HAL open science

Truncated Log-concave Sampling for Convex Bodies with Reflective Hamiltonian Monte Carlo

Apostolos Chalkis, Vissarion Fisikopoulos, Marios Papachristou, Elias Tsigaridas

To cite this version:

Apostolos Chalkis, Vissarion Fisikopoulos, Marios Papachristou, Elias Tsigaridas. Truncated Log-concave Sampling for Convex Bodies with Reflective Hamiltonian Monte Carlo. ACM Transactions on Mathematical Software, 2023, 49 (2), pp.1-25. 10.1145/3589505 . hal-04222039

HAL Id: hal-04222039

https://inria.hal.science/hal-04222039v1

Submitted on 28 Sep 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Truncated Log-concave Sampling for Convex Bodies with Reflective Hamiltonian Monte Carlo*

APOSTOLOS CHALKIS, National Kapodistrian University of Athens, Greece and GeomScale org, Greece
VISSARION FISIKOPOULOS, National Kapodistrian University of Athens, Greece and GeomScale org, Greece
MARIOS PAPACHRISTOU, Cornell University, USA and GeomScale org, Greece
ELIAS TSIGARIDAS, Inria Paris and Sorbonne Université, France and GeomScale org, Greece

We introduce Reflective Hamiltonian Monte Carlo (ReHMC), an HMC-based algorithm to sample from a log-concave distribution restricted to a convex body. The random walk is based on incorporating reflections to the Hamiltonian dynamics such that the support of the target density is the convex body. We develop an efficient open-source implementation of ReHMC and perform an experimental study on various high-dimensional datasets. The experiments suggest that ReHMC outperforms Hit-and-Run and Coordinate-Hit-and-Run regarding the time it needs to produce an independent sample, introducing practical truncated sampling in thousands of dimensions.

CCS Concepts: • **Mathematics of computing → Statistical software**; **Mathematical software performance**; **Probability and statistics**; • **Theory of computation → Random walks and Markov chains**; *Computational geometry*.

Additional Key Words and Phrases: statistical software, truncated sampling, geometric random walks, experiments, mixing time

ACM Reference Format:
Apostolos Chalkis, Vissarion Fisikopoulos, Marios Papachristou, and Elias Tsigaridas. 2023. Truncated Log-concave Sampling for Convex Bodies with Reflective Hamiltonian Monte Carlo. *ACM Trans. Math. Softw.* 1, 1, Article 1 (January 2023), 26 pages. https://doi.org/10.1145/3589505

1 INTRODUCTION

A fascinating and fundamental computational problem is sampling from a high-dimensional log-concave density of the form $\pi(x) \propto e^{-f(x)}$, constrained (or truncated) in a convex body (e.g., polytope, spectrahedron) $K \subset \mathbb{R}^d$, where f is an L-smooth and m-strongly-convex function, using a Markov Chain Monte Carlo (MCMC) method. It appears regularly in various areas, like machine learning [12], finance [15], numerical analysis [23], optimal control [46, 48], Bayesian inference [36], and computational geometry [33]. Example problems include, but are not limited to, (constrained) Bayesian logistic regression, Bayesian mixture models, Gaussian sampling, learning-to-rank, and flux sampling from metabolic networks [47].

*Authors in alphabetical order. Correspondence to papachristoumarios@gmail.com.

Authors' addresses: Apostolos Chalkis, achalkis@di.uoa.gr, National Kapodistrian University of Athens, Greece and GeomScale org, Greece; Vissarion Fisikopoulos, vfisikop@di.uoa.gr, National Kapodistrian University of Athens, Greece and GeomScale org, Greece; Marios Papachristou, papachristoumarios@gmail.com, Cornell University, USA and GeomScale org, Greece; Elias Tsigaridas, elias.tsigaridas@inria.fr, Inria Paris and Sorbonne Université, France and GeomScale org, Greece.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
Manuscript submitted to ACM

A straightforward algorithm to sample from a truncated distribution is *Rejection-Sampling* or the *Acceptance-Rejection* (AR) method. Roughly speaking, its main idea is to sample from a bigger region that contains the support of our distribution and keep only the samples that fall in the support. However, AR is efficient only for low dimensional problems or special target distributions [9], for which we can guarantee a high acceptance ratio. MCMC algorithms have proven superior to AR in several computational problems involving sampling from log-concave distributions [22, 63, 83]. Considering special cases of log-concave distributions, e.g., the uniform or the exponential distribution, AR cannot scale beyond a few dimensions, while MCMC algorithms have been proven quite efficient [53, 65, 77]. Other sampling algorithms like Adaptive Rejection Sampling [39], Inverse Transform Sampling [75], or Slice Sampling [72] have never turned out to be efficient when the dimension exceeds a few dozen, even for sampling from a log-concave probability density. We consider the first-order method of Hamiltonian Monte Carlo (HMC) [5, 73, 92] to sample from a log-concave density $e^{-f(x)}$. The main reason for this choice is that HMC has the best mixing time for unconstrained log-concave distributions and a relatively small cost per step due to leapfrog's small cost per step. HMC simulates an imaginary particle moving in a conservative field where the roles of its dynamic and kinetic energy are played by the (negative) log-concave function $f(x)$ and its gradient $\nabla f(x)$.

Current research on MCMC algorithms focuses on both theoretical and practical aspects. There are efficient implementations, like `TensorFlow` [1], `stan` [16], and `pyro` [8], that provide MCMC sampling methods which in turn allow fitting complex Bayesian models. However, implementations for sampling from truncated distributions seem to be unavailable, with the exceptions of either special distributions such as Gaussians [78] or special convex bodies such as cubes [16]. Recently, a new sampling strategy that uses a Piecewise-Deterministic Markov Process has been proposed for Gaussians distributions [19]. Remarkable exceptions are also some implementations of the Hit-and-Run algorithm, such as the ones in the HOPS library [50]. On the other hand, our software is far more generic as it can sample from polytopes and general (non-linear) bodies. It does so by relying on a general oracle model to calculate the intersection of the sampler's trajectory with the (boundary of the) convex body. In specific categories of convex bodies (convex polytopes and spectrahedra) we use specialized (optimized) oracles to achieve better performance in theory and practice. For the case of spectrahedra or bodies whose boundaries are given by a set of convex functions, our implementation is the first one that supports such oracles. Experimental evidence suggests that our implementation outperforms all existing software regarding the time it takes to produce an independent sample in the simple case of polytopes. More precisely, there are cases (e.g., convex polytopes) for which our software is more efficient in terms of mixing rate, as well as other cases (e.g., spectrahedra) for which our software is the sole option for sampling.

We introduce Reflective Hamiltonian Monte Carlo (ReHMC), an HMC-based algorithm to sample from a truncated log-concave density using leapfrog dynamics with boundary reflections in a *convex body* with smooth boundary (in particular, it is enough to assume that the singularities have zero measure).

Figure 1 illustrates a sample generated by ReHMC and a single step of the random walk. We prove that ReHMC converges to the truncated target distribution even when the boundary is smooth. (Theorem 1). We also provide a heuristic to decide the step-size of leapfrog after a fast preprocessing phase before sampling.

Finally, we present an open-source, high-performance implementation in C++ that is (i) optimized for polytopes (feasible sets of linear programs) and spectrahedra (feasible sets of semi-definite programs) and (ii) supports general convex bodies (when the boundary oracle is available).

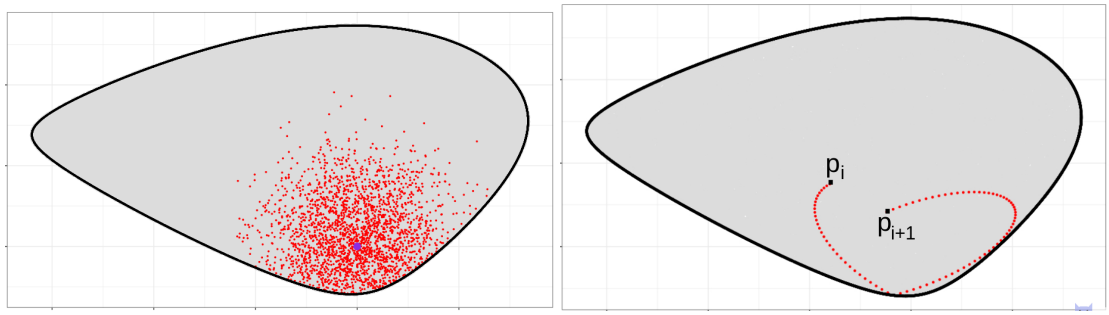

Fig. 1. Left we sample with Reflective Hamiltonian Monte Carlo (ReHMC) from the standard normal distribution $\mathcal{N}(0, I_2)$ truncated in a spectrahedron (the origin with purple color). Right a reflective leapfrog trajectory starting from p_i and ending at p_{i+1} that ReHMC computes in the i-th step for the same target distribution.

ReHMC can scale to thousands of dimensions on moderate hardware. We compare ReHMC with the Hit-and-Run (H&R) algorithm [64, 85, 88] that is commonly used in modern toolboxes, e.g., COBRA [4] and HOPS [50][1], on a wide variety of well-known convex bodies. In particular, we sample from (i) *polytopes*, namely cubes, (product of) simplices, Birkhoff, cross polytopes, and polytopes coming from structural biology; for the latter efficient sampling from a truncated log-concave density is of crucial importance [21, 47], (ii) *random spectrahedra*, and (iii) *general convex bodies*. We evaluate the performance by the rate that a mixed Markov Chain produces *independent samples* [38] using the Potential Scale Reduction Factor (PSRF) diagnostic [37]. ReHMC scales up to $\sim 10^3\times$ faster than H&R when we sample from a unit-covariance Gaussian density centered at the Chebyshev center of the body [11]. Our **code** extends the C++ library `volesti`. A **release** for this text is available at https://github.com/GeomScale/volesti/tree/v1.1.4.2 and the (post-processed) **data** we used to run the experiments are available in https://doi.org/10.5281/zenodo.7502138.

1.1 Notations and general setup

We sample from a distribution Π with PDF $\pi(x) \propto e^{-f(x)}$, where $f(x)$ is a convex function with support on $K \stackrel{\text{def}}{=} \text{supp}(\Pi) = \{x \in \mathbb{R}^d \mid \pi(x) > 0\} \subset \mathbb{R}^d$; K is a convex body with interior K^o and boundary ∂K. The function f has a minimizer at $x^* \in K^o$. The sandwiching ratio of K is $\gamma \stackrel{\text{def}}{=} \inf_{R>r>0} \{R/r \mid \mathbb{B}(x^*, r) \subseteq K \subseteq \mathbb{B}(x^*, R)\}$, where $\mathbb{B}(x^*, r)$ is the d-dimensional L_2 ball with radius r centered at x^*. We assume that f is twice differentiable in K, including the boundary ∂K, L-smooth, and m-strongly convex. The Hessian of f is $\nabla^2 f$, and has a smallest eigenvalue of m and a largest eigenvalue of L. The condition number of f is $\kappa \stackrel{\text{def}}{=} L/m$. $Q \succeq 0$ denotes that the matrix Q is positive semi-definite. We use $\Pi(A)$ to denote the measure of set A under the distribution Π whose density is π. The $\tilde{O}(\cdot)$ notation ignores polylogarithmic factors.

1.2 Related Work

First-order Unconstrained Methods. General first-order methods for sampling assume access to the (function of the) density and its gradient and include some well-known methods for sampling; Underdamped Langevin Dynamics (ULD) [58], the Metropolis-Adjusted Langevin Algorithm (MALA) [32] and HMC [27, 57] are among the most famous ones. The current work on these methods assumes distributions supported on $\mathbb{R}^d$, i.e. they do not constrain the domain of the samples. The recent bound for the mixing time of MALA is $\tilde{O}(\max\{\kappa d, \kappa^{1.5}\sqrt{d}\})$ [32] which was improved to

[1]We use the HOPS library as of 20 Jan 2021 (commit hash: 6387bfaeee71b589234c2239ef2958b5d145ed62) compiled with `MKL`.

$\tilde{O}(\kappa d)$ [57]. Also [84] proves ε-convergence time bound of $\tilde{O}(\kappa^{7/6}/\varepsilon^{1/3} + \kappa/\varepsilon^{2/3})$ in the 2-Wasserstein distance for ULD using the randomized midpoint method.

Constrained HMC. The work in [2] examines the HMC variant with reflection and refraction using a Leapfrog integrator. However, they prove the volume preservation property only for the case of an affine boundary (i.e., a convex polytope); this property is essential to establish the correctness of the algorithm. Moreover, the experimental part is restricted to low dimensions, and the code is not publicly available. The works of [20, 78] proposed HMC methods with boundary reflections where a reflective trajectory is planned, and it is rejected if the total number of reflections exceeds a certain threshold. However, in both papers, they consider only the case where the HMC flow can be computed exactly, e.g., when the Hamiltonian dynamics are relatively simple. They provide a specialized approach for sampling from a Gaussian density with mean μ and covariance matrix Σ truncated in a convex polytope or an ellipsoid. Moreover, in [20] they give an $O(\log d)$ mixing time for cubes with an $O(d)$ number of reflections per-step on expectation and also prove uniform ergodicity.

Another important family of methods for HMC-based sampling considers the inclusion of the local geometry in the Hamiltonian via barrier functions (log-barriers and sigmoid barriers) [5, 40, 59, 74, 94]; however, the barrier functions become ill-conditioned near the boundaries [93]. Finally, in this family of methods, we should also include the work of [85], which samples from a density of the form $e^{-f(x)-g(x)}$, where f and g are convex, but g is non-smooth [13, 14, 70, 80, 85]. It achieves constrained sampling using a non-smooth barrier function (such as the log barrier).

Practical performance & Software. The main paradigm in practice is Coordinate Directions Hit-and-Run (CH&R). Extended experiments [24, 34] show that H&R converges (in practice) after $\tilde{O}(d^2)$ steps. CH&R also converges after $\tilde{O}(d^2)$ steps in practice [34, 44] and when the truncation is given by a convex polytope, its cost per step is smaller than H&R's. This is the main reason why CH&R overshadowed, until recently, all other random walks in practical computations on polytopes. Recently asymptotic upper bounds have been proven for the mixing rate of CH&R [56, 71] although there is still a gap from the aforementioned empirical convergence rate. Considering software for truncated sampling, COBRA [4] provides CH&R with a rounding preprocessing step for uniform and exponential sampling from convex polytopes that appear as flux spaces of metabolic networks. HOPS [50] provides both H&R and CH&R for general distributions combined with the same rounding preprocess before sampling as in COBRA. HOPS implementation of CH&R outperforms COBRA [50]. Neither COBRA nor HOPS support sampling from more general domains than polytopes. In contrast, ReHMC provides a *specialized* implementation for sampling from polytopes and spectrahedra and a *generic* sampler for convex bodies.

Truncated Statistics. The study of truncated statistics has received a lot of attention in recent years [28, 29, 49, 76] with a focus on many "classical problems," such as linear regression and logistic regression. In this setting, the phenomenon of *output truncation* is studied, where the samples are filtered out concerning the values of the response variables. Truncation is attributed to poor measurements, data collection, and privacy concerns. These errors usually lead to biased models, i.e., models that replicate the biases of the data they have been trained on. Lastly, truncation has also been studied through the lens of more advanced generative modeling through GANs [12, 66].

2 THE ReHMC ALGORITHM

In this section, we present the Reflective Hamiltonian Monte Carlo (ReHMC) algorithm and its necessary ingredients.

2.1 Hamiltonian Dynamics

HMC simulates a moving particle to sample from a distribution Π. The state of the particle is a position vector x and a momentum vector v, with a Hamiltonian function

$$\mathcal{H}(x,v) \stackrel{\text{def}}{=} \underbrace{\tfrac{1}{2}\|v\|^2}_{\text{Kinetic Energy}} + \underbrace{f(x)}_{\text{Potential Energy}}, \tag{1}$$

where f is an L-smooth and m-strongly-convex function with condition number κ. The particle's movement evolves according to the Hamiltonian dynamics

$$\frac{dx}{dt} = +\frac{\partial \mathcal{H}}{\partial v} = +v, \quad \frac{dv}{dt} = -\frac{\partial \mathcal{H}}{\partial x} = -\nabla f(x), \tag{2}$$

that ideally preserves energy on the Hamiltonian $\mathcal{H}$, i.e., the total energy is preserved. If the particle is restricted to a convex body K, then it faces an *infinite potential barrier*. Hence, it *reflects* at ∂K and Hamiltonian dynamics embody the reflections [2, 41]. In the MCMC regime, this setting allows us to construct a Markov chain with joint stationary distribution proportional to

$$e^{-\|v\|^2/2} \cdot \underbrace{e^{-f(x)}\, \mathbf{1}\{x \in K\}}_{\text{Target density } \pi}, \tag{3}$$

where the x-marginal is the (truncated) target distribution π. The simulation of the *continuous* Markov chain, assuming that the sampler is positioned at x, is as follows: First, we draw an initial velocity $v \sim \mathcal{N}(0, I_d)$ and simulate the (reflective) Hamiltonian dynamics with initial conditions $x(0) = x$ and $v(0) = v$, for s units of time. Then, the new state $(x(s), v(s))$ is proposed, and we apply a Metropolis filter to preserve the stationary distribution of (3). That is, we perform a coin flip with bias $\min\left\{1, e^{-\mathcal{H}(x(s),v(s))+\mathcal{H}(x(0),v(0))}\right\}$. If the coin comes up heads, then the sampler moves to the proposed state $(x(s), v(s))$; otherwise, it remains at $(x(0), v(0))$. In the case of continuous dynamics, the Hamiltonian is *exactly* preserved, that is, $\dot{\mathcal{H}} = 0$, and the value of the filter equals 1, and the sampler always moves at the proposed position. For *discretized* dynamics that we use in computer simulations, the value of the filter is not one, and the proposed sample may be rejected.

2.2 Oracle Model

The oracle model has access to f, to its gradient ∇f, and to a *boundary oracle* that computes the intersection $\partial K \cap \{z \in \mathbb{R}^d | z = (1-t)x + ty, t \in [0,1], x \in K\}$, where K is a convex body given by a set of M inequalities, i.e., $K = \{x \in \mathbb{R}^d | g_i(x) \le 0, \forall i \in [M]\}$, where the g_i are convex [11, Ch. 11.1] and the normal vector is well-defined on ∂K. Suppose that the sampler is at x and has a velocity $\widehat{v}$. Then (the point) $x + \eta\widehat{v}$ may lie inside or outside K. If for all $i \in [M]$ we have that $g_i(x + \eta\widehat{v}) < 0$, then the new proposal lies within K, and we do not need to do any reflection. Otherwise, at least one inequality is violated, and thus, there exists a point where the trajectory from x to $x + \eta\widehat{v}$ intersects the boundary. So we seek the intersection parameter $t \in \operatorname{argmin}\{t_i \in [0,1], i \in [M] : g_i(x_i + \eta t_i \widehat{v}) = 0\}$. To find each t_i one has to solve $g_i(x_i + \eta t_i \widehat{v}) = 0$ subject to $t_i \in [0,1]$. We are interested in the cases where $t_i \in [0,1]$ so that the root lies in the line segment between x and $x + \eta\widehat{v}$. In the general case, the equations are solved numerically (e.g., use methods from [3] or binary search). Thus, we numerically solve (in the worst-case) $O(M)$ such equations. In specific cases, e.g., when K is a polytope or spectrahedron, the equations are univariate polynomials (linear in the case of polytopes), and we can solve them *efficiently* using dedicated algorithms; these cases cover the majority of applications. If the argmin

set is non-empty, then we pick an *active constraint* (i.e., the corresponding part of the boundary that the line segment intersects), say its index is i^*, and we perform the reflection operation with the respective (unit-length) normal, which is

$$n_{i^*} = \frac{\nabla g_{i^*}(x + \eta t_{i^*}\widehat{v})}{\|\nabla g_{i^*}(x + \eta t_{i^*}\widehat{v})\|}.$$

Our generic sampler exploits a binary search to compute the intersection of a line segment and ∂K. We can also use exponential search in this case, though its performance would be similar to binary search. In the next Section, we present efficient specialized boundary oracles for when K is a polytope or spectrahedron.

2.3 Specialized Boundary Oracles

We focus on efficient realizations of the boundary oracle for polytopes and spectrahedra. We start with *polytopes* where we follow the *Cyrus-Beck* algorithm [25]. Namely, if K has M facets and its representation is

$$K = \{x \in \mathbb{R}^d \mid Ax \leq b, \text{ where } A \in \mathbb{R}^{M\times d}, b \in \mathbb{R}^M, \text{and } \|a_i\| = 1\},$$

then the intersection point of the trajectory with the boundary of K corresponds to the smallest $t_i \in [0, 1]$ such that $a_i^\top((1 - t_i)x + t_i y) = b_i$. The computation of each t_i takes $O(d)$ operations, for $1 \leq i \leq M$. The Leapfrog integrator applies the reflection operator to $\widehat{v}$ at most ℓ times to obtain a new position $\tilde{x}' \in K$. Naïve complexity bounds yield that each reflection costs $O(Md)$, and thus the per step cost is $O(Md\ell)$. Our implementation of ReHMC —after a preprocessing— performs the first reflection in $O(Md)$ operations and each one of the subsequent $\ell - 1$ reflections costs $O(M)$ operations. Moreover, the integrator runs for w steps (w is the walk length parameter) before proposing the new position. Thus, the amortized per-step complexity bound becomes $O(M(d + \ell)w)$. The preprocessing step involves the computations of all inner products $a_i^\top a_j$ between the normal vectors of the facets; it takes $M^2 d$ operations. Now let $\widehat{v}_j$ be the velocity and $\tilde{x}_j,\ 1 \leq j \leq \ell$ the position before each reflection during a single step, with $\widehat{v}_1 = \widehat{v}$ and $\tilde{x}_1 = x$. During the computations of the first reflection, we store all the values of the inner products $a_i^\top \tilde{x}_1$ and $a_i^\top \widehat{v}_1$. For $j > 1$, the intersection time with ∂K corresponds to the smallest positive solution of the following system of linear equations,

$$a_i^\top((1 - t_j)\tilde{x}_j + t_j(\tilde{x}_j + \eta\widehat{v}_j)) = b_i,$$

where $t_j \in [0, 1]$, $\tilde{x}_j = \tilde{x}_{j-1} + \eta\widehat{v}_{j-1}$, $\widehat{v}_j = \widehat{v}_{j-1} - 2(\widehat{v}_{j-1}^\top a')a'$, a' is the normal vector of the facet that the trajectory hits at reflection $j - 1$, and t_{j-1} the solution of the reflection $j - 1$. We solve all the M equations in $O(1)$ operations based on our bookkeeping from the previous reflection and the preprocessing. When all the equations are infeasible we set $\tilde{x}' = \tilde{x}_j + \eta\widehat{v}_j$.

For *spectrahedra*, we use the oracle presented in [17], where it is used in the sampling process. Spectrahedra are the feasible sets of Semi-definite Programs (SDPs) [11]. We assume that the spectrahedron K is defined by a linear matrix inequality (LMI) of the form

$$K = \left\{x \in \mathbb{R}^d \mid F(x) = A_0 + A_1 x_1 + \ldots A_d x_d \succeq 0\right\},$$

where A_i are symmetric matrices in $\mathbb{R}^{M\times M}$. To find the intersection of a trajectory with the boundary of K, we seek the smallest $t \in [0, 1]$ such that $F(x + tv) \succeq 0$, for a point $x \in K$ and a direction $v \in \mathbb{R}^d$. This inequality yields a *polynomial eigenvalue problem* (PEP) of the form $P(\lambda)x = 0$ where $P(\lambda)$ is a univariate polynomial whose coefficients are $M \times M$ matrices. In our case, the PEP is $F(x) + t(F(v) - I_d) \succeq 0$. Reference [17, Lemma 2.1] gives an algorithm that finds the eigenvalues and eigenvectors of a PEP up to accuracy $\epsilon_P = 2^{-L_P}$ and runs in $\tilde{O}_B(M^{\omega+3}L_P)$ bit complexity (ignoring polylog factors) with $L_P = \Omega(M^3\tau_P)$ and τ_P being the bitsize of the matrix elements (for matrices with integer elements)

and has an arithmetic complexity of $\tilde{O}\left(M^{\omega} + M\log(1/\epsilon_P)\right)$. We pick the smallest eigenvalue that lies in $[0,1]$ as our solution from the set of the returned eigenvalues.

2.4 Discretization of the Dynamics

We use the symplectic method of leapfrog integration to discretize the dynamics of (2). The discretization of reflective Hamiltonian Dynamics updates the initial state (x, v) at time t to the state $(\tilde{x}', \tilde{v}')$ at time $t + \eta$ with the leapfrog integrator. The latter performs a *velocity half-update* and a *position update* as follows

$$\widehat{v} = v - \eta\nabla f(x)/2 \;\text{ and }\; \tilde{x} = x + \eta\widehat{v}. \tag{4}$$

If the new position $\tilde{x}$ is not in K we calculate the intersection of the line supporting $\tilde{x}$ and ∂K, and then perform a reflection with respect to the normal defined on the intersection point with ∂K. For this, we assume that locally the trajectory of HMC is the segment $(1-t)x + t\tilde{x}$ that intersects ∂K at a point with normal vector n_{i^*}. We reflect the velocity, $\mathrm{refl}_v(\widehat{v}) = -2(\widehat{v}^{\top} n_{i^*})n_{i^*} + \widehat{v}$, and then the position, $\mathrm{refl}_x(\tilde{x}) = \eta \cdot \mathrm{refl}(\widehat{v}) + x$. We apply the reflection operator sufficiently many times until we obtain a position inside K; let the corresponding state be $(\widehat{v}', \tilde{x}')$. Finally, we update the velocity,

$$\tilde{v}' = \widehat{v}' - \eta\nabla f(\tilde{x}')/2, \tag{5}$$

to get the final state $(\tilde{x}', \tilde{v}')$. Finally, we apply the Metropolis filter to transition from (x, v) to $(\tilde{x}', \tilde{v}')$. We can run the integrator for w steps before proposing the new state. The parameter w is called the *walk length*. The pseudocode of ReHMC can appear in Algorithm 1. We prove the following (see Appendix B.1 for the full proof):

Theorem 1. *For a smoothly differentiable negative log-density f (where $\pi(x) \propto \exp(-f(x))$), the discretized reflective Hamiltonian Dynamics are volume-preserving and time-reversible.*

In practice, before sampling, we perform a *burn-in* phase. Then, we pick the last point as a (warm) starting point for sampling. We exploit the steps we perform in the burn-in phase to learn a practical value for the leapfrog step size. We compute a sequence η_t of step sizes that converge to a value $\bar{\eta}$ in the long run; the latter is the step size we use for sampling (i.e., after the burn-in phase). In particular, we use the following online rule: We start from some η_0 which we iteratively divide with the sample average number of reflections we have seen so far. More formally, let $\ell_1, \ell_2, \ldots, \ell_t$ be the reflections observed until time t, where we count the 0-th reflection (i.e. $\ell_i = 1 +$ number of reflections at time i). Then, the step-size at time $t+1$ is

$$\eta_{t+1} = \frac{\eta_0}{\left(\frac{1}{t}\sum_{j=1}^{t}\ell_j\right)}.$$

We freeze the step size after burn-in. Figure 2 contains empirical evidence of the behaviour of $\{\eta_t\}_{t\in\mathbb{N}}$ during the burn-in phase on experiments with various polytopes. Observe that during the burn-in phase, the process $\{\eta_t\}$ is ergodic and converges to a value $\bar{\eta}$, which we use during the sampling phase after a small number of iterations. An interesting feature addition is to try to connect the step-size of the integrator with the proposition of the accepted sample so far.

2.5 Warm Start

The notion of a good starting distribution is crucial for the fast mixing of the algorithm both from a theoretical and a practical viewpoint; see, e.g., [51, 54, 57, 62] and the references therein. For this reason, it was introduced the concept of a warm start Π_0 with PDF π_0. This is a distribution that provides samples to an MCMC algorithm to start sampling

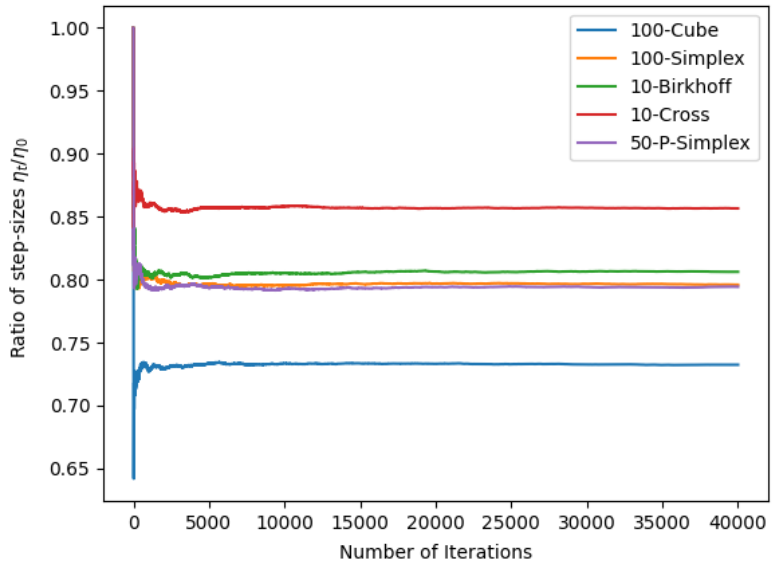

Fig. 2. Adaptive step size rule behaviour to sample from $\pi(x) \propto e^{-2\|x-x_c\|^2/R_c^2}$, where (x_c, R_c) are the Chebyshev center and radius; $\eta_0 = \frac{R_c}{10}$.

from a target distribution Π with PDF π. Formally, we say that Π_0 is β-warm-start with respect to Π if and only if

$$\sup_{x \in K} \frac{d\Pi_0}{d\Pi} = \sup_{x \in K} \frac{\pi_0(x)}{\pi(x)} \leq \beta.$$

The classical result of [62] states that the mixing time is between $1/\Phi$ and $\log \beta/\Phi^2$, where β is the *warmness* of the starting distribution, and the blocking conductance frameworks of [54], and [57] obtain mixing time bounds with a doubly logarithmic dependency on the warmness β. Therefore, a good starting distribution is a distribution for which β is small. In the untruncated case of sampling from a log-concave density $\pi(x) \propto e^{-f(x)}$ where the Hessian of f has condition number κ, a good starting distribution had $\beta = \kappa^{d/2}$. In our case, Theorem 2 shows that β is similarly $O((\kappa\gamma)^d)$.

Specifically, for ReHMC, we choose the truncated Gaussian

$$\pi_0 = \mathcal{N}_K\left(x^*, \frac{1}{L} I_d\right), \tag{6}$$

as an initial distribution and prove that

Theorem 2. *The distribution $\pi_0 = \mathcal{N}_K\,(x^*, 1/LI_d)$ is an $O(\gamma^d \kappa^d)$-warm start wrt. π, where $x^* = \arg\min_{x \in K} f(x)$, and $\gamma = \inf_{R > r > 0} \{R/r \mid \mathbb{B}(x^*, r) \subseteq K \subseteq \mathbb{B}(x^*, R)\}$ is the sandwiching ratio of K.*

The above warm start has the following advantages: First, since f is convex and we have access to ∇f, we can find the minimizer using any first-order method, such as projected gradient descent, mirror descent or interior point

	100-Cube	100-Simplex	100-S-Cube	10-Birkhoff	10-Cross	50-P-Simplex	e-coli	iAB-RBC-283	iAT-PLT-636
d (dimension)	100	100	100	81	10	100	25	130	290
ψ_{me}	1	1.0506	100	1.8	1	1.4	107.53	$1.22 \cdot 10^6$	117.1559
ReHMC									
w (walk length)	91	11	91	65	4	21	23	85	225
N_{ess}	24353	14666	8740	50309	50491	21076	15060	10	9010
PSRF	1.001	1.004	1.001	1.002	1.001	1.003	1.001	1.102	1.002
t_{is} (us)	1551	**454**	**5063**	**396**	**67**	**502**	**228**	$\mathbf{10^7}$	**698320**
Avg. Reflections ($\bar{\ell}$)	0.229	6.13	0.228	1.943	0.719	4.296	0.001	23.9943	0.0
Step size (of leapfrog)	0.008	0.001	0.008	0.004	0.058	0.0018	0.041	0.0003	0.003
H&R-HOPS									
w (walk length)	91	81	81	81	10	91	17	†	141
N_{ess}	2799	164	629	665	16886	214	24	†	467
PSRF	1.003	1.020	1.006	1.006	1.001	1.016	1.178	†	1.011
t_{is} (us)	1608	20632	6315	3496	1301	16046	310060	†	$16 \cdot 10^6$
CH&R-HOPS									
w (walk length)	91	91	91	81	10	91	17	†	197
N_{ess}	3225	293	71	4744	31734	17	11	†	453
PSRF	1.002	1.022	1.016	1.001	1.000	($\star$) 1.520	1.102	†	1.005
t_{is} (us)	**396**	4061	20645	10609	351	194182	284494	†	$7.24 \cdot 10^6$

Table 1. Experimental Results for sampling from $\pi(x) \propto e^{\|x-x_c\|^2/2}$ using ReHMC, Hit-and-Run from HOPS [50] (H&R-HOPS) and Coordinate-Hit-and-Run from HOPS (CH&R-HOPS). The HOPS library uses H&R (and CH&R) together with an initial rounding procedure in order to sample (only) from a convex polytope. The quantity ψ_{me} denotes the ratio of the maximum over the minimum axis lengths of the maximum-volume inscribed ellipsoid of K. A value of $\psi_{\text{me}} \approx 1$ indicates that the body is in John's position [52]. The † symbol denotes a failed experiment where the sampler repeatedly escaped K, and the $\star$ symbol denotes failure to mix according to the PSRF ≤ 1.2 criterion. In each case, the most efficient sampler is highlighted (lower is better). d is the dimension, w is the walk length, N_{ess} the min ESS, $\bar{\ell}$ is the average number of reflections and t_{is} is the time per independent sample. For ReHMC we also report the step size of leapfrog after burn-in (Section 3).

methods. Moreover, for practical purposes, we can use existing samplers available within our software to sample from truncated Gaussians, such as Hit-and-Run.

3 IMPLEMENTATION AND EXPERIMENTS

3.1 Implementation

We provide an open-source, scalable C++ implementation of the ReHMC algorithm for general densities with access to f and ∇f, that works on multiple OS. The software employs `eigen` [42] for linear algebra, `Spectra` for computing the eigenvalues in the PEP problems, and Intel's `MKL`[43] for high-performance[2]. It supports specialized samplers for H-polytopes and spectrahedra, as well as a generic sampler for convex bodies. Polytopes (feasible regions of linear

[2]Eigen's interface provides plug-and-play functionality with `MKL`.

Algorithm 1 ReHMC Algorithm.

```
procedure ReHMC(η, f, K, N_samples, w)
    k ← 0
    x* ← Minimize(f, K)
    Draw x_0 ~ N_K(x*, L^{-1} I_d)
    while k ≤ N_samples do
        Draw v_k ~ N(0, I_d)
        (x̃_k, ṽ_k) ← Walk(f, η, x'_k, v'_k, w)
        Draw u ~ U[0, 1]
        if u ≤ min{1, exp(H(x_k, v_k) − H(x̃'_k, ṽ'_k))} then
            x_{k+1} ← x̃'_k
        else
            x_{k+1} ← x_k
        end if
        k ← k + 1
    end while
    return {x_k}_{1≤k≤N_samples}
end procedure

procedure Walk(f, η, x, v, w)
    for 1 ≤ i ≤ w do
        (x, v) ←Leapfrog (f, η, x, v)
    end for
end procedure

procedure Leapfrog(f, η, x, v)
    v̂ ← v − (η/2) ∇f(x)
    x̃ ← x + η v̂
    if x̃ ∈ K then
        x̃' ← x̃
        v̂' ← v̂
    else
        (x̃', v̂') ← Reflect(K, x̃, x, v̂)
    end if
    ṽ' ← v̂' − (η/2) ∇f(x̃')
    return (x̃', ṽ')
end procedure

procedure Reflect(K, x̃, x, v̂)
    a = x̃ − x
    u is the point of intersection of x + ta and ∂K
    n is the normal vector at u ∈ ∂K
    x̃' ← −2(a^⊤ n)n + a + x̃
    v̂' ← −2(ṽ^⊤ n)n + v̂
    if x̃' ∉ K then
        Reflect(K, x̃', x̃, v̂')
    else
        return (x̃', v̂')
    end if
end procedure
```

programs) and spectrahedra (feasible regions of semi-definite programs) are important in convex optimization. We use the boundary oracles of Section 2.3 to compute the intersection of the leapfrog trajectory with the corresponding

Convex Body	Dimension	Distribution	w	N_{ess}	t_{is} (us)	PSRF	$\bar{\ell}$
1000-Cube	1000	Gaussian	300	30903	$3.9 \cdot 10^5$	1.001	0.2
1000-Simplex	1000	Gaussian	300	3257	$1.2 \cdot 10^6$	1.010	20
33-Birkhoff	1024	Gaussian	300	25470	$2.5 \cdot 10^5$	1.004	4
Recon1	931	Gaussian	187	409	$1.2 \cdot 10^8$	1.021	0.0
Spectra-200-15	200	Gaussian	200	11917	$4.8 \cdot 10^4$	1.005	1
Spectra-400-20	400	Gaussian	400	11896	$1.6 \cdot 10^5$	1.003	1
Spectra-600-25	600	Gaussian	600	12644	$3.5 \cdot 10^5$	1.003	1
Spectra-800-30	800	Gaussian	800	12664	$8.9 \cdot 10^5$	1.002	1
Spectra-1000-35	1000	Gaussian	1000	11866	$2.3 \cdot 10^6$	1.003	1
500-cvx-body	500	Gaussian	500	81	$7.8 \cdot 10^7$	1.022	5
1000-cvx-body	1000	Gaussian	1000	316	$5.6 \cdot 10^7$	1.013	3
e-coli	24	Exponential	18	920	$3.0 \cdot 10^4$	1.003	11.3
iAT-PLT-636	289	Exponential	216	186	$2.6 \cdot 10^7$	1.008	9.5
Recon1	931	Exponential	700	126	$4.0 \cdot 10^{10}$	1.091	7.9
iAB-RBC-283	129	Exponential	400	157	$3.6 \cdot 10^8$	1.153	912.3

Table 2. Sampling with ReHMC from spherical Gaussian and exponential distributions.

polytope or spectrahedron. We performed the experiments on a machine with 16GB of RAM and an Intel i7 CPU at 2.6GHz.

3.2 MCMC Diagnostics

To estimate the practical efficiency of our method, we measure the time needed to produce one independent sample after a total of N draws, which we define as $t_{\text{is}} = \frac{\text{Time to perform } N \text{ draws (us)}}{N_{\text{ess}}}$. The Effective Sample Size N_{ess} (ESS) measures the amount by which autocorrelation within chains increases uncertainty. Ideally, given independent, uncorrelated samples, the Central Limit Theorem implies that the estimation error of the sample mean of the observations is $O(1/\sqrt{N})$. If there is a correlation, then the estimation error is $O(1/\sqrt{N_{\text{ess}}})$ by the Monte Carlo Central Limit Theorem [38]. We use the definition and implementation of N_{ess} provided in [38], and we compute the minimum ESS.

The logic behind reporting t_{is} is that the metric balances fast performance (i.e., the time needed to produce the next sample in the chain, which may be highly correlated with the previous one) and the "bottleneck quality" of sampling (i.e., a sampling algorithm may be slower but able to produce samples with lower correlation and hence higher N_{ess}). Moreover, we measure the PSRF diagnostic [37] that measures whether a chain has mixed by comparing the variance between and the variance within the chain components. We calculate the PSRF of a chain by splitting it in half.

3.3 Convex body datasets

We experiment with the following convex bodies: *(1) Standard polytopes:* cubes, simplices, cross-polytopes, and products of simplices (P-Simplex). *(2) Application polytopes:* these include polytopes from the `BiGG` models [7] that model metabolic networks of biological systems; their feasible regions are the flux spaces of each network that makes (flux) sampling a powerful tool to study metabolism [47]. Also, we consider Birkhoff polytopes, i.e., the convex hull of the set of permutation matrices, that are of special interest and widely used in machine learning[3], computer vision, and in

[3]For example, [87] transforms inferring a ranking distribution to an optimization problem on a Birkhoff polytope.

various problems in convex optimization [35, 61]. *(3) Random Spectrahedra [26] (Spectra-d-M)*, and *(4) General Convex Bodies (d-cvx-body).* More specifically, we experiment with the following polytope (families) that in some cases, their dimension is parametrized by a natural number N:

- N-Cube. The N-dimensional cube $[-1,1]^N$.
- N-Simplex. $\Delta_N = \left\{x \in \mathbb{R}^N | \sum_{i=1}^N x_i \leq 1, x_i \geq 0\right\}$.
- N-Birkhoff. Its vertices correspond to the perfect matchings of the complete bipartite graph on N nodes. The Birkhoff polytope is the convex hull of the indicator vectors $\{\mathbf{1}_M | M$ is a perfect matching of $K_{N,N}\}$ and lies within on $(N^2 - 2N + 1)$-dimensional affine subspace of the N^2-dimensional space.
- N-Cross. The N-dimensional unit ball in ℓ_1-norm $\{x \in \mathbb{R}^N | \|x\|_1 \leq 1\}$.
- N-P-Simplex. The product $\Delta_N \times \Delta_N$.
- N-S-Cube. A skinny cube of the form $[-N, N] \times [-1,1]^{N-1}$.
- e-coli. The polytope corresponds to the core Escherichia coli metabolic model.
- iAB-RBC-283. A polytope corresponding to a proteomically derived knowledge base of erythrocyte metabolism.
- iAT-PLT-636. A polytope corresponds to a metabolic network related to the human platelet. iAT-PLT-636 is reconstructed using 33 proteomic datasets and 354 literature references. The network contains enzymes mapping to 403 diseases and 231 FDA-approved drugs.
- Recon1. The polytope corresponds to the human (homo sapiens) metabolic network.

Metabolic Polytopes. The metabolic polytopes initially had the representation $\{x \,|\, A_{eq}x = b_{eq},\ l \leq x \leq u\}$. To put them in the representation $\{x \,|\, Ax \leq b\}$ we took the following steps

- We compute the kernel of A_{eq}, say W, where each column of W is a column vector w such that $Aw = 0$.
- We compute the shift vector x_s to be the solution to the underdetermined system $A_{eq}x = b_{eq}$.
- We set A and b as
$$A = \begin{pmatrix} I_d \\ -I_d \end{pmatrix} W, \qquad b = \begin{pmatrix} u \\ -l \end{pmatrix} - \begin{pmatrix} I_d \\ -I_d \end{pmatrix} x_s.$$

The data for metabolic polytopes are publicly available from the BiGG models database.

Spectrahedra. For the random spectrahedra (Spectra-d-M), we consider the LMIs [26] defined using the following $A_1, \ldots, A_M$ matrices:

$$A_1 = ZZ^T + I_d, \qquad Z_{ij} \sim \mathcal{U}[0,1]$$
$$A_i = \begin{pmatrix} \tilde{Q}_i & O_d \\ O_d & -\tilde{Q}_i \end{pmatrix},$$
$$\tilde{Q}_i = Q_i + Q_i^T,\ Q_i \sim \mathcal{U}\left([-1,1]^{(M/2)\times(M/2)}\right) \text{ for } i \in [2,d].$$

Convex Bodies. Finally, we sample from the following generic convex body (d-cvx-body) (Figure 3 represents samples from the 2-cvx-body):

$$K = \left\{(x_1, \ldots, x_d) \in \mathbb{R}^d : g_1(x_1, \ldots, x_d) \stackrel{\text{def}}{=} \log \sum_{i=1}^d \exp(x_i) \leq 0,\ g_2(x_1, \ldots, x_d) \stackrel{\text{def}}{=} \sum_{i=1}^d x_i^2 - d^2 \leq 0\right\}.$$

Fig. 3. 2-cvx-body sample scatter plot.

The gradients of g_1, g_2 are

$$\nabla g_1(x_1, \ldots, x_d) = \left(\frac{\exp(x_i)}{\sum_{j=1}^{d} \exp(x_j)} \right)_{i \in [d]} \quad \text{(softmax function)},$$
$$\nabla g_2(x_1, \ldots, x_d) = (2x_i)_{i \in [d]} \, .$$

3.4 Comparison Experiments

We compare with HOPS-H&R and HOPS-CH&R, the standard algorithms for sampling from a truncated log-concave density in practice. We sample from a Gaussian density centered at the Chebyshev center with unit covariance. The experiments run with walk length $w \in [1, d]$ with an increment of $\lfloor d/10 \rfloor$ for all samplers (we perform grid search), while for ReHMC, we set an initial step size equal to $\eta_0 = R_c/10$ where R_c is the Chebyshev radius. For all methods, we count t_{is} over a range of parameters regarding w and report the minimum t_{is} observed given that the corresponding chain has a PSRF ≤ 1.2. For ReHMC, we also report the average number of reflections for the selected experiment and the step size used *after* burn-in. The results account for a total of 80K draws per experiment with a burn-in of 20K draws and are in Table 1. We apply a random rotation to 100-Cube, 100-Simplex, and 50-P-Simplex.

Scaling Experiments. We quantify how ReHMC scales by sampling from bodies of $\sim 10^3$ dimensions: (i) A 1000-Cube. (ii) a 1000-Simplex, (iii) the Birkhoff polytope with 33 elements ($d = 1024$), (iv) the polytope that represents the human metabolic network (Recon1) in systems biology ($d = 931, M = 4934$), (v) spectrahedra ($d \in \{200, 400, 600, 800, 1000\}$), (vi) a convex body ($d \in \{500, 1000\}$). We sample from a unit covariance Gaussian centered at the Chebyshev center for $N = 80K$ draws with a burn-in of 20K draws from a warm start. To illustrate further the efficiency of ReHMC we sample from the exponential distribution $\pi(x) \propto e^{-c^T x}$ supported on a convex polytope. In particular, to make the examples more interesting, we consider two additional polytopes −except Recon 1− that represent metabolic networks in systems biology [79]; the function $g_c(x) = c^T x$ evaluates the biomass of the network, where $c \in \mathbb{R}^n$ is its biomass vector. Sampling from $\pi(x)$ is an essential operation in metabolic network analysis [30]. The results are in Table 2. Note that HOPS *does not support* spectrahedra and general convex bodies.

4 DISCUSSION & CONCLUSION

Competitors. ReHMC can scale up to $1.03 - 1359\times$ faster than H&R-HOPS in terms of t_{is} as seen from Table 1. More specifically, while the two algorithms have comparable performance on the 100-Cube (ours surpassed H&R by 1.03×), ReHMC outperformed the H&R algorithm on the 100-Simplex (45× faster), 100-S-Cube (1.2× faster), 10-Birkhoff (8.8× faster), 10-Cross (19.4× faster), 50-P-Simplex (31.9× faster), e-coli (1359.9× faster), iAB-RBC-283, and iAT-PLT-636 ($\sim$ 22× faster). The poor performance of H&R on the simplex is due to its worst possible isotropic constant over all *simplicial polytopes* [81]. HOPS cannot sample from iAB-RBC-283 where it could not *round* the polytope due to its geometry. Regarding the CH&R-HOPS, it outperforms ReHMC only on the 100-Cube (3.9×),[4] whereas it performs up to 386.1× slower on the rest of the benchmarks. More specifically, we outperform CH&R-HOPS on 100-Simplex (8.9× faster), 100-S-Cube (4.07× faster), 10-Birkhoff (26.7× faster), 10-Cross (5.23× faster), 50-P-Simplex (386.1× faster), iAB-RBC-283, and iAT-PLT-636 ($\sim$ 10× faster). Interestingly, ReHMC computes, in all cases, a higher quality sampler, in terms of PSRF, than both H&R and CH&R. This results in a larger performance gain for ReHMC if we restrict all samplers to stop after a certain PSRF value is attained. Finally, the chosen walk length w from grid search does not seem to be related to the geometry of the body.

Scaling. ReHMC scales to a few thousand dimensions and samples with very low PSRF, whereas current implementations of *truncated HMC* [2] experimented with ≤ 50 dimensions. In particular, we efficiently sample (in a few *hours*) from a 1000-dimensional cube and simplex, a large Birkhoff polytope, a Recon1 metabolic polytope, spectrahedra, and two (general) convex bodies.

Billiard Behaviour. The average number of reflections $\bar{\ell}$ for every experiment is *ergodic and bounded* across all instances (see Tables 1 and 2), suggesting that our theoretical assumption for the boundedness of ℓ is reasonable and that the adaptive rule for determining the step-size works well in practice. Also, the average number of reflections varies among instances. For example, iAB-RBC-283 and 1000-Simplex have $\bar{\ell} \approx 24$, and $\bar{\ell} \approx 20$ respectively due to the fact that a large fraction of the Gaussian's measure is *close* to ∂K. Conversely, Recon1 has $R_c \approx 10^4$, i.e. most of the Gaussian measure is concentrated *away* from ∂K, and has $\bar{\ell} = 0$. Theoretical bounds for ℓ are the subject of chaotic billiards (e.g., see [55, 86, 89]) and lie beyond the scope of this paper. However, we believe that in the case of Gaussian distributions, we can apply the Gaussian Annulus Theorem to get an estimate of the average number of reflections $\bar{\ell}$.

Conclusions. We introduce an algorithm, ReHMC, for sampling from a truncated log-concave density $\pi(x) \propto e^{-f(x)}$. We also provide an online rule to estimate the step size during the burn-in phase, and we provide an open-source implementation that can sample from a variety of convex bodies (polytopes, spectrahedra, and general convex bodies) up to $\sim 10^3\times$ faster than H&R, subject to empirical mixing criteria. As an interesting future direction to employ different integrators to approximate the Hamiltonian dynamics and provide new mixing time guarantees, as well as use *stochastic gradients* in place of the true gradients. Finally, devising different heuristics for the step size of the leapfrog integrator, such as making the step size be also a function of the proportion of accepted samples, is interesting future work.

ACKNOWLEDGEMENTS

The authors would like to thank the Editor and the Reviewers for their insightful comments. MP wants to thank the Google Summer of Code 2020 program for making this work come to fruition, and Jon Kleinberg for his useful comments. MP was partially supported by a Cornell University Fellowship, a grant from the A.G. Leventis Foundation, a grant

[4]This is so because ψ_{me} is close to 1, the body is isotropic, and the starting point is the cube's center.

from the Gerondelis Foundation, and a LinkedIn Ph.D. Fellowship. ET is partially supported by ANR JCJC GALOP (ANR-17-CE40-0009).

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*. 265–283.

[2] Hadi Mohasel Afshar and Justin Domke. 2015. Reflection, refraction, and Hamiltonian Monte Carlo. In *Advances in neural information processing systems*. 3007–3015.

[3] Kendall Atkinson and Weimin Han. 2005. *Theoretical numerical analysis*. Vol. 39. Springer.

[4] Scott A Becker, Adam M Feist, Monica L Mo, Gregory Hannum, Bernhard Ø Palsson, and Markus J Herrgard. 2007. Quantitative prediction of cellular metabolism with constraint-based models: the COBRA Toolbox. *Nature protocols* 2, 3 (2007), 727–738.

[5] Michael Betancourt. 2013. A general metric for Riemannian manifold Hamiltonian Monte Carlo. In *International Conference on Geometric Science of Information*. Springer, 327–334.

[6] Michael Betancourt. 2017. A conceptual introduction to Hamiltonian Monte Carlo. *arXiv preprint arXiv:1701.02434* (2017).

[7] BiGG. 2020. *BiGG Polytope Database*. http://bigg.ucsd.edu/models

[8] Eli Bingham, Jonathan P Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D Goodman. 2019. Pyro: Deep universal probabilistic programming. *The Journal of Machine Learning Research* 20, 1 (2019), 973–978.

[9] Zdravko Botev and Pierre L'Ecuyer. 2019. *Simulation from the Tail of the Univariate and Multivariate Normal Distribution*. Springer International Publishing, Cham, 115–132. https://doi.org/10.1007/978-3-319-92378-9_8

[10] Nawaf Bou-Rabee and J. Sanz-Serna. 2018. Geometric Integrators and the Hamiltonian Monte Carlo method. *Acta Numerica* (01 2018), 1–92. https://doi.org/10.1017/S09624929

[11] Stephen Boyd and Lieven Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press, New York, NY, USA.

[12] Andrew Brock, Jeff Donahue, and Karen Simonyan. 2019. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

[13] Nicolas Brosse, Alain Durmus, Éric Moulines, and Marcelo Pereyra. 2017. Sampling from a log-concave distribution with compact support with proximal Langevin Monte Carlo. In *Proceedings of the 2017 Conference on Learning Theory (Proceedings of Machine Learning Research, Vol. 65)*, Satyen Kale and Ohad Shamir (Eds.). PMLR, Amsterdam, Netherlands, 319–342.

[14] Sébastien Bubeck, Ronen Eldan, and Joseph Lehec. 2018. Sampling from a log-concave distribution with projected langevin monte carlo. *Discrete & Computational Geometry* 59, 4 (2018), 757–783.

[15] Ludovic Calès, Apostolos Chalkis, Ioannis Z. Emiris, and Vissarion Fisikopoulos. 2018. Practical Volume Computation of Structured Convex Bodies, and an Application to Modeling Portfolio Dependencies and Financial Crises. In *34th International Symposium on Computational Geometry (SoCG 2018) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 99)*, Bettina Speckmann and Csaba D. Tóth (Eds.). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 19:1–19:15. https://doi.org/10.4230/LIPIcs.SoCG.2018.19

[16] Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. 2017. Stan: A probabilistic programming language. *Journal of statistical software* 76, 1 (2017).

[17] Apostolos Chalkis, Vissarion Fisikopoulos, Panagiotis Repouskos, and Elias Tsigaridas. 2021. Sampling the feasible sets of SDPs and volume approximation. *ACM Communications in Computer Algebra* 54, 3 (2021), 114–118.

[18] Y. Chen, R. Dwivedi, M. J. Wainwright, and B. Yu. 2020. Fast mixing of Metropolized Hamiltonian Monte Carlo: Benefits of multi-step gradients. *Journal of machine learning research* (2020).

[19] Augustin Chevallier, Frédéric Cazals, and Paul Fearnhead. 2022. Efficient computation of the the volume of a polytope in high-dimensions using Piecewise Deterministic Markov Processes. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 151)*, Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera (Eds.). PMLR, 10146–10160. https://proceedings.mlr.press/v151/chevallier22a.html

[20] Augustin Chevallier, Sylvain Pion, and Frédéric Cazals. 2020. Improved polytope volume calculations based on Hamiltonian Monte Carlo with boundary reflections and sweet arithmetics. *[Research Report] RR-9222, INRIA Sophia-Antipolis, France. 2018. hal-01919855v2* (2020).

[21] Benjamin Cousins. 2017. *Efficient high-dimensional sampling and integration*. Ph. D. Dissertation. Georgia Institute of Technology.

[22] Ben Cousins and Santosh Vempala. 2014. A Cubic Algorithm for Computing Gaussian Volume. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms* (Portland, Oregon) *(SODA '14)*. Society for Industrial and Applied Mathematics, USA, 1215–1228.

[23] Benjamin Cousins and Santosh Vempala. 2015. Bypassing KLS: Gaussian Cooling and an $O^*(n^3)$ Volume Algorithm. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*. 539–548.

[24] B. Cousins and S. Vempala. 2016. A practical volume algorithm. *Mathematical Programming Computation* 8 (2016). Issue 2.

[25] Mike Cyrus and Jay Beck. 1978. Generalized two-and three-dimensional clipping. *Computers & Graphics* 3, 1 (1978), 23–28.

[26] Fabrizio Dabbene, Pavel S Shcherbakov, and Boris T Polyak. 2010. A randomized cutting plane method with probabilistic geometric convergence. *SIAM Journal on Optimization* 20, 6 (2010), 3185–3207.

[27] Khue-Dung Dang, Matias Quiroz, Robert Kohn, Minh-Ngoc Tran, and Mattias Villani. 2019. *Hamiltonian Monte Carlo with energy conserving subsampling*. MIT Press.

[28] Constantinos Daskalakis, Themis Gouleakis, Christos Tzamos, and Manolis Zampetakis. 2019. Computationally and Statistically Efficient Truncated Regression. In *Conference on Learning Theory*. 955–960.

[29] Constantinos Daskalakis, Dhruv Rohatgi, and Manolis Zampetakis. 2020. Truncated Linear Regression in High Dimensions. *arXiv preprint arXiv:2007.14539* (2020).

[30] Daniele De Martino, Anna MC Andersson, Tobias Bergmiller, Călin C. Guet, and Gašper Tkačik. 2018. Statistical mechanics for metabolic networks during steady state growth. *Nature Communications* 9, 1 (Jul 2018). https://doi.org/10.1038/s41467-018-05417-9

[31] Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. 1987. Hybrid Monte Carlo. *Physics letters B* 195, 2 (1987), 216–222.

[32] Raaz Dwivedi, Yuansi Chen, Martin J Wainwright, and Bin Yu. 2019. Log-concave sampling: Metropolis-Hastings algorithms are fast. *Journal of Machine Learning Research* 20, 183 (2019), 1–42.

[33] Martin Dyer, Alan Frieze, and Ravi Kannan. 1991. A random polynomial-time algorithm for approximating the volume of convex bodies. *Journal of the ACM (JACM)* 38, 1 (1991), 1–17.

[34] I.Z. Emiris and V. Fisikopoulos. 2018. Practical polytope volume approximation. *ACM Trans. Math. Soft.* 44, 4 (2018), 38:1–38:21. https://doi.org/10.1145/3194656 Prelim. version: Proc. SoCG 2014.

[35] F. Fogel, R. Jenatton, F. Bach, and A. d'Aspremont. 2013. Convex Relaxations for Permutation Problems. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1* (Lake Tahoe, Nevada) *(NIPS'13)*. Curran Associates Inc., Red Hook, NY, USA, 1016–1024.

[36] Dani Gamerman and Hedibert F Lopes. 2006. *Markov chain Monte Carlo: stochastic simulation for Bayesian inference*. CRC Press.

[37] Andrew Gelman, Donald B Rubin, et al. 1992. Inference from iterative simulation using multiple sequences. *Statistical science* 7, 4 (1992), 457–472.

[38] Charles Geyer. 2011. Introduction to Markov chain Monte Carlo. *Handbook of Markov chain Monte Carlo* 20116022 (2011), 45.

[39] W. R. Gilks and P. Wild. 1992. Adaptive Rejection Sampling for Gibbs Sampling. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 41, 2 (1992), 337–348. http://www.jstor.org/stable/2347565

[40] Mark Girolami, Ben Calderhead, and Siu A Chin. 2009. Riemannian manifold Hamiltonian Monte Carlo. *arXiv preprint arXiv:0907.1100* (2009).

[41] Elena Gryazina and Boris Polyak. 2014. Random sampling: Billiard walk algorithm. *European Journal of Operational Research* 238, 2 (2014), 497–504.

[42] Gaël Guennebaud, Benoît Jacob, et al. 2010. Eigen v3. http://eigen.tuxfamily.org.

[43] T. Hahn. 2020. *Intel(R) Math Kernel Library*. https://software.intel.com/content/www/us/en/develop/tools/oneapi/components/onemkl.html

[44] Hulda S Haraldsdóttir, Ben Cousins, Ines Thiele, Ronan MT Fleming, and Santosh Vempala. 2017. CHRR: coordinate hit-and-run with rounding for uniform sampling of constraint-based models. *Bioinformatics* 33, 11 (2017), 1741–1743.

[45] W. K. Hastings. 1970. Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika* 57, 1 (1970), 97–109. http://www.jstor.org/stable/2334940

[46] Runxin He and Humberto Gonzalez. 2017. Numerical synthesis of pontryagin optimal control minimizers using sampling-based methods. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 733–738.

[47] H. Herrmann, B. Dyson, L. Vass, G. Johnson, and J. Schwartz. 2019. Flux sampling is a powerful tool to study metabolism under changing environmental conditions. *npj Systems Biology and Applications* 5 (2 Sept. 2019). https://doi.org/10.1038/s41540-019-0109-0

[48] Vu Anh Huynh, Sertac Karaman, and Emilio Frazzoli. 2012. An incremental sampling-based algorithm for stochastic optimal control. In *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2865–2872.

[49] Andrew Ilyas, Emmanouil Zampetakis, and Constantinos Daskalakis. 2020. A Theoretical and Practical Framework for Regression and Classification from Truncated Samples. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 4463–4473.

[50] Johann F Jadebeck, Axel Theorell, Samuel Leweke, and Katharina Nöh. 2020. HOPS: high-performance library for (non-) uniform sampling of convex-constrained models. *Bioinformatics* (2020).

[51] M. Jerrum and A. Sinclair. 1988. Conductance and the Rapid Mixing Property for Markov Chains: The Approximation of Permanent Resolved. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing* (Chicago, Illinois, USA) *(STOC '88)*. ACM, New York, NY, USA, 235–244. https://doi.org/10.1145/62212.62234

[52] Fritz John. 2014. Extremum problems with inequalities as subsidiary conditions. In *Traces and emergence of nonlinear programming*. Springer, 197–215.

[53] Adam Tauman Kalai and Santosh Vempala. 2006. Simulated Annealing for Convex Optimization. *Mathematics of Operations Research* 31, 2 (2006), 253–266. http://www.jstor.org/stable/25151723

[54] Ravi Kannan, László Lovász, and Ravi Montenegro. 2006. Blocking conductance and mixing in random walks. *Comb. Probab. Comput.* 15, 4 (2006), 541–570.

[55] Valerii Kozlov and Dmitrii Treshchev. 1991. *Billiards: A Genetic Introduction to the Dynamics of Systems with Impacts: A Genetic Introduction to the Dynamics of Systems with Impacts*. Vol. 89. American Mathematical Soc.

[56] Aditi Laddha and Santosh S. Vempala. 2021. Convergence of Gibbs Sampling: Coordinate Hit-and-Run Mixes Fast. In *SoCG*.

[57] Yin Tat Lee, Ruoqi Shen, and Kevin Tian. 2020. Logsmooth Gradient Concentration and Tighter Runtimes for Metropolized Hamiltonian Monte Carlo. *arXiv preprint arXiv:2002.04121* (2020).

[58] Yin Tat Lee, Zhao Song, and Santosh S Vempala. 2018. Algorithmic theory of ODEs and sampling from well-conditioned log-concave densities. *arXiv preprint arXiv:1812.06243* (2018).

[59] Yin Tat Lee and Santosh S Vempala. 2018. Convergence rate of Riemannian Hamiltonian Monte Carlo and faster polytope volume computation. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. 1115–1121.

[60] Xuechen Li, Yi Wu, Lester Mackey, and Murat A Erdogdu. 2019. Stochastic Runge-Kutta accelerates Langevin Monte Carlo and beyond. In *Advances in Neural Information Processing Systems*. 7748–7760.

[61] C.H. Lim and S. Wright. 2014. Beyond the Birkhoff Polytope: Convex Relaxations for Vector Permutation Problems. In *Advances in Neural Information Processing Systems*, Vol. 27. Curran Associates, Inc., 2168–2176.

[62] László Lovász and Miklós Simonovits. 1990. The mixing rate of Markov chains, an isoperimetric inequality, and computing the volume. In *Proceedings 31st annual symposium on foundations of computer science*. IEEE, 346–354.

[63] Laszlo Lovasz and Santosh Vempala. 2006. Fast Algorithms for Logconcave Functions: Sampling, Rounding, Integration and Optimization. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*. 57–68. https://doi.org/10.1109/FOCS.2006.28

[64] László Lovász and Santosh Vempala. 2006. Hit-and-run from a corner. *SIAM J. Comput.* 35, 4 (2006), 985–1005.

[65] L. Lovász and M. Simonovits. 1993. Random walks in a convex body and an improved volume algorithm. *Random Structures & Algorithms* 4, 4 (1993), 359–412. https://doi.org/10.1002/rsa.3240040402 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/rsa.3240040402

[66] Marco Marchesi. 2017. Megapixel size image creation using generative adversarial networks. *arXiv preprint arXiv:1706.00082* (2017).

[67] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. 1953. Equation of state calculations by fast computing machines. *The journal of chemical physics* 21, 6 (1953), 1087–1092.

[68] Sean P Meyn and Richard L Tweedie. 2012. *Markov chains and stochastic stability*. Springer Science & Business Media.

[69] Pierre Monmarché. 2020. High-dimensional MCMC with a standard splitting scheme for the underdamped Langevin diffusion. *arXiv e-prints* (2020), arXiv–2007.

[70] Wenlong Mou, Nicolas Flammarion, Martin J Wainwright, and Peter L Bartlett. 2019. An efficient sampling algorithm for non-smooth composite potentials. *arXiv preprint arXiv:1910.00551* (2019).

[71] Hariharan Narayanan and Piyush Srivastava. 2021. On the mixing time of coordinate Hit-and-Run. *Combinatorics, Probability and Computing* (2021), 1–13. https://doi.org/10.1017/S0963548321000328

[72] Radford M. Neal. 2003. Slice sampling. *The Annals of Statistics* 31, 3 (2003), 705 – 767. https://doi.org/10.1214/aos/1056562461

[73] Radford M Neal et al. 2011. MCMC using Hamiltonian dynamics. *Handbook of markov chain monte carlo* 2, 11 (2011), 2.

[74] Akihiko Nishimura and David Dunson. 2016. Geometrically tempered Hamiltonian Monte Carlo. *arXiv preprint arXiv:1604.00872* (2016).

[75] Sheehan Olver and Alex Townsend. 2013. Fast inverse transform sampling in one and two dimensions. arXiv:1307.1223 [math.NA]

[76] Terence J O'Neill and Simon C Barry. 1995. Truncated logistic regression. *Biometrics* (1995), 533–541.

[77] Jeffrey D Orth, Ines Thiele, and Bernhard Ø Palsson. 2010. What is flux balance analysis? *Nature biotechnology* 28, 3 (2010), 245–248.

[78] Ari Pakman. 2015. Package 'tmg'. (2015). https://cran.r-project.org/package=tmg.

[79] Bernhard Ø. Palsson. 2015. *Systems biology*. Cambridge university press.

[80] Marcelo Pereyra. 2016. Proximal Markov chain Monte Carlo algorithms. *Statistics and Computing* 26, 4 (2016), 745–760.

[81] Luis Rademacher. 2016. A simplicial polytope that maximizes the isotropic constant must be a simplex. *Mathematika* 62, 1 (2016), 307–320.

[82] Gareth O Roberts and Osnat Stramer. 2001. On inference for partially observed nonlinear diffusion models using the Metropolis–Hastings algorithm. *Biometrika* 88, 3 (2001), 603–621.

[83] Jan Schellenberger and Bernhard Ø. Palsson. 2009. Use of randomized sampling for analysis of metabolic networks. *Journal of biological chemistry* 284, 9 (2009), 5457–5461.

[84] Ruoqi Shen and Yin Tat Lee. 2019. The randomized midpoint method for log-concave sampling. In *Advances in Neural Information Processing Systems*. 2098–2109.

[85] Ruoqi Shen, Kevin Tian, and Yin Tat Lee. 2020. Composite Logconcave Sampling with a Restricted Gaussian Oracle. *arXiv preprint arXiv:2006.05976* (2020).

[86] Yakov G Sinai. 1970. Dynamical systems with elastic reflections. *Russian Mathematical Surveys* 25, 2 (1970), 137.

[87] Ashudeep Singh and Thorsten Joachims. 2018. Fairness of exposure in rankings. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2219–2228.

[88] Robert L Smith. 1996. The Hit-and-Run sampler: a globally reaching Markov chain sampler for generating arbitrary multivariate distributions. In *Proceedings Winter Simulation Conference*. IEEE, 260–264.

[89] Serge Tabachnikov. 2005. *Geometry and billiards*. Vol. 30. American Mathematical Soc.

[90] Denis Talay and Luciano Tubaro. 1990. Expansion of the global error for numerical schemes solving stochastic differential equations. *Stochastic analysis and applications* 8, 4 (1990), 483–509.

[91] Santosh Vempala. 2005. Geometric random walks: a survey. *Combinatorial and computational geometry* 52, 573-612 (2005), 2.

[92] Nisheeth K. Vishnoi. 2021. An Introduction to Hamiltonian Monte Carlo Method for Sampling. arXiv:2108.12107 [cs.DS] arXiv:2108.12107.

[93] Margaret H Wright. 1994. Some properties of the Hessian of the logarithmic barrier function. *Mathematical Programming* 67, 1-3 (1994), 265–295.

[94] Kexin Yi and Finale Doshi-Velez. 2017. Roll-back Hamiltonian Monte Carlo. *arXiv preprint arXiv:1709.02855* (2017).

A MARKOV-CHAIN-MONTE-CARLO

A.1 Markov-Chain-Monte-Carlo Algorithms

An extensive family of algorithms in the sampling regime is the *Markov-Chain-Monte-Carlo* (MCMC) (or *Metropolis-Hastings*) algorithms introduced in the seminal works of [45, 67]. The logic of an MCMC algorithm is the following: We start with an *initial density* π_0, and we simulate the following steps: First, we have a proposal step. The proposal step proposes a state $\tilde{x}$ given that the sampler is already in state x. The proposal step makes use of the proposal function $\mathcal{P} : K \times K \longrightarrow [0, \infty)$, where $\mathcal{P}(x, \cdot)$ represents a density over $x \in K$. So, at the proposal step, we sample a state $\tilde{x} \sim (x, \cdot)$, and write $\tilde{x} \sim \mathcal{P}_x$ in shorthand. In the second step, known as the *accept-reject* step, the algorithm accepts the proposal $\tilde{x}$ of the first step as the new state of the sampler with probability

$$\alpha(x, \tilde{x}) \stackrel{\text{def}}{=} \min\left\{1, \frac{\pi(\tilde{x})\mathcal{P}(\tilde{x}, x)}{\pi(x)\mathcal{P}(x, \tilde{x})}\right\} \tag{7}$$

otherwise, with probability $1 - \alpha(x, \tilde{x})$ the sampler rejects $\tilde{x}$ and remains in x. This process is also known as the Metropolis-Hastings Filter [73] and is applied to ensure that π is a stationary density for this Markov Chain. Drawing the proposal $\tilde{x}$ and applying the accept-reject step can be combined to give the overall transition kernel $\mathcal{T}(x, \tilde{x})$ defined as

$$\mathcal{T}(x, \tilde{x}) \stackrel{\text{def}}{=} \mathcal{P}(x, \tilde{x})\alpha(x, \tilde{x}) \qquad x \neq \tilde{x} \tag{8}$$

where again $\mathcal{T}(x, \cdot)$ is a probability density function which we will denote in shorthand as $\mathcal{T}_x$ and thus $\tilde{x} \sim \mathcal{T}_x$.

Sampling from a log-concave density can be performed in multiple ways since the proposal distribution $\mathcal{P}_x$ can vary between the methods. Some methods include: (a) independence sampling where $\tilde{x} \sim \mathcal{N}(0, \Sigma)$, (b) random-walk Metropolis (RWM) where $\tilde{x} \sim \mathcal{N}(x, 2\eta I_d)$, (c) Metropolis-Adjusted Langevin Algorithm (MALA) where $\tilde{x} \sim \mathcal{N}(x - \eta\nabla f(x), 2\eta I_d)$, (d) Ball-walk (BW), (e) H&R, (f) Coordinate-Hit-and-Run (CHR), (g) Underdamped Langevin Diffusion (ULD), and Hamiltonian Monte Carlo (HMC). For a more detailed discussion of the various samplers, we redirect the interested reader to [18, 18, 68, 82, 84, 90, 91] and the references therein.

A.2 Hamiltonian Dynamics

The Hamiltonian Dynamics [6, 73] is an interpretation for studying the evolution of physical systems. In this formulation, we have a particle of mass m_p with velocity v and position x. The particle moves in a conservative potential $\mathcal{U}(x)$ where it experiences a force $-\nabla\mathcal{U}(x)$ which is dependent on its position and has a Kinetic Energy $\mathcal{K}(v) = \frac{1}{2}m_p\|v\|^2$. The dynamics of the particle evolve according to Newton's Second Law, that is $m_p\dot{v} = -\nabla\mathcal{U}(x)$, or equivalently, in terms of the Hamiltonian $\mathcal{H}(x, v) = \mathcal{K}(v) + \mathcal{U}(x)$

$$\frac{dx}{dt} = +\frac{\partial\mathcal{H}}{\partial v} = +v, \quad \frac{dv}{dt} = -\frac{\partial\mathcal{H}}{\partial x} = -\frac{1}{m_p}\nabla\mathcal{U}(x) \tag{9}$$

The above system of equations preserves the Hamiltonian over time since

$$\frac{dH}{dt} = \frac{\partial\mathcal{H}}{\partial x}\frac{dx}{dt} + \frac{\partial H}{\partial v}\frac{dv}{dt} = \frac{\partial\mathcal{H}}{\partial x}\frac{\partial H}{\partial v} - \frac{\partial H}{\partial v}\frac{\partial H}{\partial x} = 0 \tag{10}$$

Therefore the system moves on the level sets of the Hamiltonian function, that is $C(E) = \{(x, v) | \mathcal{H}(x, v) = E = \text{const.}\}$. From now on, and for notational convenience, we will assume that the particle has unit mass, that is, $m_p = 1$. An alternative formulation of the Hamiltonian Dynamics defines a joint variable $z = (x, v)$ and evolves according to

$$\frac{dz}{dt} = J\nabla H(z), \quad J = \begin{pmatrix} O & I \\ -I & O \end{pmatrix} \tag{11}$$

Or in terms of a mapping we can define $\mathbb{T}_s$ such that given a state $z(t)$, it produces the state $z(t+s)$ as

$$z(t+s) = \mathbb{T}_s z(t) = z(t) + \int_t^{t+s} J\nabla H(z(t))dt \tag{12}$$

with an inverse mapping $\mathbb{T}_{-s}$, which can be obtained by negating v, applying the forward mapping, and negating v again. The Hamiltonian Dynamics are *symplectic*, that is for the Jacobian mapping B_s of $\mathbb{T}_s$ we have that $B_s^\top J^{-1} B_s = J^{-1}$. As a consequence, the operator is volume preserving as well, which means that for every region $R \subseteq \mathbb{R}^d$, we have $\text{Vol}(R) = \text{Vol}(\mathbb{T}_s R)$. This property can be proven by proving that the Jacobian of the mapping $\mathbb{T}_s$ for infinitesimal s has absolute value 1, or the divergence of the vector field $F(z) = J\nabla H(z)$ is 0.

Discretization of the Hamiltonian Dynamics. Solving the Hamiltonian Dynamics ODE in a computer setting requires discretizing the underlying ode $\dot{z} = J\nabla H(z)$. For this reason, multiple methods have been proposed. The easiest one is, perhaps, the Euler method where

$$v_{i+1} = v_i - \eta\nabla f(x_i), \quad x_{i+1} = x_i + \eta v_i \tag{13}$$

and its improvement, which uses the already computed value of v_{i+1}

$$v_{i+1} = v_i - \eta\nabla f(x_i), \quad x_{i+1} = x_i + \eta v_{i+1} \tag{14}$$

These methods, which are very simple and conceivable, are usually prone to numerical errors and may become unstable. Moreover, they have an $O(\eta^2)$ local error and an $O(\eta)$ global error. A better way to discretize the Hamiltonian Dynamics is through the *leapfrog integrator*

$$\widehat{v}_{i+1} = v_i - \frac{\eta}{2}\nabla f(x_i), \quad x_{i+1} = x_i + \eta\widehat{v}_{i+1}, \quad v_{i+1} = \widehat{v}_{i+1} - \frac{\eta}{2}\nabla f(x_{i+1}) \tag{15}$$

which has an $O(\eta^3)$ local error and an $O(\eta^2)$ global error. An even smaller error, at the expense of computational power, can be achieved with Runge-Kutta [10, 60], Runge-Kutta-Nyström, Verlet integrators [69], and Collocation Methods [58].

A.3 Hamiltonian Monte Carlo

Having gained intuition about the properties of the Hamiltonian Dynamics, we describe the Hamiltonian Monte Carlo Algorithm (HMC) [27, 31]. More specifically, HMC relies on simulating a particle (x, v) with Kinetic Energy $\mathcal{K}(v) = \frac{1}{2}\|v\|^2$ and Potential Energy $\mathcal{U}(x) = f(x)$ to draw samples from a target distribution Π. The state $z = (x, v)$ of the system evolves via the Hamiltonian Dynamics of (9). The sampler starts initially with a sample $x_0 \sim \Pi_0$ where Π_0 is the starting distribution (whose form determines how many iterations the algorithm needs to mix to the desired distribution Π) and an initial velocity $v_0 \sim \mathcal{N}(0, I_d)$ and runs an iteration, using a numerical integration method, to yield a proposal $(\tilde{x}_0, \tilde{v}_0)$. In the continuous setting, the Hamiltonian is preserved over time, which can be directly deduced

using the chain rule and (9). When, however, the ODE is solved with a computer, a discretization error is added, and the Hamiltonian is not constant in general. For this reason, the sampler either sets x_1 equal to $\tilde{x}_0$ with probability equal to $\min\{1, \exp(\mathcal{H}(x_0, v_0) - \mathcal{H}(\tilde{x}_0, \tilde{v}_0))\}$, or rejects the proposal with probability $1 - \min\{1, \exp(\mathcal{H}(x_0, v_0) - \mathcal{H}(\tilde{x}_0, \tilde{v}_0))\}$, thus setting the sample x_1 again to x_0. The procedure repeats, generating sample x_{i+1} starting from the previous sample x_i and a velocity $v_i \sim \mathcal{N}(0, I_d)$. In the case of $K = \mathbb{R}^d$ (unconstrained sampling) we can use the second-order Leapfrog Integrator and return the proposal $(\tilde{x}, \tilde{v})$ for some input (x, v). This procedure leaves the joint distribution $\pi(x, v) \propto \exp(-\mathcal{H}(x, v))$ invariant. For every "small" set $A \subseteq \mathbb{R}^d$ and set B reachable by A through $\mathbb{T}_s$, we have that the Hamiltonian is constant (over an adequately small A) $\Pi(A) = \frac{V}{Z}\exp(-H_A)$, $\Pi(B) = \frac{V}{Z}\exp(-H_B)$ and

$$\frac{V}{Z}\exp(-H_A)\min\{1, \exp(-H_B + H_A\} = \frac{V}{Z}\exp(-H_B)\min\{1, \exp(-H_A + H_B)\} \tag{16}$$

where $V = \mathrm{Vol}(A) = \mathrm{Vol}(B)$. For a more detailed introduction to the subject, we redirect the interested reader to [73], and [6].

B HAMILTONIAN MONTE CARLO FOR TRUNCATED SAMPLING

In Sections A.2, and A.3 we have discussed the case where the potential $\mathcal{U}(x) = f(x)$ is a *smooth function*, i.e. its gradient $\nabla f(x)$ does not explode at any point in the domain. In this section, we will focus on the setting where $\mathcal{U}(x)$ is non-smooth. More general, the form of $\mathcal{U}$ we assume is the following

$$\mathcal{U}(x) = \begin{cases} f(x) & x \in K \\ \infty & x \notin K \end{cases} \tag{17}$$

where, again, f is a an L-smooth and m-strongly convex function defined with domain a superset of K, and K is a convex body. A particle under such a potential, encounters an *infinite-potential barrier* and never has the energy to overcome it. The behaviour of this particle is therefore *reflective* at the boundary.

In the sampling context, the problem of sampling from this density is equivalent to sampling from

$$\pi(x) \propto \exp(-\mathcal{U}(x)) \propto \begin{cases} \exp(-f(x)) & x \in K \\ 0 & x \notin K \end{cases} \tag{18}$$

In the discretized Hamiltonian Dynamics with the leapfrog integrator, we first perform the *velocity half-update* and the *position update* from an initial state (x, v) as

$$\widehat{v} = v - \frac{\eta}{2}\nabla f(x), \qquad \tilde{x} = x + \eta\widehat{v} \tag{19}$$

Note that the newly computed position $\tilde{x}$ may not lie in K. In case it does not lie inside K, we need perform a reflection as follows

$$\widehat{v} \mapsto -2(\widehat{v}^\top n)n + \widehat{v}, \qquad \tilde{x} \mapsto -2\eta(\widehat{v}^\top n)n + \eta\widehat{v} + x \tag{20}$$

where n is the normal at the point of the intersection $\{z | z = tx + (1-t)\tilde{x}, t \in [0, 1]\} \cap K$, and the reflection can be applied multiple times until the position falls inside K yielding the state $(\tilde{x}', \widehat{v}')$. We then perform the final velocity step

$$\tilde{v}' = \widehat{v}' - \frac{\eta}{2}\nabla f(\tilde{x}') \tag{21}$$

We can prove that these dynamics are volume-preserving since we can break the transformation to 3 parts $(x, v) \mapsto (\tilde{x}, \widehat{v}) \mapsto (\tilde{x}', \widehat{v}') \mapsto (\tilde{x}', \tilde{v}')$ each of which is trivially volume preserving.

B.1 Volume preservation and time reversibility of the discretized dynamics

We prove that the discretized dynamics are volume-preserving and time-reversible; hence they constitute a valid MCMC sampling algorithm. The proof of volume preservation follows the general proof technique of proving that the absolute determinant of the Jacobian transformation on a step of the Markov chain is 1. The difference in the case of truncation on a convex body is that there is a reflection step involved between the update phases of the leapfrog integrator. However, as we show below, the reflection operator is volume preserving by itself when sampling from a convex body and *does not depend* on the current position of the sampler, and after the position half-update, we compute the intersection point and the corresponding unit normal vector, which we keep constant. At the same time, we do the reflection, we treat the boundary of the body as being *locally affine* at the intersection point of the sampler trajectory with the boundary (if any). Below, we present the proof of Theorem 1.

B.1.1 Proof of Theorem 1. **Volume preservation.** The result follows a similar pathway to [2, pp. 4-10]. We will prove the theorem for $d = 1$ and assuming that the domain has a boundary at $x = 1$ (here the assumption that K is bounded is not needed). The leapfrog dynamics map consists of the following two maps $\mathbb{G}_\eta, \mathbb{H}_\eta$ with

$$\mathbb{G}_\eta : \qquad \widehat{v} = v - \frac{\eta}{2} f'(x), \quad \widehat{x} = x + \eta \widehat{v} \tag{22}$$

$$\mathbb{H}_\eta : \qquad \tilde{v} = \widehat{v} - \frac{\eta}{2} f'(\widehat{x}), \quad \tilde{x} = \widehat{x} \tag{23}$$

We also define the reflection operator $\mathbb{U}_\eta$ as

$$\mathbb{U}_\eta : \qquad \widehat{v}' = -\widehat{v}, \quad \tilde{x}' = -\eta \widehat{v} + \tilde{x} \tag{24}$$

The volume preservation properties of $\mathbb{G}_\eta \circ \mathbb{H}_\eta$ have been proven analytically in [73]. The more general case that the reflective dynamics impose is the one of

$$\mathbb{G}_\eta \circ \underbrace{\mathbb{U}_\eta \circ \cdots \circ \mathbb{U}_\eta}_{\text{at most } \ell \text{ times}} \circ \mathbb{H}_\eta, \tag{25}$$

per iteration in the case of multiple boundary normals (trivially in the case of $x = 1$ being the only normal we have $\ell = 1$).

Therefore each iteration is volume preserving, so for each step the absolute value of the determinant of the transformation is

$$\left| \det \begin{pmatrix} 1 - \eta^2/2 f''(x) & \eta \\ -\eta/2 f''(x) & 1 \end{pmatrix} \cdot \prod_{i=1}^{k} \det \begin{pmatrix} 1 & -\eta \\ 0 & -1 \end{pmatrix} \cdot \det \begin{pmatrix} 1 & 0 \\ -\eta/2 f''(\tilde{x}') & 1 \end{pmatrix} \right| = 1 \tag{26}$$

for some $k \in \{0, \ldots, \ell\}$. Thus the dynamics are volume-preserving.

Time reversibility. The time reversibility of the dynamics can be proven by applying the operator sequence $\mathbb{G}_{-\eta} \circ \underbrace{\mathbb{U}_{-\eta} \circ \cdots \circ \mathbb{U}_{-\eta}}_{\text{at most } \ell \text{ times}} \circ \mathbb{H}_{-\eta}$ to the proposed state $(\tilde{x}', \tilde{v}')$ to obtain the initial state (x, v).

Multivariate case. Considering the reflection map in the multivariate case, when the body is a convex polytope, each iteration is volume preserving as proven in [2]. For the case of a convex body with smooth boundary, since the singularities have zero measure, the tangent plane on the intersection point is well-defined. Our algorithm uses the affine equation of the tangent plane to perform the reflection of the leapfrog trajectory. Thus, the proof in [2] also holds for this case.

C WARM STARTS

C.1 Proof of Theorem 2

Proof. Recall that from L-smoothness and m-strong-convexity for $x \in K$ and $y = x^*$ we have that

$$\frac{m}{2}\|x - x^*\|^2 \leq f(x) - f(x^*) \leq \frac{L}{2}\|x - x^*\|^2 \tag{27}$$

Equivalently, since $\exp(-t)$ is a decreasing function

$$0 \leq \exp\left(-\frac{L}{2}\|x - x^*\|^2\right) \lesssim \exp(-f(x)) \lesssim \exp\left(-\frac{m}{2}\|x - x^*\|^2\right) \tag{28}$$

Integrating inside K we have that

$$0 \leq \int_K \exp\left(-\frac{L}{2}\|x - x^*\|^2\right) dx \lesssim \int_K \exp(-f(x))dx \lesssim \int_K \exp\left(-\frac{m}{2}\|x - x^*\|^2\right) dx \tag{29}$$

We calculate the warmness function $\beta : K \to (0, +\infty)$

$$\beta(x) = \frac{d\mathcal{N}_K(x|x^*, 1/LI_d)}{d\Pi(x)} = \frac{\exp\left(-\frac{L}{2}\|x - x^*\|^2\right)}{\exp(-f(x))} \cdot \frac{\int_K \exp(-f(z))dz}{\int_K \exp\left(-\frac{L}{2}\|x - x^*\|^2\right) dx} = C \cdot \beta_1(x) \tag{30}$$

From (29) the above $\beta_1(x) \leq 1$ for all $x \in K$. We now need to bound the constant

$$C = \frac{\int_K \exp(-f(z))dz}{\int_K \exp\left(-\frac{L}{2}\|x - x^*\|^2\right) dx} \tag{31}$$

Let $\mathfrak{B}_1 = \mathbb{B}(x^*, r), \mathfrak{B}_2 = \mathbb{B}(x^*, R)$ be the two balls with radii $0 < r < R$ respectively such that

$$\mathfrak{B}_1 \subseteq K \subseteq \mathfrak{B}_2 \tag{32}$$

and $\gamma = R/r \geq 1$ is the sandwiching ratio. It is direct from the properties of integrals on non-negative and non-zero everywhere functions that

$$\frac{\int_{\mathfrak{B}_1} \exp(-f(z))dz}{\int_{\mathfrak{B}_2} \exp\left(-\frac{L}{2}\|x - x^*\|^2\right) dx} \leq C \leq \frac{\int_{\mathfrak{B}_2} \exp(-f(z))dz}{\int_{\mathfrak{B}_1} \exp\left(-\frac{L}{2}\|x - x^*\|^2\right) dx} \tag{33}$$

We are interested in the upper bound. Using strong-convexity again we have that

$$C \leq \frac{\int_{\mathfrak{B}_2} \exp\left(-\frac{m}{2}\|x - x^*\|^2\right) dx}{\int_{\mathfrak{B}_1} \exp\left(-\frac{L}{2}\|x - x^*\|^2\right) dx} \tag{34}$$

Doing a change of variables $u = \frac{1}{\sqrt{m}}(x - x^*)$ and $w = \frac{1}{\sqrt{L}}(x - x^*)$ where the volume elements become $du = m^{-d/2}dx$ and $dw = L^{-d/2}dx$ since the absolute values of the Jacobians of the corresponding transformations are $m^{-d/2}$ and $L^{-d/2}$ respectively, and the transformed domains are $\mathbb{B}(0, R/\sqrt{m})$ and $\mathbb{B}(0, r/\sqrt{L})$ we arrive at the fact that

$$C = \kappa^{d/2} \frac{\int_{\mathbb{B}(0,R/\sqrt{m})} \exp(-\|u\|^2/2)du}{\int_{\mathbb{B}(0,r/\sqrt{L})} \exp(-\|w\|^2/2)dw} \leq \kappa^{d/2} \left(\frac{\mathrm{erf}(R/\sqrt{m})}{\mathrm{erf}(r/\sqrt{L})} \right)^d < \kappa^{d/2} \left(\frac{1 - \exp(-2R^2/m)}{1 - \exp\left(-r^2/L\right)} \right)^{d/2} \tag{35}$$

by the well-known identity of the Gaussian integral in polar coordinates $\pi(1 - \exp(-a^2)) < \mathrm{erf}^2(a) < \pi(1 - \exp(-2a^2))$

where $\mathrm{erf}(t) = \int_{-t}^{t} \exp(-z^2)dz$ (we ignore the constant $\sqrt{\pi}/2$ in front of its official definition since we are interested in bounding a ratio of quantities involving the same constant). Using the fact that $R = \gamma r$, the relation $\exp(x) \geq 1 + x$ we have that $1 - \exp(-2R^2/m) \leq 2R^2/m = 2\gamma^2 r^2/m$. Moreover, using the Taylor series for $\exp(x) \approx 1 + x + O(x^2)$ for small x, we get that $1 - \exp(-r^2/L) \approx r^2/L + O(r^4/L^2)$. The fraction in question can be therefore shown to behave asymptotically as

$$C \leq \kappa^{d/2} \left(\frac{1 - \exp(-2\gamma^2 r^2/m)}{1 - \exp\left(-r^2/L\right)} \right)^{d/2} = O\left(\kappa^{d/2} \left(\frac{2\gamma^2 r^2/m}{r^2/L} \right)^{d/2} \right) = O((\kappa\gamma)^d) \tag{36}$$

since in the worst case the smaller ball becomes very small (hence the Taylor expansion for the denominator). The Taylor approximation error is of the type of $1/(1 + O(h)) \approx 1 - O(h)$ for small h. Hence $\beta(x) = O((\kappa\gamma)^d)$. The lower bound can be achieved when the convex body is a ball centered at the minimizer, where the bound reduces to its previous form.

□

C.2 Proxy Start

There are cases however that we do not have access to the minimizer, i.e. the minimizer is placed on an "unconvenient" place like the boundary of K, or the actual smoothness parameter L is not known and we have access to an estimate $\Lambda = (1 + \varepsilon_L)L$ for some $\varepsilon_L \geq 0$. In this case, we can use a "proxy" distribution

$$\mathcal{N}_K\left(z, \frac{1}{2\Lambda} I_d\right) \tag{37}$$

in order to start our sampler from. We assume that for some $\delta > 0$ we have $\|x^* - z\| \leq \delta$. We can then easily prove the following Lemma about the proxy start.

Lemma 1 (Proxy start). *The distribution*

$$\pi_0^{\text{proxy}} = \mathcal{N}_K\left(z, \frac{1}{2\Lambda} I_d\right) \tag{38}$$

is a $O(\gamma_z^d((1 + \varepsilon_L)\kappa)^d \exp((\Lambda + m/2)\delta^2)$-warm distribution with respect to π, where $\gamma_z = \inf_{R>r>0}\{R/r | \mathbb{B}(z, r) \subseteq K \subseteq \mathbb{B}(z, R)\}$.

Proof. By the triangle inequality (also appears in [18]) we can deduce that

$$\|x - z\|^2 \geq \frac{1}{2}\|x - x^*\|^2 - \|x^* - z\|^2 \tag{39}$$

and

$$\exp\left(-\frac{\Lambda}{2}\|x-z\|^2\right) \leq \exp\left(\frac{\Lambda}{2}\delta^2\right)\exp\left(-\frac{L}{4}\|x-x^*\|^2\right) \tag{40}$$

Also by exchange of x^* and z we can get

$$\|x-x^*\|^2 \geq \frac{1}{2}\|x-z\|^2 - \|x^*-z\|^2 \tag{41}$$

and therefore

$$\exp\left(-\frac{m}{2}\|x-x^*\|^2\right) \leq \exp\left(\frac{m}{2}\delta^2\right)\exp\left(-\frac{m}{4}\|x-z\|^2\right) \tag{42}$$

We now follow the same procedure as in Theorem 2

$$\begin{aligned}
\frac{d\mathcal{N}_K\left(z, \frac{1}{2\Lambda}I_d\right)}{d\Pi} &= \frac{\exp(-\Lambda\|x-z\|^2)}{\exp(-f(x))} \cdot \frac{\int_K \exp(-f(z))dz}{\int_K \exp(-\Lambda\|w-z\|^2)dw} \\
&\leq \exp(\Lambda\delta^2) \cdot \frac{\exp(-L/2\|x-z\|^2)}{\exp(-f(x))} \cdot \frac{\int_K \exp(-f(z))dz}{\int_K \exp(-\Lambda\|w-z\|^2)dw} \\
&\leq \exp((\Lambda+m/2)\delta^2) \cdot \frac{\int_K \exp(-m/4\|w-z\|)dw}{\int_K \exp(-\Lambda\|w-z\|^2)dw} \\
&\leq \exp((\Lambda+m/2)\delta^2) \cdot \frac{\int_{\mathbb{B}(z,R)} \exp(-m/4\|w-z\|)dw}{\int_{\mathbb{B}(z,r)} \exp(-\Lambda\|w-z\|^2)dw} \\
&= O(\gamma_z^d((1+\varepsilon_L)\kappa)^d \exp((\Lambda+m/2)\delta^2)
\end{aligned} \tag{43}$$

Where the first inequality is due to (39), the second inequality is due to (41) and the last two inequalities follow the exact same proof technique that Theorem 2 does.

□

The above lemma establishes the trade-off for moving the starting point and changing the Lipschitz constant with an over-estimate in terms of the sandwiching ratio around the proxy point z and the new condition number which is an $(2+2\varepsilon)$-factor apart from the original one. Moreover, the shifting from the minimizer position comes with an overhead of $\exp((\Lambda+m/2)\delta^2)$. Samples from these truncated normal distributions can be obtained by using the algorithm of [23].

C.3 Extra Marginal Plots

In Figure 4 we provide marginal plots for the first two marginals (x_1 and x_2), trace plots, and 2D scatter plots for the density $\pi(x) \propto \exp\left(-\frac{2\|x-x_c\|^2}{R_c^2}\right)$. We have used an initial step size of $\eta_0 = R_c/10$ and a walk length of $w = 100$.

D NUMERICAL ROBUSTNESS

Our implementation does not thoroughly handle robustness issues, at least compared to [20]. We rely on stable linear algebra procedures to ensure the numerical stability of the intersection points. To ensure that the point is always inside the body we use heuristics to (slightly) perturb it. This work always for all the experiments that we performed, but it does not have any theoretical guarantees.

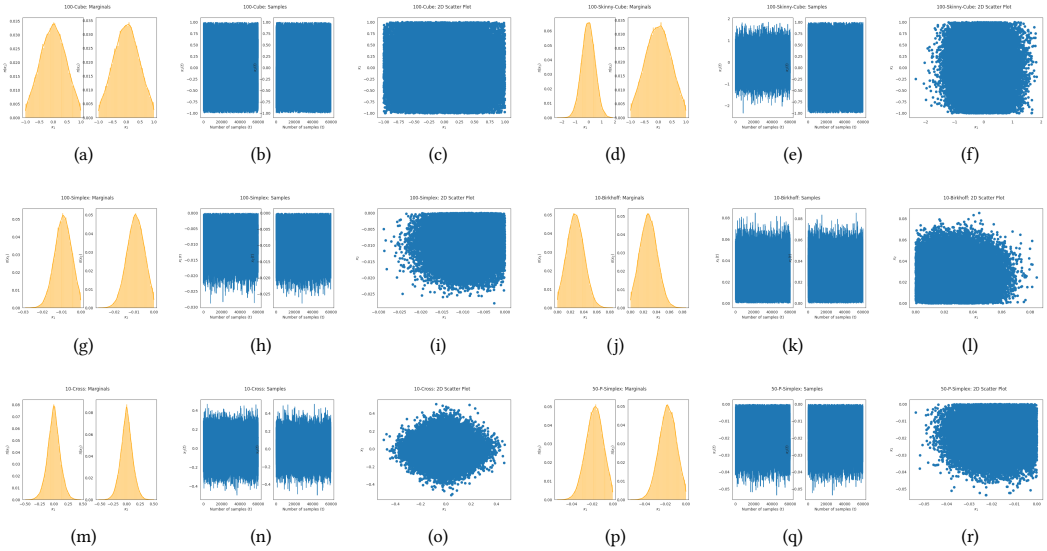

Fig. 4. Marginal, scatter, and trace plots.

E INSTRUCTIONS FOR REPRODUCING THE EXPERIMENTS

The experiments presented in Table 1 and Table 2 can be reproduced as follows:

(1) Install the software and its dependencies by following the instructions located in the README file of the codebase https://github.com/GeomScale/volesti/tree/v1.1.4.2.

(2) Compile the tests (with MKL) by running `cd test && cmake . -DUSE_MKL=ON`. MKL can be disabled by setting `-DUSE_MKL=OFF`.

(3) *Download Data.* Download the data from https://doi.org/10.5281/zenodo.5963904.

(4) *Standard & Metabolic Polytopes.* Place the data of the metabolic polytopes in `test/metabolic_full_dim` where `polytope.ine` is the polytope in `.ine` format (see the README file for the format description). Then run the tests with `cd test && ./logconcave_sampling_test -tc=benchmark_polytopes_grid_search`. To run the exponential sampling: `cd test && ./logconcave_sampling_test -tc=exponential_biomass_sampling`.

(5) *Spectrahedra.* The path `test/spectra_data` contains data about the spectrahedra. Run the spectrahedra tests with `cd test && ./logconcave_sampling_test -tc=benchmark_spectrahedra_grid_search`.

(6) *Convex Bodies.* Run the tests with `cd test && ./logconcave_sampling_test -tc=benchmark_convex_body`.