Improved polytope volume calculations based on Hamiltonian Monte Carlo with boundary reflections and sweet arithmetics

Augustin Chevallier, Sylvain Pion, Frédéric Cazals

Improved polytope volume calculations based on Hamiltonian Monte Carlo with boundary reflections and sweet arithmetics

Augustin Chevallier
Inria, France
Lancaster University, UK
chevallier.augustin@gmail.com

Sylvain S. Pion
IMS (Univ. Bordeaux / Bordeaux INP / CNRS UMR 5218), France
Inria, France
Sylvain.Pion@inria.fr

Frédéric Cazals
Université Côte d'Azur, France,
Inria, France
Frederic.Cazals@inria.fr

Abstract

Computing the volume of a high dimensional polytope is a fundamental problem in geometry, also connected to the calculation of densities of states in statistical physics, and a central building block of such algorithms is the method used to sample a target probability distribution.

This paper studies Hamiltonian Monte Carlo (HMC) with reflections on the boundary of a domain, providing an enhanced alternative to Hit-and-run (HAR) to sample a target distribution restricted to the polytope. We make three contributions. First, we provide a convergence bound, paving the way to more precise mixing time analysis. Second, we present a robust implementation based on multi-precision arithmetic – a mandatory ingredient to guarantee exact predicates and robust constructions. We however allow controlled failures to happen, introducing the *Sweeten Exact Geometric Computing* (SEGC) paradigm. Third, we use our HMC random walk to perform H-polytope volume calculations, using it as an alternative to HAR within the volume algorithm by Cousins and Vempala. The tests, conducted up to dimension 50, show that the HMC random walk outperforms HAR.

2012 ACM Subject Classification CCS → Theory of computation → Randomness, geometry and discrete structures → Computational geometry

Keywords and phrases Polytope volume, randomized algorithms, multi-phase Monte Carlo, Hamiltonian Monte Carlo, numerics, multi-precision arithmetic, exact predicates

Digital Object Identifier 10.4230/LIPIcs.SoCG.2021.yy

1 Introduction

1.1 Polytope volume calculations and related problems

Volume calculations. Computing the volume of a polytope – a bounded region of $\mathbb{R}^n$ defined by the intersection of a fixed set of half spaces (H-polytope) or the convex hull of vertices (V-polytope), is a classical problem in science and engineering. Complexity-wise, the problem is #-P hard irrespective of the representation of the polytope (H-polytope or V-polytope) [15]. This observation naturally calls for approximation algorithms [5, 28] delivering (ε, δ) approximations. Over the years, the complexity of volume calculation algorithms, measured by the number of calls to the oracle stating whether a point is inside the polytope, has been lowered from $O^\star(n^{23})$ [16] to $O^\star(n^4)$ [24], and $O^\star(n^3)$ [12], the

latter for well rounded bodies. More recently, the complexity $O^\star(hn^{2/3}hn^{\omega-1})$ has been established [26], with h the number of hyperplanes and ω the matrix multiplication exponent. The reader is referred to [26] for the full history. Interestingly, recent volume calculation algorithms are of the *multi-phase Monte-Carlo* type, an iterative strategy where each step benefits from the *progress* made at the previous step. Sketchily, the volume calculation boils down to estimating ratios in a telescoping product, each ratio being the integral over the convex of functions carefully chosen according to a *cooling schedule*. In recent algorithms, exponential functions are used [12, 14]. We also note that recent work has focused on the reduction of the cooling schedule size, using statistical tests to bound the aforementioned successive ratios [10]. The complexity of the algorithm therefore depends on (i) the number of functions in the cooling schedule, (ii) the number of points sampled at each step, and (iii) the complexity of generating a sample according to the specified distribution.

1.2 Sampling a target distribution in a bounded domain

Volume calculations require an algorithm to sample a target distribution π in a bounded domain. We describe such algorithms, providing mixing times for the uniform distribution–for the sake of conciseness–from a warm start.

General idea: MCMC. In non trivial cases, the default strategy is to build a Markov Chain leaving the target distribution π invariant. Under mild additional conditions, computing integrals of a function with respect to the target distribution can be approximated by averaging the function on the samples [7]. This method is called Markov Chain Monte Carlo (MCMC). The convergence of an MCMC schemed is usually assessed by mixing properties, e.g. based on how fast the Markov chains converges to its stationary distribution [27].

Hit-and-run and ball walk. In the case of polytopes, two important Markov chains have been introduced in the literature: Hit and Run (HAR) [29, 31, 32, 22] and Ball Walk [30]. These two Markov chains uses different strategies to stay inside the polytope: ball walk samples within a ball and rejects points outside of the polytope, while Hit and Run only proposes inside the polytope–no rejection step needed. Furthermore, HAR is amenable to several optimizations [17, 18], including the choice of the random line used, and the calculation of the facet of the polytope intersected by a line. In the context of polytope volume computation, upper bound for the mixing times of the uniform distribution from a warm start exists: HAR mixes in $O^\star(n^3)$ steps while ball-walk mixes in $O^\star(n^{2.5})$ steps [26, 32, 25].

Hamiltonian Monte Carlo. An efficient sampler is Hamiltonian Monte Carlo (HMC) [6, 39]. HMC relies on the measure preserving properties of hamiltonian flows on phase space to build a Markov Chain leaving π invariant. In a nutshell, a HMC works by adding a velocity/momentum and an HMC step involves three sub-steps which are (i) picking a random velocity p, (ii) following the Hamiltonian flow associated to $H(q,p) = \nabla \log \pi(q) + \frac{1}{2}\|p\|^2$ for a fixed time, and (iii) projecting down in position space. As seen from Hamilton's equation, the fact that the gradient of the target density is used to twist the momentum p rather than the position q helps forcing the dynamical system to be *glide* across the typical set [6], which is useful to deal with concentration phenomena. HMC generally uses a Metropolis step when a numerical integration is required for the flow of the Hamiltonian. In our case, we bypass this difficulty by using analytical trajectories.

Riemannian HMC. RHMC uses a Hamiltonian exploiting a metric defined on the domain of interest [20]. For polytopes, a metric based on the Hessian of the barrier function $(\phi(x) = -\sum_{i=1,\dots,d} \log({a_i}^\mathsf{T} x - b_i))$ can be used to constrain the random walk (RW) within

the polytope [3]. This strategy has been used to sample polytopes and compute their volume [26], which yields $O^\star(hn^{2/3})$ as mixing time in the context of polytope volume computation, with h the number of hyperplanes [26]. However, the Hessian of ϕ is ill-conditioned near barriers [41], and we are not aware of any implementation for polytope volume calculations.

Dynamical billiard. Random walks in polytopes are also connected to billiards. In two dimensions, it has been proven that there is a G_δ dense subset of ergodic billiards in the set of all possible billiards [33]. For those billiards, Birkhoff theorem implies that almost every trajectory is uniformly distributed on the phase space of the polygone. While analogous properties are unknown for billiards in dimension $n > 2$ [4], it has been conjectured that billiard trajectories provide valuable building blocks for random walks sampling uniform distributions.

Billiard walk and billiard HMC. A step of billiard walk consists of choosing a direction at random, and following the corresponding billiard trajectory for a fixed time. Properties of billiard trajectories, including their ability to escape from corners, have motivated the sampling algorithm from [37] for general n-dimensional domains, with the velocity refresh ensuring ergodicity. This work is however limited to uniform distributions. In the case of polytopes, billiard walk can be seen as a special case of HMC with reflections on boundaries [2], a strategy used in Bayesian statistics to restrict the state space. We prove the uniform ergodicity of billiard HMC (Thm 9, Sect. 2), and experimentally assess its efficiency for polytope volume estimation (Sections 3 and 4).

1.3 Robustness issues and the SEGC paradigm

It has long been known that geometric algorithms are prone to (almost) degenerate situations, which manifest even on the simplest expressions in 2D [23]. The design of robust geometric algorithms can be done in a general way using the Exact Geometric Computation paradigm [42], as it is done for example in the CGAL [1] software library. In order to do so, the CGAL kernel distinguishes between predicates and constructions. However, the EGC paradigm faces limitations of theoretical and practical nature. The former refer to the impossibility to provably determine the sign (positive, negative or zero) of some real functions in finite time in all cases. The latter relate to the added complexity which can cause the use of resources like processing time and memory space to hit limits.

Without compromising on the robustness properties of the EGC paradigm, we simply allow controlled failures by allowing the sign functions to return a fourth possible value meaning *I can't compute*. This generalization can practically be implemented using (C++) exception handling mechanisms. We name this extended paradigm Sweeten Exact Computing Paradigm (SEGC). And we refer to *sweet arithmetics* to qualify the kind of arithmetic functions libraries complying with it. In our implementation, we use the `iRRAM` C++ library [35], which provides iterative multiple precision computations for many real functions. This library does not fit into the original EGC paradigm definition, as the zero determination cannot be decided using algebraic separation bounds.

1.4 Contributions

This paper makes contributions touching upon random walks for MCMC algorithms, polytope volume calculations, and Hamiltonian Monte Carlo. More precisely:

1. In section 2, we present a robust version of billiard HMC, and prove its uniform ergodicity for convex bodies.
2. In section 3, we instantiate our random walk to sample distributions used for H-polytope volume calculations. Exploiting analytical expressions for HMC trajectories, we provide a robust implementation of the random walk based on multi-precision interval arithmetic.
3. Finally, section 4 reports experiments comparing HAR and our random walk. The first test samples a target distribution. The second one embeds our random walk into the practical polytope volume calculation of Cousins and Vempala [13]. In both cases, we show superior performances over HAR, for dimension up to $n = 50$.

It is important to note that Riemannian HMC [26] forces the random walk to stay within the domain of interest. This is achieved via a distorted metric based on the barrier function, which also yields ill conditioned numerics. We instead use billiard HMC, and control that the RW remains in the polytope using exact predicates based on multiprecision.

Notations. To conform with previous work, the following notations are used in this paper: (i) ε: criterion used to assess the quality of the volume approximation [13]; (ii) ϵ: notation used for the total variation bound on the mixing time. See theorems 9 and 12. The total variation norm is denoted $\|\cdot\|_{TV}$.

Proofs and pseudo-code. The reader is referred to the appendix.

2 Billiard Hamiltonian Monte Carlo

Consider a bounded open set $Q \subset \mathbb{R}^n$ with piecewise smooth boundary and a target probability measure with density $\pi : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ such that $\pi(x) = 0$ for all x not in the closure $\overline{Q}$ of Q.

2.1 Billiard HMC

Denoting $q^{(i)}$ and $p^{(i)}$ the i-th coordinates of position and momentum respectively, recall Hamilton's equations which state that the velocity field is orthogonal to the gradient of the Hamiltonian H:

$$\frac{dq^{(i)}}{dt} = \frac{\partial H}{\partial p^{(i)}}, \frac{dp^{(i)}}{dt} = -\frac{\partial H}{\partial q^{(i)}}. \tag{1}$$

As with HMC, we define a potential energy $U(q) = -\log(\pi(q))$ and the Hamiltonian $H(q,p) = U(q) + \frac{1}{2}\|p\|^2$ but this time restricted to $\Gamma = \overline{Q} \times \mathbb{R}^n$. We assume that π is the restriction to $\overline{Q}$ of positive smooth function defined on $\mathbb{R}^n$ and we use Φ_t the Hamiltonian flow. However, the trajectories of this flow are not included in $\overline{Q}$ even if the initial point (q,p) is in $Q \times \mathbb{R}^n$. For every $q \in Q$ and every $p \in \mathbb{R}^n$, we define $T(q,p)$ as the largest T such that for all $0 \leq t < T$, $\Phi_t(q,p) \in Q$. We also define $T(q,p) = 0$ when q is in the boundary of Q.

Following [2] and [21, 37], we modify the flow by forcing reflections on the boundary of Q. This flow, illustrated on Fig. 1, is denoted as follow:

$$\begin{cases} \tilde{\Phi}_t : \overline{Q} \times \mathbb{R}^n \to Q \times \mathbb{R}^n \\ (q,p) \mapsto \tilde{\Phi}_t(q,p) \equiv (\tilde{\Phi}_t^{(q)}(q,p), \tilde{\Phi}_t^{(p)}(q,p)). \end{cases} \tag{2}$$

Note that the latter equation defines position and velocity upon applying the flow. However, as noted in [21, 37], this new flow might exhibit problematic trajectories: some of them might not be defined for all t because of singularities on the boundary, others might have

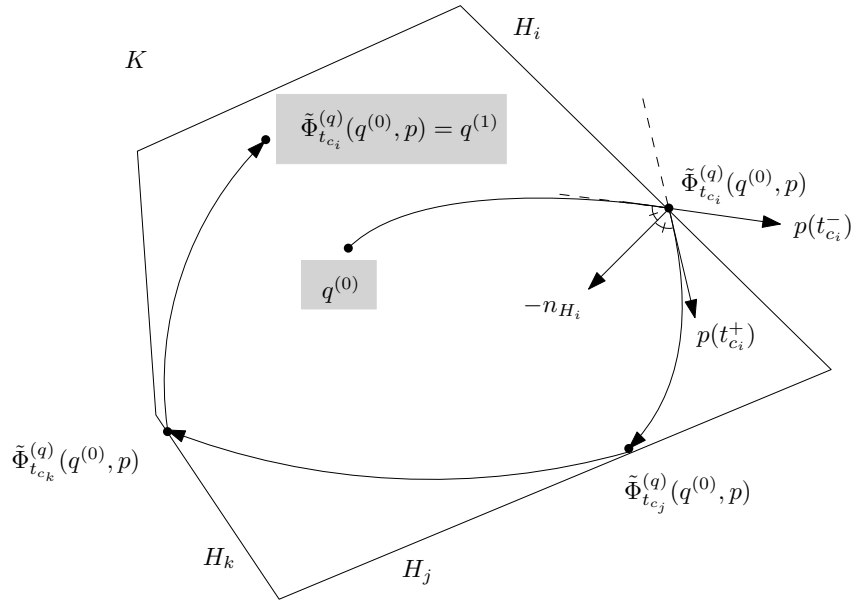

Figure 1 One HMC step starting at $q^{(0)}$, with reflections of the HMC trajectory on the boundary of the polytope K. The trajectory successively reflects on hyperplanes, before stopping at $q^{(n_L)}$. The normal to a facet is denoted n_H.

huge number of reflections or even an infinite number of reflections in finite time. This does not happen in 2D [11, Section 2.4], but we are not aware of any proof when $n > 2$ or the trajectories are curved. Hence, we introduce the upper bound $M > 0$ for the maximum number of reflections, and cull problematic trajectories accordingly.

▶ Remark 1. For $q \in Q$ and any p, $T(q,p) > 0$ and $\tilde{\Phi}_{T(q,p)}(q,p)$ is in the boundary of Q.

▶ Remark 2. When q is in the boundary of Q and $t > 0$, $\tilde{\Phi}_t(q,p)$ is only defined for the momenta p such that the open half-line with origin q and direction p, is included in Q in a neighborhood of q.

Algorithm. Let $M \in \mathbb{N}^*$ be the maximum number of reflections allowed. Given a point $q^{(t)} \in Q$, the algorithm is as follow (Algorithm 1):

Algorithm 1 Hamiltonian Monte Carlo with reflections

- (Step 1) Choose the traveling time $L \sim \text{unif}(0,1)$
- (Step 2) Pick the momentum $p \sim \mathcal{N}(0, I_n)$
- (Step 3) If the flow $\tilde{\Phi}_L(q^{(t)},p)$ is defined and does not involve more than M reflections between $t = 0$ and L, and if $\tilde{\Phi}_L(q^{(t)},p) \in Q$
  - Take $q^{(t+1)} = \tilde{\Phi}^{(q)}_L(q^{(t)},p)$
  - Else, take $q^{(t+1)} = q^{(t)}$

For a fixed $L > 0$, steps from 2. and 3. define a Markov kernel $P_{\pi,L}$. The full algorithm (steps from 1. to 3.) define a Markov kernel P_π that can be expressed with $P_{\pi,L}$.

For $L > 0$, let Γ_L be the largest subset of Γ where $\tilde{\Phi}_L$ is defined; this set admits no more than M reflections, and the trajectory does not finish in a singularity (a point of the boundary where the normal is not defined) at time L. Γ_L is open and therefore measurable.

Let

$$\bar{\Phi}(q,p) = \begin{cases} \tilde{\Phi}_L(q,p) & \text{if } (q,p) \in \Gamma_L \\ (q,p) & \text{if } (q,p) \notin \Gamma_L \end{cases}$$

the application from Γ to Γ which corresponds to steps 3 and 4.

▶ Remark 3. For any point $(q,p) \in \Gamma$, if L is small enough, $(q,p) \in \Gamma_L$. However, if L is too large, Γ_L could be empty.

2.2 Measure invariance via detailed balance

We recall the definition of detailed balance:

▶ **Definition 4** (detailed balance). *A Markov chain P is said to satisfy detailed balance (or reversibility) with respect to the measure π if for every A and B measurable subsets,*

$$\int_B P(x,A)\pi(dx) = \int_A P(x,B)\pi(dx).$$

To prove detailed balance for P_π, we establish the following intermediate result:

▶ **Theorem 5.** *For a fixed time L, we consider the Markov kernel $P_{\pi,L}$ associated to steps 2 and 3 of Algorithm 1. Then $P_{\pi,L}$ satisfies detailed balance with respect to π.*

From which one derives the invariance of P_π:

▶ **Theorem 6.** *The Markov kernel P_π associated to Algorithm 1 satisfies detailed balance with respect to π.*

It is well known that satisfying detailed balance implies the invariance of the measure.

2.3 Convergence result

Detailed balance ensures that the Markov kernel P_π leaves π invariant, but it does not imply the convergence of P_π^n to π by itself, with P_π^n is n-times iterated of P_π. In this section, we prove uniform ergodicity to get such a convergence result. This requires extra assumptions on Q.

Definition 7 is taken from [40] with P is a Markov kernel. Practically, we will use $P = P_\pi$.

▶ **Definition 7.** *A subset $C \subset \mathcal{X}$ is small (or, (n_0,ϵ,ν)-small) if there exists a positive integer n_0, a real $\epsilon > 0$, and a probability measure $\nu(.)$ on $\mathcal{X}$ such that the following minorisation condition holds:*

$$P^{n_0}(x,.) \geq \epsilon\nu(.) \quad x \in C$$

i.e. $P^{n_0}(x,A) \geq \epsilon\nu(A)$ for all $x \in C$ and all measurable $A \subset \mathcal{X}$

Intuitively, the previous definition states that whatever the starting point x–whence the adjective small, the iterated kernels $P^{n_0}(x,.)$ cover a common measure $\nu(.)$. Note that the value of n_0 is the number of steps needed to reach ν – and can be equal to one. The following theorem provides a sufficient condition for Q to be small with respect to P_π:

▶ **Lemma 8.** *If Q if convex and the gradient of the potential energy ∇U is bounded on Q, then Q is small for P_π.*

The ergodicity of P_π follows from the above Lemma 8, applied to Theorem 8 from [40]:

▶ **Theorem 9.** *If Q if convex and the gradient of the potential energy ∇U is bounded on Q, then P_π is uniformly ergodic, that is, for all $x \in Q$:*

$$\|P_\pi^n(x,.) - \pi(.)\|_{TV} \leq (1-\epsilon)^{\lfloor n/n0 \rfloor} \tag{3}$$

3 Application: computing the volume of a polytope

We now specialize Algorithm 1 and use it as a building block for polytope volume calculation from [13]. The general domain Q of Sect. 2 is now the polytope K.

3.1 Volume algorithm

Consider a polytope defined in matrix form by $Ax \leq b$. The algorithm used in [13] computes the volume of such a polytope with target relative error ε. The principle is a multi-phase Monte Carlo computation, which splits the calculation into m steps. Let $\{f_0, \ldots, f_{m-1}\}$ be m isotropic Gaussian distributions i.e. $f_i(x) = \exp(-\|x\|^2/(2\sigma_i^2))$ with $\sigma_i = 1/\sqrt{2a_i}$, or equivalently $f_i(x) = \exp(-a_i\|x\|^2)$, such that the first one is highly concentrated around a point deep inside the convex, and the last one is an almost flat distribution. The volume calculation reduces to computing the telescoping product

$$\text{Vol}(K) = \int_K f_0(x)dx \frac{\int_K f_1(x)dx}{\int_K f_0(x)dx} \cdots \frac{\int_K dx}{\int_K f_{m-1}(x)dx} \equiv \int_K f_0(x)dx \prod_{i=1,\ldots,m} R_i \tag{4}$$

Each of these ratio are estimated using Monte-Carlo integration with respect to the density

$$\pi_i(x) = \frac{f_i(x)}{\int_K f_i(y)dy} 1_K(x) \tag{5}$$

via importance sampling. The method then uses as core block an algorithm sampling the previous distribution, usually using HAR. In our case, HMC with reflections will be used instead. For $X_1, ..., X_k$ consecutive points given by the random walk, the Monte-Carlo estimation R_i^k is given by:

$$R_i^{(k)} = \frac{1}{k} \sum_{j=1}^{k} \frac{f_i(X_j)}{f_{i-1}(X_j)}. \tag{6}$$

The following stop criterion is introduced in [13]. Let $\varepsilon' = \varepsilon/\sqrt{m}$; this is the relative ratio error allocated for each ratio R_i estimation. Consider a sliding window of size $W(= 4n^2+500)$. When W consecutive estimated ratios $R_i^{(k-W+1)}, ..., R_i^{(k)}$ are within $\varepsilon'/2$, the convergence for R_i is declared.

3.2 HMC algorithm

Our HMC specialization has analytical trajectories – see also [36], whose intersection with the polytope have simple expressions.

Analytical trajectories. We build an HMC sampler for $\pi_i(x)$ from Eq. 5. Let $U(q) = \log(\exp(-a_i||q||^2)) = -a_i||q||^2$ and $H(q,p) = U(q) + 1/2\|p\|^2$. Note that the normalization constant $\frac{1}{\int_K f_i(y)dy}$ was discarded because it does not change the trajectory (its gradient

is 0). Rewriting the dynamical system associated to this Hamiltonian yields the following differential equations:

$$\frac{d^2q_j}{dt^2}(t) = -2a_iq_j(t) \text{ for } j \leq n. \tag{7}$$

Each coordinate is independent and has a solution of the form

$$q_j(t) = C_j \cos(wt + \phi_j) \tag{8}$$

With $C_j, w, \phi_j \in \mathbb{R}$. The parameter Φ_j satisfies the following 2 equations:

$$\cos(\phi_j) = \frac{q_j(0)}{C_j}, \sin(\phi_j) = \frac{-p_j(0)}{\omega C_j}. \tag{9}$$

Thus we deduce:

$$\begin{cases} \omega & = \sqrt{2a_i} \\ C_j & = \sqrt{q_j(0)^2 + p_j(0)^2/w^2} \\ \phi_j & = \arctan\left(-\frac{p_j(0)}{q_j(0)\omega}\right) + 1_{\{q_j(0)<0\}}\pi \end{cases} \tag{10}$$

Note that these equations are the same for any choice of coordinates as long as the basis is orthonormal–an observation exploited below.

Collision with convex boundary. To restrict the sampling algorithm to the convex K, trajectories should reflect on the boundary of K. The convex K is defined by a set of hyperplanes. Hence, we need to compute the intersection time of a trajectory with each hyperplane and take the smallest time. Thankfully, there is an analytical expression. The hyperplanes are defined by a matrix A and a vector b, and hyperplane i is defined by the equation $(Ax)_i = b_i$.

To compute the collision time with an hyperplane H, we make the following remark: let n_H be the normal of the hyperplane. We can complete n_H to an orthonormal basis. In this basis, the collision time depends only on what happens for the coordinate on n_H. Consider $q_{n_H}(t) =< q(t), n_h >$ and $p_{n_H}(t) =< p(t), n_H >$ the coordinates along the normal of the hyperplane. Let ω_{n_H},C_{n_H} and ϕ_{n_H} be the parameters of the trajectory along direction n_H. Finding the intersection times is equivalent to solving the equation for t:

$$C_{n_H} \cos(\omega_{n_H} t + \phi_{n_H}) = b_i \tag{11}$$

We deduce that if $|C_{n_H}| < b_i$, there is no solution, and else, the following times are solution:

$$\begin{cases} t_1 & = \left(\arccos(b_i/C_{n_H}) - \phi_{n_H}\right)/\omega_{n_H}. \\ t_2 & = \left(-\arccos(b_i/C_{n_H}) - \phi_{n_H}\right)/\omega_{n_H}. \end{cases} \tag{12}$$

One solution corresponds to the entry into K, while the other corresponds to the exit out of K. We select the exit trajectory via a dot product between the velocity at t_1 and t_2 and the outward normal n_H. In the sequel, the corresponding value is denoted t_c for time of collision.

3.3 Travel time choice with respect to a_i

Algorithm 1 can be slightly generalized by choosing a travel time L uniformly in $[0, L_{max}]$ instead of $[0, 1]$. We propose here a strategy to chose L_{max} with respect to the parameter a_i of the Gaussian sampled ($\sigma_i = 1/\sqrt{2a_i}$).

We distinguish two main cases here. First, if a_i is large, then the probability measure is concentrated in a neighborhood of 0, and the trajectories will seldom hit the boundaries and the strategy is largely unaffected by the convex. Second, if a_i is close to 0, the distribution is close the uniform distribution of the convex.

Case 1. If a_i is large, we consider $q_j(t)$ a solution of the dynamical system 7 and we introduce space-time rescaling for the trajectory:

$$\bar{q}_j(t) = \sqrt{a_i} q_j(t/\sqrt{a_i}).$$

It satisfies Eq. 7 with $a_i = 1$:

$$\frac{d^2\bar{q}_j}{dt^2}(t) = \bar{q}_j(t)$$

with initial condition

$$\frac{d\bar{q}_j}{dt}(0) = \frac{dq_j}{dt}(0) \sim \mathcal{N}(0,1). \tag{13}$$

It follows that in the absence of convex boundaries, the rescaled process $\bar{q}$ is sampling the distribution associated to $a_i = 1$, i.e. the standard normal distribution. Therefore, any sensible value for $L_{max}(1)$ taken for $a_i = 1$ should be scaled as $L_{max}(a_i) = L_{max}(1)/\sqrt{a_i}$.

Case 2. When a_i goes to 0, the previous strategy leads $L_{max}^{a_i} \to \infty$. We argue that in this case, the trajectories converges to billiard trajectories with reflections on the boundary and that the optimal L_{max} should be therefore close to the optimal L_{max} for $a_i = 0$.

Values used in experiments. We chose the following L_{max}:

$$\begin{cases} L_{max}(a_i) & = 1/\sqrt{a_i} \quad \text{if } a_i > 1 \\ & = 1 \quad\quad\quad\quad \text{if } a_i \leq 1 \end{cases} \tag{14}$$

▶ Remark 10. Further work is required to incorporate into Eq. 14 parameters depending on the geometry and scaling of the polytope. To the best of our knowledge, the optimal value for L_{max} for billiard trajectories is not known.

3.4 HMC implementation based on interval arithmetic

3.4.1 Robustness issues

The algorithm as stated before is prone to numerical rounding errors. As a particular case, one may consider the situation where rounding errors would be such that the point computed on the HMC trajectory would be outside the convex. More generally, all geometric constructions and geometric predicates on them potentially raise robustness issues – see list in section 3.4.3.

As discussed in Introduction, we guarantee robustness using the SEGC paradigm. More specifically, recall that efficient arithmetic operations usually combine two ingredients: first, an interval representation of the numbers, as non overlapping intervals yield exact predicates; second, an arbitrary precision representation of the interval bounds, as precision can be increased so as to yield exact predicates and constructions of controlled accuracy. In the sequel, we use the `iRRAM` library which provides these two ingredients [35].

For the sake of clarity, all functions for which the `iRRAM` library plays a key role are highlighted in blue.

3.4.2 iRRAM and used features

We represent points as n-dimensional points whose coordinates are of the iRRAM number type. In iRRAM, a real number is represented by two types of data: firstly a symbolic representation memorizing the way it was defined (type of function and pointers to the operands), and secondly a numeric approximation using an interval with rational endpoints guaranteed to enclose the exact real value. The accuracy of the latter interval can be increased if needed, by recomputing it using recursively increased precision of the operands intervals that defines it. Therefore, the iRRAM encoding $q^{\mathrm{iRRAM}}(t)$ of the position $q(t)$ of the HMC trajectory is numerically represented by a n-dimensional box certified to contain the exact real position.

Two specific operations of iRRAM may trigger numerical refinement:

- operator $x < y$: predicate answering the $<$ comparison operator. If the interval representations of x and y overlap, these intervals are automatically refined, a feature called *precision refinement* thereafter.
- Near_inf_double(iRRAM x) : returns the nearest double $<$ the iRRAM number x.

3.4.3 Robust operations

Our robust implementation calls for the following operations (i) Computing the trajectory parameters – Eq. 10, (ii) Finding the exit intersection time – Eq. 12, (iii) Finding the smallest exit intersection time amongst all hyperplanes, and (iv) Evolving the trajectory. In the sequel, we detail these operations, and refer the reader to the SI section B; in particular, algorithm 3 refines Algorithm 1 based on these robust primitives.

Trajectory parameters. Following Eq. 10, we construct the numbers C_j, w and ϕ_j as iRRAM number types. See Algorithm 4.

Exit intersection time t_c with one hyperplane. Intersecting the trajectory with a hyperplane yields two solutions (Eq. 12) respectively exiting and entering the convex. The collision time t_c corresponds to the former. Assuming that the normal vector to the hyperplane is oriented outwards, the value t_c is such that $< p(t_c), n_H >> 0$. The evaluation of this predicate triggers a precision refinement if needed.

Smallest exit intersection time. Since the boundary of K involves several hyperplanes, the nearest one, which corresponds to the smallest exit time, must be determined. To do so, we first construct the intersection time t_{c_i} with respect to each hyperplane. Then we compute $t_c = \min_i t_{c_i}$.

This calculation is tantamount to sorting the individual intersection times, which in turn requires the comparison operator $<$. iRRAM provides such an operator, which triggers precision refinement if needed. See Algorithm 5.

▶ Remark 11. We note that in case the trajectory would hit a face of dimension $< d-1$, an equality between exit times occurs. The absence of separation bound in iRRAM does not allow us to handle such cases, and an infinite refinement loop is entered. However, for each starting point, the measure of velocities leading to such sets is null. Practically, such a case was never faced–as expected.

The Is_strictly_in_convex(q) predicate. To constrain the trajectory within the convex, we resort to a predicate telling whether a given position q belongs to the interior K° of K. This predicate, denoted Is_strictly_in_convex(iRRAM_point_d p), checks $< op, n > < b_i$ holds for every hyperplane, and triggers the iRRAM refinement if needed so. (Nb: $< \cdot, \cdot >$ stands for the dot product of two vectors.)

Performing one HMC step. Equipped with the previous operations, our robust implementation – Algorithm 3, hinges on two operations:

- Calling the predicate Is_strictly_in_convex($q(t_c^<)$). Recall that in `iRRAM`, a n-dimensional point is represented as a box. This is in particular the case for the collision points with the hyperplanes, and for the final point returned. To ensure that all such points are strictly within the convex, we call the aforementioned Is_strictly_in_convex() .
- Computing the nearest inferior double $t_c^< =$ Near_inf_double(t_c). The intersection point between the trajectory and a hyperplane is defined analytically – Eq. 12. The `iRRAM` representation of the collision time is an interval certified to contain the exact solution, and the corresponding n-dimensional point $q^{\text{iRRAM}}(t_c^<)$ is represented as a box. We note that the box $q^{\text{iRRAM}}(t_c^<)$ intersects the interior K^{o} of the convex K. Indeed:
  - the exact collision point $q(t_c)$ lies on its defining hyperplane i.e. $q(t_c) \in H_i$
  - by definition of $t_c^<$, the exact embedding $q(t_c^<)$ satisfies $q(t_c^<) \in K^{\text{o}}$
  - the `iRRAM` box $q^{\text{iRRAM}}(t_c^<)$ corresponding to $t_c^<$ intersects K^{o} since

$$q^{\text{iRRAM}}(t_c^<) \ni q(t_c^<) \in K^{\text{o}}$$

3.5 Cube: HMC mixing time is $O(\log n)$

We make a small digression in the case of the cube. It turns out that in this special case, the mixing time of the HMC random walk is $O(\log n)$ with n the dimension, and the average complexity per step (the average number of calls to the oracle per step) is linear with the dimension. We assume without any loss of generality that the cube is $[0,1]^n$. Rigorously:

▶ **Theorem 12.** *Let $P_k^{(n)}$ the distribution after k steps in dimension n and g_n an isotropic Gaussian of parameter a_i restricted to $[0,1]^n$. Then there exists $0 < \rho < 1$ such that for every $\epsilon > 0$, every $n > 1$ and every $x \in \mathbb{R}^n$, we have for $k \geq (\log n - \log \epsilon)/(\log 1/\rho)$:*

$$\|P_k^{(n)}(x,.) - g_n\|_{TV} \leq \epsilon. \tag{15}$$

Furthermore, the average number of reflections per step is $O(n)$.

4 Experiments

4.1 Implementation and code availability

Implementation. The implementation of our algorithm for the particular case of a convex polytope $Q = K$ is provided in the Structural Bioinformatics Library (SBL, `http://sbl.inria.fr`), a state-of-the-art environment targeting molecular simulation at large [9]. The corresponding package, Hamiltonian_Monte_Carlo is ascribed to the *Core / Geometry-Topology* component of the library. The user manual of the package, as well as a jupyter notebook illustrating the main functionalities, can be accessed from `https://sbl.inria.fr/doc/Hamiltonian_Monte_Carlo-user-manual.html`.

Robustness. As explained in section 3.4.2, the main number type used to ensure robustness is the `iRRAM` number type. However, for the particular case of a polytope, we also need to check that the starting point belongs to the interior of K. (For a starting point on the boundary, one would have to check that the initial velocity is such that the trajectory enters the interior of the convex.) Since `iRRAM` does not have separation bounds to decide equality to zero to perform this initial check, we instantiate the predicate Is_strictly_in_convex using the number type `CGAL::Lazy_exact_nt<CGAL::Quotient<CGAL::MP_Float»` [19].

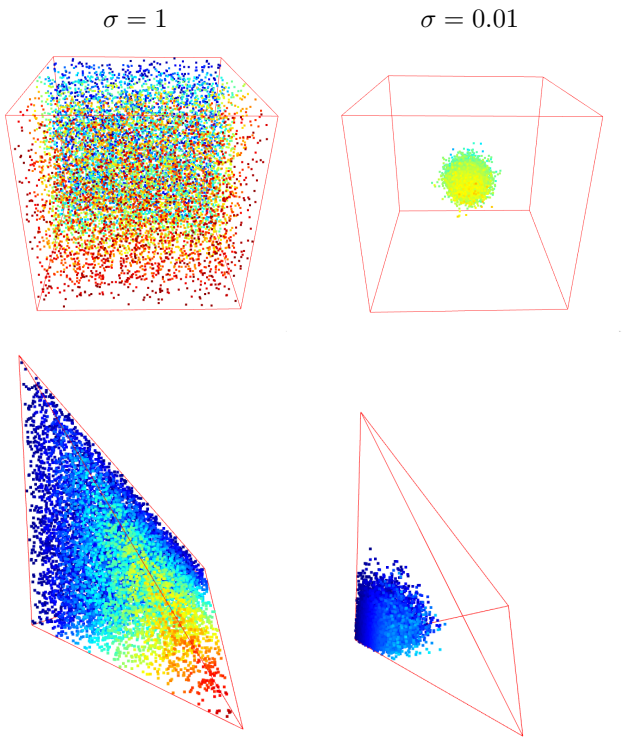

Figure 2 Using HMC to sample a Gaussian of standard deviation σ restricted to the cube $[-1, 1]^3$ and the standard simplex. In all cases, $N = 10,000$ samples. (Color gradient defined w.r.t. a coordinate value.)

Having started strictly inside the convex, with respect to the SEGC paradigm (Sec. 1.3), we never faced a situation *I can't compute*.

We also stress in passing that from dimension 10 onward, repeated runs using doubles systematically yield a significant fraction of points outside the convex, leading to crashes.

Volume calculations. As described in section 3, we also embed our random walk in the framework of [13]. We reuse the MATLAB code provided by [13] and adapt it so as to call our HMC random walk instead of the usual HAR random walk. Also following [13], we use HAR (rather than ball walk) as contender.

4.2 Illustrations of the HMC random walk

Our first illustration features samples generated by HMC for the cube $[-1, 1]^3$ and the standard simplex $\sum x_i \leq 1$ in dimension three. Varying the parameter σ rapidly yields concentrated samples (Fig. 2). Our second illustration shows the ability of the algorithm to escape corners. As opposed to HAR, HMC yields an almost uniform distribution after 10 steps even when the dimension increases (Fig. 3).

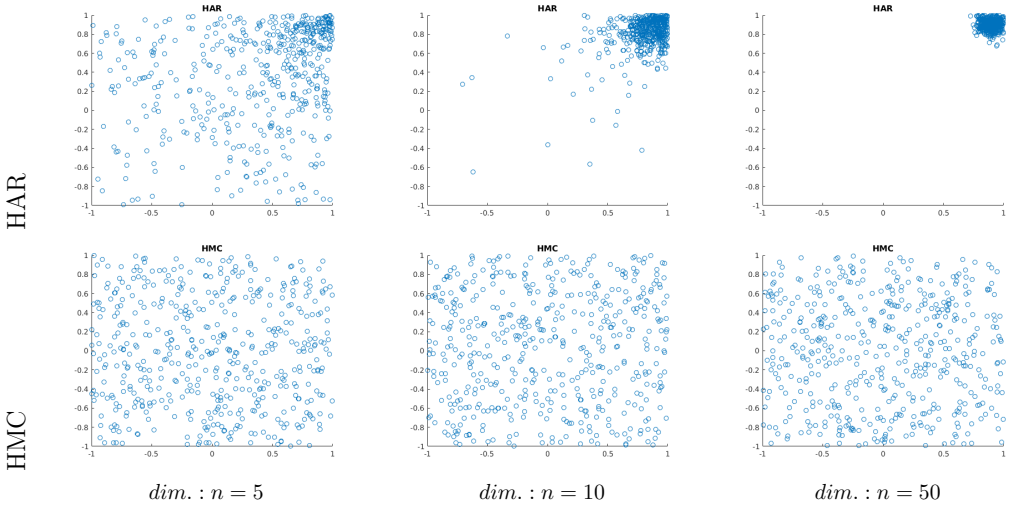

Figure 3 HMC for the cube $[-1,1]^n$: ability to escape corners, and comparison to Hit-and-run. A nearly flat isotropic Gaussian distribution was used to approach the uniform distribution ($\sigma = \sqrt{500}$). For $n > 3$, the plot displays projections onto the first two coordinates. Simulations were started from a corner ($q_i^{(0)} = 0.9$). Samples generated after 10 steps of HAR or HMC, repeated 500 times – whence 500 samples.

4.3 Analysis

4.3.1 Volume computation

We study the complexity (i.e. the number of calls to the oracle) with respect to the dimension. The MATLAB implementation of the volume computation algorithm from [13] introduced the stop criterion of Eq. 6. Intuitively, the sliding window size W should be at least as large as the mixing time of the chosen random walk. In the case of HAR, the mixing time increases with the dimension, hence the value of W chosen by [13] depends on the dimension: $W = 500 + 4n^2$. However, in the HMC case, we hope for a smaller mixing time and especially a smaller growth rate with respect to the dimension. Therefore, the growth rate of W used in [13] might be too large and impede the convergence speed for HMC. For that reason, we modified the MATLAB code to allow for different W.

Statistics. We collect the following statistics:

- the relative error of the estimated V:

$$| V - \mathrm{Vol}(K) | / \mathrm{Vol}(K). \tag{16}$$

- Number of sampled points for a single volume computation.
- Complexity, i.e. the number of calls to the oracle. For HAR, this is equal to the number of sampled points. For HMC, it takes the number of reflections into account.
- For HMC, the average number of calls to the oracle per point sampled.

Parameters used. The parameters are as follows:

- Window size. Our experiments cover the following cases:
  - dimension independent: $W = 10, 30$.
  - dimension dependent: $W = 30 + 4\sqrt{n}, 30 + 4n, 30 + 4n^{1.5}, 30 + 4n^2$.
- Target error $\varepsilon = 0.1$ – see Section 3.1 and [13].

Experiments. The overall goal of the experiments is to determine an empirical growth rate of the complexity with respect to the dimension for the algorithm with HMC and compare with HAR. Since the correct value of W is unknown, we proceed in two steps: first, we plot the error and complexity for dimensions 10...50 with different stopping criterion W; second, we select suitable values for W and use them to estimate the complexity with the dimension.

4.3.2 Models

We study polytopes featuring prototypical difficulties. First, we choose the cube, which is difficult for HAR. Second, we choose the simplex, as it has sharp angles. For the comparison with HAR, we take the isotropic simplex since it is already rounded. However for the travel time experiment, we take the standard simplex for the simplicity of its geometric features (diameter of the circumscribed and inscribed spheres).

4.3.3 Running times

We choose to study complexities instead of runninng times, as these are independent from the random walk implementations. However, as an indication, volume calculations for $d \leq 50$ took less than 10 seconds on a laptop computer.

4.4 Tests on volume calculations

4.4.1 Complexity analysis – stopping criterion

We ran the volume computation 50 times for each dimension using different values for W. Recall that the mixing time of HMC for the cube is $O(log(n))$ – Sec. 3.5. Intuitively, the window size W is related to the mixing time of the random walk. Hence we expect that using $W = cst$ for HMC on the cube would lead to a very slow growth of the error with the dimension. Since the maximum dimension is 50 and $log(50) \approx 1.7$, we do not expect to see the effect of the log with our dimension range. In addition, we expect super logarithmic values of W to yield a relative error decrease when the dimension increases.

Since the algorithm is targeting a relative error ε, we wish to identify values of W yielding a constant relative error whatever the dimension. With that in mind, for HMC, we expect to eliminate values of W for which the error would decrease with the dimension. On the contrary, for HAR, we expect to eliminate values of W for which the error increases with the dimension, since W might not grow as fast as the mixing time. We test both the cube (Fig. 4) – a model for which HMC is well understood, and the isotropic simplex (Fig. 5).

Experimental result: requirement for the values of W. As expected, the error explodes for HAR when W is too small, so that plausible values for W are $W = 30 + 4n^{1.5}$ and $W = 30+4n^2$. For HMC, the error clearly decreases for $W = 30+4n^{1.5}$ and $W = 30+4n^2$, but we cannot firmly discard any other W since no clear increase or decrease is apparent. For the cube, we expected an error decrease for any super logarithmic value of W. We conjecture that we cannot see the decrease in relative error because of a too small dimension range.

Experimental result: complexity scaling. To study the complexity (# calls to the oracle), we perform a linear regression on the complexity curves (Figs. 4, 5, bottom plots), obtaining correlation coefficients ≥ 0.997 (Table 1). This extra complexity lowers the mixing time and hence the value of W, leading to an overall smaller complexity for HMC. This increased complexity is compensated by a shorter mixing time, which makes it possible

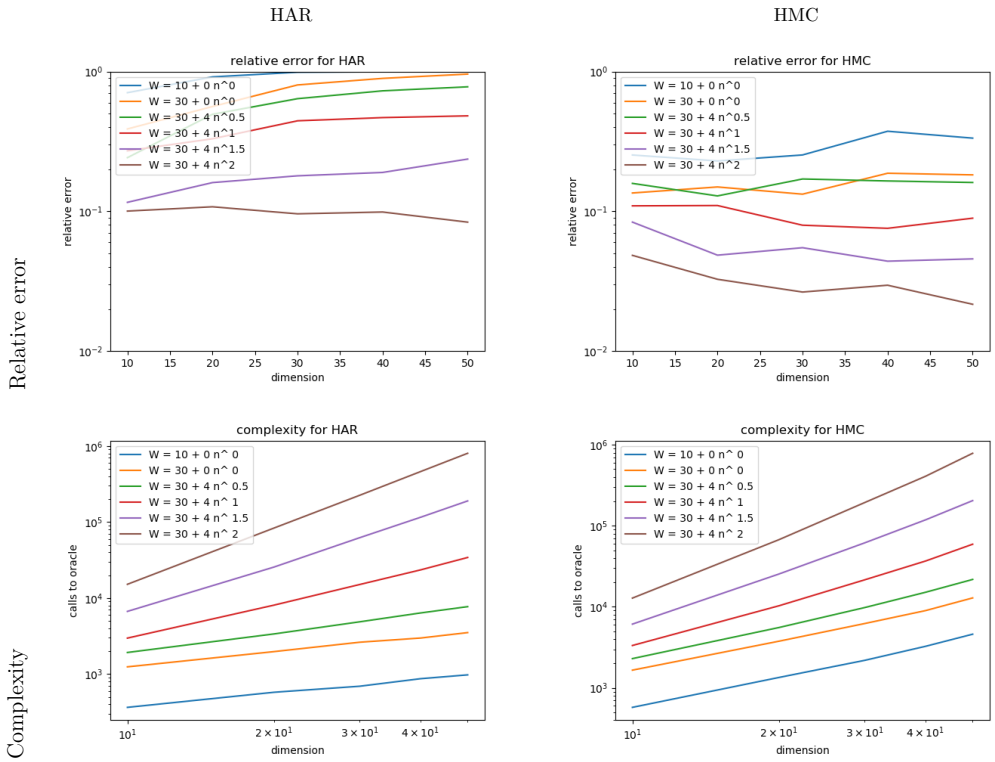

Figure 4 Cube: relative errors (top) and number of calls to the oracle(bottom) for volume computation. All quantities are averaged over 50 runs. Note that a relative error ~ 1 corresponds to a gross underestimation of the volume (Eq. 16).

to reduce the size W of the window and obtain better overall complexity for HMC. The complexity scaling (cube, simplex) for both HMC and HAR is provided in Table 1.

5 Conclusion

To compute the volume of high dimensional polytopes, this paper exploits a novel strategy based on billiard Hamiltonian Monte Carlo. This strategy exploits the ability of billiard walks to escape corners, and the simple analytical expression of the Hamiltonian makes a robust but fallible implementation based on our SEGC paradigm possible and effective, as shown by experiments on polytope volume calculations using the iRRAM and CGAL libraries.

On the theoretical side, our work leaves stimulating questions open, two of which are of prominent importance. The first one is the choice of the optimal travel time required to sample a convex uniformly, which would ideally be determined at runtime for each convex. The second one is the analysis of the incidence of reflections on the mixing time, so as to quantify the speed at which reflections decorrelate successive points.

Our work also prompts a connection with statistical physics, for the calculation of density of states (DoS), which measure the volume in phase space of the pre-image of an energy stratum. DoS are central to compute partition functions, whence all thermodynamic quantities. We anticipate that billiard on level set surfaces bounding strata will prove beneficial, with a potential impact in statistical physics at large.

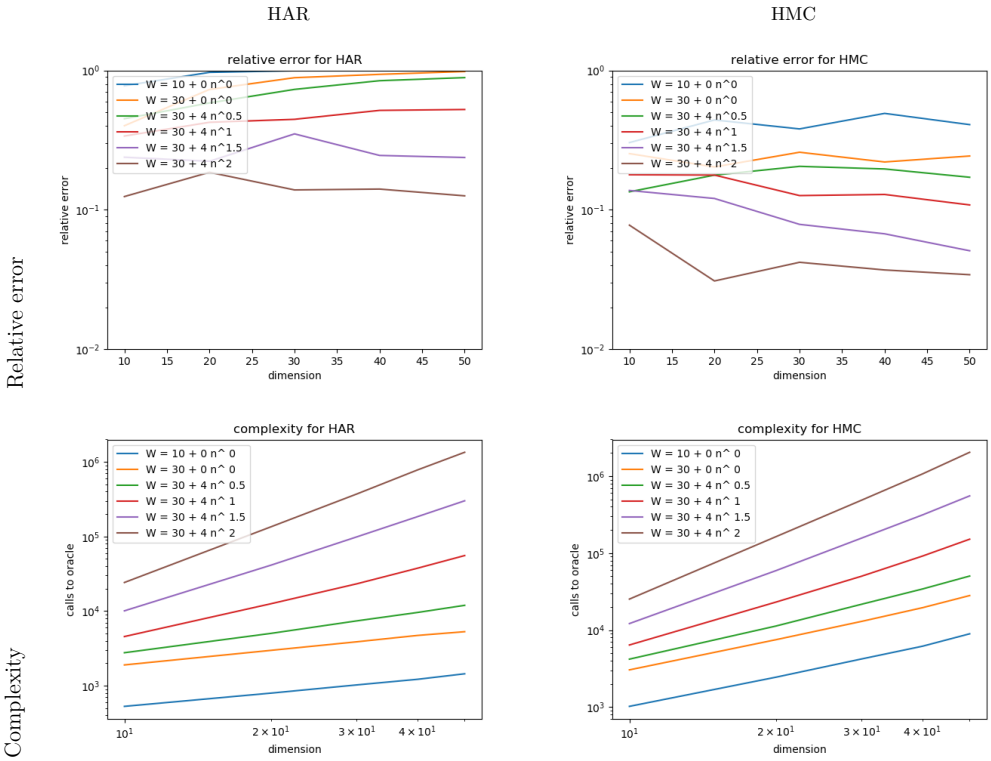

Figure 5 Iso simplex: relative errors (top) and number of calls to the oracle (bottom) for volume computation. All quantities are averaged over 50 runs. Note that a relative error ~ 1 corresponds to a gross underestimation of the volume (Eq. 16).

Window size	Complexity			
	HMC		HAR	
	Cube	Iso Simplex	Cube	Iso Simplex
$W = 10$	$O(n^{1.27})$	$O(n^{1.33})$	$O(n^{0.60})$	$O(n^{0.1})$
$W = 30$	$O(n^{1.25})$	$O(n^{1.36})$	$O(n^{0.64})$	$O(n^{0.64})$
$W = 30 + 4n^{0.5}$	$O(n^{1.39})$	$O(n^{1.54})$	$O(n^{0.86})$	$O(n^{0.90})$
$W = 30 + 4n^{1}$	$O(n^{1.77})$	$O(n^{1.95})$	$O(n^{1.54})$	$O(n^{1.51})$
$W = 30 + 4n^{1.5}$	$O(n^{2.17})$	$O(n^{2.36})$	$O(n^{2.08})$	$O(n^{2.11})$
$W = 30 + 4n^{2}$	$O(n^{2.54})$	$O(n^{2.71})$	$O(n^{2.46})$	$O(n^{2.50})$

Table 1 Scaling of the complexity with the dimension, computed using the same data as Fig. 4 and Fig. 5. Plausible values of W for each algorithm are highlighted in yellow. Exponents for the complexity growth rates were obtained with a linear regression on the complexity curves of Figs. 4, 5 – see main text.

References

1. CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.
2. H. Afshar and J. Domke. Reflection, refraction, and Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems*, pages 3007–3015, 2015.
3. Felipe Alvarez, Jérôme Bolte, and Olivier Brahic. Hessian riemannian gradient flows in convex programming. *SIAM journal on control and optimization*, 43(2):477–501, 2004.
4. Péter Bálint, Nikolai Chernov, Domokos Szász, and Imre Péter Tóth. Geometry of multi-dimensional dispersing billiards. In Wellington de Melo, Marcelo Viana, and Jean-Christophe Yoccoz, editors, *Geometric methods in dynamics (I) : Volume in honor of Jacob Palis*, number 286 in Astérisque, pages 119–150. Société mathématique de France, 2003. URL: `http://www.numdam.org/item/AST_2003__286__119_0`.
5. I. Bárány and Z. Füredi. Computing the volume is difficult. *Discrete & Computational Geometry*, 2(4):319–326, 1987.
6. M. Betancourt. A conceptual introduction to Hamiltonian Monte Carlo. *arXiv preprint arXiv:1701.02434*, 2017.
7. S. Brooks, A. Gelman, G. Jones, and X-L. Meng. *Handbook of Markov Chain Monte Carlo*. CRC press, 2011.
8. James V. Burke. Continuity and differentiability of solutions. `https://sites.math.washington.edu/~burke/crs/555/555_notes/continuity.pdf`.
9. F. Cazals and T. Dreyfus. The Structural Bioinformatics Library: modeling in biomolecular science and beyond. *Bioinformatics*, 7(33):1–8, 2017. URL: `http://sbl.inria.fr`, `doi:10.1093/bioinformatics/btw752`.
10. Apostolos Chalkis, Ioannis Z Emiris, and Vissarion Fisikopoulos. Practical volume estimation by a new annealing schedule for cooling convex bodies. *arXiv preprint arXiv:1905.05494*, 2019.
11. Nikolai Chernov and Roberto Markarian. *Chaotic billiards*. Number 127. American Mathematical Soc., 2006.
12. B. Cousins and S. Vempala. Bypassing KLS: Gaussian cooling and an $O^*(n^3)$ volume algorithm. In *ACM STOC*, pages 539–548. ACM, 2015.
13. B. Cousins and S. Vempala. A practical volume algorithm. *Mathematical Programming Computation*, 8(2):133–160, 2016.
14. B. Cousins and S. Vempala. Gaussian cooling and $O^*(n^3)$ algorithms for volume and gaussian volume. *SIAM Journal on Computing*, 47(3):1237–1273, 2018.
15. M. Dyer and A. Frieze. On the complexity of computing the volume of a polyhedron. *SIAM Journal on Computing*, 17(5):967–974, 1988.
16. M. Dyer, A. Frieze, and R. Kannan. A random polynomial-time algorithm for approximating the volume of convex bodies. *Journal of the ACM (JACM)*, 38(1):1–17, 1991.
17. I. Emiris and V. Fisikopoulos. Efficient random-walk methods for approximating polytope volume. In *Proceedings of the thirtieth annual symposium on Computational geometry*, page 318. ACM, 2014.
18. I. Emiris and V. Fisikopoulos. Practical polytope volume approximation. *ACM Trans. on Math. Software*, 44(4), 2018.
19. A. Fabri and S. Pion. A generic lazy evaluation scheme for exact geometric computations. In *Proc. 2nd Library-Centric Software Design*, pages 75–84, 2006.
20. M. Girolami and B. Calderhead. Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):123–214, 2011.
21. E. Gryazinaa and B. Polyak. Random sampling: Billiard walk algorithm. *arXiv*, 2014.
22. H. Haraldsdóttir, B. Cousins, I. Thiele, R. Fleming, and S. Vempala. CHRR: coordinate hit-and-run with rounding for uniform sampling of constraint-based models. *Bioinformatics*, 33(11):1741–1743, 2017.
23. L. Kettner, K. Mehlhorn, S. Pion, S. Schirra, and C. Yap. Classroom examples of robustness problems in geometric computations. *Computational Geometry*, 40(1):61–78, 2008.

24 László L. Lovász and S. Vempala. Simulated annealing in convex bodies and an $O^*(n^4)$ volume algorithm. *Journal of Computer and System Sciences*, 72(2):392–417, 2006.

25 Y. T. Lee and S. S. Vempala. Eldan's stochastic localization and the KLS hyperplane conjecture: An improved lower bound for expansion. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 998–1007, 2017. doi:10.1109/FOCS.2017.96.

26 Yin Tat Lee and Santosh S Vempala. Convergence rate of riemannian hamiltonian Monte Carlo and faster polytope volume computation. In *STOC*, pages 1115–1121. ACM, 2018.

27 D. Levin and Y. Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

28 S. Levy. *Flavors of Geometry*. Cambridge University Press, 1997.

29 L. Lovász. Hit-and-run mixes fast. *Mathematical Programming, Series B*, 86:443–461, 12 1999. doi:10.1007/s101070050099.

30 L. Lovász and R. Kannan. Faster mixing via average conductance. In *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, STOC '99, pages 282–287, New York, NY, USA, 1999. ACM. URL: http://doi.acm.org/10.1145/301250.301317, doi:10.1145/301250.301317.

31 L. Lovász and S. Vempala. Hit-and-run is fast and fun. *preprint, Microsoft Research*, 2003.

32 L. Lovász and S. Vempala. Hit-and-run from a corner. In *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing*, STOC '04, pages 310–314, New York, NY, USA, 2004. ACM. URL: http://doi.acm.org/10.1145/1007352.1007403, doi:10.1145/1007352.1007403.

33 Howard Masur and Serge Tabachnikov. *(Chapter 13) Rational billiards and flat structures*, pages 1015–1089. Number PART A in Handbook of Dynamical Systems. Part a edition, December 2002. doi:10.1016/S1874-575X(02)80015-7.

34 J.-M. Muller, N. Brunie, F. de Dinechin, C. Jeannerod, M. Joldes, V. Lefèvre, G. Melquiond, N. Revol, and S. Torres. Handbook of floating-point arithmetic. 2018.

35 N. Müller. The iRRAM: Exact arithmetic in C++. In *Computability and Complexity in Analysis*, pages 222–252. Springer, 2001.

36 A. Pakman and L. Paninski. Exact hamiltonian Monte Carlo for truncated multivariate gaussians. *Journal of Computational and Graphical Statistics*, 23(2):518–542, 2014. arXiv: https://doi.org/10.1080/10618600.2013.788448, doi:10.1080/10618600.2013.788448.

37 B. Polyak and E.N. Gryazina. Billiard walk-a new sampling algorithm for control and optimization. *IFAC Proceedings Volumes*, 47(3):6123–6128, 2014.

38 F. Preparata and M. Shamos. *Computational geometry: an introduction*. Springer Science & Business Media, 1985.

39 N. Radford. MCMC using Hamiltonian Dynamics. In Galin L. Jones Steve Brooks, Andrew Gelman and Xiao-Li Meng, editors, *Handbook of Markov Chain Monte Carlo*, chapter 5, pages 113–162. Chapman & Hall/CRC, 2012.

40 G.O. Roberts and J.S. Rosenthal. General state space Markov chains and MCMC algorithms. *Probability Surveys*, 2004.

41 Margaret H Wright. Some properties of the hessian of the logarithmic barrier function. *Mathematical Programming*, 67(1-3):265–295, 1994.

42 C. Yap and T. Dubé. The exact computation paradigm. In *Computing in Euclidean Geometry*, pages 452–492. World Scientific, 1995.

A Proofs

A.1 Lemmas for Thm. 6

Let A and B be measurable subsets of Q. Then let

$$A_B = \{(q,p) \in \Gamma_L | q \in A, \bar{\Phi}(q,p) \in B \times \mathbb{R}^n\}$$

be the subset of Γ of all positions in A with momenta that brings them in B after time L. Similarly, we define on Γ

$$\Psi(q,p) = (\bar{\Phi}^{(q)}(q,p), -\bar{\Phi}^{(p)}(q,p)). \tag{17}$$

▶ **Lemma 13.** *The maps $\bar{\Phi}$ and Ψ preserve the Lebesgue measure on Γ_L. ie for every $A \subset \Gamma_L$ measurable, $\lambda(\bar{\Phi}^{-1}(A)) = \lambda(A)$*

Lemma 13. Clearly it is enough to prove that $\tilde{\Phi}_t$ preserves the Lebesgue measure. In [2], it is proved that for a fixed step size, the Euler method for numerical integration applied to the Hamiltonian flow with reflections, gives rise to a flow that preserves the Lebesgue measure. Letting the step size going to zero, we see that $\tilde{\Phi}_t$ is the pointwise limit of transformations that preserves the Lebesgue measure which implies that $\tilde{\Phi}_t$ preserve the Lebesgue measure. ◀

▶ **Lemma 14.** 1. $\Psi(\Gamma_L) \subset \Gamma_L$

2. $\Psi \circ \Psi = I$ *on* Γ_L

3. *For any measurable sets A and B of $\mathbb{R}^n$,*

$$B_A = \Psi(A_B)$$

4. $H(\Psi(q,p)) = H(q,p)$ *for all* $(q,p) \in \Gamma$.

Lemma 14. 1. and 2. are simple consequences of the reversibility of the Hamiltonian flow with reflections.

3. Let A and B be measurable sets of $\mathbb{R}^n$.

$\Psi_q(A_B) \subset B$ and $\Psi(\Psi(A_B)) = A_B$ thus $\Psi(A_B) \subset B_A$.

By symmetry, $\Psi(B_A) \subset A_B$. Hence by composing with Ψ: $\Psi(\Psi(B_A)) \subset \Psi(A_B)$. Using 2., we deduce $B_A \subset \Psi(A_B)$.

4. is clear.

◀

Thm. 5. Let A and B be measurable sets of $\mathbb{R}^n$. The left hand side of the detailed balance equation becomes

$$\begin{aligned}
\int_A P_{\pi,L}(q,B) d\pi(q) &= \int_A P_{\pi,L}(q,B) \exp(-U(q)) dq \\
&= \int_A \Big[\int_{\{p|(q,p)\in\Gamma_L, \bar{\Phi}^{(q)}(q,p)\in B\}} \exp(-\|p\|^2) dp \\
&+ \int_{\{p|(q,p)\notin\Gamma_L, \bar{\Phi}^{(q)}(q,p)\in B\}} \exp(-\|p\|^2) dp \Big] \exp(-U(q)) dq \\
&= \int_{A_B} \exp(-H(q,p)) dq dp + \int_A \int_{\{p|(q,p)\notin\Gamma_L\}} 1_B(q) \exp(-H(q,p)) dq dp \\
&= \int_{A_B} \exp(-H(q,p)) dq dp + \int_{A\cap B} \int_{\{p|(q,p)\notin\Gamma_L\}} \exp(-H(q,p)) dq dp.
\end{aligned}$$

By symmetry:

$$\int_B P_{\pi,L}(q,A)d\pi(q) = \int_{B_A} \exp(-H(q,p))dqdp + \int_{A\cap B}\int_{\{p|(q,p)\notin\Gamma_L\}} \exp(-H(q,p))dqdp$$

Using lemma 14 and the measure conservation of lemma 13 we obtain:

$$\begin{aligned}\int_{A_B} \exp(-H(q,p))dqdp &= \int_{\Psi(B_A)} \exp(-H(q,p))dqdp \\ &= \int_{B_A} \exp(-H(\Psi(q,p)))dqdp \\ &= \int_{B_A} \exp(-H(q,p))dqdp\end{aligned}$$

Which concludes the proof. ◀

A.2 Proof of Lemma 8

▶ **Lemma 15.** *Let Q be an open convex subset in $\mathbb{R}^n$ that contains the ball $B(0,2\rho)$, let x be a point in Q and let v be in $B(0,\rho)-x$. Let $f:\mathbb{R}^n\to\mathbb{R}^n$ be a continuous and bounded map. Suppose that $(x(t),v(t))\in\mathbb{R}^{2n}$ is the solution of the Cauchy problem*

$$\begin{aligned}x(0) &= x\\ v(0) &= v\\ x'(t) &= v(t)\\ v'(t) &= af(t)\end{aligned}$$

where a is a real number. If $|a|\le\frac{\rho}{\|f\|_\infty}$ then for all $t\in[0,1]$, $x(t)$ is in the convex hull of x and of the ball $B(0,2\rho)$ and therefore in Q.

Lemma. 15. Suppose $|a|\le\frac{\rho}{\|f\|_\infty}$. By the mean value theorem, $v(t)=v+w(t)$ where $\|w(t)\|\le|a|\|f\|_\infty t\le\rho t$ for $t\ge 0$. The derivative of function $y(t)=x(t)-vt$ is $w(t)$, therefore by the mean value theorem, $y(t)=y(0)+tz(t)$ where $\|z(t)\|\le\rho/2$. Therefore for all $t\in[0,1]$,

$$\begin{aligned}x(t) &= y(t)+vt\\ &= (1-t)x+t((v+x)+z(t))\end{aligned}$$

is in the convex hull of x and of the ball $B(0,2\rho)$. ◀

▶ **Lemma 16.** *Let B be an open ball in $\mathbb{R}^n$ and let $\varphi:B\to\mathbb{R}^n$ be a differentiable map. If for each x in B, $\|d\varphi(x)-Id\|<1$, then φ is a diffeomorphism.*

Lemma 16. The only thing to prove is that φ is one to one. Let $x\neq y$ be two points in B. Consider the map $f:t\in[0,\|y-x\|]\to\varphi(x+tu)\cdot u$ where $u=\frac{y-x}{\|y-x\|}$. Using Schwarz inequality and the assumption we obtain

$$\begin{aligned}f'(t) &= d\varphi(x+tu)(u)\cdot u\\ &= Id(u)\cdot u+(d\varphi(x+tu)-Id)(u)\cdot u\\ &\ge 1-\|d\varphi(x+tu)-Id\|\|u\|\|u\|\\ &> 0.\end{aligned}$$

It follows that $f(\|y-x\|)>f(0)$. Now $f(0)=\varphi(x)\cdot u$ and $f(\|y-x\|)=\varphi(y)\cdot u$, hence $\varphi(x)\neq\varphi(y)$. ◀

▶ **Lemma 17.** *Let $B = B(0,r)$ be a closed ball in $\mathbb{R}^n$ of center 0 and radius $r > 0$ and let $\varphi : B \to \mathbb{R}^n$ be a one to one continuous map. If for all x in B, $d(x,\varphi(x)) \leq r/4$ then $\varphi(B)$ contains the ball $B(0,r/2)$.*

Lemma 17. By Jordan-Brouwer Theorem, the complement in $\mathbb{R}^n$ of the image Σ of the sphere $S = \partial B$ has exactly two connected components C_1 and C_2, one which is bounded, say C_1, and one which is not. By assumption the open ball $\overset{\circ}{B}(O,r/2)$ doesn't intersect Σ, hence is included in C_1 or C_2.

The image $E = \varphi(\overset{\circ}{B})$ of the interior of the ball B is included in C_i, one of the two connected components of $\mathbb{R}^n \setminus \Sigma$. On the one hand, by Jordan-Brouwer invariance of the domain theorem, E is open. On the other hand, $E = \varphi(B) \cap C_i$ and since $\varphi(B)$ is compact, $C_i \setminus E$ is open. Now C_i is connected, hence E or $C_i \setminus E$ is empty. Therefore $E = C_i$.

Since $\varphi(B)$ is compact, E is bounded, and therefore $E = C_i = C_1$. Moreover, by assumption, $\varphi(0) \in B^o(0,r/2) \cap E$ which implies that $B^o(0,r/2) \subset E$. ◀

Lemma 8. We assume without any loss of generality that $0 \in Q$.

Let $q^{(1)}, q^{(2)} \in Q$. One way to get a trajectory going from the point $q^{(1)}$ to a point very close to $q^{(2)}$, is to select a very high momentum $p_\alpha = \frac{1}{\alpha}(q^{(2)} - q^{(1)})$ and a short time $t_\alpha = \alpha$ with $\alpha > 0$. When $\alpha \to 0$, the potential energy term U of the Hamiltonian becomes less and less relevant, thus the trajectory converges to a straight line and $\lim_{\alpha\to 0} \Phi^q_{t_\alpha}(q^{(1)}, p_\alpha) = q^{(2)}$.

This intuition is formalized using the scaled variables

$$\begin{cases} \tilde{q}(t) &= q(\alpha t) \\ \tilde{p}(t) &= \alpha p(\alpha t). \end{cases} \tag{18}$$

The equation of motion becomes:

$$\frac{d\tilde{q}}{dt}(t) = \tilde{p}(t) \tag{19}$$

$$\frac{d\tilde{p}}{dt}(t) = \alpha^2 \nabla_q U(\tilde{q}(t)). \tag{20}$$

which defines a flow $\phi(\alpha, t, q, p)$. It should be noted that $\phi(-\alpha, t, q, p) = \phi(\alpha, t, q, p)$ for every α, t, q, p and that ϕ is correctly defined for $\alpha = 0$. In this case, it is easy to see that:

$$\phi(0, t, q, p) = q + pt. \tag{21}$$

As π is the restriction of a positive smooth function, ϕ is defined for every $(\alpha, t, q, p) \in [-1, 1] \times \mathbb{R}^+ \times \mathbb{R}^n \times \mathbb{R}^n$.

Furthermore, $\alpha^2 \nabla_q U$ is smooth. Hence, using the differentiability of the solutions of differential equations on parameters and initial conditions (see [8]), we see that ϕ is C^2 on $]-1, 1[\times Q \times \mathbb{R}^+ \times \mathbb{R}^n$.

Since Q is open, there exists $\rho > 0$ such that $B(0, 2\rho) \subset Q$. Let ν be the measure on Q which is the Lebesgue measure on $B(0, \rho/2)$ and zero outside the ball $B(0, \rho/2)$.

Our aim is to show that there exists $\epsilon > 0$ such that for every $q \in Q$, $P_\pi(q, .) \geq \epsilon\nu(.)$. Note that we would only need that $P^{n_0}_\pi(q, .) \geq \epsilon\nu(.)$ for some n_0, but since Q is convex we will be able to take $n_0 = 1$.

The density of the probability measure associated with momenta is $\exp(-1/2\|p\|^2)$, and taking into account the rescaling of the momenta, we define the probability density

$$\gamma(p) = \alpha \exp(-\frac{1}{2}\|\frac{1}{\alpha}p\|^2). \tag{22}$$

Q is bounded, hence there exists $\epsilon > 0$ such that for every $(q,p) \in \{(q,p) | q \in Q, p \in B(-q,\rho)\}$,

$$\gamma(p) > \epsilon$$

Let

$$A = \{(\alpha, t, q, p) : |\alpha| \leq 1/2, t \in [1/2, 3/2],\ q \in \bar{Q},\ p \in B(-q,\rho)\}.$$

A is compact, hence the first and second order derivatives of ϕ are bounded on A. Thus there exists $Z > 0$ such that for every $(\alpha, t, q, p) \in A$,

$$\|\phi^q(\alpha, t, q, p) - \phi^q(0, 1, q, p)\| \leq (\alpha + |t-1|)Z \tag{23}$$

and

$$\|d_p\phi^q(\alpha, t, q, p) - d_v\phi^q(0, 1, q, p)\| \leq (\alpha + |t-1|)Z. \tag{24}$$

Using equation 21, the two above inequalities are equivalent to

$$\|\phi^q(\alpha, t, q, p) - (q+p)\| \leq (\alpha + |t-1|)Z \tag{25}$$

and

$$\|d_p\phi^q(\alpha, t, q, p) - Id_{\mathbb{R}^n}\| \leq (\alpha + |t-1|)Z \tag{26}$$

The determinant is continuous, so there exists $0 < \beta < 1$ such that $\|d_p\phi^q(\alpha, t, q, p) - Id_{\mathbb{R}^n}\| < \beta$ implies

$$1/2 < |d_p\phi(\alpha, t, q, p)| < 2. \tag{27}$$

Finally, using Lemma 15 together with the fact that ∇U is bounded, we deduce that there exists $\alpha_0 > 0$ such that for every $0 < \alpha \leq \alpha_0$ and every $q \in Q$ and $p \in B(-q,\rho)$, the trajectory $\phi^q(\alpha, t, q, p)$ for $t \leq 1$ stays in Q. It follows that ϕ and Φ coincide, for there is no reflection.

Let $\alpha = \min(\frac{\rho}{4Z}, \frac{1}{2Z}, \frac{\beta}{2Z}, \alpha_0) > 0$. Let any $t \in [1 - \frac{\beta}{2Z}, 1]$ and q be in Q. By lemma 16 below, the map

$$f : p \to \phi^q(\alpha, t, q, p)$$

is a C^1 diffeomorphism from $B(-q,\rho)$ to $V = \phi^q(\alpha, t, q, B(-q,\rho))$ and by Lemma 17 (applied to f translated by $-q$), $B(0, \rho/2) \subset V$. Furthermore, equation 27 implies that for every $q' \in V$

$$|df^{-1}(q')| > 1/2. \tag{28}$$

Hence, the push forward measure ξ of ν by f is non zero on $B(0,\rho/2)$, and has a density

$$\xi(q') = \nu(f^{-1}(q'))|df^{-1}(q')| \geq \epsilon/2 \tag{29}$$

on $B(0,\rho/2)$. Hence, under the condition that the travel time t is in $[1 - \frac{\beta}{2Z}, 1]$, we get the following transition probability:

$$P(q, .|t \in [1 - \frac{\beta}{2Z}, 1]) \geq \frac{\epsilon}{2}\nu(.)$$

For the random walk, t is sampled uniformly in $[0,1]$, hence

$$P(q, .) \geq \frac{\epsilon}{2}\frac{\beta}{2Z}\nu(.)$$

which concludes the proof. ◀

▶ Remark 18. The previous proof uses $n_0 = 1$. We believe it should be possible to extend the proof to non convex Q by taking $n_0 > 1$, and some extra regularity assumptions on Q.

A.3 Mixing time for the cube

Lemma 12. We consider the canonical basis of $\mathbb{R}^n$. As shown before in Eq. (8), the trajectory coordinates associated to the Gaussian are all independent from each other. Furthermore, when a reflection with a boundary occurs, it means that one of the coordinates reached 0 or 1. The reflection simply switches the sign of the momentum for this coordinate, leaving other coordinates unchanged. Finally, the initial momentum vector is sampled according to $p(0) \sim \mathcal{N}(0, I_n)$, therefore each coordinate of $p(0)$ is sampled from an independent Gaussian $\mathcal{N}(0,1)$ in $\mathbb{R}$.
Hence we conclude that each coordinate has the behavior of a 1-dimensional HMC random walk sampling a 1-dimensional Gaussian, all independent from each other.

Let us consider the 1-dimensional random walk for a given Gaussian. We write $P_k(x,.)$ the distribution after k steps starting from $x \in [0,1]$, and g the probability density associated with the restriction of the Gaussian to $[0,1]$. Using theorem 9 combined with theorem 8 for the 1-D Gaussian restricted to $[0,1]$, we deduce that there exists $0 < \rho < 1$ such that for all $x \in [0,1]$,

$$\|P_k(x,.) - g\|_{TV} \leq \rho^k.$$

Observe that writing $P_k^{(n)}$ the distribution after k steps in dimension n and g_n the Gaussian restricted to $[0,1]^n$, we have

$$\begin{cases} P_k^{(n)}(x,y) = P_k(x_1,y_1)P_k^{(n-1)}((x_2,...,x_n),(y_2,...,y_n)), \\ g_n(x) = g(x_1)g_{n-1}((x_2,...,x_n)) \end{cases} \tag{30}$$

The total variation distance can be written as

$$\begin{aligned}
\|P_k^{(n)}(x,.) - g_n\|_{TV} &= \frac{1}{2}\int_{[0,1]^n} \left|P^{(n)}(x,y) - g_n(y)\right| dy \\
&= \frac{1}{2}\int_{[0,1]\times[0,1]^{n-1}} \left|P_k(x,y_1)P_k^{(n-1)}(x,y_2) - g(y_1)g_{n-1}(y_2)\right| dy_1 dy_2 \\
&= \frac{1}{2}\int_{[0,1]\times[0,1]^{n-1}} \left|\left(P_k(x,y_1) - g(y_1) + g(y_1)\right)P_k^{(n-1)}(x,y_2) - g(y_1)g_{n-1}(y_2)\right| dy_1 dy_2 \\
&\leq \frac{1}{2}\int_{[0,1]\times[0,1]^{n-1}} \left|\left(P_k(x,y_1) - g(y_1)\right)P_k^{(n-1)}(x,y_2)\right| dy_1 dy_2 \\
&+ \frac{1}{2}\int_{[0,1]\times[0,1]^{n-1}} \left|g(y_1)P_k^{(n-1)}(x,y_2) - g(y_1)g_{n-1}(y_2)\right| dy_1 dy_2 \\
&\leq \frac{1}{2}\int_{[0,1]} |P_k(x,y_1) - g(y_1)| \, dy_1 \\
&+ \frac{1}{2}\int_{[0,1]^{n-1}} \left|P_k^{(n-1)}(x,y_2) - g_{n-1}(y_2)\right| dy_2
\end{aligned}$$

We deduce

$$\|P_k^{(n)}(x,.) - g_n\|_{TV} \leq n\|P_k(x,.) - g\|_{TV} \leq n\rho^k \tag{31}$$

Hence for a fixed ϵ, if k satisfies $n\rho^k \leq \epsilon$, then for every x, $\|P_k^{(n)}(x,.) - g_n\|_{TV} \leq \epsilon$. Thus we take $k \geq (\log n - \log \epsilon)/(\log 1/\rho)$, and the mixing time is $O(\log n)$.

In addition, as each coordinate is from each other, the total number of reflections is the sum of reflections per coordinate. Hence, the number of reflections is proportional to the dimension. ◀

B Supporting information: pseudo-code

In the following, we provide the pseudo-code for the high level description of the HMC algorithm (Algorithm 1).

Algorithm 2 Hamiltonian Monte Carlo step with real RAM arithmetic model

```
HMC_step(q)
Choose a travel time L ~ Unif(0, L_max).
choose p ~ N(0, I_n)
Set dist = L
while dist > 0 do
    (intersection, t_c) ← Intersect_hyper_planes(q, p) // find intersection with hyper-planes
    if intersection = False OR dist < t_c then
        (q, p) = Update_positions_momenta(q, p, dist) // update traj. with distance dist
        Set dist = 0
    else
        (q, p) = Update_positions_momenta(q, p, t_c)
        Reflext_normal(p(t_c), n_c)
        Set dist = dist − t_c
```

Algorithm 3 Hamiltonian Monte Carlo step with `iRRAM` number type

```
HMC_step(q)
Choose a travel time L ~ Unif(0, L_max).
choose p ~ N(0, I_n) iRRAM REAL
Set dist = L
while dist > 0 do
    (intersection, t_c) ← Intersect_hyper_planes(q, p) with t_c an iRRAM REAL.
    t_c^< = Near_inf_double(t_c)
    if intersection = False OR dist < t_c^< then
        (q, p) = Update_positions_momenta(q, p, dist) // update trajectory with distance dist
    else
        (q, p) = Update_positions_momenta(q, p, t_c^<)
        Reflext_normal(p(t_c^<), n_c)
        Set dist = dist − t_c^<
    Is_strictly_in_convex(q)
    Return q
```

Algorithm 4 Find trajectory parameters for a given direction.

Compute_traj_params(q_{dir}, p_{dir})
Set $\omega = \sqrt{2a}$
Compute $C = \sqrt{q_{dir}^2 + p_{dir}^2/w^2}$
Compute $\phi = \arctan(-\frac{p_{dir}}{q_{dir}\omega})$
if $p_{dir} < 0$ and $\phi < 0$ **then**
  $\phi = \phi + \pi$
if $p_{dir} > 0$ and $\phi > 0$ **then**
  $\phi = \phi - \pi$
Return $[\omega, C, \phi]$

Algorithm 5 Intersecting the trajectory with hyperplanes bounding the polytope

```
Intersect_hyper_planes(q, p)
Set intersection = False
for each hyperplane H of equation (Ax)_i = b_i do
  Compute the outward pointing normal n_H to the hplane
  Compute the dot products q_{n_H} =< q, n_H > and p_{n_H} =< p, n_H >
  [ω, C, Φ] = Compute_traj_params(q_{n_H}, p_{n_H})
  if C > b_i then
    t1 = (arccos(b_i/C) − φ) /ω
    if t1 < 0 then
      t1 = t1 + 2π/ω
    t2 = (− arccos(b/C) − φ) /ω
    if t2 < 0 then
      t2 = t2 + 2π/ω
    t = min(t1, t2)
    if intersection = False then
      Set t_c = t
      Set t_c = n_H
      Set intersection = True
    else
      if t < t_c then
        Set t_c = t
        Set n_c = n_H
Return (intersection, t_c)
```

Algorithm 6 Reflecting the normal

Reflext_normal(p, n)
$n' = n/\|n\|$ // unit normal
Return $p - 2 < p, n' > n'$

Algorithm 7 Update trajectory with distance t

Update_positions_momenta(q, p, t)
for i from 1 to n **do**
  $[\omega, C, \Phi] =$ Compute_traj_params (q_i, p_i)
  Set $q_i = C\cos(\omega t + \phi)$
  Set $p_i = -\omega C \sin(\omega t + \phi)$
Return (q, p)

C Supporting information: results

To complement the analysis of section 4.4.1, we provide in the following plots with the variance of the statistics of interest (Figs. 6, 7, 8).

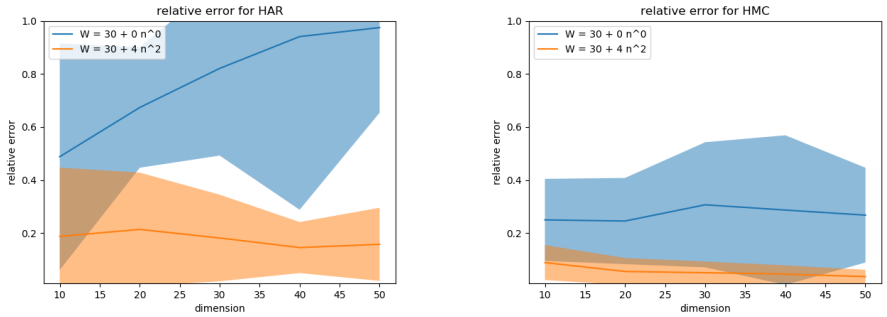

Figure 6 Error analysis: variance as a function of the dimension. Model: isotropic simplex. (Left) HAR **(Right)** HMC For the same window size, the variance of the error is lower for HMC.

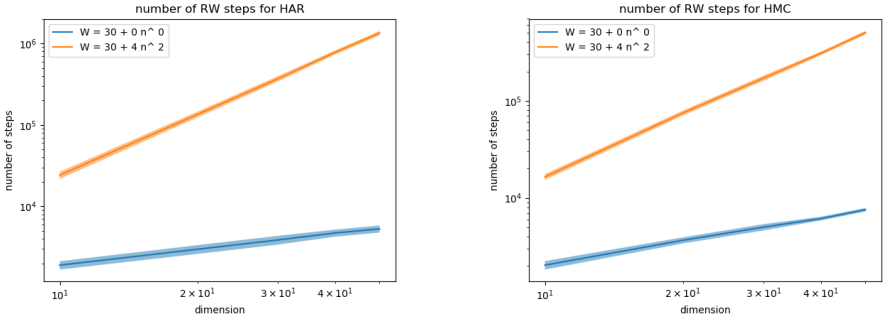

Figure 7 Number of generated points: variance. Model: isotropic simplex. (Left) HAR (Right) HMC The log scale hints at a polynomial number of points as a function of the dimension.

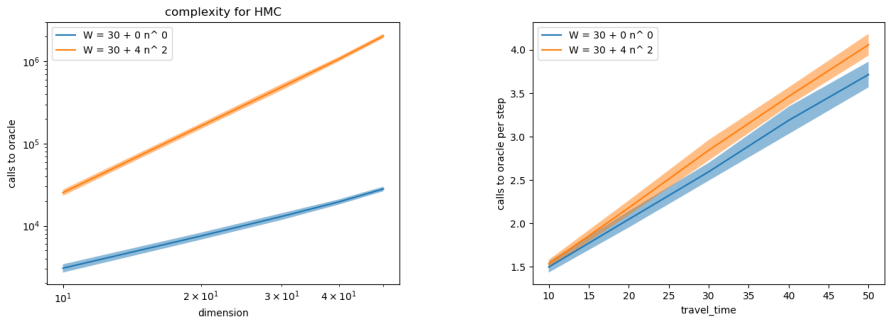

Figure 8 (Left) Number of oracle calls for HMC (Right) Ratio between the number of oracle calls and the number of points generated Plots for the isotropic simplex.