

Tutorial

# How-to Compute EPRL Spin Foam Amplitudes

Pietro Donà <sup>1</sup> and Pietropaolo Frisoni <sup>2,\*</sup>

<sup>1</sup> Center for Space, Time and the Quantum, 13288 Marseille, France; dona.pietro@gmail.com

<sup>2</sup> Department of Physics and Astronomy, University of Western Ontario, London, ON N6A 5B7, Canada

\* Correspondence: pfriponi@uwo.ca

**Abstract:** Spin foam theory is a concrete framework for quantum gravity where numerical calculations of transition amplitudes are possible. Recently, the field became very active, but the entry barrier is steep, mainly because of its unusual language and notions scattered around the literature. This paper is a pedagogical guide to spin foam transition amplitude calculations. We show how to write an EPRL-FK transition amplitude, from the definition of the 2-complex to its numerical implementation using `s12cfoam-next`. We guide the reader using an explicit example balancing mathematical rigor with a practical approach. We discuss the advantages and disadvantages of our strategy and provide a novel look at a recently proposed approximation scheme.

**Keywords:** loop quantum gravity; spin foam; numerical calculations



**Citation:** Donà, P.; Frisoni, P. How-to Compute EPRL Spin Foam Amplitudes. *Universe* **2022**, *8*, 208. <https://doi.org/10.3390/universe8040208>

Academic Editors: Lisa Glaser, Alessia Platania and Sebastian Steinhaus

Received: 8 February 2022

Accepted: 23 March 2022

Published: 26 March 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Spin foam theory provides a background-independent, Lorentz covariant path integral for general relativity. Spin foams provide dynamics to Loop Quantum Gravity, defining transition amplitudes between spin network states. A triangulation discretizes the space-time manifold, and its 2-complex regularizes the partition function.

The EPRL-FK model [1,2] (we will refer to it as just EPRL for brevity) is the state-of-the-art spin foam model. There is a large consensus in the community [3–7] that the classical continuum theory can be recovered with a double limit of finer discretization and vanishing  $\hbar$ . This observation is supported by the emergence of Regge geometries and the Regge action in the asymptotics of the 4-simplex vertex amplitude for large quantum numbers [8,9] and the recent study of many vertices transition amplitudes.

The amount of calculations possible within the models recently grew considerably. It was possible because of a paradigm shift in the field. It evolved from a theoretical framework to circumvent the difficulties in imposing the Hamiltonian constraint in the canonical approach [10] into a concrete tool where numerical calculations of transition amplitudes are possible [11–17].

With increased interest in the field, its entry barrier also increased vastly. Getting into spin foam is very difficult for a student or a researcher from a different field. There are plenty of reviews [18,19] and excellent books [20] to study and learn the basic theory. On the other end of the spectrum, we have plenty of advanced frontline papers that explore the connection of spin foam with GR [3,6,21] or possible phenomenological implications [22–25].

We noticed a hole in the literature. There are no papers that give you all the tools needed to complete a spin foam calculation, from its conception to the number. With this paper, we guide the reader through the calculation of an EPRL transition amplitude in a pedagogical manner. We use an explicit example to help them not feel disoriented dealing with abstract concepts. We hope that this paper can fill that hole and open spin foam calculation to a new generation of students and researchers.

To read this paper, advanced background knowledge on spin foam is not necessary. However, a basic understanding of the topic is helpful. We think of this work as a guide for

making spin foam calculations. We refer to targeted reviews of the EPRL model [18–20] for a comprehensive discussion of its definition, motivation, and physical significance.

We start with a brief review of the construction of the spin foam theory and the definition of the EPRL model in Section 2. In the rest of the paper, we show the reader how to compute a spin foam transition amplitude associated with a triangulation of the space-time manifold. We identify five necessary steps, each illustrated in a different section.

**Step 1.** Draw the 2-complex.

In Section 3, we describe how to build the 2-complex from the triangulation. It is crucial in regularizing the gravitational path integral and writing a finite transition amplitude.

**Step 2.** Write the EPRL spin foam amplitude.

In Section 4, we give the prescription to write the transition amplitude associated with a 2-complex, and we introduce a very convenient graphical method to represent the amplitude. For the calculation, we resort to a divide-and-conquer strategy.

**Step 3.** Divide the EPRL transition amplitude into vertex contributions.

In Section 5, we show how to divide any transition amplitude into vertex amplitudes.

**Step 4.** Compute the EPRL vertex amplitudes.

In Section 6, we discuss the calculation of the vertex amplitude in terms of  $SU(2)$  invariants and booster functions.

**Step 5.** Use `s12cfoam-next` to compute a number.

We perform the numerical evaluation of the amplitude in Section 7 using the numerical library `s12cfoam-next` and discuss the necessary approximations. In this section, we also discuss and improve the extrapolation scheme discussed in [13] as a tentative to lift, at least part of, the approximation used to calculate the amplitude.

We complement our discussion with an explicit example. We compute the EPRL transition amplitude based on the triangulation  $\Delta_4$ . It was considered first in [26] in Lorentzian the spin foams. It is 2-complex that at the same time not trivial (with more than one vertex), simple (with four vertices and some symmetry) but rich enough (with one bulk face) to require a certain degree of optimization to compute the associated amplitude. Moreover, in [26] coherent boundary data corresponding to a Lorentzian geometry was provided, allowing semiclassical calculation with some ease that we leave to future work.

## 2. Overview of the EPRL Model

Spin foam theory is a promising approach to quantize gravity. The goal is to define a path integral for general relativity in a non-perturbative and background-independent way. The spin foam partition function assigns transition amplitudes between spin network states, a basis of the Loop Quantum Gravity kinematical Hilbert space. For this reason, spin foam theory gives a dynamic to Loop Quantum Gravity, and it is often referred to as Covariant Loop Quantum Gravity [20].

In the Plebanski formulation of general relativity [27], we formulate gravity as a topological BF theory with constraints [28]. The variables of the BF theory are a 2-form  $B$  conjugated to a connection  $\omega$  (with curvature  $F$ )<sup>1</sup>.

General relativity is not topological, the simplicitiy constraints reduce the B-field in BF theory to a  $\gamma$ -simple 2-form  $B = *e \wedge e + \frac{1}{\gamma}e \wedge e$ , reducing the action to the familiar Holst action [29].

The path integral of spin foam theory is regularized on a triangulation, more precisely its 2-complex, to truncate the degrees of freedom. We discretize and quantize the topological theory first. The  $B$ -fields are assigned to faces of the 2-complex, triangles, and encode their geometry. The connection is regularized by considering only its holonomy  $g$  responsible for the parallel transport along the (half-)edges of the 2-complex (from one tetrahedra to another). The topological theory partition function consists of a collection of delta functions imposing flatness of each face of the 2-complex.

The partition function of the EPRL model is derived enforcing the simplicity constraints at the quantum level to reduce the topological theory to gravity. On a simplicial triangulation, we have a linear version of the simplicity constraints: we require the proportionality between the boost and rotation generators of  $SL(2, \mathbb{C})$   $\vec{K} = \gamma \vec{L}$  at the boundary of any 4-simplex (vertex of the 2-complex). The generalization to arbitrary tessellation is possible [30] but requires complications beyond this work's scope. Therefore, we limit ourselves to 4-simplices.

The key ingredient of the EPRL model is the  $Y_\gamma$  map. It embeds the spin  $j$   $SU(2)$  representation into the lowest spin sector of the unitary irreducible representations in the principal series of  $SL(2, \mathbb{C})$  labeled by  $\rho, k = \gamma j, j$ .

We expand the BF theory partition function in terms of matrix elements of the holonomies in irreducible representations of  $SL(2, \mathbb{C})$   $D_{jmjn}^{\rho, k}(g)$ . See Appendix B and references therein for more details. The EPRL model prescription enforces the  $Y_\gamma$  map at every vertex of the 2-complex restricts the irreducible representations to  $\gamma$ -simple ones  $D_{jmjn}^{\gamma j, j}(g)$  [1].

If the 2-complex has a boundary, the spin foam partition function maps states from the Loop quantum Gravity kinematical Hilbert space (identified with the boundary space of the spin foam with the  $Y_\gamma$  map) into the complex numbers (quantum transition amplitudes between these states).

The EPRL spin foam partition function is given as a state sum over  $SU(2)$  spins  $j_f$  on the faces and intertwiners  $i_e$  on the edges of the 2-complex:

$$Z_\Delta = \sum_{j_f, i_e} \prod_f A_f(j_f) \prod_e A_e(i_e) \prod_v A_v(j_f, i_e), \quad (1)$$

defined in terms of the face amplitude  $A_f$ , and the edge amplitude  $A_e$  and the vertex amplitude  $A_v$ . Requiring the correct convolution property of the path integral at fixed boundary, the form of the face amplitude  $A_f(j_f) = 2j_f + 1$  and the edge amplitude  $A_e(i_e) = 2i_e + 1$  are fixed [31].

We will not give an explicit form of the amplitude for an arbitrary 2-complex. They can be found in many references [1, 18, 20] if the reader is interested. Instead, we opt for a constructive approach. In Section 4, we guide the reader through a set of rules to write a general EPRL transition amplitude. In Section 5, we divide the transition amplitude in vertex amplitudes. In Section 6, we discuss the explicit form of the vertex amplitude and its form best suited for numerical calculations.

### 3. How-to Draw the 2-Complex

The spin foam partition function is regularized on the 2-complex of a triangulation of the space-time manifold. Given a triangulation, we can build its 2-complex associating a vertex to each 4-simplex.

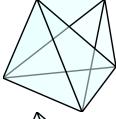
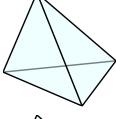
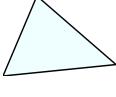
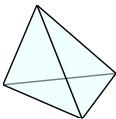
Each 4-simplex shares a tetrahedron with an adjacent 4-simplex. We associate to each tetrahedron of the triangulation an edge of the 2-complex. An edge connects two adjacent vertices. Each triangle in a 4-simplex is shared by two tetrahedra, which are generally shared with other 4-simplices. In the whole triangulation, a triangle can be shared by any number of tetrahedra and 4-simplices.

We associate to each triangle of the triangulation a face of the 2-complex. A face can contain any number of vertices and all the edges connecting them. We also assign an orientation to the faces of the 2-complex. This choice is needed for a well-defined notion of parallel transport (to identify the source and target of the holonomy uniquely).

Since two vertices share each edge for each of them, we can introduce two half-edges, one associated with each vertex. We can still picture them as dual to the tetrahedron but “seen” in the 4-simplex it belongs to. In each vertex in a given face there are two half-edges. This is sometimes referred to as a wedge. The orientation of the face allows us to identify one half-edge as the source tetrahedron (reference frame) and the other as the

target tetrahedron (reference frame) of parallel transport along the face from the first one to the last one. If a boundary is present, the edges intersected by the boundary are severed in half, leaving only one half-edge in the skeleton.

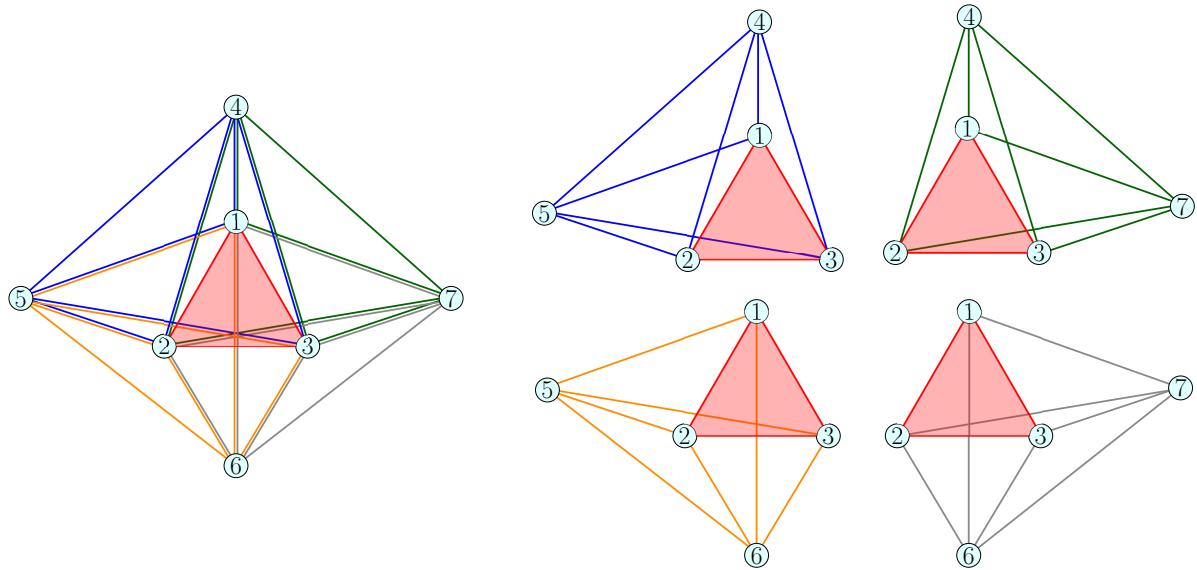
We summarize the nomenclature introduced in this section in the following:

Triangulation	2-complex
	4-simplex
	Tetrahedron
	Triangle
	Tetrahedron within 4-simplex
	Oriented couple of tetrahedra in the same simplex
	Vertex
	Edge
	Face
	Half-edge
	Wedge

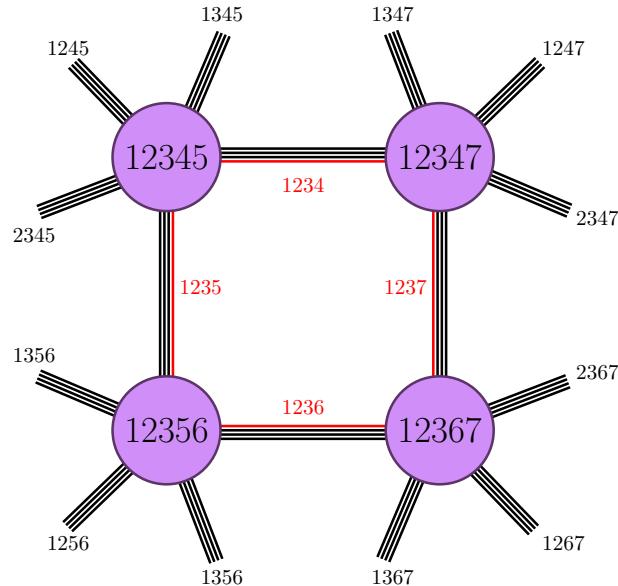
### An Example: The $\Delta_4$ Triangulation

The triangulation is formed by four 4-simplices, all sharing a triangle. The triangulation has seven points, nineteen segments, twenty-five triangles, sixteen tetrahedra (twelve in the boundary and four in the bulk), and four 4-simplices. We label the points with numbers from 1 to 7, segments with couples of different numbers (points), triangles with triples of distinct numbers (the shared triangle is 123), tetrahedra with a quadruple of distinct numbers, and 4-simplices with five distinct numbers. See Figure 1 for a pictorial representation of the triangulation.

The 2-complex of the  $\Delta_4$  triangulation has four vertices associated with a 4-simplex. It has four internal edges, each associated with a tetrahedron shared among two 4-simplices. There are also three external edges for each vertex. Each edge belongs to 4 faces, and each face is associated with a triangle of the  $\Delta_4$  triangulation. All triangles but one belong to the boundary of the triangulation. Therefore all faces but one of the 2-complex are boundary faces. The bulk face is associated with the triangle shared by all the 4-simplices. Thus, it is crossing all four vertices. We label the 2-complex in the same way of the triangulation, see Figure 2 for a representation.



**Figure 1.** The  $\Delta_4$  triangulation. The numbered circles correspond to points, while lines correspond to segments. Each color corresponds to a different 4-simplex. The bulk triangle 123 is highlighted in red. In the right panel, the 4-simplices are shown separately.



**Figure 2.** The 2-complex of the  $\Delta_4$  triangulation. We named the vertices and the tetrahedra explicitly. We avoided naming the faces explicitly not to clutter the figure. Three numbers label the faces. We find a face's name looking for the numbers in common to all the edges it belongs to. For example, the tetrahedra 1234, 1235, 1236, and 1237 all share the face 123.

#### 4. How-to Write the EPRL Spin Foam Amplitude

For each wedge we write a  $\gamma$ -simple unitary irreducible representation in the principal series of  $SL(2, \mathbb{C})$  (see Appendix B and reviews [32] and references therein for more mathematical details).

$$D_{jm,jn}^{\gamma j,j}(g_w), \quad (2)$$

where  $j \in \mathbb{N}/2$  is a spin,  $\gamma$  is the Immirzi parameter coming from the simplicity constraints,  $m, n$  are magnetic indices  $m, n = -j, -j+1 \dots, j-1, j$ , and  $g_w \in SL(2, \mathbb{C})$  is a group element associated with the wedge. This restriction results from the weak quantum implementation of the simplicity constraints in the EPRL spin foam model. The  $Y_\gamma$  map is responsible of this implementation and embeds the spin  $j$   $SU(2)$  representation in  $SL(2, \mathbb{C})$

as in (2). The group element  $g_w$  represents the holonomy responsible for the parallel transport along the wedge from the reference system of the source tetrahedron to the reference system of the target tetrahedron. We conventionally associate the row of the representation matrix, the couple  $(j, m)$  in (2), to the target and the column, the couple  $(j, n)$  in (2), to the source. In this way, the  $SL(2, \mathbb{C})$   $\gamma$ -simple representation matrices inherit the orientation of the 2-complex.

Instead of a group element for each wedge, we prefer to use a group element for each half-edge. We replace  $g_w \rightarrow g_t^{-1} g_s$  where  $s$  and  $t$  are the source and target half-edges. This choice of fundamental variables guarantees that the parallel transport on a closed path in a vertex is trivial. In other words, the product of all the holonomies on the same closed path is the identity <sup>2</sup>, or the holonomy is flat within a single vertex.

We set the spin  $j$  on each edge to be the same and contract the magnetic indices  $m, n$ . At the end of this procedure, the only non-contracted magnetic indices are on boundary half-edges. We prescribe them as part of the boundary data. A common choice to describe boundary data is to contract these magnetic indices with intertwiners in the recoupling basis or with coherent intertwiners if we are interested in representing some semi-classical geometry.

We sum over all the possible spins  $j_f$  associated with each closed face and we weight the contribution of the face with the dimensional factor  $(2j_f + 1)$  [31]

$$\sum_{j_f} (2j_f + 1) \sum_{m_w, n_w} \left( \prod_{w \subset f} D_{j_f m_w j_f n_w}^{\gamma j_f j_f} (g_w) \right), \quad (4)$$

where the product is on all the wedges belonging to the face. On non closed-faces we assign the spin as part of boundary data.

We integrate over the group element associated with each half edge using the Haar measure of  $SL(2, \mathbb{C})$ . For each vertex one integration is redundant and we remove it to regularize the amplitude as prescribed in [33].

#### 4.1. Graphical Notation

Writing all the constituent of an EPRL spin foam amplitude can quickly get out of hand. To help us be precise and clear, we rely on a graphical notation. We introduce the various elements as we need them. We represent a unitary irreducible representation in the principal series of  $SL(2, \mathbb{C})$  as an oriented line. The row labels correspond to the start of the line, and the column labels to the end of the line. We indicate the argument group element in a box and decorate the line with the needed representation labels

$$D_{jmln}^{\rho k}(g) = \xrightarrow{l, n} \boxed{g} \xrightarrow{j, m} \quad (5)$$

We contract two representations summing over all the magnetic indices (both  $j$  and  $m$  are magnetic numbers from the perspective of the infinite-dimensional irreducible representations of  $SL(2, \mathbb{C})$ ) by connecting the two lines. For example, in graphical notation, the  $SL(2, \mathbb{C})$  representation property reads

$$D_{jmln}^{\rho k}(g_2 g_1) = \sum_{\substack{i \geq k \\ |p| \leq i}} D_{jmip}^{\rho k}(g_2) D_{ipln}^{\rho k}(g_1) = \xrightarrow{l, n} \boxed{g_1 g_2} \xrightarrow{j, m} = \xrightarrow{l, n} \boxed{g_1} \xrightarrow{} \boxed{g_2} \xrightarrow{j, m} . \quad (6)$$

We denote the implementation of the  $Y_\gamma$  map (2) with a blue thick line that cuts across the representation line:

$$D_{jmjn}^{\gamma j,j}(g) = n \xrightarrow{j} \boxed{g} \xrightarrow{j} m. \quad (7)$$

If we apply the  $Y_\gamma$  map (7) to the product  $g_1g_2$  and use the decomposition (6), in the graphical notation we have one blue line at both ends:

$$D_{jmjn}^{\gamma j,j}(g_1g_2) = n \xrightarrow{j} \boxed{g_1} \xrightarrow{j} \boxed{g_2} \xrightarrow{j} m. \quad (8)$$

The (infinite) sum over two pairs of magnetic indices is implied in graphical notation, according to Equation (6). If we contract two representation lines with a  $Y_\gamma$  map we only sum over one pair of magnetic indices:

$$\sum_{|p| \leq j} D_{jmjp}^{\gamma j,j}(g_2) D_{jpjn}^{\gamma j,j}(g_1) = n \xrightarrow{j} \boxed{g_1} \xrightarrow{j} \boxed{g_2} \xrightarrow{j} m. \quad (9)$$

We denote with a thicker red line the sum over the spin associated with that representation  $j$  weighted by a dimensional factor  $(2j+1)$ :

$$\sum_j (2j+1) \sum_{m,n} D_{jmjn}^{\gamma j,j}(g_1) D_{jn jm}^{\gamma j,j}(g_2) = \text{Diagram with red thick line and loops.} \quad (10)$$

The (tensor) product of two representations is represented as two lines side by side. If the group element is the same we use a single box. Similarly for the  $Y$ -map, we use a single line. When we draw a box in amplitudes, we will always imply the integration with the Haar measure over the corresponding  $SL(2, \mathbb{C})$  group element:

$$\int dg D_{j_1 m_1 j_1 n_1}^{\gamma j_1 j_1}(g) D_{j_2 m_2 j_2 n_2}^{\gamma j_2 j_2}(g) D_{j_3 m_3 j_3 n_3}^{\gamma j_3 j_3}(g) D_{j_4 m_4 j_4 n_4}^{\gamma j_4 j_4}(g) = \text{Diagram with four input lines labeled n1, n2, n3, n4 and four output lines labeled j1, j2, j3, j4, m1, m2, m3, m4.} \quad (11)$$

#### 4.2. An Example: Writing the $\Delta_4$ Amplitude

With the general recipe discussed in this Section and the corresponding graphical representation, we write the  $\Delta_4$  spin foam amplitude associated with the 2-complex in Figure 2. We also inherit the naming convention from the 2-complex.

$A_{\Delta_4} =$  (12)

To assign a unique name to all the  $SL(2, \mathbb{C})$  group elements, we denoted as  $g$  and  $\tilde{g}$  the two group elements associated with the same (bulk) edge but belonging to different vertices. We used a small abuse of notation in writing (12). Some group elements appears as their inverse. To represent them as a single box we opted to not distinguish them. However, following our conventions, the group element in the matrix element of a target half-edge appears always as its inverse. For example, the half edge 1234 is the source of 234 and the target of 134. The group element  $g_{1234}$  appears as  $D^{\gamma j_{134}/j_{234}}(g_{1234})$  and  $D^{\gamma j_{134}/j_{134}}(g_{1234}^{-1})$ .

As mentioned above, we contracted all the boundary magnetic indices with four valent intertwiners (12 in total) as part of the prescription of the boundary data. We chose the same recoupling basis on each of them and kept the label generic for the moment ( $i_e$  with  $e$  a quadruple identifying a boundary tetrahedron).

We highlighted in red the bulk face (123), dual to the triangle 123 in the  $\Delta_4$  triangulation from Figure 1. According to Equation (10), we are implying a summation over the spin  $j_{123}$  assigned to it weighted by a dimensional factor  $2j_{123} + 1$ . As part of the boundary data, we also prescribed all the spins associated with the boundary faces. We keep them generic for the moment ( $j_f$  with  $f$  a triple identifying a boundary triangle).

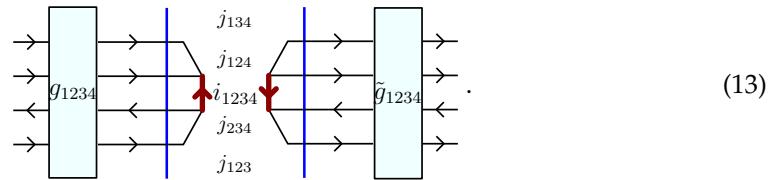
We regularized the amplitude removing one  $SL(2, \mathbb{C})$  integration for each vertex as discussed above. In Equation (12) we indicate the removed integrals with a white box. This

choice is arbitrary, and the amplitude value is independent of this choice. However, we can use this arbitrariness to simplify the numerical computation (see Section 7) by making the symmetric choice. The integral removed is always opposite to the two bulk half edges and the bulk edge (123).

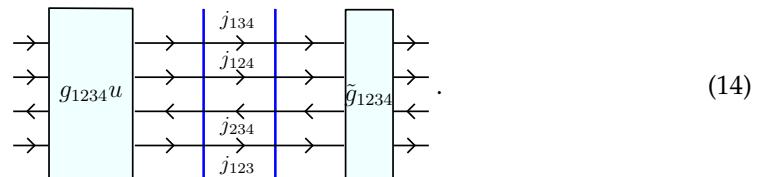
### 5. How-to Divide the EPRL Transition Amplitudes

Approaching the calculation of the full amplitude is an arduous task. The group matrix elements in unitary representations are highly oscillating functions. The integrals are group integrals over many copies (sixteen in our example) of six-dimensional non-compact groups. We divide the transition amplitude into smaller and more manageable components and compute them serialized. This approach is the most advantageous if your goal is to obtain a number from the computation of a transition amplitude. However, this could be suboptimal for semiclassical calculation due to the number of components.

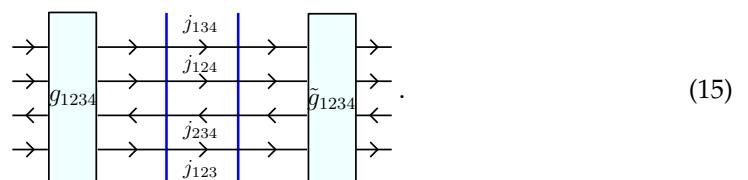
Without any loss of generality, we insert a resolution of the identity over the intertwiner space between every two vertices of (12).



We rewrite the resolution of the identity over the intertwiner space as an integral over  $SU(2)$  of four matrix elements (A23). We commute the  $SU(2)$  integral with the  $Y_\gamma$  map and bring the  $SU(2)$  group element in the  $SL(2, \mathbb{C})$  representation.



Finally, we use the invariance property of the  $SL(2, \mathbb{C})$  Haar measure to reabsorb the  $SU(2)$  group element with a change of variable, obtaining the original spin foam edge.



#### An Example: Decomposing the $\Delta_4$ Amplitude

If we divide the  $\Delta_4$  amplitude (12) inserting 4 resolutions of the identity (each one between two different vertices), the latter decomposes into a linear combination of the product of four amplitudes. That is, one per vertex. These amplitudes are commonly known as *vertex amplitudes*. Using the graphical representation, we write the full  $\Delta_4$  transition amplitude as:

$$A_{\Delta_4} = \sum_{l_f} \quad . \quad (16)$$

By separating the vertices as in (16), we have transformed the problem of calculating the full amplitude into the computation of the single building blocks: the vertex amplitudes.

## 6. How-to Compute the EPRL Vertex Amplitudes

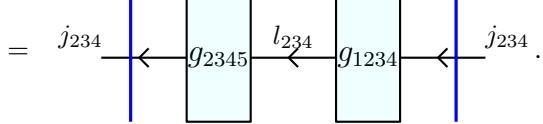
In this section, we will focus on contributions local at the vertices. For concreteness, we model the definition of the vertex amplitude on the (12345) vertex in the example (12).

$$A_{v_{12345}} = \quad . \quad (17)$$

In Equation (17) we contracted the magnetic indices of the bulk edges (1234) and (1235) with two intertwiners, labelled by  $i_{1234}$  and  $i_{1235}$ . We will see in the next section why

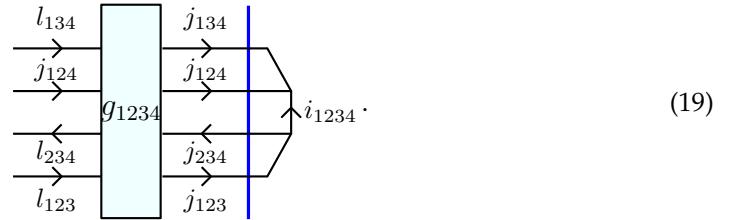
this choice is natural, and we are not losing any generality. Remember that we regularized the amplitude by fixing the group element  $g_{1245} = 1$ , graphically denoting such element by leaving it blank.

Consider the contribution from the wedge (234). We use the representation property to separate the matrix elements corresponding to the two group elements.

$$D_{j_{234}m'_{234}j_{234}n_{234}}^{\gamma j_{234}, j_{234}}(g_{2345}^{-1}g_{1234}) = \sum_{|l_{234}| \geq j_{234}} \sum_{|n_{234}| \leq l_{234}} D_{j_{234}m'_{234}l_{234}n_{234}}^{\gamma j_{234}, j_{234}}(g_{2345}^{-1}) D_{l_{234}n_{234}j_{234}m_{234}}^{\gamma j_{234}, j_{234}}(g_{1234}) \quad (18)$$


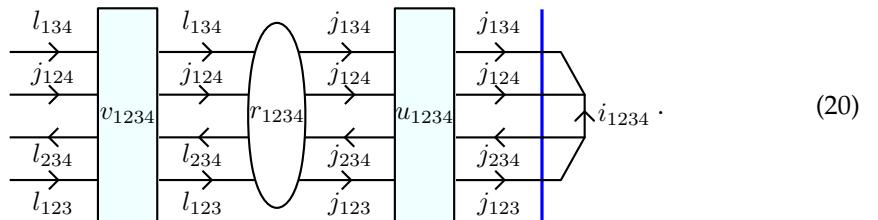
The inverse  $g_{2345}$  is due to the orientation of the wedge (234) and the conventions we are adopting.

To help the reader remember about the extra summation introduced by the representation property, we wrote spin  $l_{234}$  even if we are summing over it. This summation is bounded from below by  $j_{234}$  but is unbounded from above. It is a consequence of the non-compactness of the group (all unitary irreducible representations are infinite-dimensional). Each group element appears as the argument of four matrix elements. For example,  $g_{1234}$  appears in the matrix elements



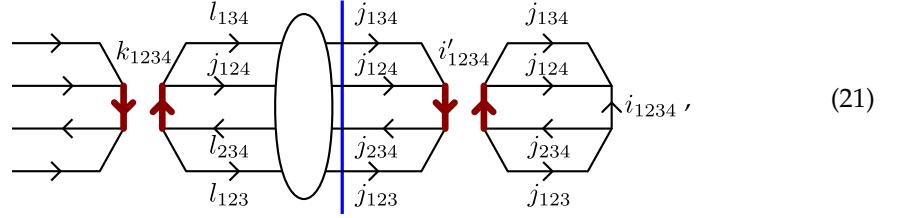
On the face (124) there is no sum over the spin  $l_{124}$ , as a consequence of the regularization choice  $g_{1245} = 1$  and the presence of the  $Y_\gamma$  map on the half-edge (1245).

We parametrize each Lorentz transformation ( $SL(2, \mathbb{C})$  group element) with an arbitrary rotation ( $SU(2)$  group element) followed by a boost in a conventional direction (the 3 direction in our case) and another arbitrary rotation: the Cartan parametrization (A35) of  $SL(2, \mathbb{C})$ . The representation matrices decompose as (A35) and the Haar measure factorizes as in (A36). We divide the contribution of the integral on the half-edge (1234) to the amplitude into



The matrix elements of  $u_{1234}$  and  $v_{1234}$  are  $SU(2)$  matrix elements elements (A38). We represent the integral over the rapidity  $r_{1234}$  of the product of four reduced matrix elements (A40) as a white oval. We wrote the arguments explicitly to help the reader to visualize the parametrization. In the following, we will omit the name of redundant integration variables.

The  $Y$  map commutes with  $SU(2)$  group elements. Therefore, we move it next to the rapidity integral. We perform the integrals over  $SU(2)$  (A25) in terms of  $(4jm)$  symbols. The contribution to the amplitude from the half-edge (20) is



where the thicker red line represent a summation over the corresponding label weighted by a dimensional factor as in (10).

The contraction of two  $(4jm)$  symbols obey the orthogonality condition (A27) and allows us to remove the summation over  $i'_{1234}$

$$(-1)^{2j_{234}} \frac{\delta_{i'_{1234} i_{1234}}}{2i_{1234} + 1}.$$

where the phase  $(-1)^{2j_{234}}$  is a consequence of the different orientation of the link (A13).

We define the booster functions  $B_4^\gamma$  as the result of the integral

$$B_4^\gamma(j_1, j_2, j_3, j_4, l_1, l_2, l_3, l_4; i, k) = k \sum_{p_1, p_2, p_3, p_4} \left( \begin{array}{cccc} l_1 & l_2 & l_3 & l_4 \\ p_1 & p_2 & p_3 & p_4 \end{array} \right)^{(k)} \left( \int_0^\infty dr \frac{1}{4\pi} \sinh^2 r \bigotimes_{f=1}^4 d_{l_f j_f p_f}^{\gamma j_f j_f}(r) \right) \left( \begin{array}{cccc} j_1 & j_2 & j_3 & j_4 \\ p_1 & p_2 & p_3 & p_4 \end{array} \right)^{(i)}.$$

The booster functions were first introduced in [34], numerically computed in [12,35], analytically evaluated in terms of complex gamma functions [36,37], and they have an interesting geometrical interpretation in terms of boosted tetrahedra [38]. The booster functions encode how the EPRL model imposes the quantum simplicity constraints and depend on the Immirzi parameter  $\gamma$ . Note that, in the definition (23), we dropped the information on the orientation of the faces. The orientation of the faces in the booster function is irrelevant. The effect of orientation change of the  $(4jm)$  symbols cancels exactly the effect of orientation change of the reduced density matrices of  $SL(2, \mathbb{C})$ , as we discuss in Appendix B. Using this definition, we can write (19) in terms of the booster functions as:

$$(-1)^{2j_{234}}.$$

We compute the contribution to the amplitude from all other half edges of (12345) with the same prescription. The  $(4jm)$  symbols in (24) contracts among themselves and form a  $\{15j\}$  symbol of the first kind (A30).

$$A_{v12345} = (-1)^{2j_{135}+2j_{234}} \sum_{l_f} \cdot \quad (25)$$

The sum over spins  $l_f$  are only bounded from below (e.g.,  $l_{123} \geq j_{123}$ ) and the intertwiners  $k_e$  are bounded by the triangular inequalities of the  $(4jm)$  symbols. To complete the calculation we recognize the  $SU(2)$ invariant as a canonical  $\{15j\}$  symbol of the first kind (A30). The vertex amplitude is

$$A_{v12345} = (-1)^{2j_{135}+2j_{234}} \sum_{l_f} \left\{ \begin{array}{ccccc} i_{1245} & j_{124} & k_{1234} & l_{234} & k_{2345} \\ j_{145} & l_{134} & l_{123} & l_{235} & j_{245} \\ l_{345} & k_{1345} & l_{135} & k_{1235} & j_{125} \end{array} \right\} B_4^\gamma(j_{235}, j_{234}, j_{345}, j_{245}, l_{235}, l_{234}, l_{345}, j_{245}; i_{2345}, k_{2345}) \\ B_4^\gamma(j_{123}, j_{135}, j_{125}, j_{235}, l_{123}, l_{135}, j_{125}, l_{235}; i_{1235}, k_{1235}) \\ B_4^\gamma(j_{134}, j_{124}, j_{234}, j_{123}, l_{134}, j_{124}, l_{234}, l_{123}; i_{1234}, k_{1234}) \\ B_4^\gamma(j_{145}, j_{345}, j_{135}, j_{134}, l_{145}, l_{345}, l_{135}, l_{134}; i_{1345}, k_{1345}) . \quad (26)$$

In general one need to change the orientation of some links to obtain the canonical  $\{15j\}$  symbol using (A13) to compute the relative phase.

We rewrote the vertex amplitude (17) as a combination of a canonical  $\{15j\}$  symbol weighted by four booster functions.

## 7. How-to Calculate Numbers

In the previous Section, we completed the formal evaluation of the amplitude. If we are satisfied with the expression (16) we can stop here. A few more steps are needed if we want to translate it into a number. We decompose each vertex amplitude in (16) as in (25). By doing so, we finally write the  $\Delta_4$  transition amplitude in the appropriate form for a numerical evaluation:

The diagram shows the EPRL spin foam amplitude  $A_{\Delta_4}$  as a sum of terms, each represented by a 4-simplex with vertices labeled with spin foam indices  $i_{ijkl}$  and edges labeled with loop momenta  $l_{ijkl}$  and  $j_{ijkl}$ . The top row shows two 4-simplices, and the bottom row shows two 4-simplices. The indices for the top row are  $i_{1245}, j_{145}, i_{1345}, j_{245}, l_{345}, j_{124}, l_{134}, i_{2345}, l_{235}, j_{123}, i_{1234}, j_{123}, l_{123}, l_{134}, l_{234}, l_{125}, l_{347}, l_{137}, l_{234}, l_{237}, i_{1247}, j_{147}, i_{1347}, j_{247}, i_{2347}, j_{1235}, i_{1235}, j_{123}, i_{1236}, j_{123}, l_{123}, l_{235}, l_{123}, l_{135}, l_{236}, l_{136}, l_{125}, l_{356}, l_{126}, l_{136}, l_{123}, l_{237}, l_{137}, l_{126}, l_{367}, l_{1367}, l_{1267}, i_{1267}, i_{2367}$ . The bottom row indices are  $i_{2356}, j_{256}, i_{1256}, j_{156}, i_{1356}, l_{235}, l_{123}, l_{135}, l_{125}, l_{356}, l_{126}, l_{136}, l_{123}, l_{237}, l_{137}, l_{126}, l_{367}, l_{1367}, i_{1267}, i_{2367}$ . The equation is  $A_{\Delta_4} = \sum_{l_f} \dots . \quad (27)$

In this paper, we rely on the library `s12cfoam-next` to perform the numerical evaluation of the EPRL spin foam amplitude. The code discussed in this Section is available in the repository in the form of notebooks [39].

### 7.1. Historical Overview

The development of a library for the numerical computation of the Lorentzian EPRL 4-simplex vertex amplitude started with `s12cfoam` [40]. The library is coded in C and is based on the decomposition of the vertex amplitude (26) in terms of booster functions. We refer to the original paper [12] for a detailed discussion of the library's performances, accuracy, and memory management.

The library computes the  $SU(2)$  invariant symbols using `wigxjpf` [41]. The invariants are stored efficiently in custom hash tables based on `khash` [42] that takes into account their symmetry properties.

`s12cfoam` computes the booster functions performing a numerical integration of the boost matrix elements (23). The integrand is rewritten as a finite sum of exponentials with complex coefficients to tame its highly oscillating behaviour. One obtains the booster function from the interference of many exponential integrals done with the trapezoidal rule. In order to reach enough numerical precision, the authors employed arbitrary precision floating point computations with the GNU libraries GMP [43], MPFR [44] and MPC [45].

The library was used to explore the numerical properties of the EPRL vertex [3,11,14,46]. The need for a much more efficient and accurate code immediately became clear, as the computational time for more complex amplitudes was definitely out of reach.

Recently `s12cfoam-next` [47], the evolution of `s12cfoam`, has been released. Furthermore, the new library is written in C, but it has an optional `julia` interface [48] which hugely simplifies its usage. Although `s12cfoam-next` computes the Lorentzian EPRL vertex amplitude in the form Equation (26), it introduces several ideas and techniques borrowed from High-Performance Computing and tensor networks. Therefore, with respect to the original version, it represents a significant improvement in performance and precision. We refer to [35] for its complete description and several usage examples.

The numerical integration of the booster functions is performed with the Gauss-Kronrod quadrature method after a weighted sub-intervals decomposition of the integration range. Furthermore, for technical reasons, the  $\gamma$ -simple unitary irreducible representations in the principal series of  $SL(2, \mathbb{C})$  are slightly different from (2) as it uses  $D^{\gamma(j+1)j}$  instead of  $D^{\gamma j j}$ .

The huge number of sums and products involved in the expression (17) is performed with optimized routines for multidimensional arrays multiplications (we will refer to them loosely as tensors in the informatics sense), such as BLAS [49] and MKL. For the description of the CPU parallelization scheme adopted, we refer to [35]. It has been recently introduced the possibility to offload tensor contractions to the GPU and parallelize them over the GPU cores with the CUDA platform by using the `julia` package `CUDA.jl` [50,51].

### 7.2. Introducing the Cut-Off

The vertex amplitude (26) is made of three distinct elements: the  $\{15j\}$  symbol, the booster functions and the combination of two together. The formula (26) is exact and `s12cfoam-next` can compute its constituents to very high numerical precision. However, the sums over the spins  $l_f$ , that appear in (26) due to the split of the representation matrix elements on the wedges, are bounded from below but not from above. This means that in order to extract a number from (26) we need to make an approximation and cut-off the 6 unbounded sums in the vertex amplitude. While unbounded the sums are convergent because the vertex amplitude is finite [33]. Therefore, in principle, it is possible to find a cut-off large enough to capture the value of the amplitude with the desired precision.

The library `s12cfoam-next` implement an homogeneous cut-off  $\Delta l$  on all the unbounded summations. We replace the sums

$$\sum_{l_f=j_f}^{\infty} \rightarrow \sum_{l_f=j_f}^{j_f+\Delta l} . \quad (28)$$

Unfortunately, we do not have a prescription to find the optimal value of  $\Delta l$ . Numerical explorations show that it depends on the details of boundary data, such as the face spins  $j_f$  and the Barbero-Immirzi parameter  $\gamma$ . At the moment, the best consolidated strategy is to set  $\Delta l$  as large as possible and estimate the error by studying the value of the amplitude  $A_{\Delta_4}(\Delta l)$  as a function of the cut-off.

Recently [13] introduced an extrapolation scheme to overcome the enormous computational cost represented by indefinitely increasing  $\Delta l$ . The extrapolation algorithm was used to calculate the self-energy spinfoam amplitude (see [46,52]), which is a divergent amplitude since it contains a bubble. Furthermore, we mention that the implementation of Markov Chain Monte Carlo methods in the study of spinfoams based on the techniques discussed in this paper is in progress [25].

### 7.3. Using the `s12cfoam-next`

Before any calculation we need to import the `s12cfoam-next` library and initialize it. In the blocks of code of this Section, we will imply that the library is correctly initialized first, and we omit the following code. We report the code initialization in Listing 1.

We set the value of the Immirzi parameter to 1.2 (for historical reasons, any value is equally possible). We define a data folder that is used both to look for the `fastwigxj` tables and to store the computed data optionally. In this way, we avoid recomputing the same vertex amplitude for a second time. We refer to the documentation of `s12cfoam-next` and the accompanying paper [35] for a detailed description of all the setup options.

#### 7.4. Computing One Vertex

We find very valuable to dedicate this paragraph to show how to use the `julia` front-end of `s12cfoam-next` to compute the EPRL vertex amplitude. We use the amplitude (26) as reference. We provide some `jupyter` notebooks <sup>3</sup> in the Git repository [39], for interactive usage examples that the reader can compile and execute. In the Listing 2 we show an essential schematic representation of the code in Listing 2.

We are omitting the initialization code in Listing 1. In lines 1–2, we specify the boundary data (all the spins equal to 1) and the cut-off  $\Delta l = 15$ . In line 3, we compute the amplitude. The function `vertex_compute` returns a tensor with five indices, one per intertwiner, computing the vertex amplitude (25) (without any phase) for all possible values of boundary intertwiners. In [39], we show how to compute a restricted range of boundary intertwiners.

**Listing 1.** Initialization of `s12cfoam-next`.

```
1 using SL2Cfoam
2 Immirzi = 1.2
3 s12c_data_folder = `$(path_to_library_data_folder)`
4 s12c_configuration = SL2Cfoam.Config(VerbosityOff, VeryHighAccuracy, 100, 0)
5 SL2Cfoam.cinit(s12c_data_folder, Immirzi, s12c_configuration)
```

The `@time` macro is used for logging purposes, tracking the computational time and memory usage. At fixed boundary spins and Immirzi parameter, the computation time depends on several parameters such as the value of the cut-off  $\Delta l$  and the accuracy level at which the library is set. With the parameters specified in Listing 1, the first time that line 3 of Listing 2 is run takes 158 s. The computation time decreases exponentially by selecting a lower cut-off  $\Delta l$ . We tested this code on a laptop with Intel(R) Core(TM) i7-10750H 2.60 GHz processor. The library distributes the workload on the available cores, according to the parallelization scheme discussed in [35]. If we store the required booster functions during the first computation, the second time we run the script takes 3.2 s. It is the time to compute, sum, and contract all the required  $\{15j\}$  symbols in the expression (25). Finally, If we store the full vertex amplitude, the computation time is negligible since nothing is calculated, and we retrieve the value from memory.

**Listing 2.** Computation of a vertex amplitude with `s12cfoam-next`.

```
1 Dl = 15
2 spins = j245, j125, j124, j145, j235, j234, j345, j123, j135, j134 = ones(10)
3 @time vs. = vertex_compute(spins, Dl);
```

If we are interested in one single vertex amplitude this is all we need to do.

#### 7.5. An Example: Computing the $\Delta_4$ Amplitude with `s12cfoam-next`

We split the computation of the amplitude (27) into two steps. First, we compute and save the value of all the necessary vertices. Then, we contract the required vertices to calculate the  $\Delta_4$  amplitude. For simplicity, we fix all boundary spins  $j$  equal to 1. The bulk spin  $j_{123}$  assumes values from 0 to  $3j$ , while bulk intertwiners  $i_{1234}, i_{1235}, i_{1236}$ , and  $i_{1237}$  assume values compatible with triangular inequalities. With the regularization choices we did, the vertex amplitudes are fully symmetric. That is, the bulk spin and bulk intertwiners always appear in the same position in each of the four vertices. Therefore, it is sufficient to compute only a single vertex amplitude for all the possible values of spins and intertwiners.

To keep the computational time reasonable, we fix the cut-off  $\Delta l$  to 15. We analyze the dependence of the amplitude on this cut-off in the next step. We report the code in Listing 3.

In lines 2–3, we set all boundary spins equal to 1 and the cut-off  $\Delta l = 15$ . In lines 4–6, we create the directory path to organize the files containing the computed amplitudes. In line 7, we define the range of the bulk spin, and from line 8, we loop over it. In lines 9–12, we assign the vertex amplitude's spins, compute the vertex amplitude, and save it for later use. Notice that we are computing the *fulltensor* vertex amplitude, namely for all the possible values of boundary intertwiners. This ensures that the  $\Delta_4$  amplitude can be calculated for any combination of the latter.

Finally we compute the whole amplitude (27) by assembling all the vertices. We report the corresponding code in Listing 4. One of the main advantages of collecting the vertex amplitudes in multidimensional arrays is that there are very efficient methods to multiply (or “contract”) the latter. For the application we discuss in this work it is unnecessary to improve upon a `for` loop, but `julia` offers the possibility to perform contractions in a wonderfully efficient and simple way, possibly using the GPU. See for example the method `contract`, provided in `s12cfoam-next` to contract vertices with coherent boundary states. Alternatively, there are packages such as `LoopVectorization.jl` (see [53] for an example) or libraries like `ITensor` [54]. In Listing 4, we are assuming that all the variables defined in Listing 3 are available.

**Listing 3.** Computation of all the vertex amplitudes needed in the computation of the transition amplitude (27).

```

1 using JLD2
2 j = 1
3 Dl = 15
4 root_dir = pwd()
5 vertex_path = `$(root_dir)/vertex_ampls/Immirzi_$(Immirzi)/j_$(j)/Dl_$(Dl)```
6 makedirs(vertex_path)
7 j_bulk_min, j_bulk_max = 0, 3j
8 for j_bulk = j_bulk_min:j_bulk_max
9   spins = [j, j, j, j, j, j, j, j_bulk, j, j]
10  vs. = vertex_compute(spins, Dl)
11  vertex = v.a
12  @save `$(vertex_path)/j_bulk_$(j_bulk)_fulltensor.jld2`` vertex
13 end

```

**Listing 4.** Computation of the transition amplitude (27). All the vertex amplitudes are pre-computed.

```

1 i_b = 2
2 i = i_b + 1
3 D4_amp = 0.0
4 for j_bulk = j_bulk_min:j_bulk_max
5   fulltensor_to_load = `$(vertex_path)/j_bulk_$(j_bulk)_fulltensor.jld2```
6   @load `$(fulltensor_to_load)` vertex
7   D4_partial_amp = 0.0
8   D = size(vertex[i,:,:,:i,i])[1]
9   for i_1234 in 1:D, i_1235 in 1:D, i_1236 in 1:D, i_1237 in 1:D
10    @inbounds D4_partial_amp += vertex[i,i_1234,i_1235,i,i]*vertex[i,i_1235,i_1236,i,i]*
11    vertex[i,i_1236,i_1237,i,i]*vertex[i,i_1237,i_1234,i,i]
12  end
13 D4_partial_amp *= (2j_bulk + 1)
14 D4_amp += D4_partial_amp
15 end
16 @show D4_amp

```

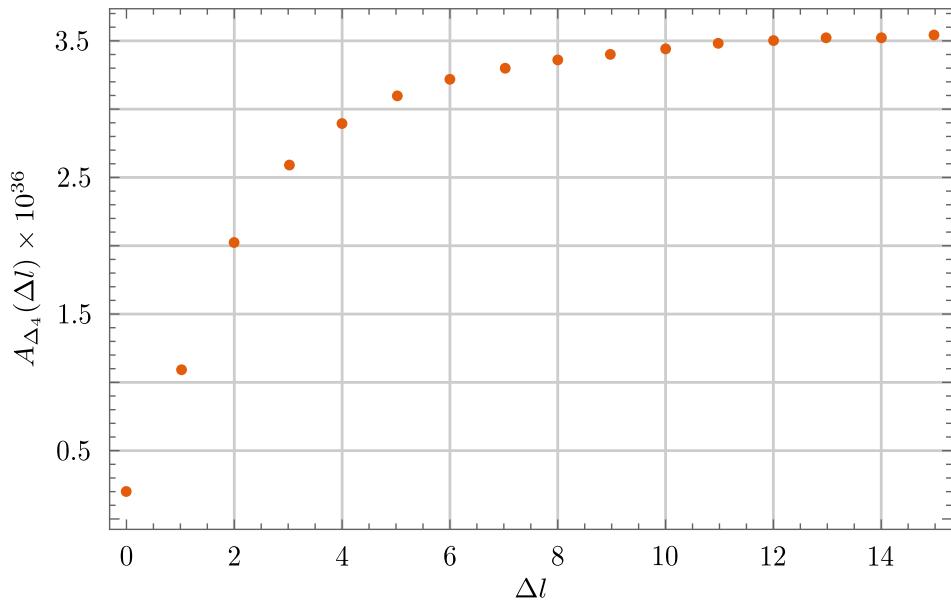
In lines 1–2, we define the boundary intertwiners. For simplicity, we pick them all equal to 2, but any other choice is also possible. Notice that in `julia` the vector's index starts from 1. Therefore, we shift its value. In line 3, we initialize the variable that will contain the amplitude. From line 4, we loop over all the possible values of the bulk spin. In lines 5–6, we load the precomputed amplitude. In line 7, we initialize the variable to store the partial amplitude. The partial amplitude is the quantity in (27) at fixed value of the bulk spin  $j_{123}$ . From lines 8 to 12, we sum over the bulk intertwiners the product of the four vertex amplitudes. In line 13, we add the dimensional factor to the full amplitude value, that we display in line 16.

### 7.6. Results and Extrapolation

We summarize the result of our calculation in Table 1 and Figure 3.

**Table 1.** Numerical values of the amplitude  $A_{\Delta_4}(\Delta l)$  in function of the cut-off.

$A_{\Delta_4}(\Delta l) \times 10^{36}$	0	1	2	3	4	5	6	7
$\Delta l$	0.202	1.09	2.03	2.59	2.90	3.09	3.21	3.29
$A_{\Delta_4}(\Delta l)$	3.36	3.40	3.44	3.47	3.50	3.51	3.53	3.54



**Figure 3.** Amplitude  $A_{\Delta_4}(\Delta l)$  in function of the cut-off.

Looking at the plot in Figure 3 seems reasonable to deduce that by increasing the cut-off  $\Delta l$ , the value of the amplitude grows and (asymptotically) converges to the value of the amplitude. In the first numerical works based on `s12cfoam` and `st2cfoam-next` [3,11] the amplitude was approximated using the value with largest available cut-off. However, we have way more information (convergence, trends, speed). Is it possible to better estimate the amplitude with what we have?

We use series acceleration techniques. We reorganize the sums in the amplitude such that it takes the form

$$A_{\Delta_4}(\Delta l) = \sum_{n=0}^{\Delta l} a_n, \quad (29)$$

where  $a_0$  is the amplitude with vanishing cut-off  $\Delta l = 0$  (also called *simplified model* in [34]),  $a_1$  encodes all the terms in the various sums of  $A_{\Delta_4}$  that appears in the amplitude cut-off  $\Delta l = 1$  but are not in  $a_0$ , and so on. The whole amplitude is recovered in the limit for infinite cutoff of (29).

While recast in this form, we can apply techniques to estimate the value of numerical convergent series like the one in Appendix C. A similar approach was attempted in [13], and here we improve it and clarify it. The technique is analog to the Aitken delta-squared process [55] applied to the the succession of the partial sum (29).

Since the amplitude is finite, the infinite cutoff limit of (29) exists, and the series defined in this way is convergent. We will assume that the ratios  $a_n/a_{n-1}$  are increasing (from a certain point onward). This assumption is backed up by numerical evidence (up to the available cutoff). The lower bound estimate in (A50) specialized for the series (29) is

$$A_{\Delta_4} \gtrsim \frac{A_{\Delta_4}(\Delta l)A_{\Delta_4}(\Delta l - 2) - A_{\Delta_4}^2(\Delta l - 1)}{A_{\Delta_4}(\Delta l) - 2A_{\Delta_4}(\Delta l - 1) + A_{\Delta_4}(\Delta l - 2)} = \frac{A_{\Delta_4}(15)A_{\Delta_4}(13) - A_{\Delta_4}^2(14)}{A_{\Delta_4}(15) - 2A_{\Delta_4}(14) + A_{\Delta_4}(13)} \approx 3.61 \times 10^{-36}, \quad (30)$$

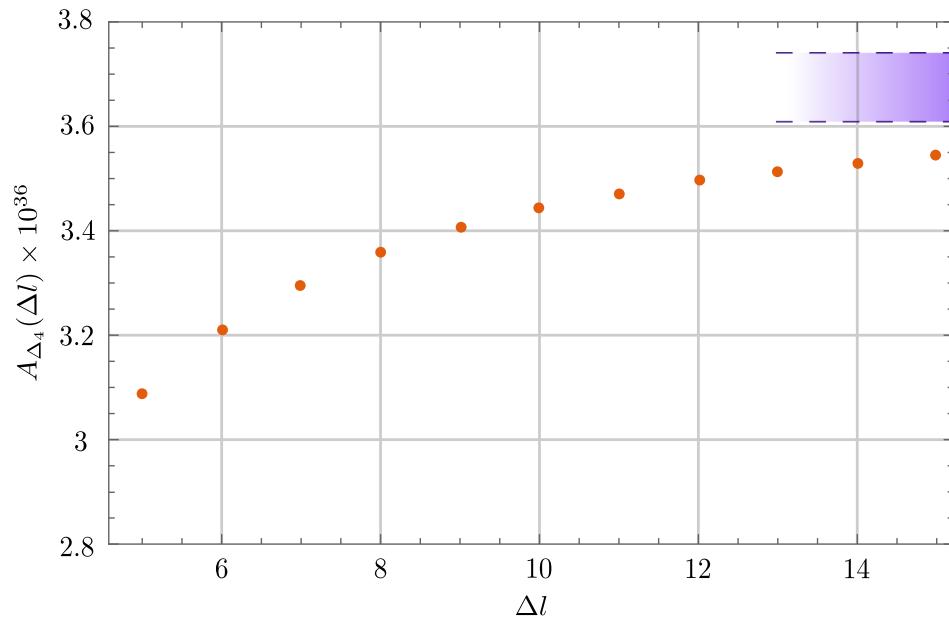
where we specified the largest maximum value of the cut-off we computed, which is  $\Delta l = 15$ . The estimate (30) is significantly different from  $A_{\Delta_4}(15)$  and does not require any additional calculation or resources. The lower bound (30) is analogous to the approximation we can obtain with the Aitken's delta-squared process. With (A50) we also obtain an upper bound to the amplitude.

$$A_{\Delta_4} \lesssim \frac{A_{\Delta_4}(\Delta l) - A_{\Delta_4}(\Delta l - 1)L}{1 - L} = \frac{A_{\Delta_4}(15) - LA_{\Delta_4}(14)}{1 - L} \approx 3.74 \times 10^{-36}, \quad (31)$$

where  $L = \lim_{\Delta l \rightarrow \infty} a_n/a_{n-1}$  which we estimate numerically with a inverse power law fit as in the example in Appendix C. We stress that the validity of this upper bound needs to be taken with a grain of salt since approximating the value of  $L$  can falsify the inequality in (31). Summarizing,

$$A_{\Delta_4} \in (3.61 \times 10^{-36}, \approx 3.74 \times 10^{-36}). \quad (32)$$

We plot in Figure 4 the amplitude together with the bound obtained from (30) and (31) to appreciate the improvement to the rough estimate  $A_{\Delta_4}(15)$ .



**Figure 4.** Amplitude  $A_{\Delta_4}(\Delta l)$  in function of the cut-off with the band (32) highlighted in blue. We excluded the points  $\Delta l < 5$  for a better plot scale.

Computational resources are precious. Up to this point, all the calculations we proposed can be done on a standard laptop. How much can we improve the estimate by increasing the cut-off using High Performance Computing? We used Compute Canada's Narval cluster to increase the cut-off from 15 to 25. The computation was distributed on 80 tasks with 10 CPUs per task, requiring about 8 min. The script used can be found in [39]. These were the resources we could employ in this project. There is still a big room for easy improvement.

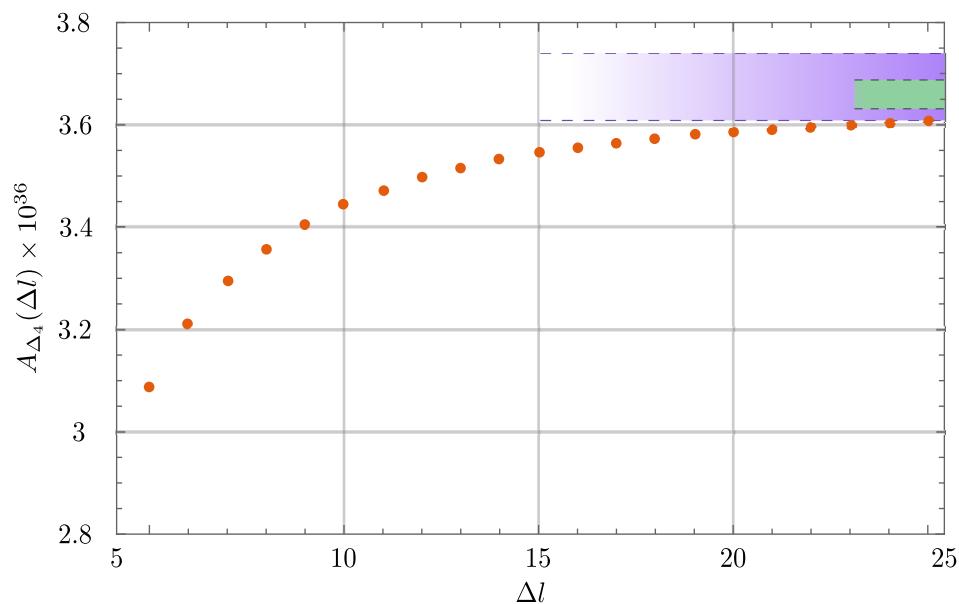
Repeating the estimate process, we find new upper and lower bounds.

$$A_{\Delta_4} \in (3.63 \times 10^{-36}, \approx 3.69 \times 10^{-36}). \quad (33)$$

The lower bound is marginally improved, as expected by comparing the numerical values in Table 2 to the ones in Table 1. However, the improvement on the upper bound is significant. Having more points to extrapolate the limit of the ratios  $L$  is essential. We plot in Figure 5 the result of the calculation.

**Table 2.** Numerical values of the amplitude  $A_{\Delta_4}(\Delta l)$  in function of the cut-off.

$\Delta l$	16	17	18	19	20	21	22	23	24	25
$A_{\Delta_4}(\Delta l) \times 10^{36}$	3.55	3.56	3.57	3.58	3.58	3.59	3.59	3.60	3.60	3.61



**Figure 5.** Amplitude  $A_{\Delta_4}(\Delta l)$  in function of the cut-off with the band (32) highlighted in purple and the band (33) highlighted in green.

The calculation we proposed is not limited to our choice of boundary data. Using the same technique and adapting the code we compute also the value of the  $A_{\Delta_4}$  amplitude for boundary intertwiners  $i_b = 1, 0$  and for other values of the Immirzi parameter  $\gamma = 1, 0.1$ . We summarize the results in Table 3

**Table 3.** Numerical calculation of the amplitude  $A_{\Delta_4}$  with different boundary data. The boundary spins are all equal  $j = 1$  and the cut-off  $\Delta l = 25$ .

	$A_{\Delta_4}(25)$	$A_{\Delta_4}(25)$
$\gamma = 1.2, i_b = 1$	$5.06 \times 10^{-37}$	$(5.09 \times 10^{-37},$
$\gamma = 1.2, i_b = 0$	$1.90 \times 10^{-37}$	$2.17 \times 10^{-37})$
$\gamma = 1.0, i_b = 2$	$9.44 \times 10^{-34}$	$(9.51 \times 10^{-34},$
$\gamma = 1.0, i_b = 1$	$1.49 \times 10^{-34}$	$1.53 \times 10^{-34})$
$\gamma = 1.0, i_b = 0$	$4.95 \times 10^{-35}$	$8.50 \times 10^{-35})$
$\gamma = 0.1, i_b = 2$	$4.28 \times 10^{-24}$	$(4.33 \times 10^{-24},$
$\gamma = 0.1, i_b = 1$	$1.24 \times 10^{-24}$	$1.64 \times 10^{-24})$
$\gamma = 0.1, i_b = 0$	$2.33 \times 10^{-25}$	$2.79 \times 10^{-25})$

**Author Contributions:** All author contributed equally in all the aspects of this paper. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was made possible through the support of the FQXi Grant FQXi-RFP-1818 and of the ID# 61466 grant from the John Templeton Foundation, as part of the “The Quantum Information Structure of Spacetime (QISS)” Project ([qiss.fr](http://qiss.fr)). This work was also supported by the Natural Science and Engineering Council of Canada (NSERC) through the Discovery Grant “Loop Quantum Gravity: from Computation to Phenomenology”.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** All the data used in this paper are freely available in [39].

**Acknowledgments:** We acknowledge the Shared Hierarchical Academic Research Computing Network (SHARCNET) and Compute Canada (accessed on 18 February 2022) ([www.computecanada.ca](http://www.computecanada.ca)) for granting access to their high-performance computing resources. We thank F. Gozzini for very insightful comments on the numeric section of our first draft. We acknowledge the Anishinaabek, Haudenosaunee, Lūnaapéewak and Attawandaron peoples, on whose traditional lands Western University is located

**Conflicts of Interest:** The authors declare no conflict of interest.

## Appendix A. $SU(2)$ Toolbox

The group  $SU(2)$  is the group of  $2 \times 2$  complex matrices with unit determinant that satisfy the unitarity condition

$$\det(u) = 1, \text{ and } u^{-1} = u^\dagger, \forall u \in SU(2). \quad (\text{A1})$$

The group is homomorphic to the rotation group  $SO(3)$  and is generated by the angular momentum algebra  $L_i$  with  $i = 1, 2, 3$  satisfying the commutation relations

$$[L_i, L_j] = i\epsilon_{ijk}L_k. \quad (\text{A2})$$

In the fundamental representation  $L_i = \sigma_i/2$  where  $\sigma_i$  are the standard Pauli matrices. The Casimir operator is  $L^2 = \vec{L} \cdot \vec{L}$  and the unitary irreducible representations are labeled by a spin  $j \in \mathbb{N}/2$  a half-integer and are  $2j + 1$  dimensional. The canonical basis for these representations diagonalizes the operator  $L_3$

$$L^2|j, m\rangle = j(j+1)|j, m\rangle, \quad L_3|j, m\rangle = m|j, m\rangle. \quad (\text{A3})$$

In this basis the matrix elements of the group are given by the Wigner matrices

$$D_{mn}^j(u) \equiv \langle j, m | u | j, n \rangle. \quad (\text{A4})$$

Their explicit expression and properties can be found in [56] and we will not report them.

In this work, we compute integrals of products of  $SU(2)$  representation matrices in terms of  $SU(2)$  invariants. We will introduce the minimal amount of tools needed and the graphical method to perform the calculations. We do not want to provide a complete introduction to recoupling theory and its graphical method that are worth books and reviews on their own [32,57,58]. We use a graphical notation that is completely analogous to the one introduced in Section 4.

We associate an oriented line to each  $SU(2)$  representation matrix. We decorate the line with a spin label and a box containing the group element

$$D_{mn}^j(u) = \begin{array}{c} \xrightarrow{j_1} \\ n \end{array} \boxed{u} \begin{array}{c} \xrightarrow{} \\ m \end{array}. \quad (\text{A5})$$

We contract two representations summing over the magnetic indices by connecting the two lines. We compute the integral over  $SU(2)$  using the unique invariant measure over the group (the Haar measure  $du$  [32]). The explicit form of the measure depends on the parametrization used for the group. We collect the boxes corresponding to the same group elements. In the following, we will always imply the integration over all the group elements in the boxes.

The integral of the product of two representation matrices is given by

$$\int du D_{m_1 n_1}^{j_1}(u) D_{m_2 n_2}^{j_2}(u) = \frac{1}{2j_1 + 1} \delta_{j_1 j_2} (-1)^{2j_1 - m_1 - n_1} \delta_{-m_1 m_2} \delta_{-n_1 n_2} = \frac{1}{2j_1 + 1} \delta_{j_1 j_2} \epsilon_{m_1 m_2}^{j_1} \epsilon_{n_1 n_2}^{j_2}, \quad (\text{A6})$$

where we defined the tensor  $e_{m_1 m_2}^{j_1} \equiv (-1)^{j_1 - m_1} \delta_{-m_1 m_2}$ , the unique invariant tensor in the product of two  $j_1$  representations. The  $\epsilon$  tensor squares to

$$\sum_{m_2} \epsilon_{m_1 m_2}^{j_1} \epsilon_{m_2 m_3}^{j_1} = \sum_{m_2} (-1)^{j_1 - m_1} \delta_{-m_1 m_2} (-1)^{j_1 - m_2} \delta_{-m_2 m_3} = (-1)^{2j_1 - m_1 + m_3} \delta_{m_1 m_3} = (-1)^{2j_1} \delta_{m_1 m_3}, \quad (\text{A7})$$

and has the symmetry property  $\epsilon_{m_1 m_2}^{j_1} = (-1)^{2j_1} \epsilon_{m_2 m_1}^{j_1}$ . We use the graphical representation to write (A6) as

$$\begin{array}{c} j_1 \\ \hline j_2 \end{array} \rightarrow \boxed{u} \rightarrow \begin{array}{c} j_1 \\ \hline j_2 \end{array} = \frac{1}{2j_1+1} \delta_{j_1 j_2} \quad \text{.} \quad (\text{A8})$$

The invariance property of the tensor  $\epsilon_{m_1 m_2}^{j_1}$  means

$$\sum_{n_1, n_2} D_{m_1 n_1}^{j_1}(u) D_{m_2 n_2}^{j_1}(u) \epsilon_{n_1 n_2}^{j_1} = \epsilon_{m_1 m_2}^{j_1}. \quad (\text{A9})$$

From (A9) we can derive the property of Wigner matrices

$$\sum_{m_2 n_2} \epsilon_{n_1 m_2}^{j_1} \epsilon_{m_1 n_1}^{j_1} D_{m_2 n_2}^{j_1}(u) = D_{m_1 n_1}^{j_1}(u^{-1}) . \quad (\text{A10})$$

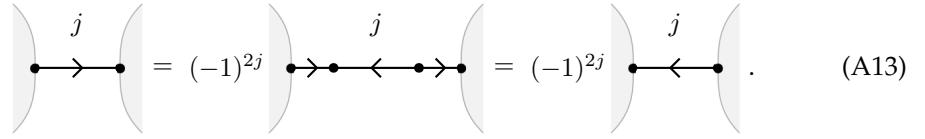
Using this property we can also perform integrals where an inverse group element appears

$$\int du D_{n_1 m_1}^{j_1}(u^{-1}) D_{m_2 n_2}^{j_2}(u) = \begin{array}{c} \text{Diagram with a central vertical line 'u' and two horizontal lines 'j_1' and 'j_2' meeting it at a central point.} \\ = \frac{1}{2j_1+1} \delta_{j_1 j_2} = \frac{1}{2j_1+1} \delta_{j_1 j_2} \end{array} \quad (\text{A11})$$

$$= \frac{1}{2j_1 + 1} \delta_{j_1 j_2} \delta_{m_1 m_2} \delta_{n_1 n_2} . \quad (\text{A12})$$

For simplicity, we will merge the  $\epsilon$  tensors with the box in the following. At first sight, it could appear as an ambiguity since one line will have a group element  $u$  in the box, while the line with the opposite orientation  $u^{-1}$  and there is no indication of which is which. However, we are integrating over  $u$ . Therefore, the name we give the group element is irrelevant. The important information is contained in the relative polarity: one group element is the inverse of the other.

Using this convention and the square property (A7), in any closed diagram, inverting the orientation of a line (without group elements) results into a phase  $(-1)^{2j}$ .



$$\text{Diagram (A13): } \text{Left: } \text{Line } j \rightarrow \text{ (in circle)} = (-1)^{2j} \text{ Line } j \rightarrow \text{ (in circle)} = (-1)^{2j} \text{ Line } j \leftarrow \text{ (in circle)} \text{. Right: } \text{Line } j \leftarrow \text{ (in circle)} \text{.}$$
(A13)

The integral of the product of three representation matrices is given by

$$\int du D_{m_1 n_1}^{j_1}(u) D_{m_2 n_2}^{j_2}(u) D_{m_3 n_3}^{j_3}(u) = \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} \begin{pmatrix} j_1 & j_2 & j_3 \\ n_1 & n_2 & n_3 \end{pmatrix}. \quad (\text{A14})$$

The tensors appearing in (A14) are the Wigner  $(3jm)$  symbols, the unique invariant tensor (or three valent *intertwiner*) in the tensor product of three  $SU(2)$  representations.

$$\sum_{n_1, n_2, n_3} D_{m_1 n_1}^{j_1}(u) D_{m_2 n_2}^{j_2}(u) D_{m_3 n_3}^{j_3}(u) \begin{pmatrix} j_1 & j_2 & j_3 \\ n_1 & n_2 & n_3 \end{pmatrix} = \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix}. \quad (\text{A15})$$

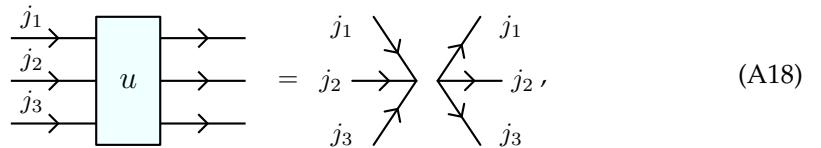
The  $(3jm)$  has the following symmetry properties (see [32,56,58] for an exhaustive list)

$$\begin{pmatrix} j_1 & j_2 & j_3 \\ n_1 & n_2 & n_3 \end{pmatrix} = \begin{pmatrix} j_2 & j_3 & j_1 \\ n_2 & n_3 & n_1 \end{pmatrix} = (-1)^{j_1+j_2+j_3} \begin{pmatrix} j_1 & j_3 & j_2 \\ n_1 & n_3 & n_2 \end{pmatrix}, \quad (\text{A16})$$

and vanishes unless the selection rules are satisfied

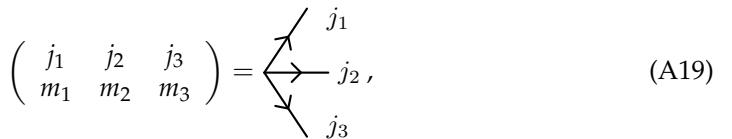
$$m_1 + m_2 + m_3 = 0, \quad |j_1 - j_2| \leq j_3 \leq j_1 + j_2, \quad j_1 + j_2 + j_3 \in \mathbb{N}. \quad (\text{A17})$$

In the graphical representation (A14) is



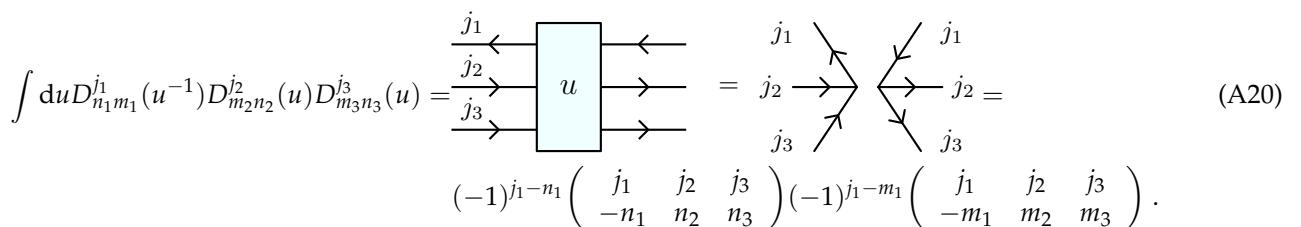
$$\text{Diagram (A18): } \text{Left: } \text{Light blue rectangle } u \text{ with } j_1, j_2, j_3 \text{ in, } j_1, j_2, j_3 \text{ out. Right: } j_1, j_2, j_3 \text{ out.}$$
(A18)

where for the  $(3jm)$  symbol



$$\text{Diagram (A19): } \text{Light blue rectangle } u \text{ with } j_1, j_2, j_3 \text{ out.}$$
(A19)

we read the spins in clockwise order if all the arrows are outgoing and in anti-clockwise order if all the arrows are ingoing. In the standard  $SU(2)$  graphical calculus, this is usually indicated with a sign next to the node [56–58]. For our calculations, this is unnecessary, and we avoid adding this extra layer of complexity. Similarly to (A11) we have



$$\int du D_{n_1 m_1}^{j_1}(u^{-1}) D_{m_2 n_2}^{j_2}(u) D_{m_3 n_3}^{j_3}(u) = \begin{pmatrix} j_1 & j_2 & j_3 \\ -n_1 & n_2 & n_3 \end{pmatrix} (-1)^{j_1-m_1} \begin{pmatrix} j_1 & j_2 & j_3 \\ -m_1 & m_2 & m_3 \end{pmatrix}. \quad (\text{A20})$$

The  $(3jm)$  satisfy the orthogonality relation

$$\sum_{m_2, m_3} \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} \begin{pmatrix} j'_1 & j_2 & j_3 \\ m'_1 & m_2 & m_3 \end{pmatrix} = \begin{array}{c} \text{Diagram of a loop with arrows} \\ \text{with labels } j_1, j'_1, j_2, j_3 \end{array} = \frac{1}{2j_1 + 1} \delta_{j_1 j'_1} \delta_{m_1 m'_1}, \quad (\text{A21})$$

and are normalized to 1

$$\sum_{m_1, m_2, m_3} \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \begin{array}{c} \text{Diagram of a loop with arrows} \\ \text{with labels } j_1, j_2, j_3 \end{array} = 1. \quad (\text{A22})$$

We also compute the integral of the product of four representation matrices with the tools we provided using a trick

$$\int du D_{m_1 n_1}^{j_1}(u) D_{m_2 n_2}^{j_2}(u) D_{m_3 n_3}^{j_3}(u) D_{m_4 n_4}^{j_4}(u) = \int du dv s. D_{m_1 n_1}^{j_1}(u) D_{m_2 n_2}^{j_2}(u) \delta(uv^{-1}) D_{m_3 n_3}^{j_3}(v) D_{m_4 n_4}^{j_4}(v), \quad (\text{A23})$$

where we doubled the number of integrals inserting a delta function. The delta function can be expanded as a sum of Wigner functions [20]

$$\delta(uv^{-1}) = \sum_i (2j+1) \sum_m D_{mm}^i(uv^{-1}) = \sum_i (2i+1) \sum_{m,n} D_{mn}^i(u) D_{nm}^i(v^{-1}). \quad (\text{A24})$$

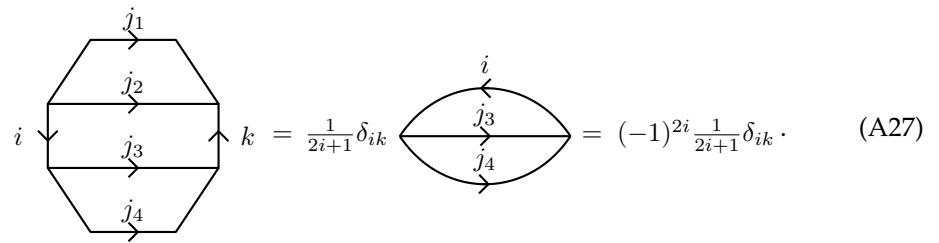
Using the graphical representation and the properties (A15) and (A20) we compute (A23)

$$\begin{array}{c} \text{Diagram of four input lines } j_1, j_2, j_3, j_4 \text{ entering a block } u \\ \text{Diagram of a block } u \text{ with four output lines } j_1, j_2, j_3, j_4 \end{array} = \begin{array}{c} \text{Diagram of a block } u \text{ with four output lines } j_1, j_2, j_3, j_4 \\ \text{Diagram of a block } v \text{ with four input lines } j_3, j_4, j_1, j_2 \end{array} = \begin{array}{c} \text{Diagram of four input lines } j_1, j_2, j_3, j_4 \text{ entering a block } u \\ \text{Diagram of a block } u \text{ with four output lines } j_1, j_2, j_3, j_4 \end{array}, \quad (\text{A25})$$

where we used a red thick line to imply a summation weighted by the dimensional factor  $(2i+1)$  and we define the  $(4jm)$  symbols (invariant tensor or four valent intertwiner) as

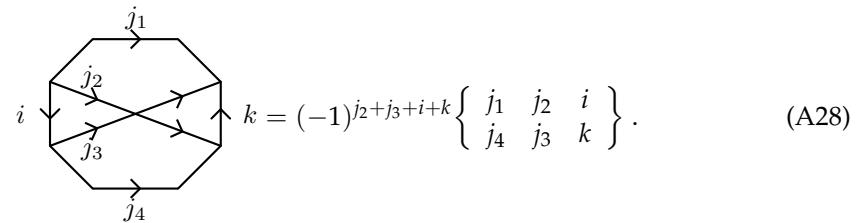
$$\begin{array}{c} \text{Diagram of four input lines } j_1, j_2, j_3, j_4 \text{ entering a block } u \\ \text{Diagram of a block } u \text{ with four output lines } j_1, j_2, j_3, j_4 \end{array} = \begin{pmatrix} j_1 & j_2 & j_3 & j_4 \\ m_1 & m_2 & m_3 & m_4 \end{pmatrix}^{(i)} = \sum_m (-1)^{i-m} \begin{pmatrix} j_1 & j_2 & i \\ m_1 & m_2 & m \end{pmatrix} \begin{pmatrix} i & j_3 & j_4 \\ -m & m_3 & m_4 \end{pmatrix}. \quad (\text{A26})$$

The way we grouped representations together in (A23) is completely arbitrary. The definition (A26) corresponds to the choice of coupling (also called recoupling basis) of the representation of spins  $j_1$  and  $j_2$  (or equivalently spins  $j_3$  and  $j_4$ ). The orthogonality condition (A21) of the  $(3jm)$  symbols imply the normalization of the  $(4jm)$  symbols



$$\text{hexagon} = \frac{1}{2i+1} \delta_{ik} \text{circle} = (-1)^{2i} \frac{1}{2i+1} \delta_{ik}. \quad (\text{A27})$$

The contraction of two  $(4jm)$  symbols in different recoupling basis forms another notable  $SU(2)$  invariant called the  $\{6j\}$  symbol.



$$\text{hexagon} = (-1)^{j_2+j_3+i+k} \left\{ \begin{array}{ccc} j_1 & j_2 & i \\ j_4 & j_3 & k \end{array} \right\}. \quad (\text{A28})$$

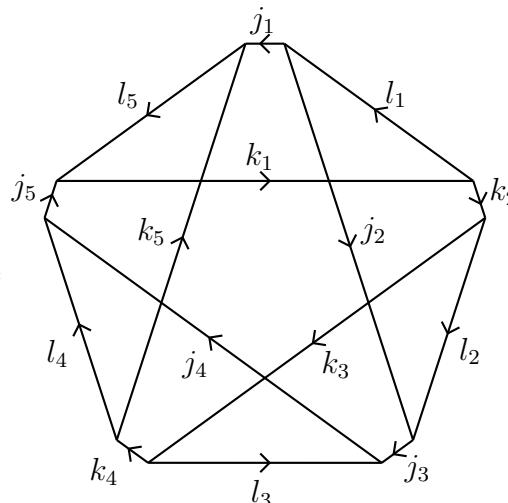
The  $\{6j\}$  symbol in terms of  $(3jm)$  symbols can be written in a canonical form as

$$\left\{ \begin{array}{ccc} j_1 & j_2 & j_3 \\ j_4 & j_5 & j_6 \end{array} \right\} = \sum_{m_1 \dots m_6} (-1)^{\sum_{i=1}^6 (j_i - m_i)} \left( \begin{array}{ccc} j_1 & j_2 & j_3 \\ m_1 & m_2 & -m_3 \end{array} \right) \left( \begin{array}{ccc} j_1 & j_5 & j_6 \\ -m_1 & m_5 & m_6 \end{array} \right) \times \left( \begin{array}{ccc} j_4 & j_5 & j_3 \\ m_4 & -m_5 & m_3 \end{array} \right) \left( \begin{array}{ccc} j_4 & j_2 & j_6 \\ -m_4 & -m_2 & -m_6 \end{array} \right). \quad (\text{A29})$$

For a numerical evaluation, it is not convenient to write the  $\{6j\}$  symbol as in (A29). It is much more efficient to rely on libraries that compute and store Wigner  $\{6j\}$  symbols optimally using recursion and symmetry properties, such as `wigxjpf` and `fastwixj` [41,59].

Another higher-order invariant that appears in our calculations is the irreducible  $\{15j\}$  symbol of the first kind (following the classification of [58]). We can write it both graphically and in terms of  $\{6j\}$  symbols as:

It must be emphasized that the  $\{15j\}$  symbol (A30) is not the most convenient choice from a numerical point of view. In fact, it is possible to choose the recoupling scheme in order to obtain reducible  $\{15j\}$  symbols (see [60] for an example), whose evaluation is much faster. However, since this aspect is not the most critical part of the performance, we prefer to have a pleasantly symmetrical symbol and sacrifice some efficiency. This also simplifies computations of spin foams transition amplitudes with many vertices, since the basis choice in the recoupling on one edge affects both vertices it connects. Therefore, choosing a symmetric  $\{15j\}$  symbol as in (A30), we are sure that the recoupling is consistent in every vertex.



$$\begin{aligned}
 & \left\{ \begin{array}{ccccc} j_1 & j_2 & j_3 & j_4 & j_5 \\ l_1 & l_2 & l_3 & l_4 & l_5 \\ k_1 & k_2 & k_3 & k_4 & k_5 \end{array} \right\} = \\
 & = (-1)^{\sum_{i=1}^5 j_i + l_i + k_i} \sum_x (2x+1) \left\{ \begin{array}{ccc} j_1 & k_1 & x \\ k_2 & j_2 & l_1 \end{array} \right\} \left\{ \begin{array}{ccc} j_2 & k_2 & x \\ k_3 & j_3 & l_2 \end{array} \right\} \\
 & \quad \times \left\{ \begin{array}{ccc} j_3 & k_3 & x \\ k_4 & j_4 & l_3 \end{array} \right\} \left\{ \begin{array}{ccc} j_4 & k_4 & x \\ k_5 & j_5 & l_4 \end{array} \right\} \left\{ \begin{array}{ccc} j_5 & k_5 & x \\ j_1 & k_1 & l_5 \end{array} \right\}. \tag{A30}
 \end{aligned}$$

## Appendix B. $SL(2, \mathbb{C})$ Toolbox

The group  $SL(2, \mathbb{C})$  is the group of  $2 \times 2$  complex matrices with unit determinant. The group is homomorphic to the proper Lorentz group (the Lorentz group part that preserve the sign of the time component) [61,62].

The algebra of  $SL(2, \mathbb{C})$  is generated by spatial rotations and boosts  $L_i$  and  $K_i$  satisfying the commutation relations

$$[L_i, L_j] = i\epsilon_{ijk}L_k, \quad [L_i, K_j] = i\epsilon_{ijk}K_k, \quad [K_i, K_j] = -i\epsilon_{ijk}L_k. \tag{A31}$$

In the spinorial representation  $L_i = \sigma_i/2$  and  $K_i = i\sigma_i/2$  where  $\sigma_i$  are the standard Pauli matrices. The two Casimir operators are  $K^2 - L^2$  and  $\vec{K} \cdot \vec{L}$ . The unitary irreducible representations in the principal series are labeled by  $\rho$  a real number and  $k$  a half-integer. In these representations the Casimirs assume the values

$$(K^2 - L^2)|\rho, k\rangle = (\rho^2 - k^2 + 1)|\rho, k\rangle, \quad \vec{K} \cdot \vec{L}|\rho, k\rangle = \rho k|\rho, k\rangle. \tag{A32}$$

The generic unitary representation  $(\rho, k)$  is infinite dimensional since the group is non-compact. However, we can decompose the representation  $(\rho, k)$  in an infinite number of  $SU(2)$  representations that diagonalize  $L^2$  with different values of the spin  $j$

$$(\rho, k) = \bigoplus_{j \geq k} j. \tag{A33}$$

The definition of the EPRL model is based on the canonical basis of  $(\rho, k)$ . In this basis we diagonalize  $L^2$  and  $L_3$

$$L^2|\rho, k; j, m\rangle = j(j+1)|\rho, k; j, m\rangle, \quad L_3|\rho, k; j, m\rangle = m|\rho, k; j, m\rangle. \tag{A34}$$

with  $j \geq k$  and  $m = -j, \dots, j$ .

The Cartan parametrization [34,62] of the group  $SL(2, \mathbb{C})$  is given by the map

$$g = ue^{\frac{i}{2}\sigma_3}v^{-1}, \tag{A35}$$

where  $u, v \in SU(2)$ ,  $r \in [0, \infty)$  is the rapidity and  $\sigma_3$  is the diagonal Pauli matrix the generator of boosts along the  $z$  axis. The Haar measure with respect to this parametrization is [34,62]

$$dg = \frac{1}{4\pi} \sinh^2 r \, dr \, du \, dv \dots \quad (\text{A36})$$

Using the Cartan parametrization (A35) the matrix elements of a group element  $g$  in the canonical basis reads

$$D_{jmln}^{\rho,k}(g) \equiv \langle \rho, k; j, m | g | \rho, k; l, n \rangle = D_{jmln}^{\rho,k}(ue^{\frac{r}{2}\sigma_3}v^{-1}) = \sum_{a,a'} D_{ma}^j(u) D_{jala'}^{\rho,k}(e^{\frac{r}{2}\sigma_3}) D_{a'n}^l(v^{-1}). \quad (\text{A37})$$

The subgroup  $SU(2) \subset SL(2, \mathbb{C})$  is generated by  $\vec{L}$  and its matrix elements are given by  $SU(2)$  Wigner matrices (A4)

$$D_{jmln}^{\rho,k}(u) = \langle \rho, k; j, m | u | \rho, k; l, n \rangle = \delta_{jl} D_{mn}^j(u) \text{ where } u \in SU(2). \quad (\text{A38})$$

Moreover,  $e^{\frac{r}{2}\sigma_3}$  is diagonal, therefore  $D_{jmln}^{\rho,k}(e^{r\sigma_3}) = \delta_{aa'} D_{jala}^{\rho,k}(e^{r\sigma_3}) \equiv \delta_{aa'} d_{jla}^{\rho,k}(r)$  where  $d_{jla}^{\rho,k}$  are called reduced matrix elements of  $SL(2, \mathbb{C})$ . Summarizing

$$D_{jmln}^{\rho,k}(g) = \sum_a D_{ma}^j(u) d_{jla}^{\rho,k}(r) D_{an}^l(v^{-1}). \quad (\text{A39})$$

The expression for  $d_{jlm}^{\rho,k}(r)$  was given in [34,62–65]

$$\begin{aligned} d_{jlm}^{\rho,k}(r) = & (-1)^{j-l} \sqrt{\frac{(i\rho - j - 1)!(j + i\rho)!}{(i\rho - l - 1)!(l + i\rho)!}} \frac{\sqrt{(2j+1)(2l+1)}}{(j+l+1)!} e^{(i\rho - k - m - 1)r} \\ & \sqrt{(j+k)!(j-k)!(j+m)!(j-m)!(l+k)!(l-k)!(l+m)!(l-m)!} \\ & \sum_{s,t} (-1)^{s+t} e^{-2tr} \frac{(k+s+m+t)!(j+l-k-m-s-t)!}{t!s!(j-k-s)!(j-m-s)!(k+m+s)!(l-k-t)!(l-m-t)!(k+m+t)!} \\ & {}_2F_1 \left[ \{l - i\rho + 1, k + m + s + t + 1\}, \{j + l + 2\}; 1 - e^{-2r} \right] \end{aligned} \quad (\text{A40})$$

where  ${}_2F_1$  is the Gauss hypergeometric function. The phase used in (A40) is the same introduced in [34], which ensures the reality of the booster function (23). The reduced matrix elements (A40) satisfy the following relation:

$$\overline{d_{jlm}^{\rho,k}(r)} = (-1)^{j-l} d_{j-l-m}^{\rho,k}(r). \quad (\text{A41})$$

As a consequence, the matrices (A38) have the property:

$$\overline{D_{jmln}^{\rho,k}(g)} = (-1)^{j-l+m-n} D_{j-m-l-n}^{\rho,k}(g). \quad (\text{A42})$$

We can write the  $SL(2, \mathbb{C})$  matrix elements of  $g^{-1}$  as:

$$D_{lnjm}^{\rho,k}(g^{-1}) = (-1)^{j-l+m-n} D_{j-m-l-n}^{\rho,k}(g) = \sum_a (-1)^{j-l+m-n} D_{-ma}^j(u) d_{jla}^{\rho,k}(r) D_{a-n}^l(v^{-1}), \quad (\text{A43})$$

where in the first equality we used (A42) (in addition to the  $SL(2, \mathbb{C})$  irrep properties) and in the second one (A39). Since there are no phases depending on the summed index, we conclude that the orientation of the  $(4jm)$  spins in the booster function (23) is irrelevant. This justifies the fact that we draw the latter without arrows.

### Appendix C. Approximation of a Convergent Series

In this appendix, we provide further details on the extrapolation scheme used in Section 7. This is analogous to the more general Aitken's delta-squared process [55], which accelerate the rate of convergence of a sequence providing a good approximation technique. Consider the series  $S = \sum_n^\infty a_n$  and cut-offed sum  $S_N = \sum_n^N a_n$ . By definition the series is the limit of  $S_N$  for infinite cut-off

$$S = \lim_{N \rightarrow \infty} S_N = \lim_{N \rightarrow \infty} \sum_n^N a_n = \sum_n^\infty a_n. \quad (\text{A44})$$

Suppose that the sequence  $a_n$  is positive and, from a certain point onwards, increasing such that

$$\lim_{N \rightarrow \infty} c_N \equiv \lim_{N \rightarrow \infty} \frac{a_N}{a_{N-1}} = \lim_{N \rightarrow \infty} \frac{S_N - S_{N-1}}{S_{N-1} - S_{N-2}} \equiv L < 1, \quad (\text{A45})$$

where the ratios increase to  $L$ . The series  $S$  is convergent by the ratio test since the ratios are increasing:

$$c_N = \frac{a_N}{a_{N-1}} < \frac{a_k}{a_{k-1}}, \quad \forall k > N. \quad (\text{A46})$$

Hence we have  $a_{N+1} = a_N \frac{a_{N+1}}{a_N} > a_N c_N$ ,  $a_{N+2} > a_{N+1} c_N > a_N c_N^2$ , and in general  $a_{N+m} > a_N c_N^m$  for  $m > 0$ . We can provide a bound on the series observing that

$$S - S_N = \sum_{n=N+1}^{\infty} a_n = \sum_{m=1}^{\infty} a_{N+m} > \sum_{m=1}^{\infty} a_N c_N^m = a_N \frac{c_N}{1 - c_N}. \quad (\text{A47})$$

Similarly, we have that  $\frac{a_k}{a_{k-1}} < L \forall k > N$  by definition of  $L$  and monotonicity of the ratios.

$$S - S_N = \sum_{n=N+1}^{\infty} a_n = \sum_{m=1}^{\infty} a_{N+m} < \sum_{m=1}^{\infty} a_N L^m = a_N \frac{L}{1 - L}. \quad (\text{A48})$$

Summarizing, we have an estimate from above and below of the value of the series as

$$S_N + a_N \frac{c_N}{1 - c_N} < S < S_N + a_N \frac{L}{1 - L}. \quad (\text{A49})$$

If the ratios are decreasing instead of increasing, we obtain an estimate analog to (A49) but with inequalities operators inverted. Let us focus on (A49) since it is the case relevant for the cut-off approximation presented in Section 7. We rewrite (A49) in terms of cut-offed sums as

$$\frac{S_N S_{N-2} - S_{N-1}^2}{S_N - 2S_{N-1} + S_{N-2}} < S < \frac{S_N - S_{N-1} L}{1 - L}. \quad (\text{A50})$$

Often, in real-world physical applications, the analytical expression of  $a_n$  is very complicated. We cannot compute  $S$  but can still calculate the cut-offed sums  $S_N$  with  $N$  as large as our numerical computational resources allow. What is the best approximation of  $S$  we can find? We will assume that we know  $S$  is convergent (so that the question is well-posed) and that the ratios  $c_N$  increase. We want to use the inequalities (A50). We can numerically compute the left-hand side of the inequality. What about the right-hand side? The convergence of  $S$  ensure that  $\lim_{N \rightarrow \infty} c_N = L$  exists, however in general we cannot compute the value of  $L$ . At the moment, there is no clear strategy on how to compute  $L$ . In this work, we will consider two possibilities. None of them is optimal, and we leave improvements to future work. Notice that no matter what approximation we decide to adopt to compute  $L$  the right inequality of (A50) will not hold anymore.

For example, we can approximate  $L$  with the largest available ratio  $L \approx c_N$ . In particular, if we insist and substitute in (A50) the approximation  $L \approx c_N$ , the right quantity

becomes equal to the left one. We have to content ourselves with a lower bound estimate of the amplitude given

$$S \gtrapprox \frac{S_N S_{N-2} - S_{N-1}^2}{S_N - 2S_{N-1} + S_{N-2}}. \quad (\text{A51})$$

The estimate (A51) is very similar to the strategy used in [13,15]. We clarified that it is a lower bound. Another possibility is to use the sequence of ratios  $c_N$  computed numerically to estimate the value of  $L$ . This extrapolation is slightly dangerous since its accuracy depends on how large we can take the cut-off  $N$ . Of course, this is in addition to the lower bound (A51).

In the following, we provide a concrete toy model example. Consider the series

$$S = \sum_{n=1}^{\infty} \frac{1}{n+1} (9/10)^n = \frac{10}{9} \log(10) - 1 \approx 1.558. \quad (\text{A52})$$

This series is exactly summable in terms of the log function. Nevertheless, we want to approximate the series pretending not to know how to sum it, ignoring the fact that any analytical calculation is straightforward, and relying only on numerical tools. Let us assume that the largest possible cut-off we have access to is  $N = 15$ . We can compute the cut-offed sums

$$S_{15} \approx 1.480, \quad S_{14} \approx 1.467, \quad S_{13} \approx 1.452. \quad (\text{A53})$$

We can immediately apply (A51) to obtain

$$S \gtrapprox \frac{S_{15} S_{13} - S_{14}^2}{S_{15} - 2S_{14} + S_{13}} \approx 1.549. \quad (\text{A54})$$

The largest cut-offed sum is 5% off the real value while the lower bound approximation (A54) is closer, being only 0.6% off. The numerical values for the ratios are summarized in Table A1.

**Table A1.** Numerical values of the ratios for (A52)

N	3	4	5	6	7	8	9	10	11	12	13	14	15
$c_N$	0.675	0.720	0.750	0.771	0.788	0.800	0.810	0.818	0.825	0.831	0.836	0.840	0.844

We extrapolate the limit at infinity of the ratios  $L$  fitting the data using the first few terms of an inverse power law and keeping the constant term. It is a cheap and dirty way of extrapolating, and one should be more careful. However, it is more than enough for our purposes. We use Wolfram's Mathematica built-in `Fit` method to perform the fit and find  $L \approx 0.889$ . If we substitute it in (A50), keeping in mind the approximations we are making, we find the estimate

$$S \lesssim \frac{S_{15} - S_{14}L}{1 - L} \approx 1.583, \quad (\text{A55})$$

which is 1.6% larger than the actual value. Combining the two estimates, we obtain a range for the series  $S \in [1.549, 1.583]$ .

## Notes

- This theory has no degrees of freedom: all the solutions of the equations of motion are gauge equivalent to the trivial one  $d_\omega B = 0$  and  $F(\omega) = 0$ . The name derives from the name of the variables used and the simple form of the action  $\int_{\mathcal{M}} B \wedge F(\omega)$ .
- Explicitly, if  $w_1, w_2$  and  $w_3$  are three wedges of the same vertex we have

$$g_{w_3} g_{w_2} g_{w_1} = g_{e_1}^{-1} g_{e_3} g_{e_1}^{-1} g_{e_2} g_{e_2}^{-1} g_{e_1} = \mathbb{1}, \quad (3)$$

where we have assumed that the wedges are oriented such that the target of  $w_1$  is the source of  $w_2$  and so on. If the orientation of one of the wedges  $w$  is the opposite we replace  $g_w$  with its inverse.

3 The code in [39] was tested with the kernel `julia 1.7.0`

## References

1. Engle, J.; Livine, E.; Pereira, R.; Rovelli, C. LQG vertex with finite Immirzi parameter. *Nucl. Phys.* **2008**, *B799*, 136–149. [\[CrossRef\]](#)
2. Freidel, L.; Krasnov, K. A New Spin Foam Model for 4D Gravity. *Class. Quantum Gravity* **2008**, *25*, 125018. [\[CrossRef\]](#)
3. Dona, P.; Gozzini, F.; Sarno, G. Numerical analysis of spin foam dynamics and the flatness problem. *Phys. Rev. D* **2020**, *102*, 106003. [\[CrossRef\]](#)
4. Engle, J.S.; Kaminski, W.; Oliveira, J.R. Addendum to ‘EPRL/FK asymptotics and the flatness problem’. *Class. Quantum Gravity* **2021**, *38*, 119401. [\[CrossRef\]](#)
5. Asante, S.K.; Dittrich, B.; Haggard, H.M. Effective Spin Foam Models for Four-Dimensional Quantum Gravity. *Phys. Rev. Lett.* **2020**, *125*, 231301. [\[CrossRef\]](#) [\[PubMed\]](#)
6. Han, M.; Huang, Z.; Liu, H.; Qu, D. Complex critical points and curved geometries in four-dimensional Lorentzian spinfoam quantum gravity. *arXiv* **2021**, arXiv:2110.10670.
7. Engle, J.; Rovelli, C. The accidental flatness constraint does not mean a wrong classical limit. *arXiv* **2021**, arXiv:2111.03166.
8. Barrett, J.W.; Dowdall, R.J.; Fairbairn, W.J.; Hellmann, F.; Pereira, R. Lorentzian spin foam amplitudes: Graphical calculus and asymptotics. *Class. Quantum Gravity* **2010**, *27*, 165009. [\[CrossRef\]](#)
9. Dona, P.; Speziale, S. Asymptotics of lowest unitary  $SL(2, C)$  invariants on graphs. *Phys. Rev. D* **2020**, *102*, 86016. [\[CrossRef\]](#)
10. Reisenberger, M.P.; Rovelli, C. Sum over surfaces form of loop quantum gravity. *Phys. Rev. D* **1997**, *56*, 3490–3508. [\[CrossRef\]](#)
11. Dona, P.; Fanizza, M.; Sarno, G.; Speziale, S. Numerical study of the Lorentzian Engle-Pereira-Rovelli-Livine spin foam amplitude. *Phys. Rev. D* **2019**, *100*, 106003. [\[CrossRef\]](#)
12. Dona, P.; Sarno, G. Numerical methods for EPRL spin foam transition amplitudes and Lorentzian recoupling theory. *Gen. Relativ. Gravit.* **2018**, *50*, 127. [\[CrossRef\]](#)
13. Frisoni, P.; Gozzini, F.; Vidotto, F. Numerical analysis of the self-energy in covariant LQG. *arXiv* **2021**, arXiv:2112.14781.
14. Sarno, G.; Speziale, S.; Stagno, G.V. 2-vertex Lorentzian spin foam amplitudes for dipole transitions. *Gen. Relativ. Gravit.* **2018**, *50*, 43. [\[CrossRef\]](#)
15. Frisoni, P. Studying the epri spinfoam self-energy. *arXiv* **2021**, arXiv:2112.08528.
16. Bahr, B.; Steinhaus, S. Hypercuboidal renormalization in spin foam quantum gravity. *Phys. Rev. D* **2017**, *95*, 126006. [\[CrossRef\]](#)
17. Allen, C.; Girelli, F.; Steinhaus, S. Numerical evaluation of spin foam amplitudes beyond simplices. *arXiv* **2021**, arXiv:2201.09902.
18. Perez, A. The Spin-Foam Approach to Quantum Gravity. *Living Rev. Relativ.* **2013**, *16*, 3. [\[CrossRef\]](#)
19. Ashtekar, A.; Bianchi, E. A short review of loop quantum gravity. *Rep. Prog. Phys.* **2021**, *84*, 42001. [\[CrossRef\]](#)
20. Rovelli, C.; Vidotto, F. *Covariant Loop Quantum Gravity: An Elementary Introduction to Quantum Gravity and Spinfoam Theory*; Cambridge University Press: Cambridge, UK, 2015.
21. Han, M.; Huang, Z.; Liu, H.; Qu, D.; Wan, Y. Spinfoam on a lefschetz thimble: Markov chain monte carlo computation of a lorentzian spinfoam propagator. *Phys. Rev. D* **2021**, *103*, 84026. [\[CrossRef\]](#)
22. D’Ambrosio, F.; Christodoulou, M.; Martin-Dussaud, P.; Rovelli, C.; Soltani, F. End of a black hole’s evaporation. *Phys. Rev. D* **2021**, *103*, 106014. [\[CrossRef\]](#)
23. Christodoulou, M.; Rovelli, C.; Speziale, S.; Vilensky, I. Planck star tunneling time: An astrophysically relevant observable from background-free quantum gravity. *Phys. Rev. D* **2016**, *94*, 84035. [\[CrossRef\]](#)
24. Gozzini, F.; Vidotto, F. Primordial fluctuations from quantum gravity. *Front. Astron. Space Sci.* **2021**, *7*, 118. [\[CrossRef\]](#)
25. Frisoni, P.; Gozzini, F.; Vidotto, F. Numerical study of the 4-simplex graph refinement with MCMC methods in covariant LQG. **2022, in preparation.**
26. Asante, S.K.; Dittrich, B.; Padua-Arguelles, J. Effective spin foam models for Lorentzian quantum gravity. *Class. Quantum Gravity* **2021**, *38*, 195002. [\[CrossRef\]](#)
27. Plebanski, J.F. On the separation of Einsteinian substructures. *J. Math. Phys.* **1977**, *18*, 2511–2520 [\[CrossRef\]](#)
28. Baez, J.C. An Introduction to Spin Foam Models of BF Theory and Quantum Gravity. *Lect. Notes Phys.* **2000**, *543*, 25–93
29. Holst, S. Barbero’s Hamiltonian derived from a generalized Hilbert-Palatini action. *Phys. Rev. D* **1996**, *53*, 5966–5969 [\[CrossRef\]](#)
30. Kaminski, W.; Kisielowski, M.; Lewandowski, J. Spin-Foams for All Loop Quantum Gravity. *Class. Quantum Gravity* **2010**, *27*, 95006; Erratum in *Class. Quantum Gravity* **2012**, *29*, 049502. [\[CrossRef\]](#)
31. Bianchi, E.; Regoli, D.; Rovelli, C. Face amplitude of spinfoam quantum gravity. *Class. Quantum Gravity* **2010**, *27*, 185009. [\[CrossRef\]](#)
32. Martin-Dussaud, P. A primer of group theory for loop quantum gravity and spin-foams. *Gen. Relativ. Gravit.* **2019**, *51*, 110. [\[CrossRef\]](#)
33. Engle, J.; Pereira, R. Regularization and finiteness of the Lorentzian LQG vertices. *Phys. Rev. D* **2009**, *79*, 84034. [\[CrossRef\]](#)
34. Speziale, S. Boosting Wigner’s nj-symbols. *J. Math. Phys.* **2017**, *58*, 32501. [\[CrossRef\]](#)
35. Gozzini, F. A high-performance code for epri spin foam amplitudes. *Class. Quantum Gravity* **2021**, *38*, 225010. [\[CrossRef\]](#)
36. Anderson, R.L.; Raczka, R.; Rashid, M.A.; Winternitz, P. Clebsch-gordan coefficients for the coupling of  $sl(2, c)$  principal-series representations. *J. Math. Phys.* **1970**, *11*, 1050–1058. [\[CrossRef\]](#)

37. Kerimov, G.A.; Verdiev, I.A. Clebsch-Gordan Coefficients of the  $SL(2, c)$  Group. *Rept. Math. Phys.* **1978**, *13*, 315–326. [\[CrossRef\]](#)

38. Dona, P.; Fanizza, M.; Martin-Dussaud, P.; Speziale, S. Asymptotics of  $SL(2, C)$  coherent invariant tensors. *Commun. Math. Phys.* **2021**, *389*, 399–437. [\[CrossRef\]](#)

39. Dona, P.; Frisoni, P. HowToSpinFoamAmplitude. Available online: <https://github.com/PietropaoloFrisoni/HowToSpinFoamAmplitude> (accessed on 8 February 2022).

40. Dona, P.; Sarno, G. Sl2cfoam. Available online: <https://github.com/qg-cpt-marseille/sl2cfoam> (accessed on 8 February 2022).

41. Johansson, H.T.; Forssén, C. Fast and accurate evaluation of wigner  $3j$ ,  $6j$ , and  $9j$  symbols using prime factorization and multiword integer arithmetic. *SIAM J. Sci. Comput.* **2016**, *38*, A376–A384. [\[CrossRef\]](#)

42. Klib. Available online: <https://github.com/attractivechaos/klib> (accessed on 8 February 2022).

43. Granlund, T. GNU Multiple Precision Arithmetic Library 4.1.2. 2002. Available online: <https://gmplib.org/> (accessed on 8 February 2022).

44. mpfr: A multiple-precision binary floating-point library with correct rounding. RR5753, INRIA. 2005. p.15. ffinria-00070266f. Available online: <https://www.mpfr.org/> (accessed on 8 February 2022).

45. Enge, A.; Gastineau, M.; Théveny, P.; Zimmermann, P. INRIA, 1.1.0 version, 2018; mpc—A Library for Multiprecision Complex Arithmetic with Exact Rounding. Available online: <http://mpc.multiprecision.org/> (accessed on 8 February 2022).

46. Dona, P. Infrared divergences in the EPRL-FK Spin Foam model. *Class. Quantum Gravity* **2018**, *35*, 175019. [\[CrossRef\]](#)

47. Gozzini, F. Sl2cfoam-next. Available online: <https://github.com/qg-cpt-marseille/sl2cfoam-next> (accessed on 8 February 2022).

48. Bezanson, J.; Edelman, A.; Karpinski, S.; Shah, V.B. Julia: A fresh approach to numerical computing. *SIAM Rev.* **2017**, *59*, 65–98. [\[CrossRef\]](#)

49. Blackford, L.S.; Petitet, A.; Pozo, R.; Remington, K.; Whaley, R.C.; Demmel, J.; Dongarra, J.; Duff, I.; Hammarling, S.; Henry, G.; et al. An updated set of basic linear algebra subprograms (blas). *ACM Trans. Math. Softw.* **2002**, *28*, 135–151.

50. Besard, T.; Foket, C.; Sutter, B.D. Effective extensible programming: Unleashing Julia on GPUs. *IEEE Trans. Parallel Distrib. Syst.* **2018**, *30*, 827–841. [\[CrossRef\]](#)

51. Besard, T.; Churavy, V.; Edelman, A.; Sutter, B.D. Rapid software prototyping for heterogeneous and distributed platforms. *Adv. Eng. Softw.* **2019**, *132*, 29–46. [\[CrossRef\]](#)

52. Riello, A. Self-Energy of the Lorentzian EPRL-FK Spin Foam Model of Quantum Gravity. *Phys. Rev. D* **2013**, *88*, 24011. [\[CrossRef\]](#)

53. Frisoni, P.; Gozzini, F. Star Spinfoam Model. Available online: <https://github.com/PietropaoloFrisoni/Star-spinfoam-model> (accessed on 8 February 2022).

54. Fishman, M.; White, S.R.; Stoudenmire, E.M. The ITensor software library for tensor network calculations. *arXiv* **2020**, arXiv:2007.14822.

55. Aitken, A. On Bernoulli's Numerical Solution of Algebraic Equations. *Proc. R. Soc. Edinb.* **1927**, *46*, 289–305. [\[CrossRef\]](#)

56. Aleksandrović, V.D.; Moskalev, A.N.; Kel'manović, K.V. *Quantum Theory of Angular Momentum: Irreducible Tensors, Spherical Harmonics, Vector Coupling Coefficients, 3nj Symbols*; World Scientific: Singapore, 1988.

57. Mäkinen, I. Introduction to  $SU(2)$  Recoupling Theory and Graphical Methods for Loop Quantum Gravity. *arXiv*, **2019** arXiv:1910.06821

58. Yutsis, A.P.; Levinson, I.B.; Vanagas, V.V. *Mathematical Apparatus of the Theory of Angular Momentum*; Israel Program for Scientific Translation: Jerusalem, Israel, 1962.

59. Rasch, J.; Yu, A.C.H. Efficient storage scheme for precalculated wigner  $3j$ ,  $6j$  and gaunt coefficients. *SIAM J. Sci. Comput.* **2004**, *25*, 1416–1428. [\[CrossRef\]](#)

60. Dona, P.; Fanizza, M.; Sarno, G.; Speziale, S.  $SU(2)$  graph invariants, regge actions and polytopes. *Class. Quantum Gravity* **2018**, *35*, 45011. [\[CrossRef\]](#)

61. Carmeli, M.; Leibowitz, E.; Nissan, N. *Gravitation: SL(2,C) Gauge Theory and Conservation Laws*; World Scientific: Singapore, 1990.

62. Ruhl, W. *The Lorentz Group and Harmonic Analysis*; W.A. Benjamin, Inc.: New York, NY, USA, 1970.

63. Dao, V.D.; Nguyen, V.H. On the theory of unitary representations of the  $sl(2, c)$  group. *Acta Phys. Hung.* **1967**, *22*, 201–219.

64. Rashid, M. Boost matrix elements of the homogeneous lorentz group. *J. Math. Phys.* **1979**, *20*, 1514–1519. [\[CrossRef\]](#)

65. Basu, D.; Srinivasan, S. A unified treatment of the groups  $so(4)$  and  $so(3,1)$ . *Czechoslov. J. Phys. B* **1977**, *27*, 629–635. [\[CrossRef\]](#)