

Computing the Lorentzian 10J Symbol

**Joshua L. Willis
Abilene Christian University**

with

**Dan Christensen
U. of Western Ontario**

11 August 2007

Outline of Talk

- Review computational challenges for Lorentzian spin foam models
- Summary of existing method for the tetrahedral network (6J)
- Recoupling Theory for $SL(2, \mathbb{C})$
- The analogue of the Christensen-Egan algorithm for the Lorentzian 10J
- First attempts at numerical implementation

Spin Foam Models of Quantum Gravity

- Assign partition function to 2-complexes in spacetime

$$Z = \sum_{\text{colorings}} \prod_{\text{faces}} A_F(f) \prod_{\text{edges}} A_E(e) \prod_{\text{vertices}} A_V(v)$$

- Would like to study numerically to investigate phase structure, semiclassical limit
- Definition involves a sum over labellings of 2-complexes, and possibly over different complexes as well.
- But evaluating summand is computationally hard for just **one** labeling, because the vertex amplitude A_V is hard to compute

Why are they computationally hard?

- For Riemannian models ($\text{Spin}(4)$ gauge group) efficient algorithm known that re-expresses $10J$ as a sum over $6J$ symbols (Racah coefficients)
- For Lorentzian models ($\text{SL}(2, \mathbb{C})$ gauge group) no such efficient algorithm was known; $6J$ symbols themselves are hard.
- Why? They are defined by integrals that are **high-dimensional** with **oscillatory integrands**.

$$6J = \int_{H^4} \prod_{i=1}^4 dx_i K_{\rho_1}(x_1, x_2) \cdots K_{\rho_6}(x_3, x_4)$$

$$10J = \int_{H^5} \prod_{i=1}^5 dx_i K_{\rho_1}(x_1, x_2) \cdots K_{\rho_{10}}(x_4, x_5)$$

$$K_{\rho}(x, y) = \frac{\sin(\rho r)}{\rho \sinh r} \quad r = d_{\text{hyp}}(x, y)$$

A Better Algorithm for Lorentzian 6J

- Using group-theoretic techniques, can re-express the Lorentzian 6J as a sum of products of Clebsch-Gordan coefficients for $SL(2, \mathbb{C})$. Analogous to similar formula for $SU(2)$ Racah coefficients:

$$6J \propto \sum_J (2J + 1) C_{00}^{0\rho_1} C_{J0}^{0\rho_5} C_{J0}^{0\rho_4} C_{00}^{0\rho_6} C_{J0}^{0\rho_4} C_{J0}^{0\rho_3} C_{00}^{0\rho_2} C_{J0}^{0\rho_5} C_{J0}^{0\rho_3}$$

- These coefficients can be calculated recursively; thus, very efficiently.
- Much more efficient than direct integration, but convergence can still require many terms.
- Can further speed convergence by using asymptotic form of Clebsch-Gordan coefficients (this is the hard part of both the derivation and coding).

Tet(1,1,1,1,1,1)

Vegas Monte-Carlo Integration

Calls	Value	Time (sec)
10^3	0.041267 ± 21.4%	0.0070
10^4	0.126242 ± 4.03%	0.0430
10^5	0.122350 ± 1.53%	0.4309
10^6	0.118190 ± 0.490%	4.933
10^7	0.117902 ± 0.192%	69.99
10^8	0.118459 ± 0.0532%	436.6

Summation Algorithm

Terms	Value	Time (sec)
10^2	0.118087292	≈ 0.00002
10^3	0.118283570	≈ 0.0002
10^4	0.118306260	0.002
10^5	0.118299794	0.0200
10^6	0.118300212	0.198
10^7	0.118300200	1.98
10^8	0.118300196	19.9

Accelerated Summation Algorithm

Terms	Value	Time (sec)
10^2	0.1183001969	≈ 0.0002
10^3	0.1183001969	≈ 0.002
10^4	0.1183001969	0.0170

Toward the 10J

- The reason for calculating the 6J is to use it in calculating the 10J, hoping that this method is more efficient or more accurate than the direct integration.
- To do this, we need to use recoupling theory for $SL(2, \mathbb{C})$ in the same way that the Riemannian algorithms rely on recoupling theory for $SU(2)$.
- This can be done, and leads to diagrammatic techniques similar to those used for $SU(2)$ spin networks
- Such techniques can be proven using known identities for $SL(2, \mathbb{C})$ matrix elements and Clebsch-Gordan coefficients.

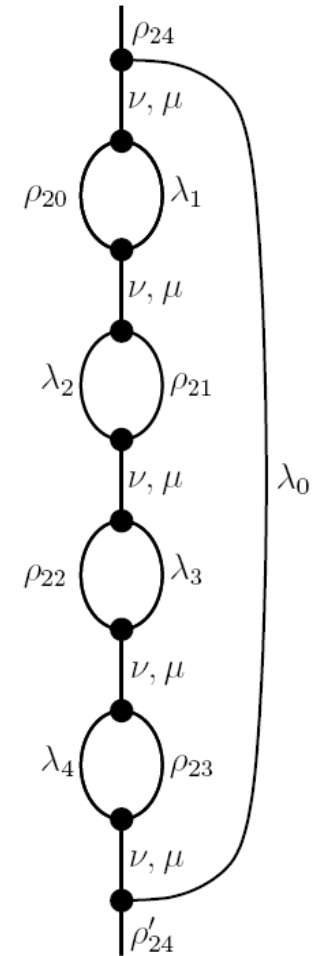
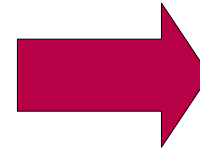
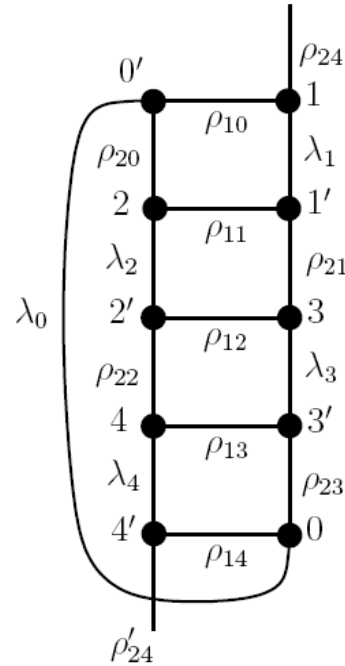
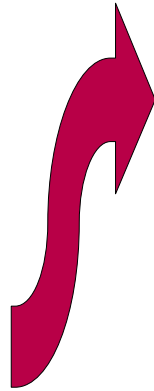
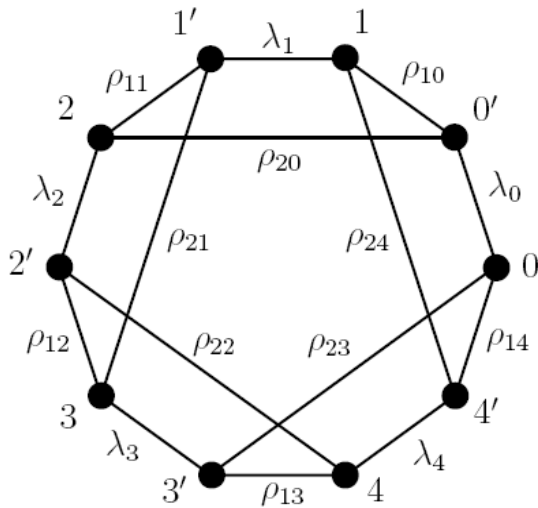
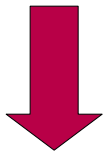
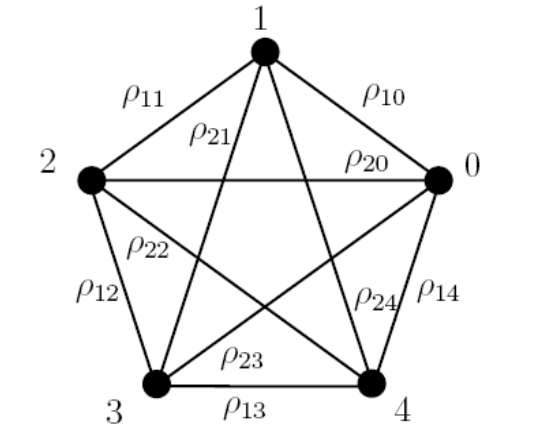
SL(2,C) Recoupling

- After suitably renormalizing the 3-valent vertex, can prove recoupling for **SL(2,C)** spin networks

$$\begin{array}{c} \beta \\ \diagdown \\ \bullet \\ \diagup \\ \alpha \end{array} \text{---} \lambda \text{---} \begin{array}{c} \bullet \\ \diagup \\ \gamma \\ \diagdown \\ \delta \end{array} = \sum_{\nu} \int (\mu^2 + 4\nu^2) d\mu \left\{ \begin{array}{ccc} \alpha & \beta & \nu, \mu \\ \gamma & \delta & \lambda \end{array} \right\} \begin{array}{c} \beta \\ \diagdown \\ \bullet \\ \diagup \\ \gamma \\ \nu, \mu \\ \bullet \\ \diagup \\ \alpha \\ \diagdown \\ \delta \end{array}$$

- Note the appearance of a non-simple representation in the recoupling formula: this is unavoidable and an exactly analogous situation occurs in the Riemannian case, when we consider recoupling for **Spin(4)** spin networks.

Evaluating the 10J



A Formula for 10J's in terms of 6J's

- Combining all of these steps we get a formula analgous to the Christensen-Egan algorithm for the Riemannian **10J**:

$$10J = \int \prod_{i=0}^4 (\lambda_i^2 d\lambda_i) \sum_{\nu} \int (\mu^2 + 4\nu^2) d\mu \quad (\text{Prod of CG's})$$

$$\left\{ \begin{matrix} \rho_{20} & \lambda_0 & \nu, \mu \\ \rho_{24} & \lambda_1 & \rho_{10} \end{matrix} \right\} \left\{ \begin{matrix} \rho_{21} & \lambda_1 & \nu, \mu \\ \rho_{20} & \lambda_2 & \rho_{11} \end{matrix} \right\} \left\{ \begin{matrix} \rho_{22} & \lambda_2 & \nu, \mu \\ \rho_{21} & \lambda_3 & \rho_{12} \end{matrix} \right\} \left\{ \begin{matrix} \rho_{23} & \lambda_3 & \nu, \mu \\ \rho_{22} & \lambda_4 & \rho_{13} \end{matrix} \right\} \left\{ \begin{matrix} \rho_{24} & \lambda_4 & \nu, \mu \\ \rho_{23} & \lambda_0 & \rho_{14} \end{matrix} \right\}$$

- Expresses **10J** as a six-dimensional integral and one-dimensional sum over **6J** symbols.
- Thus, dimension of integral is reduced (from 9 to 6) and experimentation seems to indicate the integrand is in general less oscillatory: when triangle inequalities are violated it decays exponentially.

Numerical Implementation

- First implementation: perform 6-dim integral using importance-sampled Monte-Carlo; ignoring sum over ν for now.
 - Not as fast as direct integral of kernel, since $6J$ still too slow
- Next approach (with Dan Christensen, UWO): Five of six dimensions of integral are λ 's that appear only two at a time in each factor. Can use this to do an iterated integral as matrix multiplication.
 - Mimics Riemannian algorithm.
 - Allows accuracy of one-dimensional methods but without the usual exponential growth in running time of iterated one-dimensional integrals.
 - This method seems faster than direct integral, but we must still deal with sum over ν .

Outlook

- Improvements to be tested:
 - Need to improve asymptotics in $6J \rightarrow$ faster $6J$
 - Try to combine benefits of importance sampling with iterated matrix algorithm: use Gauss-Laguerre quadrature for 1-dim method.
 - Must implement sum over ν .
- Hope to test these improvements in next couple of months.
- Thanks to: NSF grant OISE-0401966; Dan Christensen, Wade Cherrington, and Igor Khavkine for discussions.