Markov chain Monte Carlo methods for graph refinement in spinfoam cosmology

Pietropaolo Frisoni\textsuperscript{1,*}, Francesco Gozzini\textsuperscript{2} and Francesca Vidotto\textsuperscript{3}

\textsuperscript{1} Department of Physics & Astronomy, Western University, London, ON N6A 3K7, Canada
\textsuperscript{2} Institut für Theoretische Physik, Philosophenweg 16, 69120 Heidelberg, Germany
\textsuperscript{3} Department of Physics & Astronomy, Department of Philosophy, and Rotman Institute, Western University, London, ON N6A 3K7, Canada

E-mail: pfrisoni@uwo.ca

Received 17 October 2022; revised 13 February 2023
Accepted for publication 20 March 2023
Published 14 April 2023

Abstract
We study the behavior of the Lorentzian Engle-Pereira-Rovelli-Livine spin
foam amplitude with homogeneous boundary data, under a graph refinement
going from five to twenty boundary tetrahedra. This can be interpreted as a
wave function of the Universe, for which we compute boundary geometrical
operators, correlation functions, and entanglement entropy. The numerical
calculation is made possible by adapting the Metropolis-Hastings algorithm,
along with recently developed computational methods appropriate for the deep
quantum regime. We confirm that the transition amplitudes are stable against
such refinement. We find that the average boundary geometry does not change,
but the new degrees of freedom correct the quantum fluctuations of the bound-
ary and the correlations between spatial patches. The expectation values are
compatible with their geometrical interpretation and the correlations between
neighboring patches decay when computed across different spinfoam vertices.

Keywords: graph, refinements, spinfoam, cosmology, Monte Carlo,
Loop Quantum Gravity, spin foam

(Some figures may appear in colour only in the online journal)

\textsuperscript{*} Author to whom any correspondence should be addressed.

Original Content from this work may be used under the terms of the Creative Commons Attribution
4.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the
title of the work, journal citation and DOI.
1. Introduction

The last years have seen a lively development of numerical methods in the covariant, or spin-foam, formulation of loop quantum gravity (LQG) [1]. A key step in this direction has been the introduction of Markov chain Monte Carlo (MCMC) methods and their application to the study of the Engle-Pereira-Rovelli-Livine (EPRL) propagator [2]. In [2] the authors exploited the known properties of the single-vertex semiclassical limit to perform a Monte Carlo sampling over particular subspaces of the complexified parameter space, using semiclassical boundary states [3]. Exploiting the stationary phase approximation [4], the authors found a good agreement with the semiclassical results obtained via analytical methods.

The evaluation of spinfoam amplitudes in the EPRL model [5–8], the introduction of effective models [9, 10] and the numerical study of cuboid renormalization [11] have shed considerable light on several aspects of the theory, such as the role of the Immirzi parameter, the accidental flatness constraints, and the refinement limit.

This paper introduces a technique that can be applied to some calculations in the regime where the number of degrees of freedom is large, but the relevant spin quantum numbers are small. We combine the Metropolis-Hastings algorithm [12] with some recently developed high performance computing techniques in covariant LQG [13]. Not based on analytical approximations, this method requires minimal knowledge of spinfoam geometry. On the other hand, the algorithm becomes source-demanding as the complexity of the spinfoam increases. We test the algorithm in the case of a single 4-simplex, and then we use it to study a spinfoam with six vertices in the bulk and 20 nodes on the boundary. This corresponds to the cellular decomposition obtained from one elementary 4-simplex by splitting each of the five boundary tetrahedra into four tetrahedra. The resulting spinfoam is a refinement of the 4-simplex vertex which does not add any internal (dynamical) face to the spinfoam two-complex. In the following, we refer to it as the ‘star’ spinfoam.

We restrict the calculation to the homogeneous sector where the spins of all boundary links have the same value. In this sector, the spinfoam degrees of freedom are given by the boundary intertwiners, which encode the shapes of the boundary tetrahedra. We compute the amplitude as a function of these variables numerically and use the Monte Carlo sampling to study expectation values of different boundary operators and their correlations. For the sake of completeness, we investigate both the BF and EPRL models.

Spinfoam amplitudes with a regular graph on the boundary and homogeneous data can be interpreted as cosmological states [14]. In particular, when a single boundary state is considered, the amplitude can be seen as a transition from nothing into a three-dimensional geometry, compatible with the four-dimensional Lorentzian bulk [15, 16]. More precisely, the computed amplitude is a truncation of the spinfoam vertex expansion of the nothing-to-geometry transition amplitude. This provides a spinfoam Lorentzian version of the Hartle-Hawking wave function of the Universe [17]. We refer to the literature in spinfoam cosmology for the physical interpretation of these states [18–32].

These states are rich enough to describe a boundary geometry that is regular on average but allows quantum fluctuations. The recent advance in numerical methods applied to the computation of spinfoam amplitudes has opened the possibility to compute concrete observables with spinfoam cosmological states. The first computation of this kind was introduced in [33] using a single 4-simplex. In this paper, we would like to focus on numerical methods. The cosmological spinfoam states provide an interesting framework to investigate novel techniques.

A key open question in the spinfoam approach to quantum gravity is the convergence of the amplitudes under refinement of the two complexes of the spinfoam. In this regard, we are able to confirm numerically that the refinement studied is stable for the boundary observable, in
the sense that there is excellent agreement between expectation values computed on the single vertex graph and on the refined graph. The correlations turn out to be different, as well as the quantum information entropy between different boundary nodes, reflecting the finer scale at which they become accessible. The results provide a quantitative estimate of the quantum correlations between different spatial patches in the manifold boundary whose truncation is represented by the boundary spin network.

The paper is organized as follows. In section 2 we define the boundary state that we use in our analysis. In section 3 we discuss the MCMC method applied to the spin sums over the boundary degrees of freedom. In section 4 we test the Monte Carlo sampler to the case of single 4-simplex, where calculations can be performed using deterministic approaches. In section 5 we study the star spinfoam, investigating the numerical results of geometrical operators (boundary angles and volumes) and the related correlations. We also discuss the entanglement entropy between boundary nodes, considering different partitions into subsystems.

The code used for all the computations described in this paper is available on GitHub [34]. All the computations were performed on Compute Canada’s Cedar, Graham, Beluga and Narval clusters (www.computecanada.ca). The computational resources employed for this paper can be quantified as approximately $8 \times 10^4$ CPU hours.

2. The boundary state

We study the boundary state $|\psi_0\rangle$ introduced in [33], which studied the simplest triangulation of a 3-sphere, emerging from a single 4-simplex. In general, let $\Gamma$ be a graph with $L$ links and $N$ nodes. The LQG Hilbert space for the graph is:

$$H_\Gamma = L^2_{SU(2)}/SU(2)^N.$$  \hfill (1)

The spin network basis in $H_\Gamma$ is made by the states $|{j_l}, \{i_n\}\rangle$ (from now on, we omit the $\Gamma$ subscript), where $\{j_l\}$ is a set of half-integer spins and $\{i_n\}$ an intertwiner set, $n = 1 \ldots N$, $l = 1 \ldots L$. An intertwiner $i_n$ is a basis element of the invariant subspace of the tensor product of $4\ SU(2)$ representations at the node $n$. In the following, we fix all the spins to be equal, namely $j_l = j$. We denote a boundary spin network state of this reduced space as:

$$|j, \{i_n\}\rangle := |j, i_1\rangle \otimes \ldots \otimes |j, i_N\rangle,$$  \hfill (2)

suppressing the curly brackets for the spin label $j$, as there is one common spin attached to all the links. We define the state $|\psi_0\rangle$ in the Hilbert space (1) by

$$\langle j, \{i_n\}|\psi_0\rangle = A(j, \{i_n\}),$$  \hfill (3)

where $A(j, \{i_n\})$ is the LQG amplitude of the state in the spin network basis. The amplitude (3) can be interpreted as the amplitude associated with the transition nothing-to-$|j, \{i_n\}\rangle$. Hence $|\psi_0\rangle$ gives the natural state that is projected out of the empty state by the LQG dynamics. The amplitude function depends on the common spin $j$ on the links and on all the $N$ intertwiner indices. The state $|\psi_0\rangle$ is therefore defined as:

$$|\psi_0\rangle = \sum_{\{i_n\}} A(j, \{i_n\}) |j, \{i_n\}\rangle.$$  \hfill (4)

The sum is over all possible values of all the intertwiners in the set $\{i_n\}$, compatible with triangular inequalities. If $j_l = j$ then every intertwiner $i_n$ can assume integer values between 0 and

---

4 The code works on any operating system with an updated version of the Julia programming language (the code was tested with Julia 1.6.2).
This gives a total of \((2j + 1)^N\) boundary basis elements that enter the sum (4). Following the geometrical interpretation of the covariant LQG phase space in terms of twisted geometries we might interpret the constraint \(j_i = j\) as imposing strongly at the quantum level that all the areas of the faces of the boundary tetrahedra must be equal. The intertwiner degrees of freedom model the ‘shape’ of the boundary tetrahedra, and these are relational observables at a given value \(j\). They are directly linked to the boundary 3d dihedral angles, as discussed in section 5.2.1. The definition (4) does not depend on the details of the triangulation. In fact, the triangulation determines how the amplitude \(A(j, \{i_n\})\) must be computed.

### 2.1. Expectation values

We consider local geometrical operators acting on single boundary nodes of \(H\). For each operator, we specify the matrix elements in the basis states (5). We start defining the normalized expectation value on the boundary state (4) of an operator \(O_k\), acting on the Hilbert space associated to the \(k\)th node, as:

\[
\langle O_k \rangle \equiv \frac{1}{Z} \langle \psi_0 | O_k | \psi_0 \rangle .
\]  

The normalization factor is computed as:

\[
Z \equiv \langle \psi_0 | \psi_0 \rangle = \sum_{\{i_n\}} A(j, \{i_n\})^2 .
\]  

From (4) we write:

\[
\langle \psi_0 | O_k | \psi_0 \rangle = \frac{1}{Z} \sum_{\{i_n\}} \sum_{\{i'_n\}} A(j, \{i_n\}) A(j, \{i'_n\}) \langle j, \{i'_n\} | O_k | j, \{i_n\} \rangle .
\]  

By using the orthogonality of the spin-network states (2) we find:

\[
\langle j, \{i'_n\} | O_k | j, \{i_n\} \rangle = \delta_{i'_1, i_1} \cdots \langle j, \{i'_n\} | O_k | j, \{i_n\} \rangle \delta_{i'_m, i_m} ,
\]  

therefore we conclude:

\[
\langle O_k \rangle = \frac{1}{Z} \sum_{\{i_n\}} \sum_{i'_n = 0}^{2i} A(j, \{i_n\}) A(j, \{i'_n\}) \langle j, \{i'_n\} | O_k | j, i_k \rangle ,
\]  

where \(A(j, \{i_n\}, \{i'_n\})\) is defined as:

\[
A(j, \{i_n\}, \{i'_n\}) \equiv A(j, i_1, \ldots, i'_n, \ldots, i_m) ,
\]  

namely, the amplitude computed with \(i'_n\) in place of \(i_k\). Since \(i_k \in \{i_n\}\), the sum over \(i_k\) is contained in the sum over the set \(\{i_n\}\). It is now straightforward to compute \(\langle O_k O_m \rangle\), which turns out to be:

\[
\langle O_k O_m \rangle = \frac{1}{Z} \sum_{\{i_n\}} \sum_{i'_n = 0}^{2i} A(j, \{i_n\}) A(j, \{i'_n\}) \langle j, \{i'_n\} | O_k | j, i_k \rangle \langle j, i'_m | O_m | j, i_m \rangle ,
\]  

where the meaning of \(A(j, \{i_n\}, \{i'_n\}, \{i'_m\})\) is transparent by looking at (10). That is, we refer to the amplitude with \(i'_n\) instead of \(i_k\) and \(i'_m\) in place of \(i_m\). In the case of diagonal operators \(D_k\) in the spin-network basis, equations (9) and (11) become respectively:

\[
\langle D_k \rangle = \frac{1}{Z} \sum_{\{i_n\}} A^2(j, \{i_n\}) \langle j, i_k | D_k | j, i_k \rangle ,
\]  

where the meaning of \(A(j, \{i_n\}, \{i'_n\}, \{i'_m\})\) is transparent by looking at (10). That is, we refer to the amplitude with \(i'_n\) instead of \(i_k\) and \(i'_m\) in place of \(i_m\). In the case of diagonal operators \(D_k\) in the spin-network basis, equations (9) and (11) become respectively:
\[ \langle D_k D_m \rangle = \frac{1}{Z} \sum_{\{i_n\}} A^2 (j, \{i_n\}) \langle j, i_n | D_k | j, i_k \rangle \langle j, i_m | D_m | j, i_m \rangle . \]  

Normalized correlations are defined as:

\[ C(O_k, O_m) = \frac{\langle O_k O_m \rangle - \langle O_k \rangle \langle O_m \rangle}{\langle \Delta O_k \rangle \langle \Delta O_m \rangle} , \]  

where the quantum spread is:

\[ \Delta O_k = \sqrt{\langle O_k^2 \rangle - \langle O_k \rangle^2} . \]  

The fact that the connected correlation function (14) between the nodes \( k \) and \( m \) is non-vanishing turns out to be a necessary condition in order to have correlated fluctuations between the shapes of the tetrahedra dual to nodes \( k \) and \( m \) [39–43].

3. Monte Carlo over intertwiner space

As discussed in section 2.1, the expectation value of an operator requires to sum over all possible eigenstates of the quantum system. Numerically, this rapidly becomes intractable as the number of degrees of freedom increases. In the present context, for a graph with \( N \) boundary intertwiners there are \((2j + 1)^N\) values to compute and to sum. Suppose that the amplitude function \( A(j, \{i_n\}) \) can be computed in \( 10^{-6} \) s on a reference hardware (the real time is typically orders of magnitude larger). For 20 boundary tetrahedra, which is the case of the star spinfoam discussed in section 5, a spin \( j = 2 \) computation would take three years. Obviously, we cannot use blind summation if we want to approach this problem numerically. Clearly even parallelizing the computation on multiple machines cannot solve this issue in the case of many boundary degrees of freedom.

A solution is Monte Carlo summation. This is a technique that is used to compute expectation values of random variables. We obtained the best results by adapting the Metropolis-Hastings algorithm [12, 44] to the discrete sums over the boundary intertwiners. For the sake of completeness, we briefly describe the Metropolis-Hastings algorithm\(^5\) in its general form.

3.1. Metropolis-Hastings algorithm

Let’s consider a quantity \( O \) which can be computed as:

\[ O = \sum_{x \in \chi} \tilde{f}_\chi (x) o(x) , \]  

where \( x \) is a (possibly multidimensional) discrete variable on a state space \( \chi \) which must be summed over, while \( \tilde{f}_\chi \) is a probability distribution function on \( \chi \), so that:

\[ \sum_{x \in \chi} \tilde{f}_\chi (x) = 1 . \]  

From now on, we define \( \tilde{f}_\chi \) as the target distribution and we omit the state space label \( \chi \). Since the target distribution is normalized, we write:

\[ \tilde{f}(x) = f(x) / \sum_x f(x) , \]  

\(^5\) We refer to the original paper [12] or to the numerous texts available for a deeper description of the algorithm.
from which:
\[
O = \frac{\sum f(x) o(x)}{\sum f(x)} .
\]  
(19)

If the target distribution (18) can be computed up to a multiplying constant, the Metropolis-Hastings algorithm allows constructing on state space \( \chi \) an ergodic Markov chain with length \( N_{MC} \): \( x_1, x_2, x_3, \ldots, x_n, \ldots, x_{N_{MC}} \) such that \( x_n \) is converging (in distribution) to \( \tilde{f} \), exploring the space \( \chi \) progressively. If we define:
\[
O_{N_{MC}} = \frac{1}{N_{MC}} \sum_{n=1}^{N_{MC}} o(x_n) ,
\]  
(20)
then, since the chain can be considered as a statistical sample, the law of large numbers ensures that:
\[
\lim_{N_{MC} \to \infty} O_{N_{MC}} = O .
\]  
(21)

The computation is stochastic in nature and the correct result is found only in the limit of an infinite number of samples. This allows us to write:
\[
O_{N_{MC}} \approx O \quad \text{for} \quad N_{MC} \gg 1 .
\]  
(22)
That is, we obtain an estimate of the original sum (16). The soundness of the procedure comes from known theorems on Monte Carlo summation and we can estimate the error done by comparing many different runs. Since the simulation is Markovian and the chain itself can be considered as a statistical sample, the latter usually depends on the starting value. In the following, we choose randomly the starting point of the Markov chain. The initial steps, while the chain is in the thermalization phase, are typically removed as burn-in iterations. In order to transit from the chain state \( x_n \) to \( x_{n+1} \), we require a proposal distribution \( g \) defined on space \( \chi \). If \( g \) is positive everywhere, then the Metropolis-Hastings algorithm preserves \( \tilde{f} \) as the stationary distribution to which the chain is progressively converging. In the random walk Metropolis-Hastings, the proposal distribution consists of a local exploration of the neighborhood of the current value \( x_n \) of the Markov chain. That is, the proposed value \( x'_n \) is simulated as:
\[
x'_n = x_n + \delta x_n ,
\]  
(23)
where \( \delta x_n \) is a random perturbation with distribution \( g \). That is, the proposed state \( x'_n \) is sampled from a probability distribution \( g(x'_n|x_n) \), which suggests a candidate given the previous sample value \( x_n \). As proposal distribution, we choose a truncated normal distribution rounded to integers centered around \( x_n \) with standard deviation \( \sigma \):
\[
g(x'_n|x_n) = \mathcal{N}_{d,t}(x_n, a, b; \sigma) ,
\]  
(24)
where the definition of \( \mathcal{N}_{d,t} \) is reported in appendix A. The full algorithm is summarized in the flowchart 1, in which we report the steps in order to implement the random walk Metropolis-Hastings and build the Markov chain. The multiplicity factors and the storage of \( f(x)'s \) have been introduced just as a matter of efficiency. In fact, this considerably speeds up the algorithm and the consequent computation of operators. Technically, the Markov chain obtained at the end of the algorithm 1 has a length less than \( N_{MC} \) (as this depends on the acceptance ratio). Since it is sufficient to take into account the multiplicity of each single chain state in order to restore the original length, in the following we refer to the Markov chain obtained at the end of the algorithm 1 as having length \( N_{MC} \) without losing any generality.
Algorithm 1. Random walk Metropolis-Hastings.

1: Choose the number of iterations $N_{MC}$, the burn-in parameter $b$ and the standard deviation $\sigma$ of the proposal distribution $g$.
2: Set a random initial configuration $x$ and compute $f(x)$
3: Set initial multiplicity to 1
4: for $n = 1 \ldots N_{MC}$ do
5: Generate a candidate $x'$ from $x$ according to the proposal distribution $g$
6: if $x' = x$ then
7: Increase the multiplicity by 1
8: continue
9: else
10: Compute $f(x')$
11: Compute $p = \min\left\{ 1, \frac{f(x')}{f(x)} \frac{g(x|x')}{g(x'|x)} \right\}$
12: Generate a uniform random number $r$ between 0 and 1
13: if $r < p$ then
14: if $n > b$ then
15: Store $x$ and $f(x)$ with the corresponding multiplicity
16: Set $x = x'$, $f(x) = f(x')$
17: Set the multiplicity to 1
18: else
19: Increase the multiplicity by 1
20: Dump to disk the $x$’s, $f(x)$’s and the corresponding multiplicities.

Since $x_n$ depends on the previous element along the Markov chain, this induces a non-zero correlation between $x_n$ and $x_{n+d}$. The correlation between $x_n$ and $x_{n+d}$ is defined as the autocorrelation at lag $d$. For a Markov chain that converges to a stationary distribution, the autocorrelation should indeed decrease as the lag increases. Although the most common approach is to evaluate the autocorrelation of operators, a measure of the degree of autocorrelation of the Markov chain is represented by the autocorrelation of the sequence of the amplitudes of the states. If we define the average of the states’ amplitude as:

$$\bar{f}(x) = \frac{1}{N_{MC}} \sum_{n=1}^{N_{MC}} f(x_n) ,$$

the definition of the autocorrelation function at lag $d$ associated with the sequence $f(x_1) \ldots f(x_{N_{MC}})$ can be written as:

$$\text{ACF}(x_1 \ldots x_{N_{MC}}; d) = \frac{\sum_{n=1}^{N_{MC}-d} (f(x_n) - \bar{f}(x)) (f(x_{n+d}) - \bar{f}(x))}{\sum_{n=1}^{N_{MC}} (f(x_n) - \bar{f}(x))^2} .$$

In order to obtain an unbiased estimate of the statistical fluctuations due to the Monte Carlo sampling, we can compare the results of operators’ expectation values over different runs. That is, we store multiple Markov chains according to algorithm 1, computing operators for each one of them. This is extremely useful in determining the convergence of the Markov Chain to the stationary distribution and the corresponding unbiased statistical dispersion of the operators. This is discussed in section 3.2.
3.2. Expectation values with Monte Carlo

In order to apply the Metropolis-Hastings algorithm discussed in section 3.1 to the computation of spinfoam observables of section 2.1, a direct comparison between equations (12) and (16) is enlightening. Namely, if we associate to the state space $\chi$ the intertwiners’ boundary space (2), so that $x = \{\iota_a\}$, then the target distribution becomes:

$$
\tilde{f}(x) = \frac{A^2(j, \{\iota_a\})}{\sum_{\{\iota_a\}} A^2(j, \{\iota_a\})}.
$$

Namely, we run a stochastic sampling routine that extracts draws of intertwiners $[i_1 \ldots i_N]$ from their whole configuration space, according to the Markov chain. The proposal distribution corresponds to a discrete multivariate truncated normal distribution and each intertwiner is proposed sampling from an independent one-dimensional distribution. The $a$, $b$ parameters in (24) for each intertwiner are 0 and 2, respectively. After storing the intertwiner draws we can use them to compute the expectation values of operators. We can summarize the introduction of the Monte Carlo with the following substitution in the formulas of the expectation values of the operators:

$$
\sum_{\{\iota_a\}} A^2(j, \{\iota_a\}) o(\{\iota_a\}) \approx \sum_{\{\iota\}} o(\{\iota\}).
$$

In the right side of equation (28), the sum over the intertwiners is intended as the sum over the stored draws $[\iota_a] \equiv [i_1 \ldots i_N]$ in which the intertwiners have a fixed value compatible with triangular inequalities. That is, we are no longer considering all the independent summations over the intertwiners. This hugely reduces the computational cost, making the computation feasible in the case of many boundary degrees of freedom. With (28), the normalization factor (6) becomes:

$$
N \approx \sum_{\{\iota\}} = \text{Number of MC iterations} \equiv N_{MC}.
$$

We can easily find the expression for the expectation values of non-diagonal operators by multiplying and dividing for $A(j, \{\iota_a\})$ and then using (28), remembering (29). In fact, (9) becomes:

$$
\langle O_k \rangle \approx \frac{1}{N_{MC}} \sum_{\{\iota\}} \sum_{i_j}^{2j} \frac{A(j, [\iota], i'_j)}{A(j, \{\iota\})} \langle j, i'_j | O_k | j, i_j \rangle.
$$

Equation (11) becomes:

$$
\langle O_k O_m \rangle \approx \frac{1}{N_{MC}} \sum_{\{\iota\}} \sum_{i_j=0}^{2j} \sum_{i'_j=0}^{2j} \frac{A(j, [\iota], i'_j, i'_m)}{A(j, \{\iota\})} \langle j, i'_j | O_k | j, i_j \rangle \langle j, i'_m | O_m | j, i_m \rangle.
$$

In case of diagonal operators, from (12) and (13) we obtain:

$$
\langle D_k \rangle \approx \frac{1}{N_{MC}} \sum_{\{\iota\}} \langle j, i_k | D_k | j, i_k \rangle,
$$

$$
\langle D_k D_m \rangle \approx \frac{1}{N_{MC}} \sum_{\{\iota\}} \langle j, i_k | D_k | j, i_k \rangle \cdot \langle j, i_m | D_m | j, i_m \rangle.
$$

Notice that in the case of diagonal operator, it is not necessary to compute any amplitude except those necessary for sampling the draws of intertwiners. This makes the computation
of diagonal operators several orders of magnitude faster than non-diagonal ones. We consider the numerical analysis up to the value $j = 6$ for the spins associated with the boundary links. In fact, this numerical approach is intended to be applied in the full quantum regime, i.e. when the spin quantum numbers are small and the semiclassical approximation is not valid. In addition, the computational complexity represented by increasing $j$ strongly depends on the type of the considered operator, as well as on the Metropolis-Hastings parameters. The value $j = 6$ allows computing all the operators and correlation functions that we consider with a stable precision of up to three significant digits.

Crucially, notice that this approach is not based on analytical approximations and requires minimal knowledge of spinfoam geometry, which typically becomes quite complicated for models beyond the single 4-simplex. On the other hand, the price to pay is the calculation of the spinfoam amplitude at each iteration of the algorithm 1. Much effort has been devoted in recent times to the efficient computation of spinfoam amplitudes [8, 13, 45]. At present, the best numerical framework to compute BF and EPRL vertex amplitudes, which can be seen as the elementary building blocks of more general triangulations, is the sl2cfoam-next library [13].

The calculation of the spinfoam amplitude with high performance computing techniques is the fundamental ingredient that allows applying the Metropolis-Hastings algorithm discussed in 3.1. The most recent developments have made it possible to compute potentially divergent spinfoam amplitudes with many internal faces [7, 46]. This makes the algorithm presented in this paper a good candidate to be used in the case of spinfoams with a non-trivial dynamic structure.

After we have stored a number $C$ of Markov chains, each with the same length $N_{MC}$ and Metropolis-Hastings parameters, we can compute the expectation value (30) of an operator $O_k$ for each chain $\langle O_k \rangle_1 \ldots \langle O_k \rangle_C$ and then consider the corresponding average and standard deviation:

$$\mu_{\langle O_k \rangle} = \frac{\sum_{c=1}^{C} \langle O_k \rangle_c}{C},$$
$$\sigma_{\langle O_k \rangle} = \sqrt{\frac{\sum_{c=1}^{C} (\mu_{\langle O_k \rangle} - \langle O_k \rangle_c)^2}{C}}.$$

For each considered operator, we plot the corresponding Gaussian distribution:

$$G(x; \mu_{\langle O_k \rangle}, \sigma_{\langle O_k \rangle}) = \frac{1}{\sigma_{\langle O_k \rangle} \sqrt{2\pi}} e^{-\frac{1}{2} \left( \frac{x - \mu_{\langle O_k \rangle}}{\sigma_{\langle O_k \rangle}} \right)^2}.$$

The number of Monte Carlo iterations $N_{MC}$ for each Markov chain, the number $C$ of averaged chains, and the relevant parameters in the Metropolis-Hastings algorithm are all listed in tables in the appendix B.

### 4. The 4-simplex

The 4-simplex, or (referring to the dual triangulation) the vertex, is the simplest geometrical triangulation of the 3-sphere. It is formed by five tetrahedra glued on ten faces. There are five boundary degrees of freedom and there are no internal faces. As show in figure 1, there is a complete self-duality between the geometrical triangulation of the 4-simplex and the corresponding boundary graph.
The geometry in figure 1 does not depend on the Lorentzian character of the triangulation or not. Namely, it is the same for the $SU(2)$ topological BF or the Lorentzian EPRL 4-simplex. In order to emphasize the difference between the two models we need to explicitly represent the spinfoam associated with the amplitude. We denote such amplitudes associated with a single vertex as $V_{\text{BF}}$ and $V_{\text{EPRL}}$. We write the corresponding expression in the general form, in which all spins have different values. As discussed in section 2, in the present context we focus on the case in which all spins are equal.

### 4.1. The BF and EPRL vertex amplitudes

The vertex amplitude of the topological BF model can be written as an $SU(2)$ invariant Wigner $\{15j\}$ symbol. The choice of the recoupling basis on each intertwiner determines if the symbol can be reduced to the product of lower-dimensional symbols. We choose the irreducible symmetric $\{15j\}$ symbol of first kind, following the convention of [47]. The definition in terms of $\{6j\}$ symbols is:

$$V_{\text{BF}}(j_{ab}; i_a) = (-1)^{\sum_{k=1}^3 (j_k + i_k)} \sum_i (2s + 1) \left\{ \begin{array}{ccc} i_1 & j_{25} & s \\ i_5 & j_{14} & j_{15} \end{array} \right\} \left\{ \begin{array}{ccc} j_{14} & i_5 & s \\ j_{35} & i_4 & j_{45} \end{array} \right\} \left\{ \begin{array}{ccc} i_4 & j_{35} & s \\ i_3 & j_{24} & j_{34} \end{array} \right\} \times \left\{ \begin{array}{ccc} j_{24} & i_3 & s \\ j_{13} & i_2 & j_{23} \end{array} \right\} \left\{ \begin{array}{ccc} i_2 & j_{13} & s \\ i_1 & j_{25} & j_{12} \end{array} \right\}$$

(37)
where $a, b = 1 \ldots 5, a \neq b$. In order to avoid weighing down the notation, boundary dimensional factors attached to intertwiners $i_e$ and spins $j_{ab}$ have been neglected. The $\{6j\}$ Wigner symbols in (37) can be computed efficiently with libraries such as wigxjpf and especially fastwigxj [48, 49].

The EPRL vertex amplitude is built from the topological $SL(2, C)$ spinfoam vertex amplitude once the simplicity constraints have been imposed [50, 51]. We use the form of amplitude originally derived in [52], which results in a linear combination of $\{15j\}$ symbols weighted by one booster functions $B_j^\gamma$ per edge (see appendix C for explicit formulas). We write the vertex amplitude according to the graphical notation discussed in detail in [8]:

$$V_{EPRL}^{\gamma}(j_{ab}, i_u, \Delta l) = \sum_{l_{eq}=j_{eq}} \sum_{k_e} \left( \prod_{e} (2k_e + 1) B_{j_{eq}}^\gamma(j_{eq}; i_{eq}, k_e) \right) V_{BF}(j_{1e}, l_{eq}; k_e, i_{a})$$

$$= \sum_{l_{eq}=j_{eq}} \sum_{j_{eq}^{+\Delta l}} \left( \prod_{e} (2k_e + 1) B_{j_{eq}}^\gamma(j_{eq}; i_{eq}, k_e) \right) V_{BF}(j_{1e}, l_{eq}; k_e, i_{a})$$

(38)

where $e, q = 2 \ldots 5, e \neq q$. We introduced the $\Delta l$ parameter in order to truncate the summation over the auxiliary spins $l_{eq}$. The role of this truncation parameter in the context of infrared divergences has been deeply discussed in detail in [7, 46, 53]. In this paper we consider $\Delta l = 20$ and $\gamma = 1.2$ as value of the Barbero–Immirzi parameter. It is interesting to notice how the truncation parameter $\Delta l$ seems to play a minimal role in the computation of boundary observables, as already observed in [7] in the case of infrared divergent amplitudes. In the present context, except for a slight systematic shift in the expectation values of the dihedral angles of the star model (discussed in section 5.2.1), we found no difference\(^6\) between the case $\Delta l = 0$ and $\Delta l = 20$. From a computational point of view, the advantage of using a small $\Delta l$ parameter is remarkable. The 4 spins associated with the gauge-fixed edge are $(j_{12}, j_{13}, j_{14}, j_{15})$, as the elimination of a redundant $SL(2, C)$ integration along one edge in the EPRL vertex (38) is necessary to ensure that the corresponding amplitude is well defined [54]. As in (37) we neglected the dimensional factors attached to the boundary intertwiners and spins.

4.2. Testing the Monte Carlo sampler

The 4-simplex is an excellent model to test the sampling routine over the intertwiners space discussed in section 3.2. In fact, since the number $N = 5$ of boundary degrees of freedom is

---

\(^6\) In order not to be redundant, we do not explicitly report the data in the case $\Delta l = 0$. 
low, it is possible to compute observables and related functions without resorting to Monte Carlo methods, allowing for a direct comparison. In order to perform the sampling algorithm it is necessary to compute the vertex amplitudes (38)–(37) at a fixed value of boundary spin $j$ and for all the possible $(2j+1)^2$ values of intertwiners. For the EPRL model, as mentioned in section 4.1, the computational time considerably increases with $j$, especially with a high value of the truncation parameter $\Delta l$. The computation of the EPRL vertex amplitude (38) for $j = 0, 1, \ldots , 6$ has been distributed over several machines and hundreds of CPUs and it has taken about 5 days to complete. The amplitudes are available at the public repository [34], along with the corresponding BF counterparts (37). We start by looking at the autocorrelation function (26) of the vertex amplitudes sampled during the random walk. Obviously, in order to measure the autocorrelation we set $b = 0$ in the flowchart 1, while all the other parameters are the same reported in appendix B. We report the data in figure 2.

Obviously, the most important parameter to verify the effectiveness of the sampler is the computation of observables. In this respect, a direct comparison is made easier by the fact that the external dihedral angle operator in this simple model has been studied in [33]. The dihedral angle operator is the simplest operator to compute in the intertwiner basis, and it describes the cosine of the external dihedral angle $\cos(\theta_{ab})$ between two faces $a$ and $b$ of a boundary tetrahedron. Faces $a$ and $b$ depend on the recoupling basis chosen for the invariant $SU(2) \{15j\}$ symbol, which appears both in the EPRL (38) and BF amplitude (37). The external dihedral angle of the tetrahedron dual to the node $n$ in the symmetry-reduced space basis states (2) is [33]:

$$\langle j, i_n | \cos(\theta) | j, i_n \rangle = \frac{i_n(i_n+1) - 2j(j+1)}{2j(j+1)} .$$

The dihedral angle operator (39) is diagonal in the spin-network basis, therefore we can compute it very fast with equation (32). We show the statistical fluctuations (36) of the expectation values (32) in the case of the dihedral angle operator (39) in figure 3 for some values of $j$. All the parameters used in the Metropolis-Hastings algorithm are reported in appendix B. The results show that the angle average corresponds to a regular tetrahedron, which is the result originally
Figure 3. Gaussian distribution (36) of the expectation values (32) of the dihedral angle operator (39) in the 4-simplex model. We averaged over several runs, computing the (average) angle \( \mu_{<\cos\theta>} \) defined on a single node and the corresponding standard deviation \( \sigma_{<\cos\theta>} \) for different values of \( j \). Obtained in [33] with deterministic calculations. It is clear how stochastic fluctuations in the random walk over intertwiner space tend to grow much faster for EPRL, rather than for BF, as boundary spin \( j \) increases. This interesting behavior is the main reason why we used a number of \( N_{MC} \) iterations larger than one order of magnitude in the analysis of the star model, as discussed in section 5. In order to have a stable precision up to the third significant digit, especially for large values of \( j \), it is necessary to set a sampling number \( N_{MC} \sim 10^6 \), compared to \((2j+1)^5\) sums to be carried out in the exact calculation (12). Although this indicates that the sampler works as expected, this makes the use of the random walk Metropolis-Hastings algorithm extremely inefficient in the case of a single 4-simplex. However, as for other (Markov Chain) Monte Carlo methods, the advantage obtained in the case of many degrees of freedom emerges surprisingly, as we show in the case of the star spinfoam amplitude. This is due to the fact that the Metropolis-Hastings algorithm (as well as other MCMC methods) is not affected by the problem known as curse of dimensionality.

As mentioned in section 3, the statistical fluctuations in figure 3 are very useful in determining the convergence of the Markov Chain to the stationary distribution, as well as the dispersion of the operators. For the star model, which is the main element of analysis in the present context, we explicitly report the fluctuations (36) computed for all the considered operators and for the entanglement entropy.

5. The star

After testing the algorithm discussed in section 3 to the 4-simplex case, we are ready to study the star spinfoam model, in which a computation without resorting to Monte Carlo methods would be impossible. The two-complex of the star is composed of six vertices (one completely internal), five edges, and has no internal faces. The boundary graph is a refinement of the 4-simplex graph obtained by splitting each of the five nodes into four nodes. The final result of this refinement process is that we obtain 20 nodes on the boundary, which correspond dually to 20 boundary tetrahedra. Therefore the full triangulated manifold is composed of five 4-simplices glued on five internal tetrahedra, each 4-simplex showing four tetrahedra on its boundary. The triangulation of the star graph is shown in figure 4, along with the boundary spin network.
Figure 4. Left: geometry of the star triangulation. Numbered circles correspond to points and each letter is associated with a unique 4-simplex. Colored lines are shared by three tetrahedra belonging to the same 4-simplex. Right: the corresponding boundary spin network. Each red circle corresponds to a boundary node and each line corresponds to a boundary link.

The triangulation of the star model is non-regular since there are segments that are shared between three tetrahedra and other segments that are shared by six tetrahedra. Notice that the dual graph is a sort of ‘magnification’ of the dual 4-simplex. Iterating the same procedure we obtain a fractal structure. The EPRL star amplitude is constructed starting from the vertex amplitudes (38), shown in figure 5. The diagram for the BF is similar, with the difference that each vertex amplitude is simply given by (37). That is, with respect to the graphical amplitude in figure 5, there are no booster functions (and therefore no Y-map). In order not to be redundant, we do not report also the BF spinfoam diagram explicitly. The labels refer to the triangulation shown in figure 4. The 4-simplices are labeled with one letter and four points. The boundary intertwiners are labeled by one letter and three points, which indicate the corresponding tetrahedron in the triangulation (as there is one intertwiner for each node). The links shared by three 4-simplices are labeled by three points, as they are dual to triangles. Those connecting two nodes belonging to the same 4-simplex are labeled with one letter and two points. Finally, the intertwiners connecting the booster functions with the \( \{15j\} \) symbols are labeled with the position of the node in the corresponding 4-simplex. We can write the analytical expression of the amplitude associated with the star spinfoam EPRL and BF respectively as:

\[
A_{\text{EPRL}}^{\gamma}(j, i_b, \Delta l) = \sum_{i_1 \ldots i_5} V_{\text{EPRL}}^{\gamma}(j, i_1, i_2, i_3, i_4, i_5, \Delta l) \prod_{a=1}^{5} V_{\text{EPRL}}^{\gamma}(j, i_a, i_b, \Delta l) ,
\]

\[
A_{\text{BF}}(j, i_b) = \sum_{i_1 \ldots i_5} V_{\text{BF}}(j, i_1, i_2, i_3, i_4, i_5) \prod_{a=1}^{5} V_{\text{BF}}(j, i_a, i_b) ,
\]

where we used the expressions for the EPRL and BF vertex amplitude (38)–(37). The dependence on intertwiners for each vertex (apart from those on which it is necessary to sum over to assemble the amplitude) has been generically indicated with \( i_b \) in order not to weight down the notation. The combinatorial structure of the spinfoam should be clear by looking at figure 5.
Figure 5. Graphical representation of the EPRL spinfoam amplitude corresponding to the star triangulation geometry described in figure 4. Excluding the intertwiners connecting the booster functions with the \( \{15\} \) symbols in the same vertex, there are 5 internal intertwiners that need to be summed over.

5.1. A simple benchmark

Before discussing the expectation values, it is interesting to estimate the computational time of the Metropolis-Hastings algorithm, discussed in section 3.2, applied to the star spinfoam amplitude. As shown in equations (40)–(41) and in the flowchart 1, at each step of the Markov chain we need to perform the contraction of the vertex amplitudes over five bulk intertwiners. This is undoubtedly the computationally most expensive part of the algorithm. For low spins, it is sufficient to perform the contraction with HPC techniques exploiting solely the CPU. We found the best performance using the LoopVectorization Julia package. The offloading of tensor contractions on the GPU with parallelization on the GPU cores [55], exploiting the recent tensor network techniques [56], will be implemented in future works. In fact, the best improvement is obtained for large values of the spins [13], making this approach more suitable for a study of the semiclassical limit of spinfoams rather than the quantum regime. We show a benchmark of the random walk sampling algorithm in figure 6 for increasing values of the total number of iterations \( N_{MC} \) in the Markov chain.
The sampling in figure 6 has been carried out on a laptop with processor Intel(R) Core(TM) \textit{i7-10750H 2.60GHz}. The acceptance rate of intertwiners draws has been set between 30\% and 33\%, with a burn-in parameter $b = 10^3$. In the code available at the repository [34] the Markov chains are automatically parallelized on the number of available CPUs, eventually distributing the computation on multiple machines. As discussed in section 3.2, building more Markov chains is useful for improving accuracy and estimating the error committed due to the statistical fluctuations of the random walk.

5.2. \textit{Numerical results: operators}

We now describe the numerical values obtained for the expectation value of local geometric operators (9) with the boundary state (4) for the star spinfoam amplitude. We also compute the quantum spread (15) and correlation functions (11) between different nodes. For each geometrical operator, we discuss the results obtained both with the BF model and the EPRL model. The parameters used for the sampling of the draws employed for the computation of the operators are discussed in detail in appendix B.

5.2.1. \textit{The dihedral angle operator.} The dihedral angle operator has already been discussed in section 4.2, as we used it in order to test the Monte Carlo sampler. In the star model, since all boundary tetrahedra are equal and regular, we can improve the statistic with a further average\textsuperscript{7}. Namely, we can compute the expectation value of the dihedral angle (39) and the corresponding quantum spread (15) for all 20 nodes of the spinfoam independently at fixed boundary spin $j$, then averaging the results, which are shown in figure 7. The results show that the expectation value of the boundary dihedral angle (39) is peaked to the value corresponding to an equilateral

\textsuperscript{7} This step is justified \emph{a posteriori} once it has been verified that the expectation value of the operator over all the nodes is identical.
Figure 7. Expectation values (32) of the dihedral angle operator (39), averaged over all the 20 nodes of the spinfoam. The orange line shows the value of the cosine of dihedral angle of a regular tetrahedron, which is \( \cos(\theta_{\text{regular}}) = -\sqrt{3} \).

Figure 8. Expectation values of the spread (15) for the dihedral angle operator (39), averaged over all the 20 nodes of the spinfoam. In the EPRL model, the quantum spread increases faster as a function of the boundary spin \( j \).

tetrahedron, which is the same result obtained in section 4 with the simplest possible triangulation of a 3-sphere. This indicates that in the evolution from 1 to 6 vertices, the spatial metric of the boundary state still averages to that of the 3-boundary of a regular 4-simplex, i.e. to that of a 3-sphere. This is not a trivial consequence of the reduction (2) but turns out to be a dynamical result of the global geometry. In fact, in (12) we are considering the sum over all the spinfoam boundary intertwiners. Different geometries might give different (non-regular) average boundary angles, which is for example the case of the \( \Delta_3 \) spinfoam graph [5].

To get an idea of the gain obtained with respect to the blind summation, it is sufficient to consider that at spin \( j = 6 \) it is possible to compute the expectation value of an operator (for example the dihedral angle operator) stably up to the third significant digit with a number of Monte Carlo iterations \( N_{\text{MC}} \sim 10^7 \), as in the 4-simplex model. The exact sum would require performing \((2j + 1)^{20} \sim 10^{22}\) sums. Therefore, the Metropolis-Hastings algorithm adapted to the spinfoam formalism allows to reduce the computation complexity of about 15 orders of magnitude.

The corresponding quantum spread \( \Delta \cos(\theta) \) is shown in figure 8. It turns out to be rapidly increasing for EPRL and slightly increasing for the BF model. As originally noticed in [33], this suggests that quantum fluctuations of the metric in the Lorentzian model are wide, and are not suppressed in the asymptotic regime with few vertices. This however might be a simple
Figure 9. Gaussian distribution (36) of the expectation values (32) of the dihedral angle operator (39). We averaged over several runs, computing the (average) angle $\mu_{\cos \theta}$ defined on a single node and the corresponding standard deviation $\sigma_{\cos \theta}$ for each $j$.

Figure 10. Expectation values (33) of the correlations between dihedral angle operators (39). The correlations computed respectively for the EPRL and the BF models are essentially indistinguishable.

consequence of the boundary state (2), which fixes the areas of the boundary triangles at the quantum level, implying that the boundary angles are quantum totally spread. The results suggest that, even if it is not a regular triangulation, the star model is suitable to discretize a closed geometry as a simplicial manifold bounded by a topological 3-sphere. In fact, in addition to the similarity with the results obtained in [33], preliminary results on the 16-cell spinfoam model, which constitutes the second regular triangulation of the 3-sphere after the 4-simplex, exhibit a striking similar behavior [57]. The Gaussian distributions (36), measuring the statistical fluctuations in the Monte Carlo sampling, are shown in figure 9. For the sake of clarity, we underline that the statistical fluctuations in figure 9 were computed by averaging the expectation value (32) for the operator (39) on a single node over several runs, according to (36), while in figure 7 we performed a further average on the 20 nodes of the spinfoam.

We computed the correlation functions (14) between dihedral angles for all the 190 independent node combinations of the spinfoam. The result is shown in figure 10. For the dihedral angle operator (39), we found that correlations can assume two types of values, both for correlations between operators defined on nodes belonging to the same vertex and for different vertices as well, finding four different total possible numerical values. In the case of nodes on the same vertex, we get the same correlations originally computed in [33]. In the second case,
we observe that also angles between distant vertices can be (only) positively or negatively correlated, and the absolute value of the correlations is small compared to the first case. This is in agreement with the results on the entanglement entropy, as discussed in section 5.3.

The numerical results show that the EPRL and BF models give rise to essentially indistinguishable dynamic correlations in the case of the dihedral angle operator (39). This suggests that, at least in the approximation described in section 2, the SU(2) topological model (typically much easier to compute) provides an excellent approximation for studying dynamical correlations.

5.2.2. The volume operator. There are two slightly different prescriptions for the volume operator in LQG. Here we follow the Rovelli-Smolin prescription in [58]. Since the general expression of the volume matrix elements in the spin-network basis is not trivial [1], here we limit ourselves to the equations in symmetric-reduced space of (1) in which all the spins have the same value \( j \) and the basis states are given by (2).

Let \( A \) be the \((2j + 1) \times (2j + 1)\) Hermitian matrix:

\[
A = i \times \begin{pmatrix}
0 & -a_1 & 0 & 0 & \ldots & 0 \\
a_1 & 0 & -a_2 & 0 & \ldots & 0 \\
0 & a_2 & 0 & -a_3 & \ldots & 0 \\
& & & & & \\
& & & & & \\
& & & & & \\
& & & & & \\
& & & & & \\
& & & & & \\
& & & & & \\
& & & & & \\
& & & & & \\
& & & & & \\
& & & & & \\
& & & & & \\
\end{pmatrix}
\]

(42)

where the coefficients \( a_k \) are defined as:

\[
a_k = \frac{1}{4} k^2 \frac{(2j + 1)^2 - k^2}{\sqrt{4k^2 - 1}}.
\]

(43)

Let \( q_k \) be its real eigenvalues and \( \langle q_k \rangle \) the corresponding eigenvectors. For each \( j \) the eigenvalues come in pairs of opposite signs, plus one 0 eigenvalue for \( j \) integer. The volume operator matrix can be written as:

\[
\langle j, i_n | V | j, i'_n \rangle = \sqrt{2} \frac{\pi G \hbar \gamma}{3} \sum_k \sqrt{|q_k|} \langle j, i'_n | q_k \rangle \langle q_k | j, i_n \rangle.
\]

(44)

Contrary to the dihedral angle (39), the volume operator (44) is not diagonal in the basis (2). In terms of expectation values (30) and correlations (31), this involves recomputing the amplitude function for each element of the sampling, hence it is much slower than the corresponding diagonal evaluation (however it is still incomparably faster than blind summation (11) which would be required without the Monte Carlo approximation (28)). This means that computing the expectation value (30) in the case of the volume operator (44) for more than one node of the spinfoam takes too long. The expectation values of the volumes are shown in figure 11, in which we neglected all the constant factors in the expression (44) since it simply corresponds to a homogeneous re-scaling of all points. It turns out that the scaling of the boundary volume, as a function of the boundary spin \( j \), corresponds to that actually existing between the volume of a regular tetrahedron and the area of one of its faces. That is, \( V \propto j^{3/2} \), as the eigenvalue of the area operator is proportional to \( \sqrt{j(j+1)} \approx j \). This is what we observe in both BF and EPRL models, despite the fact that the spectrum is not the same. An interesting feature of the volume operator spectrum is that there is a systematic shift between integer spins and half-integers spins. That is, these are two slightly shifted curves.

The quantum spread of the volume operator is shown in figure 12. The shift between the curves corresponding to integer spins and half-integers is manifest. Differently with respect to the angle operator 8, for the volume operator the quantum spread increases faster for BF rather
Figure 11. Expectation values (30) of the volume operator (44). The orange line is proportional to the functional dependence between the volume of a regular tetrahedron and the area of one of its faces: $V \propto j^{3/2}$.

Figure 12. Expectation values of the quantum spread (15) for the volume operator (44). As in figure 7, it is evident that the spectrum of the volume operator gives rise to two distinct curves for integer and half-integers spins, which turn out to be shifted with respect to each other.

than EPRL. The Gaussian distributions (36) are shown in figure 13. The average values of the volumes used as mean in the Gaussian distributions (36) are the same plotted in figure 11.

The probability distribution for non-diagonal operators in equation (30) is not strictly positive. Interestingly, as shown in figure 13, this does not affect the convergence. Volume correlations are shown in figure 14. As for the angles, the volumes correlations between nodes belonging to the same vertex are much higher than those between different vertices. For the volumes, the latter appears to be essentially zero. It is interesting to notice that, contrary to what happens with the angles, there is only one type of correlation between volumes. That is, only exists one common value for all the correlations between volumes of tetrahedra on the same vertex, and the same is true for non-adjacent tetrahedra.

5.3. Numerical results: entanglement entropy

We now discuss the results of the computation of the entanglement entropy between different nodes, viewed as quantum subsystems of the whole graph. The entanglement entropy turns out to be the most relevant quantity to study the degree of correlation between operators, as it gives the mutual information between different subsystems [59, 60]. In the topological BF
Figure 13. Gaussian distribution (36) of the expectation values (30) of the volume operator (44). We averaged over several runs, computing the (average) volume \( \mu_{<V>} \) defined on a single node and the corresponding standard deviation \( \sigma_{<V>} \) for each \( j \).

Figure 14. Expectation values (31) of the correlations between volumes. While in the BF model the correlations between volumes on the same node seem to slowly increase with the boundary spin \( j \), in EPRL this does not happen. Unlike angles (see figure 10), all nodes on the same vertex have the same correlation, while nodes on different vertices are essentially uncorrelated.

model, the way in which we defined the boundary state (4) in section 2 coincides with the definition of the Bell-Network states, whose entanglement entropy is computed in [60] for different combinations of graphs and subsystems.

In general, a quantum system composed of two subsystems \( A \) and \( \bar{A} \) has a Hilbert space given by the tensor product:

\[
\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_{\bar{A}}.
\]  

(45)

Given the boundary state \( |\psi_0\rangle \) in the Hilbert space (1), the normalized reduced density matrix of the subsystem \( A \) is defined by the partial trace over its complement \( \bar{A} \):

\[
\rho_A = \frac{1}{Z} \text{Tr}_{\bar{A}} |\psi_0\rangle \langle \psi_0|.
\]  

(46)

The entanglement entropy of the subsystem \( A \) is then defined as the von Neumann entropy of the reduced density matrix

\[
S_A = -\text{Tr} (\rho_A \log \rho_A).
\]  

(47)
Using the expression (4) in (46), after some algebraic manipulations the normalized reduced density matrix (46) can be written as:

$$\rho_A = \frac{1}{2} \sum_{\{i_a\}} \sum_{\{i'_a\}} M(j, \{i_a\}, \{i'_a\}) \otimes_a |j; \{i_a\}\rangle \langle j; \{i'_a\}|,$$

(48)

where \(a \in A\). The coefficients \(M(j, \{i_a\}, \{i'_a\})\) are defined by tracing over the intertwiners \(i_a\) in the complement subsystem \(\bar{A}\):

$$M(j, \{i_a\}, \{i'_a\}) = \sum_{\{i'_a\}} A(j, \{i_a\}, \{i'_a\}) A(j, \{i'_a\}, \{i'_a\}).$$

(49)

We replaced the sum over the full set \(\{i_a\}\) with \(\{i'_a\}\), namely, the intertwiners involved in the partition (45). By introducing the Monte Carlo approximation (28), the expression for the density matrix becomes:

$$\rho_A \approx \frac{1}{N_{MC}} \sum_{|i_a\rangle} \sum_{|i'_a\rangle} A(j, \{i_a\}, \{i'_a\}) A(j, \{i'_a\}, \{i'_a\}) \otimes_a |j; i_a\rangle \langle j; i'_a|.$$

(50)

The notation \([i_a]\) is a label for the set of intertwiners draws \([i_a]\) in which the nodes belonging to the subsystem \(A\) have a value compatible with the position in the density matrix, and the meaning of the amplitude \(A(j, \{i'_a\}, \{i'_a\})\) should be clear by looking at (10). Notice that the density matrix is symmetric and \(\text{Tr}(\rho_A) = 1\). The entropy (47) becomes:

$$S_A \approx -\nu_i \log \nu_i,$$

(51)

where \(\nu_i\) is the \(i\)-th eigenvalue of the density matrix (50). Notice that the computational time of the density matrix considerably increases as the number of nodes \(N_A\) in subsystem \(A\) grows, as the matrix (50) has dimensions \((2j + 1)^{N_A} \times (2j + 1)^{N_A}\). Furthermore, at a fixed number of Monte Carlo iterations \(N_{MC}\), the statistical fluctuations increase along with \(N_A\) since each matrix element is sampled by a set that becomes smaller and smaller. For these reasons, in the numerical computation of the density matrices, we introduced a second multi-threading parallelization scheme using multiple machines. That is, each node computed the density matrix using a different Markov chain by distributing the calculation over multiple CPUs. The same hybrid parallelization scheme can be used to speed up the calculation of non-diagonal operators (30) and corresponding correlations (31).

5.3.1 Subsystem with 1 and 2 nodes. The values of the entanglement entropy as a function of the boundary spin \(j\) are shown in figure 15 for the subsystem \(A\) in the partition (45) consisting in 1 and 2 nodes. We used the parameters listed in appendix B for the data in figure 15. Notice that, by choosing a common value \(j\) for all the links of the star spinfoam (see figure 5), we cannot distinguish correlations between nodes belonging to distinct vertices whether or not they are connected by the same link. For example, labeling the nodes according to the notation illustrated in figure 5, computing the entanglement entropy for the subsystems \(A = \{C235, A235\}\), \(A' = \{C235, D134\}\) and \(A'' = \{C235, B134\}\) we obtain the same numerical value. The value of the EPRL entropy for the subsystem consisting of a single node is similar to the value obtained in [33] for the single vertex graph. It is interesting to notice that the EPRL entropy of all considered subsystems seems to tend asymptotically to a constant value as the boundary spin \(j\) increases.

The statistical fluctuations of entropy are shown in figure 16. Notice that, with the same number of iterations \(N_{MC}\), statistical fluctuations in figure 16 are larger for \(N_A = 2\) for the
Figure 15. Values of the entanglement entropy (51) for different subsystems $A$ in the partition (45) with $N_A = 1$ and $N_A = 2$ for the star spinfoam in figure 5.

Figure 16. Gaussian distribution (36) of the entanglement entropy (51) for the subsystems in figure 15. We proceeded as in the case of angles (figure 9) and volumes (figure 13). Top panel: fluctuations for the subsystem with $N_A = 1$. Bottom panel: fluctuations for the subsystem with $N_A = 2$.

Reasons discussed above. The fact that the entropy of a subsystem composed of nodes on different 4-simplices is slightly greater than the one of the subsystem composed of nodes on the same 4-simplex, is connected to the smaller value of the correlations between nodes belonging to different 4-simplices. In order to discuss this point, we first define the mutual information $I(k,m)$ between two generic nodes $k$ and $m$ as:

$$I(k,m) = S_k + S_m - S_{km},$$

(52)
where $S_{km}$ is the entropy of the subsystem $A$ composed by the nodes $k$ and $m$. It turns out that the mutual information (52) between $k$ and $m$ actually provides a bound on correlations [59, 60]:

$$\frac{(\langle O_k, O_m \rangle - \langle O_k \rangle \langle O_m \rangle)^2}{2\|O_k\|^2\|O_m\|^2} \leq I(k, m),$$

where $\|O_k\|$ is the norm of the local operator $O$ on the node $k$. Therefore, equations (52) and (53), along with the results in figure 15 imply that the correlation function (14) has a more stringent upper bound for the subsystem containing nodes defined on different 4-simplices.

### 5.3.2. Subsystem with 4 nodes.

For completeness, we show in figure 17 the values computed for the entropy of subsystem $A$ composed of 4 adjacent nodes, that is, 4 nodes on the same 4-simplex $A = \{i_{C235}, i_{C234}, i_{C345}, i_{C245}\}$. We show in figure 18 the corresponding Gaussian distributions. For this computation, we limited the analysis to a maximum value $j = 5$ since the computational cost is significantly higher than the other calculations reported in this paper. Unlike the parameters in appendix B, for this specific calculation we set $N_{MC} = 10^7$ for $j = 0.5, 1, \ldots, 5$, averaging over 17 independent runs both for BF and EPRL. Following to the hybrid parallelization scheme discussed at the beginning of this section, we used 17 processes,
each one with 64 CPUs, for a total of 1088 CPUs. With this configuration, the total computation time for the data in figure 17, including the sampling of the intertwiners draws, took about 3 days.

6. Conclusions

In this paper, we combined the Metropolis-Hastings algorithm [12] with recently developed high-performance codes in LQG [13, 45] to compute the expectation value and correlation functions of operators over large spinfoam graphs in the low spins regime. After testing the method, we applied it to the computations of boundary geometrical observables, correlation functions, and entanglement entropy in a spinfoam model with 20 boundary nodes, obtained as a refinement of the 4-simplex graph. We investigated both the EPRL and the BF models. Our results show that the BF and, more significantly, the EPRL model have a well-defined behavior under the refinement of the boundary graph. The computed boundary geometry agrees in terms of expectation value with the geometric interpretation of the operators. We found that correlations are present in neighboring patches but decay sharply when moving to patches that belong to different vertices, opening the way to the study of spinfoams composed of many vertices glued together. We also showed that the dynamical correlations between boundary operators in the BF and EPRL models are surprisingly similar in our case study, while the entanglement entropy shows a significant difference.

Our work provides important hints on the well-definiteness of spinfoam refinement. By refinement, we mean a step toward a less radical truncation. This is implemented by replacing a given triangulation with a finer one. A single boundary graph allows the construction of a quantum state that describes the full evolution of a (discretized) geometry in the (intrinsic, discrete) time provided by the area observables (here assumed equal). When stepping from a given triangulation to a finer one, the same overall continuous spacetime geometry is truncated as a different discretized geometry, that of course has single cells with a different volume and single triangles with a different area, when, say, the overall volume or the overall area of a maximum sphere are given. By stability under refinement, we mean that the quantum state computed with the first triangulation can be seen as a good first approximation to the one computed with the finer one since geometrical operators scale as a function of boundary spins in the same manner before and after the refinement.

The method presented in this paper can be applied to spinfoam models with Lorentzian or Euclidean signatures and to compute bulk observables. An interesting perspective would be to compare the results obtained in the spinfoam-like path integral expansion formalism [61] with the ones in the full spinfoam one. This would give interesting insights into the canonical-covariant relation. Numerical methods currently allow performing computations using spinfoams with a much richer bulk structure than the one considered in this paper. For example, the study of infrared divergences [7, 46] or the analysis of the $\Delta_3$ and $\Delta_4$ triangulation [8]. The methods described in this paper can also be applied to different choices of boundary state, although for complex coherent states one is dealing with fluctuating sampling probabilities and different Monte Carlo techniques might be more effective. The proposed approach provides a needed complement to already existing numerical techniques in covariant LQG [2]. It is effective in the regime of low spins quantum numbers with a large number of degrees of freedom, for which the other existing methods are not tailored for. The next step in the development of this work consists in applying the algorithm presented here to study the correlation functions defined on a spinfoam model with 16 cells on the boundary: this is the next regular triangulation of the 3-sphere after the 4-simplex considered in [33]. This is model is studied in [57].
Data availability statement

No new data were created or analysed in this study.

Acknowledgments

We thank Carlo Rovelli for many discussions on this project. We acknowledge the Shared Hierarchical Academic Research Computing Network (SHARCNET) for granting access to their high-performance computing resources. We thank in particular the Compute/Calcul Canada staff for the constant support provided with the Cedar and Graham clusters. This work was supported by the Natural Science and Engineering Council of Canada (NSERC) through the Discovery Grant ‘Loop Quantum Gravity: from Computation to Phenomenology’. We acknowledge support also from the QISS JFT Grant 61466. F V’s research is supported by the Canada Research Chairs Program at Western University. F V is affiliated to the Perimeter Institute for Theoretical Physics. Research at Perimeter Institute is supported by the Government of Canada through Industry Canada and by the Province of Ontario through the Ministry of Economic Development and Innovation. Western University and Perimeter Institute are located in the traditional lands of Anishinaabek, Haudenosaunee, Lünaapéewak, Attawandaron and Neutral peoples.

Appendix A. Discrete truncated normal distribution

We report in this appendix the definition of truncated normal distribution rounded to integers. For simplicity, we write the equations in the case of a one-dimensional variable. The probability density function of a normal distribution \( N(x, \sigma) \) with mean zero and standard deviation \( \sigma \) is defined as:

\[
N(x, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}},
\]

where \( x \in \mathbb{R} \). We can define the probability distribution function of a normal distribution with mean zero and standard deviation \( \sigma \) rounded to integers as:

\[
N_d(n, \sigma) = \Phi(n + 0.5, \sigma) - \Phi(n - 0.5, \sigma),
\]

where \( n \in \mathbb{N} \) and \( \Phi(x, \sigma) \) is the cumulative distribution function of a normal distribution with mean zero and standard deviation \( \sigma \), defined as:

\[
\Phi(x, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{t^2}{2\sigma^2}} dt.
\]

For convenience, let us also define:

\[
\Phi(a, b; \sigma) = \Phi(b, \sigma) - \Phi(a, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} \int_{a}^{b} e^{-\frac{t^2}{2\sigma^2}} dt.
\]

The cumulative distribution function of a discrete (integer) Gaussian is written as:

\[
\Phi_d(n_1, n_2; \sigma) = N_d(n_1, \sigma) + N_d(n_1 + 1, \sigma) + \ldots + N_d(n_2, \sigma).
\]
With the above definitions, we can define the probability distribution function of a truncated normal distribution rounded to integers \(N_{d,t}(n,n_1,n_2;\sigma)\) between \(n_1\) and \(n_2\) as:

\[
N_{d,t}(n,n_1,n_2;\sigma) = \frac{N_d(n,\sigma)}{\Phi_d(n_1,n_2;\sigma)}.
\]  
(A.6)

**Appendix B. Metropolis-Hastings parameters**

We report in the table B1 below the parameters used in the Metropolis-Hastings algorithm. These are the parameters used for all calculations in this paper except for the data in figures 17 and 6.

A general difference that we observed between the BF and EPRL model is a greater statistical fluctuation in the expectation values of operators for EPRL as \(j\) increases, despite the dimension of the intertwiners’ space is the same. In order to reduce the statistical fluctuations in EPRL, we tried both to increase the number of Markov chains to be averaged by an order of magnitude (in the code [34] the latter are automatically parallelized on the available cores) and to increase the number of Monte Carlo iterations \(N_{MC}\). We found good precision in both cases, and for this paper we decided to use the data obtained with the second approach, as shown in the table B1.

Notice the role that the Metropolis-Hastings parameters play in the sampling process. While increasing the number of chains to be averaged has the effect of improving the accuracy in the determination of the operator’s mean value (and the corresponding standard deviation), increasing the number of Monte Carlo iterations \(N_{MC}\) implies reducing the standard deviation of the statistical sampling. A satisfying statistical precision is therefore achieved when these two parameters are sufficiently high and balanced. While we did not find relevant differences by modifying the number of burn-in iterations, we set the optimal standard deviation of the proposal distribution \(\sigma\) by requiring an acceptance rate of intertwiners draws \([i_n]\) around 30% in the sampling algorithm.
Table B1. Parameters used in the Metropolis-Hastings algorithm. From left to right: $j$ is the spin attached to the links of the star spinfoam, $N_{MC}$ is the number of Monte Carlo iterations, $b$ is the number of burn-in iterations, $\sigma$ is the standard deviation of the truncated normal proposal distribution and $C$ corresponds to the number of Markov chains that we averaged to improve the statistic and measure the statistical fluctuations.

|       | $N_{MC}$ | $b$ | $\sigma$ | $C$ |
|-------|----------|-----|----------|-----|
| **BF 4-simplex** |          |     |          |     |
| $j$    | $N_{MC}$ |     | $\sigma$ | $C$ |
| 0.5    | $10^6$   | $10^2$ | 0.97     | 5   |
| 1.0    | $10^6$   | $10^2$ | 0.94     | 5   |
| 1.5    | $10^6$   | $10^2$ | 0.90     | 5   |
| 2.0    | $10^6$   | $10^2$ | 0.90     | 5   |
| 2.5    | $10^7$   | $10^2$ | 0.90     | 5   |
| 3.0    | $10^7$   | $10^2$ | 0.90     | 5   |
| 3.5    | $10^7$   | $10^2$ | 0.90     | 5   |
| 4.0    | $10^7$   | $10^2$ | 0.90     | 5   |
| 4.5    | $10^7$   | $10^3$ | 0.90     | 5   |
| 5.0    | $10^7$   | $10^3$ | 0.90     | 5   |
| 5.5    | $10^7$   | $10^3$ | 0.90     | 5   |
| 6.0    | $10^7$   | $10^3$ | 0.90     | 5   |
| **EPRL 4-simplex** |          |     |          |     |
| $j$    | $N_{MC}$ |     | $\sigma$ | $C$ |
| 0.5    | $10^7$   | $10^3$ | 0.97     | 5   |
| 1.0    | $10^7$   | $10^3$ | 0.94     | 5   |
| 1.5    | $10^7$   | $10^3$ | 0.90     | 5   |
| 2.0    | $10^7$   | $10^3$ | 0.90     | 5   |
| 2.5    | $10^7$   | $10^3$ | 0.90     | 5   |
| 3.0    | $10^7$   | $10^3$ | 0.90     | 5   |
| 3.5    | $10^7$   | $10^3$ | 0.90     | 5   |
| 4.0    | $10^7$   | $10^3$ | 0.90     | 5   |
| 4.5    | $10^7$   | $10^3$ | 0.90     | 5   |
| 5.0    | $10^7$   | $10^3$ | 0.90     | 5   |
| 5.5    | $10^7$   | $10^3$ | 0.90     | 5   |
| 6.0    | $10^7$   | $10^3$ | 0.90     | 5   |
| **BF star** |          |     |          |     |
| $j$    | $N_{MC}$ |     | $\sigma$ | $C$ |
| 0.5    | $10^6$   | $10^3$ | 0.40     | 32  |
| 1.0    | $10^6$   | $10^3$ | 0.39     | 32  |
| 1.5    | $10^6$   | $10^3$ | 0.37     | 32  |
| 2.0    | $10^6$   | $10^3$ | 0.35     | 32  |
| 2.5    | $10^6$   | $10^3$ | 0.35     | 32  |
| 3.0    | $3 \cdot 10^6$ | $10^3$ | 0.35     | 32  |
| 3.5    | $3 \cdot 10^6$ | $10^3$ | 0.35     | 32  |
| 4.0    | $5 \cdot 10^6$ | $10^3$ | 0.35     | 32  |
| 4.5    | $5 \cdot 10^6$ | $10^3$ | 0.35     | 32  |
| 5.0    | $5 \cdot 10^6$ | $10^3$ | 0.35     | 32  |
| 5.5    | $7 \cdot 10^6$ | $10^3$ | 0.35     | 32  |
| 6.0    | $7 \cdot 10^6$ | $10^3$ | 0.35     | 32  |

(Continued.)
### Appendix C. Booster functions

The booster functions [45, 62], also known as B4 functions [52], are the non compact residuals of the $SL(2, C)$ integrals. These functions turn out to encode all the details of the EPRL model, such as the $Y$–map. We define them as:

$$B_k(j_f, l_f; i, k) = \frac{1}{4\pi \sum (j_f \sigma)} \left( j_f \sigma \right)^{(j_f \sigma)} \left( \int_0^\infty dr \sinh^2 r \prod_{j} d_{(j_f, j_p)}^{(j_i, j_p)}(r) \right) \left( l_f \sigma \right)^{(l_f \sigma)}, \quad \text{(C.1)}$$

where $d_{(j_f, j_p)}^{(j_i, j_p)}(r)$ are the boost matrix elements for $\gamma$-simple irreducible representations of $SL(2, C)$ in the principal series and $\gamma$ is the Immirzi parameter. In their most general formulation, the booster functions turn out to be the $SL(2, C)$ analogs of the usual Clebsch–Gordan coefficients for the rotation group $SU(2)$. The semi-classical limit of booster functions was discussed in [63]. The general explicit form of the boost matrix elements can be found in the literature [52, 64]. In the case of simple irreducible representations, these turn out to be [52]:

$$d_{j_f, j_p}^{(j_i, j_p)}(r) = (-1)^{(j_i \sigma)} \frac{\Gamma(j + i \gamma j + 1) \Gamma(l - i \gamma j + 1) \sqrt{2^j + 1} \sqrt{2^l + 1}}{\Gamma(j + i \gamma j + 1) \Gamma(l - i \gamma j + 1) (j + l + 1)!} \times \left[ (2j)! (l+j)! (l-j)! (l+p)! (l-p)! \right]^{1/2} e^{-(j-i \gamma j + p + 1)} \sum_s \frac{(-1)^s e^{-2sr}}{s! (l-j-s)!} _2F_1,$$

where $_2F_1[a, b, c, z]$ is the hypergeometric function.

---

8 In this appendix we do not indicate the dependence on multiple variables with the curly brackets in order not to weigh down the notation.
References

[1] Rovelli C and Vidotto F 2015 *Covariant Loop Quantum Gravity* (Cambridge: Cambridge University Press)
[2] Han M, Huang Z, Liu H, Qu D and Wan Y 2021 Spinfoam on a Lefschetz thimble: Markov chain Monte Carlo computation of a Lorentzian spin foam propagator *Phys. Rev.* D **103** 084026
[3] Bianchi E, Magliaro E and Perini C 2010 Coherent spin-networks *Phys. Rev.* D **82** 024012
[4] Bianchi E, Magliaro E and Perini C 2009 LQG propagator from the new spin foams *Nucl. Phys.* B **822** 245–69
[5] Dona P, Gozzini F and Sarno G 2020 Numerical analysis of spin foam dynamics and the flatness problem *Phys. Rev.* D **102** 106003
[6] Dona P, Fanizza M, Sarno G and Speziale S 2019 Numerical study of the Lorentzian Engle-Pereira-Rovelli-Livine spin foam amplitude *Phys. Rev.* D **100** 106003
[7] Frisoni P, Gozzini F and Vidotto F 2022 Numerical analysis of the self-energy in covariant loop quantum gravity *Phys. Rev.* D **103** 084026
[8] Asante S K, Dittrich B and Haggard H M 2020 Effective spin foam models for four-dimensional quantum gravity *Phys. Rev. Lett.* **125** 231301
[9] Bianchi E, Rovelli C and Vidotto F 2010 Towards spinfoam cosmology *Phys. Rev.* D **82** 084035
[10] Vidotto F 2011 Many-nodes/many-links spinfoam: the homogeneous and isotropic case *J. Phys.: Conf. Ser.* **314** 012049
[11] Hartle J B and Hawking S W 1983 Wave function of the universe *Phys. Rev.* D **28** 2960–75
[12] Livine E R and Martin-Benito M 2013 Classical setting and effective dynamics for spinfoam cosmology *Class. Quantum Grav.* **30** 035006
[13] Kisielowski M, Lewandowski J and Puchta J 2013 One vertex spin-foams with the dipole cosmology boundary *Class. Quantum Grav.* **30** 025007
[14] Hellmann F 2011 On the expansions in spin foam cosmology *Class. Quantum Grav.* **28** 015009
[15] Gielen S, Oriti D and Sindoni L 2013 Cosmology from group field theory formalism for quantum gravity *Phys. Rev. Lett.* **111** 031301
[28] Gielen S, Oriti D and Sindoni L 2014 Homogeneous cosmologies as group field theory condensates J. High Energy Phys. JHEP06(2014)013
[29] Gielen S and Oriti D 2014 Quantum cosmology from quantum gravity condensates: cosmological variables and lattice-refined dynamics New J. Phys. 16 123004
[30] Gielen S and Sindoni L 2016 Quantum cosmology from group field theory condensates: a review SIGMA 12 082
[31] Vilenisky I 2017 Spinfoam cosmology with the proper vertex amplitude Class. Quantum Grav. 34 225015
[32] Bahr B, Kloser S and Rabuffo G 2017 Towards a cosmological subsector of spin foam quantum gravity Phys. Rev. D 96 086009
[33] Gozzini F and Vidotto F 2021 Primordial fluctuations from quantum gravity Front. Astron. Space Sci. 7 629466
[34] Frisoni P 2021 Github repository for the star spinfoam model (available at: https://github.com/PietropaoloFrisoni/Markov_Chain_Monte_Carlo_spinfoams)
[35] Granlund T et al 2002 GNU multiple precision arithmetic library 4.1.2 (available at: http://swox.com/gmp/)
[36] Fousse L et al 2005 MPFR: a multiple-precision binary floating-point library with correct rounding (INRIA) pp 15 RR5753
[37] Freidel L and Speziale S 2010 Twisted geometries: a geometric parametrisation of SU(2) phase space Phys. Rev. D 82 084040
[38] Bianchi E and Ding Y 2012 Lorentzian spinfoam propagator Phys. Rev. D 86 104040
[39] Robert C P 2015 The Metropolis-Hastings algorithm (arXiv:1504.01896)
[40] Dona P and Sarno G 2018 Numerical methods for EPRL spin foam transition amplitudes and Lorentzian recoupling theory Gen. Relativ. Gravit. 50 127
[41] Yutsin A P, Levinson I B and Vanagas V V 1962 Mathematical Apparatus of the Theory of Angular Momentum (Jerusalem: Israel Program for Scientific Translation)
[42] Johansson H T and Forssén C 2016 Fast and accurate evaluation of Wigner 3j, 6j and 9j symbols using prime factorization and multiword integer arithmetic SIAM J. Sci. Comput. 38 A376–84
[43] Engle J, Livine E, Pereira R and Rovelli C 2008 LQG vertex with finite Immirzi parameter Nucl. Phys. B 799 136–49
[44] Engle J, Pereira R and Rovelli C 2007 The loop-quantum-gravity vertex-amplitude Phys. Rev. Lett. 99 161301
[45] Speziale S 2017 Boosting Wigner’s nj-symbols J. Math. Phys. 58 032501
[46] Frisoni P 2021 Studying the EPRL spinfoam self-energy (arXiv:2112.08528)
[47] Garatli F and Vidotto F 2022 Primordial fluctuations from quantum gravity: 16-cell topological model (in preparation)
[58] Rovelli C and Smolin L 1995 Discreteness of area and volume in quantum gravity Nucl. Phys. B \textbf{442} 593–622

[59] Nielsen M A and Chuang I L 2010 \textit{Quantum Computation and Quantum Information: 10th Anniversary Edition} (Cambridge: Cambridge University Press)

[60] Bianchi E, Donà P and Vilenky I 2019 Entanglement entropy of Bell-network states in loop quantum gravity: analytical and numerical results Phys. Rev. D \textbf{99} 086013

[61] Bodendorfer N, Han M, Haneder F and Liu H 2021 Path integral renormalization in loop quantum cosmology Phys. Rev. D \textbf{103} 126021

[62] Donà P, Fanizza M, Sarno G and Speziale S 2019 Numerical study of the Lorentzian Engle-Pereira-Rovelli-Livine spin foam amplitude Phys. Rev. D \textbf{100} 106003

[63] Donà P and Speziale S 2020 Asymptotics of lowest unitary $SL(2, \mathbb{C})$ invariants on graphs Phys. Rev. D \textbf{102} 086016

[64] Ruhl W 1970 \textit{The Lorentz Group and Harmonic Analysis} (New York: W.A. Benjamin, Inc.)