Ricochet Monte Carlo

Anonymous Authors[1]

Abstract

We introduce a novel dual-purpose algorithm for MCMC sampling and constrained nonlinear optimization. The method simulates a particle bouncing across a surface whose shape is determined by the target distribution or objective function, using a notion of entropic loss that resembles simulated annealing without any acceptance step. Similarly, the simple closed-form parabolic trajectories means we can compute near-exact candidate samples. This enables a more efficient dynamical simulation, reducing total evaluations of the target. Our method easily extends to handle nonlinear constraints. We demonstrate the effectiveness of our method on several distributions and nonlinear test functions, as well as a concrete prediction task.

1. Introduction

1.1. Related work

1.2. Outline of work

2. Ricochet Monte Carlo

The intuition behind Ricochet Monte Carlo (RMC) is to treat our target distribution or objective as a sort of hilly terrain across which a point particle freely bounces along. The points where the particle collides with this surface are candidate samples to either be accepted and rejected. The particle will continue to bounce, until it settles in some "valley", at which time it is picked up and tossed in a random direction to repeat the process.

Let $F(\boldsymbol{\theta})$ be some probability distribution to sample from, defined as

$$F(\boldsymbol{\theta}) \coloneqq \frac{E(\boldsymbol{\theta})}{Z}$$

where Z is some intractable normalizing constant and $E(\boldsymbol{\theta})$ is the associated joint distribution, which is assumed tractable and known. We are interested in generating samples $\boldsymbol{\theta}_i \sim F$ that live in a continuous d-dimensional vector space, where i means the ith sample. We perform the standard transformation as done in HMC and define our *surface* as

$$S(\boldsymbol{\theta}) \coloneqq -\log E(\boldsymbol{\theta}).$$

If instead $F(\boldsymbol{\theta})$ is a nonlinear objective function to minimize, then we simply set $S(\boldsymbol{\theta}) \coloneqq F(\boldsymbol{\theta})$.

2.1. Auxillary variables

For each value of $\boldsymbol{\theta}$, we associate an auxillary *height* scalar h. We define a $(d+1)$-length *position* vector $\mathbf{q} \coloneqq [\boldsymbol{\theta}; h]$. We also create an auxillary $(d+1)$-length *position* vector $\mathbf{p}$. We define

$$\boldsymbol{\theta}(\mathbf{q}_i) \coloneqq \boldsymbol{\theta}_i$$
$$h(\mathbf{q}_i) \coloneqq h_i$$

to respectively refer to the specific sample and height components of any particular $\mathbf{q}_i$. Similarly, we overload $\boldsymbol{\theta}(\mathbf{p}_i)$ to be the vector of the first d entries of $\mathbf{p}_i$.

We maintain an invariant where the particle is always "above" the surface S, i.e.

$$C_S(\mathbf{q}) \coloneqq h(\mathbf{q}) - S(\boldsymbol{\theta}(\mathbf{q})) > 0. \tag{1}$$

Equation (1) defines the *surface collision* constraint C_S. This terminology will be used to distinguish it from any later nonlinear constraints that will be handled in the optimization formulation. All together, $\mathbf{q}$, $\mathbf{p}$, and S fully describe the state of our system.

2.2. Simulation with Hamiltonian dynamics

To simulate the particle's trajectory across the surface, we start with its associated Hamiltonian equations. Given scalars $m > 0$ and $g > 0$, we define

$$H(\mathbf{q}, \mathbf{p}) \coloneqq U(\mathbf{q}) + K(\mathbf{p})$$
$$U(\mathbf{q}) \coloneqq m \cdot g \cdot h(\mathbf{q})$$
$$K(\mathbf{p}) \coloneqq \frac{\|\mathbf{p}\|^2}{2m}.$$

We respectively call m and g the *mass* and *gravity*, and both are hyperparameters of RMC. For convenience of notation,

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

let $\mathbf{g} \coloneqq [\mathbf{0}_d; g]$ where $\mathbf{0}_d$ is a d-length zero vector. The dynamics of the particle are governed by the following system of differential equations:

$$\frac{d\mathbf{q}}{dt} = \frac{\partial H}{\partial \mathbf{p}} = \frac{dK}{d\mathbf{p}} = \frac{\mathbf{p}}{m} \tag{2}$$

$$\frac{d\mathbf{p}}{dt} = -\frac{\partial H}{\partial \mathbf{q}} = -\frac{dU}{d\mathbf{q}} = -m\mathbf{g}. \tag{3}$$

In order to simulate the particle trajectory, HMC employs Euler's "leapfrog" integration, which requires an acceptance step to account for the case when the Hamiltonian H isn't conserved. In RMC's case, because Equations (2) and (3) don't involve what could be an arbitrarily complex $S(\boldsymbol{\theta})$, we can easily solve for the time-parametrized closed forms:

$$\mathbf{q}(t) = \mathbf{q}_{i-1} + \frac{\mathbf{p}_{i-1}}{m}t - \frac{\mathbf{g}}{2}t^2 \tag{4}$$

$$\mathbf{p}(t) = \mathbf{p}_{i-1} - m\mathbf{g}t. \tag{5}$$

We use Equations (4) and (5) to find the next candidate sample $\boldsymbol{\theta}_i$ by computing the earliest t' for which $C_S(\mathbf{q}(t')) = 0$. This search is done using a two-step process that increases t' in doubling increments until a collision, and then does a binary search within a time-bounded window to find the near-exact t' representing the point of collision while still satisfying $C_S(\mathbf{q}(t')) > 0$. [ALGORITHM TODO] outlines this in fuller detail.

2.3. Sampling at collisions

Once t' is had, we compute the position $\mathbf{q}_i \coloneqq \mathbf{q}(t')$ and momentum $\mathbf{p}_i \coloneqq \mathbf{p}(t')$, and the candidate sample is thus $\boldsymbol{\theta}(\mathbf{q}_i)$. The acceptance probability of the sample is defined in terms of *lateral momentum*: ignoring the height component, if the angle of the reflection is particularly sharp, we are more likely to reject it. The reasoning is that these "hard" bounces represent regions of low probability in $F(\boldsymbol{\theta})$. If

$$\mathbf{n} \coloneqq \frac{\nabla C_S(\mathbf{q}_i)}{\|\nabla C_S(\mathbf{q}_i)\|}$$

is the unit normal at $\mathbf{q}_i$, we can define the momentum after reflecting off the surface as

$$\overleftarrow{\mathbf{p}_i} \coloneqq \mathbf{p}_i - 2 \cdot \langle \mathbf{p}_i, \mathbf{n} \rangle \cdot \mathbf{n}. \tag{6}$$

Our acceptance probability is thus

$$\Pr\left(\text{ACCEPT} \mid \mathbf{p}_i, \overleftarrow{\mathbf{p}_i}\right) \coloneqq \frac{1}{2}\left(1 + \frac{\left\langle \boldsymbol{\theta}(\mathbf{p}_i), \boldsymbol{\theta}(\overleftarrow{\mathbf{p}_i}) \right\rangle}{\|\boldsymbol{\theta}(\mathbf{p}_i)\| \|\boldsymbol{\theta}(\overleftarrow{\mathbf{p}_i})\|}\right). \tag{7}$$

If accepted, then we set $\boldsymbol{\theta}_i \leftarrow \boldsymbol{\theta}(\mathbf{q}_i)$. Whether the sample is accepted or rejected, we still continue on with the simulation, starting from $(\mathbf{q}_i, \mathbf{p}_i)$.

2.4. Entropic dissipation

We modify Equation (6)

$$\overleftarrow{\mathbf{p}_i} \coloneqq \left(\mathbf{p}_i - 2 \cdot \langle \mathbf{p}_i, \mathbf{n} \rangle \cdot \mathbf{n}\right). \tag{8}$$

2.4.1. Refreshing momentum

2.5. Handling boundary constraints

2.6. Initialization

2.7. Full algorithm

3. Experiments

3.1. Distribution sampling

3.1.1. Single gaussian

3.1.2. Mixture of varied gaussians

3.1.3. Lopsided distributions

3.1.4. Graphical model with boundary constraints

3.2. Nonlinear optimization

3.3. Modeling task: Skin segmentation

4. Theoretical analysis & discussion

4.1. Potential extensions

5. Conclusion

References

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

A. You *can* have an appendix here.

You can have as much text here as you want. The main body must be at most 8 pages long. For the final version, one more page can be added. If you want, you can use an appendix like this one, even using the one-column format.