

# Deep Learning with Bayesian Principles

Mohammad Emtiyaz Khan

RIKEN Center for AI Project, Tokyo

<http://emtiyaz.github.io>



With a significant help from

Roman  
Bachmann  
(RIKEN-AIP)

Xiangming  
Meng  
(RIKEN-AIP)



# The Goal of My Research

*“To discover the **fundamental principles of learning from data** and use them to **develop algorithms** that can learn like living beings.”*



Human Learning at  
the age of 6 months.



Human Learning at  
the age of 6 months.



Human Learning at  
the age of 6 months.



Converged at the  
age of 12 months





Converged at the  
age of 12 months



Converged at the  
age of 12 months



Transfer  
skills  
at the age  
of 14  
months





Transfer  
skills  
at the age  
of 14  
months





Transfer  
skills  
at the age  
of 14  
months



# Human learning $\neq$ Deep learning

Life-long learning from  
small chunks of data in  
a non-stationary world

Bulk learning from a  
large amount of data in  
a stationary world

Parisi, German I., et al. "Continual lifelong learning with neural networks: A review." *Neural Networks* (2019)

**Human learning**  $\neq$  **Deep learning**

**Life-long** learning from  
**small** chunks of data in  
a **non-stationary** world

**Bulk** learning from a  
**large** amount of data in  
a **stationary** world

My current research focuses on reducing this gap!

Parisi, German I., et al. "Continual lifelong learning with neural networks: A review." *Neural Networks* (2019)

# Bayesian ~~Human~~ learning $\neq$ Deep learning

Life-long learning from  
small chunks of data in  
a non-stationary world

Bulk learning from a  
large amount of data in  
a stationary world

My current research focuses on reducing this gap!

Parisi, German I., et al. "Continual lifelong learning with neural networks: A review." *Neural Networks* (2019)

# Bayesian ~~Human~~ learning $\neq$ Deep learning

Life-long learning from  
small chunks of data in  
a non-stationary world

Bulk learning from a  
large amount of data in  
a stationary world

My current research focuses on reducing this gap!

Parisi, German I., et al. "Continual lifelong learning with neural networks: A review." *Neural Networks* (2019)  
Friston, K. "The free-energy principle: a unified brain theory?." *Nature reviews neuroscience* (2010)  
Geisler, W. S., and Randy L. D. "Bayesian natural selection and the evolution of perceptual systems." *Philosophical Transactions of the Royal Society of London. Biological Sciences* (2002)

# Bayesian learning

Bayesian models

(GPs, BayesNets, PGMs,)

Bayesian inference

(Bayes rule)

# Deep learning

Deep models

(MLP, CNN, RNN etc.)

Stochastic training

(SGD, RMSprop, Adam)

# Bayesian learning

## Bayesian models

(GPs, BayesNets, PGMs,)

## Bayesian inference

(Bayes rule)

# Deep learning

## Deep models

(MLP, CNN, RNN etc.)

## Stochastic training

(SGD, RMSprop, Adam)

	Bayes	DL
Can handle large data and complex models?	✗	✓
Scalable training?	✗	✓
Can estimate uncertainty?	✓	✗
Can perform sequential / active /online / incremental learning?	✓	✗

# **Bringing the two together**

To combine their complimentary strengths to solve challenging learning problems



# Deep Learning with Bayesian Principles

# Deep Learning with Bayesian Principles

- Bayesian principles as a general principle
  - To design/improve/generalize learning-algorithms
  - By computing “posterior approximations”

# Deep Learning with Bayesian Principles

- Bayesian principles as a general principle
  - To design/improve/generalize learning-algorithms
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc

# Deep Learning with Bayesian Principles

- Bayesian principles as a general principle
  - To design/improve/generalize learning-algorithms
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning

# Deep Learning with Bayesian Principles

- Bayesian principles as a general principle
  - To design/improve/generalize learning-algorithms
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning
- Impact: Everything with one common principle.

**Is this different from  
Bayesian Deep Learning?**

# Scope of the Tutorial

- Audience: Deep learners and Bayesians
- Goal: To bring the two together
- This tutorial is not about
  - Bayesian deep-learning methods
  - Classical Bayesian inference methods
  - Approximate Bayesian Inference
  - Uncertainty estimation
  - Generative Models, VAE, etc.
  - Gaussian processes and NN architectures

# Disclaimer

- I might not have time to discuss many important/relevant works
  - If you think I should have included some of those, please send me email and I will try to include it the next time
- The content of the tutorial is based on my own biased opinion (and expertise)
  - A lot of it is based on my own work (about 40% or so)



# **Deep Learning vs Bayesian Learning**

# Deep Learning (DL)

Frequentist: Empirical Risk Minimization (ERM) or Maximum Likelihood Principle, etc.

$$\min_{\theta} \ell(\mathcal{D}, \theta)$$

The diagram illustrates the components of the empirical risk minimization formula. The expression  $\min_{\theta} \ell(\mathcal{D}, \theta)$  is shown. Below the  $\ell$  in the function, the word "Loss" is written in blue. Below the  $\mathcal{D}$ , the word "Data" is written in blue. Below the  $\theta$ , the words "Model Params" are written in blue. Two blue arrows point upwards: one from "Data" to  $\mathcal{D}$  and another from "Model Params" to  $\theta$ .

# Deep Learning (DL)

Frequentist: Empirical Risk Minimization (ERM) or Maximum Likelihood Principle, etc.

$$\min_{\theta} \ell(\mathcal{D}, \theta) = \sum_{i=1}^N [y_i - f_{\theta}(x_i)]^2 + \gamma \theta^T \theta$$

Diagram illustrating the components of the equation:

- $\ell(\mathcal{D}, \theta)$ : Loss function, taking Data ( $\mathcal{D}$ ) and Model Params ( $\theta$ ) as input.
- $\sum_{i=1}^N$ : Sum over  $N$  data points.
- $[y_i - f_{\theta}(x_i)]^2$ : Squared error term, where  $y_i$  is the target and  $f_{\theta}(x_i)$  is the prediction from the Deep Network.
- $\gamma \theta^T \theta$ : Regularization term, where  $\gamma$  is the regularization coefficient and  $\theta^T \theta$  is the squared L2 norm of the model parameters.

# Deep Learning (DL)

Frequentist: Empirical Risk Minimization (ERM) or Maximum Likelihood Principle, etc.

$$\min_{\theta} \ell(\mathcal{D}, \theta) = \sum_{i=1}^N [y_i - f_{\theta}(x_i)]^2 + \gamma \theta^T \theta$$

Diagram illustrating the components of the loss function:

- $\ell$ : Loss
- $\mathcal{D}$ : Data
- $\theta$ : Model Params
- $f_{\theta}$ : Deep Network

DL Algorithm:  $\theta \leftarrow \theta - \rho H_{\theta}^{-1} \nabla_{\theta} \ell(\theta)$

# Deep Learning (DL)

Frequentist: Empirical Risk Minimization (ERM) or Maximum Likelihood Principle, etc.

$$\min_{\theta} \ell(\mathcal{D}, \theta) = \sum_{i=1}^N [y_i - f_{\theta}(x_i)]^2 + \gamma \theta^T \theta$$

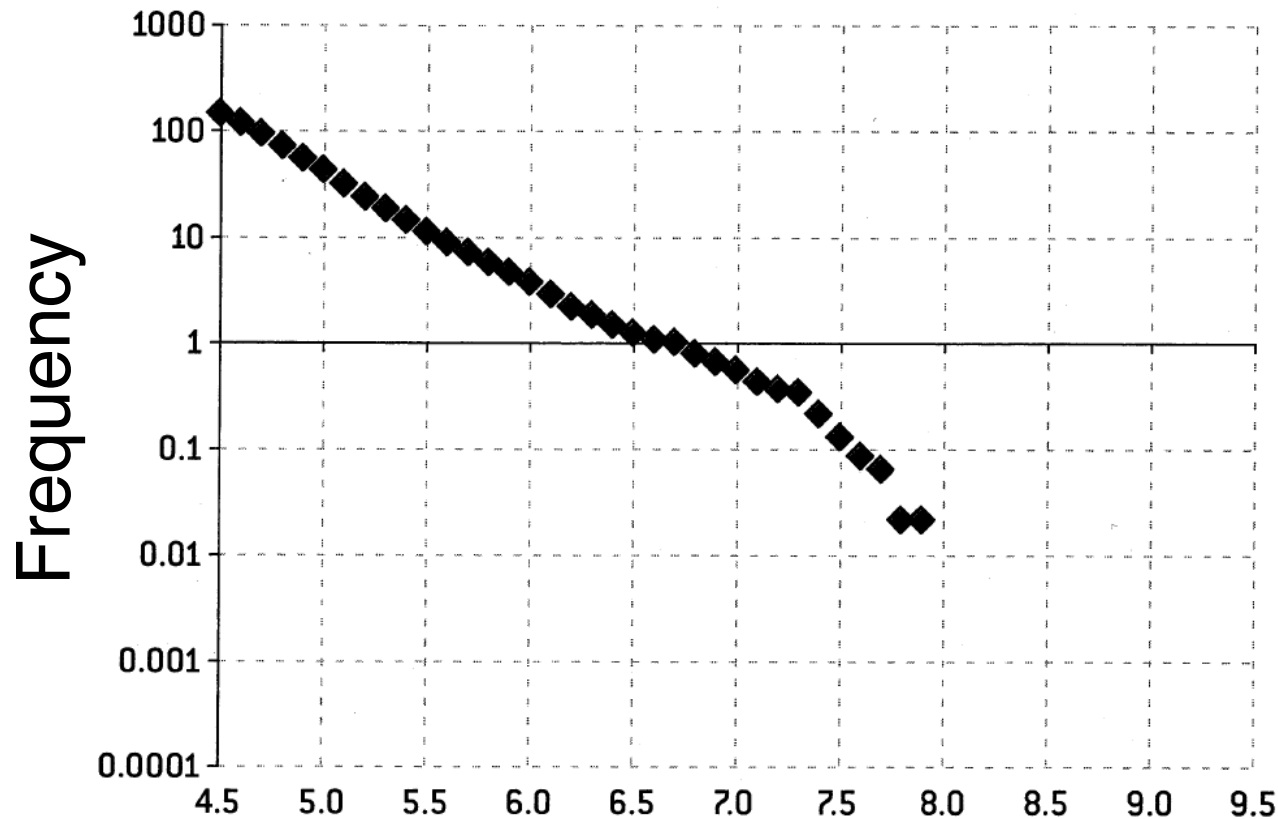
Diagram illustrating the components of the loss function:

- Loss** points to  $\ell(\mathcal{D}, \theta)$
- Data** points to  $\mathcal{D}$
- Model Params** points to  $\theta$
- Deep Network** points to  $f_{\theta}(x_i)$

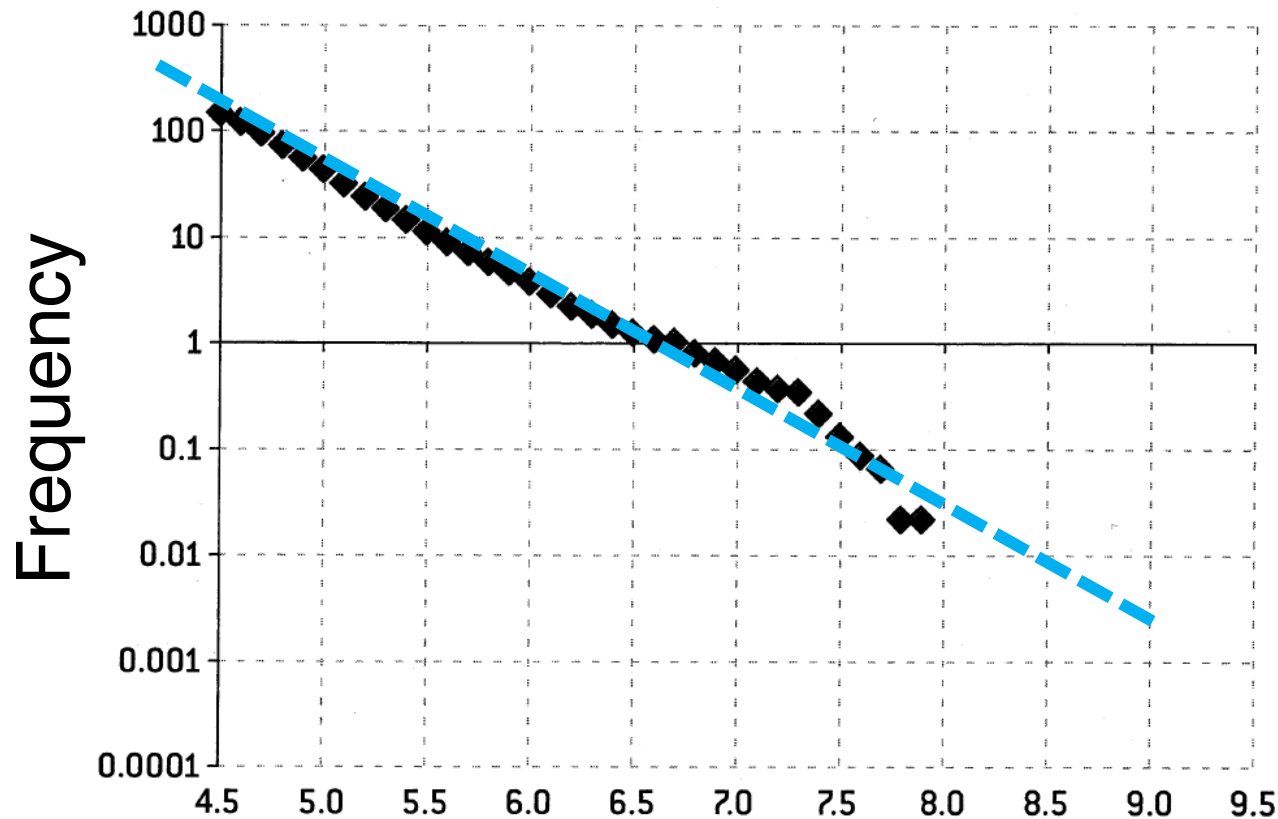
DL Algorithm:  $\theta \leftarrow \theta - \rho H_{\theta}^{-1} \nabla_{\theta} \ell(\theta)$

Scales well to large data and complex model, and very good performance in practice.

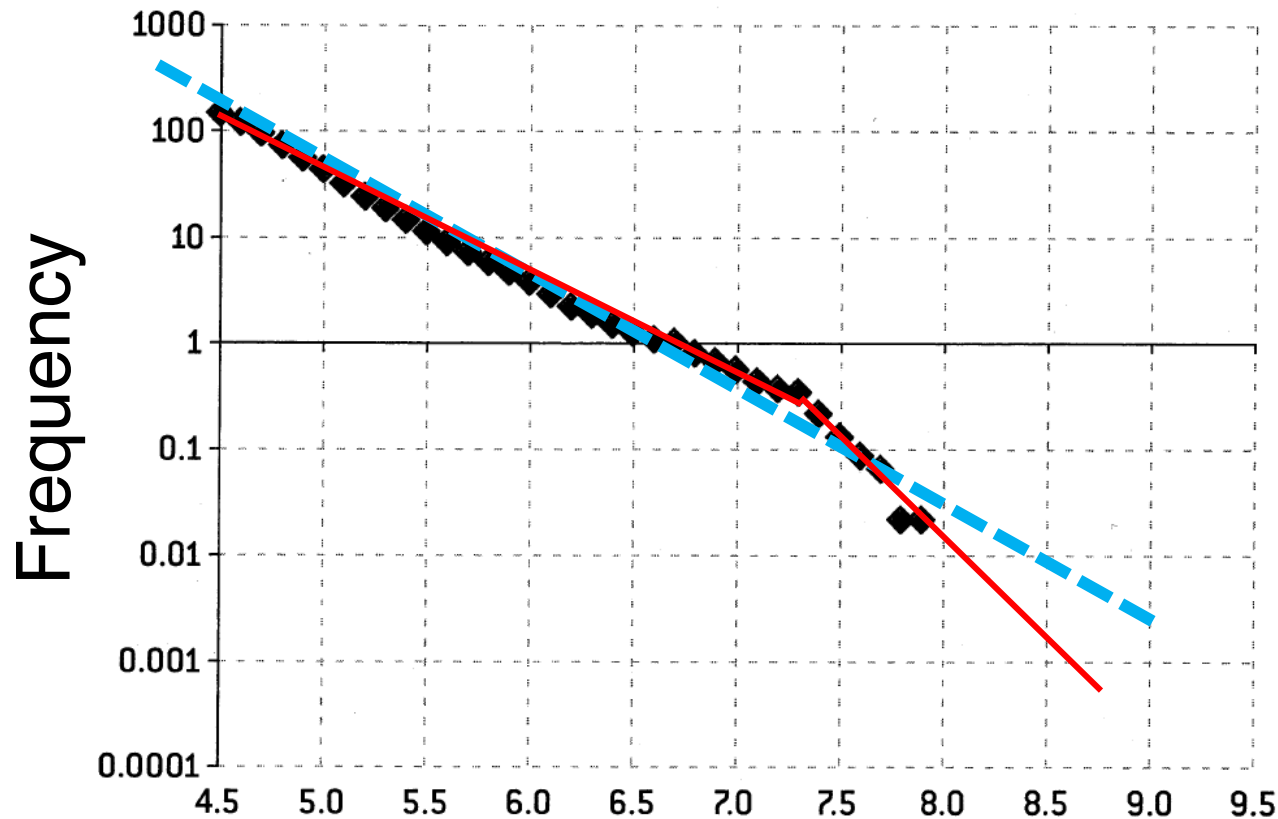
# Example: Which is a Better Fit?



# Example: Which is a Better Fit?

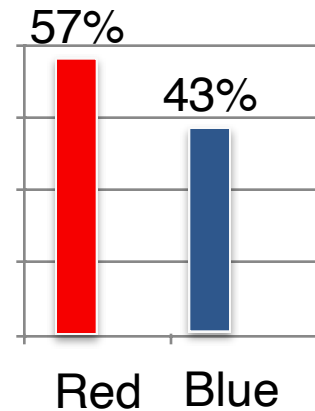
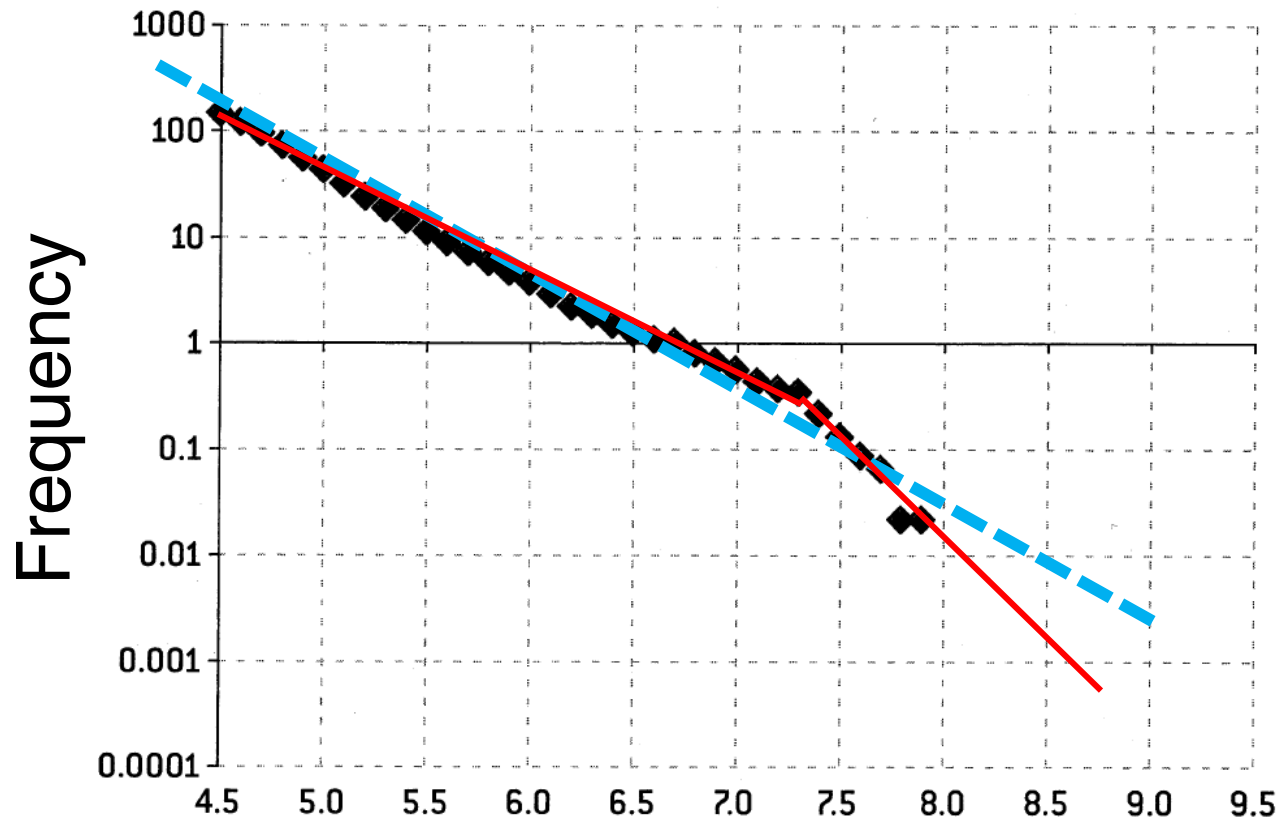


# Example: Which is a Better Fit?

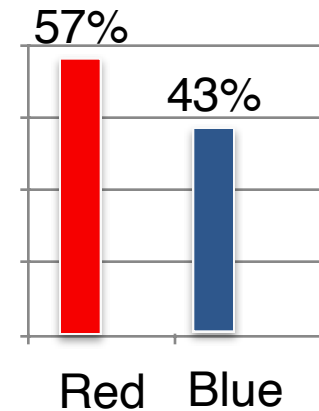
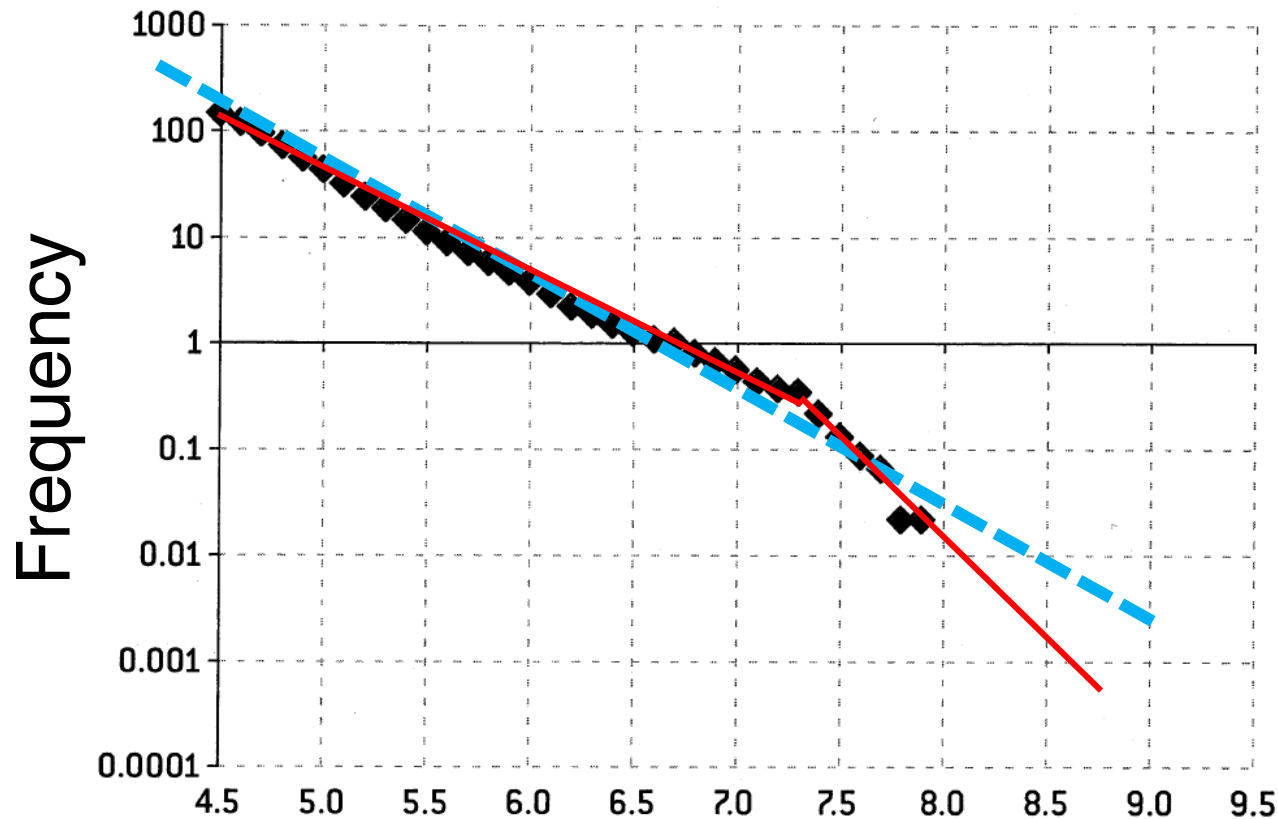




# Example: Which is a Better Fit?

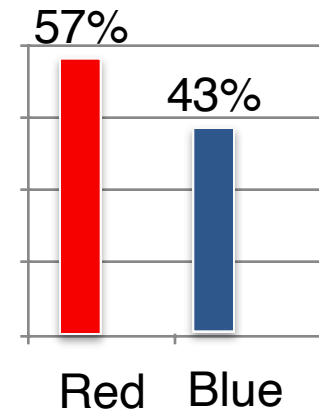
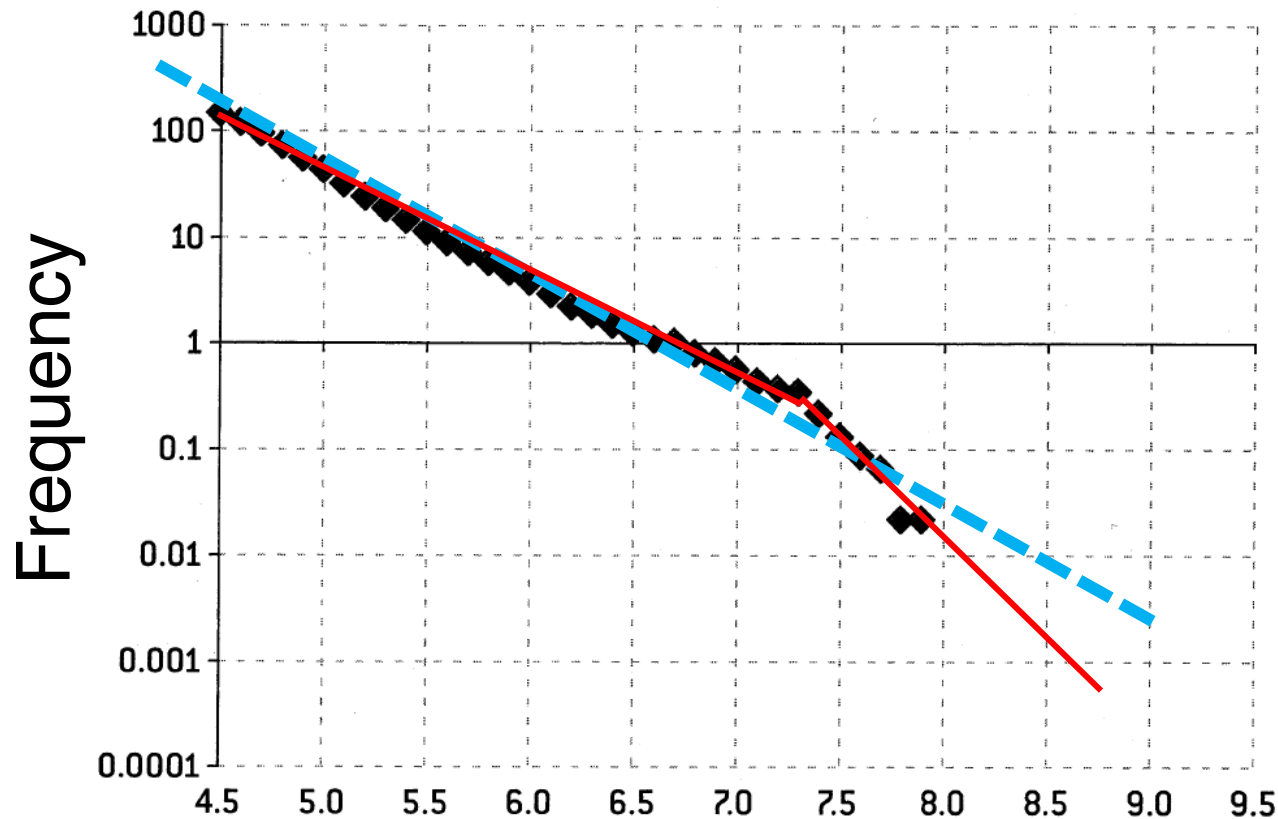


# Example: Which is a Better Fit?



Magnitude of Earthquake

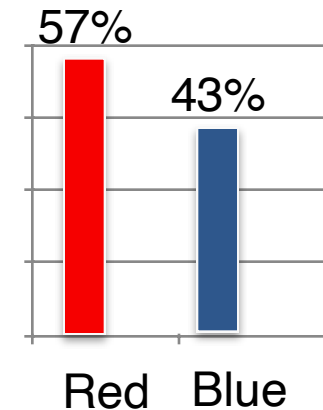
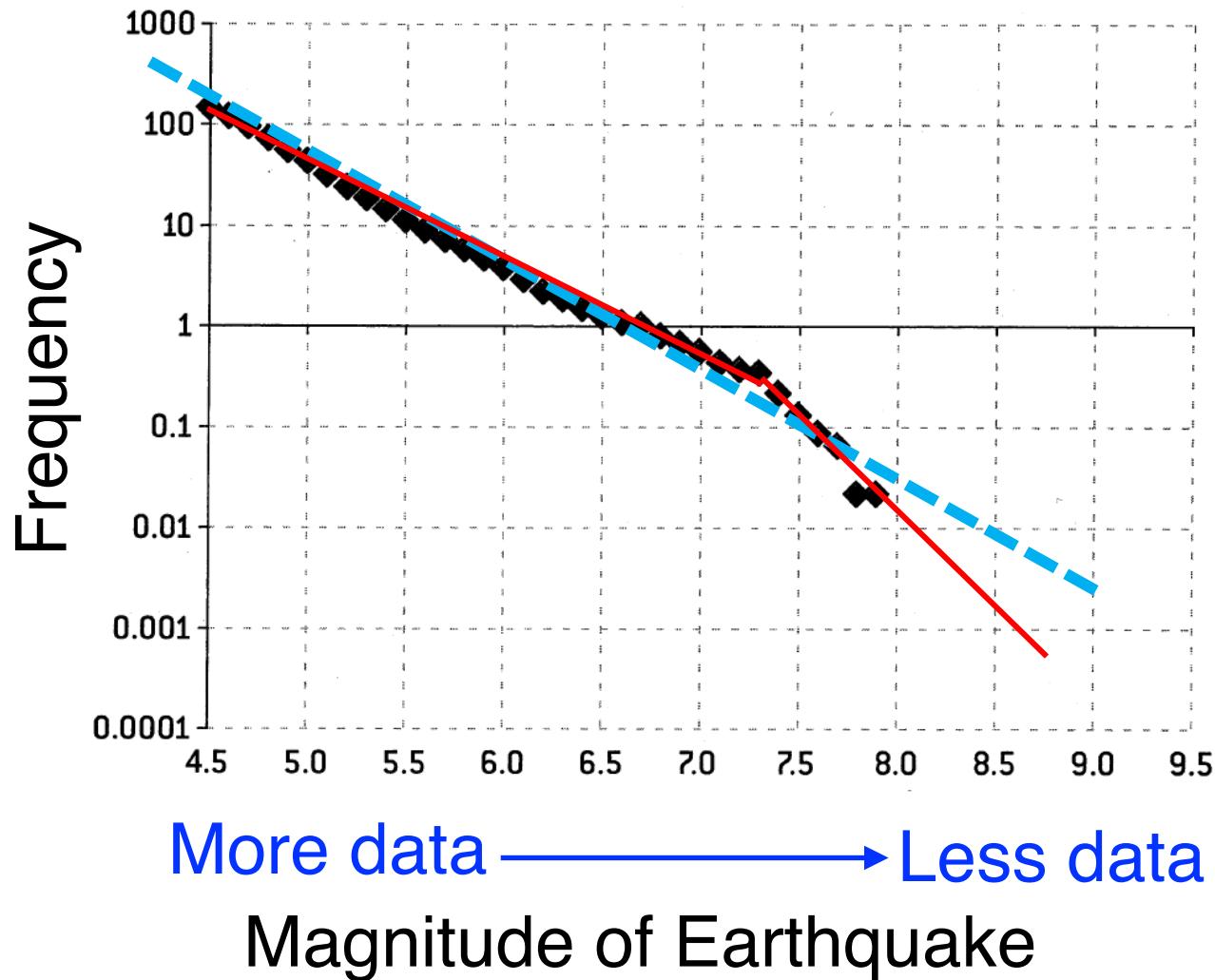
# Example: Which is a Better Fit?



More data  $\longrightarrow$  Less data

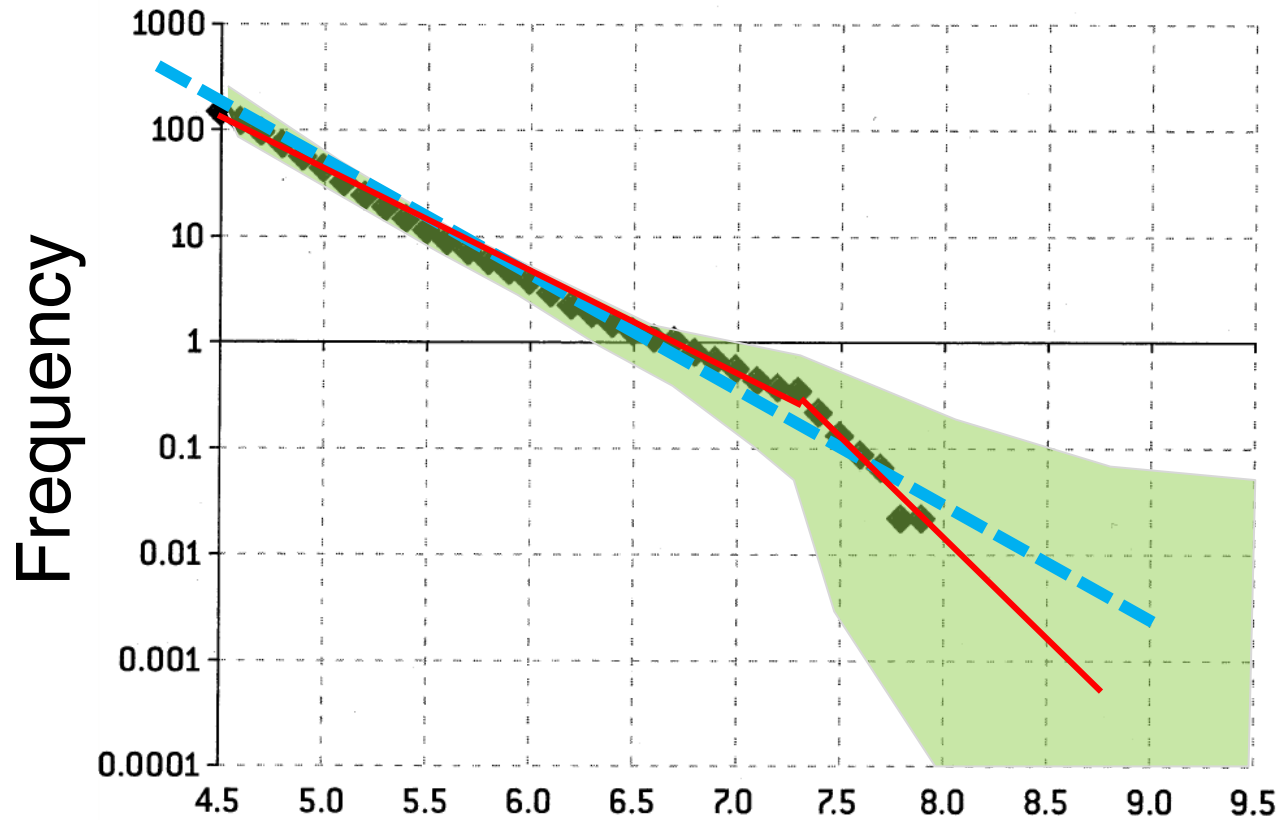
Magnitude of Earthquake

# Example: Which is a Better Fit?



Red is more  
risky than  
the blue

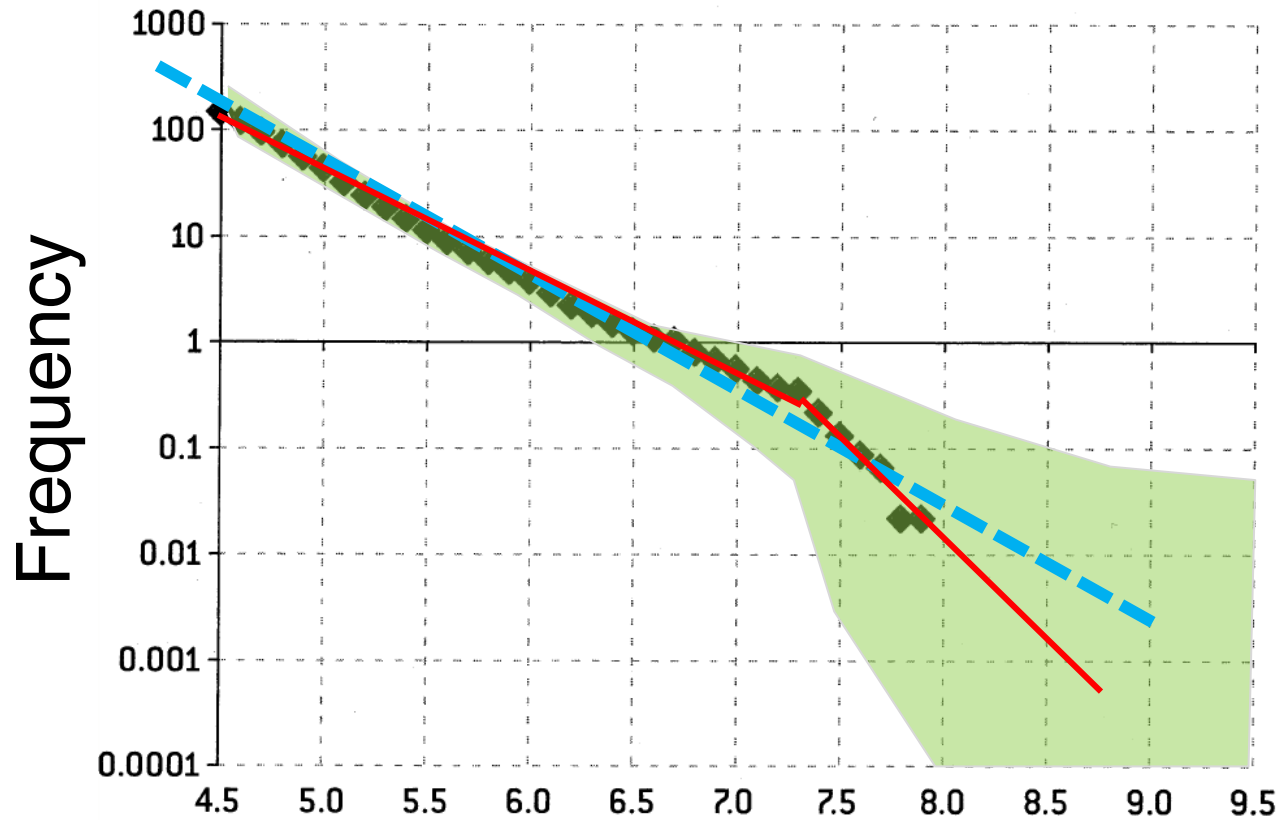
# Example: Which is a Better Fit?



Uncertainty:  
“What the  
model does  
not know”

More data —————> Less data  
Magnitude of Earthquake

# Example: Which is a Better Fit?

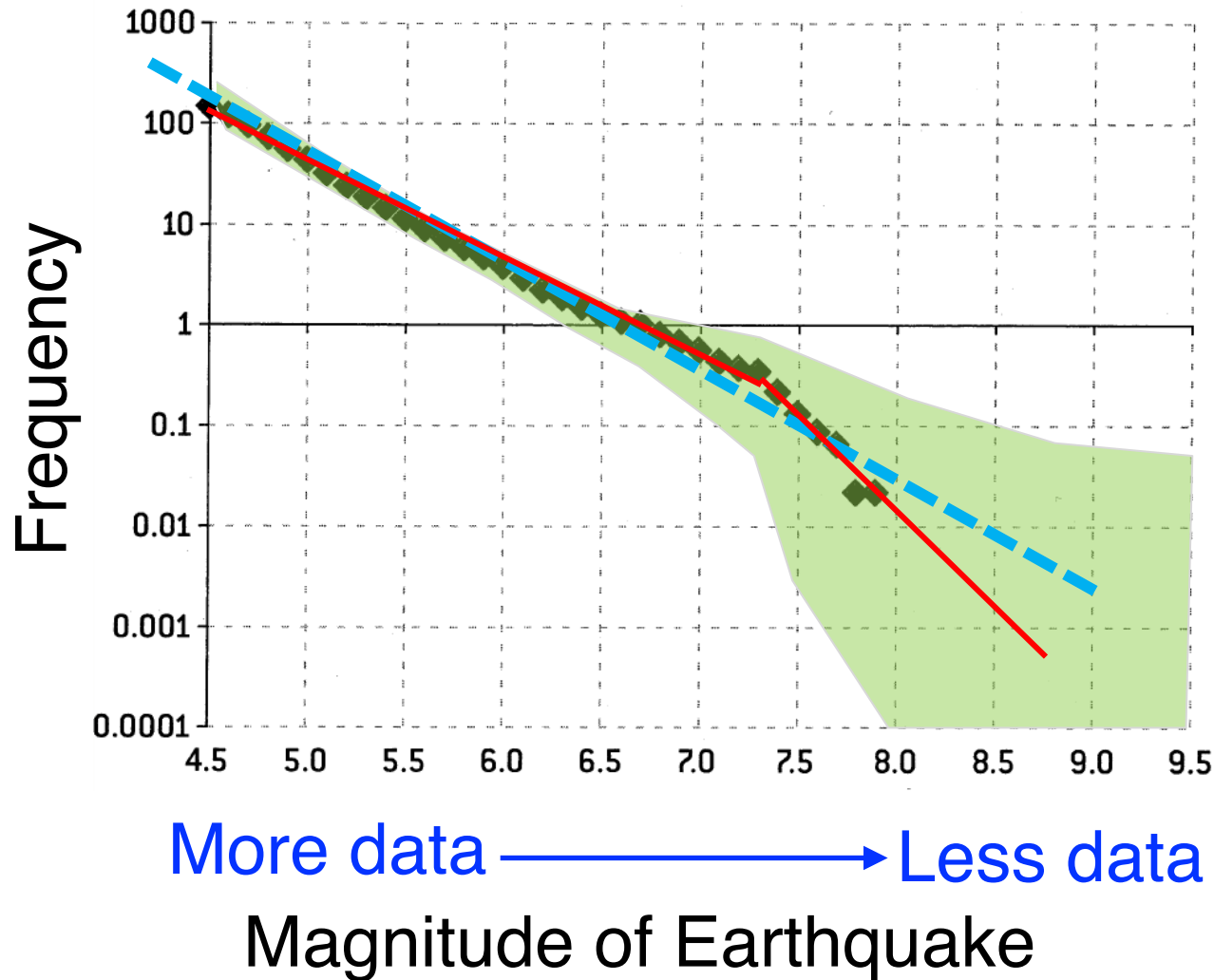


Uncertainty:  
“What the  
model does  
not know”

Choose less  
risky options!

More data  $\longrightarrow$  Less data  
Magnitude of Earthquake

# Example: Which is a Better Fit?

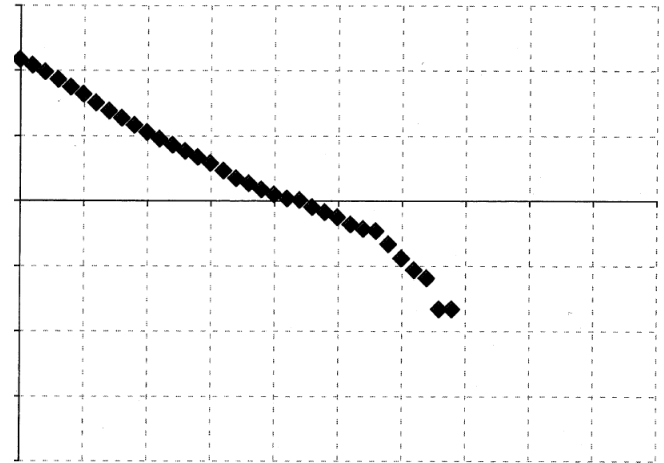


Uncertainty:  
“What the  
model does  
not know”

Choose less  
risky options!

Avoid data  
bias with  
uncertainty!

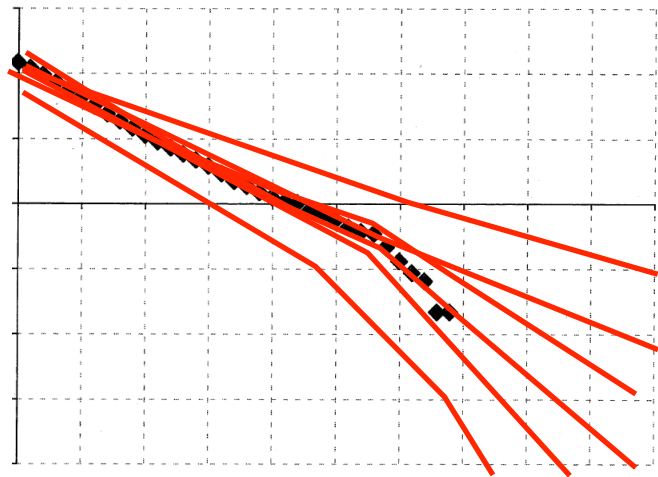
# Bayesian Principles





# Bayesian Principles

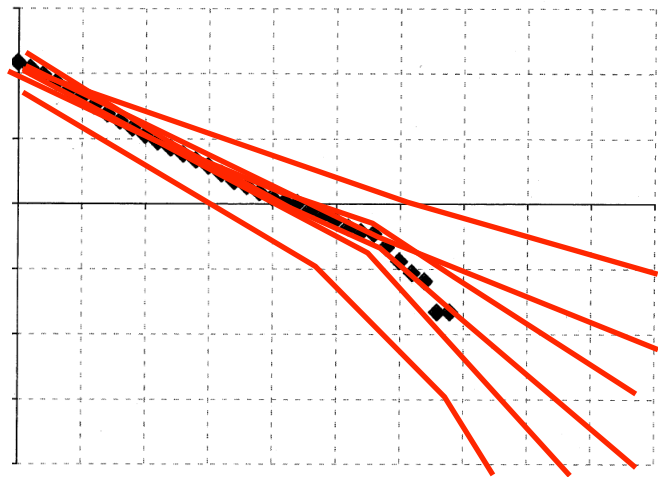
1. Sample  $\theta \sim p(\theta)$  prior



# Bayesian Principles

1. Sample  $\theta \sim p(\theta)$  prior

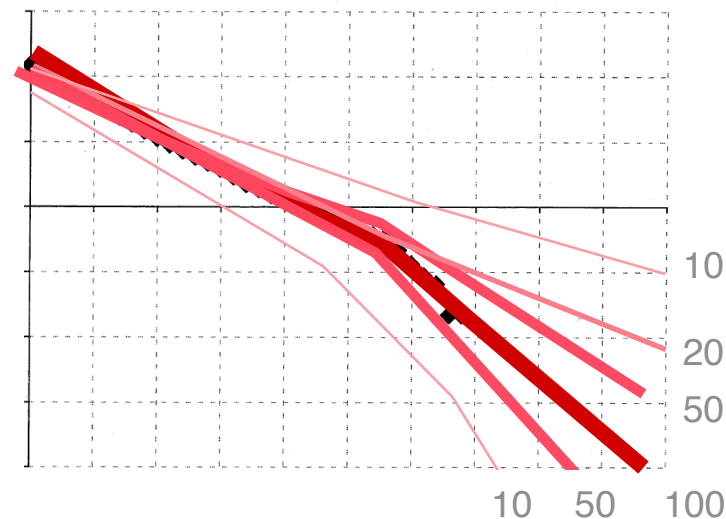
2. Score  $p(\mathcal{D}|\theta) = \prod_{i=1}^N p(y_i | f_{\theta}(x_i))$  Likelihood



# Bayesian Principles

1. Sample  $\theta \sim p(\theta)$  prior

2. Score  $p(\mathcal{D}|\theta) = \prod_{i=1}^N p(y_i | f_{\theta}(x_i))$  Likelihood



# Bayesian Principles

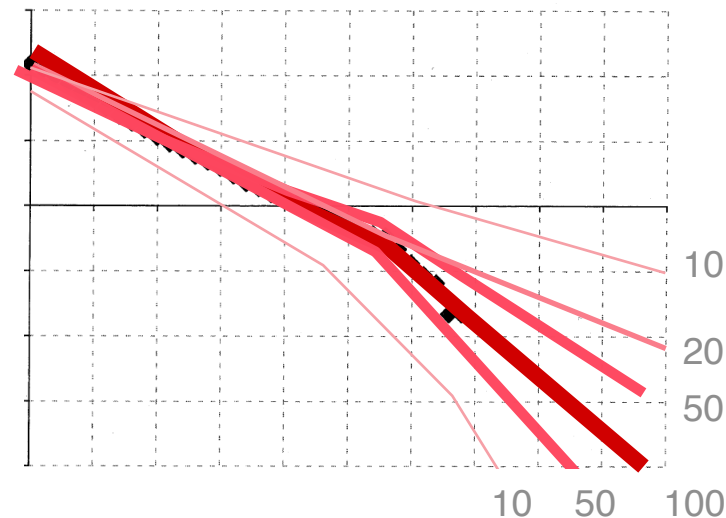
1. Sample  $\theta \sim p(\theta)$  prior

2. Score  $p(\mathcal{D}|\theta) = \prod_{i=1}^N p(y_i | f_{\theta}(x_i))$  Likelihood

3. Normalize

Posterior Likelihood x Prior

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



# Bayesian Principles

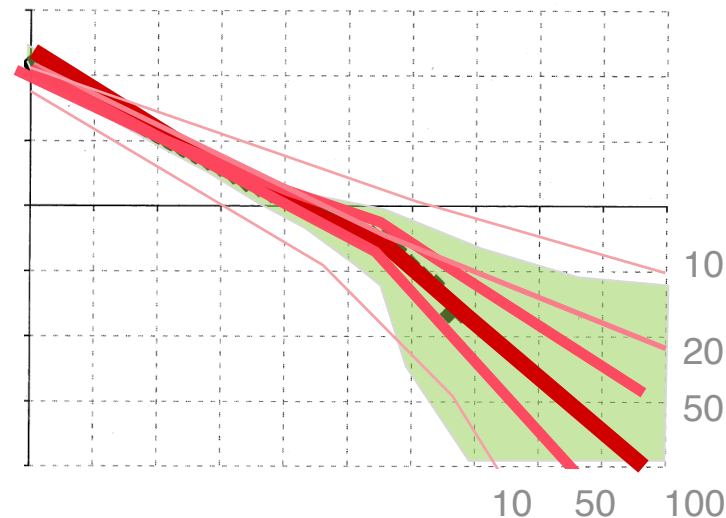
1. Sample  $\theta \sim p(\theta)$  prior

2. Score  $p(\mathcal{D}|\theta) = \prod_{i=1}^N p(y_i | f_{\theta}(x_i))$  Likelihood

3. Normalize

Posterior Likelihood x Prior

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



# Bayesian Principles

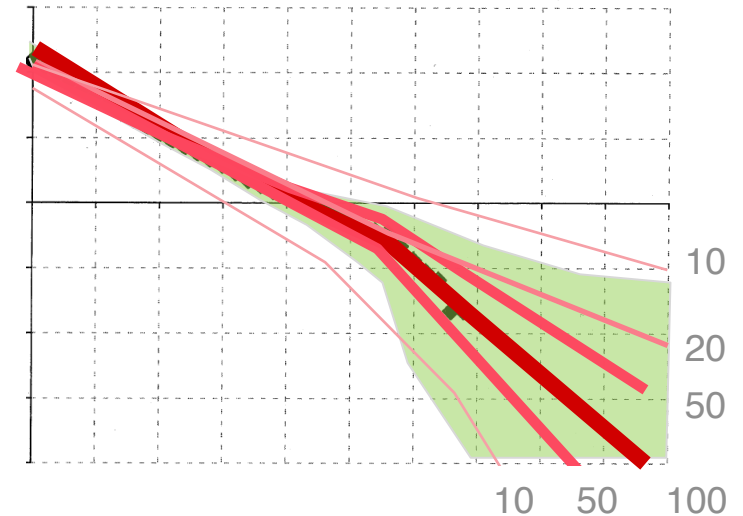
1. Sample  $\theta \sim p(\theta)$  prior

2. Score  $p(\mathcal{D}|\theta) = \prod_{i=1}^N p(y_i | f_{\theta}(x_i))$  Likelihood

3. Normalize

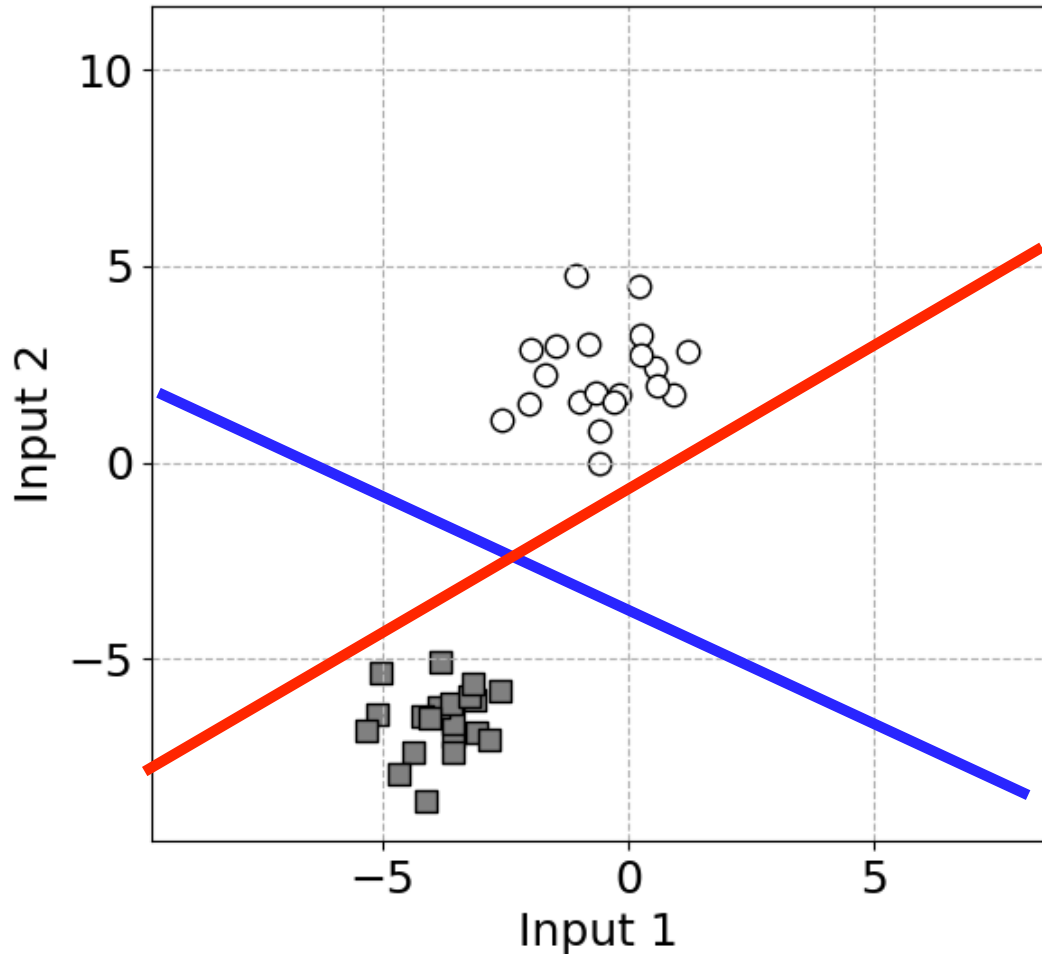
Posterior Likelihood x Prior

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

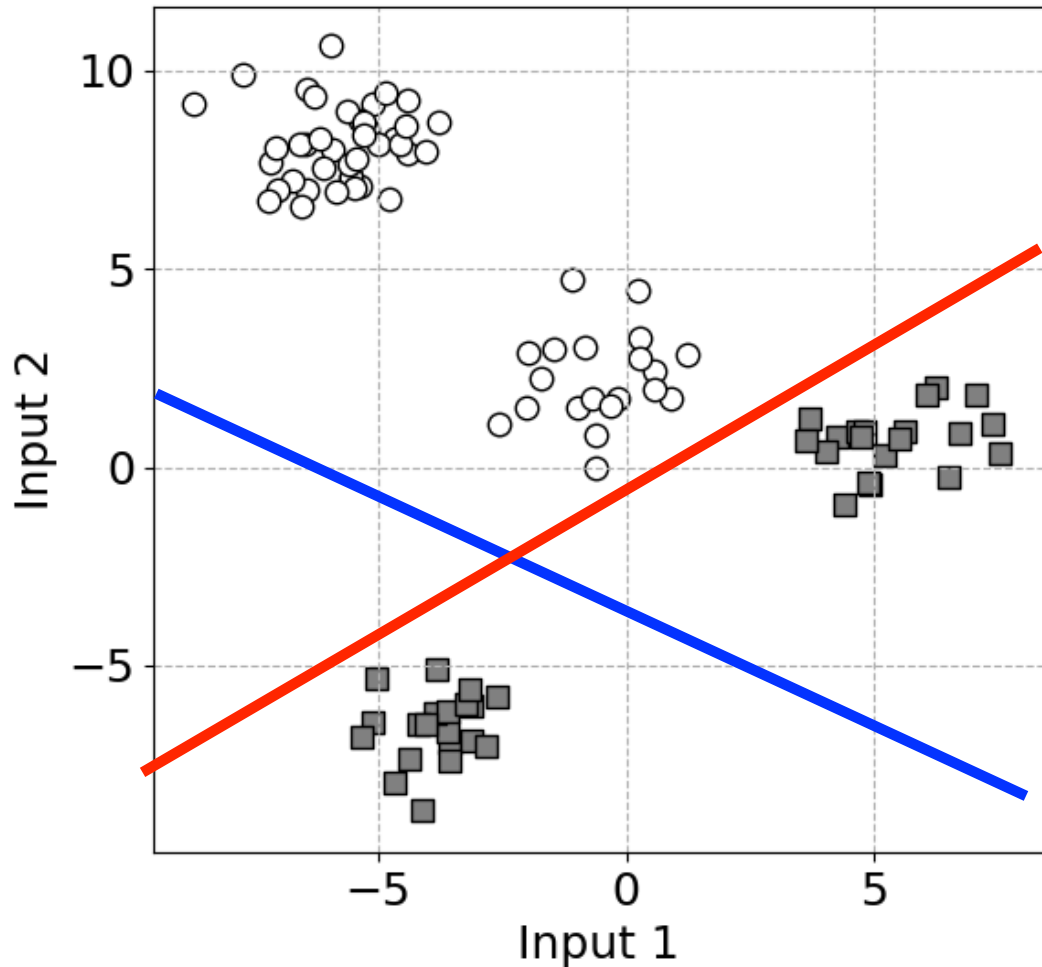


A global method: Integrates over all models  
Does not scale to large problem

# Which is a good classifier?

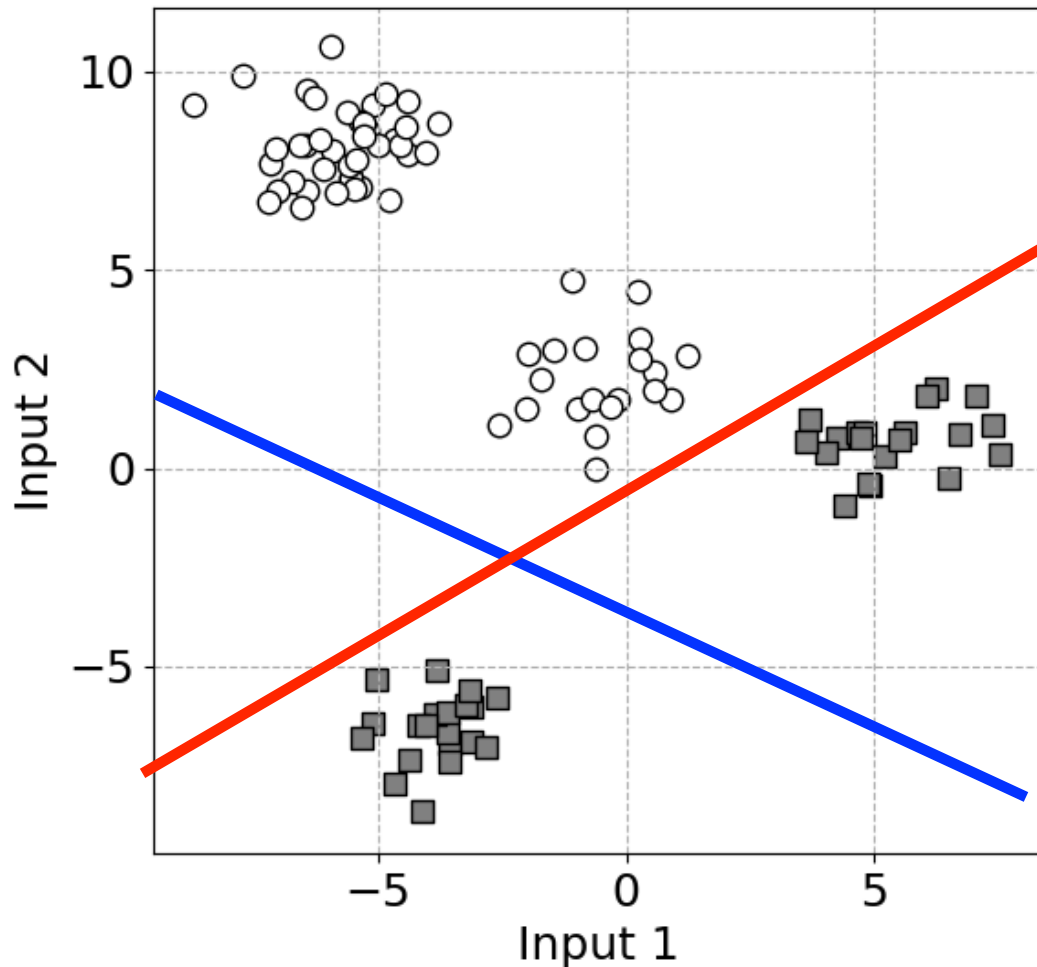


# Which is a good classifier?



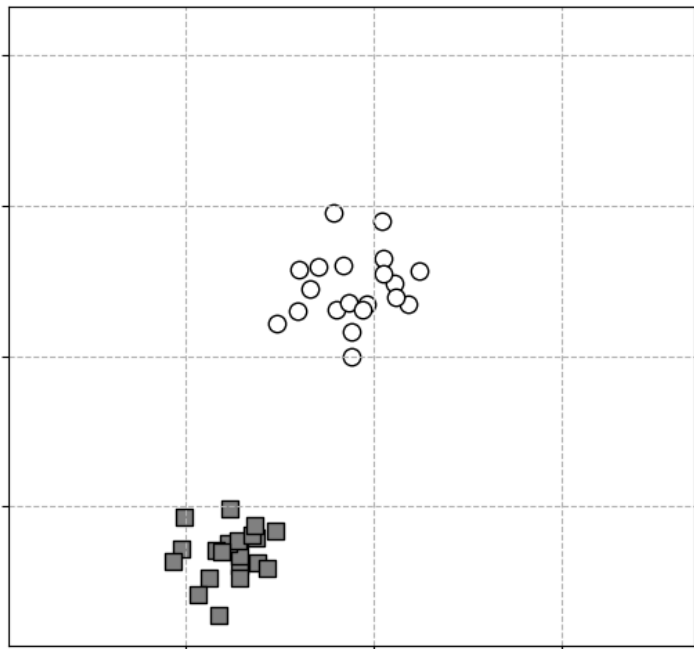


# Which is a good classifier?



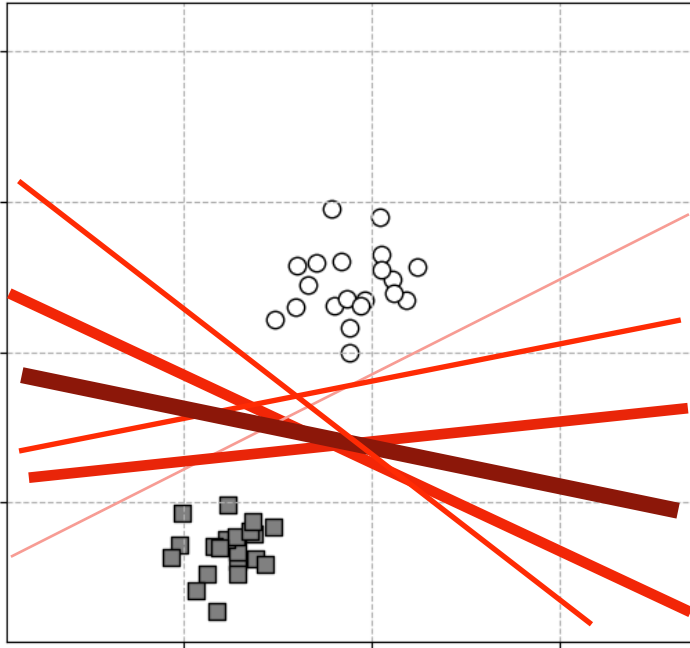
“What the model  
does not know”

# Sequential Bayesian Inference



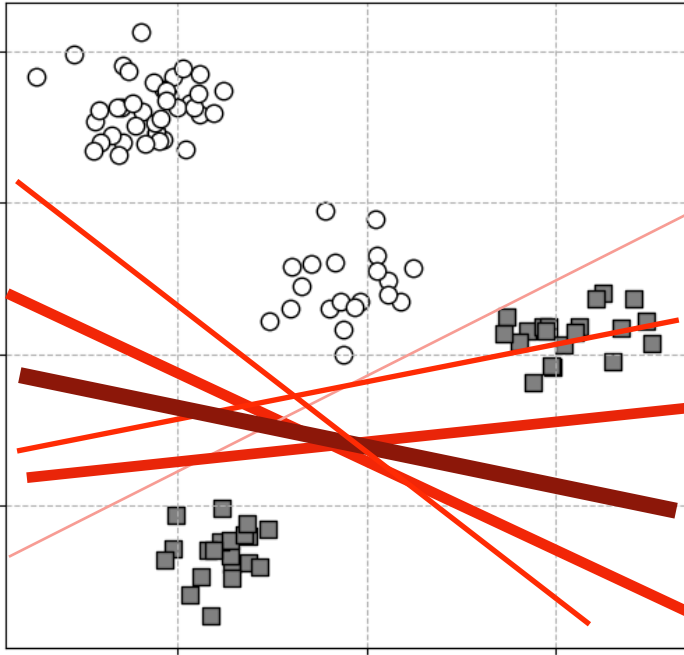
$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

# Sequential Bayesian Inference



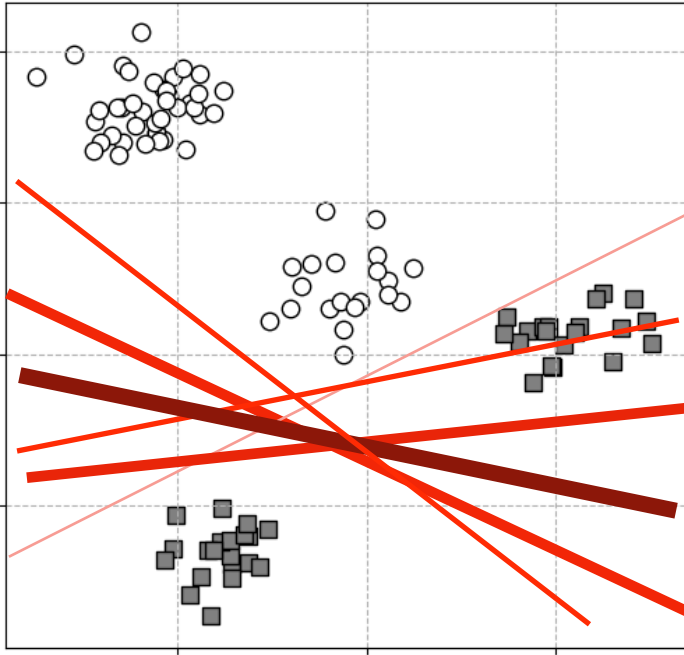
$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

# Sequential Bayesian Inference



$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

# Sequential Bayesian Inference

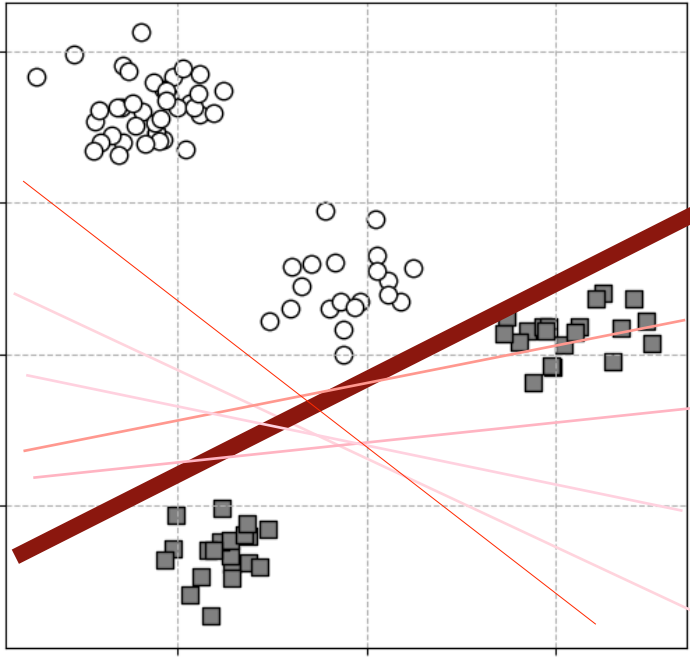


$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Set the prior to the previous posterior and recompute:

$$p(\theta|\mathcal{D}_2, \mathcal{D}_1) = \frac{p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)}{\int p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)d\theta}$$

# Sequential Bayesian Inference

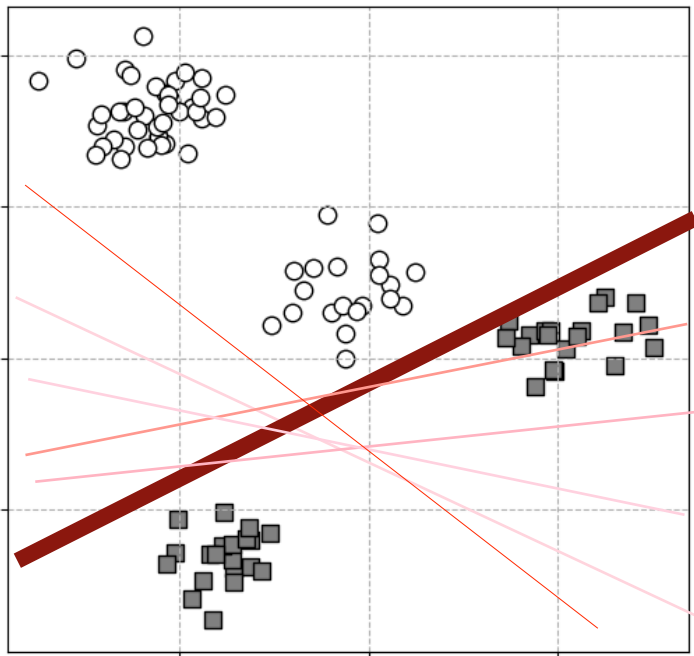


$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Set the prior to the previous posterior and recompute:

$$p(\theta|\mathcal{D}_2, \mathcal{D}_1) = \frac{p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)}{\int p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)d\theta}$$

# Sequential Bayesian Inference



$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Set the prior to the previous posterior and recompute:

$$p(\theta|\mathcal{D}_2, \mathcal{D}_1) = \frac{p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)}{\int p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)d\theta}$$

The global property enables sequential update

# Bayesian learning

Integration (global)

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

# Deep learning

Differentiation (local)

$$\theta \leftarrow \theta - \rho H_{\theta}^{-1} \nabla_{\theta} \ell(\theta)$$

	Bayes	DL
Can handle large data and complex models?	✗	✓
Scalable training?	✗	✓
Can estimate uncertainty?	✓	✗
Can perform sequential / active /online / incremental learning?	✓	✗



# Deep Learning with Bayesian Principles

- Bayesian principles as common principles
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning
- Impact: Many learning-algorithms with a common set of principles.

# Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them

Complex  Simple

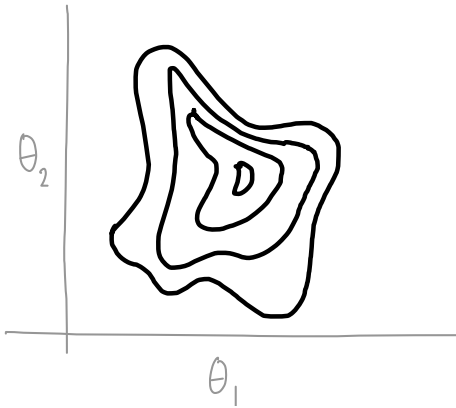
# Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them

Complex



Simple



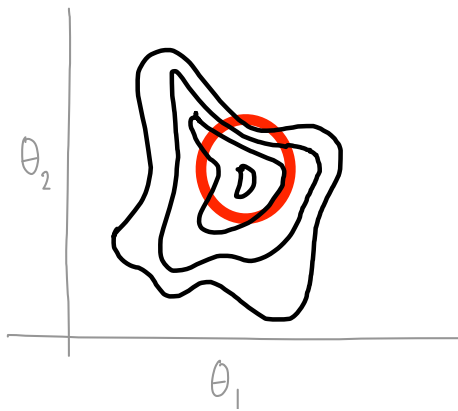
# Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them

Complex



Simple



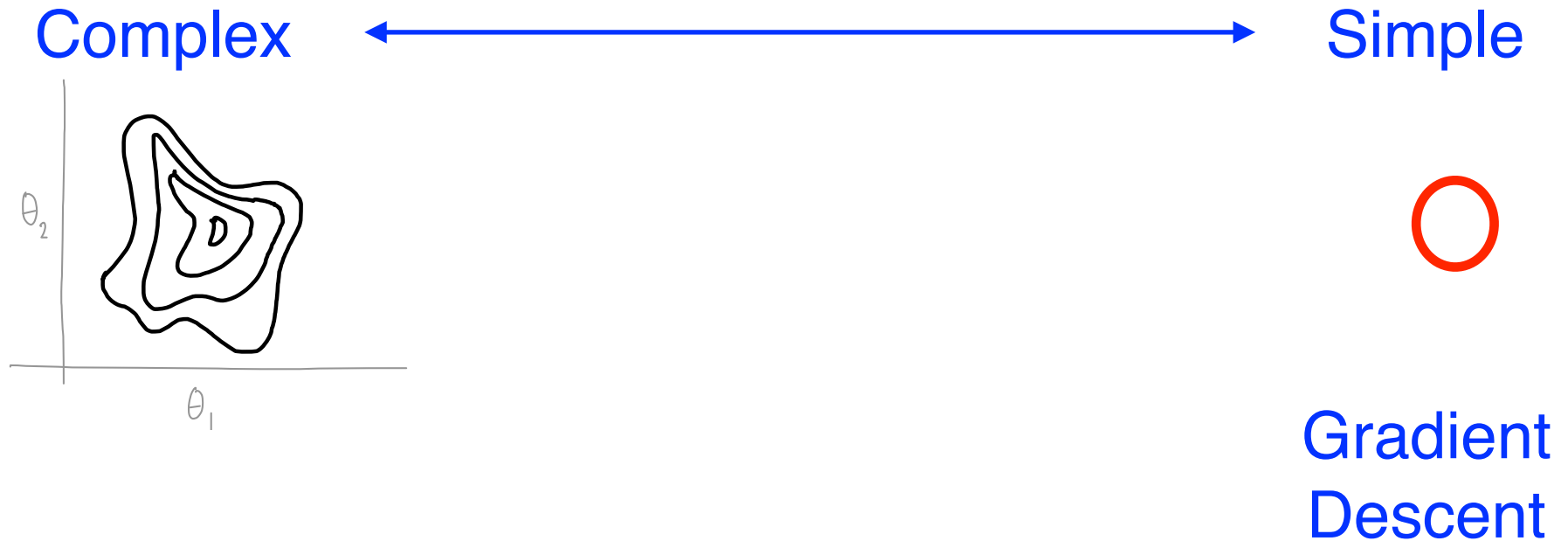
# Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them



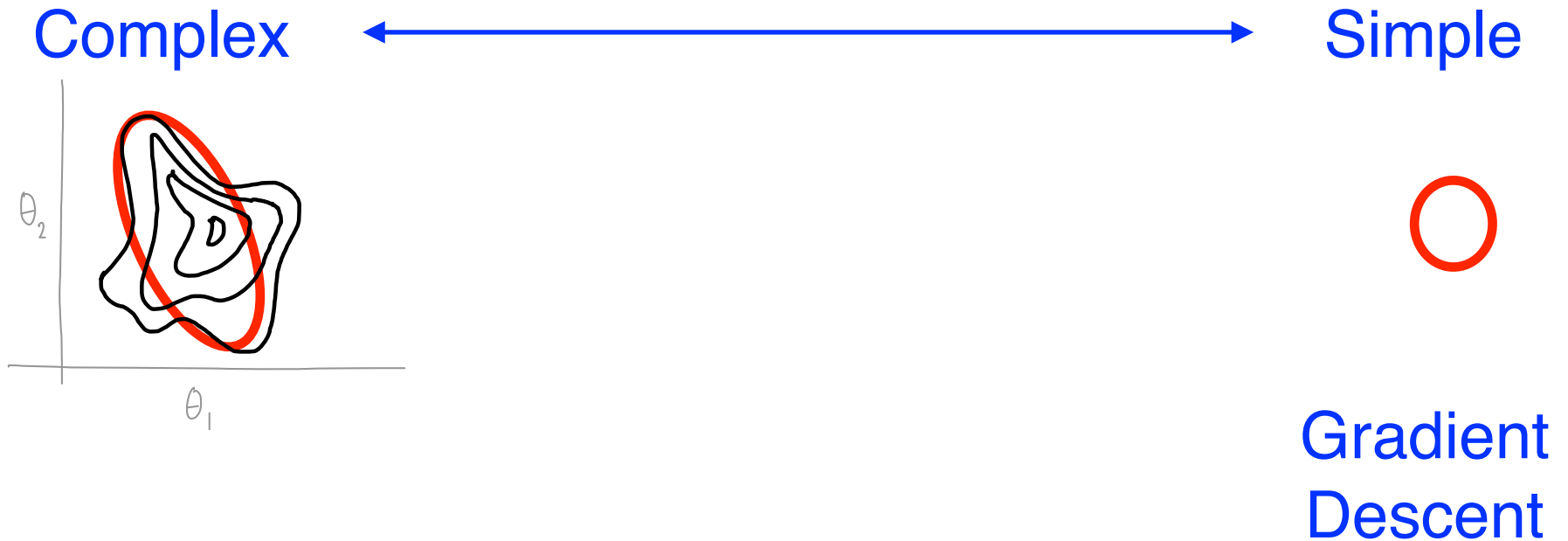
# Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them



# Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them



# Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them





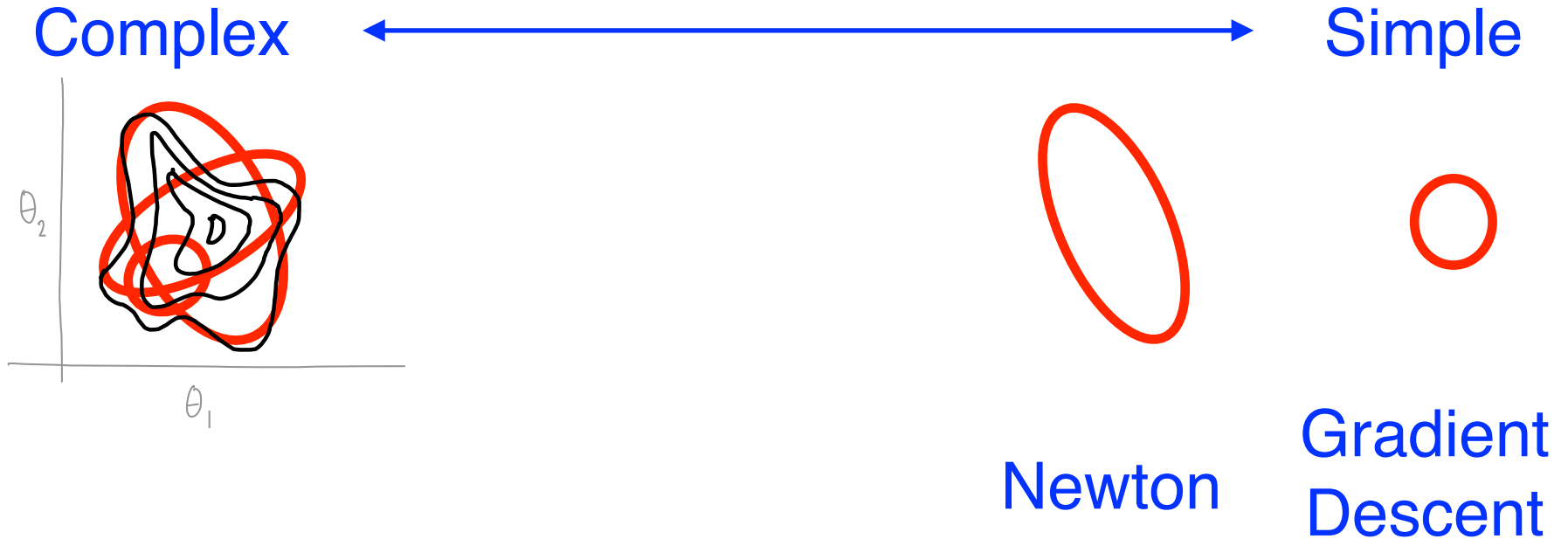
# Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them



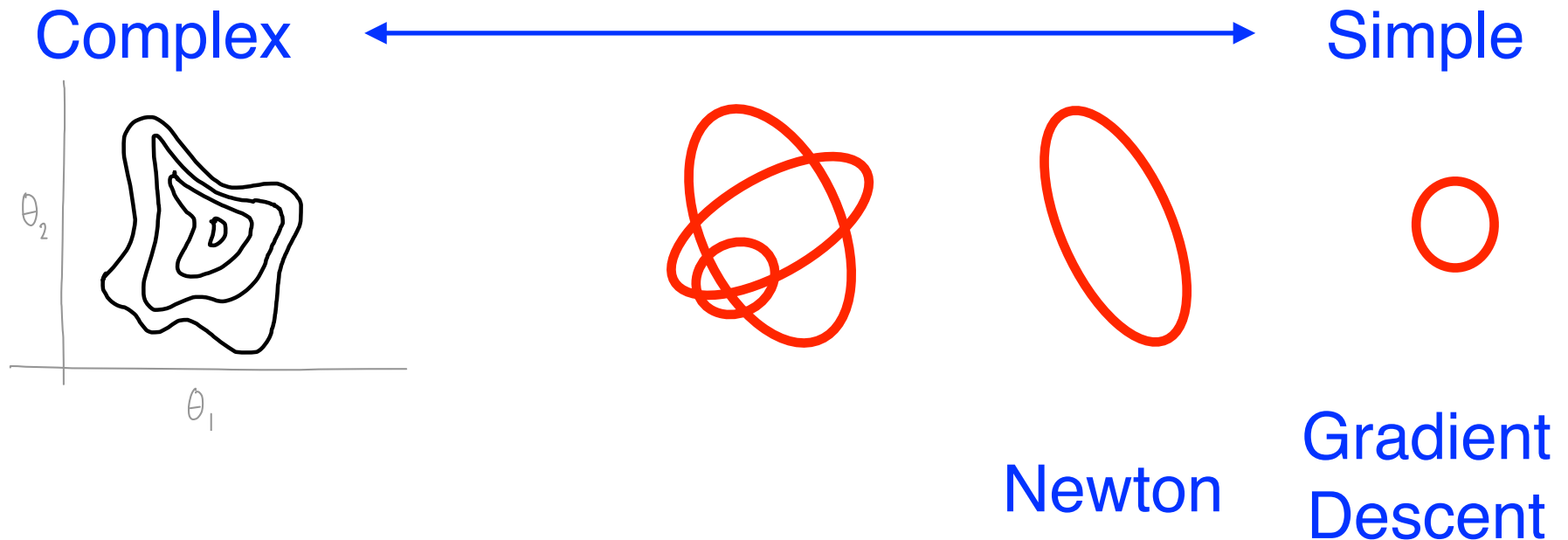
# Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them



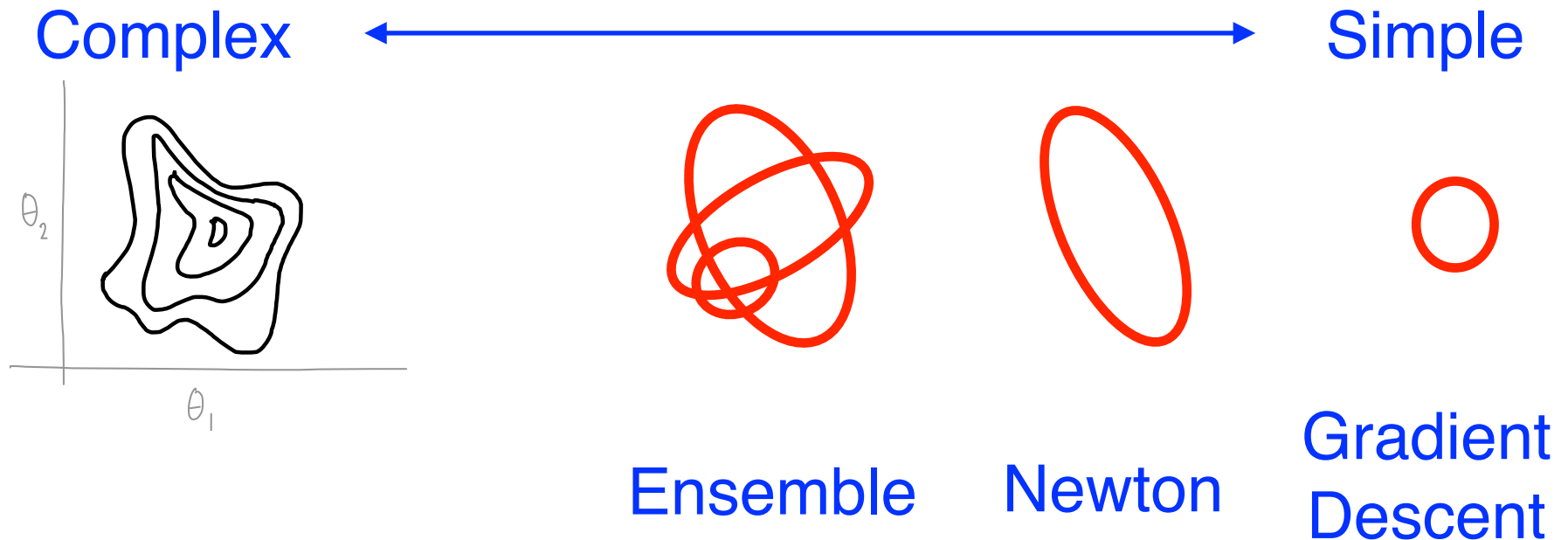
# Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them



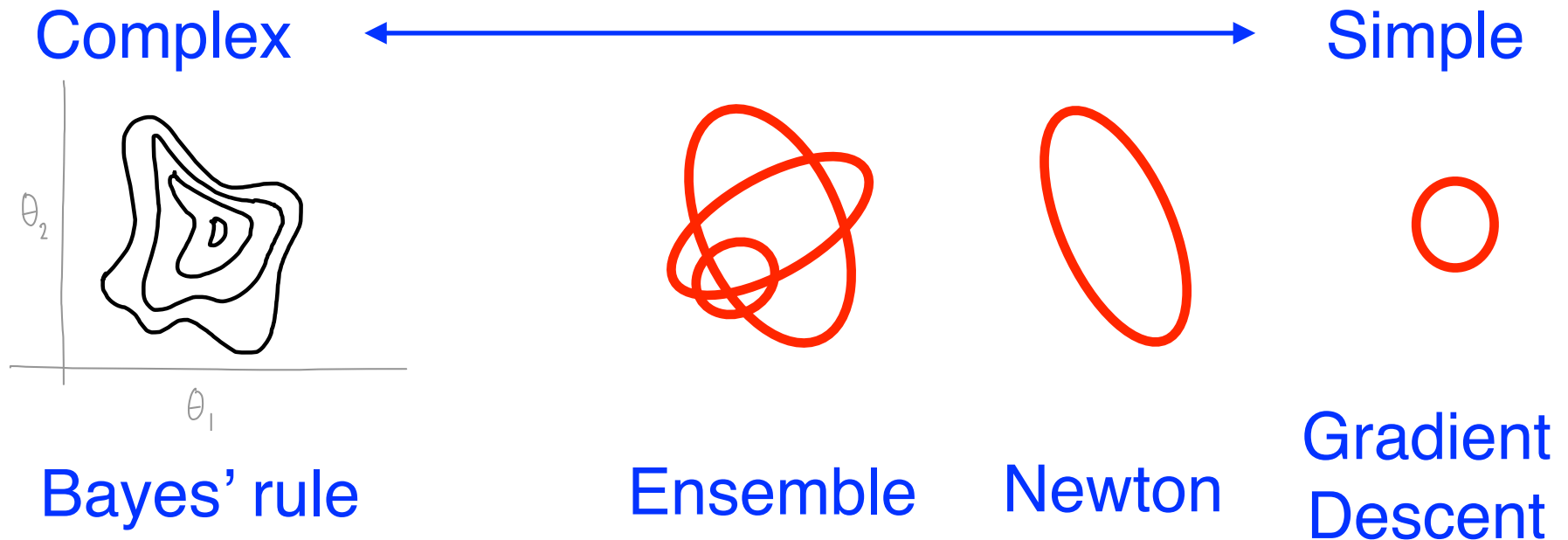
# Bayesian principles to derive Learning-Algorithms

Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them



# Bayesian principles to derive Learning-Algorithms

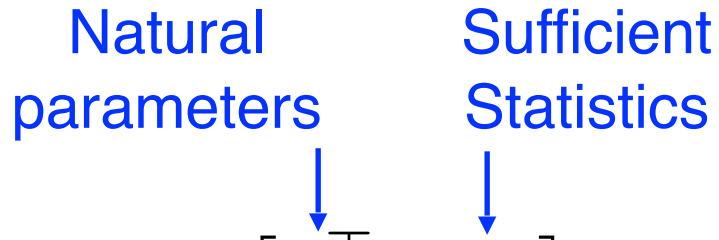
Main ideas: Introduce “posterior approximations” and the “Bayesian learning rule” to estimate them



# Exponential Family Approximations

Natural  
parameters

Sufficient  
Statistics


$$q(\theta) \propto \exp [\lambda^{\top} T(\theta)]$$

# Exponential Family Approximations

Natural  
parameters

Sufficient  
Statistics

$$q(\theta) \propto \exp \left[ \lambda^\top T(\theta) \right]$$

$$\mathcal{N}(\theta|m, S^{-1}) \propto \exp \left[ -\frac{1}{2}(\theta - m)^\top S(\theta - m) \right]$$

# Exponential Family Approximations

Natural  
parameters

Sufficient  
Statistics

$$q(\theta) \propto \exp \left[ \lambda^\top T(\theta) \right]$$

$$\begin{aligned} \mathcal{N}(\theta|m, S^{-1}) &\propto \exp \left[ -\frac{1}{2}(\theta - m)^\top S(\theta - m) \right] \\ &\propto \exp \left[ (Sm)^\top \theta + \text{Tr} \left( -\frac{S}{2} \theta \theta^\top \right) \right] \end{aligned}$$



# Exponential Family Approximations

Natural  
parameters

Sufficient  
Statistics

Expectation  
parameters

$$q(\theta) \propto \exp \left[ \lambda^\top T(\theta) \right]$$

$$\mu := \mathbb{E}_q[T(\theta)]$$

$$\begin{aligned} \mathcal{N}(\theta|m, S^{-1}) &\propto \exp \left[ -\frac{1}{2}(\theta - m)^\top S(\theta - m) \right] \\ &\propto \exp \left[ (Sm)^\top \theta + \text{Tr} \left( -\frac{S}{2} \theta \theta^\top \right) \right] \end{aligned}$$

# Exponential Family Approximations

Natural  
parameters

Sufficient  
Statistics

Expectation  
parameters

$$q(\theta) \propto \exp \left[ \lambda^\top T(\theta) \right]$$

$$\mu := \mathbb{E}_q[T(\theta)]$$

$$\begin{aligned} \mathcal{N}(\theta|m, S^{-1}) &\propto \exp \left[ -\frac{1}{2}(\theta - m)^\top S(\theta - m) \right] \\ &\propto \exp \left[ (Sm)^\top \theta + \text{Tr} \left( -\frac{S}{2} \theta \theta^\top \right) \right] \end{aligned}$$

Gaussian distribution

$$q(\theta) := \mathcal{N}(\theta|m, S^{-1})$$

Natural parameters

$$\lambda := \{Sm, -S/2\}$$

Expectation parameters

$$\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta \theta^\top)\}$$

# Bayesian Learning Rule

$$\min_{\theta} \ell(\theta) \quad \text{vs} \quad \min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \underbrace{\mathcal{H}(q)}_{\text{Entropy}}$$

# Bayesian Learning Rule

$$\min_{\theta} \ell(\theta) \quad \text{vs} \quad \min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \underbrace{\mathcal{H}(q)}_{\text{Entropy}}$$

Deep Learning algo:  $\theta \leftarrow \theta - \rho H_{\theta}^{-1} \nabla_{\theta} \ell(\theta)$

# Bayesian Learning Rule

$$\min_{\theta} \ell(\theta) \quad \text{vs} \quad \min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \underbrace{\mathcal{H}(q)}_{\text{Entropy}}$$

Deep Learning algo:  $\theta \leftarrow \theta - \rho H_{\theta}^{-1} \nabla_{\theta} \ell(\theta)$

Bayes learning rule:  $\lambda \leftarrow \lambda - \rho \nabla_{\mu} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$

$\uparrow$   $\uparrow$   
Natural and Expectation parameters of  
an exponential family distribution  $q$

# Bayesian Learning Rule

$$\min_{\theta} \ell(\theta) \quad \text{vs} \quad \min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

Entropy

Deep Learning algo:  $\theta \leftarrow \theta - \rho H_{\theta}^{-1} \nabla_{\theta} \ell(\theta)$

Bayes learning rule:  $\lambda \leftarrow \lambda - \rho \nabla_{\mu} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$

↑                      ↑  
Natural and Expectation parameters of  
an exponential family distribution  $q$

Deep Learning algorithms can be obtained by

1. Choosing an appropriate approximation  $q$ ,
2. Giving away the “global” property of the rule

# Deep Learning with Bayesian Principles

- Bayesian principles as common principles
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning
- Impact: Many learning-algorithms with a common set of principles.

# Gradient Descent from Bayes

Gradient descent:  $\theta \leftarrow \theta - \rho \nabla_{\theta} \ell(\theta)$



# Gradient Descent from Bayes

Gradient descent:  $\theta \leftarrow \theta - \rho \nabla_{\theta} \ell(\theta)$

Derived by choosing **Gaussian with fixed covariance**

Gaussian distribution  $q(\theta) := \mathcal{N}(m, 1)$

Natural parameters  $\lambda := m$

Expectation parameters  $\mu := \mathbb{E}_q[\theta] = m$

Entropy  $\mathcal{H}(q) := \log(2\pi)/2$

# Gradient Descent from Bayes

Gradient descent:  $\theta \leftarrow \theta - \rho \nabla_{\theta} \ell(\theta)$

$$\lambda \leftarrow \lambda - \rho \nabla_{\mu} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$$

Derived by choosing **Gaussian with fixed covariance**

Gaussian distribution  $q(\theta) := \mathcal{N}(m, 1)$

Natural parameters  $\lambda := m$

Expectation parameters  $\mu := \mathbb{E}_q[\theta] = m$

Entropy  $\mathcal{H}(q) := \log(2\pi)/2$

# Gradient Descent from Bayes

Gradient descent:  $\theta \leftarrow \theta - \rho \nabla_{\theta} \ell(\theta)$

$$m \leftarrow m - \rho \nabla_{\textcolor{red}{m}} \mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow \lambda - \rho \nabla_{\textcolor{red}{\mu}} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$$

Derived by choosing **Gaussian with fixed covariance**

Gaussian distribution  $q(\theta) := \mathcal{N}(m, 1)$

Natural parameters  $\lambda := m$

Expectation parameters  $\mu := \mathbb{E}_q[\theta] = m$

Entropy  $\mathcal{H}(q) := \log(2\pi)/2$

# Gradient Descent from Bayes

Gradient descent:  $\theta \leftarrow \theta - \rho \nabla_{\theta} \ell(\theta)$

“Global” to “local”

$$\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$$

$$m \leftarrow m - \rho \nabla_{\textcolor{red}{m}} \mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow \lambda - \rho \nabla_{\textcolor{red}{\mu}} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$$

Derived by choosing **Gaussian with fixed covariance**

Gaussian distribution  $q(\theta) := \mathcal{N}(m, 1)$

Natural parameters  $\lambda := m$

Expectation parameters  $\mu := \mathbb{E}_q[\theta] = m$

Entropy  $\mathcal{H}(q) := \log(2\pi)/2$

# Gradient Descent from Bayes

Gradient descent:  $\theta \leftarrow \theta - \rho \nabla_{\theta} \ell(\theta)$

Bayes Learn Rule:  $m \leftarrow m - \rho \nabla_m \ell(m)$

“Global” to “local”

$$\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$$

$$m \leftarrow m - \rho \nabla_{\textcolor{red}{m}} \mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow \lambda - \rho \nabla_{\textcolor{red}{\mu}} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$$

Derived by choosing **Gaussian with fixed covariance**

Gaussian distribution  $q(\theta) := \mathcal{N}(m, 1)$

Natural parameters  $\lambda := m$

Expectation parameters  $\mu := \mathbb{E}_q[\theta] = m$

Entropy  $\mathcal{H}(q) := \log(2\pi)/2$

# Gradient Descent from Bayes

Gradient descent:  $\theta \leftarrow \theta - \rho \nabla_{\theta} \ell(\theta)$

Bayes Learn Rule:  $m \leftarrow m - \rho \nabla_m \ell(m)$

“Global” to “local”

$$\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$$

$$m \leftarrow m - \rho \nabla_{\textcolor{red}{m}} \mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow \lambda - \rho \nabla_{\textcolor{red}{\mu}} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$$

Derived by choosing **Gaussian with fixed covariance**

Gaussian distribution  $q(\theta) := \mathcal{N}(m, 1)$

Natural parameters  $\lambda := m$

Expectation parameters  $\mu := \mathbb{E}_q[\theta] = m$

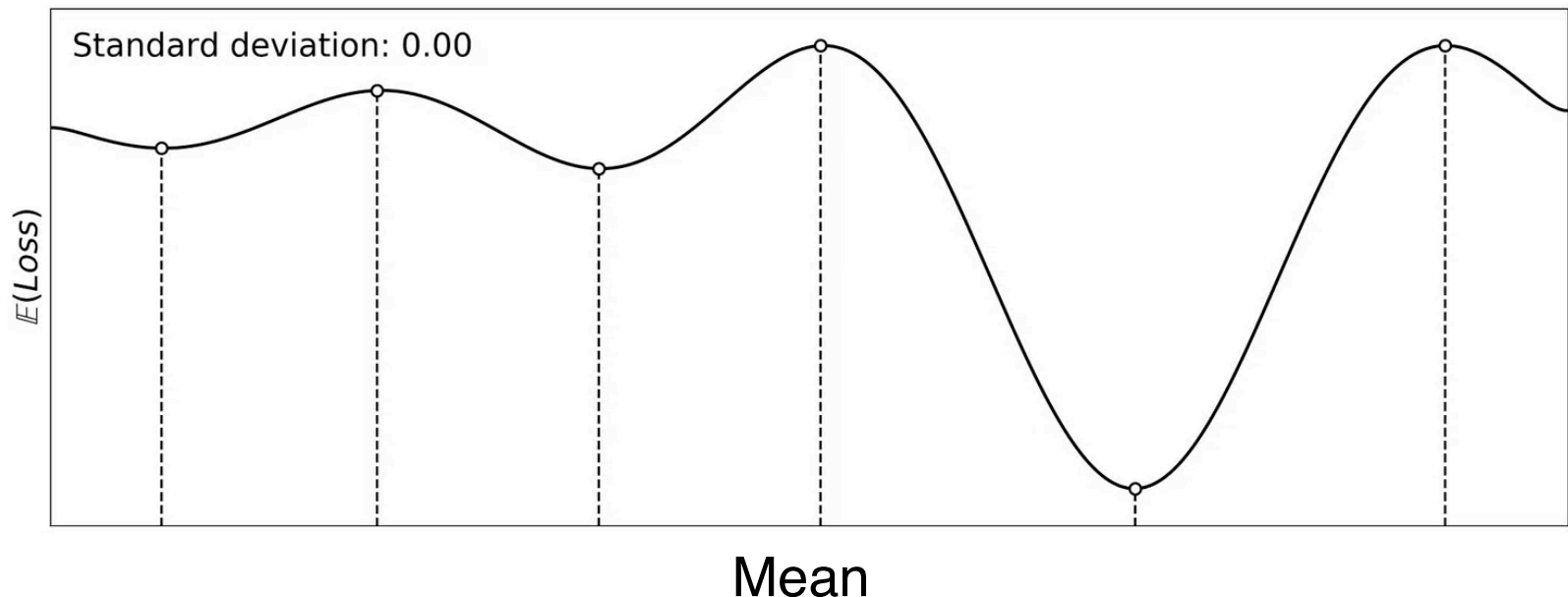
Entropy  $\mathcal{H}(q) := \log(2\pi)/2$

Using  
stochastic  
gradients,  
we get SGD

# From “Global” to “Local”

$$\mathcal{L}(m, \sigma) := \mathbb{E}_q[\ell(\theta)]$$

We can interpolate between “Global” to “local” information by changing the variance,

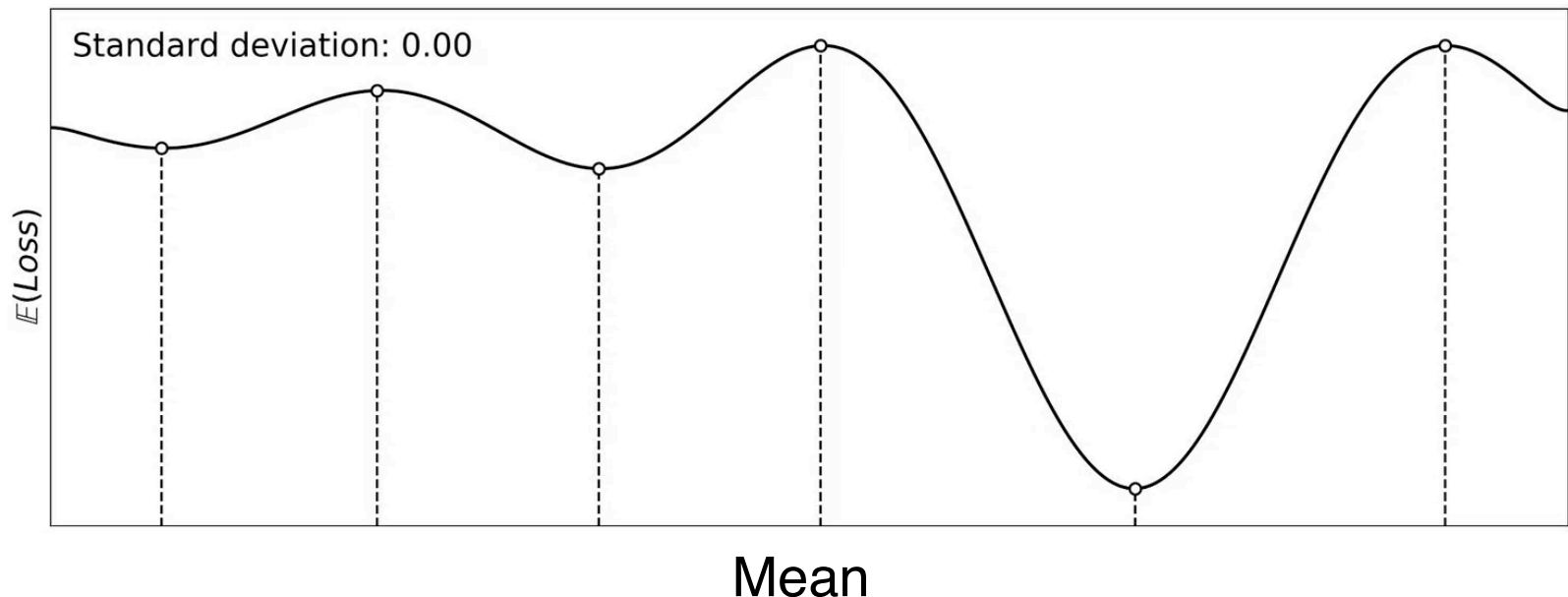


1. Huszar’s blog, [Evolution Strategies, Variational Optimisation and Natural ES \(2017\)](#)
2. Khan et al. "Variational adaptive-Newton method for explorative learning." *arXiv* (2017).

# From “Global” to “Local”

$$\mathcal{L}(m, \sigma) := \mathbb{E}_q[\ell(\theta)]$$

We can interpolate between “Global” to “local” information by changing the variance,



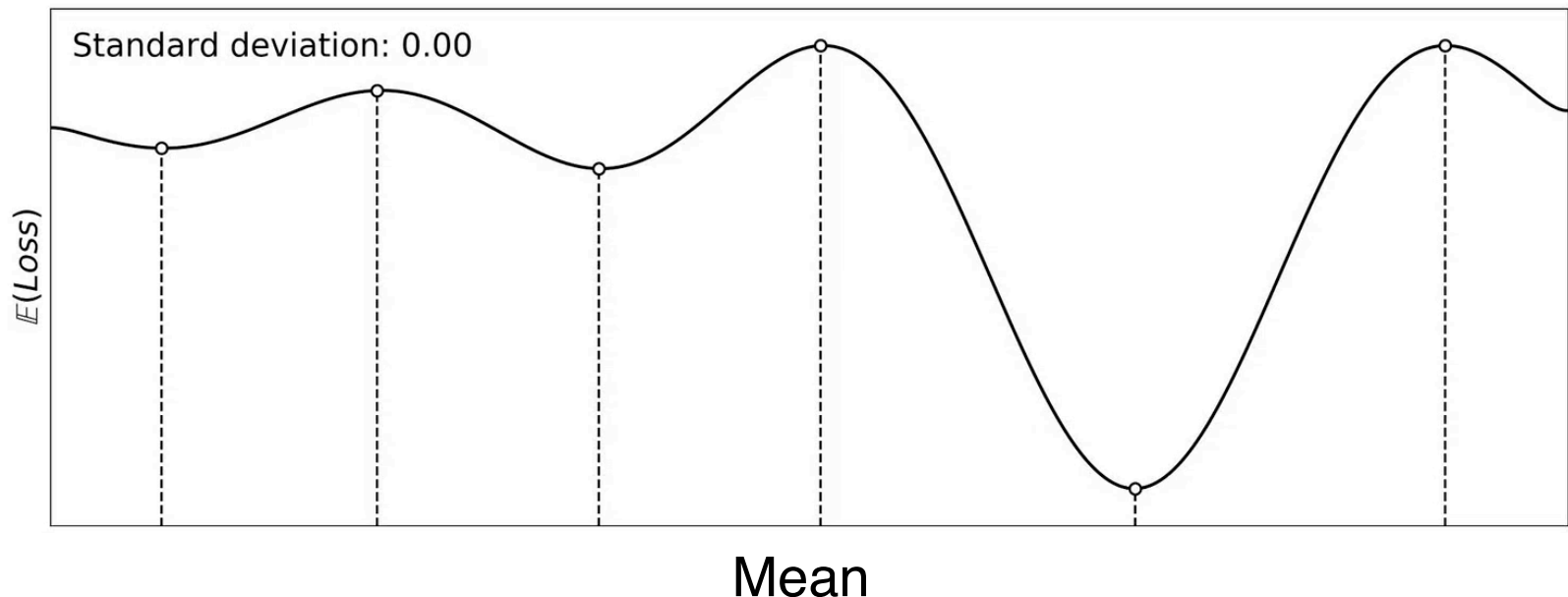
1. Huszar’s blog, [Evolution Strategies, Variational Optimisation and Natural ES \(2017\)](#)
2. Khan et al. "Variational adaptive-Newton method for explorative learning." *arXiv* (2017).



# From “Global” to “Local”

$$\mathcal{L}(m, \sigma) := \mathbb{E}_q[\ell(\theta)]$$

We can interpolate between “Global” to “local” information by changing the variance,

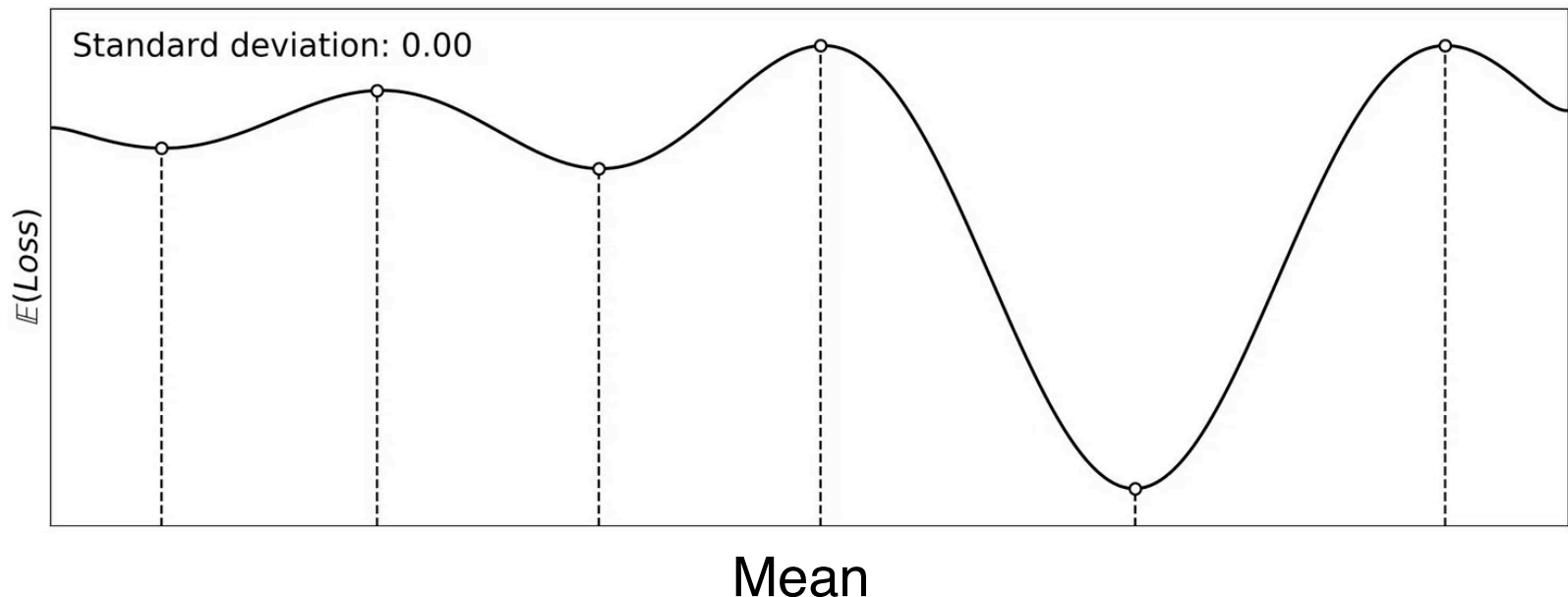


1. Huszar’s blog, [Evolution Strategies, Variational Optimisation and Natural ES \(2017\)](#)
2. Khan et al. "Variational adaptive-Newton method for explorative learning." *arXiv* (2017).

# From “Global” to “Local”

$$\mathcal{L}(m, \sigma) := \mathbb{E}_q[\ell(\theta)] \approx \ell(m)$$

We can interpolate between “Global” to “local” information by changing the variance,



1. Huszar’s blog, [Evolution Strategies, Variational Optimisation and Natural ES \(2017\)](#)
2. Khan et al. "Variational adaptive-Newton method for explorative learning." *arXiv* (2017).

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

Derived by choosing a **multivariate Gaussian**

Gaussian distribution  $q(\theta) := \mathcal{N}(\theta|m, S^{-1})$

Natural parameters  $\lambda := \{Sm, -S/2\}$

Expectation parameters  $\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^{\top})\}$

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

$$\lambda \leftarrow \lambda - \rho \nabla_{\mu} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$$

Derived by choosing a **multivariate Gaussian**

Gaussian distribution  $q(\theta) := \mathcal{N}(\theta|m, S^{-1})$

Natural parameters  $\lambda := \{Sm, -S/2\}$

Expectation parameters  $\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^{\top})\}$

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

$$\lambda \leftarrow \lambda - \rho \nabla_{\mu} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q)) \quad \boxed{-\nabla_{\mu} \mathcal{H}(q) = \lambda}$$

Derived by choosing a **multivariate Gaussian**

Gaussian distribution  $q(\theta) := \mathcal{N}(\theta|m, S^{-1})$

Natural parameters  $\lambda := \{Sm, -S/2\}$

Expectation parameters  $\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^{\top})\}$

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

$$\lambda \leftarrow \lambda - \rho (\nabla_{\mu} \mathbb{E}_q[\ell(\theta)] + \lambda)$$

$$-\nabla_{\mu} \mathcal{H}(q) = \lambda$$

Derived by choosing a **multivariate Gaussian**

Gaussian distribution  $q(\theta) := \mathcal{N}(\theta|m, S^{-1})$

Natural parameters  $\lambda := \{Sm, -S/2\}$

Expectation parameters  $\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^{\top})\}$

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

$$\lambda \leftarrow (1 - \rho)\lambda - \rho \nabla_{\mu} \mathbb{E}_q[\ell(\theta)]$$

$$-\nabla_{\mu} \mathcal{H}(q) = \lambda$$

Derived by choosing a **multivariate Gaussian**

Gaussian distribution  $q(\theta) := \mathcal{N}(\theta|m, S^{-1})$

Natural parameters  $\lambda := \{Sm, -S/2\}$

Expectation parameters  $\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^{\top})\}$



# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

$$Sm \leftarrow (1 - \rho)Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow (1 - \rho)\lambda - \rho \nabla_{\mu} \mathbb{E}_q[\ell(\theta)]$$

$$-\nabla_{\mu} \mathcal{H}(q) = \lambda$$

Derived by choosing a **multivariate Gaussian**

Gaussian distribution  $q(\theta) := \mathcal{N}(\theta|m, S^{-1})$

Natural parameters  $\lambda := \{Sm, -S/2\}$

Expectation parameters  $\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^{\top})\}$

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

$$\begin{aligned} Sm &\leftarrow (1 - \rho)Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] \\ -\frac{1}{2}S &\leftarrow -(1 - \rho)\frac{1}{2}S + \rho \nabla_{\mathbb{E}_q(\theta\theta^{\top})} \mathbb{E}_q[\ell(\theta)] \end{aligned}$$

$$\lambda \leftarrow (1 - \rho)\lambda - \rho \nabla_{\mu} \mathbb{E}_q[\ell(\theta)]$$

$$-\nabla_{\mu} \mathcal{H}(q) = \lambda$$

Derived by choosing a **multivariate Gaussian**

Gaussian distribution  $q(\theta) := \mathcal{N}(\theta|m, S^{-1})$

Natural parameters  $\lambda := \{Sm, -S/2\}$

Expectation parameters  $\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^{\top})\}$

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

$$Sm \leftarrow (1 - \rho)Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)]$$

$$S \leftarrow (1 - \rho)S - \rho 2 \nabla_{\mathbb{E}_q(\theta\theta^{\top})} \mathbb{E}_q[\ell(\theta)]$$

$$\lambda \leftarrow (1 - \rho)\lambda - \rho \nabla_{\mu} \mathbb{E}_q[\ell(\theta)]$$

$$-\nabla_{\mu} \mathcal{H}(q) = \lambda$$

Derived by choosing a **multivariate Gaussian**

Gaussian distribution  $q(\theta) := \mathcal{N}(\theta|m, S^{-1})$

Natural parameters  $\lambda := \{Sm, -S/2\}$

Expectation parameters  $\mu := \{\mathbb{E}_q(\theta), \mathbb{E}_q(\theta\theta^{\top})\}$

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

$$\begin{aligned} S_m &\leftarrow (1 - \rho) S_m - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] \\ S &\leftarrow (1 - \rho) S - \rho 2 \nabla_{\mathbb{E}_q(\theta \theta^{\top})} \mathbb{E}_q[\ell(\theta)] \end{aligned}$$

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

Express in terms of gradient and Hessian of loss:

$$\nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[\nabla_{\theta} \ell(\theta)] - 2\mathbb{E}_q[H_{\theta}]m$$

$$\nabla_{\mathbb{E}_q(\theta\theta^{\top})} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[H_{\theta}]$$

$$\begin{aligned} Sm &\leftarrow (1 - \rho)Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] \\ S &\leftarrow (1 - \rho)S - \rho 2 \nabla_{\mathbb{E}_q(\theta\theta^{\top})} \mathbb{E}_q[\ell(\theta)] \end{aligned}$$

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

“Global” to “local”  
 $\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$

Express in terms of gradient and Hessian of loss:

$$\nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[\nabla_{\theta} \ell(\theta)] - 2\mathbb{E}_q[H_{\theta}]m$$

$$\nabla_{\mathbb{E}_q(\theta\theta^{\top})} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[H_{\theta}]$$

$$\begin{aligned} Sm &\leftarrow (1 - \rho)Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] \\ S &\leftarrow (1 - \rho)S - \rho 2 \nabla_{\mathbb{E}_q(\theta\theta^{\top})} \mathbb{E}_q[\ell(\theta)] \end{aligned}$$

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

$$\begin{aligned} m &\leftarrow m - \rho \mathbf{S}^{-1} \nabla_m \ell(m) \\ S &\leftarrow (1 - \rho)S + \rho \mathbf{H}_m \end{aligned}$$

“Global” to “local”  
 $\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$

Express in terms of gradient and Hessian of loss:

$$\nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[\nabla_{\theta} \ell(\theta)] - 2\mathbb{E}_q[\mathbf{H}_{\theta}]m$$

$$\nabla_{\mathbb{E}_q(\theta\theta^{\top})} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[\mathbf{H}_{\theta}]$$

$$\begin{aligned} Sm &\leftarrow (1 - \rho)Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] \\ S &\leftarrow (1 - \rho)S - \rho 2 \nabla_{\mathbb{E}_q(\theta\theta^{\top})} \mathbb{E}_q[\ell(\theta)] \end{aligned}$$

# Newton's Method from Bayes

Newton's method:  $\theta \leftarrow \theta - H_{\theta}^{-1} [\nabla_{\theta} \ell(\theta)]$

Set  $\rho=1$  to get  $m \leftarrow m - H_m^{-1} [\nabla_m \ell(m)]$

$$m \leftarrow m - \rho \mathbf{S}^{-1} \nabla_m \ell(m)$$

$$\mathbf{S} \leftarrow (1 - \rho) \mathbf{S} + \rho \mathbf{H}_m$$

“Global” to “local”

$$\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$$

Express in terms of gradient and Hessian of loss:

$$\nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[\nabla_{\theta} \ell(\theta)] - 2\mathbb{E}_q[\mathbf{H}_{\theta}]m$$

$$\nabla_{\mathbb{E}_q(\theta\theta^{\top})} \mathbb{E}_q[\ell(\theta)] = \mathbb{E}_q[\mathbf{H}_{\theta}]$$

$$Sm \leftarrow (1 - \rho)Sm - \rho \nabla_{\mathbb{E}_q(\theta)} \mathbb{E}_q[\ell(\theta)]$$

$$\mathbf{S} \leftarrow (1 - \rho)\mathbf{S} - \rho 2 \nabla_{\mathbb{E}_q(\theta\theta^{\top})} \mathbb{E}_q[\ell(\theta)]$$



# RMSprop/Adam from Bayes

## RMSprop

$$\begin{aligned}s &\leftarrow (1 - \rho)s + \rho[\hat{\nabla}\ell(\theta)]^2 \\ \theta &\leftarrow \theta - \alpha(\sqrt{s} + \delta)^{-1}\hat{\nabla}\ell(\theta)\end{aligned}$$

## Bayesian Learning rule for multivariate Gaussian

$$\begin{aligned}S &\leftarrow (1 - \rho)S + \rho(\textcolor{red}{H}_{\theta}) \\ m &\leftarrow m - \alpha\textcolor{red}{S}^{-1}\textcolor{red}{\nabla}_{\theta}\ell(\theta)\end{aligned}$$

# RMSprop/Adam from Bayes

Bayesian Learning rule for multivariate Gaussian

RMSprop

$$\begin{aligned}s &\leftarrow (1 - \rho)s + \rho[\hat{\nabla}\ell(\theta)]^2 \\ \theta &\leftarrow \theta - \alpha(\sqrt{s} + \delta)^{-1}\hat{\nabla}\ell(\theta)\end{aligned}$$

$$\begin{aligned}S &\leftarrow (1 - \rho)S + \rho(H_{\theta}) \\ m &\leftarrow m - \alpha S^{-1}\nabla_{\theta}\ell(\theta)\end{aligned}$$

To get RMSprop, make the following choices

- Choose Gaussian with diagonal covariance
- Replace Hessian by square of gradients
- Add square root for scaling vector

# RMSprop/Adam from Bayes

Bayesian Learning rule for multivariate Gaussian

RMSprop

$$\begin{aligned}s &\leftarrow (1 - \rho)s + \rho[\hat{\nabla}\ell(\theta)]^2 \\ \theta &\leftarrow \theta - \alpha(\sqrt{s} + \delta)^{-1}\hat{\nabla}\ell(\theta)\end{aligned}$$

$$\begin{aligned}S &\leftarrow (1 - \rho)S + \rho(\mathbf{H}_{\theta}) \\ m &\leftarrow m - \alpha \mathbf{S}^{-1} \nabla_{\theta} \ell(\theta)\end{aligned}$$

To get RMSprop, make the following choices

- Choose Gaussian with diagonal covariance
- Replace Hessian by square of gradients
- Add square root for scaling vector

For Adam, use a Heavy-ball term with KL divergence as the momentum (Appendix E in [1])

# Summary

- Gradient descent is derived using a Gaussian with fixed covariance, and estimating the mean
- Newton's method is derived using multivariate Gaussian
- RMSprop is derived using diagonal covariance
- Adam is derived by adding heavy-ball momentum term
- For “ensemble of Newton”, use Mixture of Gaussians [1]
- STE is derived using Bernoulli distribution for Binary NN [2]
- To derive DL algorithms, we need to switch from a “global” to “local” approximation  $\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$
- Then, to improve DL algorithms, we just need to add some “global” touch to the DL algorithms

1. Linu, Khan, Schmidt. "Fast and Simple Natural-Gradient Variational Inference with Mixture of Exponential-family Approximations." *ICML* (2019).

2. Meng, Bachman, Khan, Training Binary Neural Networks using the Bayesian Learning Rule *ICML* (2019).

# Deep Learning with Bayesian Principles

- Bayesian principles as common principles
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning
- Impact: Many learning-algorithms with a common set of principles.

# Bayes as Optimization

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

# Bayes as Optimization

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

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)$$

# Bayes as Optimization

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

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)$$

$$= \arg \min_{q \in \mathcal{P}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distribution      Distribution      Entropy



# Bayes as Optimization

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

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)$$

$$= \arg \min_{q \in \mathcal{P}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distributionDistributionEntropy

$$= \mathbb{E}_q[\ell(\theta)] + \mathbb{E}_q[\log q(\theta)]$$

# Bayes as Optimization

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

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)$$

$$= \arg \min_{\substack{q \in \mathcal{P} \\ \uparrow \\ \text{All distribution}}} \mathbb{E}_{\substack{q(\theta) \\ \uparrow \\ \text{Distribution}}} [\ell(\theta)] - \underbrace{\mathcal{H}(q)}_{\text{Entropy}}$$

$$= \mathbb{E}_q[\ell(\theta)] + \mathbb{E}_q[\log q(\theta)] = \mathbb{E}_q \left[ \log \frac{q(\theta)}{e^{-\ell(\theta)}} \right]$$

# Bayes as Optimization

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

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)$$

$$= \arg \min_{\substack{q \in \mathcal{P} \\ \uparrow \\ \text{All distribution}}} \mathbb{E}_{\substack{q(\theta) \\ \uparrow \\ \text{Distribution}}} [\ell(\theta)] - \underbrace{\mathcal{H}(q)}_{\text{Entropy}}$$

$$\begin{aligned} &= \mathbb{E}_q[\ell(\theta)] + \mathbb{E}_q[\log q(\theta)] &= \mathbb{E}_q \left[ \log \frac{q(\theta)}{e^{-\ell(\theta)}} \right] \\ \implies q_*(\theta) &\propto e^{-\ell(\theta)} \end{aligned}$$

# Bayes as Optimization

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

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)$$

$$= \arg \min_{\substack{q \in \mathcal{P} \\ \uparrow \\ \text{All distribution}}} \mathbb{E}_{\substack{q(\theta) \\ \uparrow \\ \text{Distribution}}} [\ell(\theta)] - \underbrace{\mathcal{H}(q)}_{\text{Entropy}}$$

$$\begin{aligned} &= \mathbb{E}_q[\ell(\theta)] + \mathbb{E}_q[\log q(\theta)] = \mathbb{E}_q \left[ \log \frac{q(\theta)}{e^{-\ell(\theta)}} \right] \\ \implies q_*(\theta) &\propto e^{-\ell(\theta)} \propto p(\mathcal{D}|\theta)p(\theta) \end{aligned}$$

# Bayes as Optimization

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

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)$$

$$= \arg \min_{q \in \mathcal{P}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distribution      Distribution      Entropy

$$= \mathbb{E}_q[\ell(\theta)] + \mathbb{E}_q[\log q(\theta)] = \mathbb{E}_q \left[ \log \frac{q(\theta)}{e^{-\ell(\theta)}} \right]$$
$$\implies q_*(\theta) \propto e^{-\ell(\theta)} \propto p(\mathcal{D}|\theta)p(\theta) \propto p(\theta|\mathcal{D})$$

# Bayes as Optimization

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

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)$$

$$= \arg \min_{q \in \mathcal{P}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

All distribution      Distribution      Entropy

$$= \mathbb{E}_q[\ell(\theta)] + \mathbb{E}_q[\log q(\theta)] = \mathbb{E}_q \left[ \log \frac{q(\theta)}{e^{-\ell(\theta)}} \right]$$
$$\implies q_*(\theta) \propto e^{-\ell(\theta)} \propto p(\mathcal{D}|\theta)p(\theta) \propto p(\theta|\mathcal{D})$$

Good news: This holds for a generic loss function!

# References for Bayes as Optimization

$$\arg \min_{q \in \mathcal{P}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

# References for Bayes as Optimization

$$\arg \min_{q \in \mathcal{P}} \mathbb{E}_{q(\theta)} [\ell(\theta)] - \mathcal{H}(q)$$

- Bayesian statistics

1. Jaynes, Edwin T. "Information theory and statistical mechanics." *Physical review* (1957)
2. Zellner, A. "Optimal information processing and Bayes's theorem." *The American Statistician* (1988)
3. Bissiri, Pier Giovanni, Chris C. Holmes, and Stephen G. Walker. "A general framework for updating belief distributions." *RSS: Series B (Statistical Methodology)* (2016)



# References for Bayes as Optimization

$$\arg \min_{q \in \mathcal{P}} \mathbb{E}_{q(\theta)} [\ell(\theta)] - \mathcal{H}(q)$$

- Bayesian statistics

1. Jaynes, Edwin T. "Information theory and statistical mechanics." *Physical review* (1957)
2. Zellner, A. "Optimal information processing and Bayes's theorem." *The American Statistician* (1988)
3. Bissiri, Pier Giovanni, Chris C. Holmes, and Stephen G. Walker. "A general framework for updating belief distributions." *RSS: Series B (Statistical Methodology)* (2016)

- PAC-Bayes

4. Shawe-Taylor, John, and Robert C. Williamson. "A PAC analysis of a Bayesian estimator." COLT 1997.
5. Alquier, Pierre. "PAC-Bayesian bounds for randomized empirical risk minimizers." *Mathematical Methods of Statistics* 17.4 (2008): 279-304.

# References for Bayes as Optimization

$$\arg \min_{q \in \mathcal{P}} \mathbb{E}_{q(\theta)} [\ell(\theta)] - \mathcal{H}(q)$$

- Bayesian statistics

1. Jaynes, Edwin T. "Information theory and statistical mechanics." *Physical review* (1957)
2. Zellner, A. "Optimal information processing and Bayes's theorem." *The American Statistician* (1988)
3. Bissiri, Pier Giovanni, Chris C. Holmes, and Stephen G. Walker. "A general framework for updating belief distributions." *RSS: Series B (Statistical Methodology)* (2016)

- PAC-Bayes

4. Shawe-Taylor, John, and Robert C. Williamson. "A PAC analysis of a Bayesian estimator." COLT 1997.
5. Alquier, Pierre. "PAC-Bayesian bounds for randomized empirical risk minimizers." *Mathematical Methods of Statistics* 17.4 (2008): 279-304.

- Online-learning (Exponential Weight Aggregate)

6. Cesa-Bianchi, Nicolo, and Gabor Lugosi. *Prediction, learning, and games*. 2006.

# References for Bayes as Optimization

$$\arg \min_{q \in \mathcal{P}} \mathbb{E}_{q(\theta)} [\ell(\theta)] - \mathcal{H}(q)$$

- Bayesian statistics

1. Jaynes, Edwin T. "Information theory and statistical mechanics." *Physical review* (1957)
2. Zellner, A. "Optimal information processing and Bayes's theorem." *The American Statistician* (1988)
3. Bissiri, Pier Giovanni, Chris C. Holmes, and Stephen G. Walker. "A general framework for updating belief distributions." *RSS: Series B (Statistical Methodology)* (2016)

- PAC-Bayes

4. Shawe-Taylor, John, and Robert C. Williamson. "A PAC analysis of a Bayesian estimator." COLT 1997.
5. Alquier, Pierre. "PAC-Bayesian bounds for randomized empirical risk minimizers." *Mathematical Methods of Statistics* 17.4 (2008): 279-304.

- Online-learning (Exponential Weight Aggregate)

6. Cesa-Bianchi, Nicolo, and Gabor Lugosi. *Prediction, learning, and games*. 2006.

- Free-energy principle

7. Friston, K. "The free-energy principle: a unified brain theory?." *Nature neuroscience* (2010)

# Bayes with Approximate Posterior

$$\arg \min_{q \in \mathcal{P}} \mathbb{E}_{q(\theta)} [\ell(\theta)] - \mathcal{H}(q)$$

All distribution      Distribution      Entropy

Restrict the set of distribution from P to Q

$$\arg \min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)} [\ell(\theta)] - \mathcal{H}(q)$$

# Bayes with Approximate Posterior

$$\arg \min_{q \in \mathcal{P}} \mathbb{E}_{q(\theta)} [\ell(\theta)] - \mathcal{H}(q)$$

All distribution      Distribution      Entropy

Restrict the set of distribution from P to Q

$$\arg \min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)} [\ell(\theta)] - \mathcal{H}(q)$$

This is known as **Variational Inference**, but along with the Bayesian learning rule, it enables us to derive many more algorithms (including Bayes' rule). So this is not just a method, but a principle.

# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta)$$

# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top}T(\theta)$$



# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} T(\theta) \leftarrow \text{Sufficient statistics of } q$$

# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} T(\theta) \leftarrow \text{Sufficient statistics of } q$$

$$\ell(\theta) := (y - X\theta)^{\top} (y - X\theta) + \gamma\theta^{\top} \theta$$

# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} T(\theta) \leftarrow \text{Sufficient statistics of } q$$

$$\begin{aligned}\ell(\theta) &:= (y - X\theta)^{\top} (y - X\theta) + \gamma\theta^{\top}\theta \\ &= -2\theta^{\top} (X^{\top}y) + \text{Tr} [\theta\theta^{\top} (X^{\top}X + \gamma I)] + \text{cnst}\end{aligned}$$

# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} T(\theta) \leftarrow \text{Sufficient statistics of } q$$

$$\begin{aligned}\ell(\theta) &:= (y - X\theta)^{\top} (y - X\theta) + \gamma\theta^{\top} \theta \\ &= -2\theta^{\top} (X^{\top} y) + \text{Tr} [\theta\theta^{\top} (X^{\top} X + \gamma I)] + \text{cnst}\end{aligned}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu$$

# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} T(\theta) \leftarrow \text{Sufficient statistics of } q$$

$$\begin{aligned}\ell(\theta) &:= (y - X\theta)^{\top} (y - X\theta) + \gamma\theta^{\top} \theta \\ &= -2\theta^{\top} (X^{\top} y) + \text{Tr} [\theta\theta^{\top} (X^{\top} X + \gamma I)] + \text{cnst}\end{aligned}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}^{\top} \mu \implies \nabla_{\mu} \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} T(\theta) \leftarrow \text{Sufficient statistics of } q$$

$$\begin{aligned}\ell(\theta) &:= (y - X\theta)^{\top} (y - X\theta) + \gamma\theta^{\top} \theta \\ &= -2\theta^{\top} (X^{\top} y) + \text{Tr} [\theta\theta^{\top} (X^{\top} X + \gamma I)] + \text{cnst}\end{aligned}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}^{\top} \mu \implies \nabla_{\mu} \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

$$\lambda \leftarrow \lambda - \rho \nabla_{\mu} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$$

# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} T(\theta) \leftarrow \text{Sufficient statistics of } q$$

$$\begin{aligned}\ell(\theta) &:= (y - X\theta)^{\top} (y - X\theta) + \gamma\theta^{\top}\theta \\ &= -2\theta^{\top} (X^{\top}y) + \text{Tr} [\theta\theta^{\top} (X^{\top}X + \gamma I)] + \text{cnst}\end{aligned}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu \implies \nabla_{\mu}\mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

$$\lambda \leftarrow \lambda - \rho (\nabla_{\mu}\mathbb{E}_q[\ell(\theta)] + \lambda)$$

# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} \boxed{T(\theta)} \leftarrow \begin{array}{l} \text{Sufficient} \\ \text{statistics of } q \end{array}$$

$$\begin{aligned} \ell(\theta) &:= (y - X\theta)^{\top} (y - X\theta) + \gamma\theta^{\top} \theta \\ &= -2\theta^{\top} (X^{\top} y) + \text{Tr} [\theta\theta^{\top} (X^{\top} X + \gamma I)] + \text{cnst} \end{aligned}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu \implies \nabla_{\mu} \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

$$\lambda \leftarrow \lambda - \rho(-\lambda_{\mathcal{D}} + \lambda)$$



# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} T(\theta) \leftarrow \text{Sufficient statistics of } q$$

$$\begin{aligned}\ell(\theta) &:= (y - X\theta)^{\top} (y - X\theta) + \gamma\theta^{\top} \theta \\ &= -2\theta^{\top} (X^{\top} y) + \text{Tr} [\theta\theta^{\top} (X^{\top} X + \gamma I)] + \text{cnst}\end{aligned}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu \implies \nabla_{\mu} \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

$$\lambda \leftarrow \lambda - \rho(-\lambda_{\mathcal{D}} + \lambda) \implies \lambda_* = \lambda_{\mathcal{D}}$$

# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} T(\theta) \leftarrow \text{Sufficient statistics of } q$$

$$\begin{aligned}\ell(\theta) &:= (y - X\theta)^{\top} (y - X\theta) + \gamma\theta^{\top}\theta \\ &= -2\theta^{\top} (X^{\top}y) + \text{Tr} [\theta\theta^{\top} (X^{\top}X + \gamma I)] + \text{cnst}\end{aligned}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu \implies \nabla_{\mu}\mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

$$\lambda \leftarrow \lambda - \rho(-\lambda_{\mathcal{D}} + \lambda) \implies \lambda_* = \lambda_{\mathcal{D}}$$

$$S_* = X^{\top}X + \gamma I$$

# Conjugate Bayesian Inference from Bayesian Principles

Ex: Linear model, Kalman filters, HMM, etc.

$$\ell(\theta) := -\log p(\mathcal{D}|\theta)p(\theta) = -\lambda_{\mathcal{D}}^{\top} T(\theta) \leftarrow \text{Sufficient statistics of } q$$

$$\begin{aligned}\ell(\theta) &:= (y - X\theta)^{\top} (y - X\theta) + \gamma\theta^{\top}\theta \\ &= -2\theta^{\top} (X^{\top}y) + \text{Tr} [\theta\theta^{\top} (X^{\top}X + \gamma I)] + \text{cnst}\end{aligned}$$

$$\implies \mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}\mu \implies \nabla_{\mu}\mathbb{E}_q[\ell(\theta)] = -\lambda_{\mathcal{D}}$$

$$\lambda \leftarrow \lambda - \rho(-\lambda_{\mathcal{D}} + \lambda) \implies \lambda_* = \lambda_{\mathcal{D}}$$

$$S_* = X^{\top}X + \gamma I \quad m_* = (X^{\top}X + \gamma I)^{-1}X^{\top}y$$

# Conjugate Bayesian Inference from Bayesian Principles

The following algorithms can be obtained by setting  $\lambda_* = \lambda_{\mathcal{D}}$

- Forward-backward algorithm [2]
  - Kalman filters, HMM etc.
- Stochastic Variational Inference [3]
- Variational message passing [4]

1. Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." *Alstats* (2017).
2. Binder et al.. Space-Efficient Inference in Dynamic Probabilistic Networks. *IJCAI* (1997).
3. Hoffman et al. Stochastic variational inference. *JMLR* (2013)
4. Winn and Bishop. "Variational message passing." *JMLR* (2005)

# Laplace Approximation

Derived by choosing a multivariate Gaussian, then running the following Newton's update

$$m \leftarrow m - \rho \mathbf{S}^{-1} \nabla_m \ell(m)$$

$$\mathbf{S} \leftarrow (1 - \rho) \mathbf{S} + \rho \mathbf{H}_m \leftarrow \text{Hessian at } m$$

# Laplace Approximation

Derived by choosing a multivariate Gaussian, then running the following Newton's update

$$\begin{aligned} m &\leftarrow m - \rho \mathbf{S}^{-1} \nabla_m \ell(m) \\ \mathbf{S} &\leftarrow (1 - \rho) \mathbf{S} + \rho \mathbf{H}_m \end{aligned}$$

← Hessian at  $m$

Bayesian principles we discussed are general principles to derive learning algorithms

# Laplace Approximation

Derived by choosing a multivariate Gaussian, then running the following Newton's update

$$\begin{aligned} m &\leftarrow m - \rho \mathbf{S}^{-1} \nabla_m \ell(m) \\ \mathbf{S} &\leftarrow (1 - \rho) \mathbf{S} + \rho \mathbf{H}_m \end{aligned}$$

← Hessian at  $m$

Bayesian principles we discussed are general principles to derive learning algorithms

Calling them variational inference limits their scope!

# References for Posterior Approximations

$$\arg \min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$



# References for Posterior Approximations

$$\arg \min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)} [\ell(\theta)] - \mathcal{H}(q)$$

- Variational inference

1. Hinton, Geoffrey, and Drew Van Camp. "Keeping neural networks simple by minimizing the description length of the weights." *COLT* 1993.
2. Jordan, Michael I., et al. "An introduction to variational methods for graphical models." *Machine learning* 37.2 (1999): 183-233.

# References for Posterior Approximations

$$\arg \min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)} [\ell(\theta)] - \mathcal{H}(q)$$

- Variational inference

1. Hinton, Geoffrey, and Drew Van Camp. "Keeping neural networks simple by minimizing the description length of the weights." *COLT* 1993.
2. Jordan, Michael I., et al. "An introduction to variational methods for graphical models." *Machine learning* 37.2 (1999): 183-233.

- Entropy-regularized / Maximum-entropy RL

3. Williams, Ronald J., and Jing Peng. "Function optimization using connectionist reinforcement learning algorithms." *Connection Science* 3.3 (1991): 241-268.
4. Ziebart, Brian D. Modeling purposeful adaptive behavior with the principle of maximum causal entropy. Diss. figshare, 2010. (see chapter 5)

# References for Posterior Approximations

$$\arg \min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)} [\ell(\theta)] - \mathcal{H}(q)$$

- Variational inference

1. Hinton, Geoffrey, and Drew Van Camp. "Keeping neural networks simple by minimizing the description length of the weights." *COLT* 1993.
2. Jordan, Michael I., et al. "An introduction to variational methods for graphical models." *Machine learning* 37.2 (1999): 183-233.

- Entropy-regularized / Maximum-entropy RL

3. Williams, Ronald J., and Jing Peng. "Function optimization using connectionist reinforcement learning algorithms." *Connection Science* 3.3 (1991): 241-268.
4. Ziebart, Brian D. Modeling purposeful adaptive behavior with the principle of maximum causal entropy. Diss. figshare, 2010. (see chapter 5)

- Parameter-Space Exploration in RL

5. Rückstieß, Thomas, et al. "Exploring parameter space in reinforcement learning." *Paladyn, Journal of Behavioral Robotics* 1.1 (2010): 14-24.
6. Plappert, Matthias, et al. "Parameter space noise for exploration." *arXiv preprint arXiv:1706.01905* (2017)
7. Fortunato, Meire, et al. "Noisy networks for exploration." *arXiv preprint arXiv:1706.10295* (2017).

# More References for Posterior Approximations

- Evolution strategy  $\arg \min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)} [\ell(\theta)]$ 
  1. Ingo Rechenberg, *Evolutionsstrategie – Optimierung technischer Systeme nach Prinzipien der biologischen Evolution* (PhD thesis) 1971.
- Gaussian Homotopy
  2. Mobahi, Hossein, and John W. Fisher III. "A theoretical analysis of optimization by Gaussian continuation." *Twenty-Ninth AAAI Conference on Artificial Intelligence*. 2015.
- Smoothing-based Optimization
  3. Leordeanu, Marius, and Martial Hebert. "Smoothing-based optimization." *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008.
- Graduated Optimization
  4. Hazan, Elad, Kfir Yehuda Levy, and Shai Shalev-Shwartz. "On graduated optimization for stochastic non-convex problems." *International conference on machine learning*. 2016.
- Stochastic Search
  5. Zhou, Enlu, and Jiaqiao Hu. "Gradient-based adaptive stochastic search for non-differentiable optimization." *IEEE Transactions on Automatic Control* 59.7 (2014): 1818-1832.

# Bayesian Learning Rule and Related Works

$$\min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

Bayes learning rule:  $\lambda \leftarrow \lambda - \rho \nabla_{\mu} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$

Natural-Gradient VI:  $\lambda \leftarrow \lambda - \rho F_q^{-1} \nabla_{\lambda} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$

 Fisher Information Matrix

1. Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." *Alstats* (2017).
2. Raskutti, Garvesh, and Sayan Mukherjee. "The information geometry of mirror descent." *IEEE Transactions on Information Theory* 61.3 (2015): 1451-1457.

# Bayesian Learning Rule and Related Works

$$\min_{q \in \mathcal{Q}} \mathbb{E}_{q(\theta)}[\ell(\theta)] - \mathcal{H}(q)$$

Bayes learning rule:  $\lambda \leftarrow \lambda - \rho \nabla_{\mu} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$

Natural-Gradient VI:  $\lambda \leftarrow \lambda - \rho F_q^{-1} \nabla_{\lambda} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$

 Fisher Information Matrix

Also equivalent to a mirror-descent algorithm. The Geometry of the mirror-descent is defined by the log partition function of the posterior approximation.

1. Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." *Alstats* (2017).
2. Raskutti, Garvesh, and Sayan Mukherjee. "The information geometry of mirror descent." *IEEE Transactions on Information Theory* 61.3 (2015): 1451-1457.

# References for Step C: Natural-Gradient VI

1. Sato, Masa-aki. "Fast learning of on-line EM algorithm." Technical Report, ATR Human Information Processing Research Laboratories (1999).
2. Sato, Masa-Aki. "Online model selection based on the variational Bayes." *Neural computation* 13.7 (2001): 1649-1681.
3. Winn, John, and Christopher M. Bishop. "Variational message passing." *Journal of Machine Learning Research* 6.Apr (2005): 661-694.
4. Honkela, Antti, et al. "Approximate Riemannian conjugate gradient learning for fixed-form variational Bayes." *Journal of Machine Learning Research* 11.Nov (2010): 3235-3268.
5. Knowles, David A., and Tom Minka. "Non-conjugate variational message passing for multinomial and binary regression." *NeurIPS*. (2011).
6. Hoffman, Matthew D., et al. "Stochastic variational inference." *JMLR* (2013).
7. Salimans, Tim, and David A. Knowles. "Fixed-form variational posterior approximation through stochastic linear regression." *Bayesian Analysis* 8.4 (2013): 837-882.
8. Sheth, Rishit, and Roni Khardon. "Monte Carlo Structured SVI for Two-Level Non-Conjugate Models." *arXiv preprint arXiv:1612.03957* (2016).
9. Khan and Lin. "Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models." *Alstats* (2017).
10. Khan and Nielsen. "Fast yet simple natural-gradient descent for variational inference in complex models." (2018) *ISITA*.
11. Zhang, Guodong, et al. "Noisy natural gradient as variational inference." *ICML* (2018).

# Black-Box VI & Bayesian Learning rule

Bayes learning rule:  $\lambda \leftarrow \lambda - \rho \nabla_{\mu} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$

Black-Box VI [1]:  $\lambda \leftarrow \lambda - \rho \nabla_{\lambda} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$

Black-box VI is more generally applicable (beyond exponential-family), but we cannot derive learning-algorithms from it (even for conjugate Bayesian models)



# Learning-Algorithms from Bayesian Principles

Bayesian learning rule:  $\lambda \leftarrow \lambda - \rho \nabla_{\mu} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$

Given a loss, we can recover a variety of learning algorithms by choosing an appropriate  $q$

- Classical algorithms: Least-squares, gradient descent, Newton's method, Kalman filters, Baum-Welch, Forward-backward, etc.
- Bayesian inference: EM, Laplace's method, SVI, VMP.
- Deep learning: SGD, RMSprop, Adam.
- Reinforcement learning: parameter-space exploration, natural policy-search.
- Continual learning: Elastic-weight consolidation.
- Online learning: Exponential-weight average.
- Global optimization: Natural evolutionary strategies, Gaussian homotopy, continuation method & smoothed optimization.

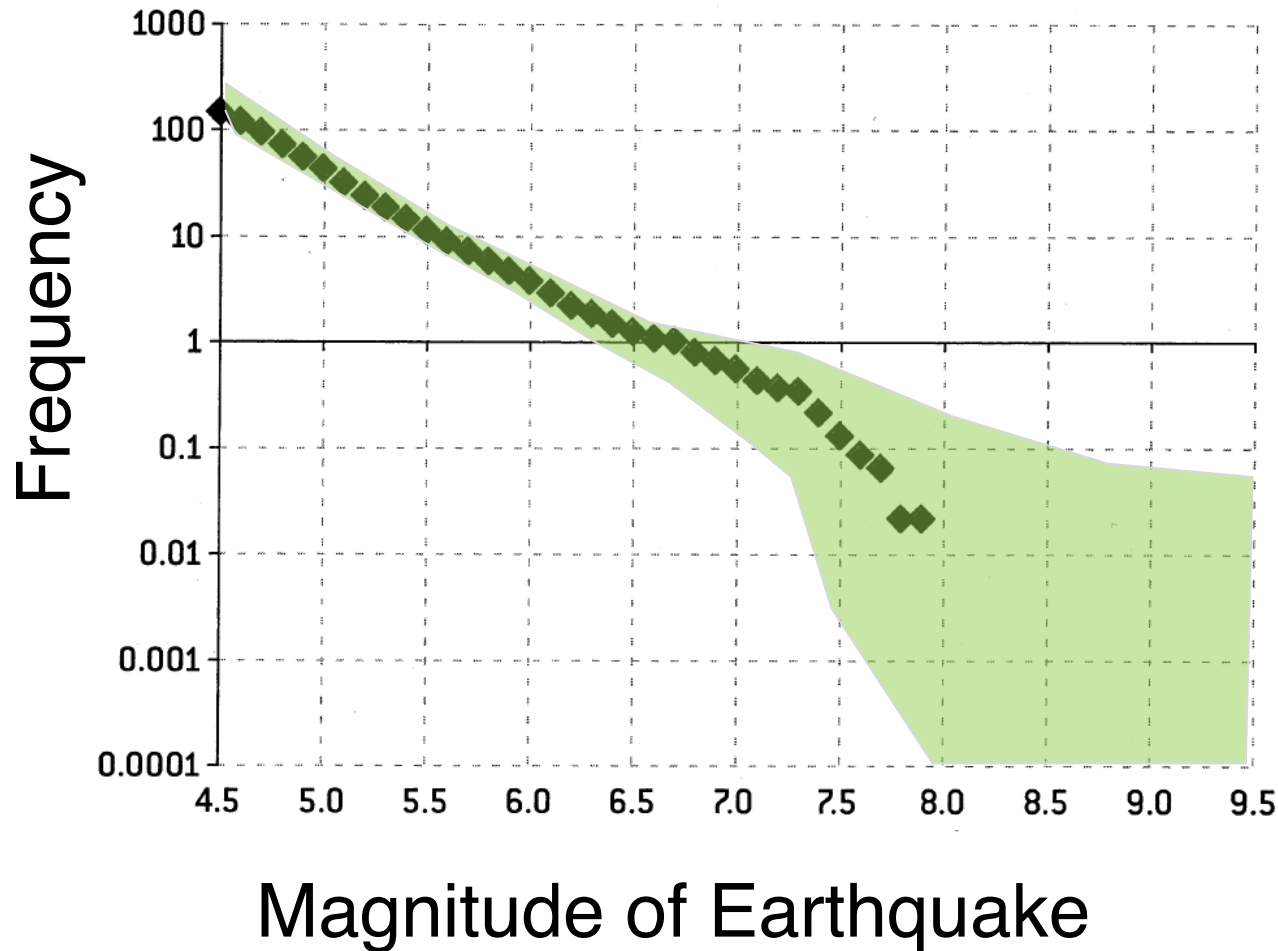
# Deep Learning with Bayesian Principles

- Bayesian principles as common principles
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning
- Impact: Many learning-algorithms with a common set of principles.

# **Uncertainty Estimation for Deep Learning**

New deep-learning algorithms

# Uncertainty for Robust Decisions



Uncertainty:  
“What the  
model does  
not know”

Choose less  
risky options!

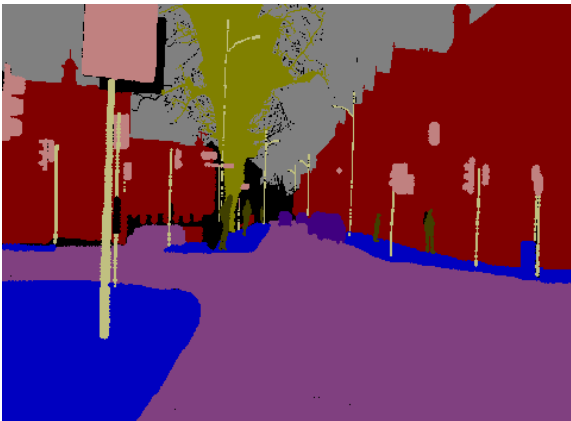
Avoid data  
bias with  
uncertainty!

# Uncertainty Estimation for Image segmentation

Image



True Segments



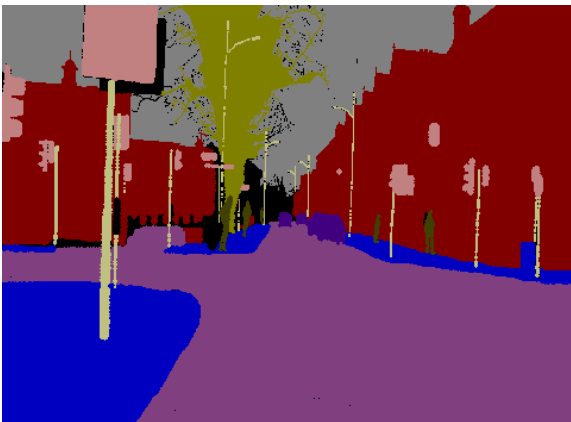
Kendall, Alex, Yarin Gal, and Roberto Cipolla. "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics." *CVPR*. 2018.

# Uncertainty Estimation for Image segmentation

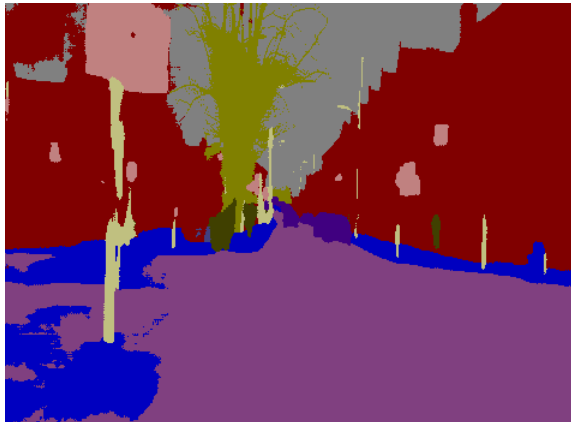
Image



True Segments



Prediction



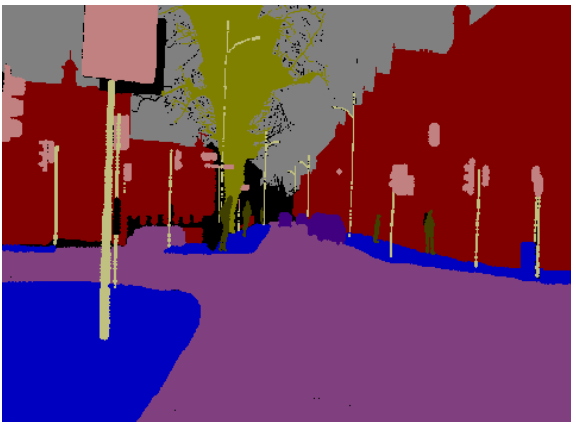
Kendall, Alex, Yarin Gal, and Roberto Cipolla. "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics." *CVPR*. 2018.

# Uncertainty Estimation for Image segmentation

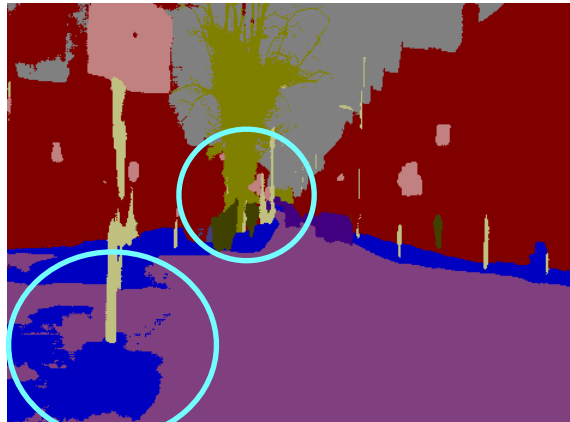
Image



True Segments



Prediction



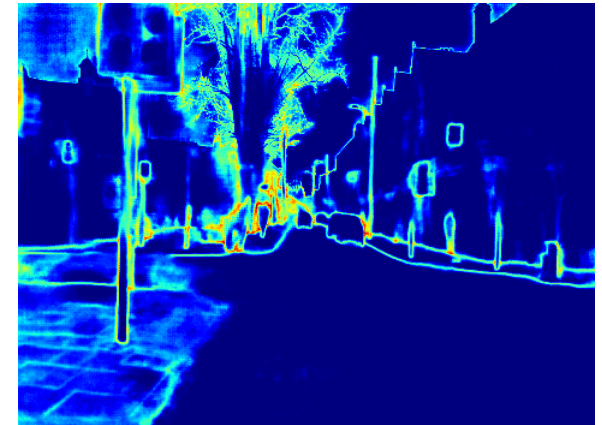
Kendall, Alex, Yarin Gal, and Roberto Cipolla. "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics." *CVPR*. 2018.

# Uncertainty Estimation for Image segmentation

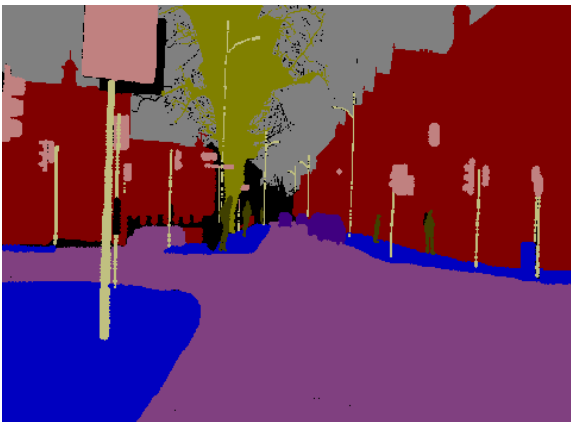
Image



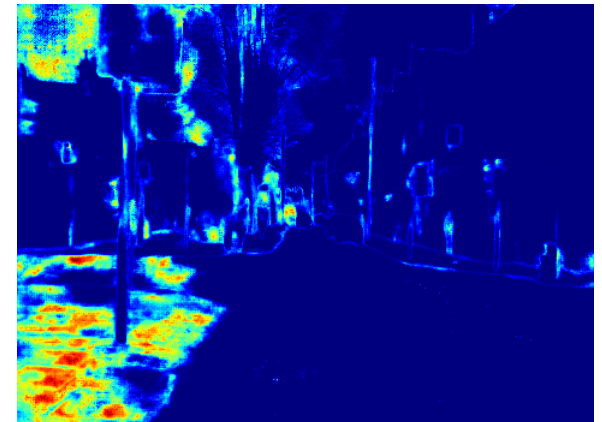
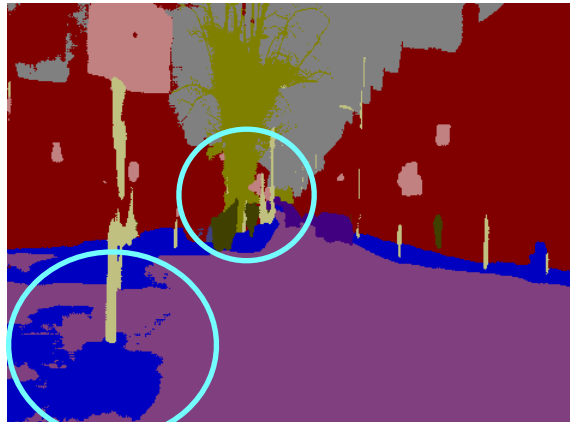
Uncertainty



True Segments



Prediction



Kendall, Alex, Yarin Gal, and Roberto Cipolla. "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics." *CVPR*. 2018.



# (Some) Bayesian Deep Learning Methods

1. Gal and Ghahramani. "Dropout as a bayesian approximation..." *ICML*. 2016.
2. Maddox, Wesley, et al. "A simple baseline for bayesian uncertainty in deep learning." *arXiv* (2019).
3. Ritter et al. "A scalable laplace approximation for neural networks." (2018).
4. Graves, Alex. "Practical variational inference for neural networks." *NeurIPS* (2011).
5. Blundell, Charles, et al. "Weight uncertainty in neural networks." *ICML* (2015).

# (Some) Bayesian Deep Learning Methods

- SGD based (MC-dropout [1], SWAG [2], Laplace [3])
  - Pros: Scales well to large problems
  - Cons: Not flexible

1. Gal and Ghahramani. "Dropout as a bayesian approximation..." *ICML*. 2016.
2. Maddox, Wesley, et al. "A simple baseline for bayesian uncertainty in deep learning." *arXiv* (2019).
3. Ritter et al. "A scalable laplace approximation for neural networks." (2018).
4. Graves, Alex. "Practical variational inference for neural networks." *NeurIPS* (2011).
5. Blundell, Charles, et al. "Weight uncertainty in neural networks." *ICML* (2015).

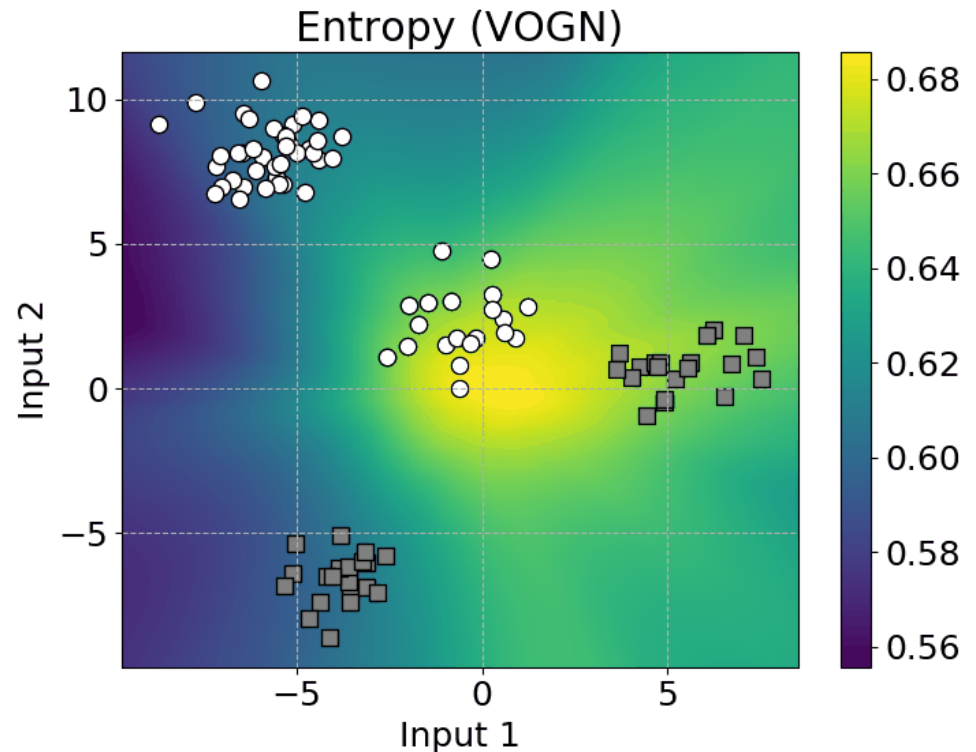
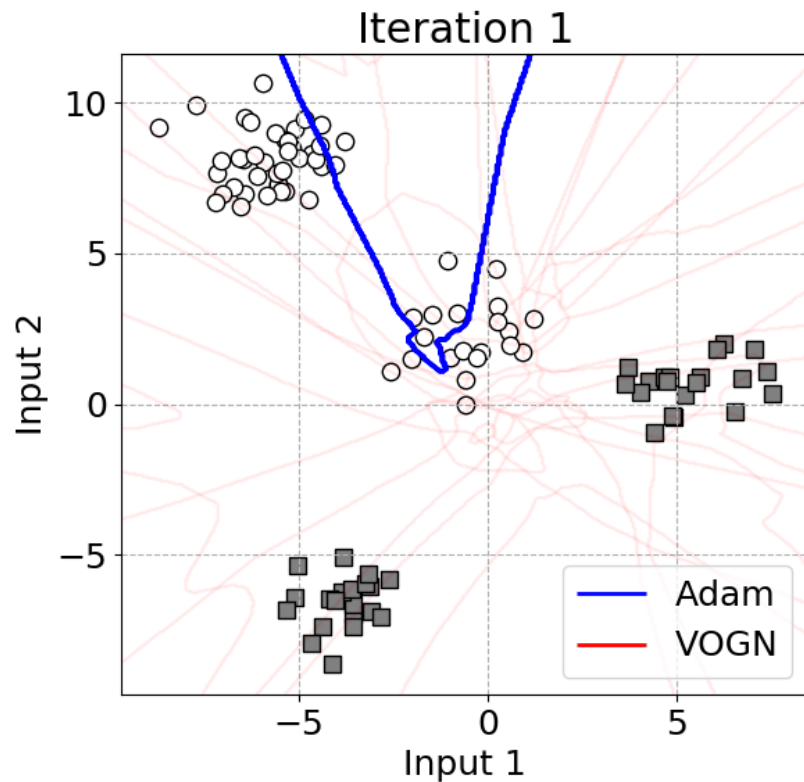
# (Some) Bayesian Deep Learning Methods

- SGD based (MC-dropout [1], SWAG [2], Laplace [3])
  - Pros: Scales well to large problems
  - Cons: Not flexible
- Variational inference methods [4,5]
$$\lambda \leftarrow \lambda - \rho \nabla_{\lambda} (\mathbb{E}_q[\ell(\theta)] - \mathcal{H}(q))$$
  - Pros: Enable flexible distributions
  - Cons: Do not scale to large problems (ImageNet)

1. Gal and Ghahramani. "Dropout as a bayesian approximation..." *ICML*. 2016.
2. Maddox, Wesley, et al. "A simple baseline for bayesian uncertainty in deep learning." *arXiv* (2019).
3. Ritter et al. "A scalable laplace approximation for neural networks." (2018).
4. Graves, Alex. "Practical variational inference for neural networks." *NeurIPS* (2011).
5. Blundell, Charles, et al. "Weight uncertainty in neural networks." *ICML* (2015).

# Scaling up VI to ImageNet

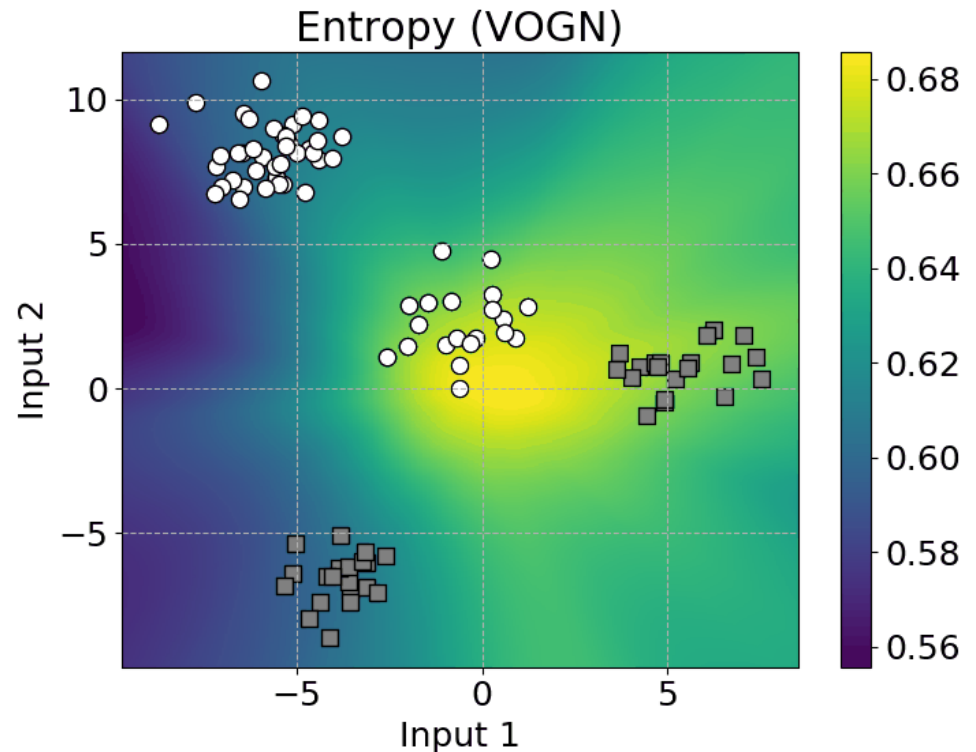
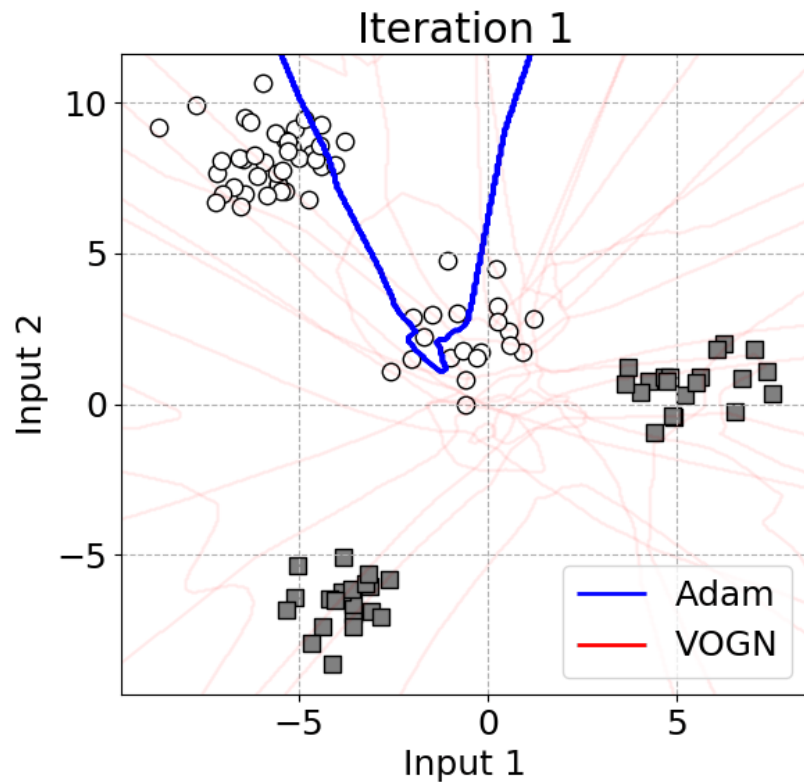
VOGN, an Adam-like algorithm, for uncertainty



1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).
2. Osawa et al. "Practical Deep Learning with Bayesian Principles." *NeurIPS* (2019).

# Scaling up VI to ImageNet

VOGN, an Adam-like algorithm, for uncertainty



1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).
2. Osawa et al. "Practical Deep Learning with Bayesian Principles." *NeurIPS* (2019).

# Variational Online Gauss-Newton

- Improve RMSprop with the Bayesian “touch”
  - Remove the “local” approximation  $\mathbb{E}_q[\ell(\theta)] \approx \ell(m)$
  - Use a second-order approximation
  - No square root of the scale
- Improve VOGN by using deep learning tricks
  - Momentum, batch norm, data augmentation etc

## RMSprop

$$\begin{aligned} g &\leftarrow \hat{\nabla} \ell(\theta) \\ s &\leftarrow (1 - \rho)s + \rho g^2 \\ \theta &\leftarrow \theta - \alpha(\sqrt{s} + \delta)^{-1} g \end{aligned}$$

## VOGN

$$\begin{aligned} g &\leftarrow \hat{\nabla} \ell(\theta), \text{ where } \theta \sim \mathcal{N}(m, \sigma^2) \\ s &\leftarrow (1 - \rho)s + \rho(\Sigma_i g_i^2) \\ m &\leftarrow m - \alpha(s + \gamma)^{-1} \nabla_{\theta} \ell(\theta) \\ \sigma^2 &\leftarrow (s + \gamma)^{-1} \end{aligned}$$

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).
2. Osawa et al. "Practical Deep Learning with Bayesian Principles." *NeurIPS* (2019).

# Adam to VOGN

“Adam” to “VOGN” in two lines of code change.

```
import torch
+import torchsso

train_loader = torch.utils.data.DataLoader(train_dataset)
model = MLP()

-optimizer = torch.optim.Adam(model.parameters())
+optimizer = torchsso.optim.VOGN(model, dataset_size=len(train_loader.dataset))
```

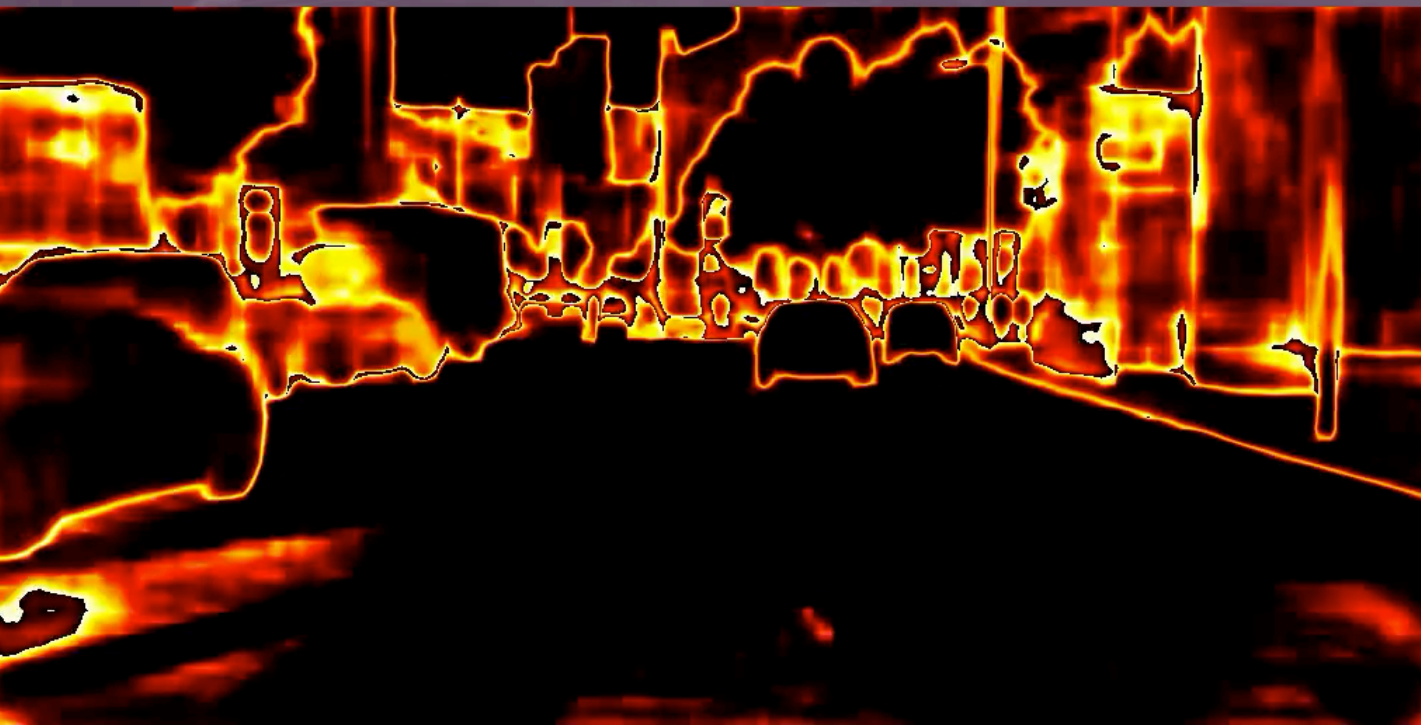
Available at <https://github.com/team-approx-bayes/dl-with-bayes>

## Uses many practical tricks of DL to scale Bayes

1. Khan, et al. "Fast and scalable Bayesian deep learning by weight-perturbation in Adam." *ICML* (2018).
2. Osawa et al. "Practical Deep Learning with Bayesian Principles." *NeurIPS* (2019).



Image  
Segmentation

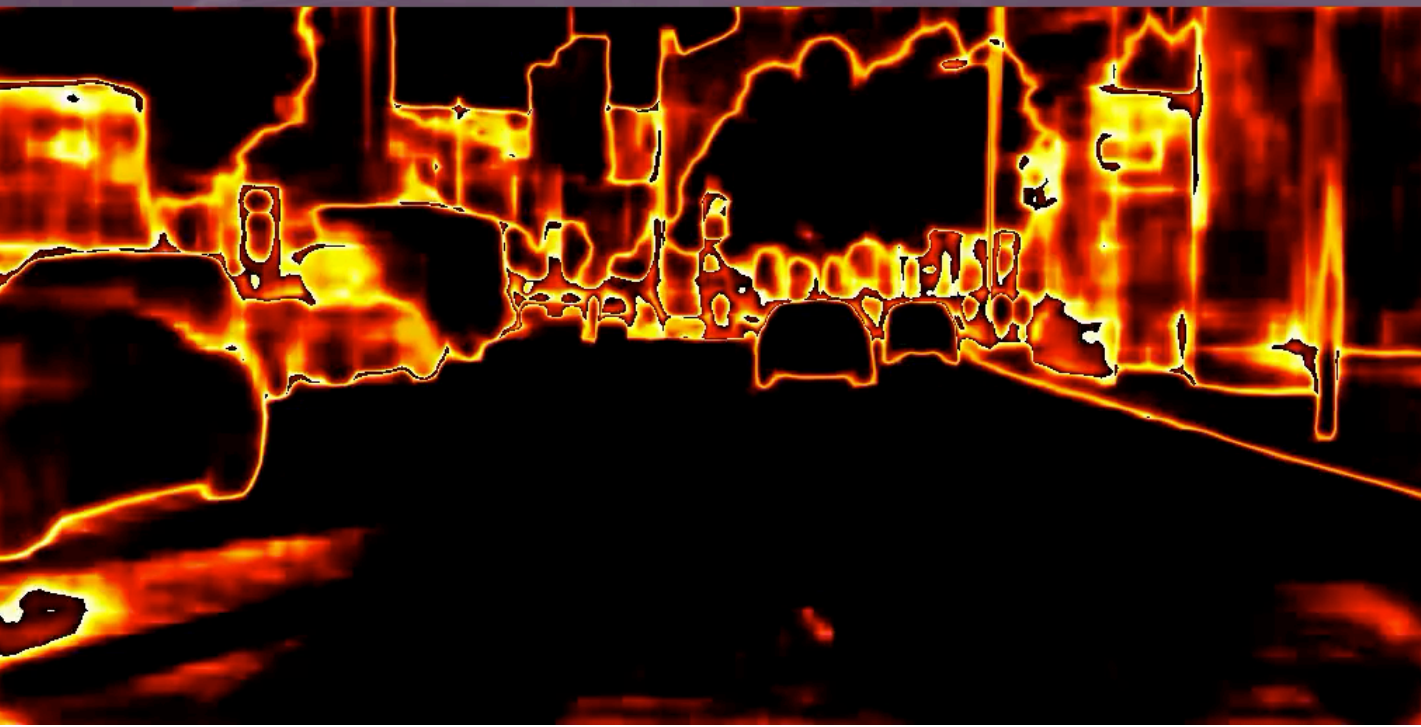


Uncertainty  
(entropy of  
class probs)





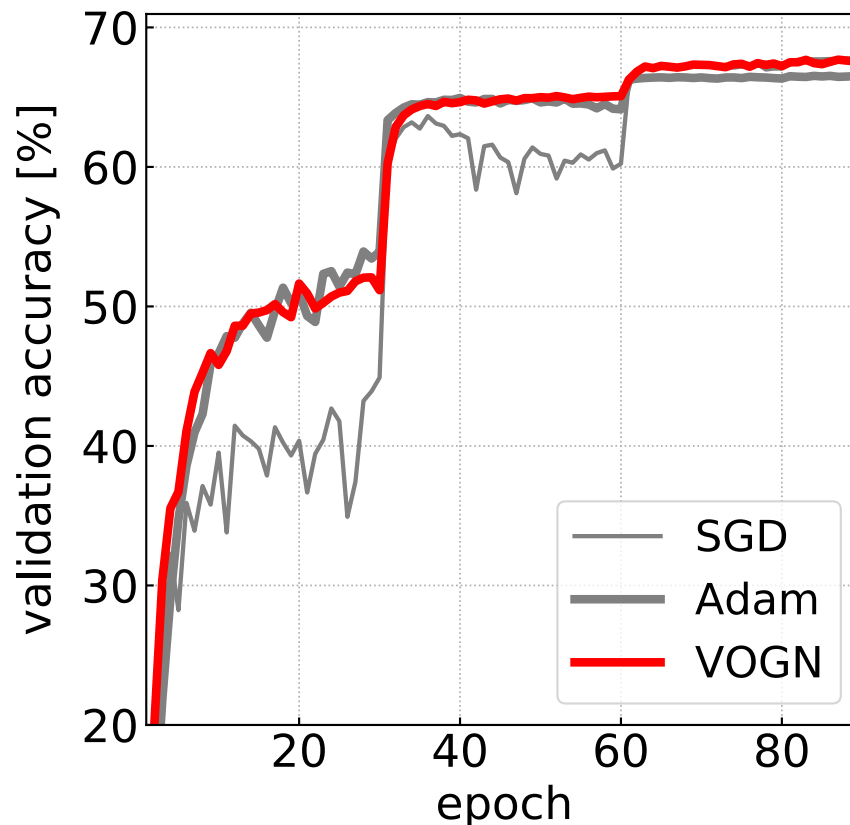
Image  
Segmentation



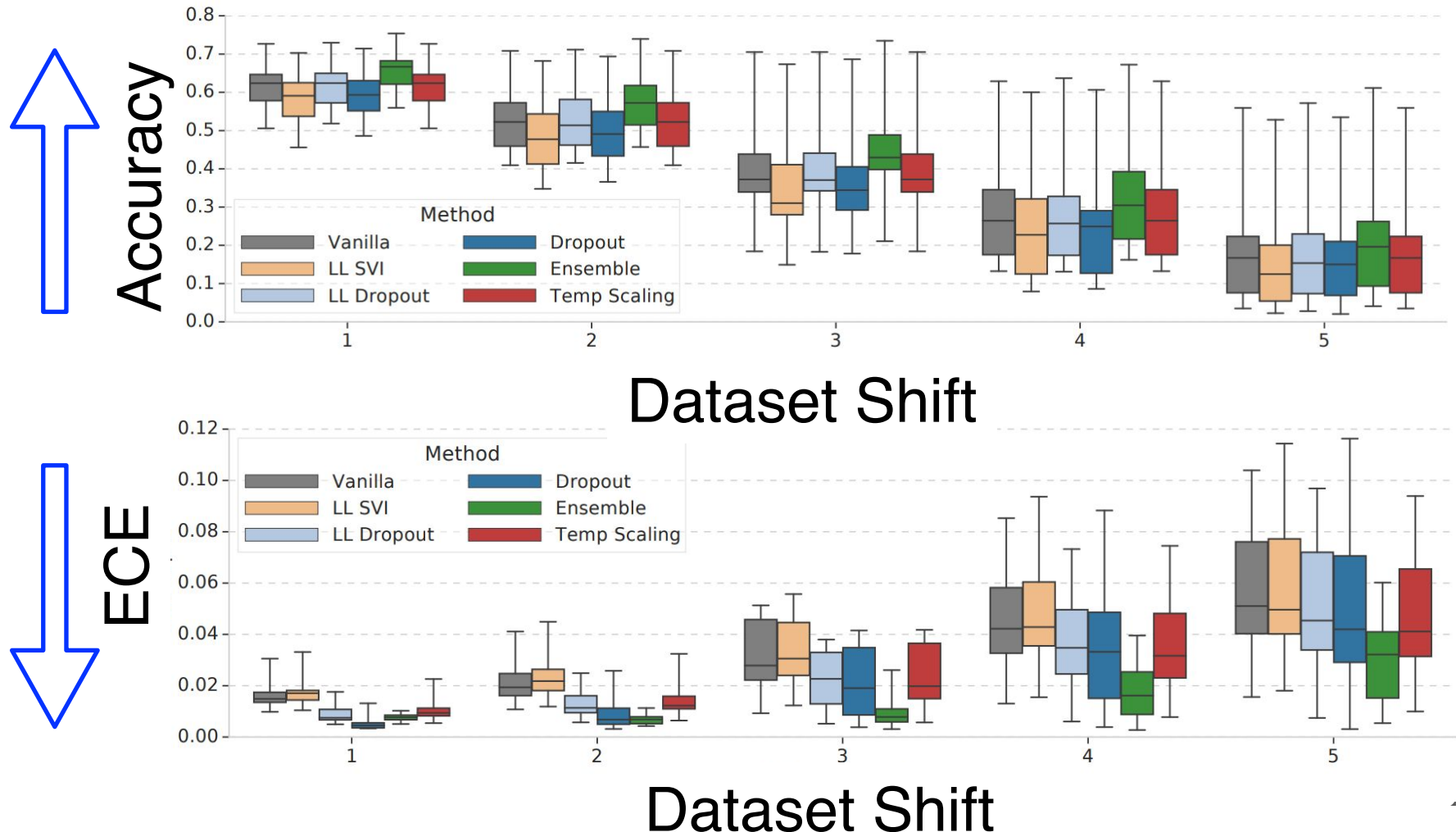
Uncertainty  
(entropy of  
class probs)

# VOGN on ImageNet

State-of-the-art performance and convergence rate, while preserving benefits of Bayesian principles



# BDL methods do not really know that they are performing badly under dataset shift



1. Ovadia, Yaniv, et al. "Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift." *NeurIPS* (2019).

# Resources for Uncertainty in DL

- Yarin Gal's tutorial (<http://bdl101.ml/>)
- Benchmarks by OATML (<http://bdlb.ml/>)

## List of Benchmarks

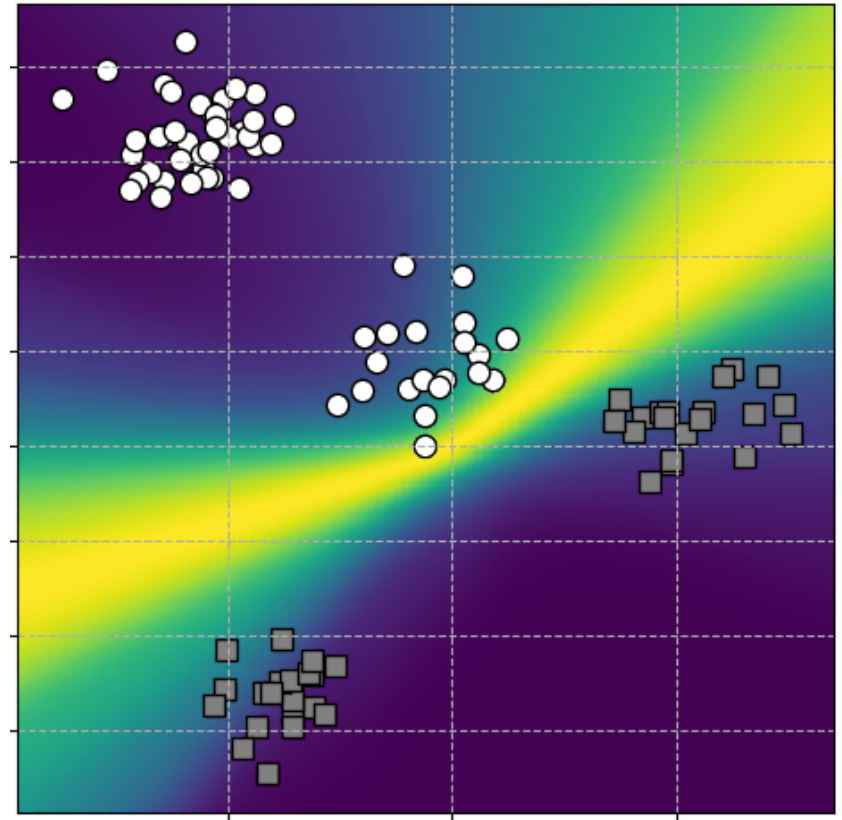
---

**Bayesian Deep Learning Benchmarks** (BDL Benchmarks or `bdlb` for short), is an open-source framework that aims to bridge the gap between the design of deep probabilistic machine learning models and their application to real-world problems. Our currently supported benchmarks are:

- ☒ [Diabetic Retinopathy Diagnosis](#) (in `alpha`, following [Leibig et al.](#))
  - ☒ [Deterministic](#)
  - ☒ [Monte Carlo Dropout](#) (following [Gal and Ghahramani, 2015](#))
  - ☒ [Mean-Field Variational Inference](#) (following [Peterson and Anderson, 1987](#), [Wen et al., 2018](#))
  - ☒ [Deep Ensembles](#) (following [Lakshminarayanan et al., 2016](#))
  - ☒ [Ensemble MC Dropout](#) (following [Smith and Gal, 2018](#))
- ☐ [Autonomous Vehicle's Scene Segmentation](#) (in `pre-alpha`, following [Mukhoti et al.](#))
- ☐ [Galaxy Zoo](#) (in `pre-alpha`, following [Walmsley et al.](#))
- ☐ [Fishyscapes](#) (in `pre-alpha`, following [Blum et al.](#))

# Challenges in Uncertainty Estimation

- For non convex problem
  - Different local minima correspond to various solutions
  - Local approximations only capture “local uncertainty”
- Solutions
  - More flexible approximations?

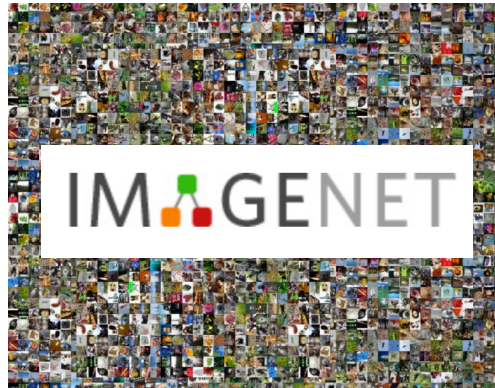


# Deep Learning with Bayesian Principles

- Bayesian principles as common principles
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and Life-Long learning
- Impact: Many learning-algorithms with a common set of principles.

# Continual Life-Long Learning

Standard  
Deep  
Learning

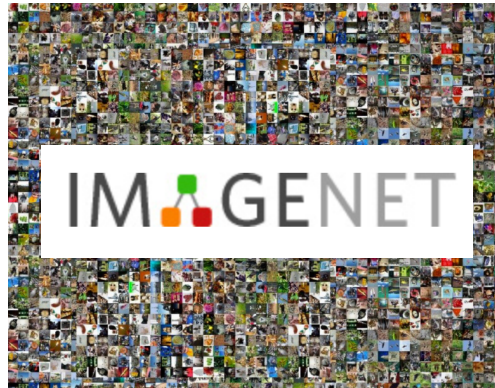


1. Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks." *PNAS* (2017)
2. Parisi et al. "Continual lifelong learning with neural networks: A review." *Neural Networks* (2019)

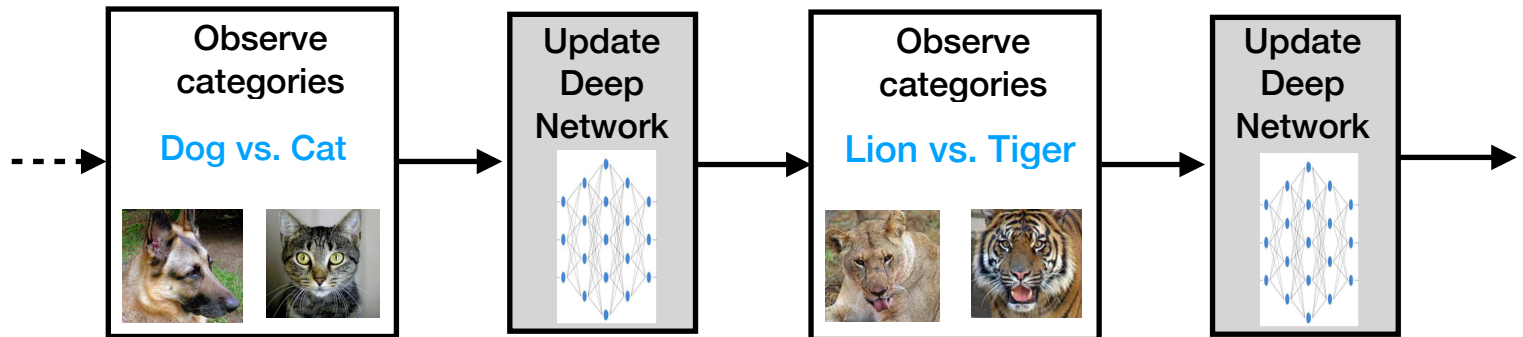


# Continual Life-Long Learning

Standard  
Deep  
Learning



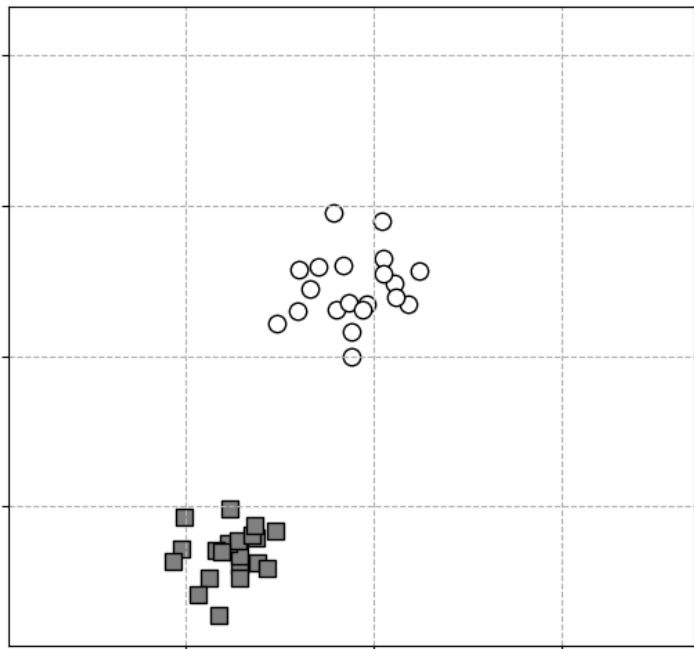
Continual Learning: past classes never revisited



1. Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks." *PNAS* (2017)
2. Parisi et al. "Continual lifelong learning with neural networks: A review." *Neural Networks* (2019)

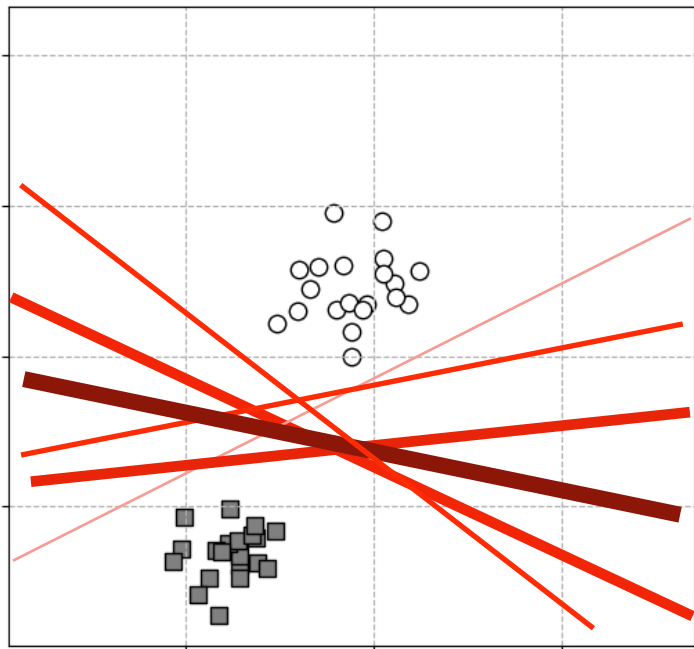


# Life-Long Learning with Bayes



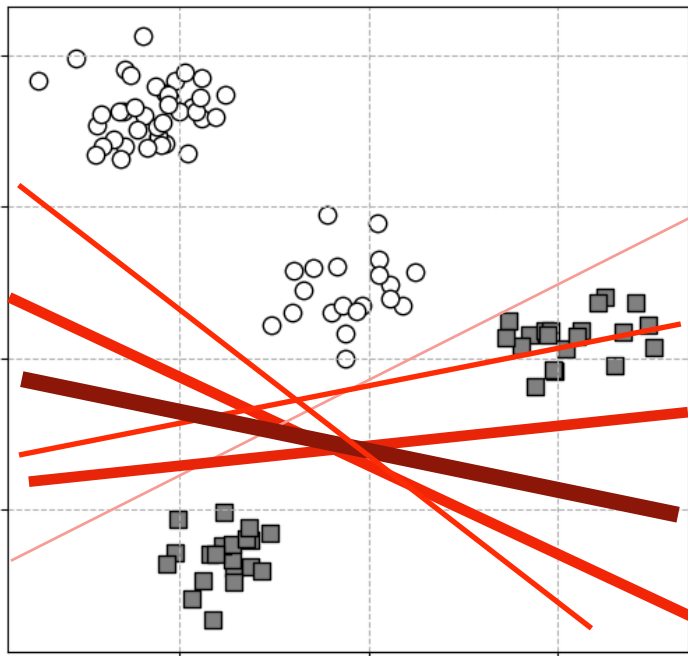
$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

# Life-Long Learning with Bayes



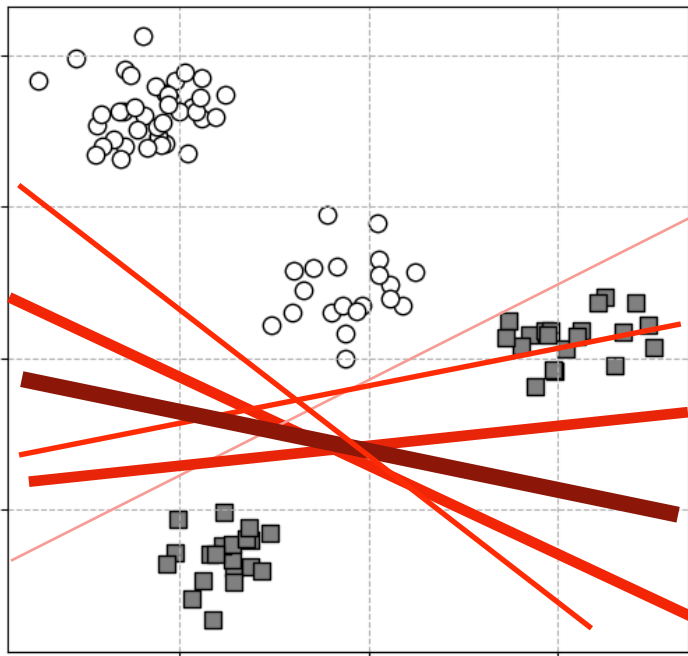
$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

# Life-Long Learning with Bayes



$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

# Life-Long Learning with Bayes

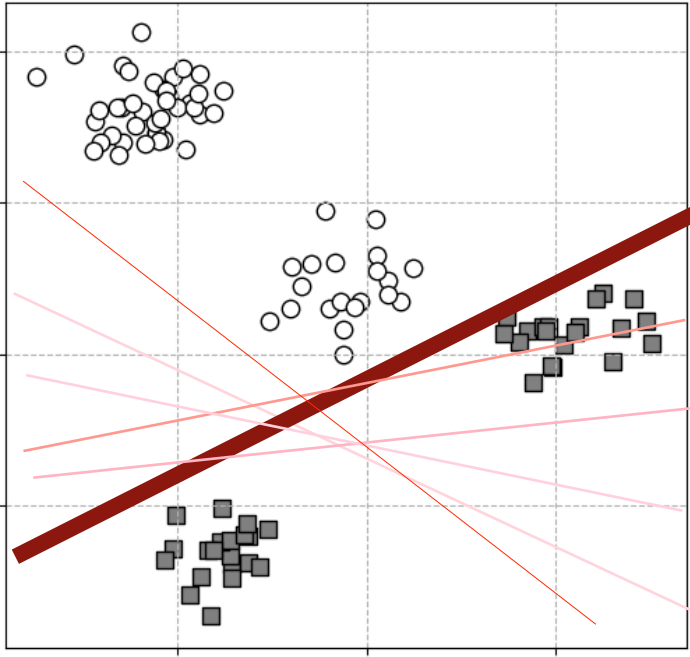


$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Set the prior to the previous posterior and recompute:

$$p(\theta|\mathcal{D}_2, \mathcal{D}_1) = \frac{p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)}{\int p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)d\theta}$$

# Life-Long Learning with Bayes

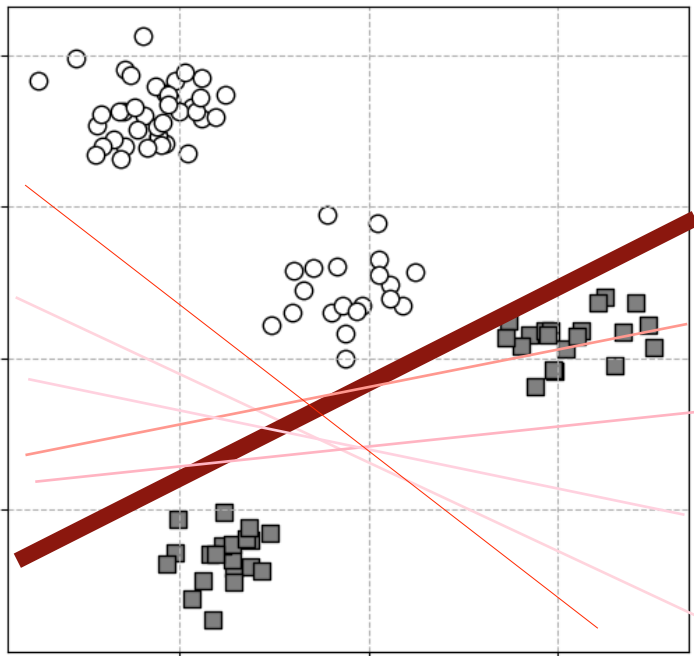


$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

Set the prior to the previous posterior and recompute:

$$p(\theta|\mathcal{D}_2, \mathcal{D}_1) = \frac{p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)}{\int p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)d\theta}$$

# Life-Long Learning with Bayes



$$p(\theta|\mathcal{D}_1) = \frac{p(\mathcal{D}_1|\theta)p(\theta)}{\int p(\mathcal{D}_1|\theta)p(\theta)d\theta}$$

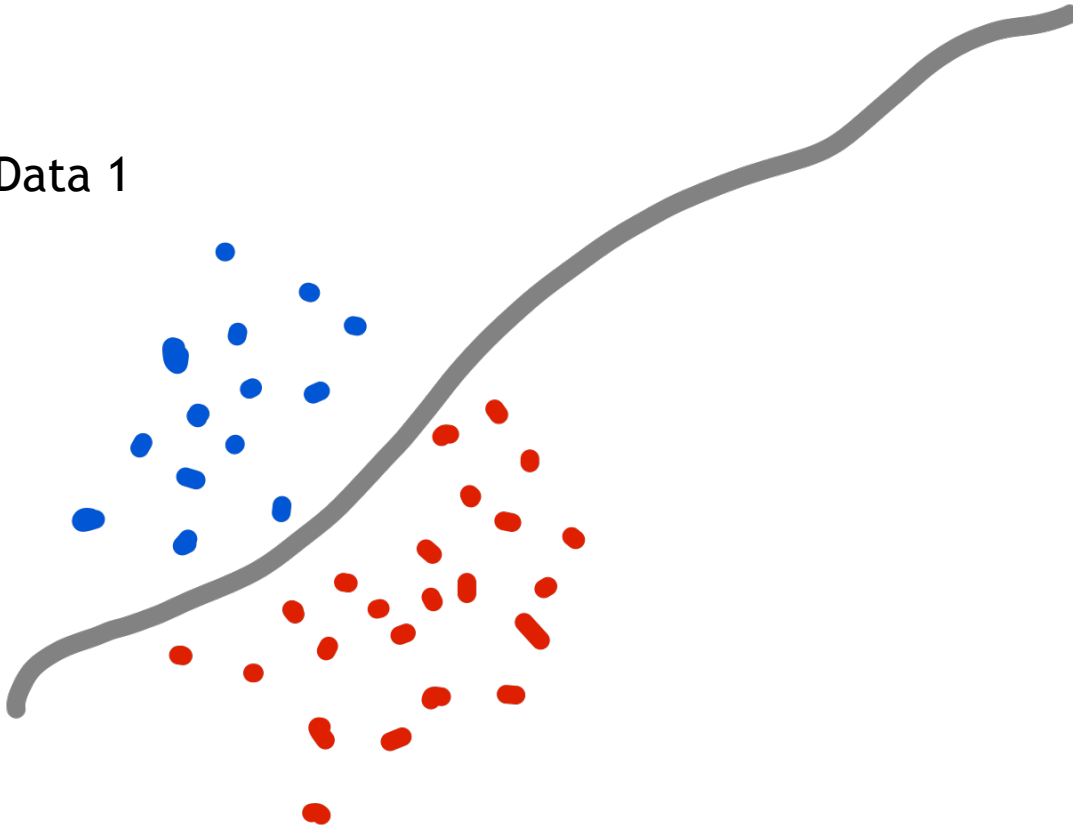
Set the prior to the previous posterior and recompute:

$$p(\theta|\mathcal{D}_2, \mathcal{D}_1) = \frac{p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)}{\int p(\mathcal{D}_2|\theta)p(\theta|\mathcal{D}_1)d\theta}$$

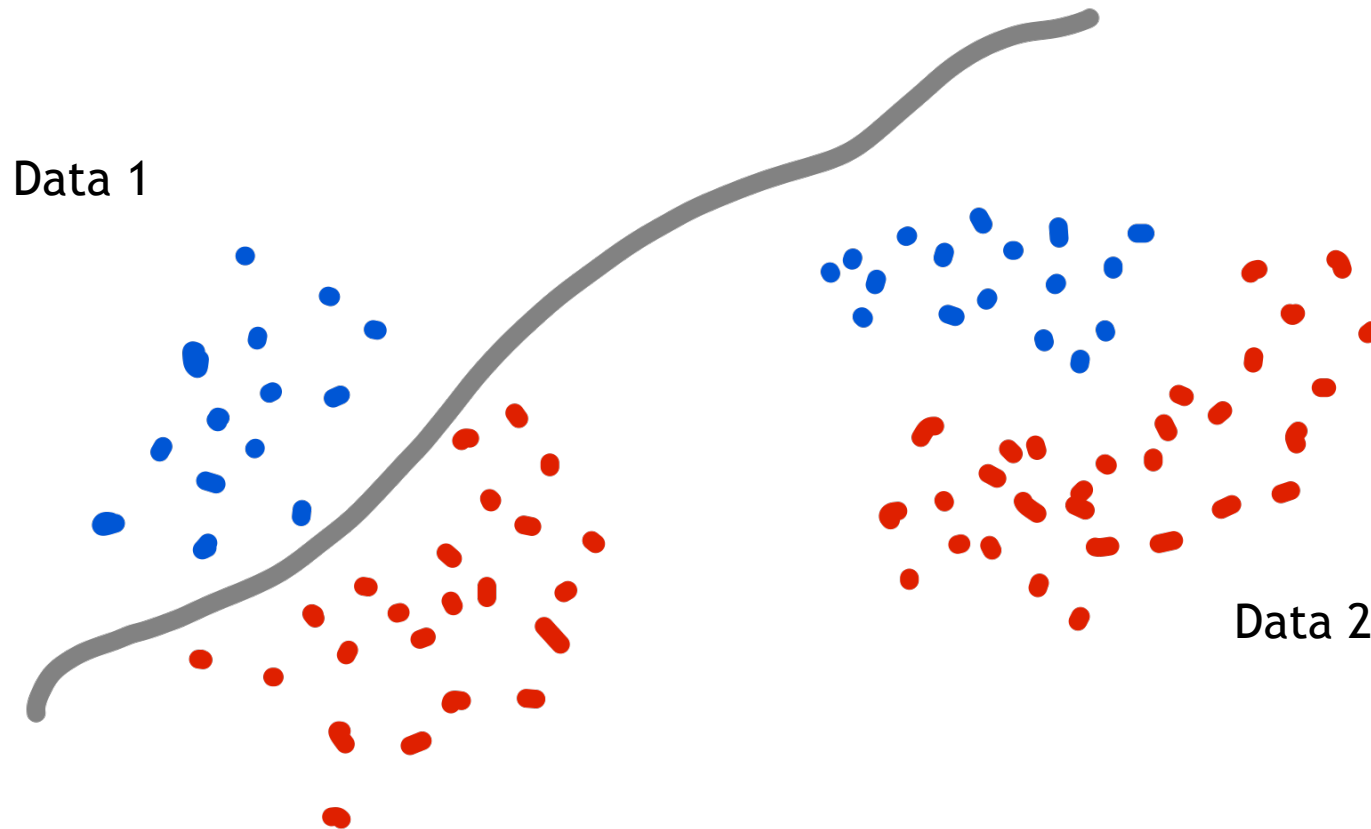
Computationally challenging. Approximations do not work well. This is an open problem!

# A Key Idea for Life-Long Learning

Data 1

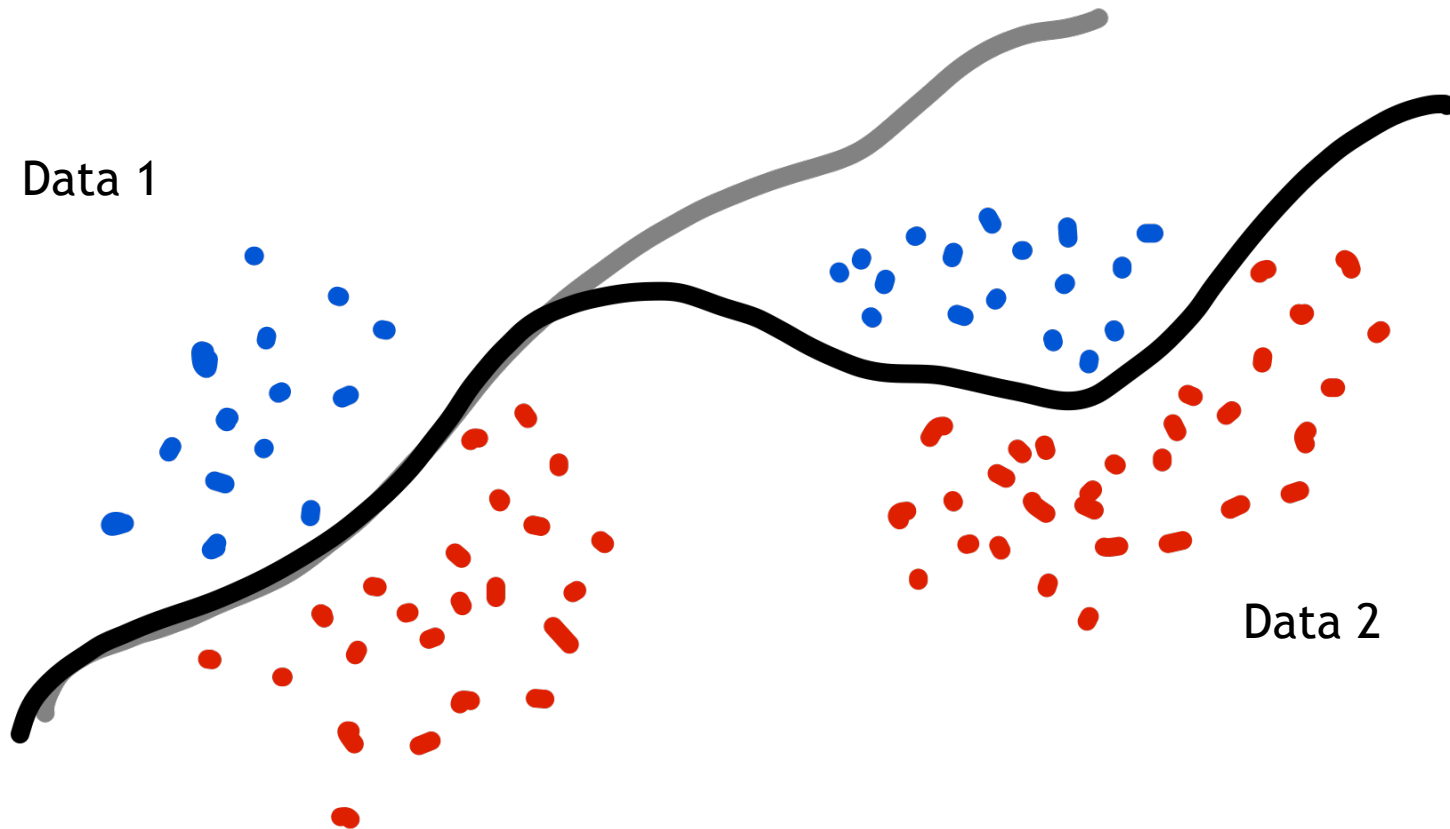


# A Key Idea for Life-Long Learning

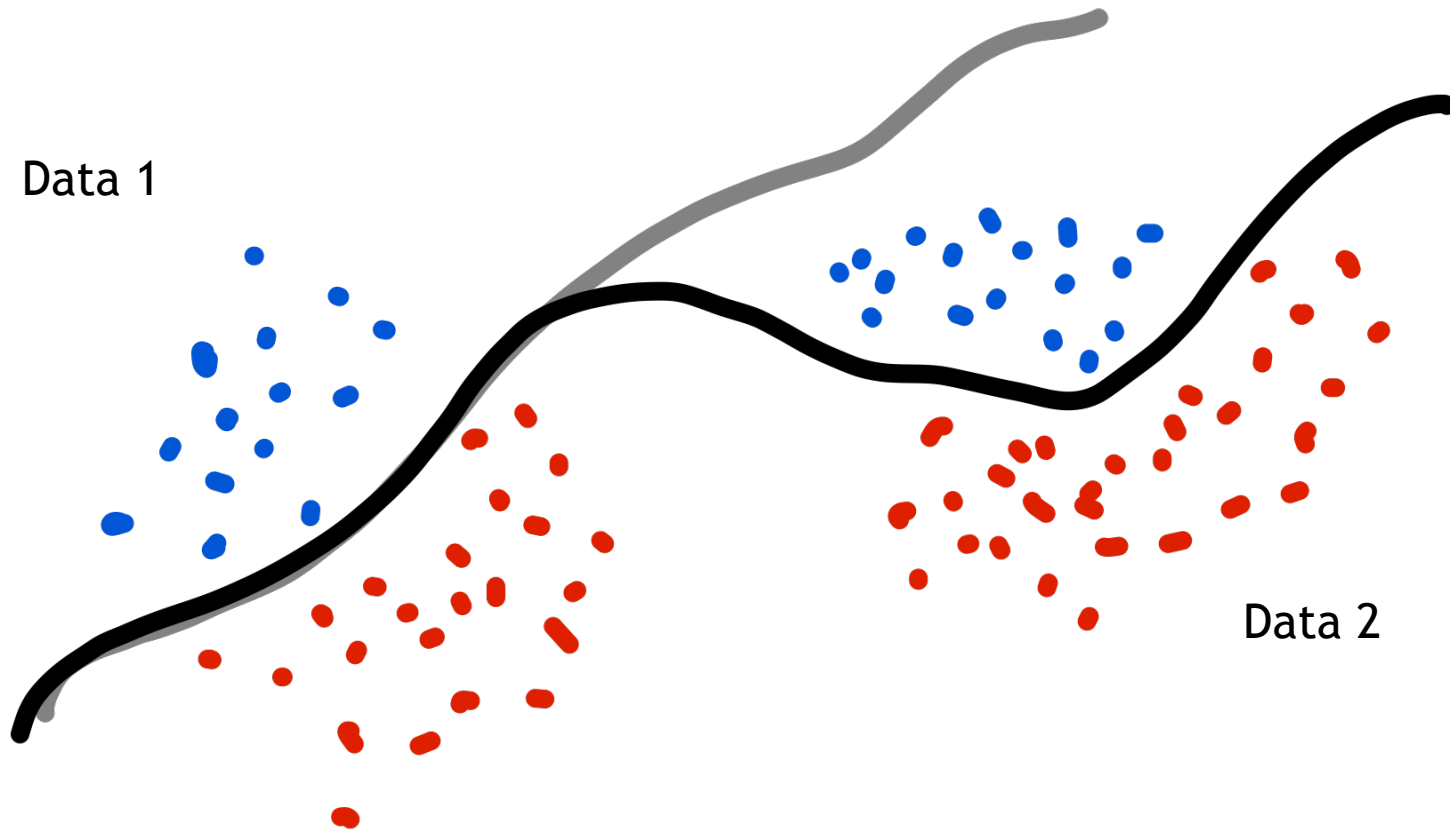




# A Key Idea for Life-Long Learning



# A Key Idea for Life-Long Learning



Change the network **weights** to match the network output (**function**) at Data 1 while classifying Data 2

# Weight-Space vs Function-Space

## Weight/parameter space

- Finite dimensional
- Easy to train
  - e.g., deep learning + SGD is scalable
- Difficult to regularize
  - Actual value of the weights do not matter
  - Complex relationship between network outputs and weights

## Function/data space

- Infinite dimensional
- Difficult to train
  - e.g. Gaussian Process is not
- Easy to regularize
  - Properties of the network outputs are easier to specify and interpret.

# Life-Long Learning with Bayesian Principles

- Connect the weight and function spaces.
  - Cheap algorithms to train in the weight space while regularizing in the function space.
- Background
  - Linear models and Gaussian Process (GP)
  - Neural Nets and GPs (**requires infinite-width nets**)
- DNN2GP
  - Convert **trained finite-widths nets** to GPs
  - Convert **the iterates of DL algorithms** to GPs
- Applications to Continual Learning

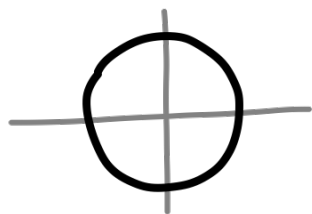
# Linear model and GPs

Gaussian prior on weights induces GP prior on functions

# Linear model and GPs

Gaussian prior on weights induces GP prior on functions

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



function  $\downarrow$   $f(x) = \phi(x)^T w$

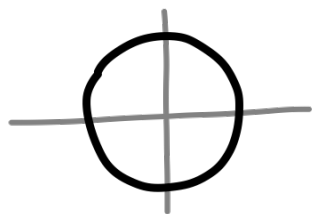
feature  $\downarrow$

weights  $\downarrow$

# Linear model and GPs

Gaussian prior on weights induces GP prior on functions

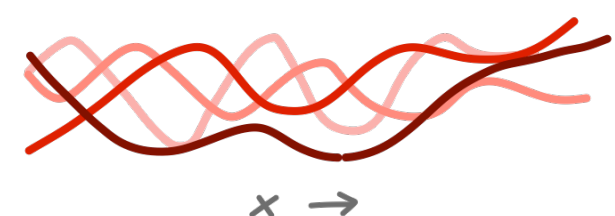
$$w \sim \mathcal{N}(0, I)$$



function  $\downarrow$  feature

$$f(x) = \phi(x)^T w$$

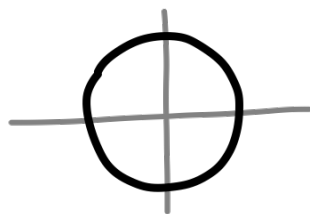
weights  $\uparrow$

$$f(x) \sim \mathcal{GP}\left(\underbrace{0}_{\text{mean}}, \underbrace{\phi(x)^T \phi(x')}_{\text{kernel } L}\right)$$


# Linear model and GPs

Gaussian prior on weights induces GP prior on functions

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



function  $\downarrow$  feature

$$f(x) = \phi(x)^T w$$

weights  $\uparrow$

$$f(x) \sim \mathcal{GP}\left(\underbrace{0}_{\text{mean}}, \underbrace{\phi(x)^T \phi(x')}_{\text{kernel } L}\right)$$
A plot showing a function  $f(x)$  as a red line, with multiple overlapping red lines representing the GP prior. The x-axis is labeled  $x \rightarrow$ .

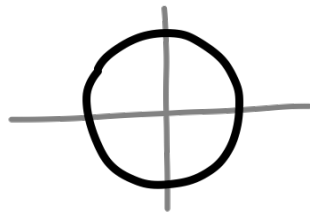
Gaussian posterior on  $w$  induces a GP posterior on  $f$



# Linear model and GPs

Gaussian prior on weights induces GP prior on functions

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

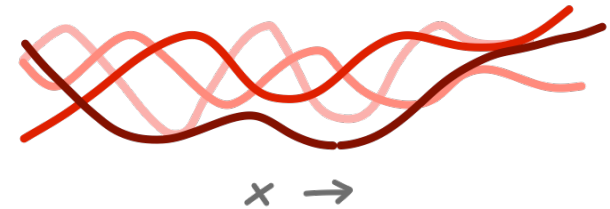


function  $\downarrow$  feature

$$f(x) = \phi(x)^T w$$

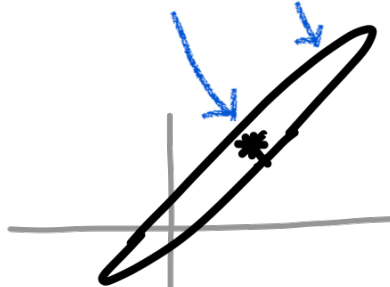
weights  $\uparrow$

$$f(x) \sim \mathcal{GP}\left(\underbrace{0}_{\text{mean}}, \underbrace{\phi(x)^T \phi(x')}_{\text{kernel } L}\right)$$



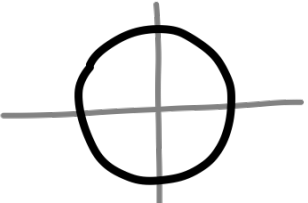
Gaussian posterior on  $w$  induces a GP posterior on  $f$

$$w \sim \mathcal{N}(w_*, \Sigma_*)$$



# Linear model and GPs

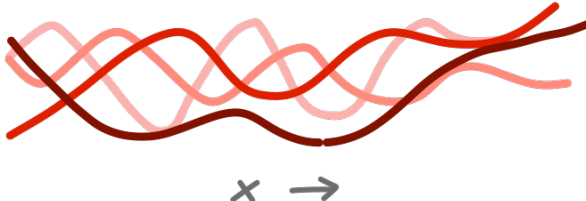
Gaussian prior on weights induces GP prior on functions

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


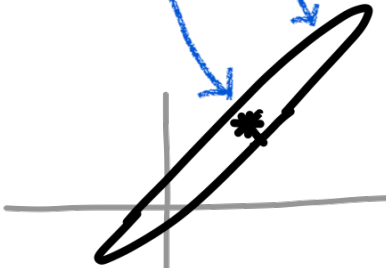
function  $\downarrow$   $f(x) = \phi(x)^T w$

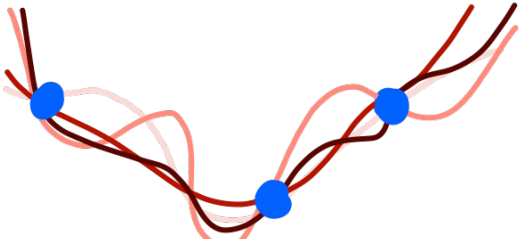
feature  $\downarrow$

weights  $\downarrow$

$$f(x) \sim \text{GP}\left(\underbrace{0}_{\text{mean}}, \underbrace{\phi(x)^T \phi(x')}_{\text{kernel } L}\right)$$


Gaussian posterior on  $w$  induces a GP posterior on  $f$

$$w \sim \mathcal{N}(w_*, \Sigma_*)$$


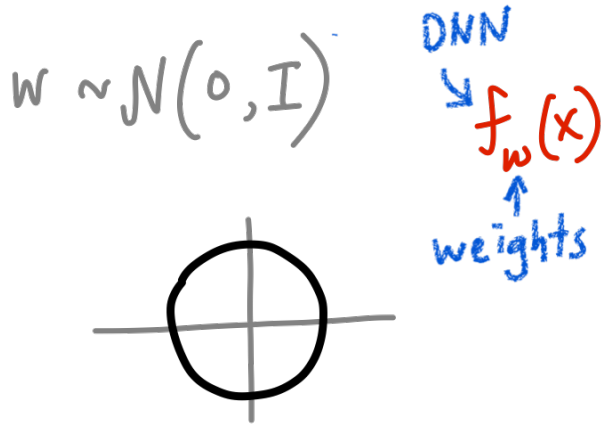
$$f(x) \sim \text{GP}\left(\phi(x)^T w_*, \phi(x)^T \Sigma_* \phi(x')\right)$$


# Deep Networks and GPs

Gaussian prior on weights induces GP prior on functions

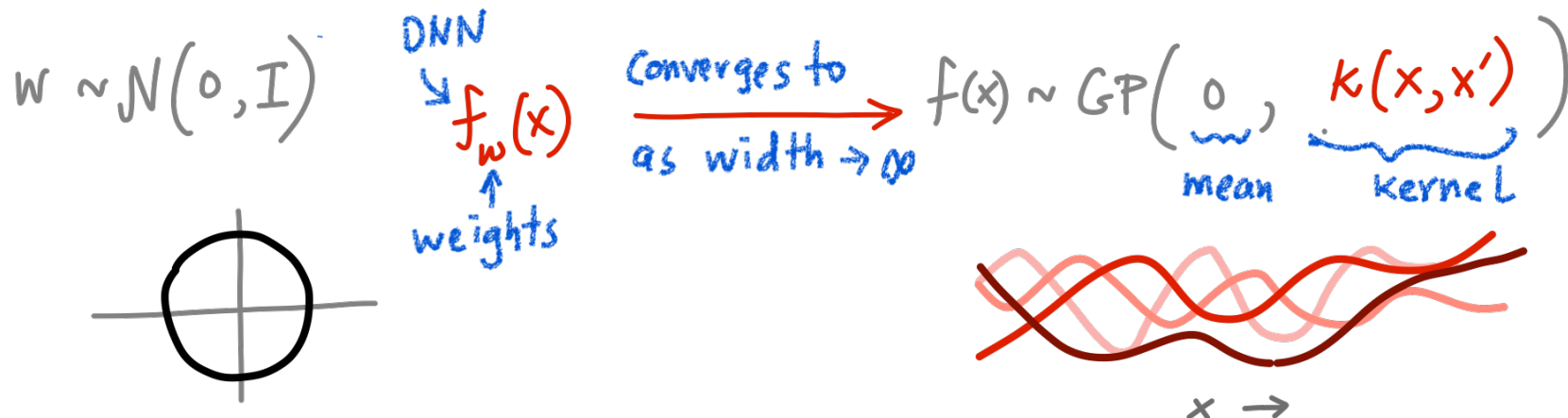
# Deep Networks and GPs

Gaussian prior on weights induces GP prior on functions



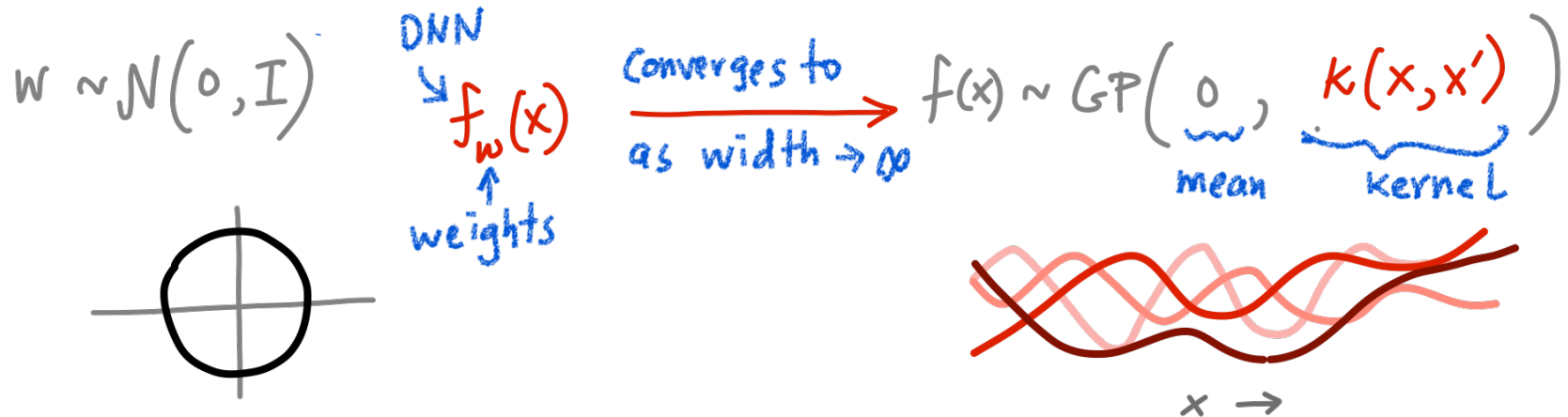
# Deep Networks and GPs

Gaussian prior on weights induces GP prior on functions



# Deep Networks and GPs

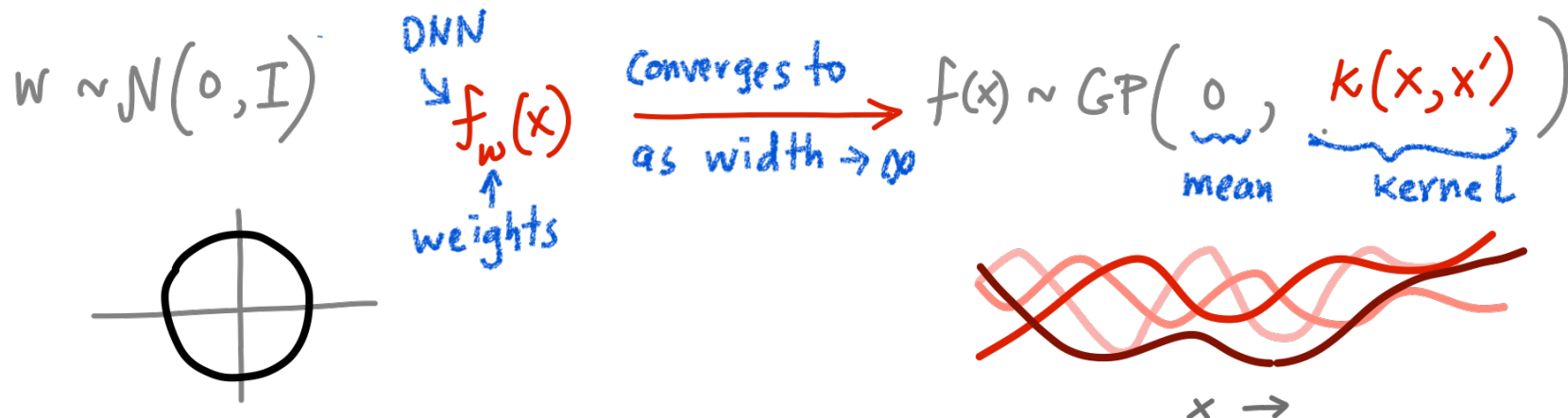
Gaussian prior on weights induces GP prior on functions



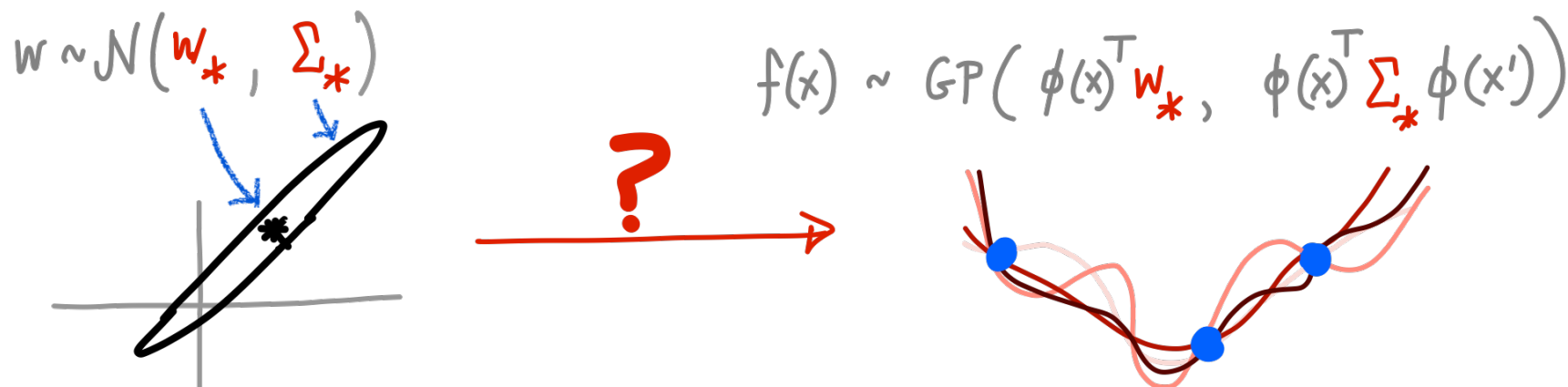
Q: Does this hold at finite width? And for posteriors?

# Deep Networks and GPs

Gaussian prior on weights induces GP prior on functions

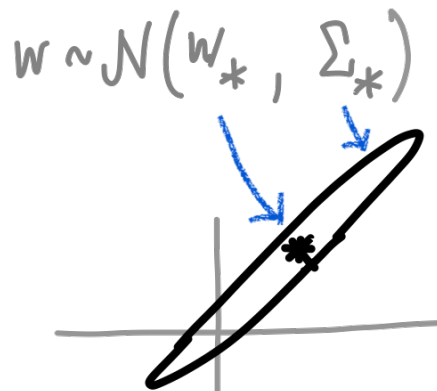
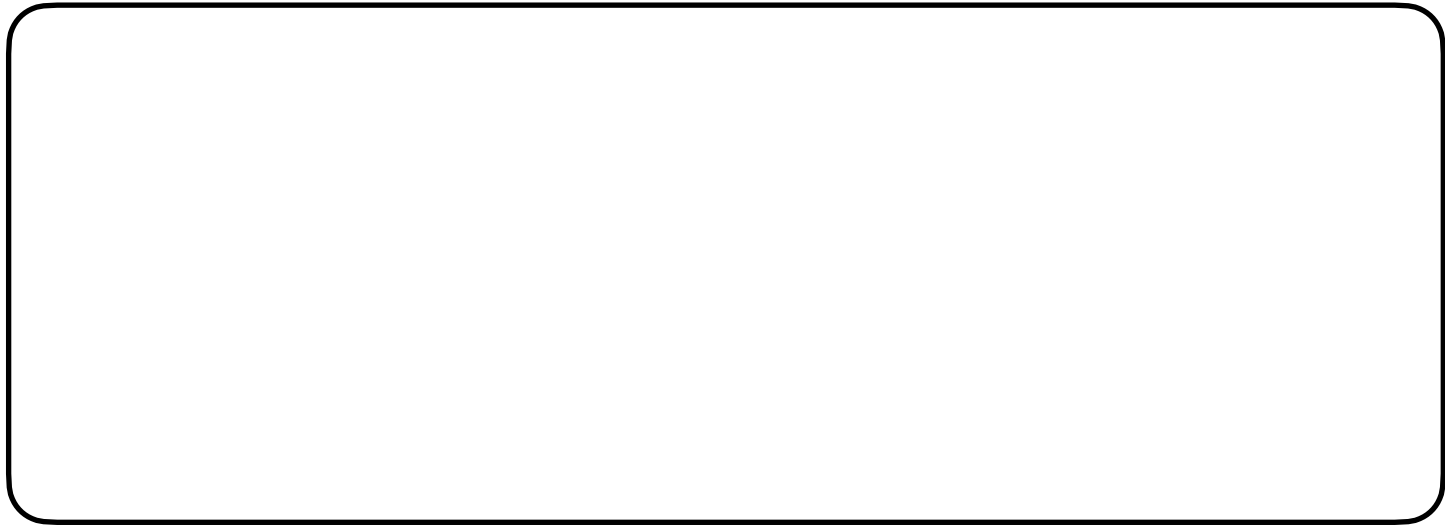


Q: Does this hold at finite width? And for posteriors?

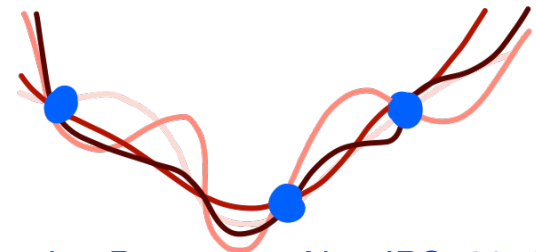


# DNN2GP for regression

Using DNN2GP, we can convert a trained network into GP



$f(x) \sim \text{GP}(f_{w_*}(x), J_{w_*}(x) \Sigma_* J_{w_*}(x'))$



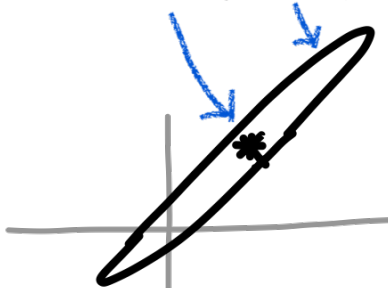


# DNN2GP for regression

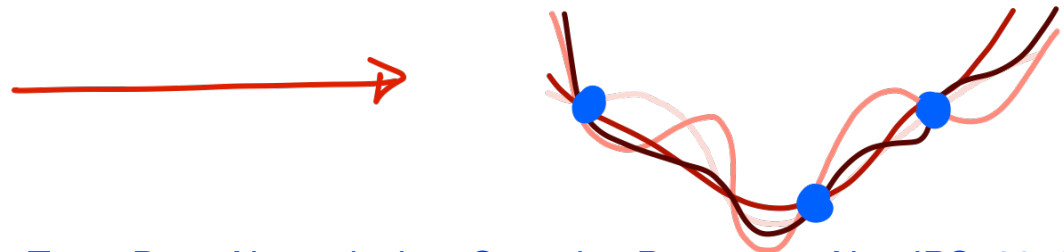
Using DNN2GP, we can convert a trained network into GP

$$w_* = \arg \min_w \underbrace{\sum_{i=1}^N (y_i - f_w(x_i))^2}_{\text{squared loss}} + \underbrace{\delta w^T w}_{L_2 \text{ prior}}$$

$$w \sim \mathcal{N}(w_*, \Sigma_*)$$



$$f(x) \sim \text{GP}(f_{w_*}(x), J_{w_*}(x) \Sigma_* J_{w_*}(x'))$$



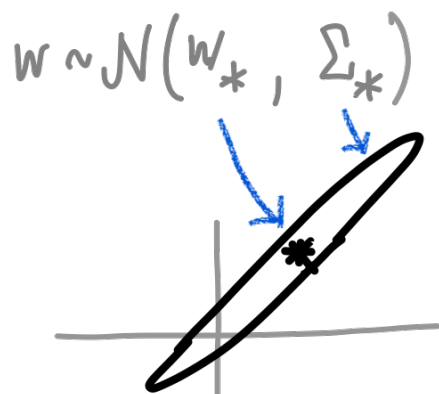
# DNN2GP for regression

Using DNN2GP, we can convert a trained network into GP

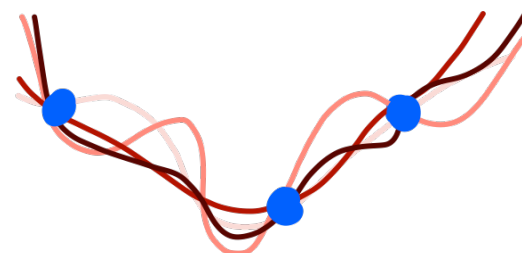
$$w_* = \arg \min_w \sum_{i=1}^N \underbrace{(y_i - f_w(x_i))^2}_{\text{squared loss}} + \underbrace{\delta w^T w}_{L_2 \text{ prior}}$$

$$\Sigma_*^{-1} = \sum_{i=1}^N \nabla_w f_{w_*}(x_i) \underbrace{\nabla_w f_{w_*}(x_i)^T}_{J_{w_*}(x_i)} + \delta I \quad \leftarrow \text{Gauss-Newton Curvature}$$

$J_{w_*}(x_i) \leftarrow \text{Jacobian}$



$$f(x) \sim \text{GP} \left( f_{w_*}(x), J_{w_*}(x) \Sigma_* J_{w_*}(x') \right)$$

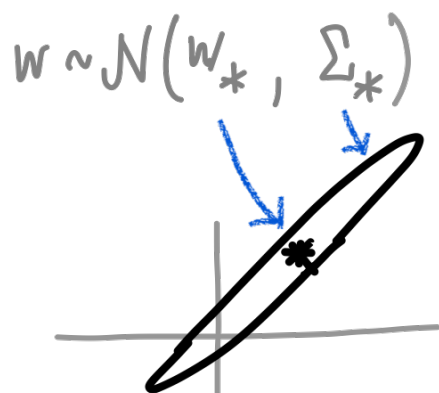


# DNN2GP Generalization

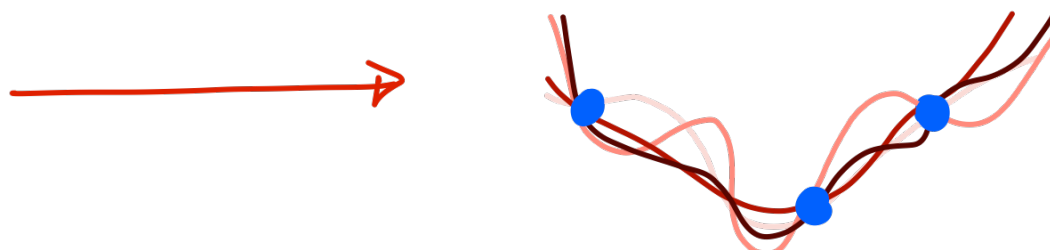
This generalizes to twice differentiable loss and priors

$$w_* = \arg \min_w \sum_{i=1}^N \underbrace{\ell(y_i, \underbrace{\sigma(f_w(x_i))}_{\text{Link function}})}_{\text{Diff- Loss}} + \underbrace{\delta R(w)}_{\text{Convex prior}}$$

$$\Sigma_*^{-1} = \sum_{i=1}^N \nabla_w f_{w_*}(x_i) \underbrace{\nabla_{ff}^2 \ell(\cdot)}_{\Lambda_{w_*}(x)} \nabla_w f_{w_*}(x_i)^T + \delta \nabla_{ww}^2 R(w) \leftarrow \begin{array}{l} \text{Generalized} \\ \text{Gauss-Newton} \\ \text{Curvature} \end{array}$$



$$f(x) \sim \text{GP}(\underbrace{\sigma(f_{w_*}(x))}_{\lambda_{w_*}(x)}, \underbrace{\lambda_{w_*}(x)}_{\lambda_{w_*}(x)}^T \underbrace{J_{w_*}(x)}_{J_{w_*}(x)} \underbrace{\Sigma_*}_{\Sigma_*} \underbrace{J_{w_*}(x')^T}_{J_{w_*}(x')^T} \underbrace{\lambda_{w_*}(x')}_{\lambda_{w_*}(x')})$$



# Deep Learning as GP inference

Iterations of algorithms too can be written as GP inference

$$\begin{aligned} w_{t+1} &\leftarrow w_t - \rho \underbrace{(S_t + \delta I)^{-1}}_{\Sigma_x} \left[ \sum_{i \in \mathcal{M}} g_i \right] \leftarrow \text{minibatch gradient} \\ \text{Scale matrix} \rightarrow S_{t+1} &\leftarrow (1-\rho) S_t + \sum_{i \in \mathcal{M}} J_{w_t}(x_i) J_{w_t}(x_i)^T \leftarrow \text{minibatch GN Curvature} \end{aligned}$$

# Deep Learning as GP inference

Iterations of algorithms too can be written as GP inference

$$\begin{aligned} w_{t+1} &\leftarrow w_t - \rho \underbrace{(S_t + \delta I)^{-1}}_{\Sigma_x} \left[ \sum_{i \in \mathcal{M}} g_i \right] \leftarrow \text{minibatch gradient} \\ \text{Scale matrix} \rightarrow S_{t+1} &\leftarrow (1-\rho) S_t + \sum_{i \in \mathcal{M}} J_{w_t}(x_i) J_{w_t}(x_i)^T \leftarrow \text{minibatch GN Curvature} \end{aligned}$$

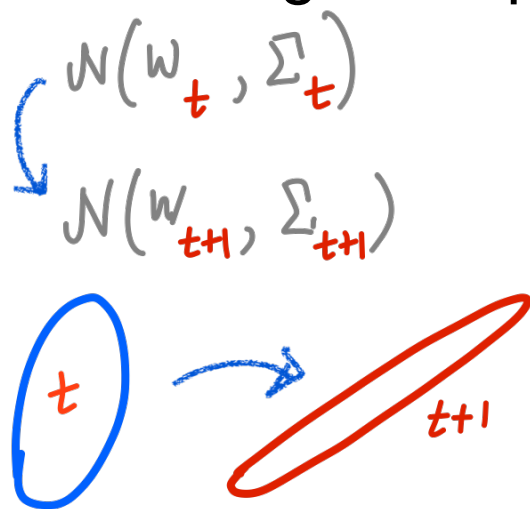
Training in  $w$  space induces a sequence in  $f$  space

# Deep Learning as GP inference

Iterations of algorithms too can be written as GP inference

$$\begin{aligned}
 w_{t+1} &\leftarrow w_t - \rho \underbrace{(S_t + \delta I)^{-1}}_{\Sigma_x} \left[ \sum_{i \in \mathcal{M}} g_i \right] \leftarrow \text{minibatch gradient} \\
 \text{Scale matrix} \rightarrow S_{t+1} &\leftarrow (1-\rho) S_t + \sum_{i \in \mathcal{M}} J_{w_t}(x_i) J_{w_t}(x_i)^T \leftarrow \text{minibatch GN Curvature}
 \end{aligned}$$

Training in  $w$  space induces a sequence in  $f$  space

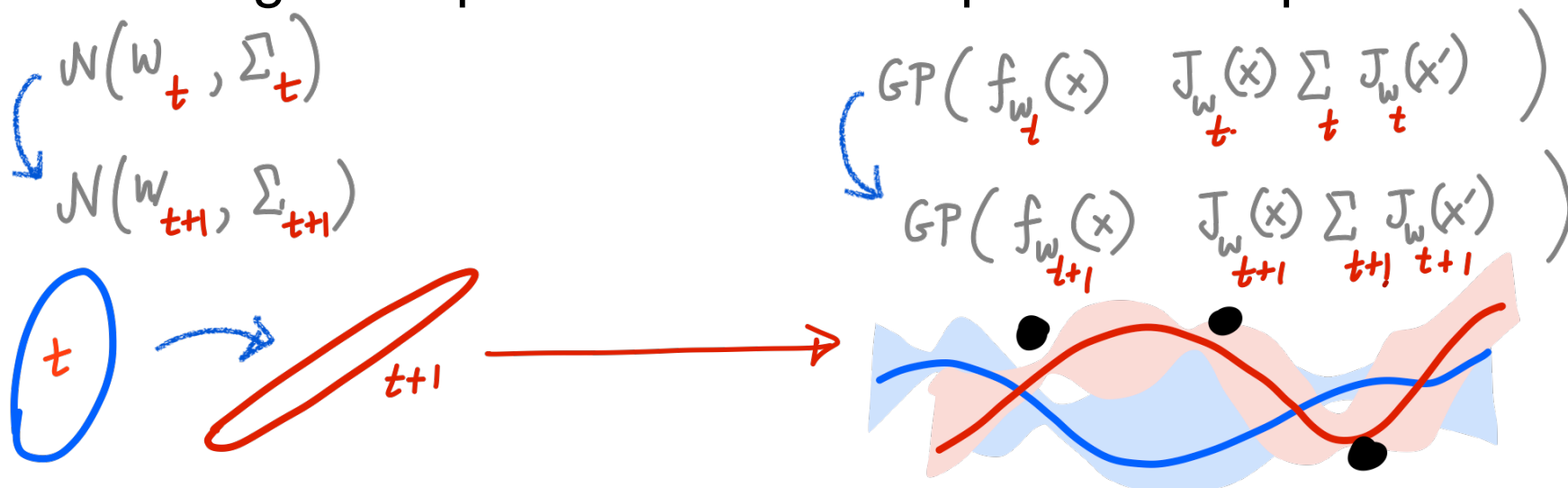


# Deep Learning as GP inference

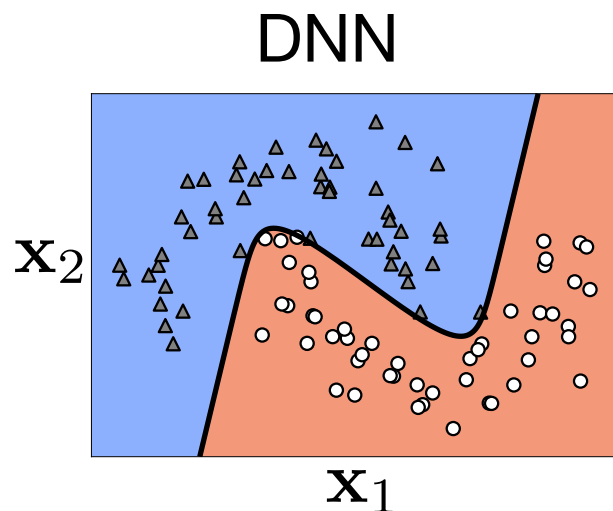
Iterations of algorithms too can be written as GP inference

$$\begin{aligned}
 w_{t+1} &\leftarrow w_t - \rho \underbrace{(S_t + \delta I)^{-1}}_{\Sigma_x} \left[ \sum_{i \in \mathcal{M}} g_i \right] \leftarrow \text{minibatch gradient} \\
 \text{Scale matrix} \rightarrow S_{t+1} &\leftarrow (1-\rho) S_t + \sum_{i \in \mathcal{M}} J_{w_t}(x_i) J_{w_t}(x_i)^T \leftarrow \text{minibatch GN Curvature}
 \end{aligned}$$

Training in  $w$  space induces a sequence in  $f$  space

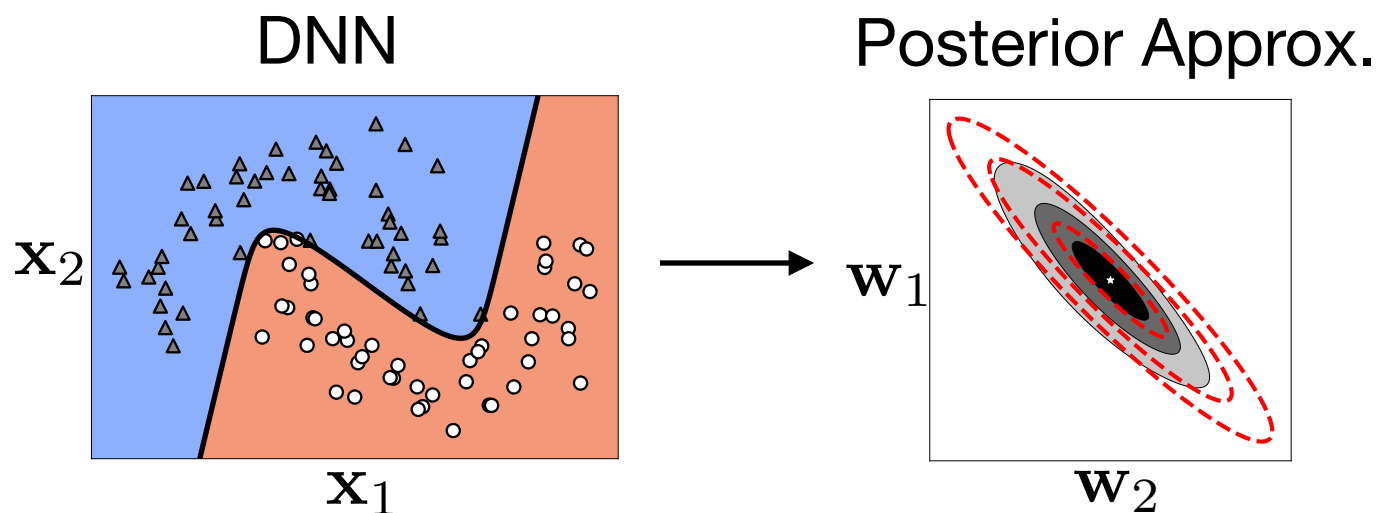


# Outline of the derivation

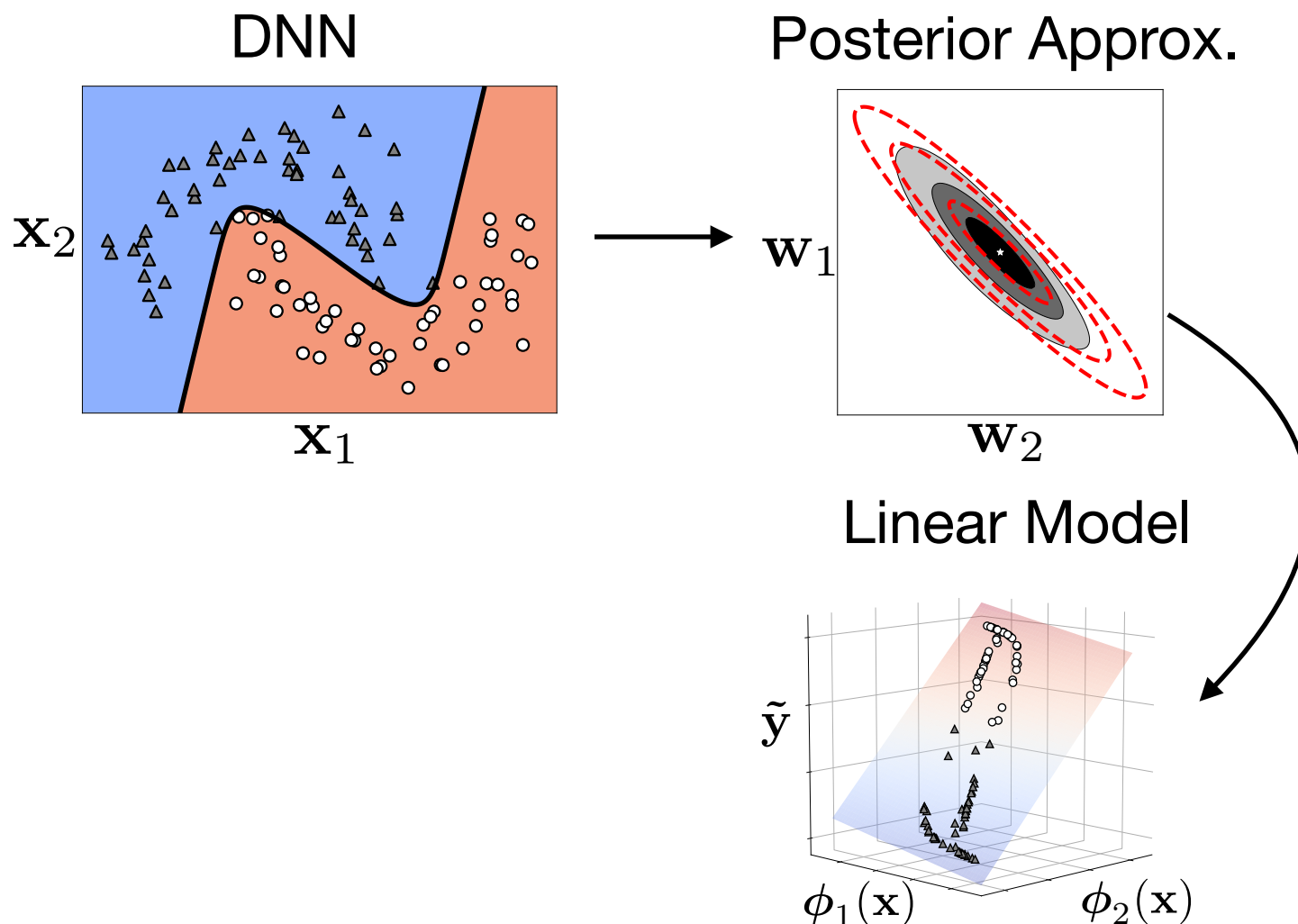




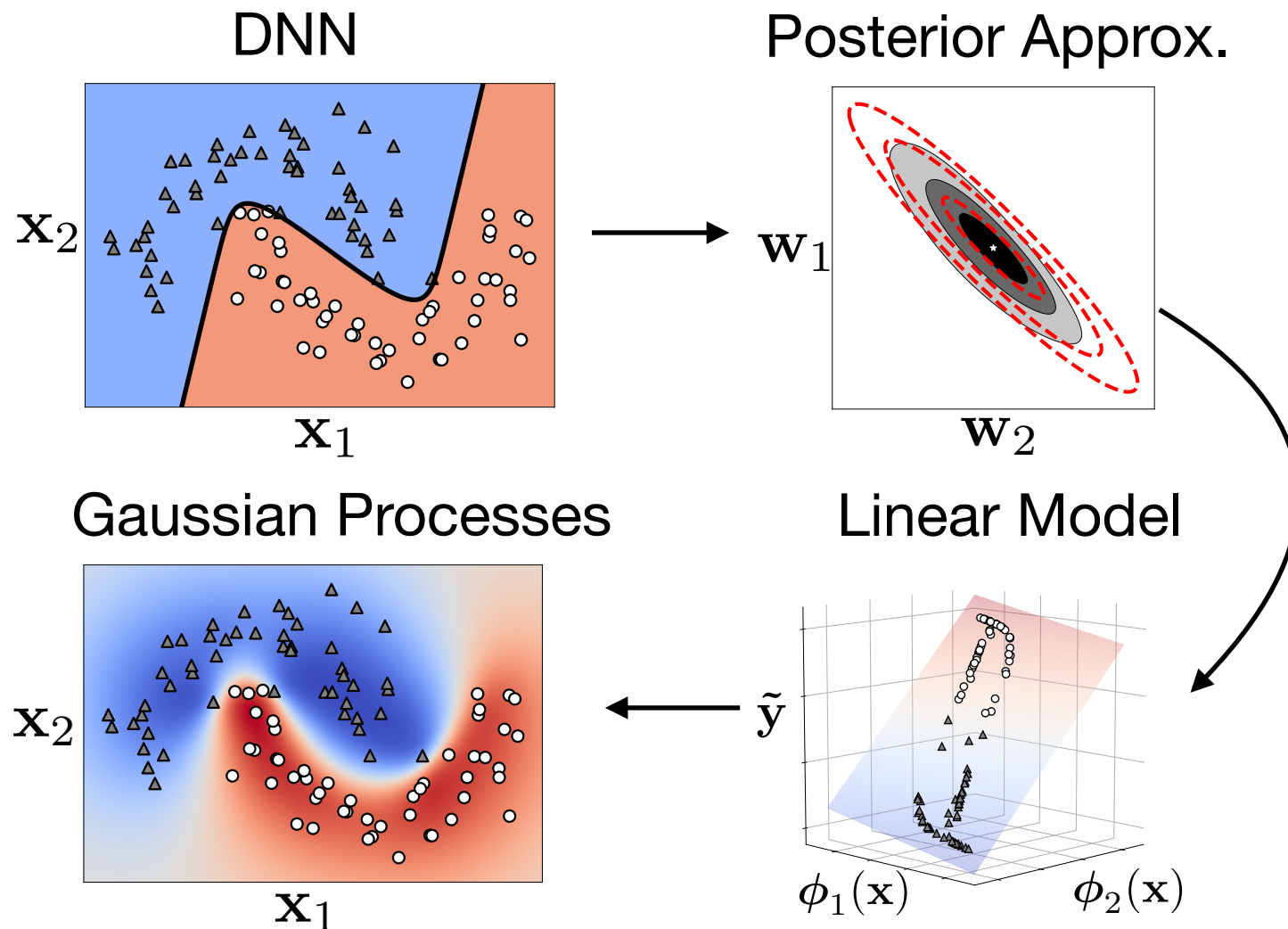
# Outline of the derivation



# Outline of the derivation



# Outline of the derivation

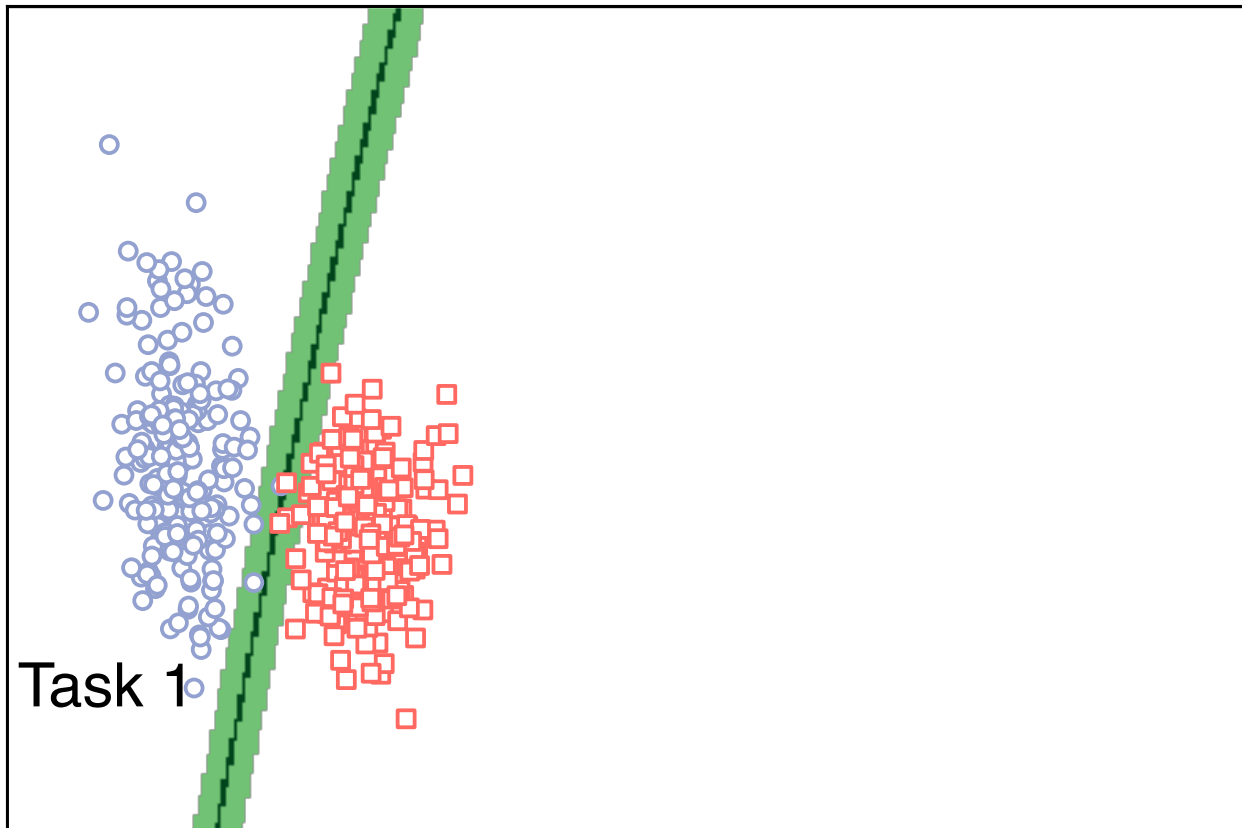


# Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP

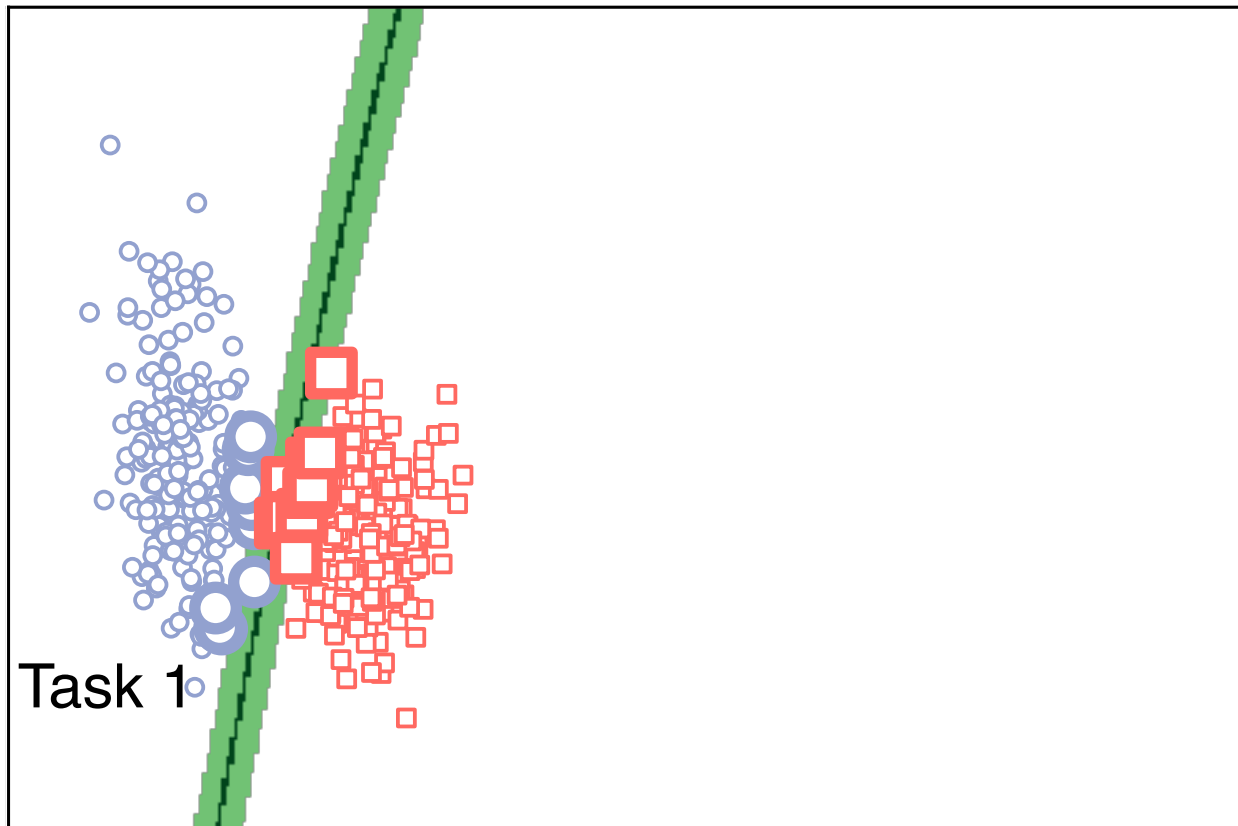
# Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP



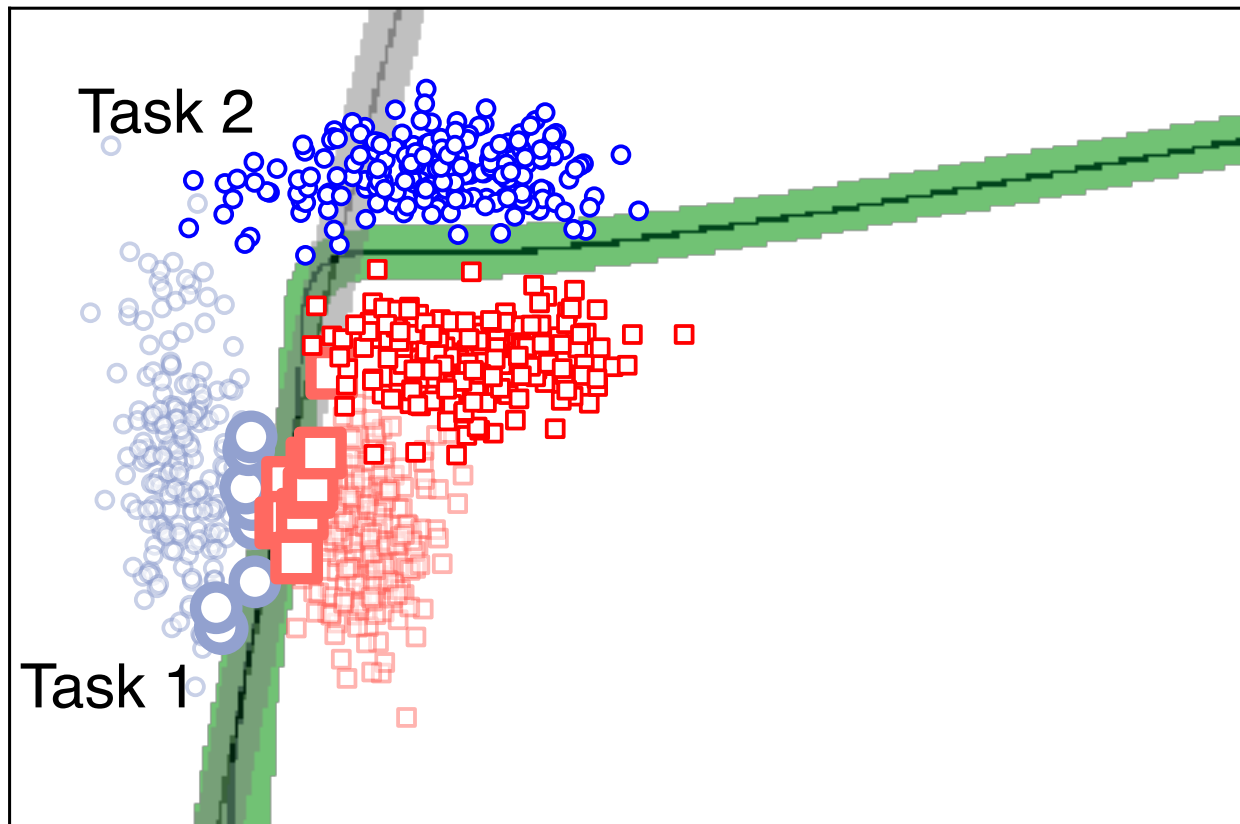
# Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP



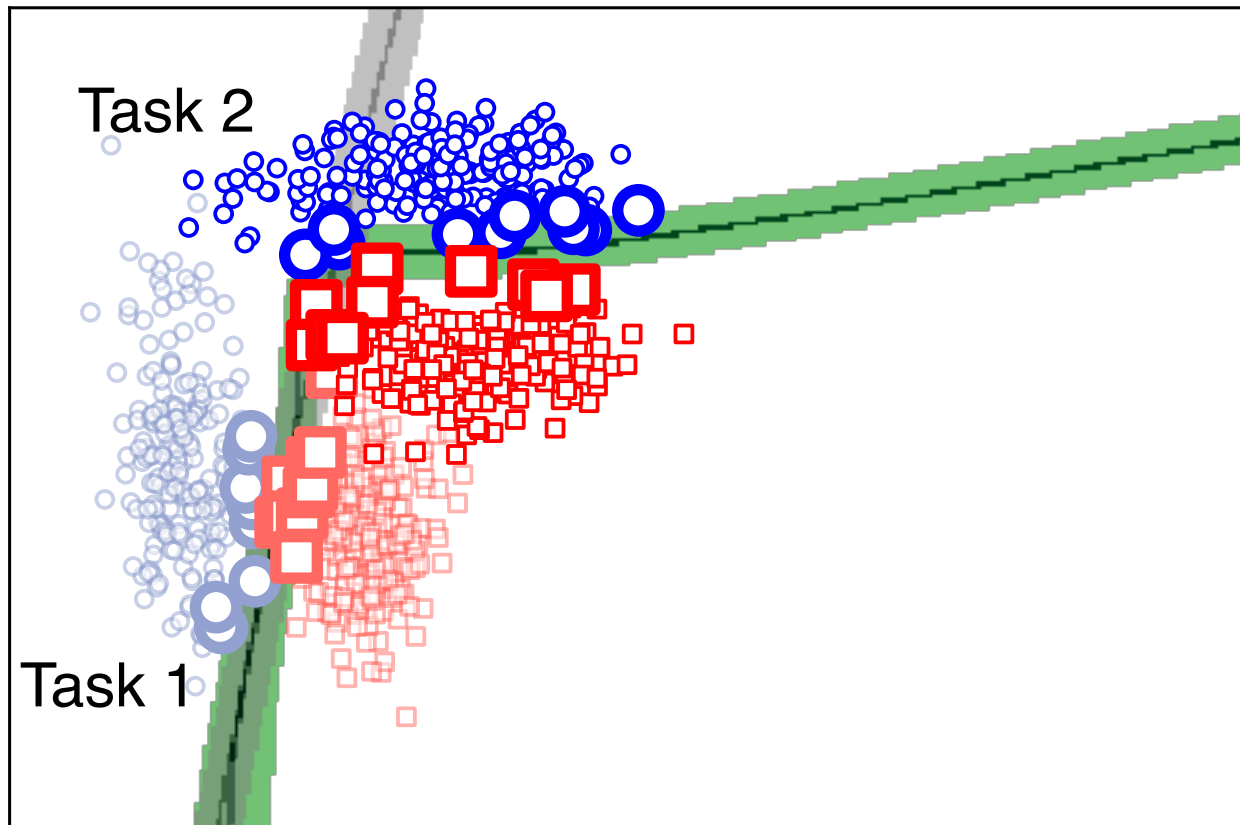
# Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP



# Functional Regularization of Memorable Past (FROMP)

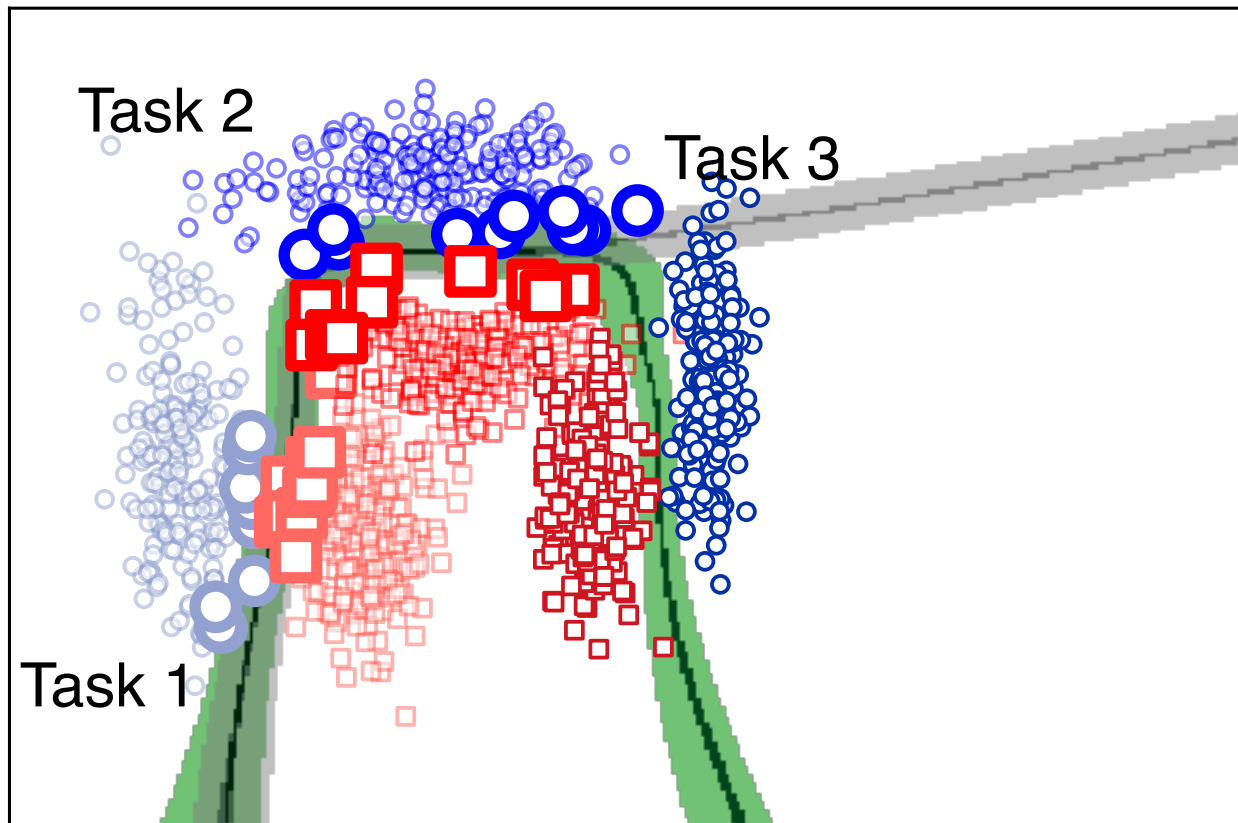
Identify, memorize, and regularize the past obtained using DNN2GP





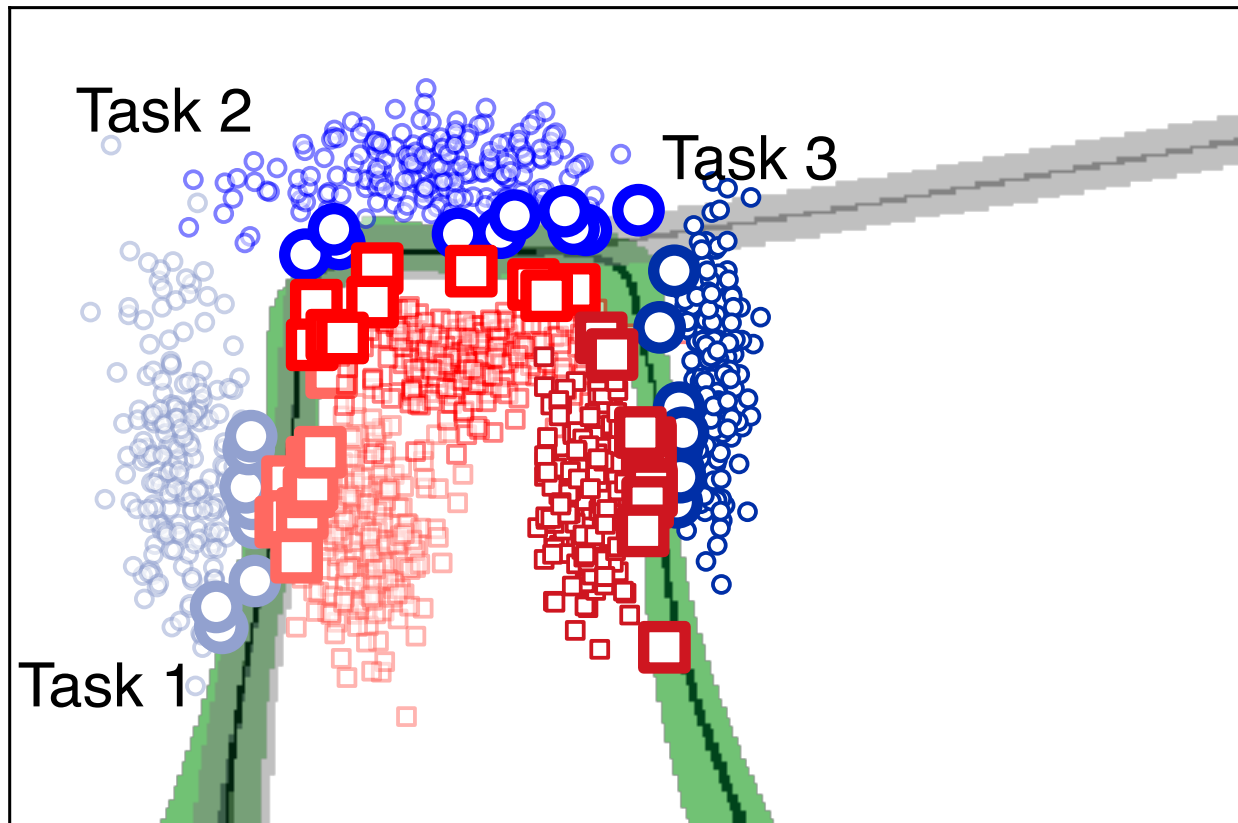
# Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP



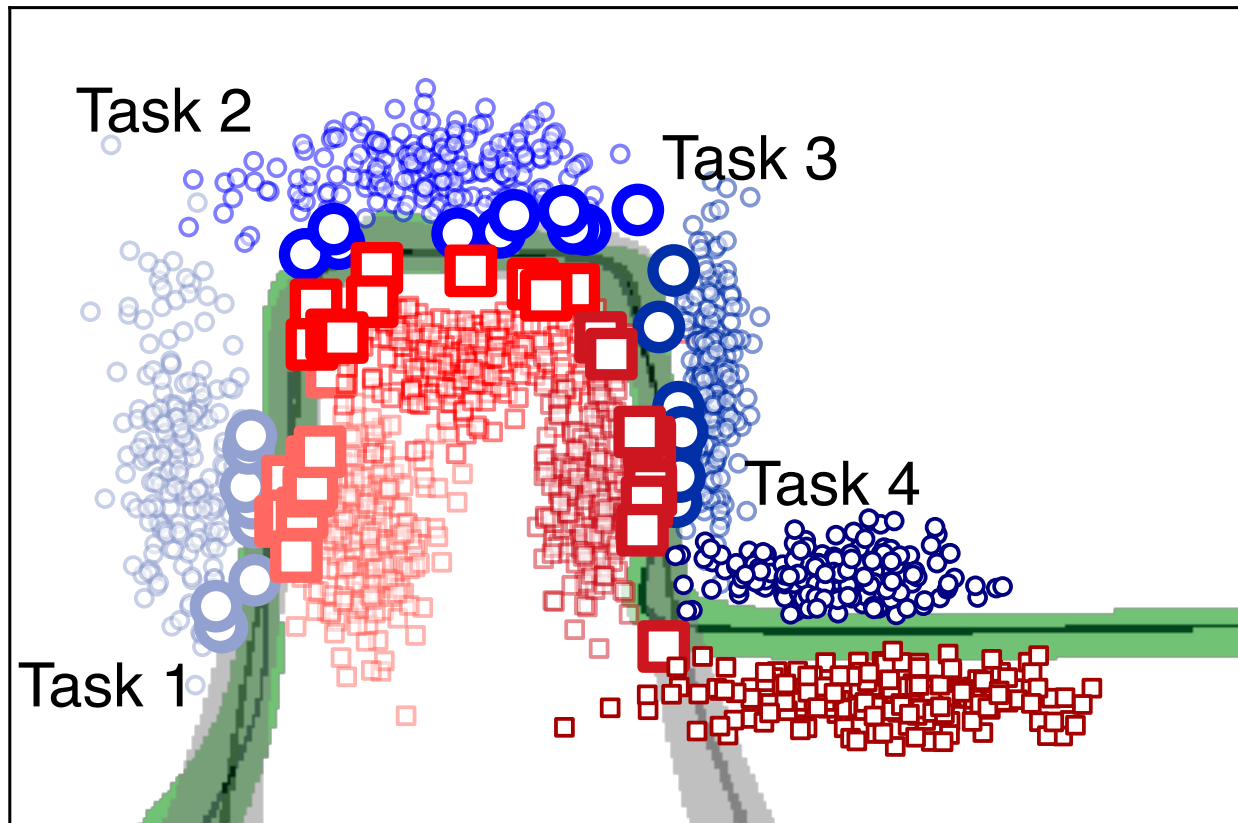
# Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP



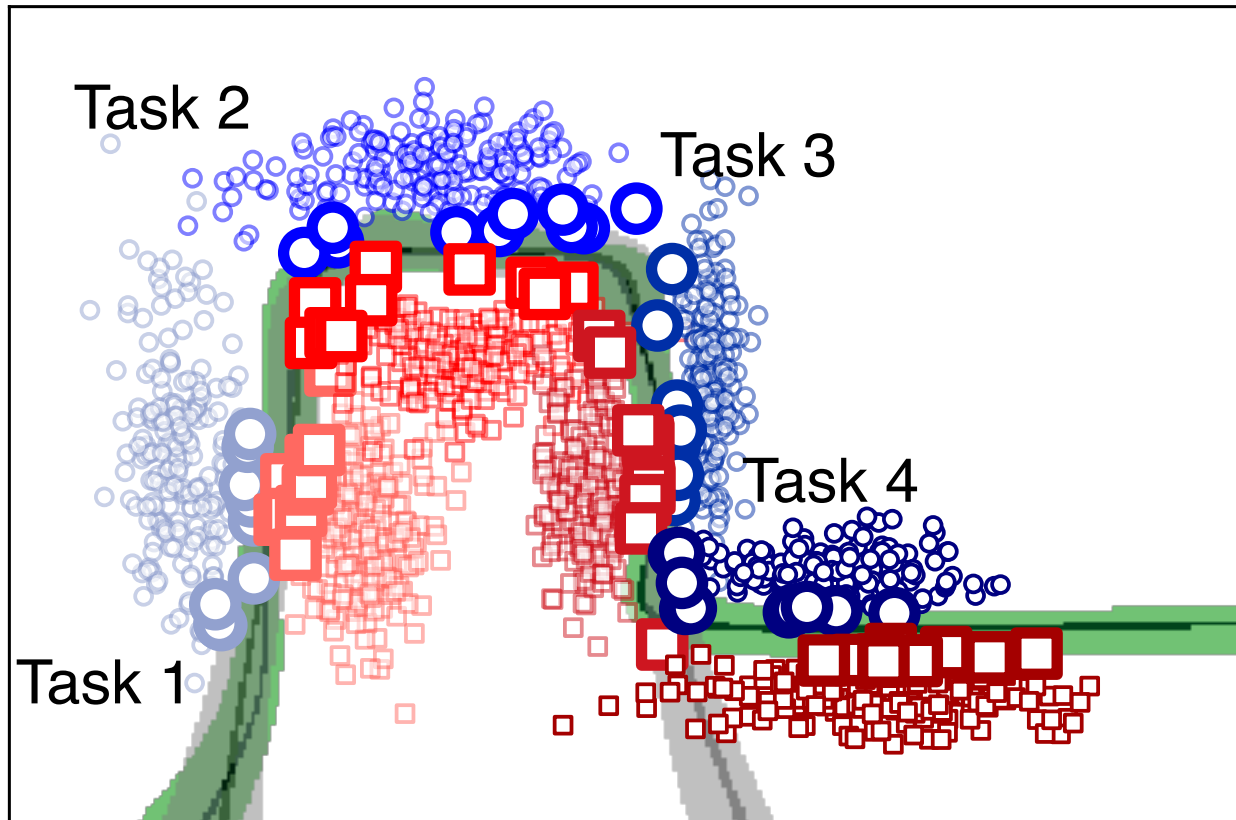
# Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP



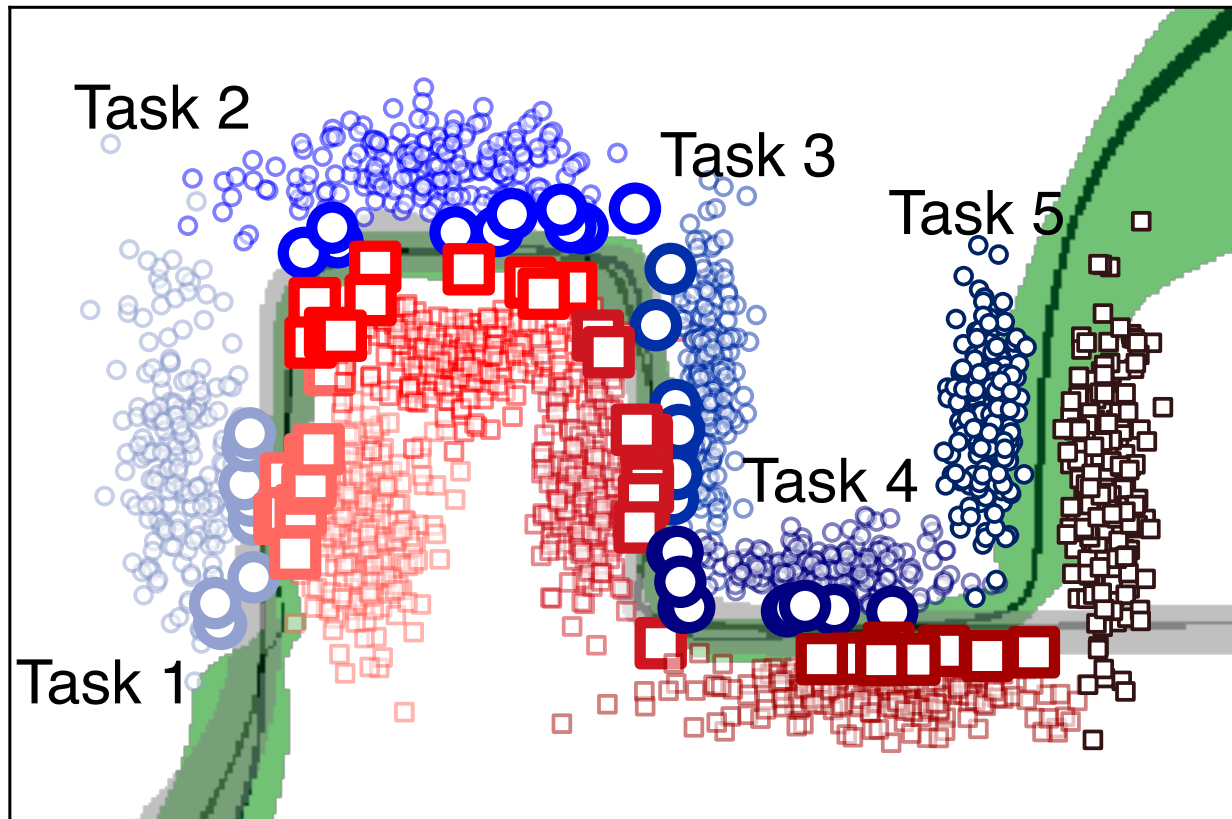
# Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP



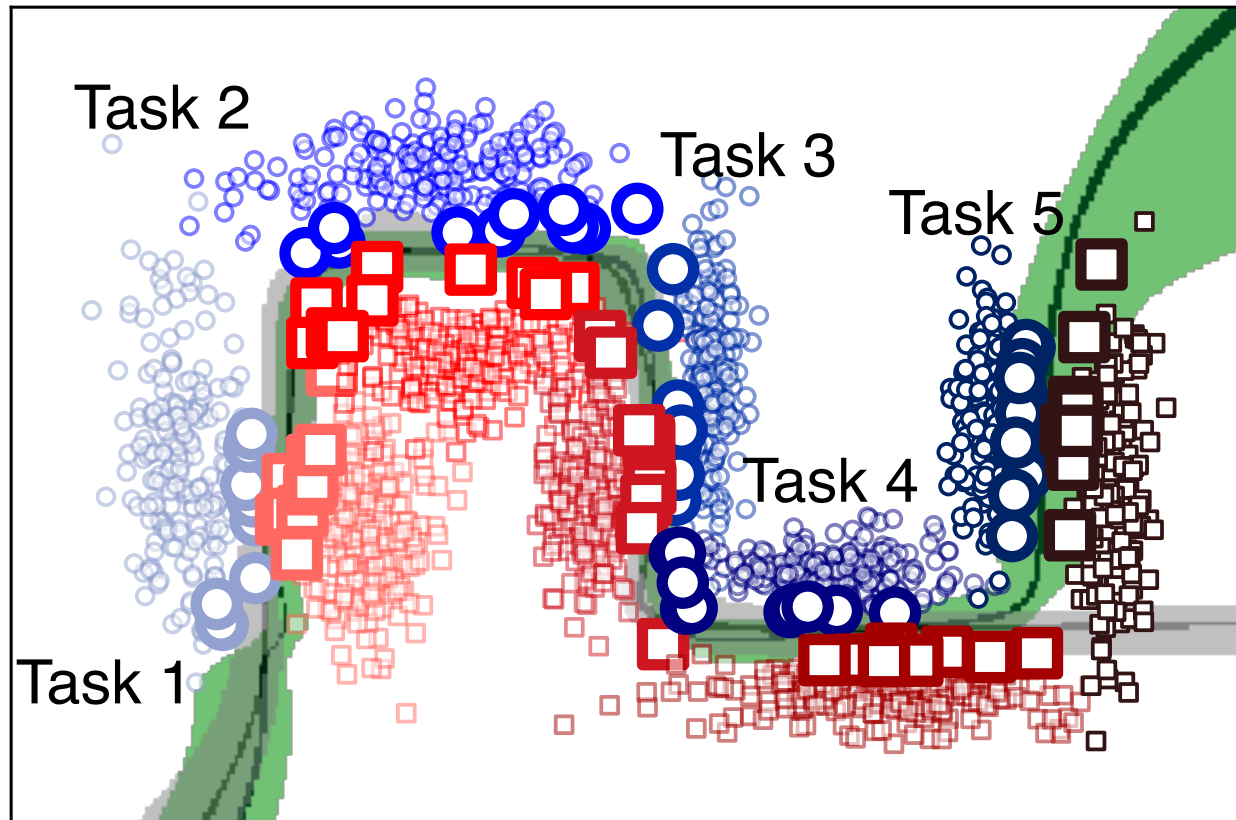
# Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP



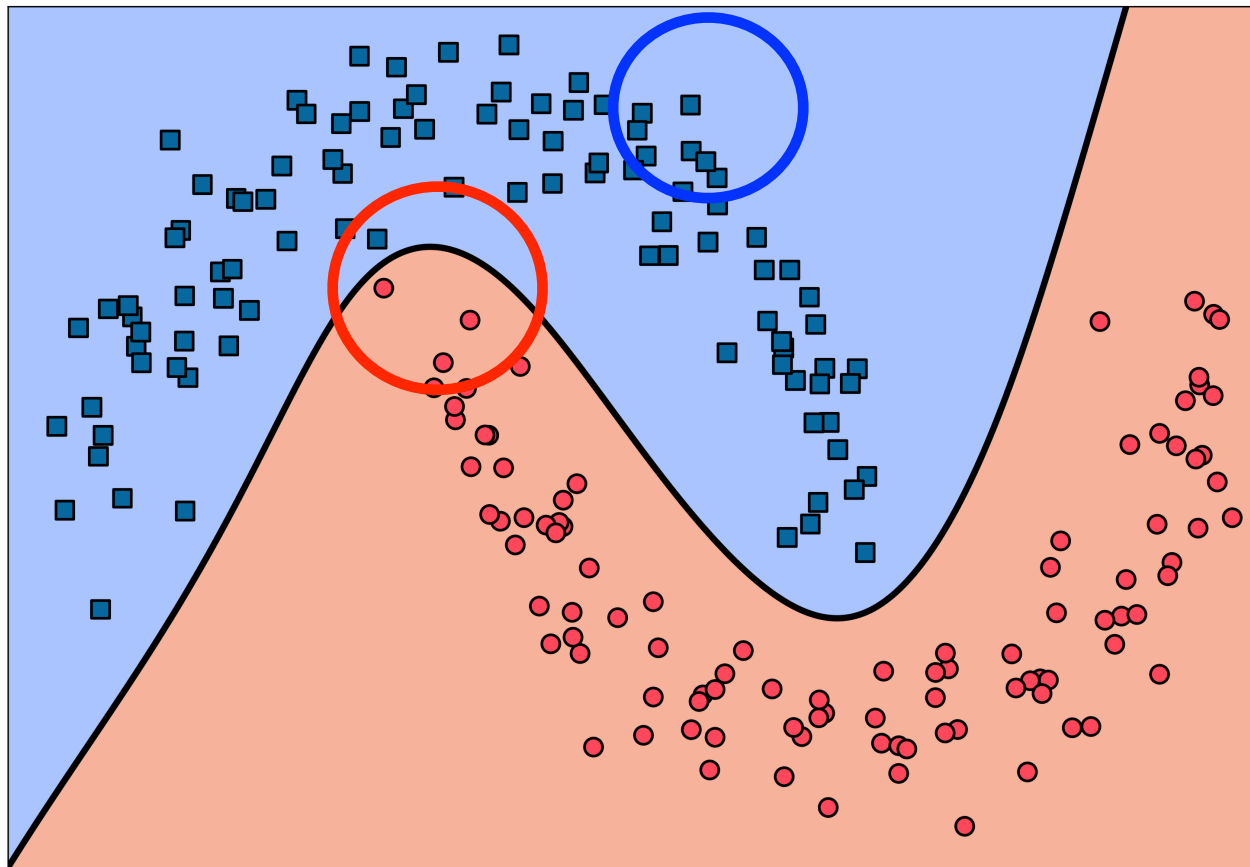
# Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP



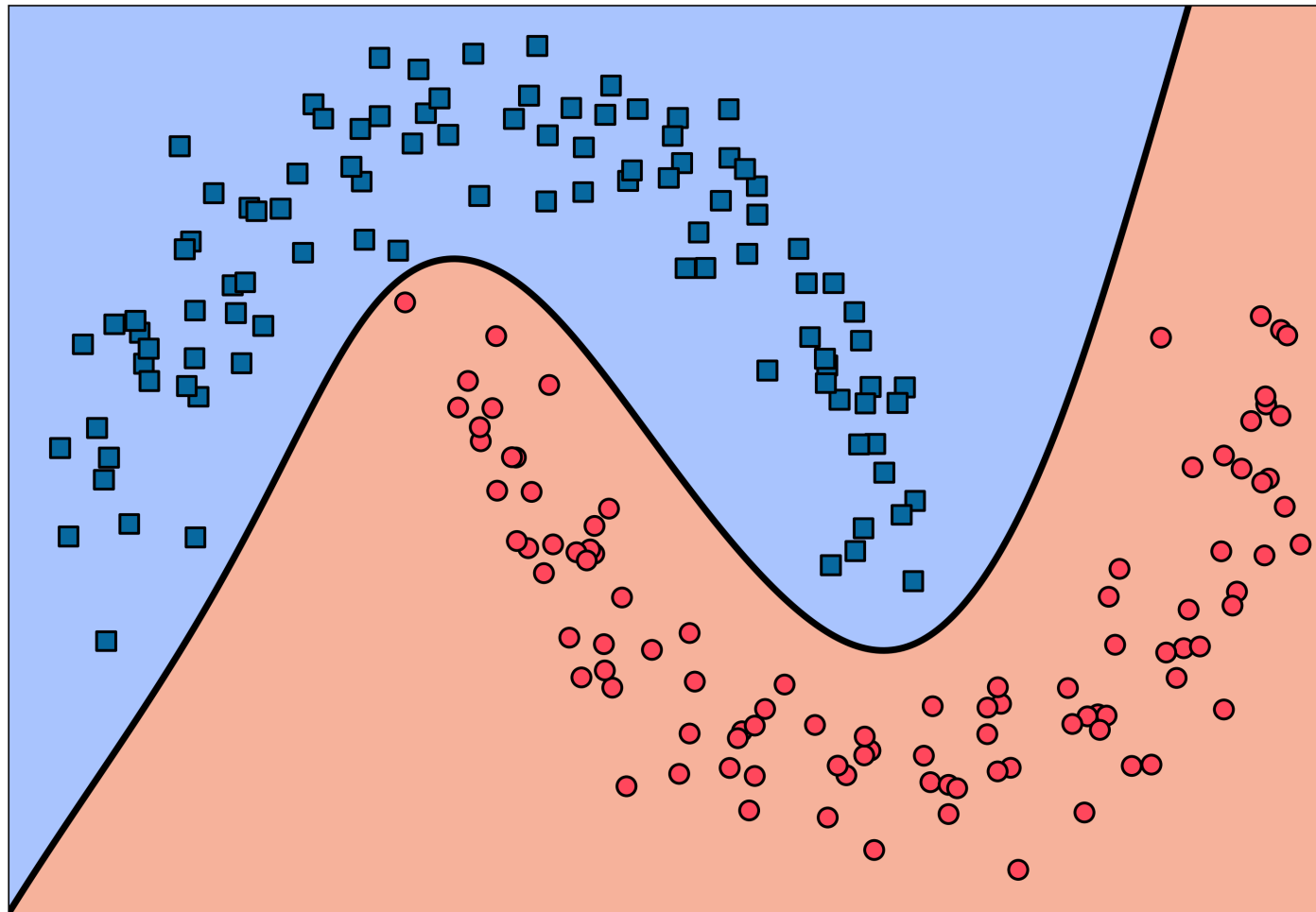
# Memorable Past

Which examples are most relevant for the classifier? Red circle vs Blue circle.



# Model view vs Data view

DNN2GP provides a measure of relevance

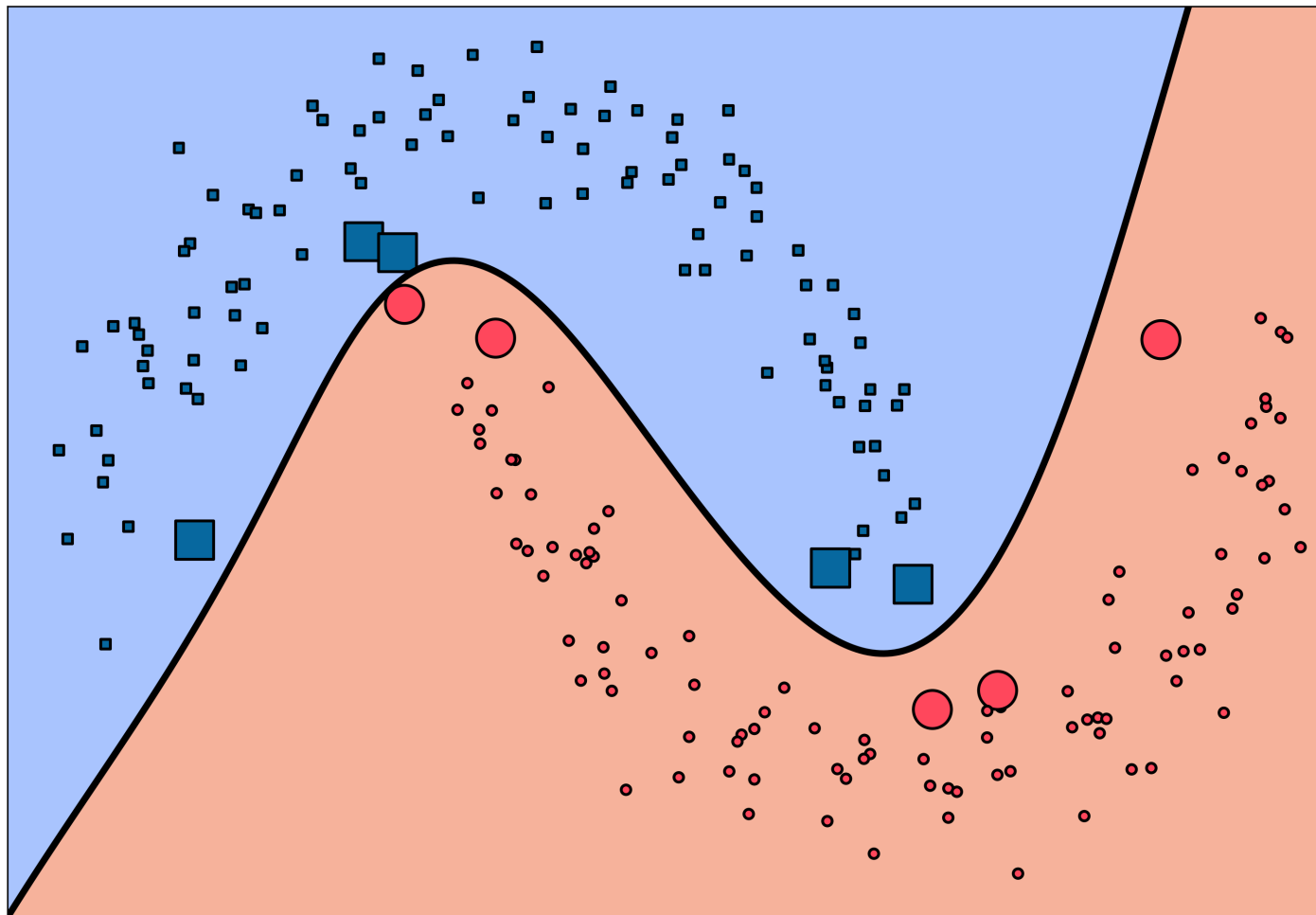


Model  
view



# Model view vs Data view

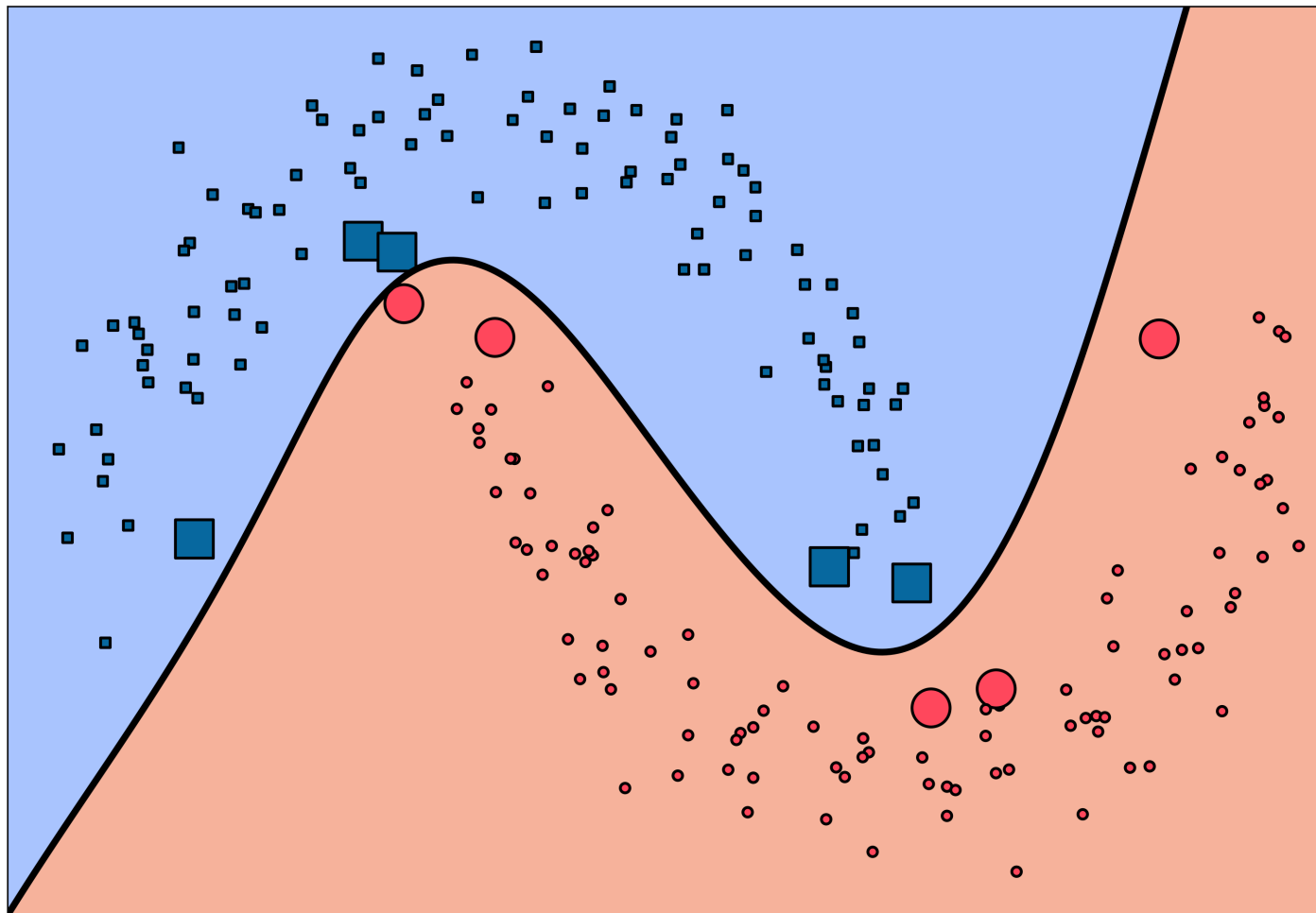
DNN2GP provides a measure of relevance



Data  
view

# Model view vs Data view

DNN2GP provides a measure of relevance

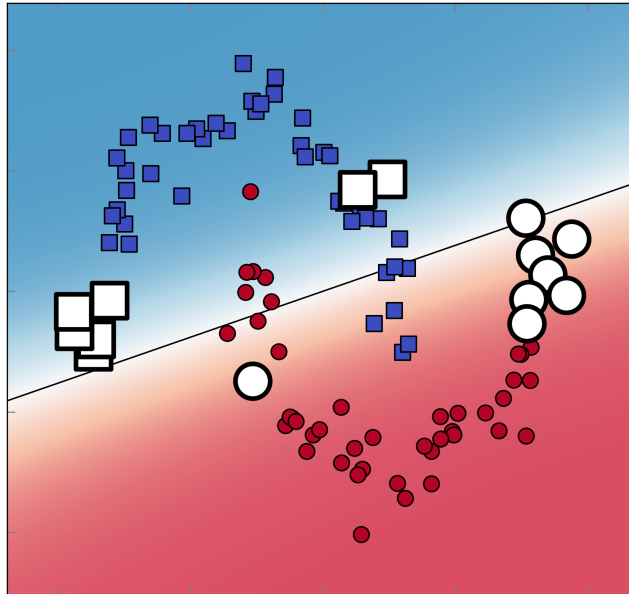


Data  
view

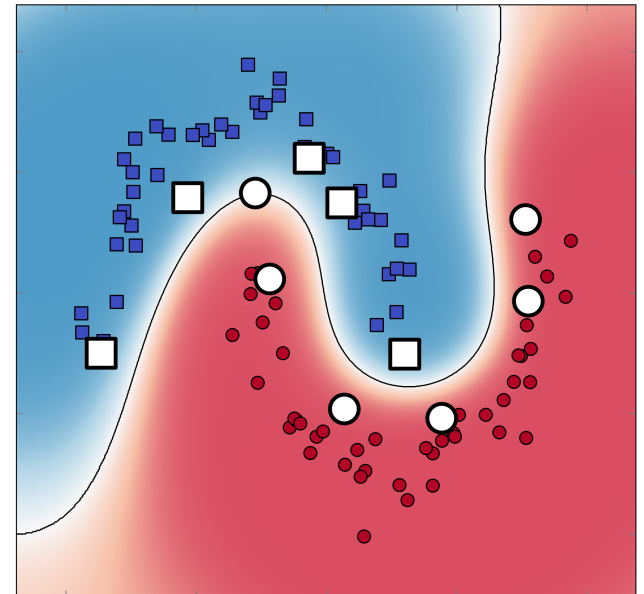
Sort  
 $\pi_i(x)$   
 $= \nabla_{ff}^2 \ell(y_i, f(x))$

# Memorable Experiences of Learning Machines

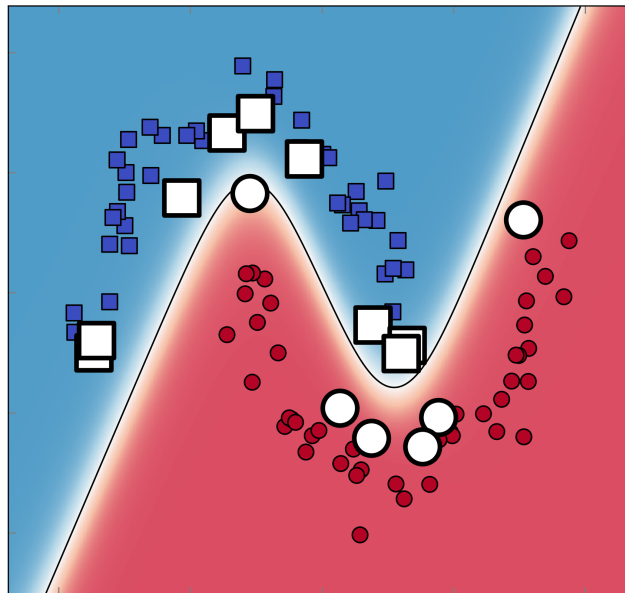
## Bayes Logistic Reg



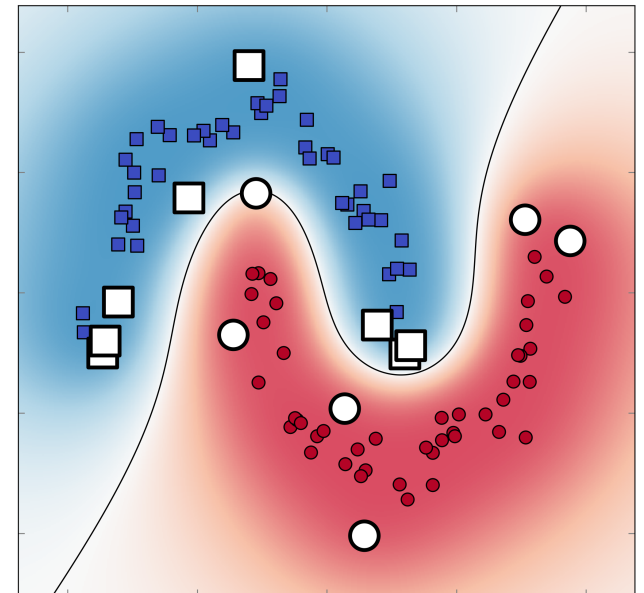
## Support Vector Machine



## Neural Network



## Gaussian Process



# Least Relevant

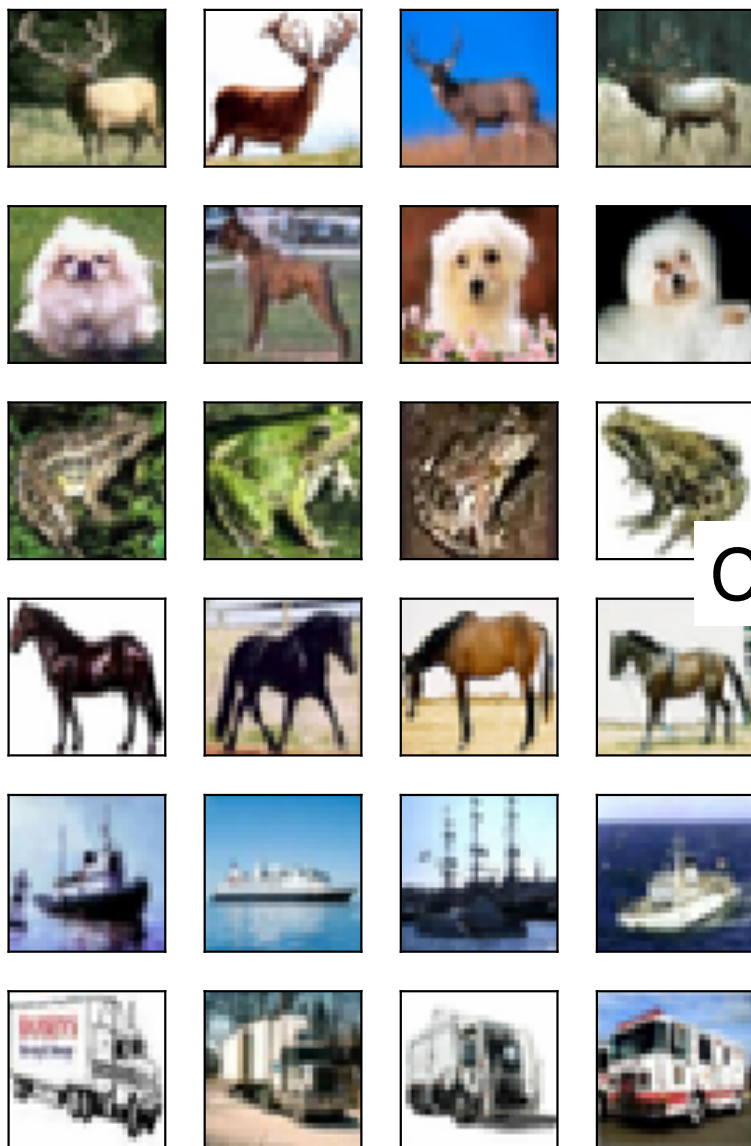


# Most Relevant

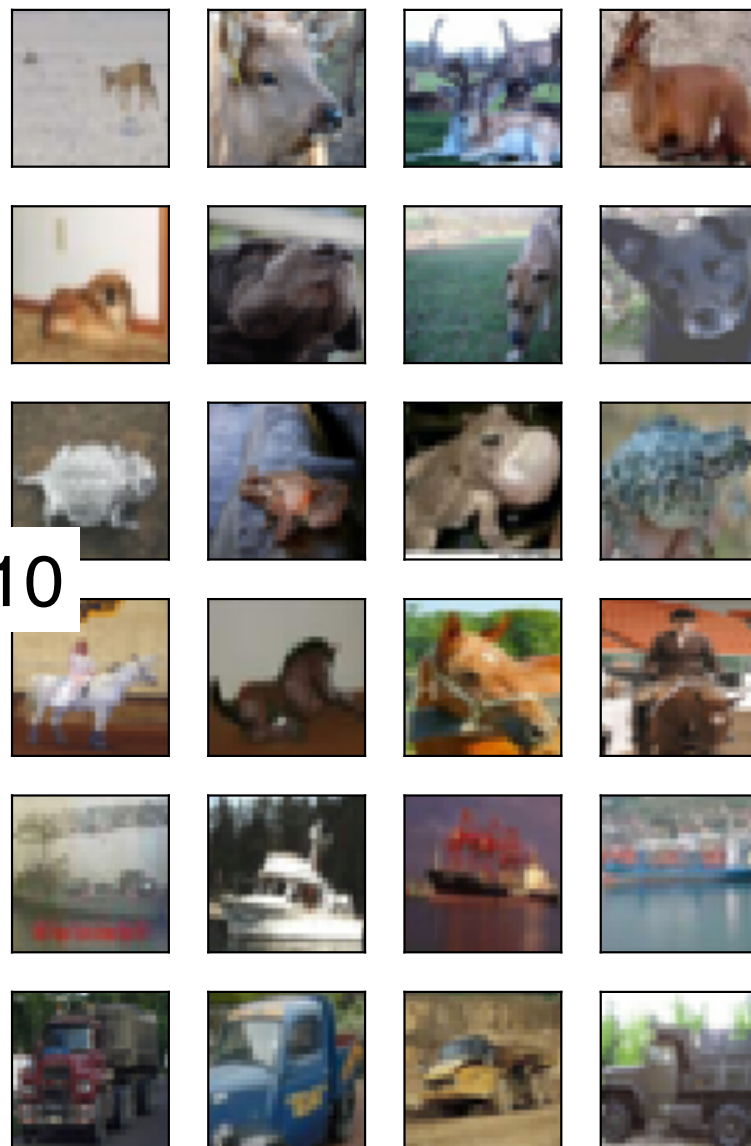


MNIST

# Least Relevant



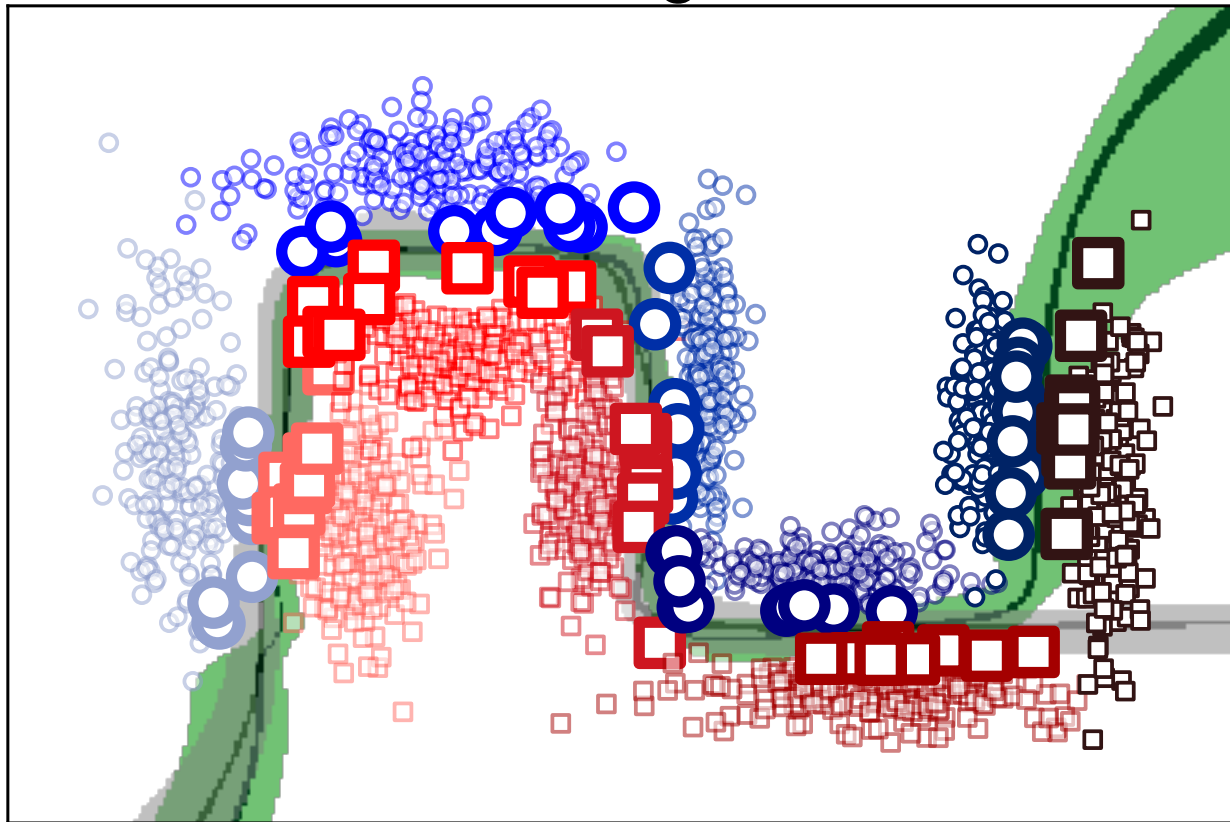
# Most Relevant



CIFAR-10

# Functional Regularization of Memorable Past (FROMP)

Identify, memorize, and regularize the past obtained using DNN2GP

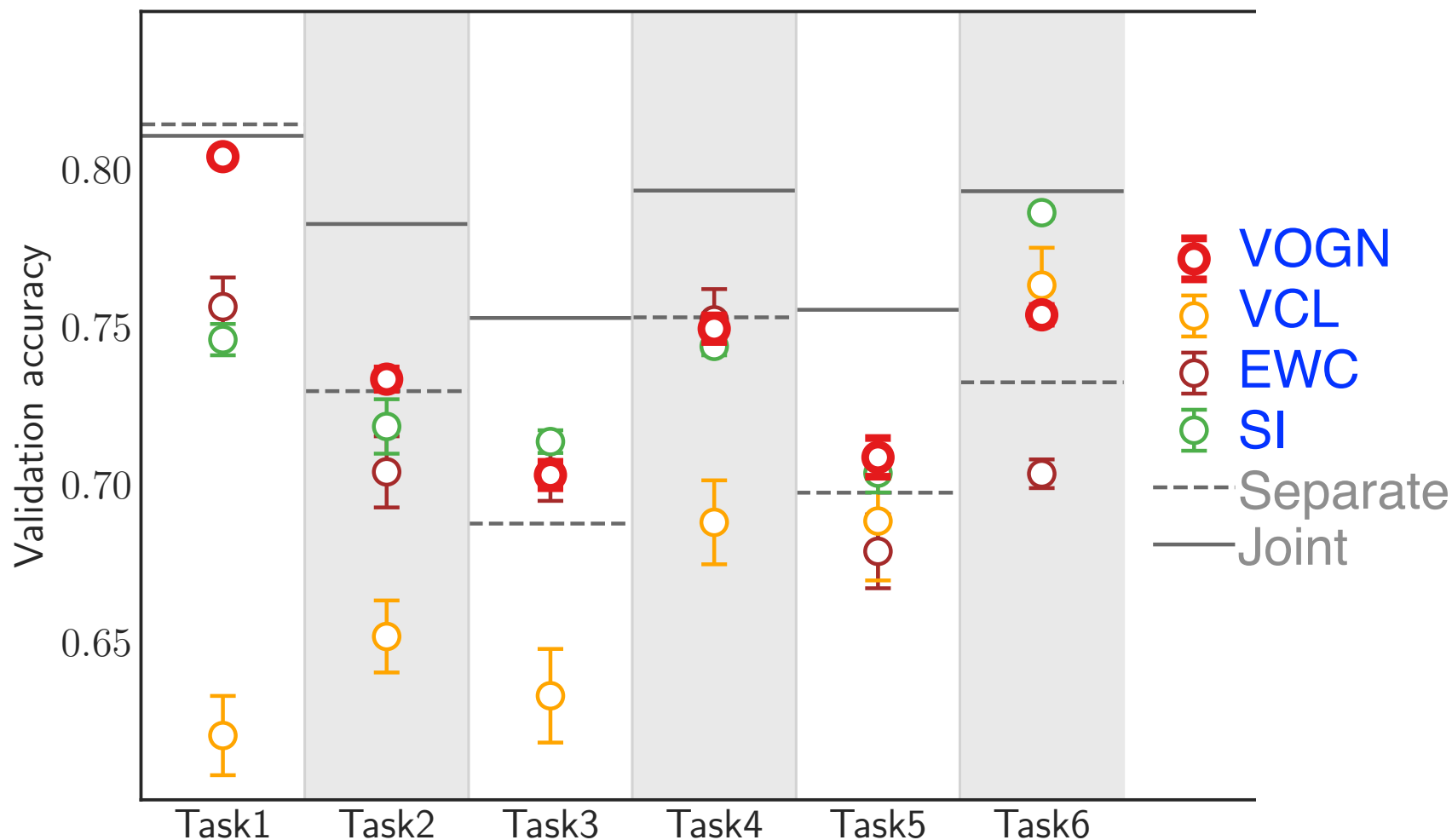


# (Some) Regularization-based Continual Learning Methods

- Elastic-weight consolidation (EWC) [1]
  - Based on a diagonal Laplace approximation
  - [2] considers structured Laplace
- Synaptic Intelligence (SI) [3]
- Variational Continual learning (VCL) [4]
  - Based on variational inference
- Functional Regularization [5]
- With better approximations, we expect the accuracy to improve, but unfortunately we don't see this!

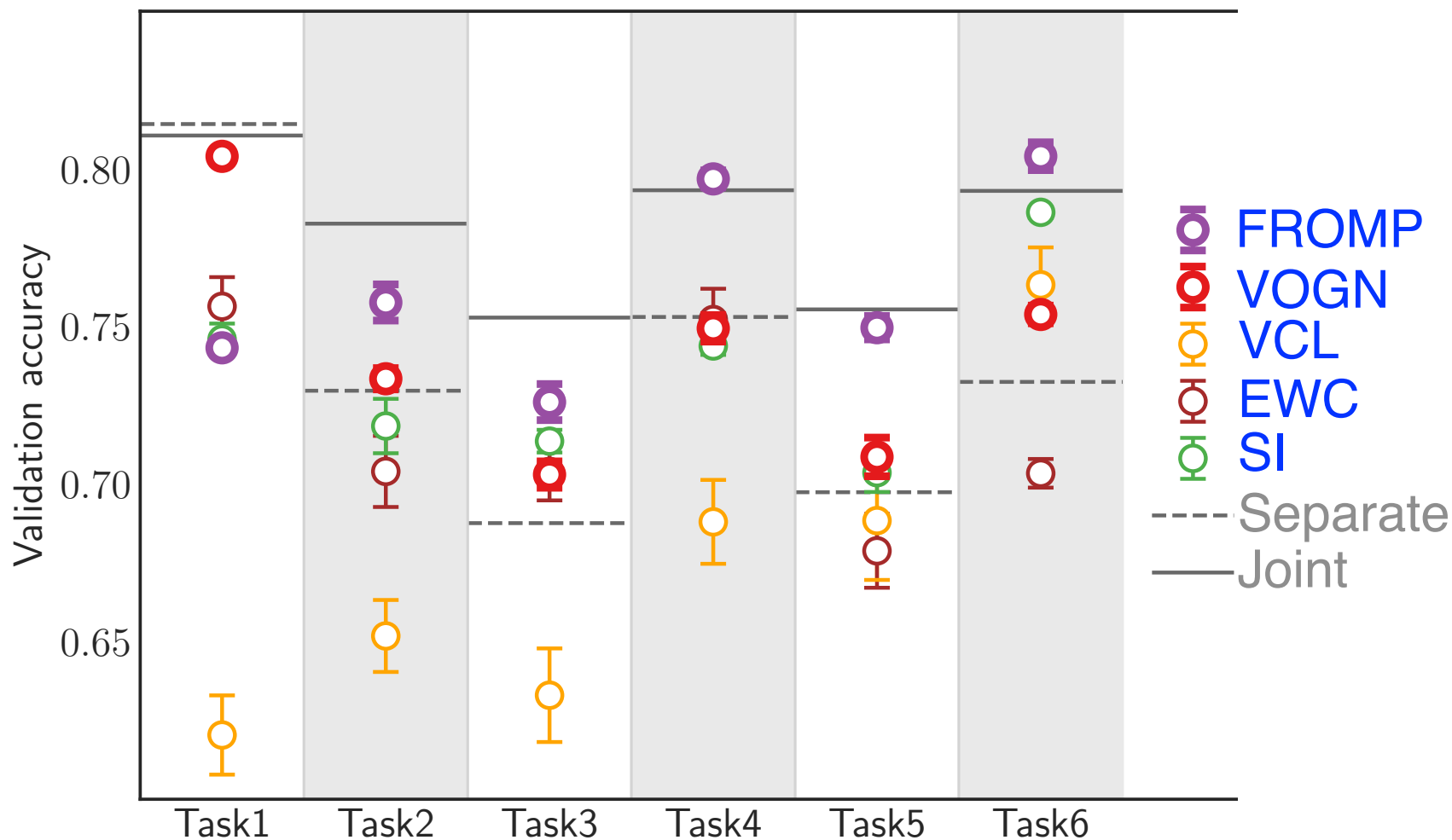
1. Kirkpatrick, James, et al. "Overcoming catastrophic forgetting in neural networks." *PNAS* (2017).
2. Ritter et al. "Online structured laplace ... for overcoming catastrophic forgetting." *NeurIPS*. 2018.
3. Zenke et al. "Continual learning through synaptic intelligence." *ICML*, 2017.
4. Nguyen et al. "Variational continual learning." *arXiv preprint arXiv:1710.10628* (2017).
5. Titsias et al. "Functional Regularisation for Continual Learning with Gaussian Processes." *ICLR* (2019).

# FROMP improves over EWC!





# FROMP improves over EWC!

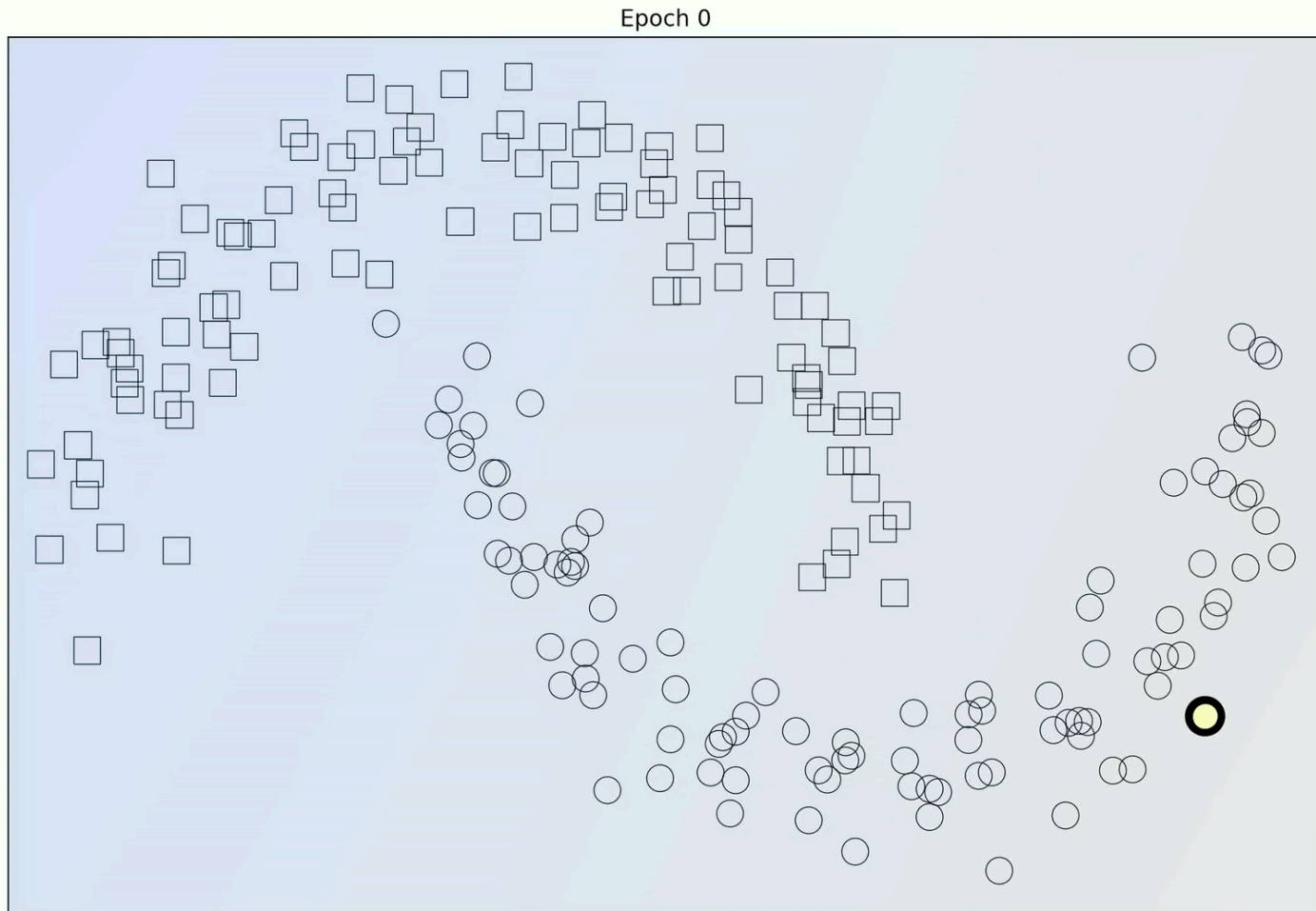


# Challenges in Life-Long Learning

- Computing exact posterior is not tractable
- Approximations do not always behave the way we want them to
  - They can miss important information from the past and lead to forgetting
- Working with the function space is one solution.
- There are plenty of non-Bayesian solutions, but my personal (biased) opinion is that they are in fact related to Bayesian principles

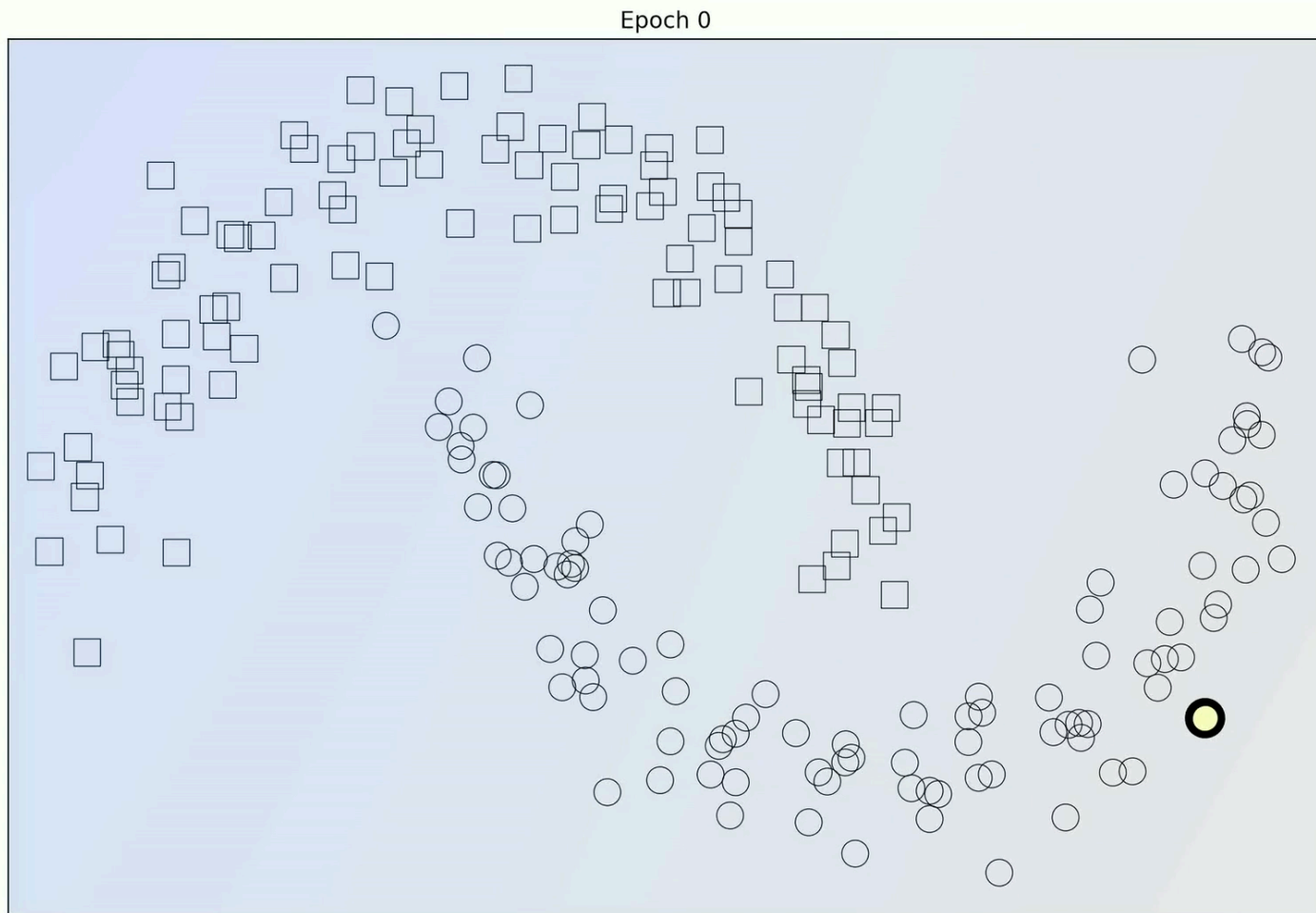
# Active Deep Learning

Select “Important” examples while training with Adam



# Active Deep Learning

Select “Important” examples while training with Adam



# Deep Learning with Bayesian Principles

- Bayesian principles as common principles
  - By computing “posterior approximations”
- Derive many existing algorithms,
  - Deep Learning (SGD, RMSprop, Adam)
  - Exact Bayes, Laplace, Variational Inference, etc
- Design new deep-learning algorithms
  - Uncertainty estimation and life-long learning
- Impact: Many learning-algorithms with a common set of principles.

# Open Challenges

# Open Challenges

- How to achieve Life-long deep learning?

# Open Challenges

- How to achieve Life-long deep learning?
- How to compute better posterior approx?



# Open Challenges

- How to achieve Life-long deep learning?
- How to compute better posterior approx?
- How to compute higher-order gradients?

# Towards Life-long learning

# Towards Life-long learning

- For life-long learning, we need

# Towards Life-long learning

- For life-long learning, we need
  - Perception: how you want to see the world?

# Towards Life-long learning

- For life-long learning, we need
  - Perception: how you want to see the world?
  - Action: what you want to see in the world?

# Towards Life-long learning

- For life-long learning, we need
  - Perception: how you want to see the world?
  - Action: what you want to see in the world?
- Posterior approximation connects the two

# Towards Life-long learning

- For life-long learning, we need
  - Perception: how you want to see the world?
  - Action: what you want to see in the world?
- Posterior approximation connects the two
  - Models are representation of the world

# Towards Life-long learning

- For life-long learning, we need
  - Perception: how you want to see the world?
  - Action: what you want to see in the world?
- Posterior approximation connects the two
  - Models are representation of the world
  - Approximations are representation of the model



# Towards Life-long learning

- For life-long learning, we need
  - Perception: how you want to see the world?
  - Action: what you want to see in the world?
- Posterior approximation connects the two
  - Models are representation of the world
  - Approximations are representation of the model
  - They help us learn the model through actions

# Towards Life-long learning

- For life-long learning, we need
  - Perception: how you want to see the world?
  - Action: what you want to see in the world?
- Posterior approximation connects the two
  - Models are representation of the world
  - Approximations are representation of the model
  - They help us learn the model through actions
  - Act to appropriately “fill” the function/data space

# Learning-Algorithms from Bayesian Principles

Coming soon!

A preliminary version is at  
[https://emtiyaz.github.io/papers/  
learning\\_from\\_bayes.pdf](https://emtiyaz.github.io/papers/learning_from_bayes.pdf)



Havard Rue (KAUST)

# Acknowledgements

Slides, papers, & code  
are at [emtiyaz.github.io](https://emtiyaz.github.io)



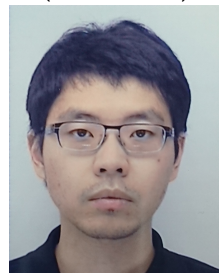
Wu Lin  
(Past: RA)



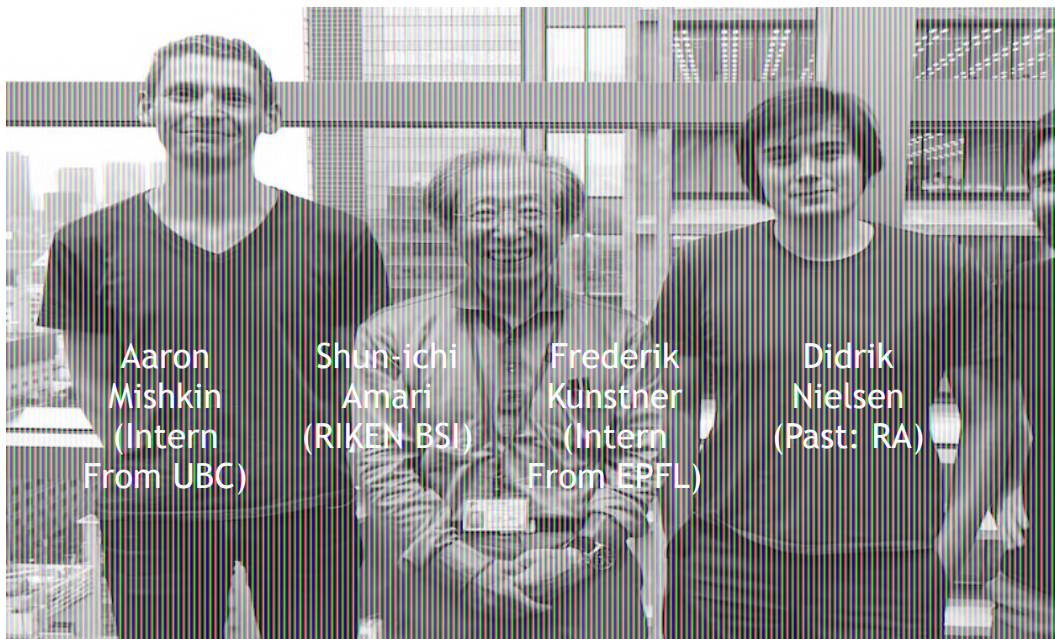
Nicolas Hubacher  
(Past: RA)



Masashi Sugiyama  
(Director RIKEN-AIP)



Voot Tangkaratt  
(Postdoc, RIKEN-AIP)



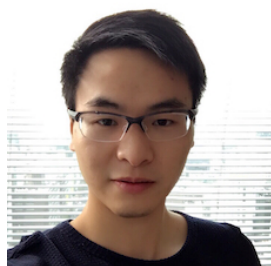
Aaron  
Mishkin  
(Intern  
From UBC)

Shun-ichi  
Amari  
(RIKEN-BSI)

Frederik  
Kunstner  
(Intern  
From EPFL)

Didrik  
Nielsen  
(Past: RA)

## External Collaborators



Zuozhu Liu  
(Intern from SUTD)



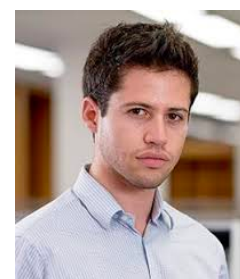
RAIDEN



Mark Schmidt  
(UBC)



Reza Babanezhad  
(UBC)



Yarin Gal  
(UOxford)



Akash Srivastava  
(UEdinburgh)

# Acknowledgements

Slides, papers, & code  
are at [emtiyaz.github.io](https://emtiyaz.github.io)



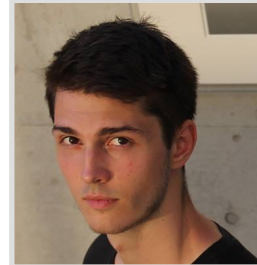
Kazuki Osawa  
(Tokyo Tech)



Rio Yokota  
(Tokyo Tech)



Anirudh Jain  
(Intern from  
IIT-ISM, India)



Runa Eschenhagen  
(Intern from  
University of  
Osnabruck)



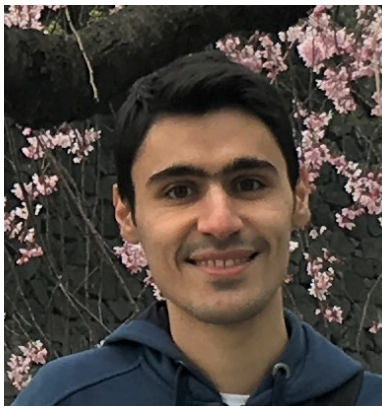
Siddharth  
Swaroop  
(University of  
Cambridge)



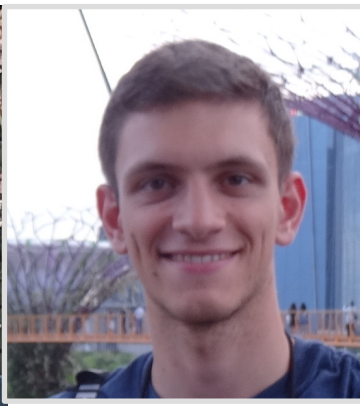
Rich Turner  
(University of  
Cambridge)



Alexander Immer  
(Intern from EPFL)



Ehsan Abedi  
(Intern from EPFL)



Maciej Korzepa  
(Intern from DTU)



Pierre Alquier  
(RIKEN AIP)



Havard Rue  
(KAUST)

+ PingBo Pan(UT Sydney)





# Approximate Bayesian Inference Team

