Hamiltonian Monte Carlo explained

Dec 19, 2016 • Alex Rogozhnikov •

MCMC (Markov chain Monte Carlo) is a family of methods that are applied in computational physics and chemistry and also widely used in bayesian machine learning.

It is used to simulate physical systems with Gibbs canonical distribution:

$$p(\mathbf{x}) \propto \exp\left(-\frac{U(\mathbf{x})}{T}\right)$$

Probability $p(\mathbf{x})$ of a system to be in the state $\mathbf{x}$ depends on the energy of the state $U(\mathbf{x})$ and temperature T.

This distribution describes positions and velocities of particles in the gas, for instance. In bayesian machine learning, it defines distribution of model parameters (such as weights of a neural network).

Example: normal distribution is a Gibbs distribution

What is Monte Carlo (and why is it needed)?

Suppose that you want to study the properties of some model with thousands of variables (for lattice models that's very few!). For instance, average energy:

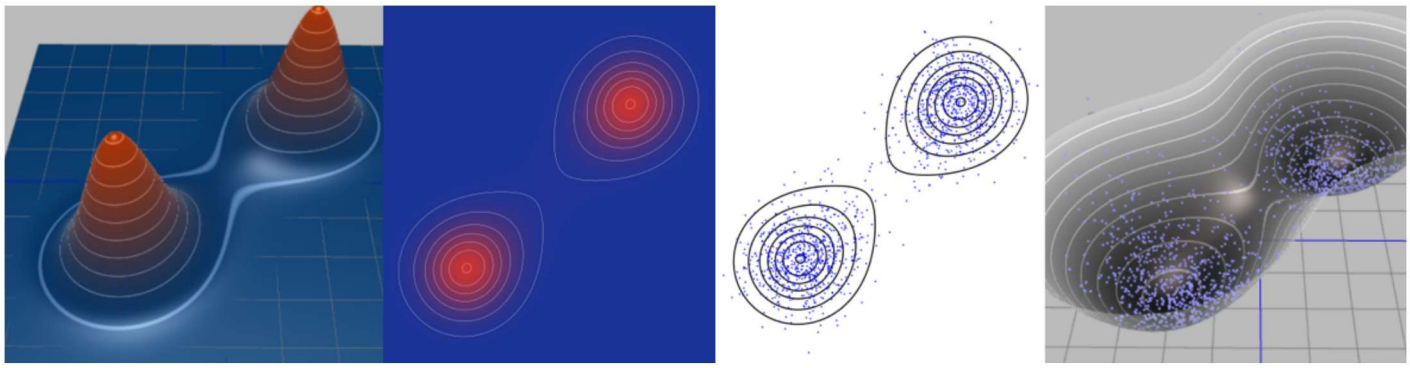
$$< U >= \int U(\mathbf{x})\, p(\mathbf{x}) d\mathbf{x}$$

is an integral over distribution. Computing this integral isn't possible, even using numerical approximations (lattice over 1000 variables is incredibly large).

By sampling from $p(\mathbf{x})$ an empirical estimate can be obtained:

$$< U >= \frac{1}{N}\sum_{i=1}^{N} U(\mathbf{x}_i), \qquad \mathbf{x}_i \sim p(\cdot)$$

This is **the idea of Monte Carlo**: to compute average energy / speed of molecules in the gas, take random molecules and average their energy / speed.



Integration with distribution pdf (left) is replaced with averaging over samples from distribution (blue points on the right).

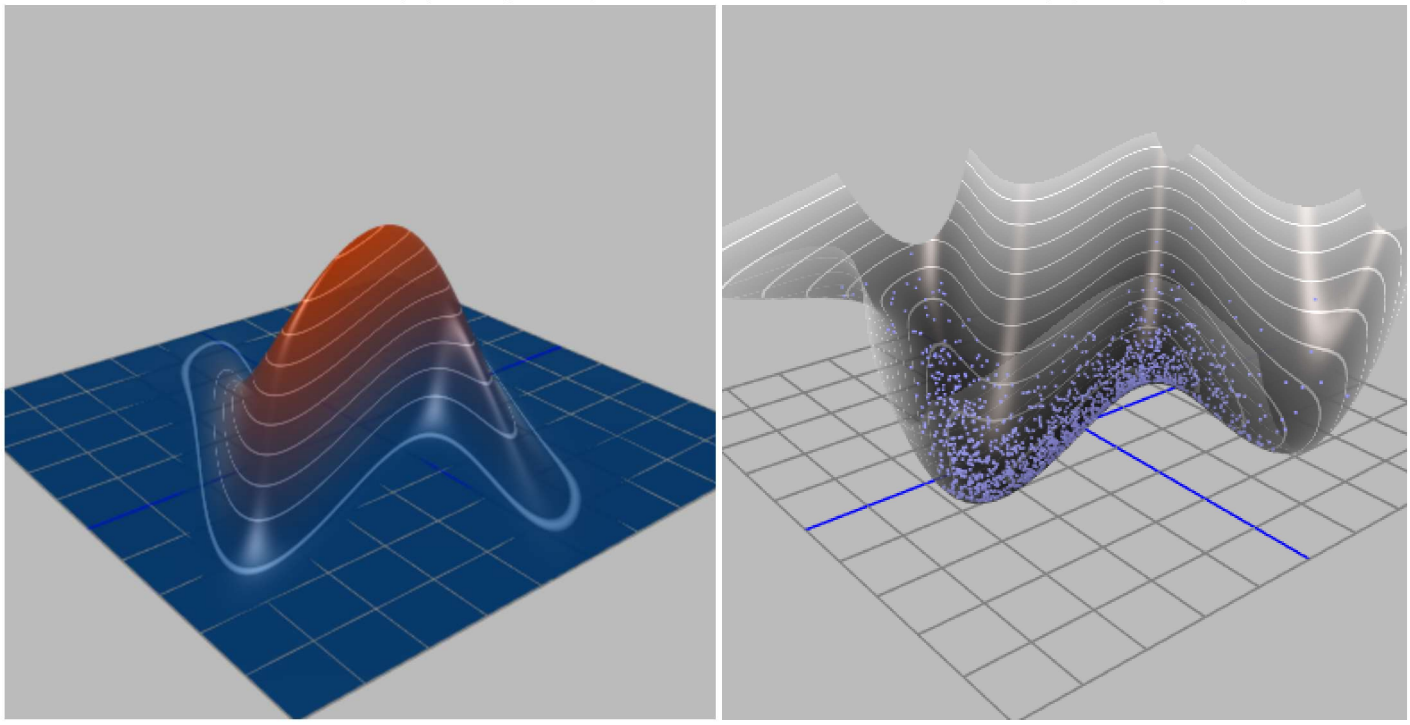
Rightmost plot demonstrates true samples on the energy surface, thus we can see corresponding energy $U(\mathbf{x})$.

In modern programming languages there are functions to sample from simple distributions: uniform, normal, Poisson... There is no simple procedure to sample from Gibbs distribution: it is times more complicated, but possible using Markov Chains.

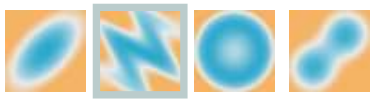
That's **our goal**: learn to sample from the canonical distribution.

How temperature influences canonical distribution:

unnormalized pdf $p(\mathbf{x}) = p(x_1, x_2)$ | Energy $U(\mathbf{x}) = U(x_1, x_2)$



Distribution



Show

☑ true samples from distribution

temperature T: 0.1

Play with a temperature and look at different distributions!

As intuition says, system has higher probability of staying in the states with lower energies. As temperature goe down, imbalance becomes stronger. When T tends to zero, the system stays in the state(s) with the lowest energy.

— So, we minimize energy?

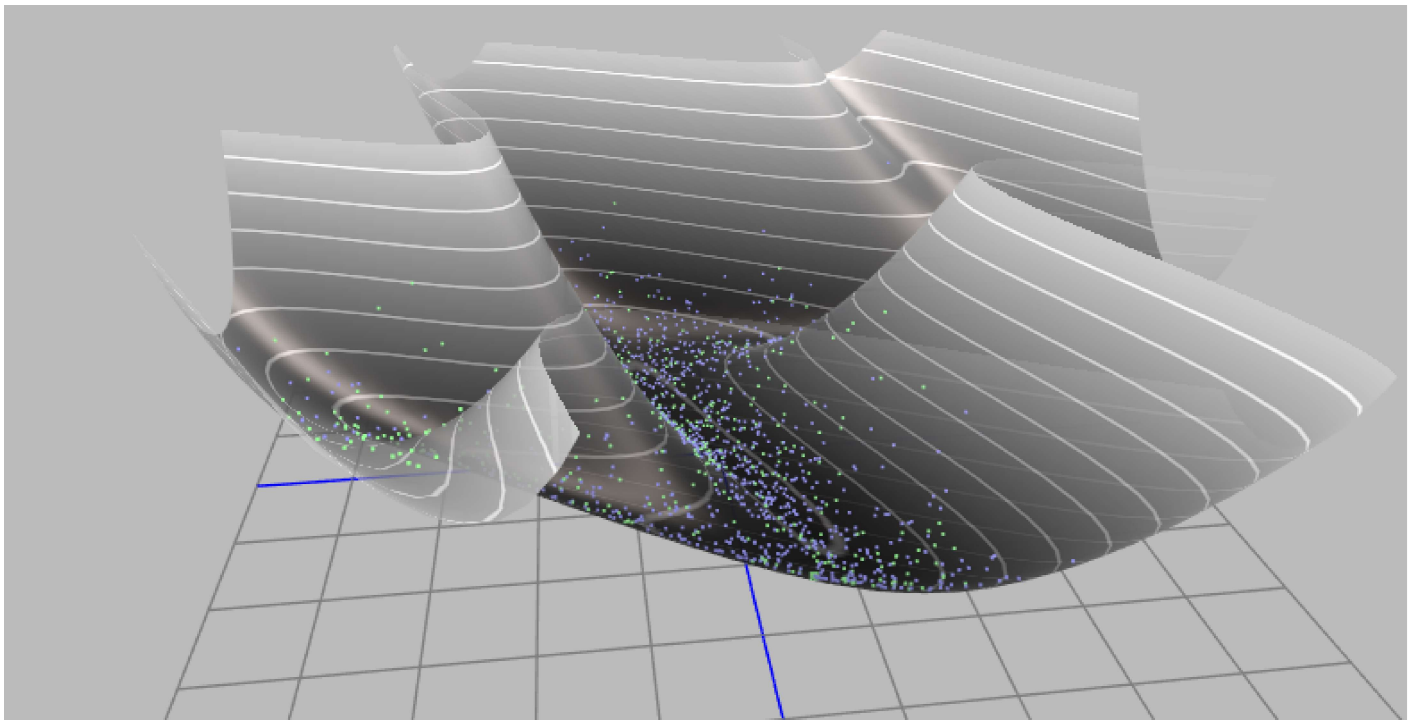
Not really. Minimal-energy state is very important, but it is wrong to think that system can exist only in this state. To get good estimates of the system properties, **a representative sampling** of possible states is required (blue points demonstrate the desired distribution).

Pro tip: for bayesian people using maximum a posteriori estimation is the same as taking state with the lowest energy, while sampling corresponds to using bayesian posterior distribution. The latter has its benefits (though it's not simple to obtain!).

Why sampling from Gibbs distribution is complex?

Metropolis-Hastings algorithm for MCMC

The simplest Markov Chain process that can sample from the distribution picks the neighbour of the current sta and either accepts it or rejects depending on the change in energy:



Algorithm produces a chain of states: $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$ (green points).
Each time a candidate from a neighbourhood of the last state is selected $\mathbf{x}'_n = \mathbf{x}_n + noise$ (noise is usually take to be gaussian with some spread σ).

With probability $p = \min\left[1, \exp\left(\frac{U(\mathbf{x}_n) - U(\mathbf{x}'_n)}{T}\right)\right]$ system accepts new state (jumps to the new state): $\mathbf{x}_{n+1} = \mathbf{x}'_n$ an with probability $1 - p$ new state is rejected: $\mathbf{x}_{n+1} = \mathbf{x}_n$.

Note, when energy is lower in new state $U(\mathbf{x}'_n) < U(\mathbf{x}_n)$ it is always accepted: $p = 1$. This way we give preference to the states with lower energies, while not restricting the algorithm to always decrease the energy. The lower temperature, the lower probability to increase energy.

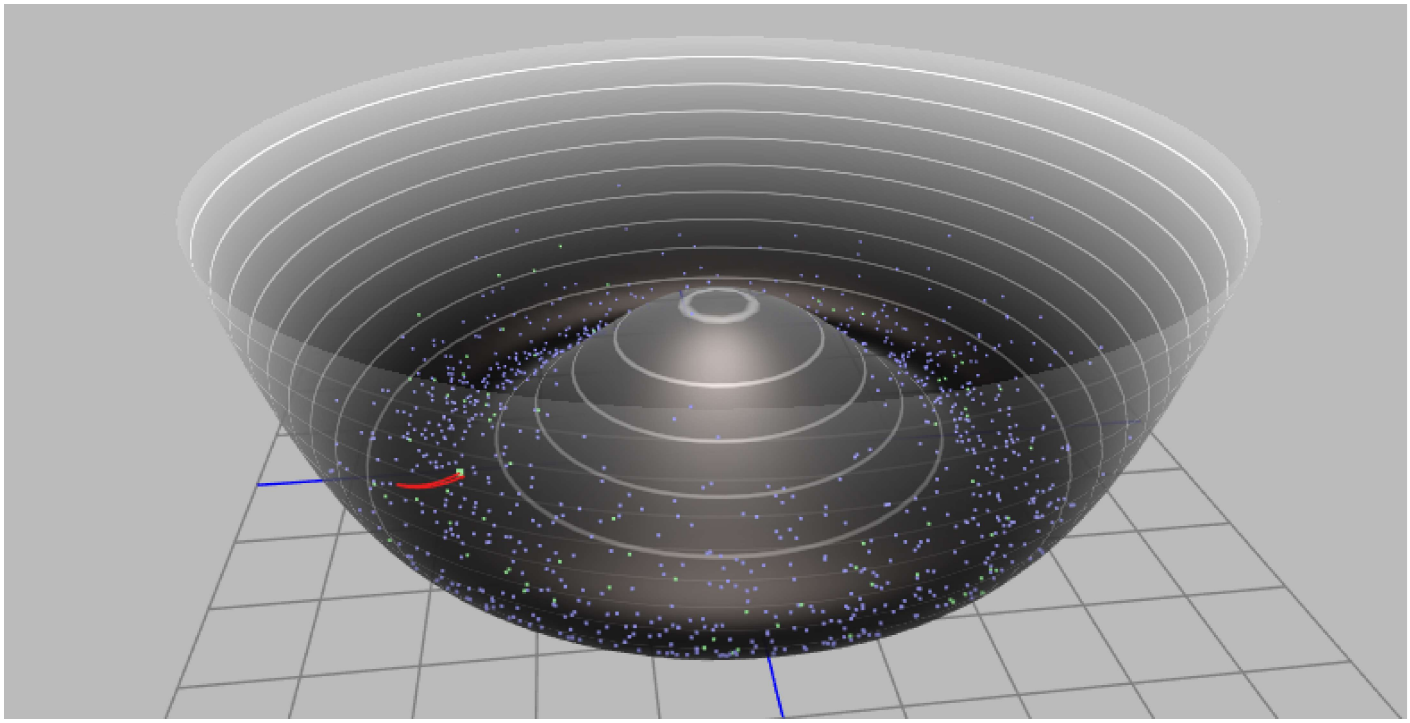
Why does Metropolis-Hastings work?

Problems of Metropolis-Hastings algorithm

Sampling high-dimensional distributions with MH becomes very inefficient in practice. A more efficient scheme i called Hamiltonian Monte Carlo (HMC).

Hamiltonian Monte Carlo

Physical analogy to Hamiltonian MC: imagine a hockey pluck sliding over a surface without friction, being stopped at some point in time and then kicked again in a random direction.



Hamiltonian MC employs the trick developed by nature (and well-known in statistical physics). Velocity v is adde to the parameters describing the system. Energy of the system consists of potential and kinetic parts:

$$E(\mathbf{x}, \mathbf{v}) = U(\mathbf{x}) + K(\mathbf{v}), \qquad K(\mathbf{v}) = \sum_i \frac{m\, v_i^2}{2}$$

Thus, velocities $\mathbf{v}$ and positions $\mathbf{x}$ have **independent** canonical distributions:

$$p(\mathbf{x}, \mathbf{v}) \propto \exp\left(\frac{-E(\mathbf{x}, \mathbf{v})}{T}\right) = \exp\left(\frac{-U(\mathbf{x})}{T}\right) \exp\left(\frac{-K(\mathbf{v})}{T}\right) \propto p(\mathbf{x})\, p(\mathbf{v}).$$

So once we can sample from joint distribution $p(\mathbf{x}, \mathbf{v})$, we also can sample $\mathbf{x}$ by ignoring computed velocities $\mathbf{v}$.

— How can we perform joint sampling?

— Is everything that simple?

Pros and cons of Hamiltonian Monte Carlo

There is no free lunch, and Hamiltonian MC has its price: HMC uses not only energy $U(\mathbf{x})$, but also it's gradient Hence possible applications are limited to the case when gradient exists and can be computed in reasonable time.

The major differences compared to Metropolis-Hastings are:

- distances between successive generated points are typically large, so we need less iterations to get representative sampling
- 'price' of a single iteration is higher, but HMC is still significantly more efficient
- Hamiltonian MC in most cases accepts new states (take a look at rejected samples in the demonstrations!)
- still, HMC has problems with sampling from distributions with isolated local minimums
  Investigate last distribution at low temperatures — 'puck' doesn't have enough energy to jump from the first minimum to the second over the energy barrier.

Bonus part: Passing energy barriers in Hamiltonian MC

Links

1. Radford M. Neal MCMC using Hamiltonian dynamics
   Interactive presentation mostly based on this summary.
2. Charles J. Geyer, Introduction to Markov Chain Monte Carlo
   Metropolis-Hastings algorithm together with MCMC practicalities
3. HMC implementation in theano
   When you don't want to compute derivatives in HMC...
4. Interactive demonstrations of machine learning
   A collection of nice visualizations

Plots in this demonstration are powered by webGL shaders and three.js library, so mobile (and quite old) browsers may have problems with rendering plots correctly.

Thanks to Tatiana Likhomanenko for reading previous versions of the post.