Log-concave Sampling (Part 1)

Marios Papachristou

GeomScale, NTUA

June 1, 2020

Mentors: A. Chalkis (NKUA), V. Fisikopoulos (NKUA), E. Tsigaridas (INRIA)
Homepage: `https://github.com/GeomScale/volume_approximation`

About today's talk / tutorial

Today's talk will concentrate on
Sampling from high-dimensional log-concave densities

1. Introduction to log-concave sampling.
2. ODE Solvers.
3. Boundary Oracles.

Google Summer of Code 2020

The current GSoC project aims to provide implementations (and theoretical insights) to log-concave sampling problems for the GeomScale project.

Milestones

1. Milestone I (ODE Solvers)
    - Implement ODE solvers (Euler, Runge-Kutta, Collocation, etc.)
    - Efficiently address boundary oracles
2. Milestone II (Samplers)
    - Implement samplers (HMC, Langevin etc.).
    - Provide theoretical guarantees on truncated settings.
3. Milestone III (R bindings)
    - Port C++ functionality of

Today's talk will mostly concentrate on **Milestone I**.

Basics

Our project involves taking samples from distributions with probability density functions of the form

$$\pi(x) \propto \exp(-f(x)) \qquad x \in K$$

where K is either: (a) $\mathbb{R}^d$, or (b) a convex body, and f is a convex function that is L-smooth and m-strongly convex.

Convex Functions I

A domain K is convex iff (if and only if) for all $x, y \in K$ it holds that for all $t \in [0, 1]$

$$tx + (1 - t)y \in K$$

The domain K is a convex body iff it is convex, closed and bounded. A function $f : K \to \mathbb{R}$ is convex iff for all $x, y \in K$ we have that for all $t \in [0, 1]$

$$f(tx + (1 - t)y) \leq tf(x) + (1 - t)f(y)$$

Convex functions have some very nice properties, and their use is widespread in optimization.

Convex Functions II

If the function is twice differentiable with gradient ∇f and Hessian matrix $\nabla^2 f$ then

- We say that f is L-smooth iff $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$ or $\nabla^2 f(x) \preceq L \cdot I_d$.
- We say that f is m-strongly convex iff

  $$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{m}{2}\|x - y\|^2$$

  or $\nabla^2 f(x) \succeq m \cdot I_d$.
- We define the **condition number** of f to be the ratio of max/min eigenvalues of the Hessian, that is $\kappa = L/m$.

Log-concave Sampling I

Our goal is sampling from $\pi(x) \propto \exp(-f(x))$.

Directly sampling from $\pi(x)$ is very difficult since one has to account for the normalization constant $\int_K \exp(-f(x))dx$ which is in general **intractable**.

Idea: The distribution $\pi(x)$ can be thought as the stationary measure of a Markov Chain that is $\pi(x) = \lim_{k\to\infty} \pi_k(x)$.

Log-concave Sampling II

One of the first algorithms to do it is the Metropolis-Hastings Algorithm. The general idea of Metropolis Hastings is

1. Assume that you are at a state x
2. Perform a transition to a new nearby state y and make a proposal
3. Accept the proposal to move to y with probability (Metropolis Filter)

$$\min\left\{1, \frac{a(x,y)\pi(y)}{a(y,x)\pi(x)}\right\}$$

   where a is a transition probability function.

It can be shown analytically that the above process converges to a stationary distribution $\pi(x)$.
Intuition when $a(x,y) = a(y,x)$: The sampler has incentive to move towards higher-density areas (but lower density areas are also allowed)

Sampling in a continuous setting

Hamiltonian Monte Carlo I

The state-space is continuous and the samples can be proposed via solving Hamilton's equations for a particle with position x and velocity v under a conservative potential $f(x)$ that applies a force $-\nabla f(x)$.
The Hamiltonian of the particle is defined as

$$H(x, v) = \frac{1}{2}\|v\|^2 + f(x)$$

and Hamilton's equations simulate the particle's behaviour

$$\begin{aligned} \dot{x} &= v \\ \dot{v} &= -\nabla f(x) \end{aligned}$$

Hamiltonian Monte Carlo II

We start by choosing a direction $v \sim \mathcal{N}(0, I_d)$ and simulate one/many steps of the ODE arriving at a proposal $(\tilde{x}, \tilde{v})$.

The Metropolis Filter in this case for a proposal $(\tilde{x}, \tilde{v})$ given a state (x, v) is $\min \{1, \exp(H(\tilde{x}, \tilde{v}) - H(x, v)\}$.

Ideally note that $\dot{H} = \langle \nabla_{x,v} H, (\dot{x}, \dot{v}) \rangle = \langle (v, \nabla f(x)), (-\nabla f(x), v) \rangle = 0$ and hence the Metropolis probability is always 1.

However, the ODE must be discretized and the **discretization error** makes the decision non-trivial.

Hamiltonian Monte Carlo III

Finally, the ODE admits a separable stationary measure proportional to

$$\pi(x, v) \propto \exp(-H(x, v))$$

The marginal density with respect to x is therefore

$$\pi(x) = \int_{\mathbb{R}^d} \pi(x, v) dv \propto \exp(-f(x))$$

Langevin Dynamics I

Another method for sampling is via solving the Langevin Stochastic Differential Equation which is the Newton's Second Law together with a Brownian Motion W.

$$\begin{aligned} \dot{x} &= v \\ \dot{v} &= -\gamma v - \nabla f(x) + \sqrt{2\epsilon\gamma}\dot{W} \end{aligned}$$

where $\dot{W}$ is the derivative of the Brownian motion, that is $dW \sim \mathcal{N}(0, dt)$. Under mild conditions the SDE accepts a stationary measure proportional to $\exp\left(-\frac{1}{2}\|v\|^2 - f(x)\right)$. The parameters γ (damping factor), ϵ determine the nature of the dynamics

1. when $\gamma > 1$ the system is overdamped (OLD equation)
2. when $\gamma < 1$ the system is underdamped (ULD equation)
3. when $\gamma = 1$ the system is critically damped

Langevin Dynamics II

Of particular interest is the ULD equation

$$\begin{aligned}\dot{x} &= v\\ \dot{v} &= -2v - u\nabla f(x) + 2\sqrt{u}\dot{W}\\ u &= 1/L\end{aligned}$$

which is widely used in log-concave sampling. (for more information see [LST20, LSV18, GP14]).

Sampling Applications

1. Integral Calculation (Volume Calculation etc.)
2. Control systems
3. Generative Adversarial Networks
4. Logistic Regression
5. Financial Modeling
6. Probabilistic Graphical Models

ODE Solvers

General Setting I

Our goal is to solve an ODE of the form

$$\dot{x}(t) = F(x(t), t) \qquad x(0) = x_0$$

Theorem

If F is Lipschitz continuous in x and continuous in t then the above has a unique solution $x(t) = \phi(t)$

The HMC equations have $F(x(t), v(t), t) = \begin{pmatrix} v(t) \\ -\nabla f(x(t)) \end{pmatrix}$ which is $\max\{1, L\}$-Lipschitz (continuous) since f is L-smooth and $v(t)$ is 1-Lipschitz

General Setting II

In a discrete setting the equation is solved at discrete timesteps $t_n = t_{n-1} + \eta$ where $\eta > 0$ is the step-size.

Let x_n denote the solution provided by the discrete solver at step n and $\phi_n = \phi(t_n)$ be the "ideal point" at step n

We define the **error** ϵ_n to be

$$\epsilon_n = x_n - \phi_n$$

The dynamical behaviour of $\{\epsilon_n\}_{n \geq 0}$ provides inshights regarding the methods' accuracy.

Euler Solver

The Euler Solver is the simplest one

$$\begin{aligned} t_n &= t_{n-1} + \eta \\ x_n &= x_{n-1} + \eta F(x_n, t_n) \end{aligned}$$

It can be proven that

$$\|\epsilon_n\| \leq \frac{\eta m}{2L} \left(\exp(t_n - t_0) - 1 \right) = K(t_n) \cdot \eta$$

Runge-Kutta Methods I

The idea is to "break" every step of size η to smaller sub-steps and interpolate to find the next position. Each Runge-Kutta (RK) method is given by the following table (Butcher Tableau)

$$\begin{array}{c|ccccc}
0 & & & & \\
c_2 & a_{21} & & & \\
c_3 & a_{31} & a_{32} & & \\
\vdots & & & & \\
c_m & a_{m1} & \dots & a_{m,m-1} & \\
\hline
 & b_1 & \dots & b_{m-1} & b_m
\end{array}$$

Table: Butcher's Tableau

where $\sum_{j=1}^{m} b_j = 1$ and $c_j = \sum_{r=1}^{j-1} a_{jr}$

Runge-Kutta Methods II

The RK iteration proceeds in sub-steps where

$$
\begin{aligned}
t_n^j &= t_{n-1} + c_j \eta \qquad j \in [m] \\
k_j &= F\left(\sum_{r=1}^{j-1} a_{j,r} k_r, t_n^j \right) \\
x_{n+1} &= \sum_{j=1}^{m} b_j k_j \\
t_{n+1} &= t_n + \eta
\end{aligned}
$$

The global truncation error $\|\epsilon_n\|$ is $O(\eta^m)$.

Collocation Methods I

The collocation method assumes that the solution is locally approximated as

$$x(t) = \sum_{j=0}^{m} a_j \phi_j(t) \tag{1}$$

where $\{\phi_j\}_{0 \le j \le m}$ are basis functions (e.g. polynomials). The constants $\{a_j\}_{0 \le j \le m}$ are found by interpolation on the derivative of x at points given by $t_{n+1}^{j} = t_n + c_j \eta$ as in the RK methods.

Collocation Methods II

The system of equations for the interpolation is given by

$$
\begin{aligned}
t_{n+1}^{j} &= t_n + c_j \eta \\
a_{n+1}^{0} &= x_n \\
\dot{x}_{n+1}^{j} &= F(x_{n+1}^{j})(c_j - c_{j-1})\eta \\
\dot{x}_{n+1}^{j} &= \sum_{j=0}^{m} a_{n+1}^{j} \dot{\phi}_{n+1}^{j}(t_{n+1}^{j})
\end{aligned}
$$

Alternatively one solves an $m \times m$ system of the form $\dot{\Phi}_{n+1} a_{n+1} = \dot{X}_{n+1}$. If the matrix of the basis derivatives is not full-rank then a solution to $\min_{a_{n+1}} \frac{1}{2} \|\dot{\Phi}_{n+1} a_{n+1} - \dot{X}_{n+1}\|_2^2$ is seeked (e.g. using SVD).

Collocation Methods III

As choices for bases one has many choices, some of which being

1. Polynomials $\phi_n^j(t) = (t - t_n)^j$
2. Lagrange polynomials $\phi_n^j(t) = \prod_{r \neq j} \frac{t - t_r}{t_j - t_r}$
3. Rational functions $\phi_n^j(t) = \frac{p_n^j(t)}{q_n^j(t)}$ with $q_n^j(t) \neq 0$ in the ROIs.

Boundary Oracles

Boundary Conditions I

In HMC the domain of (x, v) is $K \times \mathbb{R}^d \subseteq \mathbb{R}^d \times \mathbb{R}^d$. Where $K \neq \mathbb{R}^d$ one has to account for **boundary conditions** for the position x.

There are three main types of boundary conditions

1. Neumann Conditions (Boundary Reflections) where $\frac{\partial x}{\partial n} = 0$
2. Dirichlet Conditions $x = g$
3. Robin (mixed) Conditions $a\frac{\partial x}{\partial n} + g = 0$

where the domain of a, g is the boundary ∂K.
It has been proven [PP14] that HMC admits boundary conditions equivalent to the **Neumann Conditions**.

Boundary Conditions II

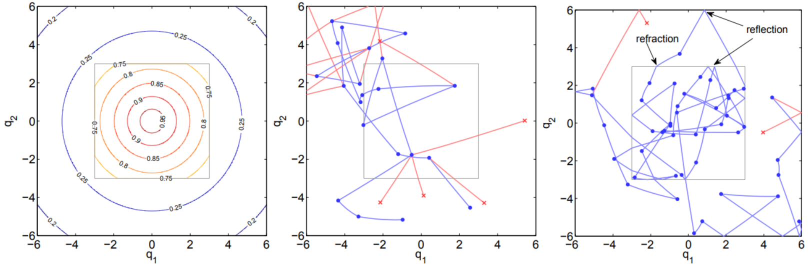


Figure: Baseline and Reflective HMC. Taken from [AD15].

The Reflection Operator I

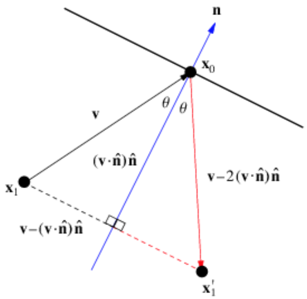
A point x reflects at the boundary point $\tilde{x}$ with normal n.

We define the reflection operator refl such that

$$\mathrm{refl}(x) = -2(a^T n)n + a + \tilde{x}$$

where $a = \tilde{x} - x$ is the ray between the initial and the boundary points. Note that in general $\mathrm{refl}(x)$ may not lie in K. We compose the reflection operator k times such that $\mathrm{refl}^k(x) = \mathrm{refl} \circ \cdots \circ \mathrm{refl}(x) \in K$. In our setting we assume that at each step the proposal point cannot reflect more than $\ell \in \mathbb{N}^*$ times.

The Reflection Operator II



$\mathbf{n}$
$\mathbf{x}_0$
θ
θ
$\mathbf{v}$
$(\mathbf{v}\cdot\hat{\mathbf{n}})\hat{\mathbf{n}}$
$\mathbf{v}-2(\mathbf{v}\cdot\hat{\mathbf{n}})\hat{\mathbf{n}}$
$\mathbf{x}_1$
$\mathbf{v}-(\mathbf{v}\cdot\hat{\mathbf{n}})\hat{\mathbf{n}}$
$\mathbf{x}_1'$

Figure: Reflection Illustration. x_1' is the reflection of x_1 about x_0 with normal n. Source: `https://mathworld.wolfram.com/Reflection.html`

Computing Intersections with ∂K I

Of particular interest is the computation of the intersection of an (implicit) curve between a point x inside the convex body K and a proposal $\tilde{x} \notin K$.

Case 1. The curve is a *line segment* and K is a convex polytope. We parametrize the line segment between x and $\tilde{x}$ with $\gamma(t) = tx + (1-t)\tilde{x}$ where $t \in [0,1]$. We seek $t_u = \sup\{t \in [0,1] | \gamma(t) \in \partial K\}$ and $u = \gamma(t_u)$ as the solution to the boundary intersection problem.

We use the Cyrus-Beck [CB78] algorithm

Computing Intersections with ∂K II

Let $z \in \partial K$ be known and let n represent the normal vector at z. We compute the quantity

$$n^T(\gamma(t) - z) \begin{cases} = 0 & \gamma(t) \in \partial K \\ < 0 & \gamma(t) \notin K \\ > 0 & \gamma(t) \in K \setminus \partial K \end{cases}$$

Solving the equation $n^T(\gamma(t) - z)$ for t we get

$$t = \frac{n^T(z - x)}{n^T(\tilde{x} - x)}$$

We compute the above for all the N normals of the polytope and keep the maximum value that lies in $[0, 1]$. The min value can also be kept in case we want the other intersetion point as well. Complexity is $O(Nd)$

Computing Intersections with ∂K III

Case 2. The curve has the form $\gamma(t) = \sum_{i=1}^{m} a_i \phi_i(t)$, $\{\phi_j\}_{j \in [m]}$ are basis functions, and K is a convex polytope.

We use the same procedure as above, however now we cannot solve directly for t. We, for example, can use the Newton-Raphson root finder to solve the transcendental equation.

$$t^{(r+1)} = t^{(r)} - \frac{\sum_{j \in [m]} (n^T a_j) \phi_j(t^{(r)}) - n^T z}{\sum_{j \in [m]} (n^T a_j) \dot{\phi}_j(t^{(r)})} \qquad r \in [R]$$

Complexity is $O(NdR)$ where R is the maximum number of iterations the NR solver must be called to find a root.
Problems. Convergence, Well-posedness (denominator getting too small)

Computing Intersections with ∂K IV

Case 3. The convex body K has the form $K = \{x \in \mathbb{R}^d | g(x) \le 0\}$ where $g(x) = \max_{1 \le i \le M} g_i(x)$ where $g_1, \dots, g_M$ are twice-differentiable convex functions that are μ-strongly-convex.

Examples. L_2 Balls, Spectrahedra etc.

Idea. Linearize the convex body around $x + h$

$$0 \ge g_i(x+h) \ge g_i(x) + \langle \nabla g_i(x), h \rangle + \frac{\mu \|h\|^2}{2}$$

The linearized convex polytope $P(x)$ around x is

$$J(x) h \le b$$

where $J(x)$ is the Jacobian matrix around x with entries $J_{ij}(x) = \frac{\partial g_i(x)}{\partial x_j}$ and b has entries $b_i = -g_i(x)$.

Computing Intersections with ∂K V

The linear approximation error is $O(\|h\|^2)$. A high-level algorithm proceeds as follows.

We are given a curve $\gamma(t)$ and a starting point $x_0 = \gamma(0)$, an accuracy $\epsilon > 0$, and a step counter i initialized at 0.

1. Find $P(x_i)$ around x_i and the intersection point of $\gamma(t)$ with $P(x_i)$ (see Case 1, Case 2). Let that point be $x_{i+1} = \gamma(t_{i+1})$
2. Calculate $g(x_{i+1}) = \max_{1 \le j \le M} g_j(x_{i+1})$. If $|g(x_{i+1)}| \le \epsilon$, output x_{i+1}, t_{i+1}, else repeat.

Next Talk(s)

Next talk(s) will be occupied with

1. Algorithmic Issues for the sampling problem (mixing time, bounds etc.).
2. Theoretical contributions to the problem.
3. Implementation details.

Thank you!

References I

Hadi Mohasel Afshar and Justin Domke.

Reflection, refraction, and hamiltonian monte carlo.

In *Advances in neural information processing systems*, pages 3007–3015, 2015.

Mike Cyrus and Jay Beck.

Generalized two-and three-dimensional clipping.

Computers & Graphics, 3(1):23–28, 1978.

Elena Gryazina and Boris Polyak.

Random sampling: Billiard walk algorithm.

European Journal of Operational Research, 238(2):497–504, 2014.

References II

Yin Tat Lee, Ruoqi Shen, and Kevin Tian.
Logsmooth gradient concentration and tighter runtimes for metropolized hamiltonian monte carlo.
arXiv preprint arXiv:2002.04121, 2020.

Yin Tat Lee, Zhao Song, and Santosh S Vempala.
Algorithmic theory of odes and sampling from well-conditioned log-concave densities.
arXiv preprint arXiv:1812.06243, 2018.

Ari Pakman and Liam Paninski.
Exact hamiltonian monte carlo for truncated multivariate gaussians.
Journal of Computational and Graphical Statistics, 23(2):518–542, 2014.