Discretization Approaches for SU(2) and Their Effects on Lattice Gauge Theories

---

Bachelorarbeit

in Physik

von Timo Jakobs,

angefertigt im
Helmholtz-Institut für Strahlen- und Kernphysik,

---

vorgelegt der Mathematisch-Naturwissenschaftlichen Fakultät
der Universität Bonn, Juli 2021.

1. Gutachter: Prof. Dr. Carsten Urbach

2. Gutachter: Johann Ostmeyer

Ich versichere, dass ich diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie die Zitate kenntlich gemacht habe.

Bonn, den _____ Unterschrift _____

In this thesis we will take a look at different approaches to approximating the group SU(2) by a finite subset in its function as the gauge group of a quantum field theory. This includes finite subgroups of SU(2) like the popular icosahedral approximation, four-dimensional versions of the Platonic solids and geodesic polytopes, as well as a generalized Fibonacci lattice. In order to test these approximations, Metropolis Monte Carlo simulations were implemented and run.

Contents

1 Introduction

Gauge theories are an essential building block of the standard model of particle physics, and therefore our understanding of modern physics. One of the early examples is quantum electrodynamics, which can be solved perturbatively and matched experiments to an enormous precision. With the introduction of further gauge theories, such as quantum chromodynamics, these perturbative approaches however quickly found their limits, especially when met with bigger coupling constants.

This was solved by putting these theories onto a space-time lattice and conducting Monte Carlo simulations of these systems. These however have their own limitations, which is why one now aims to simulate such theories on quantum computers.

For gauge theories this inevitably requires a parameterization of the gauge group G. As one of the biggest limitations of today's quantum hardware is the limited amount of quantum memory, this needs to be done efficiently.

This is why in the following work we take a look at different discretizations for one of the simplest non-trivial examples for such a gauge group, given by SU(2). On classical computers, the group elements are typically stored as four single or double precision numbers, taking up $4 \cdot 32 = 128$ and $4 \cdot 64 = 256$ bits of memory. In this way one can parameterize a vast amount of different group elements, such that for most practical intents and purposes this can be treated as an exact representation.

The most advanced quantum computers of today however still have significantly less than 100 qbits, rendering floating-point representations impractical and wasteful. Instead, we will discuss in the following, how one could pick a small *representative* subset of SU(2), and simulate the effects these approximations have on a lattice gauge theory.

2 The Lattice Action

The action of a pure gauge theory of gauge group G is given by

$$S_G = -\frac{1}{4} \int \mathrm{d}^4x \, \mathrm{Tr}\left[F_{\mu\nu}(x) F^{\mu\nu}(x)\right]$$

where the matrices $F_{\mu\nu}$ are given by

$$F_{\mu\nu} = \partial_\mu A_\nu - \partial_\nu A_\mu - ig[A_\mu, A_\nu].$$

with gauge field $A_\mu = A_\mu^a t_a$ in terms of the generators t_a of G as well as the coupling constant g [1]. In order to treat such a theory numerically, one first transitions to imaginary time. This is done by the transition $x^0 \to -ix_4$ which leads to $S_G \to iS_G^{(\mathrm{eucl.})}$ with

$$S_G^{(\mathrm{eucl.})} = \frac{1}{4} \int \mathrm{d}^4x \sum_{\mu,\nu} \mathrm{Tr}\left[F_{\mu\nu}(x) F_{\mu\nu}(x)\right].$$

where now $\eta_{\mu\nu} = \delta_{\mu\nu}$ and therefore $F_{\mu\nu} = F^{\mu\nu}$. Furthermore, one introduces a four dimensional space-time lattice with N^4 lattice sites, periodic boundary conditions and lattice spacing a. This is achieved by the following replacements:

$$\begin{aligned} x^\mu &\to a\, n^\mu \\ \int \mathrm{d}^4x &\to a^4 \sum_{n \in \mathbb{N}_N^4} \\ A_\mu(x) &\to A_\mu(an) \equiv A_\mu(n). \end{aligned}$$

where

$$\mathbb{N}_N^4 \in \left\{(a,b,c,d)^T \in \mathbb{N}^4 \,\middle|\, 0 \leq a,b,c,d < N\right\}.$$

In order to treat the interaction of gauge fields with additional fermion or scalar particles, it is also typical to introduce the so-called link variables U_μ given by

$$U_\mu(n) = \exp\left(igaA_\mu(n)\right) \in G.$$

With this in place, one can introduce the plaquette product $U_{\mu\nu}$:

$$U_{\mu\nu}(n) = U_\mu(n)\, U_\nu(n+e_\mu)\, U_\mu^\dagger(n+e_\nu) U_\nu^\dagger(n)$$

where the term plaquette denotes the path ordered product around a loop on the lattice. Using the Baker-Campbell-Hausdorf formula, this can be calculated to be

$$U_{\mu\nu}(n) = \exp\left(iga^2 F_{\mu\nu}(n)\right) + \mathcal{O}(a^5) = \mathbb{1} + iga^2 F_{\mu\nu}(n) - \frac{g^2a^4}{2}(F_{\mu\nu}(n))^2 + \mathcal{O}(a^5)$$

and thus

$$U_{\mu\nu}^\dagger(n) = \exp\left(-iga^2 F_{\mu\nu}(n)\right) + \mathcal{O}(a^5) = \mathbb{1} - iga^2 F_{\mu\nu}(n) - \frac{g^2a^4}{2}(F_{\mu\nu}(n))^2 + \mathcal{O}(a^5).$$

Such simplifications are reasonable, as the physical continuum limit is obtained by taking the lattice spacing $a \to 0$. Terms up to order four in a need to be considered, as they are required to transition from the sum over n to the original integral again. Higher order terms, however, will just vanish when taking the continuum limit and can therefore be dropped.

With this we can now express $F_{\mu\nu}$ in terms of $U_{\mu\nu}$:

$$a^4 (F_{\mu\nu})^2 \approx \frac{2}{g^2}\left(\mathbb{1} - \frac{1}{2}\left(U_{\mu\nu} + U_{\mu\nu}^\dagger\right)\right),$$

which leaves us with the euclidean lattice action

$$\begin{aligned} S_G^{(\text{lat.})} &= \frac{1}{4}\sum_{n\in\mathbb{N}_N^4}\sum_{\mu,\nu} a^4 \operatorname{Tr}\left[\left(F_{\mu\nu}(n)\right)^2\right] \\ &= \frac{1}{2}\sum_{n\in\mathbb{N}_N^4}\sum_{\mu<\nu} a^4 \operatorname{Tr}\left[\left(F_{\mu\nu}(n)\right)^2\right] \\ &\approx \frac{\beta}{2}\sum_{n\in\mathbb{N}_N^4}\sum_{\mu<\nu} \operatorname{Tr}\left[\mathbb{1} - \frac{1}{2}\left(U_{\mu\nu}(n) + U_{\mu\nu}^\dagger(n)\right)\right], \end{aligned}$$

where $\beta = \frac{2}{g^2}$. For $G = \mathrm{SU}(2)$ the generators t will be 2×2 matrices given by $t_a = \frac{\sigma_a}{2}$ where σ_a denotes the three Pauli matrices. This means that $\operatorname{Tr}[\mathbb{1}] = 2$. As can be seen in the explicit parameterization given later in eq. 1 also $\operatorname{Tr}[U_\mu] = \operatorname{Tr}\left[U_\mu^\dagger\right]$ holds for elements of SU(2). This allows one to simplify the action to be

$$S_{\mathrm{SU}(2)}^{(\text{lat.})} = \beta \sum_{n\in\mathbb{N}_N^4}\sum_{\mu<\nu}\left(1 - \frac{1}{2}\operatorname{Tr}\left[U_{\mu\nu}(n)\right]\right),$$

which will be the action used in the rest of this work and consequently referred to simply as S_G. This derivation was taken from [2] where it can be found in more detail and generality.

3 Discretization of SU(2)

A general Element $U \in \mathrm{SU}(2)$ in the fundamental representation can be written as

$$U = \begin{pmatrix} u & w \\ -\overline{w} & \overline{u} \end{pmatrix} \qquad \text{with} \quad u, w \in \mathbb{C} \quad \text{and} \quad \det U = |u| + |w| = 1. \tag{1}$$

The group composition in this representation is simply implemented by matrix multiplication. If one sets $u = a + ib$ and $w = c + id$ the determinant constraint implies that the point $(a, b, c, d)^T$ lies on the unit sphere S_3 embedded in $\mathbb{R}^4$. In order to obtain the group composition in terms of the real parameters (a, b, c, d) one simply carries out the matrix multiplication $U'' = UU'$ for two points $U \equiv (a, b, c, d)$ and $U' \equiv (a', b', c', d')$:

$$\begin{aligned} a'' &= aa' - bb' - cc' - dd' \\ b'' &= ab' + ba' + cd' - dc' \\ c'' &= ac' - bd' + ca' + db' \\ d'' &= ad' + bc' - cb' + da'. \end{aligned}$$

In order to represent this group composition more conveniently, one can introduce the quaternions $\mathbb{H}$, which will be the notation used in the following chapter. They form an extension of the complex numbers to four dimensions. The basis is usually notated as $(1, i, j, k)$ with the non-commutative multiplication defined by

$$i^2 = j^2 = k^2 = ijk = -1.$$

Therefore the group composition can be implemented by the quaternion product $q'' = qq'$ for two quaternions $q = a + bi + cj + dk$ and $q' = a' + b'i + c'j + d'k$. One can furthermore introduce the norm $\|q\| = \sqrt{a^2 + b^2 + c^2 + d^2}$ similar to the euclidean norm of $\mathbb{R}^4$ constraining elements of SU(2) to the unit quaternions $\mathbb{H}_1$:

$$\mathbb{H}_1 = \{x \in \mathbb{H} : \|x\| = 1\}.$$

To implement the previously discussed group action in terms of unit quaternions, we also need to be able to calculate the traces of group elements. This can be achieved by

$$\mathrm{Tr}\,[U] = \mathrm{Tr}\left[\begin{pmatrix} a + ib & c + id \\ -c + id & a - ib \end{pmatrix}\right] = 2a =: 2\mathrm{Re}(q).$$

In the following, different approaches to finding well distributed finite subsets of $\mathbb{H}_1$ will be discussed. The goal here is to approximate SU(2) in its function as a gauge group, which is why these subsets will be referred to as gauge sets. Explicitly, this means the Link Variables U_μ will be restricted to members of these subsets. The main metric for success is then given by the results obtained in chapter 5, where numerical simulations of the gauge theory will be conducted.

3.1 Finite Subgroups of SU(2)

The first thing to take a look at are the finite subgroups of SU(2). As SU(2) is a double cover of the group of rotations SO(3) in $\mathbb{R}^3$ these can be constructed by taking the Cartesian product of the cyclic group of order 2 with the subgroups of SO(3). The finite subgroups of SO(3) can be constructed by all the symmetry transformations of regular polygons as well as rotations of the Platonic solids [3]. The groups generated by the regular polygons were however not considered in this work, as their elements are restricted to planes within $\mathbb{H}_1$. Therefore, they will most likely

group	order	elements
$\overline{T}$	24	all sign combinations of $\left\{\pm 1, \pm i, \pm j, \pm k, \frac{1}{2}\left(\pm 1 \pm i \pm j \pm k\right)\right\}$
$\overline{O}$	48	all sign combinations and permutations of $\left\{\pm 1, \pm i, \pm j, \pm k, \frac{1}{2}\left(\pm 1 \pm i \pm j \pm k\right), \frac{1}{\sqrt{2}}\left(\pm 1 \pm i\right)\right\}$
$\overline{I}$	120	all sign combinations and even permutations of $\left\{\pm 1, \pm i, \pm j, \pm k, \frac{1}{2}\left(\pm 1 \pm i \pm j \pm k\right), \frac{1}{2}\left(1 + \tau i + \frac{j}{\tau} + 0k\right)\right\}$

Table 1: Quaternionic Representation of $\overline{T}$, $\overline{O}$ and $\overline{I}$ as found in [4], where $\tau = \frac{1+\sqrt{5}}{\sqrt{2}}$ denotes the golden ratio

not be a good approximation of SU(2).

The five Platonic solids give rise to three distinct subgroups. The first one would be the tetrahedral group T with 12 elements, which as the name suggests is generated by the orientation preserving rotations of a tetrahedron. Next up is the octahedral group O with 24 elements, generated by the rotational symmetries of the octahedron (or the Cube respectively, as it is the geometric dual of the octahedron). Finally there is the icosahedral group I with 60 elements generated by the rotational symmetries of the icosahedron and dodecahedron.

Expanding these groups to SU(2) doubles the element counts and therefore gives rise to the so-called binary tetrahedral group $\overline{T}$, the binary octahedral group $\overline{O}$ and the binary icosahedral group $\overline{I}$, with 24, 48 and 120 elements respectively. The quaternionic representation used in the following can be found in table 1.

3.2 Regular Polytopes in Four Dimensions

Another useful property of Platonic solids is, that their vertices are, if scaled accordingly, a well distributed set of points on S_2. As we are trying to distribute points on S_3, there is a good chance, that the vertices of something like a four dimensional Platonic solid would make for a useful gauge set. For higher dimensions than three, one refers to such contraptions as a regular polytope, and there are six of these in four dimension. They are denoted by the number of cells (i.e. three-dimensional faces) they have. The vertex count ranges from the 5 vertices of the 5-cell C_5 to the 600 vertices of the 120-cell C_{120}. The vertices of all six can be found in table 2.

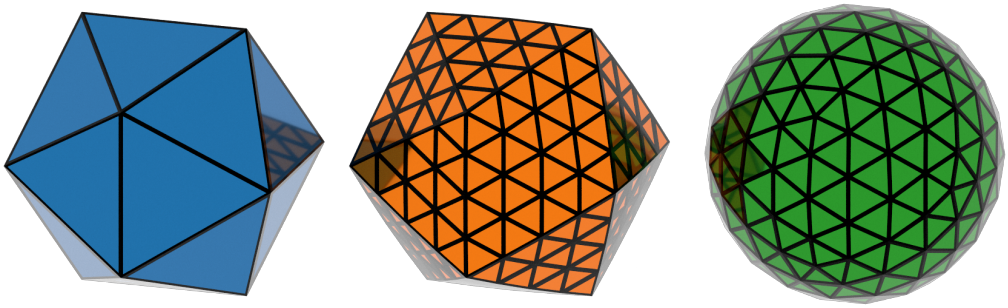
It is noteworthy that the vertices of C_{24} and C_{600} coincide with $\overline{T}$ and $\overline{I}$ respectively. The vertices of C_{16} also form a group called the binary dihedral group $\overline{D}_4$ [4].

3.3 Geodesic Polytopes in Four Dimensions

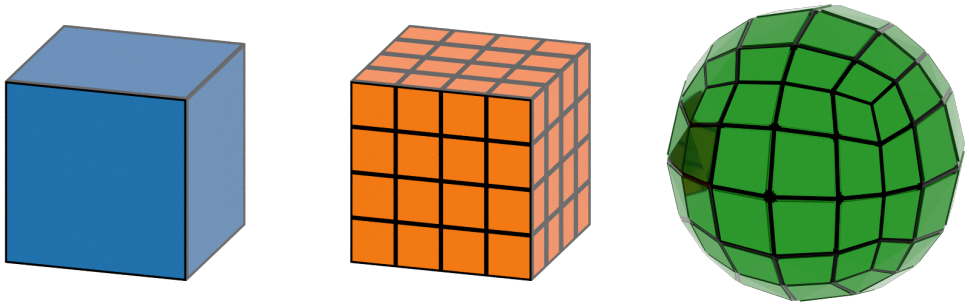
Furthermore, we can make use of the fact, that the faces of the Platonic solids are regular polygons. This allows for the construction of so-called geodesic polyhedra [5]. As seen in figure 1a the basic idea here is to subdivide the faces of a Platonic solid into triangles and then project the obtained vertices onto the sphere. Although the cells of the regular polytopes in four dimensions are the Platonic solids, such a procedure is complicated by the fact that most of the Platonic solids do not tile space on their own. Therefore, finding appropriate subdivision in a scaleable way is not a trivial task.

polytope	# vertices	vertices
C_5	5	$\{1, -\frac{1}{4} + \eta i + \eta j + \eta k, -\frac{1}{4} + \eta i + \eta j + \eta k,$ $-\frac{1}{4} + \eta i + \eta j + \eta k, -\frac{1}{4} + \eta i + \eta j + \eta k\}$
C_{16}	8	$\{\pm 1, \pm i, \pm j, \pm k\}$
C_8	16	all sign combinations of $\left\{\frac{1}{2}(\pm 1 \pm i \pm j \pm k)\right\}$
C_{24}	24	$C_8 \cup C_{16}$ (same as $\overline{T}$)
C_{600}	120	$C_{24} \cup \left\{\text{sign comb. and even perm. of } \frac{1}{2}\left(1 + \tau i + \frac{j}{\tau} + 0k\right)\right\}$ (same as $\overline{I}$)
C_{120}	600	$\{ab \,\vert\, a \in C_5,\, b \in C_{600}\}$

Table 2: The vertices of the regular polytopes in four dimensions



(a) Construction of an icosphere



(b) Construction of a cube-sphere

Figure 1: Construction of geodesic polytopes: One takes a platonic solid (blue), subdivides its surface (orange) and projects the obtained points onto the sphere (green).

The only exception to this would be C_8, as its eight cells are cubes. In this case any of the eight *surface cubes* can be trivially filled with m^3 cubes of side length $\frac{1}{m}$ for $m \in \mathbb{N}/\{0\}$. As seen in figure 1b the grid obtained by this procedure in three dimensions is somewhat reminiscent of a volleyball. So in the following we will denote such a *volleyball* lattice with n subdivision by V_n.

As any of the eight *surface cubes* is fixed by holding one of the coordinates at $\pm\frac{1}{2}$ the set of coordinates before projection onto the sphere $\tilde{V}_n$ is obtained by

$$\tilde{V}_n = \left\{ \text{all permutations of } \frac{1}{2}\left(\pm 1 + v_n(a)i + v_n(b)j + v_n(c)k\right) \,\middle|\, a,b,c \in \mathbb{N} : 0 \le a,b,c \le n+1 \right\}$$

with v_n being defined as

$$v_n(m) = 1 - \frac{2m}{n+1}\,.$$

V_n is then simply obtained by dividing out the norm for every point in $\tilde{V}_n$:

$$V_n = \left\{ \frac{q}{\|q\|} \,\middle|\, q \in \tilde{V}_n \right\}.$$

The number of distinct elements of V_n can be calculated to be

$$|V_n| = \underbrace{16}_{\substack{\text{vertices}\\ \text{of } C_8}} + \underbrace{8n^3}_{\substack{n^3 \text{ vertices added in}\\ \text{every one of the 8 cells}}} + \underbrace{24n^2}_{\substack{n^2 \text{ vertices added on}\\ \text{every one of the 24 faces}}} + \underbrace{32n}_{\substack{n \text{ vertices added on}\\ \text{every one of the 32 edges}}}.$$

The main advantage of the *volleyball* lattice is that one can vary the number of subdivisions, and therefore allows some control over the granularity of the gauge set. Due to the method of construction, the density of vertices will however increase towards the vertices of C_8. This will probably lead to some systematic deviations when using V_n as an approximation of SU(2).

3.4 Fibonacci Lattice on SU(2)

The final discretization of SU(2) considered in this work is a higher dimensional version of the so-called Fibonacci lattice. They offer an elegant and deterministic solution to the problem of distributing a given amount of points on a two-dimensional surface. They are used in numerous fields of research such as numerical analysis or computer graphics, mostly to approximate spheres (as e.g. shown in figure 2). Mainly inspired by [6] we will now attempt to construct a similar lattice for S_3.

The two-dimensional Fibonacci lattice is usually constructed within a unit square $[0,1)^2$ as

$$\tilde{L}_n = \left\{ \tilde{t}_m \middle| 0 \le m < n,\ m \in \mathbb{N} \right\}$$
$$\text{with} \qquad \tilde{t}_m = \begin{pmatrix} x_m \\ y_m \end{pmatrix} = \begin{pmatrix} \frac{1}{\tau} & \mod & 1, \frac{m}{n} \end{pmatrix}.$$

This can be generalized to the hypercube $[0,1)^N$ embedded in $\mathbb{R}^N$:

$$L_n = \{ t_m | 0 \le m < n,\ m \in \mathbb{N} \}$$
$$t_m = \begin{pmatrix} t_m^1 \\ t_m^2 \\ \vdots \\ t_m^N \end{pmatrix} = \begin{pmatrix} \frac{m}{n} & & \\ a_1\, m & \mod & 1 \\ \vdots & & \\ a_{N-1}\, m & \mod & 1 \end{pmatrix} \qquad \text{with} \qquad \frac{a_i}{a_j} \notin \mathbb{Q} \quad \text{for} \quad i \neq j.$$

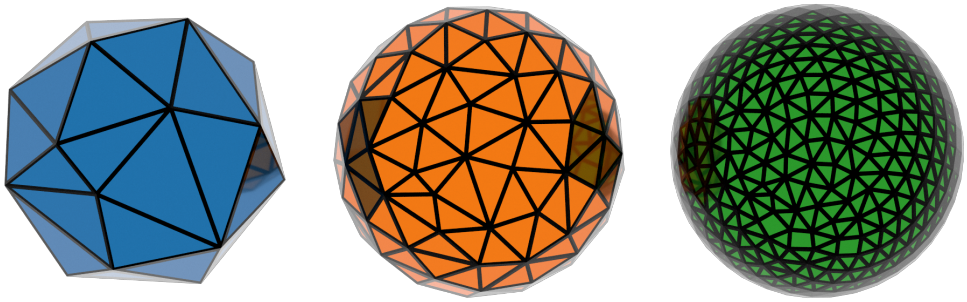


Figure 2: Fibonacci lattices on S_2 with 20 (blue), 100 (orange) and 500 (green) vertices

A simple choice for the constants a_i would be the square roots of the prime numbers:

$$(a_1, a_2, a_3, \dots) = (\sqrt{2}, \sqrt{3}, \sqrt{5}, \dots)$$

The points in L_n are then evenly distributed within the given Volume. All that is left to do is to map these points onto a given compact manifold M, in our case SU(2). In order to maintain the even distribution of the points, such a map Φ needs to be volume preserving in the sense that

$$\int_{\Omega \subseteq [0,1)^N} \mathrm{d}^N x = \frac{1}{\mathrm{Vol}(M)} \int_{\Phi(\Omega) \subseteq M} \mathrm{d}V_M \tag{2}$$

holds for all regions Ω.

To find such a map for S_3 (and therefore SU(2)) we start by embedding S_3 into $\mathbb{H}$ by introducing spherical coordinates

$$\begin{gathered} z(\psi, \theta, \phi) = \cos\psi + \sin\psi\cos\theta\, i + \sin\psi\sin\theta\cos\phi\, j + \sin\psi\sin\theta\sin\phi\, k \\ \text{with} \quad \psi \in [0, \pi), \quad \theta \in [0, \pi) \quad \text{and} \quad \phi \in [0, 2\pi). \end{gathered}$$

Therefore, the metric tensor g_{ij} in terms of the spherical coordinates $(y_1, y_2, y_3) := (\psi, \theta, \phi)$, treating $(1, i, j, k)$ as a euclidean basis, is given by

$$g_{ij} = \frac{\partial z^a}{\partial y^i} \frac{\partial z^b}{\partial y^j} \delta_{ab} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \sin^2\psi & 0 \\ 0 & 0 & \sin^2\psi \sin^2\theta \end{pmatrix}_{ij}.$$

From this one can calculate the Jacobian $\sqrt{|g|}$ to be

$$\sqrt{|g|} = \sin^2\psi \sin\theta.$$

As $\sqrt{|g|}$ factorizes nicely into functions only dependent on one coordinate, one can construct a bijective map Φ^{-1} mapping S_3 to $[0,1)^3$ given by $\Phi^{-1}(\psi, \theta, \phi) = \left(\Phi^{-1}(\psi), \Phi^{-1}(\theta), \Phi^{-1}(\phi)\right)$ with

$$\begin{aligned} \Phi^{-1}(\psi) &= \frac{\int_0^\psi \mathrm{d}\tilde{\psi} \sin^2\tilde{\psi}}{\int_0^\pi \mathrm{d}\tilde{\psi} \sin^2\tilde{\psi}} &&= \frac{1}{\pi}\left(\psi - \frac{1}{2}\sin(2\psi)\right) \\ \Phi^{-1}(\theta) &= \frac{\int_0^\theta \mathrm{d}\tilde{\theta} \sin\theta}{\int_0^\pi \mathrm{d}\tilde{\theta} \sin\theta} &&= \frac{1}{2}\left(1 - \cos(\theta)\right) \\ \Phi^{-1}(\phi) &= \frac{\int_0^\phi \mathrm{d}\tilde{\phi}}{\int_0^{2\pi} \mathrm{d}\tilde{\phi}} &&= \frac{1}{2\pi}\phi. \end{aligned}$$

Looking at some region $\Omega = \Phi^{-1}(\tilde{\Omega})$ one can see that the inverse map $(\Phi^{-1})^{-1} := \Phi$ trivially fulfills equation 2. A Fibonacci-like lattice on S_3 would therefore be given by

$$F_n = \left\{ z\left(\psi_m(t_m^1), \theta_m(t_m^2), \phi_m(t_m^3)\right) \middle| \, 0 \leq m < n, \; m \in \mathbb{N} \right\}$$

$$\begin{aligned} \text{with} \quad \psi_m(t_m^1) = f^{-1}\left(t_m^1\right) \quad &= f^{-1}\left(\frac{m\pi}{n+1}\right), \qquad f(x) = x - \frac{1}{2}\sin(2x), \\ \theta_m(t_m^2) = \cos^{-1}\left(t_m^2\right) &= \cos^{-1}\Big(1 - 2(m\sqrt{2} \mod 1),\Big) \\ \text{and} \quad \phi_m(t_m^3) = 2\pi t_m^3 \qquad &= 2\pi(m\sqrt{3} \mod 1). \end{aligned}$$

The Fibonacci approach offers two important advantages, compared to the previously proposed discretizations. First, it can generate a representative gauge set for any given number of elements. As the motivation for these discretizations is to be used for simulations on quantum computers, this effectively means that one will always be able to achieve maximum granularity for a given amount of quantum memory.

The way the four dimensional version was constructed also suggests, that one can obtain similar procedures for other compact manifolds and therefore other continuous gauge groups. To put this statement into rigorous mathematical terminology or to provide a proof is beyond the scope of this work.

However, as long as one manages to introduce coordinates $(y_1, ..., y_n)$ on the manifold, and the metric g, obtained by a pullback from $n+1$ dimensional euclidean space, takes a form s.t. $|g| = \prod_i f_i(y_i)$, one should be able to obtain a map similar to Φ^{-1}. Therefore, there is at least a good chance that such lattices can e.g. be constructed for SU(3) and therefore be used for QCD simulations.

The only disadvantage might be that in spite of the uniform distribution of the generated points, some residual bias might be left. This could lead to some systematic deviations showing for smaller lattice sizes n. In contrast to the volleyball lattice, these should however get smaller for bigger n.

4 Metropolis Monte Carlo Method

In order to evaluate the effects of the proposed discretizations of SU(2) the so-called Metropolis Monte Carlo Method was used. The goal of this method is to estimate the expectation value of an operator $\mathcal{O}$ by numerically evaluating the path integral given by

$$\begin{aligned} \langle \mathcal{O} \rangle &= \frac{1}{Z} \int \left(\prod_\mu \mathcal{D}U_\mu \right) \mathcal{O} \exp\left(-S_G(U)\right) \\ \text{with} \qquad Z &= \int \left(\prod_\mu \mathcal{D}U_\mu \right) \exp\left(-S_G(U)\right) \end{aligned}$$

This is done by randomly generating lattice configurations $\boldsymbol{U} := \{U_\mu(n)\}$ with a probability distribution given by

$$\rho(\boldsymbol{U}) = \frac{\exp\left(-S_G(\boldsymbol{U})\right)}{Z}. \tag{3}$$

For a set of N configurations $\{\boldsymbol{U}^i\}$ the expectation value of $\mathcal{O}$ is then obtained by a simple arithmetic average

$$\langle \mathcal{O} \rangle = \frac{1}{N} \sum_{\boldsymbol{U} \in \{\boldsymbol{U}^i\}} \mathcal{O}(\boldsymbol{U}).$$

There are numerous ways of generating lattice configurations $\{\boldsymbol{U}^i\}$. A popular choice is the so-called *Metropolis algorithm*, first proposed in [7] and applied to quantum simulations in [8]. A simple implementation can be found in algorithm 1.

Algorithm 1 Metropolis Monte Carlo

```
1:  ⟨O⟩ := 0
2:  U = (Some initial Lattice configuration)
3:  for N iterations do
4:    for all U_μ(n) ∈ U  do          // Iteration over all link variables
5:      U' := U                       // Copy the configuration U into a new
6:                                    //    configuration U'
7:      U'_μ(n) ← U_rand              // Replace U'_μ(n) with a random
8:                                    //    member of the gauge set U_rand ∈ G
9:      ΔS := S_G^(eucl.)(U) − S_G^(eucl.)(U')   // Calculate the difference in action
10:                                   //    for the transition U → U'
11:     if ΔS < 0 ∨ exp(−ΔS) > r then   // with 0 ≤ r ≤ 1 beeing a random
12:                                   //    number with uniform distribution
13:       U ← U'                      // replace U with U'
14:     end if
15:   end for
16:   ⟨O⟩ ← ⟨O⟩ + O(U)/N              // Measure O(U)
17: end for
```

As an initial configuration, one usually either chooses a cold starting configuration, i.e. $U_\mu(n) = 1$ or a hot one, where every link variable is set to a random element of SU(2). After that it takes numerous iterations until the configuration reaches *thermal equilibrium*. A common way to figure out how many iterations this takes is to run the simulation once with a hot and once with a cold start, and see where the measured values start to agree.

After equilibrium is reached, every successive iteration will generate a new random lattice configuration distributed according to eq. 3. Therefore, $\langle \mathcal{O} \rangle$ can then be measured to arbitrary precision by simply generating a sufficient amount of lattice configurations. The only limit to this would be the finite precision of floating-point operations on digital computers, as well as the limited randomness of pseudo random number generators.

A key element of such simulations takes place in line 11 with the generation of a new random element. The implementation of this for the continuous as well as the discrete case will be presented in the following sections.

4.1 The Continous Case

The first idea that comes to mind for the continuous case is just to pick a random point on the sphere S_3. Procedures to find uniformly distributed random points on spheres of arbitrary dimensions are well known [9]. One such procedure to get the Euclidean coordinates x_{rand} of a

G	$N(G)$ 1
C_{16}	$\{\pm i, \pm j, \pm k\}$
$\overline{O}$	$\left\{\frac{1}{2}(1+a)\middle\vert a \in \{\text{sing comb. of } \pm i \pm j \pm k\}\right\}$
$\overline{T}$	$\left\{\frac{1}{\sqrt{2}}(1 \pm a)\middle\vert a \in \{i, j, k\}\right\}$
$\overline{I}$	$\left\{\frac{1}{2}(\tau + a)\middle\vert a \in \left\{\text{sign comb. and even perm. of } \left(\pm 0i \pm 1j \pm \frac{k}{\tau}\right)\right\}\right\}$

Table 3: caption

random point on S_n would be given by

$$x_{\text{rand}} = \frac{1}{\sqrt{\sum_{i=1}^{n+1}(x^i_{\text{rand}})^2}} \begin{pmatrix} x^1_{\text{rand}} \\ x^2_{\text{rand}} \\ \vdots \\ x^{n+1}_{\text{rand}} \end{pmatrix} \tag{4}$$

where $\{x^i_{\text{rand}}\}$ are random numbers generated by a normal distribution with expectation value $\mu = 0$ and an arbitrary standard deviation $\sigma \neq 0$.

The problem with this approach is that for high values of β line 11 in the algorithm will quite often evaluate to `false`, as ΔS is likely to be big. This leads to little change in the configuration per iteration, and therefore high run times to generate sufficient statistics.

This is why one usually aims to generate an element $U^{\text{new}}_\mu(n)$ which is in the vicinity of the old element U^{old}_μ. This achieved by

$$U^{\text{new}}_\mu(n) = V\, U^{\text{old}}_\mu(n)$$

where V is a random group element within the angular distance $\pi\delta$ of 1. V can be found as

$$V = \cos(\pi d) + \sin(\pi d)\,(ix_1 + jx_2 + kx_3) \tag{5}$$

where d is a random number uniformly distributed within the interval $[-\delta, \delta]$ and (x_1, x_2, x_3) are a random point on S_2 found by applying eq. 4.

4.2 Finite Gauge Sets

Furthermore, we need to deal with finite gauge sets G. As proposed in [10] one can employ a similar procedure to the continuous case for the finite subgroups of SU(2) by simply setting V to a random element of the group adjacent to 1. These would be given by the elements $N(G)$ that minimize the value of $\|V - 1\|$ and can be found explicitly in table 3. Multiplying them from the left amounts to a small rotation, mapping all vertices to one of their adjacent vertices.

All other lattices do not possess group properties, so the transition to a neighboring vertex needs to be implemented in another way. In the case of C_5 all five vertices are adjacent to one another, so one can simply pick U^{new}_μ from one of the four vertices that is not U^{old}_μ.

Finding the four adjacent vertices of a given vertex of C_{120} is a less trivial task. As quaternionic multiplication from the left can be interpreted as a rotation in $\mathbb{R}^4$ the construction of C_{120} found in table 2 suggests that the vertices of C_{120} are given by the vertices of $\overline{I}$ as well as four rotated copies of it. In order to keep track of where we are on C_{120}, we can therefore just keep track of

where we are on $\overline{I}$ (for which we can make use of the group properties) and whether one of the four rotations needs to be applied.

Unfortunately we can not use the five elements of C_5 as these rotations, as they implement rotations by about $\cos^{-1}\left(\frac{1}{4}\right) \approx 75\,^\circ$ and therefore will most certainly not map a vertex of $\overline{I}$ to an adjacent one of C_{120}.

In order to obtain a useful set of rotation matrices, we take a look at the vertices $1, i, j$ and k of $\overline{I}$. As these translate to the basis vectors of $\mathbb{R}^4$ the columns of the rotation matrices must be given by the neighbors of these vertices. To figure out which of the $(4!)^3 = 13824$ combinations of these columns leads to the right matrices, a brute force symbolic calculation was used. This indeed gave a unique solution given by

$$R = \left\{ \begin{bmatrix} \rho & \theta & \theta & \theta \\ -\theta & \rho & \theta & -\theta \\ -\theta & -\theta & \rho & \theta \\ -\theta & \theta & -\theta & \rho \end{bmatrix}, \begin{bmatrix} \rho & -\theta & -\theta & \theta \\ \theta & \rho & \theta & \theta \\ \theta & -\theta & \rho & -\theta \\ -\theta & -\theta & \theta & \rho \end{bmatrix}, \begin{bmatrix} \rho & -\theta & \theta & -\theta \\ \theta & \rho & -\theta & -\theta \\ -\theta & \theta & \rho & -\theta \\ \theta & \theta & \theta & \rho \end{bmatrix}, \begin{bmatrix} \rho & \theta & -\theta & -\theta \\ -\theta & \rho & -\theta & \theta \\ \theta & \theta & \rho & \theta \\ \theta & -\theta & -\theta & \rho \end{bmatrix} \right\}$$

with

$$\rho = \frac{1+3\sqrt{5}}{8} \qquad \text{and} \qquad \theta = \frac{\sqrt{5}-1}{8}.$$

To implement this we introduced a rotation index $r \in \{0,1,2,3,4\}$ which indicates by which of the four matrices in R the vertex $U \in \overline{I}$ needs to be rotated. $r = 4$ covers the case that no rotation needs to be applied. With this we can then implement the following algorithm, to get one of the four adjacent vertices of C_{120}, in terms of the old vertex specified by r_{old} and U_{old}, as well as a random integer $0 \leq d_{\text{rand}} \leq 3$.

Algorithm 2 Neighbor Vertex Algorithm

Require: $U_{\text{old}} \in \overline{I}, \quad r_{\text{old}} \in \{0,1,2,3,4\}, \quad d_{\text{rand}} \in \{0,1,2,3\}$

if $r_{\text{old}} = 4$ **then** // The current vertex is also a vertex of $\overline{I}$.
  $r_{\text{new}} \leftarrow d_{\text{rand}}$ // Therefore we pick a random rotation
  $U_{\text{new}} \leftarrow U_{\text{old}}$ // matrix and leave u unchanged
else if $d_{\text{rand}} = 3$ **then** // If the current vertex is on one of the
  $r_{\text{new}} \leftarrow 4$ // rotated copies of $\overline{I}$, there is a one
  $U_{\text{new}} \leftarrow U_{\text{old}}$ // in four chance of it rotating back
else
  $r_{\text{new}} \leftarrow r_{\text{old}} + d_{\text{rand}} + 1 \mod 4$ // Pick a new rotation index s.t. $r_{\text{new}} \neq r_{\text{old}}$
  for $V \in N(\overline{I})$ **do**
    if isNeighbor $(r_{\text{new}}, VU_{\text{old}})$ **then** // Check whether r_{new} and VU_{old} give rise
      $U_{\text{new}} \leftarrow VU_{\text{old}}$ // to an adjacent vertex
      break
    end if
  end for
end if
return U_{new}, r_{new}

Next up is V_n. For this we simply keep track of the corresponding vertex of $\tilde{V}_n$, as here addition or subtraction of $\frac{1}{n+1}$ to any component of a given vertex will lead to an adjacent vertex. The only restriction to this would be that all components need to be within the interval $\left[-\frac{1}{2}, \frac{1}{2}\right]$, as well as that at least one component needs to equal $\pm\frac{1}{2}$. For $n = 0$ this also covers C_8.

Lastly, we are left with the Fibonacci lattice. As best seen in figure 2 the notion of an adjacent vertex is quite hard to define for the Fibonacci lattice, as the amount of adjacent vertices, as well as their exact distances vary for each vertex. Therefore, here we went with the less elegant approach of simply picking a random element from the whole lattice.

5 Results

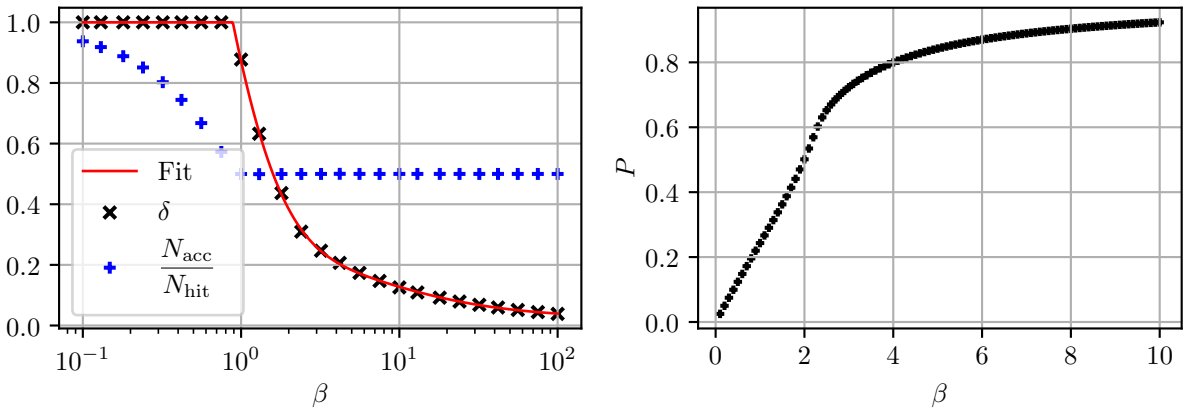
Finally, it is time to code up the simulations. This was done in `C++` making use of Nvidia's `CUDA` library to allow for parallel execution on GPUs. The source code including further documentation can be found at `https://github.com/teajay99/bachelor_thesis`.

5.1 Validation of the Continuous Case

First up, we need to figure out how to choose the δ parameter when generating the transition elements V in eq. 5. This is a bit of a balancing act, as for bigger values of δ fewer configuration changes will be accepted, leading to successive configurations being very similar. With small δ, more of the configuration changes will be accepted, but now these changes are very small and the successive configuration will still look a lot like the previous one.

The good news is, however, that the choice of δ is not critical, as for a sufficient amount of iterations, it will not affect the measured quantities themselves. Correlation between successive configurations however need to be taken into account when estimating the uncertainties of a measured quantity. This was done using the `uwerrprimary` function from the *R* library `hadron`.

To evaluate how well δ is chosen, one can take a look at the acceptance rate $\frac{N_{\text{acc}}}{N_{\text{hit}}}$, i.e. the ratio of line 11 in algorithm 1 evaluating to `true` and the total number of evaluations. A good rule of thumb is to get the acceptance rate to about 50 %.



(a) Acceptance Rate Calibration

(b) $P(\beta)$ measured on an 8^4 lattice for 100,000 iterations. All error bars are significantly smaller than the plotted points and were therefore omitted.

Figure 3

In order to reliably meet this rule, a simply zero search for the function

$$f(\delta) = \left(\frac{N_{\text{acc}}}{N_{\text{hit}}}\right)(\delta, \beta) - 0.5$$

was implemented and run for 25 logarithmically spaced values of β between $\beta = 0.1$ and $\beta = 100$. The results with their corresponding acceptance rate can be found in figure 3a. Also, a function of the form

$$\delta(\beta) = \min\left(1, \frac{a}{\beta - e} + \frac{b}{(\beta - e)^2} + \frac{c}{(\beta - e)^3} + \frac{d}{(\beta - e)^4} + f\right)$$

was fitted to the data, with the results reading

$$\begin{aligned} a &= \ 1.6477 & b &= -6.9405 \\ c &= \ 18.415 & d &= -0.03858 \\ e &= -1.3638 & f &= \ 0.02372. \end{aligned}$$

The fit looks reasonable, and will therefore be used to serve the values for δ in the following measurements. A further inspection into e.g. the uncertainties of the fit parameters or error of regression were not done, as again the exact choice of δ is not critical, and the fit itself has no physical implications.

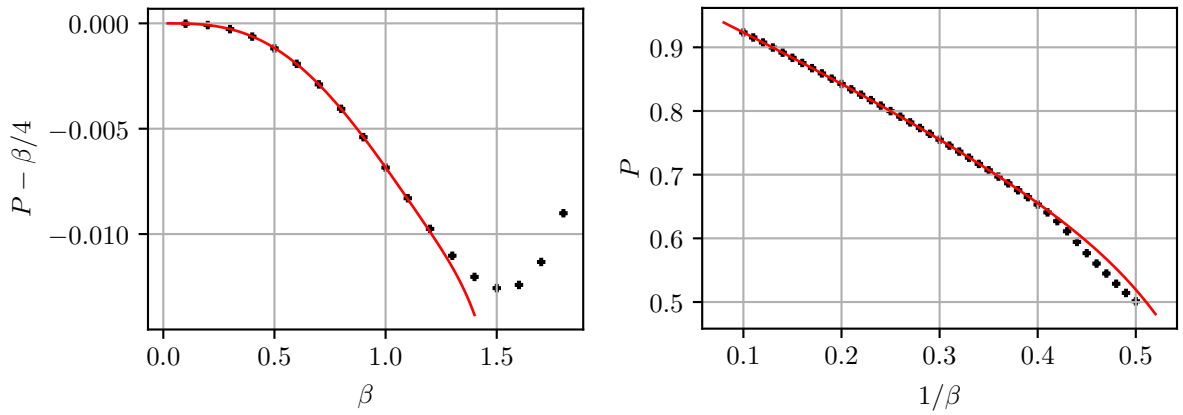
Another trick typically implemented to decrease the correlation between successive configurations is to probe each link variable multiple times before moving on to the next one. For all the simulations here, this was done 10 times.

The only thing left is to pick is an operator $\mathcal{O}$ to measure. A simple and popular choice here is to go with the average plaquette P measured by

$$P = \frac{1}{\mathcal{N}_P} \sum_{n \in \mathbb{N}_N^4} \sum_{\mu < \nu} \frac{\beta}{2} \mathrm{Tr}\left[U_{\mu\nu}(n)\right]$$

where $\mathcal{N}_P = 6N^4$ denotes the number of different plaquettes for a lattice of size N^4.

With this we then ran the continuous case for $\beta \in \{0.1, 0.2, \ldots, 9.8, 9.9, 10\}$ on an 8^4 lattice for $104,000$ iterations starting from a hot configuration. The first $4,000$ iterations were treated as thermalization iterations, and the successive $100,000$ will be used as reference data set, when evaluating the approximations of SU(2). The obtained values are plotted in figure 3b.



(a) Strong coupling expansion (red) compared to the measured data set (black). The uncertainties of the measured data are significantly smaller than the plotted points.

(b) Weak coupling expansion (red) compared to the measured data set (black). The uncertainties of the measured data are significantly smaller than the plotted points.

Figure 4

In order to verify that the measured data is sound, one can take a look at the so-called strong coupling expansion given by

$$P_{\text{s. c.}}(\beta) = \frac{1}{4}\beta - \frac{1}{96}\beta^3 + \frac{7}{1536}\beta^5 - \frac{31}{23040}\beta^7 + \frac{4451}{8847360}\beta^9 - \frac{264883}{1486356480}\beta^{11} + \frac{403651}{5945425920}\beta^{13} - \frac{1826017873}{68491306598400}\beta^{15}$$

as found in [11]. As the name suggest, this approximates P for big values of the coupling constant g, and therefore small values of β. As can be seen in figure 4a, the measured values closely agree with the prediction up to about $\beta = 1.2$.

The same article also gives a weak coupling expansion in $\frac{1}{\beta}$ up to order fifteen for lattice size 6^4:

$$P(\beta)_{\text{w. c.}} = \sum_{i=1}^{15} \frac{\gamma_i}{(\beta)^i},$$

where the coefficients γ_i can be found in table 4. For this, another data set with $10,000$ iterations was generated to account for the smaller lattice size. The results can be found in figure 4b. The values agree within the error bars up to about $1/\beta = 0.25$, and then start to slowly deviate. Nevertheless, they confirm that the simulation is working correctly.

i	1	2	3	4	5	6	7	8
γ_i	0.7498	0.1511	0.1427	0.1747	0.2435	0.368	0.5884	0.98

i	9	10	11	12	13	14	15
γ_i	1.6839	2.9652	5.326	9.7234	17.995	33.690	63.702

Table 4: Weak coupling expansion for a 6^4 lattice

As we now have a solid data set to compare against, we can take a look at the proposed approximations of SU(2). This will be done in two steps. As pointed out in [10], finite gauge groups usually show a phase transition towards higher values of β. Finding these will therefore be our first concern. In the second step, we will also take a look at systematic deviations for some specific values of β.

5.2 Phase Transitions

For each of the considered gauge sets, 4000 thermalization iterations were run, and another 3000 for measurements afterwards. This was repeated twice, once with a hot start, and once with a cold one. The results are plotted in figures 5 to 9. Shown are the deviations $P - P_{\text{ref}}$ from the previously generated reference data. The y-axis was set to be linear for the interval $[-10^{-3}, 10^{-3}]$ and logarithmic beyond that, in order to display both the statistical fluctuations, as well as systematic deviations in a meaningful way.

We begin with the finite subgroups $\overline{T}$, $\overline{O}$ and $\overline{I}$ for which the results can be found in figure 5. Here one can see that for small β, the measured values mostly agree with the continuous case. Towards bigger values of β, some systematic deviations show up, until the lattice configuration undergoes a phase transition and *freezes* at $P \approx 1$. To measure $P = 1$ all link variables need to be at the same gauge set element. This happens, as ΔS, because of the discretization, now has a

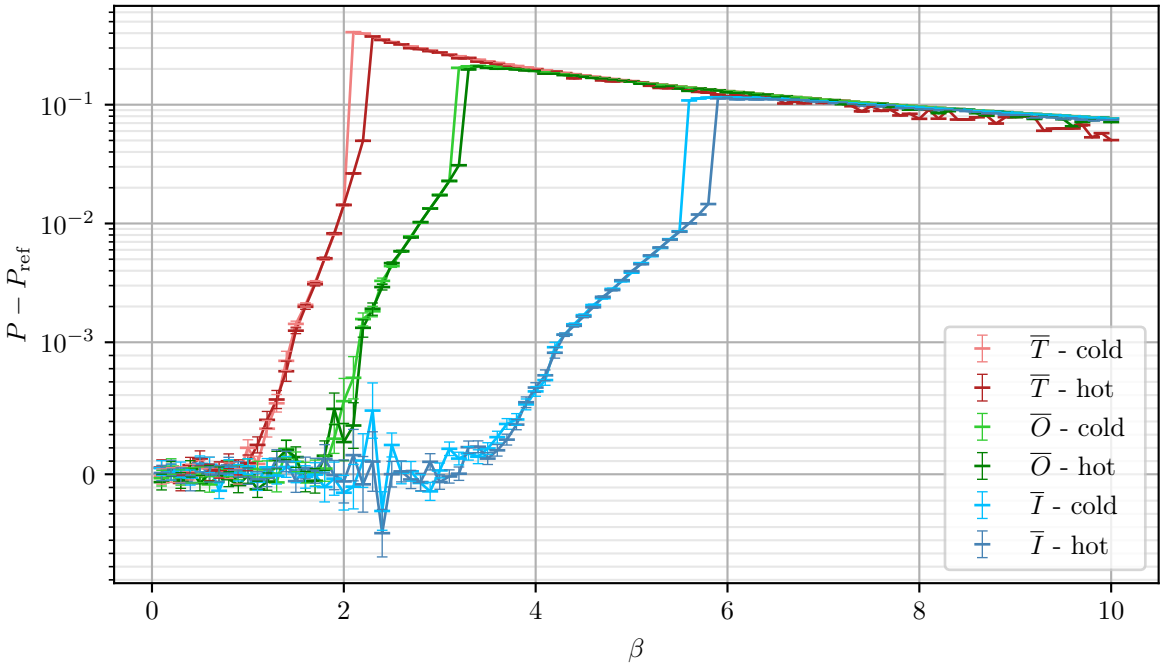


Figure 5: Scan in β for the finite subgroups of SU(2)

minimum value. This would be given by the difference in action of a cold lattice, and one where one of the link variables transitioned to an adjacent vertex. For a given $\beta_{\text{ph.}}$, ΔS then reaches a critical value, such almost all fluctuations will be rejected.

The continuous case does not show this behavior, as for any β configuration changes with a sufficiently small ΔS exist. This therefore always allows for some small fluctuations, decreasing the measured value of P. Thus, all the proposed approximations will only be useful for values where $\beta < \beta_{\text{ph.}}$.

Another interesting feature of figure 5 is that the cold and hot starting configurations form a hysteresis loop around the phase transition. This indicates, that around $\beta_{\text{ph.}}$ multiple semi stable configurations exist. It can be expected that this can be partially fixed, by applying more thermalization iterations. Further, testing on $\overline{I}$ however showed, that these loops do not entirely disappear, even for significantly higher numbers of iterations.

In [10] the exact positions of the phase transitions were found to be $\beta_{\text{ph.}}(\overline{T}) = 2.15 \pm 0.05$, $\beta_{\text{ph.}}(\overline{O}) = 3.25 \pm 0.01$ and $\beta_{\text{ph.}}(\overline{I}) = 6.01 \pm 0.01$. Estimating these transitions, by taking the center of the corresponding hysteresis loop as $\beta_{\text{ph.}}$ and the width of the loop as two standard deviations we can read of $\beta_{\text{ph.}}(\overline{T}) = 2.15 \pm 0.15$, $\beta_{\text{ph.}}(\overline{O}) = 3.2 \pm 0.1$ and $\beta_{\text{ph.}}(\overline{I}) = 5.7 \pm 0.2$. Thus, the values for $\overline{T}$ and $\overline{O}$ agree with [10], while $\beta_{\text{ph.}}(\overline{I})$ was measured slightly lower. The reason for this difference is most likely the limited computing power of the time.

The remaining regular polytopes C_5, C_{16}, C_8 and C_{120} for which the results can be found in figure 6 give a similar picture. As maybe expected due to the small vertex count, C_5, C_{16} and C_8 performed not that well with the phase transition as low as $\beta_{\text{ph.}}(C_5) = 1.2 \pm 0.2$, $\beta_{\text{ph.}}(C_{16}) = 1.15 \pm 0.15$ and $\beta_{\text{ph.}}(C_8) = 1.9 \pm 0.2$. C_{120} with it's 600 vertices however pretty much matches SU(2) for the tested range of β, with only slight systematic deviations towards the higher values of β, and no phase transition found.

Another thing to note here is, that beyond the phase transition the values of the hot and cold starting configuration start to diverge again. This is most likely due to an insufficient amount of

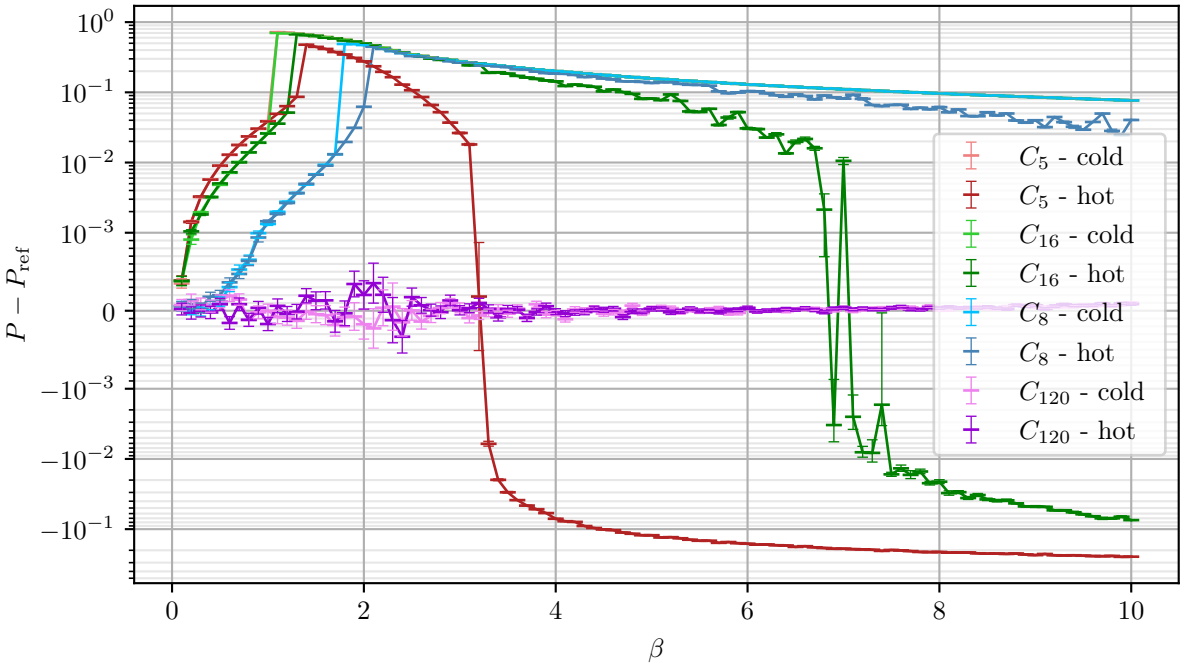


Figure 6: Scan in β for the remaining regular polytopes

thermalization iterations for the hot lattice to *freeze*. As we are however not too concerned with that region anyway and the hot and cold starting configuration agree to the left of the phase transition, this is not a problem.

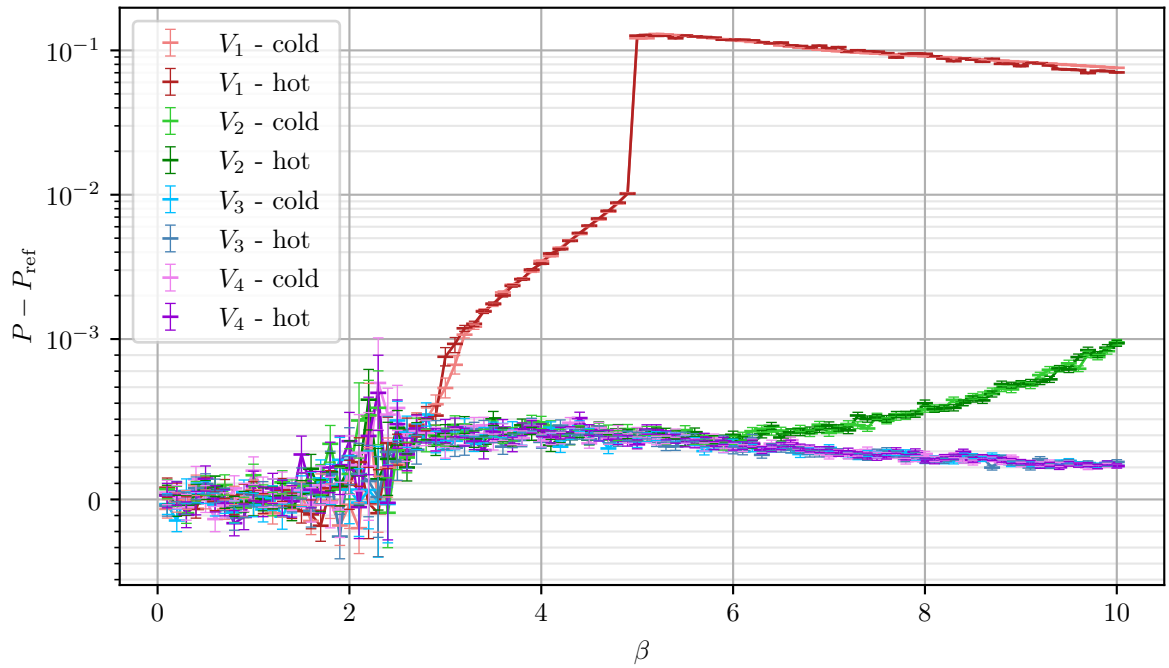


Figure 7: Scan in β for Volleyball lattices

Of the volleyball lattices, only V_1 showed a phase transition at $\beta_{\text{ph.}}(V_1) = 4.95 \pm 0.05$. This makes sense as it only has 80 vertices, while V_2 already has 240, V_3 has 544 and V_4 has 1040. For V_2, V_3 and V_4 the expected systematic deviations show up prominently for $\beta > 2$. For V_3 and V_4 they seem to be consistently around $5 \cdot 10^{-4}$ with a slight decrease towards higher values of β. The increase in deviation for V_2 towards bigger values of β is most likely an indication for an upcoming phase transition.

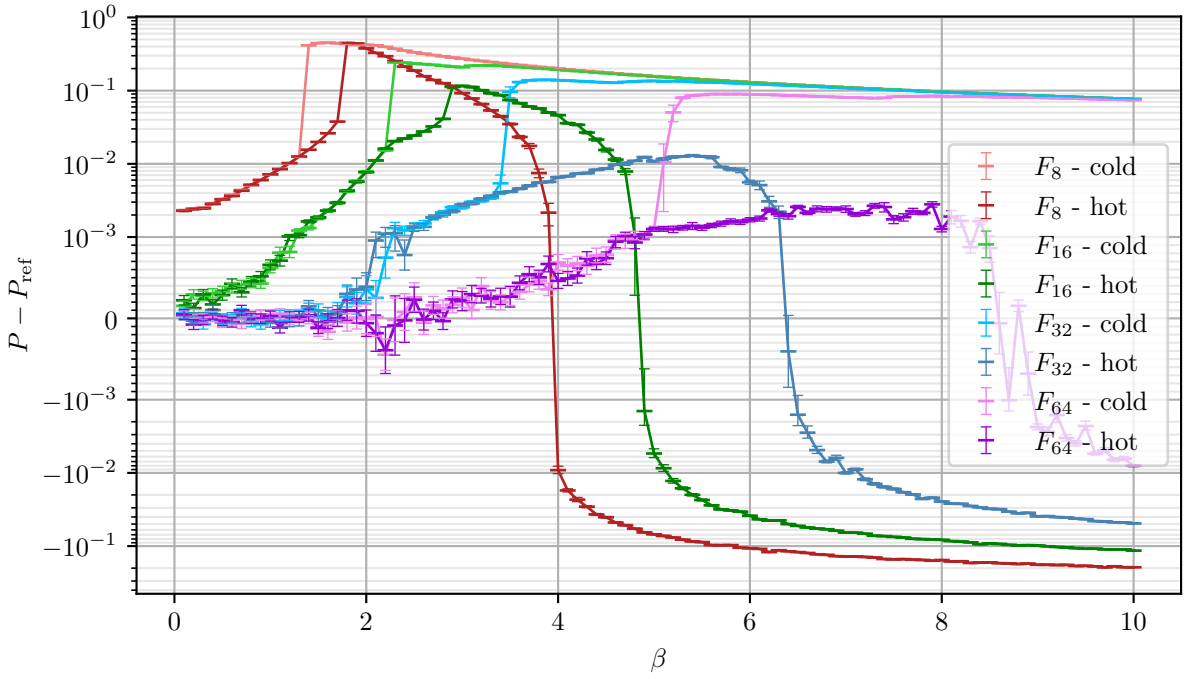


Figure 8: Scan in β for Fibonacci lattices on SU(2) (Part I)

Lastly, we take a look at the Fibonacci lattices. These were tested for lattice sizes 8, 16, 32, 64, 128, 256 and 512. The results can be found in figures 8 and 9. Reading of the positions of the phase transitions in the same way as before is not really possible here, as for lattice sizes bigger than 16 the hot configurations didn't really freeze. This is most likely due to the inefficient generation of new random elements. Nevertheless, we can put a lower bound on $\beta_{\text{ph.}}$ by taking a look at where the cold starting configuration transitions. With this we get $\beta_{\text{ph.}}(F_8) = 1.55 \pm 0.25$, $\beta_{\text{ph.}}(F16) = 2.55 \pm 0.35$, $\beta_{\text{ph.}}(F_{32}) > 3.5$, $\beta_{\text{ph.}}(F_{64}) > 5.2$ and $\beta_{\text{ph.}}(F_{128}) > 7.8$. F_{256} and F_{512} reproduced the reference data set for the whole tested range of β, with only slight systematic deviations for F_{256} towards higher β. This is probably indicating an approaching phase transition beyond the measured range.

Surprisingly the Fibonacci lattices seem to perform significantly better than e.g. the subgroups in terms of phase transitions. F_{32} shows systematic deviations at around $\beta = 2$, similar to $\overline{O}$. The actual phase transition of F_{32}, however, happens for a slightly higher value of β, although it only has two thirds of the vertices. In the same way, $\overline{I}$ is significantly outperformed by F_{128} with only eight vertices more.

To get a better idea of how $\beta_{\text{ph.}}$ scales with the lattice size n for the Fibonacci lattice, we ran another scan for 40 logarithmically spaced lattice sizes between $n = 8$ and $n = 256$. For this β was increased in steps of 0.1 until $P - P_{\text{ref}} > 4 \cdot 10^{-2}$. This was chosen as the criterion for the lattice to be frozen, in which case we moved on to a larger lattice and continued. To

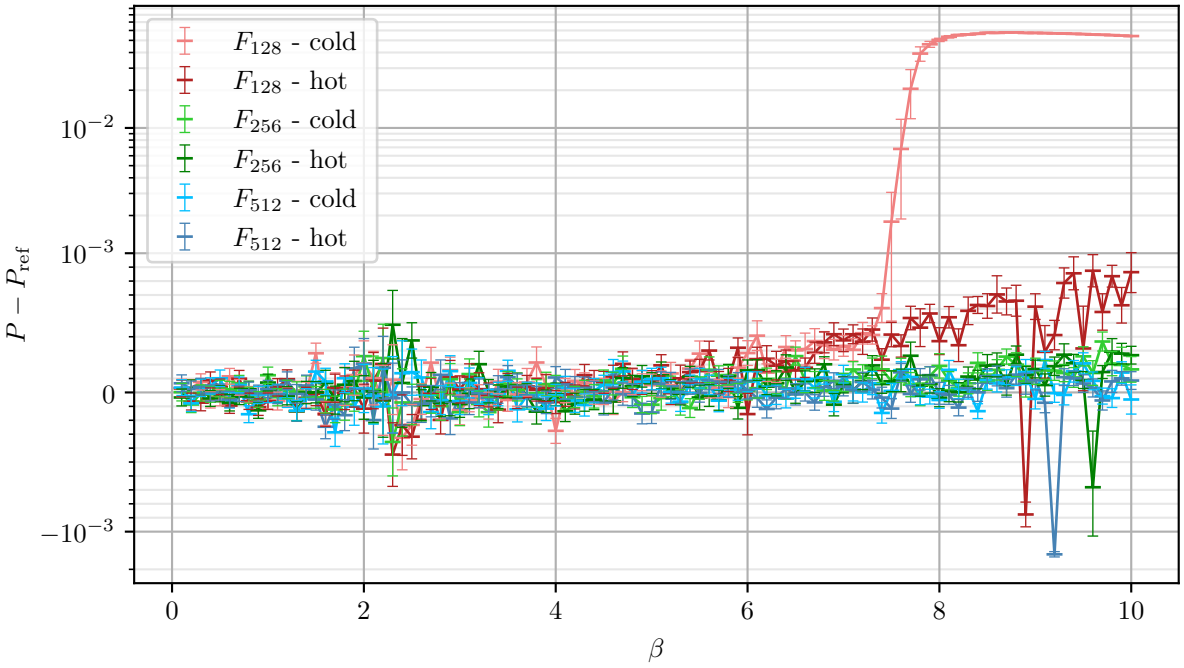


Figure 9: Scan in β for Fibonacci lattices on SU(2) (Part II)

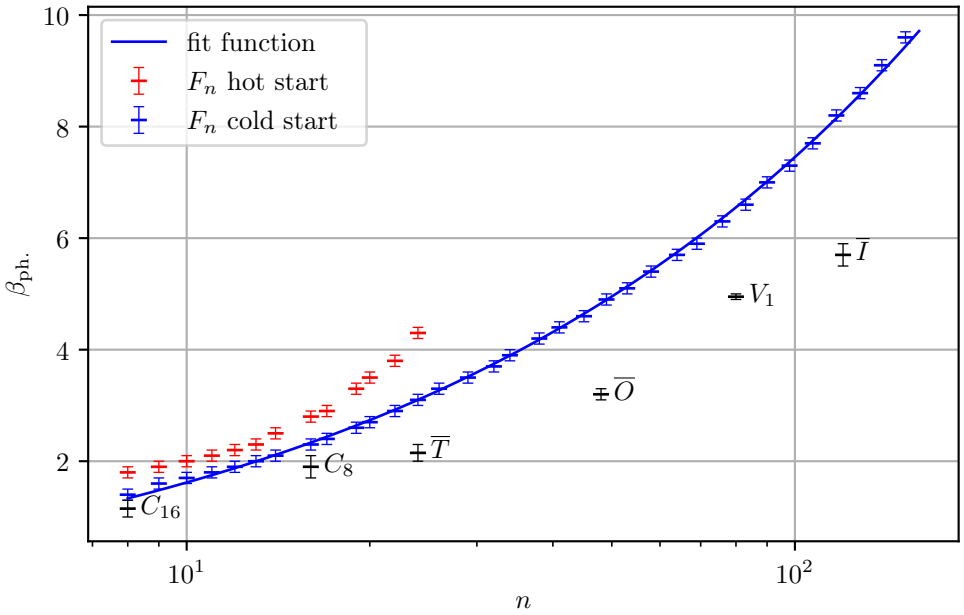


Figure 10: Phase transitions for different sized Fibonacci lattices, compared to the previous approximations of SU(2).

deal with the inefficiencies of the random element generation for the Fibonacci lattice 20,000 thermalization iterations were run, with 1000 measuring iterations.

The results can be found in figure 10. As can be seen by the lack of hot starting configuration data points for $n > 24$, the higher amount of thermalization iterations did not help much there. The results towards the larger lattice sizes however seem to have significantly improved, reading

now $\beta_{\text{ph.}}(F_{64}) > 5.7$ and $\beta_{\text{ph.}}(F_{128}) > 8.6$. It should be noted that the chosen criterion for a frozen lattice might also play a role here. We went with $P - P_{\text{ref}} > 4 \cdot 10^{-2}$ as it seemed to cover all the previously discussed phase transitions reasonably well.

Nevertheless, it becomes obvious that the phase transitions for the Fibonacci lattice can be found at significantly higher values of β compared to any of the other lattices. To get an estimate of the phase transition for a given lattice size, the function

$$\beta_{\text{ph.}}^{(\text{fib.})}(n) = a\sqrt{n} + b \qquad \text{with} \quad a = 0.8532 \pm 0.0037 \qquad \text{and} \quad b = -1.080 \pm 0.027$$

was fitted to the cold start data points. With a $\chi^2_{\text{red.}} = 0.37$ this seems to reproduce the data reasonably well. Whether this holds for lattice sizes outside the tested range might however be questionable.

5.3 Systematic Deviations

Lastly, we took a more detailed look at some systematic deviations that might occur for the different approximations of SU(2). For this we ran another reference simulation for $\beta = 0.1$ and $\beta = 1.0$ with 10^6 iterations, and then tested the approximations with 10^5 iterations. The results can be found in figures 11 and 12. For $\beta = 0.1$ only C_5 and C_{16} as well as F_8 and F_{16} significantly deviate from the continuous case.

For $\beta = 1$ also $\overline{T}$ and C_8 started to show significant deviations. The slightly higher deviation for V_4 turned out to just be a statistical fluctuation, as verified by running the same test for different values of β.

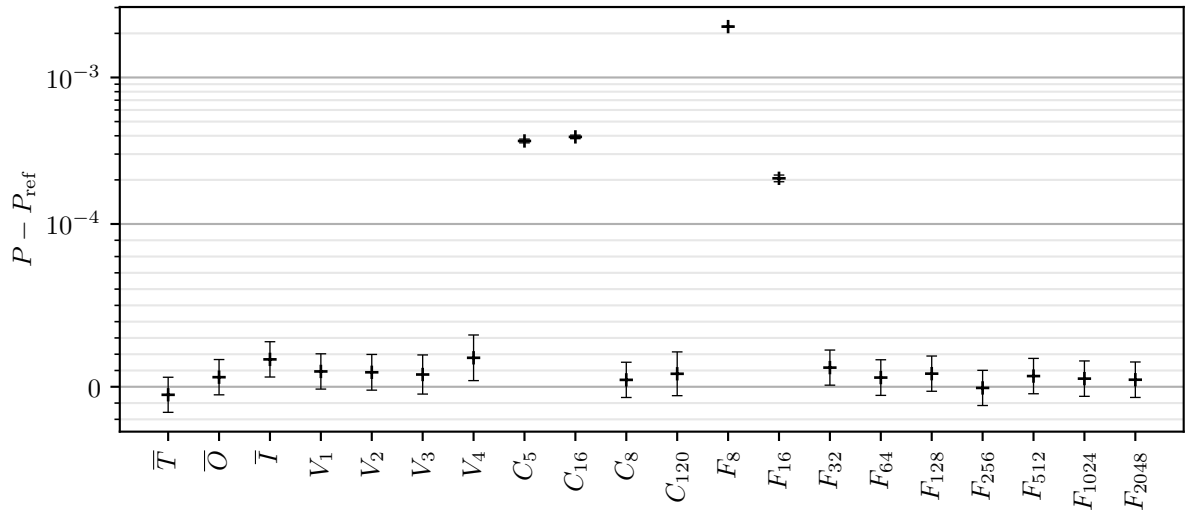


Figure 11: Systematic deviations for $\beta = 0.1$

The deviations for F_8 and F_{16} at $\beta = 0.1$ are quite noteworthy, as contrary to the behavior discussed for phase transitions the eight vertices of C_{16} are a lot closer to the reference data set, while the 16 vertices of C_8 do not even show any significant deviation.

At $\beta = 1$ however, one is already approaching the phase transitions for these lattices. Therefore, here the Fibonacci lattices are closer to the reference data set, for similar vertex count.

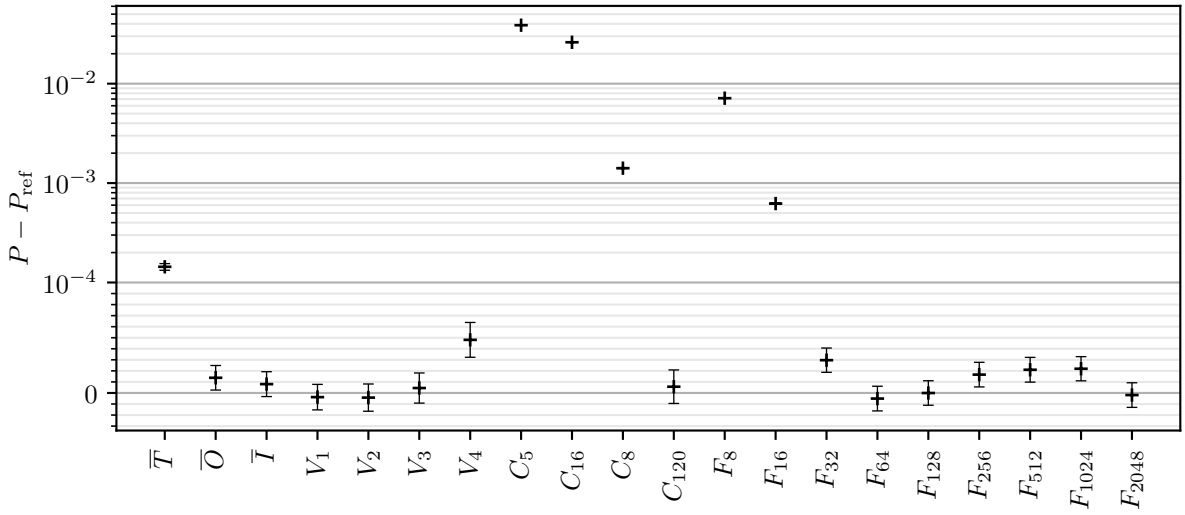


Figure 12: Systematic deviations for $\beta = 1.0$

This can be explained as follows. For very small values of β, the acceptance rate will be very high. This means the behavior of the Monte Carlo algorithm is very close to just picking random elements from the gauge set, and averaging them. As most of the lattices are constructed with some symmetry in mind, they are *balanced* in the sense that

$$\sum_{v \in \{\text{vertices}\}} v = 0$$

holds for them. This however is not the case for the Fibonacci lattice, which is why for small β, such a lack of *balance* shows up as a systematic deviation. With increasing the size of the Fibonacci lattices, these effects, however, decrease.

6 Conclusion

This concludes the testing of the proposed approximations. The clear winners within the tested range of β are the Fibonacci lattices, as their phase transitions were found at significantly higher values of β, than for similar sized lattices generated by subgroups or the regular polytopes. The lack of symmetry of the Fibonacci lattices only really lead to systematic deviations for small lattice sizes, which meant that for very small values of β the subgroups and polytopes where closer to the continuous case.

Thus, subgroups and polytopes are a good choice, whenever one does measurements for $\beta \ll \beta_{\text{ph.}}$, while in any other case one will be better off with a Fibonacci lattice of similar size.

The volleyball lattices turned out to not be that useful, as they show significant systematic deviations due to their lack of a totally uniform distribution. These showed up prominently for $\beta > 2$ for all lattice sizes. The only advantage here is the relatively simple construction, which might or might not help when implemented on quantum hardware.

References

[1] Michael E. Peskin and Daniel V. Schroeder. *An Introduction to quantum field theory*. Reading, USA: Addison-Wesley, 1995. ISBN: 978-0-201-50397-5.

[2] H.J. Rothe. *Lattice Gauge Theories: An Introduction (Third Edition)*. World Scientific Lecture Notes In Physics. World Scientific Publishing Company, 2005. ISBN: 9789813102095. URL: https://books.google.de/books?id=W-47DQAAQBAJ.

[3] J. L. Britton. "Lectures on the Icosahedron and the Solution of Equations of the Fifth Degree By Felix Klein. Translated by G. G. Morrice. Second revised edition, reprinted. Pp. xvi 289. $1.85. 1956. (Dover Publications)". In: *The Mathematical Gazette* 42.340 (1958), pp. 139–140. DOI: 10.2307/3609425.

[4] P. Du Val. *Homographies, Quaternions, and Rotations*. Oxford mathematical monographs. Clarendon Press, 1964. URL: https://books.google.de/books?id=UxW27aOcfZoC.

[5] Magnus J. Wenninger. *Spherical Models*. Dover Publications, 1999. ISBN: 9780486409214.

[6] K. Sadri (https://math.stackexchange.com/users/388484/k-sadri). *Can the Fibonacci lattice be extended to dimensions higher than 3?* Mathematics Stack Exchange. URL:https://math.stackexchange.com/q/3297830 (version: 2021-05-24). eprint: https://math.stackexchange.com/q/3297830. URL: https://math.stackexchange.com/q/3297830.

[7] Nicholas Metropolis et al. "Equation of State Calculations by Fast Computing Machines". In: *The Journal of Chemical Physics* 21.6 (1953), pp. 1087–1092. DOI: 10.1063/1.1699114. eprint: https://doi.org/10.1063/1.1699114. URL: https://doi.org/10.1063/1.1699114.

[8] M Creutz and B Freedman. "A statistical approach to quantum mechanics". In: *Annals of Physics* 132.2 (1981), pp. 427–462. ISSN: 0003-4916. DOI: https://doi.org/10.1016/0003-4916(81)90074-9. URL: https://www.sciencedirect.com/science/article/pii/0003491681900749.

[9] Yoshihiro Tashiro. "On methods for generating uniform random points on the surface of a sphere". In: *Annals of the Institute of Statistical Mathematics* 29.1 (Dec. 1977), pp. 295–300. ISSN: 1572-9052. DOI: 10.1007/BF02532791. URL: https://doi.org/10.1007/BF02532791.

[10] D. Petcher and D. H. Weingarten. "Monte Carlo Calculations and a Model of the Phase Structure for Gauge Theories on Discrete Subgroups of SU(2)". In: *Phys. Rev. D* 22 (1980), p. 2465. DOI: 10.1103/PhysRevD.22.2465.

[11] A. Denbleyker et al. "Series expansions of the density of states in SU(2) lattice gauge theory". In: *Physical Review D* 78 (Aug. 2008). DOI: 10.1103/PhysRevD.78.054503.