Bayesian Computations – MCMC

Felipe Uribe

Computational Engineering
School of Engineering Sciences
Lappeenranta-Lahti University of Technology (LUT)

Special Course on Inverse Problems
Lappeenranta, FI — January-February, 2024

Overview

MCMC changed our the emphasis from "closed form" solutions to algorithms, expanded our impact to solving "real" applied problems, expanded our impact to improving numerical algorithms using statistical ideas, and led us into a world where "exact" now means "simulated"!!

1. Introduction

Bayesian inverse problems

- All unknowns are represented as random variables with prior densities defined e.g., with respect to the Lebesgue measure (or counting measure if discrete).
- Prior information given probabilistically, i.e., probability density functions; used as regularization.
- We define a Bayesian inverse problem, as the task of characterizing the density

$$\pi(\boldsymbol{x}) := \pi_{\text{pos}}(\boldsymbol{x} \mid \boldsymbol{y}) = \frac{1}{Z}\pi_{\text{like}}(\boldsymbol{y} \mid \boldsymbol{x})\pi_{\text{pr}}(\boldsymbol{x}). \tag{1}$$

  - $\pi_{\text{pr}}(\boldsymbol{x})$ is the prior probability density.
  - $\pi_{\text{like}}(\boldsymbol{y} \mid \boldsymbol{x})$ is the likelihood function.
  - $Z = \int_{\mathbb{R}^d} \pi_{\text{like}}(\boldsymbol{y} \mid \boldsymbol{x})\, \pi_{\text{pr}}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}$ is the normalizing constant of the posterior density, called the model evidence.

Summarizing posterior inferences

- The *posterior mean* (PM) (or conditional mean) [1]:

$$\boldsymbol{x}_{\mathrm{PM}} = \mathbb{E}_{\pi_{\mathrm{pos}}}[\boldsymbol{x}] = \int_{\mathbb{R}^d} \boldsymbol{x}\, \pi_{\mathrm{pos}}(\boldsymbol{x} \mid \boldsymbol{y})\, \mathrm{d}\boldsymbol{x}. \tag{2}$$

- Posterior credible sets, posterior quantiles (abusing the notation):

$$\mathbb{P}[\boldsymbol{x} > \boldsymbol{a}] = \int_{\boldsymbol{a}}^{\infty} \pi_{\mathrm{pos}}(\boldsymbol{x} \mid \boldsymbol{y})\, \mathrm{d}\boldsymbol{x}. \tag{3}$$

- Posterior moments and statistics: standard deviation, median, MAD, etc.
- Posterior realizations for direct assessment. Posterior marginals (e.g., via a `plotmatrix`).

[1] Optimality: it minimizes the expected MSE (Bayes risk) for any prior and likelihood, as long as the second order moments exist.

General comments (i)

- Until now we have simply assumed that we can draw random variables, vectors and fields from any desired distribution.
- For some problems, we cannot do this either at all, or in a reasonable amount of time. It is often feasible however to draw dependent samples whose distribution is close to and indeed approaches the desired one.
- In Markov chain Monte Carlo (MCMC) we do this by sampling $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ from a Markov chain constructed so that the distribution of $\boldsymbol{x}_i$ approaches the target distribution.
- The primary method is the Metropolis algorithm, which was named one of the ten most important algorithms of the twentieth century.

General comments (ii)

- We have multiple MCMC algorithms to draw samples from a target posterior distribution.
- As usual, we estimate an expectation $\mu = \mathbb{E}[f(\boldsymbol{x})] = \int f(\boldsymbol{x})\pi(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$:

$$\mu \approx \hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}_i), \tag{4}$$

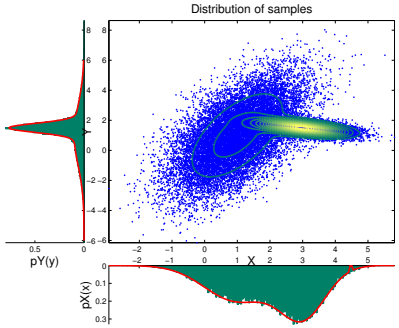
in the MCMC context, we have two main challenges:

(i) The draws $\boldsymbol{x}_i$ have a distribution that approaches π, which is usually not equal to π, hence, the estimate is biased.

(ii) The $\boldsymbol{x}_i$ are (in general) statistically dependent, and thus $\hat{\mu}$ is harder to estimate. In extreme cases, the $\boldsymbol{x}_i$ can get stuck in some subset of their domain and then $\hat{\mu}$ will fail to converge.

MCMC: basic timeline

- Nicholas Metropolis (1953) - Metropolis algorithm for generating samples from the Boltzmann distribution.
- Wilfred K. Hastings (1970) - Metropolis–Hastings algorithm, the most common MCMC.
- Julian Besag - Gibbs sampler (1974), MALA algorithm (1994).
- Simon Duane et al. (1987) - Hamiltonian MC.
- Peter Green (1995) - Reversible jump MCMC.
- Joris Bierkens (2015) - non-reversible MCMC.



This lecture...

- The lecture is based on multiple references. However, we mostly follow Chapters 11 and 12 of the book by **Art Owen**[2], which is freely available online.

[2] A. B. Owen. *Monte Carlo theory, methods and examples*. `artowen.su.domains/mc/`, 2018.

2. Markov chains

Markov chains

- Our solution to hard sampling problems will be to run a Markov chain for a long time so that the values of the chain have a distribution which approaches our target density π which is defined on $\mathcal{X}$.

- In plain terms, a Markov chain is a "memoryless" stochastic process: to know a future state, we just need to know the current state. This is called the *Markov property*.

- Formally, a discrete time Markov chain $X = \{X_n\}$ with a discrete state space $\mathcal{X}$ satisfies the Markov property:

$$\mathbb{P}[X_{n+1} = j \mid X_n = i, X_{n-1} = i_{n-1}, \ldots, X_0 = i_0] = \mathbb{P}[X_{n+1} = j \mid X_i = i]\,; \qquad (5)$$

  "past and future are conditionally independent given the present".

Markov chains

- In practice, we work with **time-homogeneous** Markov chains for which the *transition probabilities* $\mathbb{P}[X_{n+1} = j \mid X_n = i] = q_{ij}$ are independent of n. In this case, we can arrange the transition probabilities as a (transition) matrix $\mathbf{Q} \in \mathbb{R}^{m \times m}; \{q_{ij}\}$ (rows sum to one).

- Suppose that at step n, X_n has distribution $\boldsymbol{s} \in \mathbb{R}^{1 \times m}$ (row probability vector); think of this as a PMF. Here, the element $s_i = \mathbb{P}[X_n = i]$ denotes the probability that the chain is in state i at step n.

- A natural question is to what is the probability that the chain has state value j at step $n+1$? Using the total probability theorem,

$$\mathbb{P}[X_{n+1} = j] = \sum_i \mathbb{P}[X_{n+1} = j \mid X_n = i]\, \mathbb{P}[X_i = i] = \sum_i s_i\, q_{ij} \quad j\text{th entry of } \boldsymbol{s}\mathbf{Q}; \quad (6)$$

  successive iteration of this equation describes the evolution of the chain, i.e., $\boldsymbol{s}\mathbf{Q}$ is the distribution at $n+1$: "one step in the future, right multiply by $\mathbf{Q}$".

Markov chains

Some properties:

- We say that a Markov chain is **irreducible** (connected), if it is possible to get from anywhere to anywhere. Otherwise, it is called *reducible*.
- We say that a state is **recurrent**, if starting there, the chain has probability 1 of returning to that state. Otherwise, it is called *transient*.
- We say that a Markov chain is **periodic**, if the states start repeating themselves with a given period. Otherwise, it is called *aperiodic*.
- We say that $\boldsymbol{s}$ is **stationary** for a Markov chain with transition matrix $\boldsymbol{Q}$, if $\boldsymbol{s}\mathbf{Q} = \boldsymbol{s}$.

Markov chains

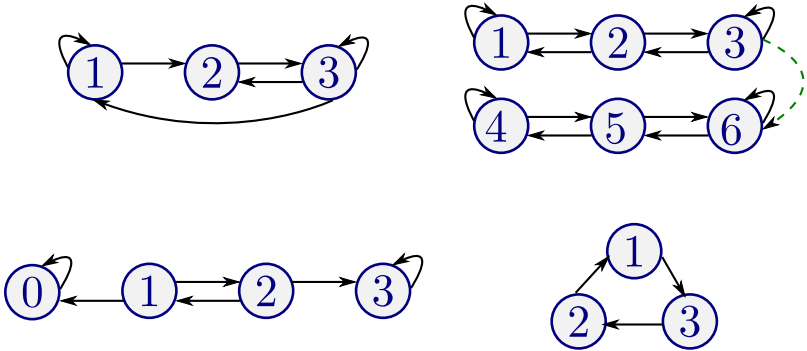


Figure: Four different Markov chains. How can they be classified?

Markov chains

- For an irreducible Markov chain (with finitely many states), we have the following[3]:
  - (i) The stationary distribution $\boldsymbol{s}$ exists.
  - (ii) The stationary distribution is unique.
  - (iii) If the Markov chain is further aperiodic ($\mathbf{Q}^m$ is strictly positive for some m), then $\mathbb{P}[X_n = i] \to s_i$ as $n \to \infty$.

- We normally work with Markov chains that are "easy to deal with". This class of chains is called *time reversible*.

- A Markov chain with transition matrix $\mathbf{Q}$ is **reversible**, if there is a $\boldsymbol{s}$ such that

$$s_i\, q_{ij} = s_j\, q_{ji} \quad \forall i, j \qquad \text{(detailed balance condition)}. \tag{7}$$

- If the Markov chain is reversible with respect to $\boldsymbol{s}$, then $\boldsymbol{s}$ is the stationary distribution.

[3] see Theorem 1.8.3 in [12]

Markov chains (example)

- Suppose the state space are (Rain, Sunny, Cloudy) and weather follows a Markov process. Hence, the probability of tomorrow's weather simply depends on today's weather, and not any other previous days (example from [17]).
- The probability transitions are given by

$$\mathbf{Q} = \begin{bmatrix} 0.5 & 0.25 & 0.25 \\ 0.5 & 0 & 0.5 \\ 0.25 & 0.25 & 0.5 \end{bmatrix}. \tag{8}$$

- Suppose today is sunny (and then rainy). What is the expected weather two days from now? Seven days?

  If $\boldsymbol{s}_0 = [0, 1, 0]$ $\quad \boldsymbol{s}_2 = \boldsymbol{s}_0\mathbf{Q}^2 = [0.375, 0.25, 0.375]$ $\quad \boldsymbol{s}_7 = \boldsymbol{s}_0\mathbf{Q}^7 = [0.4, 0.2, 0.4]$

  If $\boldsymbol{s}_0 = [1, 0, 0]$ $\quad \boldsymbol{s}_7 = \boldsymbol{s}_0\mathbf{Q}^7 = [0.4, 0.2, 0.4].$

  Note that after a sufficient amount of time, the expected weather in independent of the starting value, i.e., the chain has reached a stationary distribution.

Markov chains

- The basic idea of discrete-state Markov chains can be generalized to a continuous state Markov process by having a probability kernel $P(\boldsymbol{x}, \boldsymbol{x}')$ that satisfies

$$\int_{\mathcal{X}} P(\boldsymbol{x}, \boldsymbol{x}') \, \mathrm{d}\boldsymbol{x}' = 1, \qquad \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}. \tag{9}$$

- Now, the reversibility (detailed balance) condition is:

$$\int_A \pi(\boldsymbol{x}) P(\boldsymbol{x}, \boldsymbol{x}') \, \mathrm{d}\boldsymbol{x} = \int_B \pi(\boldsymbol{x}') P(\boldsymbol{x}', \boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}', \tag{10}$$

  where $\boldsymbol{x} \in A$, $\boldsymbol{x}' \in B$, and $A, B \subset \mathcal{X}$, $P(\cdot, \cdot)$ denotes a Markov transition kernel.

- Different approaches exist to generate kernels P that ensure eq. (10), and hence setting π as the stationary distribution.

3. MCMC – Metropolis–Hastings

Metropolis–Hastings

- If we make a good selection for the transition kernel, it could asymptotically converge to a target distribution independently of where we started from.
- More importantly, we can use the realizations of the Markov chain in Monte Carlo estimators i.e., we can average across the path. However, even if X_n were exact draws, they are not independent anymore.
- In Metropolis–Hastings (MH), the condition eq. (10) is satisfied by separating the transition into two stages: the proposal and the acceptance/rejection steps.
- These are represented respectively by the **proposal distribution** $q(\boldsymbol{x} \mid \boldsymbol{y})$ which accounts for the conditional probability of $\boldsymbol{x}$ given the proposed state $\boldsymbol{y}$, and the **acceptance probability** $\alpha(\boldsymbol{x}, \boldsymbol{y})$ which expresses the conditional probability of accepting the proposed state $\boldsymbol{y}$.

Metropolis–Hastings

- The transition kernel can be written as:

$$P(\boldsymbol{x} \mid \boldsymbol{y}) = \underbrace{\int_B q(\boldsymbol{x} \mid \boldsymbol{y})\alpha(\boldsymbol{x}, \boldsymbol{y}) \, \mathrm{d}\boldsymbol{y}}_{\text{acceptance}} + \underbrace{\mathbb{1}(\boldsymbol{x} \in B) \int_{\mathcal{X}} q(\boldsymbol{x} \mid \boldsymbol{y})(1 - \alpha(\boldsymbol{x}, \boldsymbol{y})) \, \mathrm{d}\boldsymbol{y}}_{\text{rejection}}. \tag{11}$$

- Substituting eq. (11) into eq. (10), yields

$$\int_A \pi(\boldsymbol{x}) \int_B q(\boldsymbol{x} \mid \boldsymbol{y})\alpha(\boldsymbol{x}, \boldsymbol{y}) \, \mathrm{d}\boldsymbol{y} \, \mathrm{d}\boldsymbol{x} = \int_B \pi(\boldsymbol{y}) \int_A q(\boldsymbol{y} \mid \boldsymbol{x})\alpha(\boldsymbol{y}, \boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{y}, \tag{12}$$

- Which can be written in compact form as

$$\int_{A \times B} \pi(\boldsymbol{x}) q(\boldsymbol{x} \mid \boldsymbol{y})\alpha(\boldsymbol{x}, \boldsymbol{y}) \, \mathrm{d}\boldsymbol{y} \mathrm{d}\boldsymbol{x} = \int_{A \times B} \pi(\boldsymbol{y}) q(\boldsymbol{y} \mid \boldsymbol{x})\alpha(\boldsymbol{y}, \boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \mathrm{d}\boldsymbol{y}. \tag{13}$$

Metropolis–Hastings

- The equality eq. (13) holds if

$$\pi(\boldsymbol{x})q(\boldsymbol{x} \mid \boldsymbol{y})\alpha(\boldsymbol{x}, \boldsymbol{y}) = \pi(\boldsymbol{y})q(\boldsymbol{y} \mid \boldsymbol{x})\alpha(\boldsymbol{y}, \boldsymbol{x}) \tag{14a}$$

$$\frac{q(\boldsymbol{x} \mid \boldsymbol{y})\alpha(\boldsymbol{x}, \boldsymbol{y})}{q(\boldsymbol{y} \mid \boldsymbol{x})\alpha(\boldsymbol{y}, \boldsymbol{x})} = \frac{\pi(\boldsymbol{y})}{\pi(\boldsymbol{x})} \tag{14b}$$

$$\frac{\alpha(\boldsymbol{x}, \boldsymbol{y})}{\alpha(\boldsymbol{y}, \boldsymbol{x})} = \frac{\pi(\boldsymbol{y})q(\boldsymbol{y} \mid \boldsymbol{x})}{\pi(\boldsymbol{x})q(\boldsymbol{x} \mid \boldsymbol{y})}. \tag{14c}$$

- An acceptance mechanism that fulfills the condition in eq. (14c) is proposed by [9] (based on [10], which uses a symmetric proposal distribution)

$$\alpha(\boldsymbol{x}, \boldsymbol{y}) = \min\left(1, \frac{\pi(\boldsymbol{y})q(\boldsymbol{y} \mid \boldsymbol{x})}{\pi(\boldsymbol{x})q(\boldsymbol{x} \mid \boldsymbol{y})}\right); \tag{15}$$

  an unnormalized π can be used as the normalizing constants cancel out in the ratio.

Metropolis–Hastings

Algorithm 1: Metropolis–Hastings

Initialize $\boldsymbol{x}_0$;
for $i = 1$ *to* n **do**
    Sample $\boldsymbol{x}^\star \sim q(\cdot \mid \boldsymbol{x}_{i-1})$;
    Compute the acceptance probability

$$\alpha = \min\left(1, \frac{\pi(\boldsymbol{x}^\star) q(\boldsymbol{x}_{i-1} \mid \boldsymbol{x}^\star)}{\pi(\boldsymbol{x}_{i-1}) q(\boldsymbol{x}^\star \mid \boldsymbol{x}_{i-1})}\right)$$

    Sample $u \sim \mathcal{U}(0, 1)$;
    if $u \leq \alpha$ **then**
        Accept $\boldsymbol{x}_i = \boldsymbol{x}^\star$
    else
        Reject $\boldsymbol{x}_i = \boldsymbol{x}_{i-1}$
    end
end

Metropolis–Hastings (burn-in and thinning)

- One of the difficulties with MH is the potentially strong dependence in the Markov chain states (samples). A common practice is ignore the first $n_b < n$ generated points and estimate μ by

$$\hat{\mu} = \frac{1}{n - n_b} \sum_{i=n_b+1}^{n} f(\boldsymbol{x}_i). \tag{16}$$

- This practice is called **burn-in** or warm-up. The distribution of $\boldsymbol{X}_i$ usually only approaches π as i increases and so the first few observations might be very unrepresentative. Including them could bias the answer.

- We could think of the burn-in period as one way of finding a good starting point for the simulation. Some authors recommend $n_b = 0.5n$.

- We can also "thin" the chain by selecting every other n_tth sample (discarding those in between). This process is also called *subsampling*.

Metropolis–Hastings (general comments)

- The MH acceptance probability is a default choice for $\alpha(\boldsymbol{x}, \boldsymbol{y})$ that provides detailed balance. With this default in hand we can then search for good proposal distributions $q(\boldsymbol{x}, \boldsymbol{y})$ to suit any given problem.

- It is recommended to formulate the algorithm using log densities, for stability.

- The performance of MH deteriorates with increasing dimension of the parameter space. In this case, the acceptance probability at each step becomes very small as the dimension increases, resulting in slow convergence rates and a large number of repeated samples.

Barker MCMC ⋆

- Recall the Metropolis–Hastings acceptance probability that holds the reversibility condition:

$$\alpha_{\mathrm{MH}}(\boldsymbol{x}, \boldsymbol{y}) = \min\left(1, \frac{\pi(\boldsymbol{y}) q(\boldsymbol{y} \mid \boldsymbol{x})}{\pi(\boldsymbol{x}) q(\boldsymbol{x} \mid \boldsymbol{y})}\right). \tag{17}$$

- Several other choices are possible. One alternative proposed by Barker[4] is the acceptance probability

$$\alpha_{\mathrm{B}}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\pi(\boldsymbol{y}) q(\boldsymbol{y} \mid \boldsymbol{x})}{\pi(\boldsymbol{x}) q(\boldsymbol{x} \mid \boldsymbol{y}) + \pi(\boldsymbol{y}) q(\boldsymbol{y} \mid \boldsymbol{x})}. \tag{18}$$

- In general α_{MH} is preferred over α_{B} because, for the same choice of proposal $q(\boldsymbol{y} \mid \boldsymbol{x})$, α_{MH} will result in Markov chains that produce ergodic averages with smallest asymptotic variance[5]. In particular, α_{MH} will maximize the probability of moving from $\boldsymbol{x}$ to $\boldsymbol{y}$.

[4] A. A. Barker. "Monte Carlo calculations of the radial distribution functions for a proton-electron plasma". In: *Australian Journal of Physics* 18 (1965), pp. 119–1347.

[5] L. Tierney. "A note on Metropolis-Hastings kernels for general state spaces". In: *Annals of applied probability* (1998), pp. 1–9.

Independence sampler

- Perhaps the simplest proposal mechanism is to take iid proposals from some distribution q that does not depend on the present location $\boldsymbol{x}$. Then $q(\boldsymbol{x}^\star \mid \boldsymbol{x}) = q(\boldsymbol{y})$.

- The MH proposal for this independence sampler, simplifies to

$$\alpha(\boldsymbol{x}, \boldsymbol{x}^\star) = \min\left(1, \frac{\pi(\boldsymbol{x}^\star)q(\boldsymbol{x})}{\pi(\boldsymbol{x})q(\boldsymbol{x}^\star)}\right); \tag{19}$$

  the independence sampler is also called the Metropolized independence sampler.

- The independence sampler is closely related to importance sampling. It is generally safer to have slightly heavy tails in an independence sampler proposal q instead of light tail.

Random walk Metropolis

- A random walk is a process where the increments $\boldsymbol{z}_i = \boldsymbol{x}_i - \boldsymbol{x}_{i-1}$ are iid.
- In **random walk Metropolis** (RWM) the proposals take the form $\boldsymbol{x}_i^\star = \boldsymbol{x}_i + \boldsymbol{z}_i$, where $\boldsymbol{z}_i$ are iid random vectors.
- Suppose that $\boldsymbol{z} \sim q$. Then $q(\boldsymbol{x}^\star \mid \boldsymbol{x})$ can be written $q(\boldsymbol{x}^\star - \boldsymbol{x})$, where we now use q to also represent the probability density of $\boldsymbol{z}$.
- We will focus on random walks where the distribution q is symmetric (e.g., a Gaussian) and

$$\alpha(\boldsymbol{x}, \boldsymbol{x}^\star) = \min\left(1, \frac{\pi(\boldsymbol{x}^\star)}{\pi(\boldsymbol{x})}\right), \tag{20}$$

because the proposal ratio cancels.

Random walk Metropolis

- It is common in MCMC that we have to tune our proposals. For this we have to specify the so-called **proposal scale** β. Under a Gaussian proposal, this is equal to a standard deviation.

- If the acceptance rate is very small, we can infer that β is too large and then try a smaller value. Conversely, a very high acceptance rate suggests that we should raise β. Under a famous result from [7], it is optimal to tune β so that about 23.4% of proposals are accepted.

- In [7], Gaussian targets and Gaussian proposal were considered, so they were able to study how the proposal variance should depend on the dimension d. They recommend $\beta = 2.4$ for $d = 1$, and $\beta = 2.38/\sqrt{d}$ for large d.

- The acceptance rate when using the optimal β is about 44% for $d = 1$ and decreases rapidly to a limiting value of about 23.4% as $d \to \infty$. In general, as long as the rejection rate is between 15% and 40% the efficiency is close to that of the optimal β.

Random walk Metropolis (example)

- Obtain samples from the probability density,

$$\pi(x) \propto 0.3\exp(-0.2x^2) + 0.7\exp(-0.2(x-10)^2).$$

- We use a Gaussian proposal $\mathcal{N}(y; x, \beta^2)$ (centered at x), and the initial state is defined as $x_0 = 5$.

- To show the effect of the proposal scale, simulations for different values of the standard deviation, $\beta = 0.1$, $\beta = 1$, $\beta = 14$ and $\beta = 50$ are performed.

- We run $n = 5 \times 10^3$ samples without burn-in.

Random walk Metropolis (example)

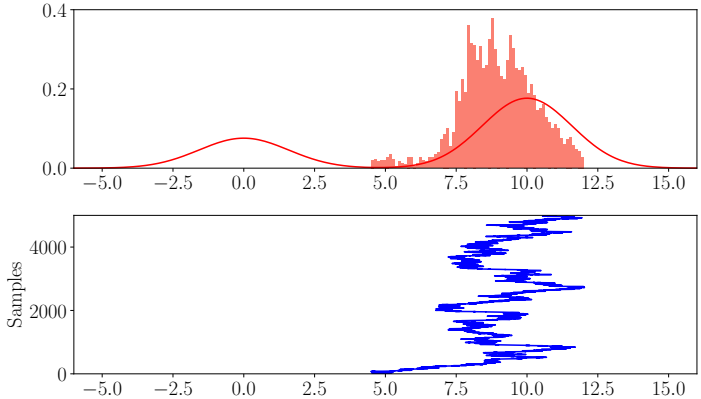


Figure: Bimodal target: scale parameter $\beta = 0.1$. Acceptance probability 0.98. The plot at the bottom is called a **trace plot**.

Random walk Metropolis (example)

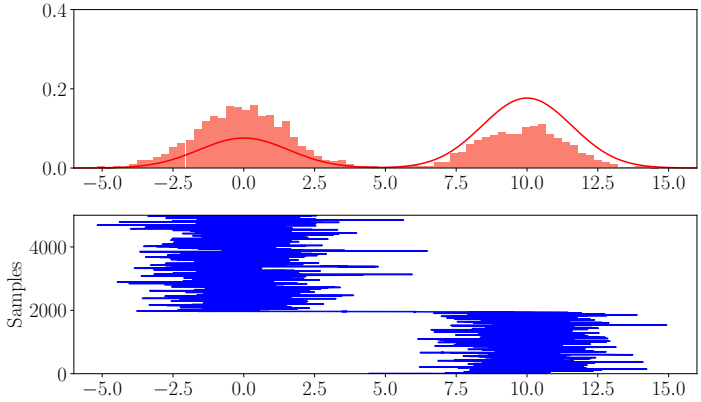


Figure: Bimodal target: scale parameter $\beta = 0.5$. Acceptance probability 0.80.

Random walk Metropolis (example)

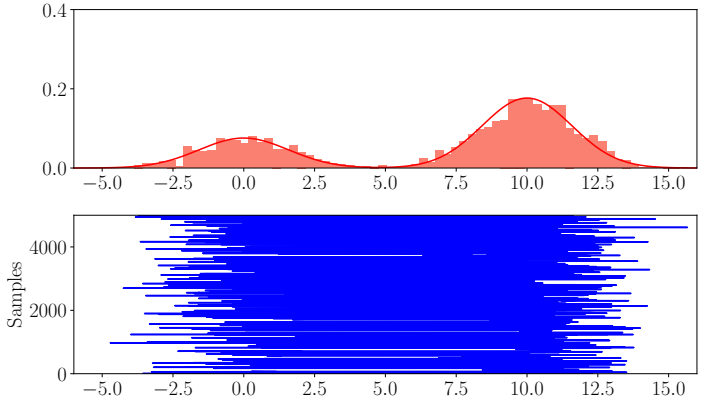


Figure: Bimodal target: scale parameter $\beta = 14$. Acceptance probability 0.23.

Random walk Metropolis (example)

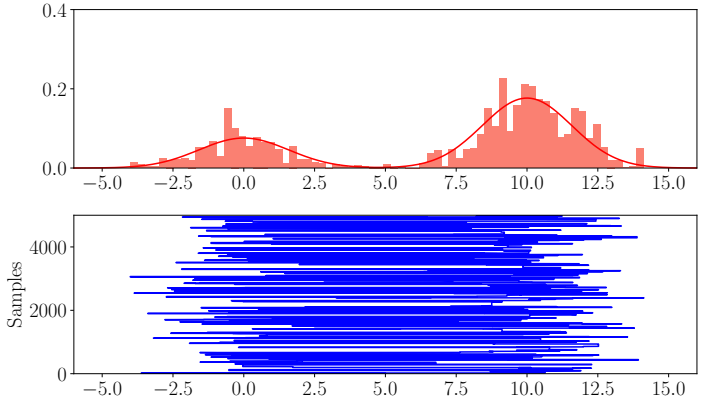


Figure: Bimodal target: scale parameter $\beta = 50$. Acceptance probability 0.08.

Adaptive RWM (I)

- We have to specify a suitable proposal scaling for our MCMC algorithm. This requires some parameter study. We can, however, employ different techniques that perform adaptation of the proposal scale such that the acceptance probability of the sampler is optimal.

- Learn a better proposal $q(\boldsymbol{x}^\star \mid \boldsymbol{x})$ from past samples: (i) an appropriate proposal scale, and (ii) an appropriate proposal orientation and anisotropy; this is essential in problems with strong correlation in π.

- Adaptive Metropolis scheme of [8]
  - (i) Covariance matrix at step i

$$\boldsymbol{\Sigma}_i^\star = \beta^2 \, \widehat{\mathrm{Cov}}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i) + \beta^2 \epsilon \mathbf{I}_d, \tag{21}$$

    where $\epsilon > 0$ and $\beta^2 = 2.4^2/d$.
  - (ii) Proposals are Gaussians at $\boldsymbol{x}_i$. Use a covariance $\boldsymbol{\Sigma}_0$ for the first i_0 steps, then use $\boldsymbol{\Sigma}_i^\star$.
  - (iii) Chain is not Markov. Nonetheless, one can prove that the chain converges to π.

Adaption with global adaptive scaling (II)

- Vanishing adaptation ensures that the chain depends less and less on recently visited states of the chain. This sets controlled MCMC algorithms that produce samples asymptotically distributed according to π.
- Adaptive Metropolis scheme of [1]. At step i update the proposal scale as

$$\log(\beta_{i+1}) = \log(\beta_i) + \gamma_{i+1}(\alpha(x_i, x^\star) - \alpha^\star),$$

  where $\{\gamma_i\} \subset (0, \infty)$ is a sequence of stepsizes which ensures that the variations of the chain states $\{\boldsymbol{x}_i\}$ vanish, $\alpha^\star$ is the desired acceptance rate. Note that this is just a standard *Robbins–Monro* recursion.
- The standard approach consists of choosing the sequence $\{\gamma_i\}$ deterministic and non-increasing, but it is also possible to choose $\{\gamma_i\}$ random. A standard choice is to make $\gamma_i = C/i^c$, for $c \in (1/(1+\beta), 1]$ and constant C.

4. MCMC – diagnostics

MCMC error and diagnostics

- Two of the hardest problems in MCMC are:
  (i) deciding whether the distribution of $\boldsymbol{x}_i$ has nearly converged to π, and
  (ii) deciding whether the $\boldsymbol{x}_i$ are mixing well.

- While traces are quite useful, they are only one way diagnostics. When they show us that the chain is not sampling π well, we can believe it. Unfortunately a trace can look perfectly good even when the sample is poor.

- The autocorrelation function has the same problem. If it shows slowly decaying autocorrelations, we know there is a problem, but if they decay rapidly we might still have missed part of the space.

- One approach to generating diagnostics is to run multiple independently generated Markov chains, starting them in different places.

MCMC error and diagnostics

- Recall from Monte Carlo that the mean estimator had a variance $\mathbb{V}[\hat{\mu}_n] = \sigma^2/n$, but the samples were iid. In MCMC, we have:

$$\hat{\mu}_n = \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{X}_i) \quad \text{and} \quad \mathbb{V}[\hat{\mu}_n] = \frac{\sigma^2}{n}\tau_f, \tag{22}$$

  where τ_f is the integrated autocorrelation time (or *inefficiency factor*) for the chain $f(\boldsymbol{X}_i)$.

- The *integrated autocorrelation time* (IACT) for a (weakly) stationary process is defined as[6]

$$\tau_f = \sum_{k=-\infty}^{\infty} \rho_f(k) = 1 + 2\sum_{k=1}^{\infty} \rho_k, \tag{23}$$

  where ρ_k is the normalized autocorrelation function at lag k.

[6] this is from an asymptotic formula for the variance of the average of a correlated sequence.

MCMC error and diagnostics

- An unbiased estimator of the IACT is

$$\tau_f \approx \hat{\tau}_f = 1 + 2\sum_{k=1}^{n-1}\left(1 - \frac{k}{n}\right)\rho_k; \tag{24}$$

  at longer lags k, the estimator starts to contain more noise than signal, and summing all the way out to n will result in a very noisy estimate of τ_f. Hence, we typically evaluate this estimator at sample size $m \ll n$.

- We have that the variance of the mean estimator using MCMC samples is:

$$\mathbb{V}[\hat{\mu}_n] \approx \frac{\sigma^2}{n}\hat{\tau}_f. \tag{25}$$

MCMC error and diagnostics

- The inefficiency factor can be used to derive the effective sample size:

$$n_{\text{eff}} = \frac{n}{1 + 2\sum_{k=1}^{\infty} \rho_k} \approx \frac{n}{1 + 2\sum_{k=1}^{n-1} \left(1 - \frac{k}{n}\right) \rho_k}. \tag{26}$$

  the effective sample size is utilized to compare between the variance estimated via correlated MCMC samples and the ideal case of a variance computed from independent draws. Thus, the aim is to obtain an n_{eff} as close as possible to n.

- Other relevant metric used as an indicator of how fast the MCMC chains are mixing is the *mean square jump* (MSJ) distance, defined as

$$\text{MSJ} = \frac{1}{n} \sum_{i=1}^{n} \|x_i - x_{i-1}\|_2^2, \tag{27}$$

  the larger the magnitude of the MSJ, the better the mixing.

MCMC error and diagnostics[7]

- A common test for convergence is the **Geweke test**. This method splits the chain (after burn-in period) into two parts, the first 10% and the last 50%. If the chain is stationary, the averages of these two parts should be approximately equal.

- Other diagnostics are: Gelman–Rubin diagnostic (which compares multiple chains), the Raftery-Lewis diagnostic (which is a method to find a proper burn-in), computing distance between distributions, etc.

- There has long been controversy over whether it is better to simulate one long chain of length n or m chains of length n/m each. There are settings where the single long chain has less bias. Parallel computing changes the picture.

[7] We recommend ArviZ to analize and postprocess MCMC results (`https://python.arviz.org/en/stable/`)

MCMC error and diagnostics (RWM example)

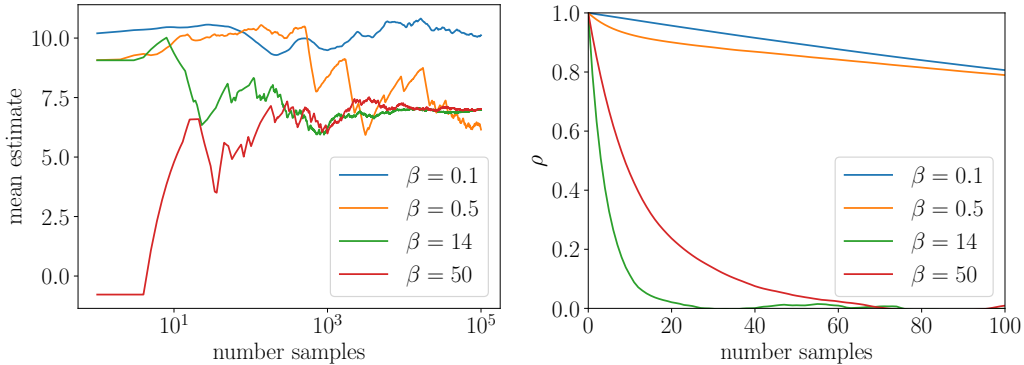


Figure: Cumulative mean and autocorrelation for Markov chain obtained with RWM for proposal different scaling. The n_{eff} is for each case: $[91, 128, 10875, 3578]$, here $n = 10^5$ and $n_b = 0.2n$.

5. MCMC – Gibbs sampler

Gibbs sampler

- We want to sample from $\pi(\boldsymbol{x})$ where $\boldsymbol{x} = [x_1, \ldots, x_{j-1}, x_j, x_{j+1}, \ldots, x_d]$. It might be difficult to construct a good proposal q that changes the whole vector at once.
- However, suppose we can construct good proposals that changes only one (or a few) component(s) of $\boldsymbol{x}$. For example, we consider that $\boldsymbol{x}$ is comprised of x_j and the remaining elements, which we can lump together into $\boldsymbol{x}_{-j}$.
- In the Gibbs sampler, we repeatedly sample one component after another from the appropriate conditional distribution. Many models used in statistics and machine learning have simple conditional distributions for which the Gibbs sampler is easy to use.

Gibbs sampler

- We write the **full conditional distribution** of x_j given $\boldsymbol{x}_{-j}$ as $\pi(x_j \mid \boldsymbol{x}_{-j})$, with marginal $\pi(\boldsymbol{x}_{-j})$.

- There are two versions: in the **random scan** Gibbs sampler, the component to update is chosen at random from $1, \ldots, d$. In the **systematic scan** Gibbs sampler, the components are updated sequentially.

- We can show directly that sampling component j from its full conditional distribution preserves the stationary distribution π. Suppose that $\boldsymbol{x} \sim \pi$ and that we replace x_j by a value $z \sim \pi(x_j|\boldsymbol{x}_{-j})$, obtaining the point $\boldsymbol{y}$ with $y_j = z$ and $y_k = x_k$ for $k \neq j$. Then

$$\pi(\boldsymbol{y}_{-j})\pi(y_j \mid \boldsymbol{y}_{-j}) = \pi(\boldsymbol{y}) \tag{28a}$$

$$\pi(\boldsymbol{x}_{-j})\pi(z \mid \boldsymbol{x}_{-j}) = \pi(\boldsymbol{x}). \tag{28b}$$

Gibbs sampler

- We may also understand the Gibbs sampler by relating it to MH. Suppose that we are at point $\boldsymbol{x}$ and have decided to modify component j of $\boldsymbol{x}$ to take the value z. Let $\boldsymbol{y}$ be the point with $y_j = z$ and $y_k = x_k$ for $k \neq j$. If we use $\boldsymbol{y}$ as the proposal in MH, then

$$\frac{\pi(\boldsymbol{y}_{-j})\pi(z \mid \boldsymbol{x}_{-j})\pi(x_j \mid \boldsymbol{x}_{-j})}{\pi(\boldsymbol{x}_{-j})\pi(x_j \mid \boldsymbol{x}_{-j})\pi(z \mid \boldsymbol{x}_{-j})} = 1, \tag{29}$$

  thus, if we update component j of $\boldsymbol{x}$ by sampling from its full conditional distribution, then we can view this as a MH proposal that is never rejected.

- Random scan Gibbs has detailed balance, while fixed scan does not, but we can construct a reversible Gibbs sampler with symmetric scan.

Gibbs sampler

- Gibbs sampling provides an alternative generation scheme based on successive generations from the full conditional distributions. The Gibbs sampler described now involves a complete scan over the components.

- Given a state $\boldsymbol{x}^{(k)} = \left[x_1^{(k)}, x_2^{(k)}, \ldots, x_d^{(k)}\right]^{\mathsf{T}}$, samples $(k = 1, \ldots, n)$ from π are drawn component by component, as follows

$$
\begin{aligned}
x_1^{(k+1)} &\sim \pi\left(x_1 \,\middle|\, x_2^{(k)}, x_3^{(k)}, \ldots, x_d^{(k)}\right), \\
x_2^{(k+1)} &\sim \pi\left(x_2 \,\middle|\, x_1^{(k+1)}, x_3^{(k)}, \ldots, x_d^{(k)}\right), \\
&\vdots \\
x_d^{(k+1)} &\sim \pi\left(x_d \,\middle|\, x_1^{(k+1)}, x_2^{(k+1)}, \ldots, x_{d-1}^{(k+1)}\right).
\end{aligned}
\tag{30}
$$

- There are many other possible updating strategies for visiting the components of $\boldsymbol{x}$.

Gibbs sampler

Algorithm 2: Random scan Gibbs.

set initial state $\boldsymbol{x}_0$;
for $i = 1$ *to* n **do**
    sample an index j from $\mathcal{U}(1, d)$;
    sample component $z \sim \pi(\cdot \mid \boldsymbol{x}_{-j}^{(i-1)})$;
    set $\boldsymbol{x}^{(i)} = \boldsymbol{x}^{(i-1)}$ and $x_j^{(i)} = z$;
end

Algorithm 3: Fixed scan Gibbs.

set initial state $\boldsymbol{x}_0$;
for $i = 1$ *to* n **do**
    set index $j = ((i-1) \bmod d) + 1$;
    sample component $z \sim \pi(\cdot \mid \boldsymbol{x}_{-j}^{(i-1)})$;
    set $\boldsymbol{x}^{(i)} = \boldsymbol{x}^{(i-1)}$ and $x_j^{(i)} = z$;
end

Gibbs sampler

- The Gibbs sampler can fail to be irreducible when the space is disconnected.
- For the Gibbs sampler to work properly, it must be possible to reach any point from any other, using only moves parallel to the coordinate axes.
- Main difficulties:
  - (i) If the variables are strongly correlated (negatively or positively) then it may take too long to reach the stationary distribution. In this case, the problem would benefit from a reparametrization in terms of less correlated random variables.
  - (ii) Need to be able to show that the Gibbs sampler Markov chain is ergodic (irreducible and aperiodic). Obvious in many circumstances but sometimes an issue.

Gibbs sampler (example)

- Draw samples from the following (mixed) joint distribution of with $x = \{0, 1, ..., m\}$ (discrete) and $0 \leq y \leq 1$ (continuous),

$$\pi_{XY}(x, y) \propto \binom{m}{x} y^{x+\alpha-1} (1-y)^{m-x+\beta-1}; \tag{31}$$

  where, $m = 16, \alpha = 2, \beta = 4$.

- The conditional densities are:

$$\pi(x \mid y) = \text{Binomial}(m, y) \quad \text{and} \quad \pi(y \mid x) = \text{beta}(\alpha + x, n - \beta + x). \tag{32}$$

- The power of the Gibbs sampler is that by computing a sequence of these univariate conditional random variables, we can compute any feature of either marginal distribution.

Gibbs sampler (example)

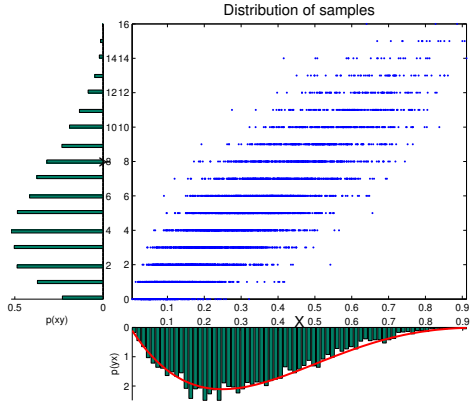


Figure: Samples from the mixed distribution.

Metropolis-within-Gibbs sampler

- Sometimes we can sample most, but not all of the full conditional distributions needed for the Gibbs sampler. For example, suppose that we can sample from $\pi_{j \mid -j}$ for every j from 1 to d, except for one value $j^\star$.
- The hybrid Gibbs or Metropolis-within-Gibbs algorithm uses a MH update in place of the missing $j^\star$th full conditional distribution.
- The sampling schemes inside the Gibbs structure only require the generation of one sample for every Gibbs iteration. This is because the Markov chains associated to the individual conditionals are not required to reach stationarity at each Gibbs iteration. The reasons for this are discussed in [15, p.393].
- Nevertheless, some authors suggest that performing a few extra within-Gibbs iterations can be beneficial to accelerate the convergence of the Gibbs chain and to reduce correlation in the samples.

6. MCMC – preconditioned Crank-Nicolson

Preconditioned Crank–Nicolson (I)

- The performance of the RWM algorithms deteriorates with increasing dimension of the parameter space. The preconditioned Crank–Nicolson (pCN) method avoids this issue by designing proposal that perform-well in function spaces.

- Given a target posterior, the *preconditioned overdamped Langevin* equation is,

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = -\boldsymbol{P}\nabla\Psi(\boldsymbol{x};\boldsymbol{y}) + \sqrt{2\boldsymbol{P}}\,\frac{\mathrm{d}W_t}{\mathrm{d}t}, \tag{33}$$

where W_t is a standard Brownian motion process, $\boldsymbol{P}$ is a symmetric and positive semi-definite preconditioner, and the (unnormalized) logarithm of the posterior density Ψ and its gradient can be written as:

$$\Psi(\boldsymbol{x};\boldsymbol{y}) = \frac{1}{2}\left\|\boldsymbol{\Sigma}_{\text{pr}}^{-1/2}\boldsymbol{x}\right\|_2^2 + \Phi(\boldsymbol{x};\boldsymbol{y}), \qquad \nabla\Psi(\boldsymbol{x};\boldsymbol{y}) = \boldsymbol{\Sigma}_{\text{pr}}^{-1}\boldsymbol{x} + \nabla\Phi(\boldsymbol{x};\boldsymbol{y}). \tag{34}$$

Preconditioned Crank–Nicolson (II)

- We can now substitute the gradient in eq. (34) into eq. (33) and solve the resulting stochastic differential equation using a Crank–Nicolson scheme with discretization step Δ [14].
- It can be shown that by making the preconditioner equal to the prior covariance, the following proposal mechanism arises from the Crank–Nicolson discretization [4]:

$$\boldsymbol{x}^\star = \frac{(2-\Delta)}{(2+\Delta)}\boldsymbol{x} + \frac{\sqrt{8\Delta}}{(2+\Delta)}\boldsymbol{\xi} \quad \longrightarrow \quad \boldsymbol{x}^\star = \sqrt{1-\beta^2}\boldsymbol{x} + \beta\boldsymbol{\xi}, \tag{35}$$

  where $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\text{pr}})$ and $\beta = \sqrt{8\Delta}/\,(2+\Delta)$.
- A common choice for the scaling parameter is $\beta \in (0, 1]$ for discretization steps $\Delta \in (0, 2]$ (in general $\beta \to 0$ as $\Delta \to \infty$).

Preconditioned Crank–Nicolson (III)

- Given a current state $\boldsymbol{x}^{(i)}$, one has from eq. (35) that the associated proposal distribution is Gaussian with mean vector $\sqrt{1-\beta^2}\boldsymbol{x}^{(i)}$ and covariance matrix $\beta^2\boldsymbol{\Sigma}_{\text{pr}}$.

- The acceptance probability of the pCN algorithm only requires evaluation of the potential function [5]

$$\alpha(\boldsymbol{x}, \boldsymbol{x}^\star) = \min\left(1, \exp\left(\Phi(\boldsymbol{x}; \boldsymbol{y}) - \Phi(\boldsymbol{x}^\star; \boldsymbol{y})\right)\right). \tag{36}$$

- pCN is applicable when the posterior distribution has density with respect to a Gaussian reference measure (prior). However, it can also be extended to more general priors by applying a transformation.

7. MCMC – using gradients

Metropolis-adjusted Langevin algorithm

- In the Metropolis-adjusted Langevin algorithm (**MALA**), new states are proposed using (overdamped) Langevin dynamics, and these are accepted or rejected using the MH algorithm[8].
- Let π denote a probability density function on $\mathbb{R}^d$, one from which it is desired to draw an ensemble of iid samples. We consider the overdamped Langevin Itô diffusion

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \frac{1}{2}\nabla \log \pi(\boldsymbol{x}) + \frac{\mathrm{d}W_t}{\mathrm{d}t}, \tag{37}$$

  driven by the time derivative of a standard Brownian motion W.

- In the limit, as $t \to \infty$, the probability distribution $X(t)$ approaches a stationary distribution, which is also invariant under the diffusion. It turns out that this distribution is π.

[8] If the acceptance/rejection is not applied the resulting approximate method is called **ULA** (unadjusted Langevin algorithm).

Metropolis-adjusted Langevin algorithm

- Approximate sample paths of the Langevin diffusion can be generated by the Euler–Maruyama method with a fixed time step $\varepsilon > 0$. We set $\boldsymbol{x}_0$ and then recursively define an approximation to the true solution by

$$\boldsymbol{x}_{k+1} := \boldsymbol{x}_k + \frac{\varepsilon^2}{2} \nabla \log \pi(\boldsymbol{x}_k) + \varepsilon \, \boldsymbol{\xi}_k, \tag{38}$$

  where each $\boldsymbol{\xi}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$.

- In contrast to the Euler–Maruyama method for simulating the Langevin diffusion. MALA incorporates an additional step. We consider the above update rule as defining a proposal $\tilde{\boldsymbol{x}}$ for a new state:

$$\tilde{\boldsymbol{x}}_{k+1} := \boldsymbol{x}_k + \frac{\varepsilon^2}{2} \nabla \log \pi(\boldsymbol{x}_k) + \varepsilon \, \boldsymbol{\xi}_k; \tag{39}$$

  this proposal is accepted or rejected according to the MH algorithm.

Metropolis-adjusted Langevin algorithm

- That is, the acceptance probability is:

$$\alpha(\boldsymbol{x}_k, \tilde{\boldsymbol{x}}_{k+1}) = \min\left(1, \frac{\pi(\tilde{\boldsymbol{x}}_{k+1}) q(\boldsymbol{x}_k \mid \tilde{\boldsymbol{x}}_{k+1})}{\pi(\boldsymbol{x}_k) q(\tilde{\boldsymbol{x}}_{k+1} \mid \boldsymbol{x}_k)}\right); \tag{40}$$

  where the proposal has the form

$$q(\boldsymbol{x}' \mid \boldsymbol{x}) = \mathcal{N}\left(\boldsymbol{x}'; \boldsymbol{x} + \frac{\varepsilon^2}{2} \nabla \log \pi(\boldsymbol{x}), \varepsilon^2 \mathbf{I}_d\right). \tag{41}$$

- The combined dynamics of the Langevin diffusion and the MH algorithm satisfy the detailed balance conditions necessary for the existence of a unique, invariant, stationary distribution.

- For limited classes of target distributions, the optimal acceptance rate for this algorithm can be shown to be 0.574; this can be used to tune ε.

Hamiltonian MC (intro)

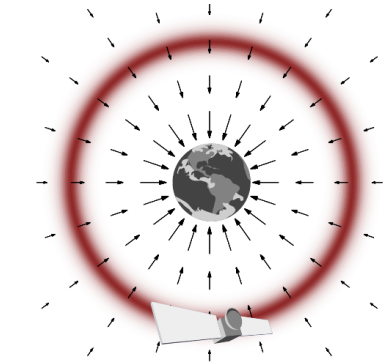


Figure: The mode of the target as a planet and the gradient of the target as its gravitational field. Taken from [3].

- So far we discussed classic MCMC approaches to draw samples from a target distribution $\pi(\boldsymbol{x})$.
- Hamiltonian Monte Carlo (HMC): use Hamiltonian dynamics to simulate particle trajectories ([6]).
- Define a Hamiltonian function in terms of the target distribution.
- Introduce an auxiliary *momentum* variables, which typically have independent Gaussian distributions.

Hamiltonian MC (intro)

- Hamiltonian dynamics operate on a d-dimensional *position* vector $\boldsymbol{q}$, and a d-dimensional *momentum* vector $\boldsymbol{p}$, so that the full state space has $2d$ dimensions. The system is described by a function of $\boldsymbol{q}$ and $\boldsymbol{p}$ known as the *Hamiltonian* $H(\boldsymbol{q}, \boldsymbol{p})$.

- In HMC, one uses Hamiltonian functions that can be written as (closed-system dynamics):

$$H(\boldsymbol{q}, \boldsymbol{p}) = \underbrace{U(\boldsymbol{q})}_{\text{potential energy}} + \underbrace{K(\boldsymbol{p}, \boldsymbol{q})}_{\text{kinetic energy}} . \tag{42}$$

- The potential energy is completely determined by the target distribution, indeed $U(\boldsymbol{q})$ is equal to the logarithm of the target distribution π.

- The kinetic energy is unconstrained and must be specified by the implementation.

Hamiltonian MC

- The Hamiltonian is an energy function for the joint state of 'position-momentum', and so defines a joint distribution for them as follows:

$$\pi(\boldsymbol{q}, \boldsymbol{p}) = \frac{1}{Z} \exp\left(-\frac{H(\boldsymbol{q}, \boldsymbol{p})}{T}\right) = \frac{1}{Z} \exp\left(-U(\boldsymbol{q})\right) \exp\left(-K(\boldsymbol{p})\right). \tag{43}$$

- There are several ways to set the kinetic energy (density of the auxiliary momentum) [3]:
  - (i) Euclidean–Gaussian kinetic energy: using a fixed covariance $\mathbf{M}$ estimated from the position parameters, $K(\boldsymbol{p}, \boldsymbol{q}) = \frac{1}{2}\boldsymbol{p}^{\mathsf{T}}\mathbf{M}^{-1}\boldsymbol{p} + \ln(|\mathbf{M}|) + \text{const.}$
  - (ii) Riemann–Gaussian kinetic energy: unlike the Euclidean metric, varies as one moves through parameter space, $K(\boldsymbol{p}, \boldsymbol{q}) = \frac{1}{2}\boldsymbol{p}^{\mathsf{T}}\boldsymbol{\Sigma}(\boldsymbol{q})^{-1}\boldsymbol{p} + \frac{1}{2}\ln(|\boldsymbol{\Sigma}(\boldsymbol{q})|) + \text{const.}$
  - (iii) Non-Gaussian kinetic energies.

Hamiltonian MC

- Hamilton's equations read as follows:

$$\frac{\mathrm{d}\boldsymbol{q}}{\mathrm{d}t} = +\frac{\partial H}{\partial \boldsymbol{p}} = [\mathbf{M}^{-1}\boldsymbol{p}] \tag{44a}$$

$$\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = -\frac{\partial H}{\partial \boldsymbol{q}} = -\frac{\partial K}{\partial \boldsymbol{q}} - \frac{\partial U}{\partial \boldsymbol{q}}, \tag{44b}$$

  where $\frac{\partial U}{\partial \boldsymbol{q}}$ is the gradient of the logarithm of the target density.

- Discretizing Hamilton's equations:
  - (i) Euler's method (**no**).
  - (ii) Modified Euler's method (**a bit better**).
  - (iii) Symplectic integrators: the leapfrog method (**the standard choice**).

Hamiltonian MC (remarks)

Properties

- Hamiltonian dynamics are time-reversible and volume-preserving.
- The dynamics keep the Hamiltonian invariant. A Hamiltonian trajectory will (if simulated exactly) move within a hypersurface of constant probability density.

Each iteration of the HMC algorithm has two steps. Both steps leave the joint distribution of $\pi(q, p)$ invariant (detailed balance) [11].

- In the first step, new values of $\boldsymbol{p}$ are randomly drawn from their Gaussian distribution, independently of the current $\boldsymbol{q}$.
- In the second step, a Metropolis update is performed, using Hamiltonian dynamics to propose a new state.
- Optimal acceptance rate is 0.65. The step size ε and trajectory length L need to be tuned.

Hamiltonian MC (example)

- More typical behavior of HMC and RWM is illustrated by a 100-dimensional multivariate Gaussian distribution in which the variables are independent, with means of zero, and standard deviations of $0.01, 0.02, ..., 0.99, 1$.
- Suppose that we have no knowledge of the details of this distribution, so we will use HMC with the same simple, rotationally symmetric kinetic energy function.
- Consistent with this, we use HMC to this distribution using trajectories with $L = 150$ and with ε randomly selected for each iteration, uniformly from $(0.0104, 0.0156)$; here $n = 10^3$. We compare with a RWM with proposal standard deviation drawn uniformly from $(0.0176, 0.0264)$; with a lag period of 150.
- These are close to optimal settings for both methods. The rejection rate was 0.13 for HMC and 0.75 for RWM.

Hamiltonian MC (example)

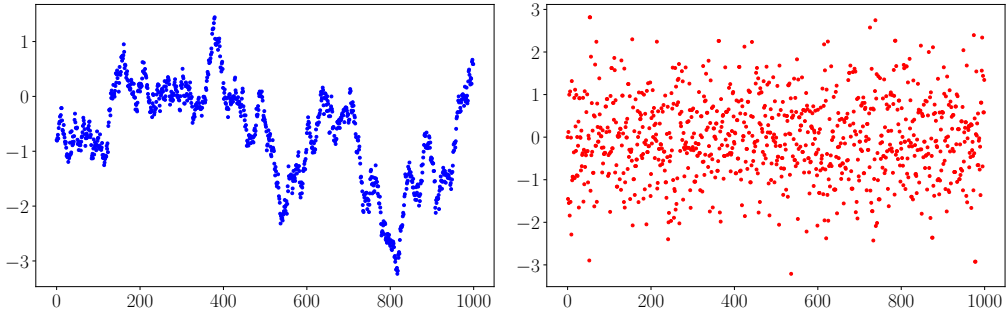


Figure: Values for the variable with largest standard deviation for the 100-dimensional example, from a RWM run and an HMC run.

Hamiltonian MC (example)

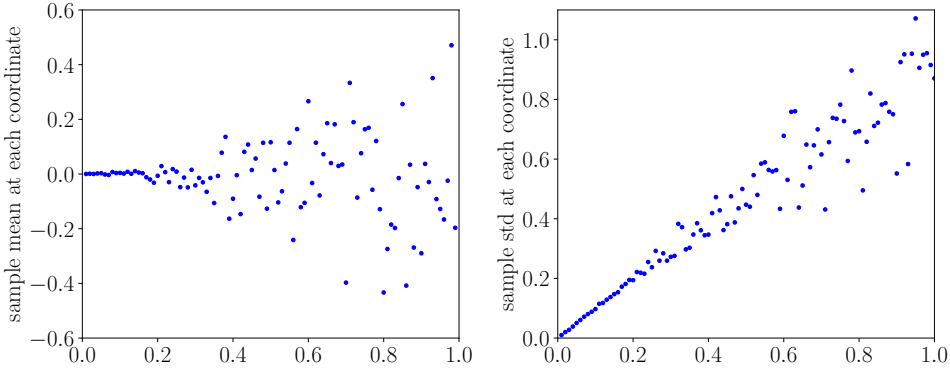


Figure: Estimates of mean and standard deviations for the 100-dimensional example, using RWM.

Hamiltonian MC (example)

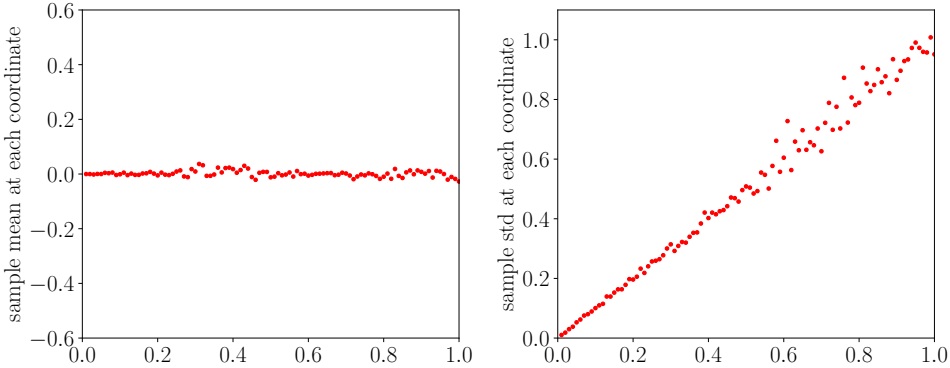


Figure: Estimates of mean and standard deviations for the 100-dimensional example, using HMC.

The no-U-turn sampler (NUTS)

- HMC is an algorithm that avoids the random walk behavior and sensitivity to correlated parameters that plague many MCMC methods by taking a series of steps informed by first-order gradient information.
- However, HMC's performance is highly sensitive to two user-specified parameters: a step size ε and a desired number of steps L.
- The No-U-Turn Sampler (NUTS), an extension to HMC that eliminates the need to set a number of steps L, as well as the step-size.
- We simulate in discrete time steps, and to make sure you explore the parameter space properly you simulate steps in one direction and the twice as many in the other direction, turn around again, etc. At some point you want to stop this and a good way of doing that is when you have done a U-turn (i.e., appear to have gone all over the place).

The no-U-turn sampler (NUTS)

- NUTS begins by introducing an auxiliary variable with conditional distribution. After re-sampling from this distribution, NUTS uses the leapfrog integrator to trace out a path forwards and backwards in fictitious time. First running forwards or backwards 1 step, then forwards or backwards 2 steps, then forwards or backwards 4 steps, etc.
- This doubling process implicitly builds a balanced binary tree whose leaf nodes correspond to position-momentum state. The doubling is stopped when the subtrajectory from the leftmost to the rightmost nodes of any balanced subtree of the overall binary tree starts to double back on itself (i.e., the fictional particle starts to make a "U-turn").
- At this point NUTS stops the simulation and samples from among the set of points computed during the simulation, taking care to preserve detailed balance.

The no-U-turn sampler (NUTS)

- To adapt the step-size, NUTS uses a modified dual averaging algorithm during the burn-in phase.
- The good thing about NUTS is that proposals are made based on the shape of the posterior and they can happend at the other end of the distribution. In contrast, MH makes proposals within a ball, and Gibbs sampling only moves along one (or at least very few) dimensions at a time.

8. Approximation methods

The Laplace approximation (I)

- As an alternative to simulation of integrals, we can also attempt analytic approximations.
- One of the oldest and most useful approximations is the integral Laplace approximation. It is based on the following argument: Suppose that we are interested in evaluating the integral

$$\int_A f(x \mid y)\mathrm{d}x, \tag{45}$$

for a fixed y, and f non-negative and integrable.

- Write

$$f(x \mid y) = \exp(nh(x \mid y)),$$

where n is a parameter that can go to infinity.

The Laplace approximation (II)

- Use a Taylor series expansion of $h(x \mid y)$ about a point x_0 to obtain

$$h(x \mid y) \approx h(x_0 \mid y) + (x - x_0)h'(x_0 \mid y) + \frac{(x - x_0)^2}{2}h''(x_0 \mid y) + \frac{(x - x_0)^3}{3!}h'''(x_0 \mid y) + R_n(x)$$

  while the remainder $R_n(x)$ satisfies $\lim_{x \to x_0} R_n(x)/(x - x_0)^3 = 0$.

- Now choose $x_0 = x^\star$, the value that satisfies $h'(x^\star \mid y) = 0$ and maximizes $h(x \mid y)$ for a given value of y. Then, the linear term in the Taylor series is zero and we have the approximation

$$\int_A \exp(nh(x \mid y))\mathrm{d}x = \exp(nh(x^\star \mid y)) \int_A \exp\left(n\left(\frac{(x - x^\star)^2}{2}\right)h''(x^\star|y)\right) \tag{46a}$$

$$\exp\left(n\left(\frac{(x - x^\star)^3}{3!}\right)h'''(x^\star|y)\right)\mathrm{d}x \tag{46b}$$

  which is valid within a neighborhood of $x^\star$.

The Laplace approximation (III)

- The cubic term in the exponent is now expanded in a series around $x^\star$ and using the Taylor expansion of the exponential function, we obtain:

$$1 + n\left(\frac{(x-x^\star)^3}{3!}\right)h'''(x^\star|y) + n^2\left(\frac{(x-x^\star)^6}{2!(3!)^2}\right)[h'''(x^\star|y)]^2 \tag{47}$$

and

$$\int_A \exp(nh(x \mid y))\mathrm{d}x = \exp(nh(x^\star \mid y))\int_A \exp\left(n\left(\frac{(x-x^\star)^2}{2}\right)h''(x^\star|y)\right) \tag{48a}$$

$$\left[1 + n\left(\frac{(x-x^\star)^3}{3!}\right)h'''(x^\star|y) + n^2\left(\frac{(x-x^\star)^6}{2!(3!)^2}\right)[h'''(x^\star|y)]^2 + R_n(x)\right]\mathrm{d}x \tag{48b}$$

- Excluding R_n, we call the integral approximations of a *first-order*, if it includes only the first term in the right-hand side; of a *second-order*, if it includes the first two terms; and of a *third-order*, if it includes all three terms.

The Laplace approximation (IV)

- We can evaluate these expressions further since the above integrand is the kernel of a Gaussian density with mean $x^\star$ and variance $1/(nh''(x^\star|y))$. This Gaussian approximation is the so-called Laplace approximation.

- More precisely, letting Φ denote the standard Gaussian CDF, and taking $A = [a, b]$, we can evaluate the integral in the **first-order approximation** to obtain (with $n = 1$)

$$\int_a^b \exp(h(x \mid y))\mathrm{d}x = \exp(h(x^\star \mid y))\sqrt{\frac{2\pi}{-h''(x^\star \mid y)}} \tag{49a}$$

$$\left[\Phi\left(\sqrt{-h''(x^\star \mid y)}(b - x^\star)\right) - \Phi\left(\sqrt{-h''(x^\star \mid y)}(a - x^\star)\right)\right]. \tag{49b}$$

The Laplace approximation (V)

- The Laplace approximation is reasonable in the central region of the density, it becomes quite unacceptable in the tails.
- In problems where Monte Carlo calculations are prohibitive because of computing time, the Laplace approximation can be useful as a guide to the solution of the problem [15].
- Also, the corresponding Taylor series can be used as a proposal density (for MCMC), which is particularly useful in problems where no obvious proposal exists.

The saddlepoint approximation (I)

- The saddlepoint approximation, in contrast to the Laplace approximation, is mainly a technique for approximating a function rather than an integral, although it naturally leads to an integral approximation (Proposed by Henry E. Daniels in 1954).
- Suppose that we are interested in evaluating the integral

$$g(y) = \int_A \pi(x \mid y) \mathrm{d}x, \tag{50}$$

  for a range of values of y.
- One interpretation of a saddlepoint approximation is that for each value of y, we do a Laplace approximation centered at $x^\star$ (the saddle point[9]).
- One way to derive the saddlepoint approximation is to use an Edgeworth expansion (aka Gram–Charlier A series).

[9] also minimax point; it is a point on the surface of a function where the slopes (derivatives) in orthogonal directions are all zero, but which is not a local extremum of the function.

The saddlepoint approximation (II)

- As a result of a quite detailed derivation, we obtain the approximation to the density of X to be

$$\pi_X(x) = \frac{\sqrt{n}}{\sigma}\varphi\left(\frac{x-\mu}{\sigma/\sqrt{n}}\right)\left\{1 + \frac{\kappa}{6\sqrt{n}}\left(\left(\frac{x-\mu}{\sigma/\sqrt{n}}\right)^3 - 3\left(\frac{x-\mu}{\sigma/\sqrt{n}}\right)\right) + \mathcal{O}(1/n)\right\}. \quad (51)$$

- Ignoring the term within braces produces the usual Gaussian approximation, which is accurate to $\mathcal{O}(1/\sqrt{n})$.

- If we are using eq. (51) for values of x near μ, then the value of the expression in braces is close to zero, and the approximation will then be accurate to $\mathcal{O}(1/n)$. The trick of the saddlepoint approximation is to make this always be the case,

- To do so, we use a family of densities such that, for each x, we can choose a density from the family to cancel the term in braces in eq. (51).

The saddlepoint approximation (III)

- One method of creating such a family is through a technique known as exponential tilting (aka or Exponential Change of Measure). The result of the exponential tilt is a family of Edgeworth expansions for $\pi_X(x)$ indexed by a parameter τ, that is

$$\pi_X(x) = \exp\left(-n[\tau x - K(\tau)]\right) \frac{\sqrt{n}}{\sigma_\tau} \varphi\left(\frac{x-\mu_\tau}{\sigma_\tau/\sqrt{n}}\right) \tag{52a}$$

$$\left\{1 + \frac{\kappa_\tau}{6\sqrt{n}}\left(\left(\frac{x-\mu_\tau}{\sigma_\tau/\sqrt{n}}\right)^3 - 3\left(\frac{x-\mu_\tau}{\sigma_\tau/\sqrt{n}}\right)\right) + \mathcal{O}(1/n)\right\}. \tag{52b}$$

- As the parameter τ free to choose, for each x we choose $\tau = \tau(x)$ so that the mean satisfies $\mu_\tau = x$. This choice cancels the middle term in the square brackets), thereby improving the order of the approximation.

The saddlepoint approximation (IV)

- If $K(\tau) = \log(\mathbb{E}[\exp \tau X])$ is the cumulant generating function, we can choose τ so that $K'(\tau) = x$, which is the saddlepoint equation.
- Denoting $\sigma_\tau = K''(\tau)$, and $\hat{\tau}$ the value obtained from the saddlepoint equation, we get the saddlepoint approximation (with $n = 1$):

$$\pi_X(x) = \exp\left(K(\hat{\tau}) - \hat{\tau}x\right) \frac{1}{\sigma_{\hat{\tau}}} \varphi(0) + (1 + \mathcal{O}(1)) \tag{53a}$$

$$\approx \frac{1}{\sqrt{2\pi\sigma_{\hat{\tau}}}} \exp\left(K(\hat{\tau}) - \hat{\tau}x\right). \tag{53b}$$

- The saddlepoint can also be used to approximate the tail area of a distribution [15].
- This better error rates are obtained by renormalizing the approximation so that it integrates to 1.

Final remarks

- Other algorithms we did not cover in class are:
  - (i) Auxiliary variable: slice sampler, simulated annealing, simulated tempering, Hit-and-run.
  - (ii) Sequential algorithms: sequential importance sampling, population Monte Carlo, etc.
  - (iii) Approximate methods: Laplace approximations, approximate Bayesian computation (likelihodod-free), variational Bayesian inference, transport maps, Stein variational gradient descent, etc.
- We covered the most common MCMC algorithms used in practice. The particular choice of a method will depend on your application and computational resources.
- Quite often we have to work with approximate methods. Sampling is a computationally intensive task, which complicates application of UQ in general inverse problems.
- UQ keeps growing by the day; new samplers and techniques are addressing complicated inference tasks.
- Check this cool demo ! Link Chi Feng's MCMC demo.

References

[1] C. Andrieu et al. "A tutorial on adaptive MCMC". In: *Statistics and Computing* 18 (2008), 343—373.

[2] A. A. Barker. "Monte Carlo calculations of the radial distribution functions for a proton-electron plasma". In: *Australian Journal of Physics* 18 (1965), pp. 119–1347.

[3] M. Betancourt. "A conceptual introduction to Hamiltonian Monte Carlo". In: `arXiv:1701.02434v2` (2017), pp. 1–60.

[4] S. L. Cotter et al. "MCMC methods for functions: modifying old algorithms to make them faster". In: *Statistical Science* 28.3 (2013), pp. 424–446.

[5] M. Dashti et al. "The Bayesian approach to inverse problems". In: *Handbook of uncertainty quantification*. Ed. by R. Ghanem et al. Springer International Publishing, 2017. Chap. 10, pp. 311–428.

[6] S. Duane et al. "Hybrid Monte Carlo". In: *Physics Letters B* 195.2 (1987), pp. 216–222.

[7] A. Gelman et al. "Efficient Metropolis jumping rules". In: *Bayesian Analysis* 5 (1996), 599–607.

[8] H. Haario et al. "An Adaptive Metropolis Algorithm". In: *Bernoulli* 7.2 (2001), pp. 223–242.

[9] W. K. Hastings. "Monte Carlo sampling methods using Markov chains and their applications". In: *Biometrika* 57.1 (1970), pp. 97–109.

[10] N. Metropolis et al. "Equation of state calculations by fast computing machines". In: *Chemical Physics* 21.6 (1953), pp. 1087–1092.

[11] R. M. Neal. "MCMC using Hamiltonian dynamics". In: *Handbook of Markov chain Monte Carlo*. Ed. by S. Brooks et al. Chapman & Hall/CRC, 2011. Chap. 5, pp. 113–162.

[12] J. R. Norris. *Markov Chains*. Cambridge University Press, 1998.

[13] A. B. Owen. *Monte Carlo theory, methods and examples*. artowen.su.domains/mc/, 2018.

[14] W. H. Press et al. *Numerical recipes in C: the art of scientific computing*. 3rd ed. Cambridge University Press, 2007.

[15] C. P. Robert et al. *Monte Carlo statistical methods*. 2nd ed. Springer, 2004.

[16] L. Tierney. "A note on Metropolis-Hastings kernels for general state spaces". In: *Annals of applied probability* (1998), pp. 1–9.

[17] B. Walsh. *Markov chain Monte Carlo and Gibbs sampling*. Lecture notes for EEB 581, version 26, 2004.