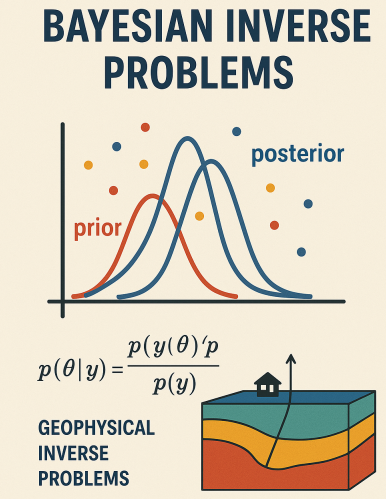
Posterior Estimation Practical: MC, McMC, HMC, V.I. Solutions

Mark Asch - MAKUTU/2025



MONTE CARLO

Ex. 1: Monte Carlo Integration

Estimate π

This is a well-known, introductory exercise in Monte Carlo integration. Compute the area under the first quadrant of the unit circle, where the function to integrate is

$$f(x,y) = \begin{cases} 1 & \text{for } x^2 + y^2 \leq 1, \\ 0 & \text{otherwise,} \end{cases}$$

and the integral

$$I = \int_0^1 \int_0^1 f(x,y)\,\mathrm{d}x\mathrm{d}y.$$

Then $I \approx \pi/4$ and $4I \approx \pi$.

1. Write a python code to estimate π.

2. Study the accuracy of the approximation as a function of n, the number of samples and deduce the rate of convergence.

3. [Optional] Plot the sample points, red for those under the curve, blue for those above.

SOLUTION:

- See: `Ex.3.1-MC-pi.ipynb` and `Ex.3.1-MC-pi-plot.ipynb`

Ex. 2: Monte Carlo Integration

Rejection Sampling

Suppose that we want to sample from a multi-modal Gaussian

$$p(x) = \alpha_1 \mathcal{N}(\mu_1, \sigma_1^2) + \alpha_2 \mathcal{N}(\mu_2, \sigma_2^2),$$

but we have no way to accurately sample it because of the two modes.

1. A suitable proposal distribution $q(x)$ would be a simple, unimodal Gaussian, centered between the two modes and with a variance large enough to cover the range of p. Propose and plot a suitable $q(x)$ and superpose it on $p(x)$.

2. By trail and error, find a factor k that is large enough to provide an envelope, i.e. such that $kq(x) \geq p(x)$. Plot again the superposition, but this time with $kq(x)$ and $p(x)$.

3. Perform rejection sampling and plot the accepted sampling points (dots and histograms) for $n = 10^3$ and $n = 10^4$.

SOLUTION:

- See: `Ex.3.2-MC-reject.ipynb`

Ex. 3: Monte Carlo Integration

Importance Sampling

Suppose that we want to estimate a tail probability, or extreme event probability, of a standard Gaussian, $\mathcal{N}(0,1)$, for $\mathrm{P}(X > 5)$.

- Ordinary MC integration will not be very efficient since almost all the samples will be rejected.

- A better option would be to use an exponential density, truncated at $x = 5$, as the importance function for Importance Sampling

1. Show that ordinary Monte Carlo would result in approximately 3 samples out of 10,000,000 from $\mathcal{N}(0,1)$. to have a value greater than 5. Conclusion?

2. We can use the exponential density truncated at 5 as the importance (proposal) function and use importance sampling to estimate the probability.

(a) Use standard MC to estimate the probability.
(b) Plot the truncated exponential density and the standard Gaussian around the value $x = 5$.
(c) Use importance sampling, with the same number of sample points, and compare the result with the theoretical value. Conclusions?

SOLUTION:

- See: `Ex.3.2-MC-importance.ipynb`

Ex. 4: MC - Rejection Sampling

Prove that rejection sampling draws samples from the desired distribution $p(\mathbf{z})$.

1. Suppose the proposal distribution is $q(\mathbf{z})$. Show that the probability of a sample value $\mathbf{z}$ being accepted is given by $\tilde{p}(\mathbf{z})/kq(\mathbf{z})$ where $\tilde{p}$ is any unnormalized distribution that is proportional to $p(\mathbf{z})$, and the constant k is set to the smallest value that ensures $kq(\mathbf{z}) \geq \tilde{p}(\mathbf{z})$ for all values of $\mathbf{z}$.

2. Note that the probability of drawing a value $\mathbf{z}$ is given by the probability of drawing that value from $q(\mathbf{z})$ times the probability of accepting that value given that it has been drawn. Make use of this, along with the sum and product rules of probability, to write down the normalized form for the distribution over $\mathbf{z}$, and show that it equals $p(\mathbf{z})$.

SOLUTION:

- The probability of acceptance follows trivially from the mechanism used to accept or reject the sample.

- The probability of a sample u drawn uniformly from the interval $[0, kq(z)]$ being less than or equal to a value $\tilde{p}(\mathbf{z}) \leq kq(\mathbf{z})$ is simply

$$p(\text{acceptance}|\mathbf{z}) = \int_0^{\tilde{p}(\mathbf{z})} \frac{1}{kq(\mathbf{z})} \, \mathrm{d}u = \frac{\tilde{p}(\mathbf{z})}{kq(\mathbf{z})}.$$

- Therefore, the probability density for drawing a sample $\mathbf{z}$ is

$$q(\mathbf{z})p(\text{acceptance}|\mathbf{z}) = q(\mathbf{z})\frac{\tilde{p}(\mathbf{z})}{kq(\mathbf{z})} = \frac{\tilde{p}(\mathbf{z})}{k}. \quad (1)$$

- Since $\tilde{p}(\mathbf{z})$ is proportional to $p(\mathbf{z})$,

$$p(\mathbf{z}) = \frac{1}{Z_{\tilde{p}}}\tilde{p}(\mathbf{z}),$$

where the constant

$$Z_{\tilde{p}} = \int \tilde{p}(\mathbf{z}) \, \mathrm{d}\mathbf{z}.$$

- Since the LHS of (1) is a probability density that integrates to 1, it follows that

$$\int \frac{\tilde{p}(\mathbf{z})}{k} \, \mathrm{d}\mathbf{z} = 1,$$

so we must have $k = Z_{\tilde{p}}$ and hence

$$\frac{\tilde{p}(\mathbf{z})}{k} = p(\mathbf{z}),$$

as required.

MARKOV CHAIN

Ex. 5a: Markov Chains

Discrete

- Finite state-space (time homogenous) Markov chain

  $\Rightarrow$ If the state space of a Markov chain takes on a finite number of distinct values, and it is time homogenous, then the transition operator can be defined by a matrix P, where the entries of P are

$$p_{ij} = p(X^{(t+1)} = j | x^{(t)} = i)$$

  $\Rightarrow$ This means that if the chain is currently in the i-th state, the transition operator assigns the probability of moving to the j-th state by the entries of i-th row of P (i.e. each row of P defines a conditional probability distribution on the state space).

- Consider the problem of predicting the weather in Berkeley, CA. Suppose that

⇒ there are only 3 weather conditions: sunny (s), foggy (f), rainy (r), i.e. a state space that takes on 3 discrete values;

⇒ weather patterns are very stable there, so a Berkeley meteorologist (based on meteorological archives) can easily predict the weather next week based on the weather today with the following transition rules:

→ if it is sunny today, then it is highly likely that it will be sunny next week

$$p(X^{(t+1)} = \mathrm{s} | X^{(t)} = \mathrm{s}) = 0.8,$$

it is very unlikely that it will be raining next week

$$p(X^{(t+1)} = \mathrm{r} | X^{(t)} = \mathrm{s}) = 0.05,$$

and somewhat likely that it will foggy next week

$$p(X^{(t+1)} = \mathrm{f} | X^{(t)} = \mathrm{s}) = 0.15$$

→ if it is foggy today, then it is somewhat likely that it will be sunny next week

$$p(X^{(t+1)} = \mathrm{s} | X^{(t)} = \mathrm{f}) = 0.4,$$

it is slightly more likely that it will be foggy next week

$$p(X^{(t+1)} = \mathrm{f} | X^{(t)} = \mathrm{f}) = 0.5,$$

and fairly unlikely that it will rainy next week

$$p(X^{(t+1)} = \mathrm{r} | X^{(t)} = \mathrm{f}) = 0.1$$

→ if it is rainy today, then it is unlikely that it will be sunny next week

$$p(X^{(t+1)} = \mathrm{s} | X^{(t)} = \mathrm{r}) = 0.1,$$

it is somewhat likely that it will be foggy next week

$$p(X^{(t+1)} = \mathrm{f} | X^{(t)} = \mathrm{r}) = 0.3,$$

and fairly likely that it will be rainy next week

$$p(X^{(t+1)} = \mathrm{r} | X^{(t)} = \mathrm{r}) = 0.6.$$

1. Write the 3×3 transition matrix P, where each row of P corresponds to the weather at iteration t (today), and each column corresponds to the weather at iteration $t + 1$ (next week.)

2. Simulate the evolution of the Markov chain, starting from an initial state of rain, for 6 months (25 week, say).

3. Print the weather probabilities at 1 week, 2 weeks, 3 months and 6 months.

4. At what point in time does the chain reach a steady equilibrium, and what are the 3 equilibrium probabilities?

5. Verify the convergence of the chain by computing the differences between the last week's probabilities and the second-last week's.

6. What would be a reasonable burn-in time?

7. Experiment with other initial states. Conclusions?

8. [Optional] Compute the theoretical steady state by an eigenvector analysis and print the errors.

SOLUTION:

$$P = \begin{bmatrix} 0.8 & 0.15 & 0.05 \\ 0.4 & 0.5 & 0.1 \\ 0.1 & 0.3 & 0.6 \end{bmatrix}$$

- See Ex3.5a-MC-discrete-weather (ipynb and pdf)

Ex. 5b: Markov Chains

Continuous

We can use the stationary distribution of a continuous state-space Markov chain in order to sample from a continuous probability distribution: we run a Markov chain for a sufficient amount of time so that it has reached its stationary distribution, then keep the states that the chain visits as samples from that stationary distribution.

In this exercise we define a continuous state-space Markov chain, by supposing:

- The transition operator is a Gaussian distribution with unit variance and a mean that is half the distance between zero and the previous state, $\mathcal{N}(0.5x, 1)$, and
- the distribution over initial conditions is a Gaussian distribution with zero mean and unit variance, $\mathcal{N}(0, 1)$.

To ensure that the chain has moved sufficiently far from the initial conditions and that we are sampling from the chain's stationary distribution, we will choose to throw away the first 50 burn-in states of the chain. We also run multiple chains simultaneously in order to sample the stationary distribution more densely. Here we will choose to run 5 chains simultaneously.

1. Write a code that simulates the 5 chains simultaneously for 1000 iterations.

2. Plot

   (a) a zoom of the first 100 iterations, and mark the burn-in cutoff,
   (b) the complete trace of the 5 chains,
   (c) the histogram of the retained states.

3. What is the stationary distribution? What are its estimated parameters?

4. [Optional] Identifying the chain as an AR(1) process, compute the theoretical variance and compare

it with the value obtained from the simulations of the Markov chain.

SOLUTION:

$$X \sim \mathcal{N}(0, 1.3)$$

- Each state follows an AR(1) process: $X_t = 0.5 \times X_{t-1} + \varepsilon_t$, with $\varepsilon_t \sim \mathcal{N}(0, 1)$. This creates a stationary process with known theoretical properties that can be compared against the simulation results.

- For stationary AR(1): $X_t = \varphi * X_{t-1} + \varepsilon_t$ where $\varepsilon_t \sim \mathcal{N}(0, \sigma^2)$, the theoretical variance $= \sigma^2/(1 - \varphi^2) = 1/(1 - 0.5^2) = 4/3 \approx 1.333$

- See `MC-continuous-Gaussian (ipynb and pdf)`

Ex. 6: McMC

Metropolis Hastings

[Optional] Go to `https://twiecki.io/blog/2015/11/10/mcmc-sampling/`, follow the explanations and (re)code this very detailed and well-explained MH version of McMC.

Ex. 7: McMC

Metropolis Hastings

This instructive exercise will be in two parts:

1. We will use the Metropolis-Hasting algorithm to compute the posterior distribution of the mean of a Gaussian distribution with known variance, from a sequence of given observations.

2. We will study the influence of the proposal distribution on the convergence of the Markov Chain generated in the first part.

In the first part, we will start with a proposal that is far from the reality, and the likelihood function will also be very approximate. In the second part, we will investigate more closely the behavior of the chain as the proposal distribution changes.

Recall Bayes' Law that gives an expression for the

posterior conditional probability of a parameter θ

$$p(\theta \mid y) = \frac{p(\theta)p(y \mid \theta)}{\int p(\theta')p(y \mid \theta') \, \mathrm{d}\theta'},$$

where y is the observation, $p(\theta)$ is the prior probability, $p(y \mid \theta)$ is the likelihood function and the denominator is a normalization factor representing the total probability of y. Very often, in practical applications, this denominator is intractable (impossible, or too expensive to compute), whereas the likelihood and prior are known. This is an ideal instance for MCMC, since we can simulate the posterior without having to know the normalizing factor.

Consider a simple problem, where we have a sequence of 5 measurements (or samples), $\{y_1, \ldots, y_5\} = \{9.37, 10.18, 9.16, 11.60, 10.33\}$, depending on a parameter θ (the unknown mean) for which we would like to obtain the (posterior) probability distribution, $p(\theta \mid y)$.

For the Metropolis-Hasting algorithm, use the

Metropolis acceptance ratio

$$r = \frac{p\left(\theta^* \mid y\right)}{p\left(\theta^{(t)} \mid y\right)} = \frac{p\left(y \mid \theta^*\right) p(\theta^*)}{p\left(y \mid \theta^{(t)}\right) p(\theta^{(t)})},$$

where θ^* is a candidate value for the chain drawn from the symmetric Gaussian proposal distribution $\mathcal{N}(0, \delta^2)$, the actual value in the chain is denoted $\theta^{(t)}$, the known prior is $\mathcal{N}(\mu, \tau^2)$, and the likelihood is computed assuming that $y_i \sim \mathcal{N}(\theta, \sigma^2)$. The exact expressions for the sample mean and variance can be computed, and are given by

$$\mu_n = \frac{(n/\sigma^2)\bar{y} + (1/\tau^2)\mu}{n/\sigma^2 + 1/\tau^2}$$

and

$$\tau_n^2 = \frac{1}{n/\sigma^2 + 1/\tau^2}.$$

1. Suppose that the proposal distribution has variance $\delta^2 = 2$, and that the prior has parameters $\mu = 5$,

$\tau^2 = 10$. Fix a chain length of 10^4 and an initial value $\theta^{(0)} = 5$. Assume that the likelihood variance is known, with $\sigma^2 = 1$.

(a) Code and simulate the MH algorithm for these values.

(b) Plot the trace (θ as a function of the iteration number, subsampling one in every ten values) and the histogram of the Markov chain (after removing a burn-in period of the first 50 values, say) and compare it with the theoretical law.

(c) Conclusions?

2. We now study how the choice of the parameter δ in the proposal distribution affects the mixing, and hence the convergence of the simulated Markov chain. Consider a sequence of values

$$\delta^2 \in \left\{ \frac{1}{32}, \frac{1}{2}, 2, 32, 64 \right\}.$$

(a) For diagnostic purposes compute the autocorrelation function and record the the lag-1 autocor-

relations. For which value of δ^2 do we have a minimum (optimum)?

(b) Plot and compare the convergence of the 5 Markov chains for the first 500 iterations, one graphic for each value of δ^2.

(c) Observations and conclusions?

SOLUTION:

- see `McMC-MH (ipynb and pdf)`
- see `McMC-MH-convg (ipynb and pdf)`

Ex. 8: McMC

Metropolis Hastings - 1D heat eq. inverse problem

- We want to estimate the conductivity (diffusivity) coefficient, D, in the 1D heat (diffusion) equation

$$\begin{cases} \dfrac{\partial u}{\partial t} - \dfrac{\partial}{\partial x}\left(D\dfrac{\partial u}{\partial x}\right) = 0, \quad x \in (0, L), \quad 0 < t \leq T, \\ u(x, 0) = f(x), \quad x \in (0, L), \\ u(0, t) = 0, \quad u(L, t) = 0, \quad t > 0, \end{cases} \tag{2}$$

  from noisy measurements (data)

$$y_i = u(x_i, T) + \epsilon_i$$

  at points x_i, $i = 1, \ldots, N$ and final time T, where the i.i.d. noise is taken as

$$\epsilon_i \sim \mathcal{N}(0, s^2).$$

SOLUTION:

- see DTbook, Ex. 11.21 (OneDHeatIP.m)

- python version: see Practicals/codes/Ex3.8-McMC-MH-1D-heat (ipynb and pdf)

Ex. 9: McMC

HMC - Harmonic Oscillator

The objective here is to demonstrates the leapfrog integration method for Hamiltonian dynamics, showing how the harmonic oscillator evolves in time, while conserving total energy in phase space. The code for the simulation of the dynamics can be utilized, as is, for Hamiltonian Monte Carlo—see below.

- The obtained results should clearly demonstrate the physics:

  $\Rightarrow$ the harmonic oscillator exhibits periodic motion,
  $\Rightarrow$ energy oscillates between kinetic and potential forms while total energy remains constant,
  $\Rightarrow$ and the phase space trajectory traces out an ellipse characteristic of conservative Hamiltonian systems.

- Suppose a ball of mass $m = 1$ is attached to a horizontal spring (no gravity). Then the spring

exerts a force on the ball

$$F = -kx,$$

where k is the spring constant, which we suppose equal to one.

- The potential energy is then

$$U(x) = \int F dx = \int -x dx = \frac{x^2}{2}$$

and the kinetic energy

$$K(v) = \frac{1}{2}mv^2 = \frac{v^2}{2} = \frac{p^2}{2} = K(p).$$

Their partial derivatives are

$$\frac{\partial U(x)}{\partial x} = x, \quad \frac{\partial K(p)}{\partial p} = p.$$

1. Write down the Hamiltonian and the 3 steps of the leapfrog method for the Hamiltonian system.

2. Simulate the dynamics, with $\delta = 0.1$ for 70 time steps.

3. Plot:

   (a) position as a function of time,
   (b) U, K and H as a function of time,
   (c) the phase plot, in the $x - p$ plane.

4. [Optional] Generate an animation of the energy conservation as a function of time.

SOLUTION:

- Hamiltonian:

$$H(p, x) = K(p) + U(x)$$
$$= \frac{p^2}{2} + \frac{x^2}{2}$$

- 3 steps of the leapfrog

$$p(t+\delta/2) = p(t) - (\delta/2)x(t)$$
$$x(t+\delta) = x(t) + (\delta)p(t+\delta/2)$$
$$p(t+\delta) = p(t+\delta/2) - (\delta/2)x(t+\delta)$$

- see Ex3.9-harm_osc_hamiltonian (ipynb, pdf, gif)

- **Recall: Leapfrog method**: (with time-step τ)

$$p_i\left(t+\frac{\tau}{2}\right) = p_i(t) - \frac{\tau}{2}\frac{\partial V}{\partial q_i}(q(t))$$
$$q_i(t+\tau) = q_i(t) + \tau\frac{p_i\left(t+\frac{\tau}{2}\right)}{m_i}$$
$$p_i(t+\tau) = p_i\left(t+\frac{\tau}{2}\right) - \frac{\tau}{2}\frac{\partial V}{\partial q_i}(q(t+\tau))$$

Ex. 10: McMC

HMC - Sampling from bivariate Gaussian

- This is a foundational example that demonstrates how HMC combines the best aspects of Hamiltonian dynamics simulation with Bayesian sampling, making it especially powerful for high-dimensional posterior distributions in modern statistical computing.

- Recall: The main idea behind Hamiltonian/Hybrid Monte Carlo is to develop a Hamiltonian function $H(\mathbf{x}, \mathbf{p})$ such that the resulting Hamiltonian dynamics allow us to efficiently explore some target distribution $p(\mathbf{x})$.

- The algorithm samples from a bivariate normal distribution with correlation coefficient 0.8. The HMC method is particularly effective for this type of problem because it can make large moves while maintaining high acceptance rates, avoiding the random walk behavior of simpler MCMC methods.

The target distribution $p(\mathbf{x})$ for this sampling exercise is a bivariate Gaussian with the following parameterization: $p(\mathbf{x}) = \mathcal{N}(\mu, \mathbf{\Sigma})$ with mean $\mu = [\mu_1, \mu_2] = [0, 0]$ and covariance

$$\mathbf{\Sigma} = \begin{bmatrix} 1 & \rho_{12} \\ \rho_{21} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}.$$

In order to sample from $p(\mathbf{x})$ (assuming that we are using a quadratic energy function), we need to determine the expressions for $U(\mathbf{x})$ and $\partial U(\mathbf{x})/\partial x_i$. Recall that the target potential energy function can be defined from the canonical form as $U(\mathbf{x}) = -\log(p(\mathbf{x}))$. So, taking the negative log of the Gaussian distribution above, we define the potential energy function

$$E(\mathbf{x}) = -\log\left(e^{-\frac{\mathbf{x}^\mathbf{T}\mathbf{\Sigma}^{-1}\mathbf{x}}{2}}\right) - \log Z,$$

where Z is the normalizing constant for a Gaussian distribution and can be ignored because it will cancel.

The potential energy function is then simply,

$$U(\mathbf{x}) = \frac{\mathbf{x^T \Sigma^{-1} x}}{2}$$

with partial derivatives

$$\frac{\partial U(\mathbf{x})}{\partial x_i} = x_i.$$

We are now in possession of all the ingredients for implementing the HMC algorithm.

- The algorithm samples from a bivariate normal distribution with correlation coefficient 0.8. The HMC method is particularly effective for this type of problem because it can make large moves while maintaining high acceptance rates, avoiding the random walk behavior of simpler McMC methods.

SOLUTION:

- see `Ex3.10-HMC-2D-Gaussian` (ipynb, pdf)

Ex. 11: FWI using HMC

- Fichtner's HMC lab paper in *GJI* (2023), 235.

  $\Rightarrow$ use HMCLab to reproduce the results

Ex. 12: FWI using VI

- Zhang, Curtis review in *Advances in Geophysics*, 62 (2021).

  $\Rightarrow$ use VIP package to reproduce the results

  $\Rightarrow$ compare with HMC

Ex.13: PPL-pymc-L-V

- **Rank plots** are histograms of the ranked posterior draws (ranked over all chains) plotted separately for each chain. If all of the chains are targeting the same posterior, we expect the ranks in each chain to be uniform, whereas if one chain has a different location or scale parameter, this will be reflected in the deviation from uniformity. If rank plots of all chains look similar, this indicates good mixing of the chains. This plot was introduced by Aki Vehtari, Andrew Gelman, Daniel Simpson, Bob Carpenter, Paul-Christian Burkner (2021): Rank-normalization, folding, and localization: An improved R-hat for assessing convergence of MCMC. Bayesian analysis, 16(2):667-718.

In this exercise, we will use `pymc` to solve a parameter estimation problem for the Lotka-Volterra (predator-prey) system

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \alpha x - \beta xy,$$
$$\frac{\mathrm{d}y}{\mathrm{d}t} = -\gamma y + \delta xy.$$

- The state vector $X(t) = [x(t), y(t)]$ comprises the densities of the prey and the predator species respectively.

- Parameters $\boldsymbol{\theta} = [\alpha, \beta, \gamma, \delta, x(0), y(0)]$ are the unknowns that we wish to infer from experimental observations.

SOLUTION:

- Based on the experiments in this notebook, the most simple and efficient method for performing Bayesian inference on the Lotka-Volterra equations was to specify the ODE system in Scipy, wrap the function as a Pytensor op, and use a Differential Evolution Metropolis (DEMetropolis) sampler in PyMC.

- see `Ex3.13-ODE_Lotka_Volterra_DIP_BIP` (ipynb, html)

Example 14: Comparison of Priors for McMC

In two Sonnet 4.5 generated examples, we compare performance of HMC and MH versions of McMC for 4 priors:

1. flat (uniform)
2. log-normal
3. half-normal
4. truncated-normal

- In Ex3.14a-bayesian_ode_priors (ipynb, pdf) we study the Lotka-Volterra system using pymc with NUTS/HMC.
- In Ex3.14b-SIR_prior_demo we study the SIR system, using a hand-coded MH sampler.