Bayesian Learning

Intelligent Architectures - 5LIL0

Martin Roa Villescas, Patrick Wijnings, Prof.dr. Henk Corporaal

Department of Electrical Engineering, Electronic Systems Group

DNNs are wonderful

TU/e

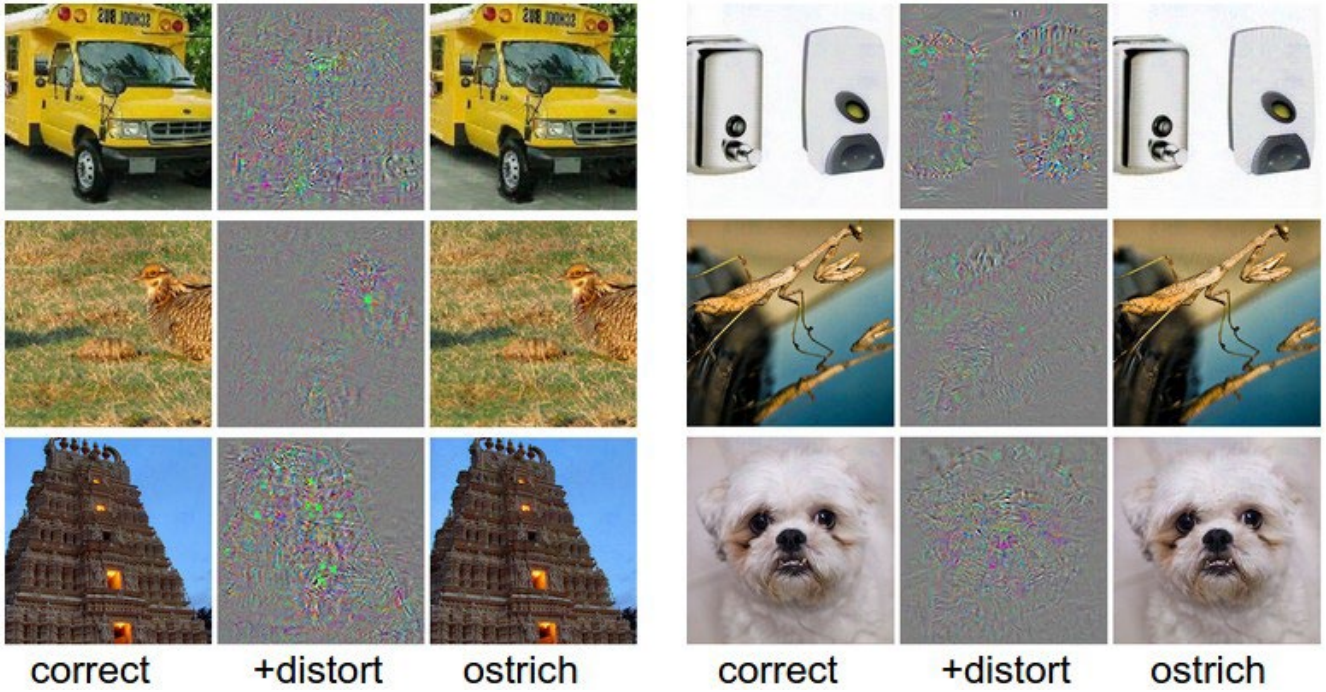
DNNs are wonderful

TU/e

DNNs are wonderful. However …

DNNs are wonderful. However …

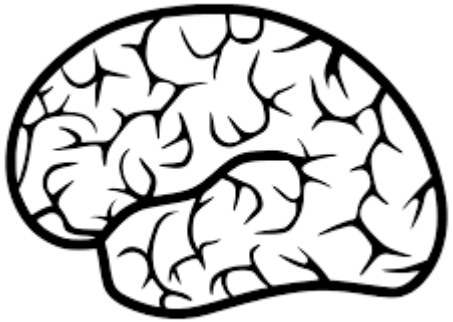
DNNs are wonderful. However …



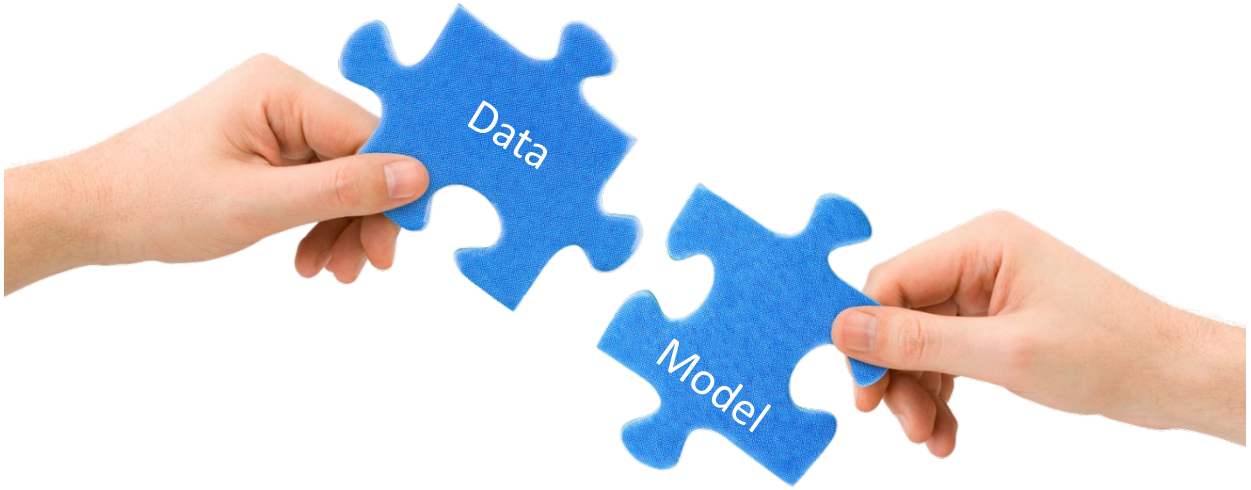
Source: “Explaining and Harnessing Adversarial Examples” by Eric J. Szegedy et al. (2014)

Bayesian Inference

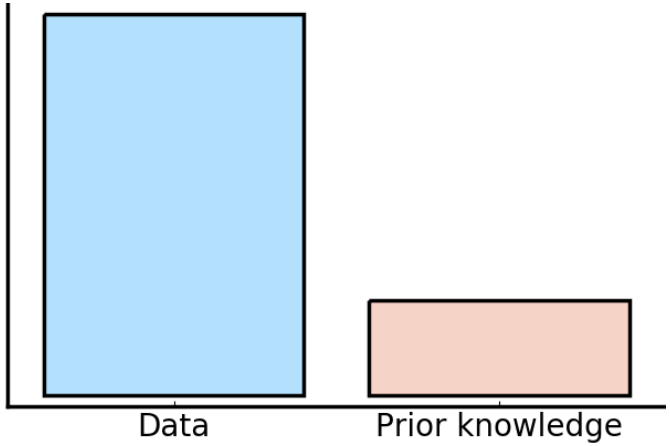
Reasoning in the presence of uncertainty

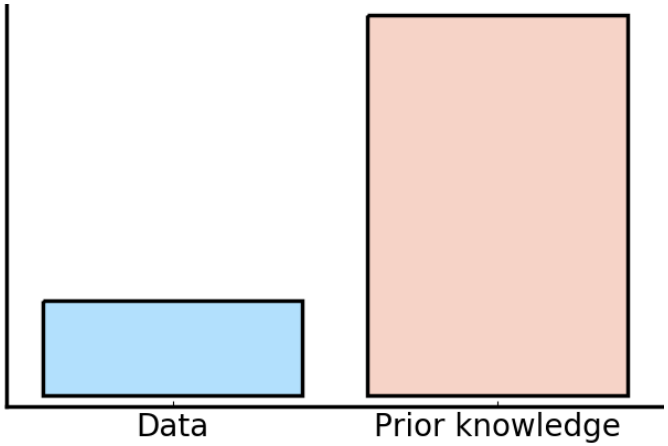


Machine Learning



Data vs Prior Knowledge Trade-off





Going Bayesian

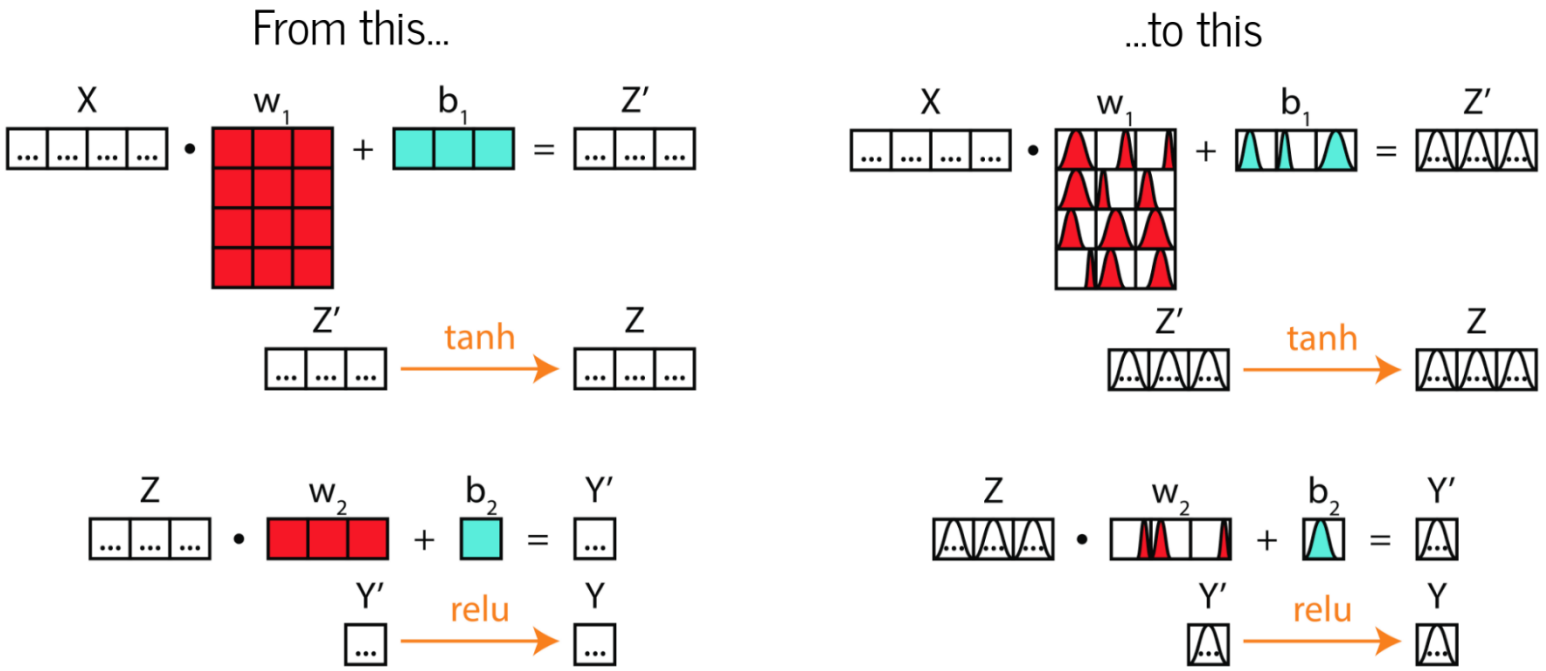
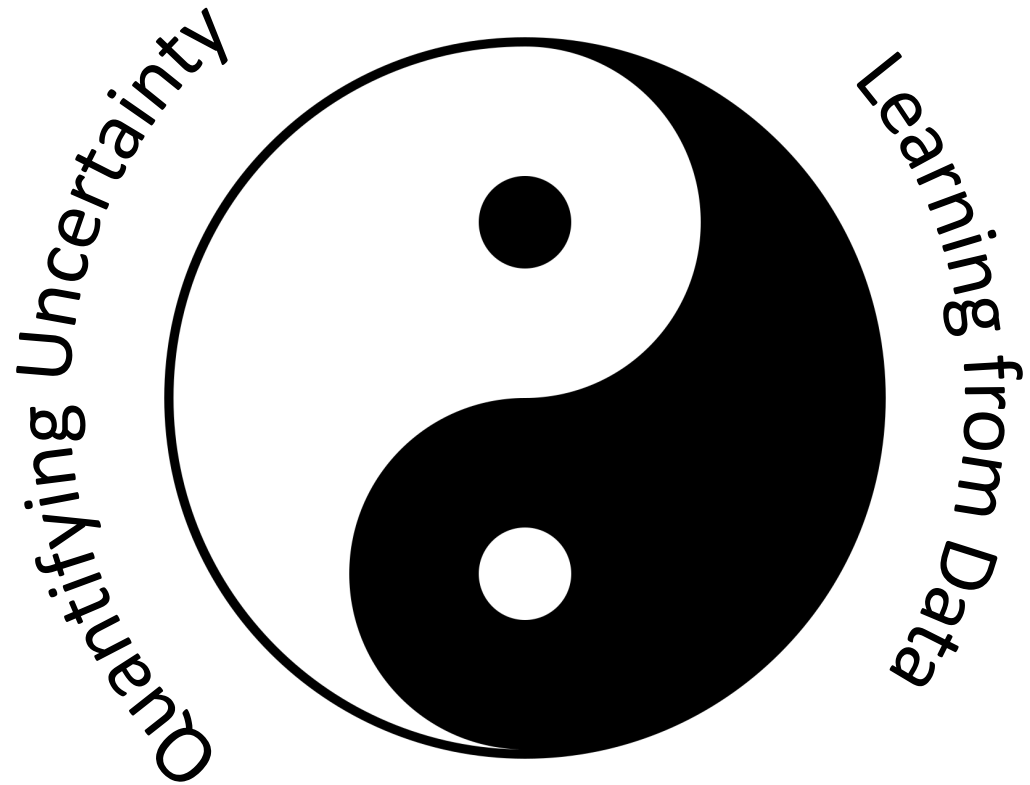
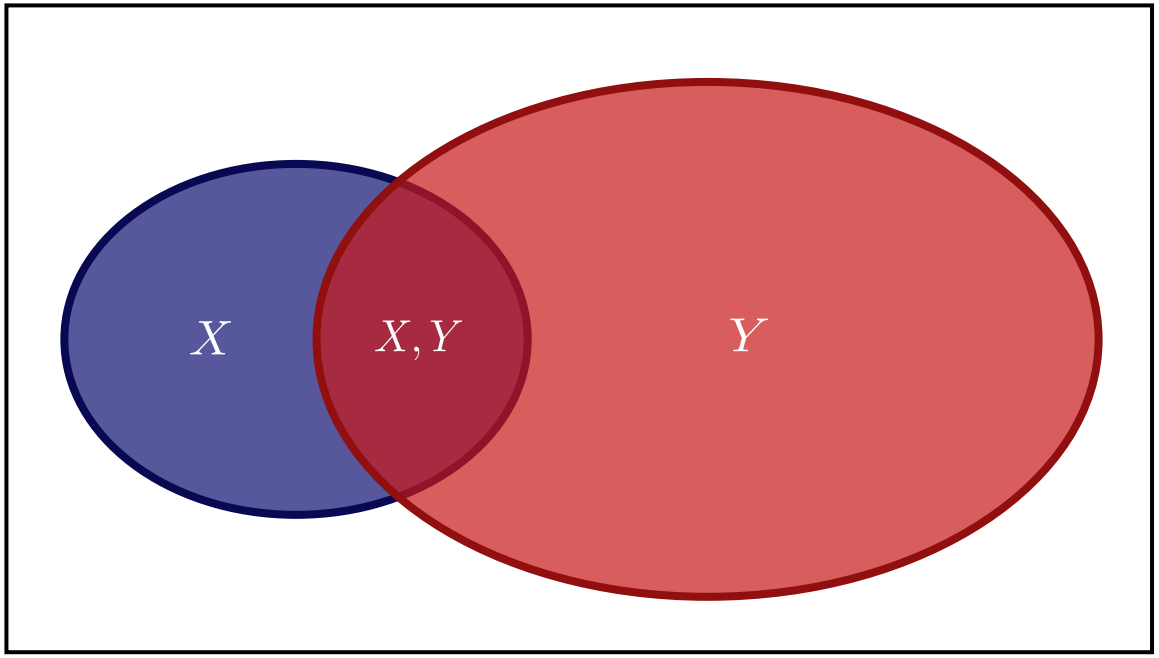


 Image source: Eric J. Ma - An Attempt At Demystifying Bayesian Deep Learning

Quantifying Uncertainty


Quantifying Uncertainty
Learning from Data

Conditional Probability

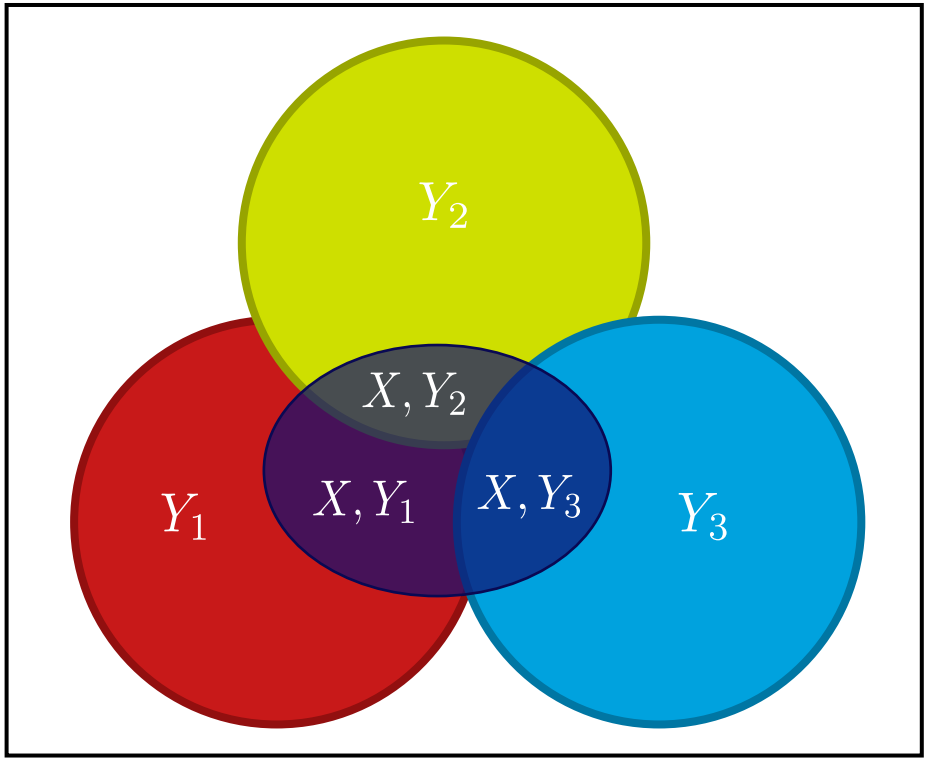


$$p(X \mid Y) = \frac{p(X, Y)}{p(Y)}$$

$$p(X, Y) = p(X \mid Y)p(Y)$$

TU/e

The Law of Total Probability



$$p(X) = p(X, Y_1) + p(X, Y_2) + p(X, Y_3)$$
$$= \sum_{Y} p(X, Y)$$

TU/e

Probability Theory Recap

Joint probability:

$p(X, Y)$

Marginal probability:

$p(X)$

Conditional probability:

$p(X \mid Y)$

Sum rule:

$$p(X) = \sum_{Y} p(X, Y)$$

Product rule:

$$p(X, Y) = p(Y \mid X)p(X) \ = p(X \mid Y)p(Y)$$

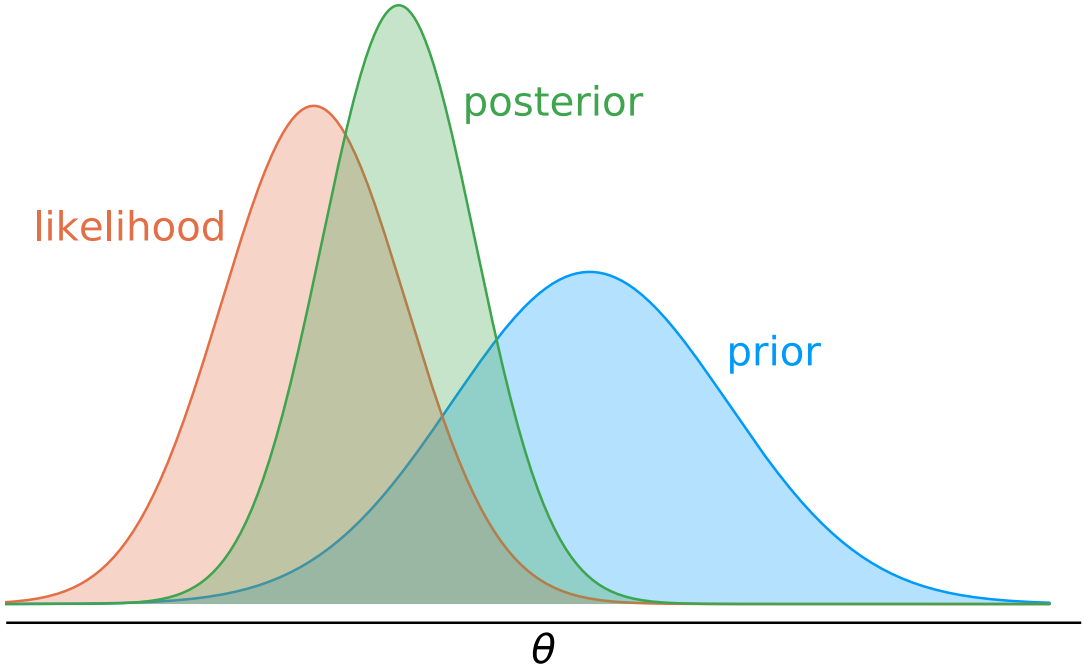
Bayes' rule:

$$p(Y \mid X) = \frac{p(X \mid Y)p(Y)}{p(X)}$$

$$= \frac{p(X \mid Y)p(Y)}{\sum_{Y} p(X \mid Y)p(Y)}$$

Bayes' Theorem

$$\underbrace{p(\boldsymbol{\theta} \mid \boldsymbol{D})}_{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{D} \mid \boldsymbol{\theta})}^{\text{likelihood}}\overbrace{p(\boldsymbol{\theta})}^{\text{prior}}}{\underbrace{p(\boldsymbol{D})}_{\text{evidence}}}$$

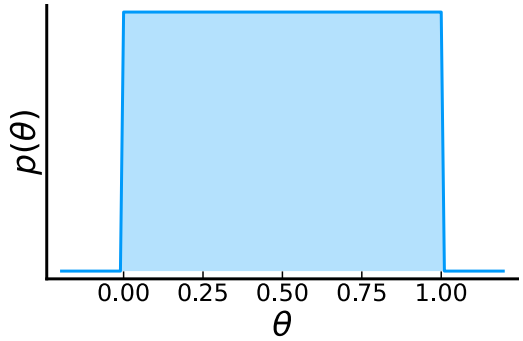


Problem 1: Estimate the Bias of a Coin

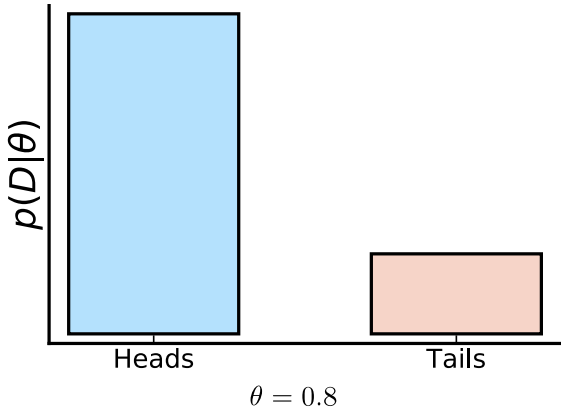
Estimate the Bias of a Coin: Model Specification

$$\underbrace{p(\boldsymbol{\theta} \mid \boldsymbol{D})}_{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{D} \mid \boldsymbol{\theta})}^{\text{likelihood}} \overbrace{p(\boldsymbol{\theta})}^{\text{prior}}}{\underbrace{p(\boldsymbol{D})}_{\text{evidence}}}$$

prior



likelihood



$\theta = 0.8$

Bayesian Machine Learning

Model specification

- Likelihood: $\{x_i\} \sim Bernoulli(\theta)$
- Prior: $\theta \sim Beta(1.0, 1.0)$

Incorporate observed data

- Virtual coin

Infer bias

- Apply Bayes' rule

DEMO

TU/e

Interpretation of Probability

Frequentist: long-run frequency of event in repeatable experiment

Bayesian: degree of belief

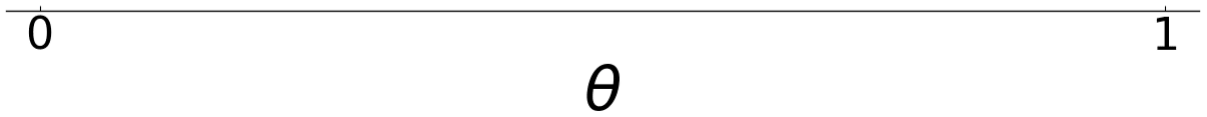
60%

40%

TU/e

What is Bayesian Inference?

Bayesian inference simply means using the rules of **probability theory** to update the probability for a **hypothesis** as more **evidence** or **information** becomes available.



TU/e

Recap: Bayesian Machine Learning

Steps of model-based ML

1. Specify the model (likelihood and prior)
2. Incorporate observed data
3. Do inference (i.e., learn, adapt)

- Iterate 2 and 3 in real-time applications
- Extend the model as required

How does a machine learn?

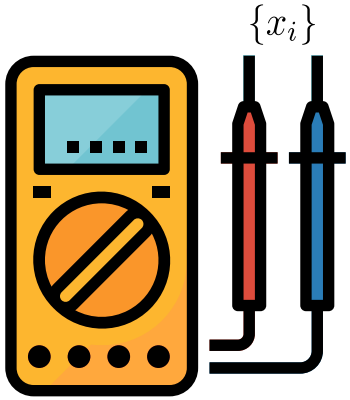
- Updates the parameters of the probabilistic model using **Bayes' rule**

$$\underbrace{p(\boldsymbol{\theta} \mid \boldsymbol{D}, m)}_{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{D} \mid \boldsymbol{\theta}, m)}^{\text{likelihood}} \overbrace{p(\boldsymbol{\theta} \mid m)}^{\text{prior}}}{\underbrace{p(\boldsymbol{D} \mid \boldsymbol{m})}_{\text{evidence}}}$$

$$\overbrace{p(\boldsymbol{\theta} \mid \boldsymbol{D}, m)}^{\text{posterior}} \propto \overbrace{p(\boldsymbol{D} \mid \boldsymbol{\theta}, m)}^{\text{likelihood}} \overbrace{p(\boldsymbol{\theta} \mid m)}^{\text{prior}}$$

Bayesian Time Series

Problem 2: Noisy Measurements



$\{x_i\}$ = 76 V, 73 V, 75 V, 72 V, 77 V

What number should we use?

Right. The mean: 74.6 V.

How do you calculate it?

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

This is inefficient. Why?

There is another more efficient way of doing this.

Icon made by Freepik from www.flaticon.com

Running Average

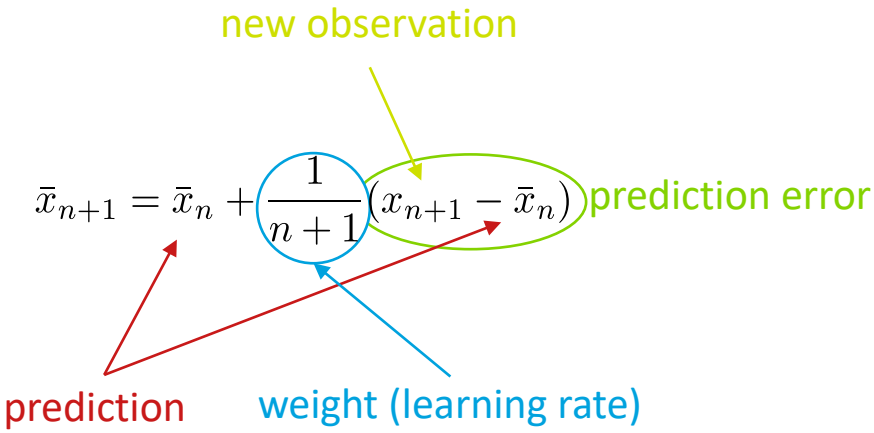
$$\bar{x}_{n+1} = \frac{1}{n+1}\sum_{i=1}^{n+1} x_i = \frac{x_{n+1}}{n+1} + \frac{1}{n+1}\sum_{i=1}^{n} x_i = \frac{x_{n+1}}{n+1} + \frac{n}{n+1}\frac{1}{n}\sum_{i=1}^{n} x_i =$$

$$= \frac{x_{n+1}}{n+1} + \frac{n}{n+1}\bar{x}_n = \bar{x}_n + \frac{x_{n+1}}{n+1} + \frac{n}{n+1}\bar{x}_n - \frac{n+1}{n+1}\bar{x}_n =$$

$$= \bar{x}_n + \frac{1}{n+1}(x_{n+1} + (n - n - 1)\bar{x}_n) = \bar{x}_n + \frac{1}{n+1}(x_{n+1} - \bar{x}_n)$$

$$\bar{x}_{n+1} = \bar{x}_n + \frac{1}{n+1}(x_{n+1} - \bar{x}_n)$$

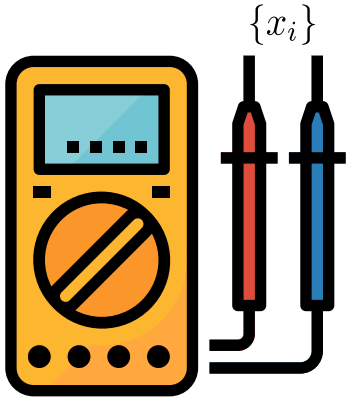
Analysis of the Update Equation



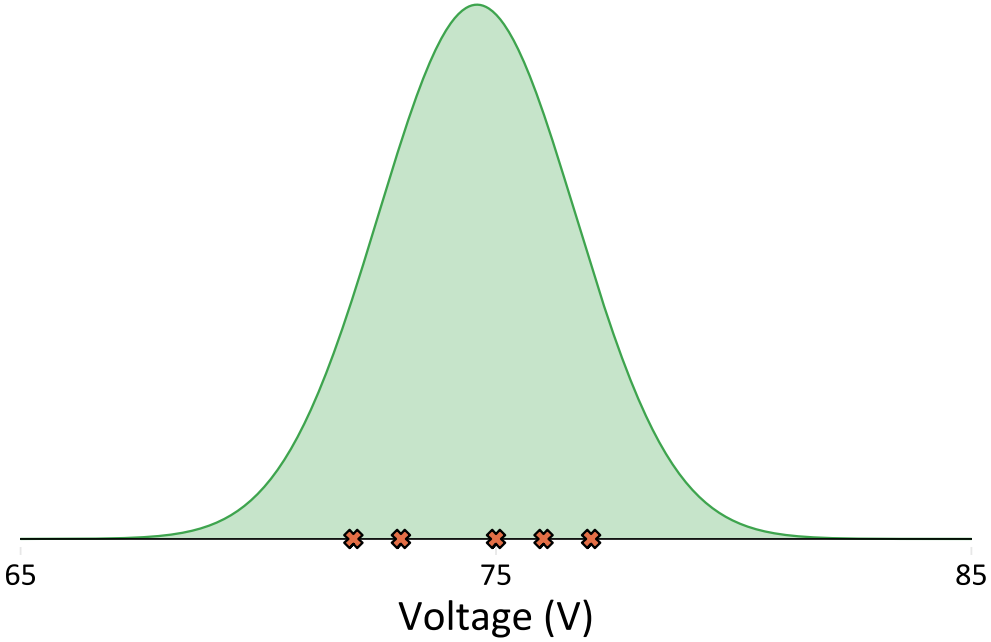
In many cases, Bayesian inference reduces to updates of this type.

TU/e

Same Problem Under a Gaussian Model

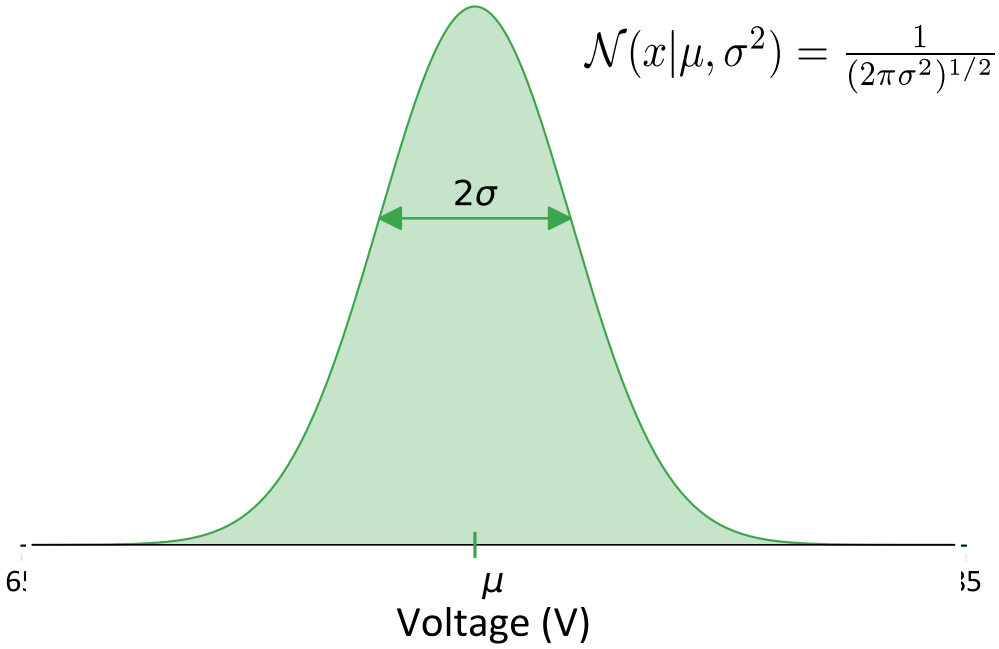


$\{x_i\} =$ 76 V, 73 V, 75 V, 72 V, 77 V

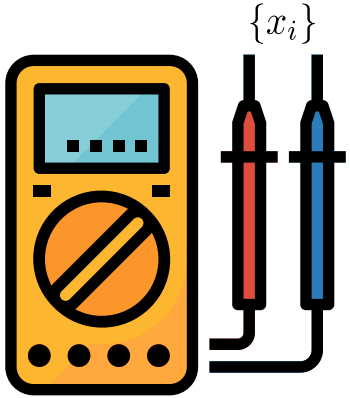


The Gaussian Distribution

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\{-\frac{1}{2\sigma^2}(x-\mu)^2\}$$



Gaussian Model Specification



$\{x_i\} =$ 76 V, 73 V, 75 V, 72 V, 77 V

Prior:

$$p(\theta) \sim \mathcal{N}(\mu_\theta, \pi_\theta^{-1})$$

Likelihood:

$$p(x|\theta) \sim \mathcal{N}(\theta, \pi_\epsilon^{-1})$$

Posterior:

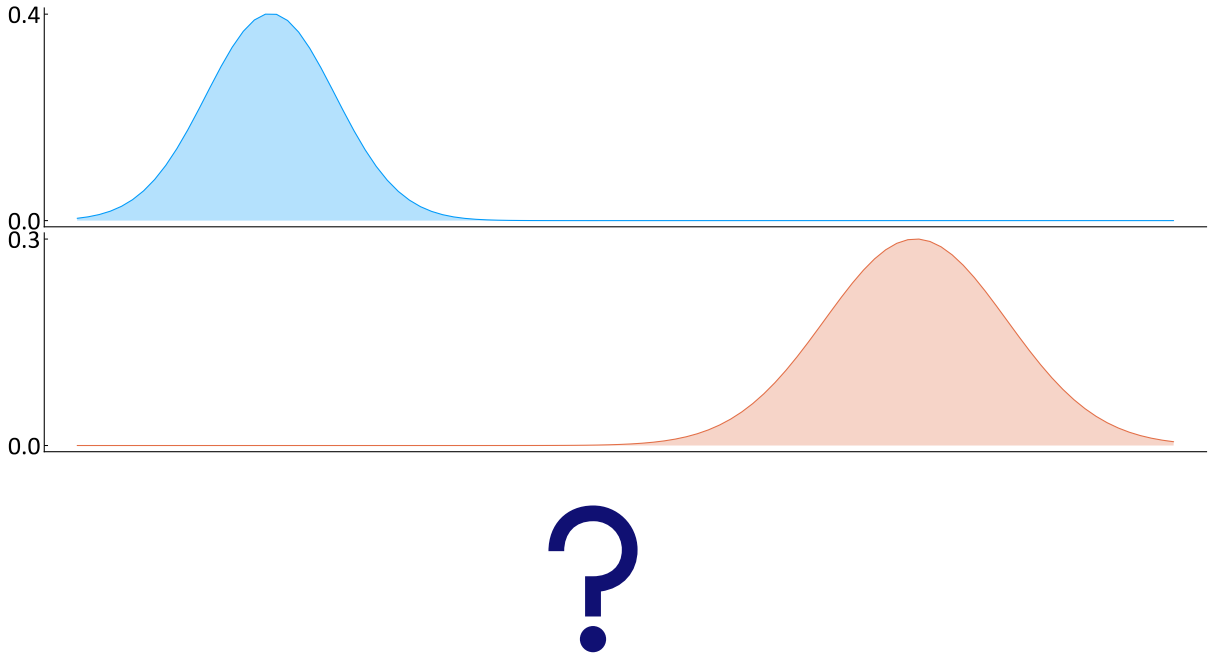
$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)}$$

$$= \frac{p(x|\theta)p(\theta)}{\int p(x|\theta')p(\theta')\,\mathrm{d}\theta'}$$

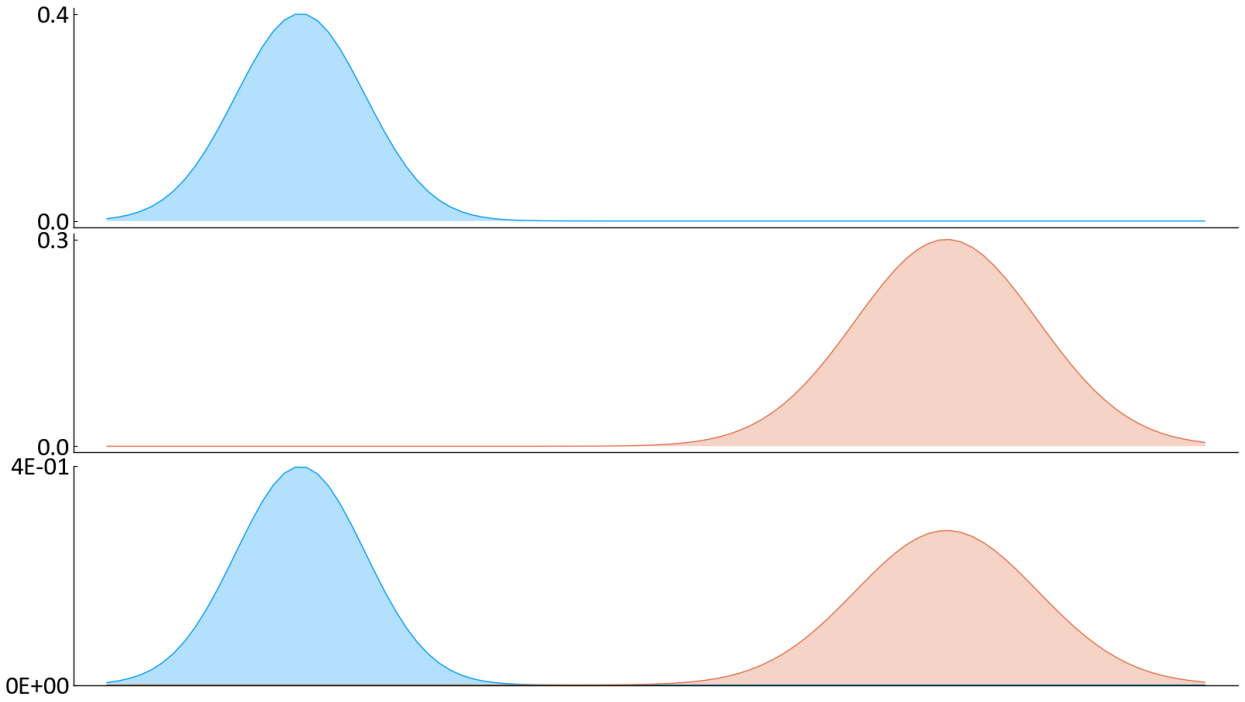
$$= \frac{\mathcal{N}(\theta, \pi_\epsilon^{-1})\mathcal{N}(\mu_\theta, \pi_\theta^{-1})}{\int \mathcal{N}(\theta', \pi_\epsilon^{-1})\mathcal{N}(\mu_\theta, \pi_\theta^{-1})\,\mathrm{d}\theta'}$$

TU/e

Product of 2 Univariate Gaussian PDFs

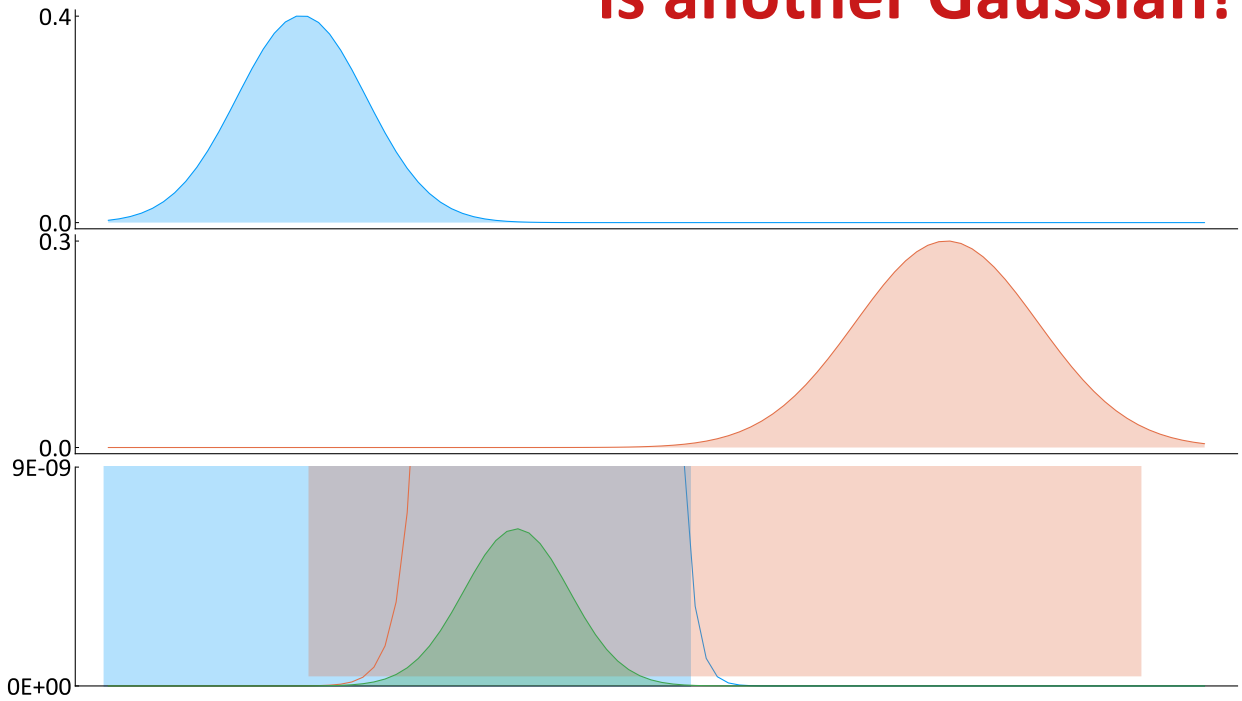


Product of 2 Univariate Gaussian PDFs



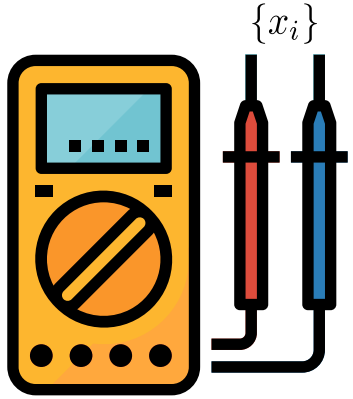
Product of 2 Univariate Gaussian PDFs

is another Gaussian!



TU/e

Same Problem Under a Gaussian Model



$\{x_i\}$ = 76 V, 73 V, 75 V, 72 V, 77 V

Prior: $p(\theta) \sim \mathcal{N}(\mu_\theta, \pi_\theta^{-1})$

Likelihood: $p(x|\theta) \sim \mathcal{N}(\theta, \pi_\epsilon^{-1})$

Posterior: **is another Gaussian!**

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)} \sim \mathcal{N}(\mu_{\theta|x}, \pi_{\theta|x}^{-1})$$

where

$$\pi_{\theta|x} = \pi_\theta + \pi_\epsilon$$

$$\mu_{\theta|x} = \mu_\theta + \frac{\pi_\epsilon}{\pi_{\theta|x}}(x - \mu_\theta)$$

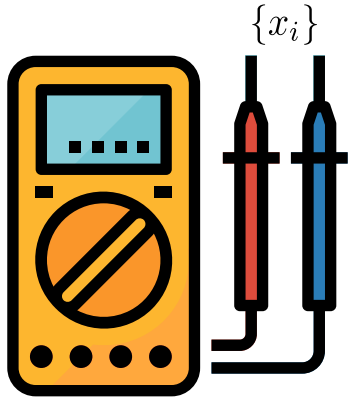
prediction error

prediction

weight (learning rate)

TU/e

Same Problem Under a Gaussian Model



$\{x_i\} =$ 76 V, 73 V, 75 V, 72 V, 77 V

Prior:

$$p(\theta) \sim \mathcal{N}(\mu_\theta, \pi_\theta^{-1})$$

Likelihood:

$$p(x|\theta) \sim \mathcal{N}(\theta, \pi_\epsilon^{-1})$$

Posterior:

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)} \sim \mathcal{N}(\mu_{\theta|x}, \pi_{\theta|x}^{-1})$$

where

$$\pi_{\theta|x} = \pi_\theta + \pi_\epsilon$$

$$\mu_{\theta|x} = \mu_\theta + \frac{\pi_\epsilon}{\pi_\theta + \pi_\epsilon}(x - \mu_\theta)$$

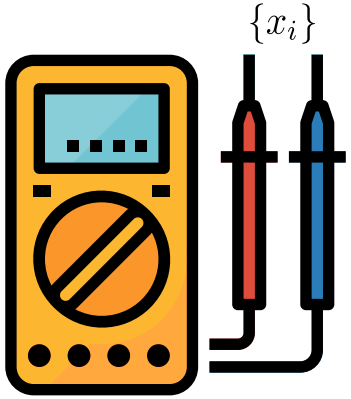
prediction error

prediction

weight (learning rate)

TU/e

Same Problem Under a Gaussian Model



$\{x_i\}$ = 76 V, 73 V, 75 V, 72 V, 77 V

Prior:
$$p(\theta) \sim \mathcal{N}(\mu_\theta, \pi_\theta^{-1})$$

Likelihood:
$$p(x|\theta) \sim \mathcal{N}(\theta, \pi_\epsilon^{-1})$$

Posterior:
$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)} \sim \mathcal{N}(\mu_{\theta|x}, \pi_{\theta|x}^{-1})$$

where

$$\pi_{\theta|x} = \pi_\theta + \pi_\epsilon$$

$$\mu_{\theta|x} = \mu_\theta + \frac{1}{\frac{\pi_\theta}{\pi_\epsilon} + 1}(x - \mu_\theta)$$

prediction error

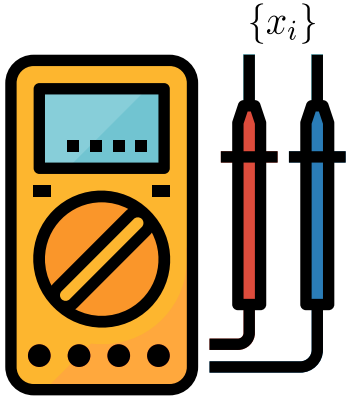
prediction

weight (learning rate)

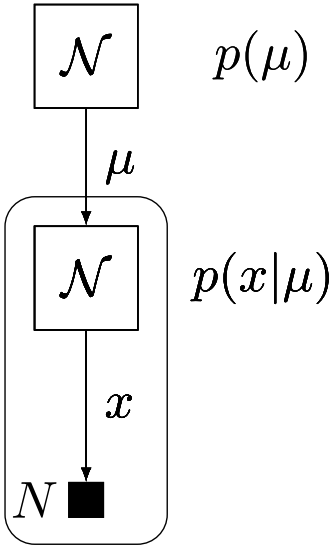
This pattern generalizes for all **exponential families** with **conjugate priors**.

TU/e

Same Problem Under a Gaussian Model



$\{x_i\}$ = 76 V, 73 V, 75 V, 72 V, 77 V



TU/e

Exponential Family

The **exponential family** is defined to be a set of probability distributions of the following form

$$f_X(x|\theta) = h(x)\exp(\eta(\theta)\cdot T(x) - A(\theta))$$

This allows Bayesian inference to be treated in a generic way for all members of the exponential family.

Most of the commonly used distributions are members of the exponential family:

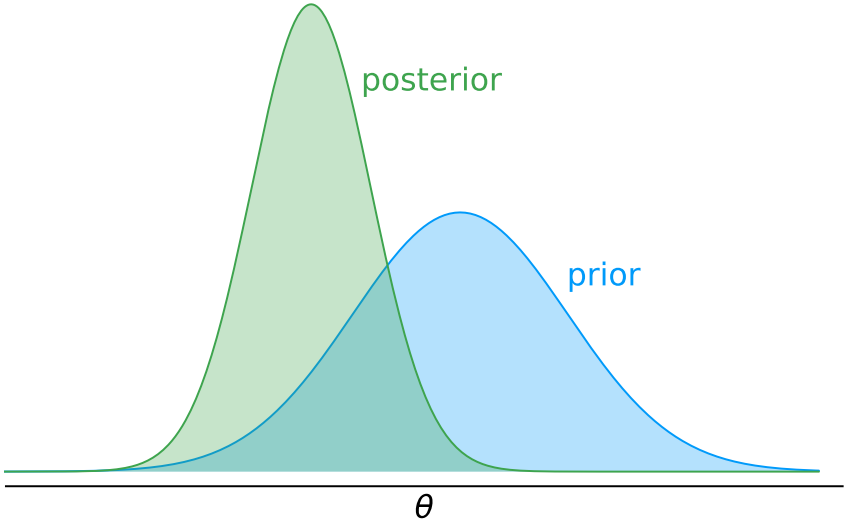
- Gaussian
- exponential
- gamma
- chi-squared
- beta
- Dirichlet
- Bernoulli
- categorical
- Poisson
- Wishart
- inverse Wishart
- geometric

Conjugate Priors

A **conjugate prior** is a probability distribution that, when multiplied by the **likelihood** and divided by the normalizing constant, yields a **posterior** probability distribution that is in the same family of distributions as the **prior**.

$$\underbrace{p(\boldsymbol{\theta} \mid \boldsymbol{D})}_{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{D} \mid \boldsymbol{\theta})}^{\text{likelihood}} \overbrace{p(\boldsymbol{\theta})}^{\text{prior}}}{\underbrace{p(\boldsymbol{D})}_{\text{evidence}}}$$

All members of the **exponential family** have **conjugate priors**



TU/e

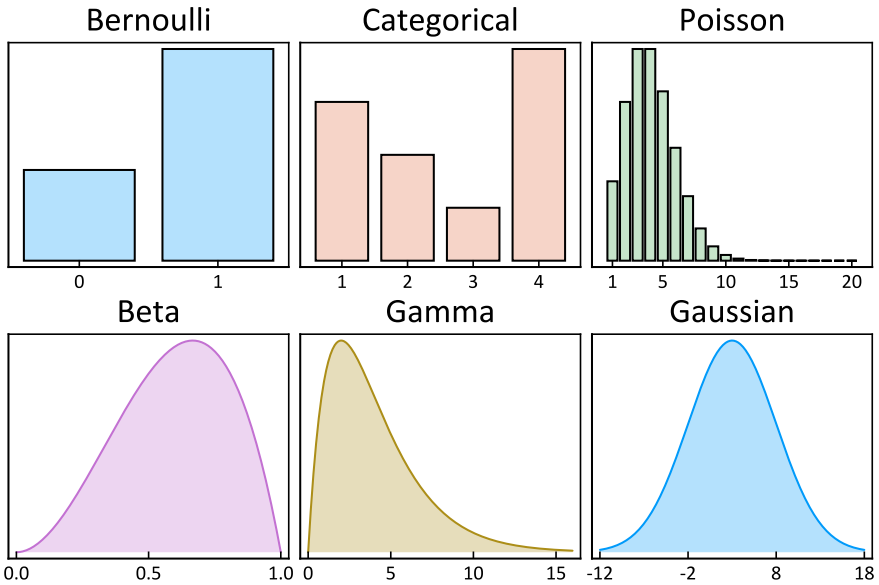
Likelihood	Model parameters	Conjugate prior distribution	Prior hyperparameters	Posterior hyperparameters	Interpretation of hyperparameters[note 1]	Posterior predictive[note 2]
Bernoulli	p (probability)	Beta	$\alpha,\ \beta$	$\alpha+\sum_{i=1}^{n}x_i,\ \beta+n-\sum_{i=1}^{n}x_i$	$\alpha-1$ successes, $\beta-1$ failures[note 1]	$p(\tilde{x}=1)=\frac{\alpha'}{\alpha'+\beta'}$
Binomial	p (probability)	Beta	$\alpha,\ \beta$	$\alpha+\sum_{i=1}^{n}x_i,\ \beta+\sum_{i=1}^{n}N_i-\sum_{i=1}^{n}x_i$	$\alpha-1$ successes, $\beta-1$ failures[note 1]	$\mathrm{BetaBin}(\tilde{x}\vert\alpha',\beta')$ (beta-binomial)
Negative binomial with known failure number, r	p (probability)	Beta	$\alpha,\ \beta$	$\alpha+\sum_{i=1}^{n}x_i,\ \beta+rn$	$\alpha-1$ total successes, $\beta-1$ failures[note 1] (i.e., $\frac{\beta-1}{r}$ experiments, assuming r stays fixed)	
Poisson	λ (rate)	Gamma	$k,\ \theta$	$k+\sum_{i=1}^{n}x_i,\ \frac{\theta}{n\theta+1}$	k total occurrences in $\frac{1}{\theta}$ intervals	$\mathrm{NB}(\tilde{x}\mid k',\theta')$ (negative binomial)
			$\alpha,\ \beta$[note 3]	$\alpha+\sum_{i=1}^{n}x_i,\ \beta+n$	α total occurrences in β intervals	$\mathrm{NB}\left(\tilde{x}\mid\alpha',\frac{1}{1+\beta'}\right)$ (negative binomial)
Categorical	$\boldsymbol{p}$ (probability vector), k (number of categories; i.e., size of $\boldsymbol{p}$)	Dirichlet	$\boldsymbol{\alpha}$	$\boldsymbol{\alpha}+(c_1,\ldots,c_k)$, where c_i is the number of observations in category i	α_i-1 occurrences of category i[note 1]	$p(\tilde{x}=i)=\frac{{\alpha_i}'}{\sum_i {\alpha_i}'}=\frac{\alpha_i+c_i}{\sum_i \alpha_i+n}$
Multinomial	$\boldsymbol{p}$ (probability vector), k (number of categories; i.e., size of $\boldsymbol{p}$)	Dirichlet	$\boldsymbol{\alpha}$	$\boldsymbol{\alpha}+\sum_{i=1}^{n}\mathbf{x}_i$	α_i-1 occurrences of category i[note 1]	$\mathrm{DirMult}(\tilde{\mathbf{x}}\mid\boldsymbol{\alpha}')$ (Dirichlet-multinomial)
Hypergeometric with known total population size, N	M (number of target members)	Beta-binomial[4]	$n=N,\alpha,\ \beta$	$\alpha+\sum_{i=1}^{n}x_i,\ \beta+\sum_{i=1}^{n}N_i-\sum_{i=1}^{n}x_i$	$\alpha-1$ successes, $\beta-1$ failures[note 1]	
Geometric	p_0 (probability)	Beta	$\alpha,\ \beta$	$\alpha+n,\ \beta+\sum_{i=1}^{n}x_i$	$\alpha-1$ experiments, $\beta-1$ total failures[note 1]	

In sum …

We have a set of probability distributions to represent beliefs about states.

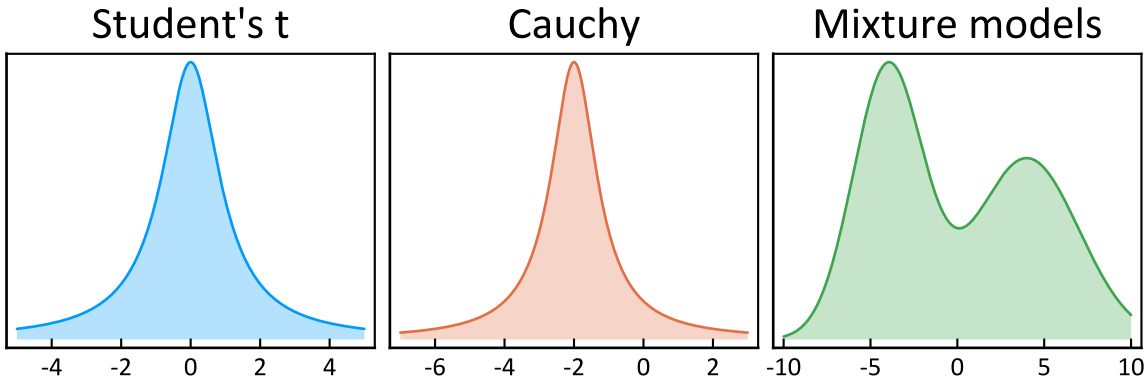
All Bayesian updates of such beliefs take the form of precision-weighted prediction errors.

These prediction errors and their precision weights are easy to compute.



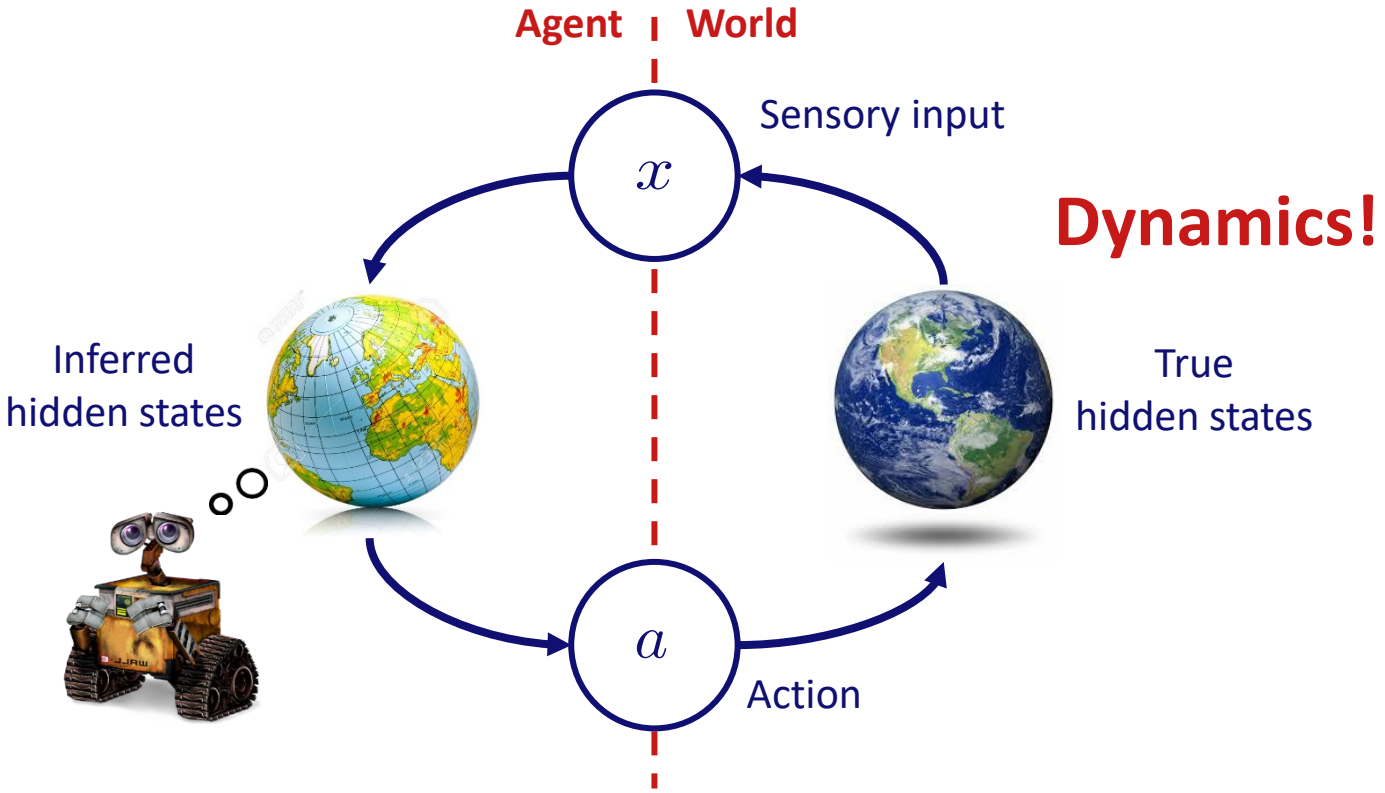
However …

There are important probability distributions that are **not** members of the exponential family



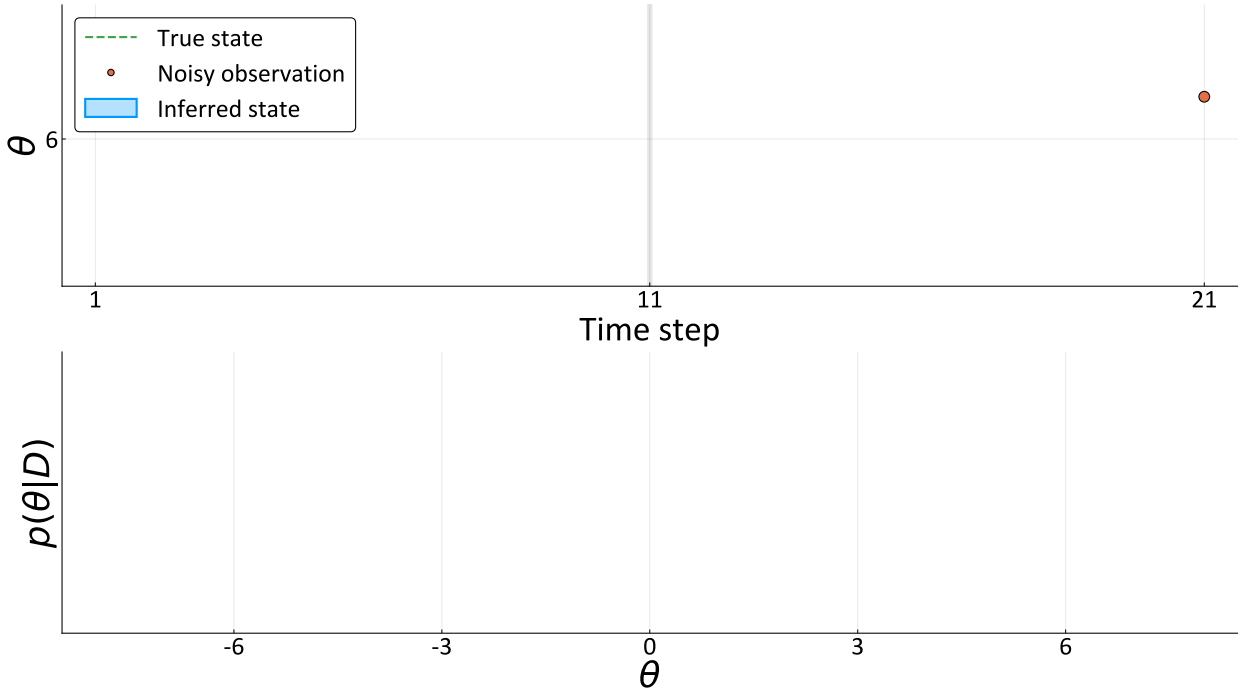
These, on the other hand, **can** be represented using appropriate **hierarchies** of distributions in the exponential family.

What have we neglected so far?

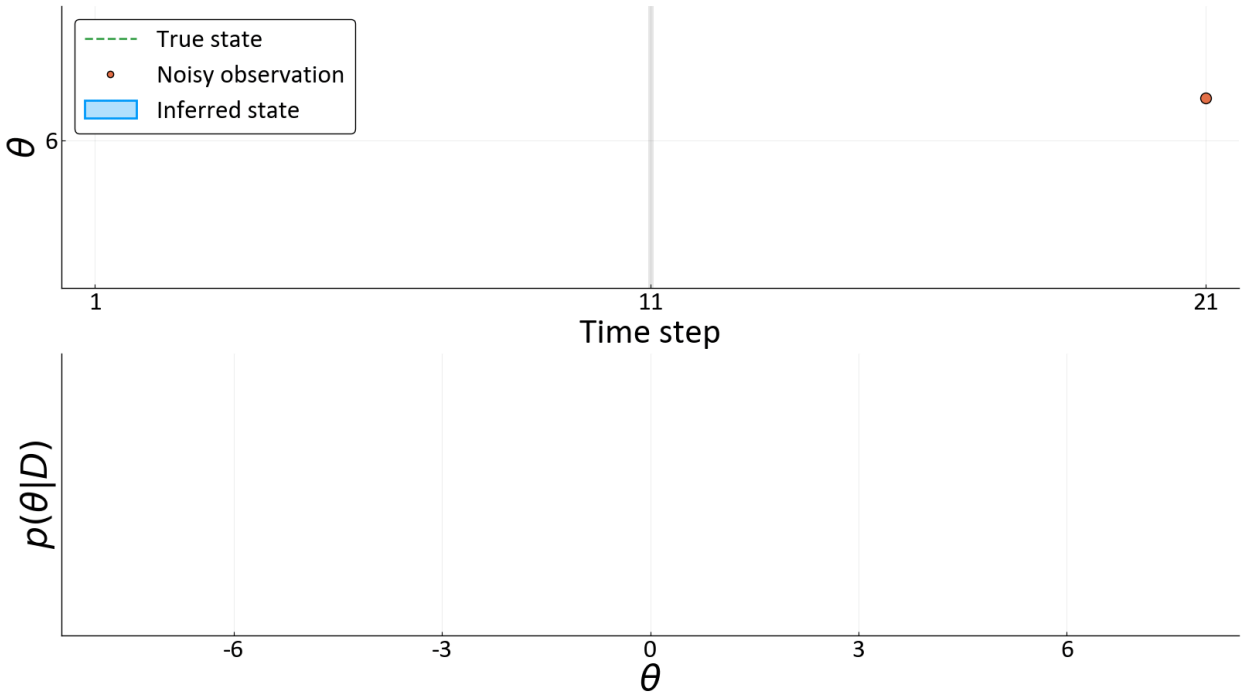


TU/e

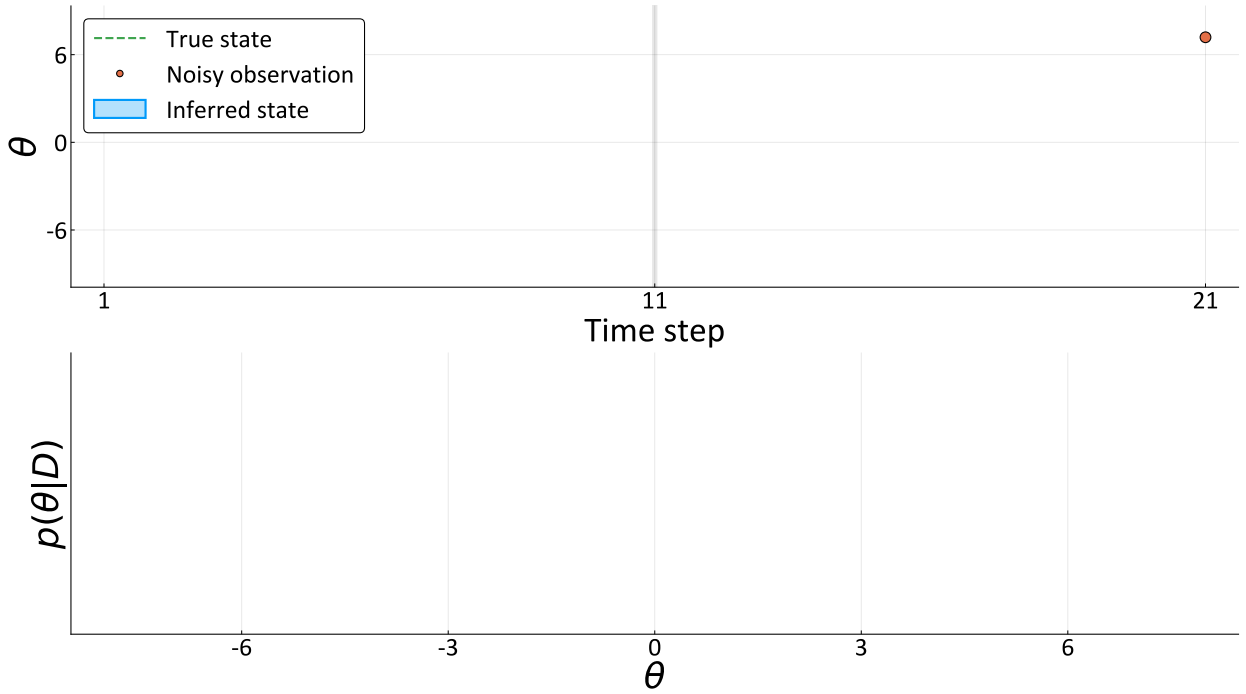
Static State



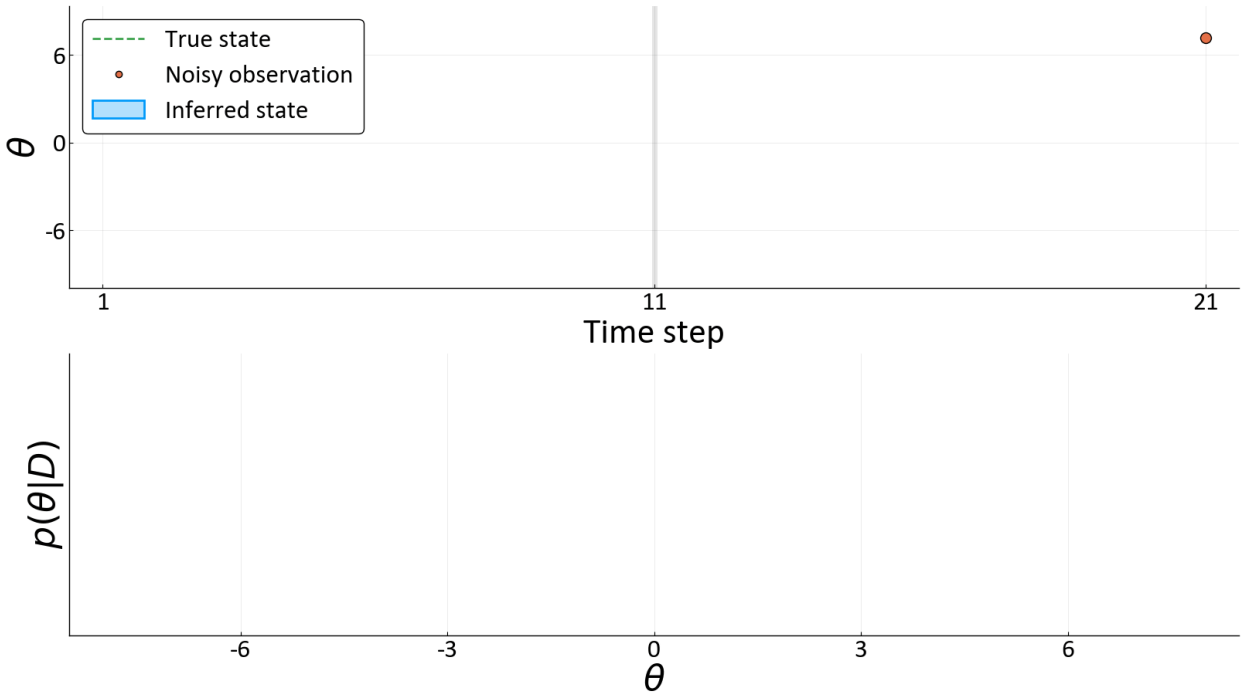
Static State



Dynamic State



Dynamic State



Accounting for Dynamics

Why isn't our model able to cope with dynamic states?

learning rate

$$\bar{x}_n = \bar{x}_{n-1} + \frac{1}{n}(x_n - \bar{x}_{n-1})$$

The learning rate decreases with the number of observations.

How can we avoid the learning rate from becoming too small?

In other words, how can we take into account that old information becomes obsolete?

Rescorla-Wagner Learning

One idea is to make it constant

learning rate

$$\bar{x}_n = \bar{x}_{n-1} + \alpha(x_n - \bar{x}_{n-1})$$

This implies taking only the last $\frac{1}{\alpha}$ data points into account.

However, if the learning rate is supposed to reflect uncertainty in Bayesian inference, then

- what should the value for α be?
- how to account for changes in uncertainty if α is constant?

Adaptive learning rates that reflect uncertainty

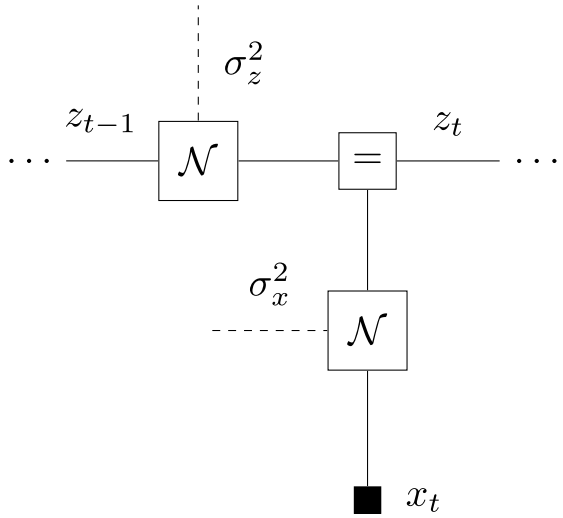
Several alternatives have been proposed in the literature:

- Kalman (1960)
- Sutton (1992)
- Nassar et al. (2010)
- Payzan-LeNestour & Bossaerts (2011)
- Mathys et al. (2011)
- Wilson et al. (2013)

Kalman Filter

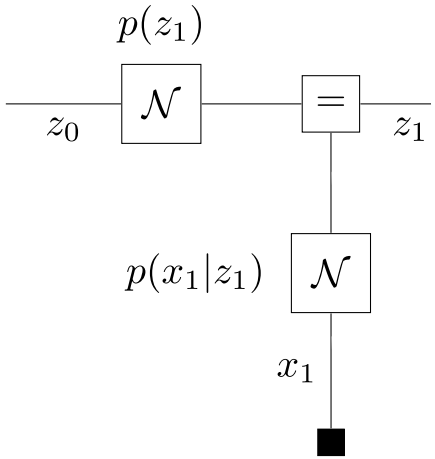
An efficient filter that estimates the internal state of a linear dynamical system from noisy measurements.

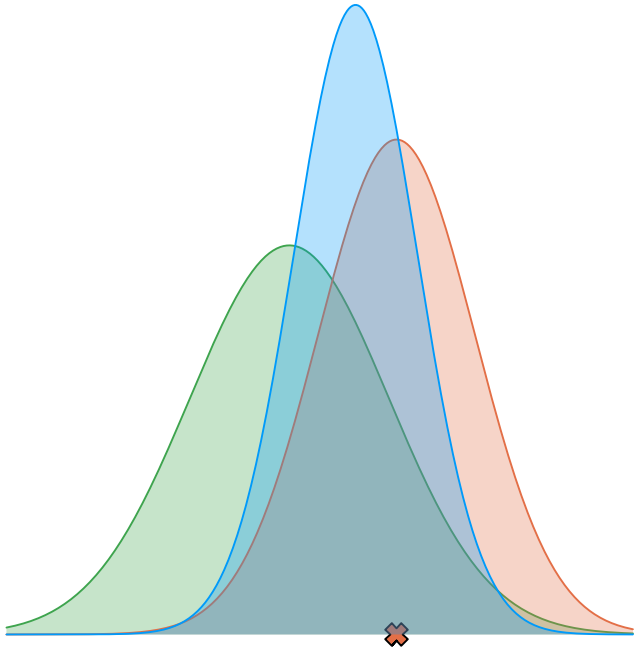
$$p(z_t \mid z_{t-1}) = \mathcal{N}(z_t \mid z_{t-1}, \sigma_z^2) \qquad \text{(state transition)}$$
$$p(x_t \mid z_t) = \mathcal{N}(x_t \mid z_t, \sigma_x^2) \qquad \text{(observation)}$$



It is **optimal** for linear dynamical systems, but realistic data usually require non-linear models.

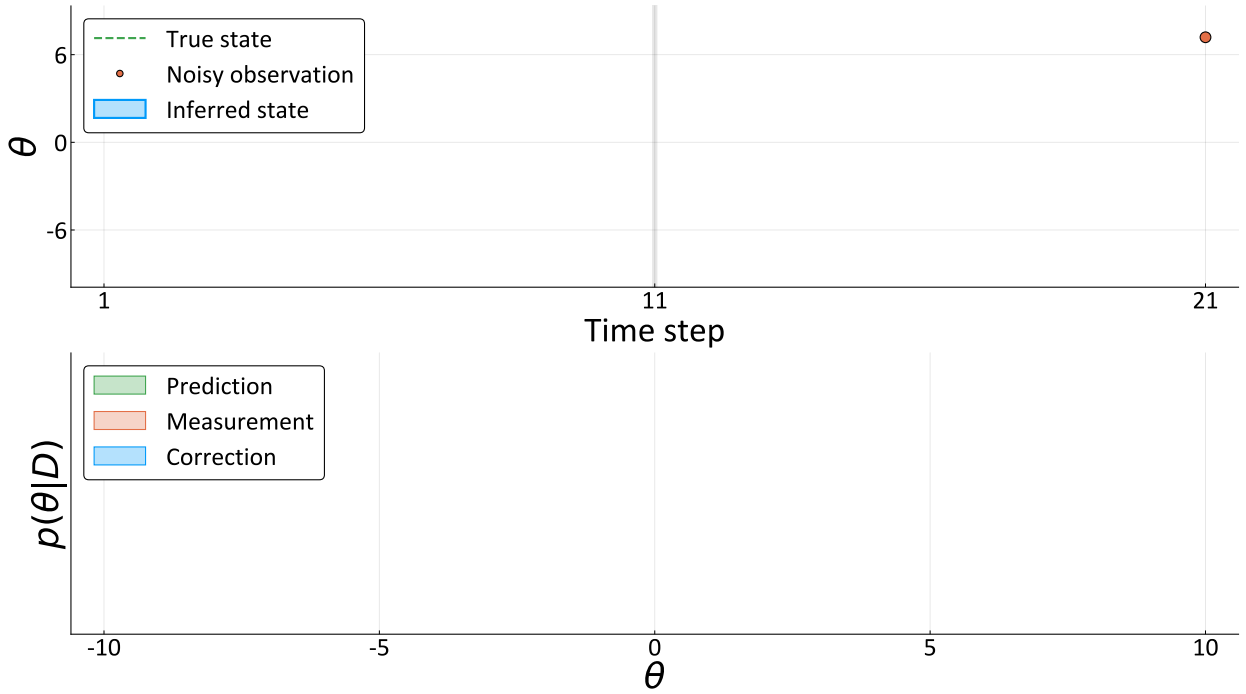
Kalman Filter



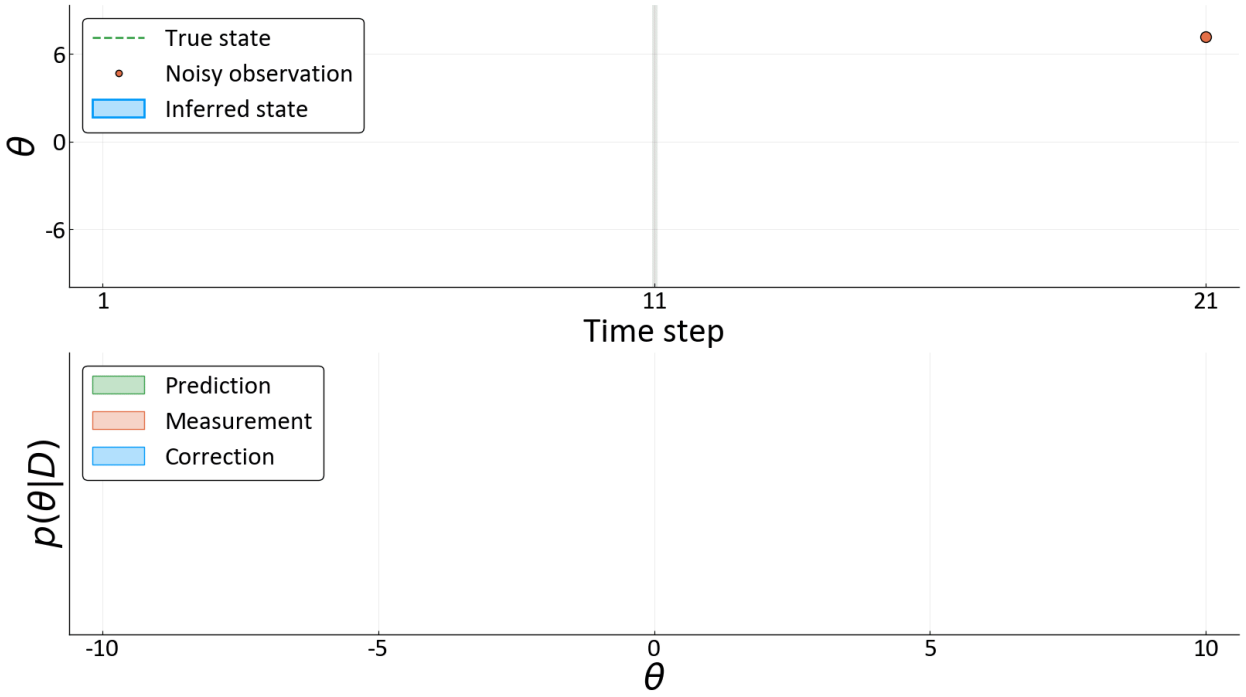


TU/e

Kalman Filter



Kalman Filter



2-Dimensional Kalman Filter

State space model:

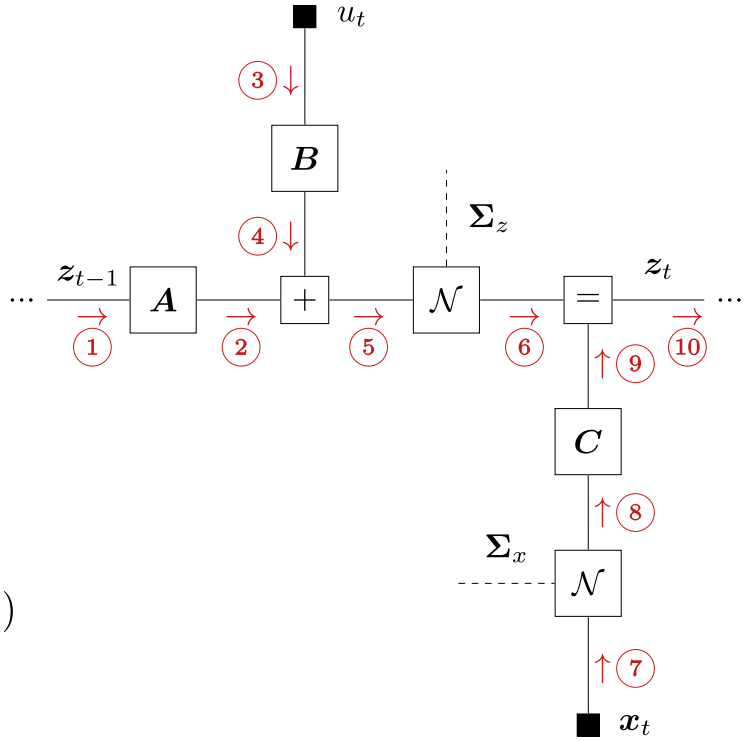
$$\boldsymbol{z}_t = \boldsymbol{A}\boldsymbol{z}_{t-1} + \boldsymbol{B}u_t + \mathcal{N}(0, \boldsymbol{\Sigma}_z)$$
$$\boldsymbol{x}_t = \boldsymbol{C}\boldsymbol{z}_t + \mathcal{N}(0, \boldsymbol{\Sigma}_x)$$

Conditional independence assumptions:

$$p(\boldsymbol{z}_t|\boldsymbol{z}_0, \ldots, \boldsymbol{z}_{t-1}) = p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1})$$
$$p(\boldsymbol{x}_t|\boldsymbol{z}_0, \ldots, \boldsymbol{z}_t) = p(\boldsymbol{x}_t|\boldsymbol{z}_t)$$

Joint probability:

$$p(\boldsymbol{z}_0, \ldots, \boldsymbol{z}_t, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_t) = p(\boldsymbol{z}_0) \prod_{i=1}^{t} p(\boldsymbol{x}_i|\boldsymbol{z}_i) p(\boldsymbol{z}_i|\boldsymbol{z}_{i-1})$$



2-Dimensional Kalman Filter

Joint probability:

$$p(\boldsymbol{z}_0, \ldots, \boldsymbol{z}_t, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_t) = p(\boldsymbol{z}_0) \prod_{i=1}^{t} p(\boldsymbol{x}_i|\boldsymbol{z}_i) p(\boldsymbol{z}_i|\boldsymbol{z}_{i-1})$$

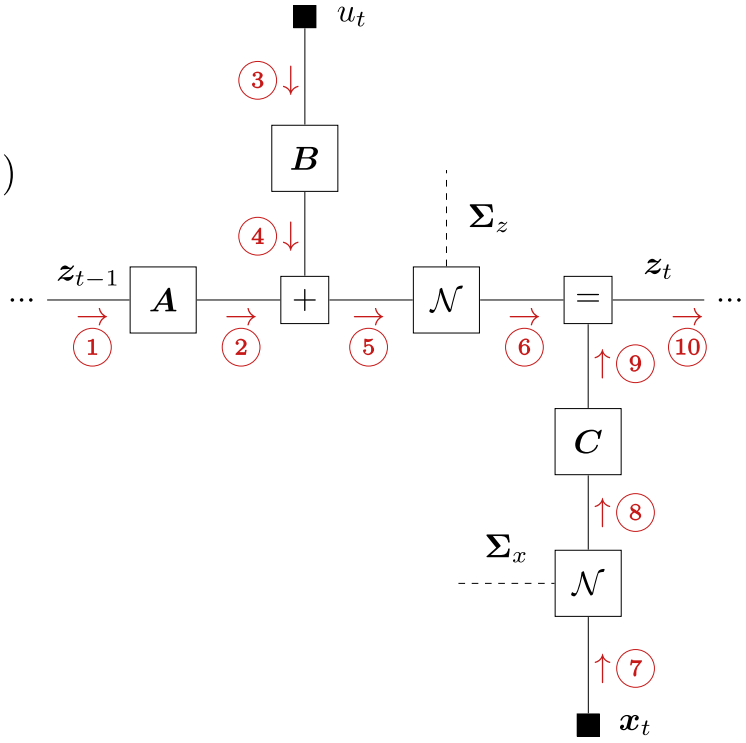
Prediction step:

$$p(\boldsymbol{z}_t|\boldsymbol{x}_{1:t-1}) = \int p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}) p(\boldsymbol{z}_{t-1}|\boldsymbol{x}_{1:t-1}) \, \mathrm{d}\boldsymbol{z}_{t-1}$$

Update step:

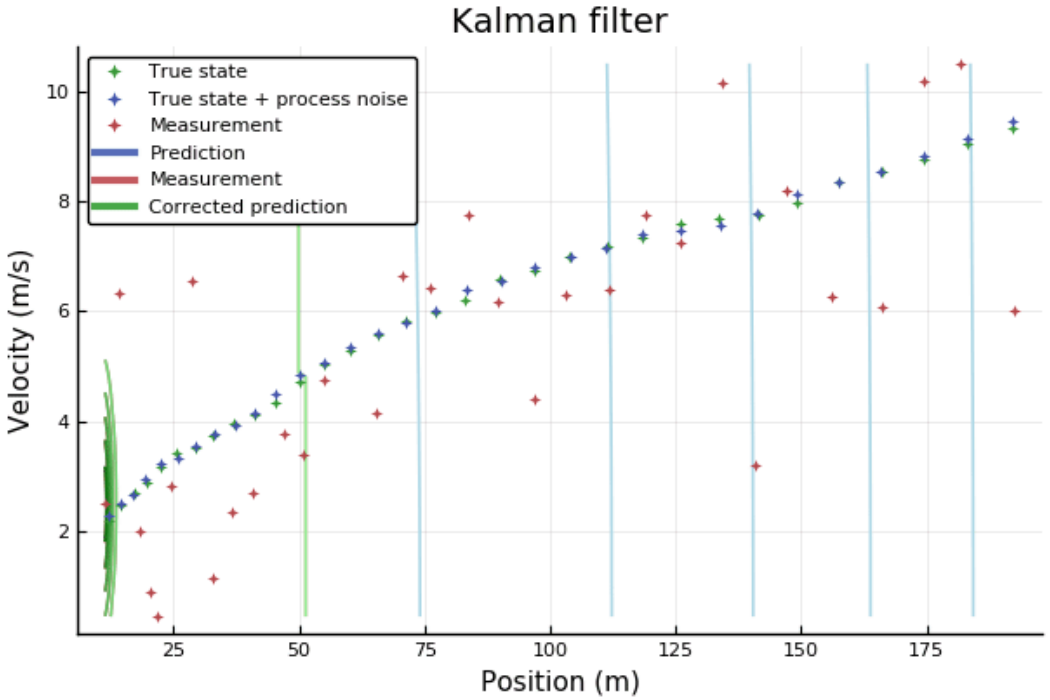
$$p(\boldsymbol{z}_t|\boldsymbol{x}_{1:t}) = \frac{p(\boldsymbol{x}_t|\boldsymbol{z}_t) p(\boldsymbol{z}_t|\boldsymbol{x}_{1:t-1})}{p(\boldsymbol{x}_t|\boldsymbol{x}_{1:t-1})}$$

$$p(\boldsymbol{x}_t|\boldsymbol{x}_{1:t-1}) = \int p(\boldsymbol{x}_t|\boldsymbol{z}_t) p(\boldsymbol{z}_t|\boldsymbol{x}_{1:t-1}) \, \mathrm{d}\boldsymbol{z}_t$$



TU/e

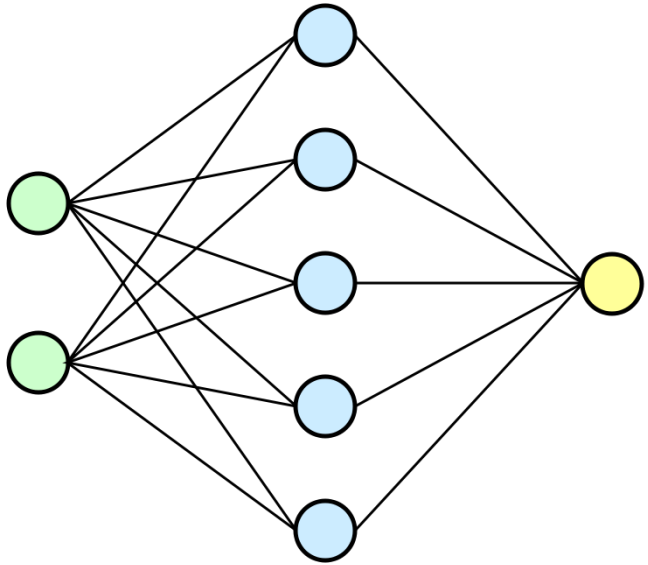
2-Dimensional Kalman Filter



Closing Remarks

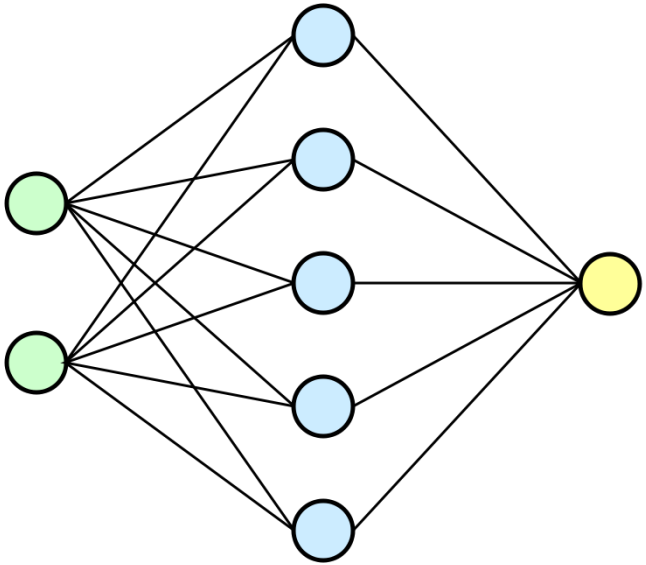
Drawbacks of Deep Learning

- Neural networks compute **point estimates**
- **Overly confident** decisions in classification, prediction and actuation tasks
- Prone to **overfitting**
- Contain many hyperparameters that may require **specific tuning**



Drawbacks of Deep Learning

- Very **data hungry**
- Very **compute-intensive** to train and deploy
- Poor at representing **uncertainty**
- **Easily fooled** by adversarial examples
- **Difficult to optimize**, e.g. choice of architecture, learning procedure, initialization, etc.
- Uninterpretable **black-boxes**, difficult to trust



Bayesian Inference

Aim: Use probability theory to express all forms of uncertainty

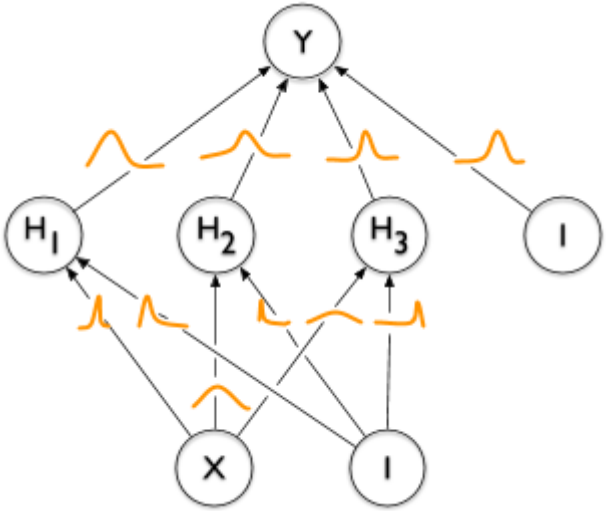
- Provides a way to **reason about uncertainty**
- AI Safety: medicine, engineering, finance, etc.

TU/e

Bayesian Neural Network

A DNN with a **prior distribution** on the weights.

- Accounts for uncertainty in weights
- Propagates this into uncertainty about predictions
- More robust against overfitting
  - Randomly sampling over network weights as a cheap form of model averaging



TU/e

Questions?

Takeaway

We've seen:

- A view-point of ML that provides a compass through the complex pile of existing ML algorithms
- A change of paradigm in the way software is programmed
- A practical tool to use when building real-world applications
- A vision of how ML can be democratized