Overview

- Hamiltonian Monte Carlo (HMC) is a powerful Markov Chain Monte Carlo (MCMC) inference algorithm [1], that has desirable geometric properties which enable it to work efficiently with high dimensional continuous models.
- Probabilistic Programming Languages (PPLs) are not random programs, but are instead languages that enable the practitioner to focus on designing the probabilistic model, rather than the complicated inference algorithms to evaluate those models [2].
- Many models contain either a mixture of discrete and continuous parameters, or just purely discrete parameters. This creates discontinuities, which greatly affect the performance of HMC as it is a gradient based method.
- We formally construct a PPL to deal with these discontinuities, and discontinuities that arise via Branching, for particular classes of functions.
- The semantics of this PPL encompasses a broad range of First Order PPLs (FOPPLs) such as STAN and BUGS.
- Constructing such a language enables us to perform inference on these types models using an adapted version of HMC, called Discontinuous HMC [3].

Examples

Hidden Markov Model

\[ x_0 \sim \text{Categorical}(p_0) \]

\[ x_i | x_{i-1} \sim \text{Categorical}(p_{x_i}), i = 1, \ldots, T = 16 \]

\[ y_i | x_i \sim \mathcal{N}(\mu_{x_i}, \sigma_{x_i}) \]

(def hmm)

(defun query)

(let (points

  (vector 0.0 0.0 0.0 0.0 -0.005
    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
    0.0 0.0 0.0 0.0 0.0 0.0)

  (get points a)))

(defun get-init-params []

  (vector 0.1 0.5 0.4

  (vector 0.2 0.2 0.6

  (vector 0.7 0.15 0.15)))

(defun get-trans-params [x]

  (matrix [0.1 0.2 0.3

  0.2 0.3 0.1

  0.3 0.1 0.6]))

(defun get-obs-dist [x]

  (vector 0.1 0.0 0.0

  0.0 0.1 0.0

  0.0 0.0 1.0)

(defun hmm-step [x]

  (matrix [0.1 0.0 0.0

  0.0 0.1 0.0

  0.0 0.0 1.0])

  (observe obs)

  (step x)

(defun condif)

(defun query)

(let (x (sample (normal 0.0 1.0)))

  (if (> x 0)

    (observe (normal 1.0 1.0) 1.0)

    (observe (normal -1.0 1.0) 1.0)))

Discontinuous Hamiltonian Monte Carlo

Algorithm 1 Discontinuous HMC

1: function DHMC(x = [x_1, x_2], p = [p_1, p_2], \epsilon, \phi = Permuate(J))
2:   p_1 \leftarrow p_1 - \frac{1}{2} \nabla_{x_1} \log U(x_1)
3:   x_1 \leftarrow x_1 + \Delta t \nabla_{p_1} K(p_1)
4:   for j in J do
5:     \Delta t \leftarrow x_j^* + \epsilon \phi_j \frac{\pi_j}{\sqrt{\pi_j}}
6:     \Delta U \leftarrow U(x_j^*) - U(x_j)
7:     if \Delta U > \Delta \epsilon then
8:       \Delta U \leftarrow \Delta U_t
9:       p_j \leftarrow p_j - \Delta U_t
10:   end if
11: end for
12: x \leftarrow x + \sum_{i} \approx \epsilon \phi_i
13: return x

Branching

\[ x \sim \mathcal{N}(0, 1) \]

\[ y | x \sim \begin{cases} 
  \mathcal{N}(-1, 1) & \text{if } x \leq 0 \\
  \mathcal{N}(1, 1) & \text{if } x > 0 
\end{cases} \]

Branching

References

