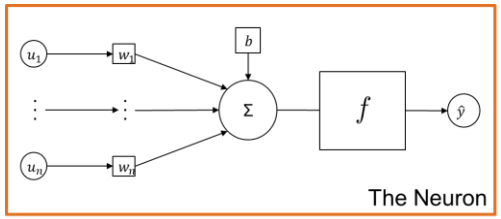
Introduction to Bayesian Neural Networks

Anh Khoa Doan – n.a.k.doan@tudelft.nl

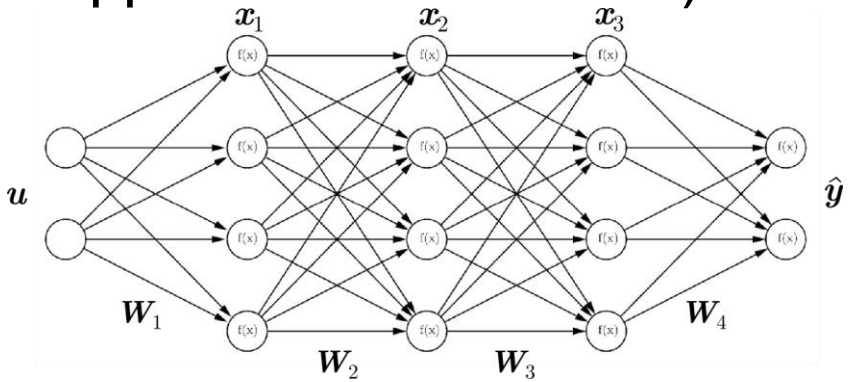
TU Delft

Recap from last week

- The neuron
  - “Base” function for regression problem
  - Single neuron with sigmoid: logistic regression (binary classification)



- Neural networks
  - Complex arrangement of neurons
  - Can “approximate” any function (universal approximator theorem)
  - Trained from data by MSE-minimization
    - With (form of) stochastic gradient descent
    - Efficient with “automatic differentiation”
    - And backpropagation



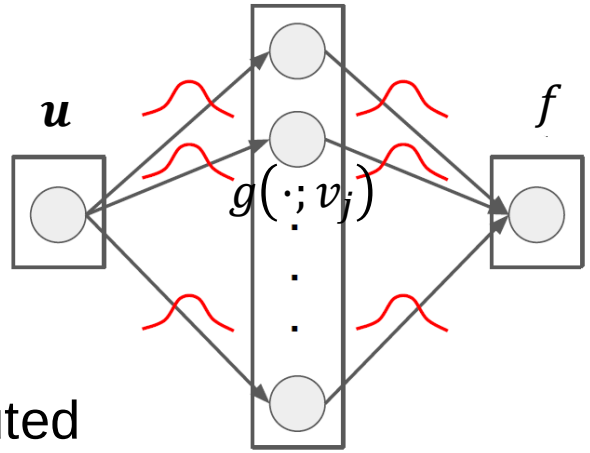
Recap from last week and why UQ with NN

- Neural Networks
  - Can be used for regression (point-based estimate)
  - Can be used for probability distribution estimation over categories (classification with negative log-probability loss (cross entropy loss))
- What is the connection with UQ?
  - Smooth out the predictions by averaging over plausible explanations
  - Get confidence interval
  - Make robust decision

NN converges towards Gaussian process

- Let's start from a single hidden layer NN with N nodes:

$$f(\boldsymbol{u}) = b + \sum_{j}^{N} w_j g(\boldsymbol{u}; v_j)$$



- Now, let's assign some distributions. Assume:
  - $b \sim \mathcal{N}(0, \sigma_b^2)$ and $w_j \sim \mathcal{N}(0, \sigma_w^2)$
  - v_j (neuron weights) are independently and identically distributed
  - σ_w^2 scales as w^2/N then

$$\mathbb{E}(f(\boldsymbol{u})) = 0$$

$$\mathbb{E}(f(\boldsymbol{u})f(\boldsymbol{u}')) = \sigma_b^2 + \sum_{j}^{N} \sigma_w^2 \mathbb{E}_x \left(g(\boldsymbol{u}; x_j) g(\boldsymbol{u}'; x_j) \right)$$

$$= \sigma_b^2 + w^2 \mathbb{E}_x \left(g(\boldsymbol{u}; x_j) g(\boldsymbol{u}'; x_j) \right)$$

  - By theorem central limit, f converges to a Gaussian process as $N \to \infty$

Neal, Bayesian Learning for Neural Networks

Structure of the lecture

- **Reminder on Bayesian Inference**
- Bayesian Neural Network

Small reminder on Bayesian Inference

- Bayesian inference: predictions by averaging over all likely explanations under the posterior distribution
- Posterior estimated with Bayes' rule

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

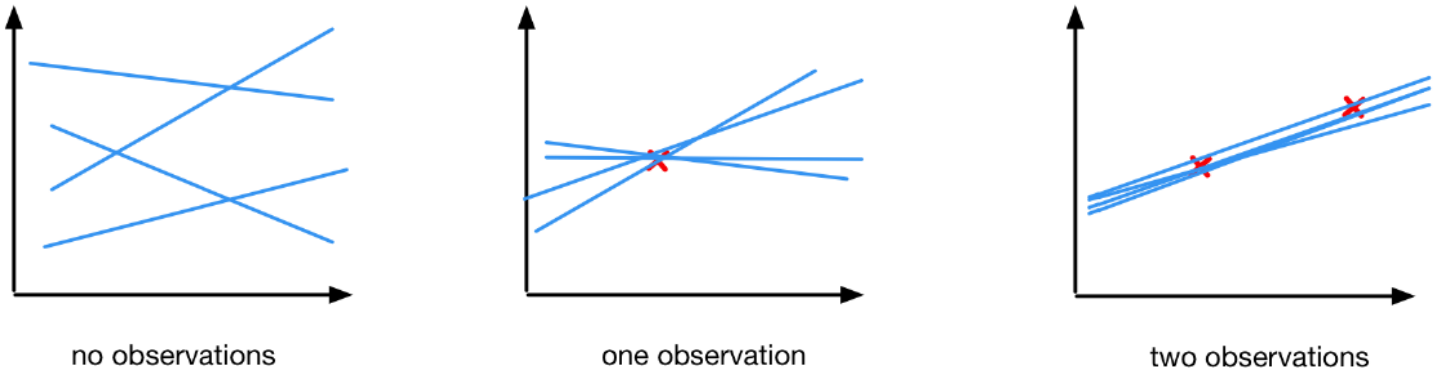
$\mathcal{D}$: distribution of observed values

- Prediction using the posterior predictive distribution:

$$p(y|\boldsymbol{u},\mathcal{D}) = \int p(\boldsymbol{w}|\mathcal{D})p(y|\boldsymbol{u},\boldsymbol{w})d\boldsymbol{w}$$

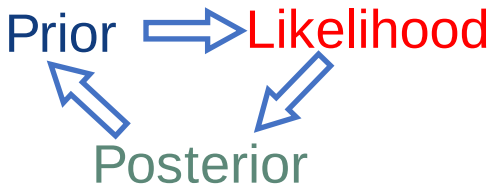
Bayesian linear regression

- Bayesian linear regression: considers various “plausible” explanation for data generation
- Prediction obtained using all possible regression weights, weighted by their posterior probability

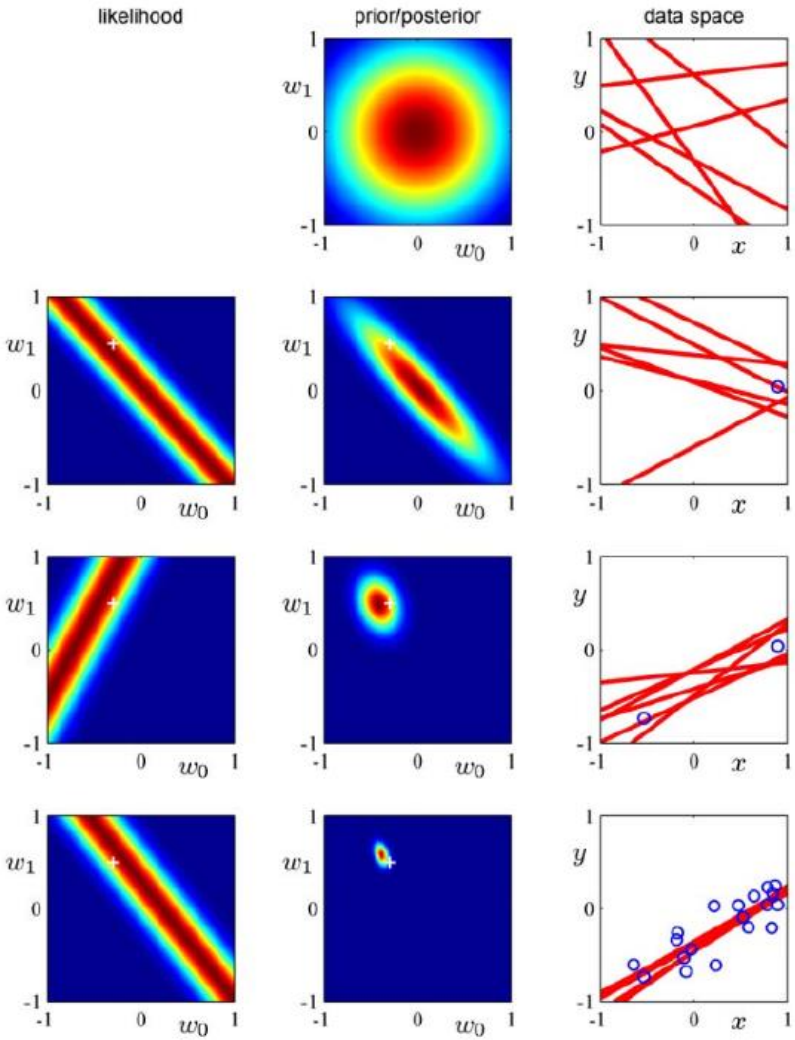


- Prior distribution: $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{S})$
- Likelihood: $p(y|\boldsymbol{u}, \boldsymbol{w}) \sim \mathcal{N}(\boldsymbol{w}^T \psi(\boldsymbol{u}), \sigma^2)$
- Assumed fixed/known $\boldsymbol{S}$ and σ^2 is a big assumption

Bayesian linear regression



1. Prior distribution
2. 3. 4. Model distribution is adjusted to data

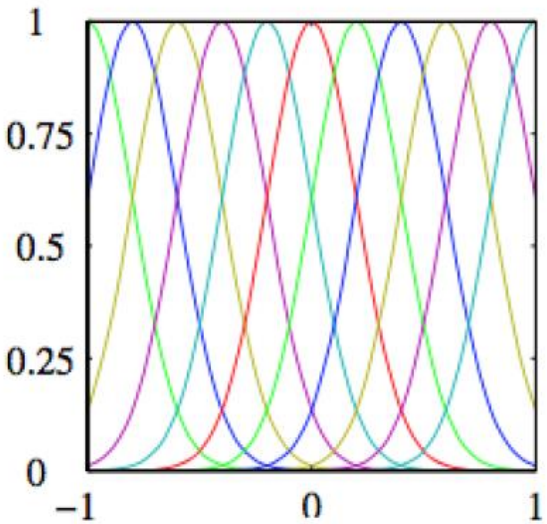


Bishop, Pattern Recognition and Machine Learning

Bayesian Regression

- Example with radial basis function features:

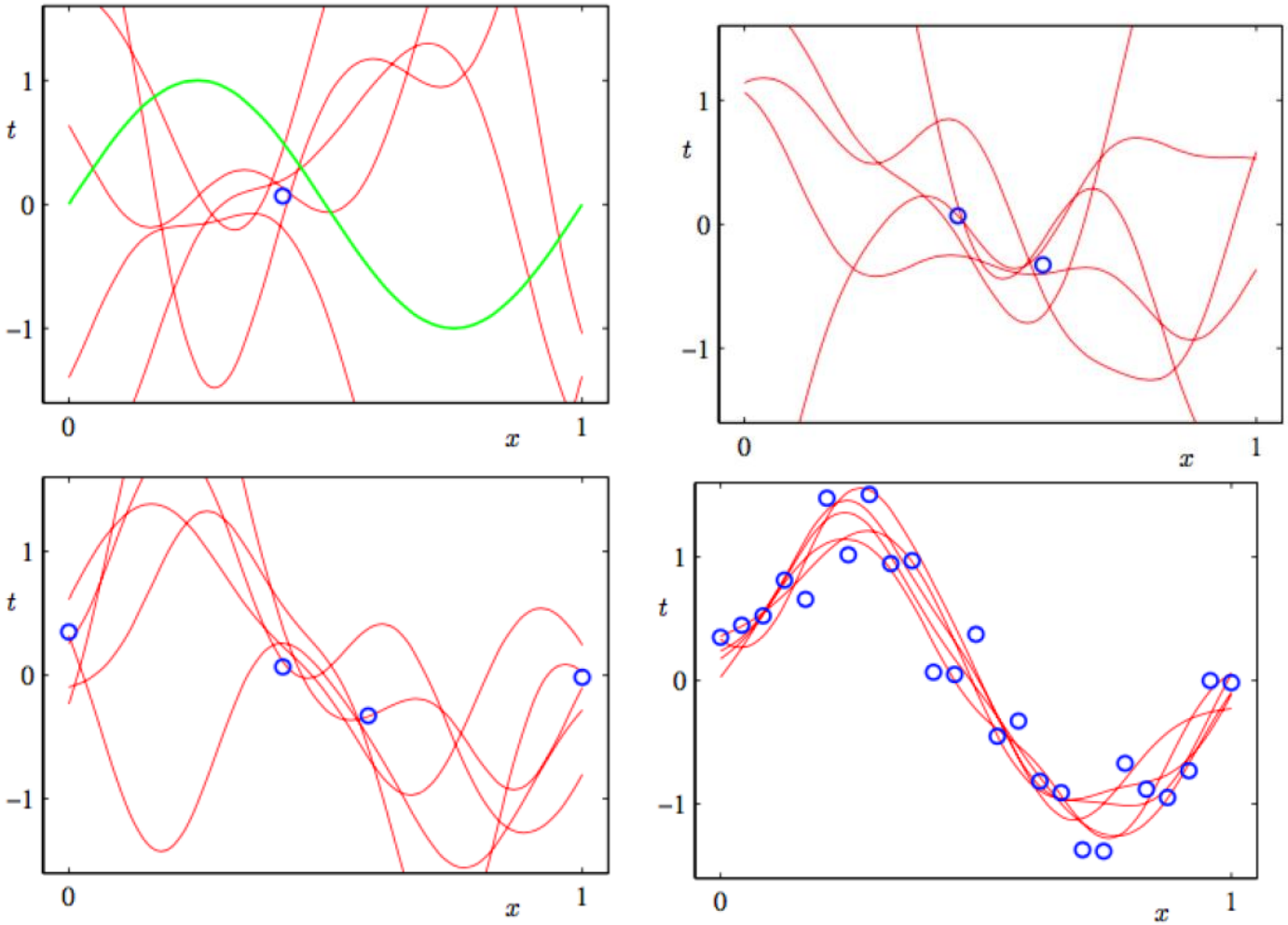
$$\phi_j(u) = \exp\left(-\frac{(x-\mu_j)^2}{2s^2}\right)$$



Bishop, Pattern Recognition and Machine Learning

Bayesian Regression

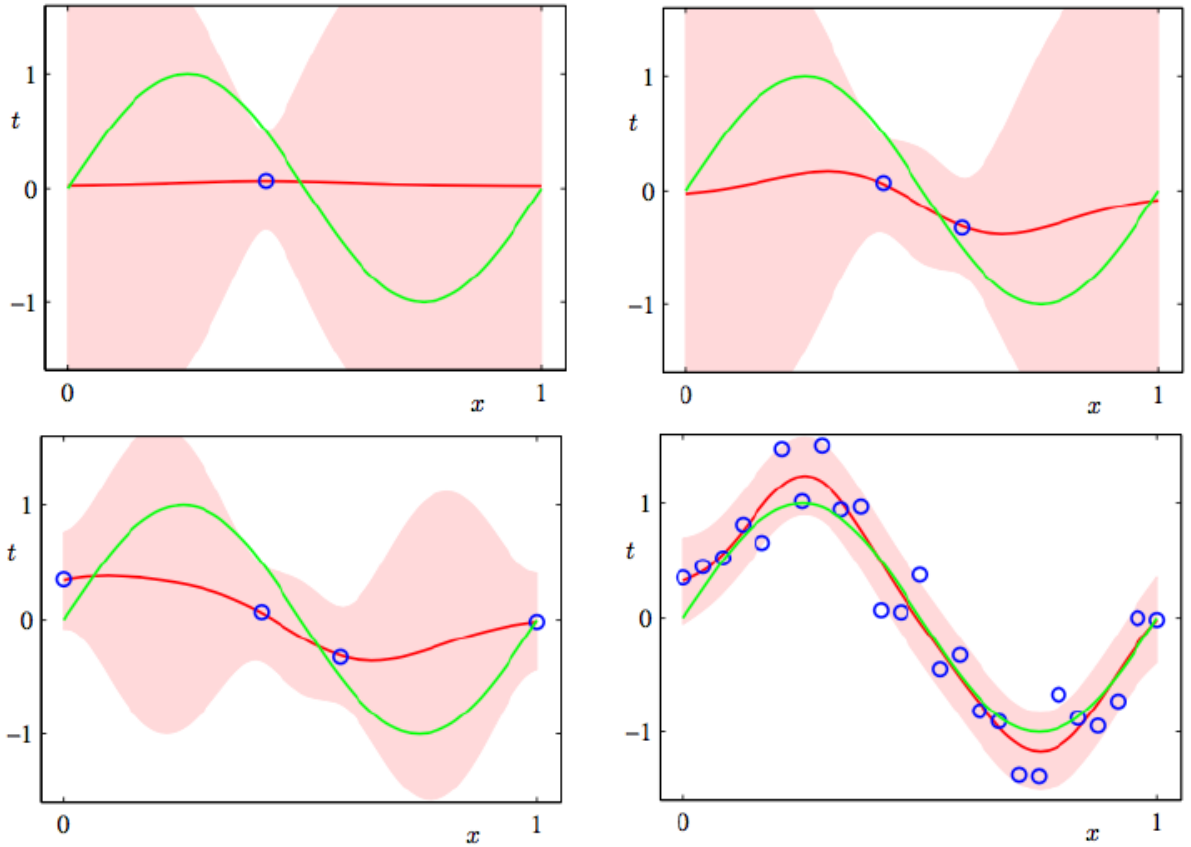
Functions sampled from the posterior:



Bishop, Pattern Recognition and Machine Learning

Bayesian Regression

- Visualization of confidence intervals based on posterior predictive mean and variance



Bishop, Pattern Recognition and Machine Learning

Structure of the lecture

- Reminder on Bayesian Inference
- **Bayesian Neural Network**

Bayesian neural network

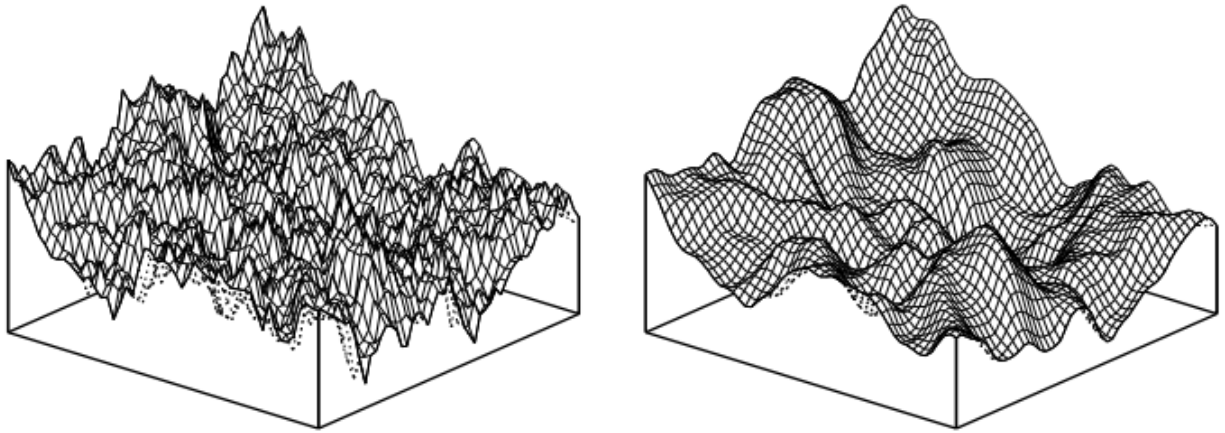
- So far: fixed basis functions…
- Can we combine the advantage of neural networks with Bayesian models?
  - Place a prior on the weights of the network, e.g. $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{0}, \eta \boldsymbol{I})$
  - Define an observation model, e.g. $p(y|\boldsymbol{u}, \boldsymbol{w}) = \mathcal{N}(y; f_{\boldsymbol{w}}(\boldsymbol{u}), \sigma^2)$
  - Apply Bayes' Rule:

$$p(\boldsymbol{w}|\mathcal{D}) \propto p(\boldsymbol{w}) \prod_{i=1}^{N} p(y^{(i)}|\boldsymbol{u}^{(i)}, \boldsymbol{w})$$

$\mathcal{D}$: dataset distribution

Samples from the prior

- How to interpret prior $p(\boldsymbol{w})$ with network kernel?
- Prior samples for a BNN with one hidden layer and 10,000 units



hard threshold activations tanh activations

Neal, Bayesian Learning for Neural Networks

BNN: what to maximize

- More generally, what we consider is:
  - Prior weight distribution: $p(\boldsymbol{w})$
  - Given a training dataset sampled distribution $\mathcal{D}$
  - Likelihood function: $p(\mathcal{D}|\boldsymbol{w}) = p(\boldsymbol{w}) \prod_{i=1}^{N} p(y^{(i)}|\boldsymbol{u}^{(i)}, \boldsymbol{w})$
- We could try to:
  - maximize $p(\mathcal{D}|\boldsymbol{w})$→ Maximum likelihood estimation (MLE)
    - Can be biased by the data → to be avoided when few data
- Alternate: Bayes rule gives:

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

- We could try to maximize:
  - $p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w})$: Maximum a posteriori (MAP) estimation

Posterior Inference

- Possible use of posterior distribution: sample set of values $\boldsymbol{w_1}, \dots, \boldsymbol{w_K}$ from the posterior distribution $p(\boldsymbol{w}|\mathcal{D})$ and average their predictive distributions:

$$p(y|\boldsymbol{u}, \mathcal{D}) \approx \frac{1}{K} \sum_{k=1}^{K} p(y|\boldsymbol{u}, \boldsymbol{w_k})$$

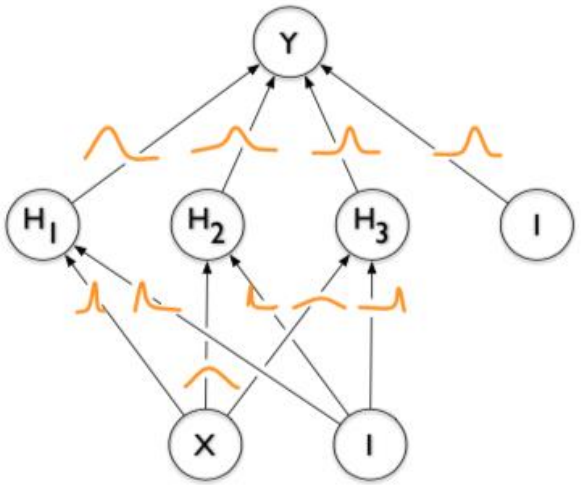
- Sample from posterior: can be obtained approximately with Markov Chain Monte Carlo
  - But can be expensive with very large dataset…
- How can we obtain $p(\boldsymbol{w}|\mathcal{D})$?
- Instead: Variational inference to estimate $p(\boldsymbol{w}|\mathcal{D})$

Posterior Inference: Variational Bayes

- Idea: Approximate complex posterior distribution with simpler (analytical) variational approximation q with parameters $\boldsymbol{\theta}$
  - Eg: Assume Gaussian posterior with diagonal covariance

$$q(\boldsymbol{w};\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{w};\boldsymbol{\mu},\boldsymbol{\Sigma})$$
$$= \prod_{j}^{D} \mathcal{N}\left(w_j;\mu_j,\sigma_j\right)$$

"each weight of the network has its own mean and variance"



Blundel et al., Weight uncertainty for neural network

Posterior Inference: Variational Bayes

- From $q(\boldsymbol{w};\boldsymbol{\theta})$, we now need to estimate the "best" $\boldsymbol{\theta}$ that gives us the best approximation of the posterior $p(\boldsymbol{w}|\mathcal{D})$
- We can use the Kullback-Leibler divergence ("measure of distance between two distributions"), D_{KL}, between the two:

$$D_{KL}\big(q(\boldsymbol{w};\boldsymbol{\theta})||p(\boldsymbol{w}|\mathcal{D})\big) \equiv \int q(\boldsymbol{w};\boldsymbol{\theta}) \log \frac{q(\boldsymbol{w};\boldsymbol{\theta})}{p(\boldsymbol{w}|\mathcal{D})} d\boldsymbol{w}$$

$$= \mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})} \log \frac{q(\boldsymbol{w};\boldsymbol{\theta})}{p(\boldsymbol{w}|\mathcal{D})}$$

Posterior Inference: Variational Bayes

$$D_{KL}\big(q(\boldsymbol{w};\boldsymbol{\theta})||p(\boldsymbol{w}|\mathcal{D})\big) = \mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})} \log \frac{q(\boldsymbol{w};\boldsymbol{\theta})}{p(\boldsymbol{w}|\mathcal{D})}$$

- With Bayes' rule for $p(\boldsymbol{w}|\mathcal{D})$

$$D_{KL}\big(q(\boldsymbol{w};\boldsymbol{\theta})||p(\boldsymbol{w}|\mathcal{D})\big) = \mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})} \log \frac{q(\boldsymbol{w};\boldsymbol{\theta})}{p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w})} p(\mathcal{D})$$

$$= \mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})} \log \frac{q(\boldsymbol{w};\boldsymbol{\theta})}{p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w})} p(\mathcal{D})$$

$$= \mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})}[\log q(\boldsymbol{w};\boldsymbol{\theta}) - \log p(\mathcal{D}|\boldsymbol{w}) - \log p(\boldsymbol{w}) + \log p(\mathcal{D})]$$

$$D_{KL}\big(q(\boldsymbol{w};\boldsymbol{\theta})||p(\boldsymbol{w}|\mathcal{D})\big) = D_{KL}\big(q(\boldsymbol{w};\boldsymbol{\theta})\big|\big|p(\boldsymbol{w})\big) - \mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})}[\log p(\mathcal{D}|\boldsymbol{w})] + \log p(\mathcal{D})$$

Posterior Inference: Variational Bayes

$$D_{KL}\big(q(\boldsymbol{w};\boldsymbol{\theta})||p(\boldsymbol{w}|\mathcal{D})\big) = D_{KL}(q(\boldsymbol{w};\boldsymbol{\theta})||p(\boldsymbol{w})) - \mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})}[\log p(\mathcal{D}|\boldsymbol{w})] + \log p(\mathcal{D})$$

- $\mathcal{F}(\mathcal{D},\boldsymbol{\theta}) \equiv D_{KL}(q(\boldsymbol{w};\boldsymbol{\theta})||p(\boldsymbol{w})) - \mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})}[\log p(\mathcal{D}|\boldsymbol{w})]$
  - Called "variational free energy"
- We can minimize $\mathcal{F}$ with respect to $\boldsymbol{\theta}$ and that will minimize $D_{KL}\big(q(\boldsymbol{w};\boldsymbol{\theta})||p(\boldsymbol{w}|\mathcal{D})\big)$
  - And $q(\boldsymbol{w};\boldsymbol{\theta})$ will approximate the posterior
- Alternate name: $\mathcal{L}(\mathcal{D},\boldsymbol{\theta}) \equiv -\mathcal{F}(\mathcal{D},\boldsymbol{\theta})$ "evidence lower bound"
  - As $\mathcal{L} \leq p(\mathcal{D})$ (as $D_{KL} \geq 0$)

Posterior Inference: Variational Bayes

- Now we need to minimize:

$$\mathcal{F}(\mathcal{D}, \boldsymbol{\theta}) \equiv D_{KL}(q(\boldsymbol{w}; \boldsymbol{\theta}) || p(\boldsymbol{w})) - \mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})}[\log p(\mathcal{D}|\boldsymbol{w})]$$

  - $D_{KL}(q(\boldsymbol{w}; \boldsymbol{\theta}) || p(\boldsymbol{w}))$: "complexity cost"
  - $\mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})}[\log p(\mathcal{D}|\boldsymbol{w})]$: "likelihood cost"
- Some re-arrangement

$$\mathcal{F}(\mathcal{D}, \boldsymbol{\theta}) = \mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})}[\log q(\boldsymbol{w}; \boldsymbol{\theta})] - \mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})}[\log p(\boldsymbol{w})] - \mathbb{E}_{q(\boldsymbol{w};\boldsymbol{\theta})}[\log p(D|\boldsymbol{w})]$$

- All terms are expectations with respect to $q(\boldsymbol{w}; \boldsymbol{\theta})$, so we can approximate them by drawing samples for $\boldsymbol{w}^{(i)}$ from $q(\boldsymbol{w}; \boldsymbol{\theta})$

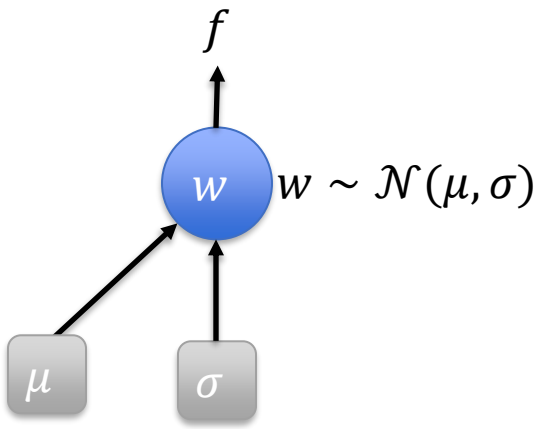
$$\mathcal{F}(\mathcal{D}, \boldsymbol{\theta}) \approx \frac{1}{N} \sum_{i}^{N} \log q\left(\boldsymbol{w}^{(i)}; \boldsymbol{\theta}\right) - \log p\left(\boldsymbol{w}^{(i)}\right) - \log p\left(D|\boldsymbol{w}^{(i)}\right)$$

→ We can minimize that with respect to $\boldsymbol{\theta}$

Note on backpropagation with random nodes

- How do we backpropagate with random variables involved?
- "Re-parametrization" trick

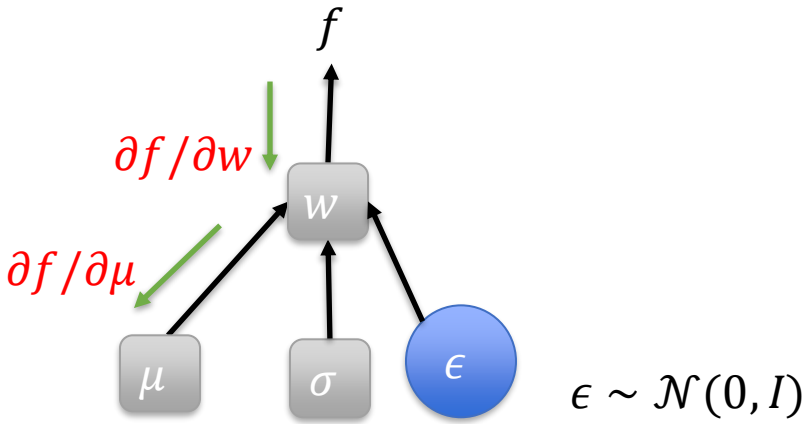
Original form:



: deterministic node

: random node

Reparametrized form:



1. Sample ϵ
2. Let $w = \mu + \sigma\epsilon$ (or $w = \mu + (1 + \log\sigma)\epsilon$)
3. Backpropagate

Kingma et al., Variational Dropout and Local Reparametrization

Training with Bayesian NN

- Train a Bayesian NN with:

$$w_j = \mu_j + \sigma_j \epsilon_j$$

  With $\epsilon_j \sim \mathcal{N}(0,1)$
- ϵ_j sampled at the beginning of training, independent of μ_j, σ_j
  - → Deterministic graph, backpropagation algorithm can be used
- If all $\sigma_j = 0$, then $\theta_j = \mu_j$, and ordinary backpropagation with deterministic neural network can be used

Structure of the lecture

- Reminder on Bayesian Inference
- Bayesian Neural Network

Bayesian Neural Network - Summary

- Approach that combines Bayesian principle with NN
- Can propagate uncertainty/account for uncertainty