

Systems biology

PolyRound: polytope rounding for random sampling in metabolic networks

Axel Theorell^{1,2,*}, Johann F. Jadebeck^{2,3}, Katharina Nöh² and Jörg Stelling ^{1,*}

¹Department of Biosystems Science and Engineering, SIB Swiss Institute of Bioinformatics, 4058 Basel, Switzerland, ²Institute of Bio- and Geosciences, IBG-1: Biotechnology, Forschungszentrum Jülich, 52425 Jülich, Germany and ³Computational Systems Biotechnology (AVT.CSB), RWTH Aachen University, 52062 Aachen, Germany

*To whom correspondence should be addressed.

Associate Editor: Alfonso Valencia

Received on December 1, 2020; revised on May 25, 2021; editorial decision on July 24, 2021; accepted on July 29, 2021

Abstract

Summary: Random flux sampling is a powerful tool for the constraint-based analysis of metabolic networks. The most efficient sampling method relies on a rounding transform of the constraint polytope, but no available rounding implementation can round all relevant models. By removing redundant polytope constraints on the go, PolyRound simplifies the numerical problem and rounds all the 108 models in the BiGG database without parameter tuning, compared to ~50% for the state-of-the-art implementation.

Availability and implementation: The implementation is available on gitlab: <https://gitlab.com/csb.ethz/PolyRound>.

Contact: axel.theorell@bsse.ethz.ch or joerg.stelling@bsse.ethz.ch

Supplementary information: [Supplementary data](#) are available at *Bioinformatics* online.

1 Introduction

Random sampling of constraint-based models of metabolism is a powerful approach to characterize the potential behaviors of metabolic networks (Schellenberger and Palsson, 2009; Herrmann *et al.*, 2019). The field is actively developed, for example, with recent extensions to model inference (Theorell and Nöh, 2020). Algorithmically, Markov Chain Monte Carlo (MCMC) based coordinate hit-and-run with rounding (CHRR) (Haraldsdóttir *et al.*, 2017) showed superior performance in computational benchmarks (Herrmann *et al.*, 2019) and it is available in a highly efficient and modular implementation (Jadebeck *et al.*, 2021).

The relevant sampling space is a polytope P : a bounded set in $\mathcal{R}^n$ constrained by hyperplanes. In constraint based models, P originates from stoichiometric reaction constraints and capacity constraints that give rise to equalities and inequalities. $P := \{x \in \mathcal{R}^n : A_{eq}x = b_{eq}, A_{ineq}x \leq b_{ineq}\}$ with matrices $A_{eq} \in \mathcal{R}^{m,n}$ and $A_{ineq} \in \mathcal{R}^{k,n}$, and vectors $b_{eq} \in \mathcal{R}^m$ and $b_{ineq} \in \mathcal{R}^k$. For Hit-and-Run samplers, such as CHRR, the asymptotic mixing time (a common efficiency measure in MCMC) depends quadratically on the sandwiching ratio, the ratio of radii of the largest sphere contained in P and the smallest sphere containing P (Lovász and Vempala, 2006). Efficient random sampling therefore relies on an efficient ‘rounding’ preprocessing step: it applies a linear transformation to make the polytope more spherical. In practice, sampling the rounded polytope converges within minutes, whereas sampling the unrounded polytope fails to converge in reasonable time.

Deterministic and stochastic algorithms for polytope rounding exist (Mangoubi and Vishnoi, 2019; Martino *et al.*, 2015). All current implementations of CHRR rely on deterministic search for the

maximum volume ellipsoid (MVE) (Zhang and Gao, 2003); after rounding via a linear transform, the MVE equals the unit sphere. An implementation that handles polytopes formulated as P is interfaced from the CobraToolbox (CT) (Heirendt *et al.*, 2019). However, we found that it rounds only about half of the models in the BiGG repository (King *et al.*, 2016) due to numerical failures. Because this strongly limits the scope of models for random sampling, we provide PolyRound, an open source Python toolbox that uses a modified reformulation and rounding scheme optimized for robustness.

2 Materials and methods

2.1 Workflow

In a first step, PolyRound reformulates P in a form with only inequality constraints and embeds it in a space where it has non-zero hypervolume. With the null space matrix $N \in \mathcal{R}^{l,n}$ of A_{eq} (computed by SVD), we express P in the null space coordinates $u = Nx + x_0$ (x_0 is an arbitrary feasible point) using only inequality constraints. However, P may still have zero hyper volume, since the inequality constraints may contain the so-called 0-facets, directions in which the width of P is 0. Therefore, prior to reformulation, PolyRound computes all facet widths by sequentially locating the minimal and maximal feasible point in the direction orthogonal to each inequality constraint. This requires solving two linear programs (LPs) per constraint. If the width is smaller than a threshold (e.g. 10^{-7}), the corresponding constraint is a de-facto equality constraint and is moved to the equality system. PolyRound also checks for redundant inequality constraints and removes these, using a third LP in which the right

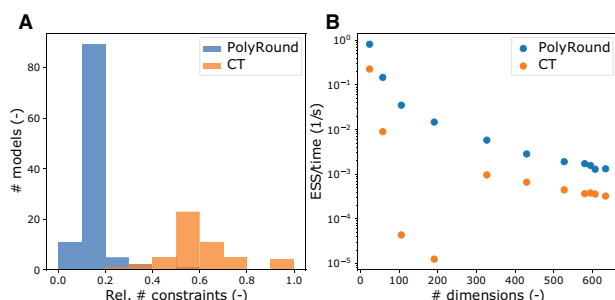


Fig. 1. (a) Fraction of constraints after rounding (number of rows of the processed inequality matrix), relative to the number for unrounded BiGG models (number of rows of A_{ineq}). Results for CT: best commit and default parameters. Due to failed rounding, the orange bars have ~50% of the surface area of the blue bars. (b) ESS per time for a selection of models (Supplementary Table S1). The number of dimensions refers to the PolyRound processed models

hand side of the constraint under investigation is relaxed. After all redundant constraints have been removed and 0-facet constraints have been refunctioned, the now smaller problem can be solved more accurately, so that new redundant constraints and 0-facets may be detected. Therefore, PolyRound iterates until no more changes of P are induced.

In a second step, PolyRound computes the MVE using the F2PD algorithm by Zhang and Gao (2003). The PolyRound MVE is an optimized python implementation of CT's default routine Bounciness/Volume-and-Sampling (Haraldsdóttir et al., 2017), including an iterative scheme that improves numerical stability (Supplementary Information S3).

2.2 Implementation

PolyRound is implemented in Python 3 and equipped with an easy-to-use command line interface. It reads constraint based models in SBML format (Hucka et al., 2003) using COBRApy (Ebrahim et al., 2013), and generic polytopes in plain HDF5 and csv representations. LPs are solved via optlang (Jensen et al., 2017), enabling easy use of different solvers. To reproduce the benchmarks, see Supplementary Information.

3 Benchmarks

We first assessed success in terms of obtaining a rounded polytope. PolyRound successfully rounded 100% of the 108 models in the BiGG database (King et al., 2016) (Fig. 1A), compared to at most 51% (Supplementary Fig. S1) for CT, using different parameters and versions. The performance difference is due to PolyRound's primary invention, removal of redundant constraints: without it, PolyRound's success rate dropped to 67% (see Supplementary Information). PolyRound and CT produced similar reductions in dimensionality, compared to the expected original dimensionality, but PolyRound achieved substantially larger reductions of the number of constraints (Fig. 1A, Supplementary Fig. S2), thus easing numerical computations in the rounding workflow. To validate that the polytopes generated by PolyRound yield efficient sampling, we collected 11 models spanning a range of sizes (Supplementary Table S1). The effective sample size (ESS) per time for uniform

sampling with the HOPS library (Jadebeck et al., 2021) was consistently higher for PolyRound than for CT (Fig. 1B).

4 Conclusion

PolyRound is an open-source, robust implementation of polytope rounding, which, by numerical craftsmanship, strongly widens the number of constraint based models for random sampling. Rounded models are maximally constraint reduced, which speeds up later computations.

Acknowledgements

The authors thank Mattia Gollub for fruitful discussions and feedback.

Funding

This work was supported by the Swiss National Science Foundation (Sinergia project #177164 to J.S.) and the Helmholtz School for Data Science in Life, Earth and Energy (HDS-LEE).

Conflict of Interest: none declared.

References

- Ebrahim, A. et al. (2013) COBRApy: constraints-based reconstruction and analysis for python. *BMC Syst. Biol.*, 7, 74.
- Haraldsdóttir, H.S. et al. (2017) CHRR: coordinate hit-and-run with rounding for uniform sampling of constraint-based models. *Bioinformatics*, 33, 1741–1743.
- Heirendt, L. et al. (2019) Creation and analysis of biochemical constraint-based models using the COBRA toolbox v.3.0. *Nat. Protocols*, 14, 639–702.
- Herrmann, H.A. et al. (2019) Flux sampling is a powerful tool to study metabolism under changing environmental conditions. *NPJ Syst. Biol. Appl.*, 5, 32.
- Hucka, M., et al.; SBML Forum. (2003) The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19, 524–531.
- Jadebeck, J.F. et al. (2021) HOPS: high-performance library for (non-) uniform sampling of convex-constrained models. *Bioinformatics*, 37, 1776–1777.
- Jensen, K. et al. (2017) Optlang: an algebraic modeling language for mathematical optimization. *J. Open Source Software*, 2, 139.
- King, Z.A. et al. (2016) BiGG models: a platform for integrating, standardizing and sharing genome-scale models. *Nucleic Acids Res.*, 44, D515–D522.
- Lovász, L. and Vempala, S. (2006) Hit-and-run from a corner. *SIAM J. Comput.*, 35, 985–1005.
- Mangoubi, O. and Vishnoi, N.K. (2019). Faster polytope rounding, sampling, and volume computation via a sub-linear ball walk. In: *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 1338–1357. IEEE.
- Martino, D.D. et al. (2015) Uniform sampling of steady states in metabolic networks: heterogeneous scales and rounding. *PLoS One*, 10, e0122670.
- Schellenberger, J. and Palsson, B.Ø. (2009) Use of randomized sampling for analysis of metabolic networks. *J. Biol. Chem.*, 284, 5457–5461.
- Theorell, A. and Nöh, K. (2020) Reversible jump MCMC for multi-model inference in metabolic flux analysis. *Bioinformatics*, 36, 232–240.
- Zhang, Y. and Gao, L. (2003) On numerical solution of the maximum volume ellipsoid problem. *SIAM J. Optim.*, 14, 53–76.