Machine Learning Fundamentals - LS2026
Bayesian learning

Czech Technical University in Prague
V. Franc

Limitations of ML and ERM learning

In MLE or ERM, we learn one parameter vector:

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta} p(T_m \mid \theta), \qquad \hat{\theta}_{\mathrm{ERM}} = \arg\min_{\theta} \frac{1}{m} \sum_{i=1}^{m} \ell(h(x_i;\theta), y_i).$$

This works well when:

- there is enough data,
- uncertainty about parameters is not needed.

But it gives limited answers to questions like:

- What to do when data is scarse ?
- How uncertain are we about the parameters?
- How uncertain is this prediction?
- Which model class is better supported by the data?

Bayesian inference

Interpret the unknown parameter $\theta \in \Theta$ as a **random** variable.

Prior knowledge encoded by selecting:

- Data distribution: parametric family of models $\{p(x, y \,|\, \theta) \,|\, \theta \in \Theta\}$,
- Prior distribution $p(\theta)$ on Θ.

The prior $p(\theta)$ and i.i.d. training data $T_m = \big((x_i, y_i) \,\big|\, i = 1, \ldots, m\big)$ define a *posterior parameter distribution*

$$p(\theta \,|\, T_m) = \frac{p(\theta) p(T_m \,|\, \theta)}{p(T_m)} \quad \text{with} \quad p(T_m \,|\, \theta) = \prod_{i=1}^{m} p(x_i, y_i \,|\, \theta).$$

The probability $p(T_m)$ is obtained by integrating over θ,
i.e. $p(T_m) = \int p(\theta) p(T_m \,|\, \theta)\, d\theta$ and does not depend on θ.

Notice that $p(T_m \,|\, \theta)$ represents the likelihood of the data T_m given the parameter θ.

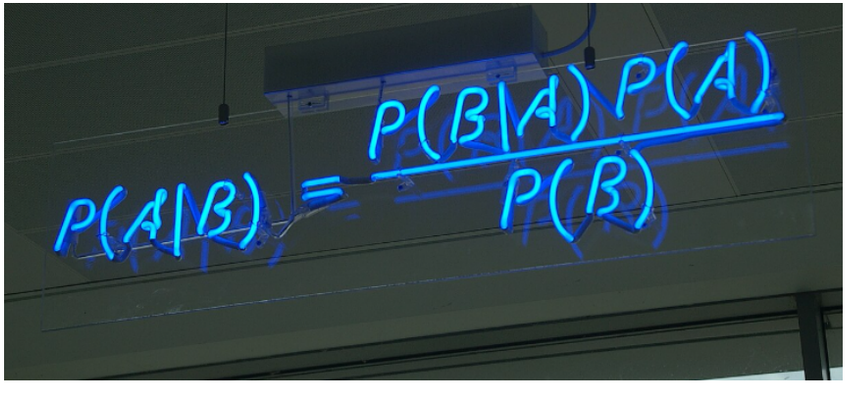
Two interpretations of probability

Frequentist

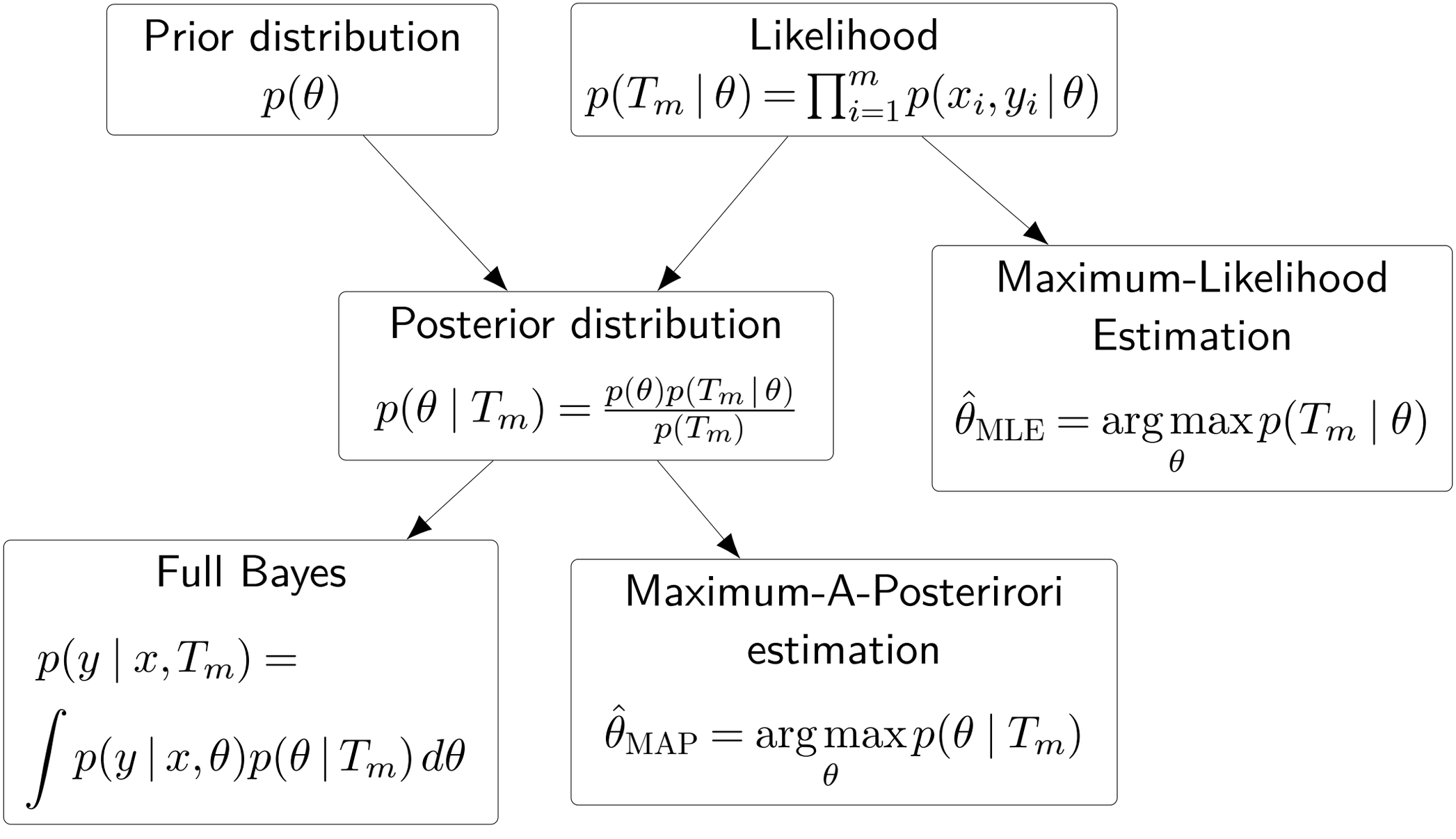
- Probability = long-run frequency
- Parameters θ are *fixed but unknown*
- Data is random
- Inference: point estimators

Bayesian

- Probability = degree of belief
- Parameters θ are *random*
- Data is observed (fixed once seen)
- Inference: a posterior distribution over θ



MLE, MAP, and full Bayesian Learning



MLE, MAP, and full Bayesian Learning

MLE: ignore the prior and choose the best likelihood value:

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta} p(T_m \mid \theta).$$

MAP: include the prior, but still return one parameter value:

$$\hat{\theta}_{\mathrm{MAP}} = \arg\max_{\theta} p(T_m \mid \theta)p(\theta) = \arg\max_{\theta} p(\theta \mid T_m).$$

Full Bayes: keep the whole posterior and predict by averaging, i.e. the predictive distribution is mixture model:

$$p(y \mid x, T_m) = \int p(y \mid x, \theta)p(\theta \mid T_m)\, d\theta.$$

MLE and MAP are optimization problems. Full Bayes is an integration problem.

The Linear-Gaussian Model

We assume a linear model with additive Gaussian noise:

$$y = \mathbf{x}^T \boldsymbol{\theta} + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2)$$

Equivalently, the conditional density of y given $\mathbf{x}$ and θ is

$$p(y \mid \mathbf{x}, \boldsymbol{\theta}) = \mathcal{N}(y \mid \boldsymbol{\theta}^T \mathbf{x}, \sigma^2) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{1}{2\sigma^2} (\boldsymbol{\theta}^T \mathbf{x} - y)^2 \right)$$

Interpretation.

- $\mathbf{x} \in \mathbb{R}^d$ is the input feature vector.
- $\boldsymbol{\theta} \in \mathbb{R}^d$ is the parameter we wish to learn.
- σ^2 is the (assumed known) noise variance.

Training Data and Likelihood

Let $T_m = \big((\mathbf{x}_i, y_i) \mid i = 1, \ldots, m\big)$ be training data drawn i.i.d. from

$$p(\mathbf{x}, y \mid \theta) = p(\mathbf{x})\, p(y \mid \mathbf{x}, \theta),$$

where $p(\mathbf{x})$ does *not* depend on the parameters.

The likelihood of $\boldsymbol{\theta}$ given T_m factorizes:

$$p(T_m \mid \boldsymbol{\theta}) = \prod_{i=1}^{m} p(\mathbf{x}_i)\, p(y_i \mid \mathbf{x}_i, \boldsymbol{\theta}) = p(\mathbf{X}_m)\, \mathcal{N}(\mathbf{y}_m \mid \mathbf{X}_m \boldsymbol{\theta}, \sigma^2 I)$$

with the design matrix and target vector

$$\mathbf{X}_m = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_m^T \end{bmatrix}, \qquad \mathbf{y}_m = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}.$$

Maximum-Likelihood Estimation

The MLE maximizes the (log-)likelihood:

$$\hat{\boldsymbol{\theta}}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} \; \log p(T_m \mid \boldsymbol{\theta}).$$

Because $p(\mathbf{X}_m)$ does not depend on $\boldsymbol{\theta}$, dropping it leaves

$$\log p(T_m \mid \boldsymbol{\theta}) = \text{const} - \frac{1}{2\sigma^2} \|\mathbf{y}_m - \mathbf{X}_m \boldsymbol{\theta}\|^2.$$

Hence

$$\hat{\boldsymbol{\theta}}_{\mathrm{ML}} = \arg\min_{\boldsymbol{\theta}} \; \frac{1}{2\sigma^2} \|\mathbf{y}_m - \mathbf{X}_m \boldsymbol{\theta}\|^2 = (\mathbf{X}_m^T \mathbf{X}_m)^{-1} \mathbf{X}_m^T y.$$

Takeaway. For Gaussian noise, MLE is exactly *ordinary least squares*.

Maximum-A-Posteriori (MAP) Estimate

The MAP estimate maximizes the posterior:

$$\hat{\boldsymbol{\theta}}_{\mathrm{MAP}} = \arg\max_{\boldsymbol{\theta}} \; \log p(\boldsymbol{\theta} \mid T_m) = \arg\max_{\boldsymbol{\theta}} \; \log \left[\frac{p(\boldsymbol{\theta})\, p(T_m \mid \boldsymbol{\theta})}{p(T_m)} \right].$$

Since $p(T_m)$ does not depend on θ, it cancels out:

$$\hat{\boldsymbol{\theta}}_{\mathrm{MAP}} = \arg\max_{\boldsymbol{\theta}} \; \big[\log p(\boldsymbol{\theta}) + \log p(T_m \mid \boldsymbol{\theta}) \big].$$

Note. MAP does *not* require computing the marginal likelihood $p(T_m)$. The prior $p(\boldsymbol{\theta})$ acts as a regularizer.

MAP with a Gaussian Prior $\Rightarrow$ Ridge Regression

Place a Gaussian prior on the parameters:

$$p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta} \mid 0, \tau^2 \mathbf{I}).$$

Combining with the Gaussian likelihood and dropping constants:

$$\begin{aligned} \hat{\boldsymbol{\theta}}_{\text{MAP}} &= \arg\min_{\boldsymbol{\theta}} \left[\frac{1}{2\sigma^2} \|\mathbf{y}_m - \mathbf{X}_m \boldsymbol{\theta}\|^2 + \frac{1}{2\tau^2} \|\boldsymbol{\theta}\|^2 \right] \\ &= \left(\mathbf{X}_m^T \mathbf{X}_m + \frac{\sigma^2}{\tau^2} I \right)^{-1} \mathbf{X}_m^T \mathbf{y}_m. \end{aligned}$$

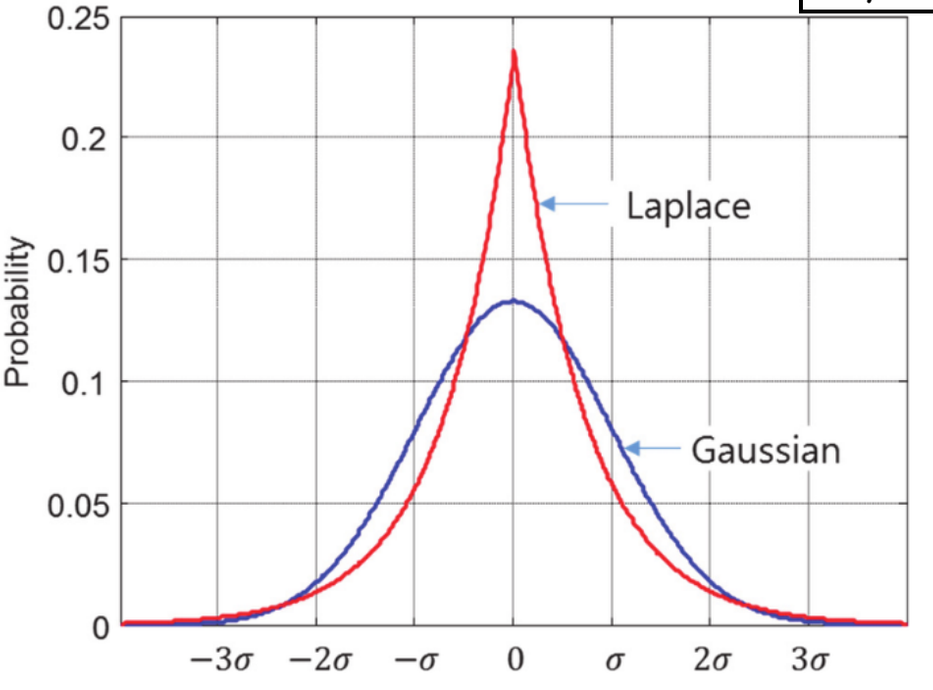
This is exactly **Ridge regression** with regularization parameter $\lambda = \frac{\sigma^2}{\tau^2}$.

- Small τ^2 (strong prior) $\Rightarrow$ large $\lambda \Rightarrow$ heavy shrinkage.
- Large τ^2 (weak prior) $\Rightarrow$ small $\lambda \Rightarrow$ MAP $\rightarrow$ MLE.

MAP with a Laplace Prior $\Rightarrow$ Lasso

Place a Laplace prior on each component:

$$p(\theta_j) = \frac{1}{2b}\exp\left(-\frac{|\theta_j|}{b}\right), \qquad j = 1, \ldots, d.$$



Then $\log p(\boldsymbol{\theta}) = -\frac{1}{b}\|\boldsymbol{\theta}\|_1 + \text{const}$, and

$$\hat{\boldsymbol{\theta}}_{\text{MAP}} = \arg\min_{\boldsymbol{\theta}} \left[\frac{1}{2\sigma^2} \|\mathbf{y}_m - \mathbf{X}_m\boldsymbol{\theta}\|^2 + \frac{1}{b}\|\boldsymbol{\theta}\|_1 \right]$$

This is exactly **Lasso** with regularization parameter $\lambda = \frac{2\sigma^2}{b}$.

The ℓ_1 penalty promotes *sparse* solutions — many components of $\hat{\boldsymbol{\theta}}_{\text{MAP}}$ become exactly zero.

Full Bayesian Inference: Posterior over θ

Take again the Gaussian prior

$$p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta} \mid 0, \tau^2 I).$$

The posterior over parameters has a closed form:

$$p(\boldsymbol{\theta} \mid T_m) = \frac{p(T_m \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{p(T_m)} = \mathcal{N}(\boldsymbol{\theta} \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$$

with

$$\boldsymbol{\Sigma}_m = \left(\frac{1}{\sigma^2} \mathbf{X}_m^T \mathbf{X}_m + \frac{1}{\tau^2} I \right)^{-1}, \qquad \boldsymbol{\mu}_m = \frac{1}{\sigma^2} \boldsymbol{\Sigma}_m \mathbf{X}_m^T \mathbf{y}_m.$$

Observe. The posterior mean $\boldsymbol{\mu}_m$ coincides with the Ridge / MAP solution. The Bayesian view additionally provides the *covariance* $\boldsymbol{\Sigma}_m$ — a measure of uncertainty about $\boldsymbol{\theta}$.

Conjugate priors

Definition: A prior $p(\boldsymbol{\theta})$ is *conjugate* to a likelihood $p(T_m \mid \boldsymbol{\theta})$ if the posterior $p(\boldsymbol{\theta} \mid T_m)$ belongs to the same family as the prior.

Likelihood	**Conjugate prior**	**Application**
Bernoulli / Binomial	Beta	coin flips
Multinomial	Dirichlet	topic models
Poisson	Gamma	count data
Gaussian (known σ^2)	Gaussian	linear regression
Gaussian (unknown μ, σ^2)	Normal-Inverse-Gamma	general Gaussian

- Why useful? Closed-form posterior requires no integrals to compute. Posterior parameters are simple updates of prior parameters.
- But many important ML models are not conjugate: logistic regression, neural networks.

Predictive Posterior

To predict y at a new input $\mathbf{x}$ we marginalize over $\boldsymbol{\theta}$:

$$\begin{aligned} p(y \mid \mathbf{x}, T_m) &= \int p(y \mid \mathbf{x}, \boldsymbol{\theta})\, p(\boldsymbol{\theta} \mid T_m)\, d\boldsymbol{\theta} \\ &= \int \mathcal{N}(y \mid \boldsymbol{\theta}^T \mathbf{x}, \sigma^2)\, \mathcal{N}(\boldsymbol{\theta} \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)\, d\boldsymbol{\theta} \\ &= \mathcal{N}\left(y \mid \boldsymbol{\mu}_m^T \mathbf{x},\ \sigma^2 + \mathbf{x}^T \boldsymbol{\Sigma}_m \mathbf{x}\right). \end{aligned}$$

Decomposition of the predictive variance:

$$\underbrace{\sigma^2}_{\text{aleatoric (noise)}} + \underbrace{\mathbf{x}^T \boldsymbol{\Sigma}_m \mathbf{x}}_{\text{epistemic (parameter uncertainty)}}.$$

- The aleatoric term is irreducible noise.
- The epistemic term *shrinks* as $m \to \infty$: more data $\Rightarrow$ smaller $\boldsymbol{\Sigma}_m$.
- Crucially, it is *larger for* $\mathbf{x}$ *far from training data* — the Bayesian model "knows what it doesn't know."

Linear Gaussian model: numerical example

Setup: Model $y = \boldsymbol{\phi}(x)^T \boldsymbol{\theta} + \varepsilon, \varepsilon \sim \mathcal{N}(0, \sigma^2)$ with $\sigma = 0.3$ and polynomial features $\boldsymbol{\phi}(x) = [1, x, x^2, x^3]$.

Maximum-Likelihood

Posterior:

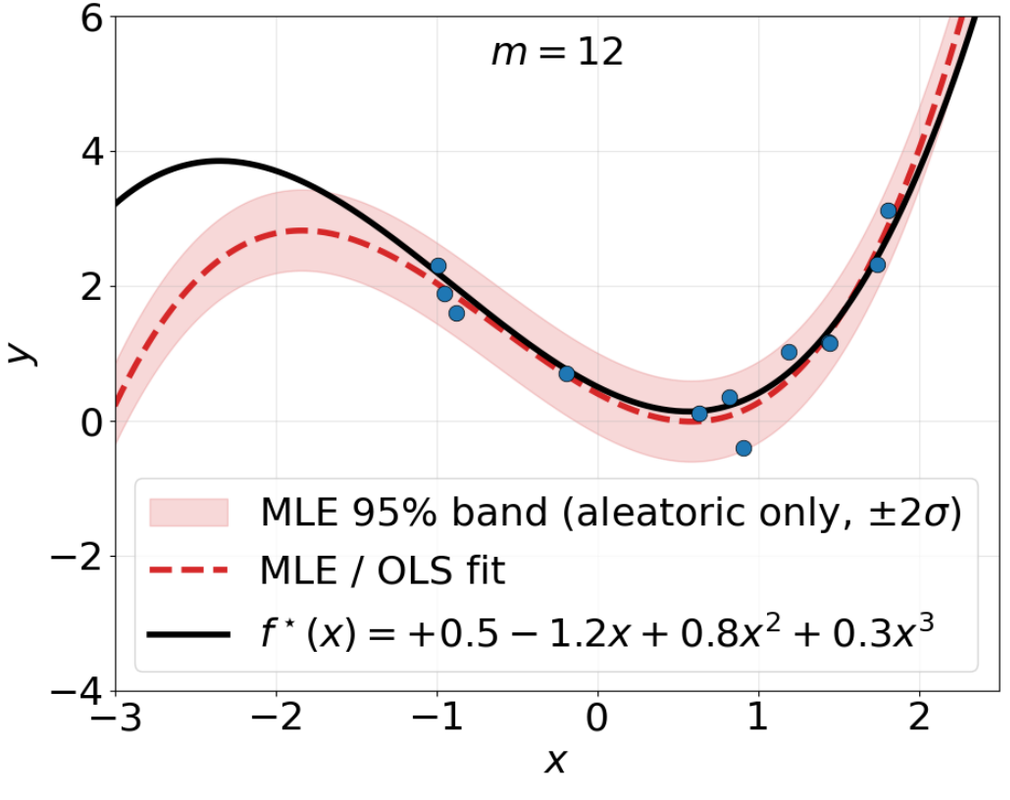
$$p(y \mid x, \theta) = \mathcal{N}(y \mid \boldsymbol{\phi}(x)^T \hat{\boldsymbol{\theta}}_{\mathrm{ML}}, \sigma^2)$$

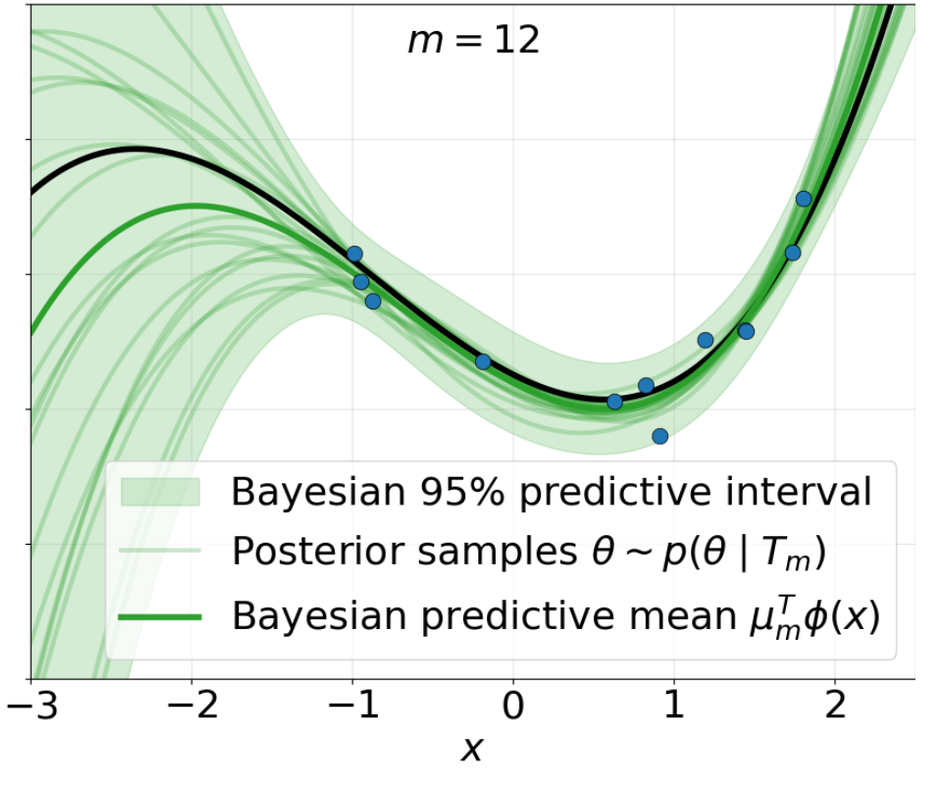
Full Bayesian inference

Prior: $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta} \mid 0, \tau^2 \mathbf{I})$ with $\tau = 1.0$

Predictive distribution:

$$p(y \mid x, T_m) = \mathcal{N}\left(y \mid \boldsymbol{\phi}(x)^T \boldsymbol{\mu}_m,\ \sigma^2 + \boldsymbol{\phi}(x)^T \Sigma_m \boldsymbol{\phi}(x)\right)$$





Linear regression: numerical example

Setup: Model $y = \boldsymbol{\phi}(x)^T\boldsymbol{\theta} + \varepsilon, \varepsilon \sim \mathcal{N}(0, \sigma^2)$ with $\sigma = 0.3$ and polynomial features $\boldsymbol{\phi}(x) = [1, x, x^2, x^3]$.

Maximum-Likelihood

Posterior:

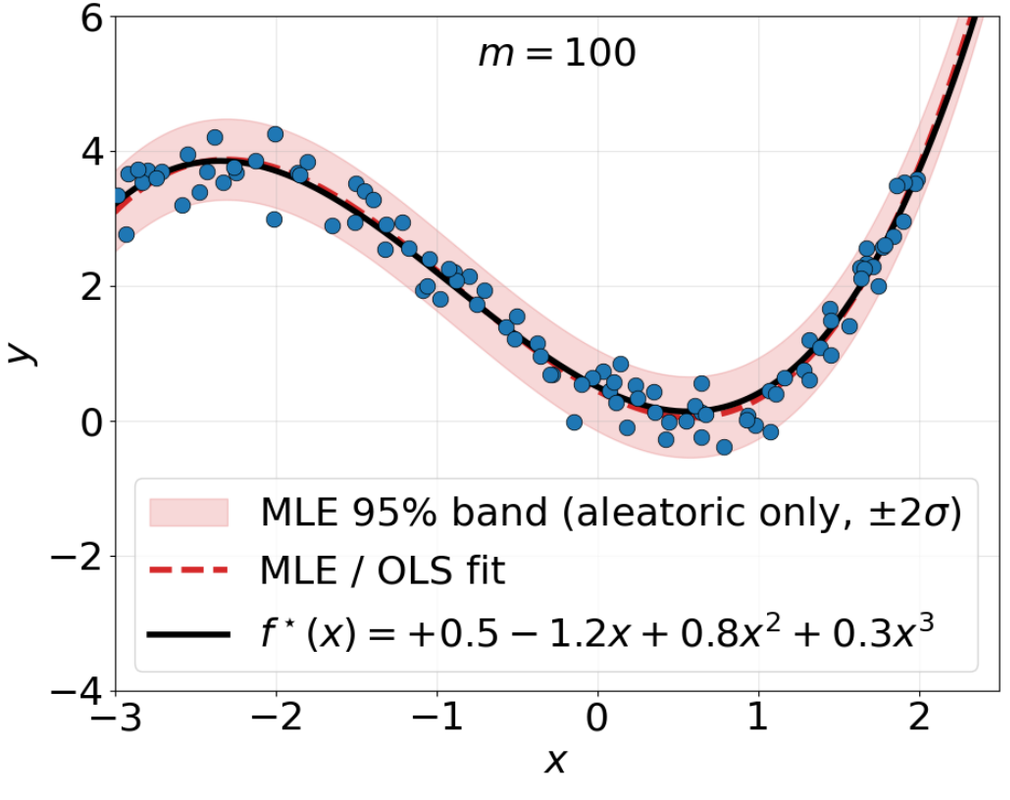
$$p(y \mid x, \boldsymbol{\theta}) = \mathcal{N}(y \mid \boldsymbol{\phi}(x)^T\hat{\boldsymbol{\theta}}_{\mathrm{ML}}, \sigma^2)$$

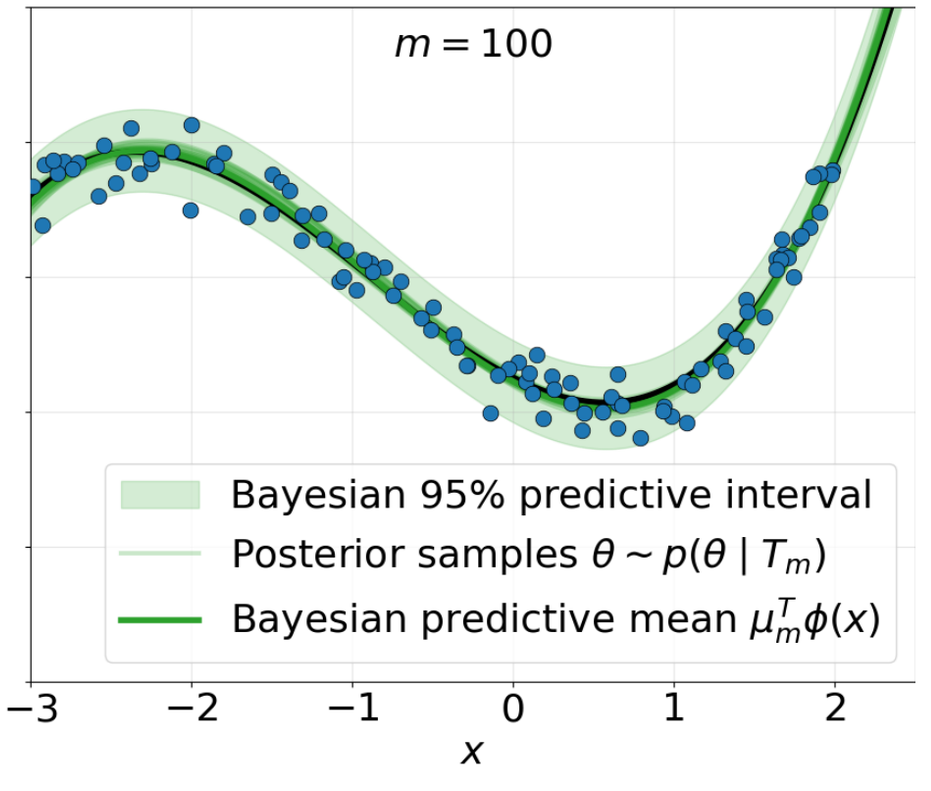
Full Bayesian inference

Prior: $p(\boldsymbol{\theta}) = \mathcal{N}(\theta \mid 0, \tau^2 I)$ with $\tau = 1.0$

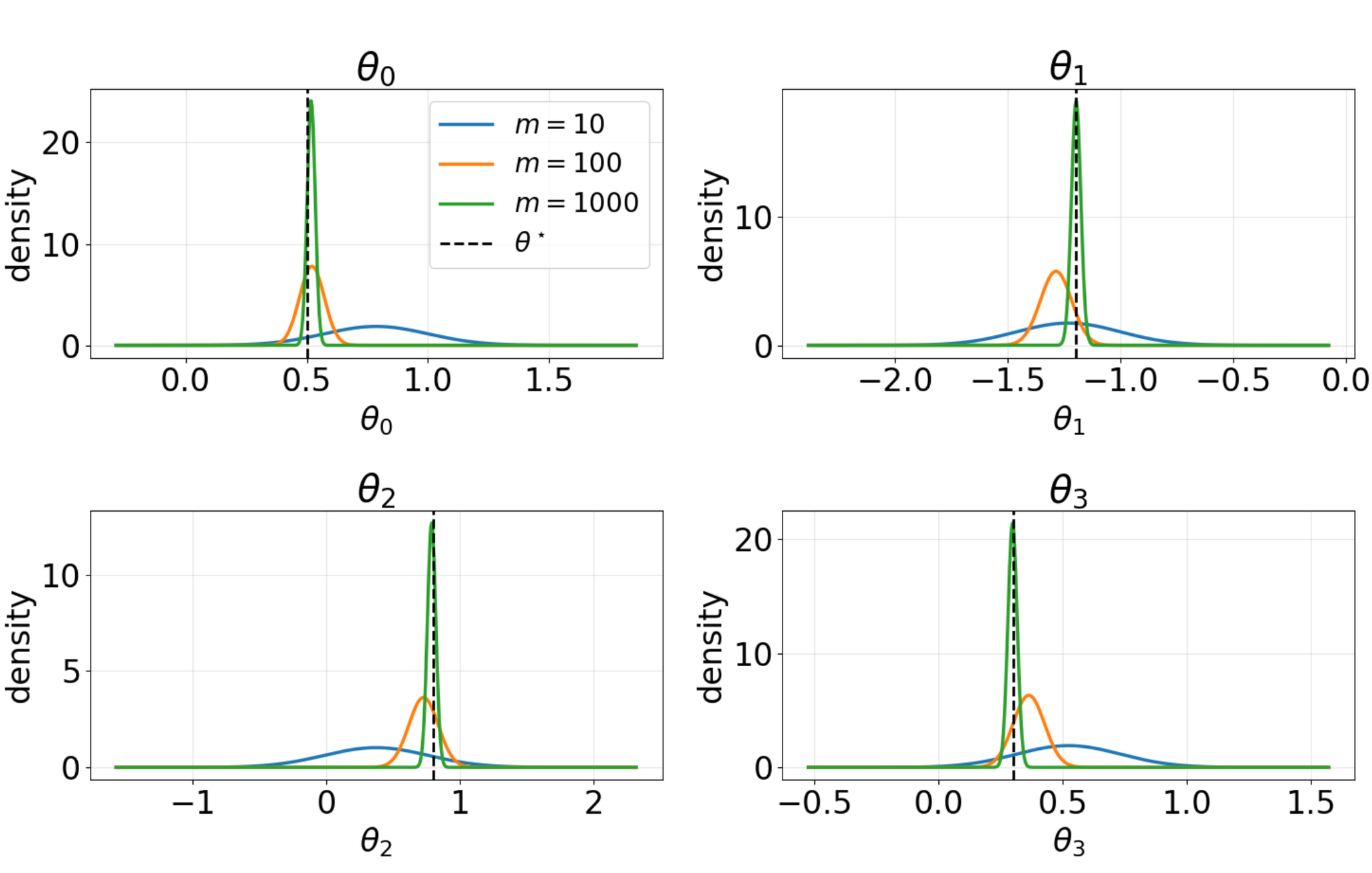
Predictive distribution:

$$p(y \mid x, T_m) = \mathcal{N}\left(y \mid \boldsymbol{\phi}(x)^T\boldsymbol{\mu}_m,\ \sigma^2 + \boldsymbol{\phi}(x)^T\Sigma_m\boldsymbol{\phi}(x)\right)$$





Linear regression: numerical example



Summary: Three Views, One Model

Method	Optimizes	Output
MLE	$\max_\theta \log p(T_m \mid \theta)$	point estimate $\hat{\theta}_{\mathrm{ML}}$
MAP	$\max_\theta [\log p(\theta) + \log p(T_m \mid \theta)]$	point estimate $\hat{\theta}_{\mathrm{MAP}}$
Bayes	$p(y \mid x, T_m) = \int p(y \mid, x, \theta) p(\theta \mid T_m) d\theta$	full distribution

Hierarchy of richness:

- **MLE** $\subset$ **MAP**: MLE is MAP with a flat prior.
- **MAP** $\subset$ **Bayes**: MAP is the mode of the Bayesian posterior; Bayes additionally captures uncertainty.

For Gaussian linear regression specifically:

MLE = OLS, MAP (Gaussian prior) = Ridge, MAP (Laplace prior) = Lasso.

What we did not cover

Bayesian learning is a much broader field. Four important directions left for further study:

- **Bayesian model selection.** Compare candidate models $\mathcal{M}_1, \ldots, \mathcal{M}_K$ via the marginal likelihood $p(T_m \,|\, \mathcal{M}_k) = \int p(T_m \,|\, \theta, \mathcal{M}_k)\, p(\theta \,|\, \mathcal{M}_k)\, d\theta$. Implements an automatic Occam's razor — balances data fit against model complexity without a held-out set.
- **Gaussian Processes.** Perform the Bayesian linear regression in high dimensional space via using the kernel function $k(x, x') = \langle \phi(x), \phi(x') \rangle$.
- **Bayesian neural networks.** Treat network weights as random variables with a prior; the predictive distribution averages over weight configurations. Provides uncertainty estimates that a standard NN lacks.
- **Approximate inference** (when the posterior has no closed form): Laplace approximation, Variational inference, Markov Chain Monte Carlo.

Conclusions

Bayesian learning treats the parameter θ as a random variable and updates our beliefs about it via Bayes' rule: $p(\theta \,|\, T_m) \propto p(T_m \,|\, \theta)\, p(\theta)$.

Three views of the same problem:

- **MLE** — ignore the prior, return a point estimate.
- **MAP** — include the prior, still a point estimate; recovers regularization (Ridge, Lasso) as a special case.
- **Full Bayes** — keep the whole posterior; predict by averaging

$$p(y \,|\, x, T_m) = \int p(y \,|\, x, \theta)\, p(\theta \,|\, T_m)\, d\theta$$

The Bayesian predictive distribution captures *aleatoric* (noise) and *epistemic* (parameter) uncertainty – the model knows what it does not know.

Bayesian learning is most valuable when data is scarce, uncertainty matters, or prior knowledge is available.