From Numerical To Analytical Amplitudes

Giuseppe De Laurentis

with Daniel Maitre

IPPP - Durham University

Table of Content

Colour Ordered Amplitudes

- Relation to the full amplitude @ tree level:

$$\mathcal{A}_n^{tree}(p_i, \lambda_i, a_i) = \ g^{n-2} \sum_{\sigma \in S_n/Z_n} \mathrm{Tr}(T^{a_{\sigma(1)}} \ldots T^{a_{\sigma(n)}}) A_n^{tree}(\sigma(1^{\lambda_1}), \ldots, \sigma(n^{\lambda_n})).$$

- Decomposition in terms of basis integrals:

$$A^{1-loop} = \sum_i d_i I^i_{Box} + \sum_i c_i I^i_{Triangle} + \sum_i b_i I^i_{Bubble} + R$$

Spinor Helicity

- The lowest-laying representations of the Lorentz group are:

(j_-, j_+)	dimensions	name	q.m. field	kinematic variables
(0, 0)	1	scalar	h	m
(0, 1/2)	2	right-handed Weyl spinor	$\chi_{R\,\alpha}$	λ_α
(1/2, 0)	2	left-handed Weyl spinor	$\chi_L^{\dot\alpha}$	$\bar\lambda^{\dot\alpha}$
(1/2, 1/2)	4	rank-two spinor/four vector	$A^\mu / A^{\dot\alpha\alpha}$	$P^\mu / P^{\dot\alpha\alpha}$
(1/2, 0) ⊕ (0, 1/2)	4	bispinor (Dirac spinor)	Ψ	ψ

- Weyl spinors are sufficient to represent the kinematics of massless particles, recall:

$$\det(P^{\dot\alpha\alpha}) = m^2$$

- They are easily related to four momenta:

$$\lambda_\alpha = \begin{pmatrix} \sqrt{p^0 + p^3} \\ \frac{p^1 + ip^2}{\sqrt{p^0+p^3}} \end{pmatrix} \quad \& \quad \lambda^\alpha = \epsilon^{\alpha\beta}\lambda_\beta = \begin{pmatrix} \frac{p^1 + ip^2}{\sqrt{p^0+p^3}} \\ -\sqrt{p^0 + p^3} \end{pmatrix}$$

- Angle and square brackets:

$$\langle ij\rangle = \lambda_i\lambda_j = (\lambda_i)^{\alpha}(\lambda_j)_{\alpha} \qquad [ij] = \bar{\lambda}_i\bar{\lambda}_j = (\bar{\lambda}_i)_{\dot{\alpha}}(\bar{\lambda}_j)^{\dot{\alpha}}$$

- And some relations:

$$s_{ij} = \langle ij\rangle[ji]$$

$$\langle i\mid(j+k)\mid l] = (\lambda_i)^{\alpha}(\not{P}_j + \not{P}_k)_{\alpha\dot{\alpha}}\bar{\lambda}_l^{\dot{\alpha}}$$

$$tr_+(ijkl) = \langle i\,|\,j\,|\,k\,|\,l\,|\,i]$$

- Examples in python:

```
oInvariants = Invariants(6)
pprint(oInvariants.invs_3[:4])
pprint(oInvariants.invs_s[:6])
```

```
[⟨1|(2+3)|1], ⟨1|(2+6)|1], ⟨1|(3+4)|1], ⟨1|(4+5)|1]]
[s_123, s_124, s_125, s_134, s_135, s_145]
```

```
oParticles = Particles(6)
oParticles.fix_mom_cons()
pprint(oParticles.compute("⟨1|2⟩") * oParticles.compute("[2|1]"))
pprint(oParticles.compute("s_12"))
```

```
-5.49884gmp(1024)-0.596308gmp(1024)j
-5.49884gmp(1024)-0.596308gmp(1024)j
```

Single Collinear Limits in Complexified Phase Space

- Collinear limits give us information about the poles of the amplitude:

$$\langle ij\rangle \to \varepsilon, \quad f \to \varepsilon^{\alpha} \;\Rightarrow\; log(f) \to \alpha \cdot log(\varepsilon)$$

$\Rightarrow$ The slope of $log(f)(\varepsilon)$ gives us the type of singularity, if any exists.

- Constructing the phase space:

```
oParticles.randomise_all()
oParticles.set("⟨1|2⟩", 10 ** -30)
oParticles.phasespace_consistency_check(oInvariants.full, silent=False);
```

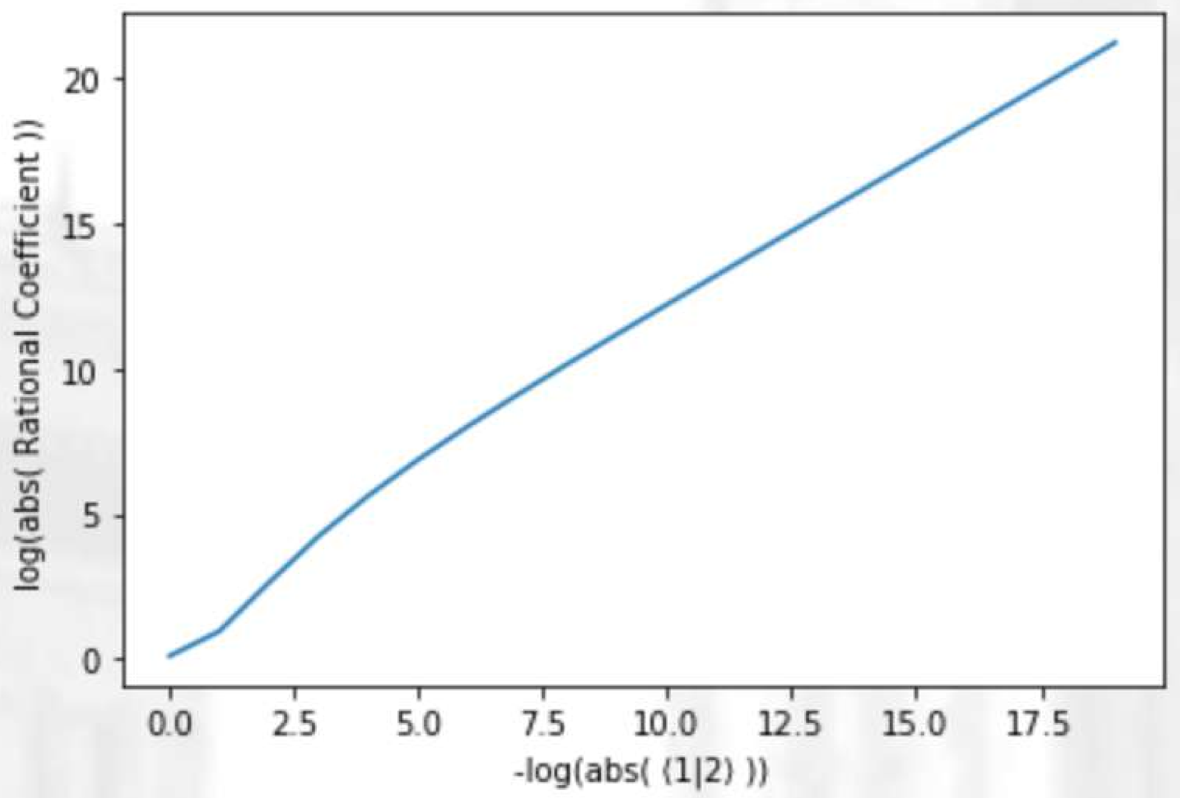
```
------------------------------------ Consistency check ------------------------------------
The largest momentum violation is 5.8668e-308gmp(1024)
The largest on shell violation is 3.20905e-307gmp(1024)
-------------------------------------------------------------------------------------------
⟨1|2⟩ = 1e-30gmp(1024)
...
-------------------------------------------------------------------------------------------
```

- The "..." in the above output hide all O(~1) spinor variables

```
oUnknown = Unknown("pmpmpm", "tree")
```

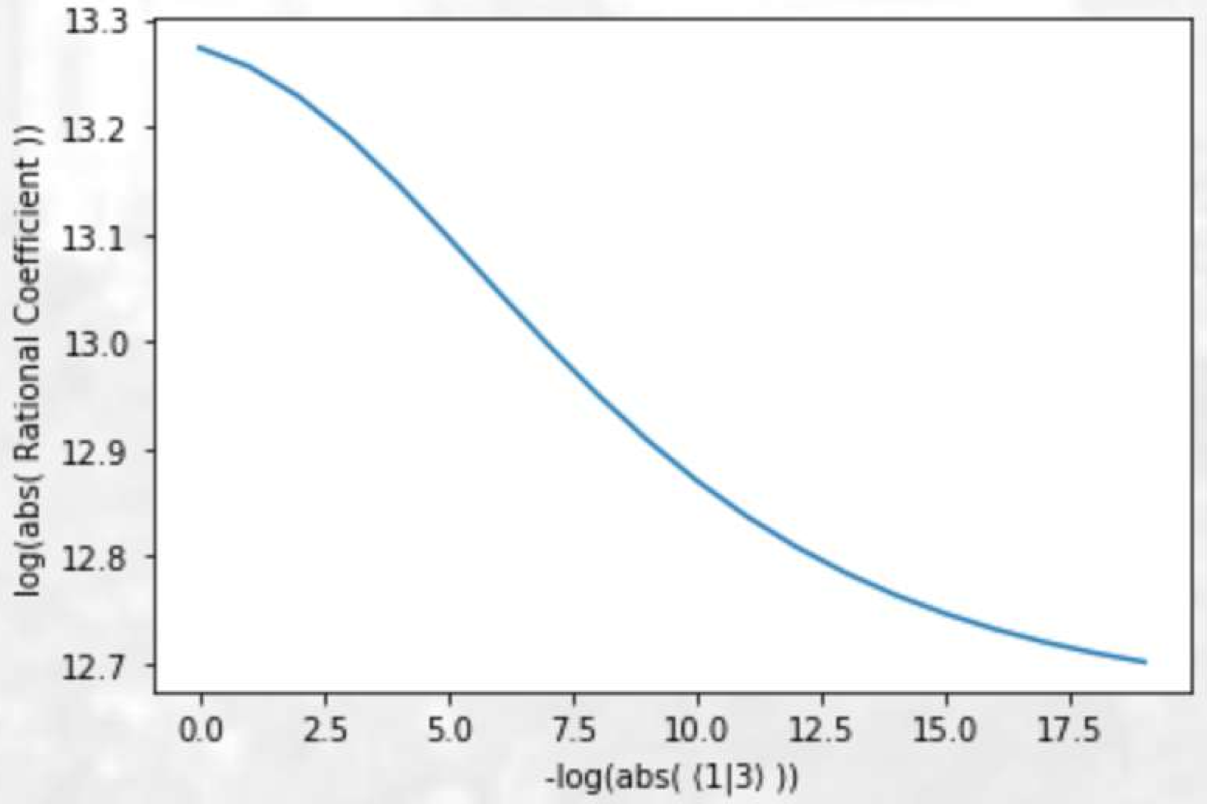
- A Simple Pole

```
oParticles.randomise_all()
x, y= [], []
for i in range(0, 20):
    oParticles.set("⟨1|2⟩", 1.2 ** -i)
    x += [i]
    y += [oUnknown.evaluate(oParticles)]
y = map(log, map(abs, y))
y = [BH.to_double(entry / log(R(1.2))) for entry in y]
plt.plot(x, y)
plt.ylabel('log(abs( Rational Coefficient ))')
plt.xlabel('-log(abs( ⟨1|2⟩ ))')
plt.show()
```



- Not a Pole

```
oParticles.randomise_all()
x, y= [], []
for i in range(0, 20):
    oParticles.set("⟨1|3⟩", 1.2 ** -i)
    x += [i]
    y += [oUnknown.evaluate(oParticles)]
y = map(log, map(abs, y))
y = [BH.to_double(entry / log(R(1.2))) for entry in y]
plt.plot(x, y)
plt.ylabel('log(abs( Rational Coefficient ))')
plt.xlabel('-log(abs( ⟨1|3⟩ ))')
plt.show()
```



- Computing this slope for all invariants gives us the full list of poles and their order:

```
oUnknown.do_single_collinear_limits()
```

```
Finished calculating single scalings. The partial result is:
/⟨1|2⟩[1|2]⟨1|6⟩[1|6]⟨2|3⟩[2|3]⟨3|4⟩[3|4]⟨4|5⟩[4|5]⟨5|6⟩[5|6]s_123s_234s_345

The mass dimension of the unknown is -2.
The phase weights of the unknown are [-2, 2, -2, 2, -2, 2].
The mass dimension of the new unknown is 16.
The phase weights of the new unknown are [-2, 2, -2, 2, -2, 2].
```

- The complexity of the numerator ansatz depends on the mass dimension.
- A mass dimension of ~16 implies an ansatz of $\mathcal{O}(10^4)$ terms.
- But reconstructing the coefficients for such an ansatz would take several days.
- Smaller denominators (i.e. a clearer pole structure) would imply easier numerators.
- This can be achieved by studying double collinear limits.

Double Collinear Limits in Complexified Phase Space

- Constructing the collinear limit works similarly, but the phase space will be less "clean".

```
oParticles.randomise_all()
oParticles.set_pair("⟨1|2⟩", 10 ** -30, "⟨2|3⟩", 10 ** -30)
oParticles.phasespace_consistency_check(oInvariants.full, silent=False);
```

```
----------------------------------- Consistency check -----------------------------------
The largest momentum violation is 1.15464e-304gmp(1024)
The largest on shell violation is 6.75449e-301gmp(1024)
-----------------------------------------------------------------------------------------
⟨1|2⟩ = 1e-30gmp(1024)
⟨2|3⟩ = 1e-30gmp(1024)
⟨2|(1+3)|4] = 1.43471e-29gmp(1024)
⟨2|(1+3)|5] = 1.47577e-29gmp(1024)
⟨2|(1+3)|6] = 2.10149e-29gmp(1024)
⟨2|(1+3)|2] = 3.39855e-29gmp(1024)
⟨2|(1+3)|(1+6)|2⟩ = 1.83765e-28gmp(1024)
⟨2|(1+3)|(3+4)|2⟩ = 1.98953e-28gmp(1024)
⟨1|3⟩ = 2.93287e-28gmp(1024)
⟨3|(1+2)|5] = 7.67695e-28gmp(1024)
⟨3|(1+2)|6] = 1.05234e-27gmp(1024)
⟨3|(1+2)|4] = 1.05266e-27gmp(1024)
⟨1|(2+3)|5] = 3.56209e-27gmp(1024)
⟨1|(2+3)|4] = 3.59802e-27gmp(1024)
⟨1|(2+3)|6] = 6.05674e-27gmp(1024)
s_123 = 8.13908e-27gmp(1024)
⟨3|(1+2)|3] = 8.13919e-27gmp(1024)
⟨1|(2+3)|1] = 8.14502e-27gmp(1024)
[4|(2+3)|(1+2)|6] = 1.29101e-26gmp(1024)
⟨2|(1+3)|(1+5)|6⟩ = 2.48514e-25gmp(1024)
⟨2|(1+3)|(3+5)|4⟩ = 3.59123e-25gmp(1024)
⟨2|(1+3)|(3+4)|5⟩ = 3.59547e-25gmp(1024)
⟨2|(1+3)|(1+6)|5⟩ = 3.60358e-25gmp(1024)
⟨1|(2+3)|(2+6)|1⟩ = 2.51402e-24gmp(1024)
⟨3|(1+2)|(2+4)|3⟩ = 3.59623e-24gmp(1024)
⟨3|(1+2)|(1+5)|6⟩ = 1.31983e-23gmp(1024)
⟨3|(1+2)|(1+6)|5⟩ = 1.80948e-23gmp(1024)
Ω_351 = 1.89267e-23gmp(1024)
⟨1|(2+3)|(3+5)|4⟩ = 1.04247e-22gmp(1024)
```

- Reconstructing the behaviour in the limit involves again the slope of a log plot. We obtain:

```
oUnknown.do_double_collinear_limits()
```

```
Finished calculating pair scalings. They are:

[⟨1|2⟩, [1|2]]: 1.0, 2  → 2
[⟨1|2⟩, ⟨1|6⟩]: 1.0, 30 → 5
[⟨1|2⟩, [1|6]]: 1.0, 3  → 2
[⟨1|2⟩, ⟨2|3⟩]: 1.0, 30 → 5
[⟨1|2⟩, [2|3]]: 1.0, 3  → 2
[⟨1|2⟩, ⟨3|4⟩]: 1.0, 2  → 2
[⟨1|2⟩, [3|4]]: 2.0, 12 → 3
[⟨1|2⟩, ⟨4|5⟩]: 1.0, 10 → 2
[⟨1|2⟩, [4|5]]: 2.0, 2  → 2
[⟨1|2⟩, ⟨5|6⟩]: 1.0, 2  → 2
[⟨1|2⟩, [5|6]]: 2.0, 12 → 3
[⟨1|2⟩, s_123]: 2.0, 10 → 3
[⟨1|2⟩, s_234]: 1.0, 2  → 2
[⟨1|2⟩, s_345]: 2.0, 10 → 3
[[1|2], ⟨1|6⟩]: 1.0, 3  → 2
[[1|2], [1|6]]: 1.0, 30 → 5
[[1|2], ⟨2|3⟩]: 1.0, 3  → 2
[[1|2], [2|3]]: 1.0, 30 → 5
[[1|2], ⟨3|4⟩]: 2.0, 12 → 3
[[1|2], [3|4]]: 1.0, 2  → 2
[[1|2], ⟨4|5⟩]: 2.0, 2  → 2
[[1|2], [4|5]]: 1.0, 10 → 2
[[1|2], ⟨5|6⟩]: 2.0, 12 → 3
[[1|2], [5|6]]: 1.0, 2  → 2
[[1|2], s_123]: 2.0, 10 → 3
[[1|2], s_234]: 1.0, 2  → 2
[[1|2], s_345]: 2.0, 10 → 3
[⟨1|6⟩, [1|6]]: 1.0, 2  → 2
[⟨1|6⟩, ⟨2|3⟩]: 1.0, 2  → 2
[⟨1|6⟩, [2|3]]: 2.0, 12 → 3
[⟨1|6⟩, ⟨3|4⟩]: 1.0, 10 → 2
[⟨1|6⟩, [3|4]]: 2.0, 2  → 2
```

- The first number column is the slope, i.e. 'α'.
- The second and third columns display information regarding how clean the phase space is in that collinear limit.
- Now we have the necessary information to split the denominator and fit smaller numerators.

Fitting the Numerators

- The ansatz can be generated automatically using information from collinear limits or inserted by hand.

```
print(oTerms)
print(oTerms.ansatze_mass_dimensions, end=", ")
print(oTerms.ansatze_phase_weights)
```

```
/⟨1|2⟩⟨2|3⟩[4|5][5|6]⟨1|(2+3)|4]⟨3|(1+2)|6]s_123

[8], [[0, 4, 0, 0, -4, 0]]
```

- The most generic ansatz for the given mass dimension and phase weights is built.

```
for entry in Obtain_Ansatze([8], [[0, 4, 0, 0, -4, 0]])[1][0]:
    print(entry)
```

```
Obtaining ansatze from Daniel's spinor solve with lM, lPW: [8], [[0, 4, 0, 0, -4, 0]].
⟨1|2⟩⟨1|2⟩⟨1|2⟩⟨1|2⟩[1|5][1|5][1|5][1|5]
⟨1|2⟩⟨1|2⟩⟨1|2⟩⟨2|3⟩[1|5][1|5][1|5][3|5]
⟨1|2⟩⟨1|2⟩⟨1|2⟩⟨2|4⟩[1|5][1|5][1|5][4|5]
⟨1|2⟩⟨1|2⟩⟨2|3⟩⟨2|3⟩[1|5][1|5][3|5][3|5]
⟨1|2⟩⟨1|2⟩⟨2|3⟩⟨2|4⟩[1|5][1|5][3|5][4|5]
⟨1|2⟩⟨1|2⟩⟨2|4⟩⟨2|4⟩[1|5][1|5][4|5][4|5]
⟨1|2⟩⟨2|3⟩⟨2|3⟩⟨2|3⟩[1|5][3|5][3|5][3|5]
⟨1|2⟩⟨2|3⟩⟨2|3⟩⟨2|4⟩[1|5][3|5][3|5][4|5]
⟨1|2⟩⟨2|3⟩⟨2|4⟩⟨2|4⟩[1|5][3|5][4|5][4|5]
⟨1|2⟩⟨2|4⟩⟨2|4⟩⟨2|4⟩[1|5][4|5][4|5][4|5]
⟨2|3⟩⟨2|3⟩⟨2|3⟩⟨2|3⟩[3|5][3|5][3|5][3|5]
⟨2|3⟩⟨2|3⟩⟨2|3⟩⟨2|4⟩[3|5][3|5][3|5][4|5]
⟨2|3⟩⟨2|3⟩⟨2|4⟩⟨2|4⟩[3|5][3|5][4|5][4|5]
⟨2|3⟩⟨2|4⟩⟨2|4⟩⟨2|4⟩[3|5][4|5][4|5][4|5]
⟨2|4⟩⟨2|4⟩⟨2|4⟩⟨2|4⟩[4|5][4|5][4|5][4|5]
```

- The linear system of equations for the coefficients is then solved by numerical inversion.
- The size of the ansatz is now 15 instead of O(~10.000).

```
oTerms.fit_numerators();
```

```
Inversion will proceed in s_123^1.0 collinear limit. No symmetry will be used.
The following symmetries will then be appended to the result [(u'234561', True), (u'345612', False)]. Accuracy set to: 24
Obtaining ansatze from Daniel's spinor solve with lM, lPW: [8], [[0, 4, 0, 0, -4, 0]].

Created new matrix of size 15x16
Time elapsed loading the matrix: 0:00:02. Created new matrix of size 15x16.
Time elapsed in row reduction: 0:00:00. 0 elements of the ansatz were redundant!
Nbr dropped redundant: 0, Nbr dropped zero: 10, Nbr dropped total: 10.

Coeff. of ⟨1|2⟩⟨1|2⟩⟨1|2⟩⟨1|2⟩[1|5][1|5][1|5][1|5]: 1*I
Coeff. of ⟨1|2⟩⟨1|2⟩⟨1|2⟩⟨2|3⟩[1|5][1|5][1|5][3|5]: -4*I
Coeff. of ⟨1|2⟩⟨1|2⟩⟨2|3⟩⟨2|3⟩[1|5][1|5][3|5][3|5]: 6*I
Coeff. of ⟨1|2⟩⟨2|3⟩⟨2|3⟩⟨2|3⟩[1|5][3|5][3|5][3|5]: -4*I
Coeff. of ⟨2|3⟩⟨2|3⟩⟨2|3⟩⟨2|3⟩[3|5][3|5][3|5][3|5]: 1*I
This piece correctly removes the singularity ([0])

Refining the fit...
Finished calculating single scalings. The partial result is:
⟨2|(1+3)|5]⁴/⟨1|2⟩⟨2|3⟩[4|5][5|6]⟨1|(2+3)|4]⟨3|(1+2)|6]s_123

The mass dimension of the unknown is -2.
The phase weights of the unknown are [-2, 2, -2, 2, -2, 2].
The mass dimension of the new unknown is 0.
The phase weights of the new unknown are [0, 0, 0, 0, 0, 0].

Inversion will proceed in s_123^1.0 collinear limit. No symmetry will be used.
The following symmetries will then be appended to the result [(u'234561', True), (u'345612', False)]. Accuracy set to: 24
Obtaining ansatze from Daniel's spinor solve with lM, lPW: [0], [[0, 0, 0, 0, 0, 0]].

Created new matrix of size 1x2
Time elapsed loading the matrix: 0:00:00. Created new matrix of size 1x2.
Time elapsed in row reduction: 0:00:00. 0 elements of the ansatz were redundant!
Nbr dropped redundant: 0, Nbr dropped zero: 0, Nbr dropped total: 0.

Coeff. of this whole term is: 1*I
This piece correctly removes the singularity ([0])
```

- Hence the result for the +-+-+- tree amplitude:

```
print(oTerms)
```

```
+1I⟨2|(1+3)|5]⁴/⟨1|2⟩⟨2|3⟩[4|5][5|6]⟨1|(2+3)|4]⟨3|(1+2)|6]s_123
(u'234561', True)
(u'345612', False)
```

- This may seem trivial. A non trivial result follows.

+-+-+- One-Loop Rational

- The analytical expression matches the numerical one:

```
oRationalAnalytical = LoadResults("pmpmpm", "rational", load_partial_results_only=False, silent=True)[0][0]
oRationalNumerical = Unknown("pmpmpm", "rational")
oParticles.randomise_all()
oParticles.fix_mom_cons()
print(oRationalAnalytical.evaluate(oParticles) - oRationalNumerical.evaluate(oParticles))
```

```
7.46485e-300gmp(1024)+1.56522e-299gmp(1024)j
```

```
RationalPDF
```

$$\frac{2/3i\langle 12\rangle^3[15]^3[23]s_{123}}{[45]\langle 1|2+3|1]^2\langle 1|2+3|4]\langle 1|2+3|6]\langle 3|1+2|6]}+$$

$$\frac{-2/3i\langle 12\rangle^3[15]^3[23]\langle 3|1+2|5]}{\langle 13\rangle[45][56]\langle 1|2+3|1]^2\langle 1|2+3|4]\langle 3|1+2|6]}+$$

$$\frac{1i\langle 12\rangle^3[15]^2\langle 23\rangle[23]^2[56]}{[45]\langle 1|2+3|1]\langle 1|2+3|4]\langle 1|2+3|6]^2\langle 3|1+2|6]}+$$

$$\frac{\langle 12\rangle^2[15]^2[23](-1i\langle 12\rangle[15]+2i\langle 23\rangle[35])}{[45]\langle 1|2+3|1]\langle 1|2+3|4]\langle 1|2+3|6]\langle 3|1+2|6]}+$$

$$\frac{1i\langle 12\rangle^3[15]^2[25]\langle 3|1+2|5]}{\langle 13\rangle^2[45][56]\langle 1|2+3|1]\langle 1|2+3|4]\langle 3|1+2|6]}+$$

$$\frac{2i\langle 12\rangle^2[15]^2[35]\langle 3|1+2|5]}{\langle 13\rangle[45][56]\langle 1|2+3|1]\langle 1|2+3|4]\langle 3|1+2|6]}+$$

$$\frac{-1i\langle 12\rangle^2[15]\langle 23\rangle[25]^2\langle 2|1+3|5]}{\langle 13\rangle^2[45][56]\langle 1|2+3|4]\langle 3|1+2|6]s_{123}}+$$

Thank you for your attention!

Questions?