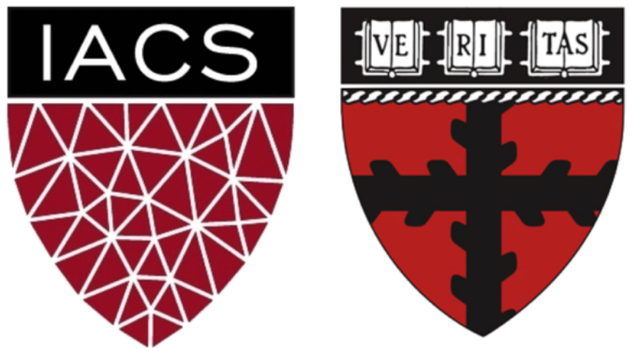
Lecture #15: Parallel Tempering and Stochastic HMC

AM 207: Advanced Scientific Computing

Stochastic Methods for Data Analysis, Inference and Optimization

Fall, 2020


IACS
VE RI TAS

Outline

1. Review of HMC
2. Parallel Tempering
3. Stochastic Gradient HMC

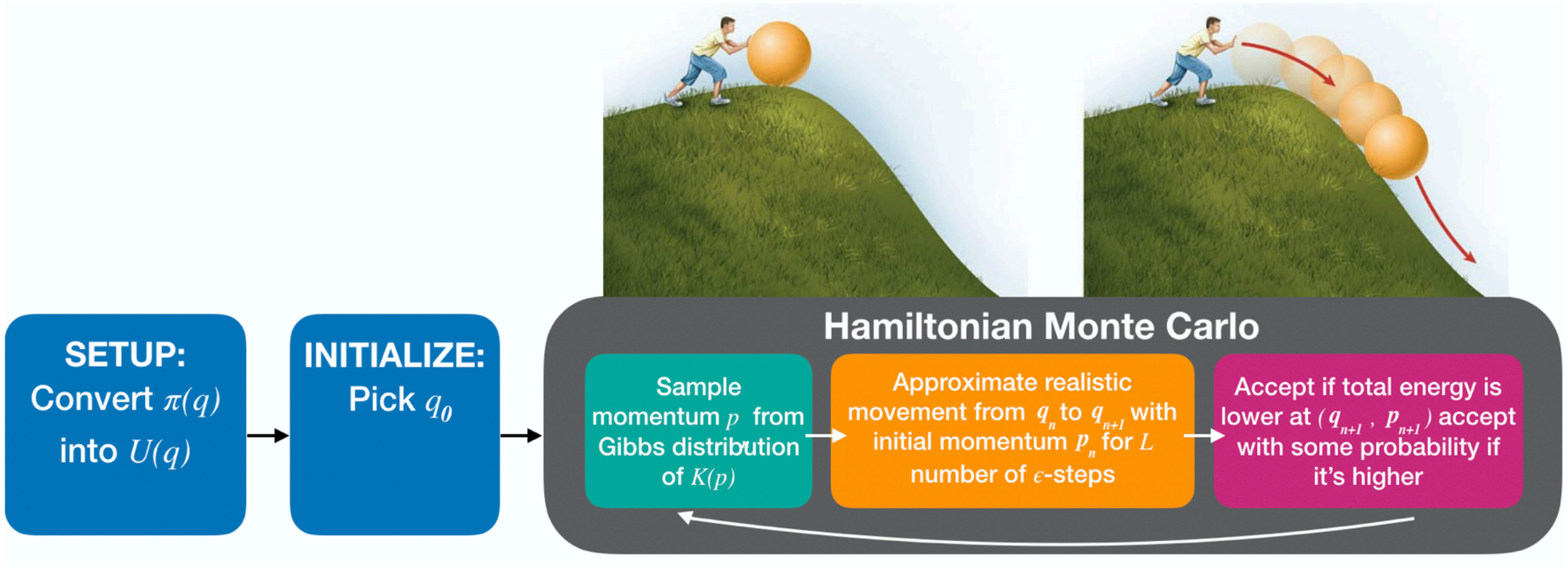
Review of HMC

Hamiltonian Monte Carlo (HMC)

We use the ***Gibbs distribution*** to transform between probability density functions and energy functions

$$U(q) = -\log \pi(q), \qquad \pi(q) = \frac{1}{Z}\exp\left\{ \frac{-U(q)}{T} \right\}, \; T = 1$$

This allows us to use gradient information when we sample from $\pi(q)$.



FAQs About HMC

1. **Question:** HMC seems complicated, do I really need to use it?
   Answer: Yes. For complex models of interest (non-conjugate, hierarchical, latent variable) HMC is the least complicated sampler you can use to perform reliable inference.

2. **Question:** Ok, but can I treat the theory as a black-box, i.e. can I just press some `model.sample()` button?
   Answer: No. HMC (like all samplers) must be tuned. That is, for many models and datasets, `model.sample()` will not have good performance. You need the theory to tell you which design choices need to be adjusted and in what way.

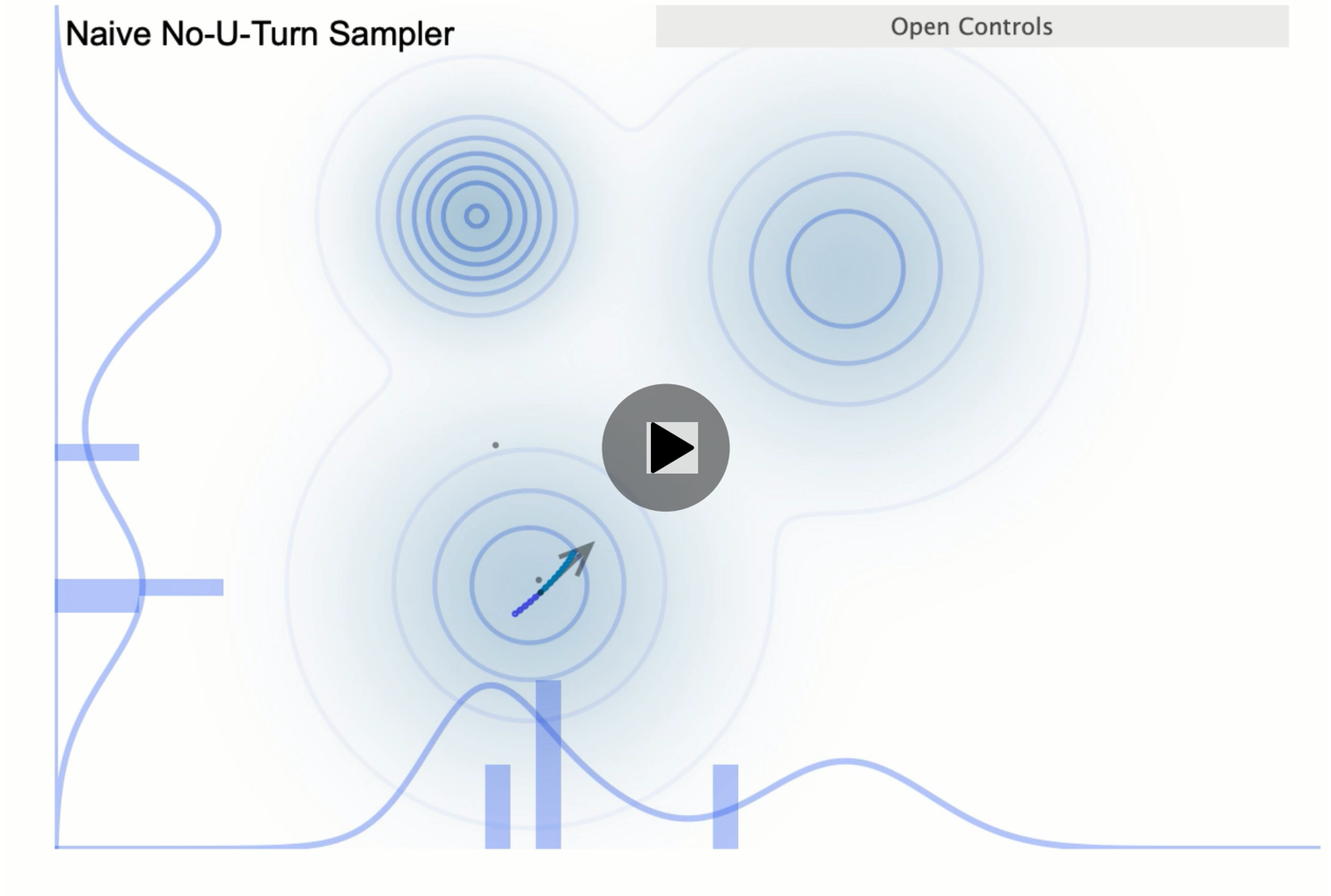
3. **Question:** But I don't need to implement it since `pymc3` has already done so, right?
   Answer: You need to implement HMC. Because `pymc3` is not going to scale well for Bayesian inference for models with neural network likelihoods and large datasets.

4. **Question:** We worked so hard to derive HMC, so it's state-of-the-art and can be applied to any Bayesian model for any dataset right?
   Answer: No. HMC may still be inefficient for complex posteriors and improving HMC is an active area of research (see this weeks readings). Futhermore, HMC **does not scale well to large datasets**.

HMC for Multimodal Distributions

In [2]:
```
HTML("""<video height="440" controls><source src="fig/hmc_multimodal.mov" type="
video/mp4"></video>""")
```

Out[2]:



Signs of Maybe Convergence?

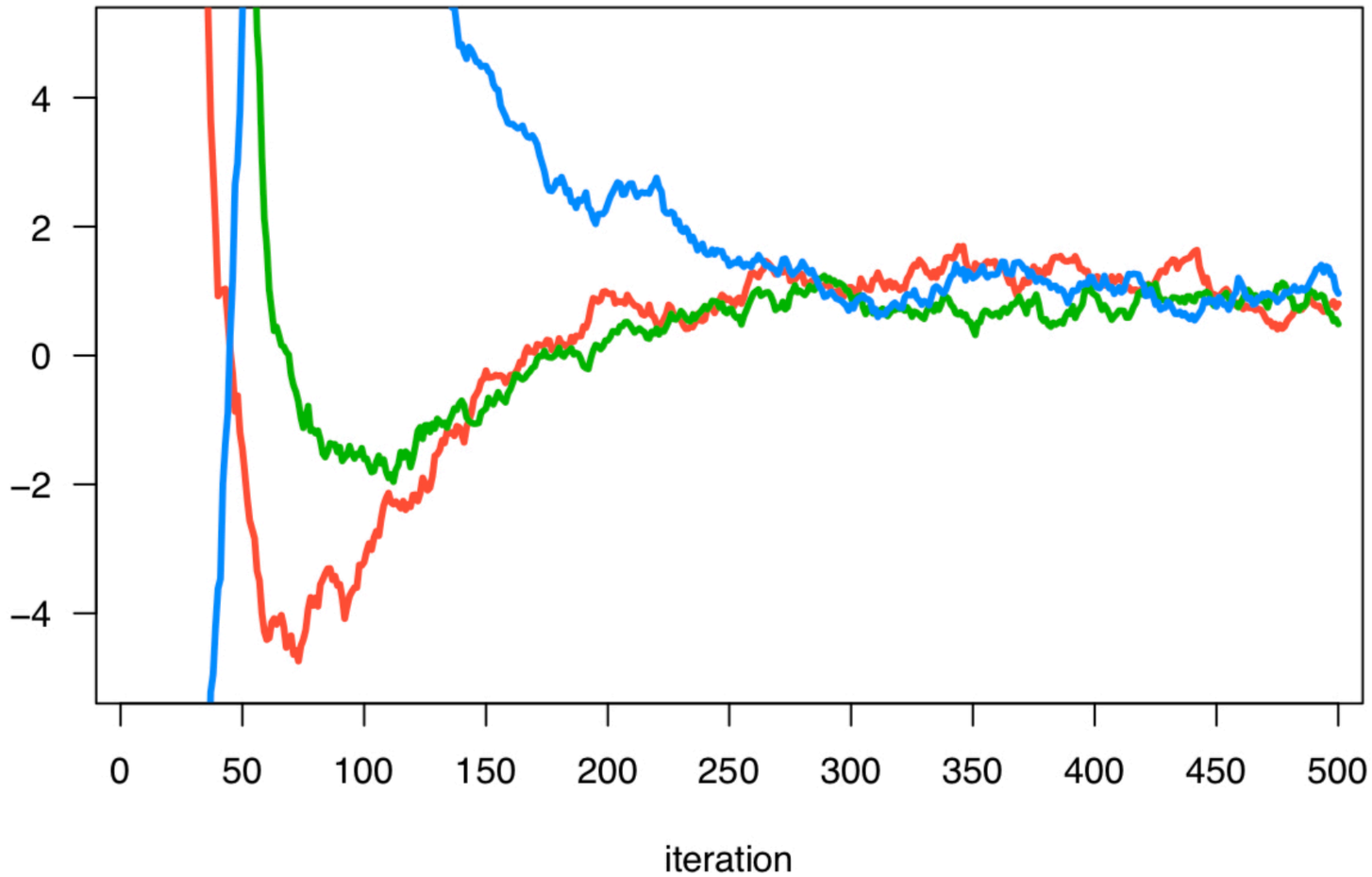
Look for:

1. Large segments of the chain should have give similar statistics (mean, variance etc)
2. Low correlations within states of the chain
3. "Reasonably high" acceptance rate of proposed steps
4. Multiple chains initialized from different initial points give similar results

Best practics:

1. Always run multiple chains initialized from very different random starting points
2. Always run your chains for as long as you can then burn and thin
3. Always check all relevant convergence diagnostics
4. Never be too certain: remember that there is no "proof" of convergence for finite chains!
5. Keep reading about best practice!

Visual Diagnostics: Traceplots of Multiple Chains



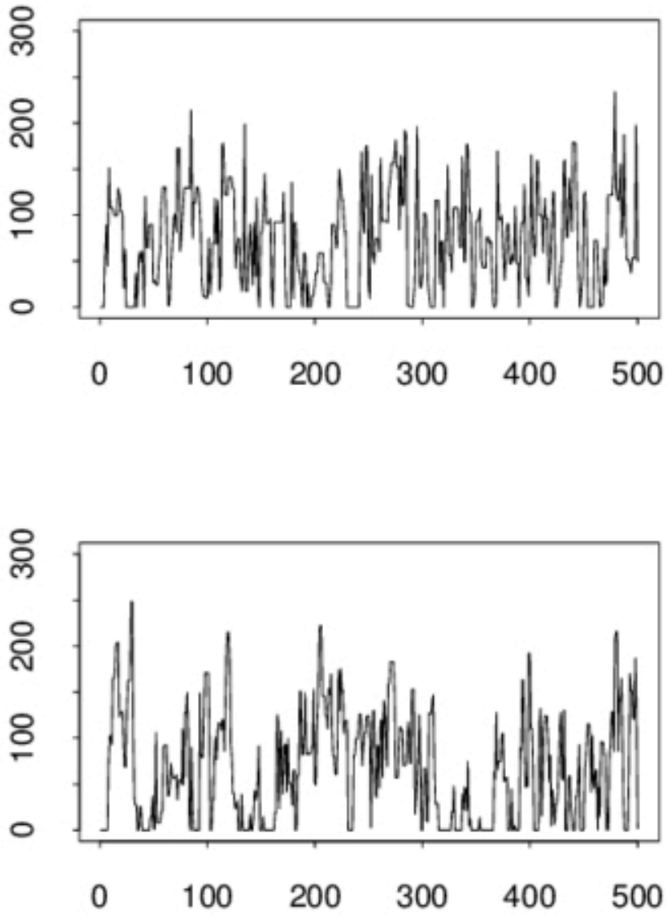
Autocorrelation: the "Effective" Sample Size

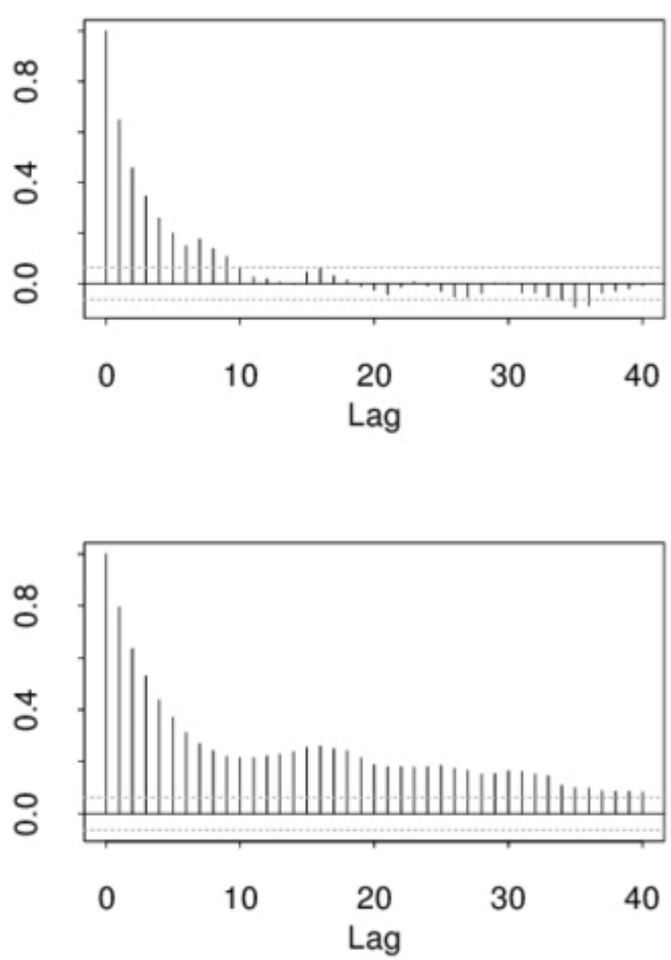
We quantify how much the samples in the chain are correlated with the samples obtained k-steps later (**the k-th lag**). The ***autocorrelation*** ρ_k is defined as

$$\rho_k = \frac{\sum_{n=1}^{N-k}(x_n - \bar{x})(x_{n+k} - \bar{x})}{\sum_{n=1}^{N}(x_n - \bar{x})^2}$$

We plot the autocorrelation for each $k = 1, \ldots, \frac{N}{2}$, and this ***autocorrelation plot*** tells us how much we to thin in order to obtain effectively independent samples. The autocorrelation plot gives us an idea of the ***effective sample size*** of the Markov chain.

Visual Diagnostics: The Autocorrelation Plot





Quantitative Diagnostics

Idea: measure between-chain and within-chain variability of a quantity of interest – if the chains have converged, these measures will be similar; otherwise, the between-chain variability will be larger.

1. ***Gelman & Rubin***: quantity $\widehat{R}_{\mathrm{GR}} = \frac{B}{W}$, which compares B the empirical variance of all the chains pooled and W the average emprical variance within each chain. If $\widehat{R}_{\mathrm{GR}}$ is large then the chains are very different (not converged). If $\widehat{R}_{\mathrm{GR}} = 1$ is ideal but in practice we accept $\widehat{R}_{\mathrm{GR}} < 1.05$.

2. ***Geweke***: takes two nonoverlapping parts (usually the first 0.1 and last 0.5 proportions) of the Markov chain and compares the means of both parts, using a difference of means test to see if the two parts of the chain are from the same distribution (the test statistic is a standard Z-score with the standard errors adjusted for autocorrelation).

Parallel Tempering

Multimodal Posteriors

But when are posteriors multimodal? Often, the posterior can be multimodal when the likelihood is ***non-identifiable***, i.e. there are multiple sets of model parameters that can explain the data equally well.

For example, the observed data likelihood of a Gaussian mixture model with 2 univariate components is:

$$p(y|\mu, \sigma^2, \pi) = \pi_1 \mathcal{N}(y; \mu_1, \sigma_1^2) + \pi_2 \mathcal{N}(y; \mu_2, \sigma_2^2)$$

But, given an observation y, there are multiple sets of model parameters μ, σ^2, π that will fit the data:

$$0.1\mathcal{N}(y; 1, 0.5) + 0.9\mathcal{N}(y; -2, 1) = 0.9\mathcal{N}(y; -2, 1) + 0.1\mathcal{N}(y; 1, 0.5)$$

That is, we can label the components however we want without changing the fit.

When we fit a Bayesian GMM, the posterior will contain multiple modes, one for each way of labeling the components.

Note: There are more non-trivial ways in which the likelihood of a GMM can be non-identifiable and, hence, its posterior multimodal!

The Effect of Temperature

Why is it hard for samplers to visit multiple modes in a target distribution?

MCMC samplers can only propose locally, so moving from one mode to another requires traveling through regions that are very unlikely under the target distribution.

In HMC terms, moving from one mode to another requires climbing a big hill -- this is called an ***energy barrier***. From simulated annealing we know that range of movement of a sampler of a Gibbs distribution is enhanced when we increase the temperature term:

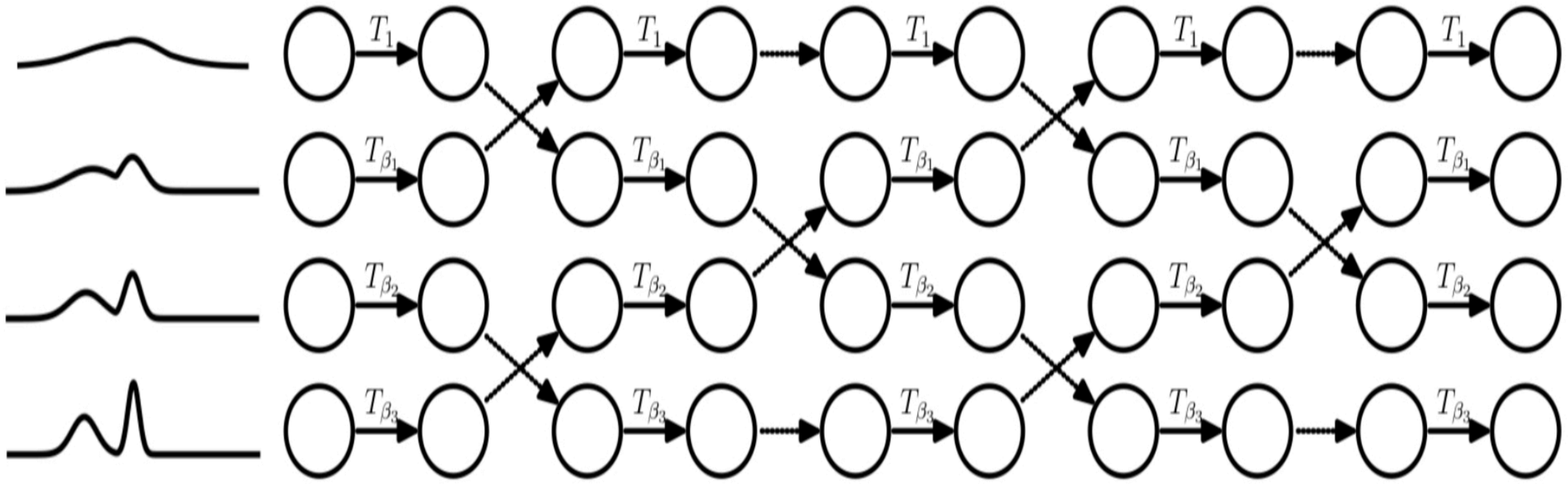
$$\pi(q) = \frac{1}{Z}\exp\left\{-\frac{-\log \pi(q)}{T}\right\}$$

Another way to say this is that when temperature is high, the poential energy landscape $\frac{-\log \pi(q)}{T}$ is flat, hence easier to explore.

But we can't simply set $T > 1$! Doing so means that we will not be sampling from the target $\pi(q)$ (i.e. the above equation will not hold).

The Idea of Parallel Tempering

Using one MCMC chain with $T > 1$ will produce incorrect samples. What if we use multiple chains: one for $T = 1$ (which will produce samples from $\pi(q)$) and other chains with $T > 1$, and we allow the chains to exchange samples once in a while?



Parallel Tempering with HMC

We set an increasing sequence of temperatures $T_0 = 1, \dots, T_M > 1$. For each temperature, denote the corresponding Gibbs distribution and potential energy function as follows:

$$\pi_m(q) = \frac{1}{Z}\exp\{-U_m(q)\}, \; U_m(q) = \frac{-\log \pi(q)}{T_m}$$

We denote the potential energy of $\pi(q)$ as $U(q) = -\log \pi(q)$.

Parallel Tempering HMC

1. initialize M number of HMC samplers: each using the same kinetic energy function, the m-th sampler using the potential energy function $U_m(q)$.

2. alternate between:

a. sample S samples from each chain independently

b. at the S-sample, the sample q_{m+1}^S from chain $m+1$ is exchanged with the sample q_m^S from chain m with the probability:

$$\alpha = \min\left\{1, \exp\left\{(1/T_m - 1/T_{m+1})(U(q_m^S) - U(q_{m+1}^S))\right\}\right\}$$

In the end, we keep the samples in the 0-th chain.

Can you show that parallel tempering is detailed balanced? What about irreducible and aperiodic?

Stochastic Gradient HMC

Problems with Scaling HMC

Let $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be our observed data (alternatively we can have a set of observed that that do not include any covariates), let the target distribution $p(\mathbf{w}|\mathcal{D})$ from which we want to sample be the posterior of the Bayesian model:

$$\begin{aligned} \mathbf{w} &\sim p(\mathbf{w}) \\ y_n|x_n, \mathbf{w} &\sim p(y_n|x_n, \mathbf{w}) \end{aligned}$$

Then the potenial energy function determined by the posterior $p(\mathbf{w}|\mathcal{D})$ is given by

$$U(\mathbf{w}) = \log p(\mathbf{w}|\mathcal{D}) = \sum_{n=1}^{N} \log p(y_n|x_n, \mathbf{w}) + \log p(\mathbf{w})$$

We see that during **each iteration** of HMC, whenever we need to compute $\nabla_{\mathbf{w}} U$, we have to evaluate $\sum_{n=1}^{N} \nabla_{\mathbf{w}} p(y_n|x_n, \mathbf{w})$ at each data point (x_n, y_n). Now if your data is large, this means that each step of HMC can take very long to compute!

Idea: use stochastic gradients for HMC! That is rather than computing the gradient over the entire dataset, we compute it over **mini-batches** of the data, just as we did in stochastic gradient descent.

Naïve Stochastic Gradient HMC

So what if we implemented HMC with mini-batch gradients? That is, what if we approximated $\nabla_{\mathbf{w}} U$ by

$$\nabla_{\mathbf{w}} U(\mathbf{w}) \approx \nabla \widetilde{U}(\mathbf{w}) = -\frac{|\mathcal{D}|}{|\widetilde{\mathcal{D}}|} \sum_{x_n \in \widetilde{\mathcal{D}}} \nabla_{\mathbf{w}} \log p(y_n | x_n, \mathbf{w}) + \nabla_{\mathbf{w}} \log p(\mathbf{w}). \;\; \widetilde{\mathcal{D}} \subset \mathcal{D}$$

Again, if the observed data are independent, then by the Central Limit Theorem we have that $\nabla \widetilde{U}(\mathbf{w}) \approx \nabla U(\mathbf{w}) + \mathcal{N}(0, V(\mathbf{w}))$, where V is the covariance of the gradient noise. So, mini-batch gradient is a noisy approximation of the true gradient $\nabla U(\mathbf{w})$.

Unfortunately, naïve stochastic gradient HMC does not preserve the target distribution $p(\mathbf{w}|\mathcal{D})$!

Corollary: The distribution $\pi(\mathbf{w}, p) \propto \exp(-H(\mathbf{w}, p))$, where p is the momentum, is no longer invariant under naïve stochastic gradient Hamiltonia dynamics.

We can correct the errors in naïve stochastic gradient Hamiltonian dynamics with a Metropolis-Hastings step (i.e. we don't accept all proposed samples), but in practice stochastic gradients produce large deviations from true Hamiltonia dynamics and hence **the acceptance rate will be low**! Furthermore, every MH step requires that we compute $p(\mathbf{w}|\mathcal{D})$, which **requires evaluating the likelihood at every data point**!

Source: Stochastic Gradient Hamiltonian Monte Carlo

Stochastic Gradient HMC with Friction

If HMC is like rolling a marble on a landscape made of ice - where the motion of the marble is determined completely by the gradient, then naïve stochastic gradient HMC is like rolling a marble on ice with a **wind that blows in random directions**.

Idea: we add friction to the surface, this will lessen if not cancel the effect of wind.

Let our kinetic energy function be $\frac{1}{2} p^\top M^{-1} p$. Naïve stochastic gradient HMC dynamics:

$$\begin{cases} d\mathbf{w} = M^{-1} p \, dt \\ d\mathbf{p} = -\nabla_{\mathbf{w}} U(\mathbf{w}) + \mathcal{N}(0, 2Bdt) \end{cases}$$

where $B = \frac{1}{2} \epsilon V(\mathbf{w})$, with V the covariance of gradient noise and ϵ the step-size.

Stochastic Gradient HMC with Friction dynamics:

$$\begin{cases} d\mathbf{w} = M^{-1} p \, dt \\ d\mathbf{p} = -\nabla_{\mathbf{w}} U(\mathbf{w}) - BM^{-1} p \, dt + \mathcal{N}(0, 2Bdt) \end{cases}$$

where $BM^{-1} p$ is a friction term that reduces the effect of the "random wind" $\mathcal{N}(0, 2Bdt)$.

If the friction term uses the exact gradient noise covariance, then SGHMC with Friction has exact dynamics - that is, **we do not need an Metropolis-Hastings correction step**!

Source: Stochastic Gradient Hamiltonian Monte Carlo

Stochastic Gradient HMC in Practice

In practice, we don't know the gradient noise covariance V and hence cannot compute B! Instead, we approximate $\widehat{B} \approx B$ and choose a user-defined friction coefficient C. Then, ***Stochastic Gradient HMC with Friction*** dynamics becomes:

$$\begin{cases} d\mathbf{w} = M^{-1} p \, dt \\ d\mathbf{p} = -\nabla_{\mathbf{w}} U(\mathbf{w}) - C M^{-1} p \, dt + \mathcal{N}(0, 2(C - \widehat{B})dt) + \mathcal{N}(0, 2Bdt) \end{cases}$$

Theorem: If $\widehat{B} = B$ then the dynamics of SGHMC with Friction yields the stationary distribution $\pi(\mathbf{w}) \propto \exp\{-H(\mathbf{w}, p)\}$.

That is, SGHMC with Friction is an MCMC sampler if our estimation $\widehat{B}$ of B (related to the true gradient noise variance) is exact. Note: this means that, **in practice, SGHMC with Friction is not an MCMC sampler**, since we can't know the exact gradient noise variance!

But **SGHMC is one of only very few ways we have for scaling MCMC methods to large datasets**.

Source: Stochastic Gradient Hamiltonian Monte Carlo