

Artificial Intelligence

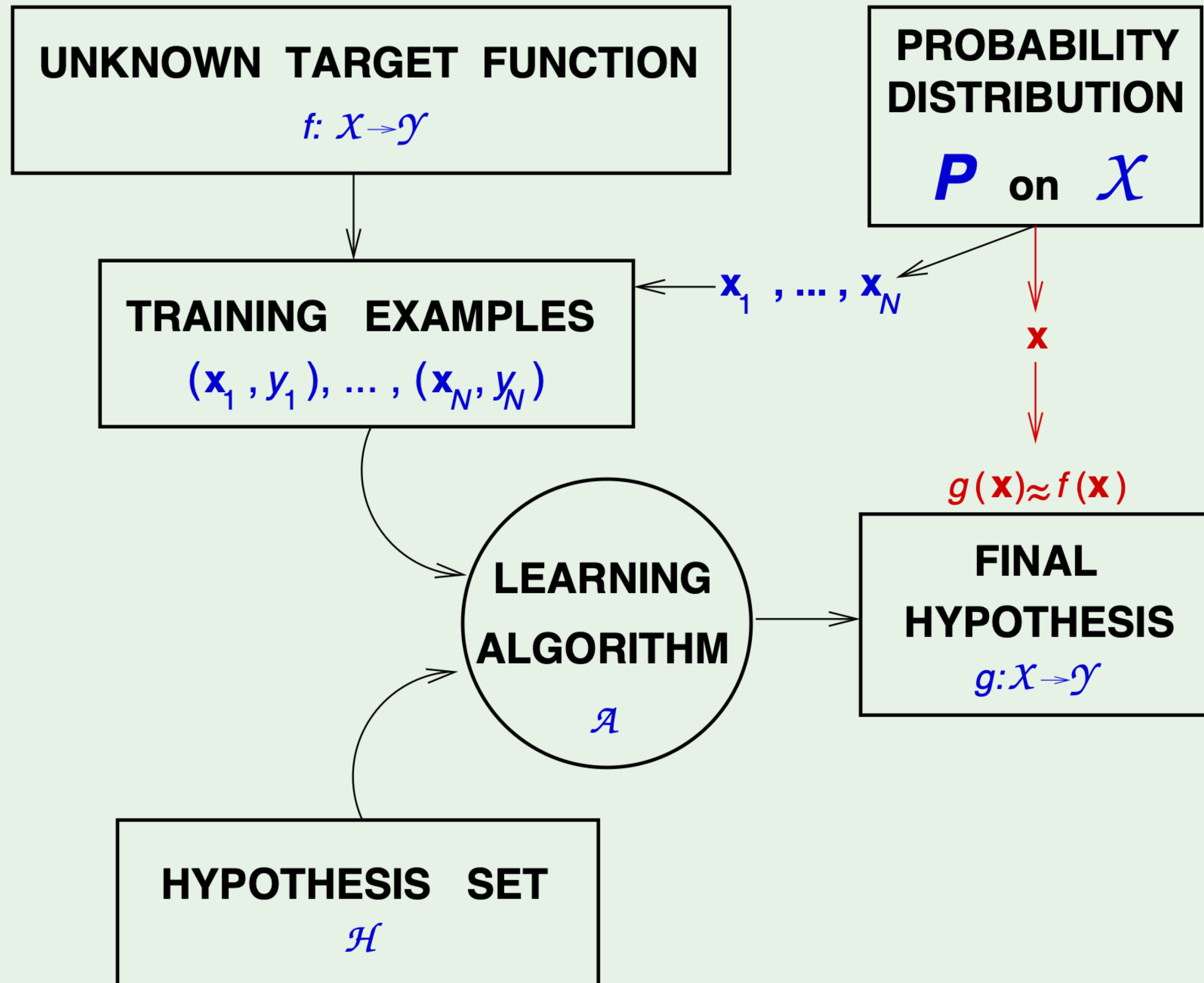
CSC 665

tyler dae devlin

Machine Learning V

11.28.2023

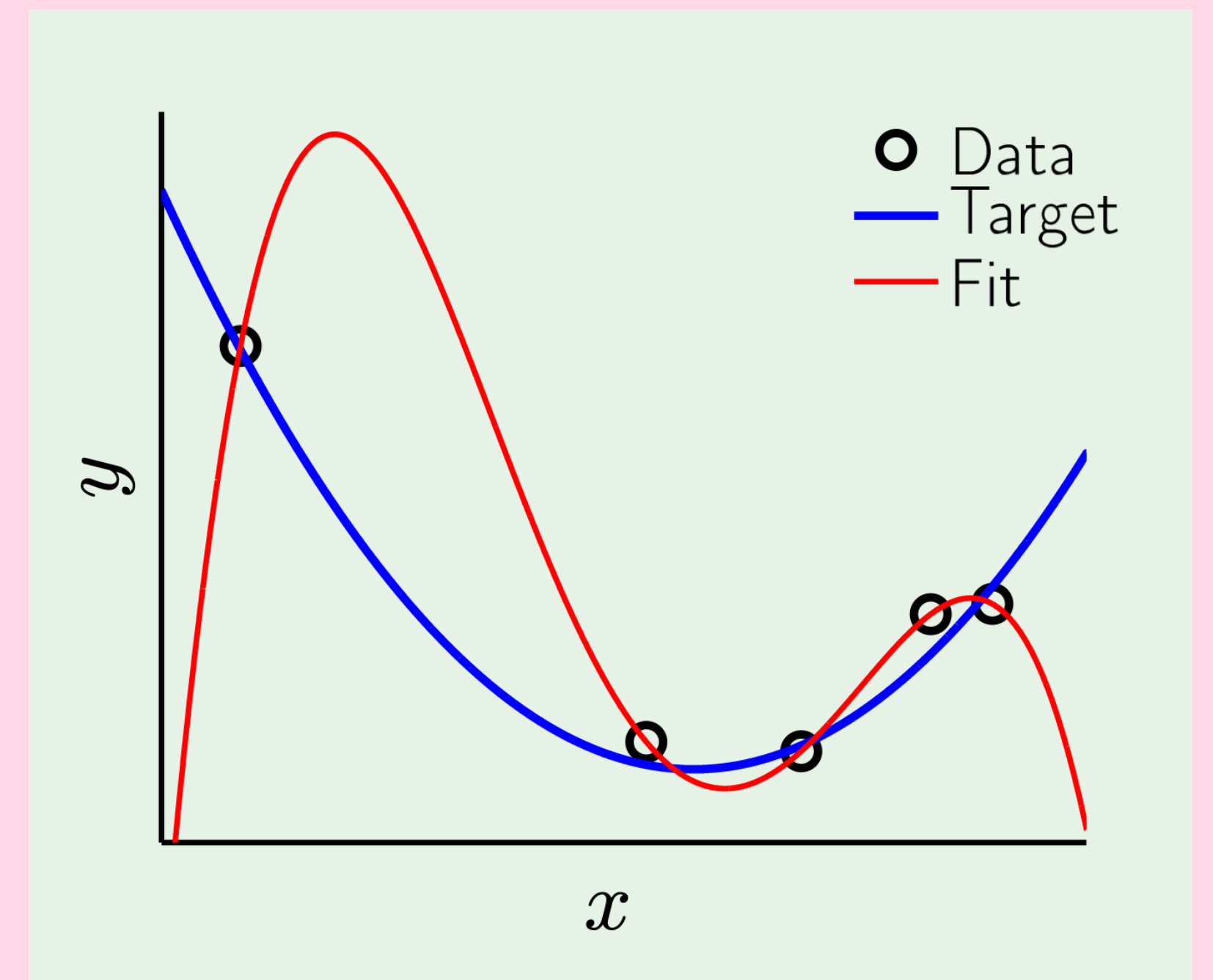
- **Search:** make decisions by looking ahead
- **Logic:** deduce new facts from existing facts
- **Constraints:** find a way to satisfy a given specification
- **Probability:** reason quantitatively about uncertainty
- **Learning:** make future predictions from past observations

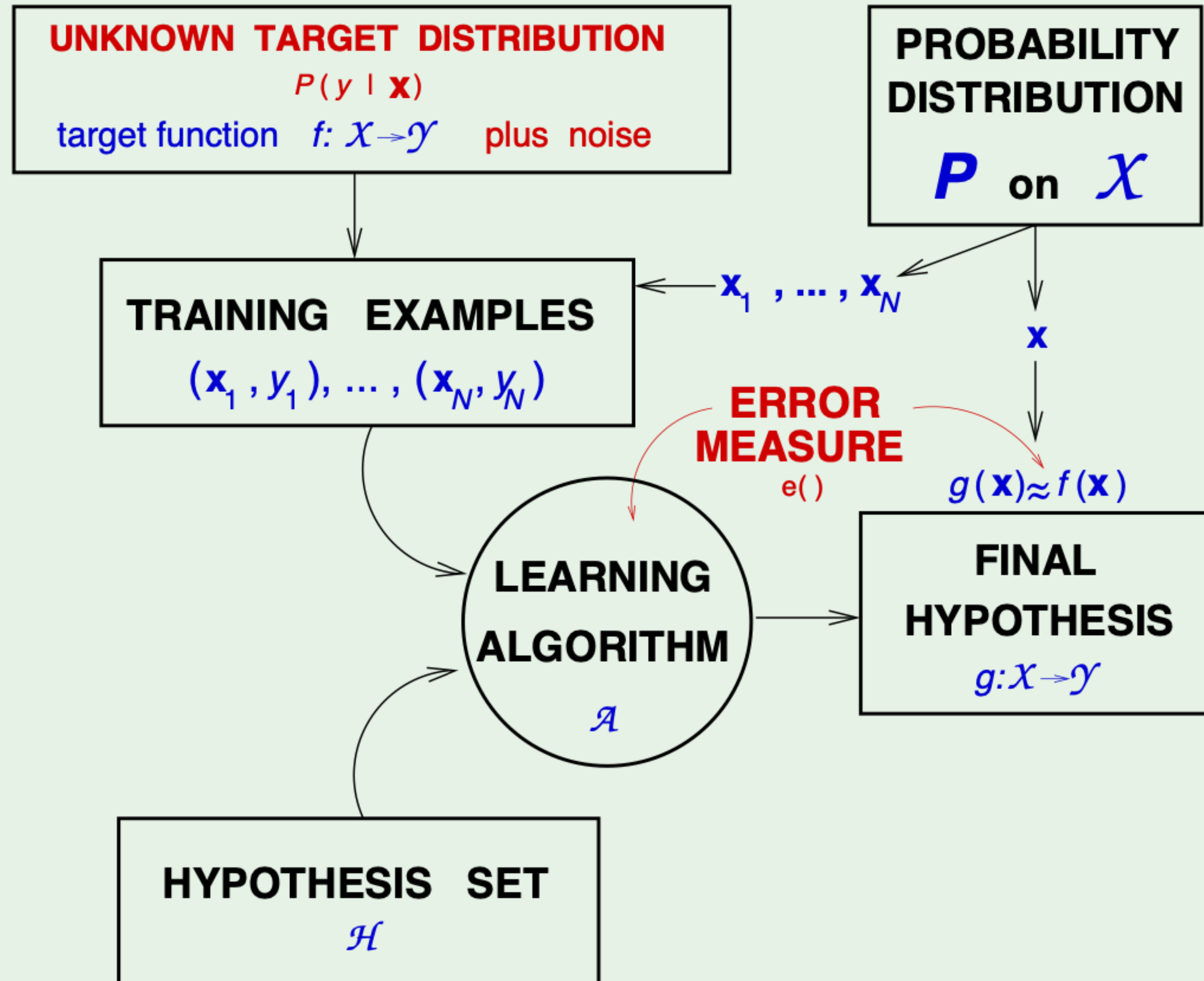


Overfitting

Overfitting by example

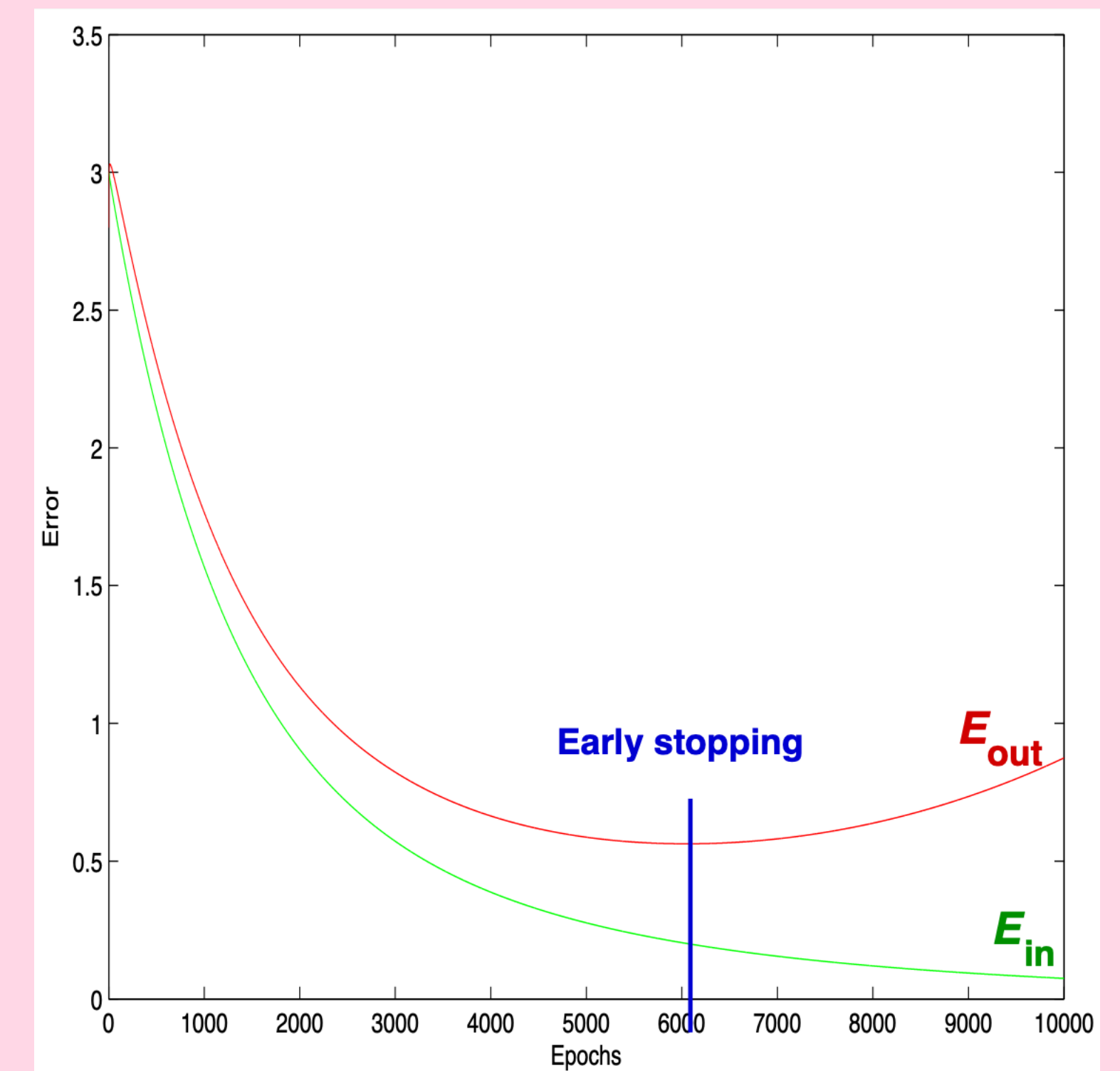
- In practice, datasets often contains **noise**
- $y = f(x) + \varepsilon$ ($\varepsilon \sim P_{\text{noise}}$)
- On the right:
 - f is a 2nd degree polynomial
 - $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ for small σ
- Since the training dataset only contains 5 points, a 4th degree polynomial **fits perfectly**
- I.e. $C_{\text{train}} = 0$ but C_{test} will be **large**
- Our hypothesis failed to generalize because *it fits the noise*





Another example

- Consider training a neural network by stochastic gradient descent
- An **epoch** is a pass through the entire training dataset
- After each epoch, measure error/loss/cost of current hypothesis on both training dataset (E_{in}) and held-out test set (E_{out})
- Before blue line: **fitting**
- After blue line: **overfitting**
- (In many modern deep learning models this figure is incomplete due to “double descent”)

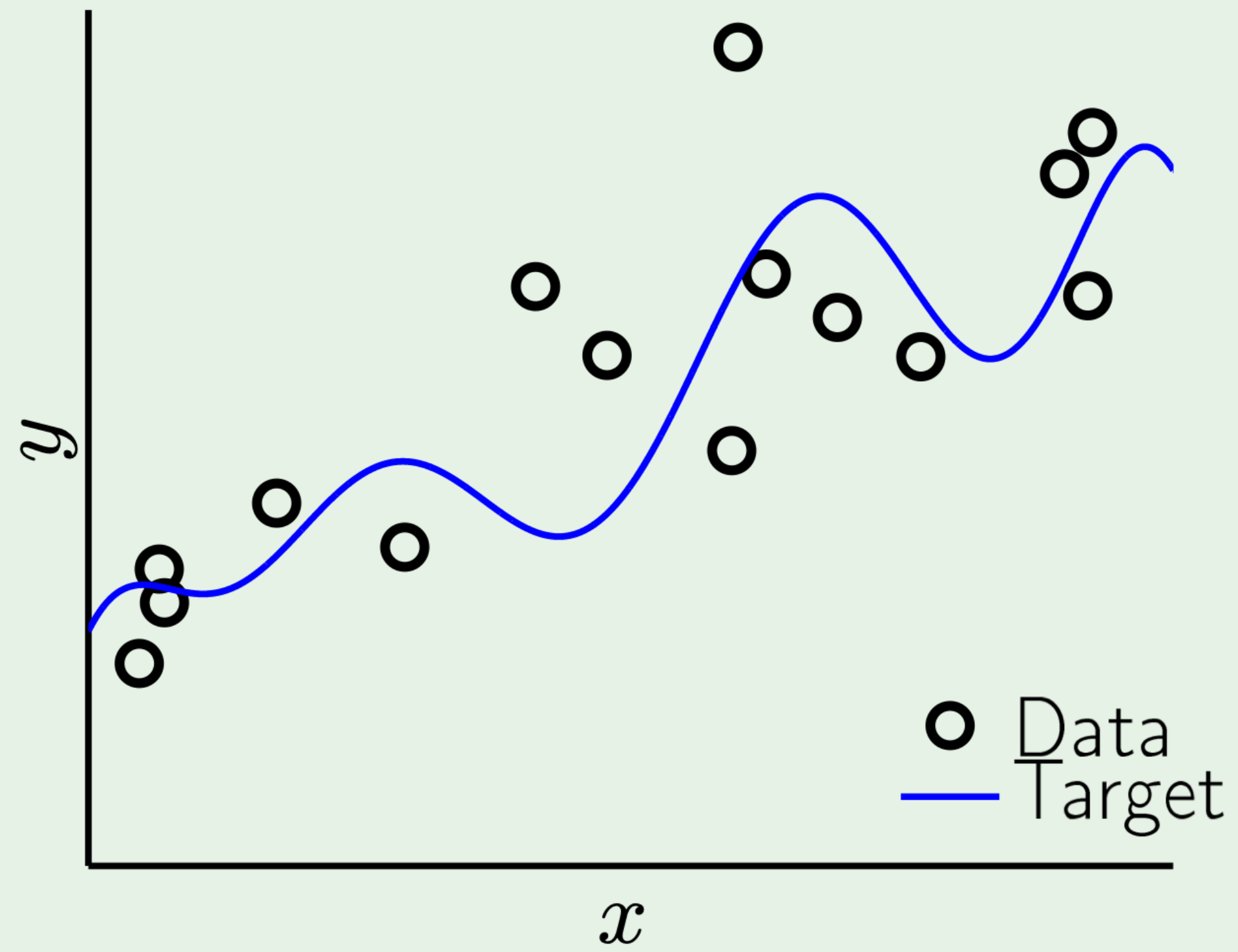


A simple definition

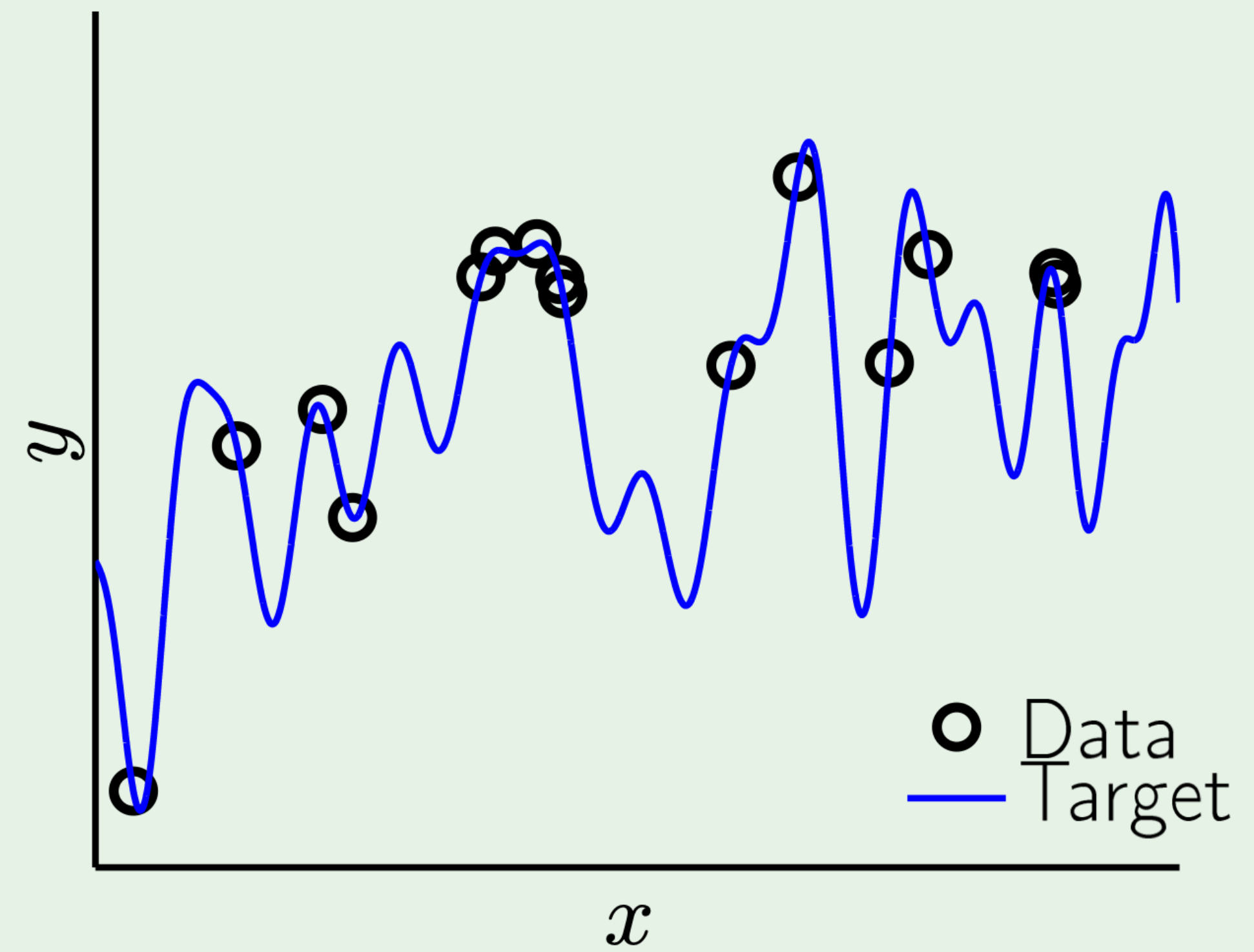
- Overfitting happens when you have **low training cost, but high test cost**
- Overfitting means your hypothesis **fails to generalize** to new unseen examples
- Overfitting happens because you have fit the data more than is warranted — you are **fitting the noise**

Case study

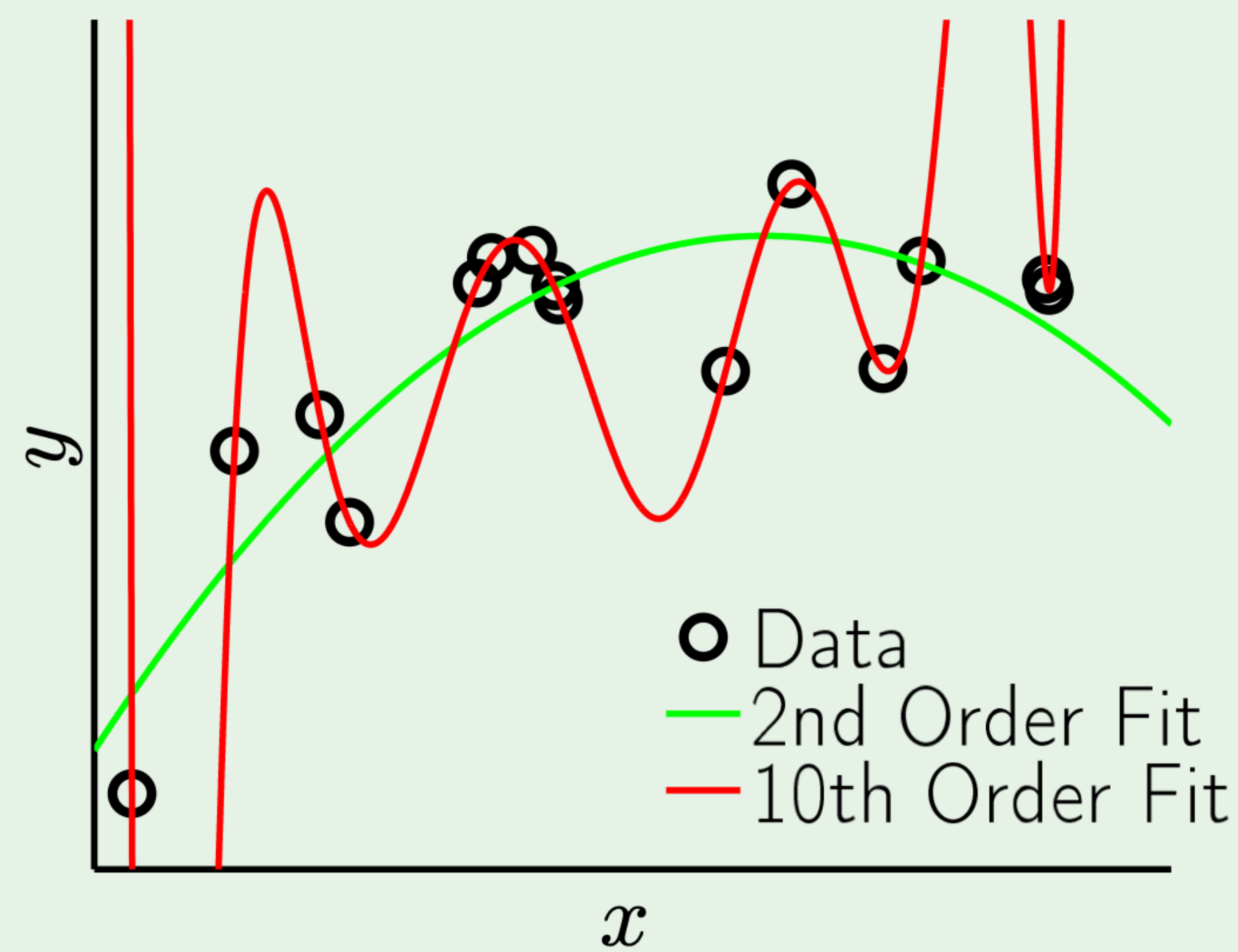
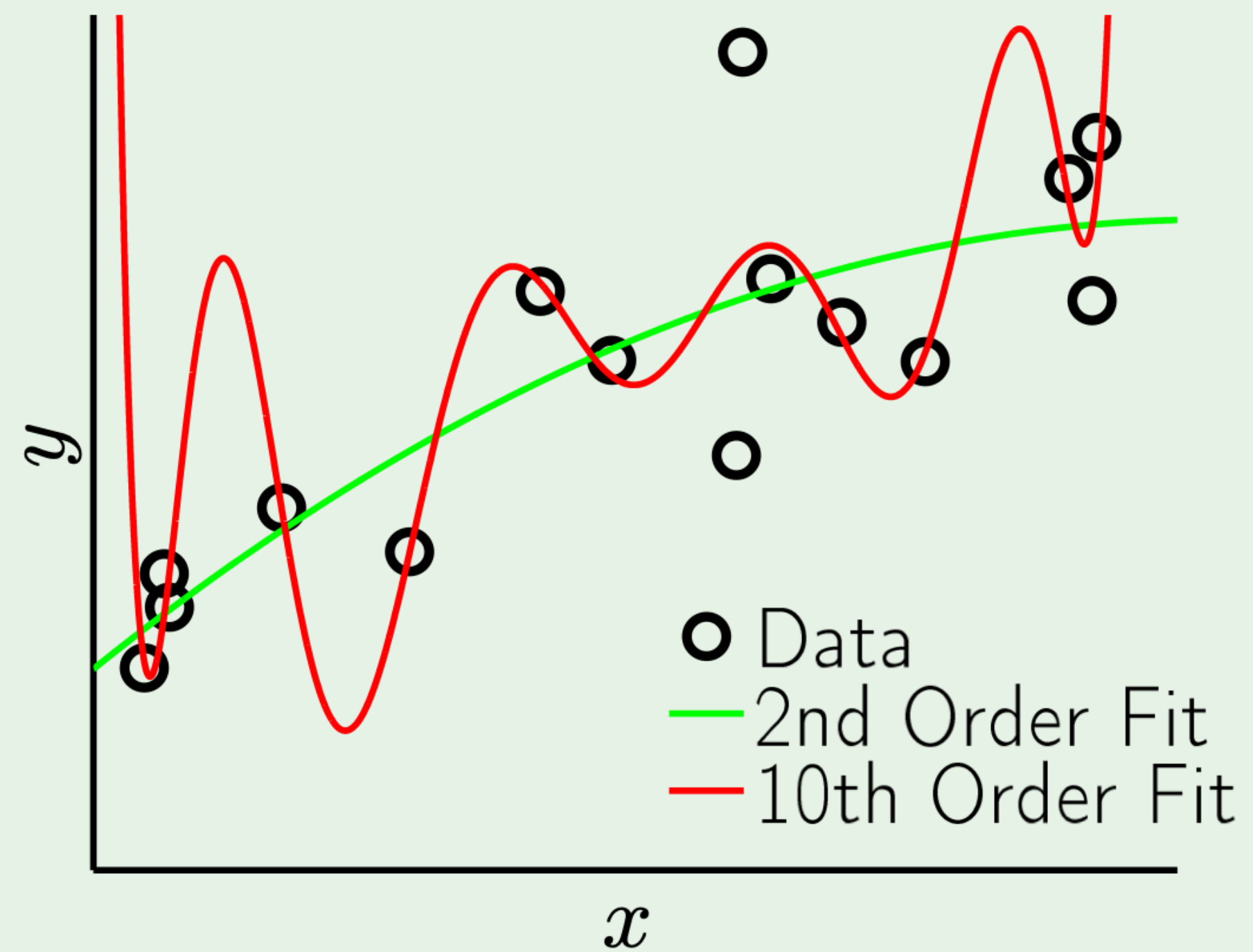
10th-order target + noise



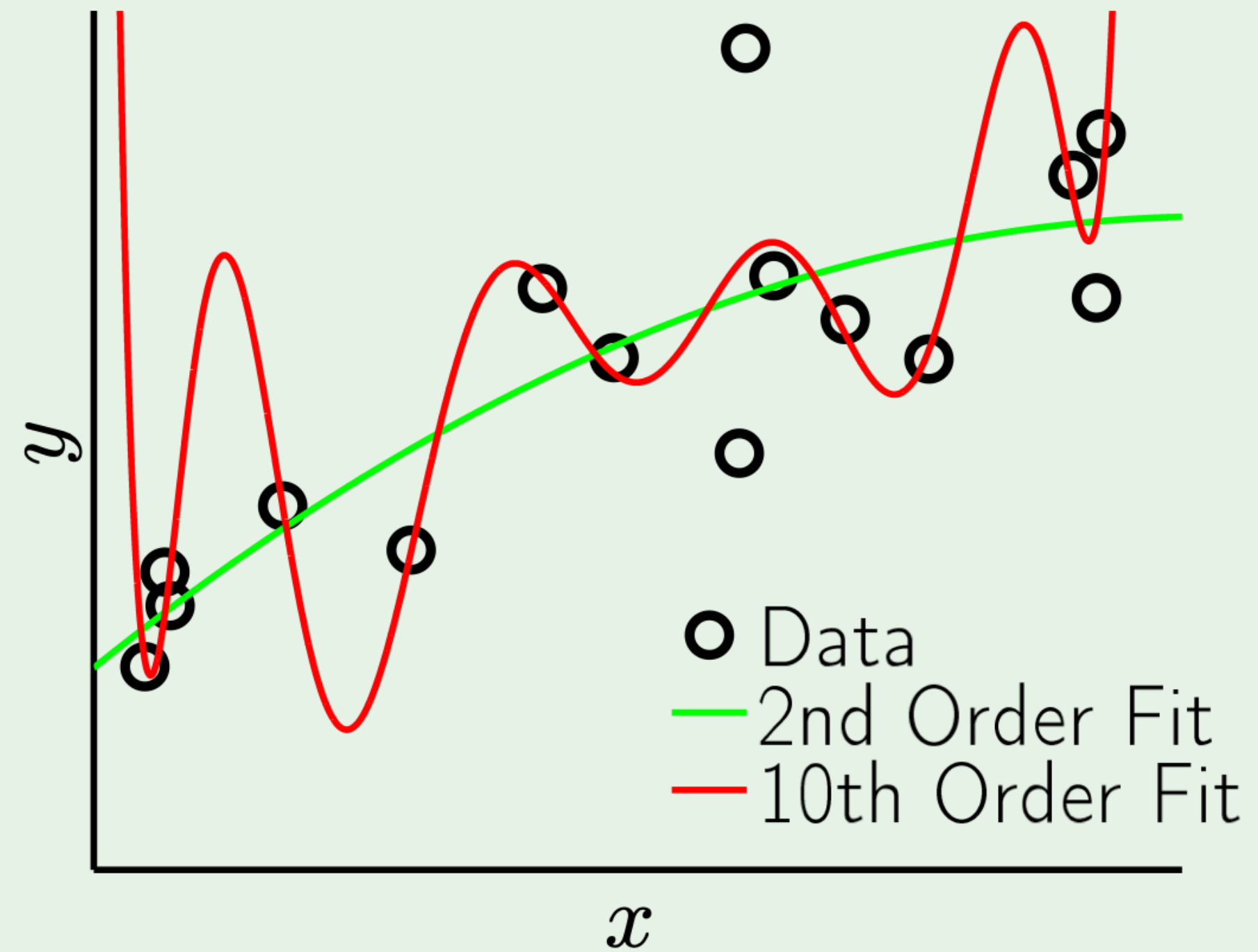
50th-order target



Two fits for each target

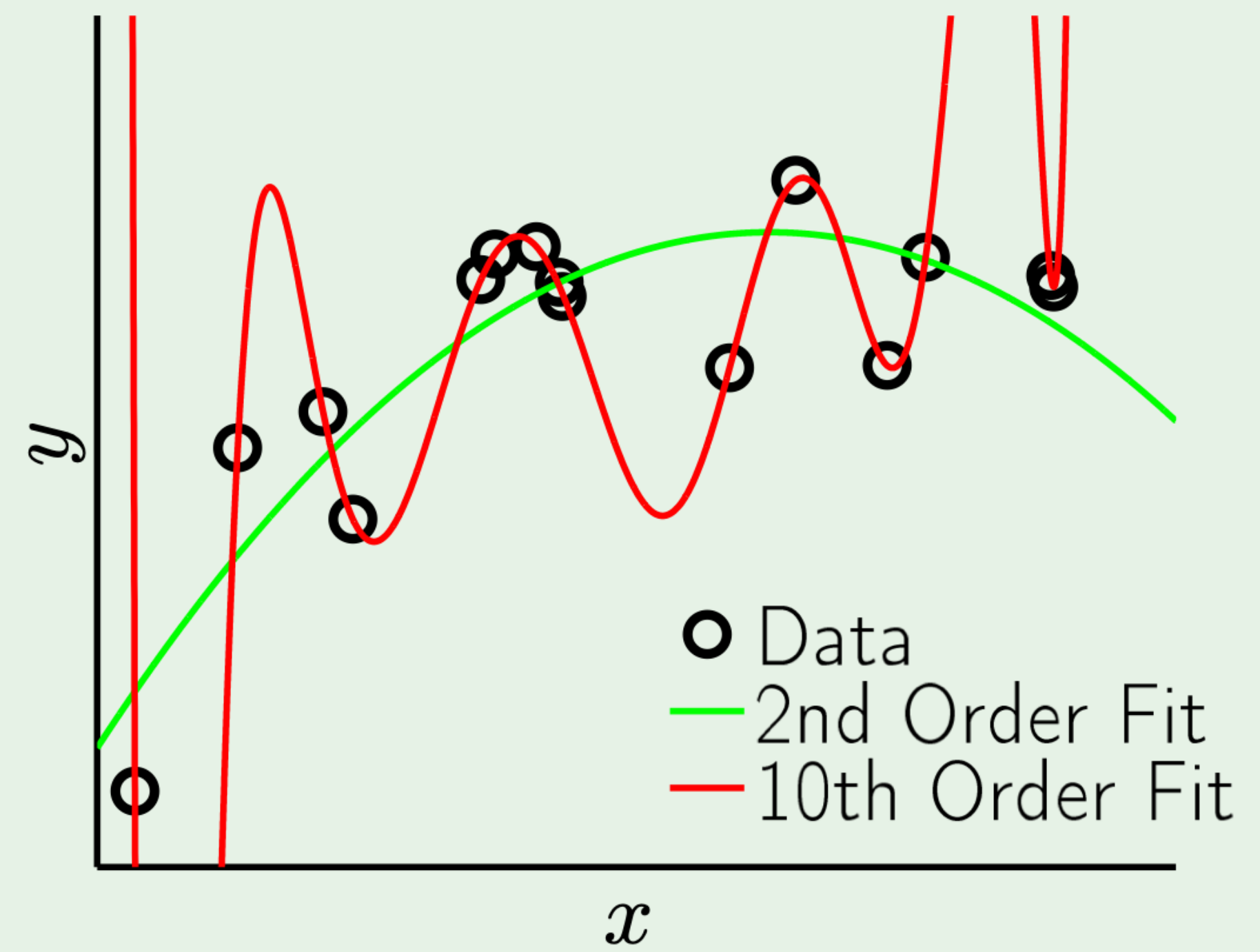


Two fits for each target



Noisy low-order target

	2nd Order	10th Order
E_{in}	0.050	0.034
E_{out}	0.127	9.00

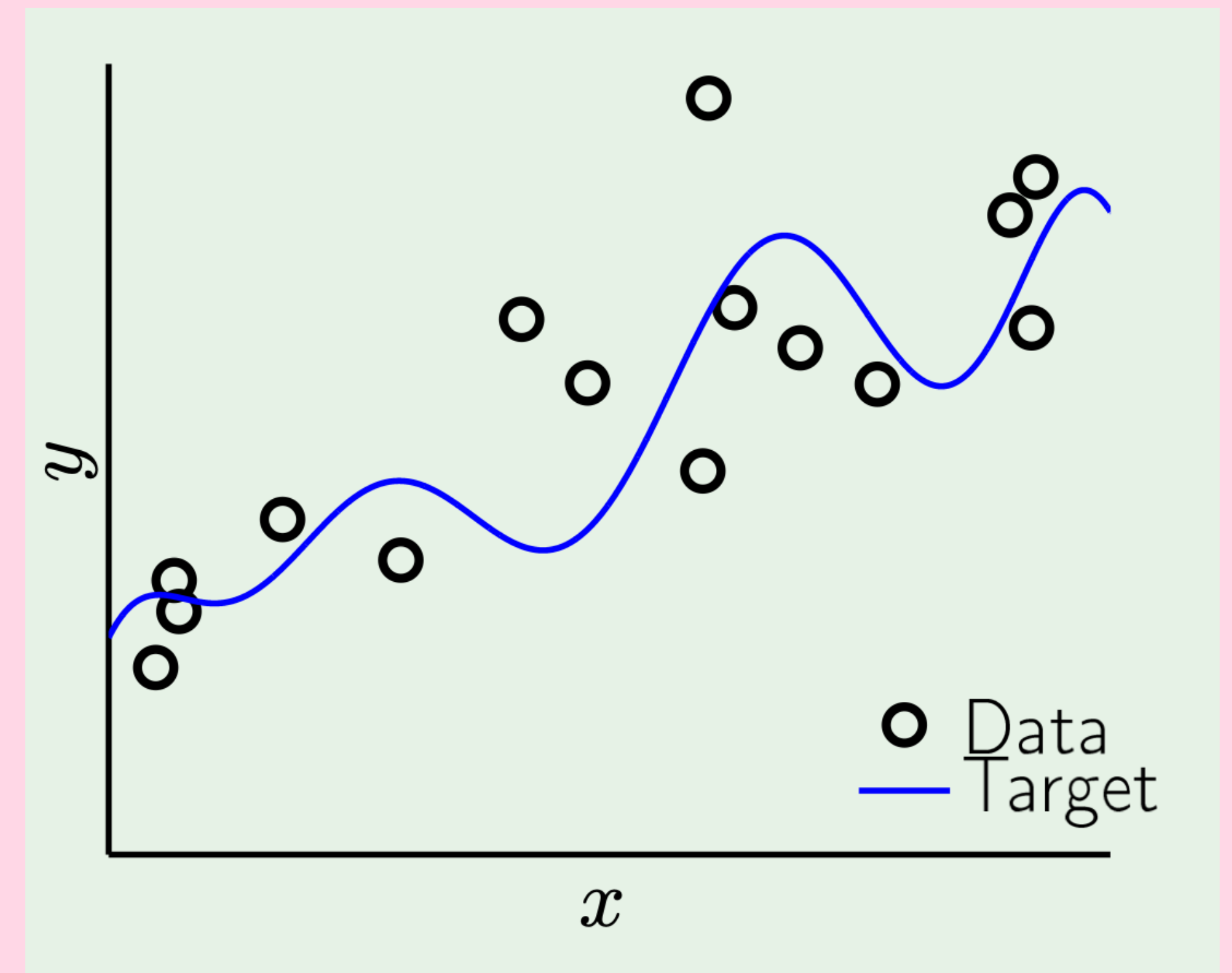


Noiseless high-order target

	2nd Order	10th Order
E_{in}	0.029	10^{-5}
E_{out}	0.120	7680

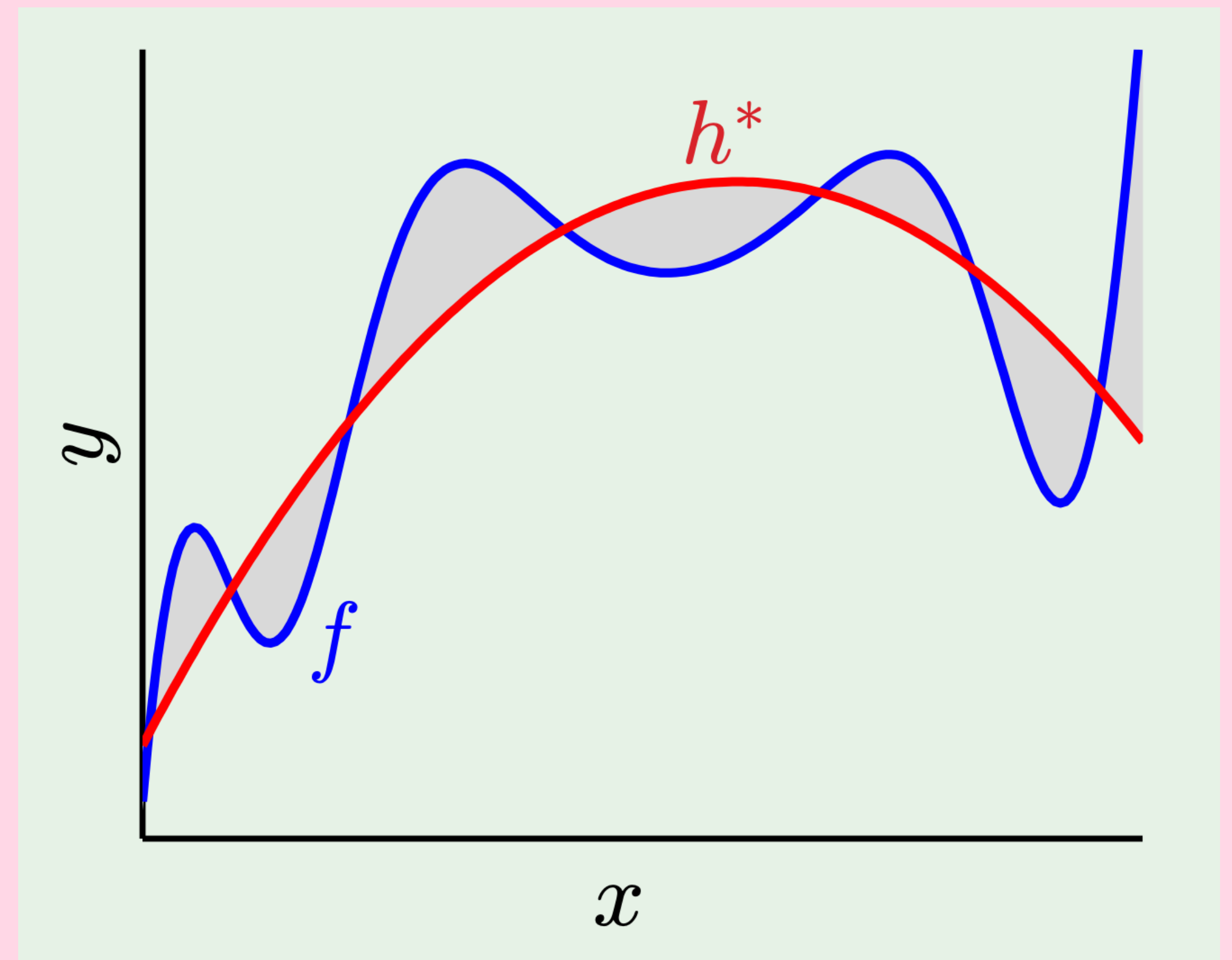
Stochastic noise

- Stochastic noise is often modeled as an additive random variable: $y = f(x) + \varepsilon$
- Fitting stochastic noise is bad, because it pulls your hypothesis away from the target f
- The best estimate of $f(x) + \varepsilon$ is $f(x)$



Deterministic noise

- Deterministic noise is the part of f that \mathcal{H} cannot capture: $f(x) - h^*(x)$
- h^* is the best hypothesis from \mathcal{H} , assuming “infinite data”
- But an $h \in \mathcal{H}$ learned from any finite dataset may be very far from h^*
- How far depends on the dataset
- Hence, “noise”



Preventing overfitting

- **Regularization:** deliberately restrict the complexity of your chosen h so that you reduce its ability to fit noise. “Putting on the brakes.”
- **Validation:** use a held-out dataset to directly estimate the thing you care about, which is the error rate on unseen examples. “Checking the bottom line.”

Regularization

Example: sine target

f

$$f : [-1, 1] \rightarrow \mathbb{R} \quad f(x) = \sin(\pi x)$$

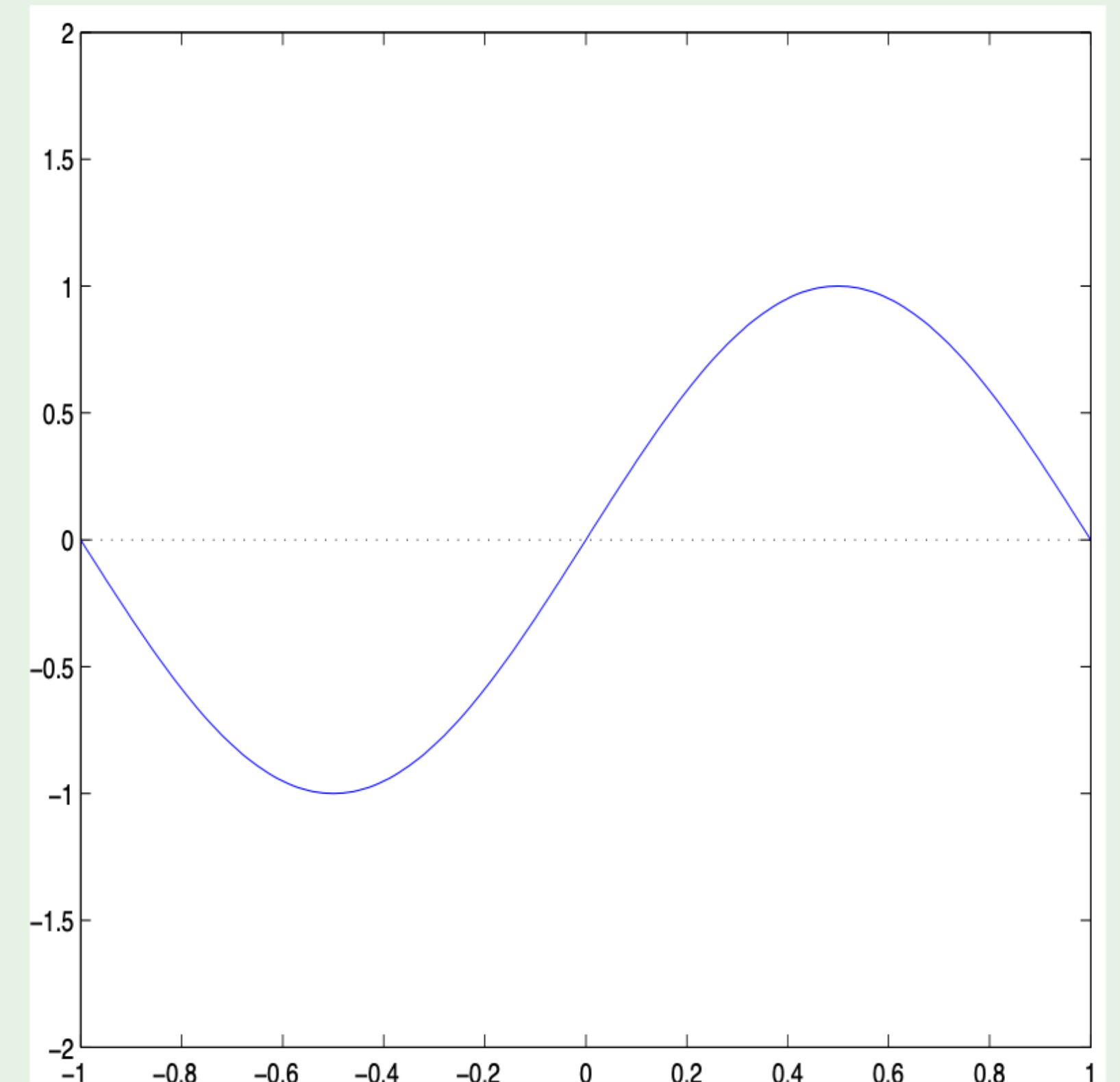
Only two training examples! $N = 2$

Two models used for learning:

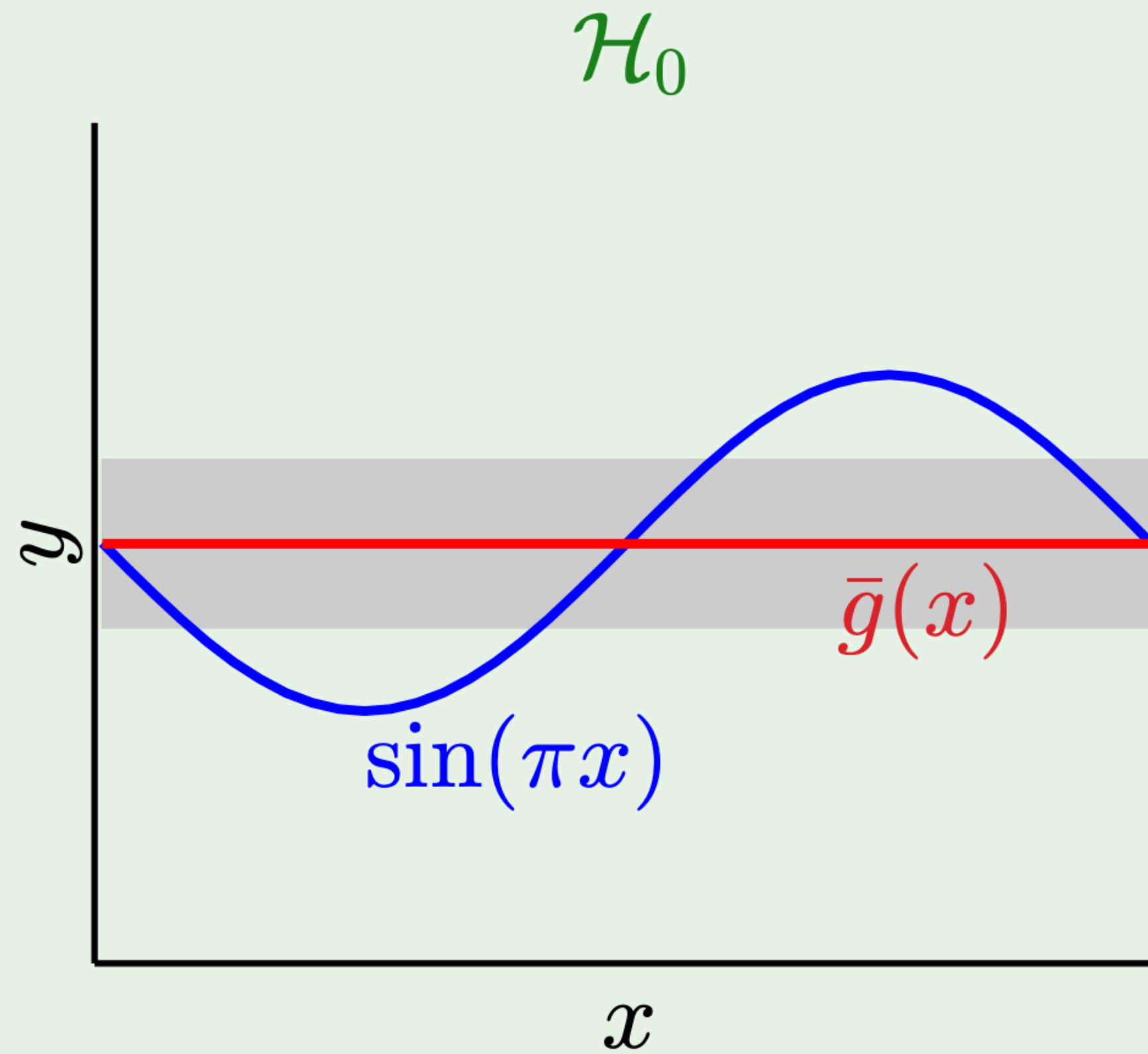
$$\mathcal{H}_0: \quad h(x) = b$$

$$\mathcal{H}_1: \quad h(x) = ax + b$$

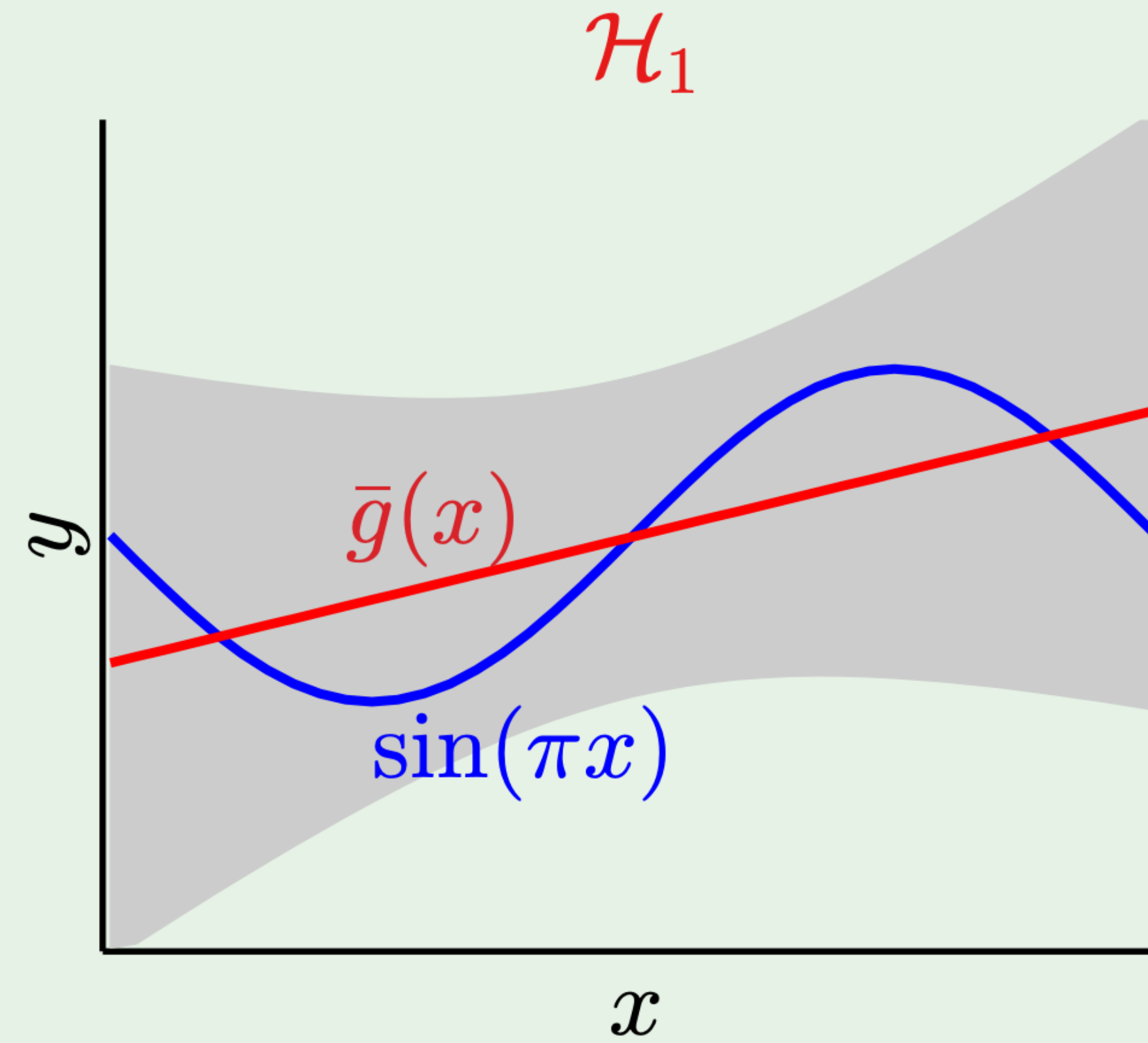
Which is better, \mathcal{H}_0 or \mathcal{H}_1 ?



and the winner is ...

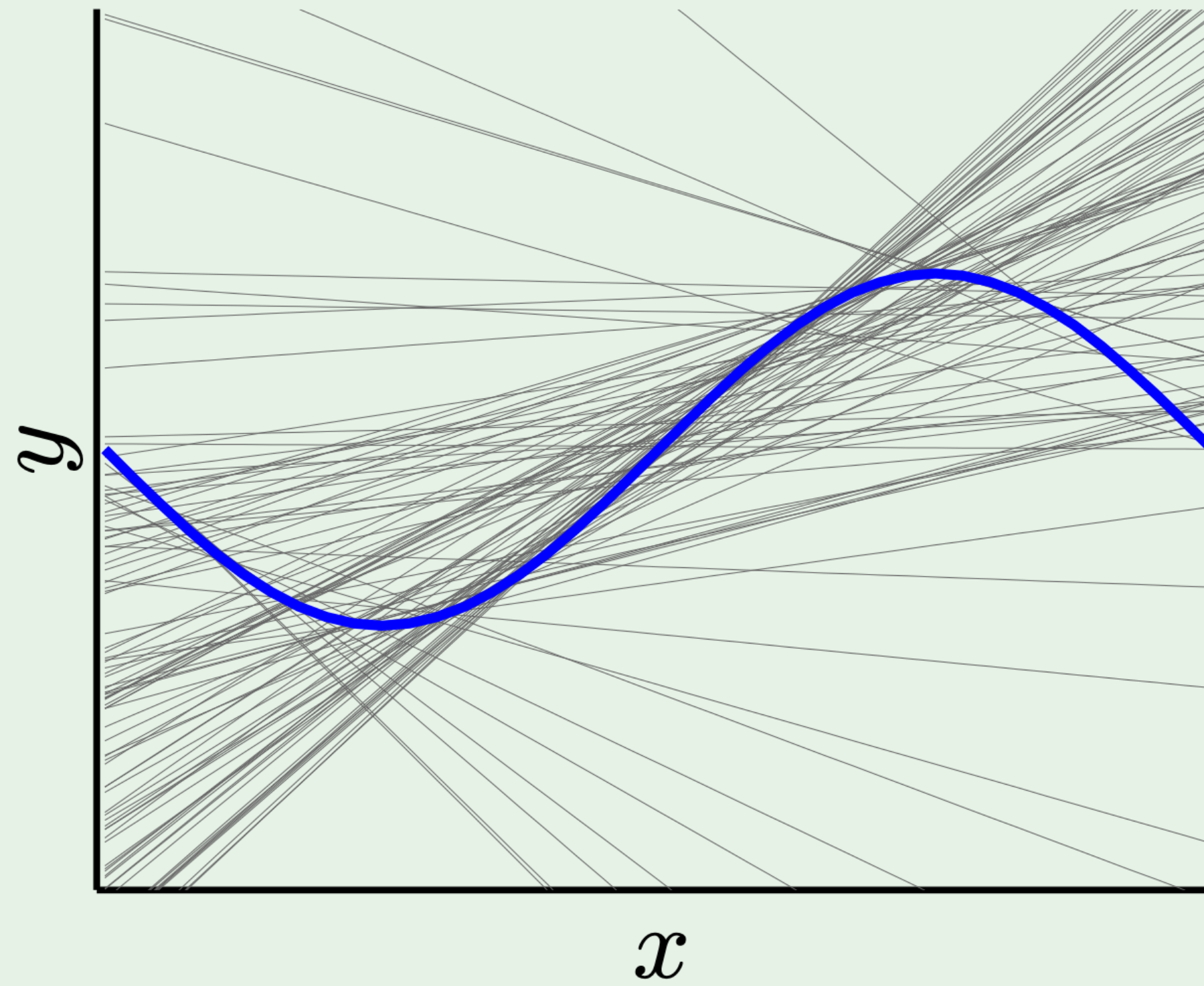


bias = **0.50** var = **0.25**

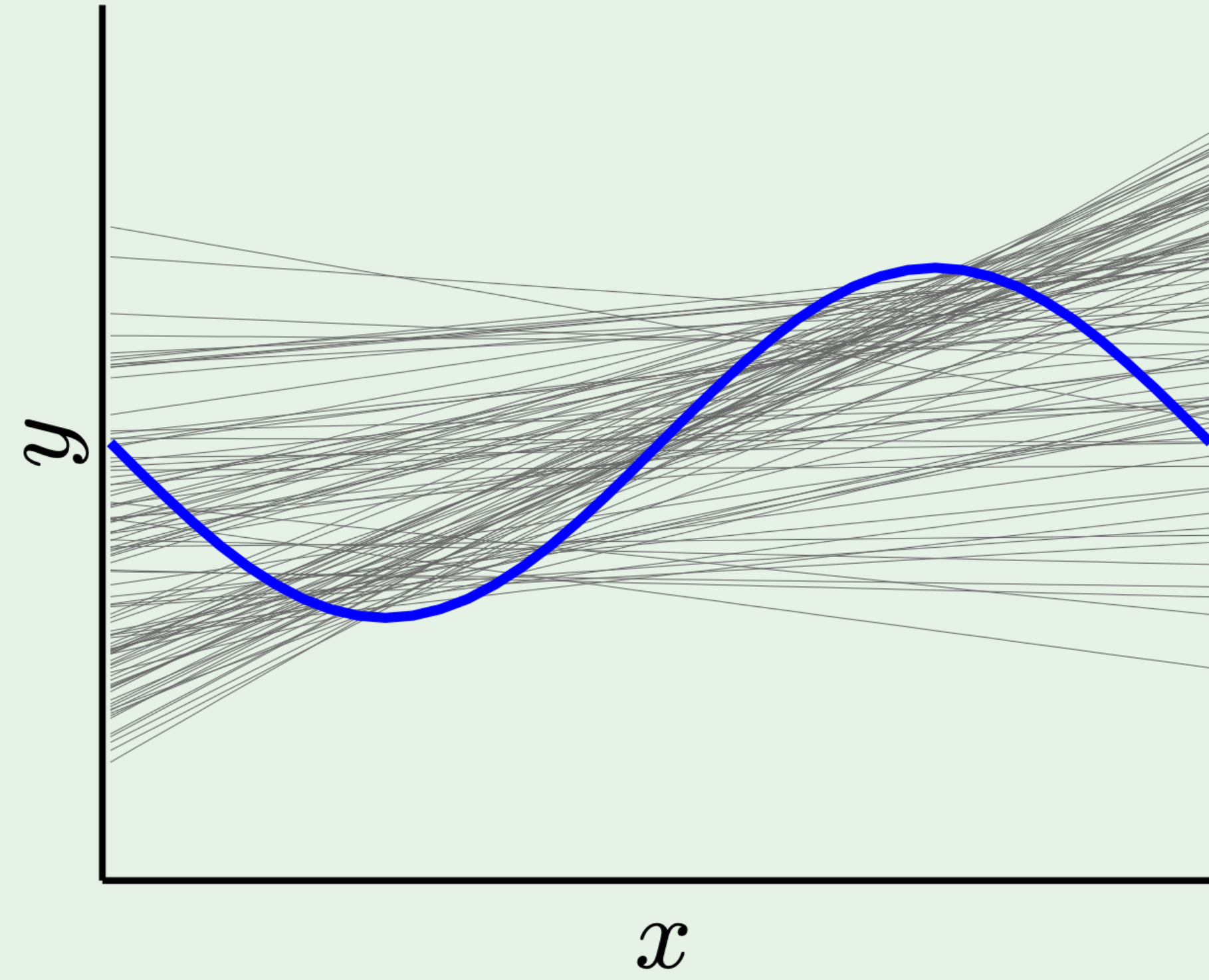


bias = **0.21** var = **1.69**

The sales pitch



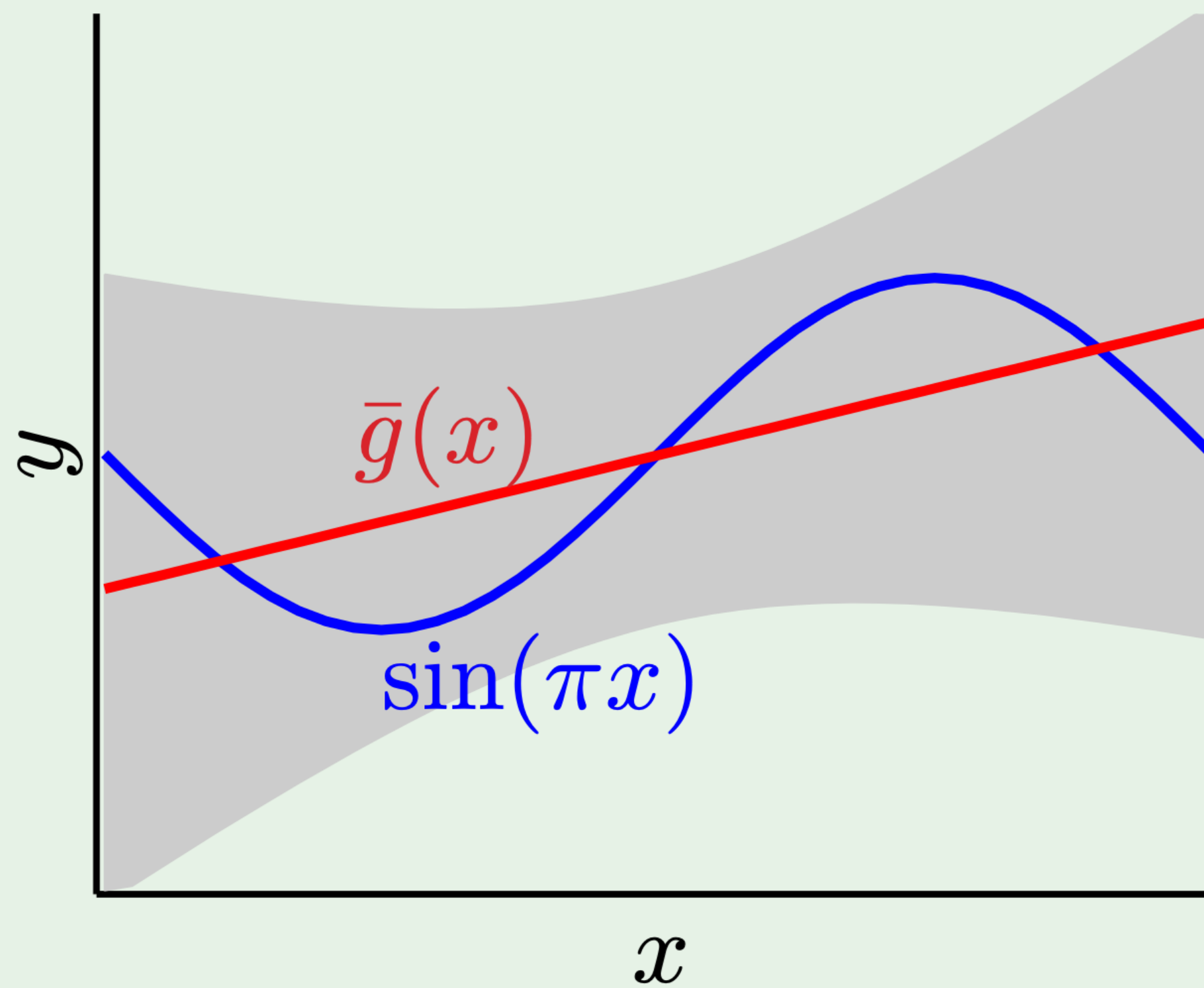
without regularization



with regularization

The sales pitch

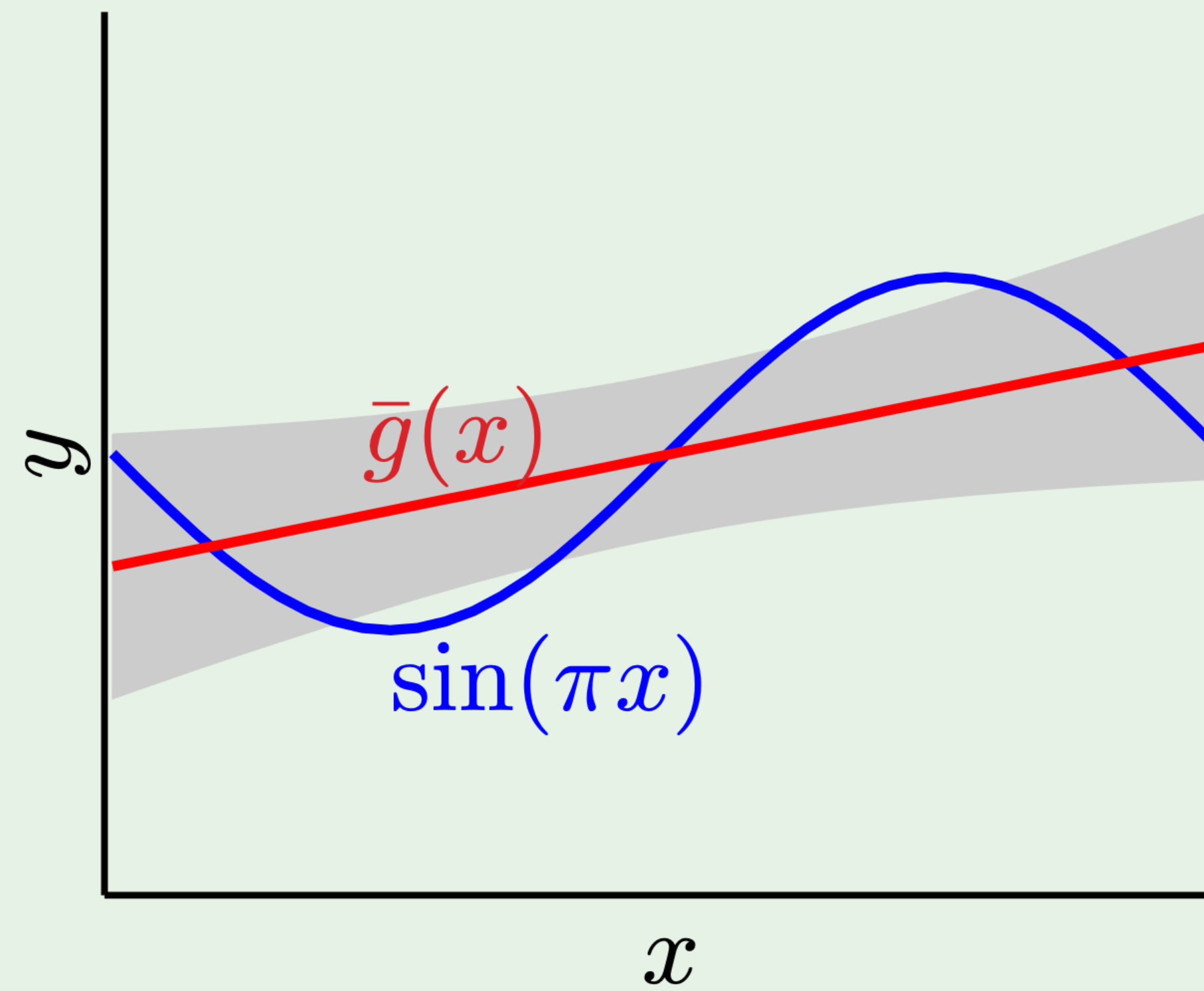
without regularization



bias = **0.21**

var = **1.69**

with regularization



bias = **0.23**

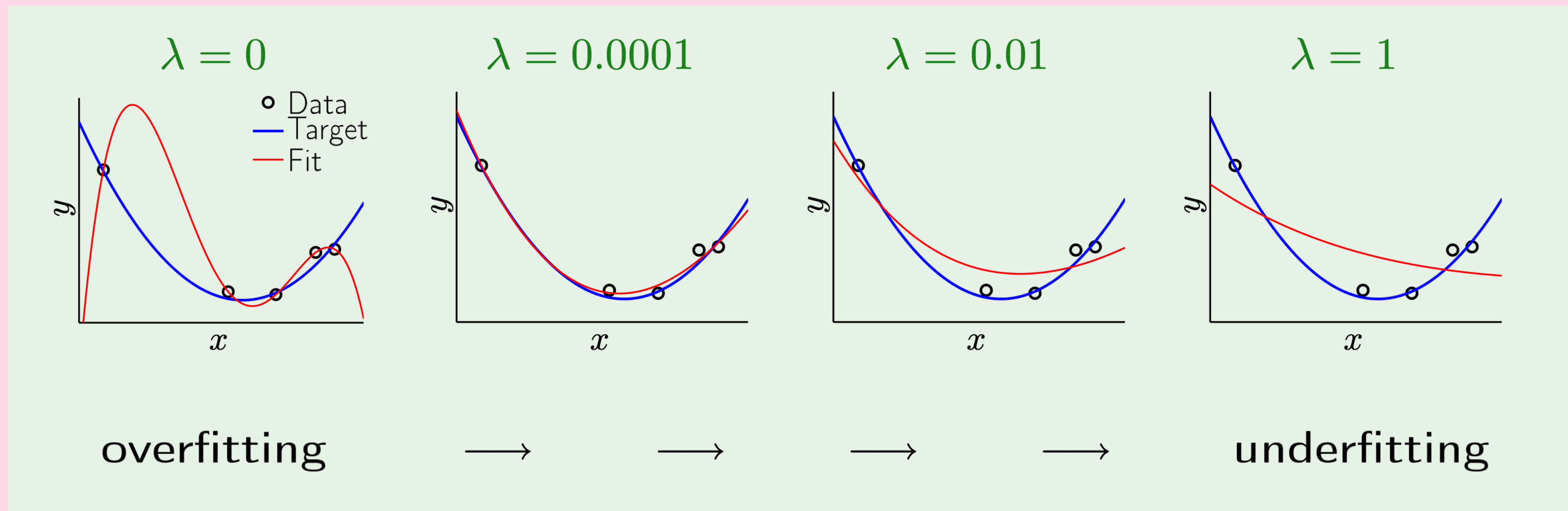
var = **0.33**

Preventing overfitting

- \mathcal{H}_0 can be thought of as a constrained version of \mathcal{H}_1 in which $w = 0$ in $h(x) = wx + b$
- This is a hard constraint on the model parameters
- Soft constraint: consider all hypotheses $wx + b$ such that $w^2 \leq A$ for some budget hyperparameter A
- w can be nonzero, but not too big
- Equivalently, minimize an augmented cost function $C(h) = \sum_{i=1}^n (h(x_i) - y_i)^2 + \lambda w^2$

Controlling the degree of fitting

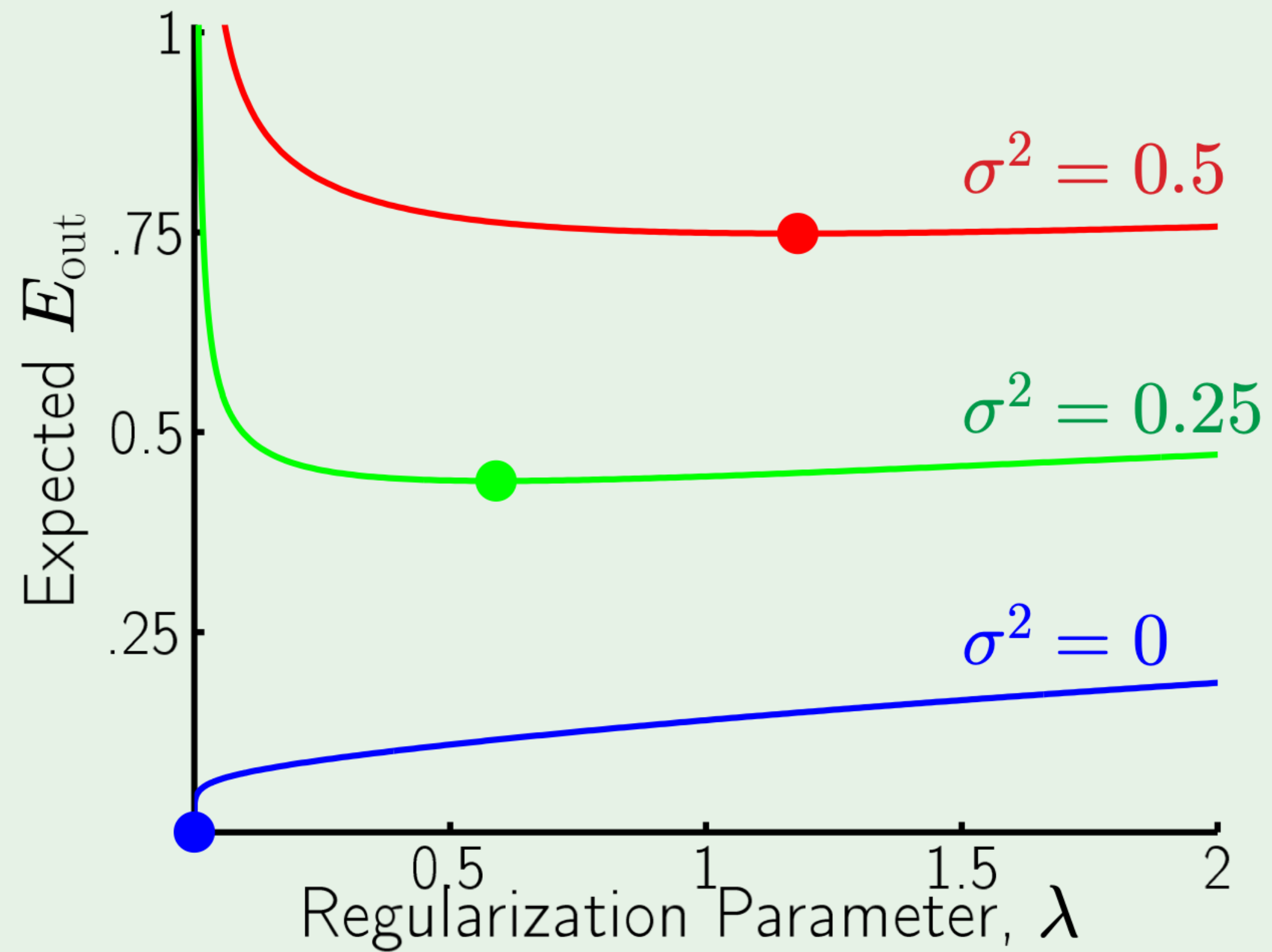
- λ is a regularization hyperparameter that controls the tradeoff between minimizing training error and using “reasonable” weights
- Large λ corresponds to small A



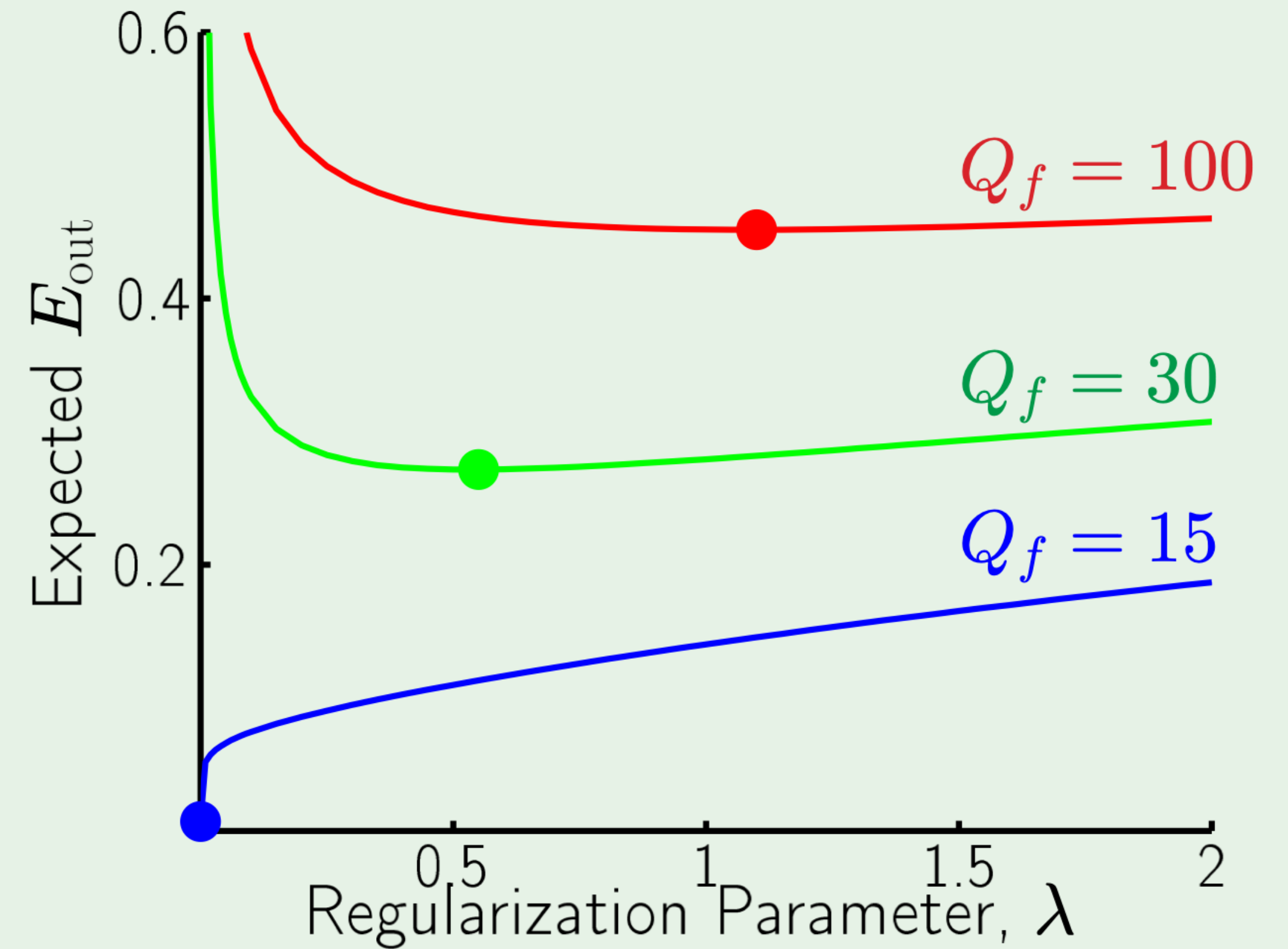
The regularizer, generalized

- More generally, if there are many weight parameters (as in a d degree polynomial regression model), $C(h) = \sum_{i=1}^n (h(x_i) - y_i)^2 + \lambda \sum_{j=1}^d w_j^2$
- This is called ℓ_2 regularization, because $\sum_{j=1}^d w_j^2 = \|w\|^2$ is the squared ℓ_2 norm of the weight vector

More noise requires more regularization



Stochastic noise



Deterministic noise

Early stopping is regularization

- The more SGD updates you make, the more opportunity the parameters have to become very large
- Early stopping is one way to prevent this growth in model complexity
- How do you know when to stop? Validation!

