Introduction to Markov Chain Monte Carlo

Olivier Le Maître
with Omar Knio (KAUST)

1 Centre de Mathématiques Appliquées, CNRS
Ecole Polytechnique, Palaiseau, France
https://perso.limsi.fr/olm/
olivier.le-maitre@polytechnique.edu

DCSE Fall school on Reduced Order Modeling and UQ
Recall the Bayesian formula:

\[ P_{\text{post}}(q|D) \propto L(D|q)P_{\text{prior}}(q). \]

- Likelihood involves discrepancies between observations and model predictions:
  \[ L(D|q) = H(D - U(q)). \]
- Generally, no close form expression for \( U(q) \), but \( P_{\text{post}}(q|D) \) can be evaluated.

**The objective is to generate samples** \( q \sim P_{\text{post}}(q|D) \).
Table of content

1. Markov chain Monte Carlo
   - Overview
   - Metropolis Algorithm
   - Quality of the MCMC chain

2. Density Estimation
   - Histogram Method
   - Kernel Density estimation
   - Smoothing parameter
Markov chain Monte Carlo (MCMC) methods: class of algorithms aimed at simulating direct draws from some complex distribution of interest.

Origin of the name: one uses the previous sample values to randomly (Monte Carlo) generate the next sample value, thus creating a Markov chain.

A Markov chain is indeed defined as a process where the transition probability between the current and following state is only a function of the current state.

Many different flavors of generating a Markov Chain. Focus on Metropolis-Hastings algorithm: a random walk using a proposal density and a method for accepting/rejecting proposed moves.
The states of the chain after a large number of steps are used as samples of the desired distribution.

The quality of the sample improves as a function of the number of steps.

The more difficult problem is to determine how many steps are needed to converge to the stationary distribution within an acceptable error: usually one needs at least $\sim 10000$ samples.

A good chain is one that has rapid mixing, i.e. the stationary distribution is reached quickly starting from an arbitrary position and the target probability is explored well and efficiently.

A common application of these algorithms is for numerically calculating multi-dimensional integrals.

Let's look in detail at the Metropolis algorithm and how to generate samples from a certain distribution.
**Metropolis (MH)**

- MH algorithm can draw samples from a target probability distribution, $\pi$, requiring only the knowledge of a function proportional to the target PDF.
- It uses a proposal distribution, $P$, to generate (Markov chain) candidates that are accepted or rejected according to a certain rule. Let $P$ be a Gaussian for simplicity.

1. Let $\xi_t$ be the current state.
MH algorithm can draw samples from a target probability distribution, \( \pi \), requiring only the knowledge of a function proportional to the target PDF.

It uses a proposal distribution, \( P \), to generate (Markov chain) candidates that are accepted or rejected according to a certain rule. Let \( P \) be a Gaussian for simplicity.

1. Let \( \xi_t \) be the current state.
2. Draw a candidate \( \xi' \) from a Gaussian centered on the current state: \( \xi' \sim \mathcal{N}(\xi_t, \text{Cov}) \) where Cov is chosen a priori.
MH algorithm can draw samples from a target probability distribution, \( \pi \), requiring only the knowledge of a function proportional to the target PDF.

It uses a proposal distribution, \( P \), to generate (Markov chain) candidates that are accepted or rejected according to a certain rule. Let \( P \) be a Gaussian for simplicity.

1. Let \( \xi_t \) be the current state.
2. Draw a candidate \( \xi' \) from a Gaussian centered on the current state: \( \xi' \sim \mathcal{N}(\xi_t, \text{Cov}) \) where Cov is chosen a priori.
3. Calculate the ratio:
\[
 r = \frac{\pi(\xi')}{\pi(\xi_t)},
\]
4. Draw a random number \( \alpha \sim \mathcal{U}(0, 1) \).
5. Chain moves (i.e. candidate is accepted/rejected) according to:
\[
 \xi_{t+1} = \begin{cases} 
 \xi' & \text{if } \alpha < r, \\
 \xi_t & \text{otherwise} 
\end{cases}
\]
6. Repeat for next step \( t \).
MH algorithm can draw samples from a target probability distribution, $\pi$, requiring only the knowledge of a function proportional to the target PDF.

It uses a proposal distribution, $P$, to generate (Markov chain) candidates that are accepted or rejected according to a certain rule. Let $P$ be a Gaussian for simplicity.

1. Let $\xi_t$ be the current state.

2. Draw a candidate $\xi'$ from a Gaussian centered on the current state: $\xi' \sim \mathcal{N}(\xi_t, \text{Cov})$ where Cov is chosen a priori.

3. Calculate the ratio:
   
   $$ r = \frac{\pi(\xi')}{\pi(\xi_t)}, $$

4. Draw a random number $\alpha \sim U(0, 1)$.
Markov chain Monte Carlo

Density Estimation

Metropolis Algorithm

**Metropolis (MH)**

- MH algorithm can draw samples from a target probability distribution, $\pi$, requiring only the knowledge of a function proportional to the target PDF.
- It uses a proposal distribution, $P$, to generate (Markov chain) candidates that are accepted or rejected according to a certain rule. Let $P$ be a Gaussian for simplicity.

1. Let $\xi_t$ be the current state.
2. Draw a candidate $\xi'$ from a Gaussian centered on the current state: $\xi' \sim \mathcal{N}(\xi_t, \text{Cov})$ where Cov is chosen a priori.
3. Calculate the ratio:
   $$r = \frac{\pi(\xi')}{\pi(\xi_t)}.$$
4. Draw a random number $\alpha \sim U(0, 1)$.
5. Chain moves (i.e. candidate is accepted/rejected) according to:
   $$\xi_{t+1} = \begin{cases} 
   \xi' & \text{if } \alpha < r, \\
   \xi_t & \text{otherwise}.
   \end{cases}$$
6. Repeat for next step $t$. 

![Diagram of the Metropolis Algorithm](image)
Suppose that you want to test MCMC to sample a certain bimodal PDF, \( \pi \), which is proportional to a mixture of two bivariate gaussians:

\[
\pi \propto 0.5 \ast N(\mu_1, \Sigma_1) + 0.5 \ast N(\mu_2, \Sigma_2).
\]

Method: Metropolis algorithm.

What about sensitivity to \( n \) and proposal amplitude?
Example

1. Choose the proposal distribution: e.g. a gaussian with covariance: $\Sigma_{prop} = 0.1 \times I_2$.
2. Choose a starting point: $\xi_0 = \{10, 10\}$.
3. Run the machinery for $n$ steps: draw a candidate, accept/reject, repeat loop.

What about sensitivity to $n$ and proposal amplitude?
Example

1. Choose the proposal distribution: e.g. a Gaussian with covariance: $\Sigma_{\text{prop}} = 0.1 \times I_2$.
2. Choose a starting point: $\xi_0 = \{10, 10\}$.
3. Run the machinery for $n$ steps: draw a candidate, accept/reject, repeat loop.

What about sensitivity to $n$ and proposal amplitude?
Suppose that you want to test MCMC to sample a certain bimodal PDF, \( \pi \), which is proportional to a mixture of two bivariate gaussians:

\[
\pi \propto 0.5 \cdot N(\mu_1, \Sigma_1) + 0.5 \cdot N(\mu_2, \Sigma_2)
\]

Method: Metropolis algorithm.

1. Choose the proposal distribution: e.g. a gaussian with covariance: \( \Sigma_{\text{prop}} = 0.1 \cdot I_2 \).
2. Choose a starting point: \( \xi_0 = \{10, 10\} \).
3. Run the machinery for \( n \) steps: draw a candidate, accept/reject, repeat loop.

4. Plot the \( n \) samples (this case \( n = 5000 \))

- The proposal amplitude, 0.1, must be varied to obtain good mixing and fast convergence.
- The number of samples, \( n \), must be as large as possible to have a reliable statistics.

What about sensitivity to \( n \) and proposal amplitude?
The proposal distribution has covariance: $\Sigma_{prop} = 0.1 \times I_2$.

Results for 3 different values of total steps $n = 500, 5000$ and $25000$.

The larger $n$, the better the approximation.
Sensitivity to proposal amplitude

- The proposal amplitude must be tuned to obtain good exploration of the space and fast convergence of the chain toward the high-probability regions.
- Results shown for $0.005 \times I_2$, $0.1 \times I_2$ and $50 \times I_2$.
- The smaller the proposal amplitude, the larger the number of the accepted moves.
- Large proposals lead to small acceptance and slow exploration of the space.
- Ideally, the acceptance rate should be between 30 to 60%.

\[
\begin{align*}
\text{amplitude} &= 0.005 \\
\text{acceptance rate} &\sim 95 \%
\end{align*}
\]

\[
\begin{align*}
\text{amplitude} &= 0.1 \\
\text{acceptance rate} &\sim 50 \%
\end{align*}
\]

\[
\begin{align*}
\text{amplitude} &= 50 \\
\text{acceptance rate} &\sim 4 \%
\end{align*}
\]
To evaluate the mixing properties of a chain:

- visually should look like a white noise.
- the autocovariance should be rapidly decaying.
- the acceptance rate should be 30 to 60%.

Before computing statistics, the initial steps before convergence should be dropped: these steps are referred to as “burn-in” period.

The burn-in period is estimated from the autocorrelation as the step at which it drops to and becomes oscillatory around zero: in this case it is about 3000 steps.
## Table of content

1. **Markov chain Monte Carlo**
   - Overview
   - Metropolis Algorithm
   - Quality of the MCMC chain

2. **Density Estimation**
   - Histogram Method
   - Kernel Density estimation
   - Smoothing parameter
Sampling strategies

- Given a vector of QoI (quantity of interest) $S \in \mathbb{R}^n$, we would like to obtain \textbf{probabilistic information} regarding $S$.
- We have seen that the mean and covariance of $S$ are given by
  $$\mathbb{E}[S] = S_0, \quad \mathbb{E}[SS^T] \in \mathbb{R}^{n \times n}.$$  
- Higher moments can also be derived, in particular \textbf{coefficient of variation}, skewness & flatness factors,\ldots
- No direct mean to assess probabilities of events, for instance
  $$P(a \leq S_i < b), \quad P(S > s), \quad P(\{S_i < a\} \cap \{S_j > b\}), \quad \ldots$$
- Such probabilities must be estimated by \textbf{sampling strategies}. 
Sampling strategies

- Assume we want to estimate the probability of the event \( S \in \mathcal{R} \):
  \[
P(S \in \mathcal{R}).
  \]

We can apply the generic recipe:

1. generate a sample set \( \{\xi(i), i = 1, \ldots, M\} \) of \( \xi \) where \( \xi(i) \sim p_\xi \),
2. construct the sample set \( \{S(i) = S(\xi(i)), i = 1, \ldots, M\} \) solving the model,
3. estimate the probability using the empirical estimator by
   \[
P(S \in \mathcal{R}) = \lim_{M \to \infty} \frac{\# \text{of samples } S(i) \in \mathcal{R}}{M}.
   \]

In other words, we make use of the **relative frequency** to estimate probabilities.

- This approach raises several concerns, regarding convergence, estimation of low probability events, . . .

- **Observe:** the empirical estimates are random variables, since they are based on a random sample set.
We denote $S$ the sample space, that is $P(S \in S) = 1$, and assume that $S$ has a (smooth) density denoted $\pi$

$$\pi : S \mapsto \mathbb{R}_+, \quad \int_S \pi(s) ds = 1.$$ 

The question becomes:

**how to approximate $\pi$** from a sample set $\{S^{(i)}\}$?
The simplest density estimation: histogram method

- Consider the 1-d case first, that is \( n = 1 \) and \( \mathbb{S} \subset \mathbb{R} \).
- We partition \( \mathbb{S} \) into uniform bins of size \( h \).
- Let \( b_i = [x_i - h/2, x_i + h/2) \) be the bin centered on \( x_i \), and define

\[
p_i = \frac{\text{# of samples } S^{(i)} \in b_i}{M},
\]

the relative frequency of the \( i \)-th bin.

- **Observe:** \( \sum_i p_i = 1 \).
- The density is then estimated by

\[
\pi(s) \approx \pi_h(s) = \frac{1}{h} \sum_i p_i \text{I}_{b_i}(s).
\]

- \( \pi_h \) is a piecewise constant approximation of \( \pi \), satisfying

\[
\forall s \in \mathbb{S}, \pi_h(s) \geq 0, \quad \int \pi_h(s)ds = \sum_i \int_{b_i} \pi_h(s)ds = \sum_i \frac{1}{h} p_i h = 1.
\]
The histogram method is simple and intuitive.

It can be easily extended to $S \subset \mathbb{R}^n$, using for instance hyper-rectangular bins or any other partition of $S$.

**But:**

- the approximation depends on the position (centroid $x_i$) of the bins (orientation too),
- it is susceptible to artifacts (choice of the bins size, outliers, . . . ),
- control of accuracy is difficult in high dimension.

A less arbitrary and more robust approach is needed.
Example: Gaussian mixture for 5000 samples
Kernel density estimation (KDE)

Recall the definition of the probability that $S$ falls in some region $\mathcal{R}$:

$$P(S \in \mathcal{R}) = \int_{\mathcal{R}} \pi(s) ds = P_{\mathcal{R}}.$$

Then, if we have $M$ vectors independently drawn at random from $\pi$, the probability that $k$ of these vectors fall in $\mathcal{R}$ is

$$P(k|M) = \binom{M}{k} P_{\mathcal{R}}^k (1 - P_{\mathcal{R}})^{M-k}.$$

It can be shown that the mean and variance of the ratio $k/M$ are

$$E[k/M] = P_{\mathcal{R}}, \quad \text{Var}[k/M] = E \left[ \left( \frac{k}{M} - P_{\mathcal{R}} \right)^2 \right] = \frac{P_{\mathcal{R}}(1 - P_{\mathcal{R}})}{M}.$$

Therefore, as $M \to \infty$ becomes large the mean fraction of the points falling within $\mathcal{R}$ is (as one would expect)

$$\frac{k}{M} = P_{\mathcal{R}}.$$
If $\mathcal{R}$ is small enough, $\pi$ should not vary much over $\mathcal{R}$ and one would expect

$$P_{\mathcal{R}} = \int_{\mathcal{R}} \pi(s')ds' \approx \pi(s)V,$$

$\forall s \in \mathcal{R}$ and where $V$ is the volume of $\mathcal{R}$.

Consequently, we shall consider the approximation

$$\pi(s \in \mathcal{R}) \approx \frac{k}{MV}.$$

Of course, we want $V$ small enough for the constant approximation to be valid, and we need $M$ large enough for the limit $k/M$ to make sense. Clearly, $M$ should increase as $V$ is decreased. This calls for a compromise.
Consider

\[ K(u) = \begin{cases} 
1 & \text{if } |u| < 1/2, \\
0 & \text{otherwise}. 
\end{cases} \]

(if \( n \) dimensions, \( K(u) = 1 \) if \( |u_j| < 1/2 \) for \( j = 1, \ldots, n \)).
In words, \( K = 1 \) for \( u \) in the unit hypercube centered at the origin. This function is known as the Parzen window.

- The quantity

\[ K \left( \frac{s - S^{(i)}}{h} \right) \]

is equal to 1 if the random sample \( S^{(i)} \) is inside the hypercube of side \( h \) centered at \( s \).

- The total number of samples inside this hypercube is

\[ k = \sum_{i=1}^{M} K \left( \frac{s - S^{(i)}}{h} \right). \]
Consider

\[ K(u) = \begin{cases} 
1 & \text{if } |u| < 1/2, \\
0 & \text{otherwise.} 
\end{cases} \]

(if \( n \) dimensions, \( K(u) = 1 \) if \( |u_j| < 1/2 \) for \( j = 1, \ldots, n \)).

In words, \( K = 1 \) for \( u \) in the unit hypercube centered at the origin. This function is known as the Parzen window.

- The quantity

\[ K \left( \frac{s - S^{(i)}}{h} \right) \]

is equal to 1 if the random sample \( S^{(i)} \) is inside the hypercube of side \( h \) centered at \( s \).

- Consequently, we shall consider the approximation

\[ \pi_{KDE}(s) = \frac{1}{Mh^n} \sum_{i=1}^{M} K \left( \frac{s - S^{(i)}}{h} \right). \]

Parzen Window resembles histogram, except bin are centered on \( s \).
Example: Gaussian mixture for 5000 samples
To understand the role of $K$, compute the expectation of $\pi_{KDE}$

$$
\mathbb{E}[\pi_{KDE}(s)] = \frac{1}{h^n} \frac{1}{M} \sum_{i=1}^{M} \mathbb{E} \left[ K \left( \frac{s - S(i)}{h} \right) \right] = \frac{1}{h^n} \int_{S} K \left( \frac{s - s'}{h} \right) \pi(s') \, ds'.
$$

$\mathbb{E}[\pi_{KDE}(s)]$ is equal to the convolution of the true density ($\pi$) with the kernel $K$. As $h \to 0$, the kernel goes to $\delta$ (Dirac delta-function), in the sense of distribution:

$$
\pi_{KDE}(s) \to \pi(s).
$$

This observation paves the way to better choice of kernel, in particular using smooth functions $K(s)$, with the property

$$
\int K(s) \, ds = 1.
$$

Typically $K : S \mapsto \mathbb{R}$ are chosen as radially symmetric, positive and unimodal.
Role of the kernel

To understand the role of $K$, compute the expectation of $\pi_{KDE}$

\[
E[\pi_{KDE}(s)] = \frac{1}{h^n} \frac{1}{M} \sum_{i=1}^{M} E \left[ K \left( \frac{s - S(i)}{h} \right) \right] = \frac{1}{h^n} \int_{\mathcal{S}} K \left( \frac{s - s'}{h} \right) \pi(s') ds'.
\]

$E[\pi_{KDE}(s)]$ is equal to the convolution of the true density ($\pi$) with the kernel $K$. As $h \rightarrow 0$, the kernel goes to $\delta$ (Dirac delta-function), in the sense of distribution:

\[
\pi_{KDE}(s) \rightarrow \pi(s).
\]

- This observation paves the way to better choice of kernel, in particular using smooth functions $K(s)$, with the property

\[
\int K(s) ds = 1.
\]

- A common choice is Gaussian Kernel:

\[
K(s) = \frac{1}{(2\pi)^{n/2}} \exp \left[ -(s^T s)/2 \right].
\]
Example: Gaussian mixture for 5000 samples

KDE with Gaussian kernel for different bandwidths:

- $h=1$
- $h=1/2$
- $h=1/4$
- $h=1/8$
- $h=1/16$
- $h=1/32$
It remains to fix $h$ (bandwidth, core-radius), which is a critical parameter:

- large $h$: too much smoothing,
- small $h$: too many spikes.

The best $h$ minimizes the error between the estimated and true densities. Using mean square error measure, we get

$$
\epsilon^2 = \mathbb{E} \left[ (\pi_{KDE}(s) - \pi(s))^2 \right] = \mathbb{E} [\pi_{KDE}(s) - \pi(s)]^2 + \mathbb{V} [\pi_{KDE}(s)].
$$

Bias-variance tradeoff:

- large $h$: reduces variance but increase bias
- small $h$: reduces bias but increase variance

How to pick $h$?
A priori selection. Assumes that the true distribution $\pi$ is Gaussian, and $K$ is the Gaussian kernel, one can get an explicit minimizer for $\epsilon$:

$$h_{opt} = 1.06\sigma M^{-1/5},$$

where $\sigma$ is the standard deviation of the distribution $\pi$.

Data based selection. For general distributions, better results are obtained using

$$h_{data} = 0.9AM^{-1/5}, \quad A = \min\left(\sigma, \frac{\text{IQR}}{1.34}\right),$$

where IQR is the inter-quantile range, defined as the difference between the 75% and 25% percentiles.
KDE with Gaussian kernel

Example: Gaussian mixture for 5000 samples

Closing remarks:
- This idea can be extended to the multivariate case (isotropic kernels)
- Use of variable $h$ to adapt local concentration of observations
- Many implementations available (MATLAB).