02477 – Bayesian Machine Learning: Lecture 8

Michael Riis Andersen

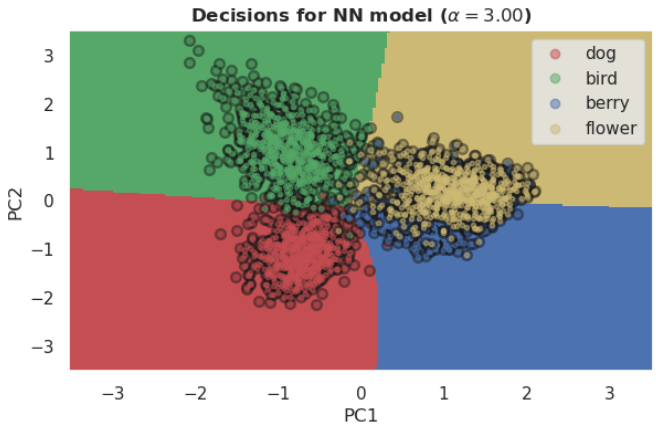
Technical University of Denmark,
DTU Compute, Department of Applied Math and Computer Science

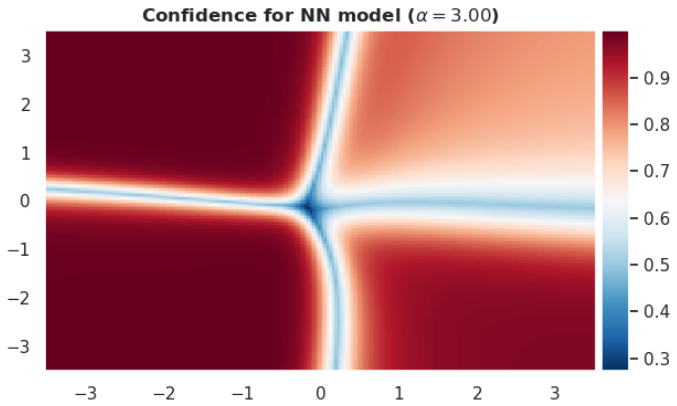
Outline

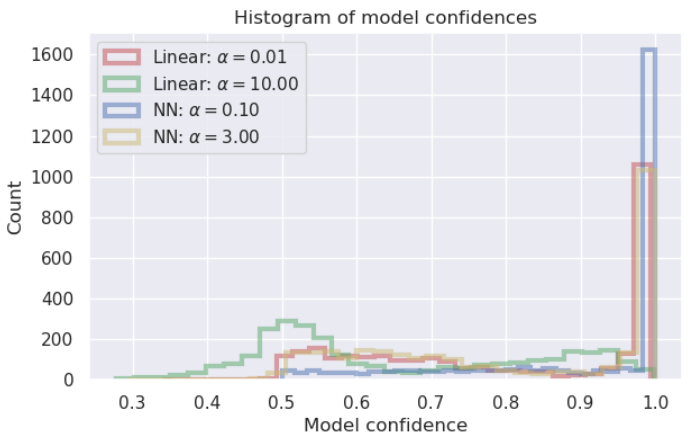
1. More on calibration
2. Monte Carlo methods
3. Simple sampling methods
4. Markov Chain Monte Carlo methods

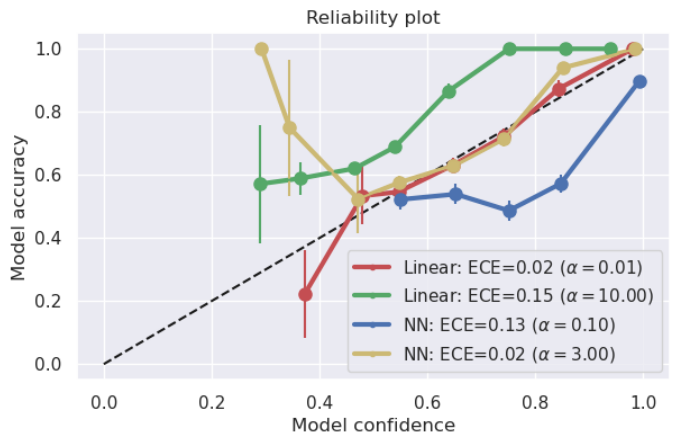
More on calibration

Calibration and reliability plots









Expected calibration error (ECE)

- Compute predictions for validation set $\mathcal{D}_{\text{val}} = \{\mathbf{x}_m^*, y_m^*\}_{m=1}^M$
- Divide unit interval in B bins such that $I_b = \left(\frac{b-1}{B}, \frac{b}{B}\right]$
- Let $\mathcal{B}_b$ be the set of indices of samples whose prediction confidence into interval and define

$$\hat{y}_m^* = \arg\max_c p(y_m^* = c|\mathbf{y}, \mathbf{x}_m^*)$$

$$\hat{p}_m^* = \max_c p(y_m^* = c|\mathbf{y}, \mathbf{x}_m^*)$$

- Then the *average accuracy* and the *average confidence* for bin b is defined as

$$\text{acc}(\mathcal{B}_b) = \frac{1}{|\mathcal{B}_b|} \sum_{m \in \mathcal{B}_b} \mathbb{I}\left[\hat{y}_m^* = y_m^*\right]$$

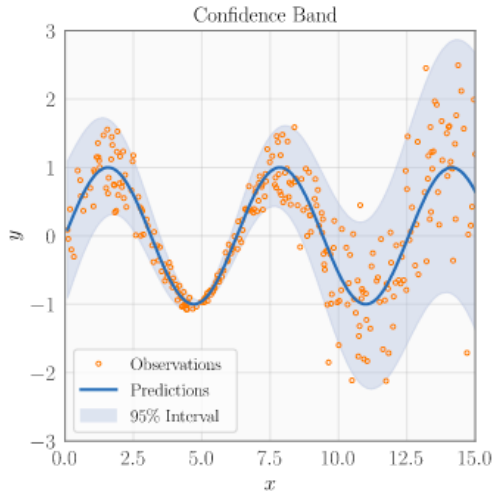
$$\text{conf}(\mathcal{B}_b) = \frac{1}{|\mathcal{B}_b|} \sum_{m \in \mathcal{B}_b} \hat{p}_m^*$$

- The *expected calibration error (ECE)* is then given by

$$\text{ECE} = \sum_{b=1}^{B} \frac{|\mathcal{B}_b|}{M} |\text{acc}(\mathcal{B}_b) - \text{conf}(\mathcal{B}_b)|$$

- Expected vs maximum calibration error

How about calibration for regression?



Images from the Python Uncertainty Toolbox (link).

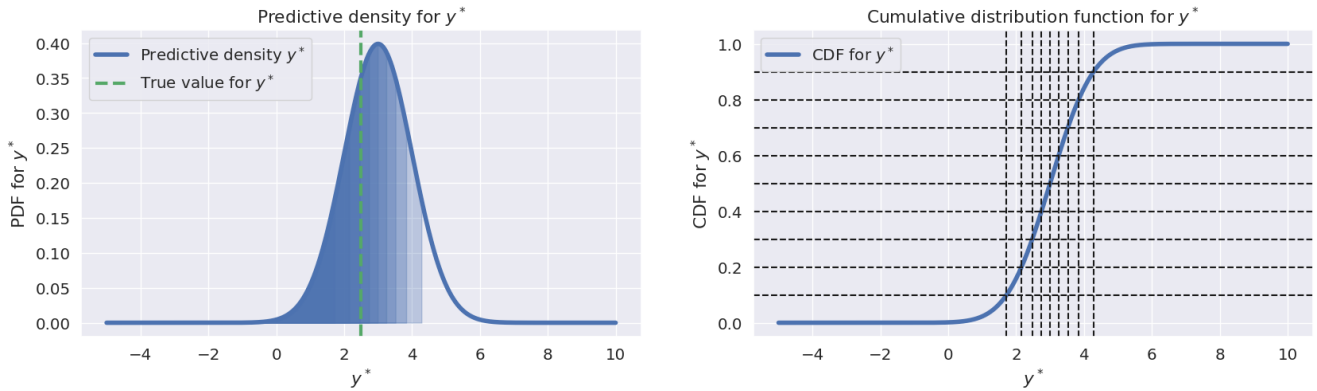
- How to measure calibration for regression? What's the natural quantity to bin?
- Recall: The CDF for predictive density $p(y^*|\mathbf{y}, \mathbf{x}^*) \equiv p(y^*)$

$$F(\tau) \equiv P(y^* \leq \tau) = \int_{-\infty}^{\tau} p(y^*) \mathrm{d}y^* \in [0, 1]$$

Calibration for regression

- Recall: The CDF for predictive density $p(y^*|\mathbf{y}, \mathbf{x}^*) \equiv p(y^*)$

$$F(\tau) \equiv P(y^* \leq \tau) = \int_{-\infty}^{\tau} p(y^*)\mathrm{d}y^* \in [0, 1]$$



- We discretize the unit interval again and for each value p we estimate

$$\hat{p} = \frac{\sum_{n=1}^{N} \mathbb{I}\left[y_n^* \leq F_n^{-1}(p)\right]}{N} \quad \overset{\text{if calibrated}}{\longrightarrow} \quad p$$

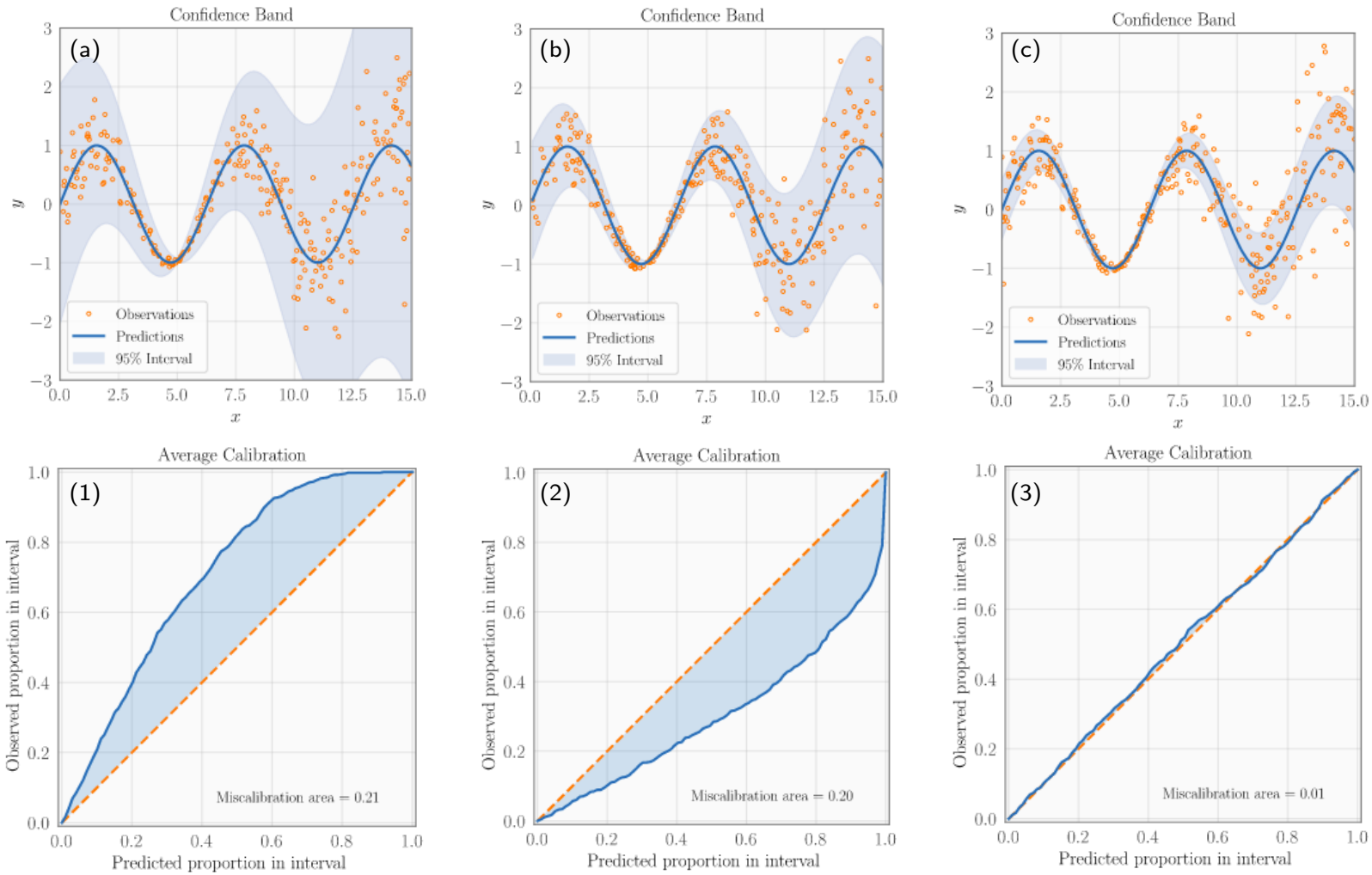
where F_n is the CDF of the predictive distribution for the n'th datapoint.

- Extension to intervals for $p_1, p_2 \in [0, 1]$

$$\frac{\sum_{n=1}^{N} \mathbb{I}\left[F_n^{-1}(p_{-1}) \leq y_n^* \leq F_n^{-1}(p_2)\right]}{N} \quad \overset{\text{if calibrated}}{\longrightarrow} \quad p_2 - p_1$$

DTU Learn quiz: Calibration for regression

Three fits, one is well-calibrated, one is overconfident, and one is underconfident. Which is which? And which fit matches which calibration curve?



Images from the Python Uncertainty Toolbox (link).

Monte Carlo methods

Bayesian methods and those annoying integrals...

- We compute the posterior distribution of the weights $\boldsymbol{w}$ given the data $\boldsymbol{y}$ using Bayes rule

$$p(\boldsymbol{w}|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{w})p(\boldsymbol{w})}{p(\boldsymbol{y})}$$

- When making predictions, we evaluate the predictive distribution

$$p(y^*|\boldsymbol{y}, \boldsymbol{x}^*) = \int p(y^*|\boldsymbol{w}, \boldsymbol{x}^*)p(\boldsymbol{w}|\boldsymbol{y})\mathrm{d}\boldsymbol{w}$$

- For most models of practical interest, both requires intractable integrals.

- Conjugate priors: often a bit like "getting the right answer to the wrong question."

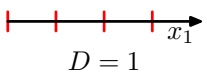
- For general Bayesian inference, we need approximate inference methods.
  1. Laplace approximations.
  2. Variational approximations.
  3. Markov Chain Monte Carlo methods.

Monte Carlo methods

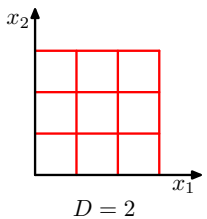
- Stanisław Ulam invented *Markov Chain Monte Carlo methods* in the 1940s while working on the first nuclear bomb with John von Neumann and Nicholas Metropolis.

- To keep their method a secret, they used a code name: *Monte Carlo*.

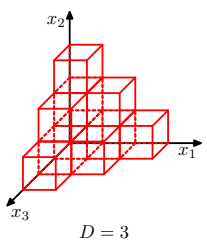
- Named after the famous casino in Monte Carlo, Monaco.



- *Monte Carlo methods* is a common term for algorithms that relies on *random sampling*.



- Monte Carlo integration breaks the *curse of dimensionality*.



$$\mathbb{E}_p[f(\boldsymbol{z})] = \int f(\boldsymbol{z}) p(\boldsymbol{z}) \mathrm{d}\boldsymbol{z}$$

Managing expectations...

Many posterior summaries of interest can be cast as expectations.

- The mean

$$\mathbb{E}[\boldsymbol{w}] = \int \boldsymbol{w} p(\boldsymbol{w}|\boldsymbol{y}) \mathrm{d}\boldsymbol{w} = \mathbb{E}[f(\boldsymbol{w})] \quad \text{for} \quad f(\boldsymbol{w}) = \boldsymbol{w}$$

- The variance

$$\mathbb{V}[\boldsymbol{w}] = \int (\boldsymbol{w} - \mathbb{E}[\boldsymbol{w}])^2 p(\boldsymbol{w}|\boldsymbol{y}) \mathrm{d}\boldsymbol{w} = \mathbb{E}[f(\boldsymbol{w})] \quad \text{for} \quad f(\boldsymbol{w}) = (\boldsymbol{w} - \mathbb{E}[\boldsymbol{w}])^2$$

- Probabilitites

$$\mathbb{P}(w > \tau) = \int_{\tau}^{\infty} p(w|\boldsymbol{y}) \mathrm{d}w = \mathbb{E}[f(w)] \quad \text{for} \quad f(w) = \mathbb{I}[w > \tau]$$

- The predictive density

$$p(y^*|\boldsymbol{y}, \boldsymbol{x}_*) = \int p(y^*|\boldsymbol{w}, \boldsymbol{x}^*) p(\boldsymbol{w}|\boldsymbol{y}) \mathrm{d}\boldsymbol{w} = \mathbb{E}[f(\boldsymbol{w})] \quad \text{for} \quad f(\boldsymbol{w}) = p(y^*|\boldsymbol{w}, \boldsymbol{x}^*)$$

Monte Carlo integration I

- Our goal is to estimate the expectation

$$\bar{f} = \mathbb{E}\left[f(\boldsymbol{z})\right] = \int f(\boldsymbol{z}) p(\boldsymbol{z}) \mathrm{d}\boldsymbol{z}$$

- Suppose we can get a number of *i.i.d.* samples from the distribution $\boldsymbol{z}^i \sim p(\boldsymbol{z})$, then the *Monte Carlo (MC) estimator* is given by

$$\bar{f} \approx \hat{f} = \frac{1}{S} \sum_{i=1}^{S} f(\boldsymbol{z}^i)$$

- The Monte Carlo estimator $\hat{f}$ is a *random variable*.

- Let's calculate the mean of the Monte Carlo estimator

$$\mathbb{E}\left[\hat{f}\right] = \mathbb{E}\left[\frac{1}{S} \sum_{i=1}^{S} f(\boldsymbol{z}^i)\right] = \frac{1}{S} \sum_{i=1}^{S} \mathbb{E}\left[f(\boldsymbol{z}^i)\right] = \frac{1}{S} \sum_{i=1}^{S} \bar{f} = \bar{f}$$

- Hence, the Monte Carlo estimator is *unbiased*.

Monte Carlo integration II

- The Monte Carlo estimator is

$$\bar{f} \approx \hat{f} = \frac{1}{S} \sum_{i=1}^{S} f(\boldsymbol{z}^i)$$

- Let's calculate the variance

$$\mathbb{V}\left[\hat{f}\right] = \mathbb{V}\left[\frac{1}{S} \sum_{i=1}^{S} f(\boldsymbol{z}^i)\right] = \frac{1}{S^2} \mathbb{V}\left[\sum_{i=1}^{S} f(\boldsymbol{z}^i)\right] = \frac{1}{S^2} \sum_{i=1}^{S} \mathbb{V}\left[f(\boldsymbol{z}^i)\right] = \frac{1}{S} \mathbb{V}\left[f(\boldsymbol{z})\right]$$

- Remarkably, the variance of $\hat{f}$ is *independent of the dimension* of $\boldsymbol{z}$.

- The variance decreases with $1/S$ and the standard deviation decreases with $1/\sqrt{S}$.

- *Law of large numbers*: If *i.i.d.* samples $\boldsymbol{z}^i \sim p(\boldsymbol{z})$, then $\hat{f}$ converges (in prob.) to $\bar{f}$ as $S \to \infty$,

$$\hat{f} = \frac{1}{S} \sum_{i=1}^{S} f(\boldsymbol{z}^i) \quad \to \quad \bar{f} = \int f(\boldsymbol{z}) p(\boldsymbol{z}) \mathrm{d}\boldsymbol{z},$$

  assuming $\bar{f}$ exists.

Monte Carlo integration III

- The Monte Carlo estimator and its variance:

$$\bar{f} \approx \hat{f} = \frac{1}{S}\sum_{i=1}^{S} f(\boldsymbol{z}^i), \qquad \mathbb{V}\left[\hat{f}\right] = \frac{1}{S}\mathbb{V}\left[f(\boldsymbol{z})\right]$$

- The *Monte Carlo Standard Error* (MCSE) is the expected difference between $\bar{f}$ and $\hat{f}$:

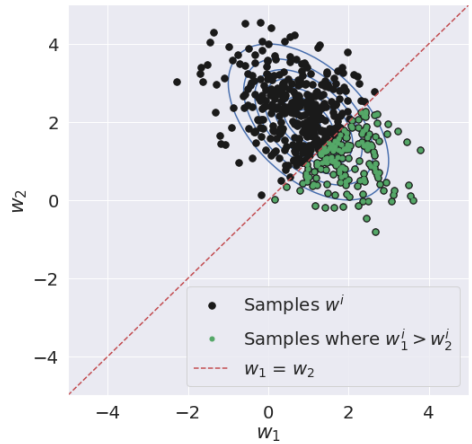
$$\text{MCSE}_f = \frac{1}{\sqrt{S}}\sqrt{\mathbb{V}\left[f(\boldsymbol{z})\right]}$$

Exercise: DTU Learn quiz: The Monte Carlo Standard Error (Lecture 8)

- Suppose we use S samples to estimate $\bar{f}$ and the resulting MCSE is 10.
- If our goal is to reduce the MCSE by a factor of 10, how many samples S' should we use instead?

Example

- Suppose our parameter of interest is $\boldsymbol{w} \in \mathbf{R}^2$ and that $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. What is the probability of $w_1 > w_2$?



- First we phrase this probability as an expectation

$$P(w_1 > w_2) = \int \mathbb{I}\left[w_1 > w_2\right] \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \mathrm{d}\boldsymbol{w} = \mathbb{E}\left[\mathbb{I}\left[w_1 > w_2\right]\right]$$

- We generate samples $\boldsymbol{w}^i \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for $S = 500$, then

$$P(w_1 > w_2) \approx \frac{1}{S} \sum_{i=1}^{S} \mathbb{I}\left[w_1^i > w_2^i\right] = 0.314$$

- The empirical variance is

$$\hat{\mathbb{V}}\left[\mathbb{I}\left[w_1^i > w_2^i\right]\right] = 0.215$$

- The Monte Carlo Standard Error (MCSE) is

$$\frac{1}{\sqrt{S}} \sqrt{\mathbb{V}\left[f(\boldsymbol{z})\right]} \approx \frac{1}{\sqrt{500}} \sqrt{\hat{\mathbb{V}}\left[f(\boldsymbol{z})\right]} \approx 0.021$$

- We conclude: $P(w_1 > w_2) \approx 0.31 \pm 0.02$

Simple sampling methods

Ancestral sampling

- Useful for *sampling* from *joint* distributions.
- Consider our linear regression model from week 3:

$$p(\boldsymbol{y}, \boldsymbol{w}|\kappa^2, \sigma^2) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{X}\boldsymbol{w}, \sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \kappa^2\boldsymbol{I})$$

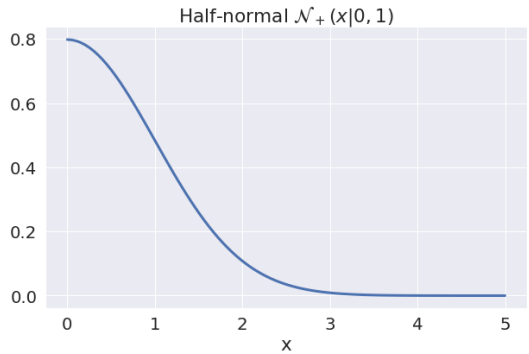
- We optimized the hyperparameters $\kappa^2 = \alpha^{-1}$ and $\sigma^2 = \beta^{-1}$ using the marginal likelihood.
- What if we want to be fully Bayesian and impose *hyperpriors* to the hyperparameters?
- Let's impose a *half-normal distribution* on κ^2 and σ^2

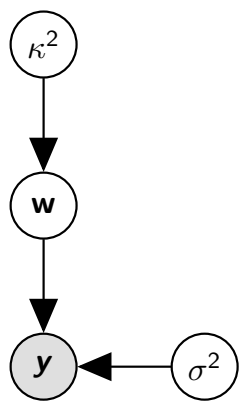
$$p(\kappa^2) = \mathcal{N}_+(\kappa^2|0, 1)$$
$$p(\sigma^2) = \mathcal{N}_+(\sigma^2|0, 1)$$

- The joint distribution of the model becomes

$$p(\boldsymbol{y}, \boldsymbol{w}, \kappa^2, \sigma^2) = p(\boldsymbol{y}|\boldsymbol{w}, \sigma^2)p(\boldsymbol{w}|\kappa^2)p(\sigma^2)p(\kappa^2)$$

- We can use *ancestral sampling* to generate samples from the joint distribution $p(\boldsymbol{y}, \boldsymbol{w}, \kappa^2, \sigma^2)$ and to understand the new model assumptions.



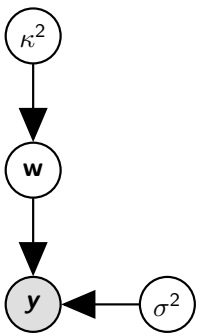


Ancestral sampling

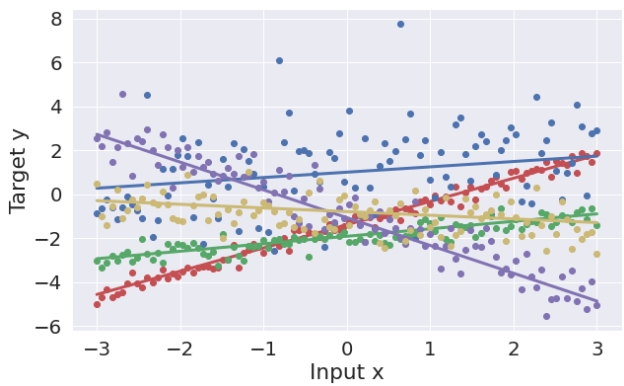
- The joint distribution of the model becomes

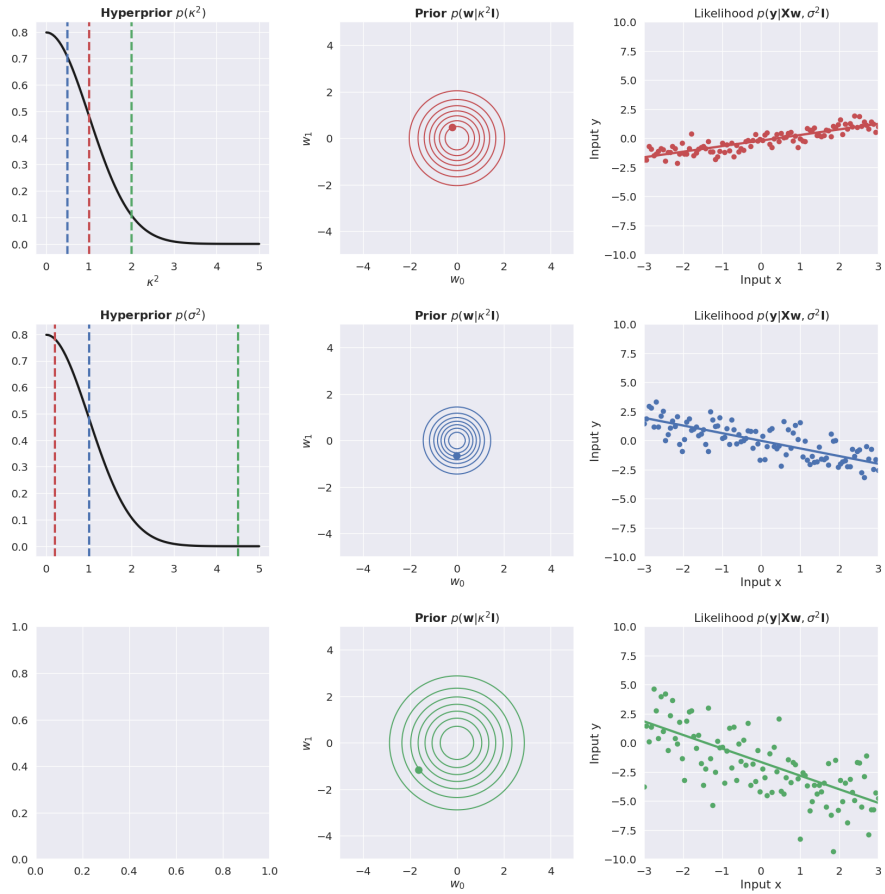
$$p(\boldsymbol{y}, \boldsymbol{w}, \kappa^2, \sigma^2) = p(\boldsymbol{y}|\boldsymbol{w}, \sigma^2)p(\boldsymbol{w}|\kappa^2)p(\kappa^2)p(\sigma^2)$$

- Let's sample some "fake" datasets using this model.



- A few more examples:





Rejection sampling

- *Rejection sampling* is a simple technique for generating samples from $p(\boldsymbol{z})$,

$$p(\boldsymbol{z}) = \frac{1}{Z}\tilde{p}(\boldsymbol{z})$$

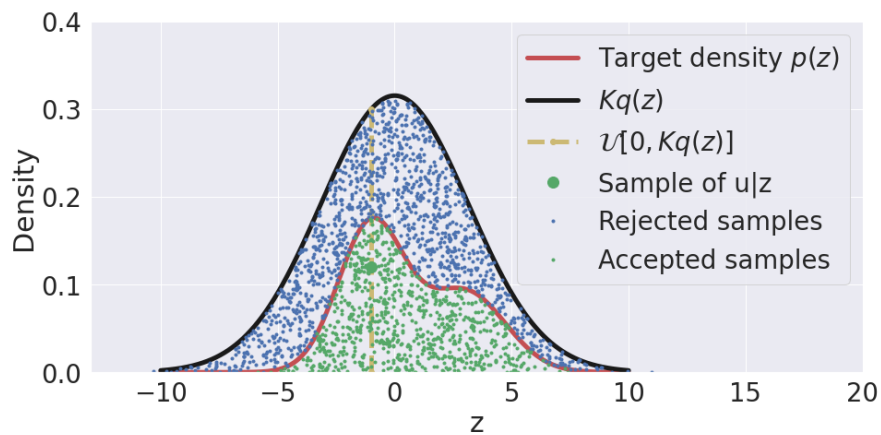
- Let $q(\boldsymbol{z})$ be an "easy" distribution and $K > 0$ a constant such that

$$\tilde{p}(\boldsymbol{z}) \leq Kq(\boldsymbol{z}) \qquad \text{for all } \boldsymbol{z}$$

- Steps in the sampling procedure:

  1. Sample $\boldsymbol{z} \sim q(\boldsymbol{z})$.
  2. Sample $u|\boldsymbol{z}$ from a uniform distribution: $u \sim \mathcal{U}\,[0, Kq(\boldsymbol{z})]$.
  3. If $u > \tilde{p}(\boldsymbol{z})$ reject sample, otherwise keep.

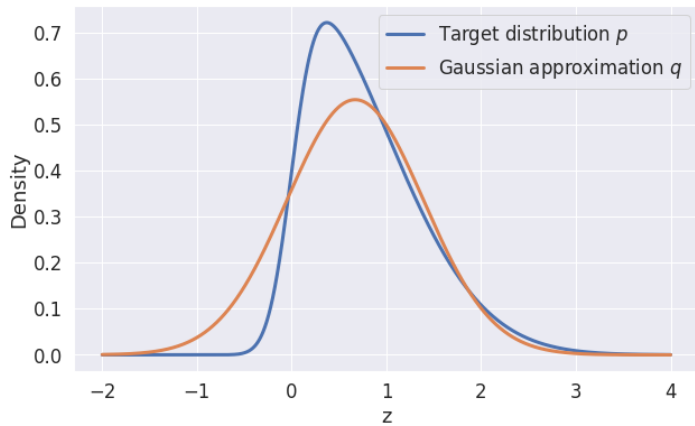
- Pros: easy to use and works well in low dimensions, Cons: In high dimensions, acceptance rate can be low.



Importance sampling

- *Importance sampling* is a technique for estimating the expectation of a distribution $p(z)$ when we cannot sample from p directly.
- Let q be an "easy" distribution.

$$
\begin{aligned}
\mathbb{E}_p\left[f(z)\right] &= \int f(z)p(z)\mathrm{d}z \\
&= \int f(z)\frac{q(z)}{q(z)}p(z)\mathrm{d}z \\
&= \int f(z)\frac{p(z)}{q(z)}q(z)\mathrm{d}z \\
&= \mathbb{E}_q\left[f(z)\frac{p(z)}{q(z)}\right] \\
&\approx \frac{1}{S}\sum_{i=1}^{S} f(z^i)\frac{p(z^i)}{q(z^i)} \equiv \hat{f} \qquad \text{for} \quad z^i \sim q(z)
\end{aligned}
$$



- The ratios $w_i = \frac{p(z^i)}{q(z^i)}$ are called importance weights. The *variance* of the estimator

$$\hat{\sigma}_q^2 = \frac{1}{S}\sum_{i=1}^{S}(w_i f(z^i) - \hat{f})^2$$

How to generate random numbers?

- Assuming we can generate random numbers in the unit interval,

$$u \sim \mathcal{U}[0, 1]$$

- Sampling the Bernoulli distribution $z \sim \text{Ber}(\pi)$:

$$z = \begin{cases} 1 & \text{if} \quad u < \pi \\ 0 & \text{otherwise} \end{cases}$$

- The Box-Muller transform:

$$z = \sqrt{-2 \ln u_1} \cos(2\pi u_2) \sim \mathcal{N}(0, 1)$$

- Location-scale families:

$$x = a + bz \sim \mathcal{N}(x|a, b^2)$$

- Sampling the Exponential-distribution $z \sim \text{Exp}(\lambda)$:

$$z = \frac{-\ln u}{\lambda}$$

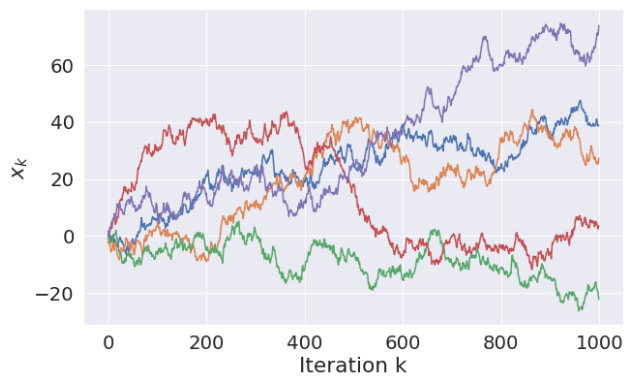
- Will take sampling from "easy" distribution for granted: uniform, Gaussian, gamma, Bernoulli, binomial, Poisson etc. and focus on more complex distributions.

Markov Chain Monte Carlo methods

Markov Chain Monte Carlo methods

- It turns out it is often really hard to obtain *i.i.d.* samples from an arbitrary distribution $p(\boldsymbol{w})$.
- *Markov Chain Monte Carlo* (MCMC) provides a way to sample from almost any distribution $p(\boldsymbol{w})$.
- *A recent survey places the Metropolis algorithm among the ten algorithms that have had the greatest influence on the development and practice of science and engineering in the 20th century.* (Andrieu, 2003).
- The idea is to give up the requirement of i.i.d. samples and construct a "random walk" with specific properties.
- The generated samples are thus correlated, so the variance of the estimator reduces more slowly than for standard Monte Carlo.
- A simple example of a Markov random walk:

$$x_{k+1} = x_k + e_k, \quad \mathcal{N}(0, \sigma^2)$$



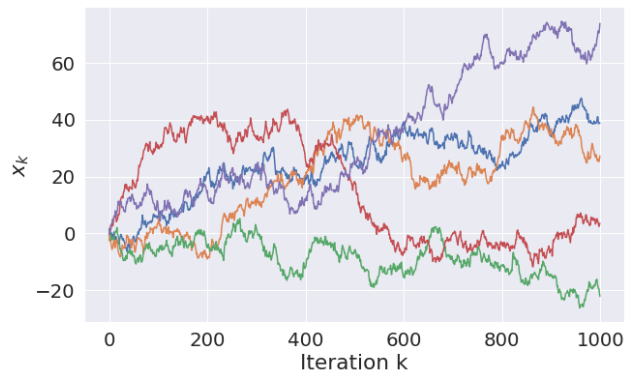
Markov chains in a nutshell

- A first-order Markov chain is defined to be a series of random variables $\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \ldots, \boldsymbol{z}^{(m)}$ with the following conditional independence property

$$p(\boldsymbol{z}^{m+1}|\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \ldots, \boldsymbol{z}^{(m)}) = p(\boldsymbol{z}^{m+1}|\boldsymbol{z}^{(m)})$$

- "If you know the present, forget about the past."

$$x_{k+1} = x_k + e_k, \quad e_k \sim \mathcal{N}(0, \sigma^2)$$



- Implications of the Markov assumption:

$$\begin{aligned} p(\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, \boldsymbol{z}_4) &= p(\boldsymbol{z}_4|\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)p(\boldsymbol{z}_3|\boldsymbol{z}_1, \boldsymbol{z}_2)p(\boldsymbol{z}_2|\boldsymbol{z}_1)p(\boldsymbol{z}_1) \\ &= p(\boldsymbol{z}_4|\boldsymbol{z}_3)p(\boldsymbol{z}_3|\boldsymbol{z}_2)p(\boldsymbol{z}_2|\boldsymbol{z}_1)p(\boldsymbol{z}_1) \end{aligned}$$

- The *transition kernel* is defined as

$$T_m(\boldsymbol{z}^{(m)}, \boldsymbol{z}^{(m+1)}) \equiv p(\boldsymbol{z}^{(m+1)}|\boldsymbol{z}^{(m)})$$

- T_m is *homogeneous* if it is the same for all m.

- Questions:

  1. What is the transition kernel for the random walk of x_k?
  2. Is it homogeneous?

Markov chains for MCMC

- The distribution of $\boldsymbol{z}^{(m+1)}$ is calculated using the *sum rule*:

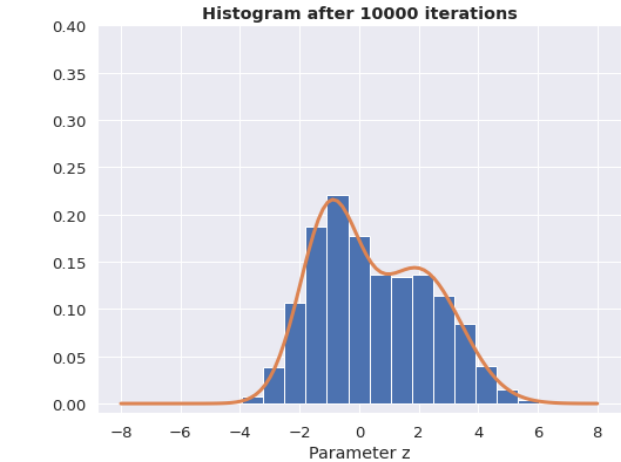
$$p(\boldsymbol{z}^{(m+1)}) = \int p(\boldsymbol{z}^{(m+1)}|\boldsymbol{z}^{(m)})p(\boldsymbol{z}^{(m)})\mathrm{d}\boldsymbol{z}^{(m)}$$

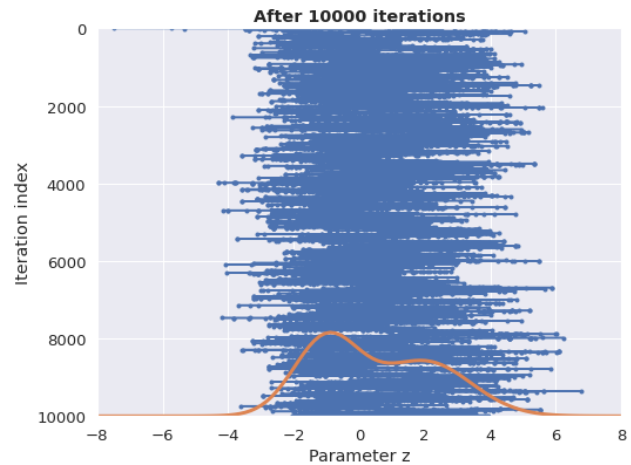
- A distribution $p^*(\boldsymbol{z})$ is said to be *invariant* or *stationary* wrt. the Markov chain if each step does not change the distribution

$$p^*(\boldsymbol{z}) = \int T(\boldsymbol{z}', \boldsymbol{z})p^*(\boldsymbol{z}')\mathrm{d}\boldsymbol{z}'$$

- Central idea of MCMC: design a Markov chain such that the target distribution $p(\boldsymbol{z})$ is invariant wrt. the chain.

- Informally: we design a very specific random walk that will explore the space of $p(\boldsymbol{z})$ in a way, where the random walk spends most time in the regions of high density.





Markov Chain Monte Carlo and the Metropolis algorithm

- The *Metropolis* and the *Metropolis-Hastings* algorithms give a recipe for designing such Markov chains.
- Suppose our goal is to sample from the target distribution $p(\boldsymbol{\theta})$ and we assume $q(\boldsymbol{\theta}^\star|\boldsymbol{\theta}^{k-1})$ to be a *symmetric proposal distribution*.

The Metropolis algorithm

- Start from some initial value $\boldsymbol{\theta}^0$.
- Repeat for $k = 1$ to K:
  1. Given the last value $\boldsymbol{\theta}^{k-1}$, generate a *candidate sample* using the proposal distribution
  $$\boldsymbol{\theta}^\star \sim q(\boldsymbol{\theta}^\star|\boldsymbol{\theta}^{k-1})$$
  2. Compute the *acceptance probability* A_k,
  $$A_k = \min\left(1, \frac{p(\boldsymbol{\theta}^\star)}{p(\boldsymbol{\theta}^{k-1})}\right)$$
  3. Simulate $u_k \sim \mathcal{U}(0,1)$ and define $\boldsymbol{\theta}^k$ as
  $$\boldsymbol{\theta}^k = \begin{cases} \boldsymbol{\theta}^\star & \text{if } u_k < A_k \\ \boldsymbol{\theta}^{k-1} & \text{otherwise} \end{cases}$$

The Metropolis algorithm: example

- Suppose we use a Gaussian proposal,

$$q(\boldsymbol{\theta}^\star|\boldsymbol{\theta}^{k-1}) = \mathcal{N}(\boldsymbol{\theta}^\star|\boldsymbol{\theta}^{k-1}, \tau \boldsymbol{I})$$

- **Key steps** in the Metropolis algorithm

1. Generate a *candidate sample*:

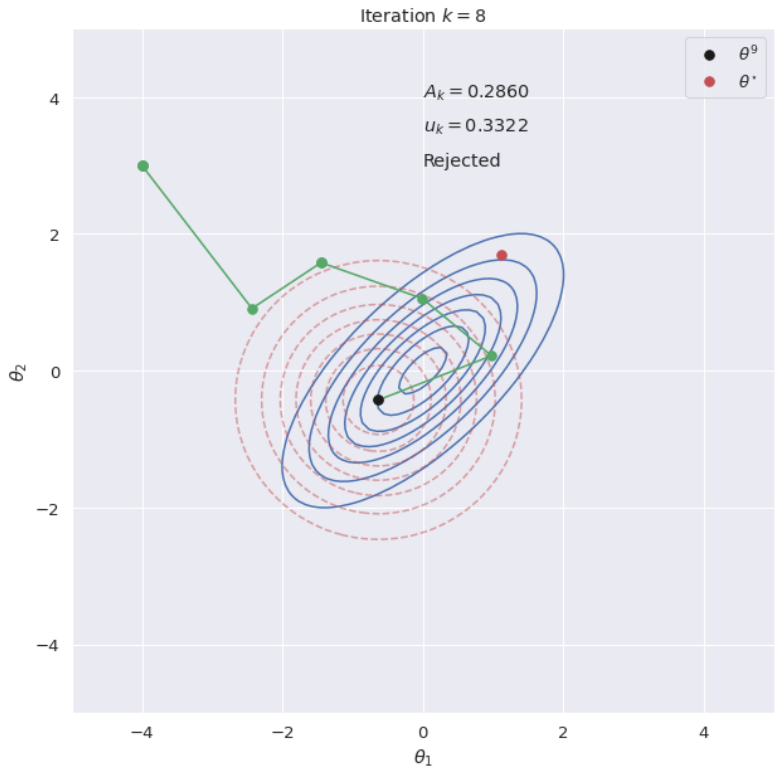
$$\boldsymbol{\theta}^\star \sim q(\boldsymbol{\theta}^\star|\boldsymbol{\theta}^{k-1})$$

2. Compute the *acceptance probability*:

$$A_k = \min\left(1, \frac{p(\boldsymbol{\theta}^\star)}{p(\boldsymbol{\theta}^{k-1})}\right)$$

3. Simulate $u_k \sim \mathcal{U}(0,1)$ and define $\boldsymbol{\theta}^k$ as

$$\boldsymbol{\theta}^k = \begin{cases} \boldsymbol{\theta}^\star & \text{if } u_k < A_k \\ \boldsymbol{\theta}^{k-1} & \text{otherwise} \end{cases}$$



A lot of questions...

- Wait? We don't know the density values for the posterior distribution?

$$p(\boldsymbol{\theta}|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{y})} = \frac{p(\boldsymbol{y},\boldsymbol{\theta})}{p(\boldsymbol{y})}$$

- *The normalization constant cancels out* in the acceptance probabilities

$$A_k = \min\left(1, \frac{p(\boldsymbol{\theta}^\star|\boldsymbol{y})}{p(\boldsymbol{\theta}^{k-1}|\boldsymbol{y})}\right)$$

- Therefore, we only need to be able evaluate $p(\boldsymbol{\theta}|\boldsymbol{y})$ up to a constant.

- When do we know whether the chain has converged?

- How many samples to collect?