HAL
open science

Geometric algorithms for sampling the flux space of metabolic networks

Apostolos Chalkis, Vissarion Fisikopoulos, Elias Tsigaridas, Haris Zafeiropoulos

To cite this version:

Apostolos Chalkis, Vissarion Fisikopoulos, Elias Tsigaridas, Haris Zafeiropoulos. Geometric algorithms for sampling the flux space of metabolic networks. The 37th International Symposium on Computational Geometry (SoCG), Jun 2021, Buffalo, United States. 10.4230/LIPIcs.SoCG.2021.21 . hal-03047049v2

HAL Id: hal-03047049

https://inria.hal.science/hal-03047049v2

Submitted on 20 Mar 2021

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.

Geometric algorithms for sampling the flux space of metabolic networks

Apostolos Chalkis

Department of Informatics & Telecommunications
National & Kapodistrian University of Athens, and
Athena Research Innovation Center, Greece
achalkis@di.uoa.gr

Vissarion Fisikopoulos

Department of Informatics & Telecommunications
National & Kapodistrian University of Athens, Greece
vfisikop@di.uoa.gr

Elias Tsigaridas

Inria Paris and IMJ-PRG,
Sorbonne Université and Paris Université
elias.tsigaridas@inria.fr

Haris Zafeiropoulos

Department of Biology, University of Crete
Institute of Marine Biology, Biotechnology and Aquaculture, Hellenic Centre for Marine Research
haris-zaf@hcmr.gr

Abstract

Systems Biology is a fundamental field and paradigm that introduces a new era in Biology. The crux of its functionality and usefulness relies on metabolic networks that model the reactions occurring inside an organism and provide the means to understand the underlying mechanisms that govern biological systems. Even more, metabolic networks have a broader impact that ranges from resolution of ecosystems to personalized medicine.

The analysis of metabolic networks is a computational geometry oriented field as one of the main operations they depend on is sampling uniformly points from polytopes; the latter provides a representation of the steady states of the metabolic networks. However, the polytopes that result from biological data are of very high dimension (to the order of thousands) and in most, if not all, the cases are considerably skinny. Therefore, to perform uniform random sampling efficiently in this setting, we need a novel algorithmic and computational framework specially tailored for the properties of metabolic networks.

We present a complete software framework to handle sampling in metabolic networks. Its backbone is a Multiphase Monte Carlo Sampling (MMCS) algorithm that unifies rounding and sampling in one pass, obtaining both upon termination. It exploits an improved variant of the Billiard Walk that enjoys faster arithmetic complexity per step. We demonstrate the efficiency of our approach by performing extensive experiments on various metabolic networks. Notably, sampling on the most complicated human metabolic network accessible today, Recon3D, corresponding to a polytope of dimension 5 335, took less than 30 hours. To our knowledge, that is out of reach for existing software.

2012 ACM Subject Classification Mathematics of computing→Mathematical software; Applied computing→Systems biology; Computing methodologies→Modeling and simulation

Keywords and phrases Flux analysis, metabolic networks, convex polytopes, random walks, sampling

Acknowledgements We would like to thank the anonymous reviewers for their helpful comments and suggestions. We also thank Ioannis Emiris for his useful comments.

1 Introduction

1.1 The field of Systems Biology

Systems Biology establishes a scientific approach and a paradigm. As a research approach, it is the qualitative and quantitative study of the systemic properties of a biological entity along with their ever evolving interactions [32, 33]. By combining experimental studies with mathematical modeling it analyzes the function and the behavior of biological systems. In this setting, we model the interactions between the components of a system to shed light on the system's *raison d'être* and to decipher its underlying mechanisms in terms of evolution, development, and physiology [27].

Initially, Systems Biology emerged as a need. New technologies in Biology accumulate vast amounts of information/data from different levels of the biological organization, i.e., genome, transcriptome, proteome, metabolome [49]. This leads to the emerging question *"what shall we do with all these pieces of information"?* The answer, if we consider Systems Biology as a paradigm, is to move away from reductionism, still the main conceptual approach in biological research, and adopt holistic approaches for interpreting how a system's properties emerge [43]. The following diagram provides a first, rough, mathematical formalization of this approach.

$$\textit{components} \rightarrow \textit{networks} \rightarrow \textit{in silico models} \rightarrow \textit{phenotype} \text{ [47].}$$

Systems Biology expands in all the different levels of living entities, from the molecular, to the organismal and ecological level. The notion that penetrates all levels horizontally is *metabolism*; the process that modifies molecules and maintains the living state of a cell or an organism through a set of chemical reactions [53]. The reactions begin with a particular molecule which they convert into some other molecule(s), while they are catalyzed by enzymes in a key-lock relationship. We call the quantitative relationships between the components of a reaction *stoichiometry*. Linked reactions, where the product of the first acts as the substrate for the next, build up metabolic pathways. Each pathway is responsible for a certain function. We can link together the aggregation of all the pathways that take place in an organism (and their corresponding reactions) and represent them mathematically using the reactions' stoichiometry. Therefore, at the species level, metabolism is a network of its metabolic pathways and we call these representations *metabolic networks*.

1.2 From metabolism to computational geometry

The complete reconstruction of the metabolic network of an organism is a challenging, time consuming, and computationally intensive task; especially for species of high level of complexity such as *Homo sapiens*. Even though sequencing the complete genome of a species is becoming a trivial task providing us with quality insight, manual curation is still mandatory and large groups of researchers need to spend a great amount of time to build such models [57]. However, over the last few years, automatic reconstruction approaches for building genome-scale metabolic models [40] of relatively high quality have been developed. Either way, we can now obtain the metabolic network of a bacterial species (single cell species) of a tissue and even the complete metabolic network of a mammal. Biologists are also moving towards obtaining such networks for all the species present in a microbial community. This will allow us to further investigate the dynamics, the functional profile, and the inter-species reactions that occur. Using the stoichiometry of each reaction, which is always the same in the various species, we convert the metabolic network of an organism to a mathematical model. Thus, the metabolic network becomes an *in silico* model of the knowledge it represents. In metabolic networks analysis mass and energy are considered to be conserved [46]. As many homeostatic states, that is steady internal conditions [54], are close to steady states (where the production rate of each metabolite equals its consumption rate [8]) we commonly use the latter in metabolic networks analysis.

Stoichiometric coefficients are the number of molecules a biochemical reaction consumes and produces. The coefficients of all the reactions in a network, with m metabolites and n reactions ($m < n$), form the

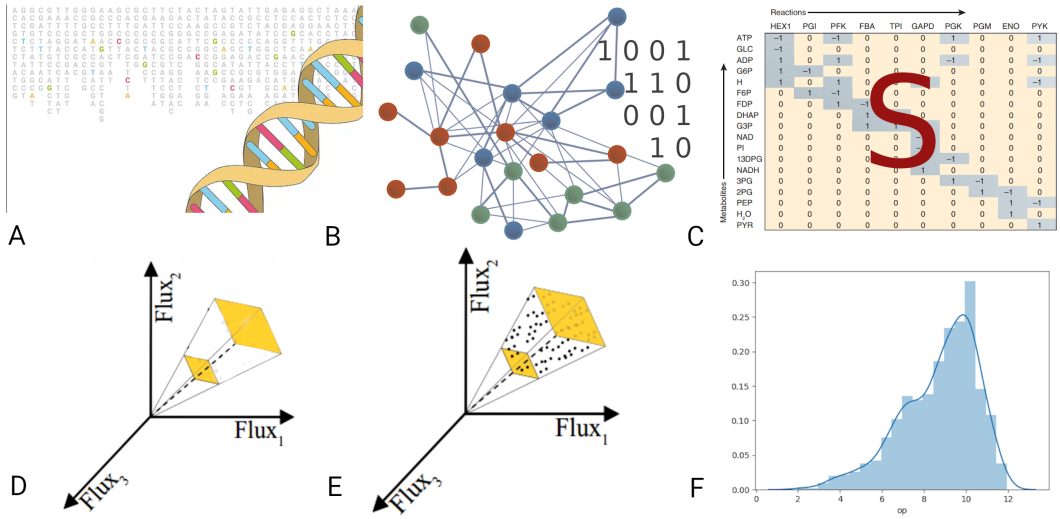

Figure 1 From DNA sequences to distributions of metabolic fluxes. (A) The genes of an organism provide us with the enzymes that it can potentially produce. Enzymes are like a blueprint for the reactions they can catalyze. (B) Using the enzymes we identify the reactions in the organism. (C) We construct the stoichiometric matrix of the metabolic model. (D) We consider the flux space under different conditions (e.g., steady states); they correspond to polytopes containing flux vectors addressing these conditions. (E) We sample from polytopes that are typically skinny and of high dimension. (F) The distribution of the flux of a reaction provides great insights to biologists.

stoichiometric matrix $S \in \mathbb{R}^{m \times n}$ [47]. The nullspace of S corresponds to the steady states of the network:

$$S \cdot x = 0, \tag{1}$$

where $x \in \mathbb{R}^n$ is the *flux vector* that contains the fluxes of each chemical reaction of the network. Flux is the rate of turnover of molecules through a metabolic pathway.

All physical variables are finite, therefore the flux (and the concentration) is bounded [47]; that is for each coordinate x_i of the x, there are $2n$ constants $x_{ub,i}$ and $x_{lb,i}$ such that $x_{lb,i} \leq x_i \leq x_{ub,i}$, for $i \in [n]$. We derive the constraints from explicit experimental information. In cases where there is no such information, reactions are left unconstrained by setting arbitrary large values to their corresponding bounds according to their reversibility properties; i.e., if a reaction is reversible then its flux might be negative as well [38]. The constraints define a n-dimensional box containing both the steady and the dynamic states of the system. If we intersect that box with the nullspace of S, then we define a polytope that encodes all the possible steady states and their flux distributions [47]. We call it the steady-state *flux space*. Fig. 1 illustrates the complete workflow from building a metabolic network to the computation of a flux distribution.

Using the polytopal representation, a commonly used method for the analysis of a metabolic network is Flux Balance Analysis (FBA) [45]. FBA identifies a single optimal flux distribution by optimizing a linear objective function over a polytope [45]. Unfortunately, this is a *biased* method because it depends on the selection of the objective function. To study the global features of a metabolic network we need *unbiased methods*. To obtain an accurate picture of the whole solution space we exploit sampling techniques [52]. If collect a sufficient number of points uniformly distributed in the interior of the polytope, then the biologists can study the properties of certain components of the whole network and deduce significant biological insights [47]. Therefore, efficient sampling tools are of great importance.

1.3 Metabolic networks through the lens of random sampling

Efficient uniform random sampling on polytopes resulting from metabolic networks is a very challenging task both from the theoretical (algorithmic) and the engineering (implementation) point of view. First,

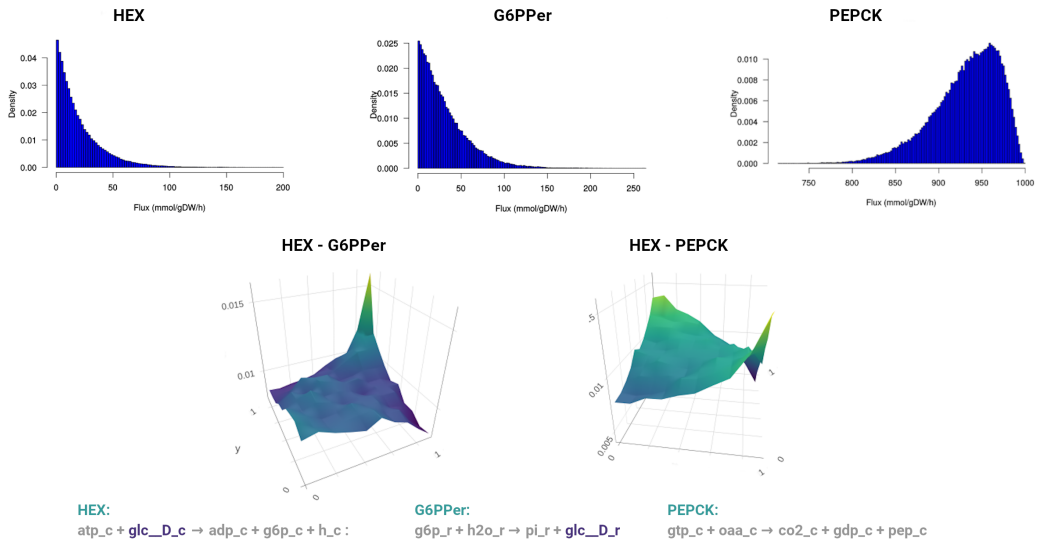

Figure 2 Flux distributions in the most recent human metabolic network Recon3D [7]. We estimate the flux distributions of the reactions catalyzed by the enzymes Hexokinase (D-Glucose:ATP) (HEX), Glucose-6-Phosphate Phosphatase, Edoplasmic Reticular (G6PPer) and Phosphoenolpyruvate carboxykinase (GTP) (PEPCK). As we sample steady states, the production rate of *glc__D_c* should be equal to its consumption rate. Thus, in the corresponding copula, we see a positive dependency between HEX, i.e., the reaction that consumes *glc__D_c* and G6PPer, that produces it. Furthermore, the PEPCK reaction operates when there is no *glc__D_c* available and does not operate when the latter is present. Thus, in their copula we observe a negative dependency between HEX and PEPCK. A copula is a bivariate probability distribution for which the marginal probability distribution of each variable is uniform. It implies a positive dependency when the mass of the distribution concentrates along the up-diagonal (HEX - G6PPer) and a negative dependency when the mass is concentrated along the down-diagonal (HEX - PEPCK). The bottom line contains the reactions and their stoichiometry.

the dimension of the polytopes is of the order of certain thousands. This requires, for example, advanced engineering techniques to cope with memory requirements and to perform linear algebra operations with large matrices; e.g., in Recon3D [7] we compute the null space of a $8\,399 \times 13\,543$ matrix. Second, the polytopes are rather skinny (Sec. 4); this makes it harder for sampling algorithms to move in the interior of polytopes and calls for novel practical techniques to sample.

There is extended on-going research concerning advanced algorithms and implementations for sampling metabolic networks over the last decades. Markov Chain Monte Carlo algorithms such as Hit-and-Run (HR) [55] have been widely used to address the challenges of sampling. Two variants of HR are the non-Markovian Artificial Centering Hit-and-Run (ACHR) [30] that has been widely used in sampling metabolic models, e.g., [51], and Coordinate Hit-and-Run with Rounding (CHRR) [24]. The latter is part of the `cobra` toolbox [25], the most commonly used software package for the analysis of metabolic networks. CHRR enables sampling from complex metabolic network corresponding to the highest dimensional polytopes so far. There are also stochastic formulations where the inclusion of experimental noise in the model makes it more compatible with the stochastic nature of biological networks [39]. The recent study in [18] offers an overview as well as an experimental comparison of the currently available samplers.

These implementations played a crucial role in actually performing in practice uniform sampling from the flux space. However, they are currently limited to handle polytopes of dimension say $\leq 2\,500$ [18, 24]. This is also the order of magnitude of the most complicated, so far, metabolic network model built, Recon3D [7]. By including 13 543 metabolic reactions and involving 4 140 unique metabolites, Recon3D provides a representation of the 17% of the functionally of annotated human genes. To our knowledge, there is no method that can efficiently handle sampling from the flux space of Recon3D.

Apparently, the dimension of the polytopes will keep rising and not only for the ones corresponding

to human metabolic networks. Metabolism governs systems biology at all its levels, including the one of the community. Thus, we are not only interested in sampling a sole metabolic network, even if it has the challenges of the human. Sampling in polytopes associated to network of networks are the next big thing in metabolic networks analysis and in Systems Biology [4, 48].

Regarding the sampling process, from the theoretical point of view, we are interested in the convergence time, or *mixing time*, of the Markov Chain, or geometric *random walk*, to the target distribution. Given a d-dimensional polytope P, the mixing time of several geometric random walks (e.g., HR or Ball Walk) grows quadratically with respect to the sandwiching ratio R/r of the polytope [36, 37]. Here r and R are the radii of the smallest and largest ball with center the origin that contains, and is contained, in P, respectively; i.e., $rB_d \subseteq P \subseteq RB_d$, where B_d is the unit ball. It is crucial to reduce R/r, i.e., to put P in well a rounded position where $R/r = \widetilde{O}(\sqrt{d})$; the $\widetilde{O}(\cdot)$ notation means that we are ignoring polylogarithmic factors. A powerful approach to obtain well roundness is to put P in *near isotropic position*. In general, $K \subset \mathbb{R}^d$ is in isotropic position if the uniform distribution over K is in isotropic position, that is $\mathbb{E}_{X\sim K}[X] = 0$ and $\mathbb{E}_{X\sim K}[X^T X] = I_d$, where I_d is the $d \times d$ identity matrix. Thus, to put a polytope P into isotropic position one has to generate a set of uniform points in its interior and apply to P the transformation that maps the point-set to isotropic position; then iterate this procedure until P is in c-isotropic position [16, 37], for a constant c. In [1] they prove that $O(d)$ points suffice to achieve 2-isotropic position. Alternatively in [24] they compute the maximum volume ellipsoid in P, they map it to the unit ball, and then apply to P the same transformation. They experimentally show that a few iterations suffice to put P in John's position [28]. Moreover, there are a few algorithmic contributions that combine sampling with distribution isotropization steps, e.g., the multi-point walk [5] and the annealing schedule [29].

An important parameter of a random walk is the *walk length*, i.e., the number of the intermediate points that a random walk visits before producing a single sample point. The longer the walk length of a random walk is, the smaller the distance of the current distribution to the stationary (target) distribution becomes. For the majority of random walks there are bounds on the walk length to bound the mixing time with respect to a statistical distance. For example, HR generates a sample from a distribution with total variation distance less than ϵ from the target distribution after $\widetilde{O}(d^3)$ [37] steps, in a well rounded convex body and for log-concave distributions. Similarly, CDHR mixes after a polynomial, in the diameter and the dimension, number of steps [34, 42] for the case of uniform distribution. However, extended practical results have shown that both CDHR and HR converges after $O(d^2)$ steps [10, 16, 24]. The leading algorithms for uniform polytope sampling are the Riemannian Hamiltonian Monte Carlo sampler [35] and the Vaidya walk [13], with mixing times $\widetilde{O}(md^{2/3})$ and $\widetilde{O}(m^{1/2}d^{3/2})$ steps, respectively. However, it is not clear if these random walks can outperform CDHR in practice, because of their high cost per step and numerical instability.

Billiard Walk (BW) [22] is a random walk that employs linear trajectories in a convex body with boundary reflections; alas with an unknown mixing time. The closest guarantees for its mixing time are those of HR and stochastic billiards [17]. Interestingly, [22] shows that, experimentally, BW converges faster than HR for a proper tuning of its parameters. The same conclusion follows from the computation of the volume of zonotopes [11]. It is not known how the sandwiching ratio of P affects the mixing time of BW. Since BW employs reflections on the boundary, we can consider it as a special case of Reflective Hamiltonian Monte Carlo [14].

For almost all random walks the theoretical bounds on their mixing times are pessimistic and unrealistic for computations. Hence, if we terminate the random walk earlier, we generate samples that are usually highly correlated. There are several *MCMC Convergence Diagnostics* [50] to check if the quality of a sample can provide an accurate approximation of the target distribution. For a dependent sample, a powerful diagnostic is the *Effective Sample Size* (ESS). It is the number of effectively independent draws from the target distribution that the Markov chain is equivalent to. For autocorrelated samples, ESS bounds the uncertainty in estimates [20] and provides information about the quality of the sample. There are several statistical tests to evaluate the quality of a generated sample, e.g., potential scale reduction factor (PSRF) [19], maximum mean discrepancy (MMD) [21], and the uniform tests [15]. Interestingly, the copula representation we employ in Fig. 2 to capture the dependence between two fluxes of reactions was also

used successfully in a geometric framework to detect financial crises capturing the dependence between portfolio return and volatility [9].

1.4 Our contribution

We introduce a Multi-phase Monte Carlo Sampling (MMCS) algorithm (Sec. 3 and Alg. 1) to sample from a polytope P. In particular, we split the sampling procedure in phases where, starting from P, each phase uses the sample to round the polytope. This improves the efficiency of the random walk in the next phase, see Fig. 3 and Table 2. For sampling, we propose an improved variant of Billiard Walk (BW) (Sec. 2 and Alg. 2) that enjoys faster arithmetic complexity per step. We also handle efficiently the potential arithmetic inaccuracies near to the boundary, see [14]. We accompany the MMCS algorithm with a powerful MCMC diagnostic, namely the estimation of Effective Sample Size (ESS), to identify a satisfactory convergence to the uniform distribution. However, our method is flexible and we can use any random walk and combination of MCMC diagnostics to decide convergence.

The open-source implementation of our algorithms[1] provides a complete software framework to handle efficiently sampling in metabolic networks. We demonstrate the efficiency of our tools by performing experiments on almost all the metabolic networks that are publicly available and by comparing with the state-of-the-art software packages as `cobra` (Sec. 4.2). Our implementation is faster than `cobra` for low dimensional models, with a speed-up that ranges from 10 to 100 times; this gap on running times increases for bigger models (Table 1). The quality of the sample our software produces is measured with two widely used diagnostics, i.e., ESS and potential scale reduction factor (PSRF) [19]. The highlight of our method is the ability to sample from the most complicated human metabolic network that is accessible today, namely Recon3D. In Fig. 2 we estimate marginal univariate and bivariate flux distributions in Recon3D which validate (a) the quality of the sample by confirming a mutually exclusive pair of biochemical pathways, and that (b) our method indeed generates steady states. In particular, our software can sample $1.44 \cdot 10^5$ points from a 5 335-dimensional polytope in a day using modest hardware. This set of points suffices for the majority of systems biology analytics. To our understanding this task is out of reach for existing software.

Last, MMCS algorithm is quite general sampling scheme and so it has the potential to address other hard computational problems like multivariate integration and volume estimation of polytopes.

2 Efficient Billiard walk

The geometric random walk of our choice to sample from a polytope is based on Billiard Walk (BW) [22], which we modify to reduce the per-step cost.

For a polytope $P = \{x \in \mathbb{R}^d \,|\, Ax \leq b\}$, where $A \in \mathbb{R}^{k \times d}$ and $b \in \mathbb{R}^k$, BW starts from a given point $p_0 \in P$, selects uniformly at random a direction, say v_0, and it moves along the direction of v_0 for length L; it reflects on the boundary if necessary. This results a new point p_1 inside P. We repeat the procedure from p_1. Asymptotically it converges to the uniform distribution over P. The length is $L = -\tau \ln \eta$, where η is a uniform number in $(0, 1)$, that is $\eta \sim \mathcal{U}(0, 1)$, and τ is a predefined constant. It is useful to set a bound, say ρ, on the number of reflections to avoid computationally hard cases where the trajectory may stuck in corners. In [22] they set $\tau \approx diam(P)$ and $\rho = 10d$. Our choices for τ and ρ depend on a burn-in step that we detail in Sec. 4.

At each step of BW we compute the intersection point of a ray, say $\ell := \{p + tv,\ t \in \mathbb{R}_+\}$, with the boundary of P, ∂P, and the normal vector of the tangent plane at the intersection point. The inner vector of the facet that the intersection point belongs to is a row of A. To compute the point $\partial P \cap \ell$ where the first reflection of a BW step takes place, we solve the following m linear equations

$$a_j^T(p_0 + t_j v_0) = b_j \Rightarrow t_j = (b_j - a_j^T p_0)/a_j^T v_0,\ j \in [k], \tag{2}$$

[1] `https://github.com/GeomScale/volume_approximation/tree/socg21`

and keep the smallest positive t_j; a_j is the j-th row of the matrix A. We solve each equation in $O(d)$ operations and so the overall complexity is $O(dk)$. A straightforward approach for BW would consider that each reflection costs $O(kd)$ and thus the per step cost is $O(\rho kd)$. However, our improved version performs more efficiently both *point* and *direction updates* by storing computations from the previous iteration combined with a preprocessing step. The preprocessing step involves the normal vectors of the facets, that takes m^2d operations, and the amortized per-step complexity of BW becomes $O((\rho + d)k)$. The pseudo-code appears in Alg. 2 in the appendix.

► Lemma 1. The amortized per step complexity of BW (Alg. 2) is $O((\rho + d)k)$ after a preprocessing step that takes $O(k^2d)$ operations, where ρ is the maximum number of reflections per step.

Proof. The first reflection of a BW step costs $O(kd)$. During its computation, we store all the values of the inner products $a_j^T x_0$ and $a_j^T v_0$. At the reflection $i > 0$, we start from a point x_i, and the solutions of the corresponding linear equations are

$$a_j^T(p_i + t_j v_i) = b_j \Rightarrow a_j^T(p_{i-1} + t_{i-1}v_{i-1}) + t_j a_j^T(v_{i-1} - 2(v_{i-1}^T a_r)a_r) = b_j$$
$$\Rightarrow t_j = \frac{b_j - a_j^T(p_{i-1} + t_{i-1}v_{i-1})}{a_j^T(v_{i-1} - 2(v_{i-1}^T a_r)a_r)}, \quad \text{for } j \in [k], \tag{3}$$

$$\text{and } \; v_{i+1} = v_i - 2(v_i^T a_l)a_l, \tag{4}$$

where a_r, a_l are the normal vectors of the facets that ℓ hits at reflection $i-1$ and i respectively, and t_{i-1} the solution of the reflection $i-1$. The index l of the normal a_l corresponds to the equation with the smallest positive t_j in (3). We solve each of the equations in (4) in $O(1)$ based on our bookkeeping from the previous reflection. We also store the inner product $v_i^T a_l$ in (4) from the previous reflection. After computing all $a_i^T a_j$ as a preprocessing step, which takes k^2d operations, the total per-step cost of Billiard Walk is $O((d+\rho)k)$. ◄

The use of floating point arithmetic could result to points outside P due to rounding errors when computing boundary points. To avoid this, when we compute the roots in Equation (2) we exclude the facet that the ray hit in the previous reflection.

3 Multiphase Monte Carlo Sampling algorithm

To sample steady states in the flux space of a metabolic network, with m metabolites and n reactions, we introduce a Multiphase Monte Carlo Sampling (MMCS) algorithm; it is multiphase because it consists of a sequence of sampling phases.

Let $S \in \mathbb{R}^{m\times n}$ be the stoichiometric matrix and x_{lb}, $x_{ub} \in \mathbb{R}^n$ bounds on the fluxes. The flux space is the bounded convex polytope

$$\text{FS} := \{x \in \mathbb{R}^n \mid Sx = 0,\ x_{lb} \le x \le x_{ub}\} \subset \mathbb{R}^n. \tag{5}$$

The dimension, d, of FS is smaller than the dimension of the ambient space; that is $d \le n$. To work with a full dimensional polytope we restrict the box induced by the inequalities $x_{lb} \le x \le x_{ub}$ to the null space of S. Let the H-representation of the box be $\left\{x \in \mathbb{R}^n \,\middle|\, \begin{pmatrix} I_n \\ -I_n \end{pmatrix} x \le \begin{pmatrix} x_{ub} \\ x_{lb} \end{pmatrix}\right\}$, where I_n is the $n \times n$ identity matrix, and let $N \in \mathbb{R}^{n\times d}$ be the matrix of the null space of S, that is $S\,N = 0_{m\times d}$. Then $P = \{x \in \mathbb{R}^d \mid Ax \le b\}$, where $A = \begin{pmatrix} I_n N \\ -I_n N \end{pmatrix}$ and $b = \begin{pmatrix} x_{ub} \\ x_{lb} \end{pmatrix} N$, is a full dimensional polytope (in $\mathbb{R}^d$). After we sample (uniformly) points from P, we transform them to uniformly distributed points (that is steady states) in FS by applying the linear map induced by N.

MMCS generates, in a sequence of sampling phases, a set of points, that is almost equivalent to n independent uniformly distributed points in P, where n is given. At each phase, it employs Billiard Walk

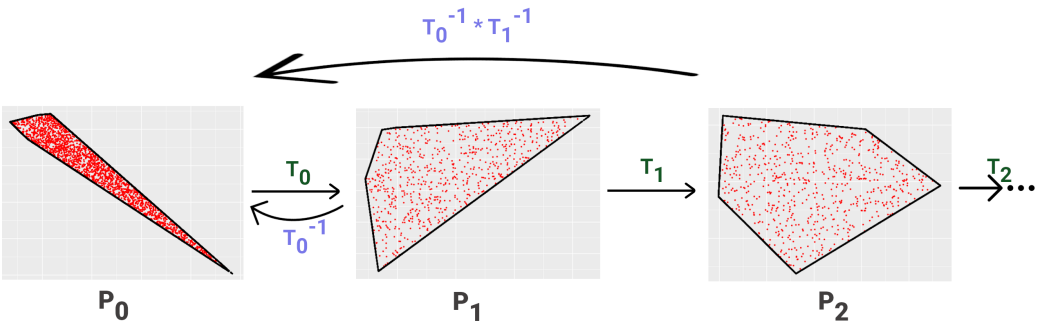

Figure 3 An illustration of our Multiphase Monte Carlo Sampling algorithm. The method is given an integer n and starts at phase $i = 0$ sampling from P_0. In each phase it samples a maximum number of points λ. If the sum of Effective Sample Size in each phase becomes larger than n before the total number of samples in P_i reaches λ then the algorithm terminates. Otherwise, we proceed to a new phase. We map back to P_0 all the generated samples of each phase.

(Section 2) to sample approximate uniformly distributed points, rounding to speedup sampling, and uses the Effective Sample Size (ESS) diagnostic to decide termination. The pseudo-code of the algorithm appears in Alg. 1.

Overview. Initially we set $P_0 = P$.

At each phase $i \geq 0$ we sample at most λ points from P_i. We generate them in chunks; we also call them *chain* of sampling points. Each chain contains at most l points (for simplicity consider $l = O(1)$). To generate the points in each chain we employ BW, starting from a point inside P_i; the starting point is different for each chain. We repeat this procedure until the total number of samples in P_i reaches the maximum number λ; we need $\frac{\lambda}{l}$ chains. To compute a starting point for a chain, we pick a point uniformly at random in the Chebychev ball of P_i and we perform $O(\sqrt{d})$ burn-in BW steps to obtain a warm start.

After we have generated λ sample points we perform a rounding step on P_i to obtain the polytope of the next phase, P_{i+1}. In particular, we compute a linear transformation, T_i, that puts the sample into isotropic position and then $P_{i+1} = T_i(P_i)$. The efficiency of BW improves from one phase to the next one because the sandwiching ratio decreases and so the average number of reflections decreases and thus the convergence to the uniform distribution accelerates (Section 4.2). That is we obtain faster a sample of better quality. Finally, the (product of the) inverse transformations maps the samples to $P_0 = P$. Fig. 3 depicts the procedure.

Termination. There are no bounds on the mixing time of BW [22], hence for termination we rely on ESS. MMCS terminates when the minimum ESS among all the univariate marginals is larger than a requested value. We chose the marginal distributions (of each flux) because they are essential for systems biologists, see [6] for a typical example. In particular, after we generate a chain, the algorithm updates the ESS of each univariate marginal to take into account all the points that we have sampled in P_i, including the newly generated chain. We keep the minimum, say n_i, among all marginal ESS values. If $\sum_{j=0}^{i} n_j$ becomes larger than n before the total number of samples in P_i reaches the upper bound λ, then MMCS terminates. Otherwise, we proceed to the next phase. In summary, MMCS terminates when the sum of the minimum marginal ESS values of each phase reaches n.

Rounding step. This step is motivated by the theoretical result in [1] and the rounding algorithms [37, 16]. We apply the linear transformation T_i to P_i so that the sandwiching ratio of P_{i+1} is smaller than that of P_i. To find the suitable T_i we compute the SVD decomposition of the matrix that contains the sample row-wise [3].

Updating the Effective Sample Size. The effective sample size of a sample of points generated by a process with autocorrelations ρ_t at lag t is function (actually an infinite series) in the ρ_t's; its exact value is unknown. Following [20], we efficiently compute ESS employing a finite sum of monotone estimators $\hat{\rho}_t$ of the autocorrelation at lag t, by exploiting Fast Fourier Transform. Furthermore, given M chains of samples,

Algorithm 1: Multiphase Monte Carlo Sampling$(P, n, l, \lambda, \rho, \tau, W)$

Input : A full dimensional polytope $P \in \mathbb{R}^d$; Requested effectiveness $n \in \mathbb{N}$; l length of each chain; upper bound for the number of generated points in each phase λ; upper bound on the number of reflections ρ; length of trajectory parameter τ; walk length W.

Output : A set of approximate uniformly distributed points $S \in P$

```
Set P_0 ← P, sum_ess ← 0, S ← ∅, i ← 0, T_0 = I_d;
do
    sum_point_phase ← 0, U ← ∅;
    do
        Generate a starting point p ∈ P_i;
        Generate a set Q of l points with Billiard Walk starting from p;
        S ← S ∪ T_i^{-1}(Q);
        U ← U ∪ Q;
        sum_point_phase ← sum_point_phase + l;
        Update ESS n_i of this phase;
        if sum_ess + n_i ≥ n then break ;
    while sum_point_phase < λ;
    sum_ess ← sum_ess + n_i;
    i ← i + 1;
    Compute T such that T(U) is in isotropic position;
    T_i ← T_{i-1} ∘ T;
while sum_ess < n;
return S ;
```

the autocorrelation estimator $\hat{\rho}_t$ is given by,

$$\hat{\rho}_t = 1 - \frac{C - \frac{1}{M}\sum_{i=1}^{M} \hat{\rho}_{t,i}}{B}, \tag{6}$$

where C and B are the within-sample variance estimate and the multi-chain variance estimate given in [19] and $\hat{\rho}_{t,i}$ is an estimator of the autocorrelation of the i-th chain at lag t. To update the ESS, for every new chain of points the algorithm generates, we compute $\hat{\rho}_{t,i}$. Then, using Welford's algorithm we update the average of the estimators of autocorrelation at lag t, as well as the between-chain variance and the within-sample variance estimators given in [19]. Finally, we update the ESS using these estimators.

► Lemma 2. Let $P = \{x \in \mathbb{R}^d \mid Ax \leq b\}$, $A \in \mathbb{R}^{k \times d}$, $b \in \mathbb{R}^k$ a full dimensional polytope in $\mathbb{R}^d$. The total number of operations per phase that Alg. 1 performs, is $O(W(\rho + d)k\lambda + \lambda^2 d + d^3)$, where W is the walk length for Billiard Walk.

Proof. The cost per step of Billiard Walk is $O((\rho + d)k)$. In each phase we generate with Billiard Walk at most λ points with walk length W. Thus, the cost to generate those points is $O(W(\rho + d)k\lambda)$.

To compute the starting point of each chain the algorithm picks a random point uniformly distributed in the Chebychev ball of P and performs $O(1)$ Billiard Walk steps starting from it. The former takes $O(d)$ operations and latter takes $O(W(\rho + d)k)$ operations. The total number of chains is $O(\lambda/l) = O(\lambda)$, as $l = O(1)$. Thus, the total cost to generate all the starting points is $O(d\lambda + W(\rho + d)k\lambda)$. The update of ESS of each univariate marginal takes $O(1)$ operations since $l = O(1)$.

If the termination criterion has not been met after generating λ points, the algorithm computes a linear transformation to put the set of points to isotropic position. We can do this by computing the SVD decomposition of the matrix that contains the set of points row-wise. This corresponds to an SVD of a $\lambda \times d$ matrix and takes $O(\lambda^2 d + d^3)$ operations [26]. ◄

In Section 4 we discuss how to tune the parameters of MMCS to make it more efficient in practice. We also comment on the (practical) complexity of each phase, based on the tuning.

4 Implementation and Experiments

In the sequel we present the implementation of our approach and the tuning of various parameters. We present experiments in an extended set of BURG models [31], including the most complex metabolic networks i.e., the human Recon2D [56] and Recon3D [7]. We end up to sample from polytopes of thousands of dimensions and show that our method can estimate precisely the flux distributions. We analyze various aspects of our method as the runtime, the efficiency and the quality of the output. We compare our method against the state-of-the-art software for the analysis of metabolic networks, which is the `Matlab toolbox` of `cobra` [25]. Our implementation for low dimensional networks is two orders of magnitude faster than `cobra`. As the dimension grows this gap on the run-time increases. The fast mixing of billiard walk allow us to use all the generated samples to approximate each flux distribution improving the flux distribution estimation.

We provide a complete open-source software framework to handle big metabolic networks. The framework loads a metabolic model in some standard format (e.g., `mat, json` files) and performs an analysis of the model e.g., compute the marginal distributions of a given metabolite. All the results in this paper are reproducible using our publicly available code[2]. The core of our implementation is in `C++` to optimize performance while the user interface is implemented in `R`. The package employs `eigen` [23] for linear algebra, `boost` [41] for random number generation, `mosek` [2] as the linear program solver, and expands `volesti` [12] an open-source package for high dimensional sampling and volume approximation. All experiments were performed on a PC with `Intel® Core™ i7-6700 3.40GHz × 8 CPU` and `32GB RAM`.

4.1 Parameter tuning for practical performance

We give details on how we tune various parameters presented in Section 3 in our implementation.

Parameters of Billiard Walk: To employ Billiard Walk (Section 2) we have to make efficient selections for the parameter τ that controls the length of the trajectory in each step, for the maximum number of reflections per step ρ, and for the walk length W of the random walk. We experimentally found that setting $W = 1$ the empirical distribution converges faster to the uniform distribution. Thus, we get a higher ESS faster than the case of $W > 1$. To set τ in phase i, first we set $\tau = 6\sqrt{d}r$ where r is the radius of the Chebychev ball of P_i. Then, we start from the center of the Chebychev ball, we perform $100 + 4\sqrt{d}$ Billiard Walk steps and we store all the points in a set Q. Then we set $\tau = \max\{\max_{q \in Q}\{\|q - p\|_2\}, 6\sqrt{d}r\}$. For the maximum number of reflections we found experimentally that $\rho = 100d$ is violated in less than 0.1% of the total number of Billiard Walk steps in our experiments.

Rounding step: In each phase i of our method, when the minimum value of ESS among all the marginals has not reached the requested threshold, we use the generated sample to perform a rounding step by mapping the points to isotropic position. After computing the SVD decomposition of the point-set we also rescale the singular values such that the smallest one is 1, to improve numerical stability as suggested in [16]. We found experimentally that setting the maximum number of Billiard Walk points per phase $\lambda = 20d$, where d is the dimension of the polytope, suffice to improve the roundness from phase to phase. When, in any phase, the ratio between the maximum over the minimum singular value is smallest than 3 we stop performing any new rounding step. In that case we stay on the current phase until we reach the requested value of ESS.

▶ Remark. Given the Stoichiometric matrix $S \in \mathbb{R}^{m \times n}$ of a metabolic network with flux bounds $x_{lb} \leq x \leq x_{ub}$, the total number of operations per phase that our implementation of Alg. 1 performs, according the

[2] `https://github.com/GeomScale/volume_approximation/tree/socg21`

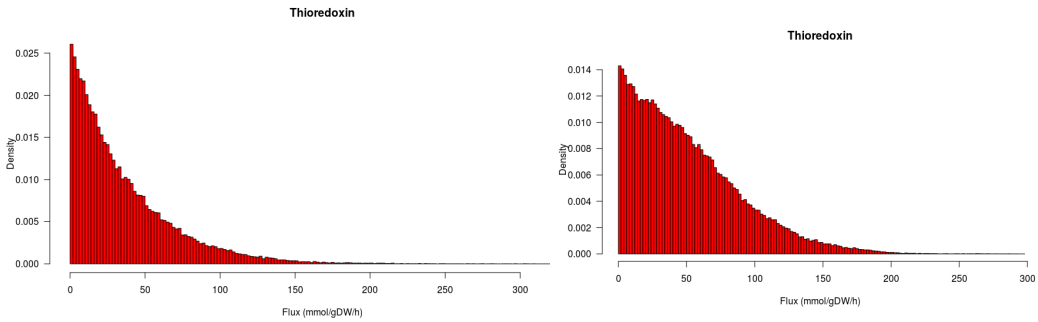

Figure 4 Our method estimation of the marginal distribution of the "Thioredoxin reductace" flux in a constraint-based model of Homo Sapiens metabolism Recon2D [56] (left) and Recon3D [7] (right).

parameterization given in this Section is $O(nd^2)$, where d is the dimension of the null space of S and n is the number of reactions occur in the metabolic network.

4.2 Experiments

We test and evaluate our software on 17 models from the BIGG database [31] and Recon2D, Recon3D from [44]. In particular, we sample from models that correspond to polytopes of dimension less than 100; the simplest model in this setting is the well known bacteria *Escherichia Coli*. We also computed with models that correspond to polytopes of dimension a few thousands; this is the case for Recon2D and Recon3D. We do not employ parallelism for any implementation, thus we report only sequential running times. To assess the quality of our results we employ a second MCMC convergence diagnostic besides the Effective Sample Size (ESS). This is the potential scale reduction factor (PSRF), introduced by Rubin and Gelman [19]. In particular, we compute the PSRF for each univariate marginal of the sample that MMCS outputs. Following [19], a convergence is satisfying according to PSRF when all the marginals have PSRF smaller than 1.1.

The workflow of `cobra` for sampling first performs a rounding step and then samples using Coordinate Directions Hit-and-Run (CDHR). To compare with `cobra` we set the walk length of CDHR according to the empirical suggestion made in [24], i.e., equal to $8d^2$, where d is the dimension of the polytope we sample. For Recon2D we follow the paradigm in [24] which shows that the method converges for walk length equal to $1.57e{+}08$. To have a fair comparison we let `cobra` to sample a minimum number of 1 000 points. If in the computed sample there is a marginal with PSRF larger than 1.1 we continue sampling until all PSRFs are smaller than 1.1.

In Table 1 we report the results of MMCS and `cobra`. We run MMCS until we get a value of ESS equal to 1 000; meaning that we stop when the sum over all phases of the minimum values of ESS among all the marginals is larger than 1 000. Moreover, in Table 1 all the marginals of the sample that MMCS returns have PSRF < 1.1. This is another statistical evidence on the quality of the generated sample. The histograms in Fig. 4 illustrate an approximation for the flux distribution of the reaction Thioredoxin as computed in Recon2D and Recon3D respectively. The same marginal flux distribution in Recon2D was estimated also in [24]. Notice that the estimated density slightly changes in Recon3D as the stoichiometric matrix has been updated and thus the corresponding marginal is affected. In Fig. 2 we also employ the copula representation to capture the dependence between two fluxes of reactions to confirm a mutually exclusive pair of biochemical pathways. Notice that the run-time of MMCS is one or two orders of magnitude smaller than the run-time of `cobra` and this gap becomes much larger for higher dimensional models such as Recon2D and Recon3D.

For some models –we report them in Table 3– we introduce a further improvement to obtain a better convergence. If there is a marginal in the generated sample from MMCS that has a PSRF larger than 1.1 then we do not take into account the k first phases, starting with $k = 1$ until we get both ESS equal to 1 000

				MMCS		cobra	
name	(m)	(n)	(d)	Time (sec)	(N)	Time (sec)	(N)
e_coli_core	72	95	24	6.50e-01	3.40e+03 (8)	7.20e+01	4.61e+06
iLJ478	570	652	59	9.00e+00	5.40e+03 (5)	4.54e+02	2.79e+07
iSB619	655	743	83	1.70e+01	8.20e+03 (5)	9.56e+02	5.51e+07
iHN637	698	785	88	2.00e+01	6.80e+03 (4)	1.03e+03	6.19e+07
iJN678	795	863	91	2.50e+01	8.10e+03 (4)	1.17e+03	6.62e+07
iNF517	650	754	92	1.70e+01	6.20e+03 (4)	1.33e+03	6.77e+07
iJN746	907	1054	116	5.70e+01	8.70e+03 (3)	2.22e+03	1.07e+08
iAB_RBC_283	342	469	130	5.20e+01	1.07e+04 (5)	7.85e+03	4.05e+08
iJR904	761	1075	227	2.98e+02	1.62e+04 (4)	8.81e+03	4.12e+08
iAT_PLT_636	738	1008	289	3.25e+02	1.04e+04 (2)	1.73e+04	6.68e+08
iSDY_1059	1888	2539	509	2.813e+03	2.31e+04 (3)	6.66e+04	2.07e+09
iAF1260	1668	2382	516	6.84e+03	5.33e+04 (6)	7.04e+04	2.13e+09
iEC1344_C	1934	2726	578	4.86e+03	3.95e+04 (4)	9.42e+04	2.67e+09
iJO1366	1805	2583	582	6.02e+03	5.14e+04 (5)	9.99e+04	2.71e+09
iBWG_1329	1949	2741	609	3.06e+03	4.22e+04 (4)	1.05e+05	2.97e+09
iML1515	1877	2712	633	4.65e+03	5.65e+04 (5)	1.15e+05	3.21e+09
Recon1	2766	3741	931	8.09e+03	1.94e+04 (2)	3.20e+05	6.93e+09
Recon2D	5063	7440	2430	2.48e+04	5.44e+04 (2)	~ 140 days	1.57e+11
Recon3D	8399	13543	5335	1.03e+05	1.44e+05 (2)	–	–

Table 1 17 metabolic networks from [31] and Recon2D, Recon3D from [44]; (m) the number of Metabolites, (n) the number of Reactions, (d) the dimension of the polytope; (N) is the total number of sampled points × walk length; for MMCS we stop when the sum of the minimum value of ESS among all the univariate marginals in each phase is 1000 (we report the number of phases in parenthesis); for `cobra` we set the walk length to $8d^2$ and 1.57e+08 for Recon2D following [24], sample at least 1000 points and stop when all marginals have PSRF < 1.1; the runtime of `cobra` for Recon2D is an estimation of the sequential time just for the purpose of the comparison in this paper.

and all the PSRF values smaller than 1.1 for all the marginals. By "do not take into account" we mean that we neither store the generated sample –for the first k phases– nor we sum up its ESS to the overall ESS considered for termination by MMCS. Note that for these models it is not practical to repeat MMCS runs for different k until we get the required PSRF value. We can obtain the final results –reported in Tables 1– in one pass. We simply drop a phase when the ESS reaches the requested value but the PSRF is not smaller than 1.1 for all the marginals. In Table 3 we separately report the MMCS runs for different k just for performance analysis reasons.

Interestingly, the total number of Billiard Walk steps –and consequently the run-time– does not increase as k increases in Table 3. This means that the performance of our method improves for these models, when we do not take into account the k first phases of MMCS. This happens because the performance of Billiard Walk improves as the polytope becomes more rounded from phase to phase. In particular, in Table 2 we analyze the performance of Billiard Walk for the model iAF1260. We sample $20d$ points per phase with walk length equal to 1 and we report the average number of reflections, the ESS, the run-time, and the ratio $\sigma_{\max}/\sigma_{\min}$ per phase. The latter is the ratio between the maximum over the minimum singular value of the point-set. The larger this ratio is the more skinny the polytope of the corresponding phase is. As the method progresses from the first to the last phase, the average number of reflections and the run-time decrease and the ESS increases. This means that as the polytope becomes more rounded from phase to phase, the Billiard Walk step becomes faster and the generated sample has better quality. This explains why the total run-time does not increase when we do not take into account the first k phases: the initial phases are slow and they contribute poorly to the quality of the final sample; the last phases are fast and contribute with more accurate samples.

Sampling from iAF1260				
Phase	Avg. #reflections	ESS	$\frac{\sigma_{\max}}{\sigma_{\min}}$	Time (sec)
1st	7819	67	43459	2271
2nd	4909	68	922	1631
3rd	3863	77	582	1278
4th	3198	71	360	1080
5th	1300	592	29	454
6th	1187	4821	3.5	417
7th	1181	4567	2.8	415

Table 2 We sample $20d = 10320$ points per phase with Billiard Walk and walk length equal to 1, where $d = 516$ is the dimension of the corresponding polytope. For each phase we report the average number of reflections per BIlliard Walk step, the the minimum value of Effective Sample Size among all the univariate marginals, the ratio between the maximum over the minimum singular value derived from the SVD decomposition of the generated sample and the run-time.

References

1 Radosław Adamczak, Alexander Litvak, Alain Pajor, and Nicole Tomczak-Jaegermann. Quantitative estimates of the convergence of the empirical covariance matrix in log-concave ensembles. *Journal of the American Mathematical Society*, 23(2):535–561, 2010. doi:10.1090/S0894-0347-09-00650-X.

2 MOSEK ApS. *The MOSEK optimization toolbox for R manual. Version 9.2.*, 2019. URL: https://docs.mosek.com/9.2/rmosek/index.html.

3 Shiri Artstein-Avidan, Haim Kaplan, and Micha Sharir. On radial isotropic position: Theory and algorithms, 2020. arXiv:2005.04918.

4 David B Bernstein, Floyd E Dewhirst, and Daniel Segre. Metabolic network percolation quantifies biosynthetic capabilities across the human oral microbiome. *Elife*, 8:e39733, 2019.

5 Dimitris Bertsimas and Santosh Vempala. Solving convex programs by random walks. *J. ACM*, 51(4):540–556, July 2004. doi:10.1145/1008731.1008733.

6 Sergio Bordel, Rasmus Agren, and Jens Nielsen. Sampling the solution space in genome-scale metabolic networks reveals transcriptional regulation in key enzymes. *PLOS Computational Biology*, 6(7):1–13, 07 2010. doi:10.1371/journal.pcbi.1000859.

7 Elizabeth Brunk, Swagatika Sahoo, Daniel C Zielinski, Ali Altunkaya, Andreas Dräger, Nathan Mih, Francesco Gatto, Avlant Nilsson, German Andres Preciat Gonzalez, Maike Kathrin Aurich, et al. Recon3D enables a three-dimensional view of gene variation in human metabolism. *Nature biotechnology*, 36(3):272, 2018.

8 Ali Cakmak, Xinjian Qi, A Ercument Cicek, Ilya Bederman, Leigh Henderson, Mitchell Drumm, and Gultekin Ozsoyoglu. A new metabolomics analysis technique: steady-state metabolic network dynamics analysis. *Journal of bioinformatics and computational biology*, 10(01):1240003, 2012.

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

10 Apostolos Chalkis, Ioannis Z. Emiris, and Vissarion Fisikopoulos. Practical volume estimation by a new annealing schedule for cooling convex bodies, 2019. arXiv:1905.05494.

11 Apostolos Chalkis, Ioannis Z. Emiris, and Vissarion Fisikopoulos. Practical volume estimation of zonotopes by a new annealing schedule for cooling convex bodies. In Anna Maria Bigatti, Jacques Carette, James H. Davenport, Michael Joswig, and Timo de Wolff, editors, *Mathematical Software – ICMS 2020*, pages 212–221, Cham, 2020. Springer International Publishing.

12 Apostolos Chalkis and Vissarion Fisikopoulos. volesti: Volume approximation and sampling for convex polytopes in R, 2020. https://github.com/GeomScale/volume_approximation. arXiv:2007.01578.

13 Yuansi Chen, Raaz Dwivedi, Martin J. Wainwright, and Bin Yu. Fast mcmc sampling algorithms on polytopes. *Journal of Machine Learning Research*, 19(55):1–86, 2018. URL: http://jmlr.org/papers/v19/18-158.html.

14 A. Chevallier, S. Pion, and F. Cazals. Hamiltonian Monte Carlo with boundary reflections, and application to polytope volume calculations. Research Report RR-9222, INRIA Sophia Antipolis, France, 2018. URL: https://hal.archives-ouvertes.fr/hal-01919855.

15 B. Cousins. *Efficient high-dimensional sampling and integration*. PhD thesis, Georgia Institute of Technology, Georgia, U.S.A., 2017.

16 Ben Cousins and Santosh Vempala. A practical volume algorithm. *Mathematical Programming Computation*, 8(2):133–160, 2016.

17 A. B. Dieker and Santosh S. Vempala. Stochastic billiards for sampling from the boundary of a convex set. *Mathematics of Operations Research*, 40(4):888–901, 2015. URL: http://www.jstor.org/stable/24540983.

18 Shirin Fallahi, Hans J Skaug, and Guttorm Alendal. A comparison of Monte Carlo sampling methods for metabolic network models. *PLOS One*, 15(7):e0235393, 2020.

19 Andrew Gelman and Donald B. Rubin. Inference from Iterative Simulation Using Multiple Sequences. *Statistical Science*, 7(4):457–472, 1992. Publisher: Institute of Mathematical Statistics. URL: https://www.jstor.org/stable/2246093.

20 Charles J. Geyer. Practical Markov chain Monte Carlo. *Statist. Sci.*, 7(4):473–483, 11 1992. doi:10.1214/ss/1177011137.

21 Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012. URL: http://jmlr.org/papers/v13/gretton12a.html.

22 Elena Gryazina and Boris Polyak. Random sampling: Billiard walk algorithm. *European Journal of Operational Research*, 238(2):497 – 504, 2014. doi:https://doi.org/10.1016/j.ejor.2014.03.041.

23 Gaël Guennebaud, Benoît Jacob, et al. *Eigen v3*, 2010. URL: http://eigen.tuxfamily.org.

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

25 Laurent Heirendt, Sylvain Arreckx, Thomas Pfau, Sebastián N Mendoza, Anne Richelle, Almut Heinken, Hulda S Haraldsdóttir, Jacek Wachowiak, Sarah M Keating, Vanja Vlasov, et al. Creation and analysis of biochemical constraint-based models using the cobra toolbox v. 3.0. *Nature protocols*, 14(3):639–702, 2019.

26 Gene H.Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2013.

27 Trey Ideker, Timothy Galitski, and Leroy Hood. A new approach to decoding life: systems biology. *Annual review of genomics and human genetics*, 2(1):343–372, 2001.

28 Fritz John. Extremum Problems with Inequalities as Subsidiary Conditions. In Giorgio Giorgi and Tinne Hoff Kjeldsen, editors, *Traces and Emergence of Nonlinear Programming*, pages 197–215. Springer, Basel, 2014. doi:10.1007/978-3-0348-0439-4_9.

29 Adam Tauman Kalai and Santosh Vempala. Simulated annealing for convex optimization. *Mathematics of Operations Research*, 31(2):253–266, 2006. URL: http://www.jstor.org/stable/25151723.

30 David E Kaufman and Robert L Smith. Direction choice for accelerated convergence in hit-and-run sampling. *Operations Research*, 46(1):84–95, 1998.

31 Zachary A King, Justin Lu, Andreas Dräger, Philip Miller, Stephen Federowicz, Joshua A Lerman, Ali Ebrahim, Bernhard Ø. Palsson, and Nathan E Lewis. Bigg models: A platform for integrating, standardizing and sharing genome-scale models. *Nucleic acids research*, 44(D1):D515–D522, 2016.

32 Edda Klipp, Wolfram Liebermeister, Christoph Wierling, and Axel Kowald. *Systems biology: a textbook*. John Wiley & Sons, 2016.

33 Peter Kohl, Edmund J Crampin, TA Quinn, and Denis Noble. Systems biology: an approach. *Clinical Pharmacology & Therapeutics*, 88(1):25–33, 2010.

34 Aditi Laddha and Santosh Vempala. Convergence of Gibbs Sampling: Coordinate Hit-and-Run Mixes Fast, 2020. `arXiv:2009.11338`.

35 Yin Tat Lee and Santosh S. Vempala. 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*, STOC 2018, page 1115–1121, New York, NY, USA, 2018. Association for Computing Machinery. `doi:10.1145/3188745.3188774`.

36 László Lovász, Ravi Kannan, and Miklós Simonovits. Random walks and an $O^*(n^5)$ volume algorithm for convex bodies. *Random Structures and Algorithms*, 11:1–50, 1997.

37 László Lovász and Santosh Vempala. Simulated annealing in convex bodies and an $O^*(n^4)$ volume algorithms. *J. Computer & System Sciences*, 72:392–417, 2006.

38 Maximilian Lularevic, Andrew J Racher, Colin Jaques, and Alexandros Kiparissides. Improving the accuracy of flux balance analysis through the implementation of carbon availability constraints for intracellular reactions. *Biotechnology and bioengineering*, 116(9):2339–2352, 2019.

39 Michael MacGillivray, Amy Ko, Emily Gruber, Miranda Sawyer, Eivind Almaas, and Allen Holder. Robust analysis of fluxes in genome-scale metabolic pathways. *Scientific Reports*, 7, 12 2017. `doi:10.1038/s41598-017-00170-3`.

40 Daniel Machado, Sergej Andrejev, Melanie Tramontano, and Kiran Raosaheb Patil. Fast automated reconstruction of genome-scale metabolic models for microbial species and communities. *Nucleic acids research*, 46(15):7542–7553, 2018.

41 Jens Maurer and Steven Watanabe. Boost random number library. Software, 2017. URL: `https://www.boost.org/doc/libs/1_73_0/doc/html/boost_random.html`.

42 Hariharan Narayanan and Piyush Srivastava. On the mixing time of coordinate hit-and-run, 2020. `arXiv:2009.14004`.

43 Denis Noble. *The music of life: biology beyond genes*. Oxford University Press, 2008.

44 Alberto Noronha, Jennifer Modamio, Yohan Jarosz, Elisabeth Guerard, Nicolas Sompairac, German Preciat, Anna Dröfn Daníelsdóttir, Max Krecke, Diane Merten, Hulda S Haraldsdóttir, Almut Heinken, Laurent Heirendt, Stefanía Magnúsdóttir, Dmitry A Ravcheev, Swagatika Sahoo, Piotr Gawron, Lucia Friscioni, Beatriz Garcia, Mabel Prendergast, Alberto Puente, Mariana Rodrigues, Akansha Roy, Mouss Rouquaya, Luca Wiltgen, Alise Žagare, Elisabeth John, Maren Krueger, Inna Kuperstein, Andrei Zinovyev, Reinhard Schneider, Ronan M T Fleming, and Ines Thiele. The Virtual Metabolic Human database: integrating human and gut microbiome metabolism with nutrition and disease. *Nucleic Acids Research*, 47(D1):D614–D624, 10 2018. `doi:10.1093/nar/gky992`.

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

46 Bernhard Ø. Palsson. Metabolic systems biology. *FEBS letters*, 583(24):3900–3904, 2009.

47 Bernhard Ø. Palsson. *Systems biology*. Cambridge university press, 2015.

48 Octavio Perez-Garcia, Gavin Lear, and Naresh Singhal. Metabolic network modeling of microbial interactions in natural and engineered environmental systems. *Frontiers in microbiology*, 7:673, 2016.

49 Robert A. Quinn, Jose A. Navas-Molina, Embriette R. Hyde, Se Jin Song, Yoshiki Vázquez-Baeza, Greg Humphrey, James Gaffney, Jeremiah J. Minich, Alexey V. Melnik, Jakob Herschend, Jeff DeReus, Austin Durant, Rachel J. Dutton, Mahdieh Khosroheidari, Clifford Green, Ricardo da Silva, Pieter C. Dorrestein, and Rob Knight. From sample to multi-omics conclusions in under 48 hours. msystems 1: e00038-16. *Crossref, Medline*, 2016.

50 Vivekananda Roy. Convergence Diagnostics for Markov Chain Monte Carlo. *Annual Review of Statistics and Its Application*, 7(1):387–412, 2020. `doi:10.1146/annurev-statistics-031219-041300`.

51 Pedro A. Saa and Lars K. Nielsen. ll-ACHRB: a scalable algorithm for sampling the feasible solution space of metabolic networks. *Bioinform.*, 32(15):2330–2337, 2016. `doi:10.1093/bioinformatics/btw132`.

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

53 John R Schramski, Anthony I Dell, John M Grady, Richard M Sibly, and James H Brown. Metabolic theory predicts whole-ecosystem properties. *Proceedings of the National Academy of Sciences*, 112(8):2617–2622, 2015.

54 Siamak S. Shishvan, Andrea Vigliotti, and Vikram S. Deshpande. The homeostatic ensemble for cells. *Biomechanics and Modeling in Mechanobiology*, 17(6):1631–1662, 2018.

55 Robert L. Smith. Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research*, 32(6):1296–1308, 1984.

56 Neil Swainston, Kieran Smallbone, Hooman Hefzi, Paul D. Dobson, Judy Brewer, Michael Hanscho, Daniel C. Zielinski, Kok Siong Ang, Natalie J. Gardiner, Jahir M. Gutierrez, Sarantos Kyriakopoulos, Meiyappan Lakshmanan, Shangzhong Li, Joanne K. Liu, Veronica S. Martínez, Camila A. Orellana, Lake-Ee Quek, Alex Thomas, Juergen Zanghellini, Nicole Borth, Dong-Yup Lee, Lars K. Nielsen, Douglas B. Kell, Nathan E. Lewis, and Pedro Mendes. Recon 2.2: from reconstruction to model of human metabolism. *Metabolomics*, 12(7):109, June 2016. doi:10.1007/s11306-016-1051-4.

57 Ines Thiele and Bernhard Ø. Palsson. A protocol for generating a high-quality genome-scale metabolic reconstruction. *Nature protocols*, 5(1):93, 2010.

A The Billiard walk algorithm

Algorithm 2: Billiard Walk(P, p, ρ, τ, W)

Input : polytope P; point p; upper bound on the number of reflections ρ; length of trajectory parameter τ; walk length W.
Require: point $p \in P$
Output : A point in P

```
for j = 1, ..., W do
    L ← −τ ln η, η ~ U(0, 1) // length of the trajectory
    i ← 0 // current number of reflections
    p_0 ← p // initial point of the step
    pick a uniform vector v_0 from the boundary of the unit ball
    do
        ℓ ← {p_i + t v_i, 0 ≤ t ≤ L} // segment
        if ∂P ∩ ℓ = ∅ then
            p_{i+1} ← p_i + L v_i ;
            break ;
        p_{i+1} ← ∂P ∩ ℓ ; // point update
        the inner vector, s, of the tangent plane at p, s.t. ||s|| = 1;
        L ← L − |P ∩ ℓ|;
        v_{i+1} ← v_i − 2(v_i^T s)s // direction update
        i ← i + 1;
    while i ≤ ρ;
    if i = ρ then p ← p_0 else p ← p_i ;
return p;
```

B Additional experiments

Sampling from iAF1260				
Do not take into account the sample of the k first phases	Time (sec)	PSRF < 1.1	(M)	(N)
0 first phases	6955	41%	6	56100
1 first phases	6943	56%	6	54100
2 first phases	6890	76%	6	55200
3 first phases	6867	95%	6	53200
4 first phases	6840	100%	6	53300
Sampling from iBWG_1329				
0 first phases	3067	50%	4	42100
1 first phases	3189	97%	5	48800
2 first phases	4652	100%	5	56500
Sampling from iEC1344_C				
0 first phases	4845	77%	4	41100
1 first phases	4721	96%	4	42500
2 first phases	4682	100%	4	39500
Sampling from iJO1366				
0 first phases	3708	66%	5	51500
1 first phases	6022	100%	5	51400

Table 3 We run our method and we do not take into account the sample of the k first phases, thus we do not also count the value of the Effective Sample Size (ESS) in those phases, before we start storing the generated sample and sum up the ESS of each phase. In all cases MMCS stops when the sum of ESS reaches 1000. For each case we report the total run-time, the percentage of the marginals that have PSRF smaller than 1.1 and the total number of phases (M) generates included the k first phases and the total number of Billiard Walk steps (N) included those performed in the k first phases.