ECOLE DOCTORALE ED352
"Physique et Sciences de la Matière"
Aix-Marseille Université

Centre de Physique Théorique
Campus de Luminy, Case 907
13288 Marseille cedex 9, France

Thèse présentée pour obtenir le grade universitaire de docteur

Discipline : Physique Fondamentale et Appliquée
Spécialité : Physique Théorique et Mathématique

François COLLET

Sous la direction de C. ROVELLI
et la co-direction de S. SPEZIALE

Short scale study of 4-simplex assembly with curvature, in euclidean Loop Quantum Gravity

Abstract

A study of symmetrical assembly of three euclidean 4-simplices using classical, Regge and quantum geometry. We analyze the geometric properties and especially the presence of curvature. We show that classical and Regge's geometry of the assembly have curvature, which evolves in function of its boundary parameters. Concerning quantum geometry, an euclidean version of the EPRL model was used with a convenient value of the Barbero-Immirzi parameter to define the transition amplitude of the assembly and its components. A C++ code was designed (Annexes B) to compute the amplitudes and numerically study the quantum geometry. We show that classical geometry, with curvature, emerges already at low spin. We also identified the appearance of degenerate configurations and their effects on the expected geometry.

Résumé français

Une étude d'un assemblage symétrique de trois 4-simplex en géométrie classique, de Regge et quantique. Nous étudions les propriétés géométriques et surtout la présence de courbure. Nous montrons que les géométries classique et de Regge de l'assemblage ont une courbure qui évolue en fonction de ses paramètres de bordure. Pour la géométrie quantique, une version euclidienne du modèle EPRL est utilisé avec une valeur pratique du paramètre Barbero-Immirzi pour définir l'amplitude de transition de l'ensemble et de ses composants. Un code C ++ est conçu (Annexes B) pour calculer les amplitudes et étudier numériquement la géométrie quantique. Nous montrons qu'une géométrie classique, avec une courbure, émerge déjà à bas spin. Nous reconnaissons également l'apparition de configurations dégénérées et de leurs effets sur la géométrie attendue.

Remerciement

Longue et difficile a été cette aventure, et je pense affirmer que je ne serais pas arrivé au bout sans le soutien infaillible des personnes qui me sont chères.

Je tiens à remercier tout d'abord Carlo Rovelli et Simone Speziale, qui m'ont offert l'opportunité inestimable de pouvoir étudier et contribuer au monde de la gravitation quantique. Merci à eux pour leurs conseils et soutien.

Je tiens également à remercier Aurélien Barrau et Karim Noui qui m'ont fait l'honneur d'être mes rapporteurs, ainsi que Etera Livine et Francesca Vidotto qui ont acceptés de faire partie de mon jury de soutenance.

Bien évidemment je remercie chaleureusement mes amis Sylvain, Wilfried, Julie, Thomas, Mathieu, Christian et ma famille qui, même dans les moments difficiles, ne m'ont jamais fait défaut et ont toujours été là pour moi. A vous qui m'avez tant donné, et à qui je suis à jamais redevable, je vous suis dévoué.

J'aurais voulu que certaines personnes soient présentes, et même si la vie nous a séparé je tiens à les remercier également pour les brefs instants que j'ai pu partager avec eux. Même si je ne les reverrai jamais, ils resteront à jamais dans ma mémoire et dans mon âme.

Merci à vous. Merci à vous tous... du fond du coeur.

Synthèse

Introduction

Dans cette thèse nous nous somme intéressé à l'étude de la théorie de la Gravitation Quantique à Boucles, et plus particulièrement à sa limite semi-classique dans le cadre d'un objet à la géométrie simple possédant classiquement de la courbure. La Gravitation Quantique à Boucles (Loop Quantum Gravity), abrégé LQG, est une théorie de quantification non-perturbative de la gravitation visant à décrire, sous un formalisme quantique approprié, les lois de la gravitation basées sur les principes de la géométrie de l'espace-temps décrit par Einstein au sein de sa théorie de la Relativité Générale. Elle vise ainsi, en tant qu'ébauche, à outrepasser les limites conceptuelles de la Relativité Générale et apporter une nouvelle physique qui permettrait de mieux comprendre les phénomènes gravitationnels à petite échelle : comme les trous noirs, ou les premiers instants de notre univers. L'étude des propriétés de cette théorie, notamment concernant ses géométries et la présence de courbure, s'avère importante pour infirmer ou confirmer sa validité avec la réalité physique de notre monde et, si tel est le cas, d'apporter des réponses sur les grands mystères cosmologique de notre temps : origine de la constante cosmologique, histoire de la naissance de l'univers, etc.

L'étude de cette théorie s'est faite, au sein de cette thèse, via l'étude d'un assemblage très simple de trois "atomes d'espace-temps". Cet assemblage, comparable à une toute petite parcelle d'espace-temps, possède à la fois une interprétation classique en terme d'assemblage d'objets géométriques –trois 4-simplex faisant office d'atomes/pièces de notre ensemble– et une formulation quantique en termes de graphes et d'états de géométrie au sein du modèle EPRL issu de la LQG :

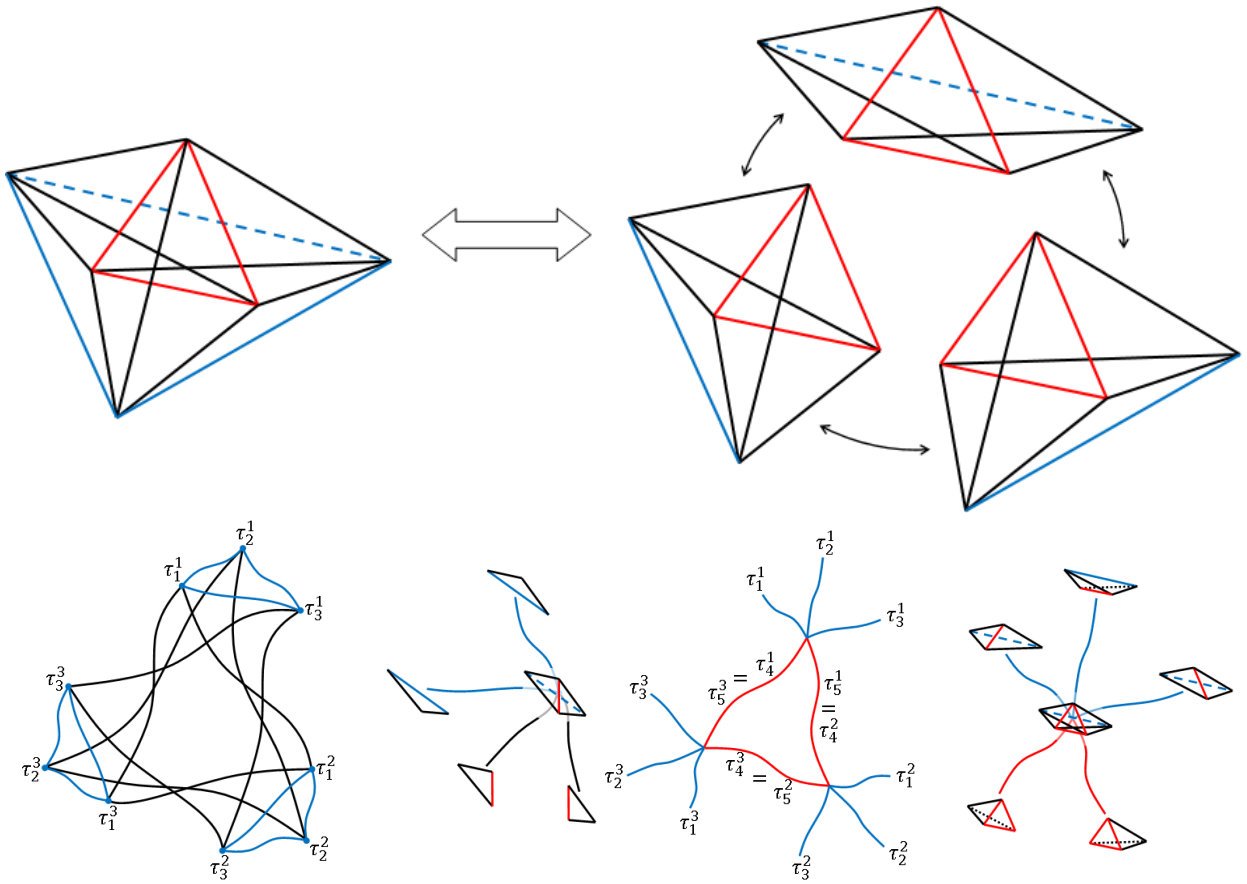


Figure 1: En premier, la représentation classique de notre assemblage en termes de 4-simplex et d'arrêtes. Ensuite, les graphes associés à la représentation quantique : Le réseau de spin, chaque lien est le dual d'une face de la bordure et chaque noeud et le dual d'un tétraèdre de la bordure ; La mousse de spin, chaques lignes representent un tetrahèdre (les bleues sont ceux de bordure, les rouges sont ceux internes) et chaque vertex represente un 4-simplex.

Le but principal est d'étudier la dynamique quantique avec courbure de cet assemblage et de la comparer à sa description classique. Cet assemblage est le cas le plus simple, possédant une représentation classique aisée avec de la courbure et permettant une dynamique de la géométrie quantique via une unique face interne.

Dans l'interprétation classique, notre assemblage peut être étudié via de simple outils de géométrie usuels et possède la propriété d'avoir une courbure qui évolue en fonction de la longueur de ces arrêtes. La présence de cette courbure est très importante car, la gravitation étant décrite par Einstein comme une manifestation de la courbure de l'espace-temps, il est nécessaire de pouvoir l'obtenir pour pouvoir être en adéquation avec physique actuelle. Le fait de pouvoir également transcrire notre objet dans le formalisme de la LQG, via le modèle EPRL, permet ainsi d'en étudier les propriétés, tout en comparant avec son interprétation classique, afin de voir si la courbure est préservée au sein de la théorie et si l'on retrouve des solutions classiques de la géométrie.

Pour cela, on défini les propriétés classiques de cet assemblage et réécrivons les paramètres usuels de la géométrie classique en fonction de paramètres qui s'avéreront utiles pour le formalisme quantique. Puis, afin de commencer à voir si une dynamique de la géométrie existe dans notre assemblage et redonne de la courbure, on adapte notre objet pour y appliquer les principes du calcul de Regge : principe de géométrie discrète, permettant d'associer une action et une dynamique à notre assemblage, et dont la théorie de la LQG possède un lien dans la limite semi-classique. Enfin, nous définissons notre objet dans le cadre de la géométrie quantique, via le modèle EPRL, avec tout ses outils, formalisme d'états (cohérents) de géométrie et amplitudes de transitions. On développe un code pour pouvoir calculer les différentes grandeurs et amplitudes quantiques associés pour, finalement, étudier et

comparer les résultats de la géométrie classique et quantique de notre assemblage.

Géométrie classique

Dans cette première partie, on se concentre à définir notre assemblage dans une interprétation de géométrie (discrète) classique ainsi que ces propriétés en fonction de ces paramètres fondamentaux. Nous mettons en avant une bijection entre les paramètres usuels de géométrie classique, que sont les longueurs des arrêtes, et les paramètres de bordure, que sont les aires et les paramètres de forme associés aux tétraèdres vivant sur la bordure 3d de notre assemblage. Nous exposons simplement les équations et l'évolution de la courbure de notre objet en fonction de ces paramètres :

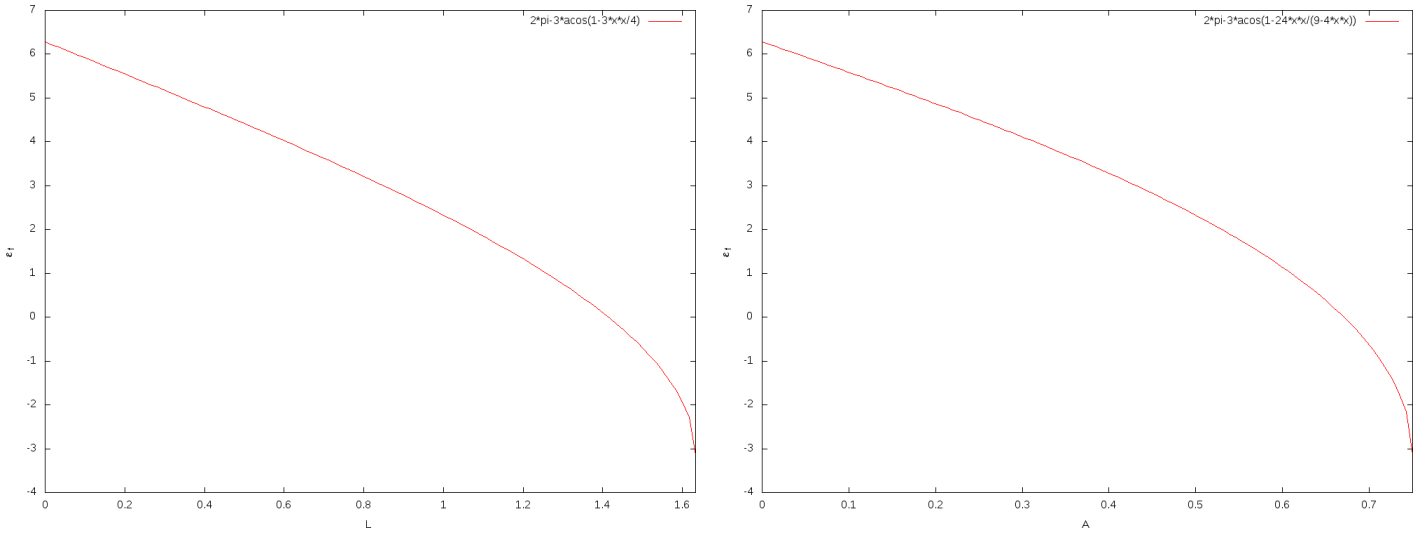


Figure 2: Evolution de la courbure de notre objet en fonction d'un paramètre de longeur (à gauche) et d'un paramètre de forme (à droite)

Nous voyons donc le lien entre la géométrie de la bordure de notre objet, donné par les paramètres de forme, et la géométrie interne de notre objet, tel la courbure, qui en découle. Nous avons donc une description de notre assemblage classique, sans dynamique, ou toute la géométrie peut être décrite en fonction des paramètres de bordure. Notre objet possède dans ce cadre de géométrie de la courbure, et elle évolue continument en fonction des dit paramètres.

Géométrie de Regge

Les géométrie de Regge [1], et plus particulièrement l'action de Regge avec sa dynamique associée, étant une limite semi-classique de la théorie de la LQG s'avéraient intéressantes à étudier. Elles permettraient ainsi de voir une première approche et définition d'une dynamique au sein de notre objet. Nous avons donc adapté, uniquement dans cette partie, notre objet pour voir comment la dynamique de Regge s'effectuait. Nous avons donc observé que le calcul de Regge était parfaitement applicable, et redonnait des solutions avec de la courbure :

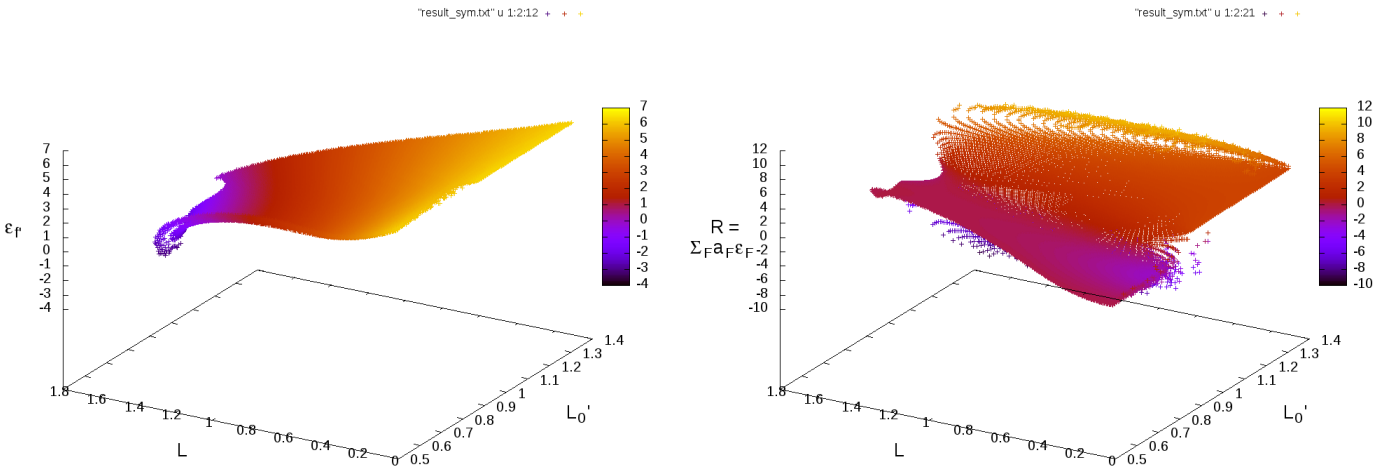


Figure 3: Evolution de la courbure solution des equations de Regge, angle déficient $\varepsilon_{f'}$ de la face interne (à gauche) et courbure de Regge $R = \sum_F a_F \varepsilon_F$ (à droite), en fonction des paramètres de longeurs.

Nous obtenons donc avec succès une dynamique de Regge fonctionnelle qui donne des solutions avec de la courbure ! Ce qui, outre le fait de renforcer les précédentes études tendant à montrer que le calcul de Regge est un bon équivalent des équations d'Einstein dans le cadre des géométries discrètes, donne confiance sur la persistance de la courbure dans la théorie de la LQG.

Géométrie quantique

Fort de ces résultats, nous avons effectué une transcription de notre objet dans le formalisme de la LQG via le modèle EPRL [2, 3]. Nous définissons les différents graphes, ainsi que les états de géométrie qui en découle. Nous définissons également les amplitudes de transitions associées à la géométrie de notre assemblage. Après l'élaboration d'un code C++ (see Annexes B) afin de calculer ces différentes grandeurs, nous avons pu étudier les résultats et les propriétés de la géométrie quantique associée. Même si, à l'image de toute théorie quantique et de leurs principes d'incertitude, une partie de l'information sur la géométrie interne est brouillée, l'étude des amplitudes de transition permit de reconstruire certaines propriétés semi-classique de la géométrie. La recherche des points cols et stationnaires des amplitudes et intégrants permit de trouver les valeurs les plus probables de certains paramètres associés à la géométrie interne. Nous avons donc étudié tout d'abord les propriétés des amplitudes individuelles des 4-simplex de notre assemblage, et retrouvé les géométries classique correspondante tout en retrouvant les résultats annoncé par Barett dans ses écrits [6, 30]. Ensuite, nous nous somme plus particulièrement interessé à l'amplitude de transition totale et ses intégrants où nous avons pu les comparer à leurs équivalent classique attendu :

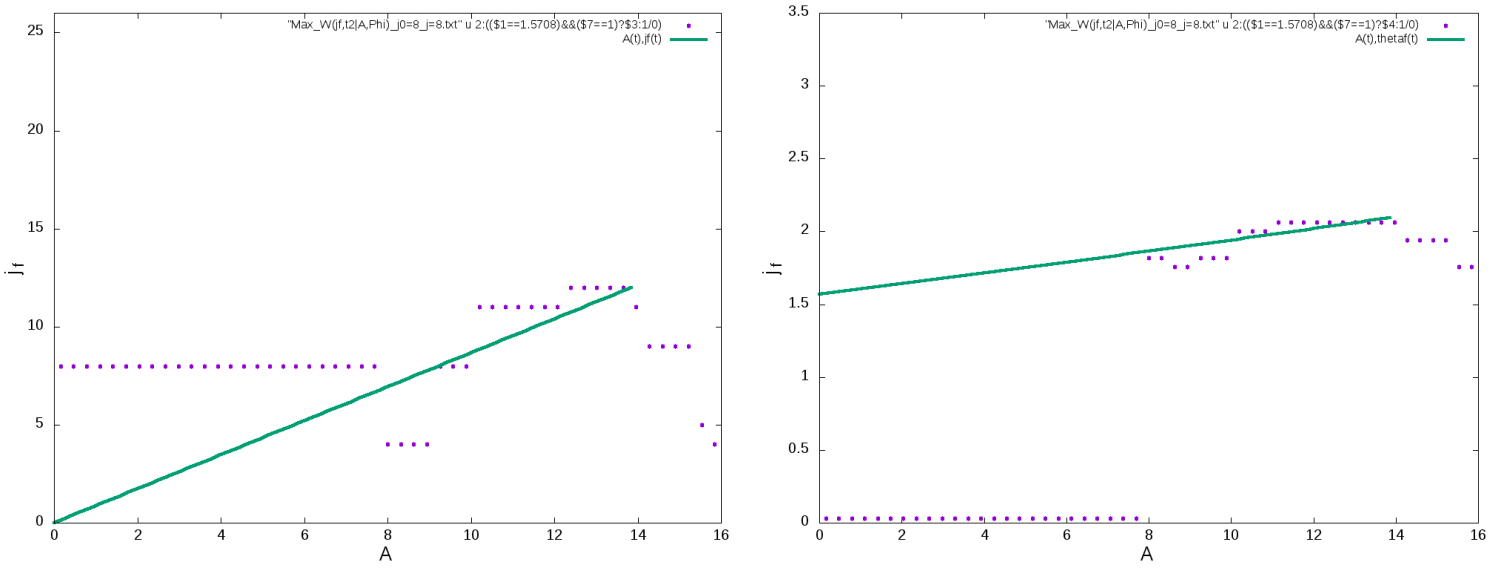


Figure 4: Évolution de l'aire d'une face interne (à gauche) et d'un angle interne (à droite) en fonction d'un paramètre de forme de la bordure. Les points violets sont les résultats fourni par les amplitude de transition de la théorie quantique et les courbes vertes représentent l'évolution des paramètres en géométrie classique.

On observe certaines régions où géométrie classique et quantique sont en accord, essentiellement dans la région où le paramètre de forme $A \in [9.23; 13.86]$, correspondant à des régions où l'équivalent classique possède une courbure $\varepsilon_f \in \left[-\pi; 2\pi - 3\arccos\left(\frac{1}{4}\right)\right]$: ce qui soutient le fait que la géométrie quantique préserve, au moins dans ces plages de valeurs, de la courbure. Cependant, même dans ces régions, l'équivalence n'est pas parfaite ce qui sous-entend que si il y a courbure elle peut être légèrement différente du cas classique. L'absence de définition d'opérateur de courbure nous empêche de statuer définitivement sur la valeur de la courbure dans le cas quantique, mais ces résultats sont un fort indice de l'émergence de la géométrie classique et de ces propriétés au sein de la théorie de la LQG. Nous remarquons aussi une région divergent fortement du cas classique pour $A \leq 7.69$ correspondant à l'influence de géométries purement quantiques.

Conclusion

Nous avons donc au cours de cette thèse fournit une contribution à la compréhension de la théorie de la LQG, et plus particulièrement au modèle EPRL. Nous avons montré que dans le cas d'un petit assemblage, pouvant posséder de la courbure dans l'interprétation classique, la théorie redonnait des résultats compatibles avec la géométrie classique. Nous voyons, dans une certaine région de paramètres, l'émergence de la géométrie classique et de sa courbure associée au sein de la LQG. L'émergence de la géométrie classique avec sa courbure et un bon indice et encouragement mettant en avant les avantages et intérêts de la LQG comme une bonne théorie quantique de la gravitation. Nous voyons cependant une région avec l'apparition de géométrie purement quantique qui diffère fortement de la géométrie classique. Cette dernière région, en raison de sa nature et de l'absence d'équivalent classique, est difficile d'interprétation, mais représente une nouvelle physique apporté avec la théorie apparaissant à faible échelle. Cette nouvelle physique, pourrais être la voie, ou du moins une piste, pour tenter de répondre à certaine questions concernant les trous noirs et les premiers instants de notre monde.

Contents

1 Introduction

Gravitation is a fundamental force of our universe which the quantum description is still unfinished and mysterious. Gravitation governs the large structure of the cosmos, its deterministic formulation in General Relativity describes it as a force from the geometry of space-time and has improved the understanding of our universe and its evolution. The gravitational field is the metric $g_{\mu\nu}$ of space-time, and the interaction between the curvature of space-time and matter is governed by Einstein's equations. General Relativity also lead to the discovery of new physics phenomena such as time dilation and space contraction, the expansion of the universe and cosmic acceleration via the cosmological constant, and peculiar solutions of Einstein's equations leading to the black hole... etc. Even if the theory of General Relativity remains the current best model to describe the world at large scale, many unsolved mysteries remain: The properties of space-time at very small scale, the fundamental sens and origin of the cosmological constant, the evaporation of black holes... are questions which are outside the domain of validity of the classical theory.

With the advent of quantum mechanics, accurately describing the world at small scales and its peculiar properties, that presents new perspectives. Matter is found composed of quanta, namely particles, whose properties escapes the intuition of the classical world. Particles move following probability waves, whose physical properties are described by states that can be superpositioned and are governed by probabilistic laws. Quantum mechanics accurately describes the subatomic world, especially via the uncertainty principle and probabilities, in contradiction with the determinism of General Relativity. A first unification with Special Relativity have yielded the quantum field theory and its applications to the world of particles via the Feynman integral and transition amplitude W. However, gravitation and quantum mechanics should be united in a new theory to draft a model describing the world at small and large scales. This draft will provide a better understanding of the universe that surrounds us, to perhaps explain its origin or at least its first moments during which the universe was both very small and dense, and thus respond to many unanswered questions.

The first approach to describe a discrete space-time was published by T. Regge [1]. He describes, with classic discrete geometry, a space-time cut into simplices associated with a action S_{Regge}. The Regge model discretizes space-time into simplices, equivalent to space-time quanta, whose geometric properties reconstruct its discrete curvature. Variables appear as the length of the simplex segments, and Regge's action depends of the geometry of the assembly. The minimization of this action, via the lengths, provides equations that govern the dynamics of this discrete space-time geometry. For space-time with dimension 3 or lower, Regge's action yields the same physics as General Relativity for vacuum (without matter) in the limit of an infinitesimal discretization. The Regge model and its associated actions are a first conceptual step for quantify space-time geometry and gravity in terms of discrete geometry and space-time quanta. However, the Regge model is not a quantum theory, but just a draft of quantification of geometry with classical objects, i.e., no uncertainties or lost information exists, all geometry is provided perfectly by the set of lengths.

The Loop Quantum Gravity theory [2, 3] involves quantum formalism associated with the concepts of Regge . The Loop Quantum Gravity theory is a nonperturbative quantification of gravity and geometry. According to this theory, General Relativity space-time is divided into quanta, "space-time atoms" like Regge, although with fluctuating, fuzzy and probabilistic geometry such as the particle properties in usual quantum mechanics. The assembly of these atoms, like puzzle pieces, pave and rebuild space-time with curvature. The deficient angles of the assembly provide the curvature [4, 5], while the states of "space-time atoms" enable a quantum description of the assembly's geometry. Formally, the states of geometry are provided in graphs representing the assembly of space-time atoms for which links are associated with group elements u. The representations D^j of the group elements are linked to the physical quantities of the geometry, such as the areas, and yield quantification of the geometry. The mathematical invariants of these graphs, in group representation theory, correspond to the probability of the associated geometries. The limit of invariants used in the theory rebuild an exponentiation of Regge's action. The group elements from the graphs of the Loop Quantum Gravity theory contain information concerning the geometry, while their integrations reconstruct the invariants, equivalent to Feynman integral with Regge's action. In this sense, the Loop Quantum Gravity theory is consistent with Regge's physics, and therefore in a certain limits to the General Relativity, while offering a quantum formulation equivalent of Feynman integrals for geometry:

$$W = \int_{graphs} du \prod D^j(u) \to \int due^{-iS_{Regge}[u]} \sim \int \mathcal{D}g_{\mu\nu} e^{-iS_{GR}[g_{\mu\nu}]} \tag{1}$$

Indeed, a celebrated theorem published by Barrett *et.al.* [6] (see also Conrady and Freidel [7]) states that the vertex amplitude [2, 8, 9, 10] of the Loop Quantum Gravity admits a geometrical interpretation in terms of the geometry of a 4-simplex, in which the properties are determined by the Regge's action of this 4-simplex. This theorem has been extended to the Lorentzian theory [11], to the physical case of positive cosmological constant [12, 13], and is at the basis of many results relating the quantum dynamics of Loop Gravity to classical General Relativity [4, 7, 14, 15, 16], which are at the foundation of the covariant formulation of Loop Quantum Gravity [3]. All these results are derived from the large spin limit, namely under the assumption that the vertex describes (to low order) a process in a region of space-time large compared to the Planck size.

The idea of this thesis was to take the EPRL model of the Loop Quantum Gravity theory, simplified in the case of a 4d euclidean space-time, applied to a simple assembly of three 4-simplices. The assembly presented is complex enough to have curvature and simple enough for analysis in the context of classical geometry, Regge's geometry and, finally, with the euclidean EPRL model. To simplify the analysis, we used a convenient "cylindrical" symmetry for the 4-simplex from the assembly, which renders makes the problem tractable. Geometrically, this corresponds to studying the assembly where the geometry of all 4-simplices is invariant under cyclic permutations of three of their tetrahedra. Thus, the main goal was to analyse the assembly to draw conclusions on the persistence and presence of curvature in the different aspects and theories discussed.

This thesis intermix classical, Regge and quantum geometry with analytic and especially numerical tools, organized as follows. We first present the object of study from a purely classical perspective. In particular, we express its geometry in terms of the natural variables in quantum gravity: the areas of the 2d triangles (corresponding to the spins of Loop Quantum Gravity) and suitable variables to capture the shape of the tetrahedra (corresponding to the intertwiners of Loop Quantum Gravity). Briefly, before using quantum geometry and the EPRL model, we adapted our assembly to study the associated Regge's geometry. We shall see, that the Regge calculus is viable for this 4d Euclidean assembly ; the Regge's equations reproduce curvature. Next, we present the spin-network, spin-foam graphs and express the quantum amplitude of a coherent boundary state with the selected symmetries. The objective was to study the quantum properties of the amplitude and identify its geometric properties via numerical analysis. All results of the transition amplitude studies were using a C++ code (Annexes B) that we designed.

Our analysis and code have three main limitations. First, the analysis involve euclidean domain instead of the physically relevant Lorentzian domain, because the Lorentzian vertices appear to be algebraically more complicated. The euclidean vertices and their assembly can be simply expressed in terms of Wigner $n-j$ symbols, which can be directly handled (numerically). Second, the euclidean theory has an intrinsic difficulty (absent in the Lorentzian one), which is that for generic values of the Barbero-Immirzi parameter γ the simplicity conditions between (discrete) spins cannot be satisfied. We have circumvented this obstacle by choosing $\gamma = 1/2$ and the appropriate closest discrete values for the spins. Finally, we limited our analysis to (Livine-Speziale [20]) boundary states with the chosen "cylindrical" symmetry.

In the wake of previous similar results [17, 18, 19], we found that the mathematically proven results in the limit $j \to \infty$ actually hold true at rather small spin j, namely for vertices representing space-time regions of Planckian size. We found evidence for the emergence of semi-classical geometry behavior already for $j \sim 10$, i.e., an order of magnitude above the Planck scale, as previously described in the article [29], which is to say an order of magnitude above the Planck scale. This might be relevant for instance in cosmology, suggesting that quantum gravitational effects could be limited to regimes very near Planckian densities.

However, we also observed the appearance of genuine quantum phenomena in the numerical result. These are first of all the spread of the amplitude around the classical values, i.e., the Heisenberg uncertainty principle. We also observed the emergence of degenerate geometries, on which we comment in closure.

2 Study object and classical geometry

The assembly studied, see figure 6, it's a assembly of three flat 4-simplex, sharing a same face. Each individual 4-simplex is a 4d-triangulation of flat space bounded by a 3d surface formed by five 3d-flat tetrahedra matching their triangle faces. As a generalization of tetrahedra assemblies, where the tetrahedra are glued face-by-face and share triangular face, each 4-simplex share tetrahedra with their neighbors. In our specific case, each 4-simplex is glued to their two neighbors by sharing two tetrahedra (one tetrahedron per neighbors) and the all shared-tetrahedra

share the same triangle base. For summary, this assembly have 1 special triangle, called f, shared between the three 4-simplices, and is a triangular base of the three shared-tetrahedra which attach the 4-simplex together. By construction, the triangle f is internal (means inside the bulk of the assembly, not in the 3d boundary) and the three shared-tetrahedra are internal. The other triangles of the shared-tetrahedra are not internal, but they belong to the boundary of the assembly. The other triangles and tetrahedra of the assembly shape the boundary.

The properties of the assembly are the following :

- The geometry is given by 15 parameters, which can be taken to be the length of its 15 segments or the triangle areas and shape variables (we see that below).
- The assembly have 3 (internal) shared-tetrahedra with joint base triangle f, and 9 (external) tetrahedra which shape the boundary.
- The assembly have 18 boundary triangles, and 1 internal face f
- The parameters of geometry can give curvature around the internal face f

The big interest of this assembly is the last point : the "size" of the 4-simplices given by the geometry parameters can give curvature around the face f. As described in the article [29], individual tetrahedra geometry can be expressed in terms of triangle area variable a and the shape variable A, Φ. In quantum interpretation of this assembly, for the boundary tetrahedra we will use coherent state $|a,(\Phi,A)\rangle$ peaked on a given shape (Φ,A). The idea is that the boundary quantum geometry can be encoded inside the coherent state $|a,(\Phi,A)\rangle$ of all boundary tetrahedra, which choose a internal geometry for the shared-tetrahedra and the face f with curvature.

2.1 Geometry of a classical tetrahedron

As the articles [20, 5] and [29] for a tetrahedron in 3d flat space with the area a_i of the labeled face $i = 1, 2, 3, 4$, geometry gives :

$$\sum_{i=1}^{4} a_i \overrightarrow{n_i} = 0 \tag{2}$$

where the $\overrightarrow{n_i}$ are the unit vector normal to the face i.

Each vector $\overrightarrow{n_i}$ can be expressed in shared coordinate system by the two S^2 angle parameters $(\theta_i, \phi_i) \in [0,\pi] \times [0, 2\pi]$. And we can define the shape variable Φ, A by the relations :

$$\cos\Phi := -\left(\frac{\overrightarrow{n_1}\wedge\overrightarrow{n_2}}{\|\overrightarrow{n_1}\wedge\overrightarrow{n_2}\|}\right)\cdot\left(\frac{\overrightarrow{n_3}\wedge\overrightarrow{n_4}}{\|\overrightarrow{n_3}\wedge\overrightarrow{n_4}\|}\right) \tag{3}$$

$$A\overrightarrow{n_A} := a_1\overrightarrow{n_1} + a_2\overrightarrow{n_2} = -\left(a_3\overrightarrow{n_3} + a_4\overrightarrow{n_4}\right) \tag{4}$$

Where $\overrightarrow{n_A}$ is just a unit vector.

The parameters, with a arbitrary orientation (gauge choice), can be computed from the six parameters (a_i, A, Φ). For example take the orientation (or "gauge") where $(\theta_1, \phi_1) = (0, 0)$ and $\phi_2 = 0$, the others parameters (θ_i, ϕ_i) are just given by the relations (3) and (4) (See the Annexes A for the full equations).

For the next, where we will have a lot of tetrahedra which share the same faces, it will be useful to take a more appropriate notation to distinguish the areas "a" and their normal vectors "$\overrightarrow{n}$". So for a specific tetrahedron called τ_k we will use the notation a_{kl} for the area of the face shared with a another tetrahedra τ_l. Of course, the notations a_{kl} and a_{lk} are equivalent and represent the same shared area between the tetrahedra τ_k and τ_l. For the normal vectors "$\overrightarrow{n}$" we need to be more precise about the indices : the vector $\overrightarrow{n_{kl}}$ will represent the vector of the tetrahedron τ_k, normal to the face of area a_{kl}, outgoing of τ_k, but in-going the next tetrahedron τ_l. Conversely, the vector $\overrightarrow{n_{lk}}$ represent the vector of the tetrahedron τ_l, normal to the same shared area a_{kl}, but outgoing of τ_l and in-going to τ_k. The vectors $\overrightarrow{n_{kl}}$ and $\overrightarrow{n_{lk}}$ are independent, because the $\overrightarrow{n_{kl}}$ is only define in the 3d-frame of the tetrahedron τ_k and the $\overrightarrow{n_{lk}}$ only in the 3d-frame of the tetrahedron τ_l. Only when the tetrahedra τ_k and τ_l are defined in the same 3d-frame, means the two tetrahedra are glued by the face-area a_{kl} in a 3d-flat assembly, we have the relation $\overrightarrow{n_{kl}} = -\overrightarrow{n_{lk}}$. With this notation the closure condition for a classical tetrahedron τ_k become just :

$$\sum_l a_{kl}\overrightarrow{n_{kl}} = a_{kl_1}\overrightarrow{n_{kl_1}} + a_{kl_1}\overrightarrow{n_{kl_1}} + a_{kl_3}\overrightarrow{n_{kl_3}} + a_{kl_4}\overrightarrow{n_{kl_4}} = 0 \tag{5}$$

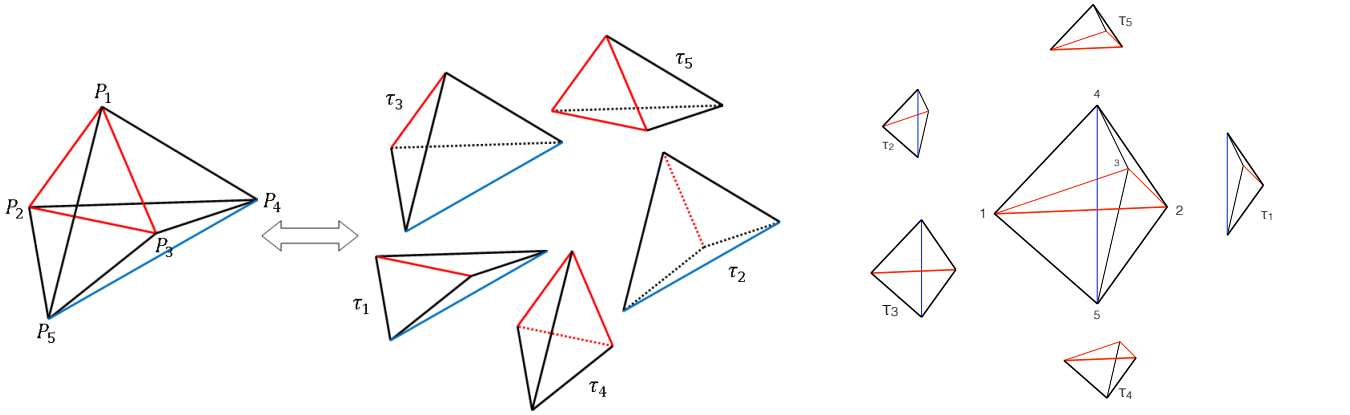
Where we have the sum over the four tetrahedra τ_l which share a face with τ_k. By definition, each unit vector $\overrightarrow{n_{kl}}$ can be expressed by the two S^2 angle parameters (θ_{kl}, ϕ_{kl}) in the shared coordinate system associated to the 3d-frame of the tetrahedron τ_k. And, from the closure condition of tetrahedron τ_k, we have the associated shape variables (A_k, Φ_k) which are fixed by their usual definition given above :

$$\cos\Phi_k = -\left(\frac{\overrightarrow{n_{kl_1}} \wedge \overrightarrow{n_{kl_2}}}{\|\overrightarrow{n_{kl_1}} \wedge \overrightarrow{n_{kl_2}}\|}\right) \cdot \left(\frac{\overrightarrow{n_{kl_3}} \wedge \overrightarrow{n_{kl_4}}}{\|\overrightarrow{n_{kl_3}} \wedge \overrightarrow{n_{kl_4}}\|}\right) \tag{6}$$

$$A_k \overrightarrow{n_{A_k}} = a_{kl_1}\overrightarrow{n_{kl_1}} + a_{kl_2}\overrightarrow{n_{kl_2}} = -\left(a_{kl_3}\overrightarrow{n_{kl_3}} + a_{kl_4}\overrightarrow{n_{kl_4}}\right) \tag{7}$$

2.2 Cylindrical symmetries for the 4-simplices and fundamental parameters

The large number of variables makes the problem hard to analyze. To simplify we will impose symmetry restrictions. The three 4-simplices used will be identical, and will have a cylindrical symmetry around the face f. With cylindrical symmetry, as in Figures 5a,5b, each 4-simplex can be seen with their two shared-tetrahedra with the same equilateral base f representing a "invariant plane" of symmetry, and the three other tetrahedra (belonging to the boundary of the assembly) are around the "invariant plane" of f. The cylindrical 4-simplex is invariant under the discrete rotation which preserve f, and transform by cyclic permutation the boundary tetrahedra to each others. For more convenience, we call τ_k^N the tetrahedron labeled k inside the 4-simplex labeled N. And for all individual 4-simplex N, we will take $k = 4, 5$ for the shared-tetrahedra and $k = 1, 2, 3$ for the boundary tetrahedra. In the 4-simplex N, we can also define P_k^N the vertex opposite to the tetrahedron τ_k^N, and define L_{kl}^N the length of the segment joining P_k^N and P_l^N.



(a) The cylindrical 4-simplex and its exploded view in tetrahedra

(b) Another view of the cylindrical 4-simplex where we can see the invariant plane of f (given by the red segments) under the discrete rotation which transform by cyclic permutation τ_1 to τ_2 and τ_3.

Figure 5: Presentation of the cylindrical symmetric 4-simplex

The cylindrical symmetric 4-simplices, and the condition where the three 4-simplices inside the assembly are identical, impose that the full geometry is entirely given by three lengths :

$$L_f := L_{12}^N = L_{23}^N = L_{31}^N \ , \ L_0 := L_{14}^N = L_{24}^N = L_{34}^N = L_{15}^N = L_{25}^N = L_{35}^N \ , \ L := L_{45}^N \ \forall N \tag{8}$$

respectively red, black and blue in Figures 5a,5b.L_f is just the length of the triangle f segments, L_0 is the length of the segments which shape the shared-tetrahedra faces on f, and L is the length which connect the top of shared-tetrahedra and close the 4-simplices. Also, the segments of length L form a equator around the internal face f on the assembly's boundary. As the following Figure :

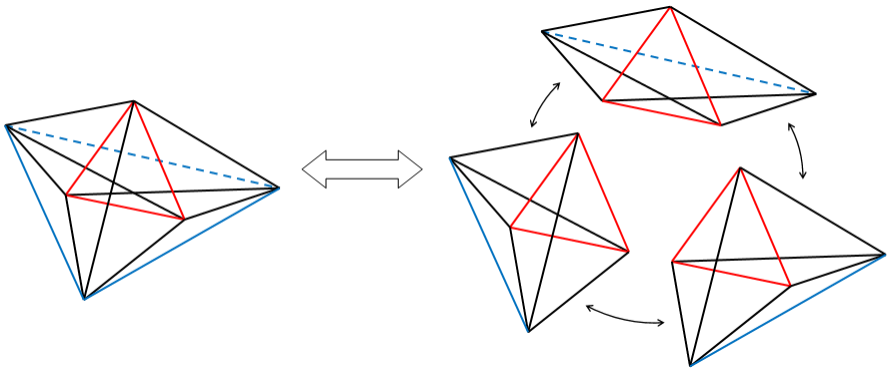


Figure 6: In the left, the study object. In the right, the exploded view of the assembly of the 4-simplices.

The set of the length (L, L_0, L_f) determinate the geometry of the 4-Simplices and, therefore, determinate the full geometry of the assembly with the cylindrical symmetry. L, L_0 and L_f belong to the boundary of the assembly, and are the fundamental parameters of the geometry in the Regge sens, we will come back about that in the Regge section for talk about the dynamics of the assembly geometry.

For the quantum section, we need to adapt the fundamental parameters to the area and shape variables. Calling a^N_{kl} the area of the triangle opposite to the segment kl, separating the tetrahedra τ^N_k and τ^N_l. The symmetries impose :

$$a^N_{12} = a^N_{23} = a^N_{31} \equiv a \ , \ a^N_{14} = a^N_{24} = a^N_{34} = a^N_{15} = a^N_{25} = a^N_{35} \equiv a_0 \ , \ a^N_{45} \equiv a_f \ \forall N \tag{9}$$

We have three sort of faces : the isosceles triangles given by two (black) segments of length L_0 and one (blue) segment of length L with the area a ; the isosceles triangles given by two (black) segments of length L_0 and one (red) segment of length L_f with the area a_0; and the equilateral triangle f given by the segments of length L_f with the area a_f. We will respectfully call these faces : f_a, f_0 and (obviously) f. Note, we will call sometimes these faces : a-faces, a_0-faces, a_f-face or j-faces, j_0-faces, j_f-face ; for remember explicitly the area or the associated quantum number of the face in the classical or quantum sections. The faces f_a and f_0 belong to the boundary of the assembly and the face f is internal (only the face is inside the bulk, because their segments belong to the boundary). Of course, the different areas can be expressed in terms of their segments length :

$$\begin{array}{rcl} a & \equiv & a\left(L, L_0\right) = \frac{L}{2}\sqrt{L_0^2 - \frac{L^2}{4}} \\ a_0 & \equiv & a_0\left(L_f, L_0\right) = \frac{L_f}{2}\sqrt{L_0^2 - \frac{L_f^2}{4}} \\ a_f & \equiv & a_f\left(L_f\right) = \frac{\sqrt{3}}{4}L_f^2 \end{array} \tag{10}$$

Remember that the geometry of individual tetrahedra can be also express in terms of triangle areas and shape variables. The geometry of boundary tetrahedra can be given by a, a_0 and their shape variables $\left(\Phi^N_k, A^N_k\right)_{k=1,2,3}$. The geometry of shared-tetrahedra can be given by a_0, a_f and their shape variables $\left(\Phi^N_k, A^N_k\right)_{k=4,5}$. Always in the context of the cylindrical symmetries [29], the three area a, a_0, a_f can be use to determine the geometry of the 4-simplices and, by assembly constraints, the geometry of the full assembly. Because the cylindrical symmetries implies :

$$\Phi^N_k = \frac{\pi}{2} \ \forall N, k \tag{11}$$

$$A^N_1 = A^N_2 = A^N_3 \equiv A \ \forall N \tag{12}$$

$$A^N_4 = A^N_5 \equiv A_f \ \forall N \tag{13}$$

The geometry of the shared-tetrahedra gives :

$$A_f = \frac{L_0 L_f}{2} = \sqrt{a_0^2 + \frac{1}{3}a_f^2} \tag{14}$$

And the assembly constraints (meaning matching faces of tetrahedra and closure condition for individual euclidean 4-simplex) gives :

$$A = \frac{LL_f}{2} = \sqrt{2A_f^2 - 2\sqrt{A_f^4 - \frac{4}{3}a^2a_f^2}} \tag{15}$$

Note, the formula for $A_f\left(a_0, a_f\right)$ and $A\left(a, a_0, a_f\right)$ are extended for the all variable set $a_f \in [0; 3a_0]$ but give only classical and physical geometric 4-simplex for $A \in \left[0; \min\left(2a_0, a\sqrt{3}\right)\right]$.

In this sens, with the symmetries and the assembly constraints, the set (a, a_0, a_f) is also a set of fundamental parameters of the full geometry of the assembly. We have a bijection between the fundamental length parameters and the fundamental areas parameters, that can be use for restore the lengths and the shape variables :

$$(a, a_0, a_f) \Leftrightarrow (L, L_0, L_f) \Rightarrow (A, A_f) \tag{16}$$

$$\begin{array}{rcl} L_f^2 & = & \frac{4}{\sqrt{3}}a_f \\ L_0^2 & = & \frac{3a_0^2 + a_f^2}{\sqrt{3}a_f} \\ L^2 & = & 2L_0^2 - 2\sqrt{L_0^4 - 4a^2} \end{array} \tag{17}$$

Alternatively, if you give the parameters (a, a_0, A), called the boundary parameters, you can compute a_f by the classical constraint from geometry :

$$A^2\frac{4a_0^2 - A^2}{4a^2 - A^2} = \frac{4}{3}a_f^2 \tag{18}$$

And also compute the full geometry :

$$\begin{array}{rcl} L_f^2 & = & 2A\sqrt{\frac{4a_0^2 - A^2}{4a^2 - A^2}} \\ L_0^2 & = & \frac{16a^2a_0^2 - A^4}{2A\sqrt{4a_0^2 - A^2}\sqrt{4a^2 - A^2}} \\ L^2 & = & 2A\sqrt{\frac{4a^2 - A^2}{4a_0^2 - A^2}} \end{array} \tag{19}$$

That will be important for the quantum section, because we expect that the coherent state of the assembly's boundary(a, a_0, A) will fix the value of a_f and the geometrical properties of the internal face f.

In summary, the classical geometry give for our assembly with cylindrical symmetries three equivalent set of parameters :

$$(L, L_0, L_f) \Leftrightarrow (a, a_0, a_f) \Leftrightarrow (a, a_0, A) \tag{20}$$

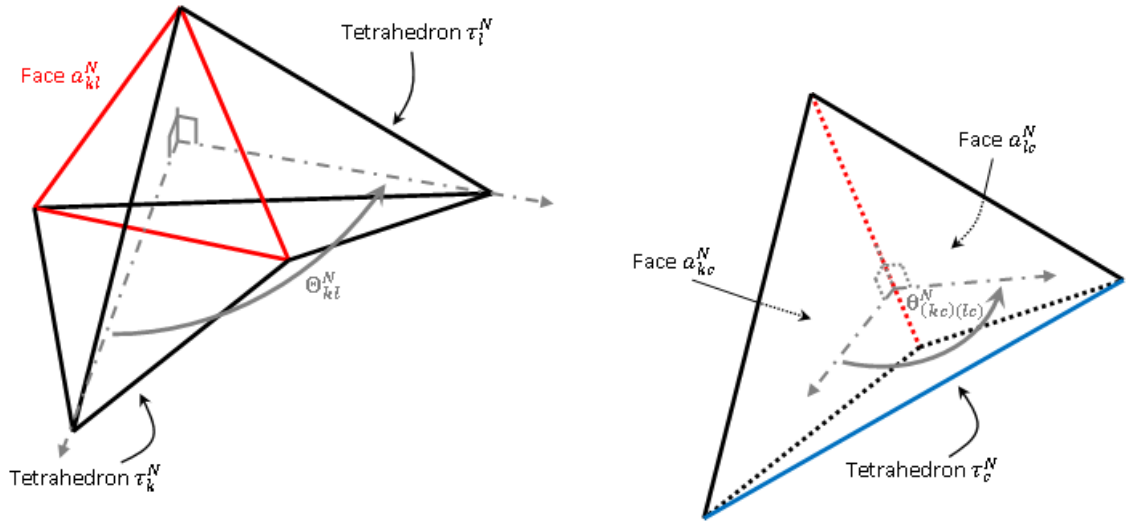
These are three equivalent classical descriptions of the geometry ; the last one is the most appropriate for the quantum theory.

2.3 Curvature of the f triangle

The most important thing about our study object, it's the presence of curvature with the classical interpretation. When you assembles the 4-simplices around the same triangle f and close the assembly, you create a deficit angle and curvature around f. And the value of the deficit angle evolve with the size of the equator (defined by the L segments) compared to the rest of the geometry. Inside our individual 4-simplices, the 4d-space is euclidean and flat, so we can compute the angle Θ_{kl}^N between two tetrahedra τ_k^N, τ_l^N along the sharing face a_{kl}^N, see Figure 7a. The definition of the Θ_{kl}^N is :

$$\cos\Theta_{kl}^N = \frac{\cos\theta_{(kc)(lc)}^N - \cos\theta_{(kl)(kc)}^N\cos\theta_{(kl)(lc)}^N}{\sin\theta_{(kl)(kc)}^N\sin\theta_{(kl)(lc)}^N} \ \forall c \tag{21}$$

Where $\theta_{(kc)(lc)}^N$ is the (dihedral) angle between the faces a_{kc}^N and a_{lc}^N. We can see visual representations of these angles below :



(a) Representation of the angle Θ^N_{kl} between the tetrahedra τ^N_k, τ^N_l of same basis a^N_{kl} inside the 4-simplex N.

(b) Representation of the dihedral angle $\theta^N_{(kc)(lc)}$ between the faces a^N_{kc}, a^N_{lc} inside the tetrahedron τ^N_c (belonging to the 4-simplex N).

Figure 7: Angles defined from the 4-Simplex geometry.

In our symmetric case, where $kl = 45$ correspond to the f triangle, $kl = 1, 2, 3$ correspond to the f_a triangles, and the others kl correspond to the f_0 triangles, we have only three sorts of Θ angles : Θ_f, Θ_{f_a} and Θ_{f_0}. With the symmetries of the assembly the deficit angle associated to the triangle f is just :

$$\varepsilon_f = 2\pi - 3\Theta_f \tag{22}$$

and we can express all the Θ angles in the three set of fundamental parameters :

- In terms of fundamental lengths (L, L_0, L_f) :

$$\cos\Theta_f = 1 - \frac{3}{2} \cdot \frac{L^2}{3L_0^2 - L_f^2} \tag{23}$$

$$\cos\Theta_{f_a} = \frac{1}{2}\left(1 - \frac{L_f^2}{4L_0^2 - L^2 - L_f^2}\right) \tag{24}$$

$$\cos\Theta_{f_0} = \frac{LL_f}{2\sqrt{3L_0^2 - L_f^2}\sqrt{4L_0^2 - L^2 - L_f^2}} \tag{25}$$

that will be useful for understand the Regge's geometries, and it's the most simple way for see the evolution of the curvature in function of the geometry. For the example $(L_0, L_f) = (1, 1)$, we can draw the deficit angle ε_f in function of the equator length L :

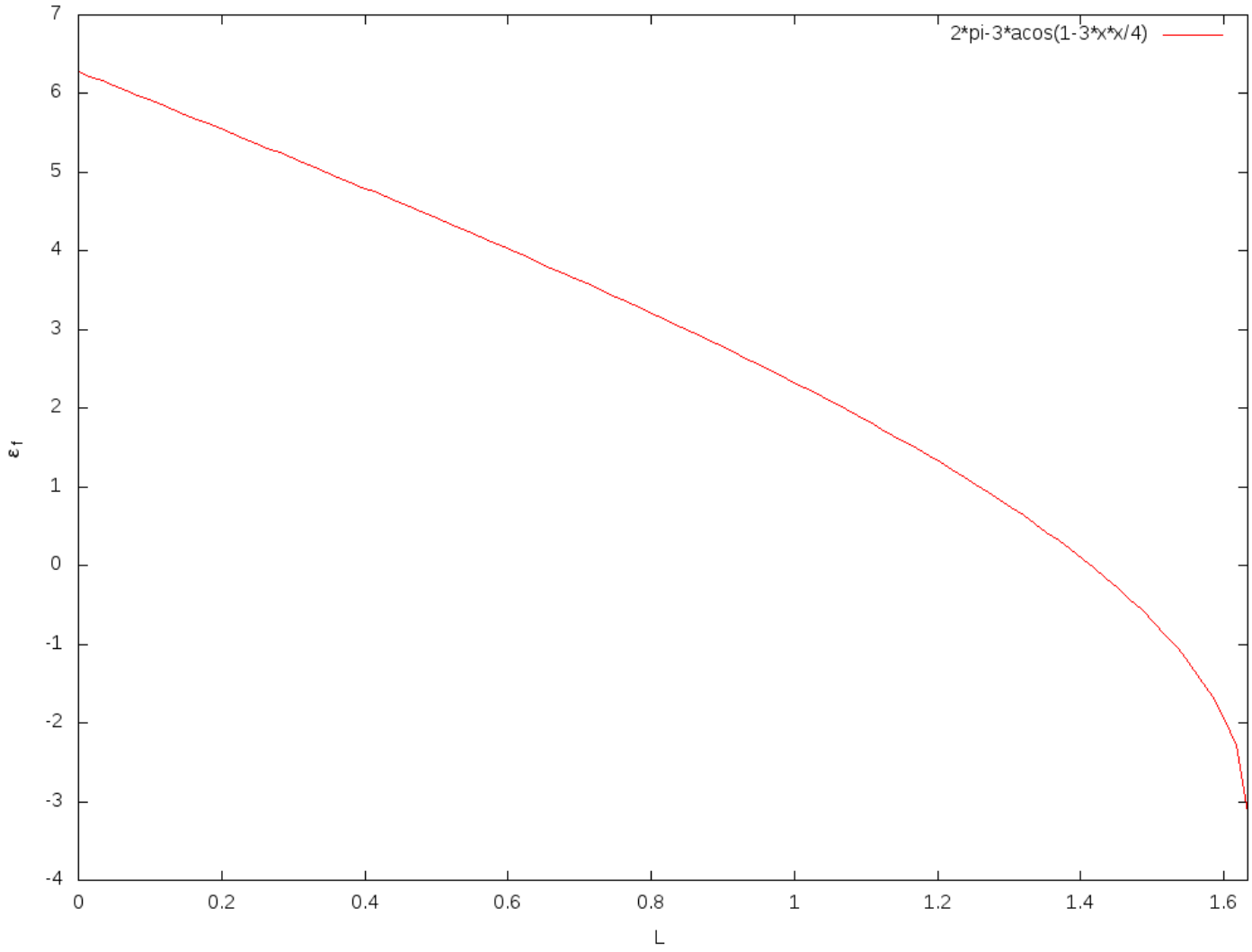


Figure 8: Curvature in function of length L for $L_0 = L_f = 1$

- In terms of fundamental areas (a, a_0, a_f) :

$$\cos\Theta_f = \frac{-4a_f^2 + 3\sqrt{\left(3a_0^2 + a_f^2\right)^2 - 12a^2a_f^2}}{9a_0^2 - a_f^2} \tag{26}$$

$$\cos\Theta_{f_a} = 1 - \frac{3a^2}{6a^2 - 3a_0^2 - a_f^2 + \sqrt{\left(3a_0^2 + a_f^2\right)^2 - 12a^2a_f^2}} \tag{27}$$

$$\cos\Theta_{f_0} = \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{9a_0^2 - a_f^2}} \cdot \frac{3a_0^2 + a_f^2 - \sqrt{\left(3a_0^2 + a_f^2\right)^2 - 12a^2a_f^2}}{\sqrt{6a^2 - 3a_0^2 - a_f^2 + \sqrt{\left(3a_0^2 + a_f^2\right)^2 - 12a^2a_f^2}}} \tag{28}$$

that will be useful for connect the classical interpretation with the quantum equivalent where the fundamental parameters are the areas. For $(a_0, a_f) = \left(\frac{\sqrt{3}}{4}, \frac{\sqrt{3}}{4}\right)$, we can draw the deficit angle ε_f in function of the area a :

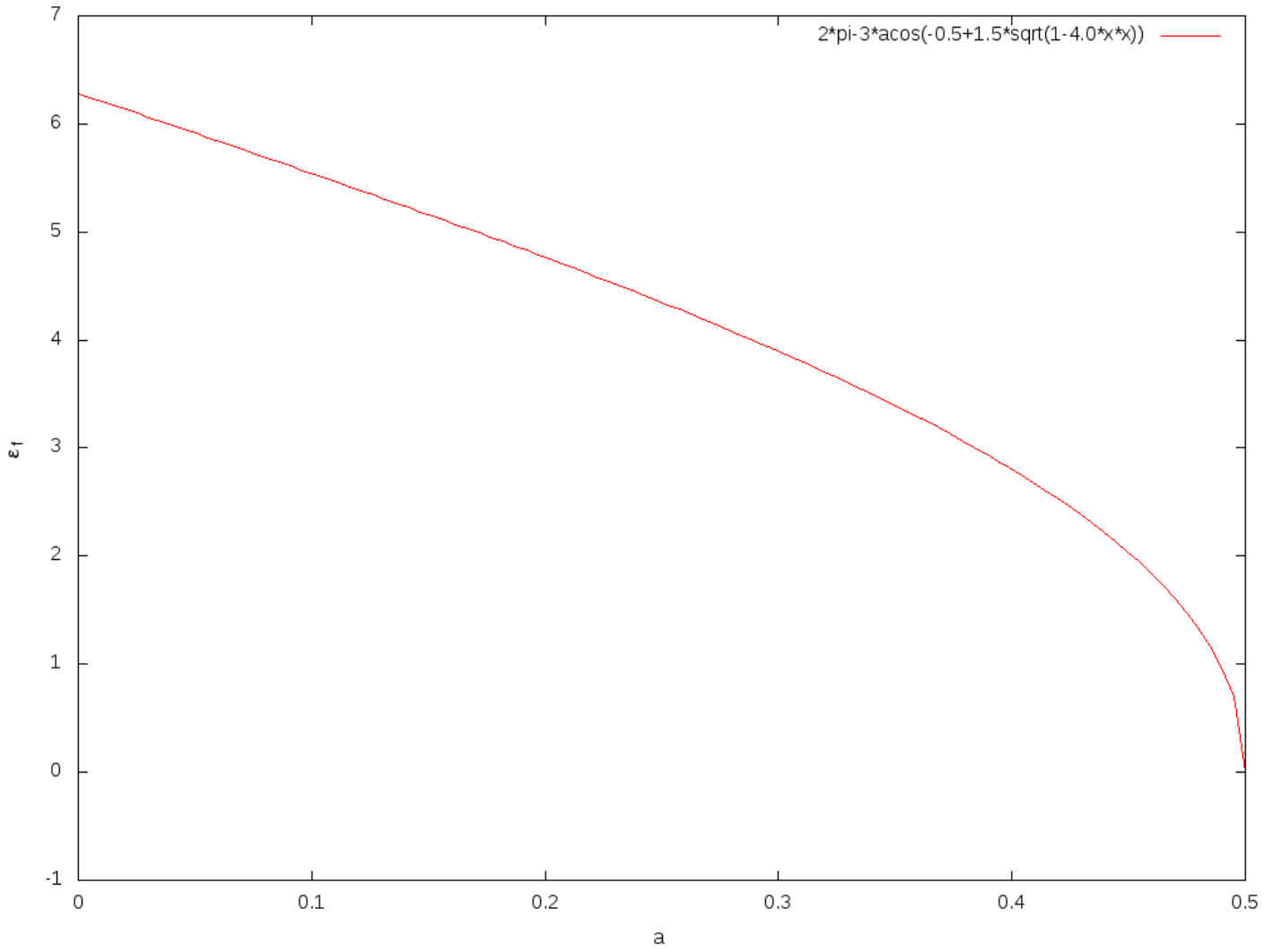


Figure 9: Curvature in function of area a for $a_0 = a_f = \frac{\sqrt{3}}{4}$

- In terms of boundary parameters (a, a_0, A) :

$$\cos\Theta_f = 1 - \frac{6A^2\left(4a^2 - A^2\right)}{12a_0^2\left(4a^2 - A^2\right) - A^2\left(4a_0^2 - A^2\right)} \tag{29}$$

$$\cos\Theta_{f_a} = \frac{1}{2}\left(1 - \frac{A^2}{4a^2 - A^2}\right) \tag{30}$$

$$\cos\Theta_{f_0} = \frac{A^2}{\sqrt{12a_0^2\left(4a^2 - A^2\right) - A^2\left(4a_0^2 - A^2\right)}} \tag{31}$$

that will be useful for understand the quantum case where the curvature evolve only with the boundary parameters given by coherent states. That means, if you fix the boundary geometry, by coherent state which give (a, a_0, A),

you will allow curvature inside the assembly. For $(a_0, a) = \left(\frac{\sqrt{3}}{4}, \frac{\sqrt{3}}{4}\right)$, you can see the evolution of deficit angle ε_f in function of A :

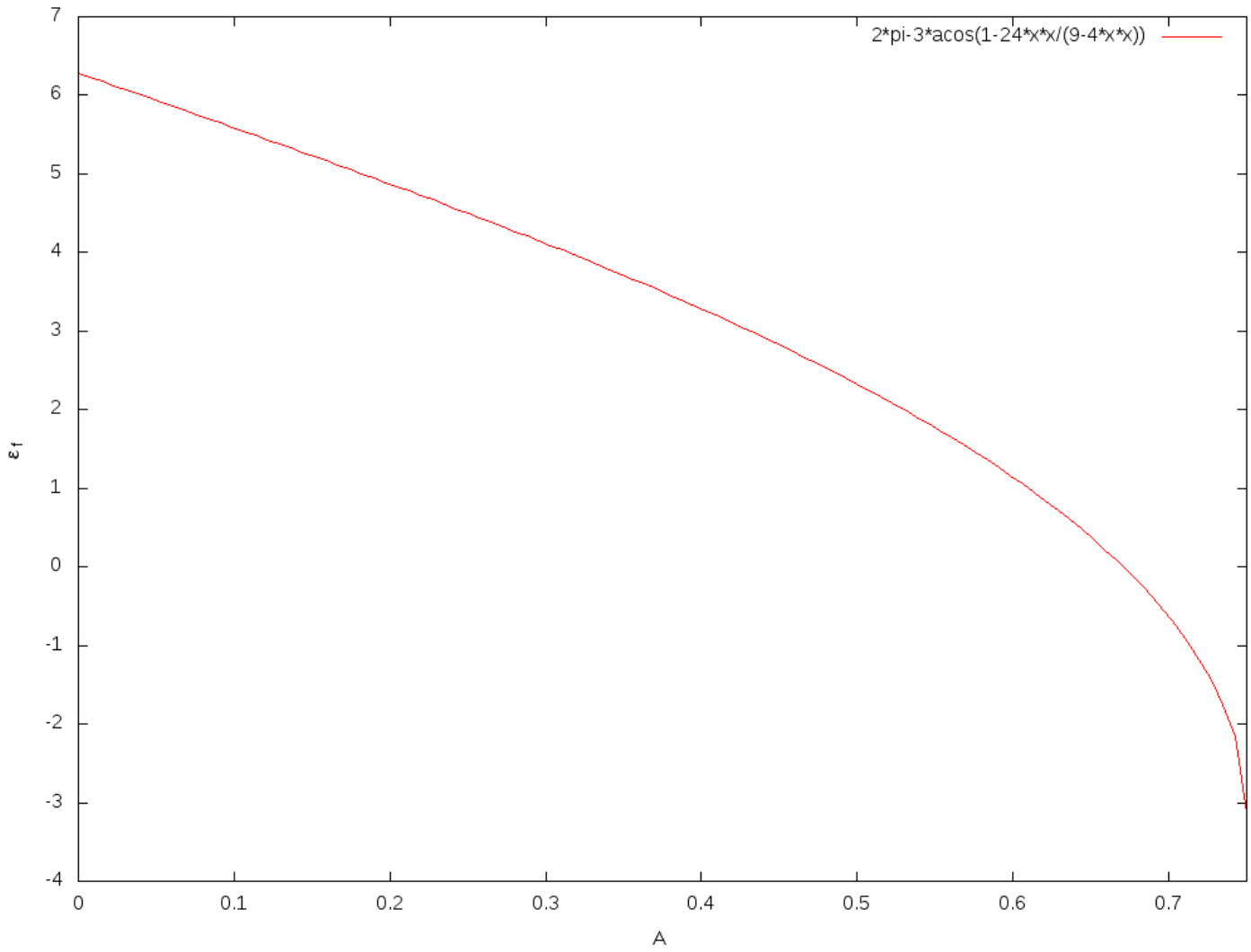


Figure 10: Curvature in function of A for $a_0 = a = \frac{\sqrt{3}}{4}$

Of course in the three descriptions, when the deficit angle ε_f is null we have a 4-dimentional flat assembly, if ε_f is positive we have positive curvature and when ε_f is negative we have negative curvature. For example, if we take the most regular case where the all lengths are equal (equivalent to have the all areas equal), the angle Θ_f give $\cos\Theta_f = \frac{1}{4}$ and the deficit angle is $\varepsilon_f = \arccos\left(-\frac{11}{16}\right) \approx 2.329$ which correspond to a positive curvature. And if we have the relation (written in the three set of variables) :

$$-3L_0^2+L^2+L_f^2=0 \;\Leftrightarrow\; 3\left(a_f^2-a_0^2\right)=2\sqrt{\left(3a_0^2+a_f^2\right)^2-12a^2a_f^2} \;\Leftrightarrow\; 4\left(4a^2-A^2\right)\left(4a_0^2-A^2\right)=16a_0^2a^2-A^4 \quad (32)$$

we have a null deficit angle and no curvature. That relation can be seen like the flatness condition of the assembly.

3 Regge's geometry with dynamics

The study of our assembly in classical geometry show us the interest of this assembly in term of simplicity and curvature : the study object (in the context of this symmetries) have only three parameters for describe its full geometry and have a curvature easily expressible. It will be interesting to see if its curvature it preserved in the context of Regge calculus. Regge calculus is a simplest way, find by T. Regge [1], for associate a physics action for a discretized geometry. The main idea of Regge, in its simplest example and interpretation, is to discretizes the space-time in simplices and associate a action given by the areas and the deficit angles of this discretization. The fundamental parameters of the Regge's action will be the length of the "skeleton bones" of the discretization : in other words, the length of the segments of the simplices. The Regge's action give the classical solution of the geometry by its minimization along the segments length ; this process is called Regge calculus. The length of segments from Regge calculus will be the fundamental variables of the geometry like the metric for the Einsteinian space-time, and the Regge's equations, from the minimization of the Regge's action by the lengths, are the equivalent of Einstein's equations from General Relativity. And for a 3d-space-time which is infinitesimally discretized, where the lengths and size of the simplices used for discretization tends to 0, the Regge calculus give exactly the same physics of General Relativity theory. In this sens, the Regge calculus is a strong equivalent of General Relativity for discretized space-time where the Regge's action is the discretized equivalent of Einstein-Hilbert's formulation. For a quantum theory of gravitation and space-time, the Regge formulation is a very interesting starting point for understand and formalize a discretized space-time in quantas that are the simplices. Moreover, the Loop Quantum Theory is linked over a group formulation of Regge principle where the transition amplitude reproduce, in a certain limit, the complex exponentiation of the Regge's action.

Our goal in this section is to see if the Regge is viable for our study object and give also curvature. The problem here it's that our study object does not have Regge's dynamics, because the all segments belong to the boundary and the Regge calculus is only viable along segments from the bulk. So, ***only in this section***, we will refine our object by the splitting of the internal face f to reveal internal segments. That will allow to make Regge calculus on this new bulk segments without affect too much the boundary data and properties from our study object. We will show for our study object, in the 4d-euclidean case with this specific splitting, that we can use Regge calculus inside and its Regge's equations give solutions with non-null curvature. We will give the definitions of Regge calculus, and give the process to adapt the calculus to our study object.

3.1 Definition of Regge's actions and equations

As explain in the classical Sections 2,2.2, the set of lengths give the full geometry of the assembly via the area and shape parameters, and reciprocally. The Regge's geometry is given by a triangulation of the part of space in simplices, with a action associated, where the length of segments are the fundamental parameters. Here, that is exactly the case. Our study object is a assembly of 4-simplices where the full geometry can be given by the length of segments and, in the cylindrical symmetry context, by the set of fundamental parameters (L, L_0, L_f).

3.1.1 4-dimensional Regge's actions and equations for a non-finite euclidean space

The Regge's action S_{Regge} for a non-finite 4-dimensional euclidean space (with no boundary) sliced/discretized in 4-simplices with the length L_{ij} for segments "ij" is :

$$S_{Regge}\left[L_{ij}\right] = \sum_{F \in \text{faces}} a_F\left[L_{ij}\right] \varepsilon_F\left[L_{ij}\right] \tag{33}$$

We have the sum over the all triangular faces F of the area a_F and the associated deficit angle ε_F, all function of the segments length L_{ij}.

The dynamics of the Regge's geometry is given by the action minimizing by the lengths, and give the Regge's equations :

$$\frac{\delta S_{Regge}}{\delta L_{ab}}\left[L_{ij}\right] = \sum_{F \in \{\text{faces} \supset L_{ab}\}} \varepsilon_F\left[L_{ij}\right] \cot \alpha_{ab,F}\left[L_{ij}\right] = 0 \qquad \forall ab \tag{34}$$

Where $\alpha_{ij,F}$ is the "view angle" of the segments "ab" from the opposite point in the face F (Figure 11).

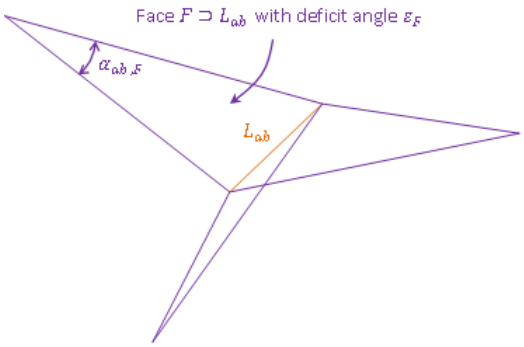


Figure 11: For the segment 'ab' of length L_{ab} (in orange), we have the set of the connected faces F (which include the 'ab' segment) and their associated deficit angle ε_F and "view angle" $\alpha_{ab,F}$. That picture help to apply the Regge equation along the segment 'ab', where you sum the deficit angle ε_F with the cotangent of the "view angle" $\alpha_{ab,F}$ for each connected faces $F \supset ab$.

The Regge's action and equations are the discretized equivalent of Einstein-Hilbert's action and equations :

$$S_{Regge} \sim S_{Einstein-Hilbert} \tag{35}$$

$$\frac{\delta S_{Regge}}{\delta L_{ab}} \sim \frac{\delta S_{E-H}}{\delta g_{\mu\nu}} \tag{36}$$

In fact, for the 3-dimensional case (where the Regge's action and equations are different from 4-dimensional case) we know they describe the same physic for a infinitesimal discretization of space :

$$S^{3d}_{Regge} \xrightarrow[L_{ij}\to 0]{} S^{3d}_{Einstein-Hilbert} \tag{37}$$

$$\frac{\delta S^{3d}_{Regge}}{\delta L_{ab}} \Rightarrow \varepsilon = 0 \xleftrightarrow[L_{ij}\to 0]{} \frac{\delta S^{3d}_{E-H}}{\delta g_{\mu\nu}} \Rightarrow R^{\sigma}_{\mu\rho\nu} = 0 \tag{38}$$

So Regge formulation is a strong equivalent and interpretation for discretized geometry and gravitation.

3.1.2 4-dimensional Regge's actions and equations for a finite euclidean space

For a finite 4-dimensional euclidean space, where boundary take a important place, we will rewrite the Regge's action in a convenient form :

$$S_{Regge}\left[L_{ij}\right] = 2\pi \sum_{F\in \text{faces}} a_F\left[L_{ij}\right] - \sum_{N\in \text{simplices}} \sum_{F\in\{\text{faces of } N\}} a_F\left[L_{ij}\right]\Theta^N_F\left[L_{ij}\right] \tag{39}$$

Here, in the left, we have the sum of "flatness space" from the all faces F with the area a_F. In the right, we have the sum over the all 4-simplices N of their individual actions, corresponding to the sum of their areas with their associated angle Θ^N_F. The angle Θ^N_F correspond to the angle between the two tetrahedra inside the simplex N along their shared-face F (from (21), see 7a). With these definitions, we have the individual Regge's action from a 4-simplex N :

$$S^N_{Regge}\left[L_{ij}\right] = \sum_{F\in\{\text{faces of } N\}} a_F\left[L_{ij}\right]\Theta^N_F\left[L_{ij}\right] \tag{40}$$

Which reproduce the usual Regge's action definition with their associated deficit angles :

$$S_{Regge} = 2\pi \sum_{F\in \text{faces}} a_F - \sum_{N\in \text{simplices}} S^N_{Regge} = \sum_{F\in \text{faces}} a_F\varepsilon_F \tag{41}$$

$$\varepsilon_F = 2\pi - \sum_{N \in \{\text{simplices} \supset F\}} \Theta_F^N \tag{42}$$

This definition is easy to study and understand because we can see the individual effect of each 4-simplices. More, this definition contain the correct definition of Regge's action for the internal geometry of assembly, the bulk, and also the extended definition of Regge's action for the external geometry of assembly, the boundary. For the all bulk part, the ε_F associated to the internal faces are just the deficit angles with exactly the same definition of before. And for the boundary part, the ε_F associated to the boundary faces are not a "complete deficit angle" because it miss the extrinsic curvature. That can be seen as the boundary effect of (missing) extrinsic curvature over the action and geometry of assembly. This last point are the most important, because the idea is than the geometry of the boundary will affect the dynamics of the internal geometry, via the Regge equation, and will create curvature : The boundary properties will affect the internal geometry and curvature.

The corresponding Regge's equations are given by minimizing Regge's action over the segment lengths in the bulk. So for each segments "ab" non-include to the boundary, we have the Regge's equations :

$$\frac{\delta S_{Regge}}{\delta L_{ab}}\left[L_{ij}\right] = \sum_{F \in \{\text{faces} \supset L_{ab}\}} \varepsilon_F\left[L_{ij}\right] \cot \alpha_{ab,F}\left[L_{ij}\right] = 0 \qquad \forall ab \in \{\text{segments in the bulk}\} \tag{43}$$

which are the same as usual Regge equation, but only effective for the bulk.

3.2 Adaptation of study object for Regge calculus

We have seen than Regge's equations take place only for the bulk lengths. The problem it's our study object have only one internal face, the all segments belong to the boundary. The set of lengths (L, L_0, L_f) of boundary give the all geometry, and their values can give classically curvature, but we have no dynamics in the Regge sens ! Even if the classical geometry give curvature with the appropriate set of (L, L_0, L_f), it will be important and instructive to see if the Regge's dynamics can reproduce the same result. But we must adapt our study object for than the Regge's equations be effective and physical.

3.2.1 Split objects

To make viable Regge calculus inside our study object the simplest way it's just to split the internal face f for reveal internal segments. In agreement with the cylindrical symmetries, we split the segments of the face f (of length L_f) by the middle in two segments of length $\frac{1}{2}L_f$ and connect the middles by new internal segments. That will split the face f in 4 triangles : one in the center, with new internal f-segments of respective lengths $L_f^{1\prime}$, $L_f^{2\prime}$, $L_f^{3\prime}$, and three faces around, with 2 segments of length $\frac{1}{2}L_f$ and 1 associated f-segments of length $L_f^{i\prime}$ each. The center face will be called f', in reference of the f-segments $L_f^{1\prime}$, $L_f^{2\prime}$, $L_f^{3\prime}$ which compose it. The others faces will be called f^i, in reference to the f-segment $L_f^{i\prime}$ which compose it with two $\frac{1}{2}L_f$ segments. We can see the splitting of the face f as follow :

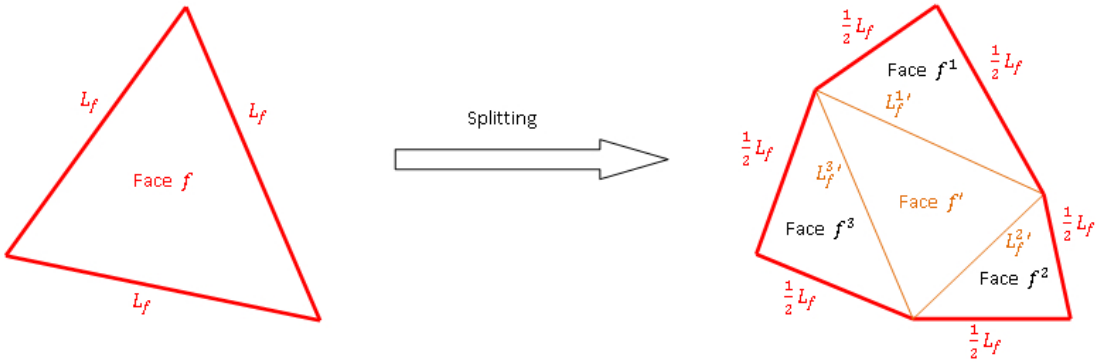


Figure 12: The splitting of the face f to make viable Regge calculus.

The splitting of f, base of the internal tetrahedra, split the faces of internal tetrahedra with new segments and new tetrahedra with bases f' and f^i. The old faces f_0 (given previously by two [black] L_0 and one [red] L_f) are split in half with secants passing by the middle of the previousL_f segments ; the lengths of the secants segments will be L_0'. We have the new boundary faces given by L_0, L_0' and $\frac{1}{2}L_f$, called f_0' and the new internal faces given by two L_0' and one L_f^i segments, called f_0^i. As the Figures :

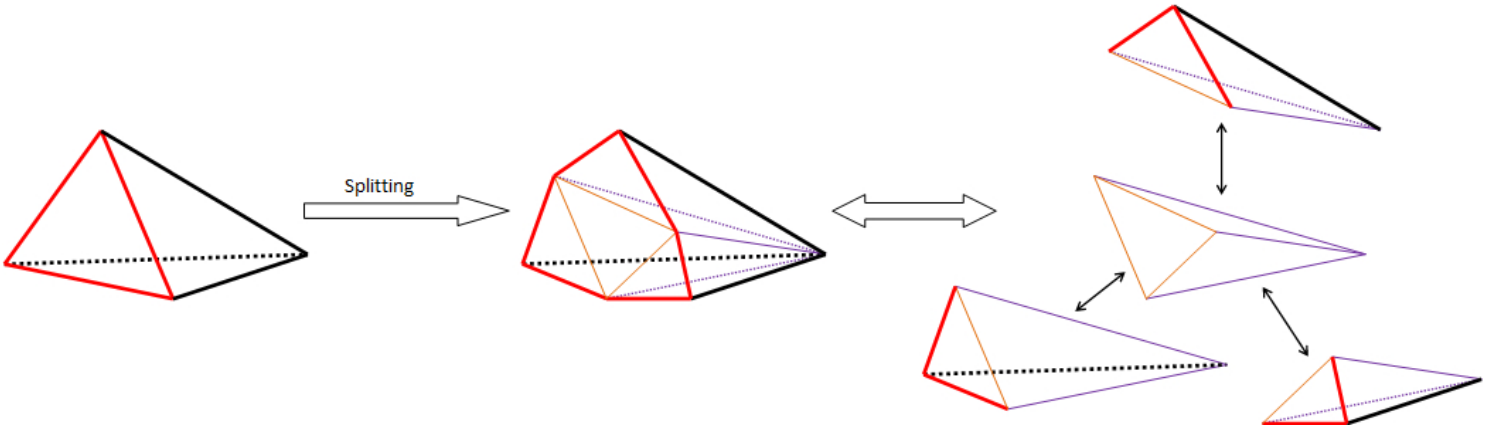


Figure 13: The splitting of the face f and internal tetrahedra. Next, the exploded view of the split internal tetrahedra.

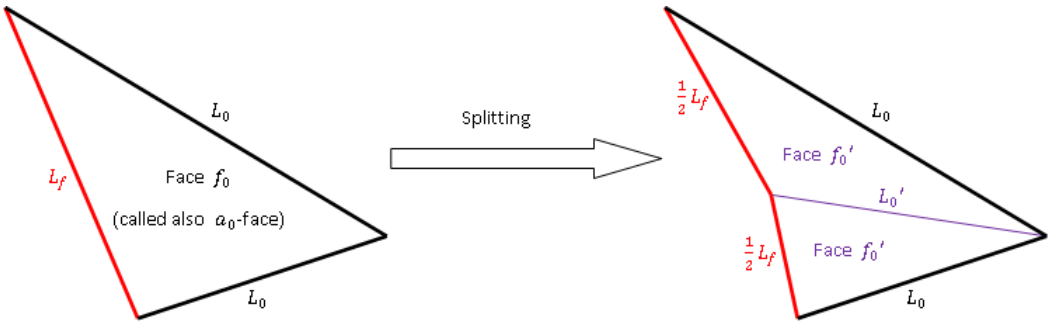


Figure 14: The splitting of the faces f_0, called also a_0-faces, given by the splitting of f.

The splitting of the old faces f_0, from shared-tetrahedra, in new faces f_0' also split the boundary tetrahedra which shared these faces. The boundary tetrahedra appear split in half, by the middle of the previous L_f segments, and create new faces given by the two L_0' and one L segments ; as in the following Figure :

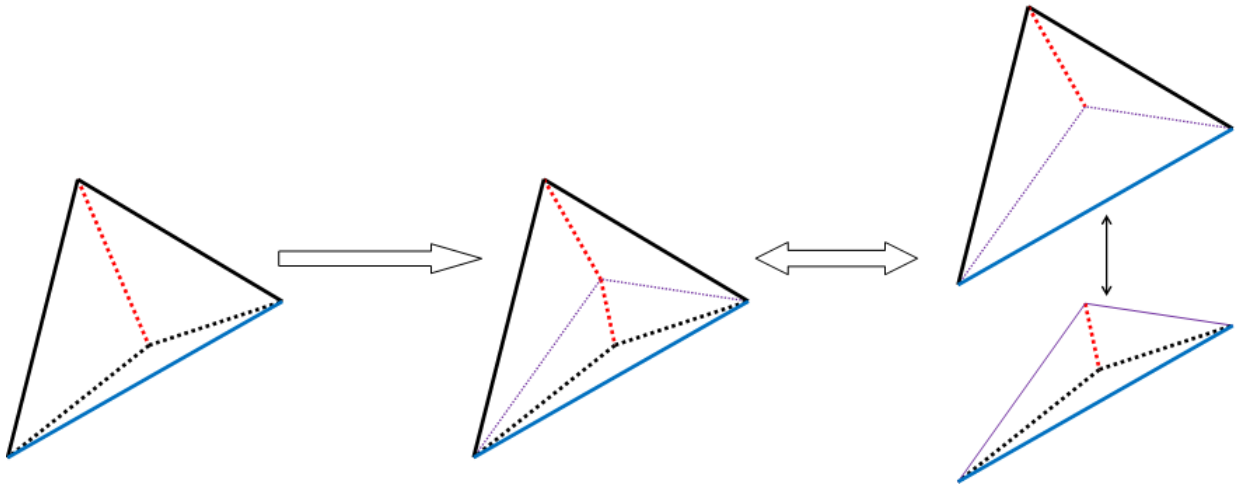


Figure 15: The splitting of the boundary tetrahedra given by the splitting of f

The new faces, with two (purple) L'_0 and one (blue) L, will be called f'_a.

Note, the new internal f-segments lengths $L_f^{1\prime}$, $L_f^{2\prime}$, $L_f^{3\prime}$ are not fixed by the boundary data, and their values will be given dynamically by the Regge's equations. The specific case where $L_f^{i\prime} = \frac{L_f}{2}$ and $L'_0 = \sqrt{L_0^2 - \frac{L_f^2}{4}}$ correspond to the case where the internal face $f' = \left(L_f^{1\prime}, L_f^{2\prime}, L_f^{3\prime}\right)$ and the internal faces$f^i = \left(L_f^{i\prime}, \frac{1}{2}L_f, \frac{1}{2}L_f\right)$ are co-planar and reproduce the equilateral triangular face f with lengths L_f. But, in general, the new internal faces are not necessary in the same plane and the contour of the set is not necessary a triangle.

This splitting is really interesting because, in addition to make internal f-segments for viable Regge calculus, we will see the geometry dynamics of the assembly for our object with boundary parameters which give curvature in the usual/classical way. The interest is to take classical parameters (L, L_0, L_f) which give curvature in the usual assembly and see the preservation of curvature from the internal f-segments and associated Regge's equations : our study object will be able to have Regge's dynamics and non-null curvature within. Moreover, we can reverse the interpretation and consider our usual study object as a part of a bigger assembly with associated curvature. Indeed, after the splitting, the central assembly given by the faces f', f_0^i and (L, L'_0, L'_0) is like our usual study object with its parameters $\left(L, L'_0, L_f^i\right)$ and its curvature $\varepsilon_{f'}$ inside a bigger object : this central part is exactly like our usual study object, can have curvature and is contained inside as a solution of Regge's equations. With this finding, we can easily imagine our usual study object, with non-null curvature and associated parameters, from a bigger assembly as a Regge's solution. So we will see with the proper boundary lengths that our study object can contain Regge's solutions with curvature and, by reverse interpretation, can be seen as a solution of Regge's equations inside a imaginary bigger assembly.

3.2.2 Parameters and geometric objects

Now, the face f of our original study object is split in 4 triangular faces with new f-segments in the bulk, where the Regge's equations will act. We have a new boundary for this assembly formed by the L and L_0 segments, the split segments of f of length $\frac{1}{2}L_f$, and the new secants segments of length L'_0 introduced by the splitting of f. The boundary geometry are given by four parameters : L, L_0 and L_f, L'_0. The drawing of the symmetrical boundary with its tetrahedra is below :

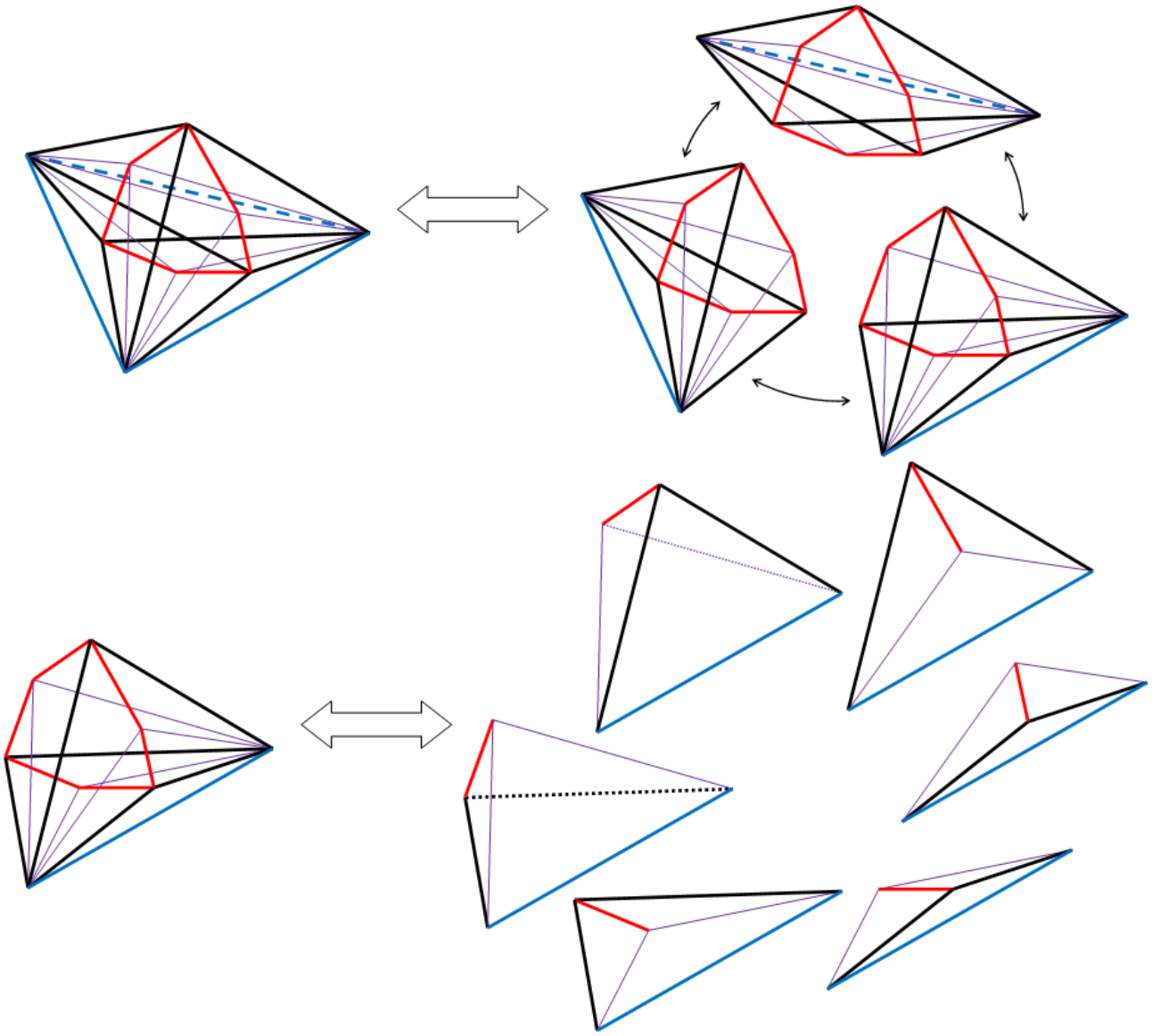


Figure 16: The first picture represent the 3d-boundary of the new assembly, and its exploded view in pieces of shape of split 4-simplex (The contour from the red segments give no face, it's the contour of a hole). The second picture represent the exploded view of shape of one split 4-simplex in new boundary tetrahedra.

As previously, the purple segments have the same length L'_0 and the red segments have the same length $\frac{L_f}{2}$. The blue and black segments have respectively the length L and L_0. In these pictures, we see the all boundary of the new object as a assembly of the same sort of tetrahedra. These tetrahedra have their geometry given by the set $\left(L, L_0, \frac{1}{2}L_f, L'_0\right)$ and share the faces f'_a (two purple L'_0 and one blue L) and f_a (two black L_0 and one blue L) for built the shape of split 4-simplex. The three shapes of split 4-simplices share the faces f'_0 (one purple L'_0, one black L_0 and one red $\frac{1}{2}L_f$) for built the boundary of the all split assembly. The drawing of the full new object, including the internal (orange) f-segments for Regge calculus and their associated faces f', f^i , is :

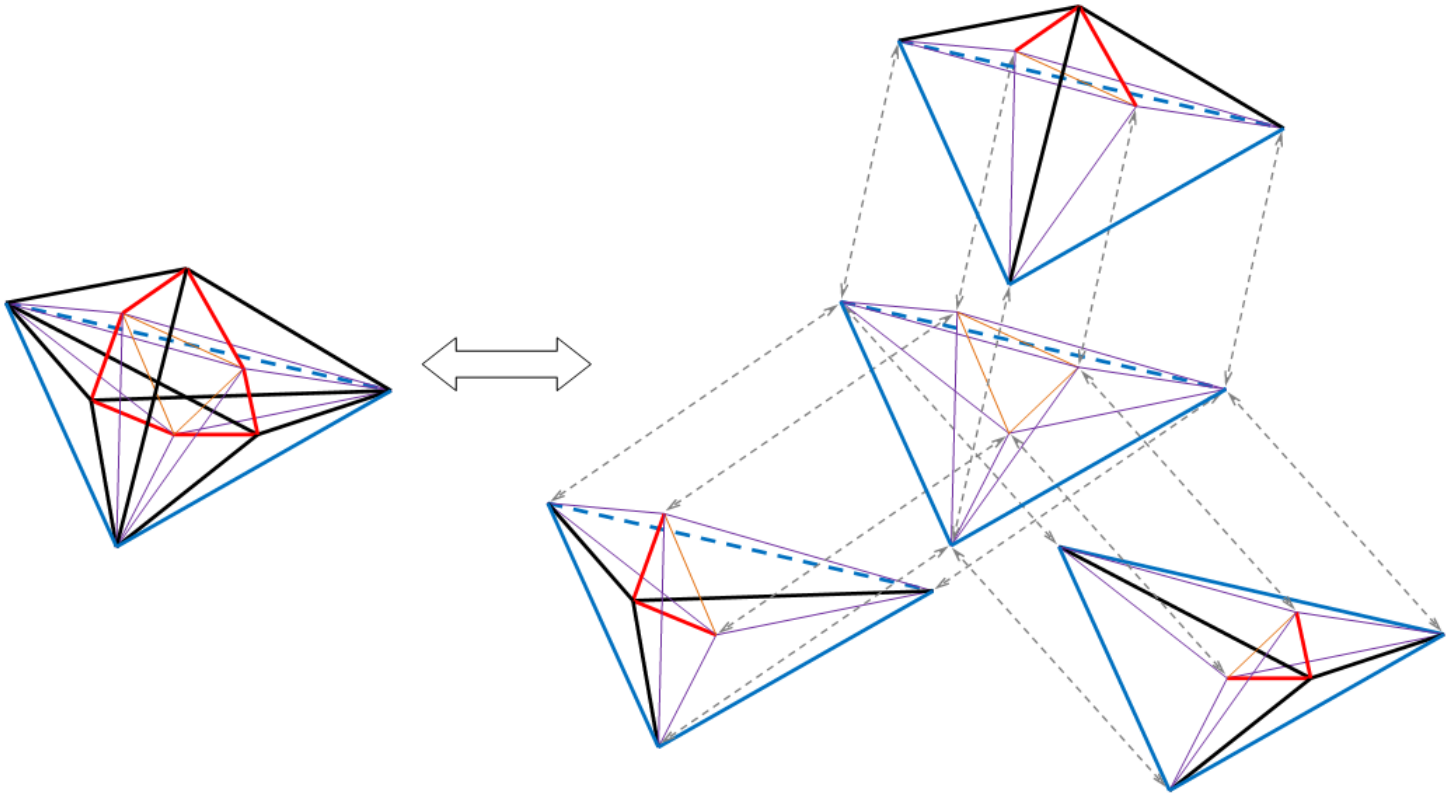


Figure 17: In the left, the split object for the Regge calculus. And its exploded view in terms of assembly of 4-simplices, in the right.

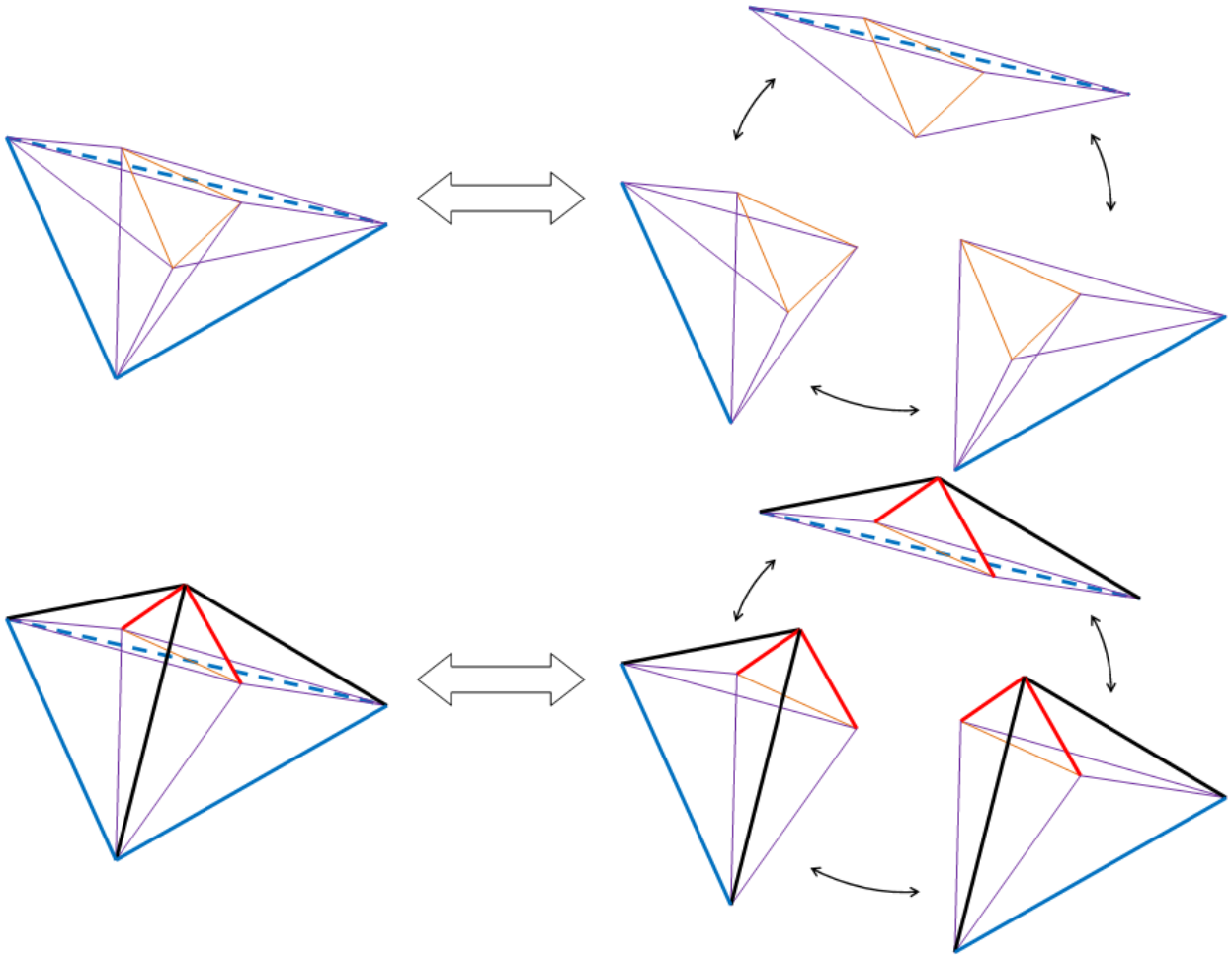


Figure 18: The exploded view of each assembly of 4-simplices for the Regge calculus object. The f-segments where we will apply the Regge's equations appear in orange.

Where we find a internal face with these f-segments in orange. Remember, in the context of Regge calculus, the length of f-segments are not necessarily equals ! The equality properties of f-segment lengths will be a consequence of Regge's action and equations from the cylindrical symmetries of boundary, not a initial constraint.

We can see the "assembly drawing" of the split object via this spin-foam graph :

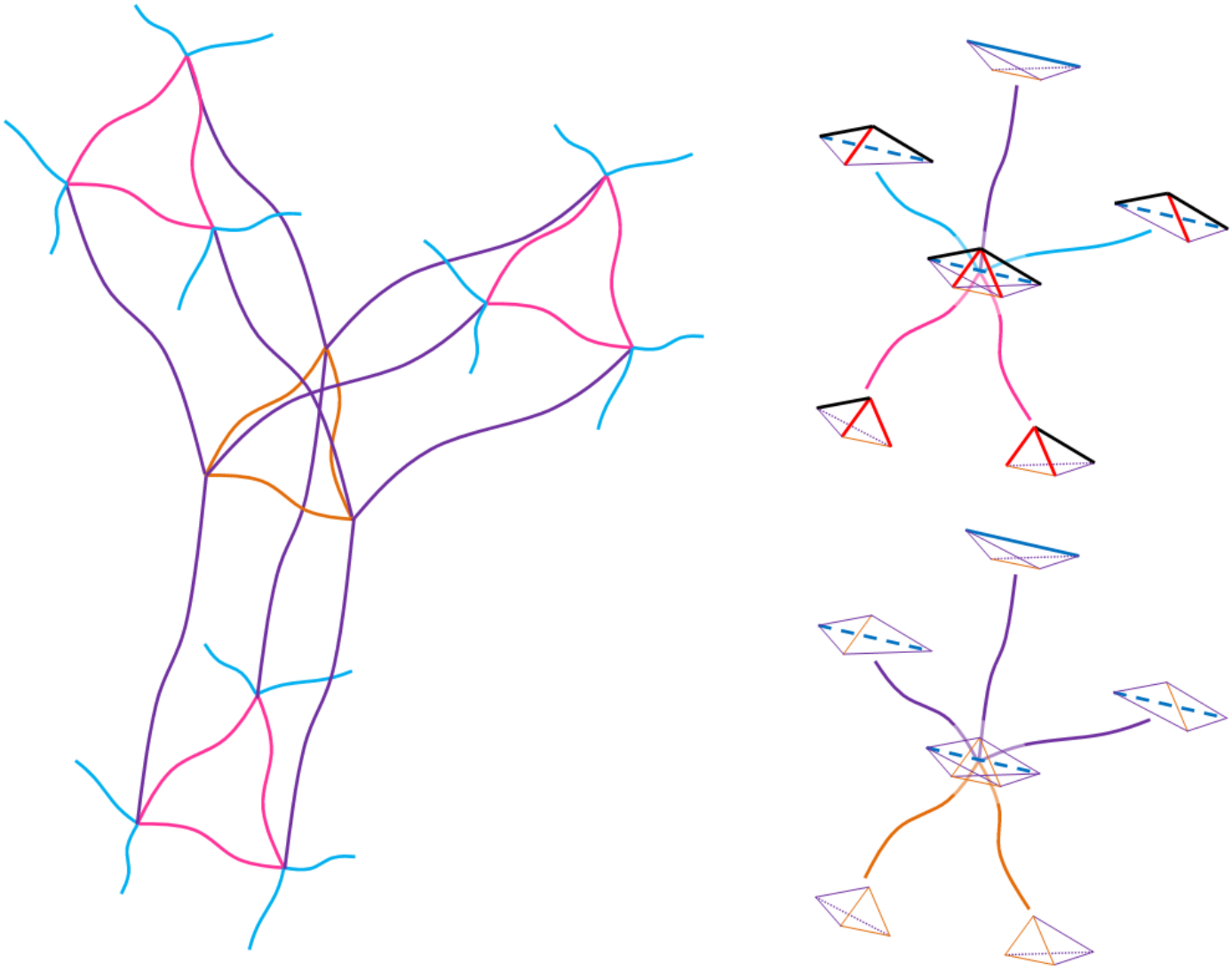


Figure 19: Spin-foam of the split assembly dual of Figures 17,18. The orange and pink edges represents the internal tetrahedra which come from of shared-tetrahedra splitting (Figure 13), the light blue edges represents the external tetrahedra which come from of boundary tetrahedra splitting (Figure 15), and the purple edges represents internal tetrahedra needed to complete the splitting and glue the new 4-simplices together. We see the associated pieces of geometry of the graph in the right.

The split assembly have the following properties :

- **30 segments :**
  - 3 internal segments : the f-segments in orange, with the lengths $L_f^{1\prime}$, $L_f^{2\prime}$, $L_f^{3\prime}$ which will be given by the Regge's equations.
  - 27 boundary segments :
    - 12 from the boundary tetrahedra of original study object :
      - 9 segments of length L_0 in black
      - 3 segments of length L in blue
    - 15 from the splitting :
      - 6 segments of length $\frac{L_f}{2}$ in red
      - 9 segments of length L_0' in purple

- **49 faces :**
  - 13 internal faces :
    - 1 "fully" internal face from the split face f : the face f' with its three (orange) f-segments.
    - 12 "simple" internal faces :
      - 3 from the split face f : the faces f^i composed by two red segments and one orange each.
      - 9 from the split internal tetrahedra : the faces f_0^i composed by two purple segments and one orange each.
  - 36 boundary faces :
    - 27 from the split of boundary tetrahedra :
      - 9 composed by two purple segments and one blue segment each.
      - 18 composed by one red, black, purple segments each.
    - 9 from the boundary tetrahedra : composed by two black segments and one blue segment each.
- **39 tetrahedra :**
  - 21 internal tetrahedra :
    - 12 from the splitting of internal tetrahedra :
      - 3 composed with the face f' and three purple segments each.
      - 9 : composed with the by two purple, two red, one black and one orange each.
    - 9 from the splitting of boundary tetrahedra : composed with the four purple, one orange and one blue segments each.
  - 18 boundary tetrahedra : composed by two purple, two black, one red and one blue each.
- **12 4-simplices :**
  - 3 composed by the purple, orange and blue segments.
  - 9 composed by the black, purple, red, orange and blue segments.

3.2.3 Equations for deficit angles

In the optics to use Regge's equations on the f-segments, we need to compute the deficit angle for the all faces connected to them. In this subsection, we will express the formulas necessary to compute the deficit angle for the face f, f^i and the face f_0^i. After that, the Regge calculus (derived from Subsection 3.1.2) just consist to compute the quantities :

$$\frac{\partial S_{Regge}}{\partial L_f^i}\left(L, L_0, L_0', \frac{1}{2}L_f, L_f^{1\prime}, L_f^{2\prime}, L_f^{3\prime}\right) = \sum_{F\in\left\{\text{faces}\supset L_f^i\right\}} \varepsilon_F \cot\alpha_{i,F}\left(L, L_0, L_0', \frac{1}{2}L_f, L_f^{1\prime}, L_f^{2\prime}, L_f^{3\prime}\right) \; i=1,2,3 \quad (44)$$

and see for what values of $L_f^{1\prime}$, $L_f^{2\prime}$, $L_f^{3\prime}$ these quantities becomes nulls.

For the central part, given by the face f' (see the Figure 18, first picture), the angle between the tetrahedra of same base f' are given by :

$$\cos\Theta_{kl}^N = \frac{\cos\theta_{(kc)(lc)}^N - \cos\theta_{(kl)(kc)}^N \cos\theta_{(kl)(lc)}^N}{\sin\theta_{(kl)(kc)}^N \sin\theta_{(kl)(lc)}^N} \text{ for } kl = f' \quad (45)$$

With the dihedral angles :

$$\cos\theta_{(kc)(lc)}^i = \frac{4\,{L_0'}^2 - 2L^2 - {L_f^i}'^2}{4\,{L_0'}^2 - {L_f^i}'^2} \quad (46)$$

$$\cos\theta_{(kl)(kc)}^i = \cos\theta_{(kl)(lc)}^i = \frac{L_f^{i\prime}}{\sqrt{4\,{L_0'}^2 - {L_f^i}'^2}} \cdot \frac{{L_f^{j}}'^2 + {L_f^{k}}'^2 - {L_f^i}'^2}{\sqrt{2\,{L_f^i}'^2\,{L_f^j}'^2 + 2\,{L_f^i}'^2\,{L_f^k}'^2 + 2\,{L_f^j}'^2\,{L_f^k}'^2 - {L_f^i}'^4 - {L_f^j}'^4 - {L_f^k}'^4}} \quad (47)$$

And with the associated deficit angle equal to $\varepsilon_{f'} = 2\pi - \sum_{i=1}^{3} \Theta^i_{f'}$. In the special case where the lengths $L_f^{1\prime}$, $L_f^{2\prime}$, $L_f^{3\prime}$ will be equal to the same length L'_f (that will be the solution that we will find in the next) we can reduce the dihedral and find a simple expression of the deficit angle of the face f' :

$$\cos\theta_{(kc)(lc)} = \frac{4\,{L'_0}^2 - 2L^2 - {L'_f}^2}{4\,{L'_0}^2 - {L'_f}^2}\ ,\ \cos\theta_{(kl)(kc)} = \cos\theta_{(kl)(lc)} = \frac{L'_f}{\sqrt{4\,{L'_0}^2 - {L'_f}^2}} \cdot \frac{1}{\sqrt{3}} \tag{48}$$

$$\Rightarrow\ \cos\Theta_{f'} = \frac{3\left({L'_0}^2 - \frac{1}{2}L^2\right) - {L'_f}^2}{3\,{L'_0}^2 - {L'_f}^2}\ \Rightarrow\ \varepsilon_{f'} = 2\pi - 3\Theta_{f'} \tag{49}$$

For the exterior parts, given by the faces f^i (see the Figure 18, second picture), the angle between the tetrahedra of same base f^i are given by the dihedral angle :

$$\cos\theta^i_{(kc)(lc)} = \frac{4\,{L'_0}^2 - 2L^2 - {L_f^i}'^2}{4\,{L'_0}^2 - {L_f^i}^2}\ \cos\theta^i_{(kl)(kc)} = \cos\theta^i_{(kl)(lc)} = \frac{4\,{L'_0}^2 + L_f^2 - 4L_0^2 - 2\,{L_f^i}'^2}{2\sqrt{4\,{L'_0}^2 - {L_f^i}^2}\sqrt{L_f^2 - {L_f^i}^2}} \tag{50}$$

That can be reduce for the special case $L_f^{1\prime} = L_f^{2\prime} = L_f^{3\prime} = L'_f$ to :

$$\cos\theta_{(kc)(lc)} = \frac{4\,{L'_0}^2 - 2L^2 - {L'_f}^2}{4\,{L'_0}^2 - {L'_f}^2}\ ,\ \cos\theta_{(kl)(kc)} = \cos\theta_{(kl)(lc)} = \frac{4\,{L'_0}^2 + L_f^2 - 4L_0^2 - 2\,{L'_f}^2}{2\sqrt{4\,{L'_0}^2 - {L'_f}^2}\sqrt{L_f^2 - {L'_f}^2}} \tag{51}$$

Which imply for the faces f^i :

$$\Rightarrow\ \cos\Theta_{f^i} = \frac{4\left(L_f^2 - {L'_f}^2\right)\left(4\,{L'_0}^2 - 2L^2 - {L'_f}^2\right) - \left(4\,{L'_0}^2 + L_f^2 - 4L_0^2 - 2\,{L'_f}^2\right)^2}{4\left(L_f^2 - {L'_f}^2\right)\left(4\,{L'_0}^2 - {L'_f}^2\right) - \left(4\,{L'_0}^2 + L_f^2 - 4L_0^2 - 2\,{L'_f}^2\right)^2}\ \Rightarrow\ \varepsilon_{f^i} = 2\pi - 3\Theta_{f^i} \tag{52}$$

For the face f_0^i, composed by 2 purple segments and one orange f-segments, shared between the central part to the exterior part, we have two type of angles $\Theta^N_{f_0^i}$: those in the central part $\Theta^a_{f_0^i}$, and those in the exterior parts $\Theta^b_{f_0^i}$. The first angles $\Theta^a_{f_0^i}$, defined in the central part connected to the face f', are given by the dihedral angles :

$$\cos\theta^i_{(kc)(lc)} = \frac{{L_f^i}'}{\sqrt{4\,{L'_0}^2 - {L_f^i}'^2}} \cdot \frac{{L_f^j}'^2 + {L_f^k}'^2 - {L_f^i}'^2}{\sqrt{2\,{L_f^i}'^2\,{L_f^j}'^2 + 2\,{L_f^i}'^2\,{L_f^k}'^2 + 2\,{L_f^j}'^2\,{L_f^k}'^2 - {L_f^i}'^4 - {L_f^j}'^4 - {L_f^k}'^4}} = \cos\theta^i_{(kl)(kc)} \tag{53}$$

$$\cos\theta^i_{(kl)(lc)} = \frac{4\,{L'_0}^2 - 2L^2 - {L_f^i}'^2}{4\,{L'_0}^2 - {L_f^i}'^2} \tag{54}$$

Which give in the special case $L_f^{1\prime} = L_f^{2\prime} = L_f^{3\prime} = L'_f$:

$$\cos\theta_{(kc)(lc)} = \frac{L'_f}{\sqrt{4\,{L'_0}^2 - {L'_f}^2}} \cdot \frac{1}{\sqrt{3}} = \cos\theta_{(kl)(kc)},\ \cos\theta_{(kl)(lc)} = \frac{4\,{L'_0}^2 - 2L^2 - {L'_f}^2}{4\,{L'_0}^2 - {L'_f}^2} \tag{55}$$

$$\Rightarrow\ \cos\Theta^a_{f_0^i} = \frac{L'_f L}{2\sqrt{3\,{L'_0}^2 - {L'_f}^2}\sqrt{4\,{L'_0}^2 - {L'_f}^2 - L^2}} \tag{56}$$

The second angles $\Theta^b_{f_0^i}$, defined in the exterior parts connected to the face f^i, are given by the dihedral angles :

$$\cos\theta^i_{(kc)(lc)} = \frac{4\,{L'_0}^2 + L_f^2 - 4L_0^2 - 2\,{L_f^i}^2}{2\sqrt{4\,{L'_0}^2 - {L'_f}^2}\sqrt{L_f^2 - {L_f^i}^2}} = \cos\theta^i_{(kl)(kc)},\ \cos\theta^i_{(kl)(lc)} = \frac{4\,{L'_0}^2 - 2L^2 - {L_f^i}^2}{4\,{L'_0}^2 - {L_f^i}^2} \tag{57}$$

Which give in the special case $L_f^{1\prime} = L_f^{2\prime} = L_f^{3\prime} = L_f'$:

$$\cos\theta_{(kc)(lc)} = \frac{4\ {L_0'}^2 + L_f^2 - 4L_0^2 - 2\ {L_f'}^2}{2\sqrt{4\ {L_0'}^2 - {L_f'}^2}\sqrt{L_f^2 - {L_f'}^2}} = \cos\theta_{(kl)(kc)},\ \cos\theta_{(kl)(lc)} = \frac{4\ {L_0'}^2 - 2L^2 - {L_f'}^2}{4\ {L_0'}^2 - {L_f'}^2} \tag{58}$$

$$\Rightarrow\ \cos\Theta^b_{f_0^i} = \frac{\left(4\ {L_0'}^2 + L_f^2 - 4L_0^2 - 2\ {L_f'}^2\right) L}{\sqrt{4\left(4\ {L_0'}^2 - {L_f'}^2\right)\left(L_f^2 - {L_f'}^2\right) - \left(4\ {L_0'}^2 + L_f^2 - 4L_0^2 - 2\ {L_f'}^2\right)^2}\sqrt{4\ {L_0'}^2 - {L_f'}^2 - L^2}} \tag{59}$$

Finally we have the deficit angle of the face f_0^i :

$$\varepsilon_{f_0^i} = 2\pi - 2\Theta^a_{f_0^i} - 2\Theta^b_{f_0^i} \tag{60}$$

3.3 Applications of Regge calculus

In order to compute and study the Regge calculus of the new object, we have designed a C++ code (not included in this thesis report) for compute the Regge's action and find the values of the f-segments who minimize it. In the code we give the boundary lengths L, L_0, L_0', $\frac{1}{2}L_f$ as constants. For each set of f-segments lengths $L_f^{1\prime}$, $L_f^{2\prime}$, $L_f^{3\prime}$ which respect the triangle inequalities of the all assembly, the code compute the individual action of each 4-simplex in function of lengths, and sum the individual actions as define above (see 3.1.2 and (39)). After, the code give the values of $L_f^{1\prime}$, $L_f^{2\prime}$, $L_f^{3\prime}$ which "locally minimize" the full action and the corresponding curvatures ε_f associated to the faces from f. Of course, because of numerical limitation, the computed values of $L_f^{1\prime}$, $L_f^{2\prime}$, $L_f^{3\prime}$ are not from a continuum ; they will be given with a precision $\delta L_f'$ inherent to the code (and so perfectly defined) and the real values where the full action is perfectly "locally minimized" are always in the intervals $\pm\delta L_f'$. The precision will be also given for each computations and cases studied.

3.3.1 Regge computation for unspecified face f

In first, we will study the general cases where the three f-segments, of respective lengths $L_f^{1\prime}$, $L_f^{2\prime}$, $L_f^{3\prime}$, can be different and "locally minimize" the action :

$$\frac{\partial S_{Regge}}{\partial L_f^i}\left(L, L_0, L_0', \frac{1}{2}L_f, L_f^{1\prime}, L_f^{2\prime}, L_f^{3\prime}\right) = 0 \qquad \text{for } i = 1, 2, 3 \tag{61}$$

For the special case where $L_0 = L_f = 1$ and $L_0' = \frac{\sqrt{3}}{2}$, $L = \sqrt{2}$, corresponding to the flat case in the original object from the classical geometry, we find obviously the solution :

- $L_f^{1\prime} = L_f^{2\prime} = L_f^{3\prime} = 0.5 \pm 0.02$ with $\varepsilon_{f'} = 0$, $\varepsilon_{f^i} = 0$, $\varepsilon_{f_0^i} = 0$

Which is just the flat solution where the split faces are in the same plane, and the splitting just cut the original object without no change.

For the special case where $L_0 = L_f = 1$ and $L_0' = \frac{\sqrt{3}}{2}$, $L = 1$, corresponding to the original object but with positive curvature induced by a "tight equator" $L < \sqrt{2}$, we find the solution :

- $L_f^{1\prime} = L_f^{2\prime} = L_f^{3\prime} = 0.711518 \pm 0.02$ with $\varepsilon_{f'} = 1.99152$, $\varepsilon_{f^i} = 1.53229$, $\varepsilon_{f_0^i} = -0.421284$

For the special case where $L_0 = L_f = 1$ and $L_0' = \frac{\sqrt{3}}{2}$, $L = 1.5$, corresponding to the original object but with negative curvature induced by a "stretched equator" $L > \sqrt{2}$, we find the solution :

- $L_f^{1\prime} = L_f^{2\prime} = L_f^{3\prime} = 0.421923 \pm 0.02$ with $\varepsilon_{f'} = -0.469524$, $\varepsilon_{f^i} = -0.422882$, $\varepsilon_{f_0^i} = 0.118673$

For the special case where $L_0 = L_f = 1$ and $L_0' = L = 1$, we have the solution :

- $L_f^{1\prime} = L_f^{2\prime} = L_f^{3\prime} = 0.866 \pm 0.02$ with $\varepsilon_{f'} = 2.59031$, $\varepsilon_{f^i} = 2.59031$, $\varepsilon_{f_0^i} = 9.9961 \times 10^{-5}$

Corresponding in fact to the exact solution :

- $L_f^{1\prime} = L_f^{2\prime} = L_f^{3\prime} = \frac{\sqrt{3}}{2}$ with $\varepsilon_{f'} = \varepsilon_{f^i} = 2\pi - 3\arccos\left(\frac{1}{3}\right)$, $\varepsilon_{f_0^i} = 0$

Here we have interesting results, because we have already a solution with non-null curvature ! We can see this solution as the local extremum if we draw the full action S_{Regge} in function of L_f^1 and $L_f^2 = L_f^3$:

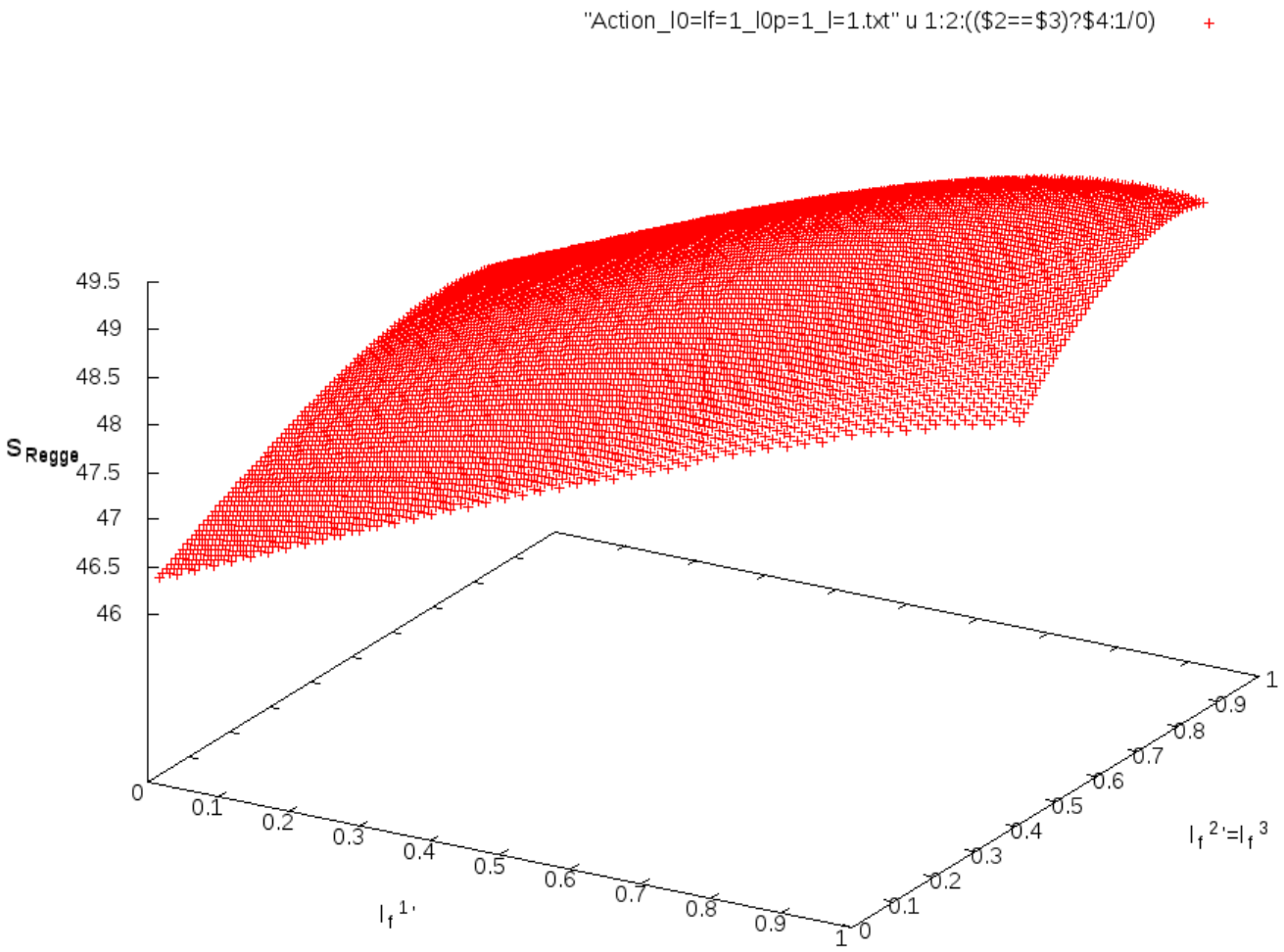


Figure 20: Evolution of Regge's action in function of the L_f^1 and $L_f^2 = L_f^3$. Of course we are off-shell of the Regge's dynamics, the Regge's equations correspond just to the extremums of this surface.

So we found several solutions from the Regge's action minimization : the solutions can have curvature, and solutions AND curvatures evolve of the boundary lengths L, L_0, L_0', L_f'.

3.3.2 Regge computation for equilateral face f

As seen previously, the solutions are always the equilateral cases where the f-segments have the same lengths. Because these solutions preserve the cylindrical symmetries of the original study assembly, will be easily comparable to the classical/quantum cases studied, and are probably the only physical solutions for cylindrical symmetries. Because of these results, we will study the simplest cases where the $L_f^{1\prime}$, $L_f^{2\prime}$, $L_f^{3\prime}$ are equal in the next.

For the equilateral cases where $L_f^{i\prime} = L_f' \ \forall i$, we can study the evolution of solution and its corresponding curvature in function of the boundary lengths L, L_0, L_0', L_f'. Because of the size space of configuration, and more convenience, we take $L_0 = 1$ and $L_f = 1$. The choice of $L_0 = 1$ is purely arbitrary, that can be understand like just

a scale choice for the assembly ; the other lengths can be compared to this scale length. The choice of $L_f = 1$ come from the choice for confine the f-segments : fundamentally the curvature associated to the face f depend to L, L'_0, $\frac{1}{2}L_f$ (when the scale $L_0 = 1$ is given) but in fact the curvature induced by the boundary depend to the shape of boundary tetrahedra and the (non-linear) "ratio" between $\left(L_0, \frac{1}{2}L_f\right)$ and (L'_0, L) ; so we can fix $L_f = 1$ and just see the evolution of induced curvature by the evolution of (L'_0, L) compared to $\left(L_0, \frac{1}{2}L_f\right) = \left(1, \frac{1}{2}\right)$.

So, with $\left(L_0, \frac{1}{2}L_f\right) = \left(1, \frac{1}{2}\right)$ parameters, we can compute the solutions L'_f and associated curvatures ε_f in function of L'_0 and L. We obtain the following drawings :

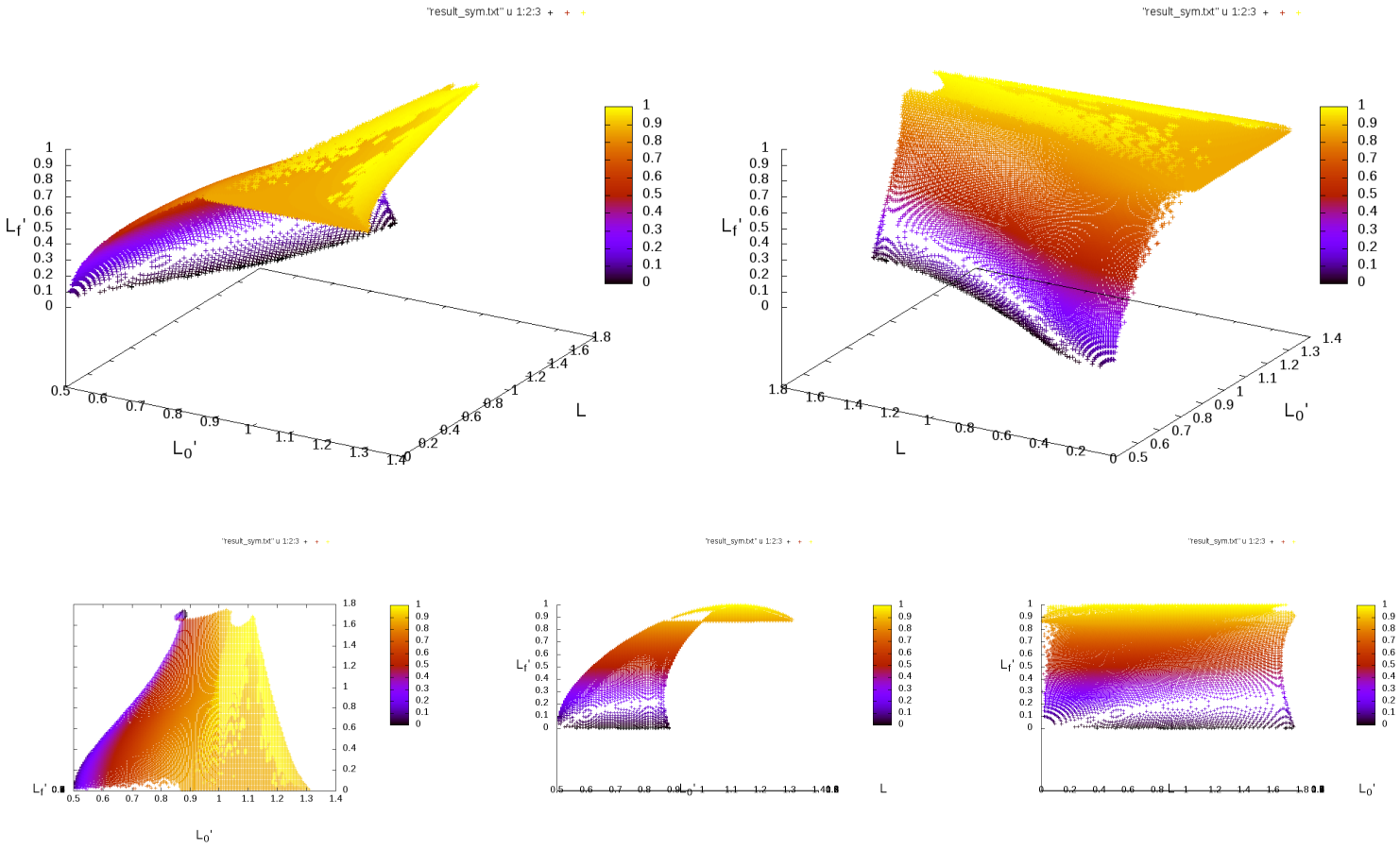


Figure 21: Length L'_f from the Regge's equations solutions in function of L'_0 and L under different views. The first and second pictures are the isometric views of the surface of solutions, the third is the top view (in the pane L'_0, L), the fourth is the front view (in the plane L'_f, L'_0) and the fifth is the side view (in the plane L'_f, L).

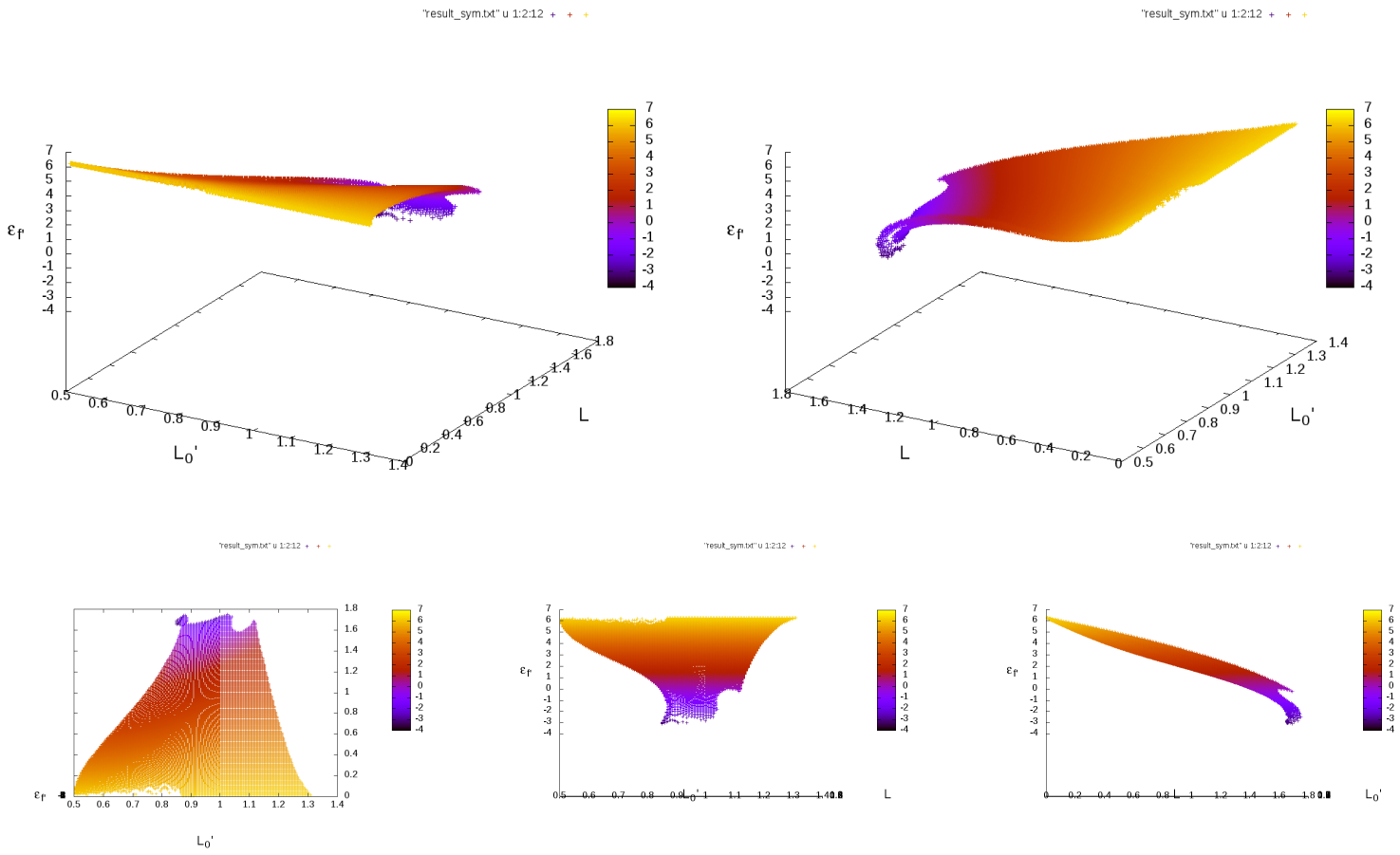


Figure 22: Curvature (deficit angle) $\varepsilon_{f'}$ from the Regge's equations solutions in function of L'_0 and L under different views. The first and second pictures are the isometric views of the surface of solutions, the third is the top view (in the pane L'_0, L), the fourth is the front view (in the plane $\varepsilon_{f'}$, L'_0) and the fifth is the side view (in the plane $\varepsilon_{f'}$, L).

Where the length L'_f and curvature $\varepsilon_{f'}$ solutions from Regge's equations evolve continuously in function of the boundary parameters L'_0, L. If we look the sections with $L'_0 = \frac{\sqrt{3}}{2}$, we have the following drawing of $\varepsilon_{f'}$, ε_{f^i} and $\varepsilon_{f_0^i}$ in function of L :

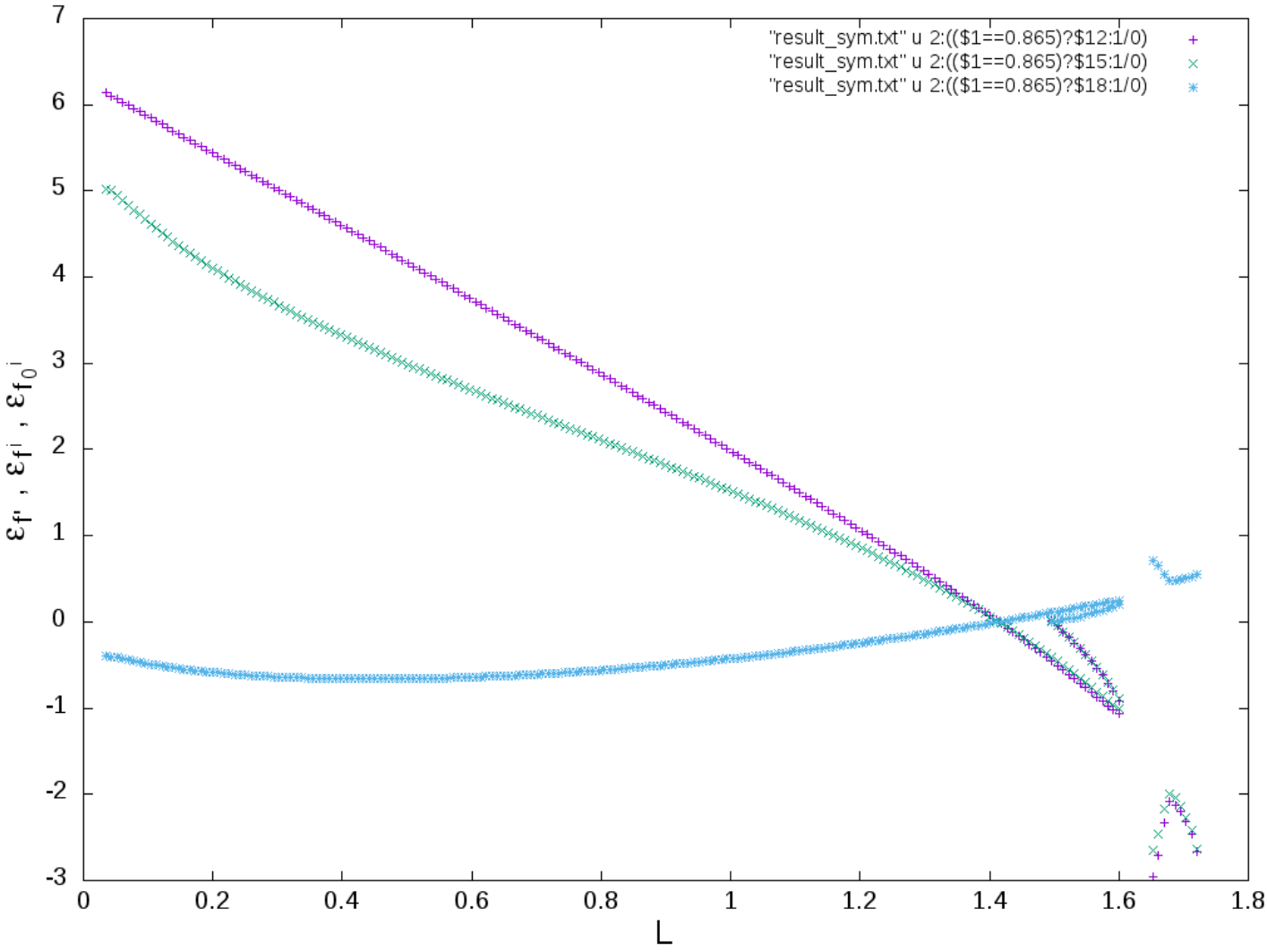


Figure 23: Evolution of $\varepsilon_{f'}$(purple), ε_{f^i} (green) and $\varepsilon_{f_0^i}$ (blue) from the Regge's equations solutions in function of L.

Where we show the curvatures from Regge's solutions evolve continuously in function of L for fixed $L_0 = L_f = 1$ and $L_0' = \frac{\sqrt{3}}{2}$. We recover the expected flat solution for the crosspoint at $L = \sqrt{2}$. We show also a discontinuity for $L = 2\sqrt{\frac{2}{3}}$, that correspond to the geometrical limit where the split 4-simplices are degenerated flat : means they are each in a 3d frame, and the length $L = 2\sqrt{\frac{2}{3}}$ correspond to the height of two regular tetrahedra with the same basis with three tetrahedra glued flatly inside. The solutions from $L > 2\sqrt{\frac{2}{3}}$ correspond to solutions where three tetrahedra for each 4-simplex are longer than the height of the two last tetrahedra : means the 4-simplex geometries are hyperbolic. We see also in the region $L \in \left]\frac{2}{3}\sqrt{5}; 2\sqrt{\frac{2}{3}}\right[$ we have two solutions for the Regge's geometries, that is the two possible way to bend the split L_f-segments : to the "exterior", means we have a convex boundary geometry, or to the "interior", means we have a concave boundary geometry.

We can also draw the evolution of $\mathrm{R} = \sum_{F \subset bulk} a_F \varepsilon_F$ (explicitly $\mathrm{R} = a_{f'}\varepsilon_{f'} + 3a_{f^i}\varepsilon_{f^i} + 9a_{f_0^i}\varepsilon_{f_0^i}$ with the symmetries) which is the equivalent of the Einstein-Hilbert's action $\int_{bulk} d^4x\sqrt{-g}R(g_{\mu\nu})$ in function of L_0 and L :

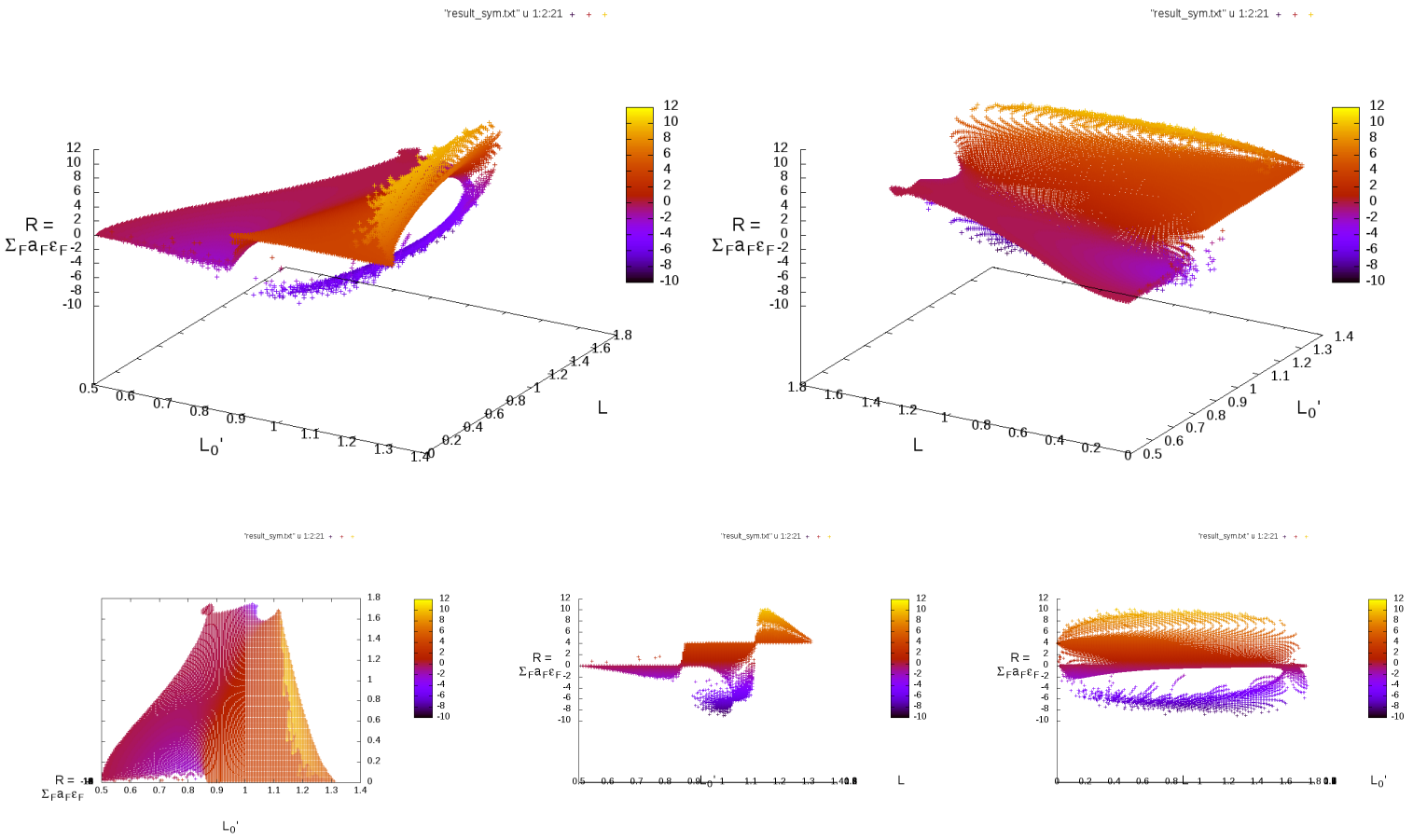


Figure 24: $\mathrm{R} = \sum_{F \subset bulk} a_F \varepsilon_F$ from the Regge's equations solutions in function of L_0' and L under different views. The first and second pictures are the isometric views of the surface of solutions, the third is the top view (in the pane L_0', L), the fourth is the front view (in the plane R, L_0') and the fifth is the side view (in the plane R, L).

Again, the results give curvature which evolve with the boundary parameters L_0', L and fixed $\left(L_0, \frac{1}{2}L_f\right) = \left(1, \frac{1}{2}\right)$. For the section where $L_0' = \frac{\sqrt{3}}{2}$:

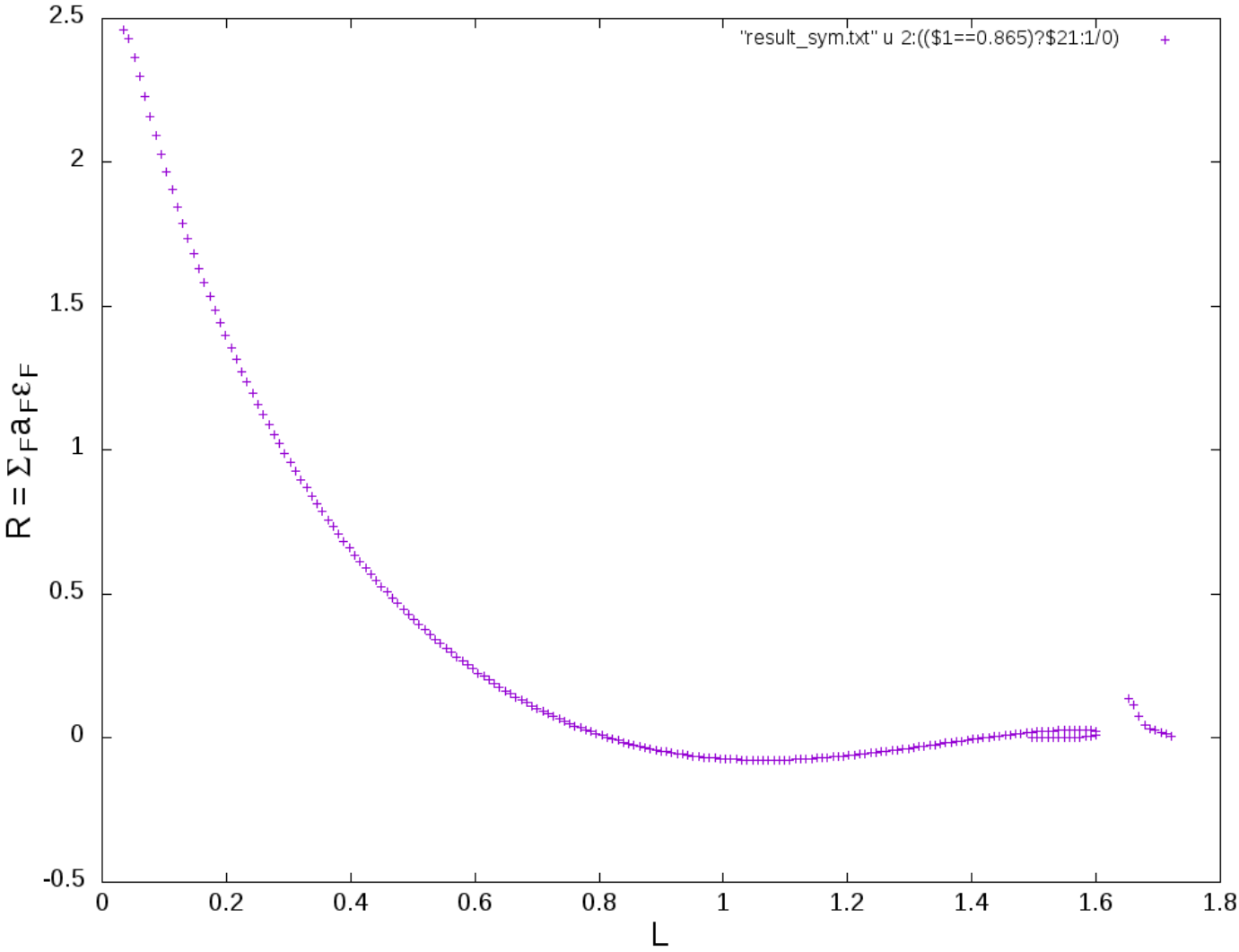


Figure 25: Evolution of $\mathrm{R} = \sum_{F \subset bulk} a_F \varepsilon_F$ from the Regge's equations solutions in function of L.

We find the values of $\mathrm{R} = \sum_{F \subset bulk} a_F \varepsilon_F$ are overall positive, with the flat case at $L = \sqrt{2}$ which give $\mathrm{R} = 0$ and the cases $0.82175 \leq L < \sqrt{2}$ which give small negative values.

3.4 Conclusion about Regge calculus

To summarize, in the last two chapters we have studied Regge calculus on two different triangulations. The first, corresponding to Δ_3 of Figure 6, has a single internal face, but all segments are boundary, thus there are no Regge's equations to be satisfied and the dynamics is trivial. The curvature associated to the internal face is then directly determined by the boundary data. This triangulation defines nonetheless a non-trivial dynamics in spin foams, because there the fundamental variables are areas instead of lengths, thus there will be internal degrees of freedom associated with the area and normals of the internal face. To study the classical dynamics in the Regge setting, we considered a natural refining of Δ_3 , given by Δ_{12} from Figures 17,19, which has 3 internal segments, and thus 3 non trivial dynamical equations to be satisfied. We study the solutions to the equations as a function of simple, axial-symmetric boundary data, obtaining results consistent with a discretization of general relativity, as expected. In particular, the curvature obtained from solving the equations varies continuously with the boundary data and

switches sign as we 'squeeze' the configuration moving from a positively curved bulk, to a flat bulk, to a negatively curved bulk, see Figures 22,23 and 24,25. Moreover, in this specific case and refining, the splitting give a assembly composed by some sub-assemblies which are "like" the study object : so you have no limitation for split again the structure, or imagine the structure inside a bigger assembly, and built more complex objects where Regge calculus is always viable and give non-null curvature !

4 Interlude for the quantum geometry

We show the Regge's geometries of our study object reproduce successfully curvature in function of the boundary data. In our original object given by three 4-simplices we have curvature, but no Regge's dynamics. In the *specific* refined object, given by twelve 4-simplices, we preserve a equivalent boundary and get Regge's dynamics with curvature. But what happen for the quantum geometry ? The need to refine our object in the previous section come to the fact of the Regge's dynamics come to the bulk segments, and our original object had no such segments. Conversely, the quantum geometry and its dynamics is defined by the areas, so we can have quantum geometry dynamics for our original object, on its internal face f, without any refine. It would be interesting to get the quantum dynamics of the refined object, and compare it with the Regge's dynamics, but unfortunately the associated quantum geometry is too difficult to do analytically and numerically : the associated spin-foam, given by the Figure 19, is too much complex to adapt for numerical computation. So let us concentrate to do the quantum definitions of our original study object (Drawing 6 and future spin-network & spin-foam Figures 27,30) and study numerically the quantum dynamics of this one.

5 Quantum geometry

We have shown that our study object can be used with the Regge's formulation and give curvature, this is a first step for understand and see if the quantum theory of geometry can give the same physic of classical geometry. But for have more clues if the quantum geometry contain, in a certain part, the usual geometry and the possible differences we need to apply the Loop Quantum Gravity theory for our object.

The goal of this section it's define the states and transition amplitude for our study object and give it a quantum formulation from the euclidean version of Loop Quantum Gravity theory. We will expose the spin-network formulation, adapt-it for the boundary of our assembly, and give the corresponding proper and coherent states of the geometry. After, we will give the spin-foam and express the transition amplitude. In the transition amplitude the Immirzi parameter γ, from the EPRL model, will be taken with the convenient value $\gamma = \frac{1}{2}$. We make this choice for obtain correct definitions of intertwiners, coherent with the euclidean version of the EPRL model. We will introduce in the next subsections.

5.1 Introduction to Loop Quantum Gravity and EPRL model (see [3, 7, 4, 14, 15])

In covariant Loop Quantum Gravity, states are defined on the 3d boundary of a space-time region. A basis of states is given by the spin-network states, that have support on a graph that can be interpreted as the dual of the 3d discretization of the space. As a dual, the spin-network graph can be seen like a "assembly drawing" of the 3d boundary : the links are the dual of the geometric faces and the nodes are the dual of the assembly of the geometric faces in polyhedra. In the context of a triangulation of 3d boundary, the links represent just the triangles and the nodes represent the tetrahedra. Formally, the quantum states of the boundary will be given as function of group elements associated at each link of the spin-network graph. The quantum parameters associated to the representation of the group will be connected to the classical parameters via some operators, as the area-operators.

The theory associates an amplitude to such boundary states. The amplitude can be computed using the spin-foam expansion: at each order the amplitude is given a by a spin-foam defined on a two-complex whose boundary is the graph of the boundary state. In particular, the spin-foam can be defined on the dual of a triangulation of the space-time region. As the spin-network graph for the 3d regions, the spin-foam graph can be seen like a "assembly drawing" of the 4d pieces of space-time where the edges represent the shared 3d polyhedra and the vertices represent the space-time pieces. For a triangulation of 4d space-time, the edges are just the tetrahedra and the vertices are

4-simplices. Finally, the amplitude will be given as integration over the group elements of the spin-foam graph with its associated spin-network states.

Of course, the all processes define in this introduction of Loop Quantum Gravity will be more explicit when we will use them for our study object.

5.1.1 Spin-network

From the boundary of a quantum geometry, we have a graph Γ dual to the 3d discretization of this boundary called the spin-network graph. The nodes, labeled "k", are the duals of the polyhedra from the boundary of the geometry. The links, labeled "kl" between the nodes k and l, can be seen are the dual of the shared faces between the polyhedra from k and l. The quantum states associated to the boundary geometry are square integral functions $\psi(u_{kl})$ of one $SU(2)$ group variable u_{kl} per each link of the spin-network graph Γ. A basis in their space is given by the spin-network functions :

$$\psi_\Gamma^{j_{kl},\{J\}_k}(u_{kl}) = \prod_{\text{nodes } k} i^{\{J\}_k} \cdot \prod_{\text{links } kl} D^{j_{kl}}(u_{kl}) \tag{62}$$

where j_{kl} are the link spins and $\{J\}_k$ are the intertwiner spins. The $D^{j_{kl}}$ are the j-representations (Wigner's representations) of the $SU(2)$ group elements u_{kl} and, as the intertwiners $i^{\{J\}_k}$, have magnetic indices ; the contraction is dictated by the topology of the graph. The intertwiner $i^{\{J\}_k}$ associated for each node k and can be obtained from the group invariance of the node :

$$\sum_{\{J\}_k} i^{\{J\}_k}_{m_{kl_1}, m_{kl_2}\ldots} i^{\{J\}_k}_{n_{kl_1}, n_{kl_2}\ldots} = \int_{SU(2)} dg_k D^{j_{kl_1}}_{m_{kl_1} n_{kl_1}}(g_k) D^{j_{kl_2}}_{m_{kl_2}, n_{kl_2}}(g_k) \ldots \tag{63}$$

That formula can be interpreted as the invariance of the geometry (like the dipheomorphism invariance) from the node k, and the intertwiner spins $\{J\}_k$ are the quantum numbers associated to the node and geometry properties from the associated polyhedra (like projected area, dihedral angle or volume. See the Subsection 5.2.1 and 5.2.3).

The states $\psi_\Gamma^{j_{kl},\{J\}_k}$ are eigenstates of the area operator $\hat{a_{kl}}$ of the faces dual to the links "kl" :

$$\hat{a_{kl}} \left| \psi_\Gamma^{j_{kl},\{J\}_k} \right\rangle = a_{kl} \left| \psi_\Gamma^{j_{kl},\{J\}_k} \right\rangle \tag{64}$$

$$a_{kl} = \frac{8\pi\gamma\hbar G}{c^3} \sqrt{j_{kl}(j_{kl}+1)} \tag{65}$$

We chose units where $8\pi\gamma\hbar G/c^3 = 1$ so we do not have to carry over the dimensional factor. The group elements u_{kl} correspond physically to the parallel transport (on the boundary) from the node k to the node l ; the conserved quantity correspond to the area, given by j_{kl}, along the parallel transport. That quantum description have strong links with the properties from the (usual) geometry dual to the spin-network graph.

5.1.2 Spin-foam

From the bulk of quantum geometry, which have a boundary defined by its spin-network graph, we have a graph Υ dual to the 4d discretization called spin-foam graph. The vertices, labeled N, are the duals of the 4-polytopes (4d generalization of polyhedra) and the edges are the duals of polyhedra. By definition, the external edges from a spin-foam correspond to the boundary polyhedra from its spin-network graph. The faces of the spin-foam, means loops and external faces (open-loops which are connected to the link from boundary spin-network), are the duals of geometric faces of the geometry.

For each vertex N, we can define a amplitude as a function of $u_{kl}^N \in SU(2)$ variables :

$$A_N\left(u_{kl}^N\right) = \int_{G^{\mathcal{N}}} dU_k^N \prod_{kl \subset \text{Simplex } N} \delta\left(Y^\dagger U_k^N \left(U_l^N\right)^{-1} Y u_{lk}^N\right) \tag{66}$$

where we have a integral over the all group elements $U_k^N \in \mathbb{G}$, from the edges connected to the vertex N ; $\mathcal{N}$ are just the number of edges connected to N and thus the number of copies of $\mathbb{G}$. $\mathbb{G}$ is the group associated to the vertex N and correspond to the group of rotations of the space-time region associated to the dual of N : $\mathbb{G} = SL(2,\mathbb{C})$ for Lorentzian space-time, $\mathbb{G} = SO(4) \simeq SU(2)^+ \times SU(2)^-$ for Euclidean space-time.

The U_k^N and U_l^N can be seen as the group elements associated to polyhedra (edges) "k" and "l" from the vertex N. The u_{kl}^N is the parallel transport element between the polyhedra (edges) "k" and "l", like in the previous spin-network subsection. In fact, the u_{kl}^N variables are the spin-network variables from the boundary of the individual vertex N ; and the Y is the map between the group elements $U_k^N \in \mathbb{G}$ and the group elements $u_{kl}^N \in SU(2)$. The map Y depend of the definition of $\mathbb{G}$, and glue the group representation of U_k^N with the group representation of u_{lk}^N :

$$Y \ : \ \begin{cases} |j,m\rangle = |\gamma j, j; j, m\rangle & \text{for } \mathbb{G} = SL(2,\mathbb{C}) \\ |j,m\rangle = \sqrt{2j+1}\sum_{m^+,m^-} \begin{pmatrix} j^+ & j^- & j \\ m^+ & m^- & m \end{pmatrix} |j^+,m^+\rangle \otimes |j^-,m^-\rangle \ ; \ j^\pm = \frac{1\pm\gamma}{2} j & \text{for } \mathbb{G} = SO(4) \end{cases} \tag{67}$$

With the definition of the $SU(2)$-delta function :

$$\delta(\bullet) \coloneqq \sum_j (2j+1)\, T_r\left[D^j(\bullet)\right] \tag{68}$$

We have formally :

$$\delta\left(Y^+ U_k^N \left(U_l^N\right)^{-1} Y u_{lk}^N\right) = \begin{cases} \sum_{j_{kl}} (2j_{kl}+1) \sum_{m_{kl}} \sum_{n_{kl}} D^{\gamma j_{kl}, j_{kl}}_{j_{kl} m_{kl}, j_{kl} n_{kl}} \left(U_k^N \left(U_l^N\right)^{-1}\right) D^{j_{kl}}_{m_{kl} n_{kl}}\left(u_{lk}^N\right) & \text{for } \mathbb{G} = SL(2,\mathbb{C}) \\ \sum_{j_{kl}} (2j_{kl}+1)^2 \sum_{m_{kl}^\pm m_{kl}} \sum_{n_{kl}^\pm n_{kl}} \begin{pmatrix} j_{kl}^+ & j_{kl}^- & j_{kl} \\ m_{kl}^+ & m_{kl}^- & m_{kl} \end{pmatrix} \begin{pmatrix} j_{kl}^+ & j_{kl}^- & j_{kl} \\ n_{kl}^+ & n_{kl}^- & n_{kl} \end{pmatrix} & \\ \quad \times D^{j_{kl}^+}_{m_{kl}^+ n_{kl}^+}\left(u_k^{N\,+}\left(u_l^{N\,+}\right)^{-1}\right) D^{j_{kl}^-}_{m_{kl}^- n_{kl}^-}\left(u_k^{N\,-}\left(u_l^{N\,-}\right)^{-1}\right) D^{j_{kl}}_{m_{kl} n_{kl}}\left(u_{lk}^N\right) & ; \ j^\pm = \frac{1\pm\gamma}{2} j \quad \text{for } \mathbb{G} = SO(4) \end{cases} \tag{69}$$

Here we see the first-fruits of the problem associated to γ for the Euclidean case : the $j^\pm$ must be integer or half-integer, that implies specific values of γ and j ; we will talk more about that when we will apply the process in our study object.

The full transition amplitude W_Υ from the spin-foam Υ is given by the integration over the all u_{kl}^N with $SU(2)$-delta functions for glue the all vertices :

$$W_\Upsilon(u_{kl}) = \int du_{kl}^N \prod_N A_N\left(u_{kl}^N\right) \prod_{\text{loops} \subset \Upsilon} \delta\left(\prod_{(ab,N)\subset \text{loop}} u_{ab}^N\right) \prod_{f_{\text{external}} \subset \Upsilon} \delta\left(u_{kl} \prod_{(ab,N)\subset f_{\text{external}}} u_{ab}^N\right) \tag{70}$$

where we have a product of δ-functions for each loops from the spin-foam, that give the delta function of the oriented product of the u_{ab}^N elements from the loop, and we have a product of δ-functions for each external faces which connect the residual u_{ab}^N with the u_{kl} from the boundary.

5.1.3 Transition amplitude

With the state from the spin-network boundary Γ and the amplitude of the associated spin-foam Υ, we can express the transition amplitude of the geometry state :

$$\left\langle W_\Upsilon | \psi_\Gamma^{j_{kl},\{J\}_k} \right\rangle = \int du_{kl} W_\Upsilon(u_{kl})\, \psi_\Gamma^{j_{kl},\{J\}_k}(u_{kl}) \tag{71}$$

which represent the quantum evolution of the geometry state $\psi_\Gamma^{j_{kl},\{J\}_k}$ from the boundary with the quantum geometrical constraint inside the definition of W_Υ. We have the geometric properties associated to the boundary, via the basis state of spin-network, and the quantum summation over the all possible bulk geometries, via the spin-foam and its integrals from the vertex amplitudes.

5.2 spin-network of our objects

Here we consider a simple case, where the 4d triangulation is formed by the assembly of three 4-simplices. Figures 26, 27 gives respectively the graph of the triangulation of the boundary of 4-simplex and 4-simplices assembly. Although formally similar, this represents actually the graphs *duals* to the boundary of these Figures 5a, 6 : points represent tetrahedra and lines represents triangles. The quantum states will be spin-network functions

$$\psi^{j_{kl},J_k}(u_{kl}) = \prod_{\text{nodes } k} i^{J_k} \cdot \prod_{\text{links } kl} D^{j_{kl}}(u_{kl}) \tag{72}$$

where j_{kl} are spins and J_k intertwiner spins for 4-valent intertwiners. The group variable u_{kl} can be understood like the group element to make the parallel transport (on the boundary) of tetrahedron τ_k to a tetrahedron τ_l through them shared face of area $a_{kl} = \sqrt{j_{kl}(j_{kl}+1)}$.

5.2.1 spin-networks for the individual cylindrical 4-simplices

For each individual 4-simplex N, we begin implementing the cylindrical symmetry by choosing boundary states where, as in 2.2 (and more especially from (9)),

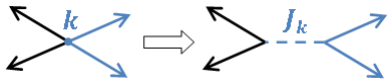
$$j^N_{12} = j^N_{23} = j^N_{31} \equiv j, \qquad j^N_{14} = j^N_{24} = j^N_{34} = j^N_{15} = j^N_{25} = j^N_{35} \equiv j_0 \qquad \text{and} \qquad j^N_{45} \equiv j_f. \tag{73}$$

The integers or half-integers j, j_0, j_f are the quantum equivalent of the areas a, a_0, a_f.

Let us now come to the intertwiners. For the intertwiners between four representations $j_1, ..., j_4$, we use a basis defined by

$$i^J_{m_1m_2m_3m_4} = \sqrt{2J+1}\sum_M (-1)^{J-M} \begin{pmatrix} j_1 & j_2 & J \\ m_1 & m_2 & M \end{pmatrix} \begin{pmatrix} j_3 & j_4 & J \\ m_3 & m_4 & -M \end{pmatrix}, \tag{74}$$

where the $\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix}$ are the Wigner 3j-symbols defining the 3-valent invariant of $SU(2)$. In the case of the boundary tetrahedra (we say boundary in the context of the boundary of the full assembly) we pair the faces with the same area and write



$$i^{J^N_k}_{m_1m_2m_3m_4} = \sqrt{2J^N_k+1}\sum_M (-1)^{J^N_k-M} \begin{pmatrix} j & j & J^N_k \\ m_1 & m_2 & M \end{pmatrix} \begin{pmatrix} j_0 & j_0 & J^N_k \\ m_3 & m_4 & -M \end{pmatrix} \qquad (\text{for } k=1,2,3 \text{ and } \forall N)$$

While for the shared-tetrahedra (which are internal for the full assembly, but belonging to the boundary of individual 4-simplex), we define the matching and the intertwiners as follows

$$i^{J^N_k}_{m_1m_2m_3m_4} = \sqrt{2J^N_k+1}\sum_M (-1)^{J^N_k-M} \begin{pmatrix} j_f & j_0 & J^N_k \\ m_1 & m_2 & M \end{pmatrix} \begin{pmatrix} j_0 & j_0 & J^N_k \\ m_3 & m_4 & -M \end{pmatrix} \qquad (\text{for } k=4,5 \text{ and } \forall N)$$

The intertwiners $i^{J^N_k}$ associated to a node determines the quantum geometry of the tetrahedron τ^N_k. The number J^N_k, integer or half-integer, is the quantum number equivalent of the projected area A^N_k (We can see the link between the two aspects in the coherent states Subsection 5.2.3). The following graph illustrates the quantum numbers defining the spin-network and the chosen pairings for the intertwiners :

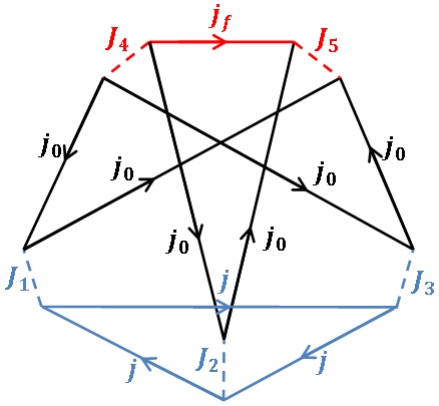


Figure 26: Spin-network of individual 4-Simplex with the specification of the spins

Explicitly, the spin-network eigenstates for each individual 4-simplex N are :

$$\begin{aligned}\psi_N^{j_0,j,j_f,J_k^N}\left(u_{kl}^N\right) &= \sum_{m,n}(-1)^{\sum(j_{kl}^N-n_{kl})} i^{J_1^N}_{-n_{12}m_{31}-n_{14}m_{51}} i^{J_2^N}_{-n_{23}m_{12}-n_{24}m_{52}} i^{J_3^N}_{-n_{31}m_{23}-n_{34}m_{53}} i^{J_4^N}_{m_{54}m_{24}m_{34}m_{14}} i^{J_5^N}_{-n_{54}-n_{52}-n_{53}-n_{51}} \\ &\times D^{j_0}_{m_{14}n_{14}}\left(u_{14}^N\right) D^{j_0}_{m_{24}n_{24}}\left(u_{24}^N\right) D^{j_0}_{m_{34}n_{34}}\left(u_{34}^N\right) D^{j_0}_{m_{51}n_{51}}\left(u_{51}^N\right) D^{j_0}_{m_{52}n_{52}}\left(u_{52}^N\right) D^{j_0}_{m_{53}n_{53}}\left(u_{53}^N\right) \\ &\times D^{j}_{m_{12}n_{12}}\left(u_{12}^N\right) D^{j}_{m_{23}n_{23}}\left(u_{23}^N\right) D^{j}_{m_{31}n_{31}}\left(u_{31}^N\right) D^{j_f}_{m_{54}n_{54}}\left(u_{54}^N\right)\end{aligned} \tag{75}$$

These states are eigenstates of the area operators of the boundary :

$$\hat{a}_{kl}^N\left|\psi^{j_0,j,j_f,J_k^N}\right\rangle = \sqrt{j_{kl}^N\left(j_{kl}^N+1\right)}\left|\psi^{j_0,j,j_f,J_k^N}\right\rangle = \begin{cases}\sqrt{j\left(j+1\right)}\left|\psi^{j_0,j,j_f,J_k^N}\right\rangle & \text{for } k,l=1,2,3 \\ \sqrt{j_f\left(j_f+1\right)}\left|\psi^{j_0,j,j_f,J_k^N}\right\rangle & \text{for } kl=45 \\ \sqrt{j_0\left(j_0+1\right)}\left|\psi^{j_0,j,j_f,J_k^N}\right\rangle & \text{else}\end{cases} \tag{76}$$

and satisfy the orthogonality relation :

$$\left\langle \psi_N^{j_0',j',j_f',J_k^{N\prime}} | \psi_N^{j_0,j,j_f,J_k^N}\right\rangle = \int_{SU(2)} du^N \overline{\psi^{j_0',j',j_f',J_k^{N\prime}}\left(u_{kl}^N\right)}\psi^{j_0,j,j_f,J_k^N}\left(u_{kl}^N\right) = \frac{\delta_{j_0,j_0'}\delta_{j,j'}\delta_{j_f,j_f'}}{\left(2j_0+1\right)^6\left(2j+1\right)^3\left(2j_f+1\right)}\prod_{k=1}^{5}\delta_{J_k^N,J_k^{N\prime}} \tag{77}$$

5.2.2 spin-networks for the boundary of the assembly

For our assembly, the spin-network associated to its boundary depend only of the boundary tetrahedra (without the shared-tetrahedra) which depend of the j, j_0, $\left\{J_k^N\right\}_{k=1,2,3}$ parameters and corresponding intertwiners. With the same pairing and definitions of intertwiners than previously, we can use the following graph showing the spin-network of the boundary from assembly :

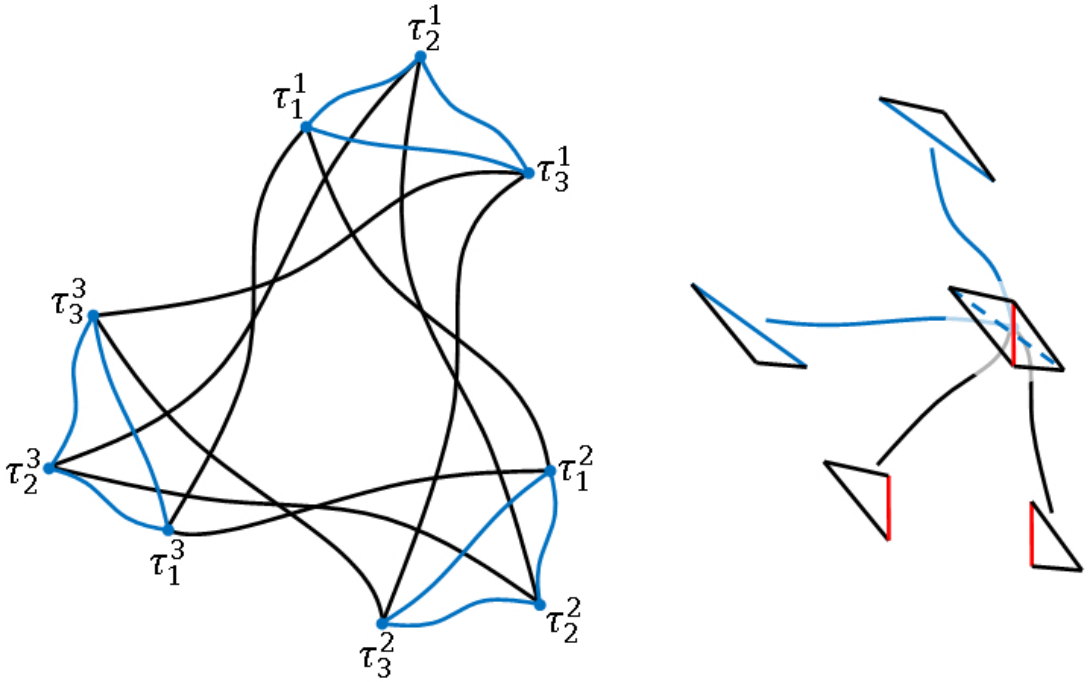


Figure 27: Spin-network of the assembly's boundary. The black links represents the faces f_0 with their associated spin j_0, and the blue links represents the faces f_a with their associated spin j. We see the associated pieces of geometry of the graph in the right.

Explicitly, with $N = 1, 2, 3$ for distinguish the 4-simplices, the boundary spin-network eigenstates are

$$\begin{aligned}
\Psi^{j,j_0,J_k^N}\left(u_{kl}^N, u_k^{NN'}\right) \quad = \sum_{m,n}(-1)^{\sum(j_{kl}^N - n_{kl})} \quad & i^{J_1^1}_{-n_{12}^1 m_{31}^1 - n_{14}^1 m_{51}^1} i^{J_2^1}_{-n_{23}^1 m_{12}^1 - n_{24}^1 m_{52}^1} i^{J_3^1}_{-n_{31}^1 m_{23}^1 - n_{34}^1 m_{53}^1} \\
& \times i^{J_1^2}_{-n_{12}^2 m_{31}^2 - n_{14}^2 m_{51}^2} i^{J_2^2}_{-n_{23}^2 m_{12}^2 - n_{24}^2 m_{52}^2} i^{J_3^2}_{-n_{31}^2 m_{23}^2 - n_{34}^2 m_{53}^2} \\
& \times i^{J_1^3}_{-n_{12}^3 m_{31}^3 - n_{14}^3 m_{51}^3} i^{J_2^3}_{-n_{23}^3 m_{12}^3 - n_{24}^3 m_{52}^3} i^{J_3^3}_{-n_{31}^3 m_{23}^3 - n_{34}^3 m_{53}^3} \\
& \times D^{j_0}_{m_{53}^1 n_{34}^2}\left(u_3^{12}\right) D^{j_0}_{m_{52}^1 n_{24}^2}\left(u_2^{12}\right) D^{j_0}_{m_{51}^1 n_{14}^2}\left(u_1^{12}\right) \\
& \times D^{j_0}_{m_{53}^2 n_{34}^3}\left(u_3^{23}\right) D^{j_0}_{m_{52}^2 n_{24}^3}\left(u_2^{23}\right) D^{j_0}_{m_{51}^2 n_{14}^3}\left(u_1^{23}\right) \\
& \times D^{j_0}_{m_{53}^3 n_{34}^1}\left(u_3^{31}\right) D^{j_0}_{m_{52}^3 n_{24}^1}\left(u_2^{31}\right) D^{j_0}_{m_{51}^3 n_{14}^1}\left(u_1^{31}\right) \\
& \times D^{j}_{m_{12}^1 n_{12}^1}\left(u_{12}^1\right) D^{j}_{m_{23}^1 n_{23}^1}\left(u_{23}^1\right) D^{j}_{m_{31}^1 n_{31}^1}\left(u_{31}^1\right) \\
& \times D^{j}_{m_{12}^2 n_{12}^2}\left(u_{12}^2\right) D^{j}_{m_{23}^2 n_{23}^2}\left(u_{23}^2\right) D^{j}_{m_{31}^2 n_{31}^2}\left(u_{31}^2\right) \\
& \times D^{j}_{m_{12}^3 n_{12}^3}\left(u_{12}^3\right) D^{j}_{m_{23}^3 n_{23}^3}\left(u_{23}^3\right) D^{j}_{m_{31}^3 n_{31}^3}\left(u_{31}^3\right)
\end{aligned} \tag{78}$$

Where u_{kl}^N are the $SU(2)$ group elements associated to the j-faces in the 4-simplex N, and the $u_k^{NN'}$ are the $SU(2)$ group elements of the j_0-faces from the shared-tetrahedra between the 4-simplices N and N'. In fact, we can construct its boundary states Ψ with the states ψ_N of the individual 4-simplices :

$$(-1)^{\sum_{k,N} j_{4k}^N} \Psi^{j_{kl}^N, J_k^N}\left(u_{kl}^N, u_k^{NN'}\right) \delta_{J_5^1 J_4^2} \delta_{J_5^2 J_4^3} \delta_{J_5^3 J_4^1} \sum_{j_f} \frac{\delta_{j_f, j_{45}^N}}{(2j_f+1)^2} \prod_{k,NN'} \frac{\delta_{j_{4k}^N, j_{5k}^{N'}}}{2j_{4k}^N + 1} \tag{79}$$

$$= \sum_{J_4,J_5} \int du_{5k} du_{4k} \quad \psi_1^{j_{kl}^1,J_k^1}\left(u_{kl}^1\right) \psi_2^{j_{kl}^2,J_k^2}\left(u_{kl}^2\right) \psi_3^{j_{kl}^3,J_k^3}\left(u_{kl}^3\right)$$
$$\times\delta\left(u_{53}^1 u_{34}^2 u_3^{21}\right) \delta\left(u_{52}^1 u_{24}^2 u_2^{21}\right) \delta\left(u_{51}^1 u_{14}^2 u_1^{21}\right)$$
$$\times\delta\left(u_{53}^2 u_{34}^3 u_3^{32}\right) \delta\left(u_{52}^2 u_{24}^3 u_2^{32}\right) \delta\left(u_{51}^2 u_{14}^3 u_1^{32}\right)$$
$$\times\delta\left(u_{53}^3 u_{34}^1 u_3^{13}\right) \delta\left(u_{52}^3 u_{24}^1 u_2^{13}\right) \delta\left(u_{51}^3 u_{14}^1 u_1^{13}\right)$$
$$\times\delta\left(u_{54}^1 u_{54}^2 u_{54}^3\right)$$

Each ψ_N states represent the boundary of the individual pieces of the assembly, the Kronecker symbols δ_{AB} and the $SU(2)$-delta function $\delta(\bullet) = \sum_j (2j+1) Tr\left[D^j(\bullet)\right]$ allow to match the shared-tetrahedra and corresponding group elements between the pieces. These states, by construction, are also eigenstates of area operator of the assembly's boundary :

$$\hat{a}_{kl}^N \left|\Psi^{j,j_0,J_k^N}\right\rangle = \sqrt{j_{kl}^N\left(j_{kl}^N+1\right)} \left|\Psi^{j,j_0,J_k^N}\right\rangle = \begin{cases} \sqrt{j(j+1)} \left|\Psi^{j,j_0,J_k^N}\right\rangle & \text{for } k,l = 1,2,3 \\ \sqrt{j_0(j_0+1)} \left|\Psi^{j,j_0,J_k^N}\right\rangle & \text{else} \end{cases} \tag{80}$$

And respect the orthogonality relation :

$$\left\langle \Psi^{j',j_0',J_k^{N\prime}} | \Psi^{j,j_0,J_k^N} \right\rangle = \int_{SU(2)} du \overline{\Psi^{j',j_0',J_k^{N\prime}}\left(u_{kl}^N, u_k^{NN\prime}\right)} \psi^{j_0,j,j_f,J_k^N}\left(u_{kl}^N, u_k^{NN\prime}\right) = \frac{\delta_{j_0,j_0'}\delta_{j,j'}}{(2j_0+1)^9 (2j+1)^9} \prod_{N=1,k=1}^{3,3} \delta_{J_k^N,J_k^{N\prime}} \tag{81}$$

5.2.3 Coherent states

The spin-network states defined in the previous section are eigenstates of the projected area A_k^N of the tetrahedra, and are therefore completely spread in the corresponding angles Φ_k^N, which do not commute with A_k^N. Therefore they are very non-classical. We are interested, instead, in wave packets that are minimally spread *both* in A_k^N and in Φ_k^N. To this aim, we use the (intrinsic) coherent states defined by Livine and Speziale [20]. These are defined as follows. The coherent link states are defined by

$$|j\overrightarrow{n}\rangle = R(\overrightarrow{n})\,|j,j\rangle = \sum_m D_{mj}^j\left(R(\overrightarrow{n})\right)|j,m\rangle \tag{82}$$

where $\overrightarrow{n}$ is the normal vector to a face of tetrahedron with area j. The group element $R(\overrightarrow{n})$ is a rotation than maps the vector $\overrightarrow{u_z}$ into the normal vector $\overrightarrow{n}$:

$$R(\overrightarrow{n}) \cdot \overrightarrow{u_z} = \overrightarrow{n} \tag{83}$$

For a tetrahedron with vectors $\overrightarrow{n_i}$ associated to its faces, the Livine-Speziale state is:

$$|j_i\overrightarrow{n_i}\rangle = \sum_m D_{m_1 j_1}^{j_1}\left(R(\overrightarrow{n_1})\right) D_{m_2 j_2}^{j_2}\left(R(\overrightarrow{n_2})\right) D_{m_3 j_3}^{j_3}\left(R(\overrightarrow{n_3})\right) D_{m_4 j_4}^{j_4}\left(R(\overrightarrow{n_4})\right) |j_1,m_1\rangle \otimes |j_2,m_2\rangle \otimes |j_3,m_3\rangle \otimes |j_4,m_4\rangle \tag{84}$$

And the projection of this state on the corresponding intertwiner gives:

$$\langle i^J | j_i\overrightarrow{n_i}\rangle = \sum_m i^J_{m_1 m_2 m_3 m_4} D_{m_1 j_1}^{j_1}\left(R(\overrightarrow{n_1})\right) D_{m_2 j_2}^{j_2}\left(R(\overrightarrow{n_2})\right) D_{m_3 j_3}^{j_3}\left(R(\overrightarrow{n_3})\right) D_{m_4 j_4}^{j_4}\left(R(\overrightarrow{n_4})\right) \tag{85}$$

Writing $\overrightarrow{n} = (\cos\phi\sin\theta, \sin\phi\sin\theta, \cos\theta)$ from the spherical coordinates system, we have

$$R(\overrightarrow{n}) = R(\theta,\phi) = e^{-\imath\phi J_Z} e^{-\imath\theta J_Y} \tag{86}$$

Where J_Z and J_Y are the generators of (usual) rotations. With this choice of R, we can express the j-representation :

$$\begin{aligned} D_{mj}^j\left(R(\overrightarrow{n})\right) = D_{mj}^j(\theta,\phi) = D_{mj}^j\left(e^{-\imath\phi J_Z} e^{-\imath\theta J_Y}\right) = e^{-\imath m\phi} d_{mj}^j(\theta) \\ = \sqrt{\tfrac{(2j)!}{(j+m)!(j-m)!}} \cdot \tfrac{\xi^{j-m}}{\left(1+|\xi|^2\right)^j} \cdot e^{-\imath j\phi} \qquad ; \qquad \xi = \tan\left(\tfrac{\theta}{2}\right) e^{\imath\phi} \end{aligned} \tag{87}$$

Where the d are the little Wigner matrices. The expression of Livine-Speziale state with his intertwiner became :

$$\left\langle i^J | j_i \overrightarrow{n_i} \right\rangle = \left(\prod_{i=1}^{4} \frac{e^{-\imath j_i \phi_i} \sqrt{(2j_i)!}}{\left(1+|\xi_i|^2\right)^{j_i}} \right) \sum_m i^J_{m_1 m_2 m_3 m_4} \prod_{i'=1}^{4} \frac{{\xi_{i'}}^{j_{i'}-m_{i'}}}{\sqrt{(j_{i'}+m_{i'})!(j_{i'}-m_{i'})!}} \tag{88}$$

Which is physically the distribution of the coherent states of tetrahedron geometry with the normal face-vectors $\overrightarrow{n_i}$ and their associated areas j_i over the intertwiner basis i^J. This distribution over the intertwiner spin J depend of the four j-areas and five angles: we have four set of (θ, ϕ) variables, one for each $\overrightarrow{n_i}$, but with the invariance under the rotations (gauge fixing) we can fix three of them. If the tetrahedron geometry is classical, that means it respect the closure condition $\sum_i a_i \overrightarrow{n_i} = \overrightarrow{0} \sim \sum_i j_i \overrightarrow{n_i} = \overrightarrow{0}$, we can reduce the angles parameters to the shape parameters A, Φ as in the classical Subsection 2.1 and Annexes A.

In a assembly of tetrahedra, as in our study object, we will take the precise notation for the Livine-Speziale of the tetrahedron τ_k^N :

$$\left\langle i^{J_k^N} | j_{kl}^N \overrightarrow{n_{kl}^N} \right\rangle = \sum_m i^{J_k^N}_{m_{kl_1} m_{kl_2} m_{kl_3} m_{kl_4}} \prod_l D^{j_{kl}^N}_{m_{kl} j_{kl}^N} \left(\theta_{kl}^N, \phi_{kl}^N\right) \tag{89}$$

In the cylindrical symmetric setting, where the tetrahedra respect the closure condition $\sum_l a_{kl}^N \overrightarrow{n_{kl}^N} = \overrightarrow{0} \sim \sum_l j_{kl}^N \overrightarrow{n_{kl}^N} = \overrightarrow{0}$, we have two types of Livine-Speziale distributions:

$$\left\langle i^{J_k^N} | j, j_0, A, \Phi \right\rangle \equiv \left\langle i^{J_k^N} | j_{kl}^N \overrightarrow{n_{kl}^N} \left(A_k^N, \Phi_k^N\right) \right\rangle \qquad \text{for the boundary tetrahedra } (k=1,2,3) \text{ from the 4-simplex } N \tag{90}$$

$$\left\langle i^{J_k^N} | j_f, j_0, A_f, \Phi_f \right\rangle \equiv \left\langle i^{J_k^N} | j_{kl}^N \overrightarrow{n_{kl}^N} \left(A_k^N, \Phi_k^N\right) \right\rangle \qquad \text{for the shared tetrahedra } (k=4,5) \text{ from the 4-simplex } N \tag{91}$$

These states are peaked around the classical geometry define by the variables (j_i, A, Φ) of each tetrahedron. A example of these distribution over the J parameter can be seen in the next Figure for $j_i = 8$, $A = \frac{2}{\sqrt{3}} j \approx 9.24$, $\Phi = \frac{\pi}{2}$ parameters :

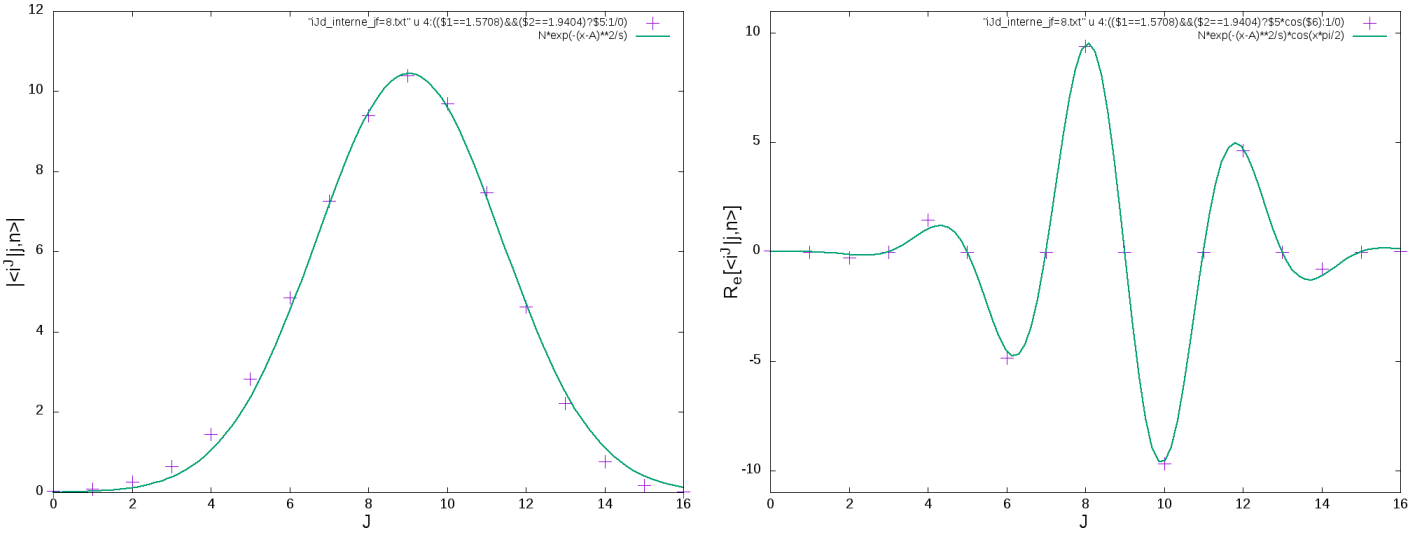


Figure 28: The norm (in the left) and the real part (in the right) from the Livine-Speziale distribution with $j_i = 8$, $A = \frac{2}{\sqrt{3}} j \approx 9.24$, $\Phi = \frac{\pi}{2}$. The points are the exact values of the Livine-Speziale distribution (define only for $J \in \frac{1}{2}\mathbb{N}$, and here in this special case $J \in \mathbb{N}$), and the green lines are the approximation in terms of Gaussian and complex phase.

Approximately:

$$\left\langle i^J | j_i \overrightarrow{n_i} \left(A, \Phi\right)\right\rangle \sim N\left(j_i, A, \Phi\right)\left(-1\right)^J e^{-iJ\Phi} e^{-\frac{(J-A)^2}{2\sigma^2(j_i,A)}} \tag{92}$$

These states

$$\psi_N^{j_{kl}^N, A_k^N, \Phi_k^N}\left(u_{kl}^N\right) = \sum_{J_k} \psi^{j_{kl}^N, J_k^N}\left(u_{kl}^N\right) \prod_k \left\langle i^{J_k^N} | j_{kl}^N \overrightarrow{n_{kl}^N}\left(A_k^N, \Phi_k^N\right)\right\rangle \tag{93}$$

$$\Psi^{j_{kl}^N, A_k^N, \Phi_k^N}\left(u_{kl}^N, u_k^{NN'}\right) = \sum_{J_k} \Psi^{j_{kl}^N, J_k^N}\left(u_{kl}^N, u_k^{NN'}\right) \prod_{k,N} \left\langle i^{J_k^N} | j_{kl}^N \overrightarrow{n_{kl}^N}\left(A_k^N, \Phi_k^N\right)\right\rangle \tag{94}$$

approximate the intrinsic classical geometry, respectively, of the 4-simplices and the assembly. The J appear to be the quantum equivalent of A as well ; the distribution is maximum for $J \sim A$ with the Gaussian part, which become a delta-function in the limit $j \longrightarrow \infty$.

A interesting feature of the Livine-Speziale distribution is the complex conjugate give the inversion symmetry of the tetrahedron geometry. That come from the properties of Wigner j-representation :

$$\begin{aligned} \overline{D_{mj}^j\left(R\left(\overrightarrow{n}\right)\right)} = \overline{D_{mj}^j\left(\theta,\phi\right)} = e^{\imath m\phi} d_{mj}^j(\theta) = D_{mj}^j\left(\theta,-\phi\right) \\ = e^{-\imath\pi m} e^{-\imath(-m)(\pi+\phi)} d_{-mj}^j(\pi-\theta) = e^{-\imath\pi m} D_{-mj}^j\left(\pi-\theta,\pi+\phi\right) = e^{-\imath\pi m} D_{-mj}^j\left(R\left(-\overrightarrow{n}\right)\right) \end{aligned} \tag{95}$$

which give :

$$\begin{aligned} \overline{\left\langle i^J | j_i \overrightarrow{n_i}\right\rangle} &= \sum_m i^J_{m_1 m_2 m_3 m_4} \prod_{i=1}^4 \overline{D^{j_i}_{m_i j_i}\left(R\left(\overrightarrow{n_i}\right)\right)} = \sum_m i^J_{m_1 m_2 m_3 m_4} (-1)^{\sum_i m_i} \prod_{i=1}^4 D^{j_i}_{-m_i j_i}\left(R\left(-\overrightarrow{n_i}\right)\right) \\ &= \sum_m (-1)^{\sum_i j_i} i^J_{-m_1 -m_2 -m_3 -m_4} \prod_{i=1}^4 D^{j_i}_{-mj_i}\left(R\left(-\overrightarrow{n}\right)\right) = (-1)^{\sum_i j_i} \left\langle i^J | j_i - \overrightarrow{n_i}\right\rangle \end{aligned} \tag{96}$$

We have the link between a coherent state to this complex conjugate by the inversion of the vectors : $|j_i \overrightarrow{n_i}\rangle \propto \langle j_i - \overrightarrow{n_i}|$. That have physical sens, because the complex conjugate correspond to a time inversion. If you have a sub-region of space which is oriented in space-time, the inversion of time reverse the orientation of the sub-region : you can see that as the PT symmetry of geometry, where the inversion of time come with a inversion of space.

5.3 Spin-foam and transition amplitude

In LQG, the spin-foam graph will represent how "build" your space-time with the 3d boundary given by the spin-network. Each external-lines will represent the boundary tetrahedra *dual* to the nodes from the spin-network (associated to the boundary, where the boundary states are defined). And each internal-lines will represent the internal tetrahedra *dual* to the "intermediate" spin-network states (see [3, 7, 4, 14, 15]). For example, Figure 29 give the "assembly drawing" of tetrahedra from its boundary (external-lines in blue) with the internal tetrahedra (internal lines in red) in assembly of 4-simplex. For our simple case, where the vertices represent just 4-simplices, we have only five-valents vertices for the spin-foam graphs.

The covariant LQG amplitude is a function of $SU(2)$ group elements u_{kl} from the associated boundary spin-network. In our euclidean 4-dimentional space, we have 5 copies of $SO(4) \simeq SU(2)^+ \times SU(2)^-$ for each vertices/4-simplices amplitude and some integrals over $SU(2)$-delta function for gluing the vertices and the group elements u_{kl} from the spin-networks. We will describe these steps in the following.

5.3.1 Spin-foam for the individual 4-simplices an specific value of γ

We now construct the amplitude of one individual 4-simplex associated to its boundary state constructed above. This is given by a single vertex, five edges (See Figure 29) and ten faces.

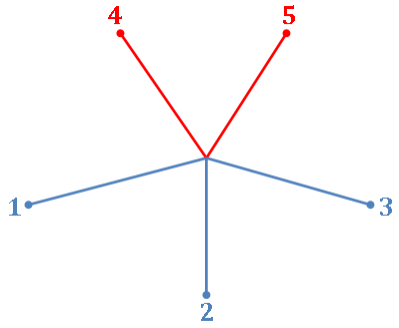


Figure 29: Spin-foam for one individual 4-simplex. The red edges will correspond to shared-tetrahedra, and blue edges to boundary tetrahedra.

The red edges correspond to the shared-tetrahedra (internal in the full assembly, but belonging to the boundary of the individual 4-simplex) and the blue edges are the boundary tetrahedra (in the context of the full assembly) ; they are connected at single the 4-simplex vertex.

The covariant LQG 4-simplex amplitude is a function of an $SU(2)$ group element u_{kl}^N per each face shared between the tetrahedra τ_k^N, τ_l^N in the 4-simplex N. It is defined as an integral over 5 copies of $SO(4) \simeq SU(2)^+ \times SU(2)^-$ as follows:

$$\int_{(SO(4))^5} dU_k^N \prod_{kl} \delta\left(Y^+ U_k^N \left(U_l^N\right)^{-1} Y u_{lk}^N\right) \tag{97}$$

Where U_k^N are the $SO(4)$ group elements associated to edges and their dual tetrahedra, δ the $SU(2)$-delta function, and Y the map between the $SO(4) \simeq SU(2)^+ \times SU(2)^-$ bulk variables and the $SU(2)$ boundary variables :

$$Y \ : \ |j,m\rangle = \sqrt{2j+1} \sum_{m^+,m^-} \begin{pmatrix} j^+ & j^- & j \\ m^+ & m^- & m \end{pmatrix} |j^+,m^+\rangle \otimes |j^-,m^-\rangle \tag{98}$$

With $j^\pm = \frac{1}{2}(1 \pm \gamma) j$ given by the *Immirzi parameter* γ*, which be taken equal to* $\frac{1}{2}$. That choice is justified by the fact that if you take $\gamma = \frac{1}{2}$ and a even number for j, the values of $j^\pm$ become integer or half-integer ! That remove the problem of the map, but need to use even number for j. For the case where j can be not even, as in the sum over the j_f in the full transition amplitude (106), we choose to take for j^+ and j^- the integers or half-integers closest to the theoretical values $\frac{1\pm\gamma}{2}$ with the constraint $j^+ + j^- = j$. We don't know how bad this choice is, but the reader (because the properties of the Wigner 3j-symbols) can choose to consider that the corresponding math objects are nulls in these cases.

Like in the article [29] vertices amplitude parts give :

$$\int_{(SO(4))^5} dU_k^N \prod_{kl} \delta\left(Y^+ U_k^N \left(U_l^N\right)^{-1} Y u_{lk}^N\right) = \sum_{j^N} \left(\prod_{kl} \left(2j_{kl}^N + 1\right)\right) \sum_{K^N} \left[K_k^N, j_{kl}^N\right] \overline{\psi_N^{j_{kl}^N, K_k^N}\left(u_{kl}^N\right)} \tag{99}$$

With the $SO(4)$ 15j-symbols :

$$\left[K_k^N, j_{kl}^N\right] = \sum_{K^\pm} \left(K^+{}_k^N, j^+{}_{kl}^N\right) \left(K^-{}_k^N, j^-{}_{kl}^N\right) \prod_k \mathcal{I}^{K_k^N}_{K^+{}_k^N \, K^-{}_k^N,} \left(j_{kl}^N\right) \tag{100}$$

Given by the $SU(2)$ 15j-symbols (K_k, j_{kl}) and the fusion coefficients $\mathcal{I}^K_{K^+,K^-}(j_a)$:

$$(K_k, j_{kl}) = \sum_p (-1)^{\sum_{kl}(j_{kl}-p_{kl})} i^{K_1}_{-p_{12}p_{13}-p_{14}p_{15}} i^{K_2}_{-p_{23}p_{12}-p_{24}p_{25}} i^{K_3}_{-p_{13}p_{23}-p_{34}p_{35}} i^{K_4}_{p_{45}p_{24}p_{34}p_{14}} i^{K_5}_{-p_{45}-p_{25}-p_{35}-p_{15}} \tag{101}$$

$$\mathcal{I}^{K}_{K^+,K^-}(j_a) = \sum_{m,m^+,m^-} i^{K}_{m_1 m_2 m_3 m_4}(j_a) i^{K^+}_{m_1^+ m_2^+ m_3^+ m_4^+}(j_a^+) i^{K^-}_{m_1^- m_2^- m_3^- m_4^-}(j_a^-) \prod_{a=1}^{4} \sqrt{2j_a+1} \begin{pmatrix} j_a^+ & j_a^- & j_a \\ m_a^+ & m_a^- & m_a \end{pmatrix} \tag{102}$$

We have the amplitude of a individual 4-simplex express in terms of its boundary eigenstates $\psi_N^{j^N_{kl},K^N_k}$ and $SO(4)$ 15j-symbols. That really important because we have a quantum definition of geometry with its geometries states given by $\psi_N^{j^N_{kl},K^N_k}$ and their weight given by the $SO(4)$ 15j-symbols $\left[K^N_k, j^N_{kl}\right]$. That physically contains some information about the geometry, in terms of areas given by $a^N_{kl} \equiv \sqrt{j^N_{kl}\left(j^N_{kl}+1\right)}$, and their probabilities to appears with $\left[K^N_k, j^N_{kl}\right]^2$.

5.3.2 Spin-foam for the assembly

We construct the spin-foam amplitude associated to our assembly and its boundary state. In our case the spin-foam is given by three vertices, with five edges and ten faces each, and interconnected by a loop of three shared edges (see Figure 30). The blue edges correspond to the boundary tetrahedra and the red edges are the shared-tetrahedra. Each face from spin-foam is the dual of the triangle from assembly : the faces given by two blue-edges (in same vertex) are the boundary j-triangles, the faces given by two blue-edges connected by one red-edge are the boundary j_0-triangles, and the red-loop is the dual of the internal triangle f.

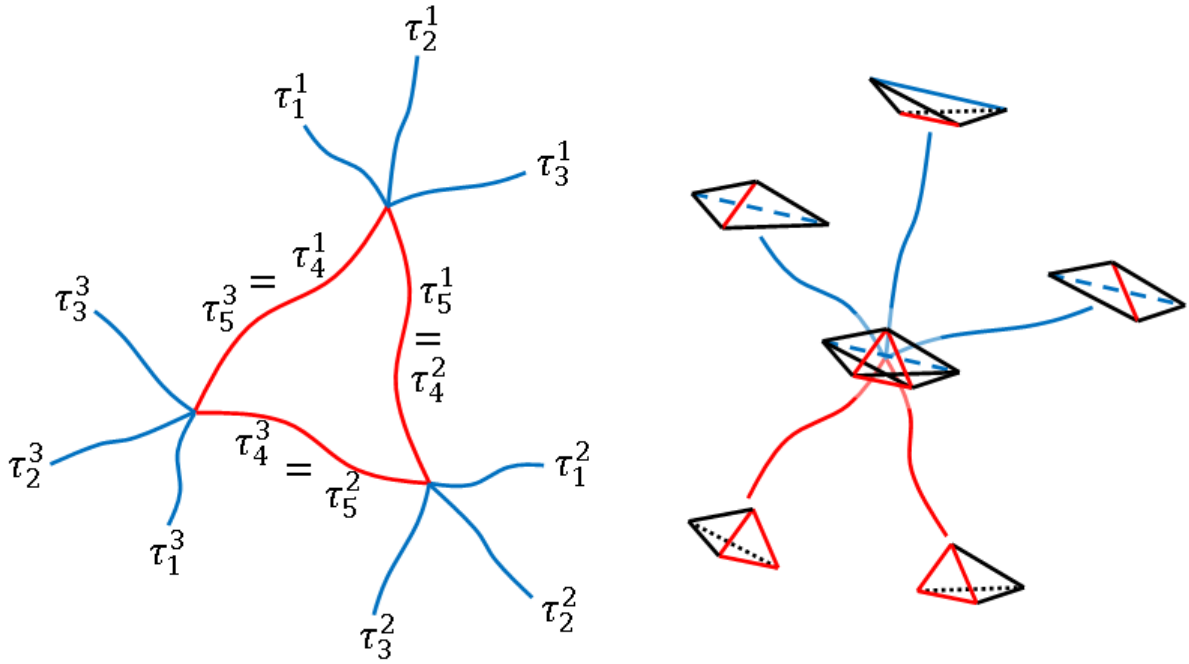


Figure 30: Spin-foam of the assembly. The red edge represents the shared-tetrahedra $\tau_4^N \left(= \tau_5^{N'}\right)$, and the blue edge represents the boundary tetrahedra. We see the associated pieces of geometry of the graph in the right.

Here the amplitude will be given as a function of $SU(2)$ group elements u^N_{kl} and $u^{NN'}_k$from the faces. The u^N_{kl} just come from of the boundary faces of the 4-simplices (who are also in the boundary of the full assembly for $k,l = 1,2,3$). The $u^{NN'}_k$ come from the boundary faces of the 4-simplices they share between them via the shared internals tetrahedra. For the amplitude, we have integrals over 15 copies of $SO(4) \simeq SU(2)^+ \times SU(2)^-$ (5 copies per each vertices) from the 4-simplices amplitude, and integrals of $SU(2)$-delta function for gluing the vertices. The

amplitude function W can be written :

$$\begin{aligned} &W\left(u_{kl}^{N},u_{k}^{NN'}\right)\\ &=\int_{SU(2)}du_{5l}du_{4l}\prod_{N}\left[\int_{(SO(4))^{5}}dU_{k}^{N}\prod_{kl}\delta\left(Y^{+}U_{k}^{N}\left(U_{l}^{N}\right)^{-1}Yu_{lk}^{N}\right)\right]\delta\left(u_{54}^{1}u_{54}^{2}u_{54}^{3}\right)\prod_{k,NN'}\delta\left(u_{5k}^{N}u_{k4}^{N'}u_{k}^{N'N}\right) \end{aligned} \tag{103}$$

With the previous definitions from the amplitude for individual 4-simplex, the covariant LQG amplitude of the full assembly is :

$$\begin{aligned} &W\left(u_{kl}^{N},u_{k}^{NN'}\right)\\ &=\int_{SU(2)}du_{5l}du_{4l}\prod_{N}\left[\sum_{j^{N}}\left(\prod_{kl}\left(2j_{kl}+1\right)\right)\sum_{K^{N}}\left[K_{k}^{N},j_{kl}^{N}\right]\overline{\psi^{j_{kl}^{N},K_{k}^{N}}\left(u_{kl}^{N}\right)}\right]\prod_{k,NN'}\delta\left(u_{5k}^{N}u_{k4}^{N'}u_{k}^{N'N}\right)\delta\left(u_{54}^{1}u_{54}^{2}u_{54}^{3}\right) \end{aligned} \tag{104}$$

And the gluing parts allow to rewrite the amplitude in terms of boundary states Ψ :

$$\begin{aligned} &W\left(u_{kl}^{N},u_{k}^{NN'}\right)\\ &=\sum_{j}\left(\prod_{N,kl}\left(2j_{kl}^{N}+1\right)\right)\sum_{j_{f}}\frac{\delta_{j_{f},j_{45}^{N}}}{\left(2j_{f}+1\right)^{2}}\sum_{K}\prod_{N}\left[K_{k}^{N},j_{kl}^{N}\right]\overline{\Psi^{j_{kl}^{N},K_{k}^{N}}\left(u_{kl}^{N},u_{k}^{NN'}\right)}\prod_{NN'}\delta_{K_{5}^{N}K_{4}^{N'}}\prod_{k,NN'}\frac{\delta_{j_{4k}^{N},j_{5k}^{N'}}}{2j_{4k}^{N}+1}\\ &=\sum_{j}\left(\prod_{N,kl\neq45}\left(2j_{kl}^{N}+1\right)\right)\sum_{K_{1}K_{2}K_{3}}\left(\sum_{j_{f}}\left(2j_{f}+1\right)\sum_{K_{4},K_{5}}\prod_{N}\left[K_{k}^{N},j_{kl}^{N}\right]_{j_{45}^{N}=j_{f}}\prod_{NN'}\delta_{K_{5}^{N}K_{4}^{N'}}\right)\\ &\qquad\times\overline{\Psi^{j_{kl},K_{k}^{N}}\left(u_{kl}^{N},u_{k}^{NN'}\right)} \end{aligned} \tag{105}$$

The amplitude of a spin-network state is just given by :

$$\begin{aligned} W^{j_{kl}^{N},J_{k}^{N}} &= \left\langle W|\Psi^{j_{kl}^{N},J_{k}^{N}}\right\rangle=\int_{SU(2)}du_{kl}W\left(u_{kl}^{N},u_{k}^{NN'}\right)\Psi^{j_{kl},J_{k}^{N}}\left(u_{kl}^{N},u_{k}^{NN'}\right)\\ &= \sum_{j_{f}}\left(2j_{f}+1\right)\left(\sum_{K}\prod_{N}\left[J_{k}^{N},K_{4}^{N},K_{5}^{N};j_{kl}^{N}\left[j_{45}^{N}=j_{f}\right]\right]\prod_{NN'}\delta_{K_{5}^{N}K_{4}^{N'}}\right) \end{aligned} \tag{106}$$

And finally give for our cylindrical symmetric boundary assembly :

$$\begin{aligned} W^{j_{kl}^{N},J_{k}^{N}} &= \left\langle W|\Psi^{j,j_{0},J_{k}^{N}}\right\rangle=\sum_{j_{f}}\left(2j_{f}+1\right)\left(\sum_{K_{4},K_{5}}\prod_{N}\left[J_{k}^{N},K_{4}^{N},K_{5}^{N};j,j_{0},j_{f}\right]\prod_{NN'}\delta_{K_{5}^{N}K_{4}^{N'}}\right)\\ &= \sum_{j_{f}}\left(2j_{f}+1\right)\left(\sum_{K_{4}}\left[J_{k}^{1},K_{4}^{1},K_{4}^{2};j,j_{0},j_{f}\right]\left[J_{k}^{2},K_{4}^{2},K_{4}^{3};j,j_{0},j_{f}\right]\left[J_{k}^{3},K_{4}^{3},K_{4}^{1};j,j_{0},j_{f}\right]\right) \end{aligned} \tag{107}$$

The mathematical structure of the amplitude of spin-network state is really interesting and have links with classical geometry :

- Each $SO(4)$ 15j-symbols $\left[K_{k}^{N},j_{kl}^{N}\right]$ represent 4-simplices and depend of the area parameters j_{kl}^{N} and the quantum equivalent of shape parameters –and projected area– K_{k}^{N}
- The Kronecker symbols $\delta_{K_{5}^{N}K_{4}^{N'}}$ glue the 4-simplices together and share the tetrahedra by summation over K_{4}^{N} and K_{5}^{N}
- The summation over j_{f} correspond to the quantum summation over the all possible area associated to f.

5.3.3 Coherent transition amplitude

With the Livine-Speziale coherent states for the quantum tetrahedra, we can construct coherent transition amplitude for the individual 4-simplex and the assembly. For the full assembly, we have the coherent transition amplitude :

$$W\left(j,j_{0},A_{k}^{N},\Phi_{k}^{N}\right)=\left\langle W|\bigotimes_{N,kl}j_{kl}^{N}\ \overrightarrow{n}_{kl}^{N}\left(A_{k}^{N},\Phi_{k}^{N}\right)\right\rangle=\sum_{J_{k}^{N}}\left\langle W|\Psi^{j,j_{0},J_{k}^{N}}\right\rangle\prod_{N,k}\left\langle i^{J_{k}^{N}}|j_{i}\overrightarrow{n}_{i}\left(A_{k}^{N},\Phi_{k}^{N}\right)\right\rangle \tag{108}$$

That will be reduce with the cylindrical symmetries to :

$$W\left(j,j_{0},A,\Phi\right)=\sum_{j_{f}}\left(2j_{f}+1\right)w_{f}\left(j,j_{0},j_{f},A,\Phi\right) \tag{109}$$

$$\left\{ \begin{array}{l} A_k^N \equiv A \\ \Phi_k^N \equiv \Phi \end{array} \right. \quad k=1,2,3 \text{ and } \forall N \tag{110}$$

where w_f are the amplitude associated to the spin-network with a specific area j_f-representation for the internal face f :

$$w_f\left(j,j_0,j_f,A,\Phi\right) = \sum_{J_k^N} \left(\sum_{K_4} \left[J_k^1,K_4^1,K_4^2;j,j_0,j_f\right]\left[J_k^2,K_4^2,K_4^3;j,j_0,j_f\right]\left[J_k^3,K_4^3,K_4^1;j,j_0,j_f\right] \right) \prod_{N,k} \left\langle i^{J_k^N} | j,j_0,A,\Phi \right\rangle \tag{111}$$

It physically represent, in a certain sens, the three 4-simplices "interacting" together for a specified face f with j_f given.

The summation over the K_4, which glue the 4-simplex together, can be expressed in terms of coherent states from the shared-tetrahedra. In fact, the Kronecker symbols $\delta_{K_4^N K_5^{N'}}$ from (106) can be expressed as below :

$$\delta_{K_4^N K_5^{N'}} = \left((2j_f+1)(2j_0+1)^3 \prod_{i=1}^4 \int \frac{d\overrightarrow{n_{f,i}^N}}{4\pi} \right) \left\langle i^{K_4^N} | j_f,j_0,\overrightarrow{n_{f,i}^N}\left[A_f^N,\Phi_f^N\right] \right\rangle \overline{\left\langle i^{K_5^{N'}} | j_f,j_0,\overrightarrow{n_{f,i}^N}\left[A_f^N,\Phi_f^N\right] \right\rangle} \tag{112}$$

where the $\left\langle i^{K_{4,5}^N} | j_f,j_0,\overrightarrow{n_{f,i}^N}\left[A_f^N,\Phi_f^N\right] \right\rangle$ are the Livine-Speziale coherent state associated to the shared-tetrahedra (Subsection 5.2.3) :

$$\begin{array}{rcl} \left\langle i^{K_{4,5}^N} | j_f,j_0,\overrightarrow{n_{f,i}^N}\left[A_f^N,\Phi_f^N\right] \right\rangle & = & \sum_m i^{K_{4,5}^N}_{m_1m_2m_3m_4} \prod_i D^{j_i^N}_{m_i j_i^N}\left(R(\overrightarrow{n_{f,i}^N})\right)\left[A_f^N,\Phi_f^N\right] \\ & = & \sum_m i^{K_{4,5}^N}_{m_1m_2m_3m_4} \prod_i D^{j_i^N}_{m_i j_i^N}\left(\theta_{f,i}^N,\phi_{f,i}^N\right)\left[A_f^N,\Phi_f^N\right] \end{array} \quad ; \begin{cases} j_1^N = j_f \\ j_i^N = j_f \quad \text{for } i=2,3,4 \end{cases} \tag{113}$$

These states depend of the face-vectors $\overrightarrow{n_{f,i}^N}$ associated to the faces of the corresponding shared-tetrahedra and implicitly of the feasible shape parameters A_f^N, Φ_f^N. *But*, only when the normal face vector$\overrightarrow{n_{f,i}^N}$ respect the closure condition (2) these states are reduced to the coherent states from the Subsection 5.2.3 :

$$\left\langle i^{K_k^N} | j_i^N \overrightarrow{n_i^N}\left[A_f^N,\Phi_f^N\right] \right\rangle \Big|_{\sum_i j_i^N \overrightarrow{n_i^N}=\overrightarrow{0}} \equiv \left\langle i^{K_k^N} | j_f,j_0,A_f,\Phi_f \right\rangle \qquad \left(\begin{array}{c} \text{with } \left(A_f^N,\Phi_f^N\right) \equiv (A_f,\Phi_f) \\ \text{from the cylindrical symmetries} \end{array} \right) \tag{114}$$

Note also the interesting property from (96) :

$$\overline{\left\langle i^{K_5^{N'}} | j_f,j_0,\overrightarrow{n_{f,i}^N}\left[A_f^N,\Phi_f^N\right] \right\rangle} \propto \left\langle i^{K_5^{N'}} | j_f,j_0,-\overrightarrow{n_{f,i}^N}\left[A_f^N,\Phi_f^N\right] \right\rangle \tag{115}$$

With these relations, we can express the w_f amplitude in function of the integration over the all geometries of the shared-tetrahedra :

$$\begin{array}{rl} w_f\left(j,j_0,j_f,A,\Phi\right) & = \prod_N \left((2j_f+1)(2j_0+1)^3 \prod_{i=1}^4 \int \frac{d\overrightarrow{n_{f,i}^N}}{4\pi} \right) \left\langle \pentagon_1 | j,j_0,j_f,A,\Phi,\overrightarrow{n_{f,i}^1}\left[A_f^1,\Phi_f^1\right],-\overrightarrow{n_{f,i}^2}\left[A_f^2,\Phi_f^2\right] \right\rangle \\ & \times \left\langle \pentagon_2 | j,j_0,j_f,A,\Phi,\overrightarrow{n_{f,i}^2}\left[A_f^2,\Phi_f^2\right],-\overrightarrow{n_{f,i}^3}\left[A_f^3,\Phi_f^3\right] \right\rangle \\ & \times \left\langle \pentagon_3 | j,j_0,j_f,A,\Phi,\overrightarrow{n_{f,i}^3}\left[A_f^3,\Phi_f^3\right],-\overrightarrow{n_{f,i}^1}\left[A_f^1,\Phi_f^1\right] \right\rangle \end{array} \tag{116}$$

where we have the coherent amplitude transition for each 4-simplex of the assembly :

$$\begin{array}{l} \left\langle \pentagon_N | j,j_0,j_f,A,\Phi,\overrightarrow{n_{f,i}^N}\left[A_f^N,\Phi_f^N\right],-\overrightarrow{n_{f,i}^{N'}}\left[A_f^{N'},\Phi_f^{N'}\right] \right\rangle \\ = \sum_{J^N,K^N} \left[J_k^N,K_4^N,K_5^N;j,j_0,j_f\right] \left\langle i^{K_4^N} | j_f,j_0,\overrightarrow{n_{f,i}^N}\left[A_f^N,\Phi_f^N\right] \right\rangle \overline{\left\langle i^{K_5^N} | j_f,j_0,\overrightarrow{n_{f,i}^{N'}}\left[A_f^{N'},\Phi_f^{N'}\right] \right\rangle} \prod_{k=1}^3 \left\langle i^{J_k^N} | j,j_0,A,\Phi \right\rangle \end{array} \tag{117}$$

For summarize, the coherent amplitude transition describes the quantum geometry of our study object with the boundary tetrahedra whose the geometries are given by the coherent states $\left\langle i^{J_k^N} | j, j_0, A, \Phi \right\rangle$. We can rewrite the coherent transition amplitude in terms of product of coherent 4-simplex amplitude where we sum (quantumly) over the all possible geometries of shared-tetrahedra :

$$\underbrace{W\left(j, j_0, A, \Phi\right)}_{\substack{\text{coherent}\\ \text{transition}\\ \text{amplitude}}} = \underbrace{\sum_{J_k^N}}_{\substack{\text{sum over}\\ \text{the}\\ \text{coherent}\\ \text{states}}} \underbrace{\left\langle W | \Psi^{j, j_0, J_k^N} \right\rangle}_{\substack{\text{transition amplitude}\\ \left(\begin{array}{c}\text{contains the all [quantum,}\\ \text{classical] possible geometries}\end{array}\right)}} \underbrace{\prod_{N,k} \left\langle i^{J_k^N} | j, j_0, A, \Phi \right\rangle}_{\substack{\text{coherent states for the}\\ \text{boundary tetrahedra}}}$$

$$= \underbrace{\sum_{j_f}}_{\substack{\text{sum over}\\ \text{the all}\\ \text{possible}\\ \text{area of } f}} \underbrace{\left(2j_f + 1\right) w_f\left(j, j_0, j_f, A, \Phi\right)}_{\substack{\text{transition amplitude of the}\\ \text{"interacting" 4-simplices}\\ \text{with the } j_f\text{-area for } f}}$$

$$= \underbrace{\sum_{j_f}}_{\substack{\text{sum over}\\ \text{the all}\\ \text{possible}\\ \text{area of } f}} \underbrace{\left(2j_f + 1\right) \prod_N \underbrace{\left(\left(2j_f + 1\right)\left(2j_0 + 1\right)^3 \prod_i \int \frac{d\overrightarrow{n_{f,i}^N}}{4\pi} \right)}_{\substack{\text{sum over the geometries of}\\ \text{one shared-tetrahedron}}} \prod_N \underbrace{\left\langle \pentagon_N | j, j_0, j_f, A, \Phi, \overrightarrow{n_{f,i}^N}, -\overrightarrow{n_{f,i}^{N'}} \right\rangle}_{\substack{\text{coherent transition amplitude of}\\ \text{one individual 4-simplex}}}}_{\substack{\text{sum over the geometries of}\\ \text{all shared-tetrahedra}}} \underbrace{\qquad\qquad}_{\substack{\text{transition amplitude of the three}\\ \text{(disjoint) 4-simplices}}} \tag{118}$$

The amplitudes contains the information about the geometry boundary and have the quantum summation over the all possible geometries. In this sens, we see the equivalence with a path integral formulation for the geometry :

$$W \sim \int_{\substack{\text{bulk}\\ \text{geometries}}} \underbrace{w}_{\substack{\text{amplitude of}\\ \text{geometries}}} \sim \int \mathcal{D} g_{\mu\nu} e^{-i S_{GR}[g_{\mu\nu}]} \tag{119}$$

The next will be to study the properties of W, w_f and of the other amplitudes, for find specific values and their corresponding geometries, especially classical solutions, and see how it contributes.

6 Numerical analysis of amplitude

Now we have the transition amplitudes and mathematical objects that describe the quantum geometry of our assembly, we will just compute them with a designed C++ code of our conception (Annexes B) and study the results. Of course, the computation of the all transition amplitudes and quantum objects are not enough for understand the quantum geometry. So we will also compute and give many interpretations and exploitations from the results for find the internal geometry of our object. The idea is, as in the classical interpretation where the choice of boundary parameters (a, a_0, A) fix the all geometry by assembly and classical constraints ((18),(14) and (19)), the transition amplitude computed for (j, j_0, A, Φ) will reproduce the full quantum geometry and its study will give some information about. First, we will compute and study the individual 4-simplex amplitude and show

that we find the classical geometry properties for the shared-tetrahedra when the coherent states of boundary are given. Next, we will compute the value of transition amplitude W in function of coherent states given by the shape variables (A, Φ) for fixed value of j and j_0 and study its properties and integrants (w_f, product of coherent 4-simplices...) for restore the full geometry information. Even if the developed code can be used for arbitrary values of j and j_0 (≤ 10), we do the all computations for the case $j = j_0 = 8$.

6.1 Transition amplitude for individual 4-simplex

Barrett *et al.*'s theorem [6] (see also Conrady and Freidel [7]) states that the vertex amplitude for a coherent boundary state is exponentially suppressed in the large spin limit ($j_{kl} \gg 1$) unless the shapes of the boundary tetrahedra are those determined non-locally by the classical flat geometry of 4-simplex, in terms of the areas of the faces, namely by the j_{kl} themselves. In the cases we are considering, this means that the shape variables A_k, Φ_k must take the "classical values", functions of j, j_0, j_f for the amplitude not to be suppressed.

We have studied these classical values in Section 2.2. For the angles, they are $\Phi_k = \frac{\pi}{2}$ for all k. For the A_k variables, they are given by the functions $A_k(j, j_0, j_f)$ defined by the constraint (15) for $k = 1, 2, 3$ and by the constraint (14) for $k = 4, 5$ (in the sense of the areas $(a, a_0, a_f) \sim (j, j_0, j_f)$ in the spin-network state, see Section 5.2.1). Thus, fixing *large* values of the spins j, j_0, j_f, we expect the 4-simplex amplitude $\langle \pentagon | j, j_0, j_f, A_k, \Phi_k \rangle$, seen as a function of the A_k and the Φ_k, to be peaked on the classical values $\Phi_k = \frac{\pi}{2}$ and $A_k = A_k(j, j_0, j_f)$. We are interested to explore what happens for *small* spins.

To this aim, we have designed a C++ program that computes the amplitude $\langle \pentagon | j, j_0, j_f, A_k, \Phi_k \rangle$ (derived from the (117)). Ideally, we would like to fix the spins and study the peakedness properties of the real function of ten variables $f_{j_0,j,j_f}(A_k, \Phi_k) = |\langle \pentagon | j, j_0, j_f, A_k, \Phi_k \rangle|$. However, the ten dimensional space A_k, Φ_k is too large to explore numerically. So, we study it gradually by exploring some of its sections.

6.1.1 Sections of the space of shapes

To start with, we fix all the angles and all the boundary projected areas to their classical values given respectively by $\Phi_k = \frac{\pi}{2}$ and by equation (14). This defines a function of two variables, the projected areas of the two shared-tetrahedra :

$$f(A_4, A_5) = f_{j_0,j,j_f}\left(A(j_0, j, j_f), A(j_0, j, j_f), A(j_0, j, j_f), A_4, A_5, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}\right) \tag{120}$$

A typical result from the numerical calculation is given in the left panel of Figure 31, where this function is plotted for $j = j_0 = j_f = 8$.

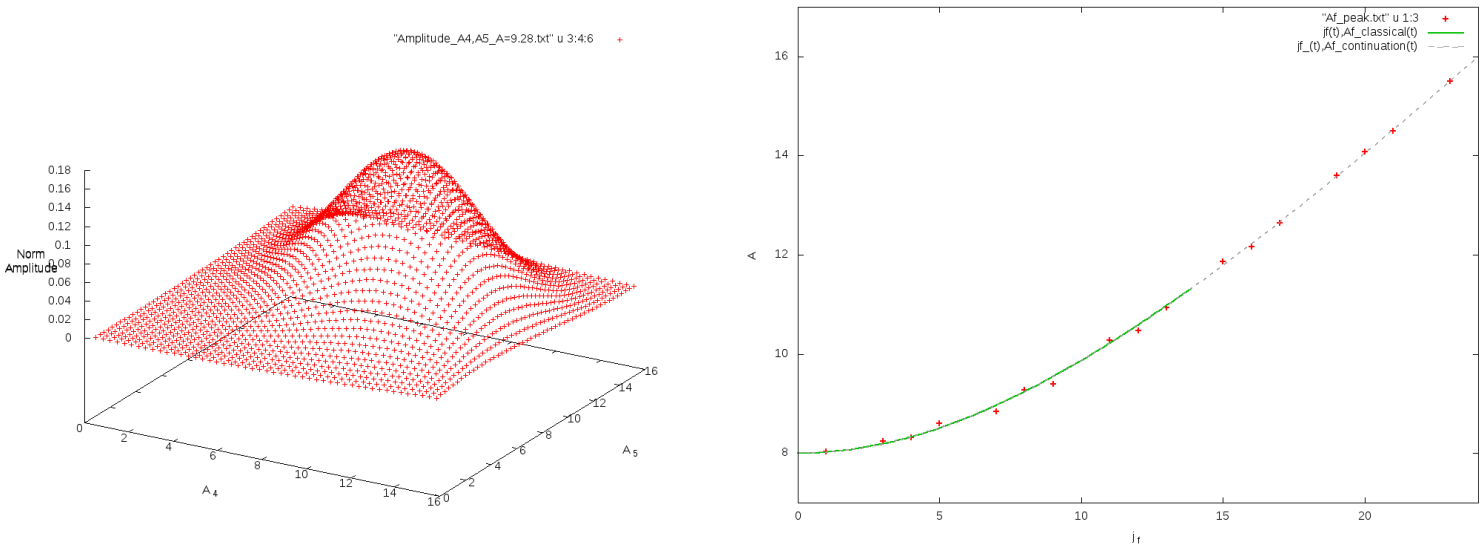


Figure 31: Left: $f(A_4, A_5)$ for $j = j_0 = j_f = 8$. Right: The position of the peak as j_f varies (crosses), compared with the classical value (line) and the analytic continuation of the classical value (dotted line).

The amplitude clearly peaks on a value of $A_4 = A_5 = A_f$, which is easily recognized precisely on the classical value $A_f = A_f(j_0, j, j_f)$. We can track the position of this peak as we change j_f and compare it with the classical value of A_f (or its analytic continuation when the triangular conditions are not respected). The result of this numerical analysis is given in the right panel of Figure 31, which shows that the peaks of the amplitude computed numerically (crosses) follow the classical value. This shows that, quite remarkably, the peakedness properties on the classical values already appears at small spins $j \sim 10$. This pattern is quite general.

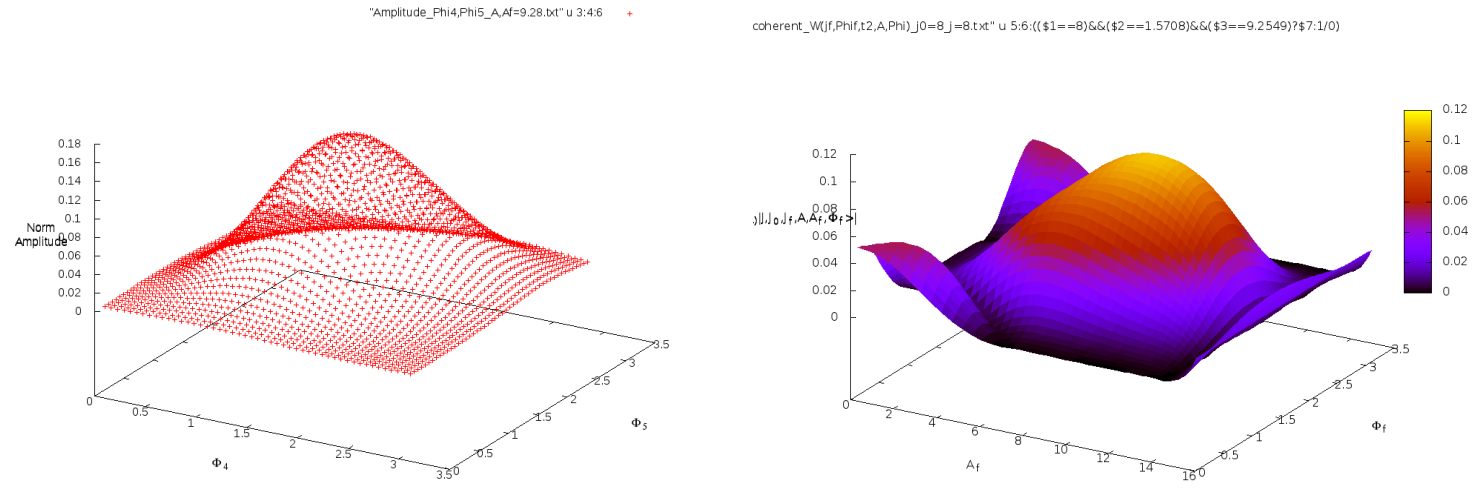


Figure 32: $f(\Phi_4, \Phi_5)$ (Left) and $f(A_f, \Phi_f)$ (Right), for $j = j_0 = j_f = 8$.

Next, we can reverse the role of the A's and the Φ's. That is, we fix all the A's to their classical value and we

compute the amplitude as a function of Φ_4 and Φ_5. That is :

$$f(\Phi_4, \Phi_5) = f_{j_0,j,j_f}\left(A(j_0,j,j_f), A(j_0,j,j_f), A(j_0,j,j_f), A_f(j_0,j_f), A_f(j_0,j_f), \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \Phi_4, \Phi_5\right) \tag{121}$$

The numerical result is given in the left panel of Figure 32. We also give the transverse section defined by :

$$f(A_f, \Phi_f) = f_{j_0,j,j_f}\left(A(j_0,j,j_f), A(j_0,j,j_f), A(j_0,j,j_f), A_f, A_f, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \Phi_f, \Phi_f\right) \tag{122}$$

The corresponding numerical result is given in the right panel of Figure 32. Again we see the peak of the amplitude on the classical values.

The last of these figures shows also that there seem to be an increase of the amplitude away from the classical values for low angles and low projected areas. To study this effect it is convenient to move away from the classical region. It is instructive to see what happens if we take a non-classical value of the projected area $A = A_1 = A_2 = A_2$ of the boundary tetrahedra. The numerical amplitude is given in Figure 33 with different values of A (the classical one is the fourth). That is, Figure 33 plots :

$$f(A_f, \Phi_f|A) = f_{j_0,j,j_f}\left(A, A, A, A_f, A_f, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \Phi_f, \Phi_f\right) \tag{123}$$

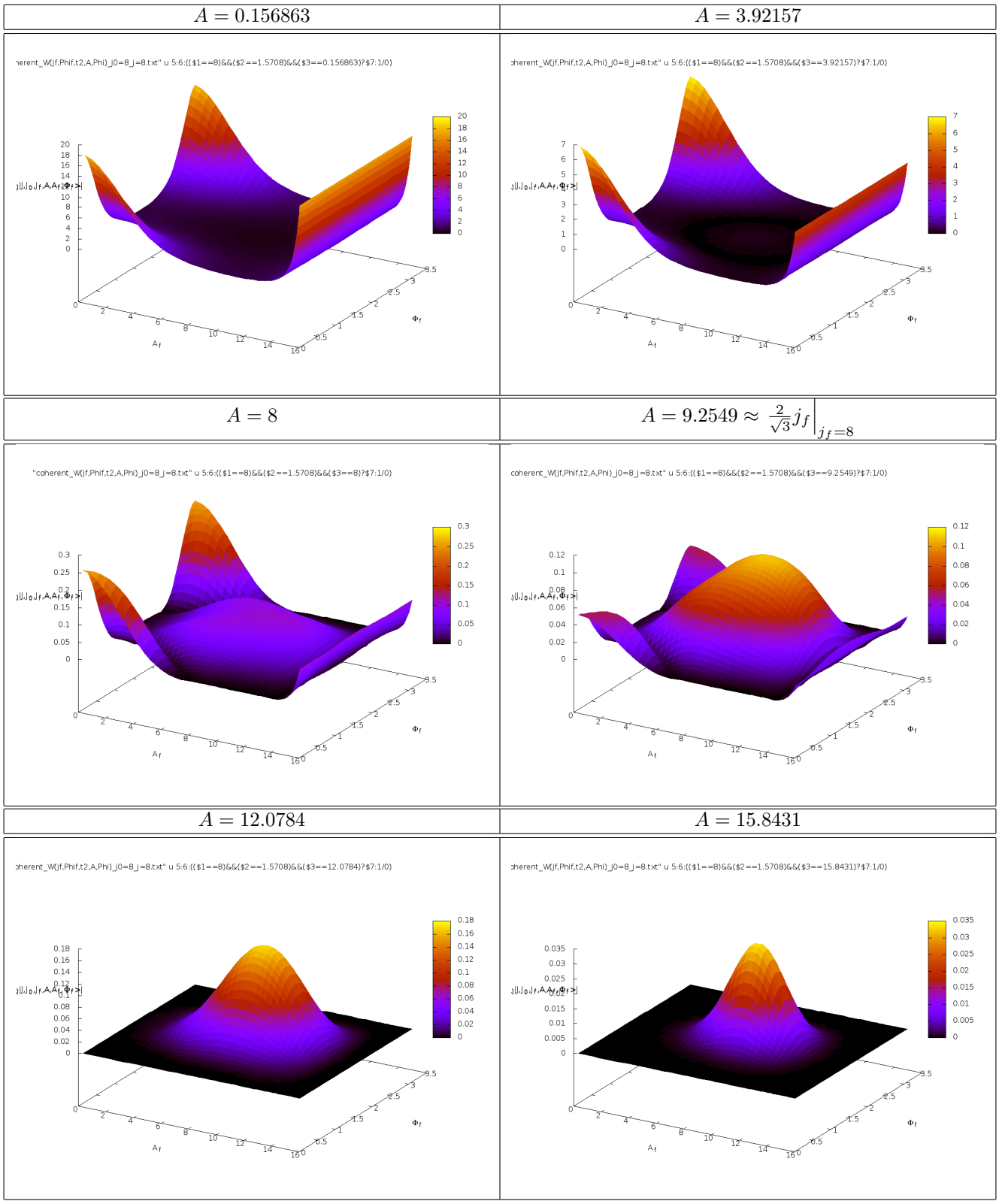


Figure 33: $f(A_f, \Phi_f|A)$, always for $j = j_0 = j_f = 8$.

Here we see an interesting phenomenon: there is large peaks for small areas and angles, which is not accounted for by the classical limit geometry. It is clearly an effect of degenerate geometries, as evident from the fact that it is at the angles $\Phi_f \sim \frac{\pi}{2} \pm \frac{\pi}{2}$ and/or at the shape parameter $A \sim \{0, 2\min(j, j_0)\}$. As A increases and get closer to its classical value, the peak at the classical values $\Phi_f \sim \frac{\pi}{2}$ and $A_f \sim A_f(j_0, j_f)$ emerges. If the value of A increase more, the peak become more "clean" around the classical values $(A_f, \Phi_f) = \left(A_f(j_0, j_f), \frac{\pi}{2}\right)$ but the value of the height of the summit decrease : we assumed that it's a trace of the presence of the classical peak for all possible value of A, but it become maximum when A get closer to its classical value. For small value of A, the classical peak is probably always present, but is completely drown in the degenerate geometries.

6.1.2 Phases and actions of individual 4-simplices

Besides the properties of the norms, which appears to be peaked around the classical geometry which fit the values given by the j-areas representations, it will be important to find inside the definition of the mathematical object the trace of the 4d-geometry parameters. So we will look the evolution of one 4-simplex amplitude with coherent states and see if we can find the 4-dimentional properties of the geometry, and more particularly if we find the definition of Regge's action. For a set of j-areas parameters, which fix the areas, we have only one set of coherent state which perfectly fit the corresponding classical geometry. As the classical geometry of the 4-simplex, the choice of the j-areas give non-locally via the 4d-geometry constraints the specific values of shape variables $A\left(j, j_0, j_f\right), \Phi = \frac{\pi}{2}$ and $A_f\left(j_0, j_f\right), \Phi_f = \frac{\pi}{2}$ given earlier (see Subsection 2.2). The idea will be to study the values of 4-simplex amplitude with the coherent states given by the shape variables "on-shell of the 4d-geometry" $A\left(j, j_0, j_f\right)$, $A_f\left(j_0, j_f\right)$ and $\Phi = \Phi_f = \frac{\pi}{2}$ in function of the j-areas parameters. We will fix j, j_0 and see the evolution of the transition amplitude in function of j_f. The expression of the 4-simplex amplitude with the coherent states previously defined is just :

$$\begin{aligned}\left\langle \pentagon_N | j, j_0, j_f, A\left(j, j_0, j_f\right), \tfrac{\pi}{2}, \overrightarrow{n_i^N}\left[A_f\left(j_0, j_f\right), \tfrac{\pi}{2}\right], -\overrightarrow{n_i^{N'}}\left[A_f\left(j_0, j_f\right), \tfrac{\pi}{2}\right] \right\rangle \\ = \sum_{J^N, K^N}\left[J_k^N, K_4^N, K_5^N; j, j_0, j_f\right] \left\langle i^{K_4^N} | j_f, j_0, A_f\left(j_0, j_f\right), \tfrac{\pi}{2}\right\rangle \overline{\left\langle i^{K_5^N} | j_f, j_0, A_f\left(j_0, j_f\right), \tfrac{\pi}{2}\right\rangle} \\ \times \prod_{k=1}^{3} \left\langle i^{J_k^N} | j, j_0, A\left(j, j_0, j_f\right), \tfrac{\pi}{2}\right\rangle\end{aligned} \tag{124}$$

which are always real, and the drawing of this amplitude for $j = j_0 = 8$ in function of j_f is :

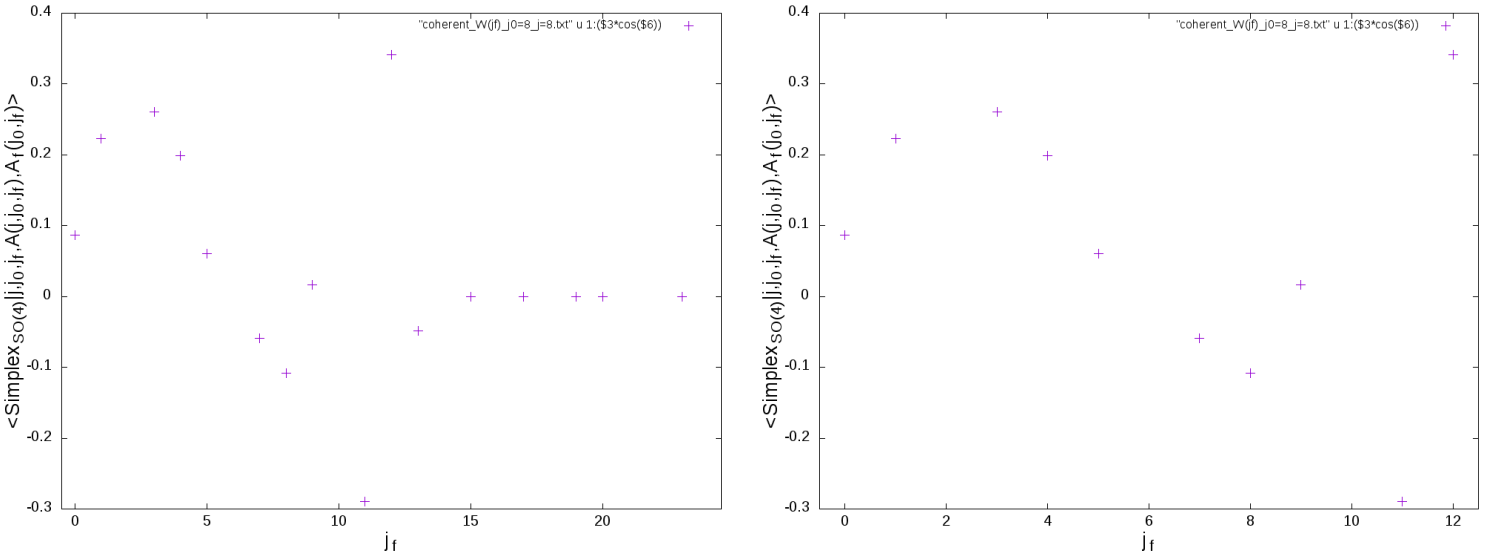


Figure 34: In the left, the 4-simplex amplitude with "on-shell 4d-geometry" coherent states for the each j_f quantumly possible : $j_f \in [0; 3j_0]$. In the right, the same but only for the j_f which have classical equivalent : $j_f \in \left[0; \frac{3}{2}j_0\right]$. The j, j_0 parameters are equal to 8.

The goal is to find a formula which approximate the results and contains inside its definition the Regge's action. As in the article [30], the integral I of the $SU(2)$ 4-simplex transition amplitude are connected to Regge's action by the stationary phase points approximation :

$$\begin{aligned} I\left(j_{kl}\right) &= \int_{(SU(2))^5} dh \prod_{kl} D^{j_{kl}}\left(h_k h_l^{-1}\right) = \sum_K \left|\left\langle \pentagon_{SU(2)} | j, j_0, j_f, K_i \right\rangle\right|^2 = \sum_K \left(K_k, j_{kl}\right)^2 \\ &\sim -\frac{(-1)^{2\sum_{kl} j_{kl}}}{2^4}\left(\sum_\sigma P\left(\sigma\right) \cos\left(\sum_{kl}\left(2j_{kl}+1\right)\Theta_{kl}\left(\sigma\right) + \chi\frac{\pi}{4}\right)\right) + D \end{aligned} \tag{125}$$

We find the connection between the 15j-symbol (K_k, j_{kl}) for the $SU(2)$ 4-simplex and the twice of Regge's action :

$$\sum_{kl}\left(2j_{kl}+1\right)\Theta_{kl} = 2S_{Regge}\left[j_{kl}, \Theta_{kl}\right] \tag{126}$$

$$\text{for } S_{Regge}\left[j_{kl}, \Theta_{kl}\right] = \sum_{kl}\left(j_{kl}+\frac{1}{2}\right)\Theta_{kl} \;\sim\; S_{Regge}\left[a_{kl}, \Theta_{kl}\right] = \sum_{kl} a_{kl}\Theta_{kl} \tag{127}$$

We have the result for coherent transition amplitude $\left\langle \pentagon_{SO(4)} | j_{kl}, A_k, \Phi_k \right\rangle$ that is a combination of 15j-symbol from the j^+ and j^- :

$$\left\langle \pentagon_{SO(4)} | j_{kl}, A_k, \Phi_k \right\rangle = \sum_K \left[K_k, j_{kl}\right] \prod_k \left\langle i^{K_k} | j_{(k)l}, A_k, \Phi_k \right\rangle \tag{128}$$

$$\left[K_k, j_{kl}\right] = \sum_{K^\pm}\left(K_k^+, j_{kl}^+\right)\left(K_k^+, j_{kl}^-\right)\prod_k \mathcal{I}^{K_k}_{K_k^+ K_k^-}\left(j_{kl}\right) \tag{129}$$

And the 15j-symbols are, in a certain sens, connected to the square-root of the integral I :

$$\begin{aligned} &\sum_K \left(K_k^\pm, j_{kl}^\pm\right)^2 = \int \frac{\overrightarrow{\mathcal{D}n^\pm}}{4\pi}\left|\sum_{K^\pm}\left(K_k^\pm, j_{kl}^\pm\right)\left\langle i^{K_k^\pm} | j_{(k)l}^\pm, \overrightarrow{n_{(k)l}^\pm}\right\rangle\right|^2 \\ &= I \sim -\tfrac{1}{2^4}\left(\sum_{\sigma^\pm} P^\pm\left(\sigma^\pm\right)\cos\left(\sum_{kl}\left(2j_{kl}^\pm+1\right)\Theta_{kl}^\pm\left(\sigma^\pm\right) + \chi^\pm\tfrac{\pi}{4}\right)\right) + D^\pm \end{aligned} \tag{130}$$

The postulate is you can express the coherent 15j-symbols as a expression in terms of Regge's action for the corresponding coherent geometry :

$$\sum_{K^{\pm}} \left[K_k^{\pm}, j_{kl}^{\pm}\right] \prod_k \left\langle i^{K_k^{\pm}} | j_{(k)l}^{\pm}, \overrightarrow{n_{(k)l}^{\pm}} \right\rangle \sim \left\{ N^{\pm} \cos\left(S_{Regge}^{\pm}\left[j_{kl}^{\pm}, \Theta_{kl}^{\pm}\right] + \alpha^{\pm} \right) + D'^{\pm} \right\}_{j_{(k)l}^{\pm}, \overrightarrow{n_{(k)l}^{\pm}}} \tag{131}$$

Which give in first approximation for the coherent $SO(4)$ transition amplitude :

$$\begin{aligned} &\left\langle \pentagon_{SO(4)} | j_{kl}, A_k, \Phi_k \right\rangle \\ &\quad = \int \tfrac{\mathcal{D}\overrightarrow{n^+}}{4\pi} \int \tfrac{\mathcal{D}\overrightarrow{n^-}}{4\pi} \left(\textstyle\sum_{K^+} \left[K_k^+, j_{kl}^+\right] \prod_k \left\langle i^{K_k^+} | j_{(k)l}^+, \overrightarrow{n_{(k)l}^+} \right\rangle \right) \left(\textstyle\sum_{K^-} \left[K_k^-, j_{kl}^-\right] \prod_k \left\langle i^{K_k^-} | j_{(k)l}^-, \overrightarrow{n_{(k)l}^-} \right\rangle \right) \\ &\qquad \times \prod_k \left(\textstyle\sum_{K_k^+, K_k^-, K_k} \left\langle j_{(k)l}^+, \overrightarrow{n_{(k)l}^+} | i^{K_k^+} \right\rangle \left\langle j_{(k)l}^-, \overrightarrow{n_{(k)l}^-} | i^{K_k^-} \right\rangle \mathcal{I}_{K_k^+ K_k^-}^{K_k}\left(j_{kl}\right) \left\langle i^{K_k} | j_{(k)l}, A_k, \Phi_k \right\rangle \right) \\ &\quad \sim \left\{ N \cos\left(S_{Regge}^{+}\left[j_{kl}^{+}, \Theta_{kl}^{+}\right] + \alpha^{+} \right) \cos\left(S_{Regge}^{-}\left[j_{kl}^{-}, \Theta_{kl}^{-}\right] + \alpha^{-} \right) + D' \right\}_{j_{(k)l}, A_k, \Phi_k} \end{aligned} \tag{132}$$

So we will try to approximate the results of the coherent transition amplitude of 4-Simplex with the formula :

$$\begin{aligned} &\left\langle \pentagon_N | j, j_0, j_f, A\left(j, j_0, j_f\right), \tfrac{\pi}{2}, \overrightarrow{n_i^N}\left[A_f\left(j_0, j_f\right), \tfrac{\pi}{2}\right], -\overrightarrow{n_i^{N'}}\left[A_f\left(j_0, j_f\right), \tfrac{\pi}{2}\right] \right\rangle \\ &\quad = N \cos\left(S_{Regge}^{+}\left[j^+, j_0^+, j_f^+, \Theta_f^+, \Theta_0^+, \Theta_a^+\right] + \alpha^+ \right) \cos\left(S_{Regge}^{-}\left[j^-, j_0^-, j_f^-, \Theta_f^-, \Theta_0^-, \Theta_a^-\right] + \alpha^- \right) + C \end{aligned} \tag{133}$$

Where the actions $S_{Regge}^{\pm}$ and the angles $\Theta^{\pm}$ are given by classical 4-simplex geometry from j^+'s and j^-'s part with the same symmetries of the classical 4-simplex from the j's :

$$S_{Regge}^{\pm}\left[j^{\pm}, j_0^{\pm}, j_f^{\pm}, \Theta_f^{\pm}, \Theta_0^{\pm}, \Theta_a^{\pm}\right] = \left(j_f^{\pm} + \frac{1}{2}\right)\Theta_f^{\pm} + 6\left(j_0^{\pm} + \frac{1}{2}\right)\Theta_0^{\pm} + 3\left(j^{\pm} + \frac{1}{2}\right)\Theta_a^{\pm} \tag{134}$$

$$\begin{aligned} \cos\Theta_f^{\pm} &= 1 - \tfrac{6A^{\pm 2}\left(4a^{\pm 2} - A^{\pm 2}\right)}{12a_0^{\pm 2}(4a^{\pm 2} - A^{\pm 2}) - A^{\pm 2}\left(4a_0^{\pm 2} - A^{\pm 2}\right)} \\ \cos\Theta_0^{\pm} &= \tfrac{A^{\pm 2}}{\sqrt{12a_0^{\pm 2}(4a^{\pm 2} - A^{\pm 2}) - A^{\pm 2}\left(4a_0^{\pm 2} - A^{\pm 2}\right)}} \\ \cos\Theta_a^{\pm} &= \tfrac{1}{2}\left(1 - \tfrac{A^{\pm 2}}{4a^{\pm 2} - A^{\pm 2}}\right) \end{aligned} \tag{135}$$

$$A^{\pm} = \sqrt{2A_f^{\pm 2} - 2\sqrt{A_f^{\pm 4} - \frac{4}{3}a^{\pm 2}a_f^{\pm 2}}}\ ,\ A_f^{\pm} = \sqrt{a_0^{\pm 2} + \frac{1}{3}a_f^{\pm 2}} \tag{136}$$

$$a^{\pm} = j^{\pm}\ ,\ a_0^{\pm} = j_0^{\pm}\ ,\ a_f^{\pm} = j_f^{\pm} \tag{137}$$

The best fit between the results and the postulate formula is given by the following Figure :

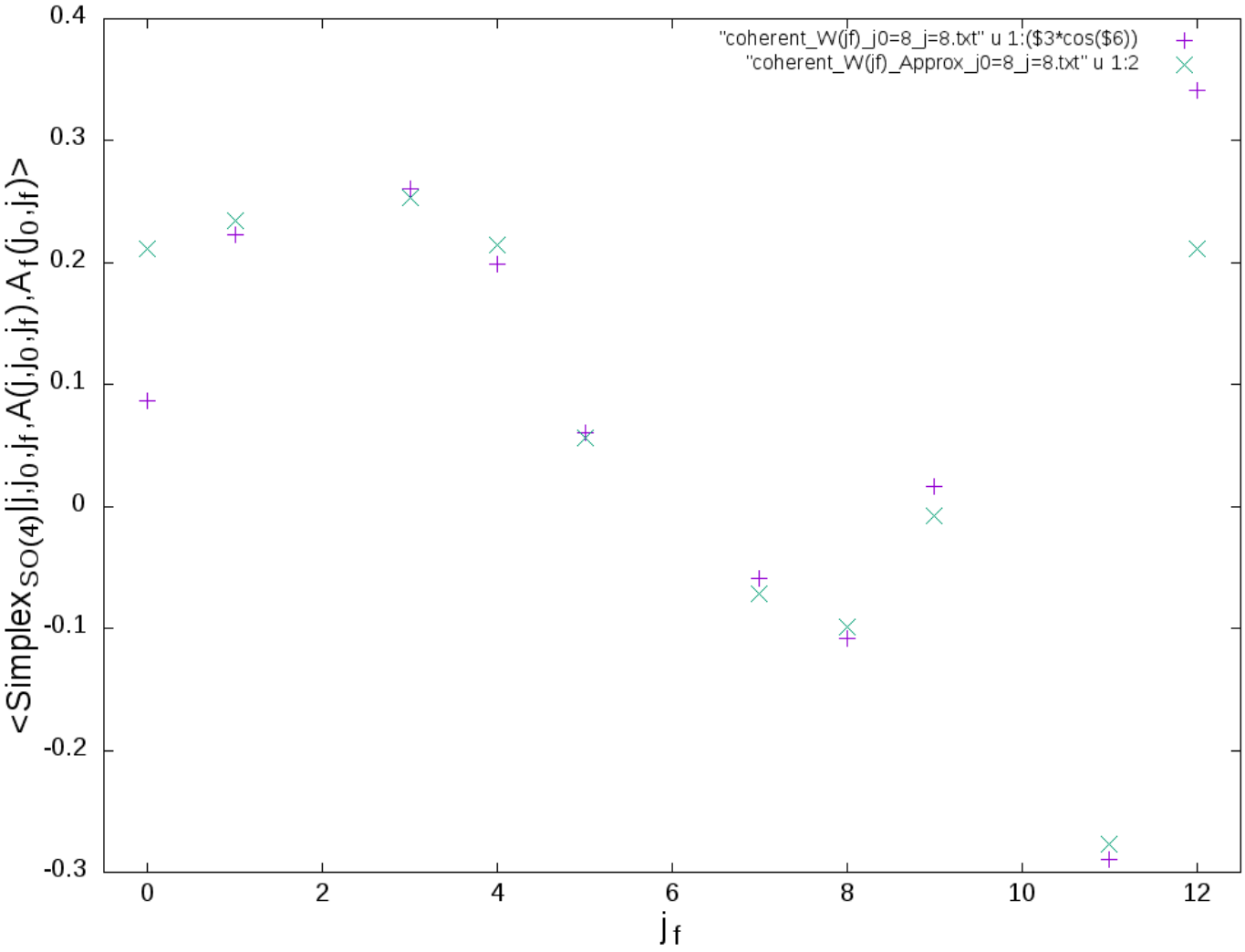


Figure 35: In purple, the points of the computed coherent 4-simplex amplitude for $j = j_0 = 8$ in function of j_f (for the region of j_f which give possible/physical classical geometries). In green, the points from the postulate formula (133) which fit the results. The postulate formula fit very well the results with the parameters $N = 0.285$, $\alpha^+ = 2.57611$, $\alpha^- = 2.89027$ and $C = -0.022$.

We have nice results, because the approximate function are really close to the real computation from coherent transition amplitude. The biggest differences between the fit and the computation are for the extremum values of j_f : $j_f = 0$ and $j_f = \frac{3}{2}j_0 = 12$ where the corresponding classical geometry are really degenerated with very flat and elongated tetrahedra. For these extremum values, the usual definitions of Regge's action, which are classical, are wrong and give bad approximation because the results are strongly quantum. But for the others values, the Regge's action give very good approximation for the quantum action from transition amplitude, and that start already the scale $j \sim 10$!

But for the transition amplitude of the assembly, we will lose the information about the value of j_f, it will be useful to check the efficiency of the approximate formula in the variable space A. Because the transition amplitude of the assembly is express in terms of boundary coherent states $|j, j_0, A, \Phi\rangle$, if we want restore the internal geometries with its action and curvature we need to study the corresponding approximate formula and corresponding action for these geometries states. With the equivalent reasoning of previously, the choice of j, j_0, A and Φ give "non-locally"

in classical geometry the value of f area and shape variables A_f, Φ_f : that fix the full geometry of individual 4-simplex. Of course in the quantum gravity, the f area is quantified with the j_f-area parameter unlike the classical case where the f area evolve continuously with the values of (j, j_0, A). So for express the evolution of the transition amplitude in function of A we will choose the quantum value of j_f closest to the classical f area given by classical geometry (18) :

$$a_f = \frac{\sqrt{3}}{2} A \sqrt{\frac{4a_0^2 - A^2}{4a^2 - A^2}} \;\rightarrow\; j_f(A) \approx \frac{\sqrt{3}}{2} A \sqrt{\frac{4j_0^2 - A^2}{4j^2 - A^2}} \tag{138}$$

The shape variables Φ and Φ_f will be taken to $\frac{\pi}{2}$ in agreement with cylindrical symmetries, and the variable A_f can be expressed in term of (j, j_0, A) via its definition with $j_f(j, j_0, A)$:

$$A_f(j, j_0, A) = \sqrt{j_0^2 + \frac{1}{3} j_f^2(j, j_0, A)} \tag{139}$$

With this approximation for the value of j_f, we can express the transition amplitude of 4-simplex in function of variables set (j, j_0, A) :

$$\begin{aligned} &\left\langle \pentagon_N | j, j_0, j_f, A, \tfrac{\pi}{2}, \overrightarrow{n_i^N}\left[A_f(j, j_0, A), \tfrac{\pi}{2}\right], -\overrightarrow{n_i^N}\left[A_f(j, j_0, A), \tfrac{\pi}{2}\right] \right\rangle \\ &= \sum_{J^N, K^N} \left[J_k^N, K_4^N, K_5^N; j, j_0, j_f(j, j_0, A)\right] \left\langle i^{K_4^N} | j_f(j, j_0, A), j_0, A_f(j, j_0, A), \tfrac{\pi}{2} \right\rangle \overline{\left\langle i^{K_5^N} | j_f(j, j_0, A), j_0, A_f(j, j_0, A), \tfrac{\pi}{2} \right\rangle} \\ &\qquad \times \prod_{k=1}^{3} \left\langle i^{J_k^N} | j, j_0, A, \tfrac{\pi}{2} \right\rangle \end{aligned} \tag{140}$$

and draw this evolution in function of A for $j = j_0 = 8$:

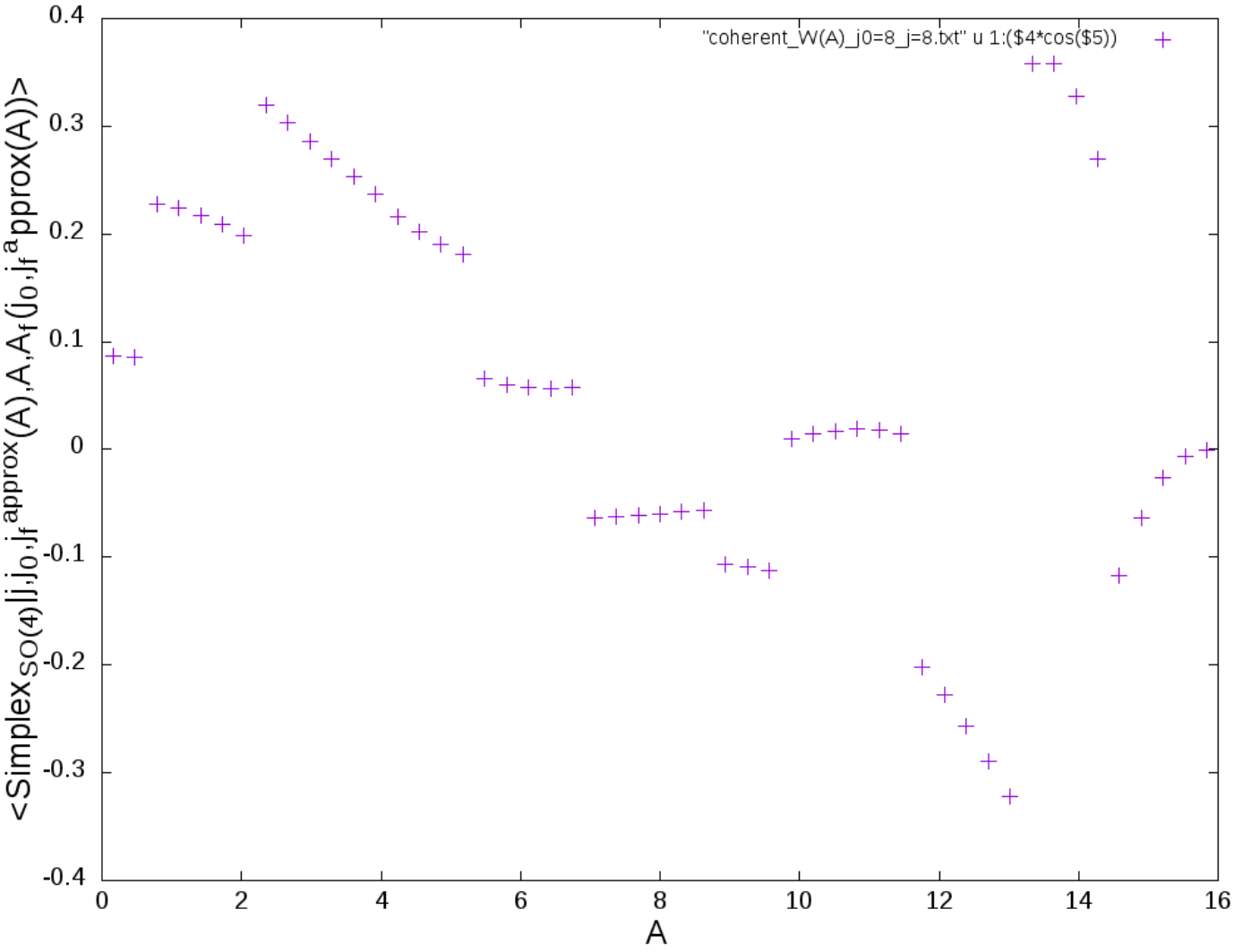


Figure 36: Amplitude transition of 4-simplex "on-shell 4d-geometry" in function of A for $j = j_0 = 8$ and the approximation $j_f(j, j_0, A)$

The many discontinuities of the Figure come from the approximation of j_f, because the non-null values of transition amplitude are given for quantum value of j_f and compatible values of j_f^+ and j_f^- from the map inside the definition of the $SO(4)$ transition amplitude. These discontinuities are non-physical and are purely the artifacts of the j_f approximation, for a fixed j_f the transition amplitude are perfectly continuous as the previous figures in the "space of shapes" Section 6.1.1. Now if we take our approximate formula define above, but with the definition of $A_f(j, j_0, A)$ and the quantum approximate value $j_f(j, j_0, A)$ as in the transition amplitude, we have the following Figure :

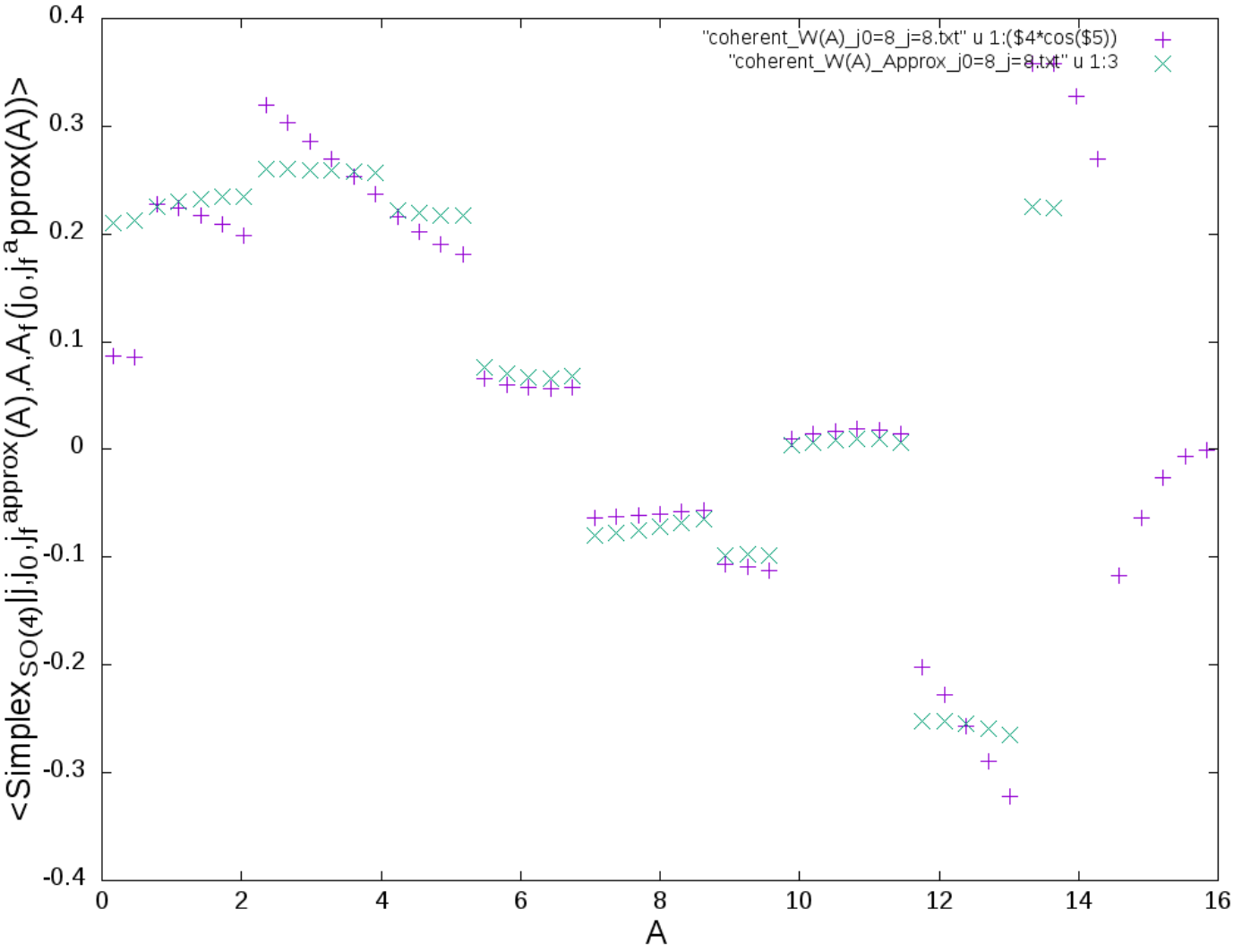


Figure 37: In purple, the points of the computed coherent transition amplitude for $j = j_0 = 8$ in function of A. In green, the points from the postulate formula with the Regge's action. The postulate formula (133) fit the results as well with the parameters $N = 0.285$, $\alpha^+ = 2.57611$, $\alpha^- = 2.89027$ and $C = -0.022$.

Again the approximate formula (133) fit very well the true results from the 4-simplex amplitude ! The region where the fit is the best are for $A \in [5.4902; 11.451]$ corresponding to a region where the corresponding geometry parameters are susceptible to give curvature for the next.

6.1.3 Short conclusion for the individual 4-simplex amplitude

The quantum geometry of a individual coherent 4-simplex is determined by the values of the areas of the triangles and the shape variables of the tetrahedra. Under cylindrical symmetry, the independent variables are $j, j_0, j_f, A_f, \Phi_f, A, \Phi$ and, in the limit of large areas, the modulus square of the amplitude is peaked on the expected classical values of A_f, Φ_f for given A, Φ. Here we have studied numerically the peakedness properties of the modulus square of the amplitude for small values of the spins, up to $j \sim 10$. We have found that the classical behaviour already emerges. In particular, the modulus of the amplitude appear to clearly peaked on the classical values of A_f, Φ_f (Figures 31,32,33). The peak in A is disturbed by the presence of high amplitude values around degenerate

configurations (Figures 33) for small-volume 4-simplices. Moreover, the study of the norm of the individual coherent 4-simplex amplitude in function of j_f show results (Figures 35,37) where we found the Regge's action as expected by Barrett [6, 30].

6.2 Transition amplitude for the assembly

Now that the properties of individual 4-simplex amplitude are studied, we will study the transition amplitude of the full assembly. We will briefly, look the norm and phase of the transition amplitude W, for fixed value of (j, j_0), in the space (A, Φ). In the all next, the value of j and j_0 will be fixed to 8.

6.2.1 Full transition amplitude

So, for $j = j_0 = 8$, we can compute the full coherent transition amplitude $W(j, j_0, A, \Phi)$ in the function of A and Φ and look the peakedness properties of its norm. The drawing of the norm is :

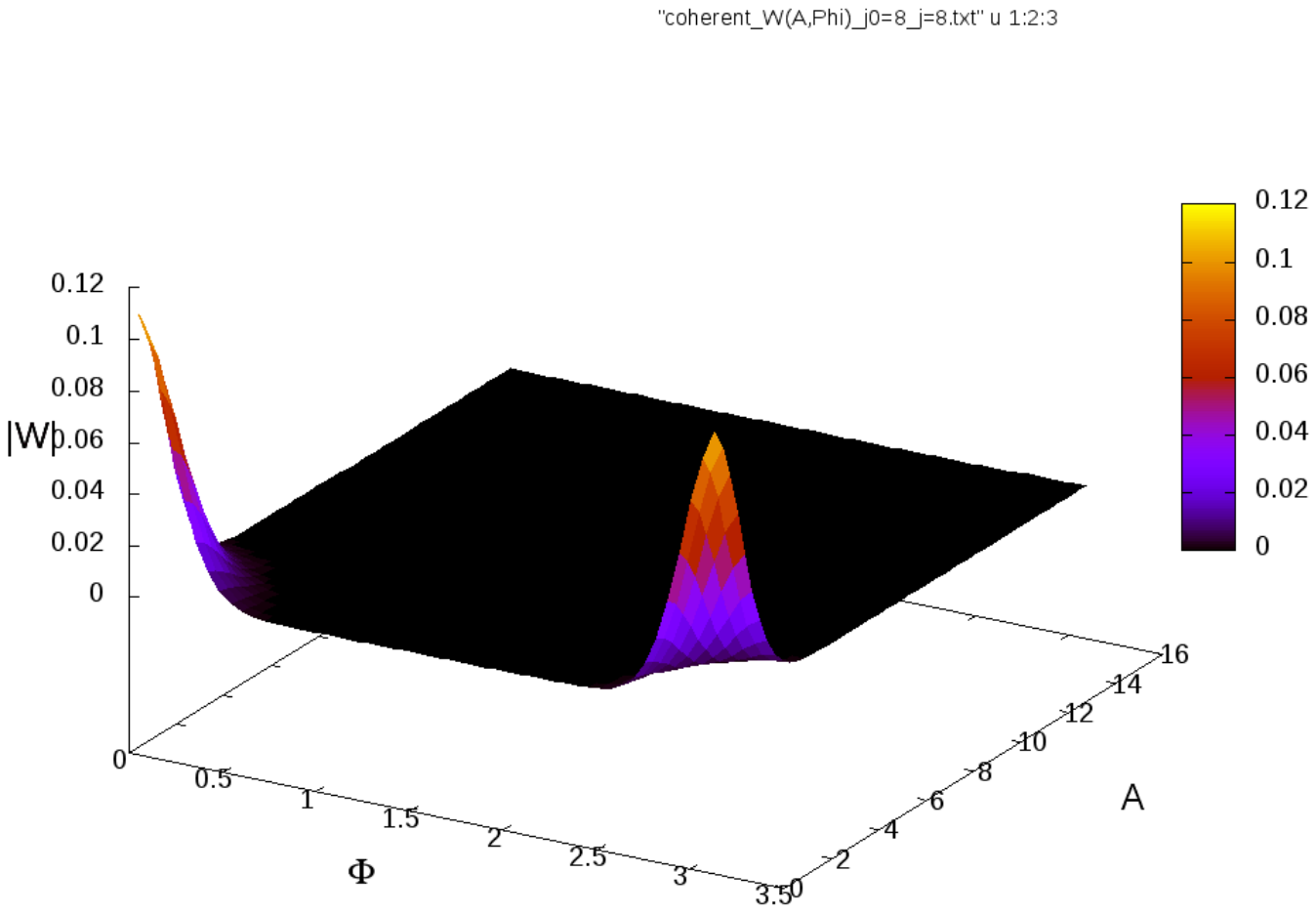


Figure 38: Drawing of the norm of $W(j, j_0, A, \Phi)$ in the space variables (A, Φ) for $j = j_0 = 8$

So, with just the properties of raw data, the transition amplitude seem to choose very degenerate geometry. We

have two high peak for $(A, \Phi) = \left(0, \frac{\pi}{2} \pm \frac{\pi}{2}\right)$ corresponding geometrically to completely flat boundary tetrahedra. That can be explain by the summation of all internal geometries, classical and quantum, and volume effect given by the norm of the states used. If we look the properties of transition amplitude in the cylindrical case, where $\Phi = \frac{\pi}{2}$, we have :

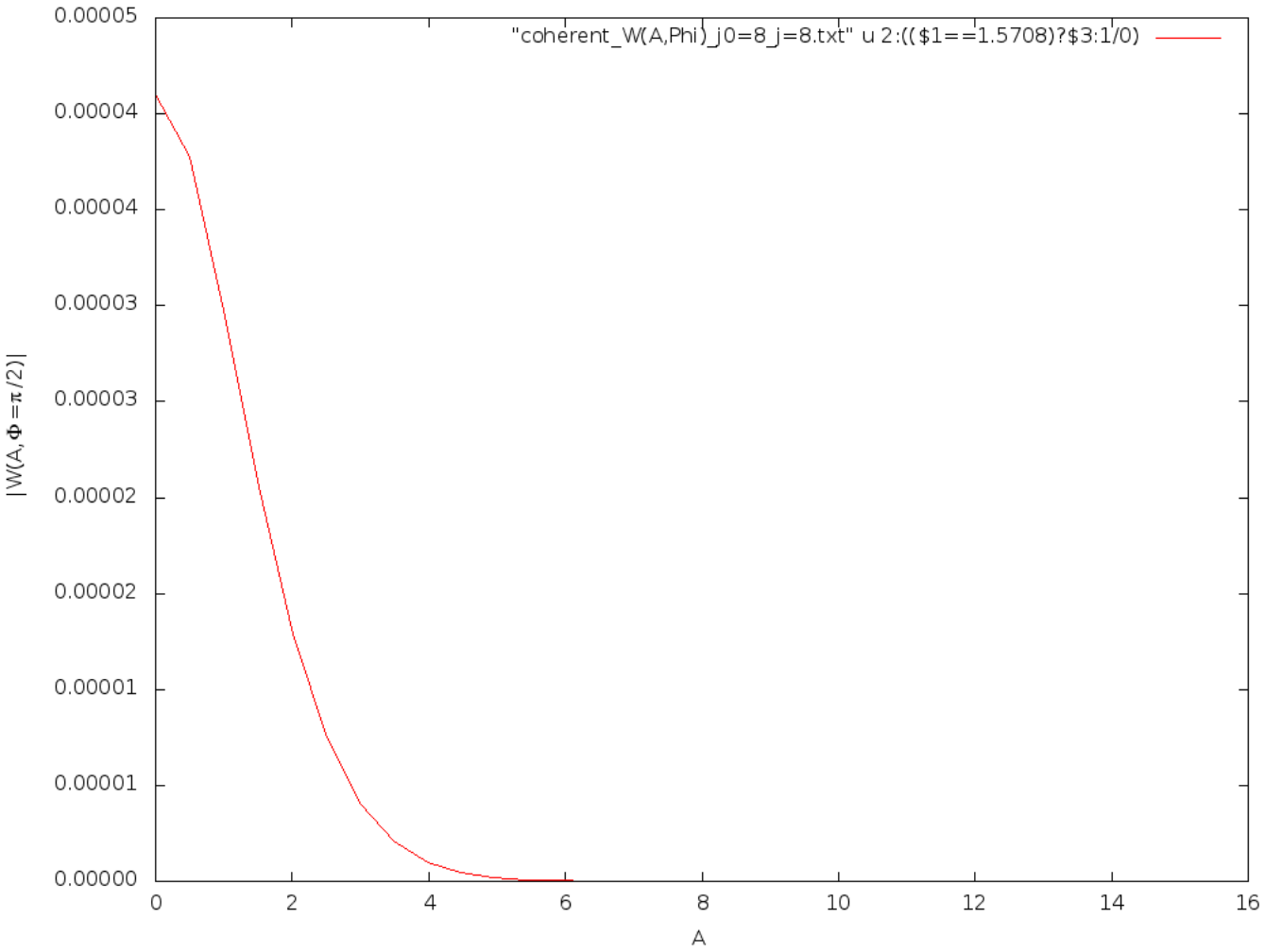


Figure 39: Drawing of the norm of $W\left(j, j_0, A, \frac{\pi}{2}\right)$ in function of A for $j = j_0 = 8$

Again we see that the transition amplitude is peaked on the degenerate geometry $A = 0$. So for j, j_0 fixed, the transition amplitude is not classical and priority choose completely flat boundary. But that make sens with the volume effect of the states, because the parameters j, j_0 and Φ are not enough for fix the full geometry. Indeed, in classical geometry the choice of a and a_0 ($\Phi = \frac{\pi}{2}$ is imposed by cylindrical symmetry) don't fix the full geometry, so we have no preferential value of A : the choice of A is arbitrary and cause all geometries from A are possible. In quantum geometries, because the volumes effects from the coherent states and intertwiners (see (92)) grow up for degenerated case, that maximize naturally the transition amplitude for degenerate geometries and make them dominant in the raw data of transition amplitude.

Now, if we draw the real and imaginary part of the phase from transition amplitude :

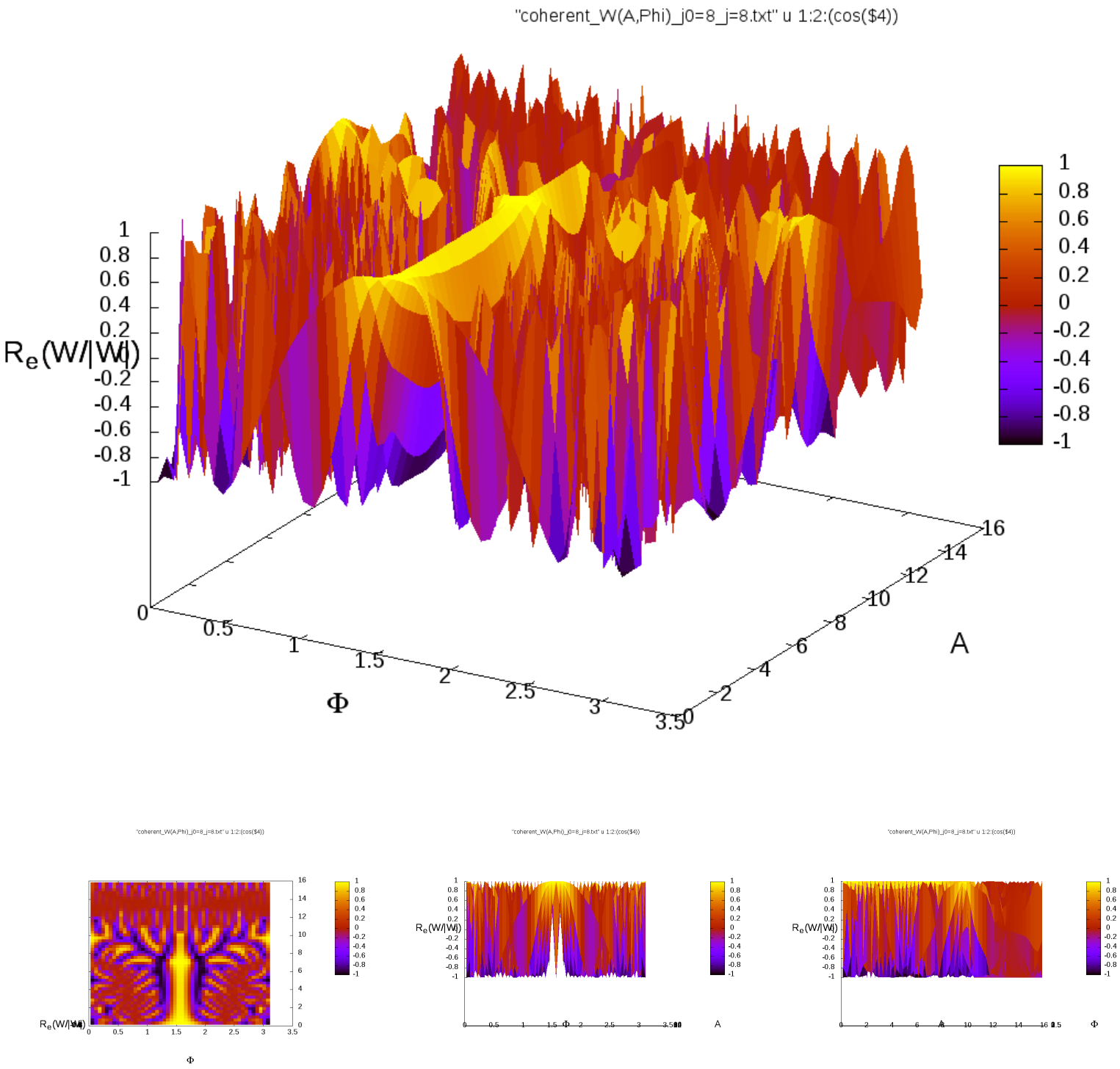


Figure 40: Real part of the phase from the transition amplitude $\mathcal{R}\left(\frac{W(j,j_0,A,\Phi)}{|W(j,j_0,A,\Phi)|}\right)$ in function of (A,Φ) under different views. The first is a isometric views of the surface, the second is the top view (in the pane A, Φ), the third is the front view (in the plane $\mathcal{R}$, Φ) and the fourth is the side view (in the plane $\mathcal{R}$, A).

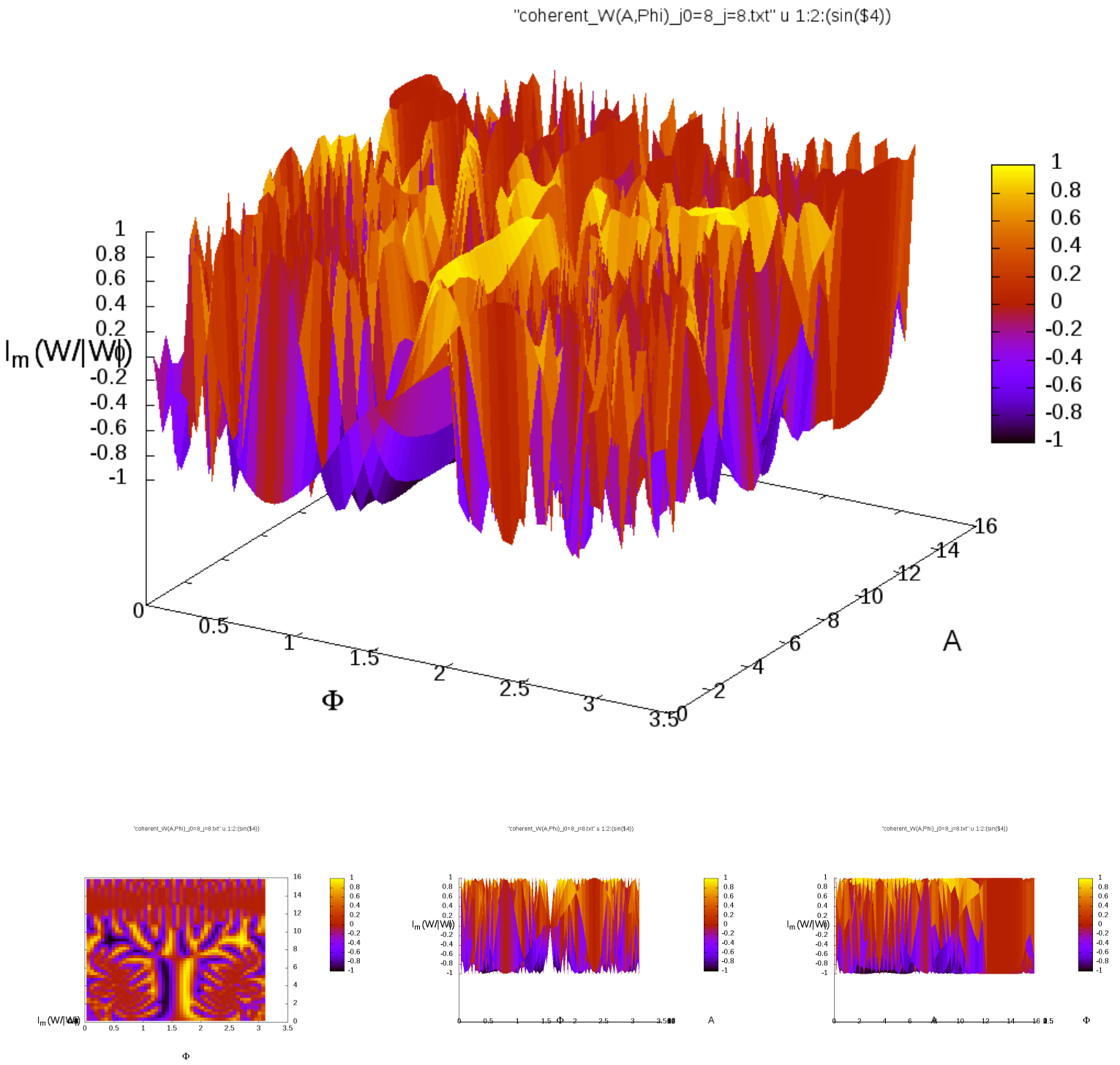


Figure 41: Imaginary part of the phase from the transition amplitude $\mathcal{I}\left(\frac{W(j,j_0,A,\Phi)}{|W(j,j_0,A,\Phi)|}\right)$ in function of (A,Φ) under different views. The first is a isometric views of the surface, the second is the top view (in the pane A, Φ), the third is the front view (in the plane $\mathcal{I}$, Φ) and the fourth is the side view (in the plane $\mathcal{I}$, A).

We show that really oscillating in the space (A, Φ) ! In fact, the most stable part seem to be for $\Phi = \frac{\pi}{2}$, corresponding again to the cylindrical symmetries. If we look more precisely what happen for the cases where $\Phi = \frac{\pi}{2}$:

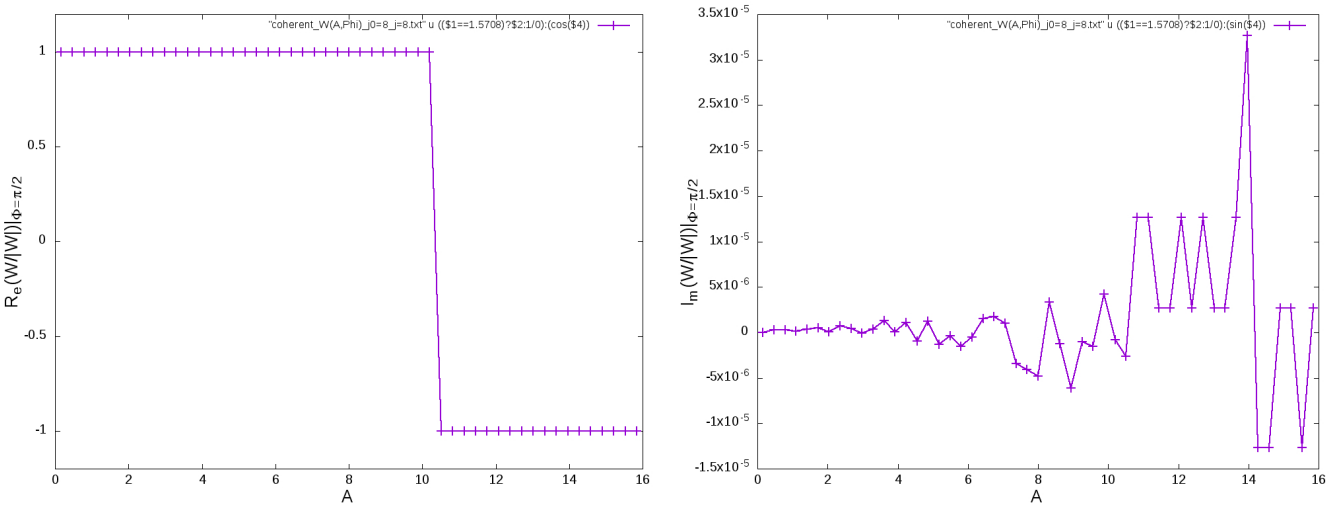


Figure 42: Drawing of real (in the left) and imaginary (in the right) parts of the phase $\frac{W(j,j_0,A,\Phi)}{|W(j,j_0,A,\Phi)|}$ in function of A for $\Phi = \frac{\pi}{2}$ and $j = j_0 = 8$

We see the real parts are dominant with just a sign change for $A > 10.1961$ and the imaginary parts are really small and erratic. At this point, it assumes that the transition amplitude is probably real for the cylindrical symmetry $\Phi = \frac{\pi}{2}$, and the discontinuities of the imaginary part just from some error in the computations. But these computation errors are relative errors of transition amplitude with the magnitude order of 10^{-5} : that are really weak, so we can consider that the computations are really precise. If we look, at A fixed (for example $A = 6.11765$), the evolution of the imaginary part :

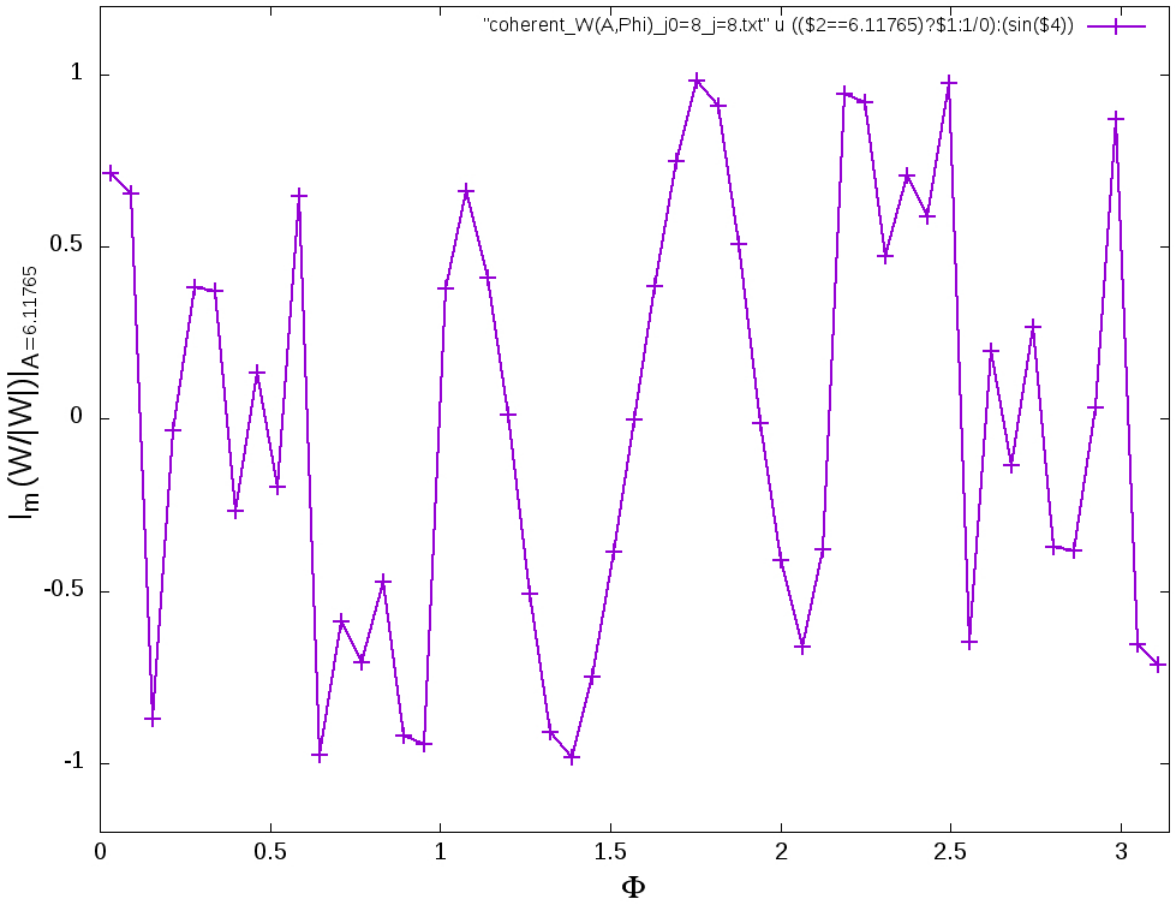


Figure 43: Drawing of the imaginary parts of the phase $\frac{W(j,j_0,A,\Phi)}{|W(j,j_0,A,\Phi)|}$ in function of Φ for $A = 6.11765$ and $j = j_0 = 8$

we have no more ambiguities : the appearing central symmetry of the imaginary part, at the point $(\Phi, \mathcal{I}) = \left(\frac{\pi}{2}, 0\right)$, has a proof of the realness of the transition amplitude for $\Phi = \frac{\pi}{2}$.

The result where the (correct) coherent state applied over the transition amplitude give real coherent transition amplitude is in agreement with the past studies of these objects [6]. The Barrett study suggests that when you put the correct associated coherent geometry to the 4-simplex you have a real results: the realness of coherent amplitude means the coherent tetrahedra states are on-shell the correct geometry. The study of the amplitudes, especially their real parts, will give essential information about the quantum geometry and if they are peaked around their (correct) classical equivalent.

6.2.2 Transition amplitude for j_f-representation

For study more the results from the full transition amplitude, we will see, for $j = j_0 = 8$, the properties of w_f in the phase space (A, Φ) in function of the possible value of j_f. We will draw the norms and phase of w_f for several value of j_f, look their properties with specifying the corresponding classical values, and discuss about the results. The drawings of the norms for the j_f, and their classical expected value of A ($A^{classical} = \frac{2}{\sqrt{3}} a_f$ for $j = j_0$), are the following :

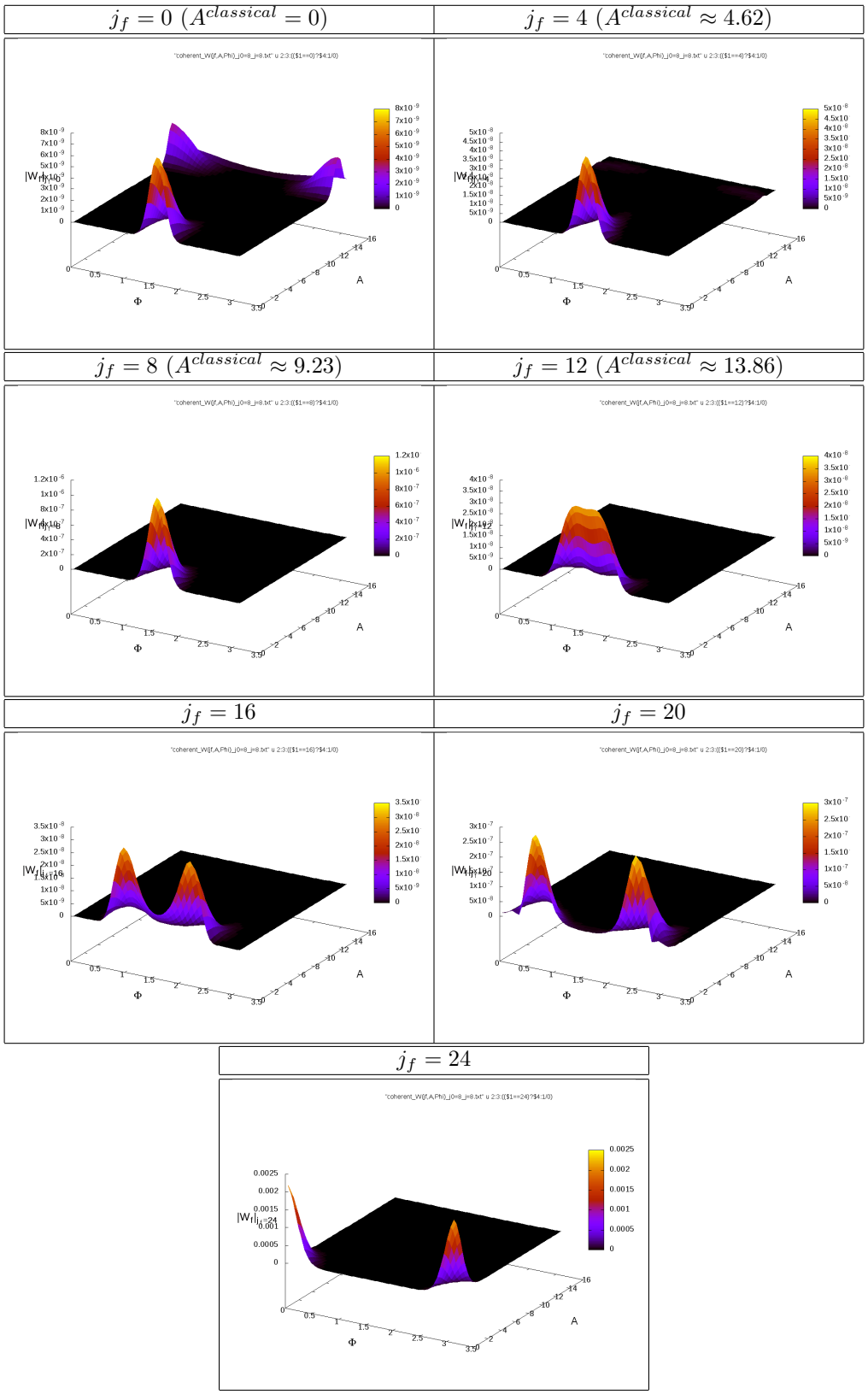


Figure 44: $|w_f(A, \Phi)|$ for some value of j_f, always for $j = j_0 = 8$.

We see interesting results, because we have a transition of the global shape over a critical value $j_f = 12$; call this specific value $j_f^{critical}$. For value of j_f close to $j_f^{critical}$ or smaller we see the transition amplitude are really peaked around the solution $(A, \Phi) = \left(0, \frac{\pi}{2}\right)$. For $j_f = 0$ we have also some degenerated cases around $(A, \Phi) \approx \left(14, \frac{\pi}{2} \pm \frac{\pi}{2}\right)$, but that not relevant because the case $j_f = 0$ is highly degenerated and quantum in itself : the corresponding classical geometry give infinitely elongated tetrahedra. For j_f bigger of $j_f^{critical}$ the transition amplitude tends to take the very degenerate cases $(A, \Phi) = \left(0, \frac{\pi}{2} \pm \frac{\pi}{2}\right)$.

The first results from the values $j_f < j_f^{critical}$ is interesting for two reasons : Primo, we find than the transition amplitude reproduce the cylindrical symmetries with $\Phi = \frac{\pi}{2}$, that means the symmetries of j-parameters reproduce the classical cylindrical symmetries. Secondo, the transition amplitude don't take the expected classical value for A. Classically, give the j, j_0 and j_f parameters fix all geometrical properties of the 4-simplex and should be give a specific value $A\left(j, j_0, j_f\right)$ as in classical geometry, but that not the value selected by the transition amplitude. Probably because the coherent state are not normalized, we have volumes terms given by the norm of coherent state over the intertwiner subspace $\sum_K \left|\left\langle i^K \right| \left\{j, \overrightarrow{n}\right\}\right\rangle\right|^2 \neq 1$, and we don't have renormalization rules for the for the spin-foam part. We have probably some kind of normalization, from the spin-network states and spin-foam, which affect the evolution of the norm of transition amplitude and don't give a clear understanding of physical properties of our quantum geometries. For exploit the physics of the data, and also determine the properties of internal geometries of the assembly, we need to study more the norm AND the phase of the transition amplitude to try to restore the spread/encrypted information. We do that in the next section.

The second results from the values $j_f > j_f^{critical}$ is also interesting, because we see the effect of degenerate geometry reproduce the full transition amplitude. In fact, the degenerate geometry from the case $j_f = 24$ reproduce virtually alone the full transition amplitude. Indeed, if you take the maximum of the degenerated peak from the full coherent transition amplitude $W\left(j, j_0, 0, \frac{\pi}{2} \pm \frac{\pi}{2}\right) \approx 0.109236$ it is composed to 97.35% from the transition amplitude $\left[\left(2j_f + 1\right) w_f\left(j, j_0, j_f, 0, \frac{\pi}{2} \pm \frac{\pi}{2}\right)\right]_{j_f = 24} \approx \left(2 \times 24 + 1\right) \times 0.00217203 \approx 0,10634127$. That clearly show the weight dominance of the degenerate geometries in the full transition amplitude.

For finish, we will talk about the physical sens of $j_f^{critical}$: it seem to be the classical maximum value of j_f for classical geometry. Because, if you take the classical condition for the existence of a 4-simplex with the areas a, a_0, a_f and A, classical constraints say the classical equivalent exist only for $A \in \left[0; \min\left(2a_0, a\sqrt{3}\right)\right]$ (See Subsection 2.2). But, the classical link between A and a_f is $A\sqrt{\frac{4a_0^2 - A^2}{4a^2 - A^2}} = \frac{2}{\sqrt{3}} a_f$ (from (18)) ; that implies for the case $j = j_0 = 8$ (meaning $a = a_0$) the maximum value of j_f : $j_f^{max} = \frac{\sqrt{3}}{2} A^{max} = \frac{3}{2} j = 12$. So for j_f bigger of $j_f^{critical} = 12$ we have no more equivalent classical geometries, only dominate quantum geometry that explain the beginning of the degeneracy from transition amplitude.

Now if we look the real part of the phase from transition amplitude for several values of j_f in the (A, Φ) space :

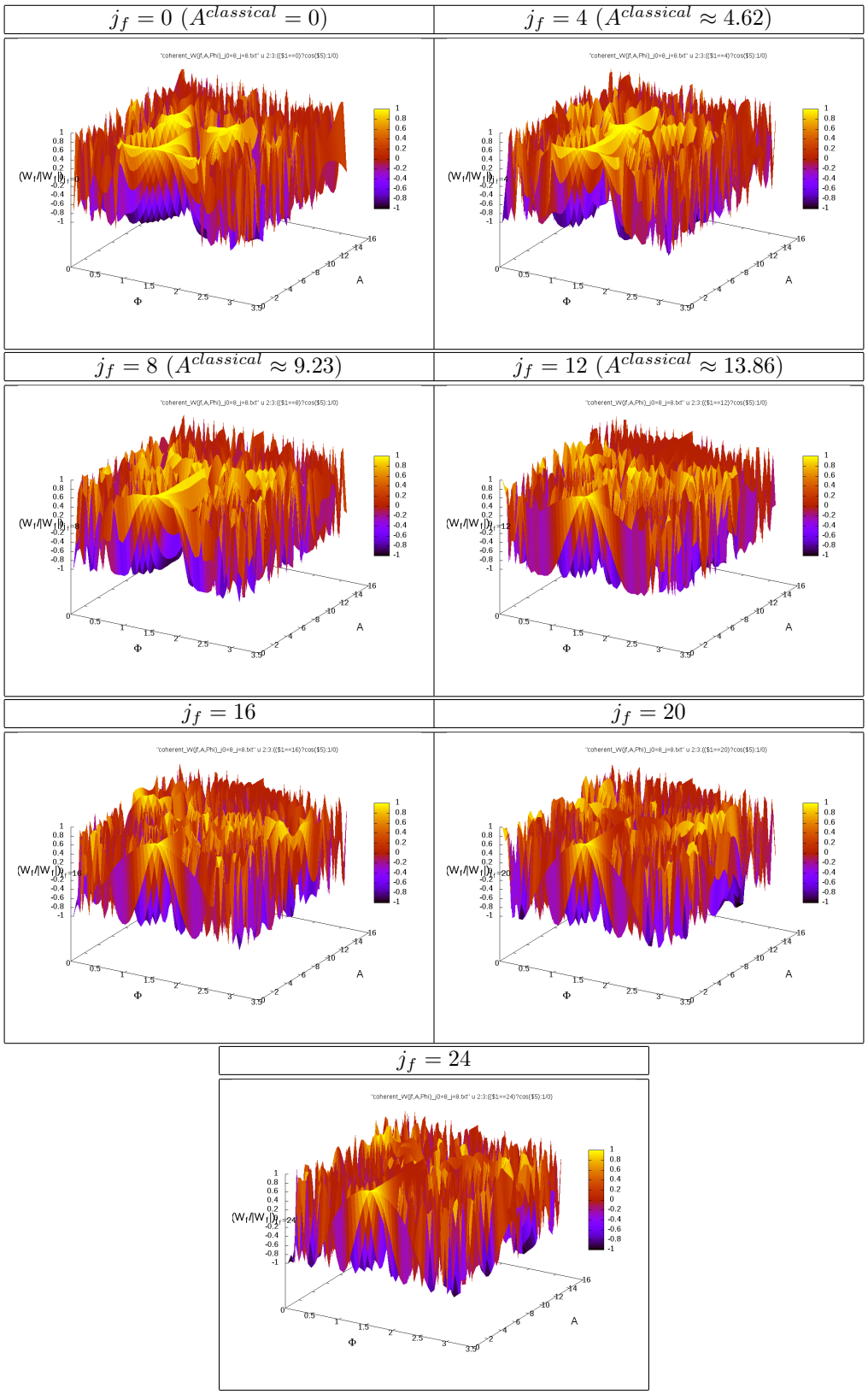


Figure 45: Real part of $\frac{w_f(A,\Phi)}{|w_f(A,\Phi)|}$ for some value of j_f, always for $j = j_0 = 8$.

We have, as the full transition amplitude, very oscillating results except for the cases $\Phi = \frac{\pi}{2}$ where the transition amplitudes are real regardless the value of j_f. So we can conclude that the most stable part of the transition amplitudes w_f are also given for $\Phi = \frac{\pi}{2}$ and preserve, in this sens, the cylindrical symmetries.

6.2.3 Conclusion about the first study of the full transition amplitude

For conclude briefly, the full transition amplitude give a predominance for the very degenerate geometries. Because the non-constrained geometry from j and j_0 promotes the degenerate geometries via the volume effects. The full transition amplitude are, for the scale studied $j = j_0 = 8$, produced essentially from the pure quantum cases where we have no classical equivalent of the geometry :

$$W(j, j_0, A, \Phi) = \sum_{j_f} (2j_f + 1) w_f(j, j_0, j_f, A, \Phi) \approx 49 w_f(j, j_0, 24, A, \Phi) \tag{141}$$

But from the maximum of the w_f norms, for $j_f < j_f^{critical}$, and from the stability region of its phase, for all j_f, we recover the cylindrical symmetry constraint $\Phi = \frac{\pi}{2}$ where the transition amplitude become real. That reality values from the transition amplitude from $\Phi = \frac{\pi}{2}$ are in agreement with the limit and the choice of correct coherent states for 4-simplex as in the Barrett paper [6].

6.3 Internal geometry

If you let free the transition amplitude, for the set of $j \sim 10$, it peak always on the degenerate cases. Because the quantum geometries of degenerate cases are dominant with volumes terms which become bigger. But for a given boundary states, where the j and the shape (A, Φ) parameters are fixed, what internal geometry is selected ? For that we need methods for restore the lost information inside the transition amplitude. So we will study the problem via the integrants of the transition amplitude W : because we can express the full amplitude in terms of sum and integrals over intermediate transition amplitudes, like w_f, which contain information about the internal geometry of our study object. In the optic to get the most probabilistic internal geometries, given the transition amplitude of the assembly, we will talk about quantum conditional probabilities ([24, 25]) and show the physical link with the intermediate transition amplitude. Of course, the quantum probabilities that we will expose are not a strict definitions or introduction to quantum conditional probabilities, just a talk and a physical meaning about the information we try to reproduce from the transition amplitude. After, we will study the norms of these transition amplitude for look if we find classical results.

6.3.1 Quantum conditional probability and transition amplitude for geometry

The conditional probabilities in quantum theories are difficult to define, because their definitions are not unique and give some problem for interpretation. One of these definition come from the extension of Bayes rules to quantum world where the probabilities are define by quantum projector. If you have a system in the state $|\Psi\rangle$, the probability to measure the quantity a associated to the state $|a\rangle$ is given by :

$$P_\Psi(a) = |\langle a|\Psi\rangle|^2 = \left\langle \Psi \left| \hat{Proj}(a) \right| \Psi \right\rangle \tag{142}$$

With the projector associated to a :

$$\hat{Proj}(a) = |a\rangle \langle a| \tag{143}$$

Of course, the corresponding probability is real and positive (because the hermitian properties of projector) and the sum of the all possible value of a (given by the integral over the state $|a\rangle$ in the Lebesgue sens and Lebesgue measure $d\mu(a)$) is equal to 1 :

$$\int d\mu(a) P_\Psi(a) = \langle\Psi| \left(\int d\mu(a) |a\rangle \langle a| \right) |\Psi\rangle = \langle\Psi| \mathbb{I} |\Psi\rangle = 1 \tag{144}$$

In classical probability, the Bayes definitions of conditional probability to have A given B is just :

$$P(A|B) = \frac{P(A,B)}{P(B)} \ , \ P(B) = \sum_A P(A,B) \tag{145}$$

Where $P(A,B)$ is the joint probability to have A and B, and $P(B)$ the probability to have B which must be the sum of all possible probabilities to have B for each compatible A. The most simplest way to extend Bayes rules to quantum with projector are :

$$\left.\begin{array}{c} P_\Psi(a,b) = \left\langle \Psi \left| \hat{Proj}(a,b) \right| \Psi \right\rangle \\ P_\Psi(b) = \int d\mu(a) P_\Psi(a,b) \end{array}\right\} \Rightarrow P_\Psi(a|b) \equiv \frac{P_\Psi(a,b)}{P_\Psi(b)} \tag{146}$$

At this step, we have a specific aspect of quantum conditional probability : the definition of joint probability in term of projector are not necessary real ! But the sum of all probabilities to measure b for each measure a compatible give the correct definition of $P_\Psi(b)$! For example, if in the specific cases where you can express $\hat{Proj}(a,b) = \hat{Proj}(a)\hat{Proj}(b)$ we have :

$$P_\Psi(a,b)^* = \left\langle \Psi \left| \left(\hat{Proj}(a)\hat{Proj}(b) \right)^\dagger \right| \Psi \right\rangle = \left\langle \Psi \left| \hat{Proj}(b)\hat{Proj}(a) \right| \Psi \right\rangle \tag{147}$$

$$= \begin{cases} P_\Psi(a,b) \Rightarrow P_\Psi(a,b) \in \mathbb{R} & \text{if } \hat{Proj}(a),\ \hat{Proj}(b) \text{ commute} \\ P_\Psi(b,a) \Rightarrow P_\Psi(a,b) \in \mathbb{C} & \text{if not} \end{cases} \tag{148}$$

$$\int d\mu(a) P_\Psi(a,b) = \left\langle \Psi \left| \left(\int d\mu(a) |a\rangle \langle a| \right) \hat{Proj}(b) \right| \Psi \right\rangle = \left\langle \Psi \left| \mathbb{I} \hat{Proj}(b) \right| \Psi \right\rangle = P_\Psi(b) \tag{149}$$

So the quantum conditional probability $P_\Psi(a|b)$ with this projector definition are in general complex, and the sum over a of these probabilities give 1. Complex number for probability seem nonphysical, but remember than the quantum mechanics are described by wave functions which can be complex. In quantum mechanics, physical measures is given by the expectation values, so the complexeness of probability itself doesn't matter : only the expectation value –sum over these probabilities– have physical sens. But, we can study the evolution of these conditional probability distributions, in terms of norm, real part, etc... for understand if we have specific value chosen by the system.

In the optics to define conditional probability in our case, we will adapt the previous methodology. The internal geometry are given by the states $\left| j^N_{f,i} \overrightarrow{n^N_{f,i}} \right\rangle$ associated to the internal tetrahedra which have the normal face-vectors $\overrightarrow{n^N_{f,i}}$ and their areas $j^N_{f,i}$, and the external geometry is given by the states $\left| j^N_i \overrightarrow{n^N_i} [A,\Phi] \right\rangle$ defined above for the coherent spin-network. The quantum geometry state is given by $|W\rangle$ and contain the sum over the all possible geometries. So the conditional probability to have a internal geometry given external geometry can be expressed as :

$$P_W\left(j^N_{f,i} \overrightarrow{n^N_{f,i}} | j^N_i \overrightarrow{n^N_i} [A,\Phi] \right) = \frac{\left\langle W \left| \hat{Proj}\left(j^N_{f,i} \overrightarrow{n^N_{f,i}}, j^N_i \overrightarrow{n^N_i} [A,\Phi] \right) \right| W \right\rangle}{\left\langle W \left| \hat{Proj}\left(j^N_i \overrightarrow{n^N_i} [A^N,\Phi^N] \right) \right| W \right\rangle} \tag{150}$$

Where the corresponding projector are "like" :

$$\hat{Proj}\left(j^N_{f,i} \overrightarrow{n^N_{f,i}}, j^N_i \overrightarrow{n^N_i} [A,\Phi] \right) \sim \bigotimes_N \left| j^N_{f,i} \overrightarrow{n^N_{f,i}} \right\rangle \left\langle j^N_{f,i} \overrightarrow{n^N_{f,i}} \right| \otimes \bigotimes_{N,kl} \left| j^N_i \overrightarrow{n^N_i} [A,\Phi] \right\rangle \left\langle j^N_i \overrightarrow{n^N_i} [A,\Phi] \right| \tag{151}$$

$$\hat{Proj}\left(j^N_i \overrightarrow{n^N_i} [A,\Phi] \right) \sim \bigotimes_{N,kl} \left| j^N_i \overrightarrow{n^N_i} [A,\Phi] \right\rangle \left\langle j^N_i \overrightarrow{n^N_i} [A,\Phi] \right| \tag{152}$$

We say "like" because the definition of the action of these states, especially the internal states, on the geometry state $|W\rangle$ are not clearly defined. The Livine-Speziale coherent states, which define the $|j\overrightarrow{n}\rangle$, act on the distribution of

the spin-networks states Ψ and ψ_N (define in the spin-network section (75),(78)) which act on the spin-foam, in the transition amplitude sens, given by W (remember Figure 30). The internal states $\left|j^N_{f,i}\overrightarrow{n^N_{f,i}}\right\rangle$ can be rewritten $\left\langle j^N_{f,i}-\overrightarrow{n^N_{f,i}}\right|$, so the projectors of internal states become :

$$\left|j^N_{f,i}\overrightarrow{n^N_{f,i}}\right\rangle\left\langle j^N_{f,i}\overrightarrow{n^N_{f,i}}\right| \propto \left\langle j^N_{f,i}-\overrightarrow{n^N_{f,i}}\right|\left\langle j^N_{f,i}\overrightarrow{n^N_{f,i}}\right| \tag{153}$$

The state $\left\langle j^N_{f,i}\overrightarrow{n^N_{f,i}}\right|$ correspond to the coherent states associated to a internal tetrahedron from a 4-simplex, and $\left\langle j^N_{f,i}-\overrightarrow{n^N_{f,i}}\right|$ correspond to the coherent states associated to the same tetrahedron but in another 4-simplex where it is reversed ! In other words, $\left\langle j^N_{f,i}-\overrightarrow{n^N_{f,i}}\right|\left\langle j^N_{f,i}\overrightarrow{n^N_{f,i}}\right|$ represent the state of the internal tetrahedron from a 4-simplex coupled to the state of this tetrahedron going in a another 4-simplex ! The projector of joint probability of internal and external geometry become :

$$\hat{Proj}\left(j^N_{f,i}\overrightarrow{n^N_{f,i}}, j^N_i\overrightarrow{n^N_i}\left[A,\Phi\right]\right) \sim \bigotimes_{N,kl}\left|j^N_i\overrightarrow{n^N_i}\left[A,\Phi\right]\right\rangle\bigotimes_N\left\{\left\langle j^N_{f,i}-\overrightarrow{n^N_{f,i}}\right|\left\langle j^N_{f,i}\overrightarrow{n^N_{f,i}}\right|\bigotimes_{kl}\left\langle j^N_i\overrightarrow{n^N_i}\left[A,\Phi\right]\right|\right\} \tag{154}$$

The left action of the projector contain only the geometry of the assembly's boundary, and the right action of the projector contain the geometry of the assembly's boundary and the internal geometry. In fact, the right action of the projector correspond to the product of individual geometry of 4-simplex boundary. With this physic interpretation, we have the joint probability :

$$\begin{aligned} P_W\left(j^N_{f,i}\overrightarrow{n^N_{f,i}}, j^N_i\overrightarrow{n^N_i}\left[A,\Phi\right]\right) &= \left\langle W\left|\hat{Proj}\left(j^N_{f,i}\overrightarrow{n^N_{f,i}}, j^N_i\overrightarrow{n^N_i}\left[A,\Phi\right]\right)\right|W\right\rangle \\ &\sim \langle W|\bigotimes_{N,kl}\left|j^N_i\overrightarrow{n^N_i}\left[A,\Phi\right]\right\rangle\left[\bigotimes_N\left\{\left\langle j^N_{f,i}-\overrightarrow{n^N_{f,i}}\right|\left\langle j^N_{f,i}\overrightarrow{n^N_{f,i}}\right|\bigotimes_{kl}\left\langle j^N_i\overrightarrow{n^N_i}\left[A,\Phi\right]\right|\right\}\right]|W\rangle \\ &\sim \left\langle W|\bigotimes_{N,kl} j^N_{kl}\,\overrightarrow{n}^N_{kl}\left(A,\Phi\right)\right\rangle\prod_N\left\langle \pentagon_N|j,j_0,j_f,A,\Phi,\overrightarrow{n^N_{f,i}},-\overrightarrow{n^N_{f,i}}\right\rangle \end{aligned} \tag{155}$$

We postulate the conditional probability :

$$\begin{aligned} P_W\left(j^N_{f,i}\overrightarrow{n^N_{f,i}}|j^N_i\overrightarrow{n^N_i}\left[A,\Phi\right]\right) &\equiv \frac{\left\langle W|\bigotimes_{N,kl} j^N_{kl}\,\overrightarrow{n}^N_{kl}(A,\Phi)\right\rangle\prod_N\left\langle \pentagon_N|j,j_0,j_f,A,\Phi,\overrightarrow{n^N_{f,i}},-\overrightarrow{n^N_{f,i}}\right\rangle}{\left|\left\langle W|\bigotimes_{N,kl} j^N_{kl}\,\overrightarrow{n}^N_{kl}(A,\Phi)\right\rangle\right|^2} \\ &= \frac{\prod_N\left\langle \pentagon_N|j,j_0,j_f,A,\Phi,\overrightarrow{n^N_{f,i}},-\overrightarrow{n^N_{f,i}}\right\rangle}{\left\langle W|\bigotimes_{N,kl} j^N_{kl}\,\overrightarrow{n}^N_{kl}(A,\Phi)\right\rangle} \end{aligned} \tag{156}$$

in accord of the internal coherent structure of W ((108) and (118)) and the definition of conditional probability. P_W can be seen as a (complex) probability density function over the $\overrightarrow{n^N_{f,i}}$ variables, where the sum over the all internal geometry of these conditional probability reproduce 1 :

$$\underbrace{\sum_{j_f}}_{\substack{\text{sum over the}\\ \text{all possible}\\ \text{area for } f}}\ (2j_f+1)\underbrace{\prod_N\underbrace{\left((2j_f+1)(2j_0+1)^3\prod_i\int\frac{d\overrightarrow{n^N_{f,i}}}{4\pi}\right)}_{\substack{\text{sum over the geometries of}\\ \text{one shared-tetrahedron}}}}_{\substack{\text{sum over the geometries of}\\ \text{all shared-tetrahedra}}}\ \underbrace{P_W\left(j^N_{f,i}\overrightarrow{n^N_{f,i}}|j^N_i\overrightarrow{n^N_i}\left[A,\Phi\right]\right)}_{\substack{\text{"conditional probability" to have:}\\ \text{internal geometries } j^N_{f,i},\overrightarrow{n^N_{f,i}}\\ \text{given}\\ \text{external geometries } j^N_i,\overrightarrow{n^N_i}\left[A,\Phi\right]}} \tag{157}$$

$$= \sum_{j_f}(2j_f+1)\prod_N\left((2j_f+1)(2j_0+1)^3\prod_i\int\frac{d\overrightarrow{n^N_{f,i}}}{4\pi}\right)\frac{\prod_N\left\langle \pentagon_N|j,j_0,j_f,A,\Phi,\overrightarrow{n^N_{f,i}},-\overrightarrow{n^N_{f,i}}\right\rangle}{\left\langle W|\bigotimes_{N,kl} j^N_{kl}\,\overrightarrow{n}^N_{kl}(A,\Phi)\right\rangle}$$
$$= \sum_{j_f}(2j_f+1)\frac{w_f(j,j_0,j_f,A,\Phi)}{\left\langle W|\bigotimes_{N,kl} j^N_{kl}\,\overrightarrow{n}^N_{kl}(A,\Phi)\right\rangle} = 1$$

Of course, the definition of $P_W\left(j^N_{f,i}\overrightarrow{n^N_{f,i}}|j^N_i\overrightarrow{n^N_i}\,[A,\Phi]\right)$ is in fact the (renormalized) transition amplitude to have the three 4-simplices given by the parameters (j,j_0,A,Φ), $\left(j_f,j_0,\overrightarrow{n^N_{f,i}}\right)$ and is like the quantum conditional probability to have internal tetrahedra with the normal face-vectors $\overrightarrow{n^N_{f,i}}$ and the area $j^N_{f,i}=(j_f,j_0,j_0,j_0)$ given the boundary parameters A, Φ and $j^N_i=(j,j,j_0,j_0)$. We can define more conditional probabilities, which are just parts from transition amplitudes, whose properties will give us information about the internal geometries.

So we want compute these transition amplitudes and study their properties : peakedness of the norms, evolution of the complex phase... etc. The idea is to extract some information about the chosen internal geometry : for example, in the case where the conditional probability $P_W\left(a|b\right)$ is purely real, that means we can physically measure a and b regardless without no quantum interaction between them. In this special case, the norm of this probability (which is just the absolute value) can be interpreted as a classical conditional probability : so the most probabilistic value of a given b is just the value of a which maximize $|P_W\left(a|b\right)|$. For the general case, where $P_W\left(a|b\right)$ is complex, you need to study also the complex phase in addition to the norm : because the complex phase contain a part of the action, and the minimization of this action give the classical evolution of probabilities and classical value of a. Simply, transition amplitudes and quantum conditional probability $P_W\left(a|b\right)$ can be seen like :

$$P_W\left(a|b\right)=\frac{\langle W|a,b\rangle}{\langle W|b\rangle}=\frac{|\langle W|a,b\rangle|\,e^{i\arg\langle W|a,b\rangle}}{|\langle W|b\rangle|\,e^{i\arg\langle W|b\rangle}}=\frac{e^{-iS_W[a,b]}}{e^{-iS_W[b]}}\;\rightarrow\;\int\mathcal{D}aP_W\left(a|b\right)=\frac{\int\mathcal{D}ae^{-iS_W[a,b]}}{e^{-iS_W[b]}}=1 \tag{158}$$

With the quantum action $S_W\left[a,b\right]\in\mathbb{C}$ which are minimized for the classical evolution of geometries with a and b in the Feynman integrals sens. So the special value of a where $\frac{\partial}{\partial a}S_W\left[a,b\right]=0\iff\frac{\partial}{\partial a}P_W\left(a|b\right)=0$ correspond to the most probabilistic/contributory value of the integral and, by definition, the classical value. Of course for the case where $P_W\left(a|b\right)$ is real, means $S_W\left[a,b\right]-S_W\left[b\right]\in i\mathbb{R}$, we recover the classical case as the maximum of $|P_W\left(a|b\right)|$. Briefly, the study of the norms and phase of transition amplitude integrants will give us the most "probabilistic" internal geometry for given boundary parameter.

6.3.2 Prelude and used conditional probabilities for the amplitude analysis

Let us recall that we have 6 internal parameters for the shared-tetrahedra that are being summed over in the path integral for build the transition amplitude: the area of the bulk face, j_f and 5 angles θ,ϕ. The latter characterize a configuration of 4 unit vectors up to a global rotation used to align one of them with the z axis, and a second one to lie on the Greenwich meridian (plane Oxz). If the model has the correct semi-classical behaviour, we expect the amplitudes to be peaked on configurations satisfying the classical conditions: first of all, it should be peaked on a closed configuration of the four vectors, which then represents a flat tetrahedron characterized by the areas and two angles; by symmetry assumptions, the internal tetrahedra θ are equal and described by the same data; then, the remaining two angles and the area j_f should be peaked on the Regge configurations determined by the boundary data as studied in Sections 2,2.2, and given by equilateral tetrahedra.

We will study the peakedness in three different ways, which can be seen via the decomposition in quantum sum from transition amplitude: (118). First (6.3.3), we keep the sums over j_f, and keep the 5 angles free. Requiring that both real and imaginary parts of the amplitudes are maximal, we will find that the chosen configuration indeed corresponds to the classical equilateral one, where the three θ are equal, and the two ϕ's are $\frac{2\pi}{3}$ and $\frac{4\pi}{2}$; furthermore (6.3.4) it gives the right classical geometric value for boundary data which are far from degenerate configurations. Second (6.3.4), we will study the reverse, keeping j_f free and integrate over the five internal angles. In this case we have again a region where the classical geometry appears, but the disparities and the degenerate geometries region are bigger. Third (6.3.6), we consider a more off-shell amplitude where we assume that the three shared-tetrahedra are equilateral with their θ equal (the ϕ are fixed by the equilateral geometry) and we allow it to vary together with j_f ; we have no more integration and have just the product of the three amplitude with just this specific case of symmetries for shared-tetrahedra. In this last case, we observe that the amplitude reproduce very well the classical geometry, but degenerate region is always present.

The corresponding probabilities and amplitudes which will be studied will be the following :

- $P\left(\overrightarrow{n_{f,i}}|A,\Phi\right)$ in the Subsection 6.3.3

– Which define the probability to have the 5 parameters $(\theta_{f,2}, \theta_{f,3}\phi_{f,3}, \theta_{f,4}\phi_{f,4})$ for the all shared-terahedra *given* the boundary shape parameters (A, Φ) for $j = j_0 = 8$. The probability is defined as :

$$P\left(\overrightarrow{n_{f,i}}|A,\Phi\right) \equiv \frac{1}{\langle W|j,j_0,A,\Phi\rangle}\sum_{j_f}\left(2j_f+1\right)\langle\pentagon|j,j_0,j_f,A,\Phi,\overrightarrow{n_{f,i}},-\overrightarrow{n_{f,i}}\rangle^3 \tag{159}$$

with the definitions for the coherent 4-simplex :

$$\begin{array}{l}\langle\pentagon|j,j_0,j_f,A,\Phi,\overrightarrow{n_{f,i}},-\overrightarrow{n_{f,i}}\rangle \\ =\sum_{J,K}\left[J_k,K_4,K_5;j,j_0,j_f\right]\left\langle i^{K_4}|j_f,j_0,\overrightarrow{n_{f,i}}\right\rangle\overline{\left\langle i^{K_5}|j_f,j_0,\overrightarrow{n_{f,i}}\right\rangle}\prod_{k=1}^{3}\left\langle i^{J_k}|j,j_0,A,\Phi\right\rangle\end{array} \tag{160}$$

and the coherent states for shared-tetrahedra :

$$\left\langle i^K|j_f,j_0,\overrightarrow{n_{f,i}}\right\rangle = \sum_m i^K_{j_f m_1 m_2 m_3} D^{j_0}_{m_1 j_0}\left(\theta_{f,2},0\right) D^{j_0}_{m_2 j_0}\left(\theta_{f,3},\phi_{f,3}\right) D^{j_0}_{m_3 j_0}\left(\theta_{f,4},\phi_{f,4}\right) \tag{161}$$

which come from 5.2.3 and can be off-shell of the tetrahedron geometry (closure condition not necessarily respected).

- $P\left(\theta_f|A\right)$ in the Subsection 6.3.4
  – Which define the probability to have parameters θ_f for the all shared-tetrahedra *given* the boundary shape parameter A for $j = j_0 = 8$. The probability is defined as :

$$P\left(\theta_f|A\right) \equiv \frac{1}{\left\langle W|j,j_0,A,\frac{\pi}{2}\right\rangle}\sum_{j_f}\left(2j_f+1\right)\left\langle\pentagon|j,j_0,j_f,A,\frac{\pi}{2},\overrightarrow{n_{f,i}}\left[\theta_f\right],-\overrightarrow{n_{f,i}}\left[\theta_f\right]\right\rangle^3 \tag{162}$$

with the definitions for the coherent 4-simplex :

$$\begin{array}{l}\left\langle\pentagon|j,j_0,j_f,A,\frac{\pi}{2},\overrightarrow{n_{f,i}}\left[\theta_f\right],-\overrightarrow{n_{f,i}}\left[\theta_f\right]\right\rangle \\ =\sum_{J,K}\left[J_k,K_4,K_5;j,j_0,j_f\right]\left\langle i^{K_4}|j_f,j_0,\overrightarrow{n_{f,i}}\left[\theta_f\right]\right\rangle\overline{\left\langle i^{K_5}|j_f,j_0,\overrightarrow{n_{f,i}}\left[\theta_f\right]\right\rangle}\prod_{k=1}^{3}\left\langle i^{J_k}|j,j_0,A,\frac{\pi}{2}\right\rangle\end{array} \tag{163}$$

and the coherent states for shared-tetrahedra :

$$\left\langle i^K|j_f,j_0,\overrightarrow{n_{f,i}}\left[\theta_f\right]\right\rangle = \sum_m i^K_{j_f m_1 m_2 m_3} D^{j_0}_{m_1 j_0}\left(\theta_f,0\right) D^{j_0}_{m_2 j_0}\left(\theta_f,\frac{2\pi}{3}\right) D^{j_0}_{m_3 j_0}\left(\theta_f,\frac{4\pi}{3}\right) \tag{164}$$

which come from 5.2.3 and can be off-shell of the tetrahedron geometry if $\cos\theta_f \neq -\frac{j_f}{3j_0}$.

- $P\left(j_f|A\right)$ in the Subsection 6.3.4
  – Which define the probability to have parameter j_f for the internal face f *given* the boundary shape parameter A for $j = j_0 = 8$. The probability is defined as :

$$P\left(j_f|A\right) \equiv \frac{\left(2j_f+1\right)w_f\left(j,j_0,j_f,A,\frac{\pi}{2}\right)}{\left\langle W|j,j_0,A,\frac{\pi}{2}\right\rangle} \tag{165}$$

with the w_f from (116) which represent the quantum summation over the shared-tetrahedra geometry for given j_f.

- $P\left(j_f,\theta_f|A\right)$ in the Subsection 6.3.6
  – Which define the probability to have parameters j_f and θ_f for the all shared-tetrahedra *given* the boundary shape parameter A for $j = j_0 = 8$. The probability is defined as :

$$P\left(j_f,\theta_f|A\right) \equiv \frac{\left(2j_f+1\right)}{\left\langle W|j,j_0,A,\frac{\pi}{2}\right\rangle}\left\langle\pentagon|j,j_0,j_f,A,\frac{\pi}{2},\overrightarrow{n_{f,i}}\left[\theta_f\right],-\overrightarrow{n_{f,i}}\left[\theta_f\right]\right\rangle^3 \tag{166}$$

with the definitions :

$$\begin{array}{l}\langle \pentagon | j, j_0, j_f, A, \Phi, \overrightarrow{n_{f,i}}\left[\theta_f\right], -\overrightarrow{n_{f,i}}\left[\theta_f\right]\rangle \\ = \sum_{J,K}\left[J_k, K_4, K_5; j, j_0, j_f\right]\left\langle i^{K_4}|j_f, j_0, \overrightarrow{n_{f,i}}\left[\theta_f\right]\right\rangle \overline{\left\langle i^{K_5}|j_f, j_0, \overrightarrow{n_{f,i}}\left[\theta_f\right]\right\rangle}\prod_{k=1}^{3}\left\langle i^{J_k}|j, j_0, A, \Phi\right\rangle\end{array} \tag{167}$$

and the coherent states for shared-tetrahedra :

$$\left\langle i^K|j_f, j_0, \overrightarrow{n_{f,i}}\left[\theta_f\right]\right\rangle = \sum_m i^K_{j_f m_1 m_2 m_3} D^{j_0}_{m_1 j_0}\left(\theta_{f,2}, 0\right) D^{j_0}_{m_2 j_0}\left(\theta_{f,3}, \phi_{f,3}\right) D^{j_0}_{m_3 j_0}\left(\theta_{f,4}, \phi_{f,4}\right) \tag{168}$$

which come from 5.2.3 and can be off-shell of the tetrahedron geometry if $\cos\theta_f \neq -\frac{j_f}{3j_0}$

6.3.3 Numerical result for $P\left(\overrightarrow{n_{f,i}}|A, \Phi\right)$

We are interested to :

$$P\left(\overrightarrow{n_{f,i}}|A, \Phi\right) \equiv \frac{1}{\langle W|j, j_0, A, \Phi\rangle}\sum_{j_f}\left(2j_f+1\right)\langle \pentagon | j, j_0, j_f, A, \Phi, \overrightarrow{n_{f,i}}, -\overrightarrow{n_{f,i}}\rangle^3 \tag{169}$$

which is linked to the transition amplitude of the three 4-simplices from the assembly, with the quantum summation over the all possible area j_f for the face f, but don't have the quantum summation over the face-vectors $\overrightarrow{n_{f,i}}$ from shared-tetrahedra (see the link with (118)). In this context, the all shared-tetrahedra have the same geometry given by the $\overrightarrow{n_{f,i}}$ vectors and their five parameters $(\theta_{f,2}, \theta_{f,3}\phi_{f,3}, \theta_{f,4}\phi_{f,4})$. For a set of j,j_0 and A, Φ given, we want compute $P\left(\overrightarrow{n_{f,i}}|A, \Phi\right)$ and see what $\overrightarrow{n_{f,i}}$ are chosen.

The main reason of this interest and simplification is because the space of variables in the general case is really huge ! In the general case, each face-vectors $\overrightarrow{n^N_{f,i}}$ of shared-tetrahedra are given by two parameters $\left(\theta^N_{f,i}, \phi^N_{f,i}\right)$ (see coherent states Subsection 5.2.3), that give 5 parameters for each shared-tetrahedron with their 4 face-vectors : you can use gauge fixing to set $\left(\theta^N_{f,1}, \phi^N_{f,1}\right) = (0,0)$ and $\phi^N_{f,2} = 0$. Give a total of 15 parameters for the three shared-tetrahedra. Because the individual 4-simplex sections 6.1.1 show some clues that the transition amplitude for the 4-simplices are peaked around the symmetric cases, where the $\overrightarrow{n^N_{f,i}}$ are the same for all internal tetrahedra, we will reduce the number of parameters and just compute this probability density function to the five parameters $(\theta_{f,2}, \theta_{f,3}\phi_{f,3}, \theta_{f,4}\phi_{f,4})$ from $\overrightarrow{n_{f,i}} = \{\overrightarrow{n_{f,1}}(0,0), \overrightarrow{n_{f,2}}(\theta_{f,2}, 0), \overrightarrow{n_{f,3}}(\theta_{f,3}\phi_{f,3}), \overrightarrow{n_{f,4}}(\theta_{f,4}\phi_{f,4})\}$. That give the definition of the coherent 4-simplex used :

$$\begin{array}{l}\langle \pentagon | j, j_0, j_f, A, \Phi, \overrightarrow{n_{f,i}}, -\overrightarrow{n_{f,i}}\rangle \\ = \sum_{J,K}\left[J_k, K_4, K_5; j, j_0, j_f\right]\left\langle i^{K_4}|j_f, j_0, \overrightarrow{n_{f,i}}\right\rangle \overline{\left\langle i^{K_5}|j_f, j_0, \overrightarrow{n_{f,i}}\right\rangle}\prod_{k=1}^{3}\left\langle i^{J_k}|j, j_0, A, \Phi\right\rangle\end{array} \tag{170}$$

and the associated coherent states for shared-tetrahedra :

$$\left\langle i^K|j_f, j_0, \overrightarrow{n_{f,i}}\right\rangle = \sum_m i^K_{j_f m_1 m_2 m_3} D^{j_0}_{m_1 j_0}\left(\theta_{f,2}, 0\right) D^{j_0}_{m_2 j_0}\left(\theta_{f,3}, \phi_{f,3}\right) D^{j_0}_{m_3 j_0}\left(\theta_{f,4}, \phi_{f,4}\right) \tag{171}$$

In this first approach, we want find some symmetry properties between the parameters $\theta_{f,i}$, $\phi_{f,i}$ for reduce again the parameters and try to get the maximum of precision for the calculus. So, for fixed j,j_0, and each shape parameters A, Φ we will compute $P\left(\overrightarrow{n_{f,i}}|A, \Phi\right)$ and take the value of $(\theta_{f,2}, \theta_{f,3}\phi_{f,3}, \theta_{f,4}\phi_{f,4})$ which maximize its norm. We can draw the "chosen" values of $(\theta_{f,2}, \theta_{f,3}\phi_{f,3}, \theta_{f,4}\phi_{f,4})$ in function of the shape parameters A, Φ :

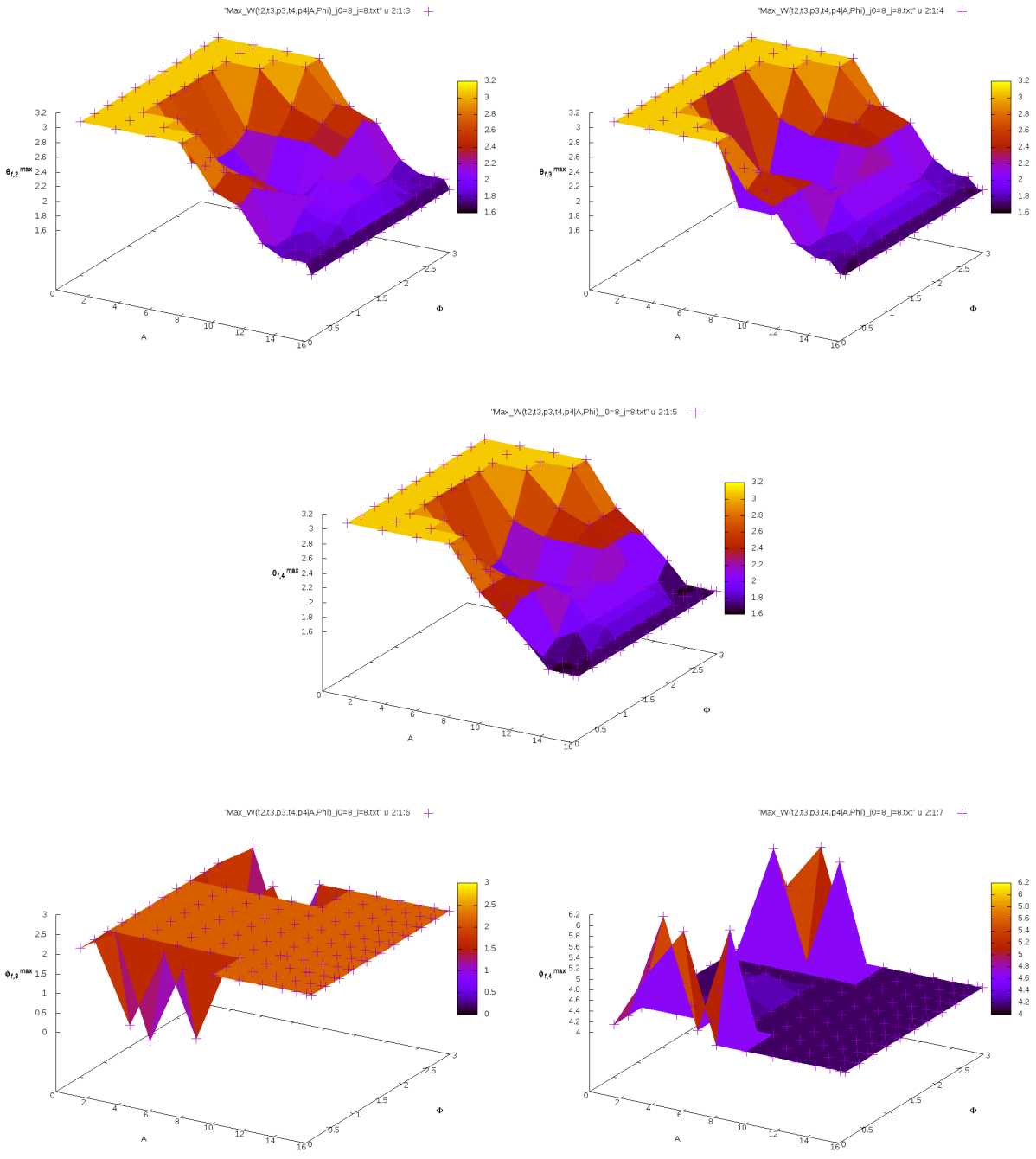


Figure 46: Value of $\theta_{f,2}$, $\theta_{f,3}$, $\theta_{f,4}$, $\phi_{f,3}$ and $\phi_{f,4}$ which maximize the norm of $P\left(\overrightarrow{n_{f,i}}|A,\Phi\right)$ in function of A, Φ. We see appearing the equilateral symmetry $\theta_{f,i}=\theta_f$ and $\phi_{f,3}=\frac{2\pi}{3}$, $\phi_{f,4}=\frac{4\pi}{3}$.

The precision of the chosen values $(\theta_{f,2},\theta_{f,3}\phi_{f,3},\theta_{f,4}\phi_{f,4})$ is not really good, because the number of parameters and the limitation of computer impose to have a incertitude of $\pm\frac{\pi}{11}$ $(\pm 9.1\%)$, but we can clearly see that the cases where the $\theta_{f,i}$ are equal and $\phi_{f,3}=\frac{2\pi}{3}$, $\phi_{f,4}=\frac{4\pi}{3}$ are the main solutions. Except for the very degenerate geometries from the regions $(A,\Phi)\sim\left(0,\frac{\pi}{2}\pm\frac{\pi}{2}\right)$, we obtain the equilateral symmetries properties for the shared-tetrahedra as

in the classical equivalent. That reinforces the idea where the cylindrical symmetries of the boundary impose only the equilateral cases for the geometry of the shared-tetrahedra, with a equilateral face f for base and same isosceles triangle for the j_0-faces. Note, the regions of degenerate geometries where the equilateral face is not preserved are the same as those from the individual 4-simplex amplitude study (Subsection 6.1.1, Figures 32,33).

Because the symmetries between the $\theta_{f,i}$ (see Figures 46), we can do the same computation with more precision for the restricted equilateral geometries where $\theta_{f,i} = \theta_f$ and $\phi_{f,3} = \frac{2\pi}{3}$, $\phi_{f,4} = \frac{4\pi}{3}$. So we can draw the specific values of θ_f which maximize the norm of $P\left(\overrightarrow{n_{f,i}}\left[\theta_f\right]|A,\Phi\right)$ for the equilateral cases in function of A, Φ :

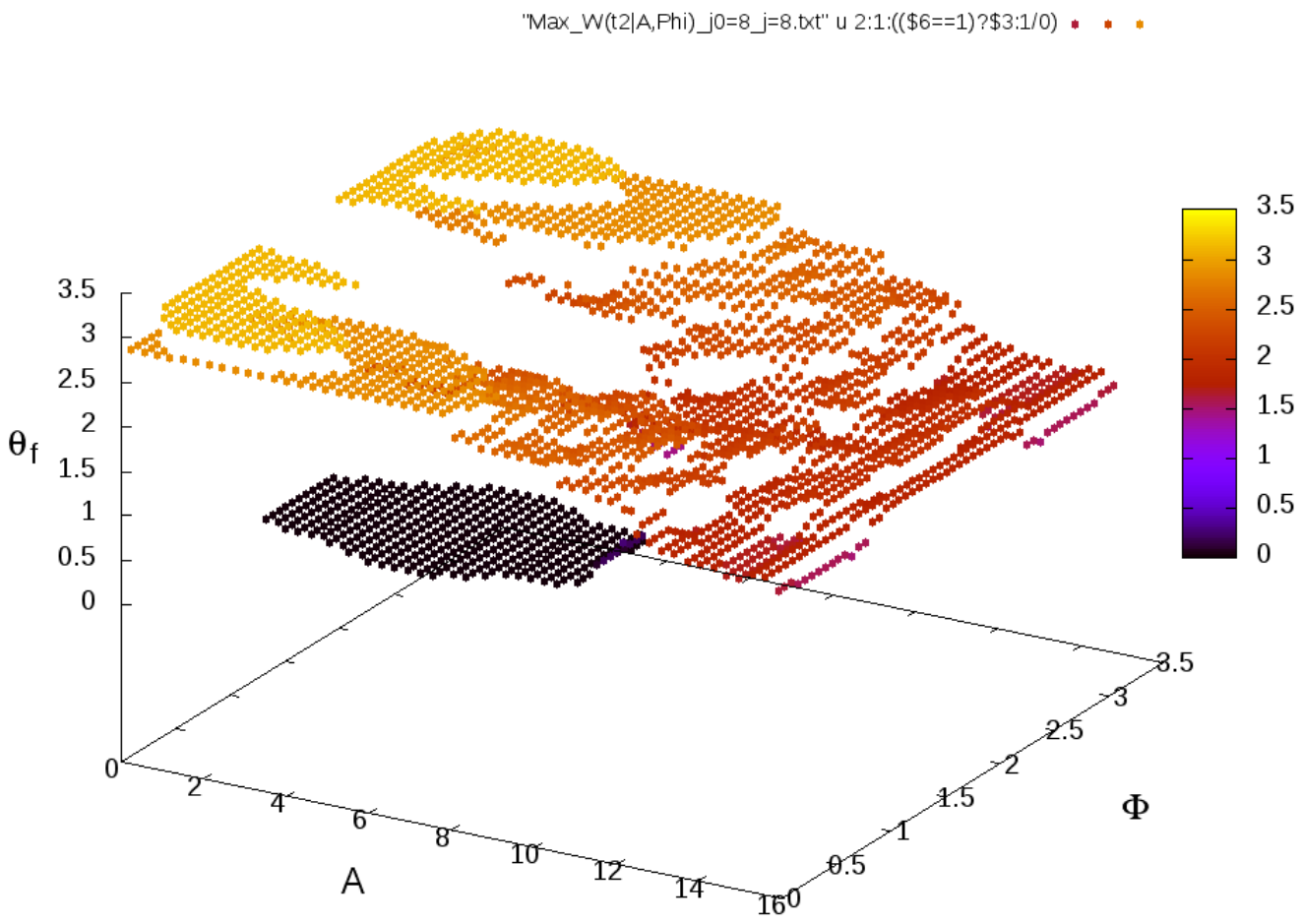


Figure 47: Value of θ_f which maximize $\left|P\left(\overrightarrow{n_{f,i}}\left[\theta_f\right]|A,\Phi\right)\right|$ for the equilateral cases $\left(\theta_{f,i} = \theta_f; \phi_{f,3} = \frac{2\pi}{3}; \phi_{f,4} = \frac{4\pi}{3}\right)$ in function A, Φ.

Now, with more precise data, we can draw and look the evolution of $P\left(\overrightarrow{n_{f,i}}\left[\theta_f\right]|A,\Phi\right)$ in function of θ_f, for fixed A, Φ :

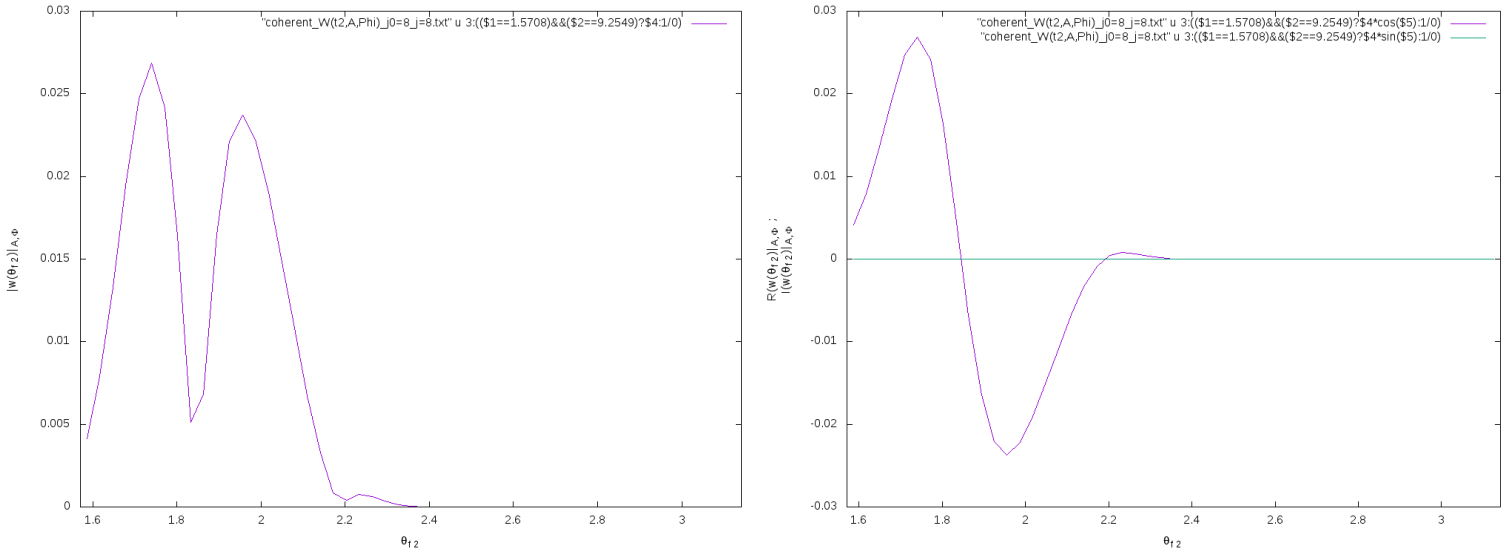


Figure 48: Norm (purple, in the left) and real & imaginary (purple & green, in the right) parts of $P\left(\overrightarrow{n_{f,i}}\left[\theta_f\right]|A,\Phi\right)$ in function of θ_f, for $A = 9.2549$ and $\Phi = \frac{\pi}{2}$.

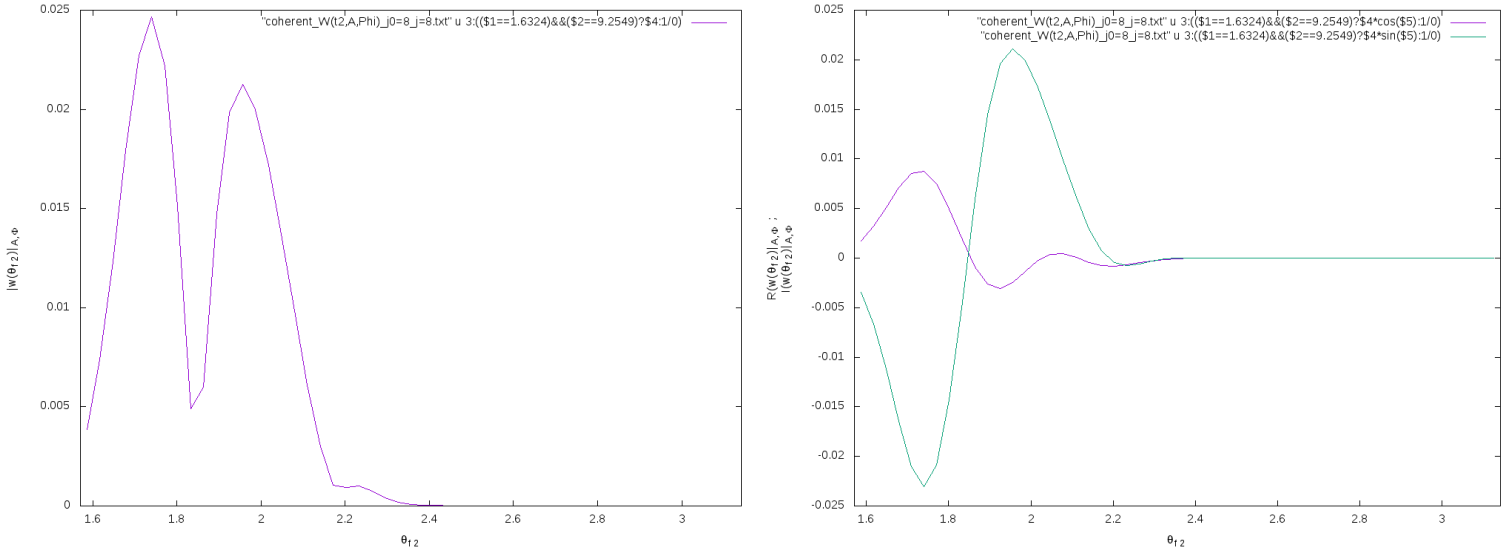


Figure 49: Norm (purple, in the left) and real & imaginary (purple & green, in the right) parts of $P\left(\overrightarrow{n_{f,i}}\left[\theta_f\right]|A,\Phi\right)$ in function of θ_f, for $A = 9.2549$ and $\Phi = 1.6324$.

We have two interesting type of results, the first from the Figure 48 correspond to the geometry with the cylindrical symmetries $\Phi = \frac{\pi}{2}$, and the second from the Figure 49 correspond to a geometry without cylindrical symmetries.

The first result from $\Phi = \frac{\pi}{2}$ (Figure 48), as the previous results from the Subsection 6.2 and the Figures 42,43, give $P\left(\overrightarrow{n_{f,i}}\left[\theta_f\right]|A,\Phi\right)$ real. By virtue of the talk about quantum conditional probabilities in Subsection 6.3.1, the realness of $P\left(\overrightarrow{n_{f,i}}\left[\theta_f\right]|A,\Phi\right)$ allow to understand this probability "like" a classical probability : the maximum values of the norm give the most probabilistic result chosen by the quantum geometry. So we have two peak from the norm, corresponding to the two most probabilistic solution of θ_f : the first, which is the

higher, is for $\theta_f \sim 1.7402$, the second is for $\theta_f \sim 1.9558$, which are very close to the classical expected value $\theta_f^{classical} = \arccos\left(-\frac{1}{\sqrt{3}} \cdot \frac{A}{2a_0}\right) \approx 1.9113$. The classical geometry is found in the quantum geometry via this second peak and correspond here to a classical solution with curvature ! Indeed, from the classical equations (21) and (22) we can express the angle between two shared-tetrahedra :

$$\cos\Theta_f = \frac{\cos\theta_A - \cos^2\theta_f}{1-\cos^2\theta_f} = \frac{\left(1-\frac{A^2}{2a_0^2}\right) - \cos^2\theta_f}{1-\cos^2\theta_f} \approx 0.2209 \tag{172}$$

and the deficit angle of f :

$$\varepsilon_f = 2\pi - 3\Theta_f \approx 2,2392 \text{ rad} \tag{173}$$

But the main solution, given by the first peak, is a priory not a classical solution. Note, if you suppose that the formula given by (21) and (22) can be extended to the quantum geometry, we find again a solution with curvature. That a not a proof, because the formula is used beyond this validity domain in this case, but it's maybe a clue of the presence of curvature.

The second result from $\Phi = 1.6324$ (Figure 49) give $P\left(\overrightarrow{n_{f,i}}\left[\theta_f\right]|A,\Phi\right)$ complex, so it's more difficult to determinate what is the "classical solution chosen" by the amplitude. By the norm, we have again two peak which almost the same of the Figure 48, that is normal because the value of Φ are close to the cylindrical/classical case $\frac{\pi}{2}$. The most interesting part is given by the real and imaginary parts of $P\left(\overrightarrow{n_{f,i}}\left[\theta_f\right]|A,\Phi\right)$, because we show that the both give a same local extremum for the solution $\theta_f \sim 1.7402$ but differ for the solution $\theta_f \sim 1.9558$. So if like in the section 6.3.1 we define the classical solution as the specific solutions where $\frac{\partial}{\partial a}P_W\left(a|b\right) = 0$, the corresponding selected solution is the first peak of the norm where the real and imaginary parts have the same local extremum. The second peak of the norms come with the local extremum of the real part, but no extremum for the imaginary part. We don't know exactly what to think, but it's important to reveal these particularities.

6.3.4 Numerical results for $P\left(\theta_f|A\right)$

Guided by the previous results, where the symmetric shared-tetrahedra with $\theta_{f,i} = \theta_f$ and $(\phi_{f,3},\phi_{f,4}) = \left(\frac{2\pi}{3},\frac{4\pi}{3}\right)$ are dominant, we will study more precisely the properties of the conditional probability $P\left(\theta_f|A\right) \equiv P\left(\overrightarrow{n_{f,i}}\left[\theta_f\right]|A,\frac{\pi}{2}\right)$. In this equilateral case, with the cylindrical symmetry $\Phi = \frac{\pi}{2}$, the all results will be real so we can just study the norms of $P\left(\theta_f|A\right)$. We can draw precisely the values of θ_f, in function of A, with a color for indicate if it correspond to a maximum or not for the fixed value of A. We have the colored drawing and chosen values of θ_f given by the following Figures :

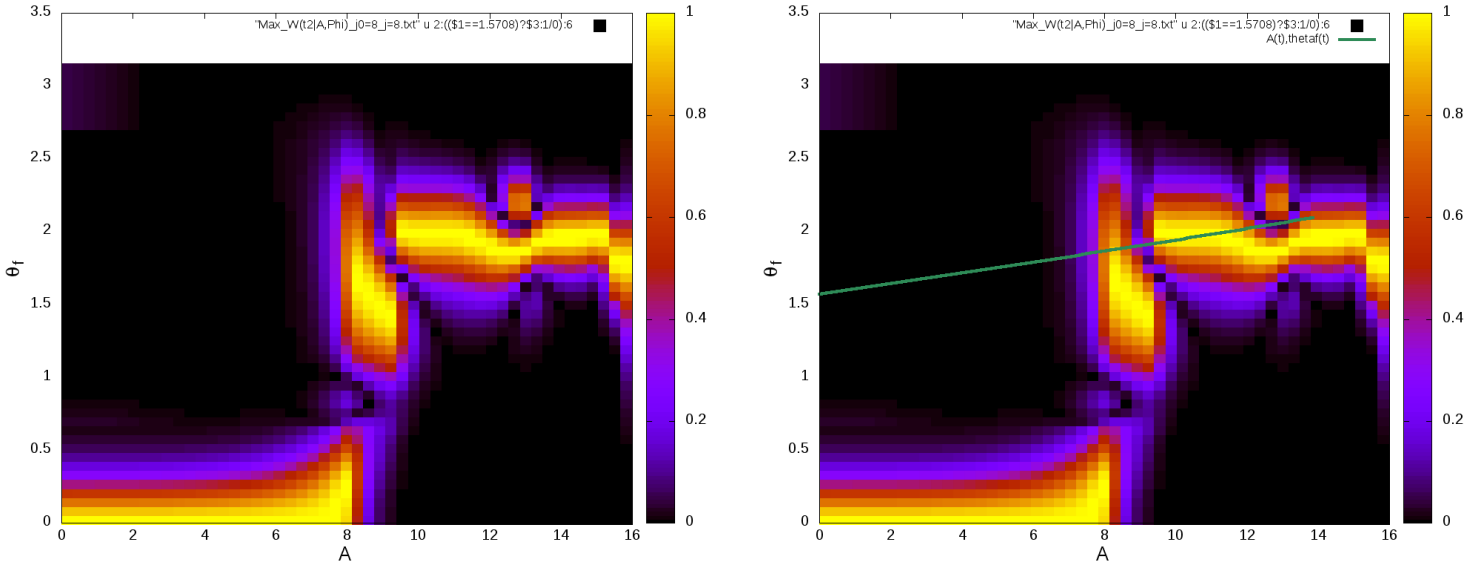


Figure 50: Representation of θ_f values in function of A for $\Phi = \frac{\pi}{2}$; the yellow correspond to the values which maximize $|P_W(\theta_f|A)|$ for the given A. The green line represent the expected solution of $\theta_f^{classical}(A) = \arccos\left(-\frac{1}{\sqrt{3}} \cdot \frac{A}{2a_0}\right)$ from classical geometry. We show the classical solution $\theta_f^{classical}$ fit the quantum results in the region $A \in]9.2376; 13.8564[$.

We notice that the parameter θ_f associated to the shared-tetrahedra and internal geometry evolve in the same way of the value $\theta_f(A) = \arccos\left(-\frac{1}{\sqrt{3}} \cdot \frac{A}{2a_0}\right)$ from the classical geometry. The evolution of θ_f in function of A are approximately the same in the region $A \in]9.2376; 13.8564[$. We see some level for the value, like a quantification of the θ_f, which are around the usual solution of classical geometry except for extreme geometries (as for $A < 9.2376$ or $A > 13.8564$). In the region $A \in]9.2549; 13.86[$, the value of θ_f given by the quantum probabilities are around to the classical solution : this region correspond in the classical interpretation to a assembly between the case where the all tetrahedra are regular (most symmetrical case possible, which contain curvature with $A = \frac{2}{\sqrt{3}}a \approx 9.2376$, $\theta_f = \arccos\left(-\frac{1}{3}\right) \approx 1.910$ and $\varepsilon_f = 2\pi - 3\arccos\left(\frac{1}{4}\right) \approx 2.3288$) and the case where the assembly is extremum hyperbolic (with $A = \sqrt{3}a \approx 13.8564$, $\theta_f = \arccos\left(-\frac{1}{2}\right) = \frac{2\pi}{3}$ and $\varepsilon_f = -\pi$). Maybe it's a clue to say the quantum geometry have the same properties of classical geometry but for "moderate curvature" ($|\varepsilon_f| < \pi$). For $A < 9.2376$ we see a huge divergence between the results and the classical geometry, that difference can be explain by the influence of degenerate geometries (see Figures 33).. For $A > 13.8564$, we have no more classical equivalent for the geometry, the physics is given only by quantum geometry.

A this step we can conclude some properties : the transition amplitude of the full assembly, with the cylindrical constrains on this boundary, choose the geometry of shared-tetrahedra in agreement with cylindrical constraints. We will have internal tetrahedra with a equilateral base f, and isosceles j_0-faces as the classical geometry where the $\theta^N_{f,i} = \theta_f \ \forall N, i$ and $(\phi_{f,3}, \phi_{f,4}) = \left(\frac{2\pi}{3}, \frac{4\pi}{3}\right)$. The evolution of θ_f is approximately the same that classical geometry for the region $A \in \left]\frac{2}{\sqrt{3}}a; \sqrt{3}a\right[$ which classically give curvature, but we have some divergence with the extremum cases and quantum regions.

6.3.5 Numerical result for $P(j_f|A)$

As the previous section, we can try to get the most probabilistic value of j_f chosen by the transition amplitude. Similarly, we have the definition of :

$$P(j_f|A) \equiv \frac{(2j_f+1)\, w_f\left(j, j_0, j_f, A, \frac{\pi}{2}\right)}{\left\langle W|j, j_0, A, \frac{\pi}{2}\right\rangle} \tag{174}$$

which is linked to the transition amplitude of the three 4-simplices from the assembly, with the quantum summation only over the all possible shared-tetrahedra geometries (see the link with (118)). We want to compute it for look what the values of j_f are selected for given A. With the cylindrical symmetries, $\Phi = \frac{\pi}{2}$, the $P(j_f|A)$ are real so we just need to study the norm of it.

If we draw the j_f in function of A with a colored view which indicate if the values j_f maximize $|P(j_f|A)|$ for given A, that give the Figure :

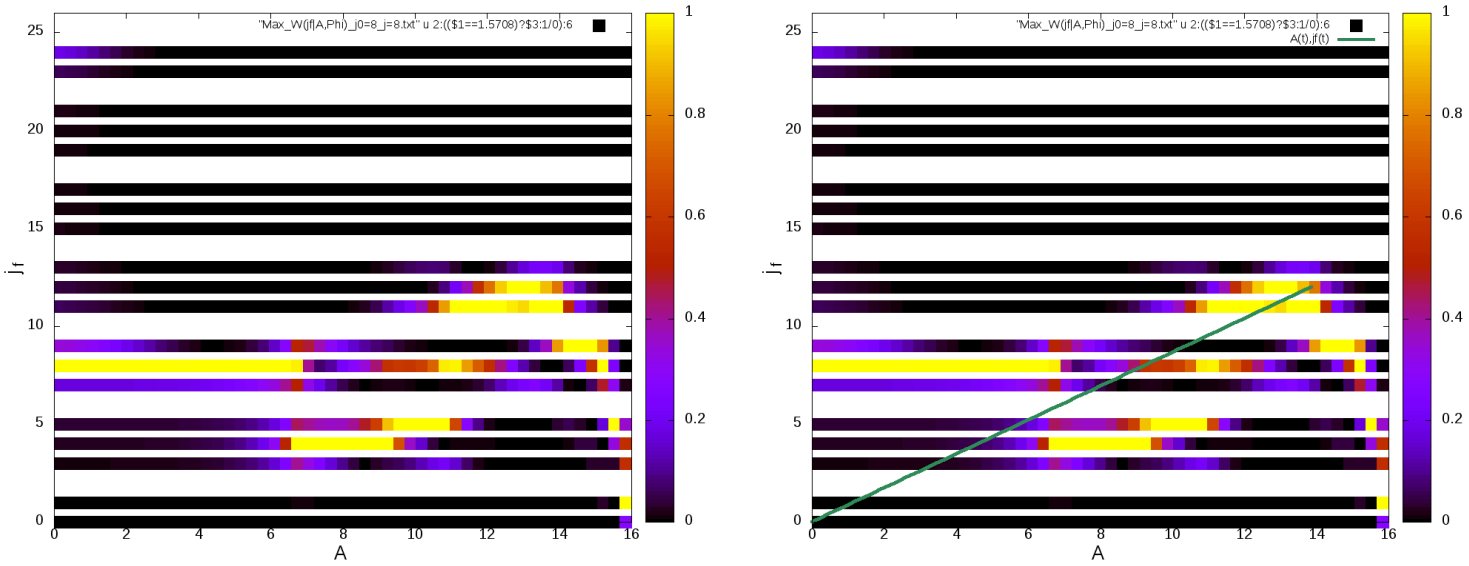


Figure 51: Representation of j_f values in function of A for $\Phi = \frac{\pi}{2}$; the yellow correspond to the values which maximize $|P_W(j_f|A)|$ for the given A. The green line represent the expected solution of $j_f^{classical}(A) = \frac{\sqrt{3}}{2}A$ from classical geometry. We show the classical solution $j_f^{classical}$ fit the quantum results in the region $A \in]10,4745; 13.8564[$.

Here we see a "similar" evolution of the value of j_f and the classical expected value $j_f^{classical}(A) = \frac{\sqrt{3}}{2}A$ from the classical constraint (18) ; the disparity is more important here. The region where the quantum and classical geometry give the same results, is for $A \in]10,4745; 13.8564[$, that more small that previously. The lower bound, $A \approx 10,4745$ probably correspond to the case where the classical Θ_f is equal to $\frac{\pi}{2}$ (for $A = 2\sqrt{\frac{3}{7}}a$ with (21) and (22)); that give the deficit angle $\varepsilon_f = \frac{\pi}{2}$ in the classical equivalent. The region $A \in \left]2\sqrt{\frac{3}{7}}a; \sqrt{3}a\right[$, where the classical and quantum geometry are in agreement, correspond to the curvature region with $\varepsilon_f \in \left]\frac{\pi}{2}; -\pi\right[$. For $A > \sqrt{3}a$, we have no more classical geometry, so the j_f are given only by the quantum geometries. For $A \in \left]6.4314; 2\sqrt{\frac{3}{7}}a\right[$ we have a moderate divergence, which become bigger beyond $A < 6.4314$. We assumes that the differences come from the degenerate geometries, but we don't know the physical meaning of the value $A = 6.4314$.

For summarize, overall the evolution of the quantum and classical geometry is the same in the region where curvature exist classically and is moderate. but we have a important differences for the quantum regions and small values of expected $j_f^{classical}$.

6.3.6 Numerical result for $P(j_f, \theta_f|A)$

For try to get more information about the internal geometries, we will study the probability to have a specific value of j_f and θ_f given the coherent states A and $\Phi = \frac{\pi}{2}$:

$$P(j_f, \theta_f|A) \equiv \frac{(2j_f+1)}{\langle W|j, j_0, A, \frac{\pi}{2}\rangle} \left\langle \pentagon|j, j_0, j_f, A, \frac{\pi}{2}, \overrightarrow{n_{f,i}}[\theta_f], -\overrightarrow{n_{f,i}}[\theta_f] \right\rangle^3 \tag{175}$$

As in the decomposition (118), that linked to the transition amplitude of the three (disjoint) 4-simplex with the same face f and their coherent data given by j,j_0,A and θ_f. Again, if we look just the norm of the probability, and take the couple of values (j_f, θ_f) which maximize the norm for each given A, we have the evolution of j_f and θ_f in function of A :

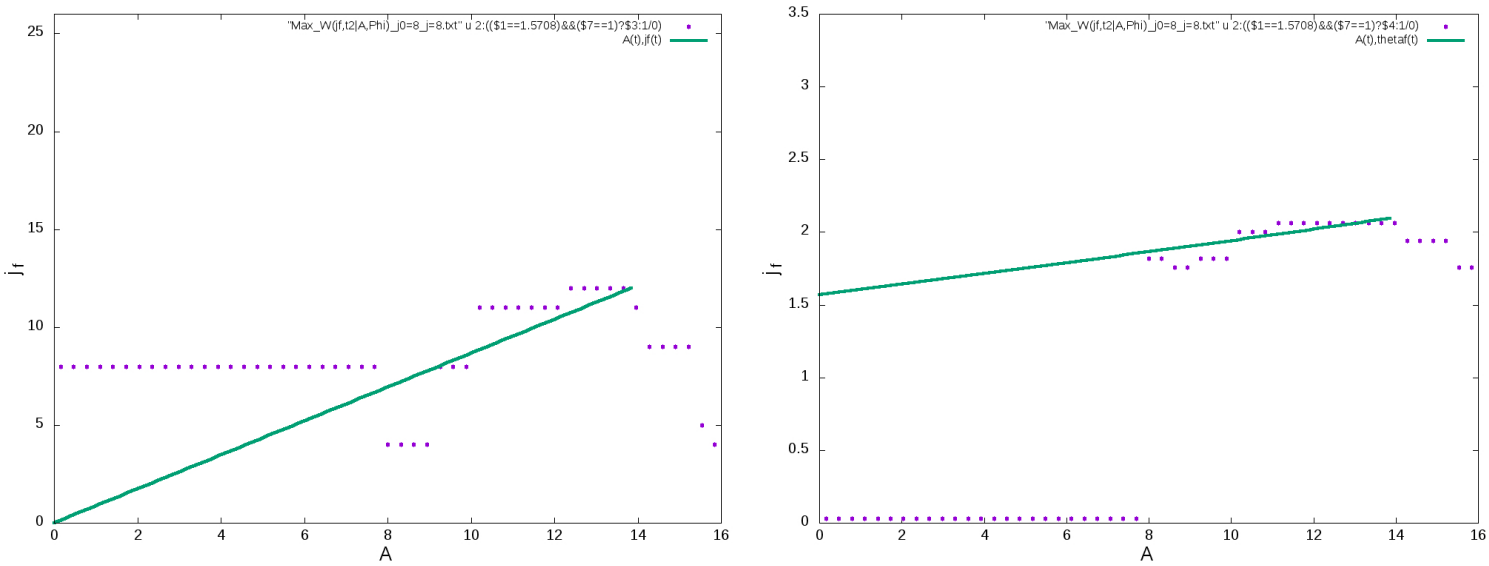


Figure 52: Representation (purple points) of the selected couple j_f (left Figure), θ_f (right Figure) values which maximize $|P(j_f, \theta_f|A)|$ in function of A. The green lines are the expected solutions $(j_f; \theta_f)^{classical} = \left(\frac{\sqrt{3}}{2}A; \arccos\left(-\frac{1}{\sqrt{3}} \cdot \frac{A}{2a_0}\right)\right)$ from classical geometry. We show the classical solutions fit very well, in the both data, the quantum results in the region $A \in]7.6862; 13.8564[$.

We see a important evolution here, because the difference between the computation and the classical expectation over j_f and θ_f have less disparities of before ! The classical case seem emerge from the quantum transition amplitude in the maximum of the probability. The matching region of the classical and quantum geometry is restored for $A \in \left]\sqrt{\frac{12}{13}}a; \sqrt{3}a\right[$, which classically correspond to geometry with moderate curvature. In fact, the value $A = \sqrt{\frac{12}{13}}a$ correspond to the case where the deficit angle, from (21) and (22), is $\varepsilon_f = \pi$; the matching region thus correspond to the curvature interval $\varepsilon_f \in]-\pi; \pi[$. We assume that the degenerate region $A < \sqrt{\frac{12}{13}}a$ is the source of the biggest differences between the quantum and classical geometry for the previous Figures 50,51.

6.4 Conclusion about the results

Overall, we see in the study of the transition amplitude properties that we find classical solutions in the region $A \in [7.69; 13.86]$ (Figure 52). That region correspond, in the classical equivalent, to geometry with curvature $\varepsilon_f \in [-\pi; \pi]$; it's a good indication that the euclidean Loop Quantum Gravity contain curvature. The specific region where the all transition amplitudes are in agreement with the classical geometry is more precisely for $A \in [9.23; 13.86]$ (50,51,52), corresponding to the curvature region $\varepsilon_f \in \left[-\pi; 2\pi - 3\arccos\left(\frac{1}{4}\right)\right]$. Naively, this could be understood as the manifestation that quantum geometry prefer negative curvature. But in presence of some classical solutions with positive curvature, and the fact where the corresponding j_f become bigger and close to the classical expected value with the increasing of A, its probably just a influence of quantum effect of geometry from low value of j_f. Although we have no explicit expressions of the curvature in the context of quantum theory, it is reasonable to assume that the curvature is found in the EPRL model. For the region $A \geq a\sqrt{3} \approx 13.86$, we have no equivalent of classical geometry, so the results come from the pure quantum geometry. The region $A \leq a\sqrt{\frac{12}{13}} \approx 7.69$, equivalent

to $j_f^{classical} < 7$, are just dominated by degenerate geometry and their volumes effects.

7 Conclusion

In this study, we investigated a simple assembly of 4-simplices with classical, Regge, and quantum geometry to identify differences and similarities between these aspects and theories. Our assembly was formed by three 4-simplices, sharing tetrahedra with a unique internal face f, in which the symmetries were devised to simplify the study.

We studied the classical geometry of the assembly and showed that the complete geometry could be restored from the lengths of its segments. We highlighted the bijection between the lengths of the segments, natural variables of geometry, and the areas and shape parameters from the boundary. We show that the assembly possessed curvature around the face f which evolved continuously in function of the lengths or, via the bijection, in function of the boundary parameters.

To investigate whether if the curvature was preserved through the Regge's geometries, we adapted the assembly by cutting it via the split of the face f. The split of f generate new segments where we have Regge's dynamics, while retaining a similar boundary structure to our study object. We then show that the Regge calculus was viable in our object and reproduced the curvature, which evolved in function of the segments length from the boundary.

Based on the Regge calculus that reproduces curvature in the context of our object, it was interesting to see what happened in the case of quantum geometry. The EPRL model dynamics, which is defined on the areas and especially the face f, was perfectly applicable to our assembly without any modification. In the context of quantum geometry, we defined the graphs and boundary states of our assembly. We expressed the transition amplitudes of the 4-simplices and their union to numerically analyze their properties. We therefore developed a C++ code (Annexes B) to compute the various mathematical objects and numerically construct the amplitudes.

Any amplitudes previously calculated for a small scale ($j = 8$) were studied from their norms and phase, as conducted in typical studies of field theory for which "classical" solutions are provided by the stationary points from the amplitudes and their integrants. After identifying these stationary points from transition amplitudes and integrants, we have found some solutions in agreement with classical geometry.

From the 4-simplex amplitude, we identified a great emergence of the classical geometry far away the degenerate region. Moreover, the 4-simplex amplitude well reproduced the semi-classical limit predicted by Barrett [6, 30] with the corresponding Regge's action.

From the full transition amplitude, we found also solutions in agreement with classical geometries that have curvature. Some classical solutions therefore possess parameters in agreement with classical geometry and are likely to have curvature as their classical equivalent. The regions where classical solutions appear were obtained for parameters intervals around to the full regular case and which would give moderate curvature. Thus, in the context of our assembly, the Loop Quantum Gravity theory is in agreement with classical geometry regarding the moderate curvature regions.

In addition to a few expected geometry results, we also highlighted the influence of degenerate geometry states. These degenerate solutions greatly affect certain geometric properties that therefore diverge from classical solutions and yield regions for which quantum and classical geometry differs. These regions correspond to geometries for which the classical equivalents would be very extreme (very elongated tetrahedra, tetrahedra nearly flat and high curvature), which mainly occur in intervals of parameters that typically yield a low associated spin to f. The geometry of these regions is purely quantum and corresponds to a new physics that is not included in classical or Regge geometry. This new physics would therefore be interesting to study, because it opens a door to possible novel insights into physics phenomena for which curvature is high and possibly quantum, such as black holes and the first instants of the universe.

References

[1] T. Regge, "General Relativity Without Coordinates" , Nuovo Cim. 19 (1961) 558-571.

[2] J. Engle, E. Livine, R. Pereira, and C. Rovelli, "LQG vertex with finite Immirzi parameter", Nucl. Phys. B799 (2008) 136149, arXiv:0711.0146.

[3] C. Rovelli and F. Vidotto, "Covariant Loop Quantum Gravity". Cambridge University Press, 2014.

[4] E. Magliaro and C. Perini, "Emergence of gravity from spin-foams", EPL (Europhysics Letters) 95 (aug, 2011) 30007, arXiv:1108.2258.

[5] B. Dittrich and S. Speziale, "Area-angle variables for general relativity", NewJ. Phys. 10 (2008) 083006, arXiv:0802.0864 [gr-qc].

[6] J. W. Barrett, R. J. Dowdall, W. J. Fairbairn, H. Gomes, and F. Hellmann, "Asymptotic analysis of the EPRL four-simplex amplitude", J.Math.Phys. 50 (2009) 112504.

[7] F. Conrady and L. Freidel, "Path integral representation of spin foam models of 4d gravity", Class. Quant. Grav. 25 (2008) 245010, arXiv:0806.4640.

[8] J. Engle, R. Pereira, and C. Rovelli, "The loop-quantum-gravity vertex-amplitude", Phys. Rev. Lett. 99 (2007) 161301, arXiv:0705.2388.

[9] L. Freidel and K. Krasnov, "A New Spin Foam Model for 4d Gravity", Class. Quant. Grav. 25 (2008) 125018, arXiv:0708.1595.

[10] W. Kaminski, M. Kisielowski, and J. Lewandowski, "Spin-Foams for All Loop Quantum Gravity", Class. Quant. Grav. 27 (2010) 95006, arXiv:0909.0939.

[11] J. W. Barrett, R. J. Dowdall, W. J. Fairbairn, F. Hellmann, and R. Pereira, "Lorentzian spin foam amplitudes: graphical calculus and asymptotics", Class. Quant. Grav. 27 (2010) 165009, arXiv:0907.2440.

[12] M. Han, "Cosmological Constant in LQG Vertex Amplitude", Phys.Rev. D84 (2011) 64010, arXiv:1105.2212.

[13] W. J. Fairbairn and C. Meusburger, "Quantum deformation of two four-dimensional spin foam models",J.Math.Phys. 53 (dec, 2010) 45, arXiv:1012.4784.

[14] E. Magliaro and C. Perini, "Regge gravity from spin-foams", International Journal of Modern Physics D 22 (2013) 1350001, arXiv:1105.0216.

[15] M. Han, Covariant Loop "Quantum Gravity, Low Energy Perturbation Theory, and Einstein Gravity", Phys. Rev. D 89 (2014) 124001, arXiv:1308.4063.

[16] H. Haggard, M. Han, W. Kaminski, and A. Riello, "SL(2,C) Chern-Simons Theory, a non-Planar Graph Operator, and 4D Loop Quantum Gravity with a Cosmological Constant: Semiclassical Geometry", arXiv:1412.7546.

[17] E. Magliaro, C. Perini, and C. Rovelli, "Numerical indications on the semi-classical limit of the flipped vertex", arXiv.org gr-qc (2007) .

[18] E. Bianchi and H. M. Haggard, "Discreteness of the volume of space from Bohr-Sommerfeld quantification", arXiv:1102.5439 [gr-qc].

[19] E. Bianchi and H. M. Haggard, "Bohr-Sommerfeld quantification of space", Physical Review D 86 (dec, 2012) 124010, arXiv:1208.2228.

[20] E. R. Livine and S. Speziale, "Physical boundary state for the quantum tetrahedron", Classical and Quantum Gravity 25 (apr, 2008) 085003, arXiv:0711.2455.

[21] E. R. Livine and S. Speziale, "Group Integral Techniques for the Spin-foam Graviton Propagator", JHEP 11 (2006) 92, arXiv:0608131 [gr-qc].

[22] B. Bahr, B. Dittrich, "(Broken) Gauge Symmetries and Constraints in Regge Calculus", Class. Quant. Grav. 26 (2009) 225011, arXiv:0905.1670 [gr-qc].

[23] M. Han, "4-dimensional Spin-foam Model with Quantum Lorentz Group", J. Math. Phys. 52 (2011) 072501, arXiv:1012.4216 [gr-qc].

[24] A. M. Steinberg, "Conditional probabilities in quantum theory, and the tunneling time controversy", Phys. Rev. A52:32-42,1995, arXiv:quant-ph/9502003.

[25] I. G. Bobo, "On Quantum Conditional Probability", 0495-4548 (2013) 28: 76; pp. 115-137

[26] R. De Pietri and C. Rovelli, "Geometry Eigenvalues and Scalar Product from Recoupling Theory in Loop Quantum Gravity", Phys. Rev. D54:2664-2690, 1996, arXiv:gr-qc/9602023.

[27] J. B. Hartle and Z. Perjes, "Solutions of the Regge's equations on some Triangulations of CP^2", J. Math. Phys. 38 (1997) 2577-2586, arXiv:gr-qc/9606005.

[28] P Khavari, "Regge Calculus as a Numerical Approach to General Relativity", 2009.

[29] F. Collet, C. Rovelli, V. Bayle, "Short-scale Emergence of Classical Geometry, in Euclidean Loop Quantum Gravity", arXiv:1603.07931 [gr-qc].

[30] J. W. Barrett, R. M. Williams, "The asymptotics of an amplitude for the 4-simplex", arXiv:gr-qc/9809032

A Tetrahedron geometry

For a classical tetrahedron with this four areas a_i and their face-vectors $\overrightarrow{n_i}$ which respect the closure condition $\sum_i a_i \overrightarrow{n_i} = \overrightarrow{0}$, we express the unit normals $\overrightarrow{n_i}$ in polar coordinates as $\overrightarrow{n_i} = (\theta_i, \phi_i)$, meaning $\overrightarrow{n_i} = (\cos\phi_i \sin\theta_i, \sin\phi_i \sin\theta_i, \cos\theta_i)$. We choose the orientation of the tetrahedron by (gauge) fixing $(\theta_1, \phi_1) = (0, 0)$, $\phi_2 = 0$ and $\phi_3 \in [0, \pi]$, $\phi_4 \in [\pi, 2\pi]$.
By using these relations (and more especially 3,4), straigthforward geometry gives:

$$\theta_1 = 0 \qquad \phi_1 = 0$$
$$\cos\theta_2 = \tfrac{A^2 - a_1^2 - a_2^2}{2a_1 a_2} \qquad \phi_2 = 0$$

$$4a_1 a_3 A^2 \cos\theta_3 = \cos\Phi \sqrt{2A^2\left(a_3^2 + a_4^2\right) - \left(a_3^2 - a_4^2\right)^2 - A^4} \sqrt{2A^2\left(a_1^2 + a_2^2\right) - \left(a_1^2 - a_2^2\right)^2 - A^4} - \left(A^2 + \left(a_3^2 - a_4^2\right)\right)\left(A^2 + \left(a_1^2 - a_2^2\right)\right)$$

$$4a_1 a_4 A^2 \cos\theta_4 = -\cos\Phi \sqrt{2A^2\left(a_4^2 + a_3^2\right) - \left(a_4^2 - a_3^2\right)^2 - A^4} \sqrt{2A^2\left(a_1^2 + a_2^2\right) - \left(a_1^2 - a_2^2\right)^2 - A^4} - \left(A^2 + \left(a_4^2 - a_3^2\right)\right)\left(A^2 + \left(a_1^2 - a_2^2\right)\right)$$

$$\cos\phi_3 = \tfrac{a_4^2 \sin^2\theta_4 - a_2^2 \sin^2\theta_2 - a_3^2 \sin^2\theta_3}{2a_2 a_3 \sin\theta_2 \sin\theta_3}$$

$$\cos\phi_4 = \tfrac{a_3^2 \sin^2\theta_3 - a_2^2 \sin^2\theta_2 - a_4^2 \sin^2\theta_4}{2a_2 a_4 \sin\theta_2 \sin\theta_4}$$

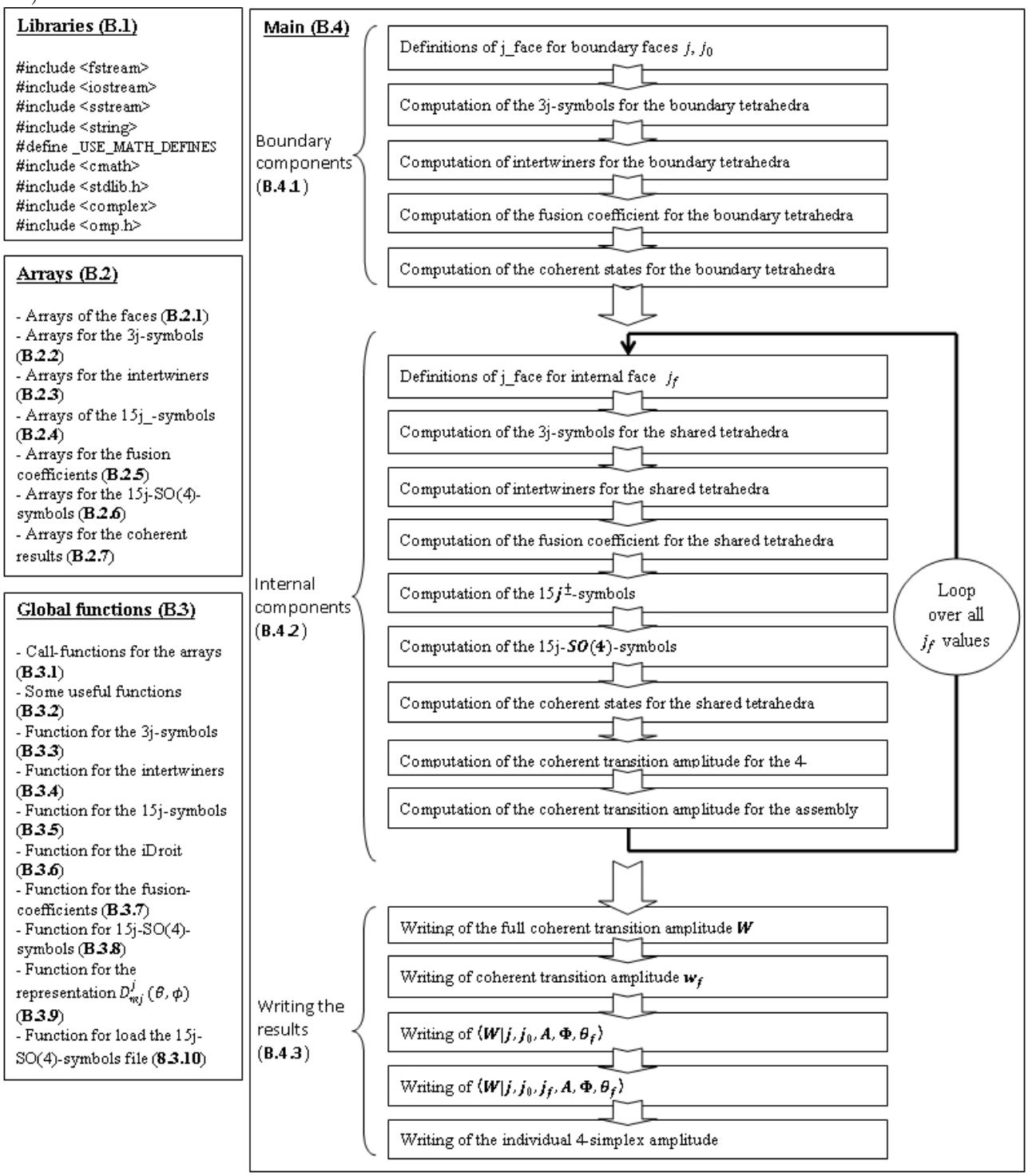
with the shape parameters of the tetrahedron A and Φ.

In the case of cylindrical symmetry ($\Phi = \frac{\pi}{2}$, $a_1 = a_2 = a$, $a_3 = a_4 = a_0$), these relations simplify. For the boundary tetrahedra, we have:

$$\theta_1 = 0 \qquad \phi_1 = 0$$
$$\cos\theta_2 = \tfrac{A^2}{2a^2} - 1 \qquad \phi_2 = 0$$
$$\cos\theta_3 = -\tfrac{A^2}{4aa_0} \qquad \cos\phi_3 = \tfrac{-A\sqrt{4a^2 - A^2}}{\sqrt{16a^2 a_0^2 - A^4}}$$
$$\theta_4 = \theta_3 \qquad \phi_4 = 2\pi - \phi_3$$

B The C++ code

In the optics to present and attest the validity of the results, we will show a brief review of the C++ code and process designed and used for the computation. In this section, we will expose (some) code parts, explain the tricks used and how the code compute the different quantities. We can show a brief sketch (functional organizational chart) of the code as below :



Where the all parts are describe more precisely in the next.

B.1 Library used

The first of all, we need to explicit the libraries that we use :

```
#include <fstream>       //Library for open/close files
                         //(and some reading/writing operations)
#include <iostream>      //Library for read/write stream
#include <sstream>       //Library providing string stream classes
#include <string>        //Library for strings
#define _USE_MATH_DEFINES       //Define some math constants, like pi
#include <cmath>         //Library for use the math-functions : cos, sqrtl...
#include <stdlib.h>      //Library for use unix commands in the code
#include <complex>       //Library for define and use the complex numbers
#include <omp.h>         //OpenMP library for parallelization

using namespace std;     //For more convenience,
                         //allows to not necessarily locally-define the std-functions
```

The stream-libraries are for allow to read/write results in some files and, with the <string> and <cmath>, are the standard libraries in a lot of C++ code. The most important libraries for the computation in itself are the <complex> and the <omp.h>. The complex-library will provide to use complex numbers, that of course necessary for the computation of coherent states and all coherent transition. The omp-library refer to a optimized libraries, the OpenMP library, for use parallelization for the computation, that will be very useful for compute more effectively some big tables of values. Indeed, to store the large tables of values, from the $15j$-symbols for example, we will have a lot of "for"-loops in the code ; the OpenMP library automatically allow to cut the loops in sub-loops which are computed simultaneously over the available processors of the machine when you use the line-code "#pragma omp parallel for".

B.2 Definitions of arrays, tables and links with the math elements

Now, for the all functions can use the intermediate results during the computation process we will define the different arrays and tables. The following arrays and tables will be defined just after the libraries and out of the "main()", that will allow all the tables available for the all code but impose to fix (manually in the code) the maximum size of the corresponding arrays.

```
float j_face[11][3];

long double trois_j_type_t[11][22][61][21];
long double trois_j_plusmoins_t[4][61][61];

long double intertwiner_t[7][22][61][21][21];

long double function_15j_plus_t[22][22][22][22][22];
long double function_15j_moins_t[22][22][22][22][22];

long double iDroit_type1_t[22][22][21][21][21];
long double iDroit_type2_t[22][22][61][21][21];

long double iCourbe_type1_t[22][22][22];
long double iCourbe_type2_t[22][22][22];
```

```
long double function_15j_so4_t[22][22][22][22][22];

long double t15j_so4_loaded[22][22][22][22][22];
bool t15j_so4_present[22][22][22][22][22];

complex<long double> so4semicoherent[51][51][22][22];
complex<long double> so4coherent[31][51][51][51][51];
complex<long double> w_coherent_f[31][51][51];
```

The indices of tables and arrays, in accordance with the represented math objects, will depend of the intertwiner representations (the J and K parameters) and the magnetic indices (the m parameters). Because of the symmetries of the math objects, the J, K parameters will be always positive integer so we can use them directly as array indices. But the magnetic indices m can be integer or half-integer (same type as their corresponding j-representation) and can be negative, so for the array indices we will use the trick to replace the m by the $\mathrm{m} = m + j = 0, 1, 2, \ldots, 2j+1$. With these indices, the maximum size of the array for a given j is for the J, K and m parameters $[2j+1]$; that means for a max scale with $j = 10$ we have the size for the arrays equal to [21] (except for the m indices linked to $j_f \in [0; 3j_0]$ which need to have the size $[2j_f^{max}+1]$=[61]). For more visibility, the tables and arrays are defined with the size [21] for the (usual) m indices, the size [61] for m indices from j_f,$j_f^{\pm}$, and the size [22] for the J, K indices. The others size are given by the indices which allow to select the type of math objects we want, we will specify that in the following. For the coherent-objects tables, the size come from the arbitrary resolution selected for : we take a resolution with 51 points for the coherent variables ; of course these objects will be defined and presented in their respective subsections.

B.2.1 Arrays of the faces

The first of all is the table of the j-representations for the faces : j_face[][]. This table will contain the values of the corresponding j for the intertwiners. The element j_face[n][p] of the table will have the j-values of the face number "p" from the 3j-symbol "n" :

$$\text{3j-symbol of type "n":} \quad \begin{pmatrix} \text{j\_face[n][1]} & \text{j\_face[n][2]} & J \\ m_1 & m_2 & M \end{pmatrix} \tag{176}$$

Because, because of symmetries of our object and our will to save computing time, all the future intertwiners can be written with a type of intertwiner "n" and their corresponding face "p". The corresponding mapping between the real math objects and the n,p variables for identify the faces is the following :

type of 3j-symbol (n)	3j-symbol	$\longrightarrow$	associated j_face[n][p]
1	$\begin{pmatrix} j & j & J \\ m_1 & m_2 & M \end{pmatrix}$	$\longrightarrow$	$\begin{cases} \text{j\_face[1][1]} = j \\ \text{j\_face[1][2]} = j \end{cases}$
2	$\begin{pmatrix} j^+ & j^+ & J^+ \\ m_1^+ & m_2^+ & M^+ \end{pmatrix}$	$\longrightarrow$	$\begin{cases} \text{j\_face[2][1]} = j^+ \\ \text{j\_face[2][2]} = j^+ \end{cases}$
3	$\begin{pmatrix} j^- & j^- & J^- \\ m_1^- & m_2^- & M^- \end{pmatrix}$	$\longrightarrow$	$\begin{cases} \text{j\_face[3][1]} = j^- \\ \text{j\_face[3][2]} = j^- \end{cases}$
4	$\begin{pmatrix} j_0 & j_0 & J \\ m_1 & m_2 & M \end{pmatrix}$	$\longrightarrow$	$\begin{cases} \text{j\_face[4][1]} = j_0 \\ \text{j\_face[4][2]} = j_0 \end{cases}$
5	$\begin{pmatrix} j_0^+ & j_0^+ & J^+ \\ m_1^+ & m_2^+ & M^+ \end{pmatrix}$	$\longrightarrow$	$\begin{cases} \text{j\_face[5][1]} = j_0^+ \\ \text{j\_face[5][2]} = j_0^+ \end{cases}$
6	$\begin{pmatrix} j_0^- & j_0^- & J^- \\ m_1^- & m_2^- & M^- \end{pmatrix}$	$\longrightarrow$	$\begin{cases} \text{j\_face[6][1]} = j_0^- \\ \text{j\_face[6][2]} = j_0^- \end{cases}$
7	$\begin{pmatrix} j_f & j_0 & J \\ m_1 & m_2 & M \end{pmatrix}$	$\longrightarrow$	$\begin{cases} \text{j\_face[7][1]} = j_f \\ \text{j\_face[7][2]} = j_0 \end{cases}$
8	$\begin{pmatrix} j_f^+ & j_0^+ & J^+ \\ m_1^+ & m_2^+ & M^+ \end{pmatrix}$	$\longrightarrow$	$\begin{cases} \text{j\_face[8][1]} = j_f^+ \\ \text{j\_face[8][2]} = j_0^+ \end{cases}$
9	$\begin{pmatrix} j_f^- & j_0^- & J^- \\ m_1^- & m_2^- & M^- \end{pmatrix}$	$\longrightarrow$	$\begin{cases} \text{j\_face[9][1]} = j_f^- \\ \text{j\_face[9][2]} = j_0^- \end{cases}$

B.2.2 Arrays for the 3j-symbols

After the definition of the j_face[][], we have the corresponding definition of the tables for the 3j-symbols. For the table trois_j_type_t[][][][][], the element trois_j_type_t[n][J][ma][mb] will contain the value of the corresponding 3j-symbols "n" :

$$\text{trois\_j\_type\_t[n][J][ma][mb]} = \begin{pmatrix} \text{j\_face[n][1]} & \text{j\_face[n][2]} & \text{J} \\ m_1 & m_2 & M \end{pmatrix} \; ; \; \begin{matrix} m_1 = \text{ma-j\_face[n][1]} \\ m_2 = \text{mb-j\_face[n][2]} \\ M = -m_1 - m_2 \end{matrix} \tag{177}$$

Next we have the table definition of the specific 3j-symbol : trois_j_plusmoins_t[][][]. It correspond to the 3j-symbol for the fusion coefficients that link the $j^+ \times j^-$ with their corresponding j, so we will use a specific notation for designate these $\{j^+ j^- j\}$-symbols. Similarly to the other 3j-symbols, we will use a "n" (valid only in the context of this table) for label the corresponding $\{j^+ j^- j\}$-symbols in the table. We have the mapping :

type of $\{j^+ j^- j\}$-symbols (n)	trois_j_plusmoins_t[n][ma][mb]
1	$\text{trois\_j\_plusmoins\_t[1][ma][mb]} = \begin{pmatrix} j^+ & j^- & j \\ \text{ma} - j^+ & \text{mb} - j^- & j - \text{ma} - \text{mb} \end{pmatrix}$
2	$\text{trois\_j\_plusmoins\_t[2][ma][mb]} = \begin{pmatrix} j_0^+ & j_0^- & j_0 \\ \text{ma} - j_0^+ & \text{mb} - j_0^- & j_0 - \text{ma} - \text{mb} \end{pmatrix}$
3	$\text{trois\_j\_plusmoins\_t[3][ma][mb]} = \begin{pmatrix} j_f^+ & j_f^- & j_f \\ \text{ma} - j_f^+ & \text{mb} - j_f^- & j_f - \text{ma} - \text{mb} \end{pmatrix}$

The trois_j_type_t[][][][][] and trois_j_plusmoins_t[][][][] tables contain the all values of the all 3j-symbols we need for built the all intertwiners and next math objects necessary.

B.2.3 Arrays for the intertwiners

The next defined table, intertwiner_t[][][][][][], will contain the all values of the all intertwiners needed. In the same way, we have a number "n" for specify the type of the intertwiner you consider, and the other parameters give the corresponding values :

type of intertwiner (n)	intertwiner_t[n][J][ma][mb][mc]
1	intertwiner_t[1][J][ma][mb][mc] $= i^{\mathrm{J}}_{(\mathrm{ma}-j)(\mathrm{mb}-j)(\mathrm{mc}-j_0)(2j+j_0-\mathrm{ma}-\mathrm{mb}-\mathrm{mc})}\,(j,j,j_0,j_0)$
2	intertwiner_t[2][J][ma][mb][mc] $= i^{\mathrm{J}}_{(\mathrm{ma}-j^+)(\mathrm{mb}-j^+)\left(\mathrm{mc}-j_0^+\right)\left(2j^++j_0^+-\mathrm{ma}-\mathrm{mb}-\mathrm{mc}\right)}\left(j^+,j^+,j_0^+,j_0^+\right)$
3	intertwiner_t[3][J][ma][mb][mc] $= i^{\mathrm{J}}_{(\mathrm{ma}-j^-)(\mathrm{mb}-j^-)\left(\mathrm{mc}-j_0^-\right)\left(2j^-+j_0^--\mathrm{ma}-\mathrm{mb}-\mathrm{mc}\right)}\left(j^-,j^-,j_0^-,j_0^-\right)$
4	intertwiner_t[4][J][ma][mb][mc] $= i^{\mathrm{J}}_{(\mathrm{ma}-j_f)(\mathrm{mb}-j_0)(\mathrm{mc}-j_0)(j_f+2j_0-\mathrm{ma}-\mathrm{mb}-\mathrm{mc})}\,(j_f,j_0,j_0,j_0)$
5	intertwiner_t[5][J][ma][mb][mc] $= i^{\mathrm{J}}_{\left(\mathrm{ma}-j_f^+\right)\left(\mathrm{mb}-j_0^+\right)\left(\mathrm{mc}-j_0^+\right)\left(j_f^++2j_0^+-\mathrm{ma}-\mathrm{mb}-\mathrm{mc}\right)}\left(j_f^+,j_0^+,j_0^+,j_0^+\right)$
6	intertwiner_t[6][J][ma][mb][mc] $= i^{\mathrm{J}}_{\left(\mathrm{ma}-j_f^-\right)\left(\mathrm{mb}-j_0^-\right)\left(\mathrm{mc}-j_0^-\right)\left(j_f^-+2j_0^--\mathrm{ma}-\mathrm{mb}-\mathrm{mc}\right)}\left(j_f^-,j_0^-,j_0^-,j_0^-\right)$

With the definition of intertwiners :

$$i^J_{m_1m_2m_3m_4}\left(j_1,j_2,j_3,j_4\right)=\sqrt{2J+1}\sum_M(-1)^{J-M}\begin{pmatrix} j_1 & j_2 & J\\ m_1 & m_2 & M\end{pmatrix}\begin{pmatrix} j_3 & j_4 & J\\ m_3 & m_4 & -M\end{pmatrix} \tag{178}$$

Inside the definition of the intertwiners, we can see the contribution of the 3j-symbol previously presented ; of course, the method for compute the value of the intertwiners table in function of the 3j-symbols table will be given later via the associated code.

B.2.4 Arrays of the $15j^{\pm}$-symbols

Next, we have the $15j^+$-symbols and $15j^-$-symbols table. At this point, the j-representation are implicitly given in the definition of the objects : the table will only depend of the intertwiners parameter J. So we have just the map :

$$\begin{array}{l}\text{fonction\_15j\_plus\_t[J1p][J2p][J3p][J4p][J5p]} = \left(\text{J1p,J2p,J3p,J4p,J5p};j_{kl}^+\right) = \\ \sum_p(-1)^{\sum_{kl}\left(j_{kl}^+-p_{kl}\right)}i^{\mathrm{J1p}}_{-p_{12}p_{13}-p_{14}p_{15}}i^{\mathrm{J2p}}_{-p_{23}p_{12}-p_{24}p_{25}}i^{\mathrm{J3p}}_{-p_{13}p_{23}-p_{34}p_{35}}i^{\mathrm{J4p}}_{p_{45}p_{24}p_{34}p_{14}}i^{\mathrm{J5p}}_{-p_{45}-p_{25}-p_{35}-p_{15}}\left(j_{kl}^+\right)\end{array} \tag{179}$$

$$\begin{array}{l}\text{fonction\_15j\_moins\_t[J1m][J2m][J3m][J4m][J5m]} = \left(\text{J1m,J2m,J3m,J4m,J5m};j_{kl}^-\right) = \\ \sum_p(-1)^{\sum_{kl}\left(j_{kl}^--p_{kl}\right)}i^{\mathrm{J1m}}_{-p_{12}p_{13}-p_{14}p_{15}}i^{\mathrm{J2m}}_{-p_{23}p_{12}-p_{24}p_{25}}i^{\mathrm{J3m}}_{-p_{13}p_{23}-p_{34}p_{35}}i^{\mathrm{J4m}}_{p_{45}p_{24}p_{34}p_{14}}i^{\mathrm{J5m}}_{-p_{45}-p_{25}-p_{35}-p_{15}}\left(j_{kl}^-\right)\end{array} \tag{180}$$

B.2.5 Arrays for the fusion coefficients

We will have the definition of a table for construct the future fusion coefficients :

$$\begin{array}{l}\text{iDroit\_type1\_t[Jp][Jm][ma][mb][mc]} = \\ \sum_{m^+,m^-}i^{\mathrm{Jp}}_{m_1^+m_2^+m_3^+m_4^+}\left(j^+,j^+,j_0^+,j_0^+\right)i^{\mathrm{Jm}}_{m_1^-m_2^-m_3^-m_4^-}\left(j^-,j^-,j_0^-,j_0^-\right)\prod_{l=1}^4\sqrt{2j_l+1}\begin{pmatrix} j_l^+ & j_l^- & j_l\\ m_l^+ & m_l^- & m_l\end{pmatrix} \\ \begin{pmatrix}\text{with :} & \begin{array}{c} m_1=\mathrm{ma}-j\\ m_2=\mathrm{mb}-j\\ m_3=\mathrm{mc}-j_0\\ \left(\ m_4=-\sum_{i=1}^3m_i\ \right)\end{array}\end{pmatrix}\end{array} \tag{181}$$

$$\begin{array}{l}\text{iDroit\_type2\_t[Jp][Jm][ma][mb][mc]} = \\ \sum_{m^+,m^-}i^{\mathrm{Jp}}_{m_1^+m_2^+m_3^+m_4^+}\left(j_f^+,j_0^+,j_0^+,j_0^+\right)i^{\mathrm{Jm}}_{m_1^-m_2^-m_3^-m_4^-}\left(j_f^-,j_0^-,j_0^-,j_0^-\right)\prod_{l=1}^4\sqrt{2j_l+1}\begin{pmatrix} j_l^+ & j_l^- & j_l\\ m_l^+ & m_l^- & m_l\end{pmatrix} \\ \begin{pmatrix}\text{with} & \begin{array}{c} m_1=\mathrm{ma}-j_f\\ m_2=\mathrm{mb}-j_0\\ m_3=\mathrm{mc}-j_0\\ \left(\ m_4=-\sum_{i=1}^3m_i\ \right)\end{array}\end{pmatrix}\end{array} \tag{182}$$

where the intertwiners and the $\{j^+j^-j\}$-symbols appear. And, obviously, a table for the fusion coefficients :

$$\text{iCourbe\_type1\_t[Jp][Jm][J]} = \mathcal{I}^{\mathrm{J}}_{\mathrm{Jp},\mathrm{Jm}}\left(j,j,j_0,j_0\right) \tag{183}$$

$$\text{iCourbe\_type2\_t[Jp][Jm][J]} = \mathcal{I}^{\mathrm{J}}_{\mathrm{Jp,Jm}}\left(j_f, j_0, j_0, j_0\right) \tag{184}$$

B.2.6 Arrays for the 15j-$SO(4)$-symbols

Finally, we have the table for the last important computation : the function_15j_so4_t[][][][][][] table to store the values of the 15j-$SO(4)$-symbols :

$$\begin{array}{c}\text{fonction\_15j\_so4\_t[J1][J2][J3][J4][J5]} = [\text{J1,J2,J3,J4,J5}; j_{kl}] = \\ \sum_{K^+,K^-}\left(K_k^+; j_{kl}^+\right)\left(K_k^-; j_{kl}^-\right)\mathcal{I}^{\mathrm{J1}}_{K_k^+,K_k^-}\left(j,j,j_0,j_0\right)\mathcal{I}^{\mathrm{J2}}_{K_2^+,K_2^-}\left(j,j,j_0,j_0\right)\mathcal{I}^{\mathrm{J3}}_{K_3^+,K_3^-}\left(j,j,j_0,j_0\right) \\ \times\mathcal{I}^{\mathrm{J4}}_{K_4^+,K_4^-}\left(j_f,j_0,j_0,j_0\right)\mathcal{I}^{\mathrm{J5}}_{K_5^+,K_5^-}\left(j_f,j_0,j_0,j_0\right)\end{array} \tag{185}$$

The two next table correspond to a temporary storage for load in memory previous values of 15j-$SO(4)$-symbols computed. Because the time part for compute the 15j-$SO(4)$-symbols is very long, we have a part of the code for write the symbols computed, and load the values already given. The t15j_so4_loaded[][][][][][] will correspond to the temporary storage in memory of the read values, and the t15j_so4_present[][][][][][] to a Boolean table for check if the values are already read/computed or not.

B.2.7 Arrays for the coherent results

Finally, the last three (complex) tables correspond to the coherent results, where :

$$\begin{aligned}\text{so4semicoherent[n][t][J4][J5]} &= \sum_{J_1,J_2,J_3}[J_i, \text{J4,J5}; j, j_0, j_f]\prod_{k=1}^{3}\left\langle i^{J_k}|j, j_0, A\left[\mathrm{n}\right], \Phi\left[\mathrm{t}\right]\right\rangle \\ \text{so4coherent[(int)}(j_f)\text{][n][t][n2][t2]} &= \sum_{J,K}[J_k, K_4, K_5; j, j_0, j_f]\left\langle i^{K_4}|j_i^N\overrightarrow{n_i^N}\left[A_f\left[\mathrm{n2}\right], \Phi_f\left[\mathrm{t2}\right]\right]\right\rangle \\ &\quad\times\overline{\left\langle i^{K_5}|j_i^{N'}\overrightarrow{n_i^{N'}}\left[A_f\left[\mathrm{n2}\right], \Phi_f\left[\mathrm{t2}\right]\right]\right\rangle}\prod_{k=1}^{3}\left\langle i^{J_k^N}|j, j_0, A\left[\mathrm{n}\right], \Phi\left[\mathrm{t}\right]\right\rangle \\ \text{w\_coheren\_f[(int)}(j_f)\text{][n][t]} &= w_f\left(j, j_0, j_f, A\left[\mathrm{n}\right], \Phi\left[\mathrm{t}\right]\right)\end{aligned}$$

That will give all the raw data we need to compute all the probabilities and study the properties of transitions amplitudes. The exact process for the computation of these quantities and the parameters “n”, “t”, “n2”, “t2” associated to the coherent states will be defined in their section of code respectively.

B.3 Definitions of global functions

Before to process to the main of the code, we need to define all the global functions we will use. The all functions necessary to compute and store the tables will be expressed and (briefly) explained below.

B.3.1 Call-functions for the arrays and tables

In the context of the code development, we have defined some functions for call the tables. The interest of this call-functions is just to give the corresponding value of the tables, in this sens the call-functions are no more necessary because you can just take the value from the tables directly. But if you want check the number of time the tables are called, or if you want add some intermediate calculation when you call a value of a table without affect the table, the user can add some code in the call-functions. That is the old interest to define these functions : look if the code compute exactly the good value and check if the process of the code is going well.

```
////////////////////////////////-Call of the tables-////////////////////////////////

long double trois_j_plusmoins(int type, int ma, int mb) {
```

```
// The type here correspond to one of these 3j-symbols :
        // 1-> (j+,j-,j), 2 -> (j0+,j0-,j0) et 3 -> (jf+,jf-,jf)
  return trois_j_plusmoins_t[type][ma][mb];
}

long double trois_j_type(int type, int K, int ma, int mb) {
// The type here correspond to one of the 9 cases of 3j-symbols
  return trois_j_type_t[type][K][ma][mb];
}

long double intertwiner_mem(int typeInter, int K, int ma, int mb, int mc){
// The typeInter here correspond to one of the 6 cases of intertwiners
  return intertwiner_t[typeInter][K][ma][mb][mc];
}

long double function_15j_moins_mem(int K1, int K2, int K3, int K4, int K5) {
  return function_15j_moins_t[K1][K2][K3][K4][K5];
}

long double function_15j_plus_mem(int K1, int K2, int K3, int K4, int K5) {
  return function_15j_plus_t[K1][K2][K3][K4][K5];
}

long double iDroit_type1_mem(int kplus,int kmoins, int ma, int mb, int mc) {
  return iDroit_type1_t[kplus][kmoins][ma][mb][mc];
}

long double iDroit_type2_mem(int kplus,int kmoins, int ma, int mb, int mc) {
  return iDroit_type2_t[kplus][kmoins][ma][mb][mc];
}

long double iCourbe_type1_mem(int kplus,int kmoins, int k) {
  return iCourbe_type1_t[kplus][kmoins][k];
}

long double iCourbe_type2_mem(int kplus,int kmoins, int k) {
  return iCourbe_type2_t[kplus][kmoins][k];
}

long double function_15j_so4_mem(int K1,int K2, int K3, int K4, int K5){
  return function_15j_so4_t[K1][K2][K3][K4][K5];
}
```

B.3.2 Some useful functions

Of course, for the next functions and many parts of the code, that will be useful to define some basic functions, as the maximum, minimum and factorial functions.

```
float max(float e,float f)       ///Maximum function///
   {
   if(e>=f)
```

```
      {return e;
      }
   else
      {return f;
      }
   }

float min(float e,float f)       ///Minimum function///
   {
   if(e<=f)
      {return e;
      }
   else
      {return f;
      }
   }

long double fact(long double j) ///Factorial function///
  {
  if ( (j<0)||((j-(long int)(j))!=0) )
    {cout <<  " problem" << j << endl;
    }
  if ((j==1)||(j==0))
    {return 1;
  }else{
    return fact(j-1)*j;
    }
  }
```

The factorial here, for more convenience with the precision from the rest of the code, is define with numbers which have long double precision. For check if no errors occur with the used values in the factorial, because whatever the precision of the number it must be physically a integer, we have a line where the code check if the number can be expressed as a integer. If not, the code give the error message "problem", that give the information to the user that some values for the factorial from the code are not consistent : that means we have a error in the code. Of course, in the all simulations done, this error message never appear : that means the values used/computed are right and (probably) this check-line is no more useful.

For the future definition of the j^+, with the constraint $j^+ \approx \frac{1+\gamma}{2} j$, we need to introduce a function which give the integer or half-integer closest to the exact value $\frac{1+\gamma}{2} j$. This function is the following :

```
float jp_approx(float Immirzi,float jf,float j_type) //Approximation function for j+
   {
   float jfp_theorique=0.5*(int)((1.0+Immirzi)*jf);
   float jfm_theorique=jf-jfp_theorique;
   if(fabs(jfp_theorique*(1.0-Immirzi)-jfm_theorique*(1.0+Immirzi))
     <fabs((jfp_theorique+0.5)*(1.0-Immirzi)-(jfm_theorique-0.5)*(1.0+Immirzi)))
      {return jfp_theorique;
      }
   if(fabs(jfp_theorique*(1.0-Immirzi)-jfm_theorique*(1.0+Immirzi))
    ==fabs((jfp_theorique+0.5)*(1.0-Immirzi)-(jfm_theorique-0.5)*(1.0+Immirzi)))
      {if((jfp_theorique-(int)(jfp_theorique))==(j_type-(int)(j_type)))
         {return jfp_theorique;
         }
```

```
        else
           {return jfp_theorique+0.5;
           }
        }
    if(fabs(jfp_theorique*(1.0-Immirzi)-jfm_theorique*(1.0+Immirzi))
      >fabs((jfp_theorique+0.5)*(1.0-Immirzi)-(jfm_theorique-0.5)*(1.0+Immirzi)))
        {return jfp_theorique+0.5;
        }
    }
```

So the jp_approx(γ,j_f,j_{type}) return the integer or half-integer closest to $\frac{1+\gamma}{2}j_f$. If the exact value $\frac{1+\gamma}{2}j_f$ is in the middle of the closest integer and half-integer, the function return the closest value which have the same type (integer or half-integer) of the arbitrary j_{type}.

B.3.3 Function for the 3j-symbols

We have the function for a 3j-symbol given by :

```
long double trois_j_sans_m3(float j1, float j2, float j3, float m1, float m2) {
  long double norme=0 ;
  long double somme=0 ;

  if (((j1+j2+j3)==(int)(j1+j2+j3))&&(abs(m1+m2)<=j3) ){
  norme=( pow(-1.0, (int)(j1-j2+m1+m2) )
       *sqrtl(fact(j1+j2-j3)*fact(j1+j3-j2)*fact(j2+j3-j1)/(fact(j1+j2+j3+1.0)))
       *sqrtl(fact(j1+m1)*fact(j1-m1))
       *sqrtl(fact(j2+m2)*fact(j2-m2))
       *sqrtl(fact(j3-m1-m2)*fact(j3+m1+m2) ) );

    for (int k=0; k < (int)(j1+j2-j3+1.0); k++ ) {
      if (((j1-m1-k)>=0)&&((j2+m2-k)>=0)&&((j3-j2+m1+k)>=0)&&((j3-j1-m2+k)>=0)){
        somme += pow(-1.0, k)/(fact(k)*fact(j1+j2-j3-k)*fact(j1-m1-k)*fact(j2+m2-k)
                 *fact(j3-j2+m1+k)*fact(j3-j1-m2+k) );
      }
    }
  }
  return norme*somme;
}
```

Which compute the $\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & -m_1-m_2 \end{pmatrix}$ symbol in agreement with its constraint :

$$j_1 + j2 + j3 \in \mathbb{N}\ ,\ |m_1 + m_2| \leq j_3 \tag{186}$$

The formula used for the computation of that is the Racah formula :

$$\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & -m_1-m_2 \end{pmatrix} = (-1)^{j_1-j_2+m1+m2}\left[\frac{(j1+j2-j3)!\,(j1+j3-j2)!\,(j2+j3-j1)!}{(j1+j2+j3+1)!}\right]^{\frac{1}{2}} \tag{187}$$

$$\times\left[(j_1+m_1)!\,(j_1-m_1)!\,(j_2+m_2)!\,(j_2-m_2)!\,(j_3+m_3)!\,(j_3-m_3)!\right]^{\frac{1}{2}} \tag{188}$$

$$\times\sum_k \frac{(-1)^k}{k!\,(j1+j2-j3-k)!\,(j1-m1-k)!\,(j2+m2-k)!\,(j3-j2+m1+k)!\,(j3-j1-m2+k)!} \tag{189}$$

B.3.4 Function for the intertwiners

We have the function for compute a intertwiner :

```
long double intertwiner(int type1, int type2, int K, int ma, int mb, int mc) {
  long double inter_temp=0;

  float j1=j_face[type1][1];
  float j2=j_face[type1][2];
  float j3=j_face[type2][1];
  float j4=j_face[type2][2];

  if ( abs(j1+j2+j3-ma-mb-mc) <= j4 ) {
  inter_temp = pow(-1, (int)(K+ma+mb-j1-j2) )*sqrtl(2.0*K+1.0)
               *trois_j_type(type1,K,ma,mb)
               *trois_j_type(type2,K,mc,(int)(j1+j2+j3+j4-ma-mb-mc));
  }

  return inter_temp;
}
```

Where we specify the type1 and type2 for choose the correct 3j-symbols, and their associated j-parameters, in there corresponding tables. The code return explicitly :

$$\begin{array}{l} \text{intertwiner(type1,type2,K,ma,mb,mc)} \\ = \sqrt{2K+1}\,(-1)^{\text{K+ma+mb}-j_1-j_2} \\ \quad \times \text{trois\_j\_type\_t[type1][K][ma][mb]} \\ \quad \times \text{trois\_j\_type\_t[type2][K][mc][(int)(j1+j2+j3+j4-ma-mb-mc)]} \\ = i^{\text{K}}_{\text{(ma-}j_1\text{)(mb-}j_2\text{)(mc-}j_3\text{)(}\sum j_i\text{-ma-mb-mc)}}\,(j_1,j_2,j_3,j_4) \end{array} \quad ; \quad \begin{array}{l} j_1 = \text{j\_face[type1][1]} \\ j_2 = \text{j\_face[type1][2]} \\ j_3 = \text{j\_face[type2][1]} \\ j_4 = \text{j\_face[type2][2]} \end{array} \tag{190}$$

B.3.5 Function for the $15j^{\pm}$-symbols

For the function of $15j^{\pm}$-symbols, we have the same sort of code. For the $15j^{+}$-symbols, we have the following function :

```
long double function_15j_plus(int K1, int K2, int K3, int K4, int K5) {

  float j12=j_face[2][1];
  float j13=j_face[2][2];
  float j14=j_face[5][1];
  float j15=j_face[5][2];
  float j23=j_face[2][1];
  float j24=j_face[5][1];
  float j25=j_face[5][2];
  float j34=j_face[5][1];
  float j35=j_face[5][2];
  float j45=j_face[8][1];

  int A=(int)(j14+j24+j34+j45);
  int B=(int)(j45-j14-j24-j34);
  int C=(int)(j12-j13+j14+j15);
  int D=(int)(j23-j12+j24+j25);
```

```
int E=(int)(j13-j23+j34+j35);

int deuxj12=(int)(2*j12);
int deuxj13=(int)(2*j13);
int deuxj14=(int)(2*j14);
int deuxj15=(int)(2*j15);
int deuxj23=(int)(2*j23);
int deuxj24=(int)(2*j24);
int deuxj25=(int)(2*j25);
int deuxj34=(int)(2*j34);
int deuxj45=(int)(2*j45);
int signe=pow(-1.0,(int)(deuxj12+deuxj13+deuxj14+deuxj23+deuxj24+deuxj34
                         + j15+j25+j35+j45));
int signe1=1;
int signe2=1;
int signe3=1;
int signe4=1;
int signe5=1;
int signe6=1;
long double temp=0;

for (int mab=0; mab<=deuxj12; mab++) {
  int deuxj12moinsmab=deuxj12-mab;
  signe2=signe1;
  for (int mac=0; mac<=deuxj13; mac++) {
    signe3=signe2;
    for (int mad=0; mad<=deuxj14; mad++) {
      signe4=signe3;

      long double intertwin1
                =intertwiner_mem(2,K1,deuxj12moinsmab,mac,deuxj14-mad);

      if((intertwin1!=0) && (C-mab+mac-mad>=0) && (C-mab+mac-mad<=deuxj15)){
        for (int mbc=0; mbc<=deuxj23; mbc++) {
          signe5=signe4;
          for (int mbd=0; mbd<=deuxj24; mbd++) {
            signe6=signe5;

            long double intertwin2
                       =intertwiner_mem(2,K2,deuxj23-mbc,mab,deuxj24-mbd)
                        *intertwin1;

            if((intertwin2!=0) && (D-mbc+mab-mbd>=0) && (D-mbc+mab-mbd<=deuxj25)){
              for (int mcd=0; mcd<=deuxj34; mcd++) {
                if ( (A-mad-mbd-mcd >=0) && (A-mad-mbd-mcd <= deuxj45) ) {

                  long double intertwin45
                 =intertwiner_mem(5,K4,A-mad-mbd-mcd,mbd,mcd)
                 *intertwiner_mem(5,K5,B+mad+mbd+mcd,D-mbc+mab-mbd,E-mac+mbc-mcd);

                  if(intertwin45!=0){
                    temp += signe6*intertwin2
```

```
                                        *intertwiner_mem(2,K3,deuxj13-mac,mbc,deuxj34-mcd)
                                        *intertwin45  ;
                                    }
                                }
                                signe6*=-1;
                            }
                        }
                        signe5*=-1;
                    }
                    signe4*=-1;
                }
            }
            signe3*=-1;
        }
        signe2*=-1;
    }
    signe1*=-1;
  }
  return temp*signe;
}
```

Which return the sum :

$$\text{function\_15j\_plus(K1,K2,K3,K4,K5)} = \sum_p (-1)^{\sum_{kl}(j^+_{kl}-m_{kl})} i^{\mathrm{K1}}_{-m_{12}m_{13}-m_{14}m_{15}} i^{\mathrm{K2}}_{-m_{23}m_{12}-m_{24}m_{25}} i^{\mathrm{K3}}_{-m_{13}m_{23}-m_{34}m_{35}} i^{\mathrm{K4}}_{m_{45}m_{24}m_{34}m_{14}} i^{\mathrm{K5}}_{-m_{45}-m_{25}-m_{35}-m_{15}} \left(j^+_{kl}\right) \tag{191}$$

Of course, because of the properties of the intertwiners (which are null if $\sum_{i=1}^4 m_i \neq 0$), we don't need to proceed the all sum over the m_{kl} ; so we can save more time-calculation just by the sum over six of them. This function do it, with the associated set of variables and notations :

$$m_{12} \leftrightarrow \text{mab}\ ,\ m_{23} \leftrightarrow \text{mbc}\ ,\ \ldots \tag{192}$$

$$\text{mab} = m_{12} + j_{12}, \text{mbc} = m_{23} + j_{23}, \ldots \tag{193}$$

The part of code for the function function_15j_moins(int K1, int K2, int K3, int K4, int K5), which give the corresponding $15j^-$-symbols, is the same except obviously for the mapping of the j-faces and associated intertwiners : the type of face and intertwiner are changed for their associated j^--elements.

B.3.6 Function for the iDroit

We have the following code for compute the future fusion coefficient associated to the boundary tetrahedra (j, j, j_0, j_0) :

```
long double iDroit_type1(int kplus,int kmoins, int ma, int mb, int mc) {

  float j1=j_face[1][1];
  float j2=j_face[1][2];
  float j3=j_face[4][1];
  float j4=j_face[4][2];
  float j1plus=j_face[2][1];
  float j2plus=j_face[2][2];
  float j3plus=j_face[5][1];
  float j4plus=j_face[5][2];
```

```
  float j1moins=j_face[3][1];
  float j2moins=j_face[3][2];
  float j3moins=j_face[6][1];
  float j4moins=j_face[6][2];

  int deuxj1=(int)(2*j1);
  int deuxj2=(int)(2*j2);
  int deuxj3=(int)(2*j3);
  int deuxj4=(int)(2*j4);
  int deuxj1plus=(int)(2*j1plus);
  int deuxj2plus=(int)(2*j2plus);
  int deuxj3plus=(int)(2*j3plus);
  int deuxj4plus=(int)(2*j4plus);
  int deuxj4moins=(int)(2*j4moins);

  int minmaplus=max(0,(deuxj1-2*j1moins-ma));
  int maxmaplus=min(deuxj1plus, (deuxj1-ma));
  int minmbplus=max(0,(deuxj2-2*j2moins-mb));
  int maxmbplus=min(deuxj2plus, (deuxj2-mb));
  int minmcplus=max(0,(deuxj3-2*j3moins-mc));
  int maxmcplus=min(deuxj3plus, (deuxj3-mc));
  long double temp=0;

  for (int maplus=minmaplus; maplus<= maxmaplus ; maplus++){
    int deuxj1moinsmaplus=(int)(deuxj1-maplus-ma);
    for (int mbplus=minmbplus; mbplus<= maxmbplus; mbplus++){
      int deuxj2moinsmbplus=(int)(deuxj2-mbplus-mb);
      for (int mcplus=minmcplus; mcplus<= maxmcplus; mcplus++){
        int A=(int)(j1plus+j2plus+j3plus+j4plus-maplus-mbplus-mcplus);
        int B=(int)(maplus+mbplus+mcplus + ma+mb+mc
                    + j1moins+j2moins+j3moins+j4moins -deuxj1-deuxj2-deuxj3);
        if ( (A>=0)&&(B>=0)&&(A<=deuxj4plus)&&(B<=deuxj4moins) ){
          int deuxj3moinsmcplus=(int)(deuxj3-mcplus-mc);
          temp+=intertwiner_mem(2,kplus,maplus,mbplus,mcplus)
 *intertwiner_mem(3,kmoins,deuxj1moinsmaplus,deuxj2moinsmbplus,deuxj3moinsmcplus)
 *trois_j_plusmoins(1,maplus,deuxj1moinsmaplus)
 *trois_j_plusmoins(1,mbplus,deuxj2moinsmbplus)
 *trois_j_plusmoins(2,mcplus,deuxj3moinsmcplus)
 *trois_j_plusmoins(2,A,B);
        }
      }
    }
  }
  return temp*sqrtl( (deuxj1+1)*(deuxj2+1)*(deuxj3+1)*(deuxj4+1) );
}
```

Which return :

$$\begin{array}{l}\text{iDroit\_type1(kplus,kmoins,ma,mb,mc)} = \\ \sum_{m^+,m^-} i^{\text{kplus}}_{m_1^+ m_2^+ m_3^+ m_4^+}\left(j^+,j^+,j_0^+,j_0^+\right) i^{\text{kmoins}}_{m_1^- m_2^- m_3^- m_4^-}\left(j^-,j^-,j_0^-,j_0^-\right)\prod_{l=1}^4\sqrt{2j_l+1}\begin{pmatrix} j_l^+ & j_l^- & j_l \\ m_l^+ & m_l^- & m_l\end{pmatrix} \\ \begin{pmatrix}\text{with :} & \begin{array}{c} m_1 = \text{ma} - j \\ m_2 = \text{mb} - j \\ m_3 = \text{mc} - j_0 \\ \left(m_4 = -\sum_{i=1}^3 m_i \right)\end{array}\end{pmatrix}\end{array} \tag{194}$$

The code for the shared-tetrahedra (j_f, j_0, j_0, j_0) is the same, with the appropriate type for the j_face, intertwiners and $\{j^+, j^-, j\}$-symbols, and define the iDroit_type2(int kplus, int kmoins, int ma, int mb, int mc) function.

B.3.7 Function for the fusion-coefficients

The fusion-coefficients of boundary tetrahedra (j, j, j_0, j_0) will be computed by the following function :

```
long double iCourbe_type1(int kplus,int kmoins, int k) {

  float j1=j_face[1][1];
  float j2=j_face[1][2];
  float j3=j_face[4][1];
  float j4=j_face[4][2];
  int deuxj1=(int)(2*j1);
  int deuxj2=(int)(2*j2);
  int deuxj3=(int)(2*j3);
  int deuxj4=(int)(2*j4);
  long double temp=0;

  for (int ma=0; ma<= deuxj1; ma++){
    for (int mb=0; mb<= deuxj2; mb++){
      for (int mc=0; mc<= deuxj3; mc++){
        temp+=iDroit_type1_mem(kplus,kmoins,ma,mb,mc)*intertwiner_mem(1,k,ma,mb,mc);
      }
    }
  }
  return temp;

}
```

Which will associate the corresponding iDroit (iDroit_type1) result with the corresponding intertwiner. Again, the code for the fusion-coefficient of shared-tetrahedra (j_f, j_0, j_0, j_0) is the same with the associated remapping of the type of iDroit and intertwiners. The two code return :

$$\text{iCourbe\_type1(kplus,kmoins,k)} = \mathcal{I}^{\text{k}}_{\text{kplus,kmoins}}(j, j, j_0, j_0) \tag{195}$$

$$\text{iCourbe\_type2(kplus,kmoins,k)} = \mathcal{I}^{\text{k}}_{\text{kplus,kmoins}}(j_f, j_0, j_0, j_0) \tag{196}$$

B.3.8 Function for 15j-$SO(4)$-symbols

Now we have a function for the computation of $SO(4)$ equivalent of 15j-symbols. The function will be combined the results given by the $15j^{\pm}$-symbols functions and the fusion-coefficients functions and return the value of the corresponding 15j-$SO(4)$-symbol as below :

```
long double function_15j_so4(int K1,int K2,int K3, int K4, int K5){

// coeffs plus
  float j12plus=j_face[2][1];
  float j13plus=j_face[2][2];
  float j14plus=j_face[5][1];
  float j15plus=j_face[5][2];
  float j23plus=j_face[2][1];
  float j24plus=j_face[5][1];
  float j25plus=j_face[5][2];
  float j34plus=j_face[5][1];
  float j35plus=j_face[5][2];
  float j45plus=j_face[8][1];
//moins
  float j12moins=j_face[3][1];
  float j13moins=j_face[3][2];
  float j14moins=j_face[6][1];
  float j15moins=j_face[6][2];
  float j23moins=j_face[3][1];
  float j24moins=j_face[6][1];
  float j25moins=j_face[6][2];
  float j34moins=j_face[6][1];
  float j35moins=j_face[6][2];
  float j45moins=j_face[9][1];

//
  float maxK1plus=2.0*min(j12plus,j14plus);
  float maxK4plus=j14plus+min(j14plus,j45plus);
  float minK4plus=(int)abs(j45plus-j14plus);
  float maxK1moins=2.0*min(j12moins,j14moins);
  float maxK4moins=(int)(j14moins+min(j14moins,j45moins));
  float minK4moins=(int)abs(j45moins-j14moins);
  long double result=0;

for(int K1m=0; K1m<=maxK1moins; K1m++){
  for(int K2m=0; K2m<=maxK1moins; K2m++){
    for(int K3m=0; K3m<=maxK1moins; K3m++){
      for(int K4m=minK4moins; K4m<=maxK4moins; K4m++){
        for(int K5m=minK4moins; K5m<=maxK4moins; K5m++){

          long double symbol_15jm = function_15j_moins_mem(K1m,K2m,K3m,K4m,K5m);
          if(symbol_15jm != 0){ // <--- if symbol_15jm non null

            long double sum=0;
            for(int K2p=0; K2p<=maxK1plus; K2p++){

              long double i_K2pK2mK2 = iCourbe_type1_mem(K2p,K2m,K2);
              if(i_K2pK2mK2 != 0){ // <--- if i_K2pK2mK2 non null !

                long double sum_temp=0;
                for(int K1p=0; K1p<=maxK1plus; K1p++){
```

```
                for(int K4p=minK4plus; K4p<=maxK4plus; K4p++){

                  long double temp2=0;
                  for(int K3p=0; K3p<K1p; K3p++){
                    long double temp=0;
                    for(int K5p=minK4plus; K5p<K4p; K5p++){
                      temp += function_15j_plus_mem(K1p,K2p,K3p,K4p,K5p)
*( iCourbe_type2_mem(K4p,K4m,K4)*iCourbe_type2_mem(K5p,K5m,K5)
+ pow(-1.0,K1+K2+K3)*iCourbe_type2_mem(K4p,K4m,K5)*iCourbe_type2_mem(K5p,K5m,K4) );
                    }
                    temp2 += temp
*( iCourbe_type1_mem(K1p,K1m,K1)*iCourbe_type1_mem(K3p,K3m,K3)
+ pow(-1.0,K1+K2+K3+K4+K5)*iCourbe_type1_mem(K1p,K1m,K3)*iCourbe_type1_mem(K3p,K3m,K1) );
                  }
                  // temp2 = sum_K5+<K4+ sum_K3+<K1+ of 15j+ * I

                  long double temp3=0;
                  for(int K3p=0; K3p<K1p; K3p++){
                    temp3 += function_15j_plus_mem(K1p,K2p,K3p,K4p,K4p)
*( iCourbe_type1_mem(K1p,K1m,K1)*iCourbe_type1_mem(K3p,K3m,K3)
+ pow(-1.0,K1+K2+K3+K4+K5)*iCourbe_type1_mem(K1p,K1m,K3)*iCourbe_type1_mem(K3p,K3m,K1) );
                  }
                  temp3 *= iCourbe_type2_mem(K4p,K4m,K4)*iCourbe_type2_mem(K4p,K5m,K5);
                  // temp3 = sum_K3+<K1+ of 15j+ * I

                  long double temp4=0;
                  for(int K5p=minK4plus; K5p<K4p; K5p++){
                    temp4 += function_15j_plus_mem(K1p,K2p,K1p,K4p,K5p)
*( iCourbe_type2_mem(K4p,K4m,K4)*iCourbe_type2_mem(K5p,K5m,K5)
+ pow(-1.0,K1+K2+K3)*iCourbe_type2_mem(K4p,K4m,K5)*iCourbe_type2_mem(K5p,K5m,K4) );
                  }
                  temp4 *= iCourbe_type1_mem(K1p,K1m,K1)*iCourbe_type1_mem(K1p,K3m,K3);
                  // temp4 = sum_K5+<K4+ of 15j+ * I

                  sum_temp += temp2 + temp3 + temp4
                              + function_15j_plus_mem(K1p,K2p,K1p,K4p,K4p)
                                *iCourbe_type1_mem(K1p,K1m,K1)
                                *iCourbe_type1_mem(K1p,K3m,K3)
                                *iCourbe_type2_mem(K4p,K4m,K4)
                                *iCourbe_type2_mem(K4p,K5m,K5);
                 // sum_temp = sum_K3+ sum_K5+ of 15j+ * I
                }
              }
              sum += sum_temp*i_K2pK2mK2;
            } // <--- end if i_K2pK2mK2 non null !
          }
          result += symbol_15jm*sum;
        } // <--- end if symbol_15jm non null !
      }
    }
  }
 }
```

```
}
return result;

}
```

Here, we needed to be very clever and found a tricky way for compute the value of function_15j_so4(K1,K2,K3,K4,K5), because the number of steps is huge and the calculation is very long. Indeed, we need to compute the object :

$$\begin{array}{c}\text{fonction\_15j\_so4(K1,K2,K3,K4,K5)} = [\text{K1,K2,K3,K4,K5}; j, j_0, j_f] = \\ \sum_{K^+,K^-} \left(K_k^+; j_{kl}^+\right) \left(K_k^-; j_{kl}^-\right) \mathcal{I}^{\text{K1}}_{K_k^+,K_k^-}\left(j,j,j_0,j_0\right) \mathcal{I}^{\text{K2}}_{K_2^+,K_2^-}\left(j,j,j_0,j_0\right) \mathcal{I}^{\text{K3}}_{K_3^+,K_3^-}\left(j,j,j_0,j_0\right) \\ \times \mathcal{I}^{\text{K4}}_{K_4^+,K_4^-}\left(j_f,j_0,j_0,j_0\right) \mathcal{I}^{\text{K5}}_{K_5^+,K_5^-}\left(j_f,j_0,j_0,j_0\right)\end{array} \tag{197}$$

which have ten sums, and each sums have the approximate size of $2j+1$! So the number of steps is of the order of $(2j+1)^{10} \sim 10^{13}$ for $j \sim 10$! For simplify the computation and save a lot of time-calculation, we use the symmetries properties inside the 15j-$SO(4)$-symbols. In our definition of 15j-symbols, we have the symmetries :

$$\begin{array}{rl}\left(K_1,K_2,K_3,K_4,K_5;j,j_0,j_f\right) & = (-1)^{K_1+K_2+K_3+K_4+K_5}\left(K_3,K_2,K_1,K_4,K_5;j,j_0,j_f\right) \\ & = (-1)^{K_4+K_5}\left(K_3,K_2,K_1,K_5,K_4;j,j_0,j_f\right)\end{array} \tag{198}$$

And from the fusion coefficients :

$$(-1)^{K^+ + K^-}\, \mathcal{I}^K_{K^+,K^-} = (-1)^K\, \mathcal{I}^K_{K^+,K^-} \tag{199}$$

So, the symmetry properties of the $15j^{\pm}$-symbols are transmitted to the 15j-$SO(4)$-symbols :

$$\begin{array}{rl}\left[K_1,K_2,K_3,K_4,K_5;j,j_0,j_f\right] & = (-1)^{K_1+K_2+K_3+K_4+K_5}\left[K_3,K_2,K_1,K_4,K_5;j,j_0,j_f\right] \\ & = (-1)^{K_4+K_5}\left[K_3,K_2,K_1,K_5,K_4;j,j_0,j_f\right]\end{array} \tag{200}$$

We will use the properties of the $15j^{\pm}$-symbols and fusion-coefficients to rewrite in a more light way (in the sens of machine resources and time) the 15j-$SO(4)$-symbols :

$$\begin{array}{l}\left[K_1,K_2,K_3,K_4,K_5;j,j_0,j_f\right] = \\ \sum_{K^-}\left(K_1^-K_2^-K_3^-K_4^-K_5^-;j,j_0,j_f\right)\sum_{K_2^+} I^{K_2^+K_2^-}_{K_2^+} \sum_{K_1^+K_4^+}\Big\{\left(K_1^+K_2^+K_1^+K_4^+K_4^+;j,j_0,j_f\right)\mathcal{I}^{K_1}_{K_1^+,K_1^-}\mathcal{I}^{K_3}_{K_1^+,K_3^-}\mathcal{I}^{K_4}_{K_4^+,K_4^-}\mathcal{I}^{K_5}_{K_4^+,K_5^-} \\ +\mathcal{I}^{K_1}_{K_1^+,K_1^-}\mathcal{I}^{K_3}_{K_1^+,K_3^-}\sum_{K_5^+<K_4^+}\left(K_1^+K_2^+K_1^+K_4^+K_5^+;j,j_0,j_f\right)\left[\mathcal{I}^{K_4}_{K_4^+,K_4^-}\mathcal{I}^{K_5}_{K_5^+,K_5^-}+(-1)^{K_1+K_2+K_3}\mathcal{I}^{K_5}_{K_4^+,K_4^-}\mathcal{I}^{K_4}_{K_5^+,K_5^-}\right] \\ +\mathcal{I}^{K_4}_{K_4^+,K_4^-}\mathcal{I}^{K_5}_{K_4^+,K_5^-}\sum_{K_3^+<K_1^+}\left(K_1^+K_2^+K_3^+K_4^+K_4^+;j,j_0,j_f\right)\left[\mathcal{I}^{K_1}_{K_1^+,K_1^-}\mathcal{I}^{K_3}_{K_3^+,K_3^-}+(-1)^{K_1+K_2+K_3+K_4+K_5}\mathcal{I}^{K_3}_{K_1^+,K_1^-}\mathcal{I}^{K_1}_{K_3^+,K_3^-}\right] \\ +\sum_{K_3^+<K_1^+}\left[\mathcal{I}^{K_1}_{K_1^+,K_1^-}\mathcal{I}^{K_3}_{K_3^+,K_3^-}+(-1)^{K_1+K_2+K_3+K_4+K_5}\mathcal{I}^{K_3}_{K_1^+,K_1^-}\mathcal{I}^{K_1}_{K_3^+,K_3^-}\right] \\ \qquad\times\sum_{K_5^+<K_4^+}\left(K_1^+K_2^+K_3^+K_4^+K_5^+;j,j_0,j_f\right)\left[\mathcal{I}^{K_4}_{K_4^+,K_4^-}\mathcal{I}^{K_5}_{K_5^+,K_5^-}+(-1)^{K_1+K_2+K_3}\mathcal{I}^{K_5}_{K_4^+,K_4^-}\mathcal{I}^{K_4}_{K_5^+,K_5^-}\right]\Big\}\end{array} \tag{201}$$

And it's exactly this formula which are coded in the function_15j_so4(K1,K2,K3,K4,K5). With it's way, the time for compute just one 15j-$SO(4)$-symbol (with the tables of the $15j^{\pm}$-symbols and fusion-coefficient already in memory) for $j \sim 10$ is to the order of 1 second.

B.3.9 Function for the representation $D^j_{mj}(\theta,\phi)$

For the coherent states we need to compute the representation $D^j_{mj}(\theta,\phi)$ associated to a arbitrary j-face. So we have the d(ma,deuxj,theta,phi) function as follow :

```
complex<long double> d(int ma, int deuxj,float theta, float phi)
{
complex<long double> phase_d(cos(phi*(deuxj/2.0-ma)),sin(phi*(deuxj/2.0-ma)));
float costemp=cos(theta/2.0);
```

```
float sintemp=sin(theta/2.0);

if(((costemp==0)&&(ma==0))||((sintemp==0)&&(ma==deuxj))){
  return phase_d;
}else{
  complex<long double> norm_d( sqrt(fact(deuxj)/(fact(ma)*fact(deuxj-ma)))
                                *pow(costemp,ma)*pow(sintemp,deuxj-ma)     , 0 );
  return phase_d*norm_d;
  }
}
```

Which return the associated value from the Wigner representation matrices :

$$\begin{aligned} \mathrm{d(ma,}2j\mathrm{,}\theta\mathrm{,}\phi\mathrm{)} &= D^{j}_{(\mathrm{ma}-j)j}(\theta,\phi) \\ &= D^{j}_{(\mathrm{ma}-j)j}\left(e^{-\imath\phi J_Z}e^{-\imath\theta J_Y}\right) = e^{-\imath(\mathrm{ma}-j)\phi}d^{j}_{(\mathrm{ma}-j)j} \\ &= \sqrt{\frac{(2j)!}{\mathrm{ma}!\,(2j-\mathrm{ma})}}e^{-\imath(\mathrm{ma}-j)\phi}\left[\cos\left(\frac{\theta}{2}\right)\right]^{\mathrm{ma}}\left[\sin\left(\frac{\theta}{2}\right)\right]^{(2j-\mathrm{ma})} \end{aligned}$$

B.3.10 Function for load the 15j-$SO(4)$-symbols file

For save more time, if some values of 15j-$SO(4)$-symbols are precalculated, it will be useful to have a save-load system for get the previous values from a past execution of the code. So we define the load_15j_so4(j,j_0,j_f) function for load the 15j-$SO(4)$-symbols values from a data file called "simplex_walking_j0=j_0_j=j_jf=j_f.txt" :

```
//////////////////////Loading function for the 15j_so4 table//////////////////////

void load_15j_so4(int j, int j0, int jf) {
    ostringstream j0s, js, jfs;//create a stringstream
    j0s << j0;//add number to the stream
    js << j;
    jfs << jf;

    string name = "simplex_walking_j0=";
    name += j0s.str();
    name += "_j=";
    name += js.str();
    name += "_jf=";
    name += jfs.str();
    name += ".txt";
    ifstream results_so4_w(name.c_str(), ios::in);  //open the 15j so4 file in read

    int K1,K2,K3,K4,K5;
    // initialization of Boolean table
    int K1max=(int)(2*min(j0,j));
    int K4max=(int)(j0+min(j0,jf));
    for (K1=0; K1<=K1max; K1++) {
      for (K2=0; K2<=K1max; K2++) {
        for (K3=0; K3<=K1; K3++) {
          for (K4=0; K4<=K4max; K4++) {
            for (K5=0; K5<=K4; K5++) {
```

```
            t15j_so4_present[K1][K2][K3][K4][K5]=false;
          }
        }
      }
    }

    // we load the file directly !
    long double v15j_so4;
    if (results_so4_w) {
      string ligne;
      while (getline(results_so4_w, ligne)) {
        results_so4_w>>K1;
        results_so4_w>>K2;
        results_so4_w>>K3;
        results_so4_w>>K4;
        results_so4_w>>K5;
        results_so4_w>>v15j_so4;
        t15j_so4_loaded[K1][K2][K3][K4][K5]=v15j_so4;
        t15j_so4_present[K1][K2][K3][K4][K5]=true;
      }
    }

  results_so4_w.close();          //close the 15j so4 file
}
```

For the specified value of j, j_0 and j_f the code recreates the name of the corresponding file "simplex_walking_j0=j_0_j=j_jf=j_f.tx and call the data-file with this name. The data-file is open for reading, and we use some loops to put beforehand the Boolean t15j_so4_present[][][][][] table to "false" : that "false" table means the values of 15j-$SO(4)$-symbols are not already loaded, and the code is ready for get the values from the file. We have a loop over the lines from the file, and we get for each line the corresponding K parameter and its corresponding value of 15j-$SO(4)$-symbol from the file ; we store the value in the t15j_so4_loaded[K1][K2][K3][K4][K5] table box and put the associated t15j_so4_present[K1][K2][K3][K4][K5] table box to "true" : that will means for the rest of the code that the value of 15j-$SO(4)$-symbol are store in its table box.

B.4 Main code

The all arrays and the all functions are defined, now we will expose and explain the main code. The main() will compute the all coherent objects before write the all results in data-files.

B.4.1 Boundary components

The first steps are to compute the boundary elements which not depend of j_f. We will define and construct the all objects needed for the boundary tetrahedra and associated coherent states. The results computed will be stored in the corresponding arrays and tables for the next steps dedicated to the transition amplitude computation.

Beginning of the main and definitions of j_face for boundary faces j, j_0 In the starting of the main, we will define and give the value of j, j_0 and Immirzi parameter that are the base for the boundary definition. With the help of the jp_approx function define above we compute the corresponding $j^{\pm}$, $j_0^{\pm}$ and stock the all values in the j_face[][] tables. The code is :

```
//////////////////////////////-CODE MAIN()-//////////////////////////////
```

```
int main(){
   cout << "\n Starting of the main ! "<< endl;

  float j0=8;
  float j=8;

  int maxJ=(int)(2*min(j0,j));

  float Immirzi=0.5;

  float j0plus=jp_approx(Immirzi,j0,1);
  float j0moins=j0-j0plus;
  float jplus=jp_approx(Immirzi,j,1);
  float jmoins=j-jplus;

  // definition des j_face
  j_face[1][1]=j;
  j_face[1][2]=j;
  j_face[2][1]=jplus;
  j_face[2][2]=jplus;
  j_face[3][1]=jmoins;
  j_face[3][2]=jmoins;
  j_face[4][1]=j0;
  j_face[4][2]=j0;
  j_face[5][1]=j0plus;
  j_face[5][2]=j0plus;
  j_face[6][1]=j0moins;
  j_face[6][2]=j0moins;
  j_face[7][2]=j0;
  j_face[8][2]=j0plus;
  j_face[9][2]=j0moins;
```

The jp_approx(Immirzi,j,1) return the integer or half-integer closest to the $\frac{1\pm\text{Immirzi}}{2}j$ value and, in case of ambiguity (see the properties of the function), the parameter "1" return the closest integer. So the all j for the boundary faces are stocked in the j_face tables, we will be able to compute the other math objects from them.

Computation of the 3j-symbols for the boundary tetrahedra Now that the boundary j are available, we have the section of the code for compute the all corresponding 3j-symbols :

```
  // populating of 3j-type arrays
  cout << "populating of 3j-type arrays"<< endl;

  for (int type=1;type<=6;type++) {
    float j1=j_face[type][1];
    float j2=j_face[type][2];
    float mamax=2*j_face[type][1];
    float mbmax=2*j_face[type][2];
    float minK=abs(j_face[type][1]-j_face[type][2]);
    float maxK=(j_face[type][1]+j_face[type][2]);
    for (int K=minK; K<=maxK; K++) {
      for (int ma=0; ma<=mamax; ma++) {
```

```
        float m1=ma-j1;
        for (int mb=0; mb<=mbmax; mb++) {
          trois_j_type_t[type][K][ma][mb]=trois_j_sans_m3(j1,j2,K,m1,mb-j2) ;
        }
      }
    }
  }

  // populating of (j+,j-,j)-symbols arrays
  cout << "populating of (j+,j-,j)-symbols arrays"<< endl;
  //type 1 -> calculation with j
  float mamax=2*jplus;
  float mbmax=2*jmoins;
  for (int ma=0; ma<=mamax; ma++) {
    float m1=ma-jplus;
      for (int mb=0; mb<=mbmax; mb++) {
       trois_j_plusmoins_t[1][ma][mb]=trois_j_sans_m3(jplus,jmoins,j,m1,mb-jmoins);
    }
  }
  // type 2 -> calculation with j0
  mamax=2*j0plus;
  mbmax=2*j0moins;
  for (int ma=0; ma<=mamax; ma++) {
    float m1=ma-j0plus;
      for (int mb=0; mb<=mbmax; mb++) {
       trois_j_plusmoins_t[2][ma][mb]=trois_j_sans_m3(j0plus,j0moins,j0,m1,mb-j0moins);
    }
  }
```

In the first block, for build the 3j-symbols for the future (boundary) intertwiners, we have a loop over the type of 3j-symbols. This loop, for the type 1 to 6, will build the six 3j-symbols from the j, j^+, j^-, j_0, j_0^+ and j_0^- respectively. For each type, we have a temporary definition of the values of faces from the j_face[][] and a computation of corresponding boundaries for the K parameter. After, we call the trois_j_sans_m3 function and stock the result in the trois_j_type[][][][][] table for the all possible indices. The next block of this part of code have the same logic and utility but for generate the $\{j^+, j^-, j\}$-symbols and the $\{j_0^+, j_0^-, j_0\}$-symbols.

Computation of intertwiners for the boundary tetrahedra Like the previous part, we will call the corresponding functions and generate the values for the intertwiners :

```
  // populating of intertwiners arrays
  cout << "populating of intertwiners arrays"<< endl;

  // type 1 -> 3j type 1 et 4:
  mamax=2*j_face[1][1];
  mbmax=2*j_face[1][2];
  float mcmax=2*j_face[4][1];
  float Kmax=min(j_face[1][1]+j_face[1][2],j_face[4][1]+j_face[4][2]);
  float Kmin=abs(j_face[1][1]-j_face[1][2]);
  for (int K=Kmin; K<=Kmax ;K++) {
    for (int ma=0; ma<=mamax; ma++) {
      for (int mb=0; mb<=mbmax; mb++) {
        for (int mc=0; mc<=mcmax; mc++){
```

```
          intertwiner_t[1][K][ma][mb][mc]=intertwiner(1,4,K,ma,mb,mc);
        }
      }
    }
  }
  // type 2 -> 3j type 2 et 5:
  mamax=2*j_face[2][1];
  mbmax=2*j_face[2][2];
  mcmax=2*j_face[5][1];
  Kmax=min(j_face[2][1]+j_face[2][2],j_face[5][1]+j_face[5][2]);
  Kmin=abs(j_face[2][1]-j_face[2][2]);
  for (int K=Kmin; K<=Kmax ;K++) {
    for (int ma=0; ma<=mamax; ma++) {
      for (int mb=0; mb<=mbmax; mb++) {
        for (int mc=0; mc<=mcmax; mc++){
          intertwiner_t[2][K][ma][mb][mc]=intertwiner(2,5,K,ma,mb,mc);
        }
      }
    }
  }
  // type 3 -> 3j type 3 et 6:
  mamax=2*j_face[3][1];
  mbmax=2*j_face[3][2];
  mcmax=2*j_face[6][1];
  Kmax=min(j_face[3][1]+j_face[3][2],j_face[6][1]+j_face[6][2]);
  Kmin=abs(j_face[3][1]-j_face[3][2]);
  for (int K=Kmin; K<=Kmax ;K++) {
    for (int ma=0; ma<=mamax; ma++) {
      for (int mb=0; mb<=mbmax; mb++) {
        for (int mc=0; mc<=mcmax; mc++){
          intertwiner_t[3][K][ma][mb][mc]=intertwiner(3,6,K,ma,mb,mc);
        }
      }
    }
  }
```

The code create and stock successively the boundary intertwiners $i^K(j,j,j_0,j_0)$, $i^K\left(j^+,j^+,j_0^+,j_0^+\right)$ and $i^K\left(j^-,j^-,j_0^-,j_0^-\right)$ in the intertwiner_t[1][K][][][], intertwiner_t[2][K][][][][] and intertwiner_t[3][K][][][][] tables. For each type of intertwiners, the code use the corresponding intertwiner() function with the corresponding parameter for select the type of 3j-symbols needed : intertwiner_t[1] → intertwiner(1,4,...), intertwiner_t[2] → intertwiner(2,5,...) etc.

Computation of the fusion coefficient for the boundary tetrahedra Here we will have the part for compute the iDroit_type1 terms and, after, compute the fusion-coefficients for the boundary tetrahedra :

```
  // Let's go for the iDroit:
  cout << "Generating the iDroit"<< endl;

  float maxKplus=2.0*min(jplus,j0plus);
  float maxKmoins=2.0*min(jmoins,j0moins);
  int deuxj=(int)(2.0*j);
  int deuxj0=(int)(2.0*j0);
```

```
for (int Kplus=0; Kplus<= maxKplus; Kplus++){
  for (int Kmoins=0; Kmoins<= maxKmoins; Kmoins++){
    for (int ma=0; ma<=deuxj; ma++){
      for (int mb=0; mb<=deuxj; mb++){
        for (int mc=0; mc<=deuxj0; mc++){
          if ( abs(ma+mb+mc-deuxj-j0)>j0  ) {
            iDroit_type1_t[Kplus][Kmoins][ma][mb][mc]=0;
          } else {
            iDroit_type1_t[Kplus][Kmoins][ma][mb][mc]=iDroit_type1(Kplus,Kmoins,ma,mb,mc);
          }
        }
      }
    }
  }
}

// And the fusion-coefficients
cout << "And the fusion-coefficients"<< endl;

maxKplus=2.0*min(jplus,j0plus);
maxKmoins=2.0*min(jmoins,j0moins);
float maxK=2.0*min(j,j0);
for (int Kplus=0; Kplus<= maxKplus; Kplus++){
  for (int Kmoins=0; Kmoins<= maxKmoins; Kmoins++){
    for (int K=0; K<= maxK; K++){
      iCourbe_type1_t[Kplus][Kmoins][K]=iCourbe_type1(Kplus,Kmoins,K);
    }
  }
}
```

We have the definitions of the K^+, K^-, K boundaries in terms of corresponding j and call the functions for store the fusion-coefficients tables.

Computation of the coherent states for the boundary tetrahedra It remains the states for finally get the all boundary elements, prelude to the calculation over the shared-tetrahedra and the internal geometries. For the coherent states from the boundary tetrahedra, we need to do the calculus :

$$\begin{aligned}\left\langle i^J | j_i \overrightarrow{n_i} \right\rangle &= \sum_m i^J_{m_1 m_2 m_3 m_4} D^{j_1}_{m_1 j_1}\left(R\left(\overrightarrow{n_1}\right)\right) D^{j_2}_{m_2 j_2}\left(R\left(\overrightarrow{n_2}\right)\right) D^{j_3}_{m_3 j_3}\left(R\left(\overrightarrow{n_3}\right)\right) D^{j_4}_{m_4 j_4}\left(R\left(\overrightarrow{n_4}\right)\right) \\ &= \sum_m i^J_{m_1 m_2 m_3 m_4} D^{j_1}_{m_1 j_1}\left(\theta_1,\phi_1\right) D^{j_2}_{m_2 j_2}\left(\theta_2,\phi_2\right) D^{j_3}_{m_3 j_3}\left(\theta_3,\phi_3\right) D^{j_4}_{m_4 j_4}\left(\theta_4,\phi_4\right)\end{aligned}$$

Which is simply with the gauge choice $(\theta_1,\phi_1)=(0,0)$, $\phi_2=0$ and the closure condition $\sum_i j_i \overrightarrow{n_1} = \overrightarrow{0}$:

$$\left\langle i^J | j, j_0, A, \Phi \right\rangle = \sum_m i^J_{j_1 m_2 m_3 m_4} D^{j_2}_{m_2 j_2}\left(\theta_2,0\right) D^{j_3}_{m_3 j_3}\left(\theta_3,\phi_3\right) D^{j_4}_{m_4 j_4}\left(\theta_4,\phi_4\right)\left[j, j_0, A, \Phi\right] \tag{202}$$

$$
\begin{aligned}
\theta_2 &= \arccos\left(\frac{A^2}{2j^2}-1\right)\\
\theta_3 &= \arccos\left(\frac{\cos(\Phi)\sqrt{4j_0^2-A^2}\sqrt{4j^2-A^2}-A^2}{4jj_0}\right)\\
\theta_4 &= \arccos\left(\frac{-\cos(\Phi)\sqrt{4j_0^2-A^2}\sqrt{4j^2-A^2}-A^2}{4jj_0}\right)\\
\phi_3 &= \arccos\left(\frac{\sin^2\theta_4-\sin^2\theta_2-\sin^2\theta_3}{2\sin\theta_2\sin\theta_3}\right)\\
\phi_4 &= 2\pi-\arccos\left(\frac{\sin^2\theta_3-\sin^2\theta_2-\sin^2\theta_4}{2\sin\theta_2\sin\theta_4}\right)
\end{aligned}
$$

But, obviously, we cannot compute that for the all continuum, that make no sens for the machine ! So we will define a resolution for the continuous parameters $A \in [0; 2\min(j, j_0)]$ and $\Phi \in [0; \pi]$ and just compute them for discrete value given by two code parameters n and t :

$$A = \frac{1+2n}{102} \times 2\min(j, j_0) \text{ for } n = 0, 1, \ldots, 50 \tag{203}$$

$$\Phi = \frac{1+2t}{102} \times \pi \text{ for } t = 0, 1, \ldots, 50 \tag{204}$$

We will have 51 points uniformly distributed for each variables A and Φ which include the middle cases ($A = \min(j, j_0)$ for $n = 25$ or $\Phi = \frac{\pi}{2}$ for $t = 25$) and exclude the extreme cases ($A = 0, \min(j, j_0)$ or $\Phi = 0, \pi$). So we will have a good resolution for the data, without the full degenerated cases where the classical geometry give absurd results in the context of the numerical calculus : tetrahedra infinitely elongated or completely flat which give some annoying "division by zero" for the machine. The associated code is :

```
cout << "Next, the coherent boundary states"<< endl;

complex<long double> iJd_externe[maxJ+1][51][51];
ofstream resultInter("iJd_externe.txt", ios::out | ios::trunc);

for(int t=0;t<=50;t++){
  float PHI=(1+2*t)*M_PI/102.0;
   //Dissymmetry parameter - angle between (n1,n2)^(n3,n4) - partner variable of K
  for(int n=0;n<=50;n++){
    float A=(1+2*n)*maxJ/102.0;
    float theta2=acos(A*A/(2.0*j*j)-1.0);
    float theta3=acos((cos(PHI)*sqrt(4.0*j0*j0-A*A)*sqrt(4.0*j*j-A*A) - A*A)
                      /(4.0*j*j0));
    float theta4=acos((-cos(PHI)*sqrt(4.0*j0*j0-A*A)*sqrt(4.0*j*j-A*A) - A*A)
                      /(4.0*j*j0));
    float phi3=acos((pow(sin(theta4),2) - pow(sin(theta2),2) - pow(sin(theta3),2))
                      /(2.0*sin(theta2)*sin(theta3)));
    float phi4=2.0*M_PI
              -acos((pow(sin(theta3),2) - pow(sin(theta2),2) - pow(sin(theta4),2))
                      /(2.0*sin(theta2)*sin(theta4)));

    // Creation of the iJd_externe
```

```
      long double norm_iJd_externe=0;
      for(int K=0;K<=maxJ;K++){
        iJd_externe[K][n][t]=0;
        for(int mb=0;mb<=deuxj;mb++){
          for(int mc=0;mc<=deuxj0;mc++){
            int md=deuxj0-mb-mc;
            if(md>=0){
              iJd_externe[K][n][t] += intertwiner_mem(1,K,deuxj,mb,mc)
                                      *d(mb,deuxj,theta2,0)
                                      *d(mc,deuxj0,theta3,phi3)
                                      *d(md,deuxj0,theta4,phi4);
            }
          }
        }
      norm_iJd_externe+=abs(iJd_externe[K][n][t])*abs(iJd_externe[K][n][t]);
      }

      if(norm_iJd_externe!=0){
        for(int K=0;K<=maxJ;K++){
          complex<long double> temp((deuxj+1.0)*(deuxj0+1.0),0);
          //complex<long double> temp(1.0/sqrt(norm_iJd_externe),0);
          iJd_externe[K][n][t]*=temp;
          resultInter<<PHI<<" "<<A<<"\t"<<norm_iJd_externe<<"\t"<<K<<"\t"
                     <<abs(iJd_externe[K][n][t])<<" "<<arg(iJd_externe[K][n][t])<<endl;
        }
        resultInter<<endl;
      }
    }
    resultInter<<endl;
  }
  resultInter.close();
```

We define the array iJd_externe[maxJ+1][51][51] which will contain the coherent tetrahedra state ; remember that the maxJ $= 2\min(j, j_0)$ correspond to the maximum boundary of the J parameter and the two [51] are for the all points given for n and t. After, we have the definition of the file "iJd_externe.txt" (open for writing) for write, in parallel to the calculation, the results from the coherent states of boundary tetrahedra.

We have the two loops over the t and n, for range the all space (Φ, A). For each step for given t and n, we compute the corresponding classical variables (θ_i, ϕ_i) and compute from them the coherent state with the associated intertwiner from table and the d($m_{(i)}$,$2j_i$,θ_i,ϕ_i) functions. We store, temporarily, the iJd_externe[][][][] table with the coherent state :

$$\text{iJd\_externe[K][n][t]} = \left\langle i^{\mathrm{K}} | j, j_0, A[\mathrm{n}], \Phi[\mathrm{t}] \right\rangle \tag{205}$$

and compute, on the fly, the norm of the state : norm_iJd_externe $= \sum_{\mathrm{K}} \left| \left\langle i^{\mathrm{K}} | j, j_0, A[\mathrm{n}], \Phi[\mathrm{t}] \right\rangle \right|^2$. The next step consist just to renormalize the non-null states (norm_iJd_externe $\neq 0$) with the j-representation and store the the iJd_externe[][][][] with this renormalization :

$$\text{iJd\_externe[K][n][t]} \rightarrow \text{iJd\_externe[K][n][t]} = (2j+1)(2j_0+1)\left\langle i^{\mathrm{K}} | j, j_0, A[\mathrm{n}], \Phi[\mathrm{t}] \right\rangle \tag{206}$$

So we have normalized coherent states in the iJd_externe[][][][] in the sens of orthogonality relations :

$$\prod_i \int \frac{d\overrightarrow{n_i}}{4\pi} \sqrt{\prod_i (2j_i+1)} \left\langle i^J | j_i \overrightarrow{n_i} \right\rangle \times \sqrt{\prod_i (2j_i+1)} \overline{\left\langle i^{J'} | j_i \overrightarrow{n_i} \right\rangle} = \delta^{J,J'} \tag{207}$$

In the renormalization part, we have a commented line corresponding to a another possible renormalization. If you uncomment this line, compile and execute the code, you will process to the following renormalization :

$$\text{iJd\_externe[K][n][t]} = \frac{\langle i^{\mathrm{K}} | j, j_0, A[\mathrm{n}], \Phi[\mathrm{t}] \rangle}{\sqrt{\sum_{\mathrm{K'}} |\langle i^{\mathrm{K'}} | j, j_0, A[\mathrm{n}], \Phi[\mathrm{t}] \rangle|^2}} \tag{208}$$

B.4.2 Loop for the face f and internal components

At this point, the all boundary objects are define and in the machine memory. We will start to expend the process for the shared object given for each compatible $j_f \in [0; 3j_0]$ and compute the objects for obtain the transition amplitudes. The corresponding process will be given by a loop over the all available j_f where will we compute the 3j-symbols, intertwiners, fusion-coefficients, $15j^{\pm}$-symbols, 15j-$SO(4)$-symbols and coherent transition amplitudes which depend of the j_f-representation. For each j_f the computed results will be stored in their arrays and tables, and full coherent transition amplitude will be done progressively for the next files-writing and study code part.

Start of the f-loop and definition of the j_face for f Now we will start the loop over the all compatible values of j_f to get the math elements from shared-tetrahedra :

```
  //Loop over the jf
  cout << "////START jf-LOOP////"<< endl;
float jf=0;
for(jf=j0-(int)(j0);jf<=3*j0;jf++){ //jf=j0-(int)(j0) implies that jf same type as j0
                                    //otherwise the elements are nulls
  float jfplus=jp_approx(Immirzi,jf,1);
  float jfmoins=jf-jfplus;
  if((jfplus-(int)(jfplus))==(j0plus-(int)(j0plus))) {  //Check if jf+ same type as j0+
                                                        //=>  Calculation required
  // definition of j_face for jf
    j_face[7][1]=jf;
    j_face[8][1]=jfplus;
    j_face[9][1]=jfmoins;
    cout << "--- Case jf="<<jf<<" (jf-="<<jfmoins<<",jf+="<<jfplus<<") ---"<< endl;
```

The starting jf value for the loop is define with j0-(int)(j0) because the non-nulls elements happen when j_0 and j_f are the same type ; the value j0-(int)(j0) term return 0 or $\frac{1}{2}$ if the j0 is integer or half-integer. The other jf values will be obtained by simple incrementation of the loop from this starting value. For each value of j_f given by the loop, the code compute the j_f^+ and j_f^- values (with the jp_approx function) in the optics to check if they are compatible ; because the non-nulls elements from the $j_f^{\pm}$ are given for j_f^+ (and j_f^-) which is same type as j_0^+ (and j_0^-). So, after the definition of $j_f^{\pm}$, we have a if-condition for check the type of j_f^+ : jfplus-(int)(jfplus) will return 0 or $\frac{1}{2}$ if ${j_f}^+$ integer or half-integer and j0plus-(int)(j0plus) will return 0 or $\frac{1}{2}$ if j_0^+ integer or half-integer, the two results are equal only if j_f^+ and j_0^+ have the same type.

At the level of the if-condition the j_f value, and its $j_f^{\pm}$, is compatible for non-nulls elements. We have filtered the non-compatible values of j_f which would have given automatically null elements, so we can store the values of j_face[][] for the calculus and proceed to the next.

Computation of the 3j-symbols for the shared-tetrahedra With the values of j_face[][] containing the j_f and $j_f^{\pm}$ we can, as the boundary part, compute the 3j-symbols :

```
    // peuplement des tableaux des 3jtype (WITH jf !!!)
    cout << "peuplement des tableaux des 3jtype (WITH jf !!!)"<< endl;
```

```
for (int type=7;type<=9;type++) {
  float j1=j_face[type][1];
  float j2=j_face[type][2];
  float mamax=2*j_face[type][1];
  float mbmax=2*j_face[type][2];
  float minK=abs(j_face[type][1]-j_face[type][2]);
  float maxK=(j_face[type][1]+j_face[type][2]);
  for (int K=minK; K<=maxK; K++) {
    for (int ma=0; ma<=mamax; ma++) {
      float m1=ma-j1;
      for (int mb=0; mb<=mbmax; mb++) {
        trois_j_type_t[type][K][ma][mb]=trois_j_sans_m3(j1,j2,K,m1,mb-j2) ;
      }
    }
  }
}
//peuplement des tableaux des troisjplusmoins (WITH jf !!!)
cout << "peuplement des tableaux des troisjplusmoins (WITH jf !!!)"<< endl;
//type 3 : on calcule avec jf
mamax=2*jfplus;
mbmax=2*jfmoins;
for (int ma=0; ma<=mamax; ma++) {
  float m1=ma-jfplus;
  for (int mb=0; mb<=mbmax; mb++) {
    trois_j_plusmoins_t[3][ma][mb]=trois_j_sans_m3(jfplus,jfmoins,jf,m1,mb-jfmoins);
  }
}
```

The first block compute and store the trois_j_type_t[][][][][] table with the trois_j_sans_m3() function for the 3j-symbols of the type 7 to 9 (dependent of j_f, j_f^+ and j_f^-). The next block give the $\{j_f^+, j_f^-, j_f\}$-symbols.

Computation of intertwiners for the shared-tetrahedra As the boundary part, the code compute the intertwiners for the shared-tetrahedra (which depend of j_f) :

```
// Populating of intertwiners arrays (WITH jf !!!)
cout << "Populating of intertwiners arrays (WITH jf !!!)"<< endl;

// type 4 -> 3j type 7 et 4:
mamax=2*j_face[7][1];
mbmax=2*j_face[7][2];
mcmax=2*j_face[4][1];
Kmax=min(j_face[7][1]+j_face[7][2],j_face[4][1]+j_face[4][2]);
Kmin=abs(j_face[7][1]-j_face[7][2]);
for (int K=Kmin; K<=Kmax ;K++) {
  for (int ma=0; ma<=mamax; ma++) {
    for (int mb=0; mb<=mbmax; mb++) {
      for (int mc=0; mc<=mcmax; mc++){
        intertwiner_t[4][K][ma][mb][mc]=intertwiner(7,4,K,ma,mb,mc);
      }
    }
  }
```

```
}
// type 5 -> 3j type 8 et 5:
mamax=2*j_face[8][1];
mbmax=2*j_face[8][2];
mcmax=2*j_face[5][1];
Kmax=min(j_face[8][1]+j_face[8][2],j_face[5][1]+j_face[5][2]);
Kmin=abs(j_face[8][1]-j_face[8][2]);
for (int K=Kmin; K<=Kmax ;K++) {
  for (int ma=0; ma<=mamax; ma++) {
    for (int mb=0; mb<=mbmax; mb++) {
      for (int mc=0; mc<=mcmax; mc++){
        intertwiner_t[5][K][ma][mb][mc]=intertwiner(8,5,K,ma,mb,mc);
      }
    }
  }
}
// type 6 -> 3j type 9 et 6:
mamax=2*j_face[9][1];
mbmax=2*j_face[9][2];
mcmax=2*j_face[6][1];
Kmax=min(j_face[9][1]+j_face[9][2],j_face[6][1]+j_face[6][2]);
Kmin=abs(j_face[9][1]-j_face[9][2]);
for (int K=Kmin; K<=Kmax ;K++) {
  for (int ma=0; ma<=mamax; ma++) {
    for (int mb=0; mb<=mbmax; mb++) {
      for (int mc=0; mc<=mcmax; mc++){
        intertwiner_t[6][K][ma][mb][mc]=intertwiner(9,6,K,ma,mb,mc);
      }
    }
  }
}
```

Again, for each block the computation of the intertwiners associate and call the intertwiner() function with their corresponding 3j-symbols type parameter : intertwiner_t[4] → intertwiner(7,4,...), intertwiner_t[5] → intertwiner(8,5,...) etc.

Computation of the fusion coefficient for the shared-tetrahedra We have the computation of the fusion-coefficients :

```
//  Let's go for the iDroit (WITH jf !!!):
cout << "generating the iDroit (WITH jf !!!)"<< endl;

int deuxjf=(int)(2.0*jf);
maxKplus=j0plus+min(jfplus,j0plus);
maxKmoins=j0moins+min(jfmoins,j0moins);
float minKplus=abs(jfplus-j0plus);
float minKmoins=abs(jfmoins-j0moins);

for (int Kplus=minKplus; Kplus<= maxKplus; Kplus++){
  for (int Kmoins=minKmoins; Kmoins<= maxKmoins; Kmoins++){
    for (int ma=0; ma<=deuxjf; ma++){
      for (int mb=0; mb<=deuxj0; mb++){
```

```
          for (int mc=0; mc<=deuxj0; mc++){
            if (abs(ma+mb+mc-jf-deuxj0)>j0) {
              iDroit_type2_t[Kplus][Kmoins][ma][mb][mc]=0;
            } else {
              iDroit_type2_t[Kplus][Kmoins][ma][mb][mc]=iDroit_type2(Kplus,Kmoins,ma,mb,mc)
            }
          }
        }
      }
    }
  }
  // And the fusion-coefficients (WITH jf !!!)
  cout << "And the fusion-coefficients (WITH jf !!!)"<< endl;

  maxKplus=j0plus+min(jfplus,j0plus);
  maxKmoins=j0moins+min(jfmoins,j0moins);
  maxK=j0+min(jf,j0);

  minKplus=abs(jfplus-j0plus);
  minKmoins=abs(jfmoins-j0moins);
  float minK=abs(jf-j0);
  for (int Kplus=minKplus; Kplus<= maxKplus; Kplus++){
    for (int Kmoins=minKmoins; Kmoins<= maxKmoins; Kmoins++){
      for (int K=minK; K<= maxK; K++){
        iCourbe_type2_t[Kplus][Kmoins][K]=iCourbe_type2(Kplus,Kmoins,K);
      }
    }
  }
```

The first block use the iDroit_type2() function for create the iDroit for the shared-tetrahedra, and store them in the iDroit_type2_t[][][][][][][] table. The second block use the previously computed iDroit_type2_t[][][][][][][] values and compute the fusion-coefficients in the iCourbe_type2_t[][][][] array.

Computation of the $15j^{\pm}$-symbols Now we have the all intertwiners, from boundary and internal quantum geometry for a specific value of j_f, so we can compute the $15j^{\pm}$-symbols :

```
  // Let's go for the 15j :
  cout << "Let's go for the  15j plus "<< endl;

  maxKplus=2.0*min(jplus,j0plus);
  float maxK4plus=j0plus+min(jfplus,j0plus);
  float minK4plus=abs(jfplus-j0plus);
  int intmaxKplus=(int)(maxKplus);

  #pragma omp parallel for
  for (int K2=0; K2<=intmaxKplus; K2++){
    for (int K1=0; K1<=maxKplus; K1++){
      for (int K3=0; K3<K1; K3++){
        for (int K4=minK4plus; K4<=maxK4plus; K4++){
          for (int K5=minK4plus; K5<K4; K5++){
            function_15j_plus_t[K1][K2][K3][K4][K5]
              =function_15j_plus(K1,K2,K3,K4,K5);
```

```
          function_15j_plus_t[K1][K2][K3][K5][K4]
            =pow(-1.0,K1+K2+K3)*function_15j_plus_t[K1][K2][K3][K4][K5];
          function_15j_plus_t[K3][K2][K1][K4][K5]
            =pow(-1.0,K1+K2+K3+K4+K5)*function_15j_plus_t[K1][K2][K3][K4][K5];
          function_15j_plus_t[K3][K2][K1][K5][K4]
            =pow(-1.0,K4+K5)*function_15j_plus_t[K1][K2][K3][K4][K5];
        }
        function_15j_plus_t[K1][K2][K3][K4][K4]
          =function_15j_plus(K1,K2,K3,K4,K4);
        function_15j_plus_t[K3][K2][K1][K4][K4]
          =function_15j_plus_t[K1][K2][K3][K4][K4];
      }
    }
    for (int K4=minK4plus; K4<=maxK4plus; K4++){
      for (int K5=minK4plus; K5<K4; K5++){
        function_15j_plus_t[K1][K2][K1][K4][K5]
          =function_15j_plus(K1,K2,K1,K4,K5);
        function_15j_plus_t[K1][K2][K1][K5][K4]
          =pow(-1.0,K2)*function_15j_plus_t[K1][K2][K1][K4][K5];
      }
      function_15j_plus_t[K1][K2][K1][K4][K4]
        =function_15j_plus(K1,K2,K1,K4,K4);
    }
  }
}
#pragma omp parallel for
for (int K2=0; K2<=intmaxKplus; K2++){
  for (int K1=0; K1<=maxKplus; K1++){
    for (int K3=0; K3<=K1; K3++){
      for (int K4=minK4plus; K4<=maxK4plus; K4++){
        for (int K5=minK4plus; K5<=K4; K5++){
          if(abs(function_15j_plus_t[K1][K2][K3][K4][K5])<1.0e-20){
            function_15j_plus_t[K1][K2][K3][K4][K5]=0;
            function_15j_plus_t[K1][K2][K3][K5][K4]=0;
            function_15j_plus_t[K3][K2][K1][K4][K5]=0;
            function_15j_plus_t[K3][K2][K1][K5][K4]=0;
          }
        }
      }
    }
  }
}


cout << "Let's go for the 15j minus "<< endl;

maxKmoins=2.0*min(jmoins,j0moins);
float maxK4moins=j0moins+min(jfmoins,j0moins);
float minK4moins=abs(jfmoins-j0moins);
int intmaxKmoins=(int)(maxKmoins);

#pragma omp parallel for
```

```
for (int K2=0; K2<=intmaxKmoins; K2++){
  for (int K1=0; K1<=maxKmoins; K1++){
    for (int K3=0; K3<K1; K3++){
      for (int K4=minK4moins; K4<=maxK4moins; K4++){
        for (int K5=minK4moins; K5<K4; K5++){
          function_15j_moins_t[K1][K2][K3][K4][K5]
            =function_15j_moins(K1,K2,K3,K4,K5);
          function_15j_moins_t[K1][K2][K3][K5][K4]
            =pow(-1.0,K1+K2+K3)*function_15j_moins_t[K1][K2][K3][K4][K5];
          function_15j_moins_t[K3][K2][K1][K4][K5]
            =pow(-1.0,K1+K2+K3+K4+K5)*function_15j_moins_t[K1][K2][K3][K4][K5];
          function_15j_moins_t[K3][K2][K1][K5][K4]
            =pow(-1.0,K4+K5)*function_15j_moins_t[K1][K2][K3][K4][K5];
        }
      }
    }
    for (int K4=minK4moins; K4<=maxK4moins; K4++){
      for (int K5=minK4moins; K5<K4; K5++){
        function_15j_moins_t[K1][K2][K1][K4][K5]
          =function_15j_moins(K1,K2,K1,K4,K5);
        function_15j_moins_t[K1][K2][K1][K5][K4]
          =pow(-1.0,K2)*function_15j_moins_t[K1][K2][K1][K4][K5];
      }
      function_15j_moins_t[K1][K2][K1][K4][K4]
        =function_15j_moins(K1,K2,K1,K4,K4);
    }
  }
}
#pragma omp parallel for
for (int K2=0; K2<=intmaxKmoins; K2++){
  for (int K1=0; K1<=maxKmoins; K1++){
    for (int K3=0; K3<=K1; K3++){
      for (int K4=minK4moins; K4<=maxK4moins; K4++){
        for (int K5=minK4moins; K5<=K4; K5++){
          if(abs(function_15j_moins_t[K1][K2][K3][K4][K5])<1.0e-20){
            function_15j_moins_t[K1][K2][K3][K4][K5]=0;
            function_15j_moins_t[K1][K2][K3][K5][K4]=0;
            function_15j_moins_t[K3][K2][K1][K4][K5]=0;
            function_15j_moins_t[K3][K2][K1][K5][K4]=0;
          }
        }
      }
    }
  }
}
```

The first block compute the $15j^+$-symbols, the second compute the $15j^-$-symbols. For each we have two parts :

The first compute the corresponding the 15j-symbols using the function_15j function and the symmetry properties :

$$\begin{aligned}(K_1,K_2,K_3,K_4,K_5;j,j_0,j_f) &= (-1)^{K_1+K_2+K_3+K_4+K_5}\,(K_3,K_2,K_1,K_4,K_5;j,j_0,j_f)\\ &= (-1)^{K_4+K_5}\,(K_3,K_2,K_1,K_5,K_4;j,j_0,j_f)\end{aligned} \tag{209}$$

for store the all function_15j table without going through all the set of K parameter and save more calculation time.

The second part take the values of the table which are smaller than 1.0×10^{-20} and replace them by 0, because they correspond in reality to null values. That was found in the development of the code, where several methods was used for compute the 15j-symbols, and show that the values from different methods was very precisely the same except when they are smaller than 1.0×10^{-20} (in these cases the values was always the same size order and similar, but not equal). So we can conclude that these values are nulls, and that the code have a precision at least 1.0×10^{-20} for the values of the 15j-symbols.

To save more calculation time, we use the parallelization library OpenMP with the "#pragma omp parallel for" lines for the loops of each parts, that will allow to make the computation of the all 15j-symbols over the all processor of the machine.

Computation of the 15j-$SO(4)$-symbols At this point, we have the all objects needed for compute the 15j-$SO(4)$-symbols ! The code will load the values of 15j-$SO(4)$-symbols from data-file (if it exist) and compute the missing 15j-$SO(4)$-symbols values before to store them in the memory and in the data-file :

```
// remains only the 15jso4
cout << "Remains only the 15jso4" << endl;

maxK=2.0*min(j0,j);
float maxK4=j0+min(j0,jf);
float minK4=abs(jf-j0);

ostringstream j0s, js, jfs;//create a stringstream
j0s << j0;//add number to the stream
js << j;
jfs << jf;

load_15j_so4(j,j0,jf);

string name = "simplex_walking_j0=";
name += j0s.str();
name += "_j=";
name += js.str();
name += "_jf=";
name += jfs.str();
name += ".txt";
ofstream results_so4_w(name.c_str(), ios::out|ios::app);  //open the 15j so4 file
results_so4_w.precision(64);
omp_lock_t writelock;
omp_init_lock(&writelock);
int intmaxK=(int)(maxK);
#pragma omp parallel for
for (int K2=0; K2<=intmaxK; K2++){
  for (int K1=0; K1<=maxK; K1++){
    for (int K3=0; K3<K1; K3++){
      for (int K4=minK4; K4<=maxK4; K4++){
        for (int K5=minK4; K5<K4; K5++){
          if ( t15j_so4_present[K1][K2][K3][K4][K5] == true ) {
            function_15j_so4_t[K1][K2][K3][K4][K5]
              =t15j_so4_loaded[K1][K2][K3][K4][K5];
```

```
      } else {
        function_15j_so4_t[K1][K2][K3][K4][K5]
          =function_15j_so4(K1,K2,K3,K4,K5);
          if(abs(function_15j_so4_t[K1][K2][K3][K4][K5])<1.0e-20)
            {function_15j_so4_t[K1][K2][K3][K4][K5]=0;}
        omp_set_lock(&writelock);
        results_so4_w<<K1<<" "<<K2<<" "<<K3<<" "<<K4<<" "<<K5<<"\t"
                      <<function_15j_so4_t[K1][K2][K3][K4][K5]<<endl;
        omp_unset_lock(&writelock);
      }
      function_15j_so4_t[K1][K2][K3][K5][K4]
        =pow(-1.0,K1+K2+K3)*function_15j_so4_t[K1][K2][K3][K4][K5];
      function_15j_so4_t[K3][K2][K1][K4][K5]
        =pow(-1.0,K1+K2+K3+K4+K5)*function_15j_so4_t[K1][K2][K3][K4][K5];
      function_15j_so4_t[K3][K2][K1][K5][K4]
        =pow(-1.0,K4+K5)*function_15j_so4_t[K1][K2][K3][K4][K5];
    }
    if ( t15j_so4_present[K1][K2][K3][K4][K4] == true ) {
      function_15j_so4_t[K1][K2][K3][K4][K4]
        =t15j_so4_loaded[K1][K2][K3][K4][K4];
    } else {
      function_15j_so4_t[K1][K2][K3][K4][K4]
        =function_15j_so4(K1,K2,K3,K4,K4);
        if(abs(function_15j_so4_t[K1][K2][K3][K4][K4])<1.0e-20)
          {function_15j_so4_t[K1][K2][K3][K4][K4]=0;}
      omp_set_lock(&writelock);
      results_so4_w<<K1<<" "<<K2<<" "<<K3<<" "<<K4<<" "<<K4<<"\t"
                    <<function_15j_so4_t[K1][K2][K3][K4][K4]<<endl;
      omp_unset_lock(&writelock);
    }
    function_15j_so4_t[K3][K2][K1][K4][K4]
      =function_15j_so4_t[K1][K2][K3][K4][K4];
  }
}
for (int K4=minK4; K4<=maxK4; K4++){
  for (int K5=minK4; K5<K4; K5++){
    if ( t15j_so4_present[K1][K2][K1][K4][K5] == true ) {
      function_15j_so4_t[K1][K2][K1][K4][K5]
        =t15j_so4_loaded[K1][K2][K1][K4][K5];
    } else {
      function_15j_so4_t[K1][K2][K1][K4][K5]
        =function_15j_so4(K1,K2,K1,K4,K5);
        if(abs(function_15j_so4_t[K1][K2][K1][K4][K5])<1.0e-20)
          {function_15j_so4_t[K1][K2][K1][K4][K5]=0;}
      omp_set_lock(&writelock);
      results_so4_w<<K1<<" "<<K2<<" "<<K1<<" "<<K4<<" "<<K5<<"\t"
                    <<function_15j_so4_t[K1][K2][K1][K4][K5]<<endl;
      omp_unset_lock(&writelock);
    }
    function_15j_so4_t[K1][K2][K1][K5][K4]
      =pow(-1.0,K2)*function_15j_so4_t[K1][K2][K1][K4][K5];
  }
```

```
        if ( t15j_so4_present[K1][K2][K1][K4][K4] == true ) {
          function_15j_so4_t[K1][K2][K1][K4][K4]
            =t15j_so4_loaded[K1][K2][K1][K4][K4];
        } else {
          function_15j_so4_t[K1][K2][K1][K4][K4]
            =function_15j_so4(K1,K2,K1,K4,K4);
            if(abs(function_15j_so4_t[K1][K2][K1][K4][K4])<1.0e-20)
              {function_15j_so4_t[K1][K2][K1][K4][K4]=0;}
          omp_set_lock(&writelock);
          results_so4_w<<K1<<" "<<K2<<" "<<K1<<" "<<K4<<" "<<K4<<"\t"
                        <<function_15j_so4_t[K1][K2][K1][K4][K4]<<endl;
          omp_unset_lock(&writelock);
        }
      }
    }
  }
  omp_destroy_lock(&writelock);
  results_so4_w.close();   //close the 15j so4 file
```

After create the name of the hypothetical 15j-$SO(4)$-symbols data-file in function of the specific value of (j, j_0 and) j_f, "simplex_walking_j0=j_0_j=j_jf=j_f.txt", the code use the load_15j_so4(j,j0,jf) function for get the precalculated values from the data-file (via the t15j_so4_loaded[][][][][][][] table inside the loading function) and get the list of them (via the t15j_so4_present[][][][][][][] Boolean table). As the 15j-symbols parts, we use the same tricks and logic for compute the 15j-$SO(4)$-symbols with the symmetries :

$$\begin{aligned}[K_1,K_2,K_3,K_4,K_5;j,j_0,j_f] &= (-1)^{K_1+K_2+K_3+K_4+K_5}\,[K_3,K_2,K_1,K_4,K_5;j,j_0,j_f] \\ &= (-1)^{K_4+K_5}\,[K_3,K_2,K_1,K_5,K_4;j,j_0,j_f]\end{aligned} \tag{210}$$

Here is really important, because the execution of each function_15j are long (~1sec for $j\sim 10$), so the use of these functions for all the set of K (~$(2j+1)^5$ steps) are VERY long ! Of course, for each set of K, before to compute the 15j-$SO(4)$-symbol associated, the code check if we have already the computed value in the t15j_so4_loaded[][][][][][][] table (due to the t15j_so4_present[K1][K2][K3][K4][K5] == true condition) and use it if that is the case. The loops over the K parameter get the all 15j-$SO(4)$-symbols and store them in the function_15j_so4_t table. Again, we use the OpenMP library line "#pragma omp parallel for" for parallelize the process, and use the specific omp_set_lock() and omp_unset_lock() functions for write correctly the missing 15j-$SO(4)$-symbols in the data-file.

Computation of the coherent states for the shared-tetrahedra For the shared-tetrahedra, with the gauge choice and closure condition, we have to compute the quantities :

$$\left\langle i^J | j_f, j_0, \theta_f, \Phi_f \right\rangle = \sum_m i^J_{j_1 m_2 m_3 m_4} D^{j_2}_{m_2 j_2}(\theta_2, 0)\, D^{j_3}_{m_3 j_3}(\theta_3, \phi_3)\, D^{j_4}_{m_4 j_4}(\theta_4, \phi_4)\, [j, j_0, \theta_f, \Phi_f] \tag{211}$$

with :

$$
\begin{aligned}
\theta_2 &= \theta_f \Rightarrow \left(A_f \coloneqq \sqrt{j_0^2 + j_f^2 + 2 j_0 j_f \cos\theta_f}\right) \\
\theta_3 &= \arccos\left(\frac{\cos(\Phi)\sqrt{4j_0^2 - A_f^2}\sqrt{2A_f^2\left(j_f^2 + j_0^2\right) - \left(j_f^2 - j_0^2\right)^2 - A_f^4} - A_f\left(A_f^2 + j_f^2 - j_0^2\right)}{4 j_f j_0 A_f}\right) \\
\theta_4 &= \arccos\left(\frac{-\cos(\Phi)\sqrt{4j_0^2 - A_f^2}\sqrt{2A_f^2\left(j_f^2 + j_0^2\right) - \left(j_f^2 - j_0^2\right)^2 - A_f^4} - A_f\left(A_f^2 + j_f^2 - j_0^2\right)}{4 j_f j_0 A_f}\right) \\
\phi_3 &= \arccos\left(\frac{\sin^2\theta_4 - \sin^2\theta_2 - \sin^2\theta_3}{2\sin\theta_2\sin\theta_3}\right) \\
\phi_4 &= 2\pi - \arccos\left(\frac{\sin^2\theta_3 - \sin^2\theta_2 - \sin^2\theta_4}{2\sin\theta_2\sin\theta_4}\right)
\end{aligned}
$$

As the coherent states part for boundary tetrahedra, we use the same sort of code for the internal tetrahedra states :

```
cout << "Next, the coherent shared states (WITH jf !!!)"<< endl;

complex<long double> iJd_interne[(int)(j0+min(j0,jf)+1.0)][51][51];

name = "iJd_interne_jf=";
name += jfs.str();
name += ".txt";
ofstream results3(name.c_str(), ios::out | ios::trunc);

for(int t2=0;t2<=50;t2++){
  float PHIf=(1+2*t2)*M_PI/102.0;
    //facteur disymétrie - angle entre (n1,n2)^(n3,n4) - variable partenaire de K
  for(int n2=0;n2<=50;n2++){
    float theta2=M_PI*(1+2*n2)/102.0;
      float Af=sqrt(j0*j0+jf*jf+2.0*j0*jf*cos(theta2));
    float theta3=acos((cos(PHIf)*sqrt(4.0*j0*j0-Af*Af)*sqrt(2.0*Af*Af*(jf*jf+j0*j0)
                        -pow(jf*jf-j0*j0,2)-Af*Af*Af*Af) - Af*(Af*Af+jf*jf-j0*j0)
                        )/(4.0*Af*j0*jf));
    float theta4=acos((-cos(PHIf)*sqrt(4.0*j0*j0-Af*Af)*sqrt(2.0*Af*Af*(jf*jf+j0*j0)
                        -pow(jf*jf-j0*j0,2)-Af*Af*Af*Af) - Af*(Af*Af+jf*jf-j0*j0)
                        )/(4.0*Af*j0*jf));
    float phi3=acos((pow(sin(theta4),2) - pow(sin(theta2),2) - pow(sin(theta3),2))
                      /(2.0*sin(theta2)*sin(theta3)));
    float phi4=2.0*M_PI
                -acos((pow(sin(theta3),2) - pow(sin(theta2),2) - pow(sin(theta4),2))
                      /(2.0*sin(theta2)*sin(theta4)));

    // Creation of the iJd_externe (WITH jf !!!)
    long double norm_iJd_interne=0;
    for(int K=minK4;K<=maxK4;K++){
      iJd_interne[K][n2][t2]=0;
```

```
        for(int mb=0;mb<=deuxj0;mb++){
          for(int mc=0;mc<=deuxj0;mc++){
            int md=3.0*j0-jf-mb-mc;
            if((md>=0)&&(md<=deuxj0)){
              iJd_interne[K][n2][t2] += intertwiner_mem(4,K,deuxjf,mb,mc)
                                        *d(mb,deuxj0,theta2,0)
                                        *d(mc,deuxj0,theta3,phi3)
                                        *d(md,deuxj0,theta4,phi4);
            }
          }
        }
      norm_iJd_interne+=abs(iJd_interne[K][n2][t2])*abs(iJd_interne[K][n2][t2]);
      }

      if(norm_iJd_interne!=0){
        for(int K=minK4;K<=maxK4;K++){
          complex<long double> temp(sqrt((deuxjf+1.0)*(deuxj0+1.0))*(deuxj0+1.0),0);
          //complex<long double> temp(1.0/sqrt(norm_iJd_interne),0);
          iJd_interne[K][n2][t2]*=temp;
          results3<<PHIf<<" "<<theta2<<"\t"<<norm_iJd_interne<<"\t"<<K<<"\t"
                  <<abs(iJd_interne[K][n2][t2])<<" "<<arg(iJd_interne[K][n2][t2])<<endl;
        }
        results3<<endl;
      }
    }
    results3<<endl;
  }
  results3.close();
```

We define the iJd_interne[(int)(j0+min(j0,jf)+1.0)][51][51] array and the "iJd_interne_jf=j_f.txt" data-file which will contain the coherent tetrahedra results. We have the loops over the two discretized parameters :

$$\Phi_f = \frac{1+2t_2}{102} \times \pi \text{ for } t_2 = 0, 1, \ldots, 50 \tag{212}$$

$$\theta_f = \frac{1+2n_2}{102} \times \pi \text{ for } n_2 = 0, 1, \ldots, 50 \tag{213}$$

$$\left(A_f \coloneqq \sqrt{j_0^2 + j_f^2 + 2 j_0 j_f \cos\theta_f}\right) \tag{214}$$

We store, temporarily, the iJd_interne[][][][] table with the coherent state :

$$\text{iJd\_interne[K][n2][t2]} = \left\langle i^{\text{K}} | j_f, j_0, \theta_f[\text{n2}], \Phi_f[\text{t2}] \right\rangle \tag{215}$$

and compute, on the fly, the norm of the state : norm_iJd_interne $= \sum_{\text{K}} \left| \left\langle i^{\text{K}} | j_f, j_0, \theta_f[\text{n2}], \Phi_f[\text{t2}] \right\rangle \right|^2$. The next step consist just to renormalize the non-null states (norm_iJd_interne $\neq 0$) with the j-representation and store the the iJd_interne[][][][] with this renormalization :

$$\text{iJd\_interne[K][n2][t2]} \to \text{iJd\_interne[K][n2][t2]} = \sqrt{(2j_f+1)(2j_0+1)^3} \left\langle i^{\text{K}} | j_f, j_0, \theta_f[\text{n2}], \Phi_f[\text{t2}] \right\rangle \tag{216}$$

We have also a commented line that you can uncomment if you would the another possible renormalization :

$$\text{iJd\_interne[K][n2][t2]} = \frac{\left\langle i^{\text{K}} | j_f, j_0, \theta_f[\text{n2}], \Phi_f[\text{t2}] \right\rangle}{\sqrt{\sum_{\text{K'}} \left| \left\langle i^{\text{K'}} | j_f, j_0, \theta_f[\text{n2}], \Phi_f[\text{t2}] \right\rangle \right|^2}} \tag{217}$$

Computation of the coherent transition amplitude for the 4-simplices We have the all objects for create the coherent 4-simplex :

```
// Computation of the 15j so4 semi-coherent
cout << "Créations des 4-simplex semi-cohérent (WITH jf !!!)"<< endl;
complex<long double> interm1(0,0);
complex<long double> interm2(0,0);
complex<long double> interm3(0,0);
for(int t=0; t<=50; t++){
  for(int n=0; n<=50; n++){
    for (int K4=minK4; K4<=maxK4; K4++){
      for (int K5=minK4; K5<=maxK4; K5++){
        complex<long double> resultat(0,0);
        for (int K3=0; K3<=maxJ; K3++){
          interm3=iJd_externe[K3][n][t];
          for (int K2=0; K2<=maxJ; K2++){
            interm2=interm3*iJd_externe[K2][n][t];
            for (int K1=0; K1<=maxJ; K1++){
              resultat+=function_15j_so4_mem(K1,K2,K3,K4,K5)*interm2*iJd_externe[K1][n][t

            }
          }
        }
        so4semicoherent[n][t][K4][K5]=resultat;
      }
    }
  }
}


// Computation of the 15j so4 coherent
cout << "Créations des 4-simplex cohérent (WITH jf !!!)"<< endl;
for(int t=0; t<=50; t++){
  //float PHI=(1+2*t)*M_PI/102.0;
  for(int n=0; n<=50; n++){
    //float A=2.0*(1+2*n)*min(j0,j)/102.0;
    for(int n2=0;n2<=50;n2++){
      for(int t2=0; t2<=50; t2++){
        complex<long double> resultat1(0,0);
        for (int K4=minK4; K4<=maxK4; K4++){
          for (int K5=minK4; K5<=maxK4; K5++){
            resultat1+=so4semicoherent[n][t][K4][K5]*iJd_interne[K4][n2][t2]*conj(iJd_int
          }
        }
        so4coherent[(int)(jf)][n][t][n2][t2]=resultat1;
      }
    }
  }
}
```

The first block will combine the boundary tetrahedra coherent states with the 15j-$SO(4)$-symbols for built the semi-coherent 4-simplex :

$$\begin{aligned}\text{so4semicoherent[n][t][K4][K5]} &= \langle \pentagon | j, j_0, j_f, A[\text{n}], \Phi[\text{t}]; i^{\text{K4}} i^{\text{K5}} \rangle \\ &= \sum_{\text{K1,K2,K3}} [\text{K1,K2,K3,K4,K5}; j, j_0, j_f] \prod_{k=1}^{3} \left((2j+1)\,(2j_0+1)\,\langle i^{\text{Kk}} | j, j_0, A[\text{n}], \Phi[\text{t}] \rangle \right)\end{aligned}$$

And the second block will combine the previous results with the shared-tetrahedra coherent states for built the coherent 4-simplex :

$$\text{so4coherent[(int)}(j_f)\text{][n][t][n2][t2]} = \sum_{\text{K4,K5}} \langle \pentagon | j, j_0, j_f, A[\text{n}], \Phi[\text{t}]; i^{\text{K4}} i^{\text{K5}} \rangle \langle i^{\text{K4}} | j_f, j_0, \theta_f[\text{n2}], \Phi_f[\text{t2}] \rangle \overline{\langle i^{\text{K5}} | j_f, j_0, \theta_f[\text{n2}], \Phi_f[\text{t2}] \rangle} \tag{218}$$

Computation of the coherent transition amplitude for the assembly and end of the f-loop Finally, we can compute the coherent transition amplitude $w_f(j, j_0, j_f, A, \Phi)$ from the semi-coherent 4-simplex table before end the loop over the j_f values :

```
    // Computation of coherent wf !!!!
    cout << "Computation of coherent wf (WITH jf !!!)"<< endl;

    for(int t=0; t<=50; t++){
      for(int n=0; n<=50; n++){
        complex<long double> temp(0,0);
        complex<long double> result(0,0);
        for (int K1=minK4;K1<=maxK4;K1++){
          for (int K2=minK4;K2<=maxK4;K2++) {
            for (int K3=minK4;K3<=maxK4;K3++) {
              temp +=  so4semicoherent[n][t][K1][K2]
                      *so4semicoherent[n][t][K2][K3]
                      *so4semicoherent[n][t][K3][K1];
            }
          }
        }
        result=temp;
      w_coherent_f[(int)(jf)][n][t]=result;      //<------Storage in memory of the result
      }
    }

  }      //<-- end of the if-condition for check if jf+ same type as j0+
}        //<-- end of the jf loop

cout << "////END jf-LOOP////"<< endl;
```

We have the loops over the shared K parameter for compute and store the sum :

$$\begin{aligned}\text{w\_coherent\_f[(int)}(j_f)\text{][n][t]} &= w_f(j, j_0, j_f, A[\text{n}], \Phi[\text{t}]) \\ &= \sum_{\text{K1,K2,K3}} \prod_{N=1}^{3} \langle \pentagon_N | j, j_0, j_f, A[\text{n}], \Phi[\text{t}]; i^{\text{Kk}} i^{\text{Kk'}} \rangle\end{aligned}$$

B.4.3 Writing the results

At the end of the loop over j_f we have the all coherent objects in the memory, so the results writing part of the code can be start. In this section we will expose the write parts of code.

Writing of the full coherent transition amplitude W The first step is to write the results from the full coherent amplitude W of our assembly, but in memory we have just the coherent amplitude w_f. So we need to take the stored values of w_f and sum them over the compatibles j_f for built the amplitude W :

```
////////////////////////////////Writing coherent data////////////////////////////////

cout << "Writing coherent W(A,Phi)"<< endl;

ostringstream j0s, js;//create a stringstream
j0s << j0;//add number to the stream
js << j;
string name = "coherent_W(A,Phi)_j0=";
name += j0s.str();
name += "_j=";
name += js.str();
name += ".txt";
ofstream results4(name.c_str(), ios::out | ios::trunc);  //open the amplitude file

complex<long double> w_coherent[51][51];
for(int t=0; t<=50; t++){
  float PHI=(1+2*t)*M_PI/102.0;
  for(int n=0; n<=50; n++){
    float A=2.0*(1+2*n)*min(j0,j)/102.0;
    w_coherent[n][t]=0;
    for (jf=j0-(int)(j0); jf<=3*j0; jf++){
      float jfplus=jp_approx(Immirzi,jf,1);
      if((jfplus-(int)(jfplus))==(j0plus-(int)(j0plus))) { //checks if jf+ same type as j0+
                                                           //=> calculation required
        complex<long double> djf(2.0*jf+1.0,0);
        w_coherent[n][t]+=djf*w_coherent_f[(int)(jf)][n][t];
      }
    }
    results4<<PHI<<" "<<A<<"\t"<<abs(w_coherent[n][t])<<" "<<arg(w_coherent[n][t])<<endl;
  }
  results4<<endl;
}
results4.close();
```

We create the W data-file with the specification of the j : "coherent_W(A,Phi)_j0=j_0_j=j.txt". We define the w_coherent[][] table which will contain the values of W in case of subsequent calculation, and fill it with the computed values :

$$\begin{aligned} \text{w\_coherent[n][t]} &= W\left(j, j_0, A[\text{n}], \Phi[\text{t}]\right) \\ &= \sum_{j_f} \left(2j_f+1\right) w_f\left(j, j_0, j_f, A[\text{n}], \Phi[\text{t}]\right) \end{aligned}$$

In the code we have the two loops over the parameter "n" and "t", which correspond to the (A, Φ) shape variables, and for each pair we have a loop over the compatibles j_f (given by the if-condition) for perform the summation.

For each j_f we get the stored value of w_f from w_coherent_f[(int)(j_f)][n][t], multiply it by the factor djf $= 2j_f + 1$, and increments the value of w_coherent[n][t]. After the j_f summation, the w_coherent[n][t] table box contain the associated value $W(j, j_0, A[\text{n}], \Phi[\text{t}])$ and is written in the data-file.

Writing of w_f-coherent In this section we will write the amplitude of w_f in function of j_f and given boundary shape parameter (A, Φ). The corresponding table will written in the data-file "coherent_W(jf,A,Phi)_j0=j_0_j=j.txt" :

```
cout << "Writing coherent wf(jf,A,Phi)"<< endl;

name = "coherent_W(jf,A,Phi)_j0=";
name += j0s.str();
name += "_j=";
name += js.str();
name += ".txt";
ofstream results5(name.c_str(), ios::out | ios::trunc);  //open the amplitude file

for(jf=j0-(int)(j0);jf<=3*j0;jf++) {
  float jfplus=jp_approx(Immirzi,jf,1);
  if((jfplus-(int)(jfplus))==(j0plus-(int)(j0plus))) { //checks if jf+ same type as j0+
                                                       //=> calculation required
    for(int t=0; t<=50; t++){
      float PHI=(1+2*t)*M_PI/102.0;
      for(int n=0; n<=50; n++){
        float A=2.0*(1+2*n)*min(j0,j)/102.0;
        if(abs(w_coherent[n][t])>0){ //check if |w_coherent[n][t]|>0
                                     //=> means that the values exist
          //jf   Phi A   "wf(jf,A,Phi)"  "P(jf|A,Phi)"
          results5<<jf<<"\t"<<PHI<<" "<<A<<"\t"
                  <<abs(w_coherent_f[(int)(jf)][n][t])
                  <<" "
                  <<arg(w_coherent_f[(int)(jf)][n][t])
                  <<"\t"
                  <<(1.0+2.0*jf)*abs(w_coherent_f[(int)(jf)][n][t])/abs(w_coherent[n][t])
                  <<" "
                  <<arg(w_coherent_f[(int)(jf)][n][t])-arg(w_coherent[n][t])<<endl;
        }else{
          results5<<jf<<"\t"<<PHI<<" "<<A<<"\t0 0\t0 0"<<endl;
        }
      }
      results5<<endl;
    }
    results5<<endl;
  }
}
results5.close();
```

For each cases, we write the j_f value, the Φ[t], the A[n] and the associated $w_f\,(j, j_0, j_f, A[\text{n}], \Phi[\text{t}])$ (norm and phase) with also the normalized $(2j_f + 1)\frac{w_f(j,j_0,j_f,A[\text{n}],\Phi[\text{t}])}{W(j,j_0,A[\text{n}],\Phi[\text{t}])}$ (norm and phase). That useful if the user want study the raw or normalized values of w_f in terms of norms and phases.

Writing of $\langle W|j, j_0, A, \Phi, \theta_f\rangle$ Similarly to previously, we compute and write the $\langle W|j, j_0, A, \Phi, \theta_f\rangle$ data :

```
cout << "Writing coherent W(t2,A,Phi) cohérent

name = "coherent_W(t2,A,Phi)_j0=";
name += j0s.str();
name += "_j=";
name += js.str();
name += ".txt";
ofstream results6(name.c_str(), ios::out | ios::trunc);  //open the amplitude file

complex<long double> w_thetaf_coherent[51][51][51];
for(int t=0; t<=50; t++){
  float PHI=(1+2*t)*M_PI/102.0;
  for(int n=0; n<=50; n++){
    float A=2.0*(1+2*n)*min(j0,j)/102.0;

    for(int n2=0;n2<=50;n2++){
      float theta2=M_PI*(1+2*n2)/102.0;

      w_thetaf_coherent[n][t][n2]=0;
      for(jf=j0-(int)(j0);jf<=3*j0;jf++) {ls
        float jfplus=jp_approx(Immirzi,jf,1);
        if((jfplus-(int)(jfplus))==(j0plus-(int)(j0plus))){ //checks if jf+ same type j0+
                                                            //=> calculation required
          complex<long double> w_thetaf_coherent_temp
                               =pow(so4coherent[(int)(jf)][n][t][n2][25],3.0);
          if(abs(w_thetaf_coherent_temp)>=0) {
            complex<long double> djf(1.0+2.0*jf,0);
            w_thetaf_coherent[n][t][n2] += djf*w_thetaf_coherent_temp;
          }
        }
      }
      //                  Phi A    theta2   "W(t2,A,Phi)"    "P(t2|A,Phi)"
      results6<<PHI<<" "<<A<<"\t"<<theta2<<"\t"
              <<abs(w_thetaf_coherent[n][t][n2])
              <<" "
              <<arg(w_thetaf_coherent[n][t][n2])
              <<"\t"
              <<abs(w_thetaf_coherent[n][t][n2])/abs(w_coherent[n][t])
              <<" "
              <<arg(w_thetaf_coherent[n][t][n2])-arg(w_coherent[n][t])<<endl;
    }
    results6<<endl;
  }
  results6<<endl;
}
results6.close();
```

As in the writing code section of W, we have the loops over (n,t) for browse the all cases of boundary coherent state, the loop over the n2 for the discrete values of θ_f[n2], and we have a loop over the compatibles j_f for make

the sum :

$$\langle W|j,j_0,A[\mathrm{n}],\Phi[\mathrm{t}],\theta_f[\mathrm{n2}]\rangle = \sum_{j_f}\prod_{N=1}^{3}\left\langle \circlearrowleft_N|j,j_0,j_f,A[\mathrm{n}],\Phi[\mathrm{t}],A_f\left(\theta_f[\mathrm{n2}]\right),\frac{\pi}{2}\right\rangle \tag{219}$$

For each case, in the file "coherent_W(t2,A,Phi)_j0=j_0_j=j.txt", the code write the values of Φ[t], A[n], θ_f[n2] and their associated $\langle W|j,j_0,A[\mathrm{n}],\Phi[\mathrm{t}],\theta_f[\mathrm{n2}]\rangle$ (norm and phase) with also the normalized $\frac{\langle W|j,j_0,A[\mathrm{n}],\Phi[\mathrm{t}],\theta_f\rangle}{\langle W|j,j_0,A[\mathrm{n}],\Phi[\mathrm{t}]\rangle}$ (norm and phase).

Writing of $\langle W|j,j_0,j_f,A,\Phi,\theta_f\rangle$ We have the same for the $\langle W|j,j_0,j_f,A,\Phi,\theta_f\rangle$:

```
cout << "Writing coherent W(jf,t2,A,Phi)"<< endl;

name = "coherent_W(jf,t2,A,Phi)_j0=";
name += j0s.str();
name += "_j=";
name += js.str();
name += ".txt";
ofstream results7(name.c_str(), ios::out | ios::trunc);  //open the amplitude file

for(int t=0; t<=50; t++){
  float PHI=(1+2*t)*M_PI/102.0;
  for(int n=0; n<=50; n++){
    float A=2.0*(1+2*n)*min(j0,j)/102.0;
    for(jf=j0-(int)(j0);jf<=3*j0;jf++) {
      float jfplus=jp_approx(Immirzi,jf,1);
      if((jfplus-(int)(jfplus))==(j0plus-(int)(j0plus))){ //checks if jf+ same type as j0+
                                                          //=> calculation required
        float maxK4=j0+min(j0,jf);
        float minK4=abs(jf-j0);
        for(int n2=0;n2<=50;n2++){
          float theta2=M_PI*(1+2*n2)/102.0;

          results7<<PHI<<" "<<A<<"\t"<<jf<<"\t"<<theta2<<"\t"
                  <<abs(so4coherent[(int)(jf)][n][t][n2][25])
                  <<" "
                  <<arg(so4coherent[(int)(jf)][n][t][n2][25])
                  <<"\t"
                  <<pow(abs(so4coherent[(int)(jf)][n][t][n2][25]),3.0)
                        /abs(w_coherent[n][t])
                  <<" "
                  <<3.0*arg(so4coherent[(int)(jf)][n][t][n2][25])
                       -arg(w_coherent[n][t])
                  <<"\t"
                  <<pow(abs(so4coherent[(int)(jf)][n][t][n2][25]),3.0)
                        /abs(w_coherent_f[(int)(jf)][n][t])
                  <<" "
                  <<3.0*arg(so4coherent[(int)(jf)][n][t][n2][25])
                      -arg(w_coherent_f[(int)(jf)][n][t])<<endl;
        }
        results7<<endl;
      }
```

```
    }
    results7<<endl;
  }
  results7<<endl;
}
results7.close();
```

where we write in the data-file "coherent_W(jf,t2,A,Phi)_j0=j_0_j=j.txt" the values of Φ[t], A[n], j_f, θ_f[n2] and their associated $\langle W|j,j_0,j_f,A[n],\Phi[t],\theta_f[n2]\rangle$ (norm and phase) with the normalized $\frac{\langle W|j,j_0,j_f,A[n],\Phi[t],\theta_f[n2]\rangle}{w_f(j,j_0,j_f,A[n],\Phi[t])}$ (norm and phase).

Writing of $\langle \pentagon_N|j,j_0,j_f,A,\Phi,A_f,\Phi_f\rangle$and end of the main Finally we write the amplitude transition of individual coherent 4-simplex :

```
cout << "écriture des W(jf,Phif,t2,A,Phi) cohérent"<< endl;

name = "coherent_W(jf,Phif,t2,A,Phi)_j0=";
name += j0s.str();
name += "_j=";
name += js.str();
name += ".txt";
ofstream results8(name.c_str(), ios::out | ios::trunc);  //open the amplitude file

for(jf=j0-(int)(j0);jf<=3*j0;jf++) {
  float jfplus=jp_approx(Immirzi,jf,1);
  if((jfplus-(int)(jfplus))==(j0plus-(int)(j0plus))){ //checks if jf+ same type as j0+
                                                      //=> calculation required
    for(int t=0; t<=50; t++){
      float PHI=(1+2*t)*M_PI/102.0;
      for(int n=0; n<=50; n++){
        float A=2.0*(1+2*n)*min(j0,j)/102.0;
        for(int n2=0;n2<=50;n2++){
          float theta2=M_PI*(1+2*n2)/102.0;
            float Af=sqrt(j0*j0+jf*jf+2.0*j0*jf*cos(theta2));
          for(int t2=0; t2<=50; t2++){
            float PHIf=(1+2*t2)*M_PI/102.0;
            results8<<jf<<"\t"<<PHI<<" "<<A<<"\t"<<theta2<<"\t"<<Af<<" "<<PHIf<<"\t"
                    <<abs(so4coherent[(int)(jf)][n][t][n2][t2])
                    <<" "
                    <<arg(so4coherent[(int)(jf)][n][t][n2][t2])<<endl;
          }
          results8<<endl;
        }
        results8<<endl;
      }
      results8<<endl;
    }
    results8<<endl;
  }
}
results8.close();
```

} //<--- END of the MAIN

in the data-file "coherent_W(jf,Phif,t2,A,Phi)_j0=j_0_j=j.txt" with the values of j_f, Φ[t], A[n], θ_f[n2], $A_f(\theta_f)$ (A priory redundant, but useful if you want draw some graphic in function of A_f),Φ_f[t2], and their associated $\langle \pentagon | j, j_0, j_f, A[n], \Phi[t], A_f(\theta_f[n2]), \Phi_f[t2] \rangle$ (norm and phase).