


DBSOP: An Efficient Heuristic for Speedy MCMC Sampling on Polytopes

A preprint

Christos Karras¹  and Aristeidis Karras¹ 

Computer Engineering and Informatics Department,
University of Patras, Patras, Hellas
{c.karras, akarras}@ceid.upatras.gr

Abstract. Markov Chain Monte Carlo (MCMC) techniques have long been studied in computational geometry subjects whereabouts the problems to be studied are complex geometric objects which by their nature require optimized techniques to be deployed or to gain useful insights by them. MCMC approaches are directly answering to geometric problems we are attempting to answer, and how these problems could be deployed from theory to practice. Polytope which is a limited volume in n -dimensional space specified by a collection of linear inequality constraints require specific approximation. Therefore, sampling across density based polytopes can not be performed without the use of such methods in which the amount of repetition required is defined as a property of error margin. In this work we propose a simple accurate sampling approach based on the triangulation (tessellation) of a polytope. Moreover, we propose an efficient algorithm named Density Based Sampling on Polytopes (DBSOP) for speedy MCMC sampling where the time required to perform sampling is significantly lower compared to existing approaches in low dimensions with complexity $\mathcal{O}^*(n^3)$. Ultimately, we highlight possible future aspects and how the proposed scheme can be further improved with the integration of reservoir-sampling based methods resulting in more speedy and efficient solution.

Keywords: Polytopes · Uniform Sampling · Data Engineering · Convex Bodies · Markov Chain · Monte Carlo · Hit-and-Run Methods

1 Introduction

The sampling process across different distributions is a major topic in statistics, probability, systems engineering, as well as other disciplines that use stochastic models ([6],[11],[12],[25],[26]). Before Monte-Carlo techniques may be used to estimate anticipated values and other integrals, sampling algorithms must first be developed and implemented. In recent decades, Markov Chain Monte Carlo (MCMC) algorithms have gained remarkable success; for example, the book [7] and the references therein discuss this issue in great detail. These tactics are predicated on the creation of a Markov model with a density function that

matches the goal distribution in which the chain is simulated for a set number of steps to generate samples. MCMC algorithms offer the benefit of requiring just wisdom of the desired density up to a ratio constant, significantly reducing the quantity of data required. On the other hand, theoretical knowledge of the MCMC methods that are employed in practice is far from adequate. It is critical to control the decomposition rate of a MCMC operation, which can be defined as the amount of repetitions required as a property of error margin, issue element n , and other variables for the chain to land on a distribution that is well within a specific range from the objective.

1.1 Problem definition

We are concerned with the issue of sampling from a uniform density across a convex polytope¹, which is a limited volume in n -dimensional space specified by a collection of linear inequality constraints, and we are interested in sampling from a convex polytope. There are several applications for this sampling issue, but we are particularly interested in its application to the sampling of weight vectors for multi-class discriminant analysis (MCDA). Previous research has shown that the method of Hit-n-Run may be often employed to this particular use case [23]. Hit-n-Run has the drawback of being a MCMC method, which necessitates that use of convergence is required checking or oversampling to confirm that convergence has been achieved. In this work, we investigate a straightforward precise sampling procedure based on the triangulation (tessellation) of a polytope. Technical abbreviations are defined the very first time they appear in the text. Ultimately, the notation used in this work is given in table 1.

Table 1. Notation of this work.

Symbol	Meaning	First in
\triangleq	Definition or equality by definition	Eq. (1)
$ \cdot $	Absolute value	Eq. (4)
$\det(\cdot)$	Determinant	Eq. (4)

2 Related Work

With a number of applications and methodologies, the challenge of equally sampling from a polytope is crucial to the success of the process. A good example is the basis for a number of ways of estimating randomised approximations to polytope volumes, such as the one described here. A lengthy history of study on sampling strategies for generating randomised estimates to the dimensions of polytopes and other convex structures can be found in works such

¹ Not to be confused with Polytopes.

as [21],[18],[3],[20],[8]. Aspects of polytope sampling that are particularly advantageous include the development of fast randomised algorithms for multiobjective problems [4] and sampling situational tables [15]. Additionally, randomised strategies for approximated solving mixed - integer linear convex programmes are being studied and developed [13]. Polytope sampling, as indicated in [16], is also associated with hard-disk model simulators in statistical physics, along with estimations of erroneous incidences for linear programming in communication [9].

In order to sample across a uniform distribution encompassing a targeted polytope, one approach follows the assumption to gain useful samples from a homogeneous proposal density which is covering the targeted polytope, for instance, a homogeneous density centred on a square hyperbox, or a Dirichlet distribution, both of which are examples of uniform proposal densities. As demonstrated in [14], the Dirichlet population is uniform across the simplex when the density factor is assigned to 1. This attribute was employed to establish homogeneity throughout the simplex in the multi-class discriminant analysis (MCDA) scenario [19]. In order to avoid a situation where the proposal density is close to the desired density, such techniques must include a rejection phase. Generally, the rate of rejection grows in an exponential way proportionally with the size of the sample space, making this strategy ineffective for large sample spaces [22]. Additionally, weights may be simulated using a variety of MCMC techniques, which are detailed below. Typically, a trade-off arises among the frequency of mixing and the rate of acceptance by the sampler. While dealing with homogeneous joints and dependent distributions, a solitary-state sampler such as Gibbs is the ideal approach [10]. In this circumstance, the rate of rejection is zero by default, and the weights can be repeatedly replicated while adhering to the linear limits and ratio limitations set out. It has been shown that using a systematic strategy of repeated sampling, there are strong connections between drawings and delayed mixing [1],[5]. Improved mixing characteristics may be achieved by modelling the weights together using random walk methods, as opposed to simulating them separately.

Numerous MCMC approaches have widely been investigated for sample processes through polytopes schemas and, more broadly, for convex bodies sampling processes. There are several preliminary observations of algorithms that perform sampling derived from broad convex bodies, including the Ball Walk shown in [21] and the hit-n-run approach proposed in [3],[20]. Despite the fact that these approaches are applicable to polytopes, they do not take use of the particular structure presented by the issue. In contrast, the Dikin walk was introduced in [15], which is tailored for polytopes and so achieves greater convergence rates than generic techniques. With its connection to methodologies for solving linear programmes using interior point approaches, the Dikin walk was the first sampling process found globally. Additionally, as stated in greater detail later in this section, it generates proposal distributions beginning with the typical logarithmic barrier for a polytope. As inducted in [24], it was shown that the Dikin walk may be extended to generic curves with subconscious barriers, which was proven in a further study.

3 Methodology

3.1 Definitions and requisites

Definition 1 (Polytope). A bounded convex n -polytope or polytope is the group of points

$$\mathcal{P} \triangleq \{ \mathbf{p} : \mathbf{A}\mathbf{p} \leq \mathbf{b} \} \quad (1)$$

in \mathbb{R}^n , where \mathbf{A} is a $r \times n$ real matrix of coefficients, \mathbf{b} is a r -vector, and the relation \leq is meant elementwise. A polytope can be determined by its vertices or extreme points \mathcal{V} (\mathcal{V} representation of polytopes).

Definition 2 (Simplex). Let $\mathbf{v}_0, \dots, \mathbf{v}_n$ be points in general position in \mathbb{R}^n . The set

$$\mathcal{S} \triangleq \left\{ \mathbf{p} : \mathbf{p} = a_0 \mathbf{v}_0 + \dots a_n \mathbf{v}_n, a_i \geq 0, \sum_{i=0}^n a_i = 1 \ \forall i = 0, \dots, n \right\} \quad (2)$$

is a n -simplex or simplex in \mathbb{R}^n . \mathcal{S} is an n -dimensional polytope.

More compactly, write $\mathbf{V}_0 = (\mathbf{v}_1 - \mathbf{v}_0, \dots, \mathbf{v}_n - \mathbf{v}_0)$ and $\mathbf{a} = (a_1, \dots, a_n)^T$, with $(^T)$ denoting transpose. Then

$$\begin{aligned} \mathbf{p} &= a_0 \mathbf{v}_0 + a_1 \mathbf{v}_1 + \dots a_n \mathbf{v}_n \\ &= (1 - \sum_{i=1}^n a_i) \mathbf{v}_0 + \sum_{i=1}^n a_i \mathbf{v}_i \\ &= \mathbf{v}_0 + \sum_{i=1}^n a_i (\mathbf{v}_i - \mathbf{v}_0) \\ &= \mathbf{v}_0 + \mathbf{V}_0 \mathbf{a}, \end{aligned}$$

and (2) becomes

$$\mathcal{S} = \{ \mathbf{p} : \mathbf{p} = \mathbf{v}_0 + \mathbf{V}_0 \mathbf{a}, \mathbf{a} \geq \mathbf{0}, \mathbf{a} \mathbf{1}^T \leq 1 \}, \quad (3)$$

where \geq, \leq are meant elementwise.

The volume of \mathcal{S} is

$$\text{Vol}(\mathcal{S}) = \frac{|\det(\mathbf{V}_0)|}{n!}, \quad (4)$$

where $|\cdot|$ means absolute value and $\det(\cdot)$ means determinant. Because $\mathbf{v}_0, \dots, \mathbf{v}_n$ are in general position, $\text{rank}(\mathbf{V}_0) = n$ and $\det(\mathbf{V}_0) \neq 0$.

Lemma 1. *Simplicial decomposition of polytopes*

An n -polytope \mathcal{P} can be decomposed into n -dimensional simplices \mathcal{S}_k , $k = 1, \dots, K$ such that $\mathcal{P} = \mathcal{S}_1 \cup \dots \cup \mathcal{S}_K$ and, for $k \neq l$, $\mathcal{S}_k \cap \mathcal{S}_l = \emptyset$ or $\mathcal{S}_k \cap \mathcal{S}_l = \mathcal{T}$, where \mathcal{T} is a lower-dimensional simplex.

Corollary 1. If $\mathcal{S}_1, \dots, \mathcal{S}_K$ is a simplicial decomposition of \mathcal{P} , then $\text{Vol}(\mathcal{P}) = \sum_{k=1}^K \text{Vol}(\mathcal{S}_k)$.

3.2 Construction of a uniform density over a simplex \mathcal{S}

Theorem 1. Assume $\mathbf{w} = [w_i]$, $i = 1, \dots, n$ to be a vector created at random with $w_i \geq 0$, $w_1 + \dots + w_n \leq 1$, and density $h(\mathbf{w})$. Then the vector $\mathbf{p} = \mathbf{v}_0 + \mathbf{V}_0 \mathbf{w} \in \mathcal{S}$ has density

$$g(\mathbf{p}) = h(\mathbf{w}) |\det(\mathbf{V}_0)|^{-1}, \quad (5)$$

Proof.

This preceding proof utilizes the multivariate principle of Change of Variables:

$$g(\mathbf{p}) = h(\mathbf{w}) \left| \det \left(\frac{d\mathbf{w}}{d\mathbf{p}} \right) \right| = h(\mathbf{w}) |\det(\mathbf{V}_0)|^{-1},$$

where $\left(\frac{d\mathbf{w}}{d\mathbf{p}} \right)_{ij} = \frac{\partial w_i}{\partial p_j} = (\mathbf{V}_0^{-1})_{ij}$. □

To construct a uniform probability density in \mathcal{S} define a random variable \mathbf{w} with uniform density on a regular simplex $\mathcal{W} = \{w_i \geq 0, \sum w_i \leq 1, i = 0, \dots, n\}$. Such is a n -dimensional Dirichlet distribution

$$h(\mathbf{w}) = \text{Dirichlet}(\mathbf{1}) = (n!)^{-1}. \quad (6)$$

From Theorem 1 the choice (6) results in a random vector \mathbf{p} with uniform density $g(\mathbf{p}) = (|\det(\mathbf{V}_0)|n!)^{-1}$ over \mathcal{S} .

3.3 Construction of a uniform density over a polytope \mathcal{P}

We denote $f(\mathbf{p})$ for the following cases as:

$$f(\mathbf{p}) = \begin{cases} g_k(\mathbf{p})P(\mathbf{p} \in \mathcal{S}_k) & , \text{ if } \mathbf{p} \in \mathcal{S}_k \subseteq \mathcal{P} \ \forall k = 1, \dots, K \\ 0 & , \text{ if } \mathbf{p} \notin \mathcal{P} \end{cases} \quad (7)$$

where $P(\mathbf{p} \in \mathcal{S}_k)$ the probability that \mathbf{p} belongs to the k -th simplex of a decomposition of \mathcal{P} as per Lemma 1. Choose $P(\mathbf{p} \in \mathcal{S}_k) = \text{Vol}(\mathcal{S}_k)/\text{Vol}(\mathcal{P})$ for all k . Then for the k -th simplex we obtain:

$$\begin{aligned} g_k(\mathbf{p})P(\mathbf{p} \in \mathcal{S}_k) &= g_k(\mathbf{p}) \left(\frac{\text{Vol}(\mathcal{S}_k)}{\text{Vol}(\mathcal{P})} \right) \\ &= \frac{1}{|\det(\mathbf{V}_{0k})|n!} \left(\frac{|\det(\mathbf{V}_{0k})|/n!}{\sum_{j=1}^K |\det(\mathbf{V}_{0j})|/n!} \right) \\ &= \frac{1}{n! \sum_{j=1}^K |\det(\mathbf{V}_{0j})|} \end{aligned}$$

subsequently (7) becomes

$$f(\mathbf{p}) = \begin{cases} (n! \sum_{j=1}^K |\det(\mathbf{V}_{0j})|)^{-1} & , \text{ if } \mathbf{p} \in \mathcal{P} \\ 0 & , \text{ if } \mathbf{p} \notin \mathcal{P} \end{cases}. \quad (8)$$

As per (8), the construction results in a uniform density distribution over the polytope \mathcal{P} .

3.4 Proposed Algorithm for Sampling

Given the results derived above, we can sample uniformly from a convex polytope \mathcal{P} as defined in algorithm 1.

Algorithm 1 Density Based Sampling On Polytope \mathcal{P} (DBSOP)

Require: Vertices $\mathbf{v}_0, \dots, \mathbf{v}_n$ of a polytope \mathcal{P}

Ensure: Uniform sampling from a convex polytope \mathcal{P}

- 1: Find the vertices $\mathbf{v}_0, \dots, \mathbf{v}_n$ of the polytope.
This can be achieved using the Avis-Fukuda pivoting algorithm as in [2].
 - 2: Decompose \mathcal{P} in n -simplices $\mathcal{S}_1, \dots, \mathcal{S}_K$ using, e.g., Delaunay triangulation (any triangulation satisfying Lemma 1 is appropriate).
 - 3: Return the vertices $\mathcal{V}(\mathcal{S}_k)$ and content $\text{Vol}(\mathcal{S}_k)$ of each simplex \mathcal{S}_k from the triangulation process.
 - 4: Set $\mathbf{q} = (q_1, \dots, q_K)$ with $q_k = \text{Vol}(\mathcal{S}_k)/\text{Vol}(\mathcal{P})$.
 - 5: **for** the i -th of N samples **do**
 - 6: Draw a random vector \mathbf{w}_i form a regular n -simplex:
 $\mathbf{w}_i \sim \text{Dirichlet}(\mathbf{1})$.
 - 7: Decide which simplex is sampled from $j_i \sim \text{Categorical}(\mathbf{q})$.
 - 8: Compute the point \mathbf{p}_i as: $\mathbf{p}_i = \mathbf{v}_{j_i 0} + \mathbf{V}_{j_i 0} \mathbf{w}_i$
 - 9: **end for**
-

3.5 Implementation

The implementation of the proposed scheme is in RStudio where several libraries were used for each step. To find the number of vertices the `findVertices` function is used derived from the `hitandrun` package which in turn uses the `rcdd` package. To perform the tessellation the `delaunayn` function is used obtained from the `geometry` package. The function `simplex.sample` to sample from the degenerate Dirichlet is used acquired from the `hitandrun` package and the function `sample` obtained from the `base` package is used to sample from the Categorical.

3.6 Complexity

Due to the fact that the number of simplices created may scale up to $n!$, triangulation is by far the most dominant term for the complexity. Hence, the algorithm is not feasible in high-dimensional space. The overall complexity is $\mathcal{O}^*(n^3)$.

4 Results

The running times are shown in table 2 and they were obtained using a 5.2 GHz Intel Core i9-10850k CPU and 32 GB of RAM for a fairly simple polytope. We refer to n as the number of n dimensions of polytopes and to k as interactions.

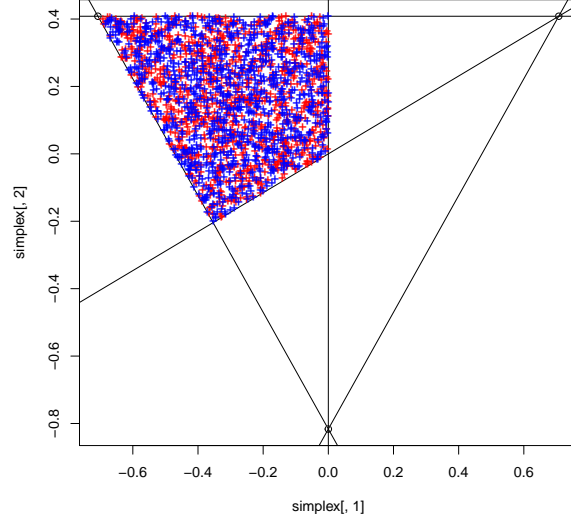


Fig. 1. Polytope sampling using the proposed method

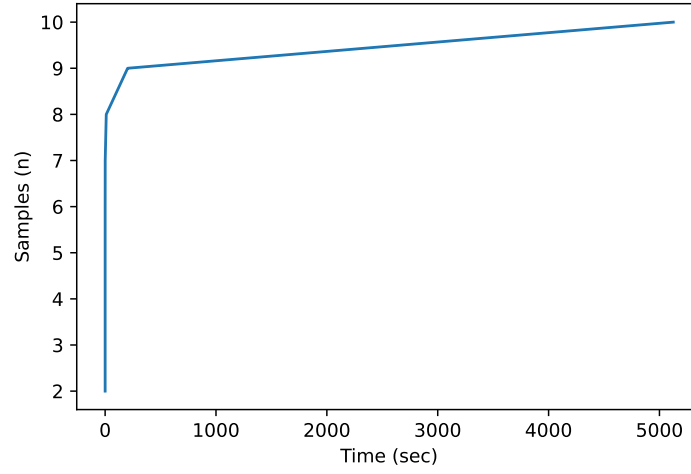


Fig. 2. Samples vs Time

As depicted in figure 2, the algorithm achieves fast sampling up to $n = 8$ and shows a steady performance across $n = 9, \dots, n = 10$.

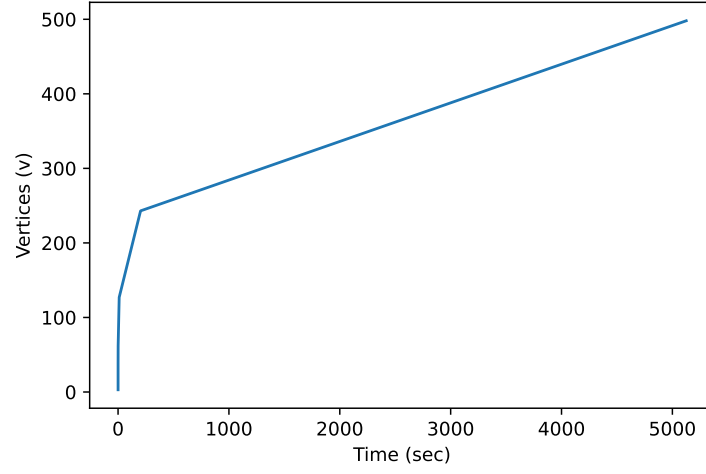


Fig. 3. Vertices vs Time

As depicted in figure 3, the vertices found by the algorithm are ≈ 200 in a relative short time interval while for ≥ 250 the process of finding vertices occurs with a stable performance.

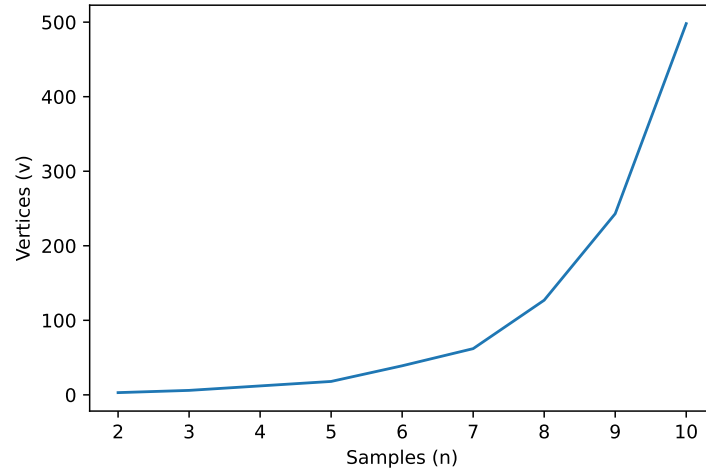


Fig. 4. Vertices vs Samples

As depicted in figure 4 the sampling of n -polytopes vs the vertices found can be expressed in a $f(x) = 2^x - 1$ way.

Table 2. Actual Results.

Actual Results			
n	t (s)	v	K
2	0.142	3	1
3	0.151	6	5
4	0.154	12	44
5	0.159	18	210
6	0.218	39	2.486
7	0.731	62	19.763
8	10.218	127	359.214
9	202.97	243	4.481.667
10	5124.75	498	62.743.338

For $n = 10$, the triangulation started swapping out of RAM (31GB used out of 32 total), and therefore this is the last actual measurement taken. Noteworthy, the running time is almost three minutes for $n = 9$. Moreover, the triangulation only becomes a dominant cost at $n = 7$, and in lower dimensions the running time could be reduced by about 45% through more efficient implementation of step 3 of the algorithm.

Rejection sampling is similarly only feasible up to about $n = 8$ [3] (note that their n is our $n + 1$). However, our algorithm is significantly faster for $n = 7$ and $n = 8$, for example rejection sampling takes over 10 seconds for $n = 8$ and about 200 seconds or 3 minutes and 20 seconds for $n = 9$.

Table 3. Predicted Results.

Predicted Results				
n	t (s)	t (days)	v	K
11	13572	0.15	1.082	125.486.676
12	34638	0.40	3.246	376.460.028
13	271484	3	12.984	1.505.840.112
14	678605	7.8	64.920	7.529.200.564
15	1678609	19.4	389.520	45.175.203.247
16	4763672	55.1	2.726.640	316.226.423.531
17	8163851	94.4	21.813.120	2.529.811.388.174
18	21263149	246.1	196.318.080	22.768.302.493.218
19	72163554	835.2	2.159.498.880	227.683.024.934.355

Because of lack of more RAM we used a machine learning model trained on the results of table 2 to predict the results for $n = 11, 12 \dots n = 19$. Table 3 depicts the results obtained by the machine learning model where the model shows that as with the actual results, the number of vertices as well as the K and the time $t(s)$ grows exponentially. Note that for $n = 19$, the prediction shows that the time required to calculate the polytope will be approximately 835 days or almost 2 years. Hence, it is crucial to readjust the algorithm in step 3 rather than trying to calculate higher dimensions or using more RAM.

5 Evaluation

In this section we evaluate the proposed method to existing techniques such as *bench* and *har* [27]. Figure 5 depicts the average time required to perform sampling for $n = 1, 2 \dots n = 10$. As shown in the figure, the proposed method outperforms the other two existing approaches by $\approx 35\%$. The evaluation metrics

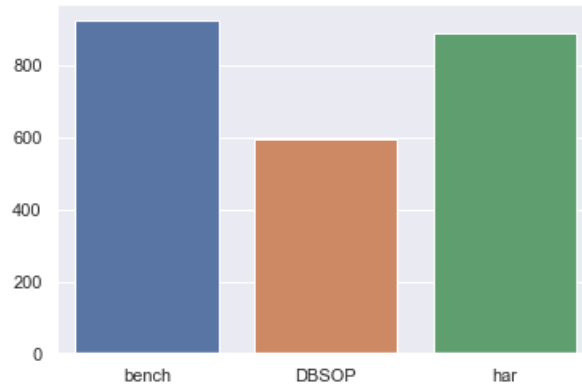


Fig. 5. Comparison of the proposed method in terms of average time

are shown in table 4. The proposed method outperforms the other two existing methods across all three metrics (values shown are average) and the overall performance achieved was higher. We moreover define shape compactness (SC) as the number of samples divided by each sampled dimension.

Table 4. Evaluation of the proposed method.

Evaluation Metric	bench	har	DBSOP
Z-value	2.1	4.9	6.4
SCE	0.04	0.10	0.27
SC	6.4	8.4	11.2

6 Conclusions and Future Work

In the context of this work, a solution to the problem of sampling from a uniform density over a convex polytope is presented, where polytope is a finite volume in n -dimensional space characterized by a combination of linear inequality constraints. This sampling problem has a variety of applications, but we are especially interested in how it might be used to the sampling of weight vectors for multi-class discriminant analysis (MCDA). The outcome of the proposed algorithm resulted in a efficient and fast sampling scheme whereabouts the time required was significantly lower than existing methods in low dimensions. However, for high dimensions we may require further investigation of the rejection rate. Future directions of this work include the readjustment of step 3 of the algorithm to decrease the time required to perform sampling. An efficient variation of this step could decrease the cost significantly resulting in a $\approx 45\%$ reduction. Moreover, another future aspect is to transform the problem of sampling in a CPU-based approach rather than using RAM memory, which will result in a parallel execution of all steps without requiring significant amount of I/Os. Ultimately, a potential path for this work in the future is the integration of reservoir-based sampling techniques as in [17] where the selection of k elements representative of the whole distribution will occur to enhance the overall performance and to further reduce the time as well as the cost required to perform sampling.

References

1. Amit, Y., Grenander, U.: Comparing sweep strategies for stochastic relaxation. *Journal of multivariate analysis* **37**(2), 197–222 (1991)
2. Avis, D., Fukuda, K.: A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra. *Discrete & Computational Geometry* **8**(1), 295–313 (1992). <https://doi.org/10.1007/BF02293050>
3. Bélisle, C.J., Romeijn, H.E., Smith, R.L.: Hit-and-run algorithms for generating multivariate distributions. *Mathematics of Operations Research* **18**(2), 255–266 (1993)
4. Bertsimas, D., Vempala, S.: Solving convex programs by random walks. *Journal of the ACM (JACM)* **51**(4), 540–556 (2004)
5. Besag, J., Green, P., Higdon, D., Mengersen, K.: Bayesian computation and stochastic systems. *Statistical science* pp. 3–41 (1995)
6. Brémaud, P.: Markov chains: Gibbs fields, Monte Carlo simulation, and queues, vol. 31. Springer Science & Business Media (2013)
7. Brooks, S., Gelman, A., Jones, G., Meng, X.L.: Handbook of markov chain monte carlo. CRC press (2011)
8. Cousins, B., Vempala, S.: A cubic algorithm for computing gaussian volume. In: *Proceedings of the twenty-fifth annual ACM-SIAM symposium on discrete algorithms*. pp. 1215–1228. SIAM (2014)
9. Feldman, J., Wainwright, M.J., Karger, D.R.: Using linear programming to decode binary linear codes. *IEEE Transactions on Information Theory* **51**(3), 954–972 (2005)

10. Gelfand, A.E.: Gibbs sampling. *Journal of the American statistical Association* **95**(452), 1300–1304 (2000)
11. Geman, S., Geman, D.: Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-6**(6), 721–741 (1984). <https://doi.org/10.1109/TPAMI.1984.4767596>
12. Hastings, W.K.: Monte carlo sampling methods using markov chains and their applications. *Biometrika* **57**(1), 97–109 (1970)
13. Huang, K.L., Mehrotra, S.: An empirical evaluation of walk-and-round heuristics for mixed integer linear programs. *Computational optimization and applications* **55**(3), 545–570 (2013)
14. Jia, J., Fischer, G.W., Dyer, J.S.: Attribute weighting methods and decision quality in the presence of response error: a simulation study. *Journal of Behavioral Decision Making* **11**(2), 85–105 (1998)
15. Kannan, R., Narayanan, H.: Random walks on polytopes and an affine interior point method for linear programming. *Mathematics of Operations Research* **37**(1), 1–20 (2012)
16. Kapfer, S.C., Krauth, W.: Sampling from a polytope and hard-disk monte carlo. In: *Journal of Physics: Conference Series*. vol. 454, p. 012031. IOP Publishing (2013)
17. Karras, C., Karras, A., Sioutas, S.: Pattern Recognition and Event Detection on IoT Data-streams. *arXiv preprint arXiv:2203.01114* (2022). <https://doi.org/10.48550/arXiv.2203.01114>
18. Lawrence, J.: Polytope volume computation. *Mathematics of computation* **57**(195), 259–271 (1991)
19. Li, T., Zhu, S., Ogiwara, M.: Using discriminant analysis for multi-class classification: an experimental investigation. *Knowledge and information systems* **10**(4), 453–472 (2006)
20. Lovász, L.: Hit-and-run mixes fast. *Mathematical programming* **86**(3), 443–461 (1999)
21. Lovász, L., Simonovits, M.: The mixing rate of markov chains, an isoperimetric inequality, and computing the volume. In: *Proceedings [1990] 31st annual symposium on foundations of computer science*. pp. 346–354. IEEE (1990)
22. Mackay, D.J.C.: Introduction to monte carlo methods. In: *Learning in graphical models*, pp. 175–204. Springer (1998)
23. Mete, H., Zabinsky, Z.: Pattern hit-and-run for sampling efficiently on polytopes. *Oper. Res. Lett.* **40**, 6–11 (01 2012). <https://doi.org/10.1016/j.orl.2011.11.002>
24. Narayanan, H.: Randomized interior point methods for sampling and optimization. *The Annals of Applied Probability* **26**(1), 597–641 (2016)
25. Revuz, D.: *Markov chains*. Elsevier (2008)
26. Ripley, B.D.: *Stochastic simulation*. John Wiley & Sons (2009)
27. Smith, R.L.: Efficient monte carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research* **32**(6), 1296–1308 (1984)