Probabilistic AI –what is it?

Arto Klami

Department of Computer Science

University of Helsinki

ProbAi — 2022

FCAI Finnish Center for Artificial Intelligence

UNIVERSITY OF HELSINKI

How does the sentence end?

Summer schools are ____________

How does the sentence end?

Summer schools are ______________

fun

interesting

useful

boring

out of order

…

How does the sentence end?

Summer schools are ______________

fun	20%
interesting	10%
useful	15%
boring	3%
out of order	0.001%
…	…

Probabilistic AI

AI methods need to acknowledge that

a) Multiple solutions are possible
b) Some of them are more likely than others
c) We are usually unsure of the probabilities as well

Probabilistic AI

AI methods need to acknowledge that

a) Multiple solutions are possible
b) Some of them are more likely than others
c) We are usually unsure of the probabilities as well

AI methods should

a) Explicitly represent and manipulate the probabilities in a consistent manner
b) Update estimates based on data
c) Communicate the uncertainty for the user and/or downstream tasks

Machine learning and AI

We mostly talk about machine learning methods, as one core element of AI, but many of the same concepts apply for other forms of AI

Probabilistic AI necessarily builds on standard probability calculus and statistics, but our focus is in

- General-purpose methods, not in description of a specific process
- Computationally practical methods

Supervised machine learning

1. Data comes from unknown distribution
2. Some parametric function from inputs to outputs
3. Parameters chosen to minimize some loss on training data

$$\{y, x\} \sim p(y, x)$$

$$\hat{y} = f(x, \theta)$$

$$\hat{\theta} = \arg\min \mathcal{L}(D, \theta)$$

$$\mathcal{D} = \{y_n, x_n\}_{n=1}^{N}$$

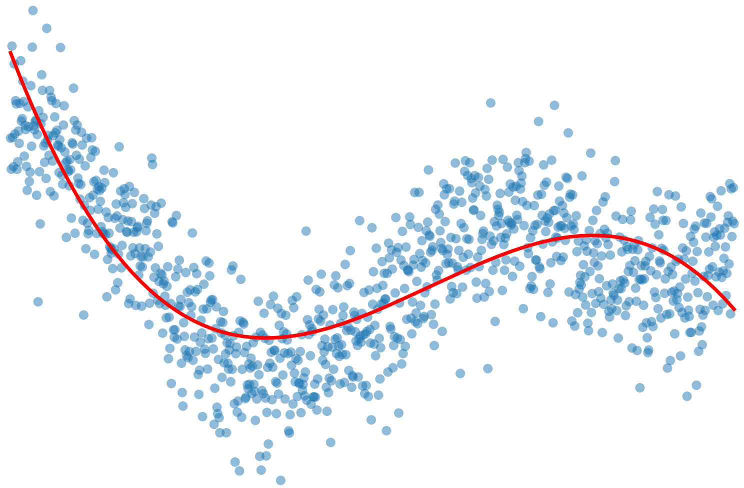
Generalization

We mostly care about how well the model works for new inputs drawn from the same distribution $p(y, x)$

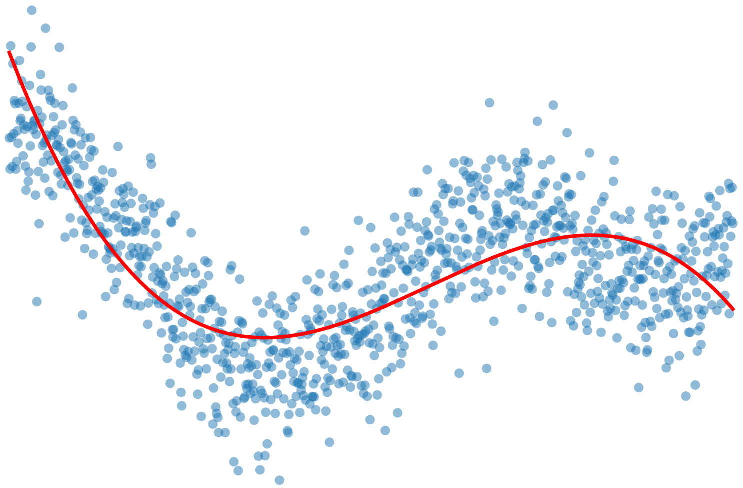
Models trained on infinite data work well for future samples

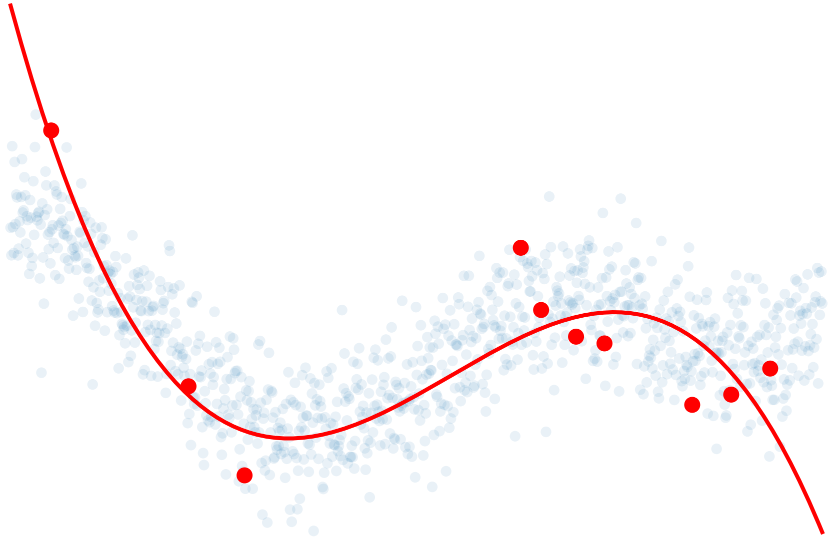
How do we ensure this if only having a finite collection of samples from the distribution?

Finite data means uncertainty

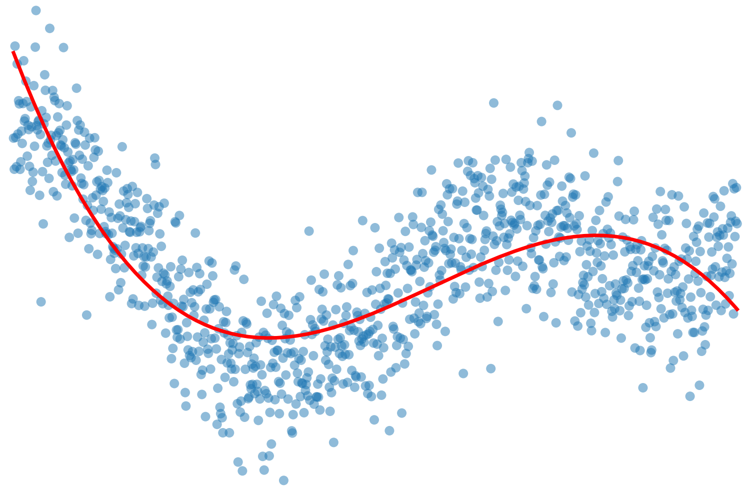


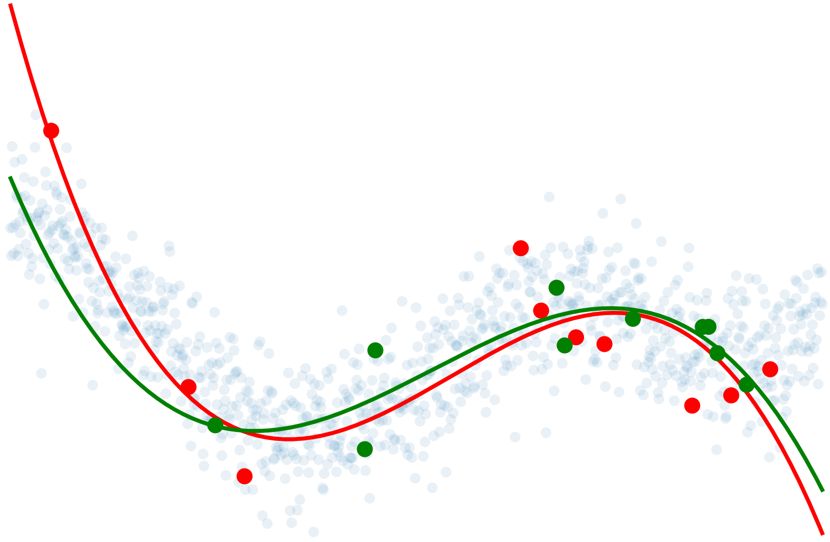
Finite data means uncertainty



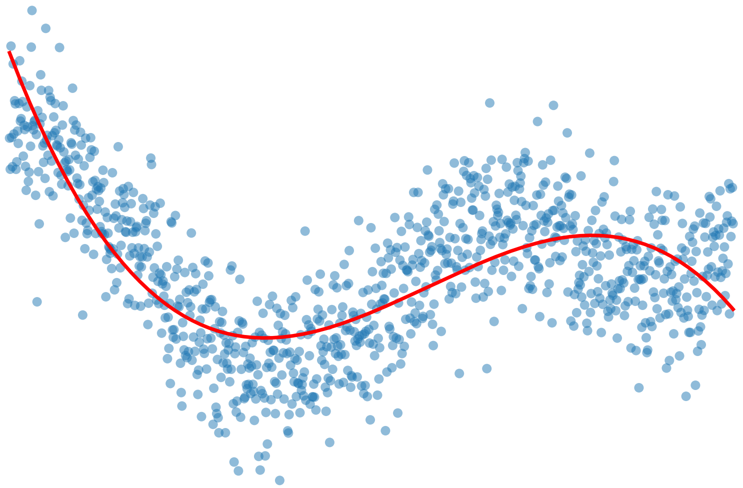


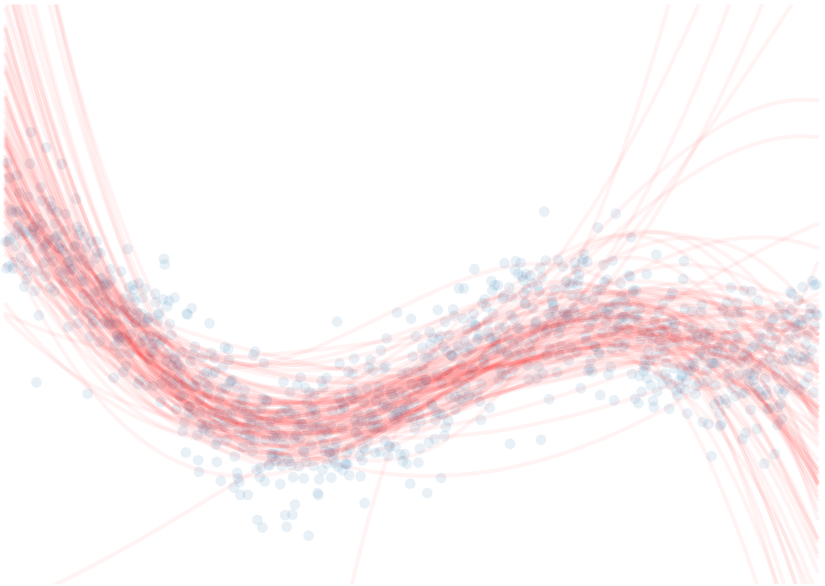
Finite data means uncertainty



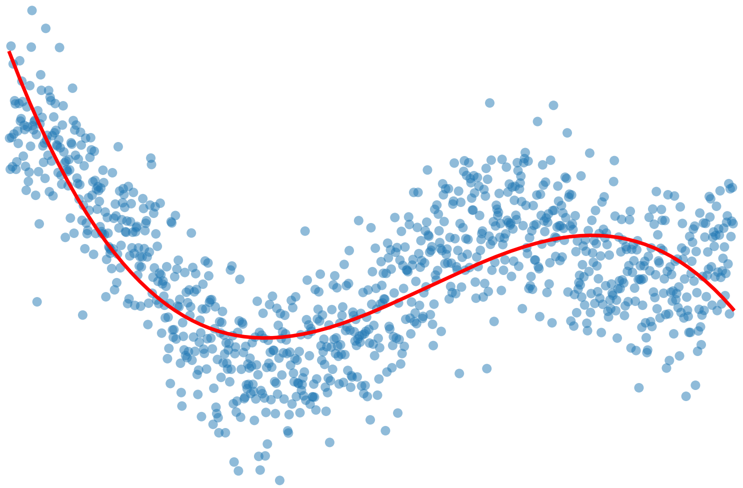


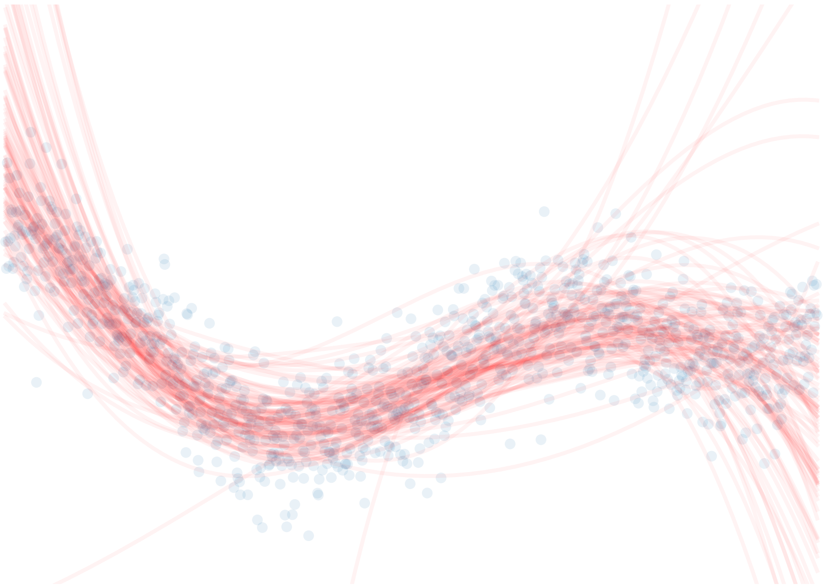
Finite data means uncertainty





Finite data means uncertainty





Problem: For understanding the uncertainty we needed to repeatedly sample new data

What if we only really have 10 samples?

Bayesian inference

Given any finite data, we can only learn a distribution over the parameter values, conditional on whatever we have observed

$$\cancel{\hat{\theta} = \arg\min \mathcal{L}(D, \theta)}$$

$$p(\theta|\mathcal{D})$$

Bayesian inference

Given any finite data, we can only learn a distribution over the parameter values, conditional on whatever we have observed

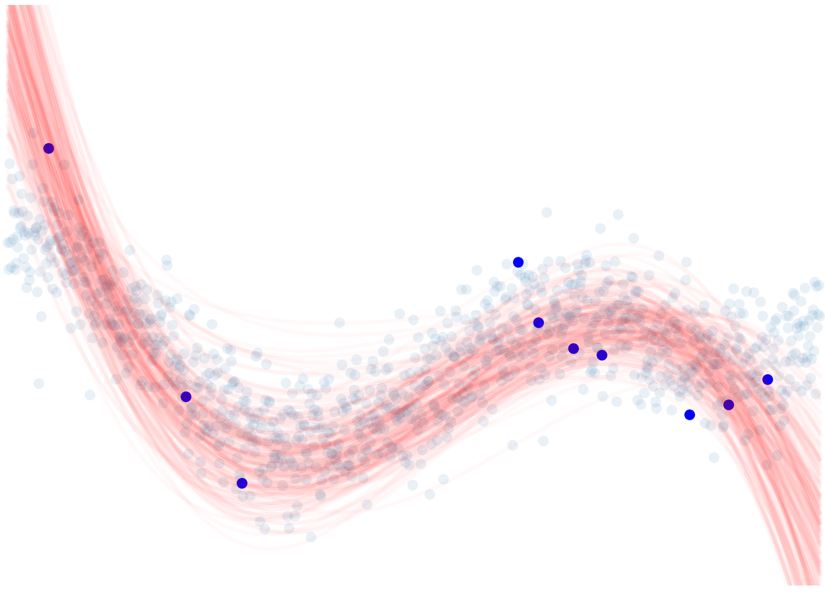
This posterior distribution is defined by the standard rule of conditional probability, also known as Bayes' rule

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}, \theta)}{p(\mathcal{D})}$$

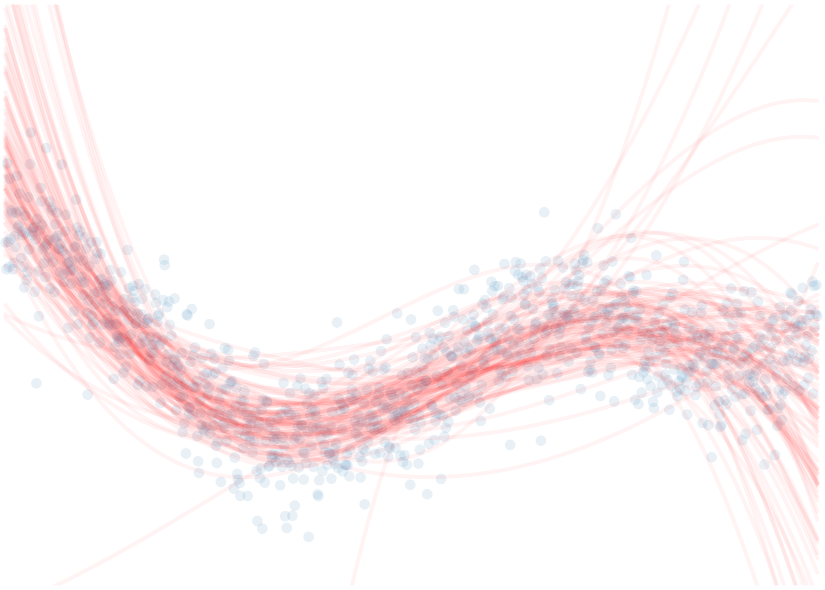
Predictions by integrating over the uncertainty

$$p(y|x) = \int p(y|x, \theta)p(\theta|\mathcal{D})d\theta$$

The posterior distribution

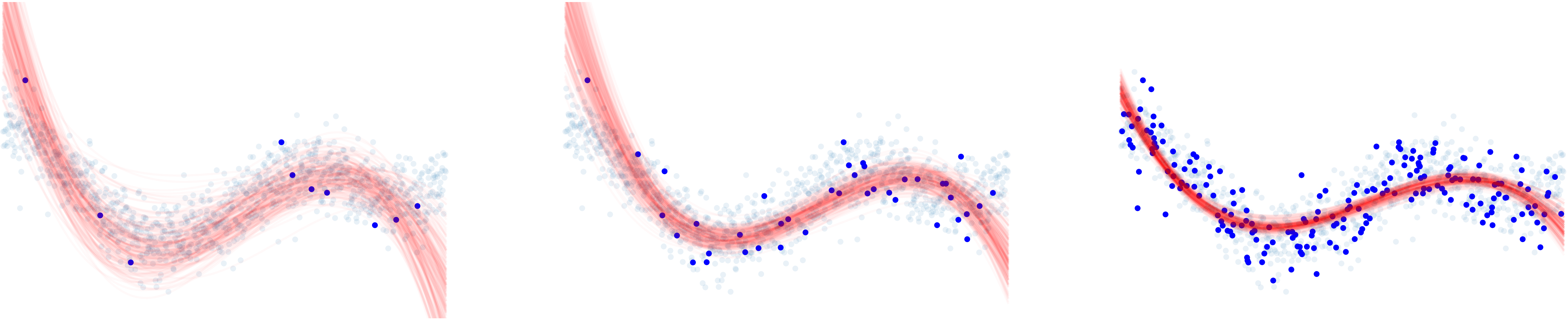


Posterior for 10 observations



Solutions for different sets of 10 observations

The posterior distribution



N=10 N=40 N=200

Observation: Even for this simple 1D problem, 200 samples is not enough for a certain result

The posterior distribution

The posterior is clearly defined for all probabilistic models that express unknown quantities as random variables

The posterior contains everything we need for optimal decisions

$$x \sim \mathrm{unif}[0, 1]$$
$$w \sim \mathcal{N}(0, 1)$$
$$\sigma^2 \sim \text{Inv-Gamma}(\alpha, \beta)$$
$$y \sim \mathcal{N}(w^T x, \sigma^2)$$

The posterior distribution

The posterior is clearly defined for all probabilistic models that express unknown quantities as random variables

The posterior contains everything we need for optimal decisions

We can rarely express the posterior distribution in analytic form and hence need to approximate it somehow

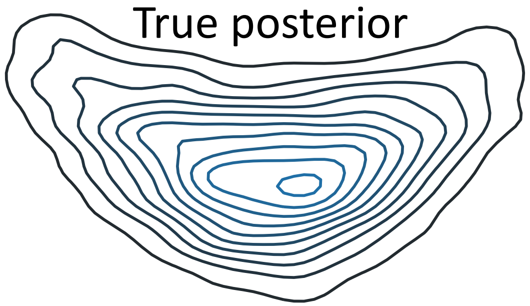
$$x \sim \mathrm{unif}[0,1]$$

$$w \sim \mathcal{N}(0,1)$$

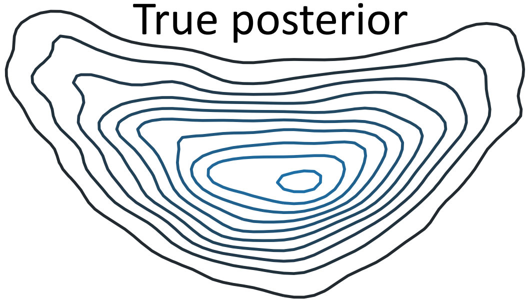
$$\sigma^2 \sim \text{Inv-Gamma}(\alpha, \beta)$$

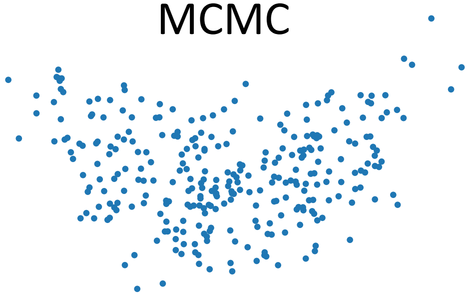
$$y \sim \mathcal{N}(w^T x, \sigma^2)$$

Posterior approximations

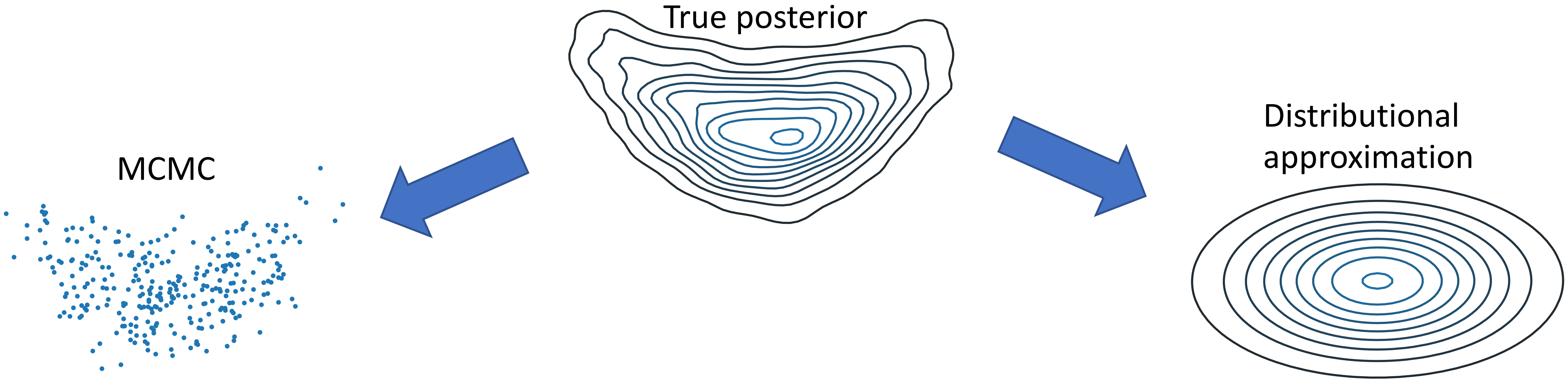


Posterior approximations

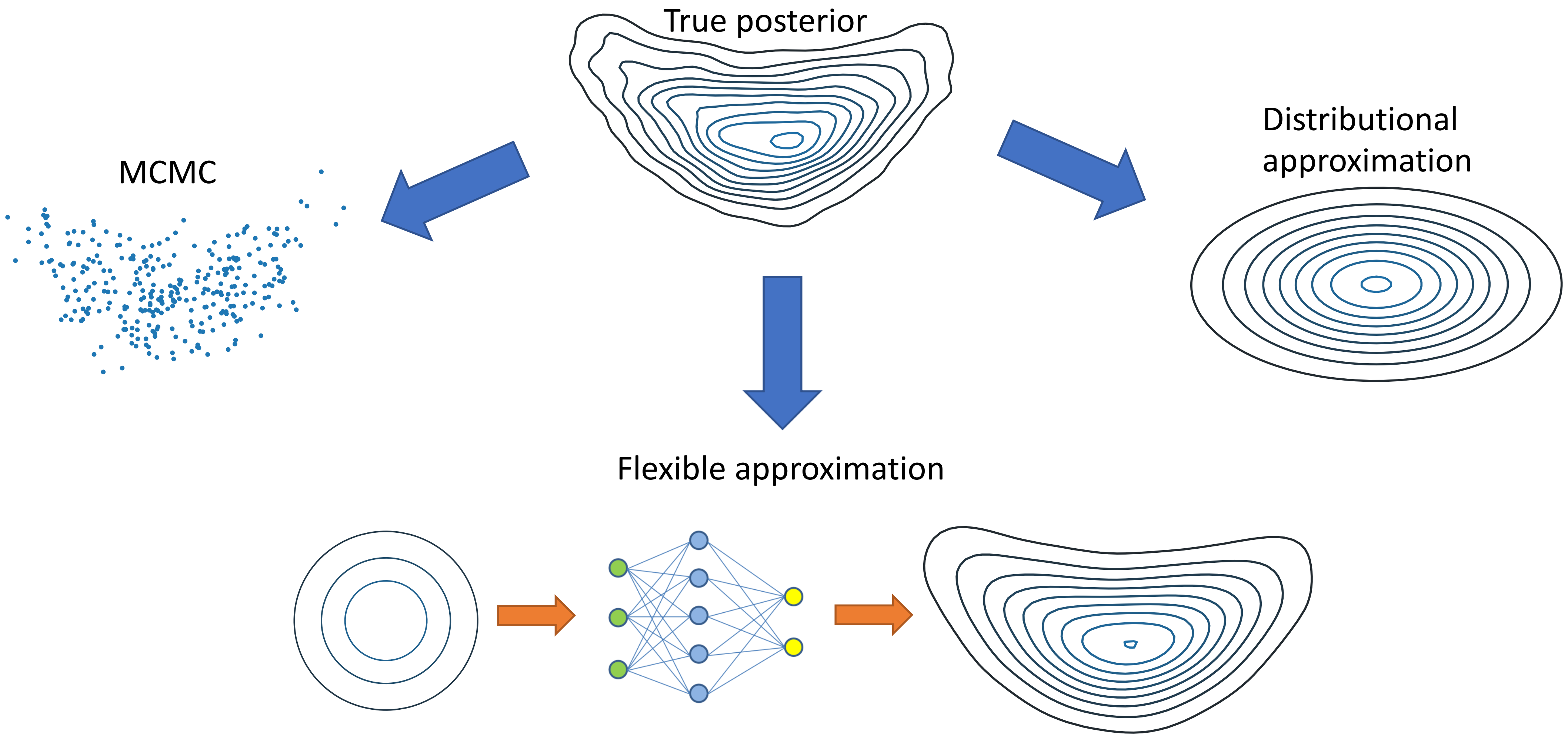




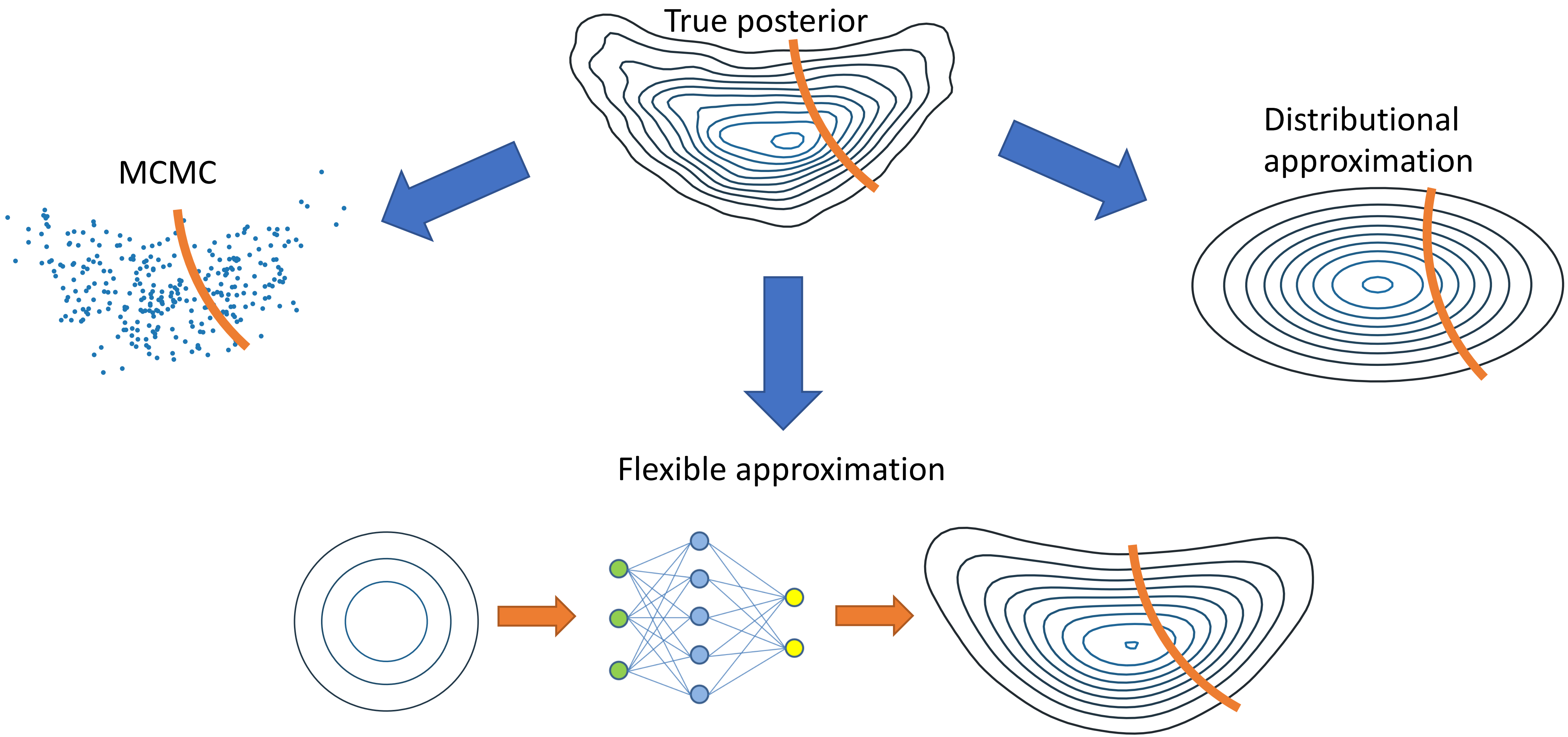
Posterior approximations



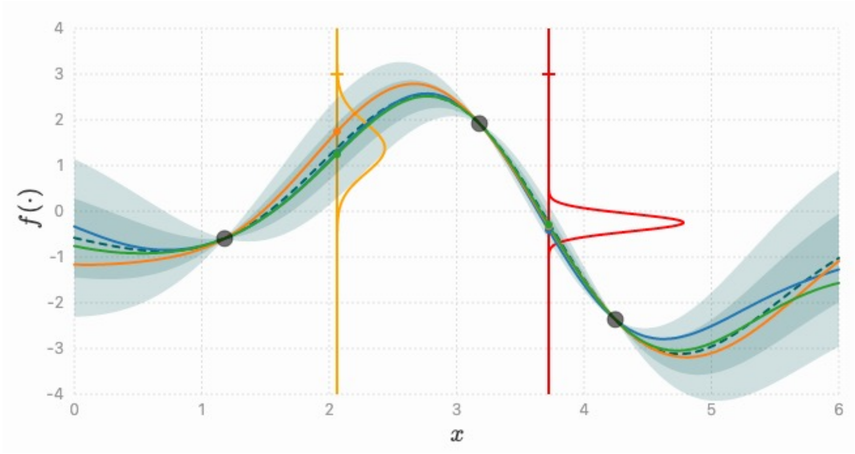
Posterior approximations



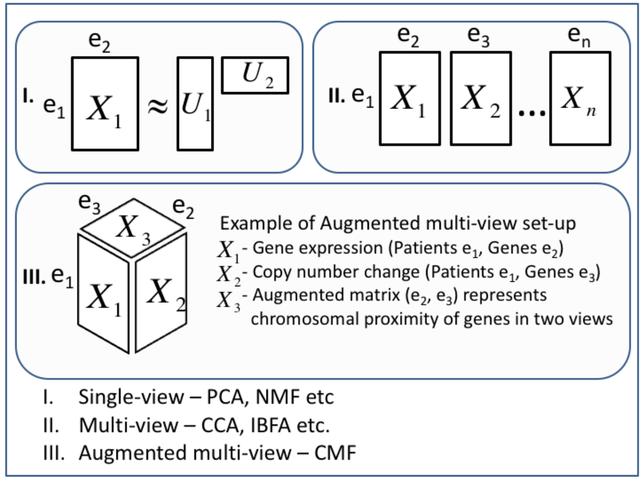
Posterior approximations



Probabilistic AI in action



Gaussian processes

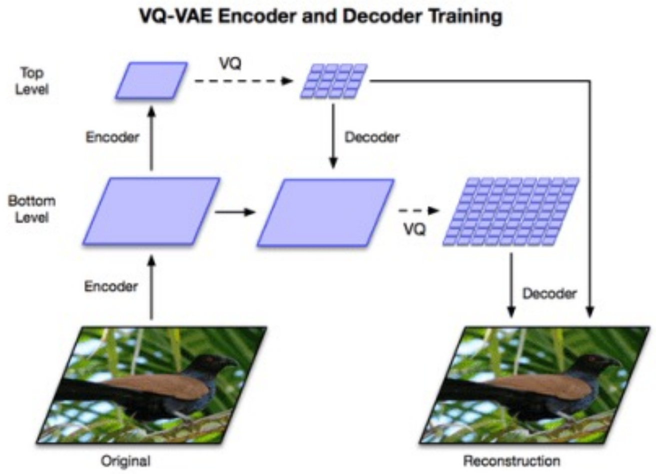


Bayesian matrix factorization

```
data {
  int<lower=0> N;                      // N >= 0
  int<lower=0,upper=1> y[N];           // y[n] in { 0, 1 }
}
parameters {
  real<lower=0,upper=1> theta;         // theta in [0, 1]
}
model {
  theta ~ beta(1,1);                   // prior
  y ~ bernoulli(theta);                // likelihood
}
```

Probabilistic programming

Probabilistic AI in action



(a) Overview of the architecture of our hierarchical VQ-VAE. The encoders and decoders consist of deep neural networks. The input to the model is a 256×256 image that is compressed to quantized latent maps of size 64×64 and 32×32 for the *bottom* and *top* levels, respectively. The decoder reconstructs the image from the two latent maps.

Variational autoencoder for image generation
[Razavi et al. arXiv:1906.00446, 2019]

Probabilistic AI in action

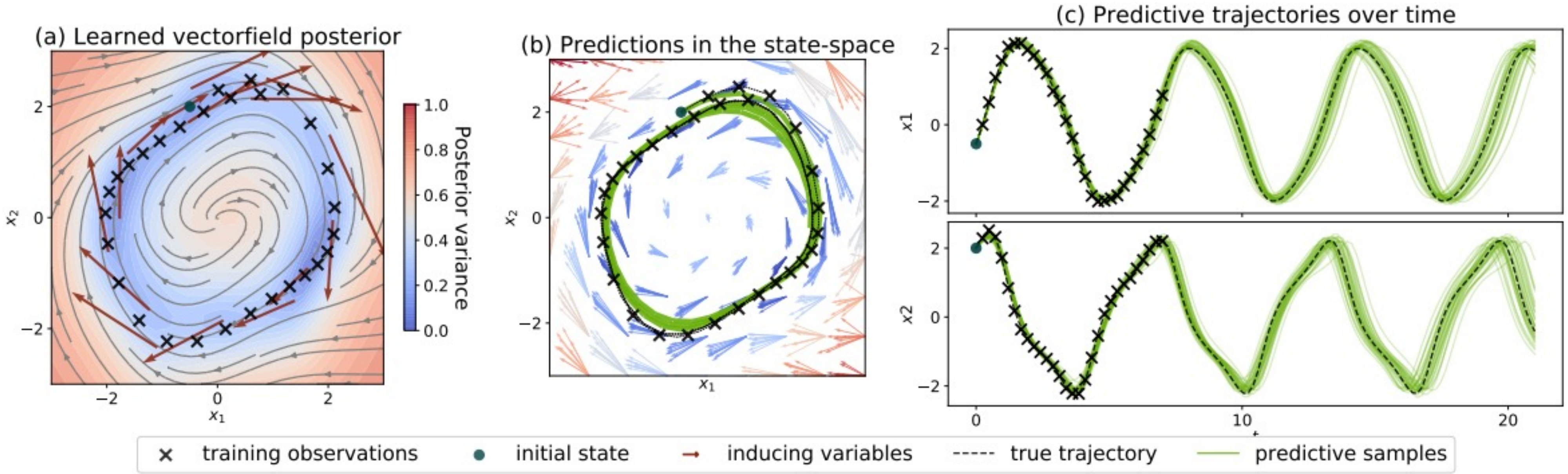


Figure 1: Illustration of GPODE: The model learns a GP posterior (a) of a vector field. Valid ODE trajectories are sampled from the posterior process as shown in (b) and (c).

Bayesian inference for differential equations
[Hegde et al. arXiv:2106.10905, 2022]

Probabilistic AI

1. AI must account for uncertainty
2. It has to be done correctly
   - Specify models using random variables
   - Standard probabilistic calculus for manipulating distributions
   - Predictions by integrating over the uncertainty
3. Exact inference usually impossible, so we need approximations
   - Our focus is largely in variational approximations
   - Modern machine learning tools help learning accurate approximations for flexible models