Inductive Bias

DSCC 251/451: Machine Learning with Limited Data

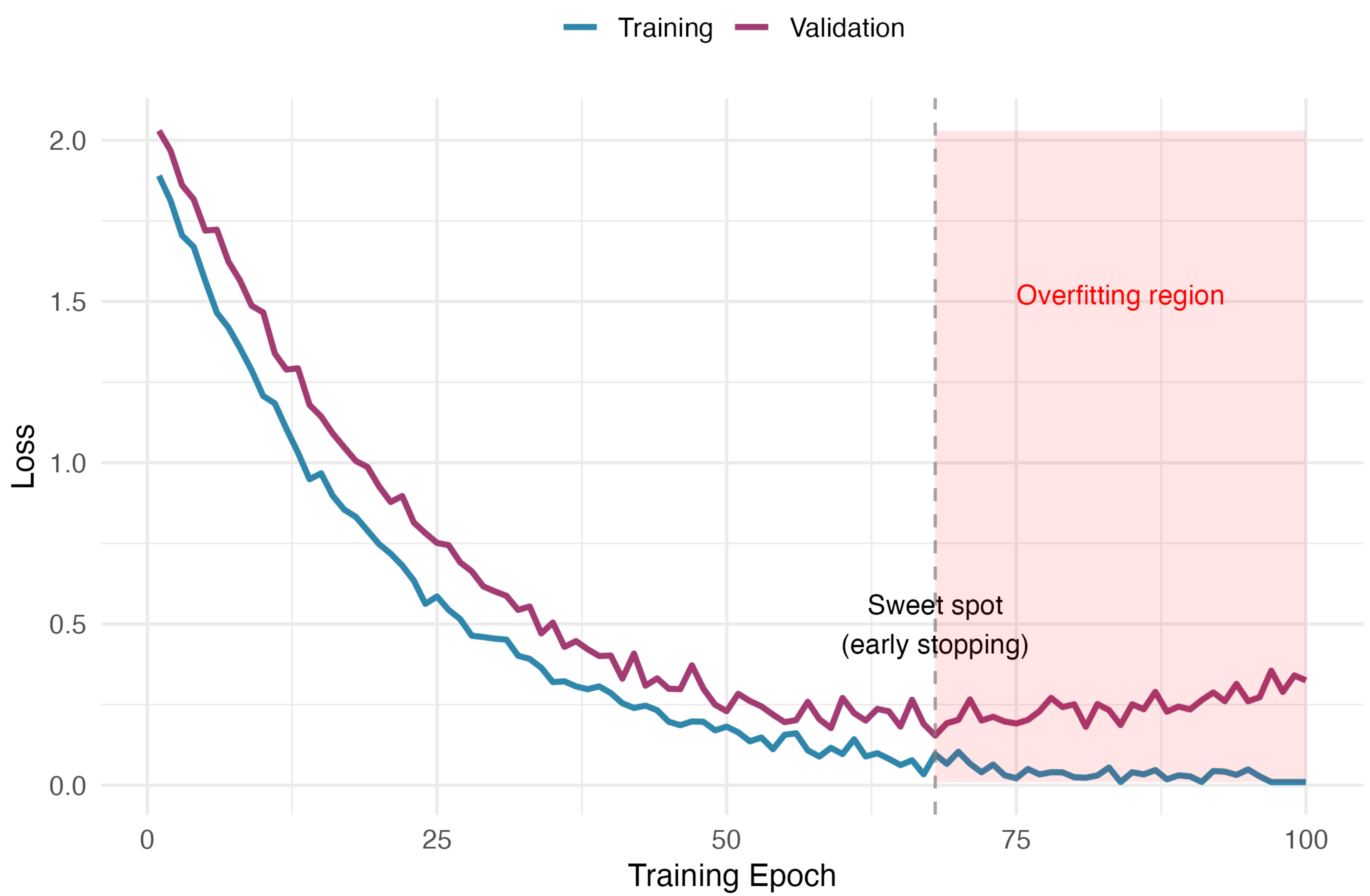
C.M. Downey

Spring 2026

Bias vs. Variance Recap

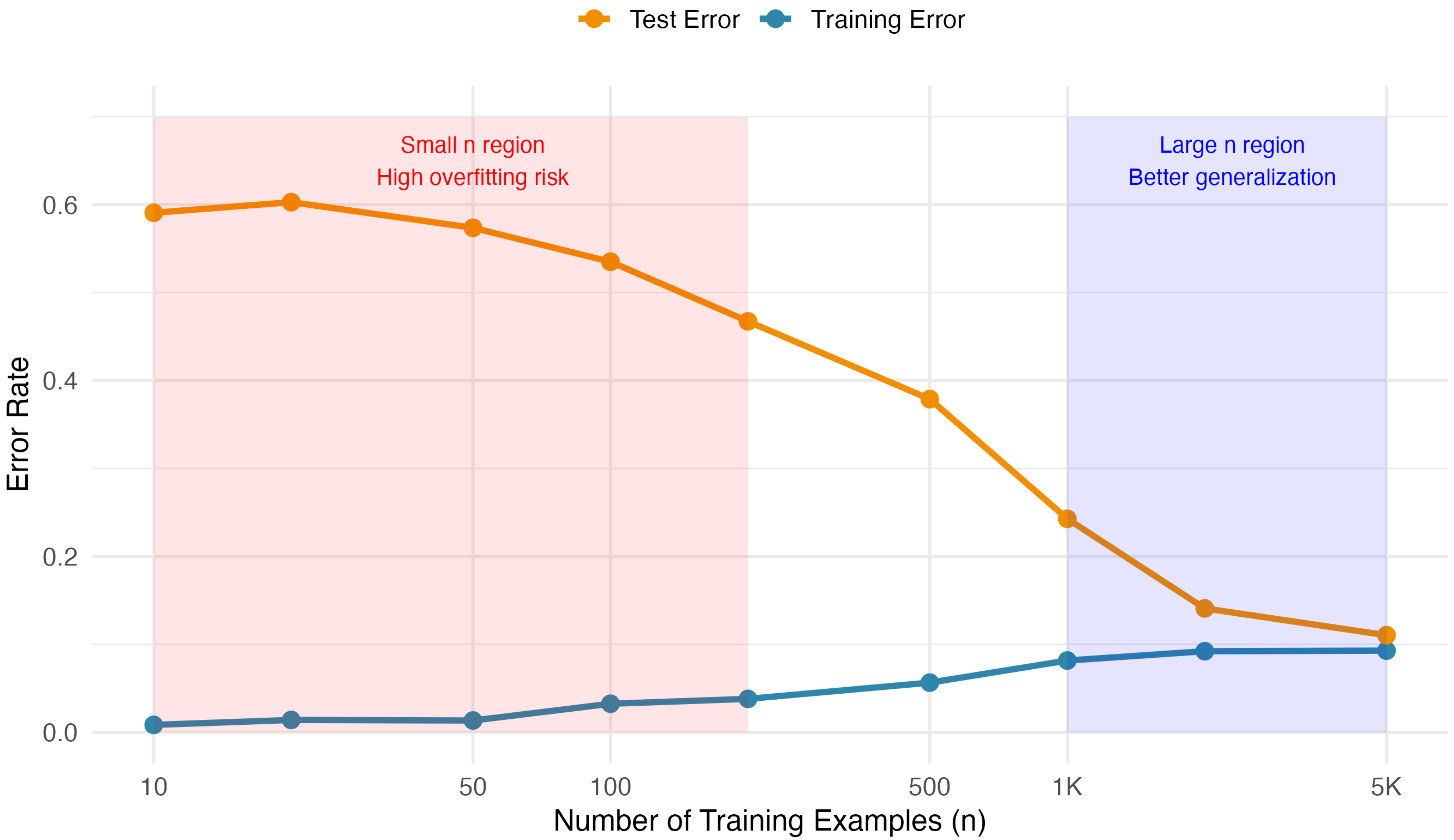
What Overfitting Looks Like

Training loss keeps improving, but validation loss increases

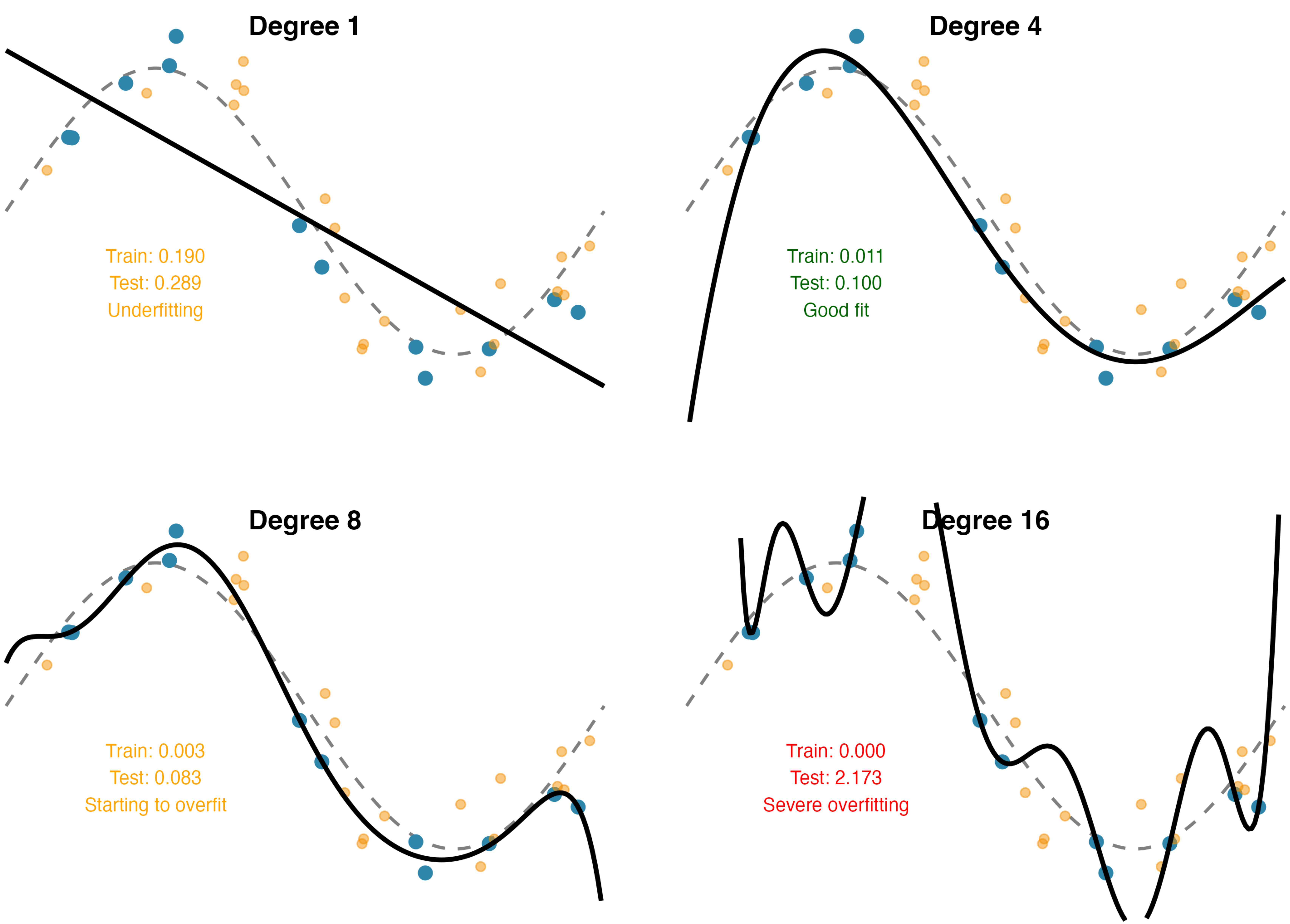


The Generalization Gap Depends on Data Size

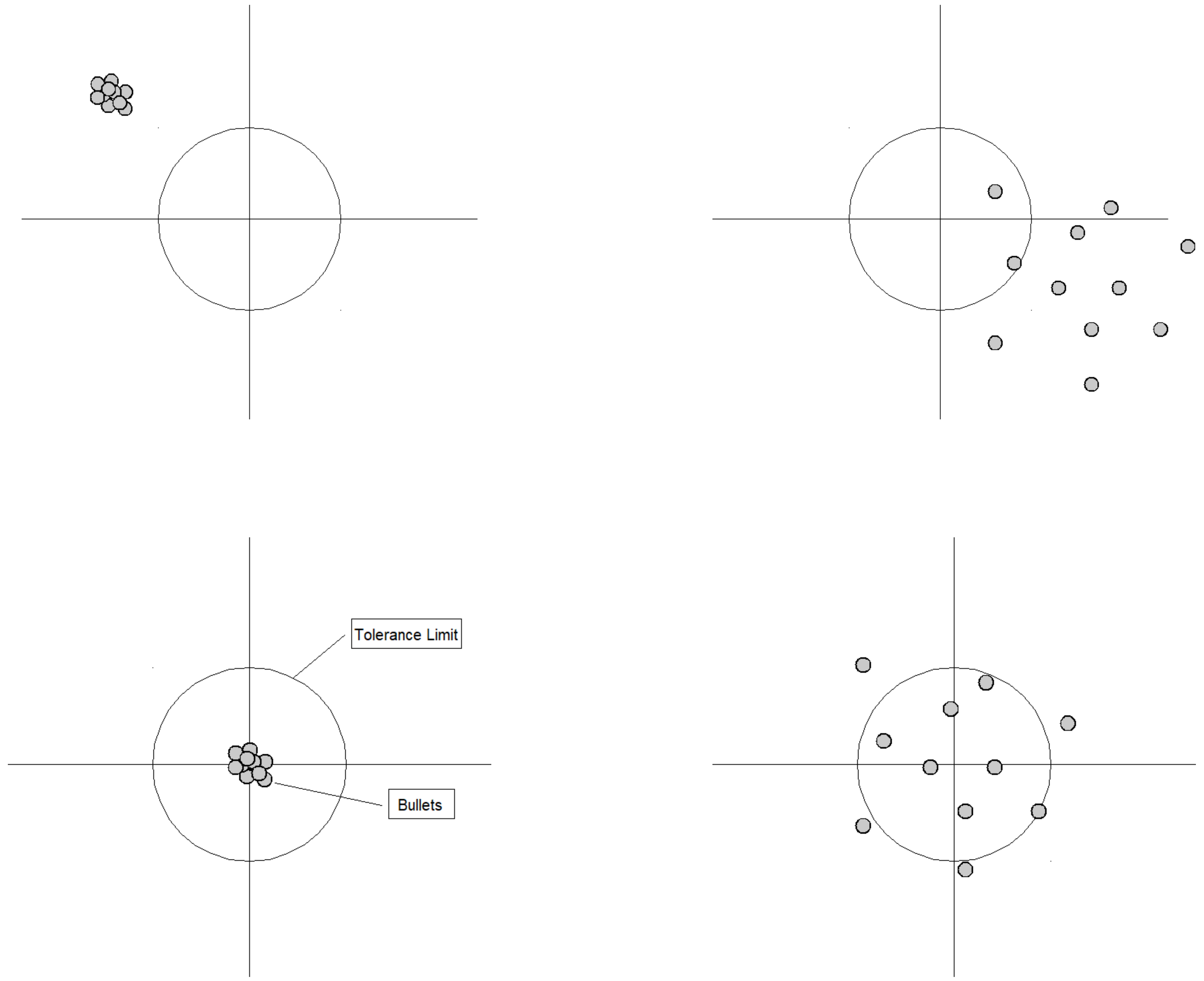
This is the fundamental picture of the course



Model Complexity vs. Overfitting
Blue = training data, Orange = test data, Dashed = true function

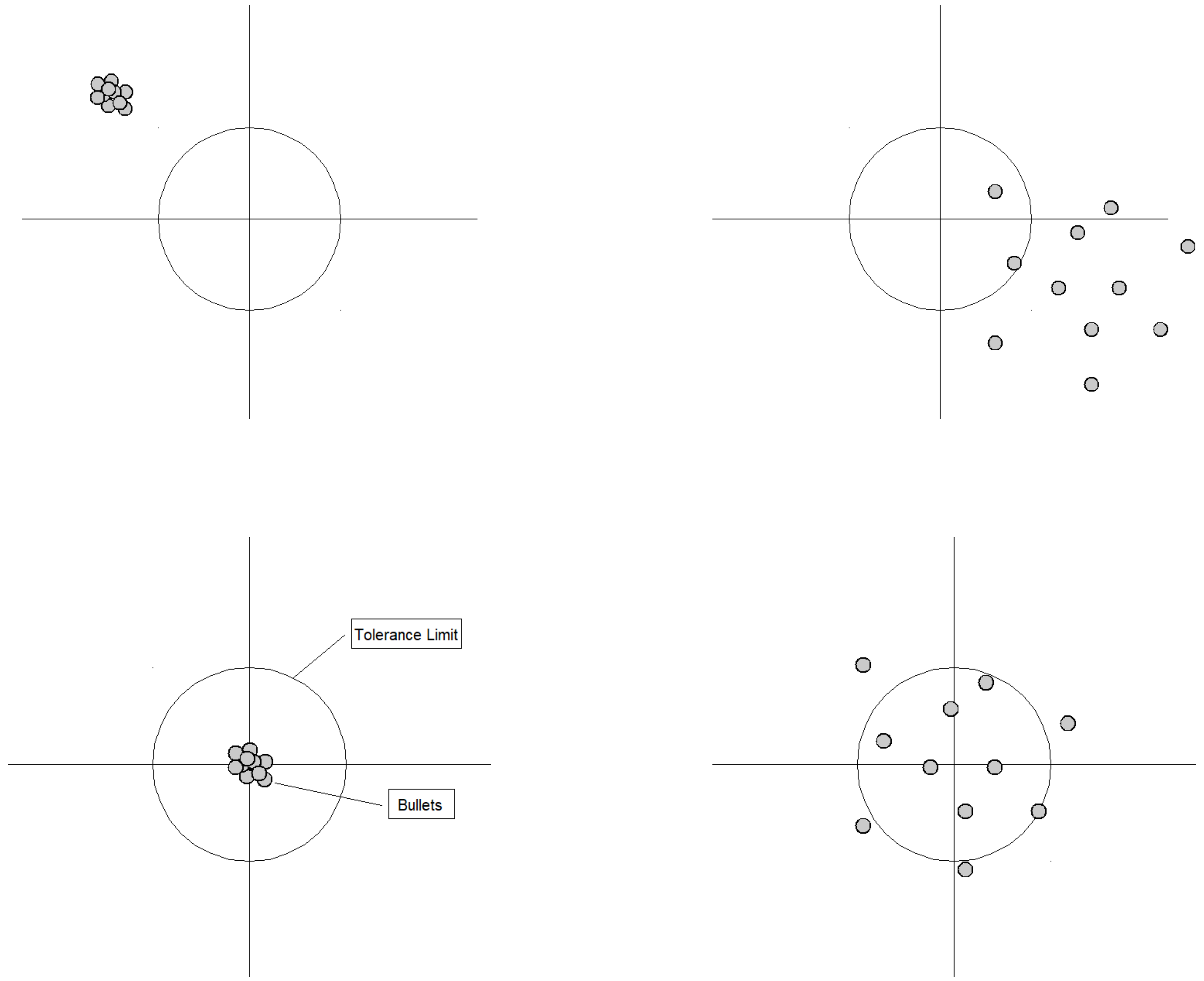


Bias and Variance



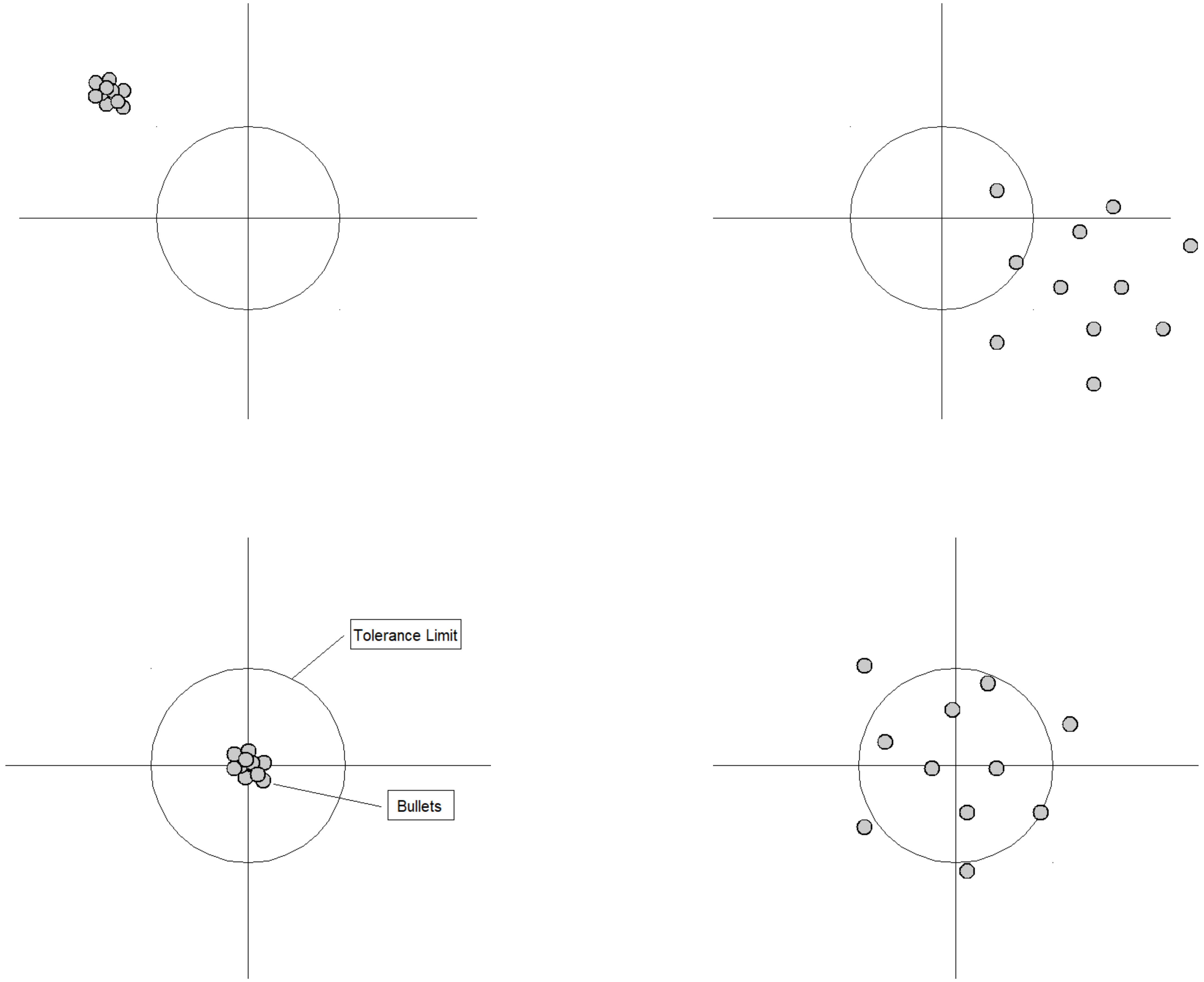
Bias and Variance

high bias
low variance



Bias and Variance

high bias
low variance



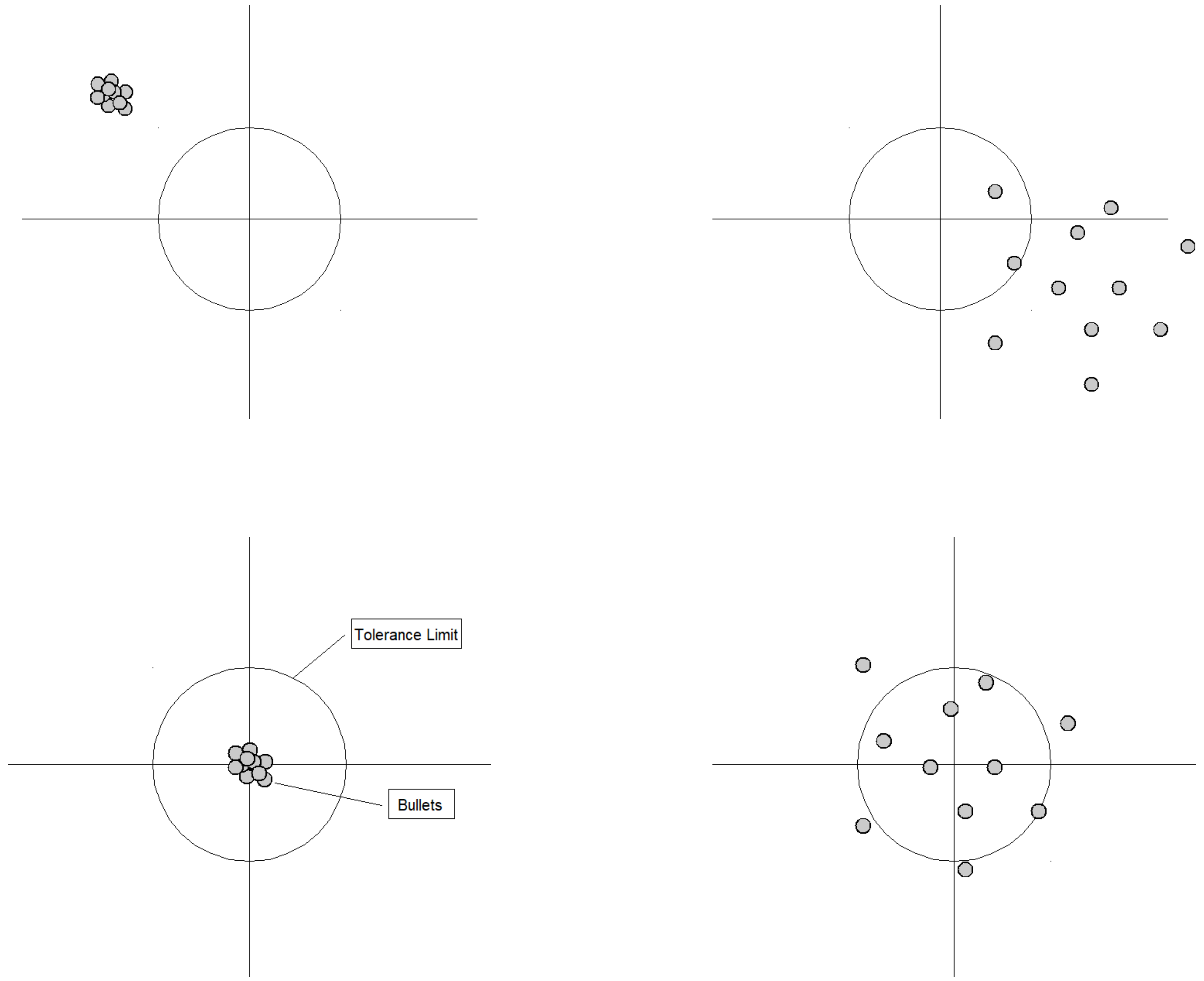
high bias
high variance

Bias and Variance

high bias
low variance

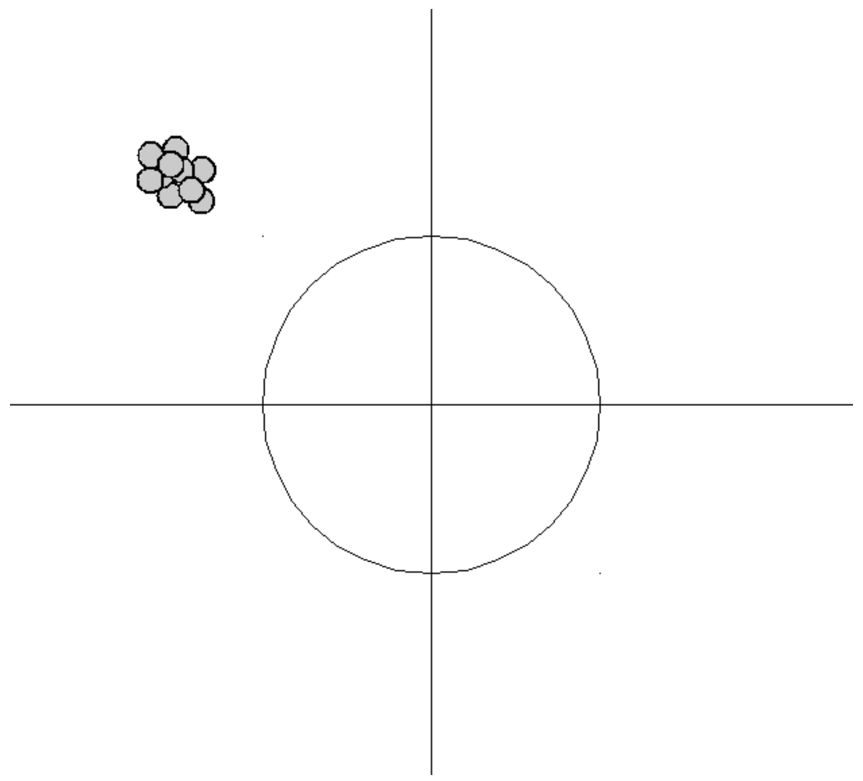
high bias
high variance

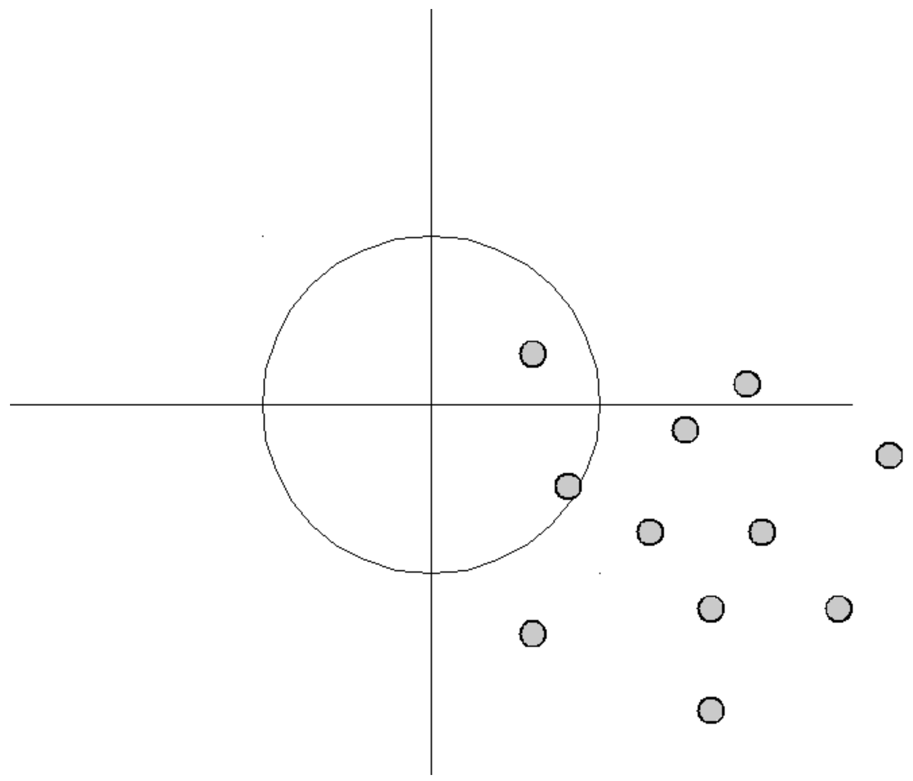
low bias
low variance



Bias and Variance

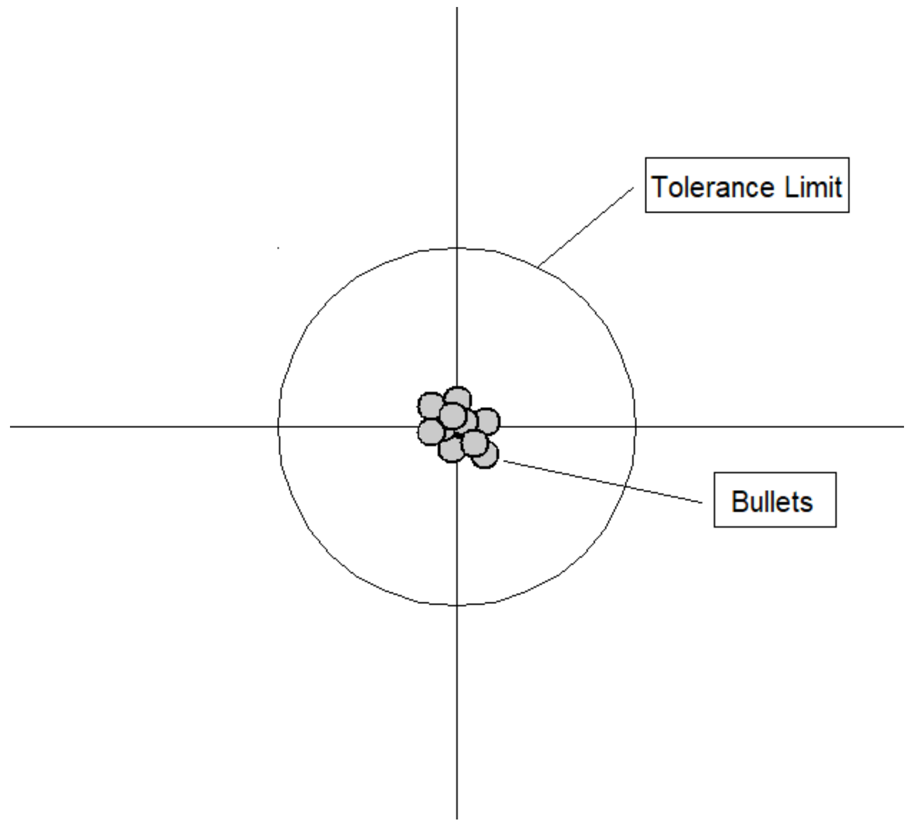
high bias
low variance

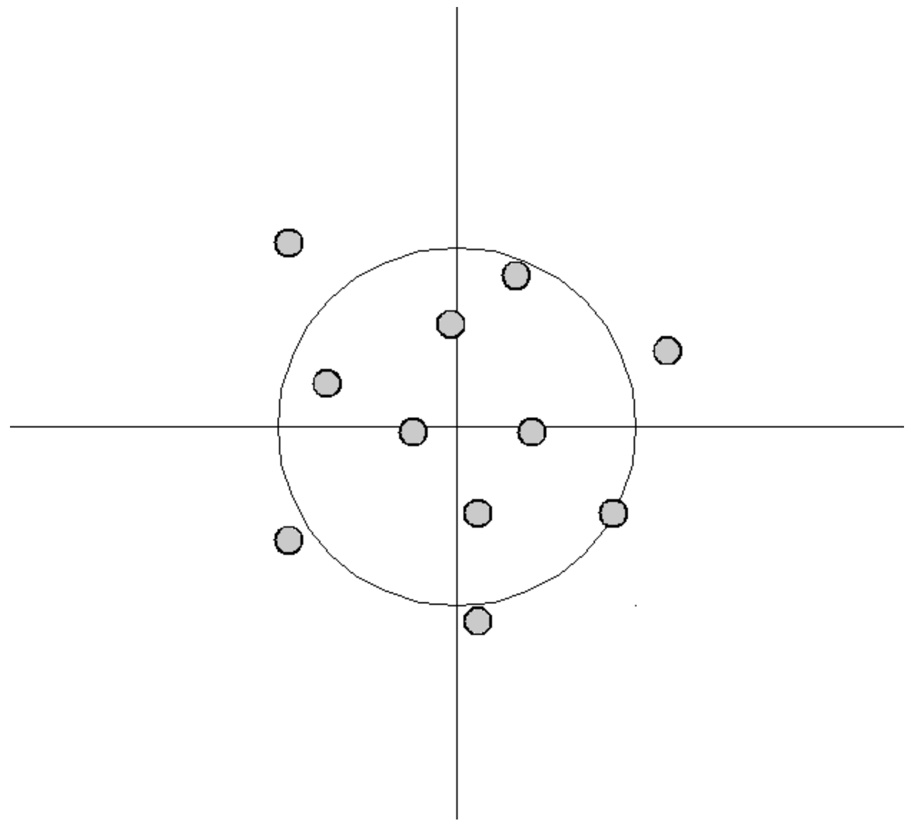




high bias
high variance

low bias
low variance

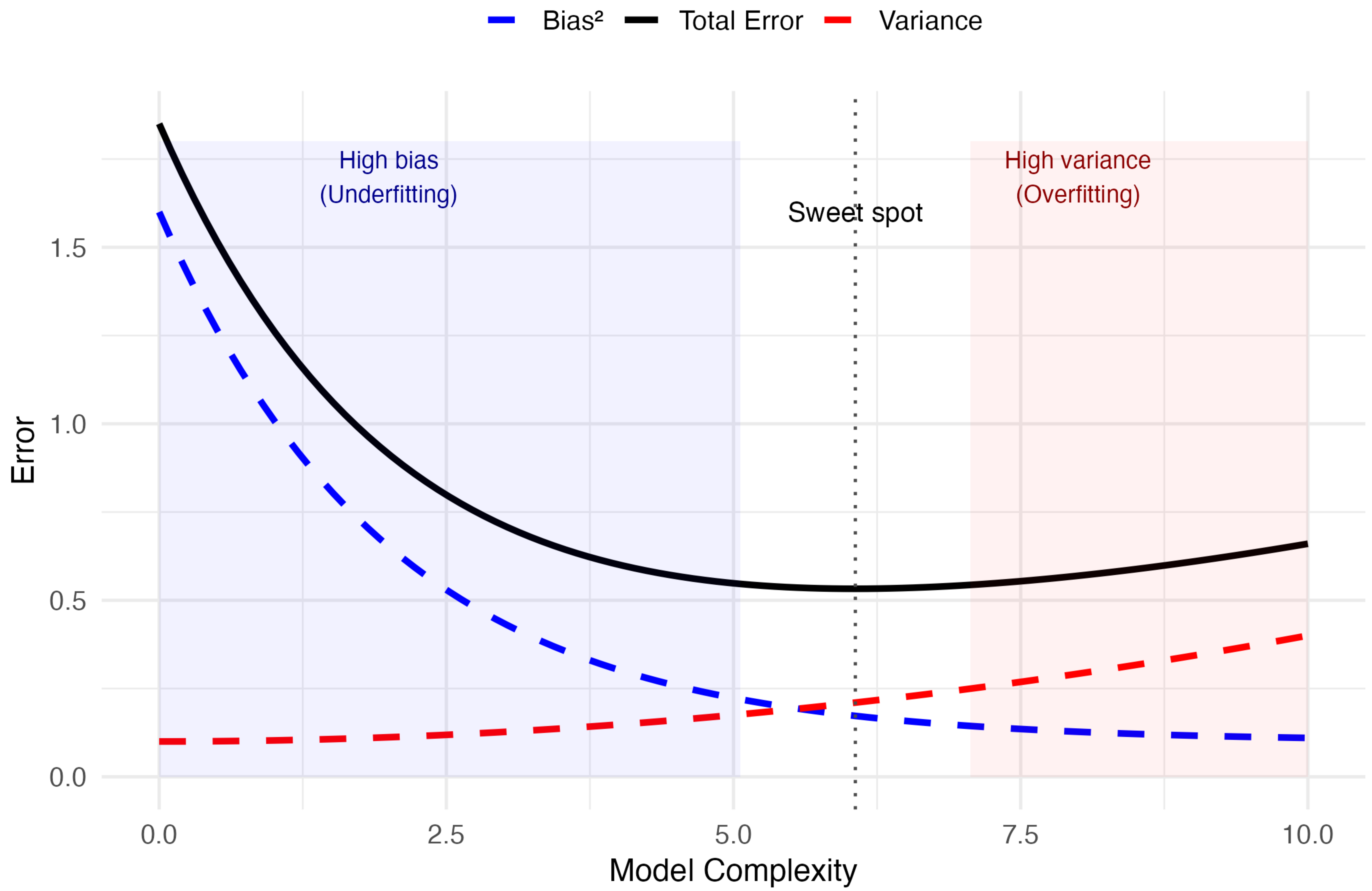




low bias
high variance

The Bias-Variance Tradeoff

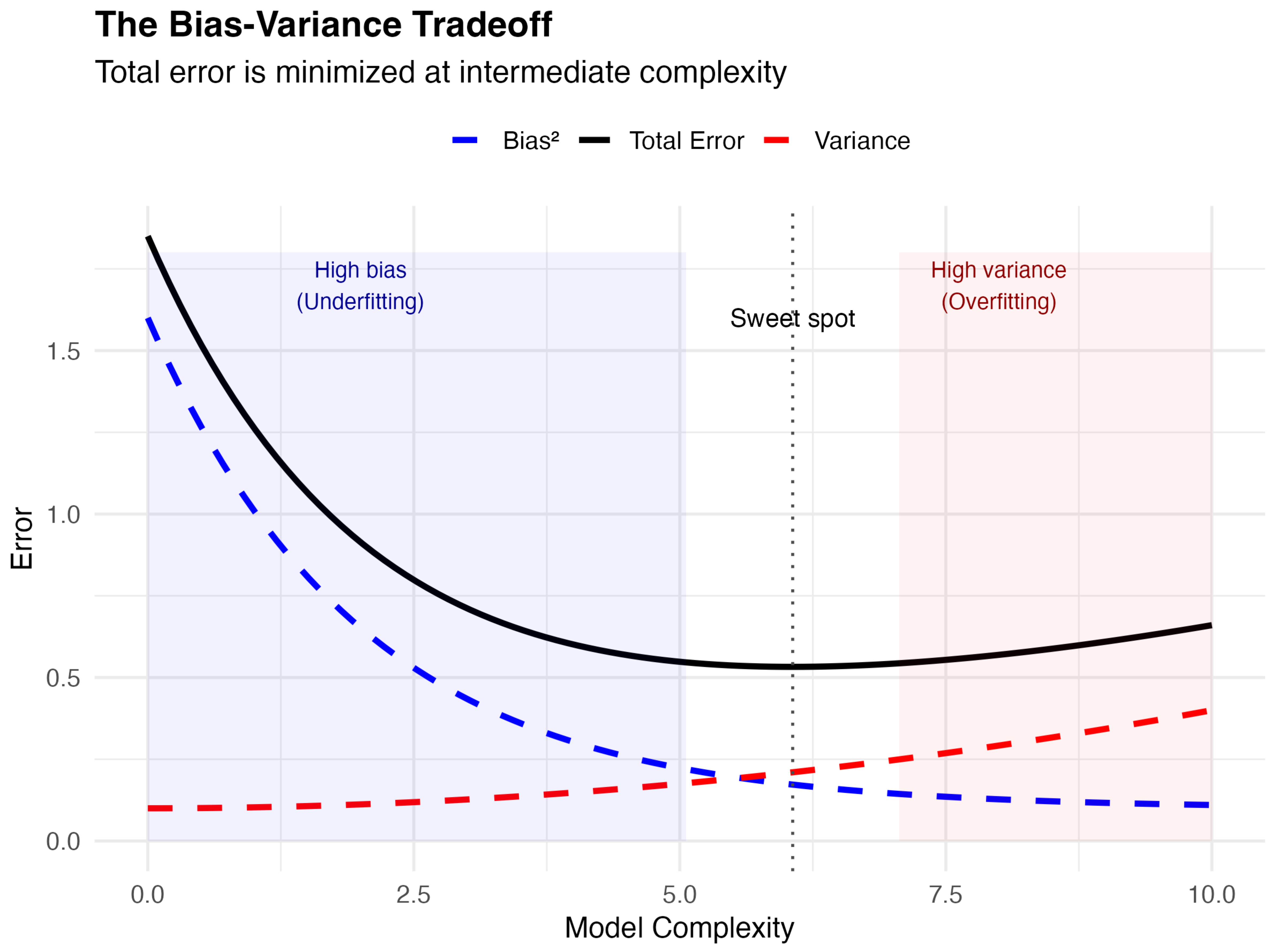
Total error is minimized at intermediate complexity



Bias/Variance Decomposition

Bias/Variance Decomposition

- We have the **intuition** of why Bias and Variance are in a tradeoff
  - What can we say **mathematically?**
- We are able to **decompose** the definition of model error:
  - $\mathbb{E}[(y - \hat{f}(x))^2]$ (model error)
  - = $\text{Bias}^2 + \text{Variance} + \text{noise}$
  - Here's how



Starting the Derivation

Starting the Derivation

- Start with a **fixed test datapoint** x. The **true relationship** we want to model is $y = f^*(x) + \epsilon$

Starting the Derivation

- Start with a **fixed test datapoint** x. The **true relationship** we want to model is $y = f^*(x) + \epsilon$
  - y: the **actual/desired output**

Starting the Derivation

- Start with a **fixed test datapoint** x. The **true relationship** we want to model is $y = f^*(x) + \epsilon$
  - y: the **actual/desired output**
  - $f^*(x)$: the **perfect model** (the function we're trying to learn)

Starting the Derivation

- Start with a **fixed test datapoint** x. The **true relationship** we want to model is $y = f^*(x) + \epsilon$
  - y: the **actual/desired output**
  - $f^*(x)$: the **perfect model** (the function we're trying to learn)
  - ϵ: **noise** (intrinsic randomness we always consider part of the data)

Starting the Derivation

- Start with a **fixed test datapoint** x. The **true relationship** we want to model is $y = f^*(x) + \epsilon$
  - y: the **actual/desired output**
  - $f^*(x)$: the **perfect model** (the function we're trying to learn)
  - ϵ: **noise** (intrinsic randomness we always consider part of the data)
- We'll assume our **training set** is **random**

Starting the Derivation

- Start with a **fixed test datapoint** x. The **true relationship** we want to model is $y = f^*(x) + \epsilon$
  - y: the **actual/desired output**
  - $f^*(x)$: the **perfect model** (the function we're trying to learn)
  - ϵ: **noise** (intrinsic randomness we always consider part of the data)
- We'll assume our **training set** is **random**
  - What is the **expected error** for the model, across **all possible training sets?**

Starting the Derivation

- Start with a **fixed test datapoint** x. The **true relationship** we want to model is $y = f^*(x) + \epsilon$
  - y: the **actual/desired output**
  - $f^*(x)$: the **perfect model** (the function we're trying to learn)
  - ϵ: **noise** (intrinsic randomness we always consider part of the data)
- We'll assume our **training set** is **random**
  - What is the **expected error** for the model, across **all possible training sets?**
  - $\mathbb{E}[(y - \hat{f}(x))^2]$ (this is what we will **decompose**)

Decomposition

Decomposition

- Start with $\mathbb{E}[(y - \hat{f}(x))^2]$ (previous slide)

Decomposition

- Start with $\mathbb{E}[(y - \hat{f}(x))^2]$ (previous slide)
- Substitute $y = f^*(x) + \epsilon$

Decomposition

- Start with $\mathbb{E}[(y - \hat{f}(x))^2]$ (previous slide)
- Substitute $y = f^*(x) + \epsilon$
  - $\mathbb{E}[(f^*(x) + \epsilon - \hat{f}(x))^2]$

Decomposition

- Start with $\mathbb{E}[(y - \hat{f}(x))^2]$ (previous slide)
- Substitute $y = f^*(x) + \epsilon$
  - $\mathbb{E}[(f^*(x) + \epsilon - \hat{f}(x))^2]$
- Add and subtract $\mathbb{E}[\hat{f}(x)]$ (trick)

Decomposition

- Start with $\mathbb{E}[(y - \hat{f}(x))^2]$ (previous slide)
- Substitute $y = f^*(x) + \epsilon$
  - $\mathbb{E}[(f^*(x) + \epsilon - \hat{f}(x))^2]$
- Add and subtract $\mathbb{E}[\hat{f}(x)]$ (trick)
  - $$\mathbb{E}\left[\Big(\underbrace{f^*(x) - \mathbb{E}[\hat{f}(x)]}_{\text{bias (constant)}} + \underbrace{\mathbb{E}[\hat{f}(x)] - \hat{f}(x)}_{\text{variance (random)}} + \underbrace{\epsilon}_{\text{noise (random)}}\Big)^2\right]$$

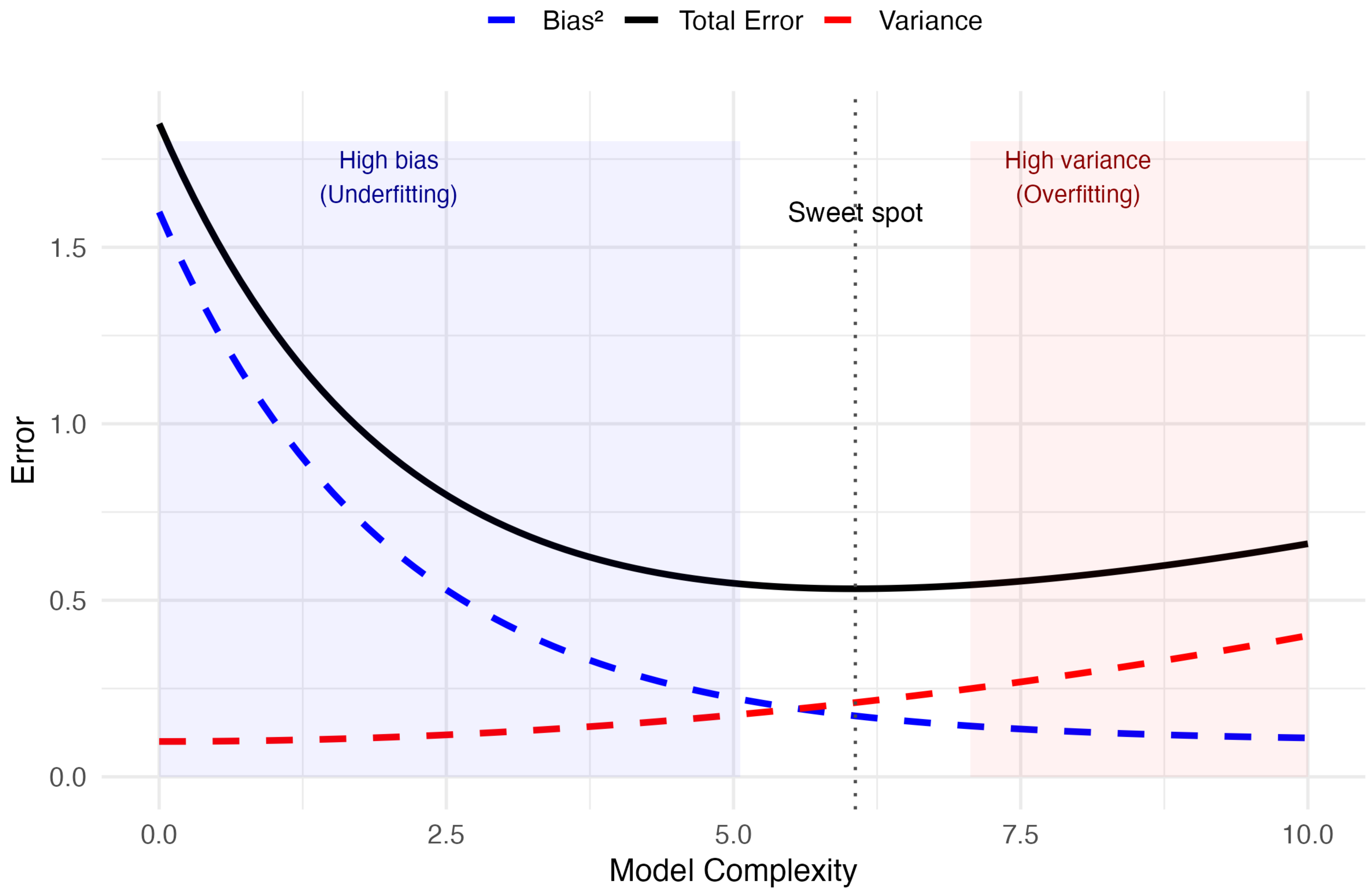
Decomposition

- Finally, with some algebra we get the **equation seen below**
- Model error is **decomposable** into Bias^2 + Variance + Noise
  - This is why there will **always be a tradeoff!**
- Bias: the difference between the **model** and the **true (ideal) function**
- Variance: the difference between the **model** and **its own mean**
- Noise: **intrinsic randomness** in the data

$$\mathbb{E}[(y - \hat{f}(x))^2] = \underbrace{(f^*(x) - \mathbb{E}[\hat{f}(x)])^2}_{\text{Bias}^2} + \underbrace{\mathbb{E}[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2]}_{\text{Variance}} + \underbrace{\sigma^2}_{\text{Irreducible}}$$

The Bias-Variance Tradeoff

Total error is minimized at intermediate complexity



Bayesian Priors

Bayes' Rule

$$P(A \mid B) := \frac{P(A \cap B)}{P(B)}$$

Def. of Conditional Probability

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

Bayes' Rule

Bayes' Rule

- **Bayesian** statistics works with **Conditional Probabilities**
  - $P(A \mid B)$: what is the probability of A **given B?**

$$P(A \mid B) := \frac{P(A \cap B)}{P(B)}$$

Def. of Conditional Probability

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

Bayes' Rule

Bayes' Rule

- **Bayesian** statistics works with **Conditional Probabilities**
  - $P(A \mid B)$: what is the probability of A **given B?**
- **Bayes' Rule:** an alternative definition useful for **statistical inference**
  - What is the probability of some **hypothesis, given observed data?**

$$P(A \mid B) := \frac{P(A \cap B)}{P(B)}$$

Def. of Conditional Probability

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

Bayes' Rule

Bayes' Rule Decomposition

"Posterior"
What we want to know

"Likelihood"
How likely is the data under each hypothesis?

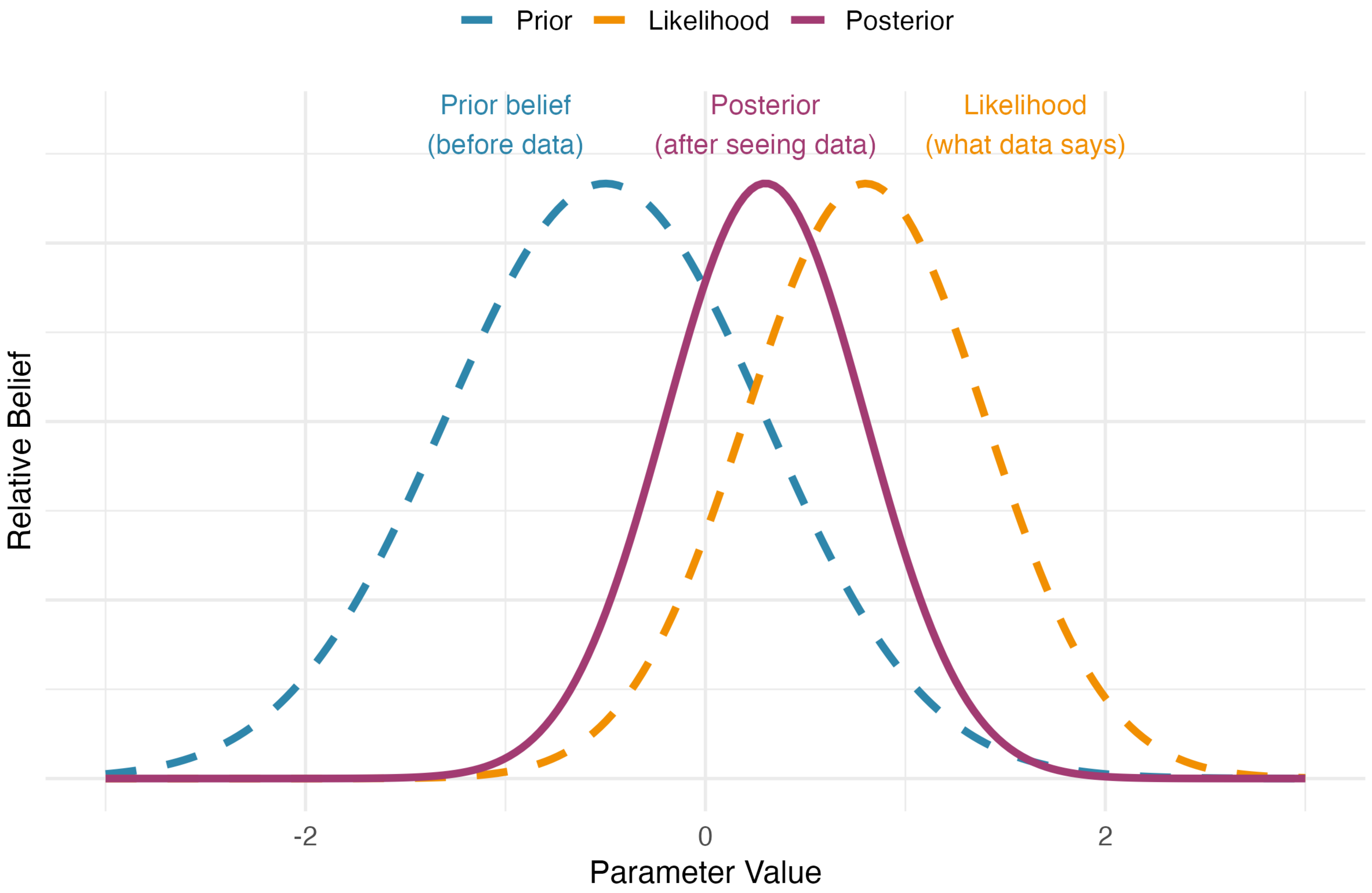
"Prior"
What is our prior belief about H?

$$P(H|D) = \frac{P(D|H)P(H)}{P(D)}$$

What is the probability of a hypothesis **given our data?**

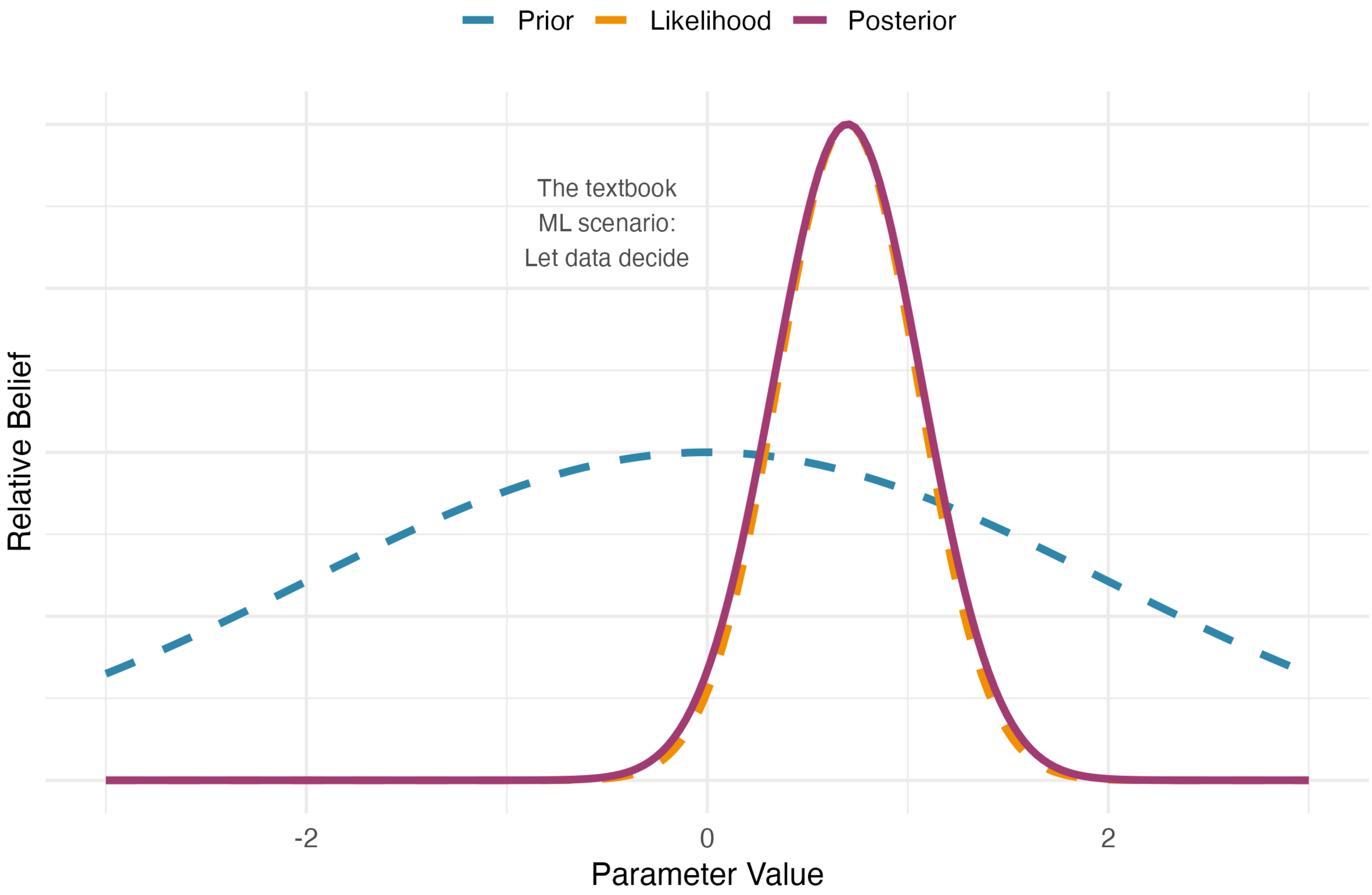
Bayesian Update: Posterior ~ Likelihood × Prior

Posterior is a compromise between what you believed and what the data tells you



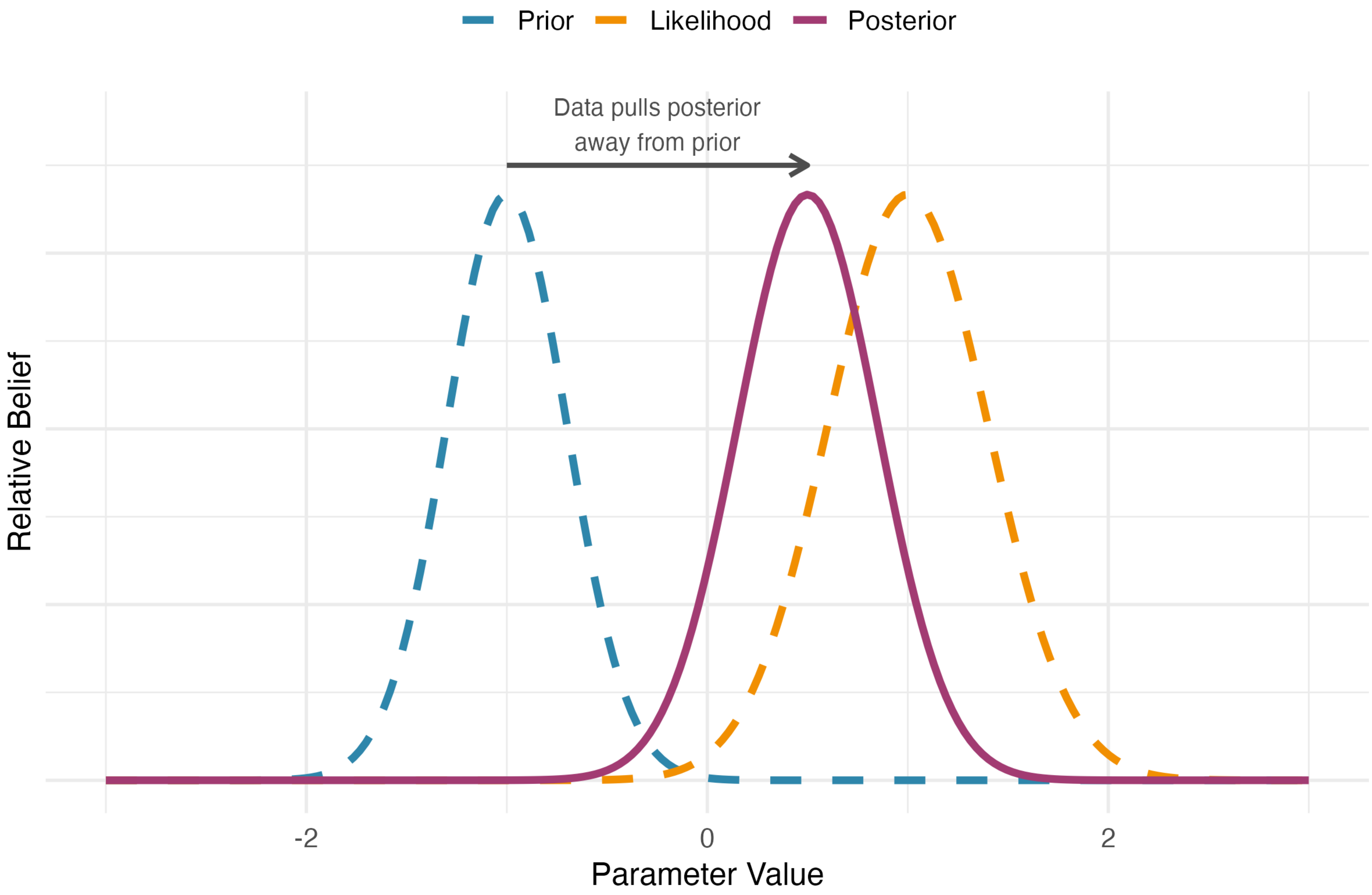
Weak Prior + Lots of Data

Data speaks for itself - posterior ≈ likelihood



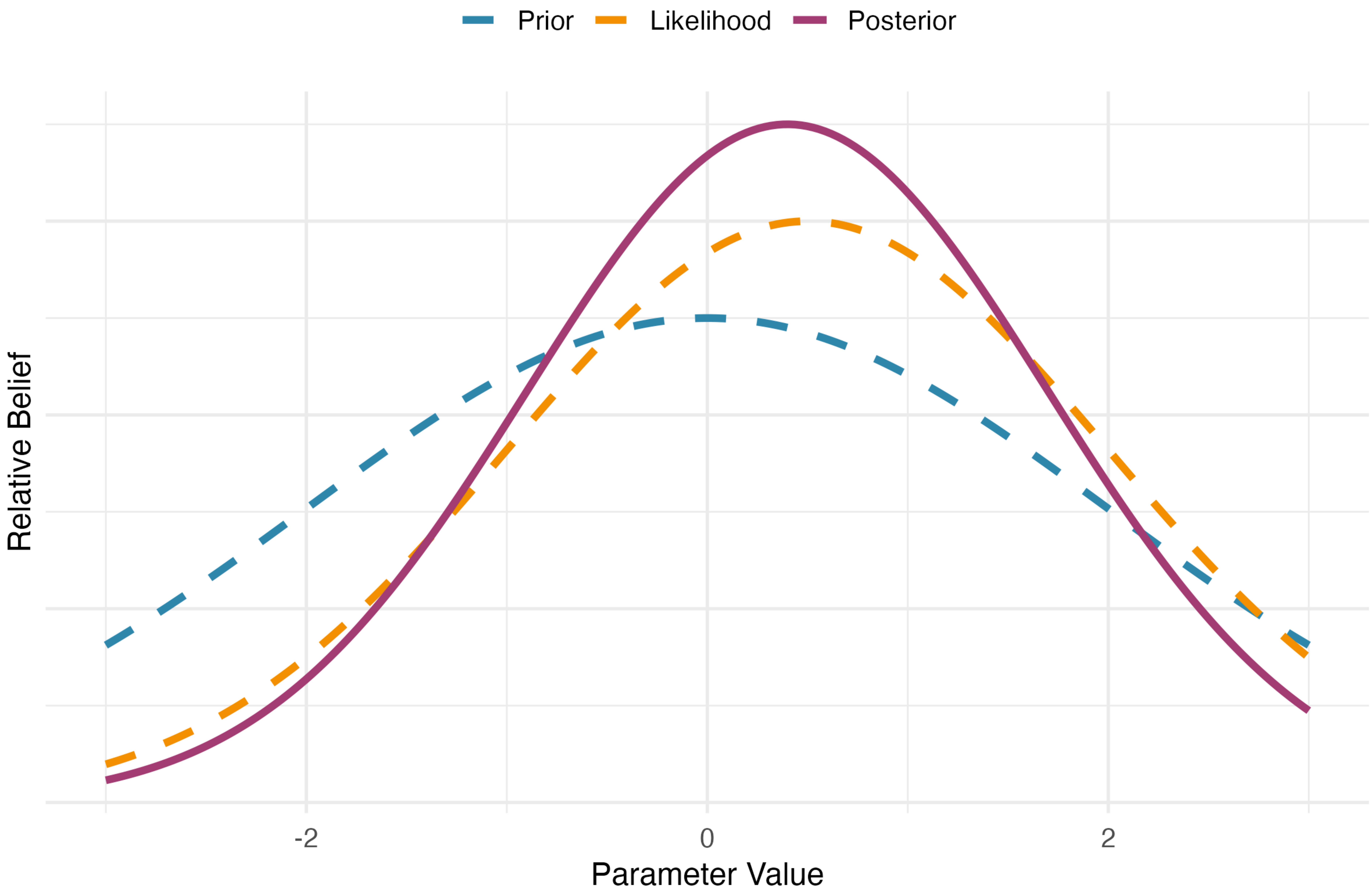
Tight Prior + Lots of Data

Strong evidence can overcome strong assumptions



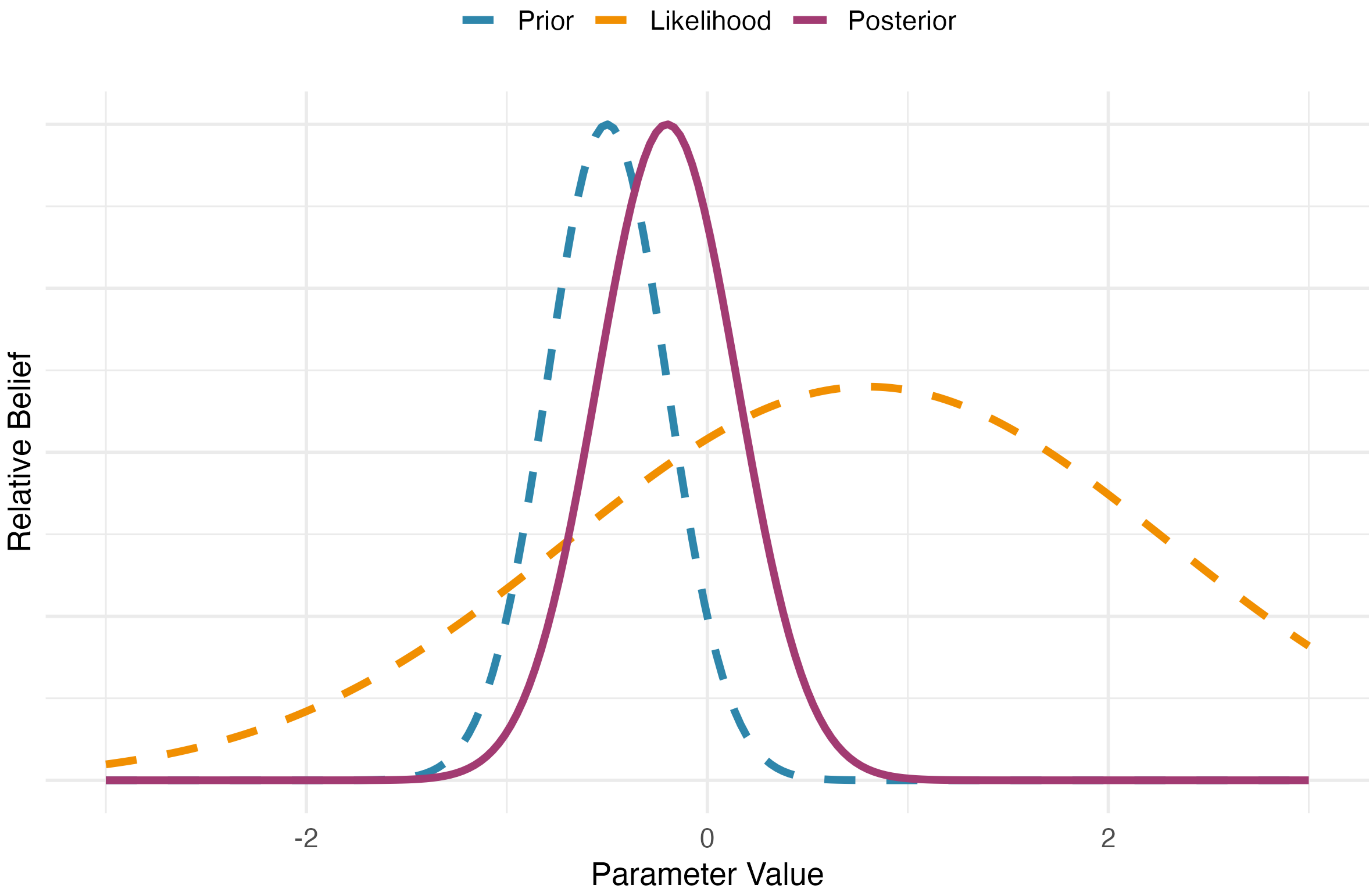
Weak Prior + Little Data

No strong assumptions + weak evidence = high uncertainty

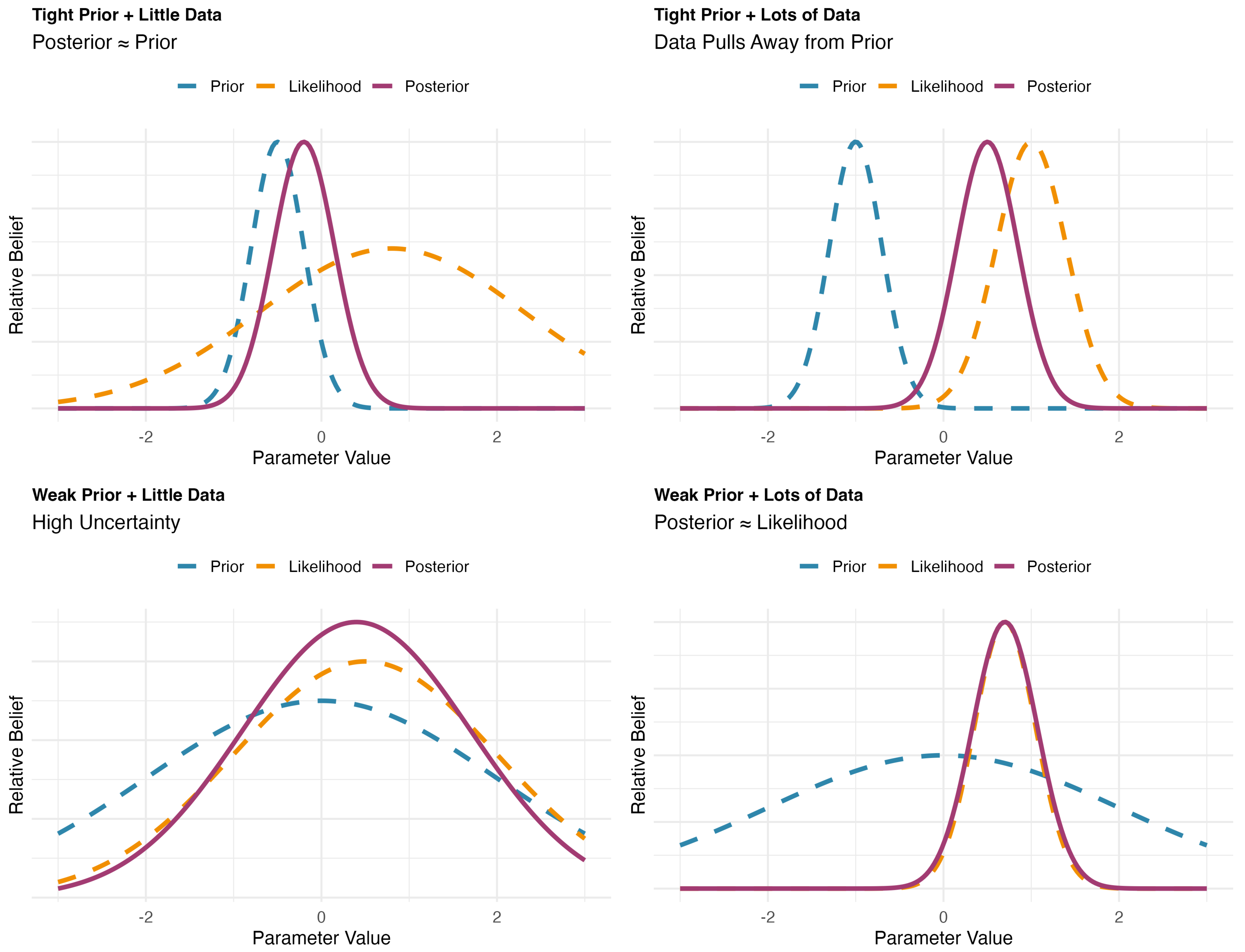


Tight Prior + Little Data

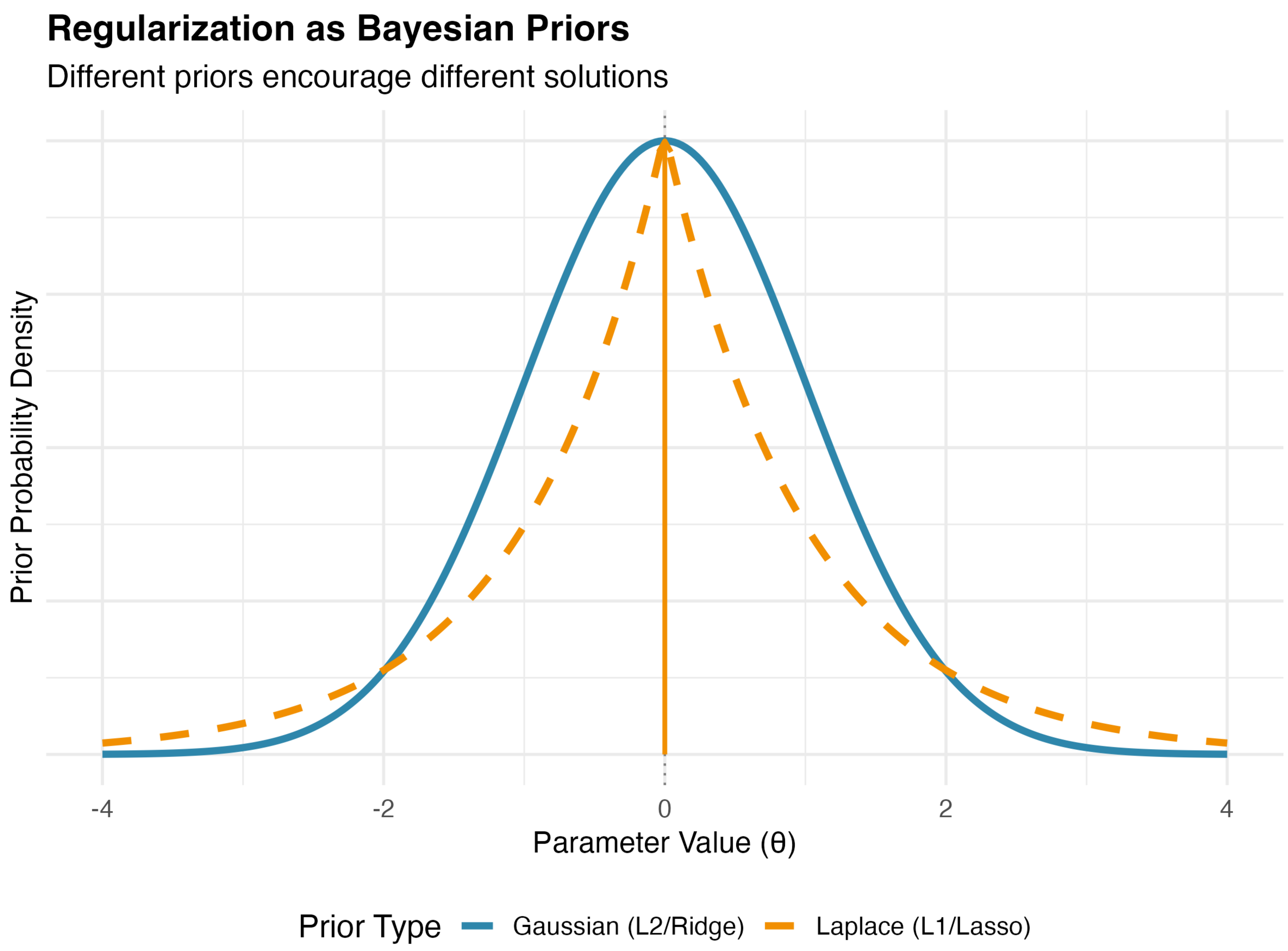
Strong assumptions dominate weak evidence - posterior ≈ prior



Prior Strength × Data Amount: Four Scenarios
Row 1: Little Data (Weak Evidence) | Row 2: Lots of Data (Strong Evidence)

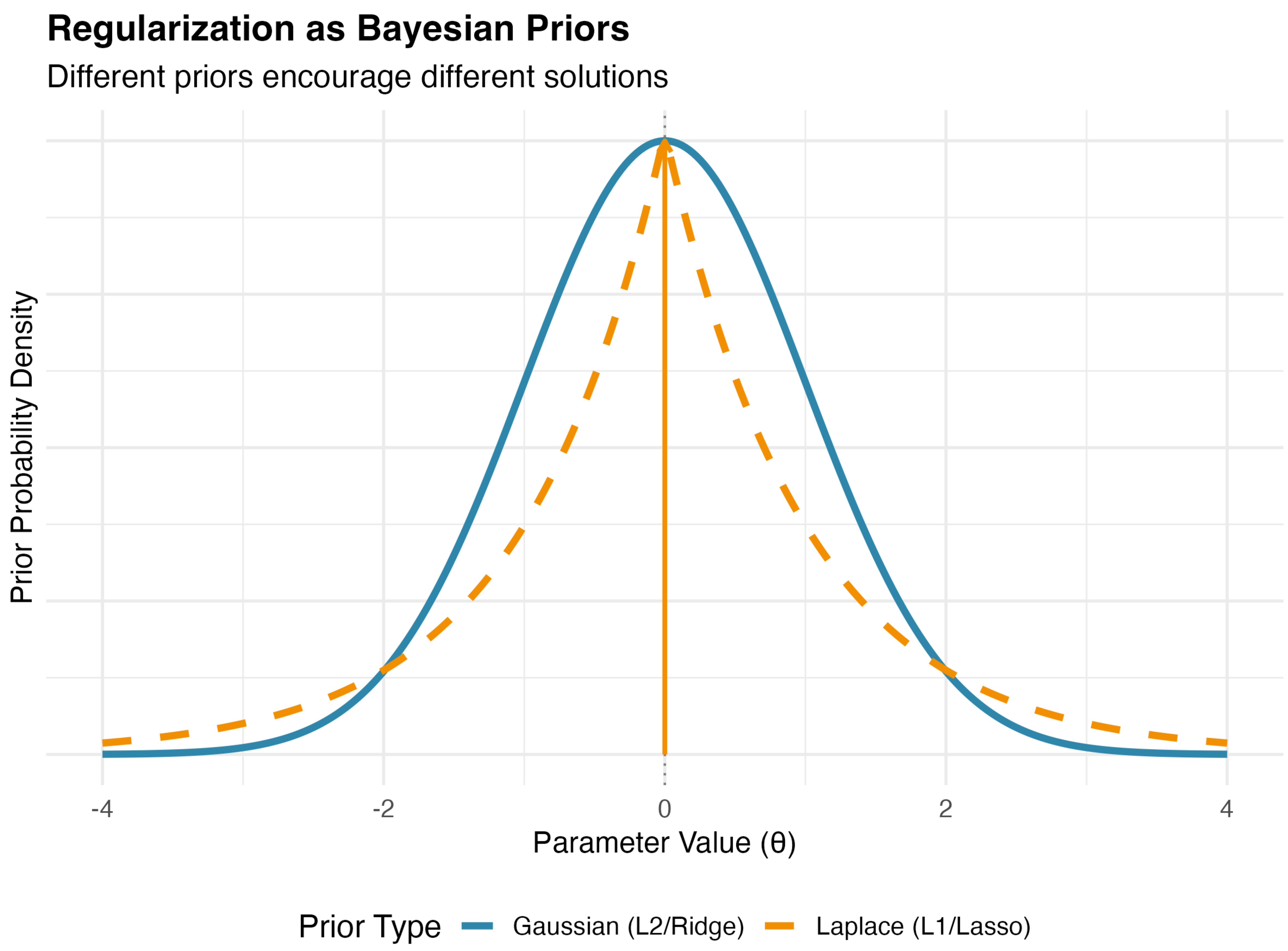


Bayesian Thinking



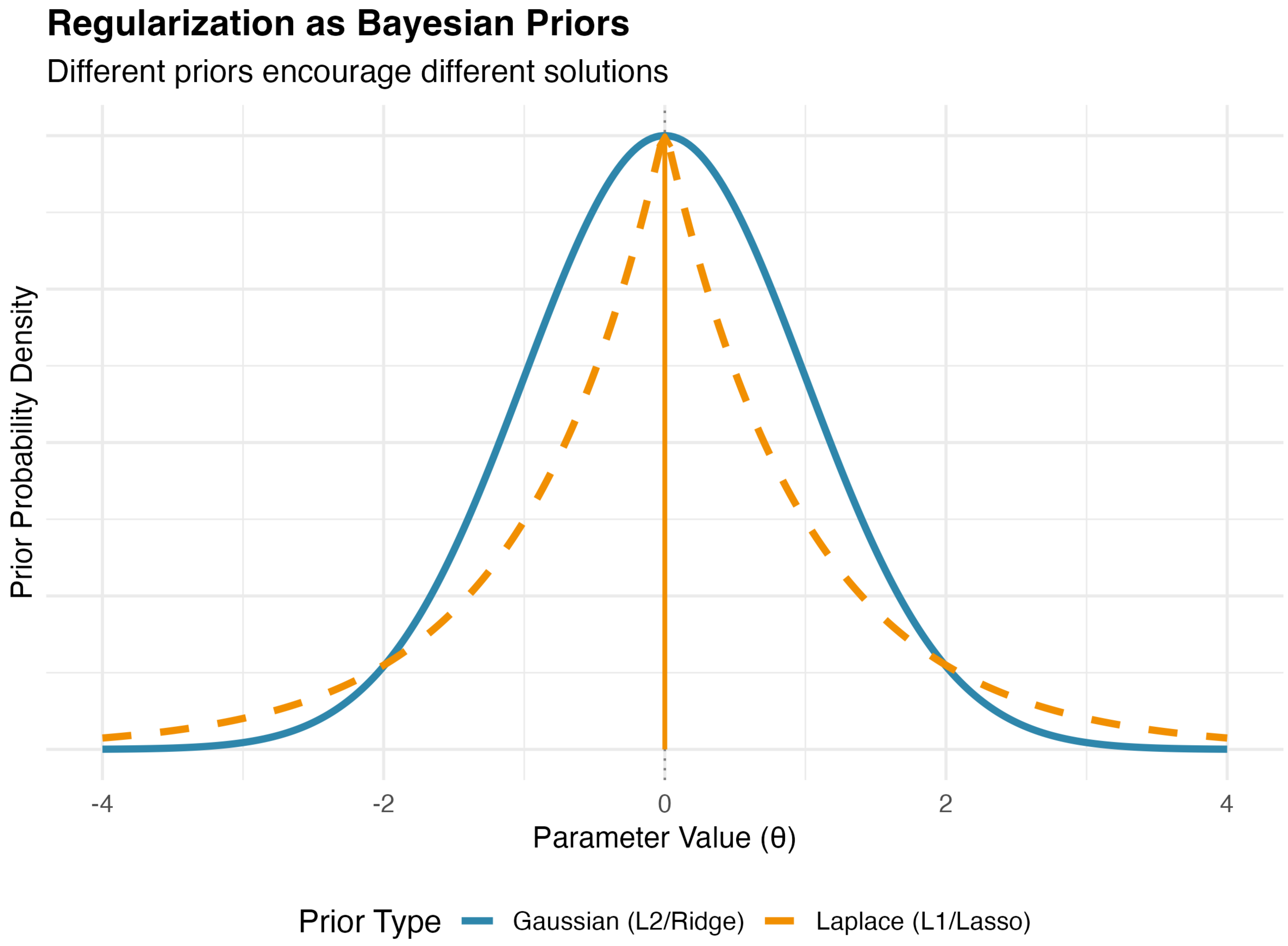
Bayesian Thinking

- Key point: we do **NOT** have to use **Bayesian models** in order to engage in **Bayesian thinking!**
  - Bayesian Machine Learning exists! But the insights **apply to other models too**

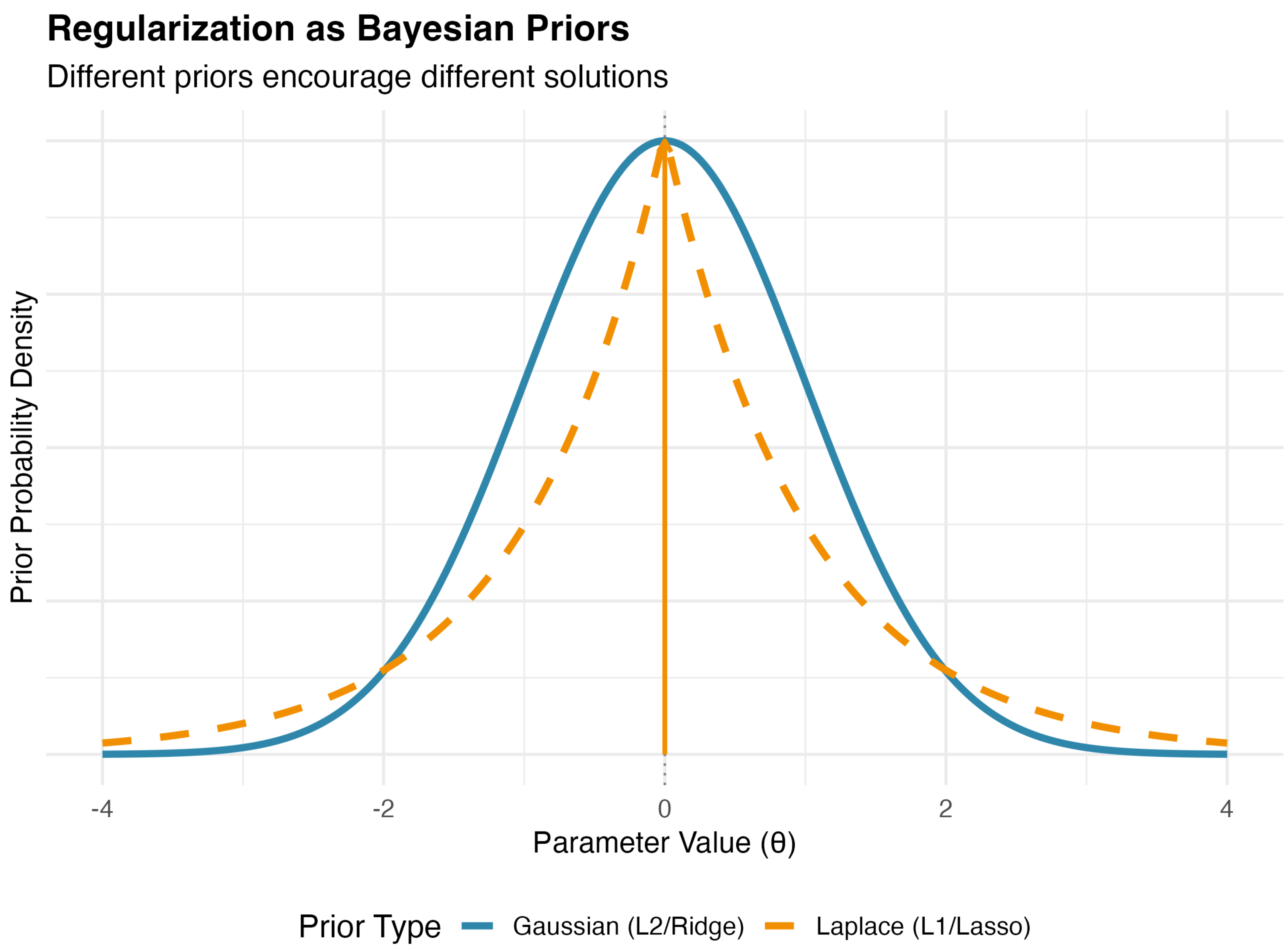


Bayesian Thinking

- Key point: we do **NOT** have to use **Bayesian models** in order to engage in **Bayesian thinking!**
  - Bayesian Machine Learning exists! But the insights **apply to other models too**
- Example: **parameter regularization** is essentially applying a **prior probability on small weights!**

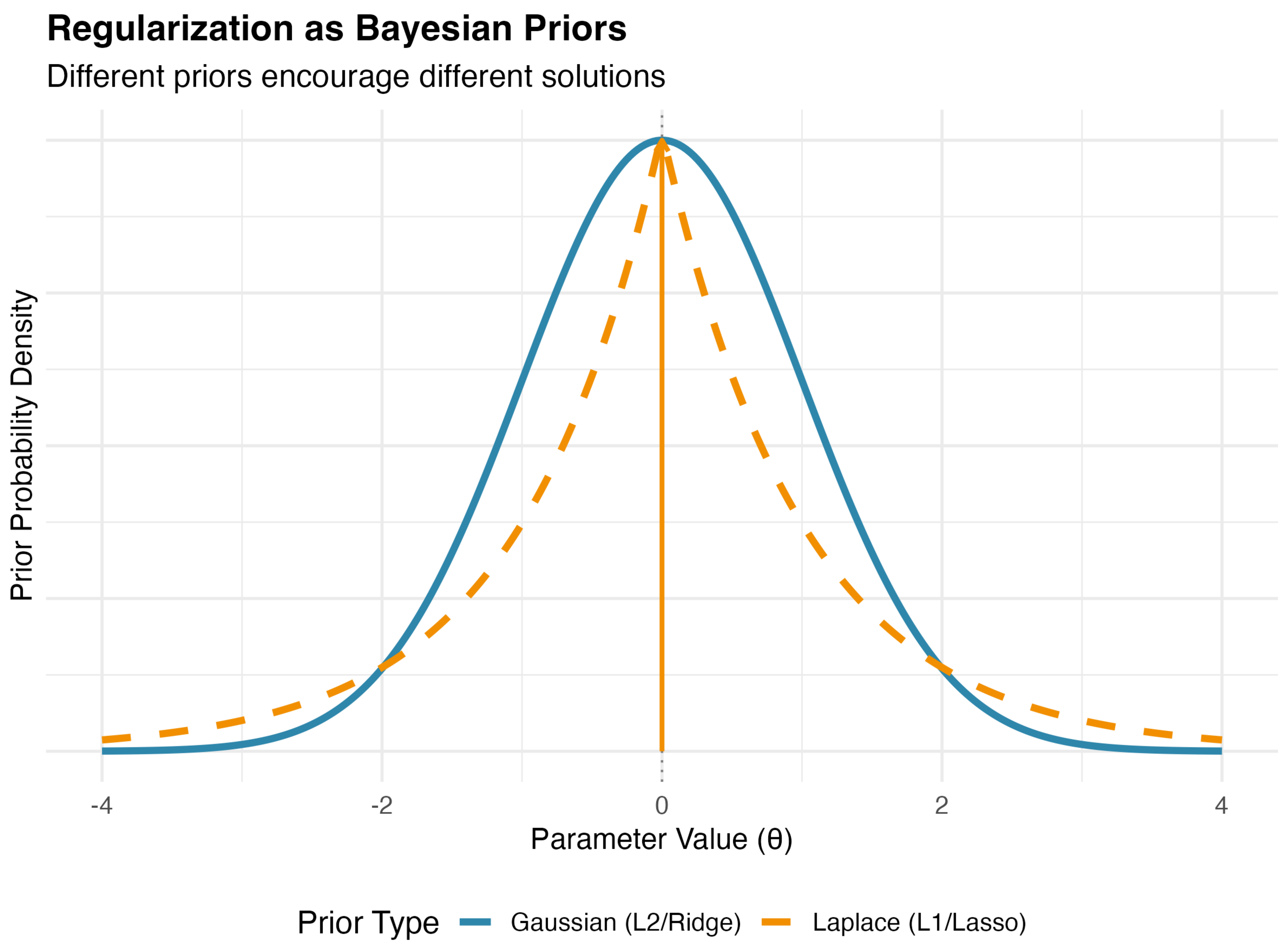


Reminder: Norm Regularization



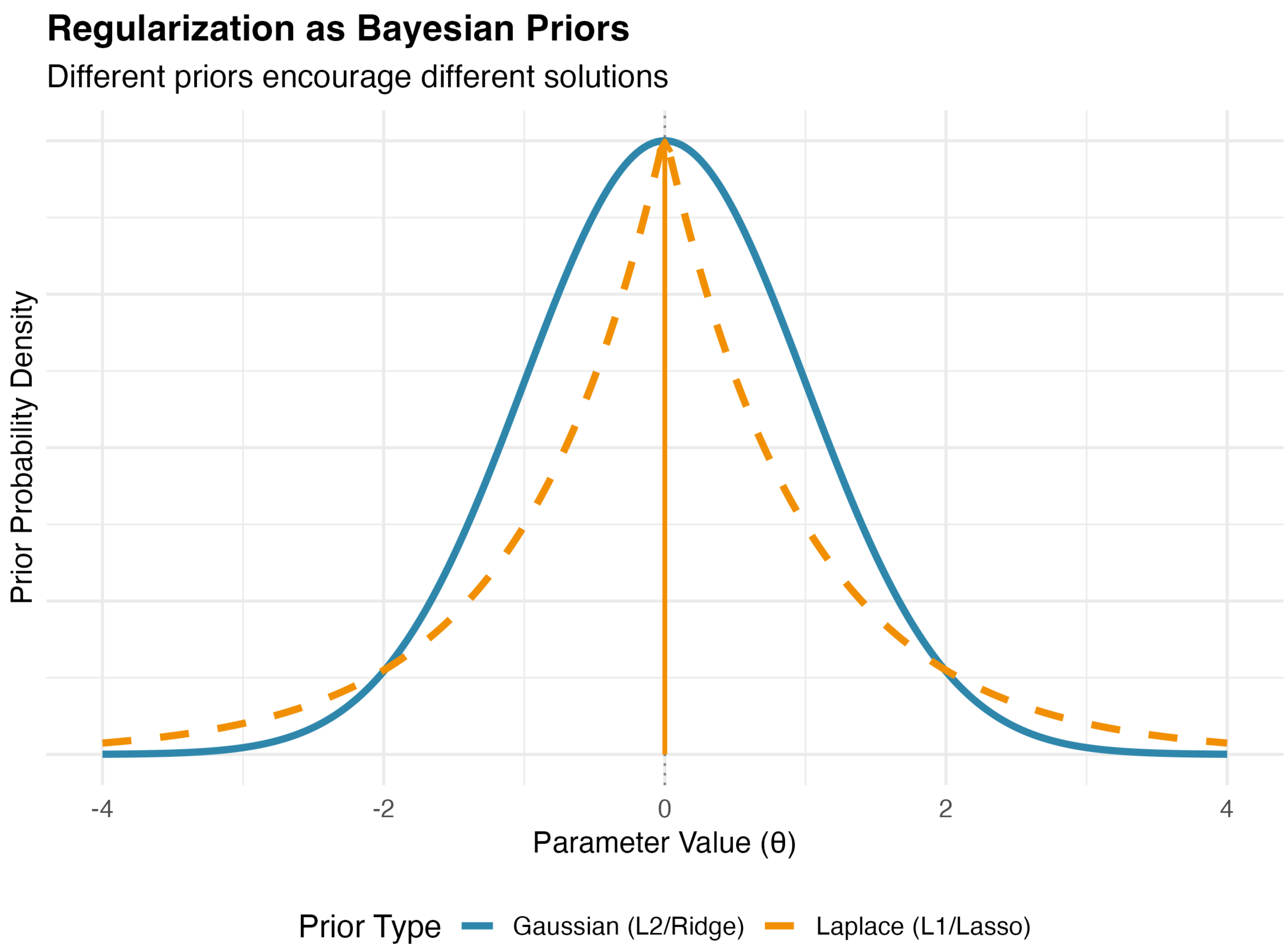
Reminder: Norm Regularization

- **L2 Regularization**: penalizes large weights
  - Strength controlled by **hyperparameter λ:** loss += $\lambda \Sigma \theta_i^2$



Reminder: Norm Regularization

- **L2 Regularization**: penalizes large weights
  - Strength controlled by **hyperparameter λ:** loss += $\lambda \Sigma \theta_i^2$
- **L1 Regularization**: penalizes large weights (in a different way)
  - Tends to **drive some weights to zero** (creating a sparse model)
  - loss += $\lambda \Sigma | \theta_i |$



Regularization Strength (λ) Controls Prior Influence

Larger λ = stronger prior = posterior pulled toward zero