UPPSALA
UNIVERSITET

U.U.D.M. Project Report 2024:16

Examensarbete C, 15 hp
Kandidatprogrammet i matematik

Juni 2024

Efficient sampling of solutions to a system of linear inequalities in the game GOALS

André Ramos Ekengren, Oscar Rotander

Matematiska institutionen, Uppsala universitet

Handledare: Jonas Rundberg

Ämnesgranskare: Jordi-Lluís Figueras

Examinator: Martin Herschend

Abstract

In this work, two algorithms for generating players in the football game GOALS are proposed and compared.

The players are generated to fulfil constraints in a 31-dimensional attribute space, as well as requirements on the distribution of the generated players.

The problem is to find a mathematical description of the problem and a solution technique so that players can be generated from a uniform distribution on the space of all possible solutions. The space of points satisfying the constraints is interpreted as the intersection of a convex polytope and a hyperplane. This is then transformed to a convex polytope in 30-dimensions where the algorithms are applied. It is shown that reversing this transformation preserves the uniform distribution and yields points satisfying all constraints.

We apply two Markov Chain Monte Carlo techniques for sampling in the convex polytope: Hit-and-run and Vaidya Walk. The samples $(X_0, \ldots, X_n)$ generated by these algorithms approach a uniform distribution for large n, but the variables X_i and X_{i+1} are highly correlated. We want low correlation between successive samples and to achieve this we utilize thinning and parallel chains.

The algorithms are compared based on autocorrelation, trace plots to detect non-random patterns and the runtime of the algorithms. It is shown that Hit-and-run has better characteristics for this problem and as such is the preferred algorithm.

Contribution

In this chapter we outline the specific contributions of each author to the thesis. In general, the implementation, testing and gathering of results were done in collaboration.

More specifically, André wrote the following sections:

- The background in section 1.1
- All the prerequisite theory in section 3
- The introduction to section 4, as well as section 4.2 regarding the Vaidya walk
- The criteria in section 5.1
- The description of the results of figure 2-6 in section 5.2
- Paragraphs 1-3 and 6-7 in the discussion
- Appendix B

And the following sections were written by Oscar:

- The abstract of the thesis
- The problem specification in section 1.2
- The mathematical interpretation and transformation of the problem, in section 2
- The hit-and-run algorithm in section 4.1
- The implementation of the algorithms in section 4.3
- Description of the results of figure 7 in section 5.2
- Paragraphs 4-5 in the discussion
- Appendix A

Contents

1 Introduction

1.1 Background

GOALS is a Swedish game studio headquartered in Stockholm, dedicated to developing a football game for both PC and console. The game features multiplayer modes where users compete against each other in teams or one-on-one matches. Users build their football teams with virtual characters, each of which is unique and randomly generated.

Each generated character has 31 attributes ("stats") that define their performance in the game, with values ranging from 10 to 99. These characters also include other data such as overall rating, team position, gender, nationality, ethnicity, height, and weight, which define their visual representation in the game.

The creation of a new random character is initiated by the user from the client, and the characters are generated on a server where they are stored in a database. Due to the large user base and high volume of concurrent users, there is a requirement to generate new characters in less than 20 milliseconds. The stats are represented as an integer array and are distributed to each client before playing a match. From this data, the game client renders the characters visually and sets their limits and abilities as footballers. During a match, each client remains connected to the server, ensuring all clients and the server have the same information about each team.

Balancing the game requires careful management of the weights and constraints related to the 31 stats to ensure matches are competitive, fair, and fun. As GOALS generates millions of characters over time, achieving a uniform distribution across all possible variations is crucial.

This thesis is done in collaboration with GOALS, and they have tasked us with finding efficient algorithms for generating the player stats in their game. We describe the problem further in the coming sections.

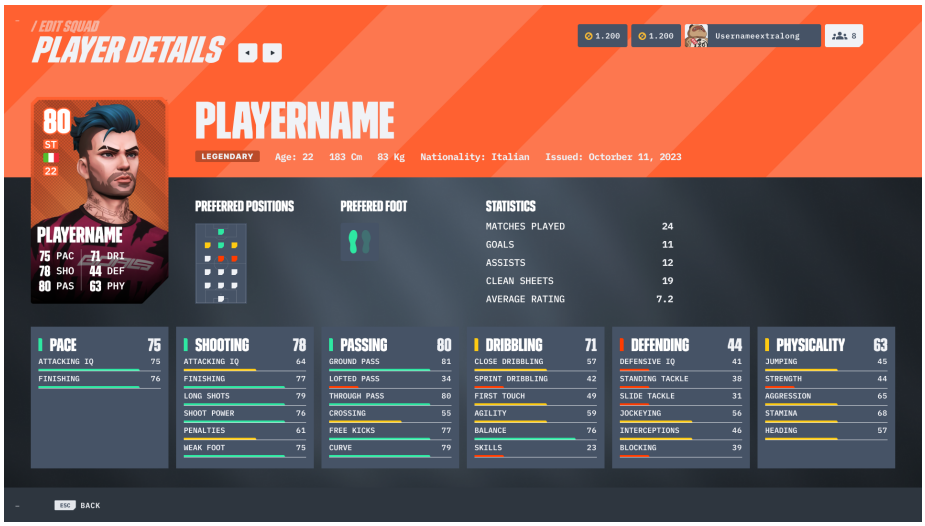

Figure 1: Player card

1.2 Problem specification

The overall rating (80) is exemplified in the top left of the player portrait in figure 1. The overall rating is calculated by a weighted sum of all 31 stats, differently for each position on the field, the difference in weighting is done to simulate that players with different positions are good at different things, for example the rating of a striker may barely be affected by the value of defensive-iq. There are also "mainstats" which are seen in figure 1 as the headers of the 6 boxes containing all stats. These are also weighted but only depending on the stats under that mainstat.

There are constraints on the stats and mainstats, for example a stat under a certain mainstat may be required to be within some distance of the value of that mainstat. Some stats may also be dependent on each other, such as finishing and penalties, meaning they also need to be within some distance of each other. It is also required that the stats lie in the range $[10, 99]$ and the mainstats in the range $[20, 99]$.

When generating a new player a desired position and overall rating is chosen randomly by the game. An additional constraint is that the overall rating when calculated with the weighted sums corresponding to the other positions are less than the desired overall rating. This is done so that every player is playing the position they are most suited for, hence a striker with rating 75 is worse at defending than a defender with rating 75. Given the position and overall rating we want to find a new player as defined by it's stats, uniformly chosen from all valid players.

2 Mathematical interpretation

As seen a player is defined completely by the values of its stats, and so we will consider a player to be a point $x \in \mathbb{R}^n$ where n is the number of stats in the game. To generate a valid player of a given position and overall rating, we need to find a point satisfying all given constraints simultaneously. We interpret each constraint as a linear inequality or equality, $a \cdot x \leq b$ or $a \cdot x = b$ where $a, x \in \mathbb{R}^n$ in the following ways:

The a_i's are different in each case and there may be multiple instances of constraints of the same form but with different a_i's and right-hand side values, these are just the general forms.

- The players rating, given a position, should equal the overall rating.

  $$a_1x_1 + a_2x_2 + ... + a_nx_n = overall\ rating$$

  Here the a_i's are the coefficients in the weighted sum for that position.

- The rating calculated with the weighted sum corresponding to a different position than the one we want should be less than the overall rating.

  $$a_1x_1 + a_2x_2 + ... + a_nx_x \leq overall\ rating$$

  Here the a_i's are the coefficients in the weighted sum for that position.

- A stat should be within some interval of the mainstat to which it belongs. $min \leq stat_i - mainstat \leq max$.

  $$min \leq -a_1x_1 - ... - (a_i - 1)x_i - ... - a_nx_n \leq max$$

  Which we separate in to two inequalities:

  $$-a_1x_1 - ... - (a_i - 1)x_i - ... - a_nx_n \leq max$$
  $$a_1x_1 + ... + (a_i - 1)x_i + ... + a_nx_n \leq -min$$

  Here the a_j's represents the coefficients in the weighted sum for the given mainstat and i is the index of that particular stat.

- Some stats should be within some distance of each other. $-spread \leq stat_i - stat_j \leq spread$.

  $$a_1x_1 + ... + a_ix_i + ... - a_jx_j + ... + a_nx_n \leq spread$$
  $$a_1x_1 + ... - a_ix_i + ... + a_jx_j + ... + a_nx_n \leq spread$$

  Where all coefficients are 0 except $a_i = 1$ and $a_j = 1$. Even though the equation only is $x_i - x_j \leq spread$ and $x_j - x_i \leq spread$ we want all the equations on the same form.

- A mainstat should be less than max and greater than min.
  $min \leq mainstat \leq max$.

$$a_1x_1 + ... + a_nx_n \leq max$$
$$-a_1x_1 - ... - a_nx_n \leq -min$$

  Here the a_i's represent the weighted sum for the given mainstat.

- A stats should be less than max and greater than min.
  $min \leq stat_i \leq max$.

$$a_1x_1 + ... + a_ix_i + ... + a_nx_n \leq max$$
$$-a_1x_1 - ... - a_ix_i - ... - a_nx_n \leq -min$$

  Where each $a_j = 0$ for $j \neq i$ and $a_i = 1$.

In our problem all constraints are of one of these forms, and there is only one constraint of the first form. Now let A be the matrix containing as rows every vector $(a_1, a_2, ..., a_n)$ and b the column vector containing the right-hand side of each inequality. And let C be the $1 \times n$ matrix having as its only row the coefficients in the weighted sum for our desired player position. Then our solution-space can be represented as the intersection between finitely many closed halfspaces and a hyperplane, i.e. a convex polytope and a hyperplane. Hence a valid player is a point $x \in \mathbb{R}^n$ s.t $Ax \leq b$ and $Cx = d$, where d is the desired overall rating.

Having an equality constraint included, however, is not optimal in our case. One reason is that one of the methods we use is specialized to convex polytopes, and is based on interior point methods that require strict inequality. Another reason is that there is an increase in the punishment of numerical errors when attempting to find points constricted to the hyperplane. The next section deals with this problem.

2.1 Isometric transformation to a convex polytope

The goal in this section is to reduce the problem from finding points $x \in \mathbb{R}^n$ satisfying $Ax \leq b$ and $Cx = d$ to finding points $y \in \mathbb{R}^{n-1}$ satisfying $A'y \leq b'$ for some A' and b' such that we are left with sampling points from a convex polytope. We show that we can find such matrices and that the solutions to $A'y \leq b'$ gives us all the solutions to our problem and that the uniform distribution is preserved under the transformation.

Theorem 1. *Let $Ax \leq b$ be a convex polytope and $Cx = d$ a hyperplane in $\mathbb{R}^n$ (A is a $m \times n$ matrix, b is a $m \times 1$ matrix, C is a $1 \times n$ matrix and d is a real scalar). Then there is a $n \times (n-1)$ orthogonal matrix N such that for all $y \in \mathbb{R}^{n-1}$ we have $C(Ny) = 0$.*

Proof. Let c_i be the first coefficient of C s.t $c_i \neq 0$ and define:

$$t := (0, ..., \frac{d}{c_i}, ..., 0)$$

Then we get.

$$C(x-t) = Cx - Ct = d - c_i\frac{d}{c_i} = 0 \qquad A(x-t) = Ax - At = b - At$$

Let $U\Sigma V^*$ be the singular value decomposition of C and since C is real we have $V^* = V^\intercal$. We know that the $n-r$ last columns of V make an orthonormal basis for $\ker C$, and in our case $r = 1$. Let N be the $n \times (n-1)$ matrix having as columns those basis vectors. Then for any $y \in \mathbb{R}^{n-1}$ we get $C(Ny) = 0$ since N consists of basis vectors for $\ker C$. And N is orthogonal. □

Let N be as in theorem 1 and let t be as in the proof of theorem 1. Then given a point $y \in \mathbb{R}^{n-1}$ such that $ANy \leq b - At$, we have $C(Ny) = 0$ and get

$$ANy \leq b - At \implies A(Ny + t) \leq b - At + At = b$$
$$C(Ny) = 0 \implies C(Ny + t) = Ct = d$$

Hence given such a y we can construct the point $(Ny + t) \in \mathbb{R}^n$ that solves our original system, $A(Ny + t) \leq b$ and $C(Ny + t) = d$.

Theorem 2. *Let $Ax \leq b$ be a convex polytope and $Cx = d$ a hyperplane in $\mathbb{R}^n$, and let N be as in theorem 1 and let t be as in the proof of theorem 1 and define $\mathcal{K} := \{x \in \mathbb{R}^n \mid Ax \leq b,\ Cx = d\}$ and $\mathcal{K}' := \{y \in \mathbb{R}^{n-1} \mid ANy \leq b'\}$. Then the affine transformation:*

$$T : \mathcal{K}' \to \mathcal{K} \qquad\qquad y \to Ny + t$$

is an isometry.

Proof. First we prove that T is injective. Let $y_1, y_2 \in \mathcal{K}'$, $y_1 \neq y_2$ and suppose for contradiction that $Ny_1 + t = Ny_2 + t$, then

$$Ny_1 + t = Ny_2 + t \iff Ny_1 = Ny_2 \implies Ny_1 - Ny_2 = 0 \implies$$
$$N(y_1 - y_2) = 0 \implies y_1 = y_2$$

The last implication holds since the columns of N are linearly independent, therefore T is injective.

Now we prove that T is surjective. Let $x \in \mathcal{K}$, then $A(x-t) \leq b - At$ and $(x-t) \in \ker C$, so $x - t$ can be written as a linear combination of the columns of N, $x - t = a_1v_1 + \ldots + a_{n-1}v_{n-1}$ where v_i is the $i : th$ column vector of N. Then the point $y \in \mathbb{R}^{n-1}$, $y = (a_1, \ldots, a_{n-1})$ is such that $x - t = Ny$ and $x = Ny + t$, furthermore since $b - At \geq A(x-t) = ANy$ we have $y \in \mathcal{K}'$, hence T is surjective.

Now we want to show that T preserves distances. Let $y_1, y_2 \in \mathcal{K}'$, $y_1 \neq y_2$ and $y_1 = (a_1, ..., a_{n-1})$, $y_2 = (b_1, ..., b_{n-1})$ then

$$\|(Ny_1 + t) - (Ny_2 + t)\| = \|Ny_1 - Ny_2\| = \|(a_1 - b_1)v_1 + ... + (a_{n-1} - b_{n-1})v_{n-1}\| =$$
$$= \sqrt{(a_1 - b_1)^2 + ... + (a_{n-1} - b_{n-1})^2} = \|y_1 - y_2\|$$

Where the third equality holds because all v_i's are pair-wise orthogonal and $\|v_i\| = 1$.

Hence T is an isometry. □

Theorem 3. *Let K, K' and T be as in theorem 2, then given a random variable X with the uniform distribution on K'. $T(X)$ is a random variable with the uniform distribution on K.*

Proof sketch. Since T is an isometry all distances are preserved and the $n-1$ volume of a subset $Z \subset \mathcal{K}'$ is the same as the $n-1$ volume of $T(Z) \subset \mathcal{K}$ and therefore the ratios $\frac{vol(Z)}{vol(\mathcal{K}')}$ and $\frac{vol(T(Z))}{vol(\mathcal{K})}$ are equal, hence the uniform distribution is preserved under T. □

And since T is bijective we know that we can find all solutions to our problem, points $x \in \mathcal{K}$, by finding points $y \in \mathcal{K}'$.

Now that the problem is described mathematically as having all feasible solutions in a convex polytope, we continue with some prerequisite theory for our solution methods.

3 Prerequisite theory

In this chapter we introduce the theory of *Markov chains* and *Markov Chain Monte Carlo*, the latter being the type of sampling method chosen to solve our problem. We use the work done by Robert and Casella [8] as a foundation for these parts. See appendix B for an introduction to fundamental measure theory, and most importantly, for the definition of transition probability kernels. We begin with the theory of Markov chains, as it is the foundation of our sampling methods.

3.1 Markov Chains

Let $n \in \mathbb{N}$, and say we have a sequence of random variables (X_n) with $X_n \in E$, and transition probabilities dictated by a transition probability kernel K on the measurable space $(E, \mathcal{E})$. Note that in this work, the focus is on state spaces E that are continuous, rather than discrete. As such, $\mathcal{E}$ will usually be assumed to be the Borel σ-algebra, $\mathcal{B}(E)$. Continuing, it will be useful to introduce the notation

$$P(X_{n+1} \in A \,|\, x_n) = K(x_n, A) = \int_A K(x_n, \mathrm{d}x) \tag{1}$$

for the one-step transition probabilities of that sequence. We proceed by defining Markov chains.

Definition 1 (Markov Chain)**.** *Given a transition probability kernel K on a measurable space $(E, \mathcal{E})$, a sequence of random variables (X_n) taking values in E is called a Markov chain if*

$$P(X_{n+1} \in A \,|\, x_0, x_1, \ldots, x_n) = P(X_{n+1} \in A \,|\, x_n) = \int_A K(x_n, \mathrm{d}x). \tag{2}$$

That is, the next transition in the chain is independent of past transitions. We will also focus exclusively on Markov chains that possess another property, called time-homogeneity. These are Markov chains whose transition dynamics are independent of time.

Definition 2 (Time-homogeneous Markov chains)**.** *A Markov chain (X_n) is time-homogeneous if for every $t \in \mathbb{N}_{>0}$, it holds that*

$$P(X_{n+t} \in A \,|\, X_n = x) = P(X_t \in A \,|\, X_0 = x) \tag{3}$$

Furthermore, for a time-homogeneous Markov chain, we would like to calculate not just one-step transitions, but also n-step transitions. If for some $(x, A) \in E \times \mathcal{E}$, we define the one-step transition kernel simply as $K^1(x, A) := K(x, A)$, then it is possible to show the following lemma about the n-step transition kernel K^n.

Lemma 1 (The n-step transition kernel)**.** *for $n > 1$, the n-step transition kernel is*

$$K^n(x, A) = \int_E K^{n-1}(y, A) K(x, \mathrm{d}y). \tag{4}$$

and we say that the n-step transition probability is

$$P(X_n \in A \,|\, X_0 = x_0) = K^n(x, A) \tag{5}$$

when K is a transition probability Kernel.

Additionally, for the purpose of simulation using Markov Chain Monte Carlo techniques, there are some properties that are essential for a Markov chain to have. Given a chain, we would for example like to analyse how the initial starting point affects, if at all, the long-time behaviour of the chain, and whether it explores the whole state space. Will the chain stabilize and converge to some distribution? These types of properties are what the rest of this section will address. We begin with defining for a set $A \in \mathcal{E}$, and a Markov chain (X_n), the *number of passages* of the chain in A.

Definition 3 (Number of passages)**.** *The number of passages of (X_n) in A is defined as the amount of times the chain enters A*

$$\eta_A = \sum_{n=1}^{\infty} \mathbb{I}_A(X_n). \tag{6}$$

We now turn to the notion of irreducibility of a Markov chain, which essentially is a property that states that no matter the initial point of the chain, all other points in the state space have a chance to eventually be reached. For Markov chains with continuous state spaces, we define a ϕ-irreducible Markov chain.

Definition 4 (ϕ-irreducible Markov chain)**.** *A Markov chain (X_n) with transition probability kernel K is called ϕ-irreducible if for a measure ϕ on $(E, \mathcal{E})$, and for every $A \in \mathcal{E}$ such that $\phi(A) > 0$, there exists $n \geq 1$ such that*

$$P(X_n \in A \,|\, X_0 = x) = K^n(x, A) > 0, \, \forall x \in E. \tag{7}$$

Next we would like to define the *period* of a Markov chain, but before we do that we need to know what a *small* set is, and what a cycle of a ϕ-irreducible Markov chain is.

Definition 5 (Small sets)**.** *Given a set $C \in \mathcal{E}$, if there exists $m \in \mathbb{N}_{>0}$ and a measure ν_m on $(E, \mathcal{E})$ such that*

$$K^m(x, A) \geq \nu_m(A), \quad \forall A \in \mathcal{E}, \, \forall x \in C \tag{8}$$

*then C is called a **small** set, and we will sometimes call such a set ν_m-small.*

Definition 6 (Cycle of a Markov chain)**.** *For a ϕ-irreducible Markov chain, let C be a ν_M-small set, and let d be defined as*

$$d = \gcd\{m \geq 1 \,:\, \exists \delta_m > 0, \text{ such that } C \text{ is small for } \nu_m \geq \delta_m \nu_M\}. \tag{9}$$

Then we say that the chain has a cycle of length d.

Subsequently, the *period* of a ϕ-irreducible Markov chain is defined as the largest cycle length. If the period is 1, the chain is called *aperiodic*, and *periodic* otherwise. The periodicity of a chain tells us in some sense whether there are some constraints put on at what time steps it is possible to return to some part of the state-space. If aperiodic, there are no constraints.

Moreover, we have the notions of *recurrence* and *transience*. Although a ϕ-irreducible Markov chain provides the possibility of reaching every set at some point in the chain, it does not provide any information of whether it actually occurs, or how often it does. Therefore, we define recurrent and transient sets, that provides exactly this information.

Definition 7 (Recurrent sets)**.** *A set A is called **recurrent** for a Markov chain if*

$$\mathbb{E}(\eta_A \,|\, X_0 = x) = +\infty, \quad \forall x \in A \tag{10}$$

Definition 8 (Transient sets)**.** *A set A is called **uniformly transient** for a Markov chain if there exists $M \in \mathbb{N}_{>0}$ such that*

$$\mathbb{E}(\eta_A \,|\, X_0 = x) < M, \quad \forall x \in A \tag{11}$$

*and **transient** if there exists a countable collection of uniformly transient sets* $\{B_i\}$ *such that*

$$A = \bigcup_i B_i. \tag{12}$$

This leads us to defining recurrent and transient Markov chains.

Definition 9 (Recurrent Markov chain)**.** *A ϕ-irreducible Markov chain (X_n) is recurrent if for every set $A \in \mathcal{E}$ such that $\phi(A) > 0$, A is recurrent.*

Definition 10 (Transient Markov Chain)**.** *A ϕ-irreducible Markov chain is transient if the state space E is transient.*

Thus, for a Markov chain to be recurrent means that the chain will keep exploring the whole state space in the long run, whereas a transient Markov chain can not make any such guarantees, possibly not returning at all to certain areas of the space. There is, however, an even stronger property than recurrence, named *Harris recurrence.*

Definition 11 (Harris recurrent set)**.** *A set A is **Harris recurrent** if*

$$P(\eta_A = +\infty \,|\, X_0 = x) = 1, \quad \forall x \in A \tag{13}$$

Definition 12 (Harris recurrent Markov chain)**.** *A ϕ-irreducible Markov chain is **Harris recurrent** if every set A such that $\phi(A) > 0$ is Harris recurrent.*

In contrast to normal recurrency, which ensures that on average all sets will be reached infinitely many times, a Harris recurrent Markov chain ensures that this will happen for all realisations of the Markov chain.

Another property of a Markov chain that is important is if the chain has a stationary distribution, or an invariant probability measure. The existence of such a measure is central to Markov Chain Monte Carlo methods.

Definition 13 (Invariant measure)**.** *Let π be a σ-finite measure on $(E, \mathcal{E})$, and let (X_n) be a Markov chain with transition kernel K. If it holds that*

$$\pi(A) = \int_E K(x, A)\pi(dx), \quad \forall A \in E. \tag{14}$$

*Then π is **invariant** for K and the chain (X_n).*

A ϕ-irreducible chain is called *positive* if it has an invariant probability measure, and *null* otherwise [6]. It is possible to show that if the chain is positive, it is also recurrent. If an invariant probability measure π exists for the chain, it is also called its *stationary distribution.*

We now turn to an important condition named the *detailed balance condition.*

Definition 14 (detailed balance condition)**.** *We say that a Markov chain (X_n) with transition probability kernel K satisfies the* ***detailed balance condition*** *if for some function f it holds that*

$$K(y,x)f(y) = K(x,y)f(x), \quad \forall (x,y) \in E \times E \tag{15}$$

We also include the definition of a *reversible* Markov chain, as it is connected to the theorem that will follow.

Definition 15 (Reversible Markov chain)**.** *A Markov chain is* ***reversible*** *if*

$$P(X_{n+2} \in A \,|\, X_{n+1} = x) = P(X_{n+1} \in A \,|\, X_n = x) \tag{16}$$

Now the following theorem gives us a condition that will ensure that some density is invariant to a Markov chain, and showcases the importance of the detailed balance condition.

Theorem 4. *If a Markov chain satisfies the detailed balance condition for a probability density f, then the Markov chain:*

1. *has f as its invariant probability density*
2. *is reversible*

Finally, most of the important properties of Markov chains that are of use to us have now been introduced, and what remains in this section is to connect these properties to convergence theorems. For this purpose, we first define the *total variation distance*.

Definition 16 (total variation distance)**.** *For two probability measures μ_1, μ_2 on $(E, \mathcal{E})$ the* ***total variation distance*** *is*

$$||\mu_1 - \mu_2||_{TV} = \sup_{A \in \mathcal{E}} |\mu_1(A) - \mu_2(A)| \tag{17}$$

Now we answer the question of when the Markov chain converges to its stationary distribution with time.

Theorem 5. *For an aperiodic Harris positive Markov chain (X_n) with probability transition kernel K, and invariant probability measure π, it holds that*

$$\lim_{n \to \infty} \left\| \int K^n(x, \cdot)\mu(\mathrm{d}x) - \pi \right\|_{TV} \to 0 \tag{18}$$

for all initial distributions μ.

We will call such Markov chains *ergodic*. For the partial sums

$$S_N(h) = \sum_{i=1}^{N} h(X_i). \tag{19}$$

we have the following result.

Theorem 6. *Let π be a σ-finite invariant probability measure for (X_n), given that it exists. Then the chain is Harris recurrent if and only if for $f, g \in L^1(\pi)$, with $\int g(x)\pi(dx) \neq 0$, it holds that*

$$\lim_{N\to\infty} \frac{S_N(f)}{S_N(g)} = \frac{\int f(x)\pi(\mathrm{d}x)}{\int g(x)\pi(\mathrm{d}x)}. \tag{20}$$

In particular, if $g(x) = 1$, we have that

$$\lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} f(X_i) = \int f(x)\pi(\mathrm{d}x). \tag{21}$$

We let these last two convergence theorems conclude the theory of Markov chains, and we proceed in the next section with the simulation technique named *Markov Chain Monte Carlo*.

3.2 Markov Chain Monte Carlo

In statistics it is sometimes of interest to simulate random variables from some distribution f. For example, the computation of the integral

$$\mathbb{E}_{X\sim f}[h(X)] = \int h(x)f(x)\,\mathrm{d}x \tag{22}$$

is sometimes approximated by directly simulating from f and taking the mean

$$\frac{1}{n}\sum_{i=1}^{n} h(X_n) \tag{23}$$

as its approximation. As in our problem, it is not always feasible to directly simulate from a distribution. Therefore, other methods have to be explored that can indirectly simulate distributions. One such method is *Markov Chain Monte Carlo (MCMC)*. In general, MCMC based methods work by designing an ergodic Markov chain with f as its stationary distribution, and we have seen in the previous chapter for example that through Theorem 6, the average partial sums of such chains also converge.

We go on to describe the *Metropolis-Hastings algorithm*, a popular MCMC technique.

3.2.1 The Metropolis-Hastings algorithm

In Metropolis-Hastings, we generate a next proposal point from a proposal conditional density q, and construct a chain such that, under some conditions on q, it is ergodic with a target density f as its invariant probability density. We present Metropolis-Hastings in Algorithm 1. Some conditions on f and q is that $\operatorname{supp} f$ is connected, and that

$$\operatorname{supp} f \subset \bigcup_{x\in\operatorname{supp} f} \operatorname{supp} q(\cdot \mid x). \tag{24}$$

Algorithm 1 *Metropolis-Hastings Algorithm*

Input: Current state X_n, target density f, proposal density q
Output: Next state X_{n+1}

1: Generate $Y_n \,|\, X_n \sim q(y \,|\, X_n)$
2: Set next state as

$$X_{n+1} = \begin{cases} Y_n, & \text{with probability } \rho(X_n, Y_n), \\ X_n, & \text{with probability } 1 - \rho(X_n, Y_n) \end{cases} \tag{25}$$

with

$$\rho(x, y) = \min\left\{1, \frac{f(y)}{f(x)} \frac{q(x \,|\, y)}{q(y \,|\, x)}\right\} \tag{26}$$

We would like the conditions of Theorem 5 to be satisfied for the Metropolis-Hastings Markov chain. That is, we want it to be an aperiodic Harris positive Markov chain. It can be shown that the transition kernel K of the chain constructed by Metropolis-Hastings satisfies the **detailed balance condition** for the target density f. Then we can conclude by Theorem 4 that the chain is positive with f as its invariant probability density. Furthermore, there are sufficient conditions on q that guarantee that the chain is also an aperiodic Harris recurrent chain. The first conditions is that the proposal density is positive

$$q(y \,|\, x) > 0, \quad \forall x, y \in \operatorname{supp} f, \tag{27}$$

and the other is that the chain is aperiodic, allowing a one-step transition back to the current state, $X_{n+1} = X_n$. That is

$$P\left(\frac{f(Y_n)}{f(X_n)} \frac{q(X_n \,|\, Y_n)}{q(Y_n \,|\, X_n)} < 1\right) > 0. \tag{28}$$

If these conditions hold, then we can invoke Theorem 5 and 6 and say that the Metropolis-Hastings chain is ergodic with invariant probability density f.

3.2.2 Reducing the dependency

The sample generated using MCMC methods is generally a dependent sample from the target distribution. Depending on the problem, this dependency might need to be reduced. One way to achieve this is to sample from different independent Markov chains. Another option is to perform a *thinning* of the chain by subsampling every k:th value instead. The new chain would then be $(X_{k \cdot m})_{m \in \mathbb{N}}$. Theoretically, subsampling with $k = \infty$ would be optimal, since then the next sample converges to the stationary distribution regardless of the previous sample. In practice this is not possible. Instead, the choice of thinning parameter could for example be chosen as the minimal lag that gives an insignificant autocorrelation of the chain, where we define the autocorrelation

function at lag k of a chain as [4]

$$\rho(k) = \lim_{n\to\infty} \frac{\mathrm{Cov}(X_n, X_{n+k})}{\mathrm{Var}(X_n)}. \tag{29}$$

With a sample of a chain of length N, the autocorrelation at lag k can be estimated as [4]

$$\hat{\rho}(k) = \frac{1}{N-k} \sum_{n=1}^{N-k} (x_n - \bar{x})(x_{n+k} - \bar{x}). \tag{30}$$

Related to this is the *integrated autocorrelation time*, which measures the amount of iterations that is needed to generate an independent sample from target distribution [4]. We define it as the sum

$$\tau = 1 + 2\sum_{k=1}^{\infty} \rho(k). \tag{31}$$

The integrated autocorrelation time can be estimated by

$$\hat{\tau} = 1 + 2\sum_{k=1}^{M} \hat{\rho}(k)$$

for some appropriately chosen $M < N$ [10]. Then the integrated autocorrelation time can be used as a guideline for the choice of thinning.

4 Sampling methods

The relevant theory surrounding Markov Chain Monte Carlo methods have been explained, and we now focus on the specific methods used to solve our problem. Let some general convex polytope be denoted by

$$\mathcal{C} = \{x \in \mathbb{R}^d : Ax \leq b\}.$$

Our problem is to generate vectors $X_i \in \mathcal{C}$, such that they are i.i.d uniformly on the polytope

$$X_1, X_2, \ldots, \overset{\text{i.i.d}}{\sim} \text{Uniform}(\mathcal{C}).$$

Since this distribution is difficult to directly simulate, with the earlier presented theory in mind, we can instead use MCMC methods to construct ergodic Markov chains with the uniform distribution as its stationary distribution. This will produce dependent samples that by Theorem 5 converges to the uniform distribution in the limit.

In the following section we present two MCMC techniques that can be used for sampling on convex polytopes: the *Hit-and-run*, and the *Vaidya walk*.

4.1 Hit-and-run

Hit and run algorithms are a class of algorithms used to simulate arbitrary distributions on $\mathbb{R}^d$ [1], we consider the case where the target distribution is the uniform distribution, referred to in [1] as the "Hypersphere Directions algorithm". Hit and run is a symmetric mixing algorithm [9] which induces a markov chain $(X_0, X_1, ...)$. The induced markov chain is ergodic and has the uniform distribution over the bounded space in which it runs as the stationary distribution. Let $\partial\mathcal{D}$ be the boundary of the d-dimensional unit-sphere and let $\mathcal{C}$ be the convex polytope we are sampling from.

Algorithm 2 Hit and run

Input: $x_0 \in \mathcal{C}$

1: $v \leftarrow uniformly\ chosen\ point\ on\ \partial\mathcal{D}$
2: $t \leftarrow uniformly\ chosen\ point\ from\ \{t \in \mathbb{R} \mid vt + x_n \in \mathcal{C}\}$
3: $x_{n+1} \leftarrow x_n + vt$

4.1.1 Choosing a random point on $\partial\mathcal{D}$

To choose a random direction in $\mathbb{R}^d$ we create a vector $w = (X_1, ..., X_d)$ consisting of d independent normally distributed variables, $w \sim \mathcal{N}_d(0, \mathbb{I}_d)$. Then the vector $v = \frac{w}{\|w\|}$ is a vector uniformly distributed over the $d - dimensional$ unit sphere.

4.1.2 Finding the distance to the boundary of the convex polytope

Since our current point x is inside the convex polytope $\mathcal{C}$ we know that we are on the "correct" side of every hyperplane defining it. Hence to find the furthest travel along the vector v in both directions we need to find the closest hyperplane along the line $L = vt+x$ in both directions. The intersection between a hyperplane $H = a \cdot x$ and a line $L = vt + x$ can be found in the following way:

$$a \cdot (vt + x) = a \cdot vt + a \cdot x = ta \cdot v + a \cdot x = H \implies t = \frac{H - a \cdot x}{a \cdot v}$$

Hence since $||v|| = 1$, $\frac{H-a\cdot x}{a\cdot v}$ is the distance along v to the hyperplane from x. Since each row a_i in A together with b_i in b defines a hyperplane we can find the distance by:

$$b = (H_1, ..., H_m) \implies b - Ax = (H_1 - a_1 \cdot x, ..., H_m - a_m \cdot x)$$
$$Av = (a_1 \cdot v, ..., a_m \cdot v)$$

Element-wise division gives us:

$$\vec{t} := (\frac{H_1 - a_1 \cdot x}{a_1 \cdot v}, ..., \frac{H_m - a_m \cdot x}{a_m \cdot v})$$

4.1.3 Choosing the next point

The smallest positive value t_+ and largest negative value t_- in $\vec{t}$ are the distances we can travel along v before hitting a hyperplane in the positive and negative direction respectively. The next point is chosen to be $vt+x$ where $t \sim \mathcal{U}(t_-, t_+)$. The algorithm is summarised above as Algorithm 2.

4.2 Vaidya Walk

The Vaidya Walk is a Metropolis-Hastings algorithm (as previously described in Algorithm 1) developed for sampling points uniformly from a convex polytope [3]. The method is inspired by interior point methods in optimization, utilizing the combination of a volumetric and logarithmic barrier function to generate new proposal points in the polytope. The logarithmic barrier function is defined as

$$\mathcal{F}(x) := -\sum_{i=1}^{n} \log\left(b_i - a_i^T x\right) \tag{32}$$

with the hessian

$$\nabla^2 \mathcal{F}(x) = \sum_{i=1}^{n} \frac{a_i a_i^T}{s_{x,i}^2} \tag{33}$$

and $s_{x,i} := b_i - a_i^T x$, being the slackness at x for the i:th constraint. The volumetric barrier function is defined as

$$v(x) = \log\det \nabla^2 \mathcal{F}(x). \tag{34}$$

The creators of the walk base their work on another barrier function, that we will call the Vaidya barrier function $\mathcal{V}(x)$, that combines the logarithmic and volumetric barrier:

$$\mathcal{V}(x) := v(x) + \beta_V \mathcal{F}(x) \tag{35}$$

with $\beta_V := d/n$. The matrix that is central to sampling new points in the Vaidya Walk is the positive definite matrix [2]

$$V(x) := \sum_{i=1}^{n} (\sigma_{x,i} + \beta_V) \frac{a_i a_i^T}{s_{x,i}^2} \tag{36}$$

where $\sigma_{x,i} = \frac{a_i^T (\nabla^2 \mathcal{F}(x))^{-1} a_i}{s_{x,i}^2}$, are the so called leverage scores. This matrix is related to the hessian of the Vaidya barrier, and it holds that

$$5v^T \nabla^2 \mathcal{V}(x)\, v \geq v^T V(x)\, v \geq v^T \nabla^2 \mathcal{V}(x)\, v, \quad \forall v \in \mathbb{R}^d. \tag{37}$$

In the Vaidya Walk, given a point in the interior of our polytope, $x \in \text{int}\,(\mathcal{C})$, and some radius r, a new point Y is proposed from the multivariate normal distribution

$$Y \,|\, X = x \sim \mathcal{N}(x, \frac{r^2}{\sqrt{nd}} V^{-1}(x)). \tag{38}$$

Thus, a scaled inverse of the matrix $V(x)$ is used as the covariance matrix in our proposal distribution, and the proposal conditional density is

$$p(y \,|\, x) = \sqrt{\det V(x)} \left(\frac{nd}{2\pi r^2}\right)^{d/2} \exp\left(-\frac{\sqrt{nd}}{2r^2}(y-x)^T V(x)(y-x)\right). \tag{39}$$

The target density is uniform on the polytope

$$f(x) = \frac{1}{\text{Vol}\,(\mathcal{C})}, \quad \forall x \in \mathcal{C} \tag{40}$$

and thus the target densities are canceled out in the acceptance probability

$$\rho(x, y) = \begin{cases} \min\left\{1, \frac{p(x \,|\, y)}{p(y \,|\, x)}\right\}, & y \in \mathcal{C} \\ 0, & y \notin \mathcal{C}. \end{cases} \tag{41}$$

The algorithm we use is presented in Algorithm 3. It is an adaptation of the original Vaidya walk that does not explicitly set $X_{n+1} = X_n$ in case of rejection. The reason for this is because we want the players generated to the user to be unique. This might, however, weaken the theoretical guarantees, as the Markov chain is not explicitly aperiodic anymore. Without the adaptation however, this algorithm would produce an ergodic Markov chain, since the proposal conditional density p satisfies both condition 27 and 28.

Algorithm 3 Vaidya Walk, edited non-repeating, non-lazy version [2]

Input: Radius $r > 0$, $x_0 \in \text{int}\,(\mathcal{C})$
Output: Sequence $x_1, x_2, \ldots$ uniformly sampled from $\mathcal{C}$

1: **for** $i = 0, 1, \ldots$ **do**
2: $\quad \xi_{i+1} \sim \mathcal{N}(0, \mathbb{I}_d)$
3: $\quad y_{i+1} \leftarrow x_i + \frac{r}{(nd)^{1/4}} V(x_i)^{-1/2} \xi_{i+1}$
4: $\quad$ **if** $y_{i+1} \notin \mathcal{C}$ **then**
5: $\quad\quad$ go to line 2
6: $\quad$ **else**
7: $\quad\quad \alpha_{i+1} \leftarrow \min\left\{1, \frac{p(x_i \mid y_i)}{p(y_{i+1} \mid x_i)}\right\}$
8: $\quad\quad U_{i+1} \sim U(0, 1)$
9: $\quad\quad$ **if** $U_{i+1} \geq \alpha_{i+1}$ **then**
10: $\quad\quad\quad$ go to line 2
11: $\quad\quad$ **else**
12: $\quad\quad\quad x_{i+1} \leftarrow y_{i+1}$
13: $\quad\quad$ **end if**
14: $\quad$ **end if**
15: **end for**

4.3 Implementation

The sample $(X_0, X_1, ..., X_n)$ generated by Hit-and-run and Vaidya Walk approach a uniform distribution as n increases. But the variables X_i and X_{i+1} are highly correlated and often close together in space. We want neither of these properties, and to mitigate this we utilize thinning, hence we instead get the sample $(X_0, X_m, ..., X_{nm})$ where m is the amount of thinning we do. For large m the previous value in the chain is "forgotten" and the value of X_{i+1} is as if chosen from the stationary distribution of the chain which in this case is the uniform distribution. The correlation between successive samples can also be reduced by running multiple chains in parallel and randomly selecting one from which to draw the next sample.

Given a position and an overall rating we construct the convex polytopes $\mathcal{K}$ and $\mathcal{K}'$ together with the matrix N and vector $\vec{t}$ as described in theorems 1 and 2. Let $p \in \mathbb{N}$ be the number of chains to run in parallel and let $m \in \mathbb{N}$ be the amount of thinning to use. Since the convex polytopes are specific for each unique pair of position and rating we need to keep track of the current state of all p chains for each such pair and continue from there the next time we generate a player of that type. An initial point to start the algorithm from can be found with any number of linear programming methods, such as the method described by Chungmok Lee and Sungsoo Park[5] for finding the Chebyshev center.

We create a new player according to Algorithm 4 where on line 5 we use either Hit and run(Algorithm 2) or Vaidya walk(Algorithm 3).

Algorithm 4 Generate random player of given position and rating

Input: $\vec{x_0} := (x_{0_0}, x_{1_0}, ..., x_{p_0}) \in \mathcal{K}'^p$, $p, m \in \mathbb{N}$, N *and* $\vec{t}$

$i \leftarrow 0$
$n \leftarrow 0$
while $i < m$ **do**
    $j \leftarrow 0$
    while $j < p$ **do**
        $\vec{x_{n+1}}[j] \leftarrow$Hit and run($\vec{x_n}[j]$) or Vaidya walk($\vec{x_n}[j]$)
        $j \leftarrow j + 1$
    end while
    $i \leftarrow i + 1$
    $n \leftarrow n + 1$
end while
$r \leftarrow$ Uniformly chosen integer from 0 to and excluding p
return $N(x_m[r]) + t$

Let $(N(X_0)+t,\ N(X_m)+t, \ldots, N(X_{nm})+t)$ be the sample generated by running the above algorithm n times, the sample $(X_0,\ X_m, \ldots, X_{nm})$ approaches the uniform distribution for large n and for large values of m and p the correlation between the $X_i's$ approaches zero. Then the $X_i's$ approach being "independent" and uniformly distributed random variables, and from theorem 3 we know that the $(N(X_i) + t)'s$ have the same properties and they are valid players of the given position and rating.

5 Results and analysis

5.1 Criteria

There are mainly three criteria and diagnostics that are of interest to us, and that will be presented in the coming results. One is the autocorrelation of the chain at different time lags, which signifies the correlation between generated players. We want this value to be low so that the players are as close to random and independent as possible. Another is the trace plot of the different stats, that gives a good overview of the general movement of the chain. If the trace plot looks as if the moves are random with wide jumps, it signifies good behaviour and low autocorrelation. Lastly, we are interested in the runtime of the algorithms. The players need to be generated fast, preferably under 20 ms per player, to satisfy the users of the game. To find suitable parameters for the methods, one must weigh the randomness of the player generation and its runtime against each other, and find a good balance.

5.2 Effects of parameters

We present here how the thinning parameter and number of parallel chains affect the different criteria for both algorithms, and the results are based on generated centre backs with overall rating of 70 with a focus on the *defending* mainstat in the plots that will follow. The effect of thinning is shown in figure 2 and 3 for the Hit-and-run and the Vaidya walk respectively.

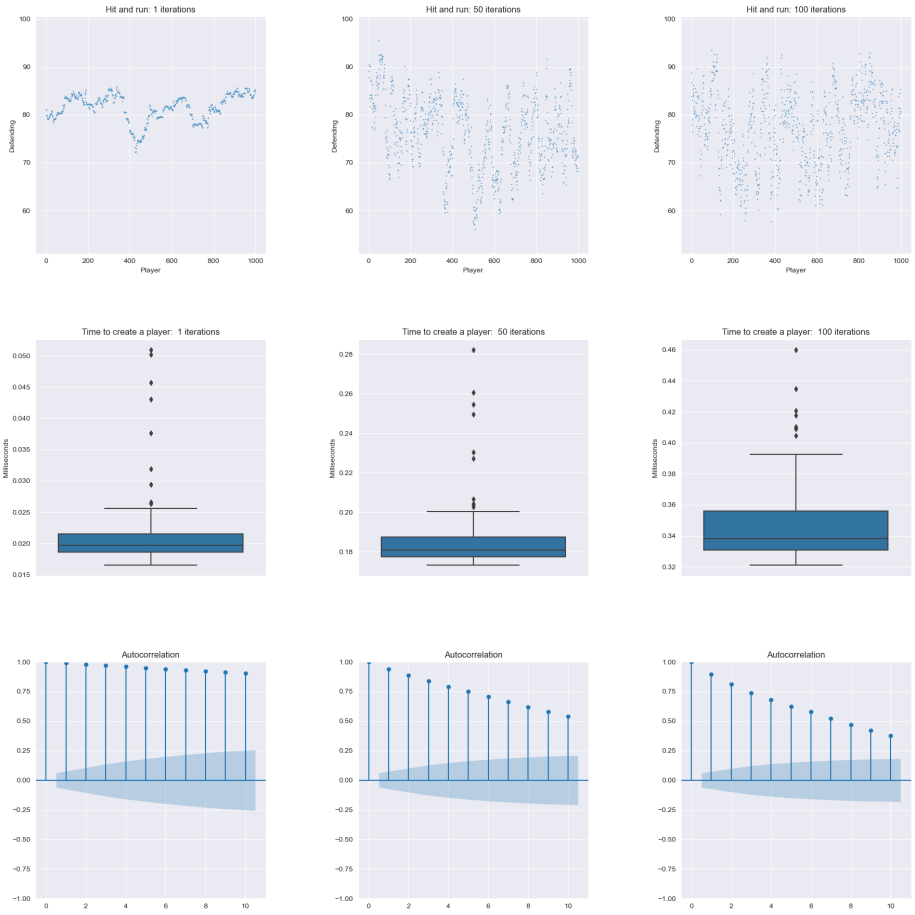

Figure 2: Hit and run effects of thinning

For the Hit-and-run, the trace plots show more variation and less autocorrelation with increasing size of thinning parameter. Furthermore, even though the runtime increases with larger thinning, it is in these examples not close to reaching the performance threshold of 20 ms. And as expected, the autocorrelation plots show a decrease with increased thinning, but subsampling every 100:th player does not result in insignificant autocorrelation.

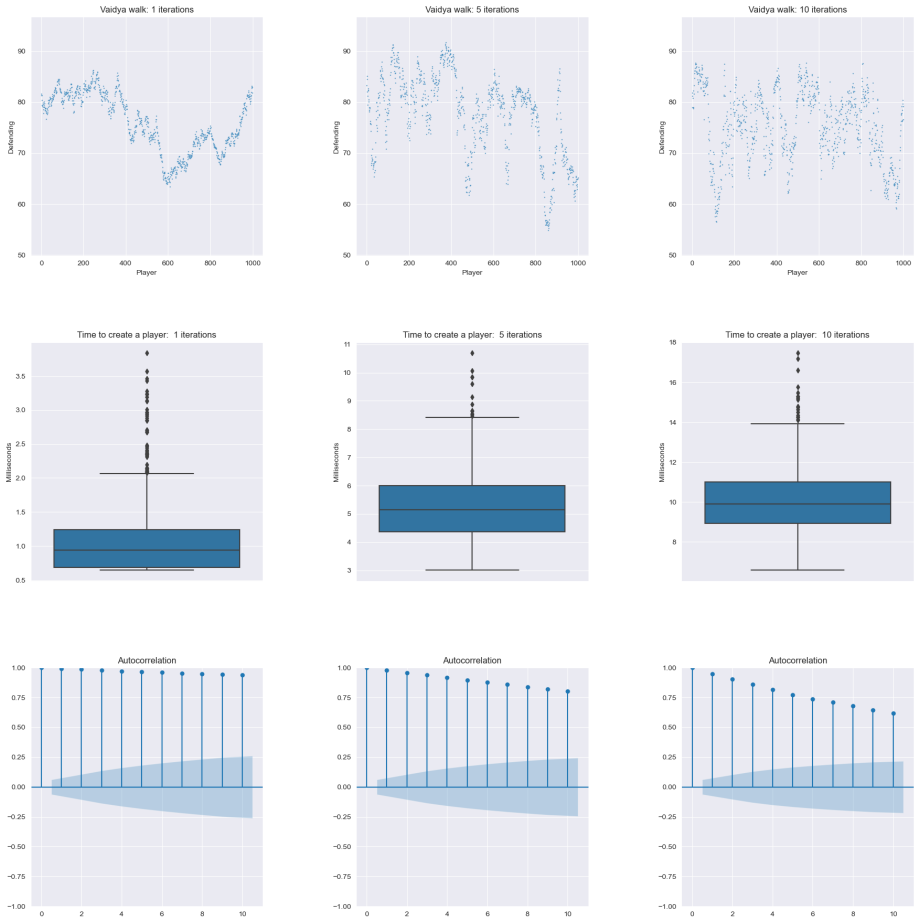

Figure 3: Vaidya walk effects of thinning

Similar results hold for the Vaidya walk. The trace plots shows increased variation and decreased autocorrelation. Note that the thinning used is much lower for the Vaidya walk. By inspecting the boxplots of sampling times, the time is already high with no thinning, and almost reaches the threshold for performance when subsampling every 10:th player, with the variation in sampling time also being high. The autocorrelation is still significant in all cases.

The effect of parallel chains is shown in figures 4 and 5.

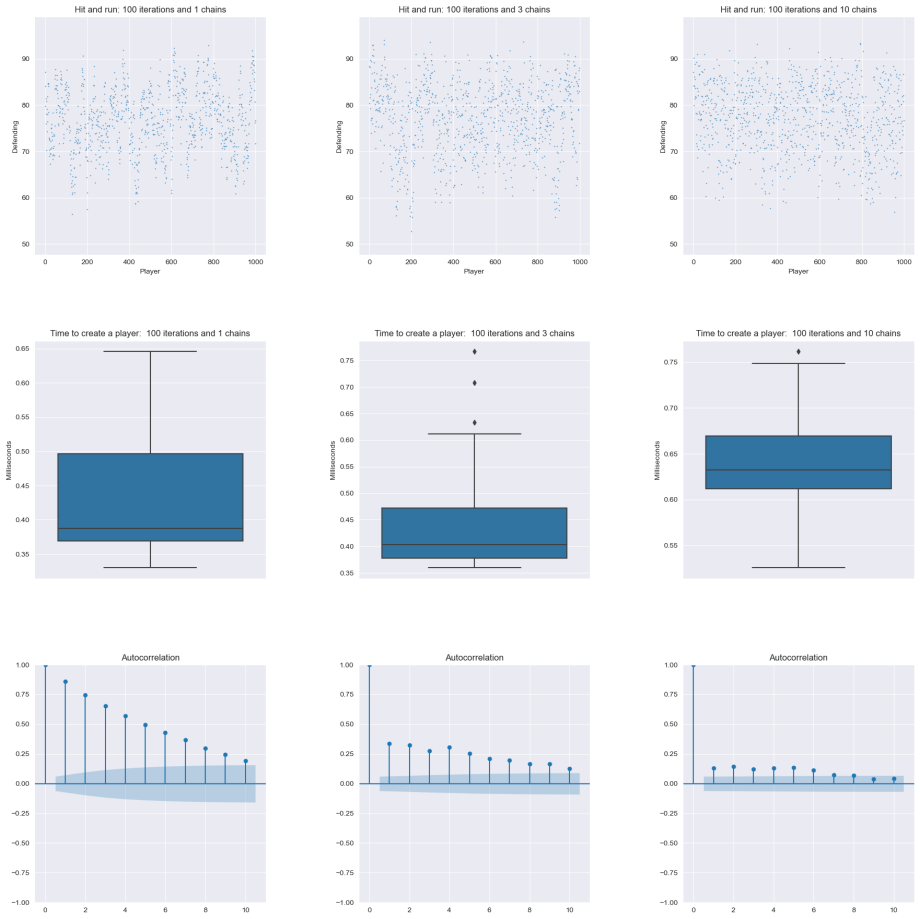

Figure 4: Hit and run effects of multiple chains

It is apparent at once that increasing the amount of chains for the Hit-and-run improves the trace plot and decreases the autocorrelations significantly, while not having a substantial negative impact on the sampling time in relation to the performance threshold.

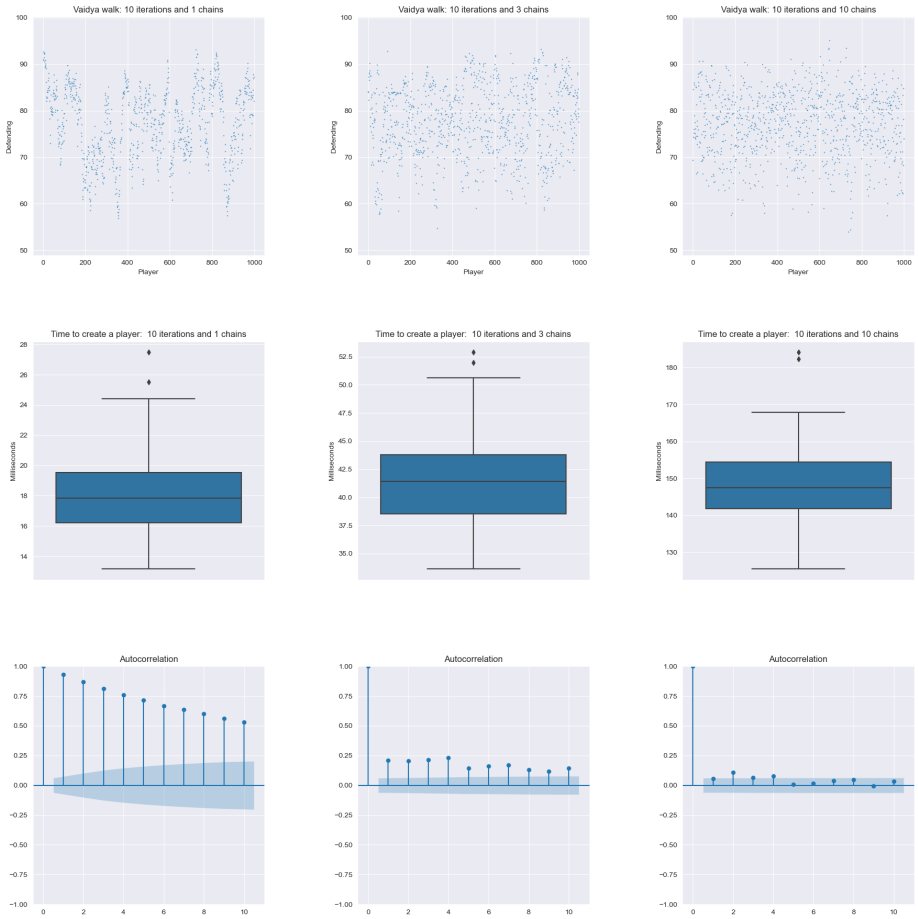

Figure 5: Vaidya walk effects of multiple chains

Once again, similar results are seen for the Vaidya walk. The trace plots are improved, and autocorrelation decreased to insignificant levels with 10 chains. With the combination of 10 iterations and several chains, however, the sampling time is high relative to the performance threshold. The integrated autocorrelation time has been mentioned before as a guideline for the choice of thinning, and we present results related to that in figure 6.

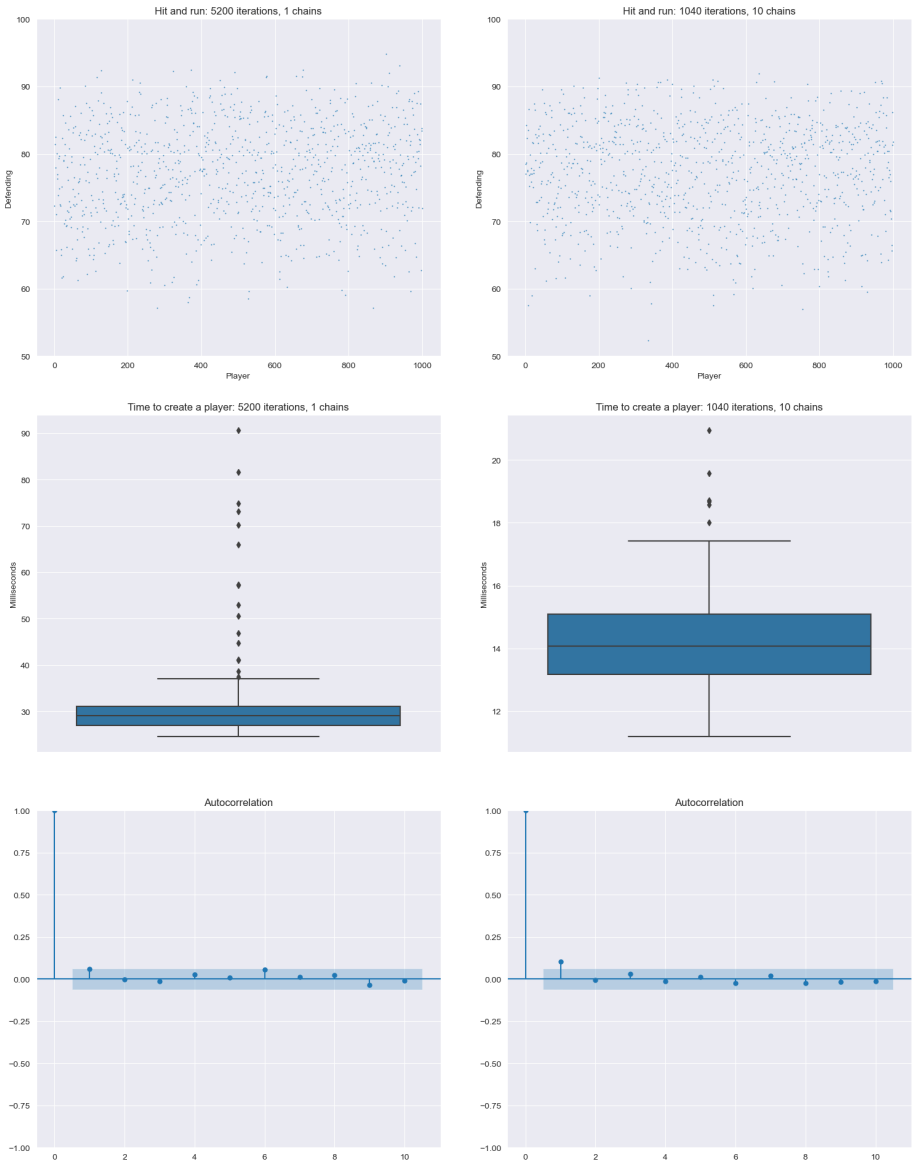

Figure 6: Hit and run using integrated autocorrelation

This plot shows hit-and-run using a thinning equal to the estimated integrated autocorrelation time (IAT) with one chain and thinning equal to one fifth of the IAT with ten chains. The trace plots appear similar, with low autocorrelation and no discernible patterns. With thinning equal to the IAT, the autocorrelations are insignficiant for all time lags, and almost insignificant for all time

lags in the other case. The average sampling times using the IAT has increased slightly above the performance threshold, with some outliers that are significantly higher. In the other case with a fifth of the thinning and 10 chains, the sampling times are not above the performance threshold, except for some outliers.

In figure 7 the effects of thinning across all stats for successive players are shown. For a player there is a marker in the row corresponding to a stat if any of the previous players(in the window=1) same stat was within the threshold(± 2 in this case).

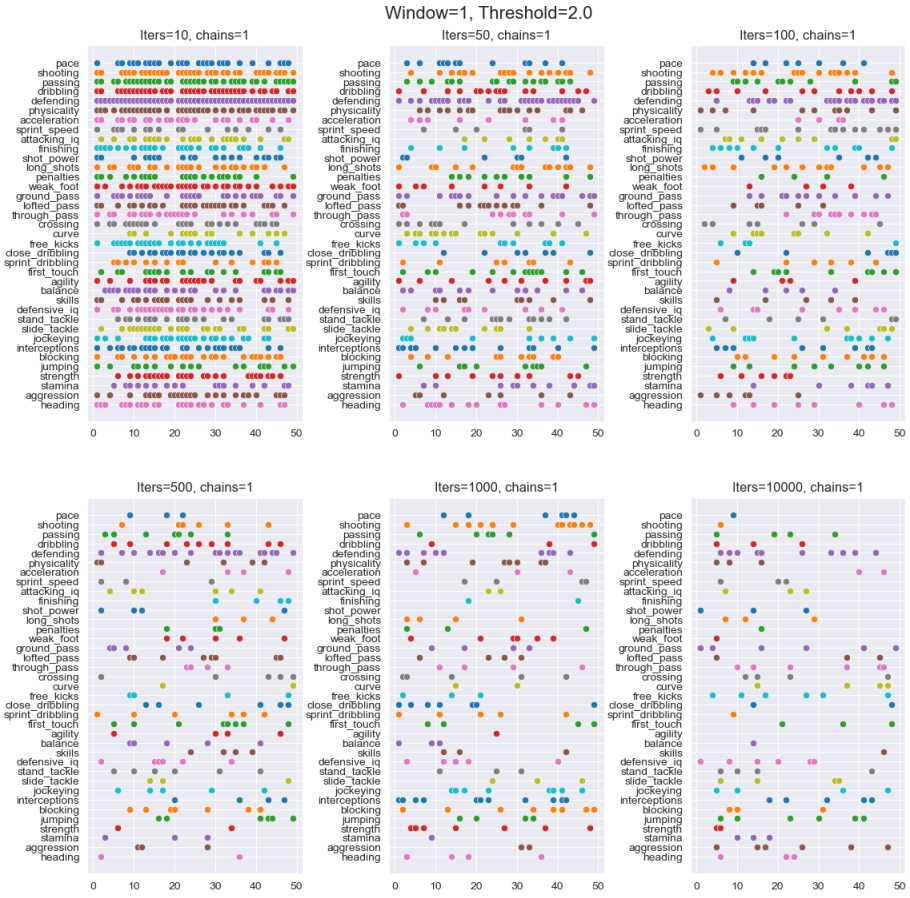

Figure 7: Effects of thinning on similarity of stats for successive players

In figure 7 we see that increasing thinning reduces the likelihood of successive players having similar stats. But even if the players were sampled from the uniform distribution we would expect to see some successive players have stats

be within the threshold.

6 Discussion

The general take-away from the results is that the Hit-and-run algorithm has a fast sampling time relative to the performance threshold, with acceptable sampling performance, while the Vaidya walk is slow for similar sampling performance. The most significant negative impact on sampling time is the thinning parameter. In the case of Hit-and-Run, it is originally so fast that the thinning parameter can be increased significantly without exceeding the performance threshold, and it is possible to increase to higher than 100 iterations to further reduce the autocorrelation. The same can not be said about the Vaidya walk, already approaching the performance threshold of the sampling time at 10 iterations. Since the results are only for a 1000 generated players, it is fully possible for the outliers in the sampling time to be drastically higher. Thus, even though the autocorrelation is not sufficiently low, thinning can not be increased much further for the Vaidya walk.

The use of parallel Markov chains has a tremendous impact on the autocorrelation for both algorithms. With 10 chains, the autocorrelation is reduced to almost insignificant levels for the Hit-and-run, and insignificant for the Vaidya walk. If the iterations are increased for the Hit-and-run, the autocorrelation would eventually reach insignificant levels as well. Parallel chains also have the effect of increasing the sampling time. Most likely this is due to the sampling finishing only when the slowest chain is finished, as well as the overhead of launching and joining threads. The increase is not significant for the Hit-and-run, but it is for the Vaidya walk, where both using 3 and 10 chains (with 10 iterations) increases the average sampling time to levels above the performance threshold. Therefore, one must choose more carefully the right amounts of thinning and parallel chains as to not make the Vaidya walk completely impractical.

There is always a balance between having samples with low correlation and having a low sampling time. As such, the integrated autocorrelation time can serve as a reference for the minimal thinning needed for "optimality", in terms of dependency between the samples. It might not be suitable with respect to the sampling time, however, as is visible in figure 6. Since the use of multiple chains has also been shown to decrease dependency while not increasing the sampling time substantially, adding more chains and lowering the thinning from the IAT makes it possible to decrease the sampling time at the same time as preserving low dependency. This is also observed in the same figure.

One of the important criteria is the uniqueness of the players that are generated. Although the algorithms show good behaviour on larger scales they still have issues on small scales, they may for example get stuck for a few iterations and yield very similar players. Increasing thinning mitigates this problem as can be seen in figure 7, but the required amount of thinning to guarantee good

behaviour on all scales is too large to be usable in real-time without further optimization. Therefore we recommend setting the thinning parameter differently depending on the immediate situation in the game. For example when a user opens a "player pack" and it is determined the user will receive more than one player of the same position and rating the thinning parameter can be set to be much larger than in regular use of the algorithm. Implementing it this way keeps the algorithm showing good behaviour on a large scale with low run-time cost while allowing it to also exhibit the desired behaviour on a small scale whenever observed by a user.

In this paper we have targeted a uniform distribution believing that it will yield the most "random" feeling players to the users. But what feels random to people is not necessarily what is mathematically random, for example generating a player while maximizing the distance to the previous player may generate a more "random feeling" player, that would however not work in the long term since the algorithm would be predictable. Furthermore, changing the algorithm may change or cause it to altogether lose some mathematical properties, which makes further development of the algorithm and interpretation of the results harder to reason about, yet it may be a worthwhile trade-off.

There is also something to be said about the complexity of the algorithms. In practice, the Hit-and-run is both simpler to understand and to implement compared to the Vaidya walk, making it especially more suitable for a company that might prefer simplicity. Furthermore, a constant radius of 1 was used in the results gathered for the Vaidya walk. In reality, the radius should be chosen differently depending on the position and overall rating, and hence polytope, that is being sampled, such that a good balance between exploration and exploitation is reached. This increases the complexity of the algorithm even further.

When taking both the results and the complexity of the algorithms into consideration, the Hit-and-run stands out as the preferred method for this particular problem.

Appendix A Linear algebra

In this appendix we shortly define the linear algebra concepts that are the foundation of the mathematical interpretation of our problem. First is the definition of a hyperplane, which is equivalent to linear equality constraints.

Definition 17 (Hyperplane in $\mathbb{R}^n$)**.** *Let $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$ then the affine subspace defined by $H := \{x \in \mathbb{R}^n \mid a \cdot x = b\}$ is a* ***Hyperplane****.*

Secondly is the idea of a closed halfspace, related to linear inequality constraints.

Definition 18 (Closed Halfspace in $\mathbb{R}^n$)**.** *Let $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$ then the set defined by $H := \{x \in \mathbb{R}^n \mid a \cdot x \leq b\}$ is a* ***Closed Halfspace****. It is the region of space one side of a hyperplane.*

And lastly, we define convex polytopes.

Definition 19 (Convex polytope in $\mathbb{R}^n$)**.** *A convex polytope is the intersection of finitely many closed halfspaces that is bounded. It is the solutions to the system of linear inequalities:*

$$\begin{aligned} a_{11}x + a_{12}x + ... + a_{1n}x &\leq b_1 \\ a_{21}x + a_{22}x + ... + a_{2n}x &\leq b_2 \\ &\vdots \\ a_{m1}x + a_{m2}x + ... + a_{mn}x &\leq b_m \end{aligned}$$

Where m is the number of halfspaces. This can be expressed as $Ax \leq b$ where the inequality is element-wise.

Appendix B Probability and Measure theory

The main purpose of this appendix is to introduce basic probability and measure theory concepts that are needed before dealing with Markov chains with continuous state space. The following theory is heavily based on the exposition done by Çinlar [11], and we refer the reader there for a more thorough introduction. We start by defining *algebras* and *σ-algebras*.

Definition 20 (Algebra)**.** *Let E be a set, and $\mathcal{E}$ be a non-empty collection of subsets of E. $\mathcal{E}$ is called an algebra on E if it is closed under finite unions and complements.*

Definition 21 (σ-Algebra)**.** *$\mathcal{E}$ is a σ-algebra on E if it is closed under countable unions and complements.*

Furthermore, we have a special type of σ-algebra for topological spaces such as $\mathbb{R}^d$ that we deal with, called *Borel* σ-algebra.

Definition 22 (Borel σ-Algebra)**.** *If E is a topological space and $\mathcal{E}$ is the smallest collection of all open subsets of E, then $\mathcal{E}$ is called the Borel σ-algebra on E, often denoted as $\mathcal{B}(E)$.*

With these definitions in place we can then go on to define the notions of measurable spaces, measures on measurable spaces, and finally, the resulting measure space.

Definition 23 (Measurable Space)**.** *If E is a set and $\mathcal{E}$ a σ-algebra on E, then the pair $(E, \mathcal{E})$ is called a measurable space, and the elements of $\mathcal{E}$ are called measurable sets.*

Connected to measurable spaces are so called *measurable functions*.

Definition 24 (Measurable function)**.** *If $(E, \mathcal{E})$ and $(F, \mathcal{F})$ are measurable spaces, and f is a mapping $f : E \mapsto F$ such that $f^{-1}B \in \mathcal{E}$ for every $B \in \mathcal{F}$, then f is called measurable relative to $\mathcal{E}$ and $\mathcal{F}$, or alternatively, $\mathcal{E}$-measurable.*

In addition, the notion of a *measure* on measurable spaces is a crucial concept.

Definition 25 (Measure)**.** *A measure on a measurable space $(E, \mathcal{E})$ is a mapping $\mu : \mathcal{E} \mapsto \bar{R}_+ = [0, +\infty]$ such that*

- $\mu(\{\emptyset\}) = 0$
- $\mu(\bigcup_n A_n) = \sum_n \mu(A_n)$ *for every disjointed sequence* $(A_n) \in \mathcal{E}$.

If there exists a measurable partition (E_n) of E such that $\mu(E_n) < \infty$ for all n, then the measure μ is called *σ-finite*.

Definition 26 (Measure Space)**.** *The triplet $(E, \mathcal{E}, \mu)$ is called a measure space.*

We continue by defining probability measures and probability spaces, which are simply special cases of measures spaces.

Definition 27 (Probability Measure)**.** *A measure μ on a measurable space $(E, \mathcal{E})$ is called a probability measure if $\mu(E) = 1$.*

Definition 28 (Probability Space)**.** *The measure space $(E, \mathcal{E}, \mu)$ is a probability space if μ is a probability measure.*

Finally, an idea that is going to be of importance to know about are transition kernels, which are going to be used to represent probabilities of transitioning into certain areas in the space.

Definition 29 (Transition Kernel)**.** *If $(E, \mathcal{E})$ and $(F, \mathcal{F})$ are measurable spaces, and K is a mapping $K : E \times \mathcal{F} \mapsto \bar{\mathbb{R}}_+$ such that*

- $\forall B \in \mathcal{F}$, $K(\cdot, B)$ *is a measurable function*
- $\forall x \in E$, $K(x, \cdot)$ *is a measure on* $(F, \mathcal{F})$

then K is called a transition kernel from $(E, \mathcal{E})$ into $(F, \mathcal{F})$.

For our purposes, we would like a measurable space $(E, \mathcal{E})$ to transition into itself, and for $K(x, \cdot)$ to be a probability measure. If the former holds, we say it is a transition kernel *on* $(E, \mathcal{E})$. If the latter holds, we call it a *transition probability kernel* or a *Markov kernel*. If both are true, we have a transition probability / Markov kernel on $(E, \mathcal{E})$.

It is possible to define a transition kernel through an integral, called an *integral kernel* [7]. If ν is some positive σ-finite measure on the measurable space $(E, \mathcal{E})$, and k a function $k : E \times E \mapsto \mathbb{R}_+$ that is measurable relative to the product σ-algebra $\mathcal{E} \otimes \mathcal{E}$, a kernel K on $(E, \mathcal{E})$ can be defined as

$$K(x, A) = \int_A k(x, y)\, \nu(\mathrm{d}y). \tag{42}$$

Then if for example $k(x, \cdot)$ is some density on $E = \mathbb{R}^d$, and ν the Lebesgue measure, we can define a Markov Kernel

$$K(x, A) = \int_A k(x, y)\, \mathrm{d}y\,. \tag{43}$$

It follows that $K(x, E) = 1$, and we call such a function k a *transition density*, and it can be interpreted as a conditional density.

References

[1] Claude J. P. Bélisle, H. Edwin Romeijn, and Robert L. Smith. Hit-and-run algorithms for generating multivariate distributions. *Mathematics of Operations Research*, 18(2):255–266, 1993.

[2] Yuansi Chen, Raaz Dwivedi, Martin J. Wainwright, and Bin Yu. Vaidya walk: A sampling algorithm based on the volumetric barrier. In *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1220–1227, 2017.

[3] Yuansi Chen, Raaz Dwivedi, Martin J. Wainwright, and Bin Yu. Fast mcmc sampling algorithms on polytopes, 2019.

[4] Daniel Foreman-Mackey, David W. Hogg, Dustin Lang, and Jonathan Goodman. emcee: The mcmc hammer. *Publications of the Astronomical Society of the Pacific*, 125(925):306, feb 2013.

[5] Chungmok Lee and Sungsoo Park. Chebyshev center based column generation. *Discrete Applied Mathematics*, 159:2251–2265, 12 2011.

[6] S. P. (Sean P.) Meyn and Richard L. Tweedie. *Markov chains and stochastic stability*. Communications and control engineering series. Springer-Vlg, Berlin ;, 1993.

[7] D. Revuz. *Markov chains*. North-Holland mathematical library ; 11. North-Holland, Amsterdam, rev. ed. edition, 1984.

[8] Christian P. Robert and George Casella. *Monte Carlo statistical methods*. Springer texts in statistics. Springer, New York, 1999.

[9] Robert L. Smith. Efficient monte carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research*, 32(6):1296–1308, 1984.

[10] A. Sokal. Monte carlo methods in statistical mechanics: Foundations and new algorithms. In *Functional Integration*, NATO ASI Series, pages 131–192. Springer US, Boston, MA.

[11] Erhan. Çınlar. *Probability and Stochastics*. Graduate Texts in Mathematics, 261. Springer New York, New York, NY, 1st ed. 2011. edition, 2011.