

- Find the parameters $\{a_0, a_1, a_2, \ldots, a_m\}$ that minimize the objective function

$$\underbrace{S = \sum_{i=0}^{n-1} [y_i - f(x_i)]^2}_{\text{Take derivatives of } S \text{ with respect to } a_0,\, a_1,\, a_2,\, \ldots,\, a_m \text{ and equate to zero.}}$$

Take derivatives of S with respect to $a_0, a_1, a_2, \ldots, a_m$ and equate to zero.

↳ If $f(x)$ is linear in $\{a_0, a_1, a_2, \ldots, a_m\}$ then this system of equations will be linear. $[\ \ ][\ ] = [\ ]$



n data points

$$\underbrace{f(x)}_{\text{yes linear in } a\text{'s.}} = a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^m$$

Polynomial of order m

How do we pick $\{a_0, a_1, a_2, \dots, a_m\}$ that minimize S

$$\frac{\partial S}{\partial a_1} = ?$$

polynomial order m.

minimize $$S = \sum_{i=0}^{n-1} [y_i - \overbrace{f(x_i)}]^2$$

$$f(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_m x^m$$

$$S = \sum_{i=0}^{n-1} [y_i - \underbrace{a_0 - a_1 x_i - a_2 x_i^2 - a_3 x_i^3 - \dots - a_m x_i^m}]^2$$

$$y_i - \left(\sum_{j=0}^{j=m} a_j (x_i)^j \right)$$

$$\boxed{2 \sum_{i=0}^{n-1} [y_i - a_0 - a_1 x_i - a_2 x_i^2 - \dots - a_m x_i^m]^{2-1} * (-x_i)}$$

write $\frac{\partial S}{\partial a_1} = 0$, similar for other a's.

Example: Finding a quadratic polynomial to fit n data points

→ 3 parameters {a_0 a_1 a_2}

$$\frac{\partial S}{\partial a_0} = -2\sum_{i=0}^{n-1}\left[y_i - a_0 - a_1x_i - a_2x_i^2\right] = 0$$

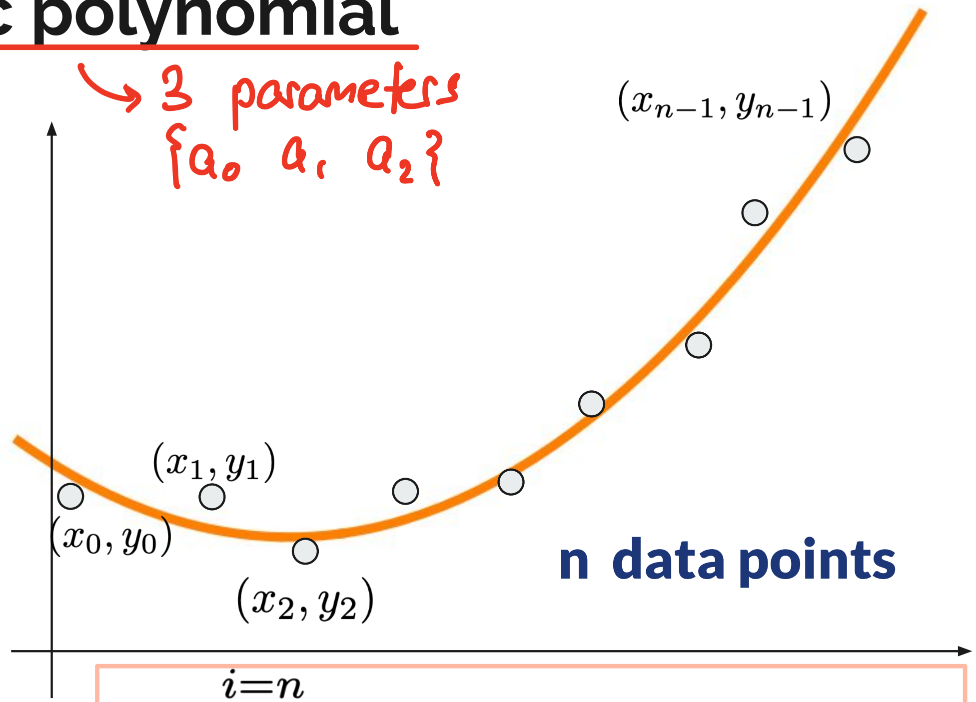
$$\implies \sum_{i=0}^{n-1}\left(a_0 + a_1x_i + a_2x_i^2\right) = \sum_{i=0}^{n-1} y_i$$

$$\frac{\partial S}{\partial a_1} = -2\sum_{i=0}^{n-1}x_i\left[y_i - a_0 - a_1x_i - a_2x_i^2\right] = 0$$

$$\implies \sum_{i=0}^{n-1}\left(a_0x_i + a_1x_i^2 + a_2x_i^3\right) = \sum_{i=0}^{n-1} y_ix_i$$

$$\frac{\partial S}{\partial a_2} = -2\sum_{i=0}^{n-1}x_i^2\left[y_i - a_0 - a_1x_i - a_2x_i^2\right] = 0$$

$$\implies \sum_{i=0}^{n-1}\left(a_0x_i^2 + a_1x_i^3 + a_2x_i^4\right) = \sum_{i=0}^{n-1} y_ix_i^2$$



$$\boxed{S = \sum_{i=0}^{i=n}\left[y_i - a_0 - a_1x_i - a_2x_i^2\right]^2}$$

Goal: Solve 3 equations for 3 unknowns.

Rearrange!

The linear system of equations for finding the best-fit polynomial of order m for n data points

unknowns   knowns {from data points}

$$\sum_{i=0}^{n-1}\left(a_0 x_i^0 + a_1 x_i^1 + a_2 x_i^2\right) = \sum_{i=0}^{n-1} y_i x_i^0$$

$$\sum_{i=0}^{n-1}\left(a_0 x_i^1 + a_1 x_i^2 + a_2 x_i^3\right) = \sum_{i=0}^{n-1} y_i x_i^1$$

$$\sum_{i=0}^{n-1}\left(a_0 x_i^2 + a_1 x_i^3 + a_2 x_i^4\right) = \sum_{i=0}^{n-1} y_i x_i^2$$

depend on $\{x_i, y_i\}$

$$\underbrace{\begin{bmatrix} \Sigma x_i^0 & \Sigma x_i^1 & \Sigma x_i^2 \\ \Sigma x_i^1 & \Sigma x_i^2 & \Sigma x_i^3 \\ \Sigma x_i^2 & \Sigma x_i^3 & \Sigma x_i^4 \end{bmatrix}}_{\text{depend on } x_i \text{ only}} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix}$$

$m = 2$

The linear system of equations for finding the best-fit polynomial of order m for n data points

$$\begin{bmatrix} \sum x_i^0 & \sum x_i^1 & \sum x_i^2 & \cdots & \sum x_i^m \\ \sum x_i^1 & \sum x_i^2 & \sum x_i^3 & \cdots & \sum x_i^{m+1} \\ \sum x_i^2 & \sum x_i^3 & \sum x_i^4 & \cdots & \sum x_i^{m+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sum x_i^m & \sum x_i^{m+1} & \sum x_i^{m+2} & \cdots & \sum x_i^{m+m} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_m \end{bmatrix} = \begin{bmatrix} \sum x_i^0 y_i \\ \sum x_i^1 y_i \\ \sum x_i^2 y_i \\ \vdots \\ \sum x_i^m y_i \end{bmatrix}$$

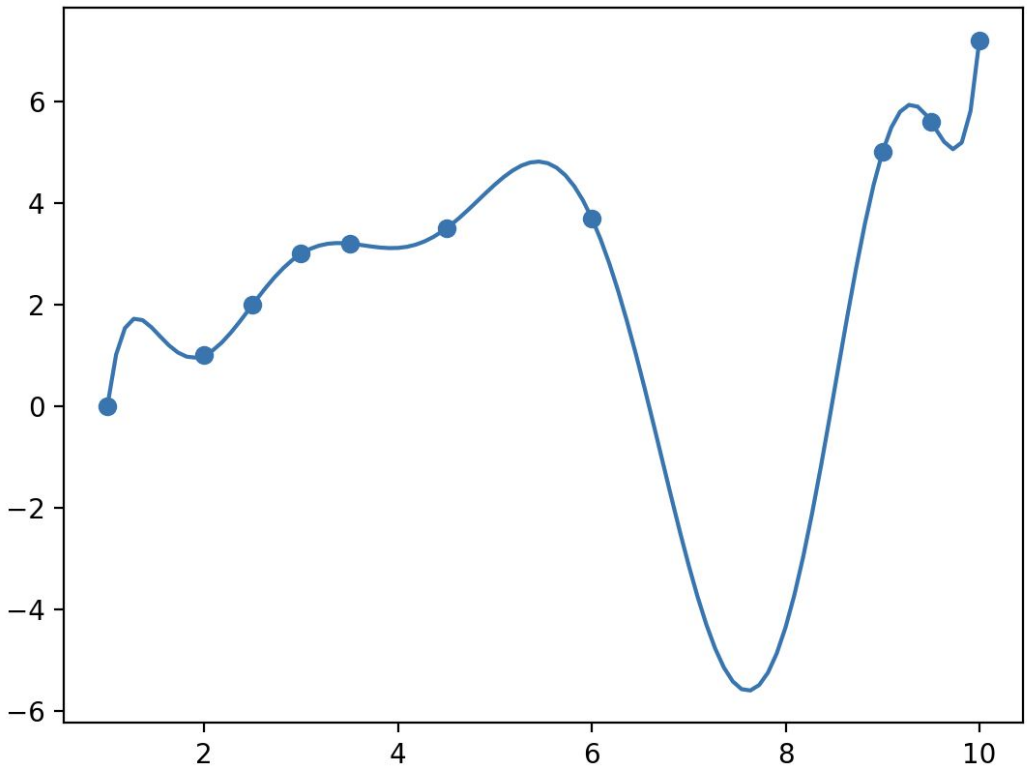
Symmetric

powers $2m$

If m is large $\longrightarrow$ ill-conditioned!

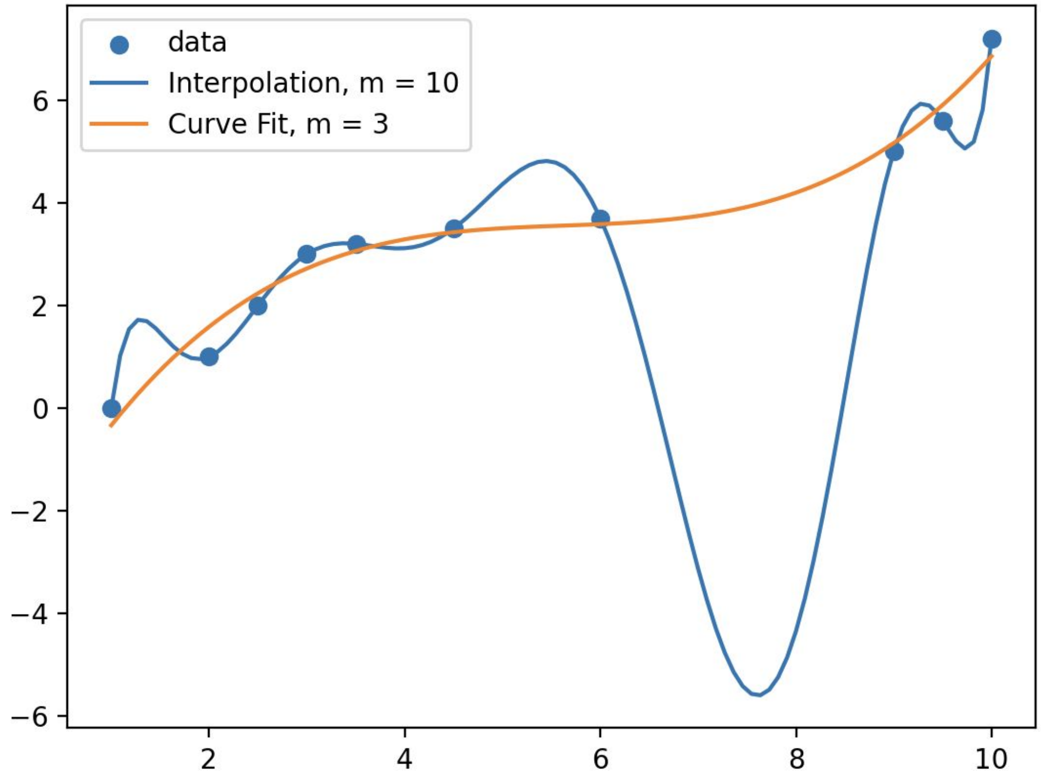
Sometimes, interpolation leads to "overfit"
And a low-order, "relaxed" curve fit suits the data better

Points from previous lecture



Sometimes, interpolation leads to “overfit”
And a low-order, “relaxed” curve fit suits the data better

Points from previous lecture



Python function for polynomial curve fitting

Write a Python program that reads these data points from a file and fits them to a quadratic polynomial.

Steps:

1. Read the data points as numpy arrays of x and y values
2. Assemble the matrix using the x values
3. Assemble the right hand side using the x and y values
4. Solve the linear system of equations Ax=b using your favorite linear solver
5. You now have the 3 coefficients a0, a1 and a2 that define the quadratic polynomial you were looking for!

Further steps:

1. Use the coefficients to actually plot the polynomial curve
2. Overlay on the points to see how well it fits

datapoints.txt

```
0.0000 2.0000
0.0500 1.8625
0.1000 1.7500
0.1500 1.6625
0.2000 1.6000
0.2500 1.5625
0.3000 1.5500
0.3500 1.5625
0.4000 1.6000
0.4500 1.6625
0.5000 1.7500
0.5500 1.8625
0.6000 2.0000
0.6500 2.1625
0.7000 2.3500
0.7500 2.5625
0.8000 2.8000
0.8500 3.0625
0.9000 3.3500
0.9500 3.6625
```

ENGR 21 Fall 2025

Resources

Lec 11.1, Tue, Nov 18

Interpolate these points: download file

Lec 11.2, Thu, Nov 20

Curve-fit these points: download file