On-Shell Diagrams, Recursion Relations, & Combinatorics

Jacob L. Bourjaily

Cracow School of Theoretical Physics
LVI Course, 2016
A Panorama of Holography

The Niels Bohr International Academy

On-Shell Diagrams, Recursion Relations, & Combinatorics

Jacob L. Bourjaily

Cracow School of Theoretical Physics
LVI Course, 2016
A Panorama of Holography

The Niels Bohr International Academy

Organization and Outline

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

Broadening the Class of Physically Meaningful Functions

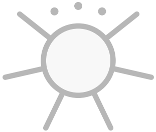
We are interested in the class of functions involving **only** observable quantities

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

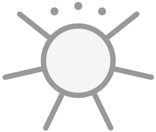
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



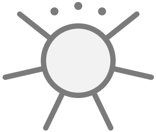
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



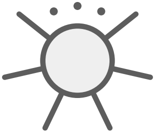
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



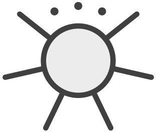
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



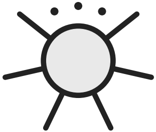
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



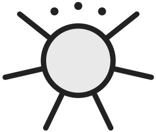
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



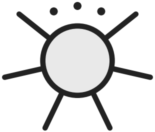
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



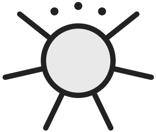
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



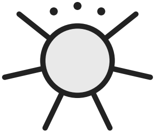
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



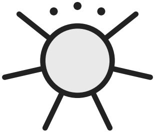
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



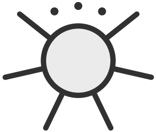
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



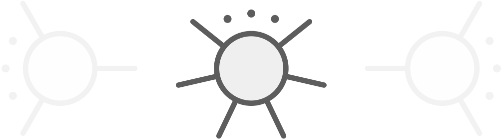
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



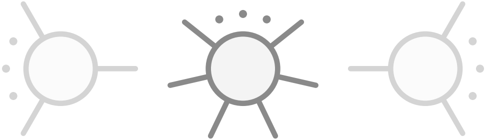
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



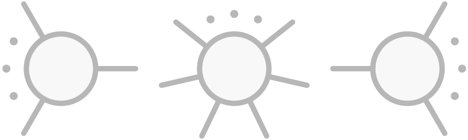
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



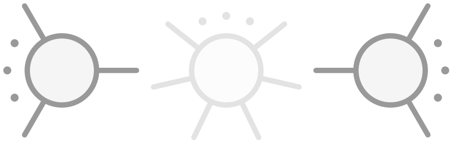
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

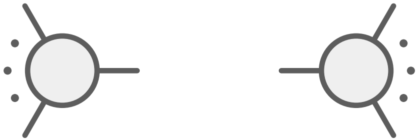


Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



Broadening the Class of Physically Meaningful Functions

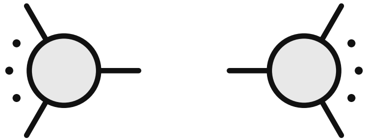
We are interested in the class of functions involving **only** observable quantities

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

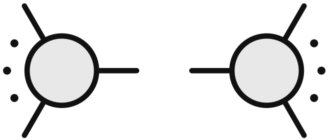


Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

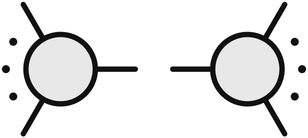
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



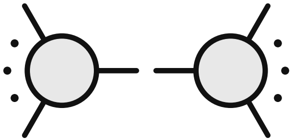
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



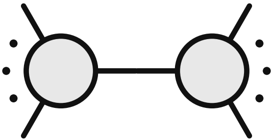
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



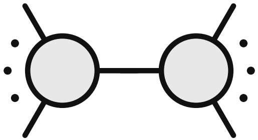
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



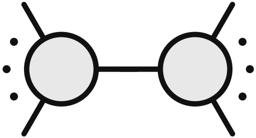
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



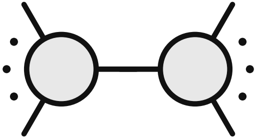
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



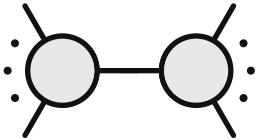
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



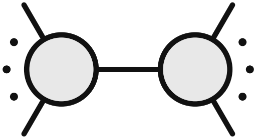
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



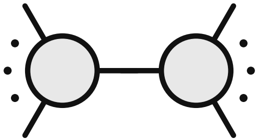
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



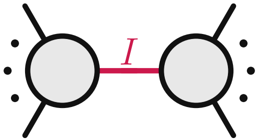
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



Broadening the Class of Physically Meaningful Functions

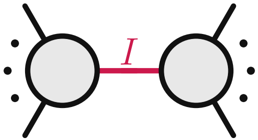
We are interested in the class of functions involving **only** observable quantities



Internal Particles:

Broadening the Class of Physically Meaningful Functions

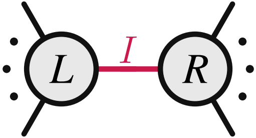
We are interested in the class of functions involving **only** observable quantities



Internal Particles: locality dictates that we multiply each amplitude,

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

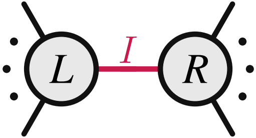


Internal Particles: locality dictates that we multiply each amplitude,

$$\mathcal{A}_L(\ldots, I) \times \mathcal{A}_R(I, \ldots)$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

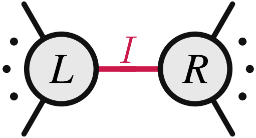


Internal Particles: locality dictates that we multiply each amplitude, and unitarity dictates that we marginalize over unobserved states

$$\mathcal{A}_L(\ldots,I)\times\mathcal{A}_R(I,\ldots)$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

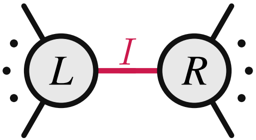


Internal Particles: locality dictates that we multiply each amplitude, and unitarity dictates that we marginalize over unobserved states—integrating over the Lorentz-invariant phase space ("LIPS") for each particle I,

$$\mathcal{A}_L(\ldots,I)\times\mathcal{A}_R(I,\ldots)$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

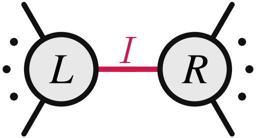


Internal Particles: locality dictates that we multiply each amplitude, and unitarity dictates that we marginalize over unobserved states—integrating over the Lorentz-invariant phase space ("LIPS") for each particle I,

$$\int d^3\text{LIPS}_I \;\; \mathcal{A}_L(\ldots, I)\times\mathcal{A}_R(I, \ldots)$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

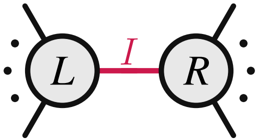


Internal Particles: locality dictates that we multiply each amplitude, and unitarity dictates that we marginalize over unobserved states—integrating over the Lorentz-invariant phase space ("LIPS") for each particle I, and summing over the possible states

$$\int d^3\mathrm{LIPS}_I \;\; \mathcal{A}_L(\ldots,I)\times\mathcal{A}_R(I,\ldots)$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

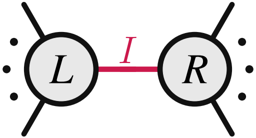


Internal Particles: locality dictates that we multiply each amplitude, and unitarity dictates that we marginalize over unobserved states—integrating over the Lorentz-invariant phase space ("LIPS") for each particle I, and summing over the possible states

$$\sum_{\text{states } I} \int d^3\text{LIPS}_I \;\; \mathcal{A}_L(\ldots, I) \times \mathcal{A}_R(I, \ldots)$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

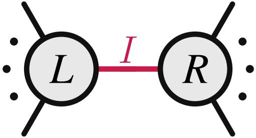


Internal Particles: locality dictates that we multiply each amplitude, and unitarity dictates that we marginalize over unobserved states—integrating over the Lorentz-invariant phase space ("LIPS") for each particle I, and summing over the possible states (helicities, masses, colours, etc.).

$$\sum_{\text{states } I} \int d^3\text{LIPS}_I \;\; \mathcal{A}_L(\ldots, I)\times\mathcal{A}_R(I, \ldots)$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

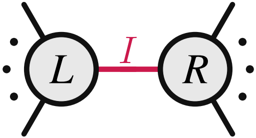


Internal Particles: locality dictates that we multiply each amplitude, and unitarity dictates that we marginalize over unobserved states—integrating over the Lorentz-invariant phase space ("LIPS") for each particle I, and summing over the possible states (helicities, masses, colours, etc.).

$$\sum_{\text{states } I} \int d^3\text{LIPS}_I \;\; \mathcal{A}_L(\ldots, I)\times\mathcal{A}_R(I,\ldots)$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

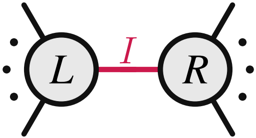


Internal Particles: locality dictates that we multiply each amplitude, and unitarity dictates that we marginalize over unobserved states—integrating over the Lorentz-invariant phase space ("LIPS") for each particle I, and summing over the possible states (helicities, masses, colours, etc.).

$$\sum_{\text{states } I} \int \frac{d^2\lambda_I d^2\widetilde{\lambda}_I}{\text{vol}(GL_1)} \mathcal{A}_L(\ldots, I) \times \mathcal{A}_R(I, \ldots)$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

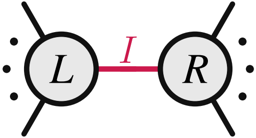


Internal Particles: locality dictates that we multiply each amplitude, and unitarity dictates that we marginalize over unobserved states—integrating over the Lorentz-invariant phase space ("LIPS") for each particle I, and summing over the possible states (helicities, masses, colours, etc.).

$$\int d^4\widetilde{\eta}_I \int \frac{d^2\lambda_I d^2\widetilde{\lambda}_I}{\mathrm{vol}(GL_1)} \mathcal{A}_L(\ldots,I)\times\mathcal{A}_R(I,\ldots)$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

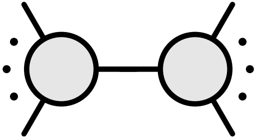


Internal Particles: locality dictates that we multiply each amplitude, and unitarity dictates that we marginalize over unobserved states—integrating over the Lorentz-invariant phase space ("LIPS") for each particle I, and summing over the possible states (helicities, masses, colours, etc.).

$$\int d^4\tilde{\eta}_I \int \frac{d^2\lambda_I d^2\tilde{\lambda}_I}{\mathrm{vol}(GL_1)} \mathcal{A}_L(\ldots, I) \times \mathcal{A}_R(I, \ldots)$$

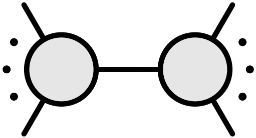
Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



Broadening the Class of Physically Meaningful Functions

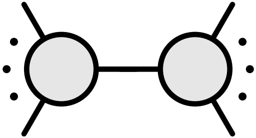
We are interested in the class of functions involving **only** observable quantities



On-Shell Functions:

Broadening the Class of Physically Meaningful Functions

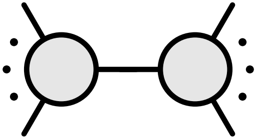
We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

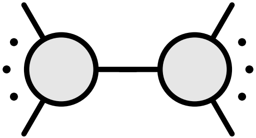


On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

$$f_\Gamma \equiv \prod_{i\in I}\Big(\sum_{\substack{h_i, c_i,\\ m_i, \cdots}} \int d^3\mathrm{LIPS}_i\Big)\prod_v \mathcal{A}_v$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

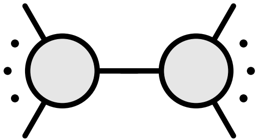


On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\text{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



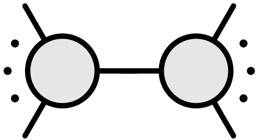
On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

$$f_\Gamma \equiv \prod_{i \in I} \Bigg(\sum_{\substack{h_i, c_i, \\ m_i, \cdots}} \int d^3\mathrm{LIPS}_i \Bigg) \prod_v \mathcal{A}_v$$

Counting Constraints:

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

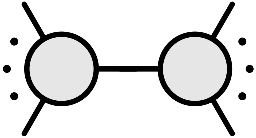
$$f_\Gamma \equiv \prod_{i \in I} \Bigg(\sum_{\substack{h_i, c_i, \\ m_i, \cdots}} \int d^3\mathrm{LIPS}_i \Bigg) \prod_v \mathcal{A}_v$$

Counting Constraints:

n_δ

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

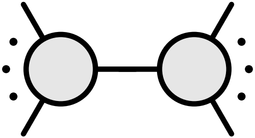
$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

Counting Constraints:

$$n_\delta \equiv 4 \times n_V$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

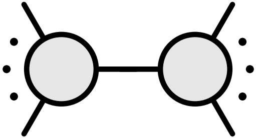
$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

Counting Constraints:

$$n_\delta \equiv 4\times n_V - 3\times n_I$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

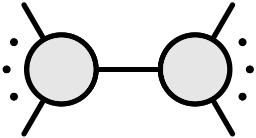
$$f_\Gamma \equiv \prod_{i \in I} \Bigg(\sum_{\substack{h_i, c_i,\\ m_i, \cdots}} \int d^3\mathrm{LIPS}_i \Bigg) \prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4 \times n_V - 3 \times n_I - 4$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

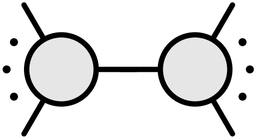
$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \;= \text{number of excess } \delta\text{-functions}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i\in I$, forming a graph Γ called an "on-shell diagram".

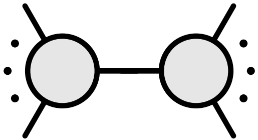
$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \;= \text{number of excess } \delta\text{-functions}$$
$$(= \text{minus number of remaining integrations})$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

$$f_\Gamma \equiv \prod_{i \in I} \Bigg(\sum_{\substack{h_i, c_i, \\ m_i, \cdots}} \int d^3\mathrm{LIPS}_i \Bigg) \prod_v \mathcal{A}_v$$

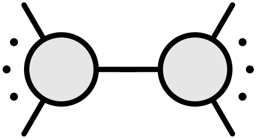
Counting Constraints:

$$\widehat{n}_\delta \equiv 4 \times n_V - 3 \times n_I - 4 \;=\; \text{number of excess } \delta\text{-functions}$$

$$(= \text{minus number of remaining integrations})$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

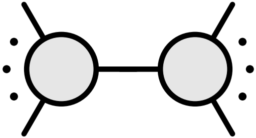
Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \;=\; \text{number of excess } \delta\text{-functions}$$

$$(= \text{minus number of remaining integrations})$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i\in I$, forming a graph Γ called an "on-shell diagram".

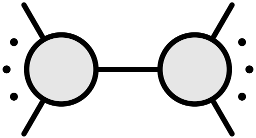
$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \;= 0$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

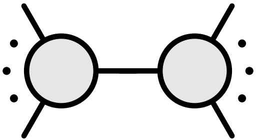
$$f_\Gamma \equiv \prod_{i \in I} \Bigg(\sum_{\substack{h_i, c_i, \\ m_i, \cdots}} \int d^3\mathrm{LIPS}_i \Bigg) \prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4 \times n_V - 3 \times n_I - 4 = 0 \quad \Rightarrow \quad \text{ordinary (rational) function}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

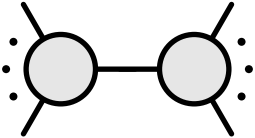
$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \;\begin{cases} >0 & \Rightarrow\quad (\widehat{n}_\delta)\text{ kinematical constraints}\\ =0 & \Rightarrow\quad \text{ordinary (rational) function}\\ <0 & \Rightarrow\quad (-\widehat{n}_\delta)\text{ non-trivial integrations}\end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

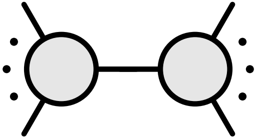
$$f_\Gamma \equiv \prod_{i \in I} \Big(\sum_{\substack{h_i, c_i, \\ m_i, \cdots}} \int d^3\mathrm{LIPS}_i \Big) \prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4 \times n_V - 3 \times n_I - 4 \begin{array}{lcl} > 0 & \Rightarrow & (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow & \text{ordinary (rational) function} \\ < 0 & \Rightarrow & (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{array}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

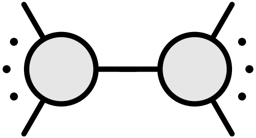
$$f_\Gamma \equiv \prod_{i\in I}\Big(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Big)\prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{array}{lcl} >0 & \Rightarrow & (\widehat{n}_\delta)\text{ kinematical constraints} \\ =0 & \Rightarrow & \text{ordinary (rational) function} \\ <0 & \Rightarrow & (-\widehat{n}_\delta)\text{ non-trivial integrations} \end{array}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

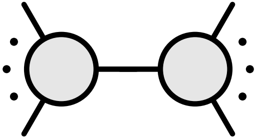
$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{array}{lcl} > 0 & \Rightarrow & (\widehat{n}_\delta)\text{ kinematical constraints}\\ = 0 & \Rightarrow & \text{ordinary (rational) function}\\ < 0 & \Rightarrow & (-\widehat{n}_\delta)\text{ non-trivial integrations}\end{array}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

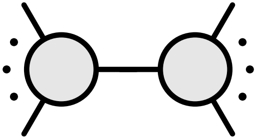
$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{array}{ll} >0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ =0 & \Rightarrow \quad \text{ordinary (rational) function} \\ <0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{array}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

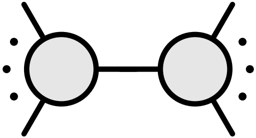
$$f_\Gamma \equiv \prod_{i \in I} \Bigg(\sum_{\substack{h_i, c_i, \\ m_i, \cdots}} \int d^3\mathrm{LIPS}_i \Bigg) \prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4 \times n_V - 3 \times n_I - 4 \begin{array}{lcl} > 0 & \Rightarrow & (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow & \text{ordinary (rational) function} \\ < 0 & \Rightarrow & (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{array}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i \in I$, forming a graph Γ called an "on-shell diagram".

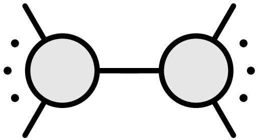
$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \;\begin{cases} >0 & \Rightarrow\quad (\widehat{n}_\delta) \text{ kinematical constraints}\\ =0 & \Rightarrow\quad \text{ordinary (rational) function}\\ <0 & \Rightarrow\quad (-\widehat{n}_\delta) \text{ non-trivial integrations}\end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i\in I$, forming a graph Γ called an "on-shell diagram".

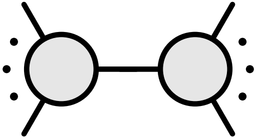
$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{array}{lcl} >0 & \Rightarrow & (\widehat{n}_\delta)\text{ kinematical constraints} \\ =0 & \Rightarrow & \text{ordinary (rational) function} \\ <0 & \Rightarrow & (-\widehat{n}_\delta)\text{ non-trivial integrations} \end{array}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



On-Shell Functions: networks of amplitudes, $\mathcal{A}_v$, connected by any number of internal particles, $i\in I$, forming a graph Γ called an "on-shell diagram".

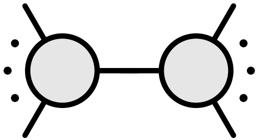
$$f_\Gamma \equiv \prod_{i\in I}\Bigg(\sum_{\substack{h_i,c_i,\\ m_i,\cdots}}\int d^3\mathrm{LIPS}_i\Bigg)\prod_v \mathcal{A}_v$$

Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \;\begin{array}{lcl} >0 & \Rightarrow & (\widehat{n}_\delta)\text{ kinematical constraints}\\ =0 & \Rightarrow & \text{ordinary (rational) function}\\ <0 & \Rightarrow & (-\widehat{n}_\delta)\text{ non-trivial integrations}\end{array}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

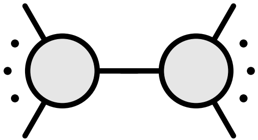


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

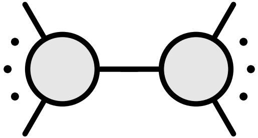


Counting Constraints:

$$\widehat{n}_{\delta} \equiv 4\times n_V - 3\times n_I - 4 \; \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_{\delta}) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_{\delta}) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

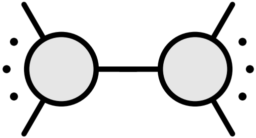


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

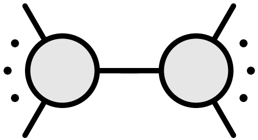


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta)\text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta)\text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

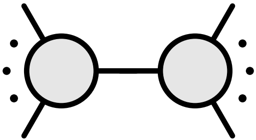


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

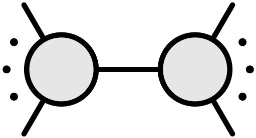


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \;\begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

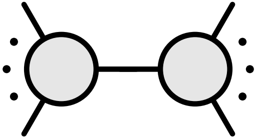


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \;\begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

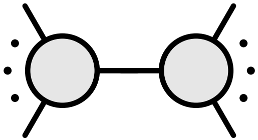


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} >0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ =0 & \Rightarrow \quad \text{ordinary (rational) function} \\ <0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

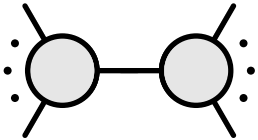


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

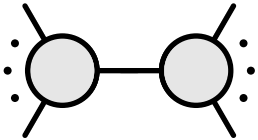


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

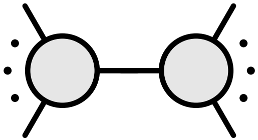


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

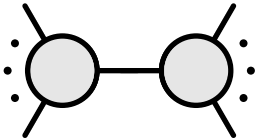


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

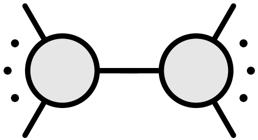


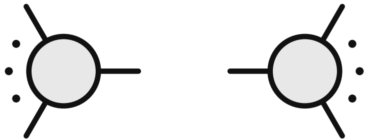
Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \;\begin{cases} >0 & \Rightarrow \quad (\widehat{n}_\delta)\text{ kinematical constraints} \\ =0 & \Rightarrow \quad \text{ordinary (rational) function} \\ <0 & \Rightarrow \quad (-\widehat{n}_\delta)\text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



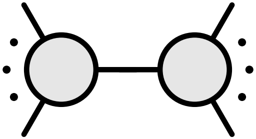


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \;\begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

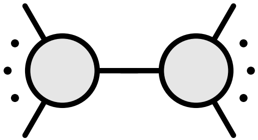


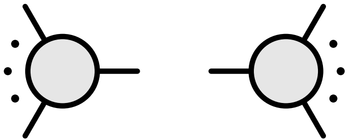
Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \;\; \begin{cases} >0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ =0 & \Rightarrow \quad \text{ordinary (rational) function} \\ <0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



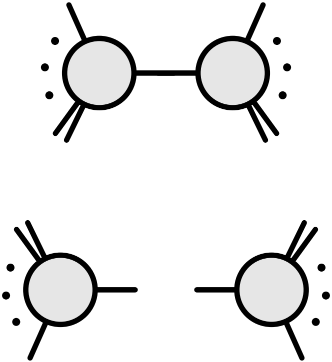


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

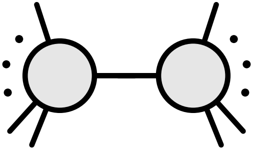


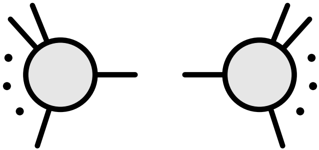
Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} >0 & \Rightarrow & (\widehat{n}_\delta) \text{ kinematical constraints} \\ =0 & \Rightarrow & \text{ordinary (rational) function} \\ <0 & \Rightarrow & (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



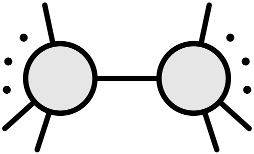


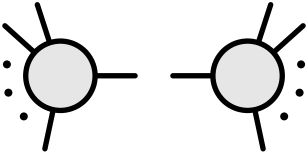
Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} >0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ =0 & \Rightarrow \quad \text{ordinary (rational) function} \\ <0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



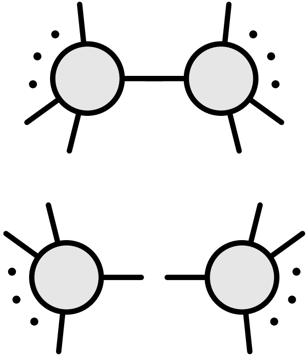


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \; \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

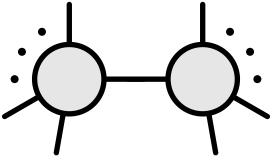


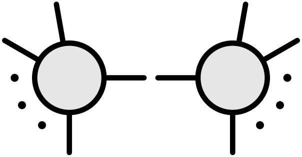
Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



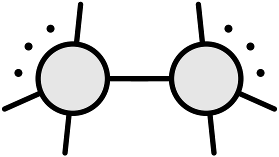


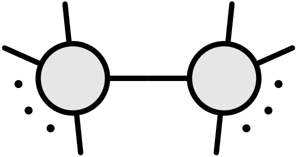
Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \quad \begin{array}{lcl} > 0 & \Rightarrow & (\widehat{n}_\delta)\text{ kinematical constraints} \\ = 0 & \Rightarrow & \text{ordinary (rational) function} \\ < 0 & \Rightarrow & (-\widehat{n}_\delta)\text{ non-trivial integrations} \end{array}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



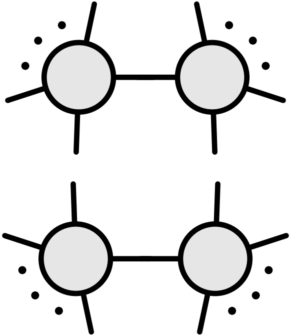


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \; (\widehat{n}_\delta)\text{ kinematical constraints} \\ = 0 & \Rightarrow \; \text{ordinary (rational) function} \\ < 0 & \Rightarrow \; (-\widehat{n}_\delta)\text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

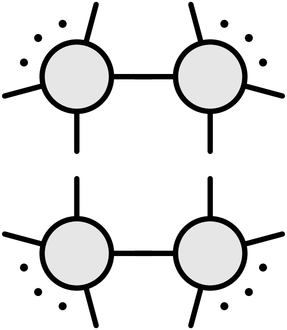


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta)\text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta)\text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

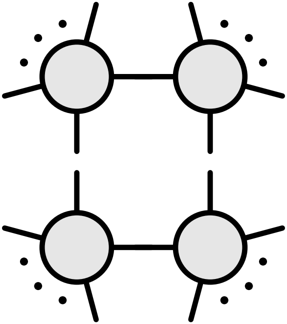


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

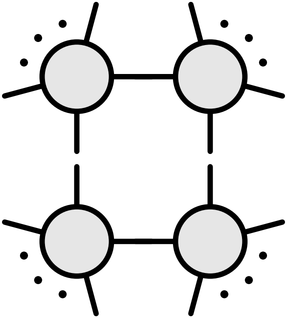


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta)\text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta)\text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

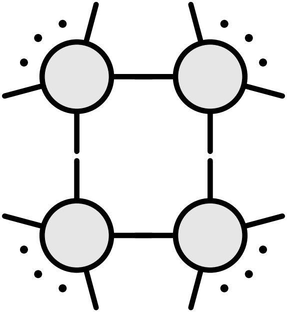


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow & (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow & \text{ordinary (rational) function} \\ < 0 & \Rightarrow & (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

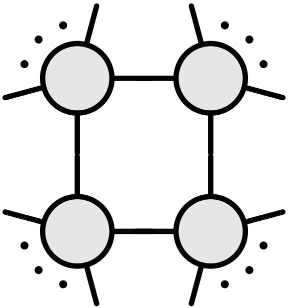


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

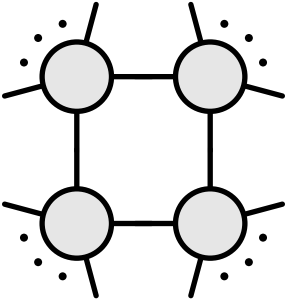


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow & (\widehat{n}_\delta)\text{ kinematical constraints} \\ = 0 & \Rightarrow & \text{ordinary (rational) function} \\ < 0 & \Rightarrow & (-\widehat{n}_\delta)\text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

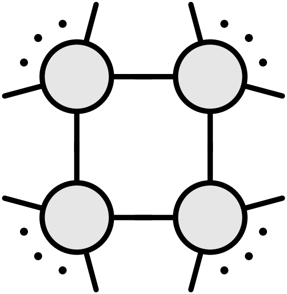


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \quad \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

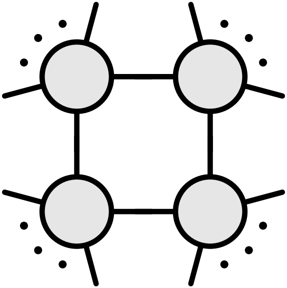


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

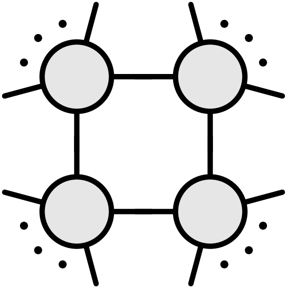


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow & (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow & \text{ordinary (rational) function} \\ < 0 & \Rightarrow & (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

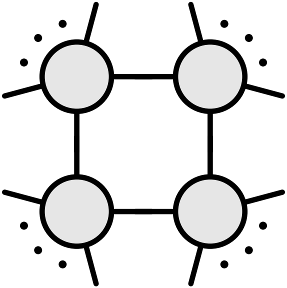


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow & (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow & \text{ordinary (rational) function} \\ < 0 & \Rightarrow & (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

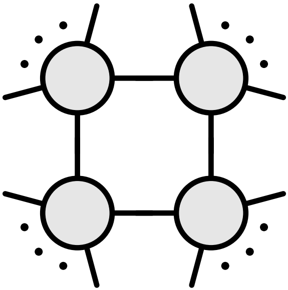


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow & (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow & \text{ordinary (rational) function} \\ < 0 & \Rightarrow & (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities

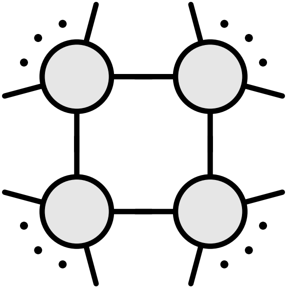


Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Broadening the Class of Physically Meaningful Functions

We are interested in the class of functions involving **only** observable quantities



Counting Constraints:

$$\widehat{n}_\delta \equiv 4\times n_V - 3\times n_I - 4 \quad \begin{cases} > 0 & \Rightarrow \quad (\widehat{n}_\delta) \text{ kinematical constraints} \\ = 0 & \Rightarrow \quad \text{ordinary (rational) function} \\ < 0 & \Rightarrow \quad (-\widehat{n}_\delta) \text{ non-trivial integrations} \end{cases}$$

Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory

Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined
to all orders of perturbation theory

Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

Amalgamating Diagrams from Three-Particle Amplitudes

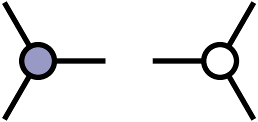
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

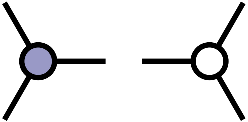
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



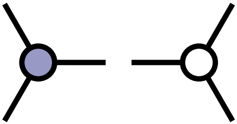
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



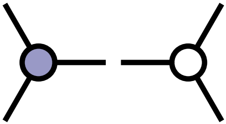
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



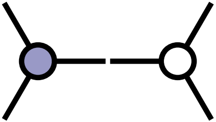
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



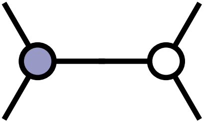
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



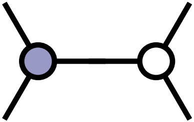
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



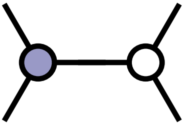
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



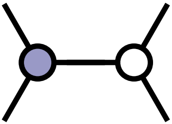
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



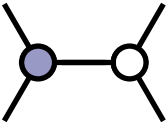
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



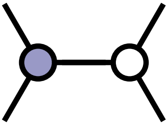
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



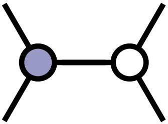
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



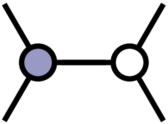
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



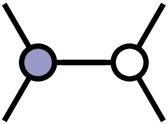
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



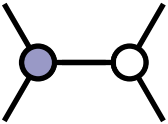
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



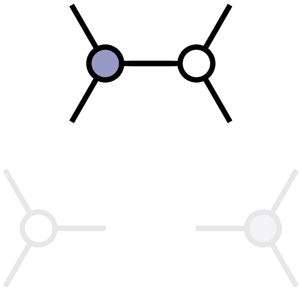
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



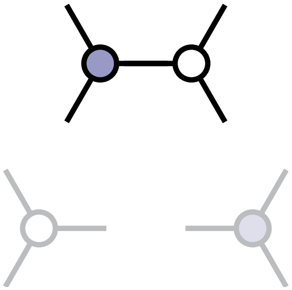
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



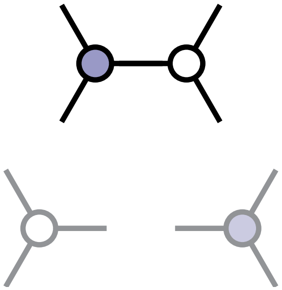
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



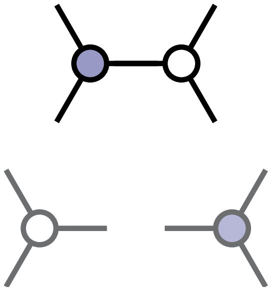
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



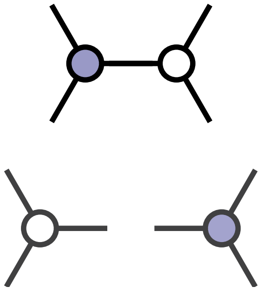
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



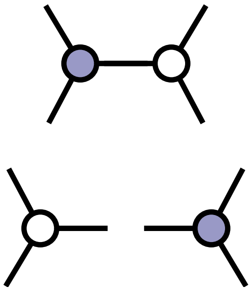
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



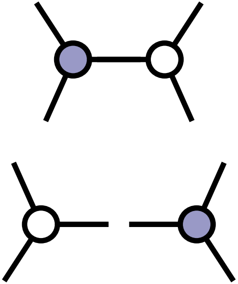
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



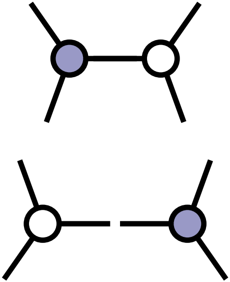
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



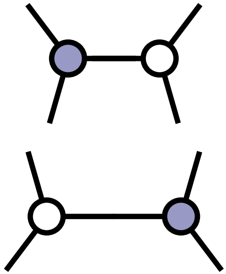
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



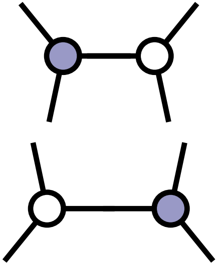
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



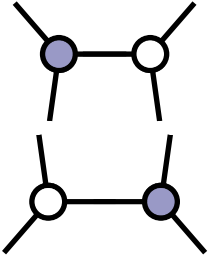
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



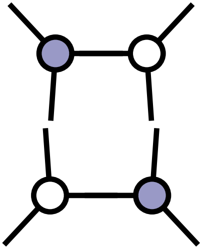
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



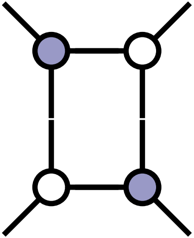
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



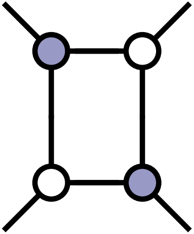
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



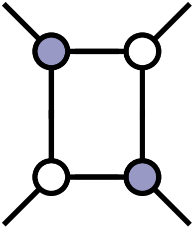
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



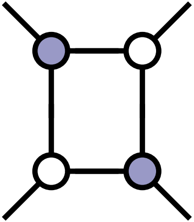
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



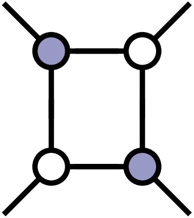
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



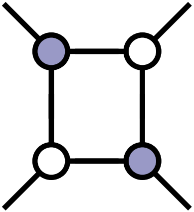
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



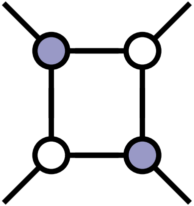
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



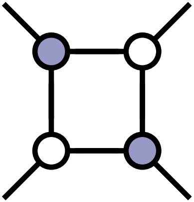
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



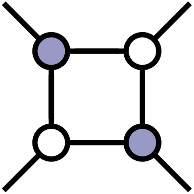
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



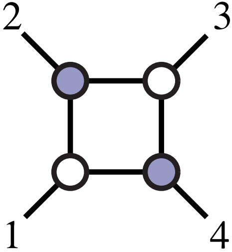
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



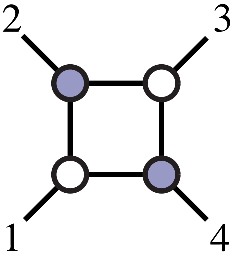
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



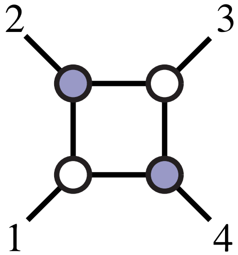
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



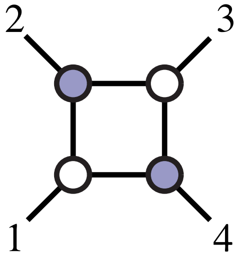
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



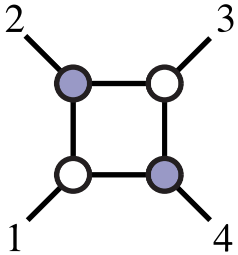
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



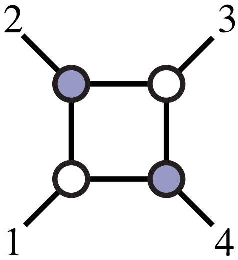
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



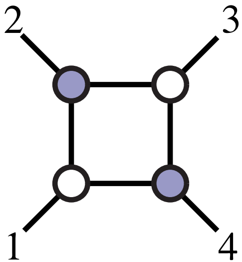
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



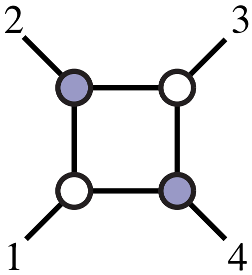
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



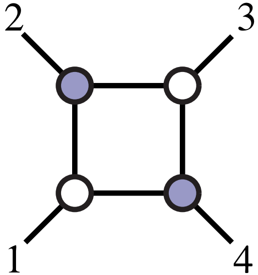
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



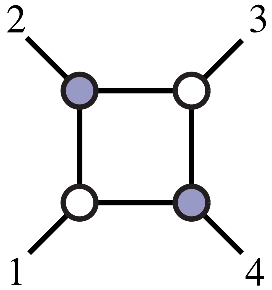
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



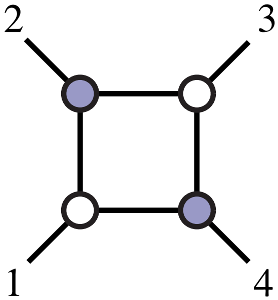
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



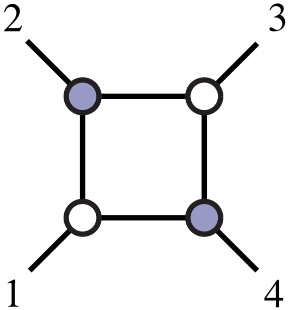
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



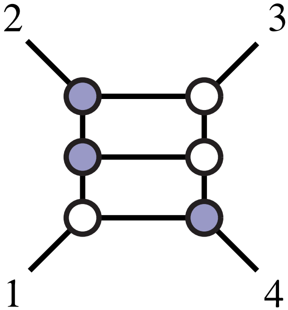
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



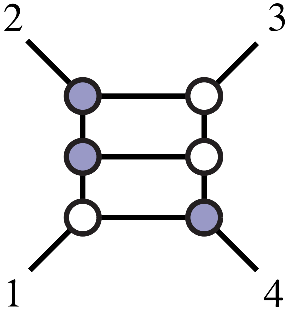
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



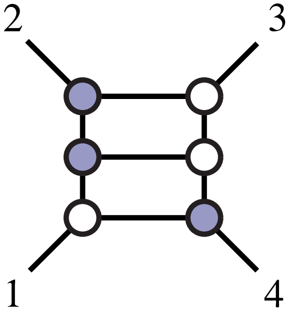
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



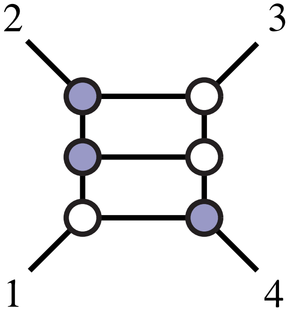
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



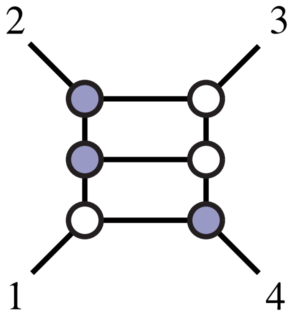
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



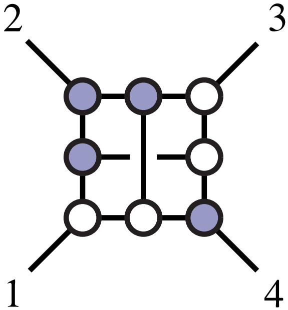
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



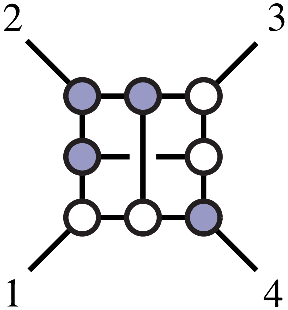
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



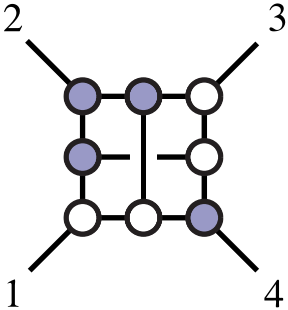
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



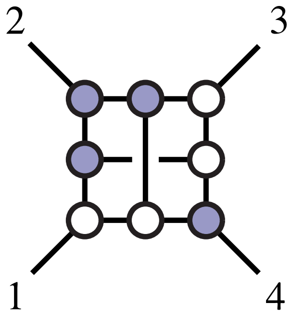
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



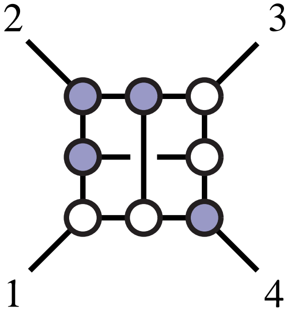
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



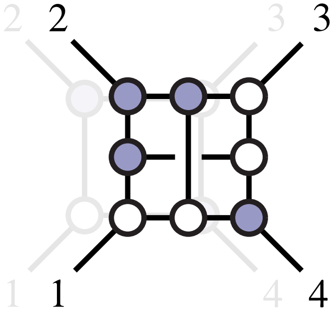
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



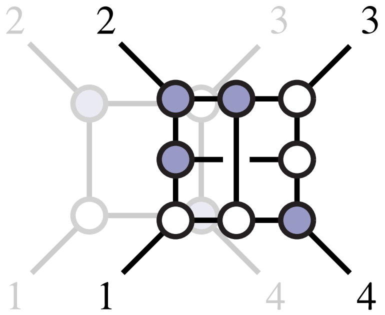
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



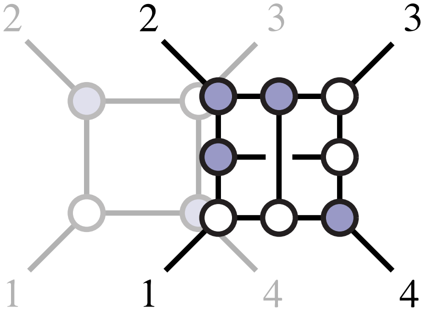
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



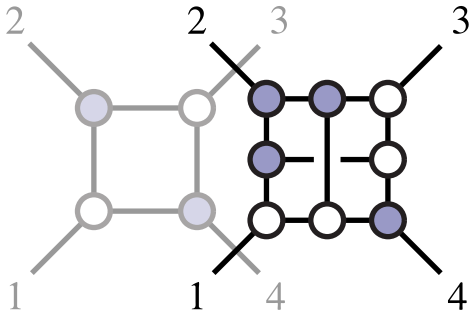
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



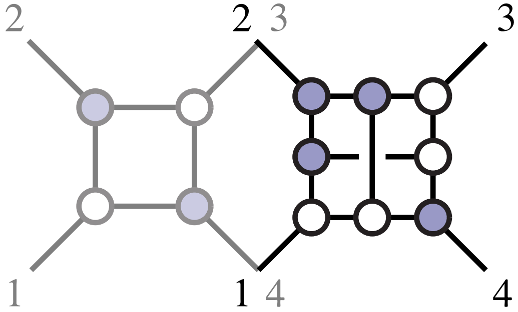
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



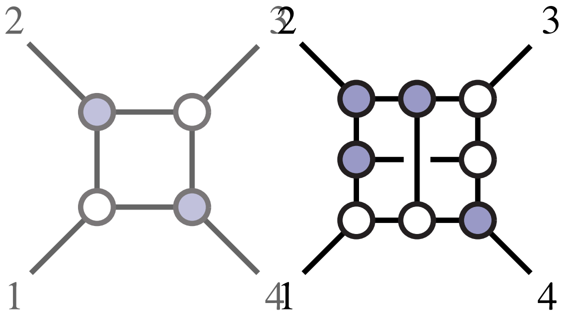
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



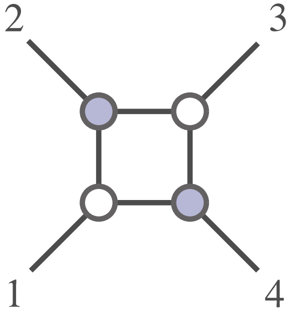
Amalgamating Diagrams from Three-Particle Amplitudes

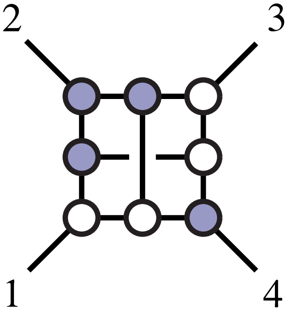
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



Amalgamating Diagrams from Three-Particle Amplitudes

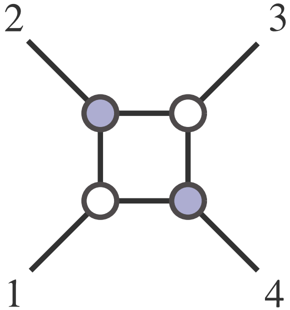
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

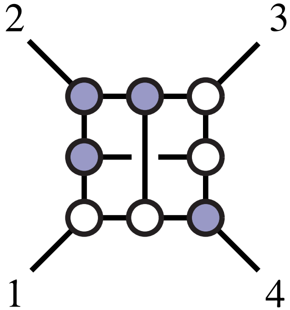




Amalgamating Diagrams from Three-Particle Amplitudes

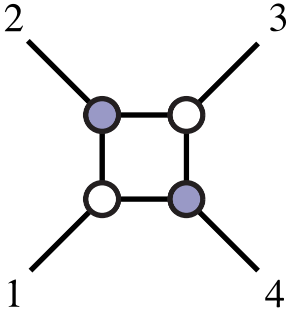
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

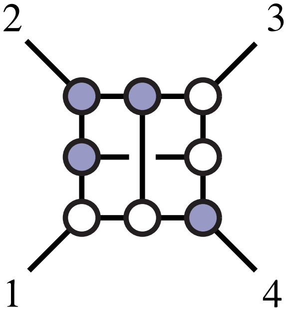




Amalgamating Diagrams from Three-Particle Amplitudes

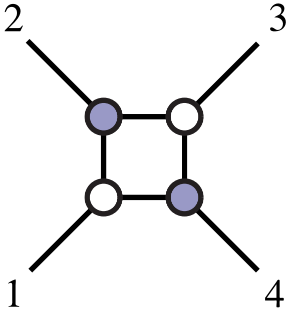
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

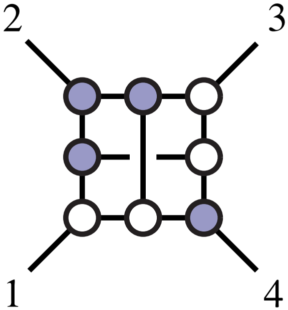




Amalgamating Diagrams from Three-Particle Amplitudes

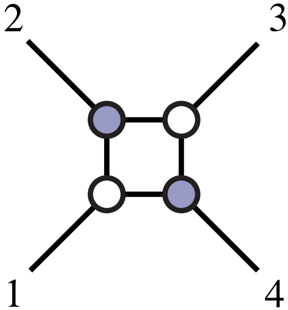
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

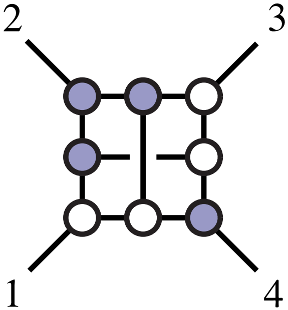




Amalgamating Diagrams from Three-Particle Amplitudes

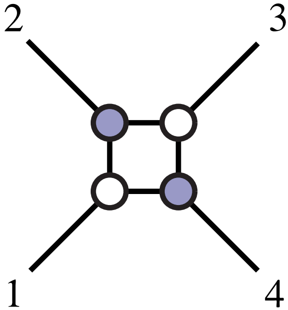
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

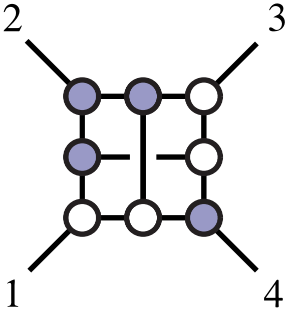




Amalgamating Diagrams from Three-Particle Amplitudes

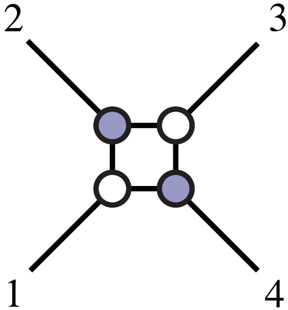
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

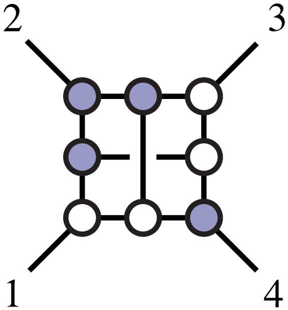




Amalgamating Diagrams from Three-Particle Amplitudes

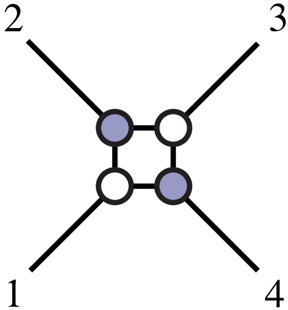
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

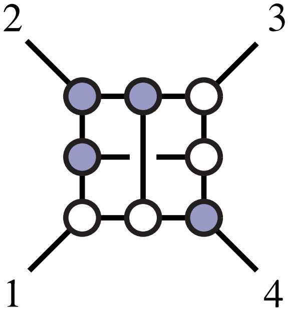




Amalgamating Diagrams from Three-Particle Amplitudes

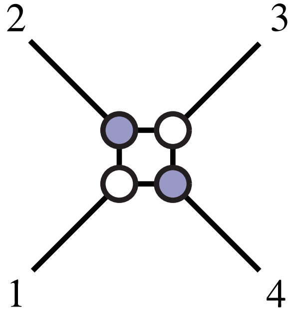
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

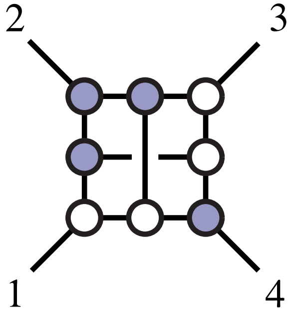




Amalgamating Diagrams from Three-Particle Amplitudes

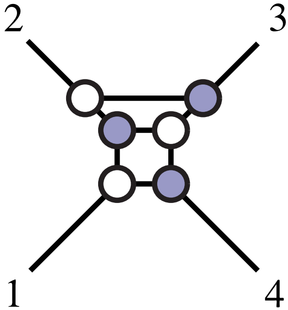
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

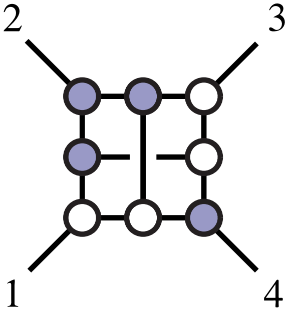




Amalgamating Diagrams from Three-Particle Amplitudes

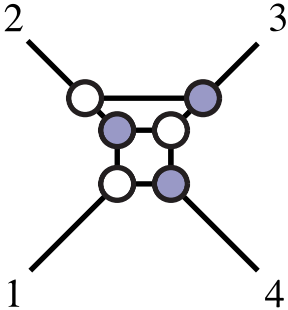
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

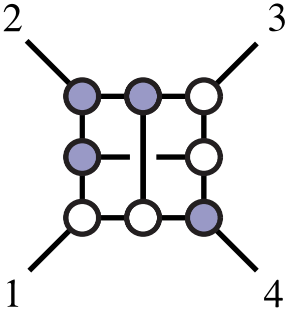




Amalgamating Diagrams from Three-Particle Amplitudes

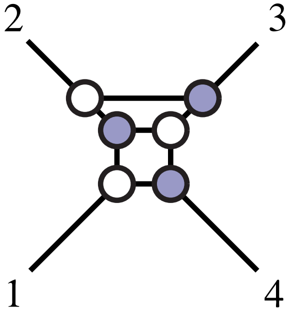
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

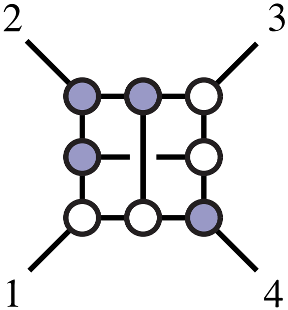




Amalgamating Diagrams from Three-Particle Amplitudes

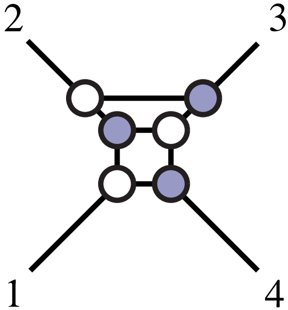
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

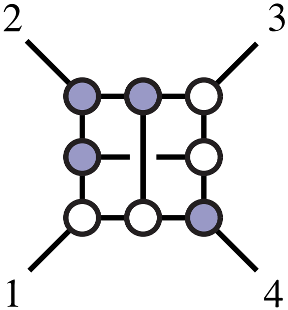




Amalgamating Diagrams from Three-Particle Amplitudes

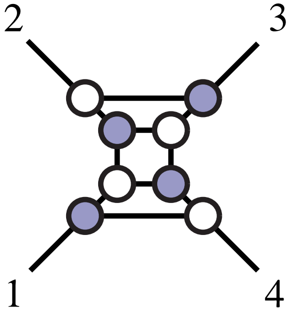
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

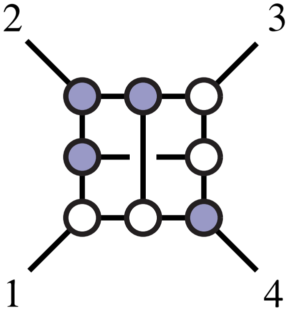




Amalgamating Diagrams from Three-Particle Amplitudes

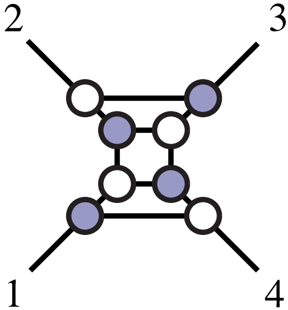
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

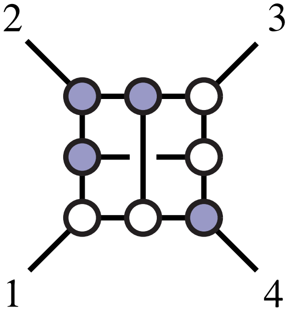




Amalgamating Diagrams from Three-Particle Amplitudes

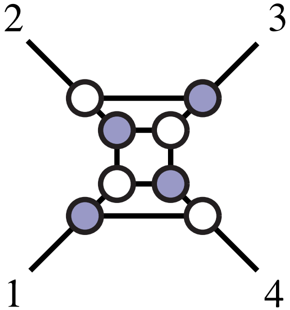
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

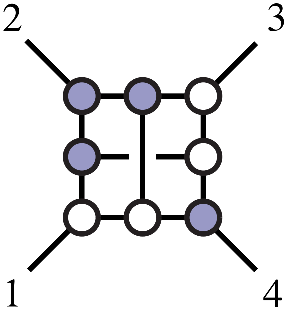




Amalgamating Diagrams from Three-Particle Amplitudes

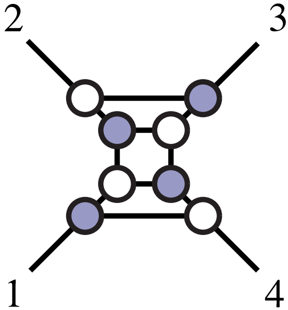
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

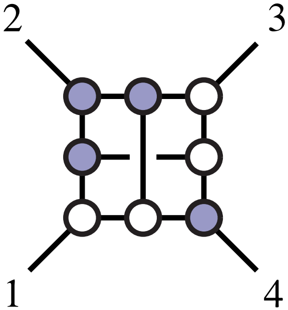




Amalgamating Diagrams from Three-Particle Amplitudes

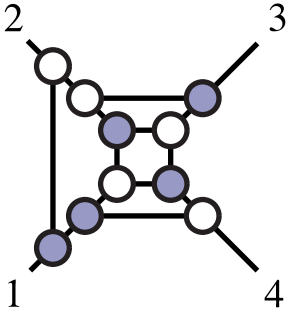
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

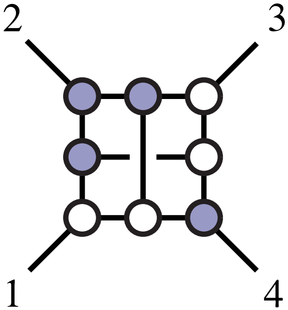




Amalgamating Diagrams from Three-Particle Amplitudes

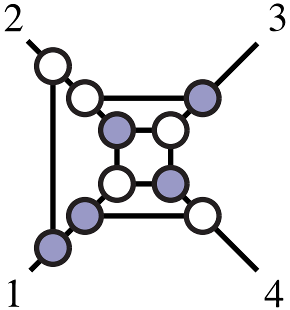
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

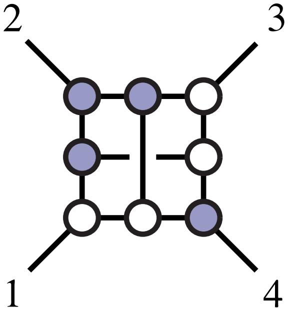




Amalgamating Diagrams from Three-Particle Amplitudes

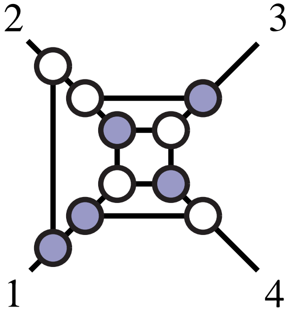
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

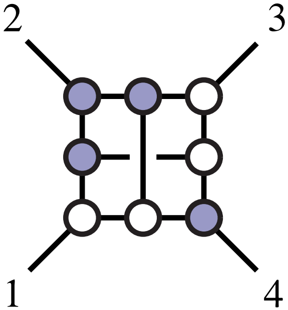




Amalgamating Diagrams from Three-Particle Amplitudes

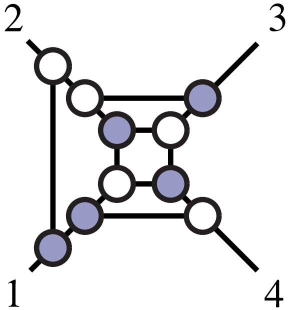
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

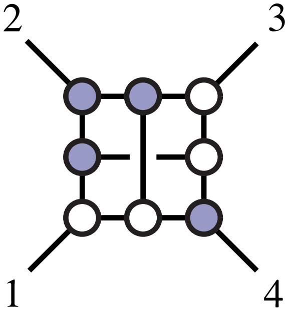




Amalgamating Diagrams from Three-Particle Amplitudes

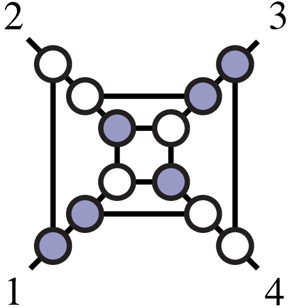
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

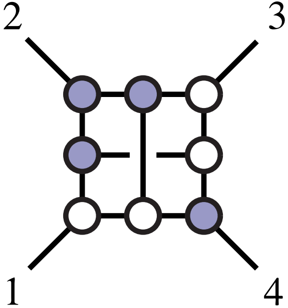




Amalgamating Diagrams from Three-Particle Amplitudes

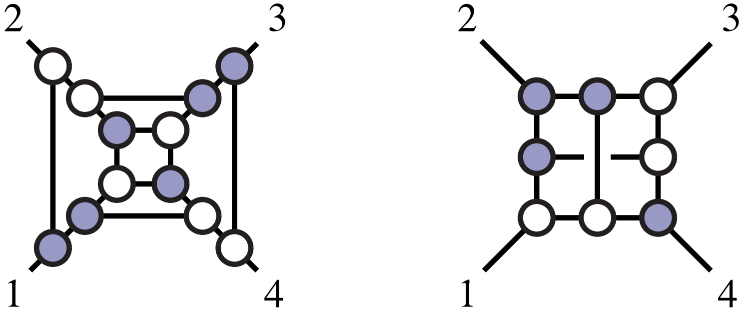
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:





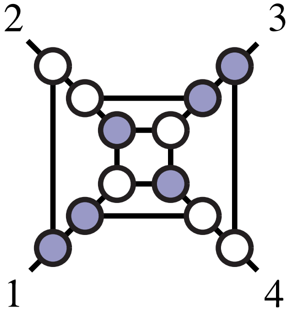
Amalgamating Diagrams from Three-Particle Amplitudes

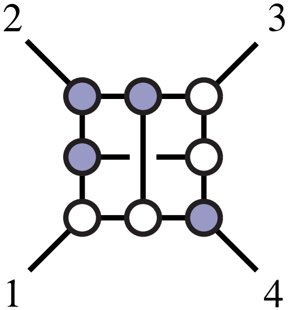
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



Amalgamating Diagrams from Three-Particle Amplitudes

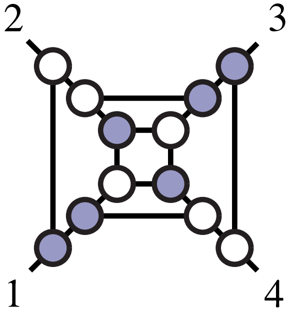
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

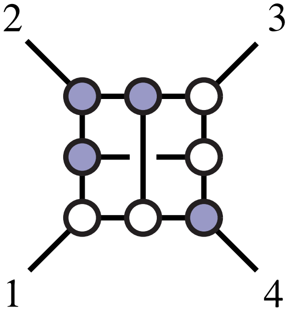




Amalgamating Diagrams from Three-Particle Amplitudes

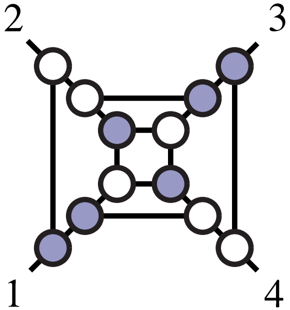
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

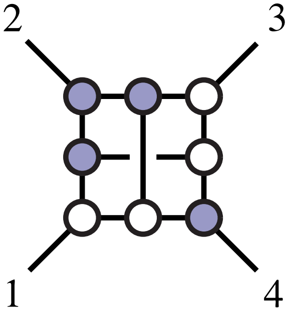




Amalgamating Diagrams from Three-Particle Amplitudes

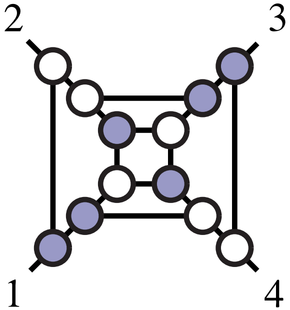
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

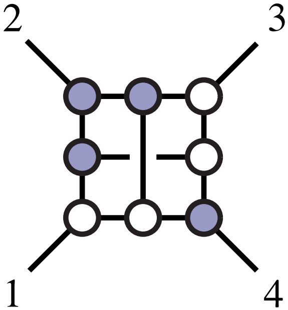




Amalgamating Diagrams from Three-Particle Amplitudes

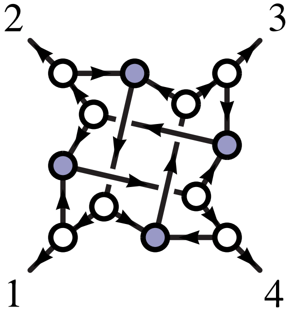
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

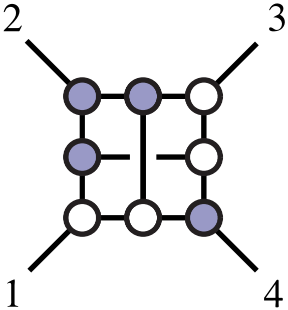




Amalgamating Diagrams from Three-Particle Amplitudes

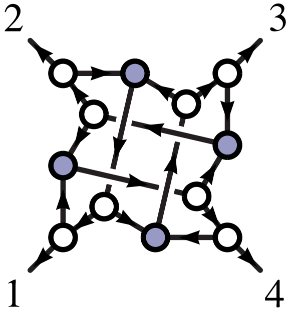
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

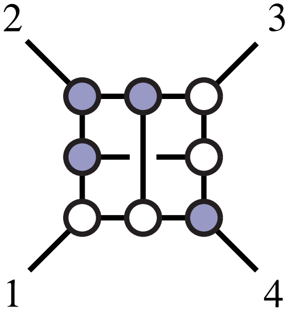




Amalgamating Diagrams from Three-Particle Amplitudes

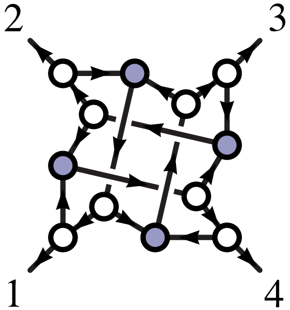
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

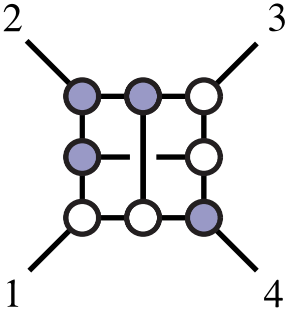




Amalgamating Diagrams from Three-Particle Amplitudes

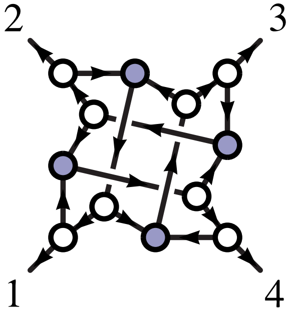
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

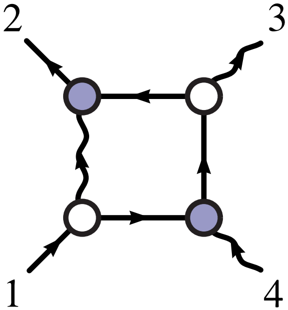




Amalgamating Diagrams from Three-Particle Amplitudes

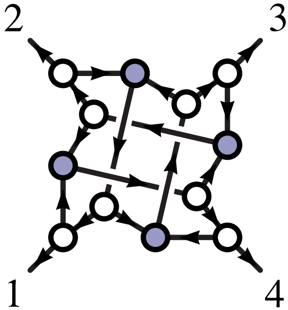
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

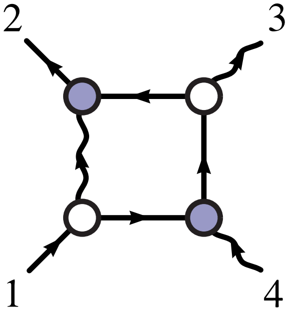




Amalgamating Diagrams from Three-Particle Amplitudes

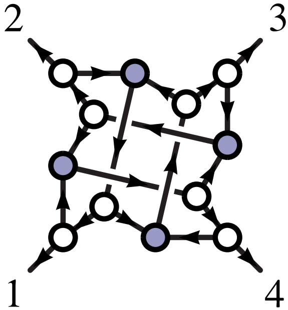
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

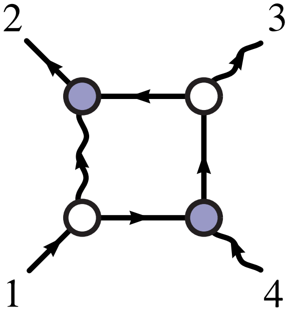




Amalgamating Diagrams from Three-Particle Amplitudes

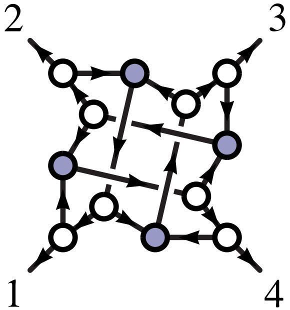
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

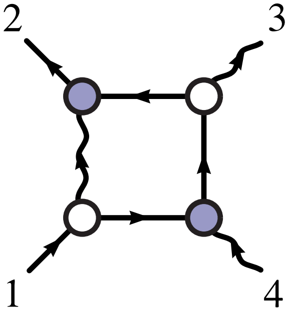




Amalgamating Diagrams from Three-Particle Amplitudes

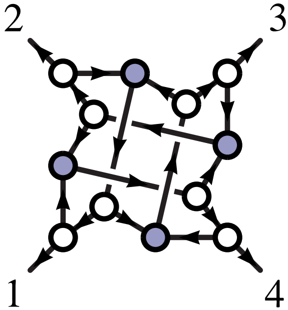
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

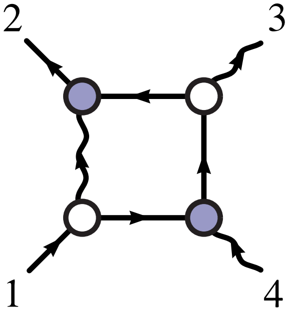




Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:





Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

Amalgamating Diagrams from Three-Particle Amplitudes

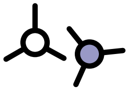
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:

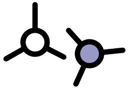
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



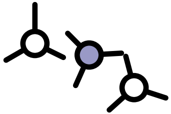
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



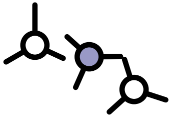
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



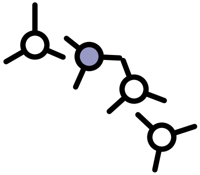
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



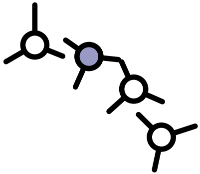
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



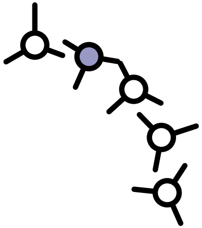
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



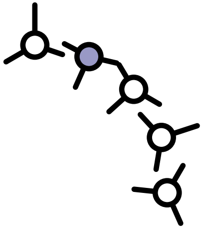
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



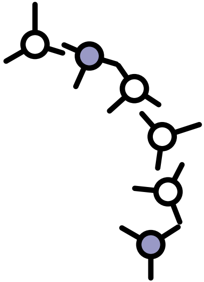
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



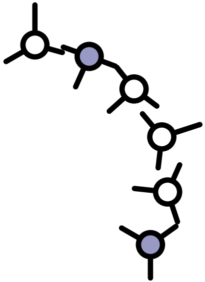
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



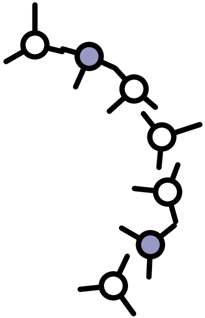
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



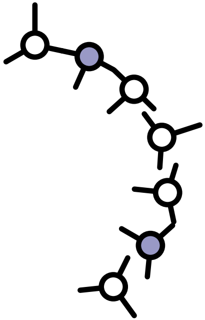
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



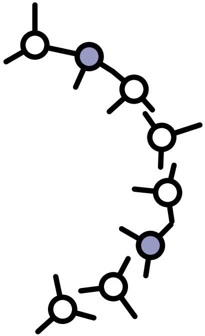
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



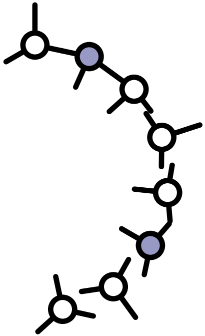
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



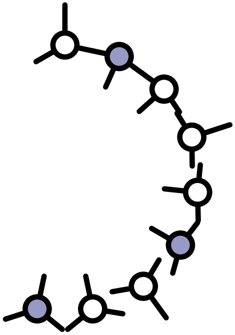
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



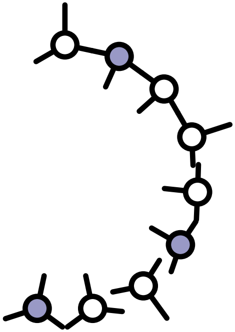
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



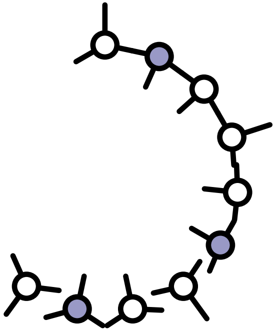
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



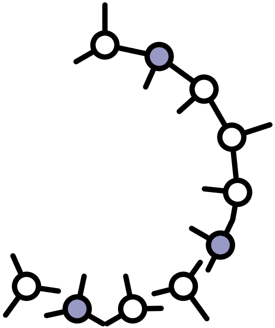
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



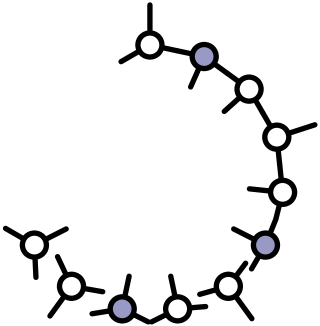
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



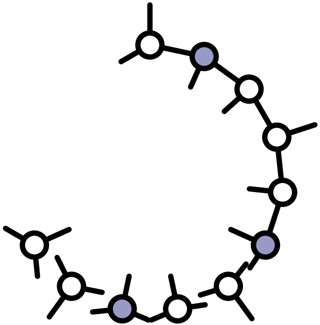
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



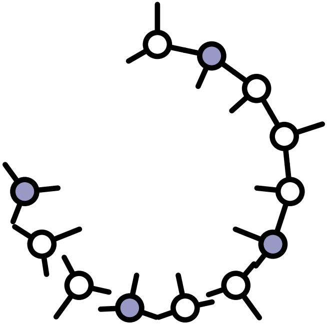
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



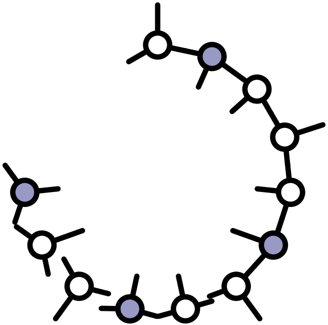
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



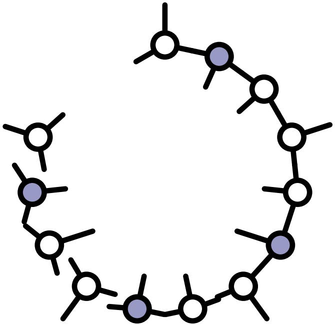
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



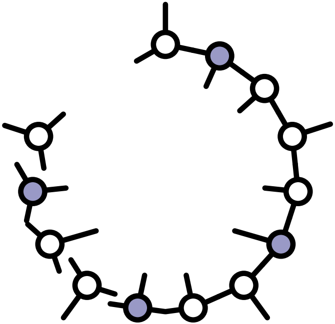
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



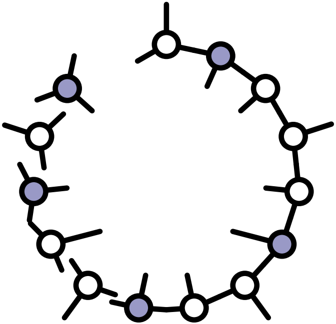
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



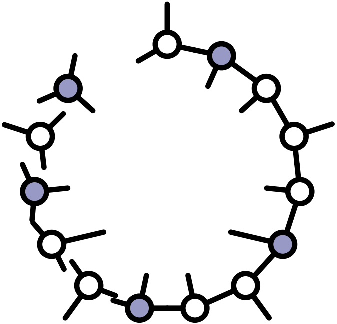
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



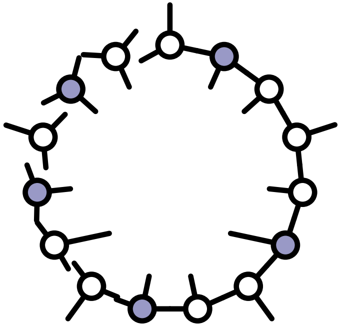
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



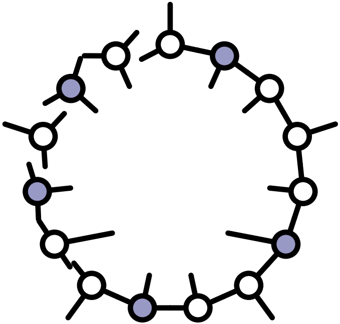
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



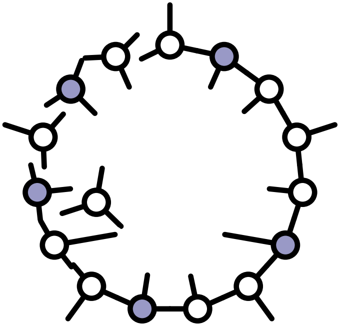
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



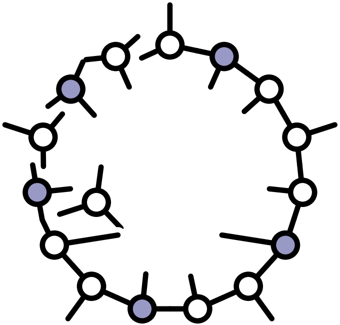
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



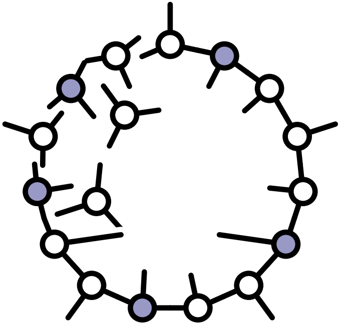
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



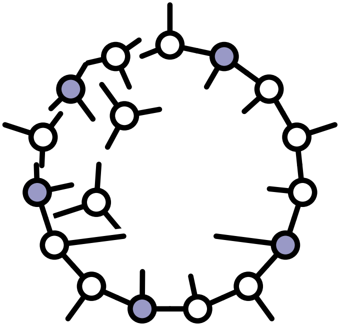
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



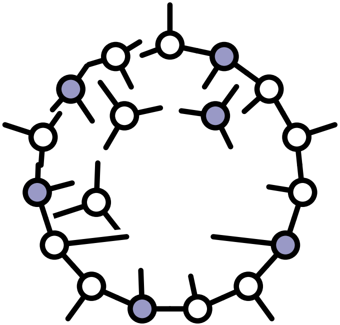
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



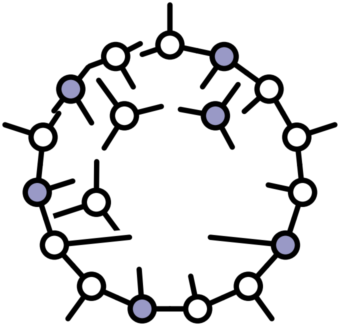
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



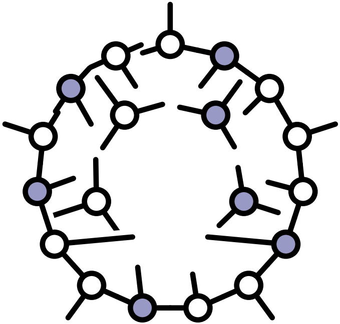
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



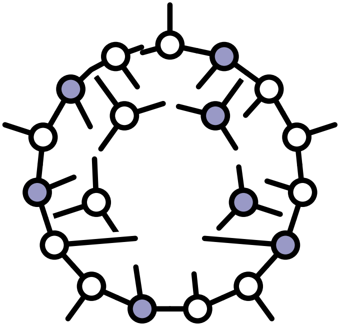
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



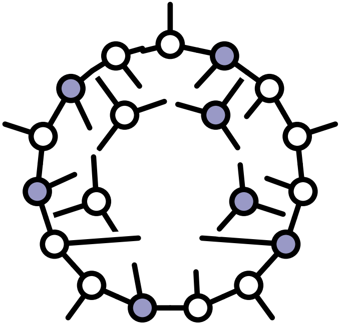
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



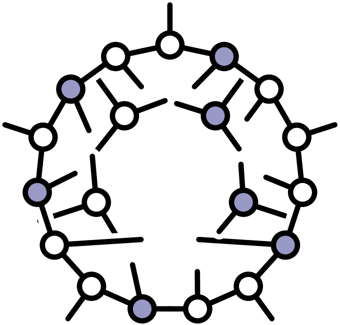
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



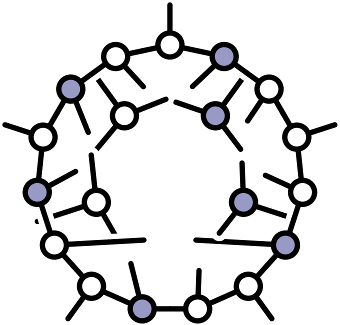
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



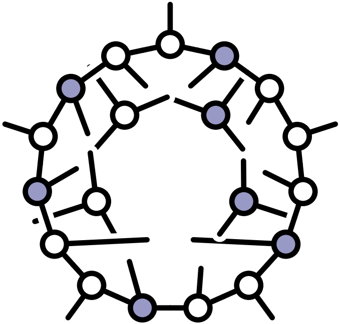
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



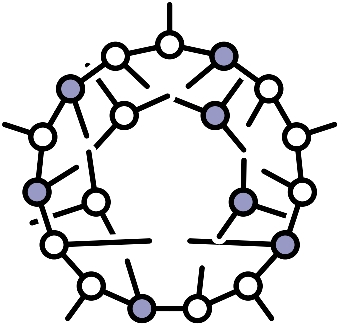
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



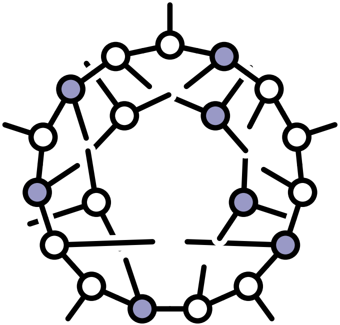
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



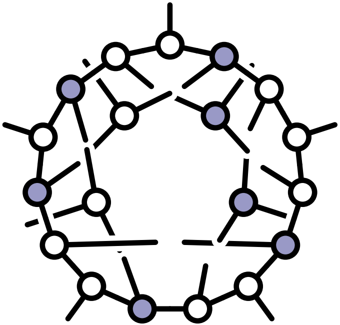
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



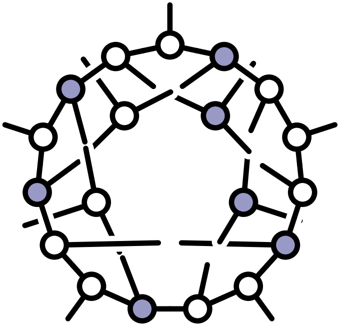
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



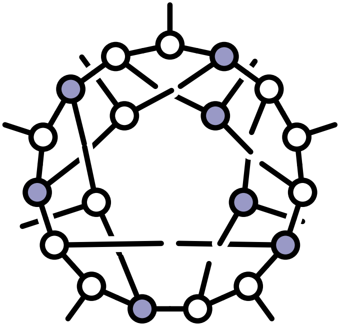
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



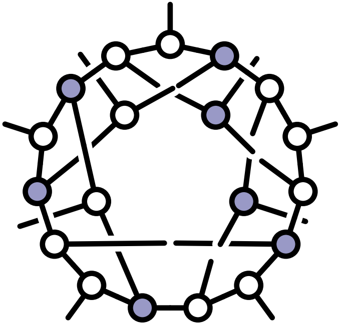
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



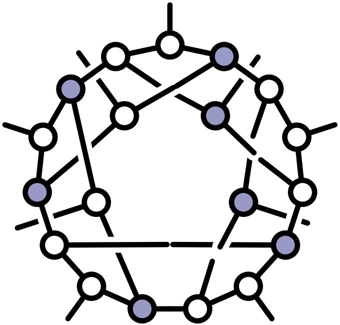
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



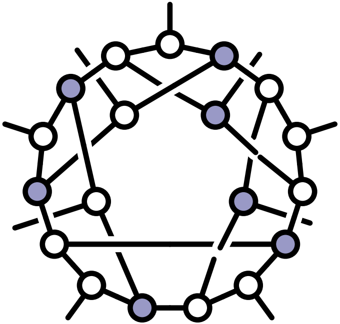
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



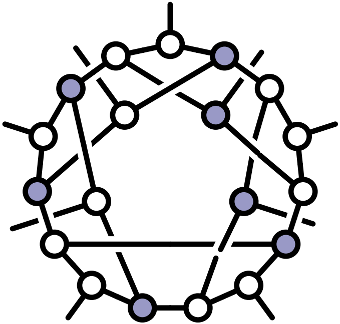
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



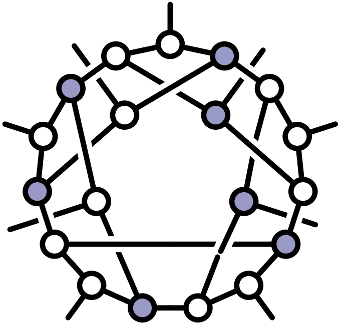
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



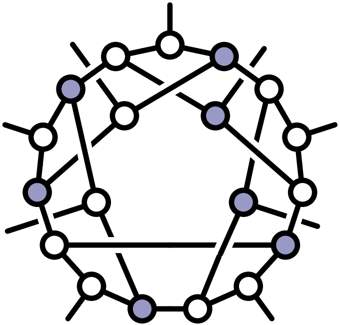
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



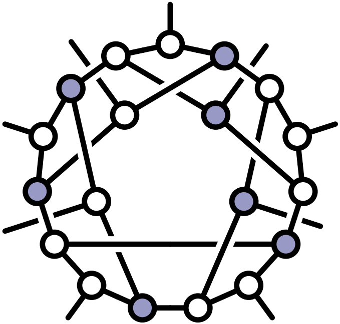
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



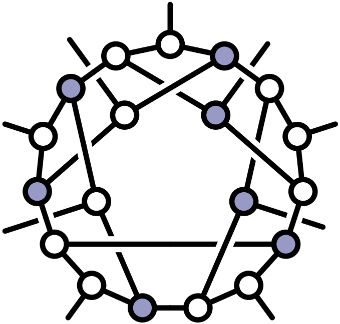
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



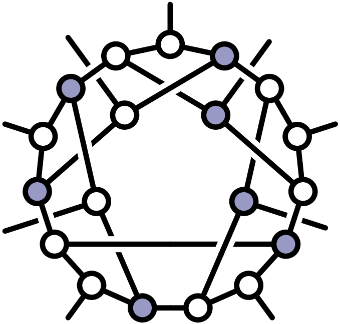
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



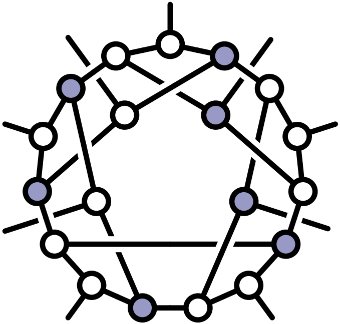
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



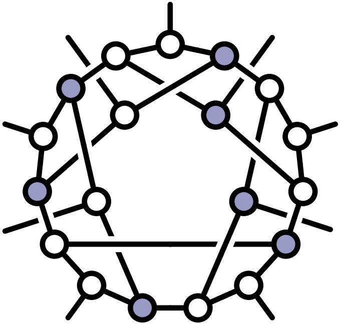
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



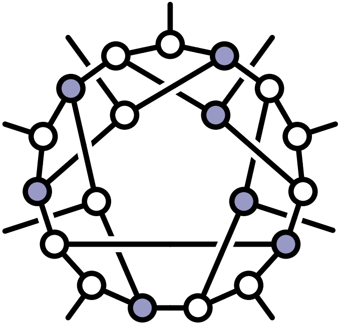
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



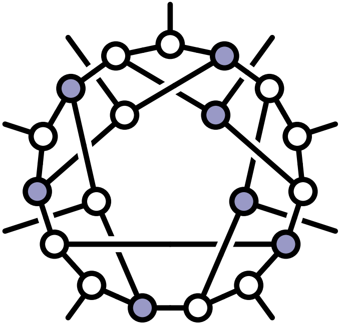
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



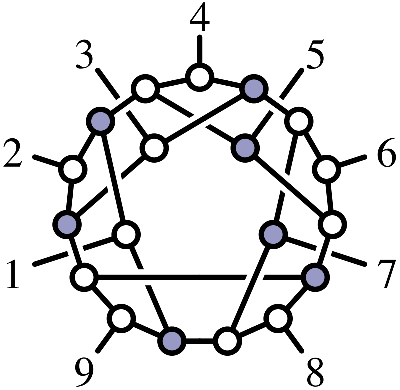
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



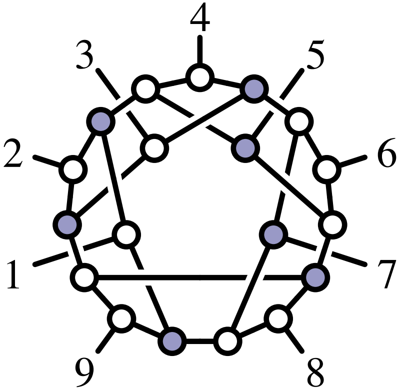
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



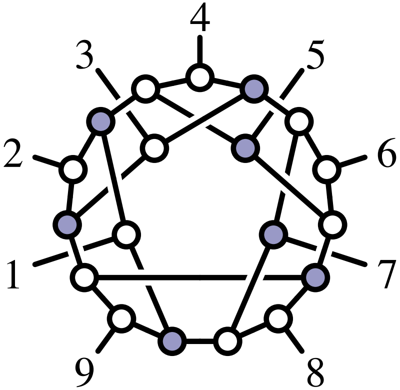
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



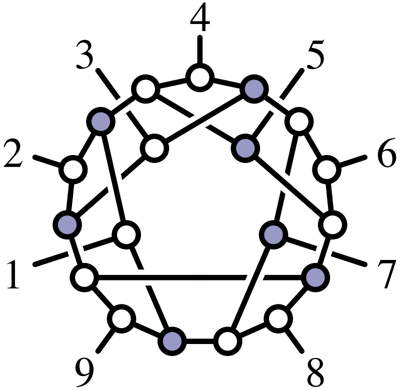
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



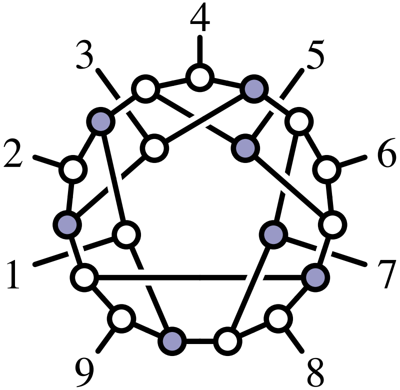
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



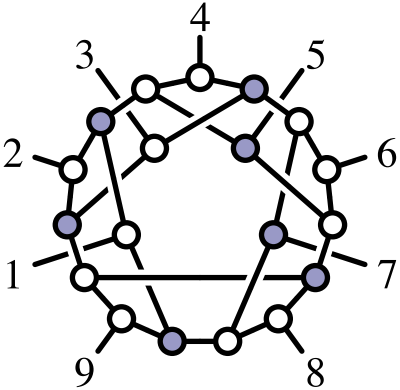
Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



Amalgamating Diagrams from Three-Particle Amplitudes

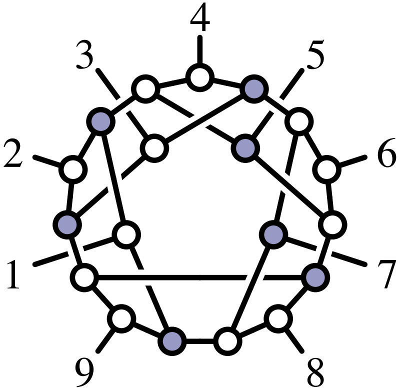
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



$$= \frac{\left(\langle 91\rangle\langle 23\rangle\langle 46\rangle - \langle 16\rangle\langle 34\rangle\langle 29\rangle\right)^2 \quad \delta^{2\times 4}(\lambda\cdot\widetilde{\eta})\,\delta^{2\times 2}(\lambda\cdot\widetilde{\lambda})}{\langle 12\rangle\langle 23\rangle\langle 34\rangle\langle 45\rangle\langle 56\rangle\langle 67\rangle\langle 78\rangle\langle 81\rangle\langle 14\rangle\langle 42\rangle\langle 29\rangle\langle 96\rangle\langle 63\rangle\langle 39\rangle\langle 91\rangle}$$

Amalgamating Diagrams from Three-Particle Amplitudes

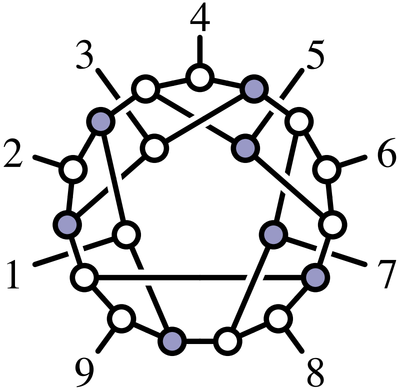
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



$$= \frac{\left(\langle 91\rangle\langle 23\rangle\langle 46\rangle - \langle 16\rangle\langle 34\rangle\langle 29\rangle\right)^2 \quad \delta^{2\times 4}(\lambda\cdot\widetilde{\eta})\,\delta^{2\times 2}(\lambda\cdot\widetilde{\lambda})}{\langle 12\rangle\langle 23\rangle\langle 34\rangle\langle 45\rangle\langle 56\rangle\langle 67\rangle\langle 78\rangle\langle 81\rangle\langle 14\rangle\langle 42\rangle\langle 29\rangle\langle 96\rangle\langle 63\rangle\langle 39\rangle\langle 91\rangle}$$

Amalgamating Diagrams from Three-Particle Amplitudes

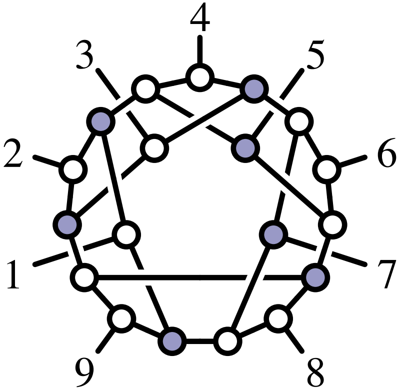
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



$$= \frac{(\langle 91\rangle\langle 23\rangle\langle 46\rangle - \langle 16\rangle\langle 34\rangle\langle 29\rangle)^2 \quad \delta^{2\times 4}(\lambda\cdot\widetilde{\eta})\delta^{2\times 2}(\lambda\cdot\widetilde{\lambda})}{\langle 12\rangle\langle 23\rangle\langle 34\rangle\langle 45\rangle\langle 56\rangle\langle 67\rangle\langle 78\rangle\langle 81\rangle\langle 14\rangle\langle 42\rangle\langle 29\rangle\langle 96\rangle\langle 63\rangle\langle 39\rangle\langle 91\rangle}$$

Amalgamating Diagrams from Three-Particle Amplitudes

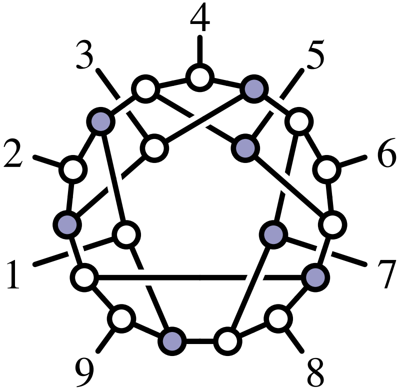
On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



$$= \frac{(\langle 91\rangle\langle 23\rangle\langle 46\rangle - \langle 16\rangle\langle 34\rangle\langle 29\rangle)^2 \quad \delta^{2\times4}(\lambda\cdot\widetilde{\eta})\delta^{2\times2}(\lambda\cdot\widetilde{\lambda})}{\langle 12\rangle\langle 23\rangle\langle 34\rangle\langle 45\rangle\langle 56\rangle\langle 67\rangle\langle 78\rangle\langle 81\rangle\langle 14\rangle\langle 42\rangle\langle 29\rangle\langle 96\rangle\langle 63\rangle\langle 39\rangle\langle 91\rangle}$$

Amalgamating Diagrams from Three-Particle Amplitudes

On-shell diagrams built out of only **three-particle amplitudes** are well-defined to all orders of perturbation theory, generating a large class of functions:



$$= \frac{\left(\langle 91\rangle\langle 23\rangle\langle 46\rangle - \langle 16\rangle\langle 34\rangle\langle 29\rangle\right)^2 \quad \delta^{2\times 4}(\lambda\cdot\widetilde{\eta})\,\delta^{2\times 2}(\lambda\cdot\widetilde{\lambda})}{\langle 12\rangle\langle 23\rangle\langle 34\rangle\langle 45\rangle\langle 56\rangle\langle 67\rangle\langle 78\rangle\langle 81\rangle\langle 14\rangle\langle 42\rangle\langle 29\rangle\langle 96\rangle\langle 63\rangle\langle 39\rangle\langle 91\rangle}$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex

Grassmannian Representations of Three-Point Amplitudes

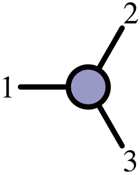
In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use)

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex:

Grassmannian Representations of Three-Point Amplitudes

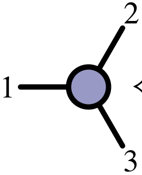
In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:

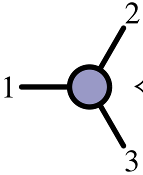


$\Leftrightarrow B$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}\big(\lambda\cdot\widetilde{\eta}\big)}{\langle 12\rangle\langle 23\rangle\langle 31\rangle}\,\delta^{2\times2}\big(\lambda\cdot\widetilde{\lambda}\big)$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:

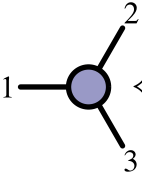


$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times 4}(\lambda \cdot \widetilde{\eta})}{\langle 12\rangle\langle 23\rangle\langle 31\rangle}\delta^{2\times 2}(\lambda \cdot \widetilde{\lambda})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex:

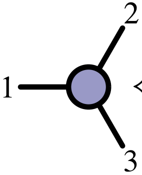


$$\Leftrightarrow B\equiv\begin{pmatrix} b_1^1 & b_2^1 & b_3^1\\ b_1^2 & b_2^2 & b_3^2\end{pmatrix}$$

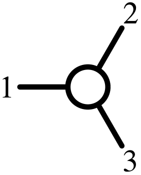
$$\mathcal{A}_3^{(2)}=\frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda})\equiv\int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\delta^{2\times2}(B\cdot\widetilde{\lambda})\ \delta^{1\times2}(\lambda\cdot B^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex:



$$\Leftrightarrow B\equiv\begin{pmatrix} b_1^1 & b_2^1 & b_3^1\\ b_1^2 & b_2^2 & b_3^2\end{pmatrix}$$

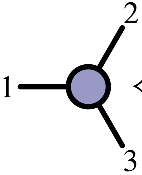


$$\mathcal{A}_3^{(2)}=\frac{\delta^{2\times4}\big(\lambda\cdot\widetilde{\eta}\big)}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}\big(\lambda\cdot\widetilde{\lambda}\big)\equiv\int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\,\frac{\delta^{2\times4}\big(B\cdot\widetilde{\eta}\big)}{(12)(23)(31)}\,\delta^{2\times2}\big(B\cdot\widetilde{\lambda}\big)\,\delta^{1\times2}\big(\lambda\cdot B^{\perp}\big)$$

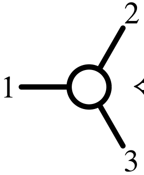
$$\mathcal{A}_3^{(1)}=\frac{\delta^{1\times4}\big(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta}\big)}{[12][23][31]}\delta^{2\times2}\big(\lambda\cdot\widetilde{\lambda}\big)$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



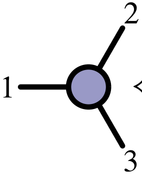
$$\Leftrightarrow W$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\,\delta^{2\times2}(B\cdot\widetilde{\lambda})\,\delta^{1\times2}(\lambda\cdot B^{\perp})$$

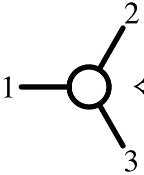
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



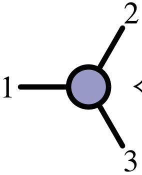
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}\left(\lambda\cdot\widetilde{\eta}\right)}{\langle 12\rangle\langle 23\rangle\langle 31\rangle}\delta^{2\times2}\left(\lambda\cdot\widetilde{\lambda}\right) \equiv \int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\,\frac{\delta^{2\times4}\left(B\cdot\widetilde{\eta}\right)}{(12)(23)(31)}\;\delta^{2\times2}\left(B\cdot\widetilde{\lambda}\right)\;\delta^{1\times2}\left(\lambda\cdot B^{\perp}\right)$$

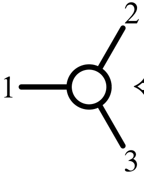
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times4}\left(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta}\right)}{[12][23][31]}\delta^{2\times2}\left(\lambda\cdot\widetilde{\lambda}\right)$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



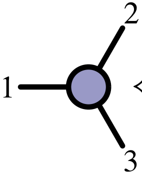
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle 12\rangle\langle 23\rangle\langle 31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\,\delta^{2\times2}(B\cdot\widetilde{\lambda})\,\delta^{1\times2}(\lambda\cdot B^{\perp})$$

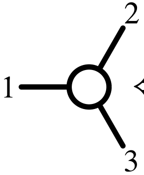
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\frac{\delta^{1\times4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\,\delta^{1\times2}(W\cdot\widetilde{\lambda})\,\delta^{2\times2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



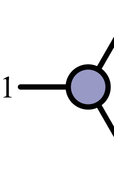
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\,\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\;\delta^{2\times2}(B\cdot\widetilde{\lambda})\;\delta^{1\times2}(\lambda\cdot B^{\perp})$$

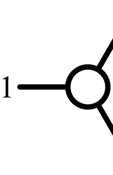
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int\frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\,\frac{\delta^{1\times4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\;\delta^{1\times2}(W\cdot\widetilde{\lambda})\,\delta^{2\times2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



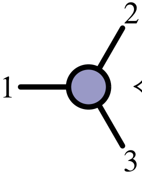
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\,\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\,\delta^{2\times2}(B\cdot\widetilde{\lambda})\;\delta^{1\times2}(\lambda\cdot B^{\perp})$$

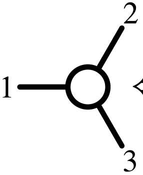
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\,\frac{\delta^{1\times4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\;\delta^{1\times2}(W\cdot\widetilde{\lambda})\,\delta^{2\times2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex:



$$\Leftrightarrow B\equiv\begin{pmatrix} b_1^1 & b_2^1 & b_3^1\\ b_1^2 & b_2^2 & b_3^2\end{pmatrix}$$



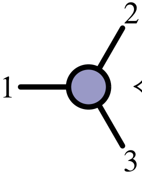
$$\Leftrightarrow W\equiv\begin{pmatrix} w_1^1 & w_2^1 & w_3^1\end{pmatrix}$$

$$\mathcal{A}_3^{(2)}=\frac{\delta^{2\times4}\big(\lambda\cdot\widetilde{\eta}\big)}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}\big(\lambda\cdot\widetilde{\lambda}\big)\equiv\int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times4}\big(B\cdot\widetilde{\eta}\big)}{(12)(23)(31)}\,\delta^{2\times2}\big(B\cdot\widetilde{\lambda}\big)\,\delta^{1\times2}\big(\lambda\cdot B^{\perp}\big)$$

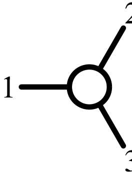
$$\mathcal{A}_3^{(1)}=\frac{\delta^{1\times4}\big(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta}\big)}{[12][23][31]}\delta^{2\times2}\big(\lambda\cdot\widetilde{\lambda}\big)\equiv\int\frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\frac{\delta^{1\times4}\big(W\cdot\widetilde{\eta}\big)}{(1)\,(2)\,(3)}\,\delta^{1\times2}\big(W\cdot\widetilde{\lambda}\big)\delta^{2\times2}\big(\lambda\cdot W^{\perp}\big)$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



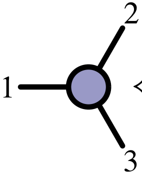
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\,\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\;\delta^{2\times2}(B\cdot\widetilde{\lambda})\;\delta^{1\times2}(\lambda\cdot B^{\perp})$$

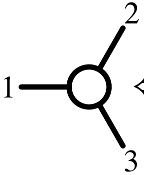
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\,\frac{\delta^{1\times4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\;\delta^{1\times2}(W\cdot\widetilde{\lambda})\,\delta^{2\times2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$



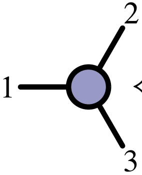
$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times 4}(\lambda\cdot\widetilde{\eta})}{\langle 12\rangle\langle 23\rangle\langle 31\rangle}\delta^{2\times 2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{2\times 3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times 4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\,\delta^{2\times 2}(B\cdot\widetilde{\lambda})\,\delta^{1\times 2}(\lambda\cdot B^{\perp})$$

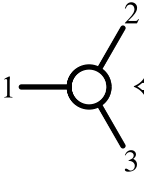
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times 4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times 2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{1\times 3}W}{\mathrm{vol}(GL_1)}\frac{\delta^{1\times 4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\,\delta^{1\times 2}(W\cdot\widetilde{\lambda})\,\delta^{2\times 2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} 1 & 0 & b_3^1 \\ 0 & 1 & b_3^2 \end{pmatrix}$$



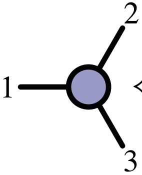
$$\Leftrightarrow W \equiv \begin{pmatrix} 1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times 4}(\lambda\cdot\widetilde{\eta})}{\langle 12\rangle\langle 23\rangle\langle 31\rangle}\delta^{2\times 2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d\,b_3^1}{b_3^1}\wedge\frac{d\,b_3^2}{b_3^2}\;\delta^{2\times 4}(B\cdot\widetilde{\eta})\;\;\delta^{2\times 2}(B\cdot\widetilde{\lambda})\;\delta^{1\times 2}(\lambda\cdot B^{\perp})$$

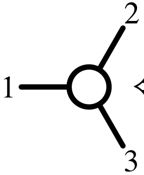
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times 4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times 2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d\,w_2^1}{w_2^1}\wedge\frac{d\,w_3^1}{w_3^1}\;\delta^{1\times 4}(W\cdot\widetilde{\eta})\;\;\delta^{1\times 2}(W\cdot\widetilde{\lambda})\,\delta^{2\times 2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex:



$$\Leftrightarrow B\equiv\begin{pmatrix} b_1^1 & 1 & 0 \\ b_1^2 & 0 & 1\end{pmatrix}$$



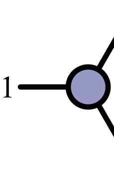
$$\Leftrightarrow W\equiv\begin{pmatrix} w_1^1 & 1 & w_3^1\end{pmatrix}$$

$$\mathcal{A}_3^{(2)}=\frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda})\equiv\int\frac{d\,b_1^1}{b_1^1}\wedge\frac{d\,b_1^2}{b_1^2}\,\delta^{2\times4}(B\cdot\widetilde{\eta})\;\;\delta^{2\times2}(B\cdot\widetilde{\lambda})\;\delta^{1\times2}(\lambda\cdot B^{\perp})$$

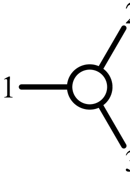
$$\mathcal{A}_3^{(1)}=\frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda})\equiv\int\frac{d\,w_3^1}{w_3^1}\wedge\frac{d\,w_1^1}{w_1^1}\,\delta^{1\times4}(W\cdot\widetilde{\eta})\;\;\delta^{1\times2}(W\cdot\widetilde{\lambda})\,\delta^{2\times2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex:



$$\Leftrightarrow B\equiv\begin{pmatrix}0 & b_2^1 & 1\\ 1 & b_2^2 & 0\end{pmatrix}$$



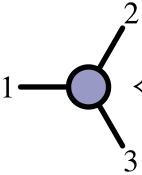
$$\Leftrightarrow W\equiv\begin{pmatrix}w_1^1 & w_2^1 & 1\end{pmatrix}$$

$$\mathcal{A}_3^{(2)}=\frac{\delta^{2\times4}\big(\lambda\cdot\widetilde{\eta}\big)}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}\big(\lambda\cdot\widetilde{\lambda}\big)\equiv\int\frac{d\,b_2^1}{b_2^1}\wedge\frac{d\,b_2^2}{b_2^2}\;\delta^{2\times4}\big(B\cdot\widetilde{\eta}\big)\;\;\delta^{2\times2}\big(B\cdot\widetilde{\lambda}\big)\;\delta^{1\times2}\big(\lambda\cdot B^{\perp}\big)$$

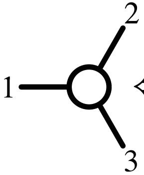
$$\mathcal{A}_3^{(1)}=\frac{\delta^{1\times4}\big(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta}\big)}{[12][23][31]}\delta^{2\times2}\big(\lambda\cdot\widetilde{\lambda}\big)\equiv\int\frac{d\,w_1^1}{w_1^1}\wedge\frac{d\,w_2^1}{w_2^1}\;\delta^{1\times4}\big(W\cdot\widetilde{\eta}\big)\;\;\delta^{1\times2}\big(W\cdot\widetilde{\lambda}\big)\delta^{2\times2}\big(\lambda\cdot W^{\perp}\big)$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



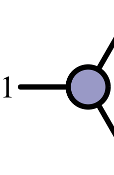
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\,\delta^{2\times2}(B\cdot\widetilde{\lambda})\,\delta^{1\times2}(\lambda\cdot B^{\perp})$$

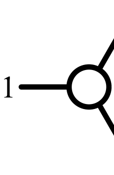
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int\frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\frac{\delta^{1\times4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\,\delta^{1\times2}(W\cdot\widetilde{\lambda})\,\delta^{2\times2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



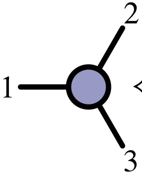
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times 4}\big(\lambda \cdot \widetilde{\eta}\big)}{\langle 12\rangle\langle 23\rangle\langle 31\rangle}\delta^{2\times 2}\big(\lambda\cdot\widetilde{\lambda}\big) \equiv \int \frac{d^{2\times 3}B}{\mathrm{vol}(GL_2)}\,\frac{\delta^{2\times 4}\big(B\cdot\widetilde{\eta}\big)}{(12)(23)(31)}\;\delta^{2\times 2}\big(B\cdot\widetilde{\lambda}\big)\;\delta^{1\times 2}\big(\lambda\cdot B^{\perp}\big)$$

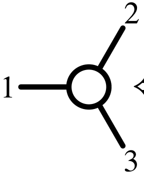
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times 4}\big(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta}\big)}{[12][23][31]}\delta^{2\times 2}\big(\lambda\cdot\widetilde{\lambda}\big) \equiv \int \frac{d^{1\times 3}W}{\mathrm{vol}(GL_1)}\,\frac{\delta^{1\times 4}\big(W\cdot\widetilde{\eta}\big)}{(1)\,(2)\,(3)}\;\delta^{1\times 2}\big(W\cdot\widetilde{\lambda}\big)\delta^{2\times 2}\big(\lambda\cdot W^{\perp}\big)$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex:



$$\Leftrightarrow B\equiv\begin{pmatrix}b_1^1 & b_2^1 & b_3^1\\ b_1^2 & b_2^2 & b_3^2\end{pmatrix}$$



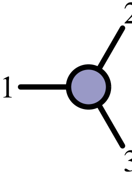
$$\Leftrightarrow W\equiv\begin{pmatrix}w_1^1 & w_2^1 & w_3^1\end{pmatrix}$$

$$\mathcal{A}_3^{(2)}=\frac{\delta^{2\times4}\big(\lambda\cdot\widetilde{\eta}\big)}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}\big(\lambda\cdot\widetilde{\lambda}\big)\equiv\int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\,\frac{\delta^{2\times4}\big(B\cdot\widetilde{\eta}\big)}{(12)(23)(31)}\;\delta^{2\times2}\big(B\cdot\widetilde{\lambda}\big)\;\delta^{1\times2}\big(\lambda\cdot B^{\perp}\big)$$

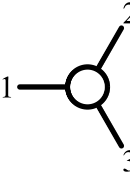
$$\mathcal{A}_3^{(1)}=\frac{\delta^{1\times4}\big(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta}\big)}{[12][23][31]}\delta^{2\times2}\big(\lambda\cdot\widetilde{\lambda}\big)\equiv\int\frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\,\frac{\delta^{1\times4}\big(W\cdot\widetilde{\eta}\big)}{(1)\,(2)\,(3)}\;\delta^{1\times2}\big(W\cdot\widetilde{\lambda}\big)\delta^{2\times2}\big(\lambda\cdot W^{\perp}\big)$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



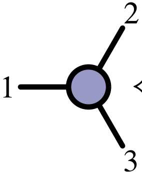
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times 4}(\lambda\cdot\widetilde{\eta})}{\langle 12\rangle\langle 23\rangle\langle 31\rangle}\delta^{2\times 2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{2\times 3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times 4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\,\delta^{2\times 2}(B\cdot\widetilde{\lambda})\,\underbrace{\delta^{1\times 2}(\lambda\cdot B^{\perp})}$$

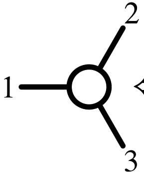
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times 4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times 2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{1\times 3}W}{\mathrm{vol}(GL_1)}\frac{\delta^{1\times 4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\,\delta^{1\times 2}(W\cdot\widetilde{\lambda})\,\delta^{2\times 2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



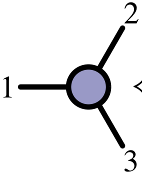
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\delta^{2\times2}(B\cdot\widetilde{\lambda})\underbrace{\delta^{1\times2}(\lambda\cdot B^{\perp})}_{B\mapsto B^*}$$

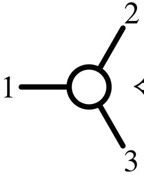
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int\frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\frac{\delta^{1\times4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\delta^{1\times2}(W\cdot\widetilde{\lambda})\delta^{2\times2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



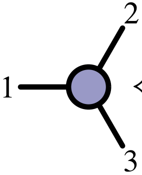
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\,\delta^{2\times2}(B\cdot\widetilde{\lambda})\,\underbrace{\delta^{1\times2}(\lambda\cdot B^{\perp})}_{B\mapsto B^{*}=\lambda}$$

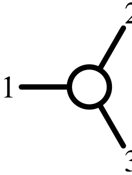
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int\frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\frac{\delta^{1\times4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\,\delta^{1\times2}(W\cdot\widetilde{\lambda})\,\delta^{2\times2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



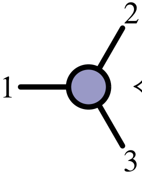
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times 4}\big(\lambda\cdot\widetilde{\eta}\big)}{\langle 12\rangle\langle 23\rangle\langle 31\rangle}\delta^{2\times 2}\big(\lambda\cdot\widetilde{\lambda}\big) \equiv \int \frac{d^{2\times 3}B}{\mathrm{vol}(GL_2)}\,\frac{\delta^{2\times 4}\big(B\cdot\widetilde{\eta}\big)}{(12)(23)(31)}\,\delta^{2\times 2}\big(B\cdot\widetilde{\lambda}\big)\,\underbrace{\delta^{1\times 2}\big(\lambda\cdot B^{\perp}\big)}_{B\mapsto B^* = \lambda}$$

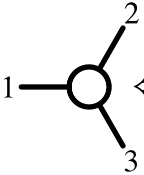
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times 4}\big(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta}\big)}{[12][23][31]}\delta^{2\times 2}\big(\lambda\cdot\widetilde{\lambda}\big) \equiv \int \frac{d^{1\times 3}W}{\mathrm{vol}(GL_1)}\,\frac{\delta^{1\times 4}\big(W\cdot\widetilde{\eta}\big)}{(1)\,(2)\,(3)}\,\delta^{1\times 2}\big(W\cdot\widetilde{\lambda}\big)\delta^{2\times 2}\big(\lambda\cdot W^{\perp}\big)$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex:



$$\Leftrightarrow B\equiv\begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2\end{pmatrix}$$



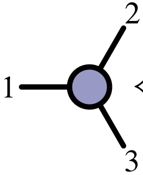
$$\Leftrightarrow W\equiv\begin{pmatrix} w_1^1 & w_2^1 & w_3^1\end{pmatrix}$$

$$\mathcal{A}_3^{(2)}=\frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda})\equiv\int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\delta^{2\times2}(B\cdot\widetilde{\lambda})\underbrace{\delta^{1\times2}(\lambda\cdot B^{\perp})}_{B\mapsto B^*=\lambda}$$

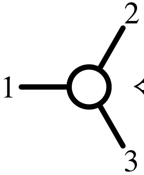
$$\mathcal{A}_3^{(1)}=\frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda})\equiv\int\frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\frac{\delta^{1\times4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\underbrace{\delta^{1\times2}(W\cdot\widetilde{\lambda})}\delta^{2\times2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



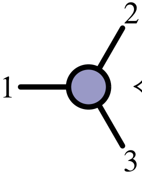
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\,\delta^{2\times2}(B\cdot\widetilde{\lambda})\,\underbrace{\delta^{1\times2}(\lambda\cdot B^{\perp})}_{B\mapsto B^*=\lambda}$$

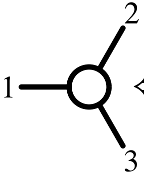
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int\frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\frac{\delta^{1\times4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\,\underbrace{\delta^{1\times2}(W\cdot\widetilde{\lambda})}_{W\mapsto W^*}\delta^{2\times2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



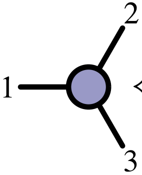
$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times 4}(\lambda\cdot\widetilde{\eta})}{\langle 12\rangle\langle 23\rangle\langle 31\rangle}\delta^{2\times 2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{2\times 3}B}{\mathrm{vol}(GL_2)}\,\frac{\delta^{2\times 4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\,\delta^{2\times 2}(B\cdot\widetilde{\lambda})\,\underbrace{\delta^{1\times 2}(\lambda\cdot B^{\perp})}_{B\mapsto B^*=\lambda}$$

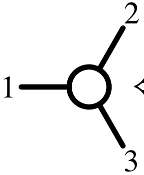
$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times 4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times 2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{1\times 3}W}{\mathrm{vol}(GL_1)}\,\frac{\delta^{1\times 4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\,\underbrace{\delta^{1\times 2}(W\cdot\widetilde{\lambda})}_{W\mapsto W^*=\widetilde{\lambda}^{\perp}}\delta^{2\times 2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex:



$$\Leftrightarrow B\equiv\begin{pmatrix}b_1^1 & b_2^1 & b_3^1\\ b_1^2 & b_2^2 & b_3^2\end{pmatrix}$$



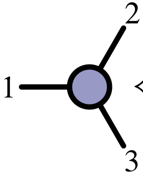
$$\Leftrightarrow W\equiv\begin{pmatrix}w_1^1 & w_2^1 & w_3^1\end{pmatrix}$$

$$\mathcal{A}_3^{(2)}=\frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda})\equiv\int\frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\,\delta^{2\times2}(B\cdot\widetilde{\lambda})\,\underbrace{\delta^{1\times2}(\lambda\cdot B^{\perp})}_{B\mapsto B^*=\lambda}$$

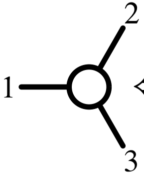
$$\mathcal{A}_3^{(1)}=\frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda})\equiv\int\frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\frac{\delta^{1\times4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\,\underbrace{\delta^{1\times2}(W\cdot\widetilde{\lambda})}_{W\mapsto W^*=\widetilde{\lambda}^{\perp}}\delta^{2\times2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of Three-Point Amplitudes

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex:



$$\Leftrightarrow B \equiv \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 \\ b_1^2 & b_2^2 & b_3^2 \end{pmatrix}$$



$$\Leftrightarrow W \equiv \begin{pmatrix} w_1^1 & w_2^1 & w_3^1 \end{pmatrix}$$

$$\mathcal{A}_3^{(2)} = \frac{\delta^{2\times4}(\lambda\cdot\widetilde{\eta})}{\langle12\rangle\langle23\rangle\langle31\rangle}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{2\times3}B}{\mathrm{vol}(GL_2)}\frac{\delta^{2\times4}(B\cdot\widetilde{\eta})}{(12)(23)(31)}\,\delta^{2\times2}(B\cdot\widetilde{\lambda})\,\underbrace{\delta^{1\times2}(\lambda\cdot B^{\perp})}_{B\mapsto B^*=\lambda}$$

$$\mathcal{A}_3^{(1)} = \frac{\delta^{1\times4}(\widetilde{\lambda}^{\perp}\cdot\widetilde{\eta})}{[12][23][31]}\delta^{2\times2}(\lambda\cdot\widetilde{\lambda}) \equiv \int \frac{d^{1\times3}W}{\mathrm{vol}(GL_1)}\frac{\delta^{1\times4}(W\cdot\widetilde{\eta})}{(1)\,(2)\,(3)}\,\underbrace{\delta^{1\times2}(W\cdot\widetilde{\lambda})}_{W\mapsto W^*=\widetilde{\lambda}^{\perp}}\delta^{2\times2}(\lambda\cdot W^{\perp})$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f\equiv\int\Omega_C\;\;\delta^{k\times4}\big(C\cdot\widetilde{\eta}\big)\delta^{k\times2}\big(C\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C^{\perp}\big)$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \;\; \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f\equiv\int\Omega_C\ \ \delta^{k\times4}\big(C\cdot\widetilde{\eta}\big)\delta^{k\times2}\big(C\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C^{\perp}\big)$$

$$C\in G(k,n)$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f\equiv\int\Omega_C\;\;\delta^{k\times4}\big(C\cdot\widetilde{\eta}\big)\delta^{k\times2}\big(C\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C^{\perp}\big)$$

$$C\in G(k,n)$$
$$k\equiv 2n_B+n_W-n_I$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f\equiv\int\Omega_C\;\;\delta^{k\times4}\big(C\cdot\widetilde{\eta}\big)\delta^{k\times2}\big(C\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C^{\perp}\big)$$

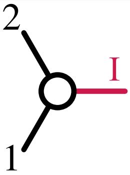
$$C\in G(k,n)$$
$$k\equiv2n_B+n_W-n_I$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \quad \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$



Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \quad \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \;\; \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n) \qquad k \equiv 2n_B + n_W - n_I$$

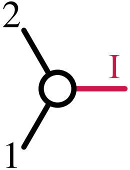
$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

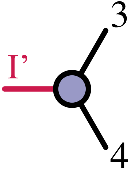
In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \;\; \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n) \qquad k \equiv 2n_B + n_W - n_I$$



$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$



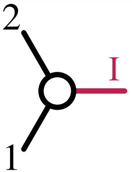
$$\begin{array}{ccc} \mathrm{I}' & 3 & 4 \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

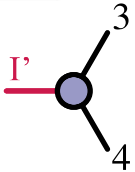
In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f\equiv\int\Omega_C\;\;\delta^{k\times4}\big(C\cdot\widetilde{\eta}\big)\delta^{k\times2}\big(C\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C^{\perp}\big)$$

$$C\in G(k,n)$$
$$k\equiv 2n_B+n_W-n_I$$



$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \multicolumn{3}{c}{\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}} \end{array}$$



$$\begin{array}{ccc} \mathrm{I'} & 3 & 4 \\ \hline \multicolumn{3}{c}{\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}} \end{array}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \;\; \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$

$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$

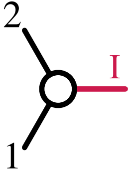
$$\begin{array}{ccc} \mathrm{I}' & 3 & 4 \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

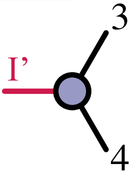
In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \;\; \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$



$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$



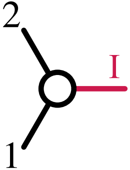
$$\begin{array}{ccc} \mathrm{I'} & 3 & 4 \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

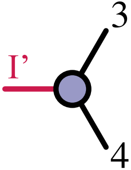
In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \;\; \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C\in G(k,n)$$
$$k\equiv 2n_B+n_W-n_I$$



$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$



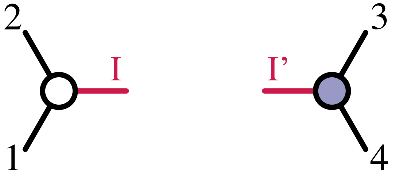
$$\begin{array}{ccc} \mathrm{I'} & 3 & 4 \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \quad \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$



$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$

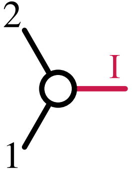
$$\begin{array}{ccc} \mathrm{I'} & 3 & 4 \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

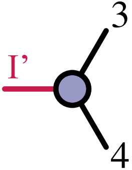
In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \quad \delta^{k\times 4}\big(C\cdot\widetilde{\eta}\big)\,\delta^{k\times 2}\big(C\cdot\widetilde{\lambda}\big)\,\delta^{2\times(n-k)}\big(\lambda\cdot C^{\perp}\big)$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$



$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \end{array}$$
$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$



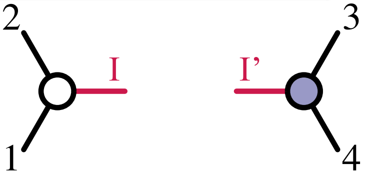
$$\begin{array}{ccc} \mathrm{I'} & 3 & 4 \\ \end{array}$$
$$\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f\equiv\int\Omega_C\quad\delta^{k\times4}(C\cdot\widetilde{\eta})\,\delta^{k\times2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C\in G(k,n)$$
$$k\equiv2n_B+n_W-n_I$$



$$\begin{array}{ccc}1&2&\mathrm{I}\\\hline\end{array}$$
$$\begin{pmatrix}1&w_2&w_{\mathrm{I}}\end{pmatrix}$$

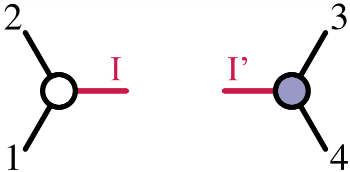
$$\begin{array}{ccc}\mathrm{I'}&3&4\\\hline\end{array}$$
$$\begin{pmatrix}1&0&b_4^1\\0&1&b_4^2\end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \quad \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$



$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$

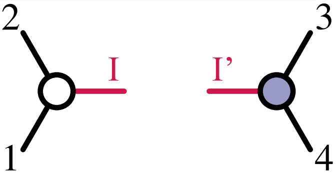
$$\begin{array}{ccc} \mathrm{I'} & 3 & 4 \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f\equiv\int\Omega_C\quad\delta^{k\times4}\big(C\cdot\widetilde{\eta}\big)\,\delta^{k\times2}\big(C\cdot\widetilde{\lambda}\big)\,\delta^{2\times(n-k)}\big(\lambda\cdot C^{\perp}\big)$$

$$C\in G(k,n)$$
$$k\equiv2n_B+n_W-n_I$$



$$\begin{array}{ccc}1 & 2 & \mathrm{I}\\ \hline \end{array}\qquad\qquad \begin{array}{ccc}\mathrm{I'} & 3 & 4\\ \hline \end{array}$$

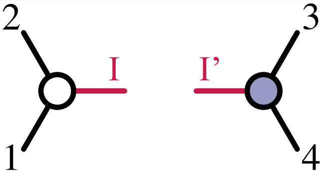
$$\begin{pmatrix}1 & w_2 & w_{\mathrm{I}}\end{pmatrix}\qquad\qquad\begin{pmatrix}1 & 0 & b_4^1\\ 0 & 1 & b_4^2\end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \quad \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$



$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$

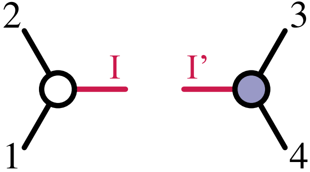
$$\begin{array}{ccc} \mathrm{I}' & 3 & 4 \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \quad \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$



$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$

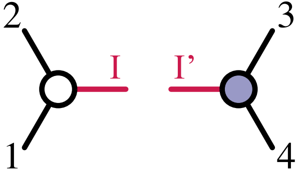
$$\begin{array}{ccc} \mathrm{I'} & 3 & 4 \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \;\; \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$



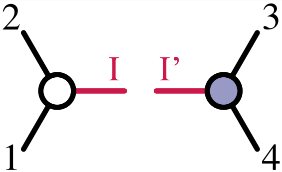
$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline 1 & w_2 & w_{\mathrm{I}} \end{array} \qquad \begin{array}{ccc} \mathrm{I'} & 3 & 4 \\ \hline 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{array}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f\equiv\int\Omega_C\;\;\delta^{k\times4}\big(C\cdot\widetilde{\eta}\big)\delta^{k\times2}\big(C\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C^{\perp}\big)$$

$$C\in G(k,n)$$
$$k\equiv2n_B+n_W-n_I$$



$$\begin{array}{ccc}1 & 2 & \mathrm{I}\\ \hline \end{array}\qquad\begin{array}{ccc}\mathrm{I'} & 3 & 4\\ \hline \end{array}$$

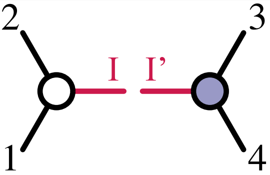
$$\begin{pmatrix}1 & w_2 & w_{\mathrm{I}}\end{pmatrix}\qquad\begin{pmatrix}1 & 0 & b_4^1\\ 0 & 1 & b_4^2\end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \;\; \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n) \qquad k \equiv 2n_B + n_W - n_I$$



$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \end{array} \qquad \begin{array}{ccc} \mathrm{I'} & 3 & 4 \\ \hline \end{array}$$

$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$

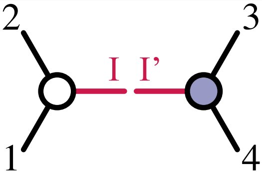
$$\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \quad \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n) \qquad k \equiv 2n_B + n_W - n_I$$



$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \end{array} \qquad \begin{array}{ccc} \mathrm{I}' & 3 & 4 \\ \hline \end{array}$$

$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$

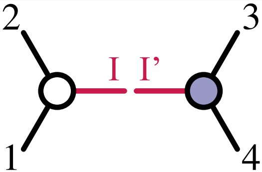
$$\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \quad \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$



$$\begin{array}{ccc} 1 & 2 & \mathrm{I} \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & w_2 & w_{\mathrm{I}} \end{pmatrix}$$

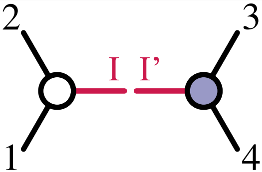
$$\begin{array}{ccc} \mathrm{I'} & 3 & 4 \\ \hline \end{array}$$
$$\begin{pmatrix} 1 & 0 & b_4^1 \\ 0 & 1 & b_4^2 \end{pmatrix}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \;\; \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$



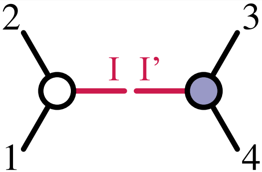
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & \mathrm{I} & \mathrm{I'} & 3 & 4 \end{array} \\ \left(\begin{array}{cccccc} 1 & w_2 & w_{\mathrm{I}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & b_4^1 \\ 0 & 0 & 0 & 0 & 1 & b_4^2 \end{array}\right) \end{array}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f\equiv\int\Omega_C\quad\delta^{k\times4}(C\cdot\widetilde{\eta})\,\delta^{k\times2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C\in G(k,n)$$
$$k\equiv2n_B+n_W-n_I$$



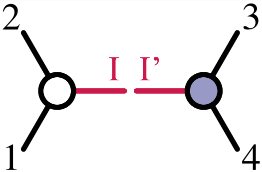
$$C\equiv\begin{array}{c}\begin{array}{cccccc}1&2&\mathrm{I}&\mathrm{I'}&3&4\end{array}\\ \left(\begin{array}{cccccc}1&w_2&w_{\mathrm{I}}&0&0&0\\0&0&0&1&0&b_4^1\\0&0&0&0&1&b_4^2\end{array}\right)\end{array}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f\equiv\int\Omega_C\;\;\delta^{k\times4}(C\cdot\widetilde{\eta})\,\delta^{k\times2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C\in G(k,n)\qquad k\equiv 2n_B+n_W-n_I$$



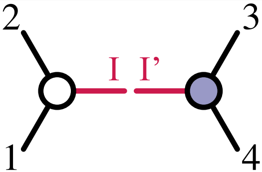
$$C\equiv\begin{array}{c}\begin{array}{cccccc}1 & 2 & \mathrm{I} & \mathrm{I}' & 3 & 4\end{array}\\ \left(\begin{array}{cccccc}1 & w_2 & w_{\mathrm{I}} & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & b_4^1\\ 0 & 0 & 0 & 0 & 1 & b_4^2\end{array}\right)\end{array}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f\equiv\int\Omega_C\;\;\delta^{k\times4}\big(C\cdot\widetilde{\eta}\big)\delta^{k\times2}\big(C\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C^{\perp}\big)$$

$$C\in G(k,n)$$
$$k\equiv 2n_B+n_W-n_I$$



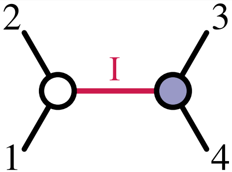
$$C\equiv\begin{array}{c}\begin{array}{cccccc}1 & 2 & \mathrm{I} & \mathrm{I'} & 3 & 4\end{array}\\ \left(\begin{array}{cccccc}1 & w_2 & w_{\mathrm{I}} & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & b_4^1\\ 0 & 0 & 0 & 0 & 1 & b_4^2\end{array}\right)\end{array}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$\boxed{f\equiv\int\Omega_C\;\;\delta^{k\times4}\big(C\cdot\widetilde{\eta}\big)\delta^{k\times2}\big(C\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C^{\perp}\big)}$$

$$C\in G(k,n)$$
$$k\equiv 2n_B+n_W-n_I$$



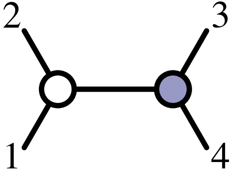
$$C\equiv\begin{array}{c}\begin{array}{cccccc}1&2&\mathrm{I}&\mathrm{I'}&3&4\end{array}\\\left(\begin{array}{cccccc}1&w_2&w_{\mathrm{I}}&0&0&0\\0&0&0&1&0&b_4^1\\0&0&0&0&1&b_4^2\end{array}\right)\end{array}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \;\; \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$



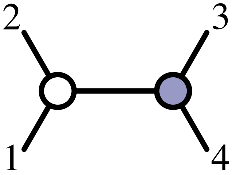
$$C \equiv \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{pmatrix} 1 & w_2 & 0 & b_4^1 \\ 0 & 0 & 1 & b_4^2 \end{pmatrix} \end{array}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$\boxed{f\equiv\int\Omega_C\;\;\delta^{k\times4}\big(C\cdot\widetilde{\eta}\big)\delta^{k\times2}\big(C\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C^{\perp}\big)}$$

$$C\in G(k,n)$$
$$k\equiv 2n_B+n_W-n_I$$



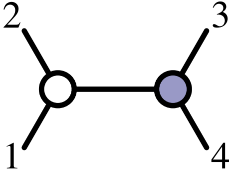
$$C\equiv\begin{array}{c}\begin{array}{cccc}1 & 2 & 3 & 4\end{array}\\ \begin{pmatrix}1 & w_2 & 0 & b_4^1\\ 0 & 0 & 1 & b_4^2\end{pmatrix}\end{array}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \quad \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n)$$
$$k \equiv 2n_B + n_W - n_I$$



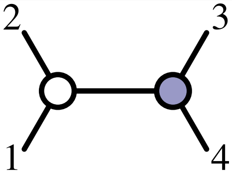
$$C \equiv \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 1 & w_2 & 0 & b_4^1 \\ 0 & 0 & 1 & b_4^2 \end{pmatrix} \end{array}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B\in G(2,3)$ and $W\in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f\equiv\int\Omega_C\quad\delta^{k\times 4}\big(C\cdot\widetilde{\eta}\big)\delta^{k\times 2}\big(C\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C^{\perp}\big)$$

$$C\in G(k,n)$$
$$k\equiv 2n_B+n_W-n_I$$



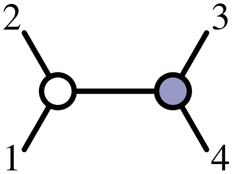
$$C\equiv\begin{array}{c}\begin{array}{cccc}1&2&3&4\end{array}\\ \left(\begin{array}{cccc}1&w_2&0&b_4^1\\0&0&1&b_4^2\end{array}\right)\end{array}$$

Grassmannian Representations of On-Shell Functions

In order to **linearize** momentum conservation at each three-particle vertex, (and to specify *which* of the solutions to three-particle kinematics to use) we introduce **auxiliary** $B \in G(2,3)$ and $W \in G(1,3)$ for each vertex—allowing us to represent all on-shell functions in the form:

$$f \equiv \int \Omega_C \;\; \delta^{k\times 4}(C\cdot\widetilde{\eta})\,\delta^{k\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times(n-k)}(\lambda\cdot C^{\perp})$$

$$C \in G(k,n) \qquad k \equiv 2n_B + n_W - n_I$$



$$C \equiv \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 1 & w_2 & 0 & b_4^1 \\ 0 & 0 & 1 & b_4^2 \end{pmatrix} \end{array}$$

Building-Up On-Shell Diagrams with “BCFW” Bridges

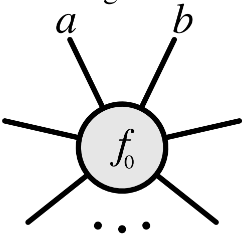
Very complex on-shell diagrams can be constructed by successively adding “BCFW” bridges to diagrams

Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):

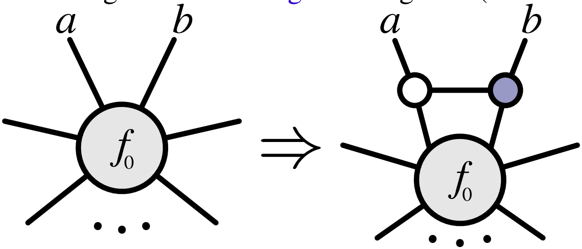
Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):



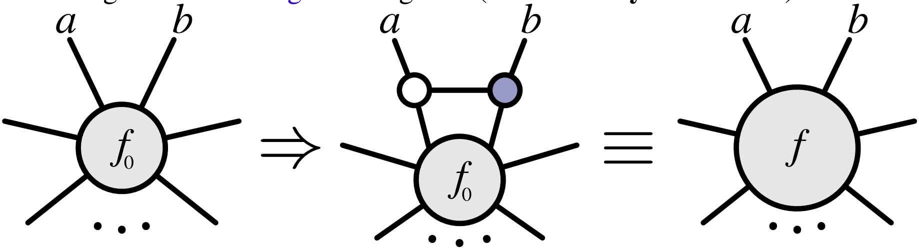
Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):



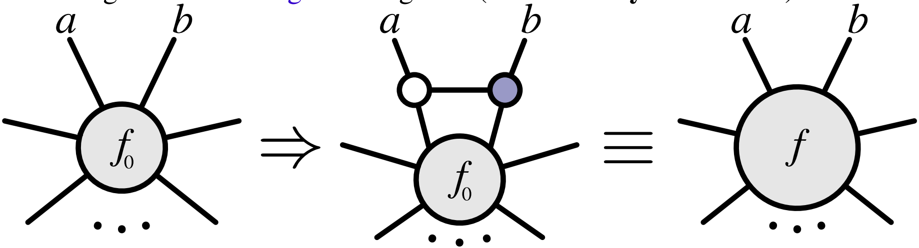
Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):



Building-Up On-Shell Diagrams with "BCFW" Bridges

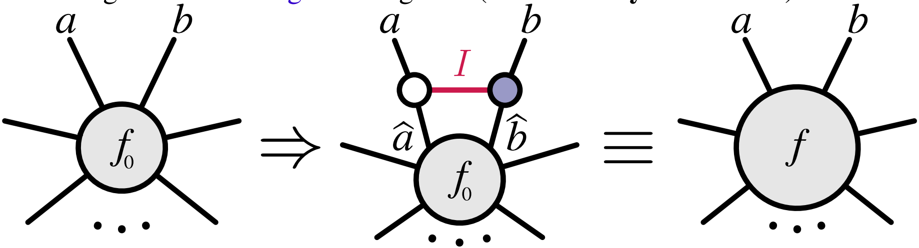
Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):



Adding the bridge has the effect of shifting the momenta p_a and p_b flowing into the diagram f_0 according to:

Building-Up On-Shell Diagrams with "BCFW" Bridges

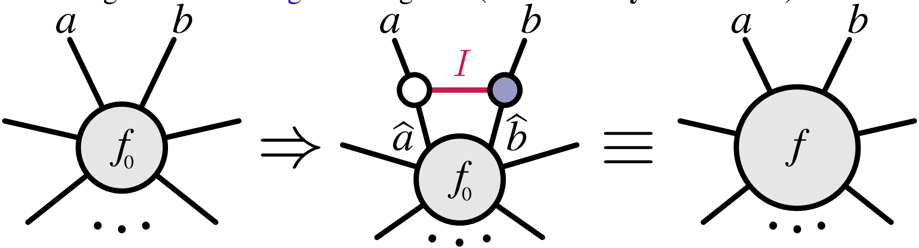
Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):



Adding the bridge has the effect of shifting the momenta p_a and p_b flowing into the diagram f_0 according to:

Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):

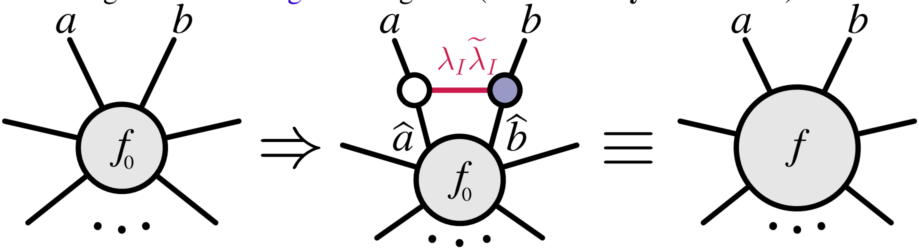


Adding the bridge has the effect of shifting the momenta p_a and p_b flowing into the diagram f_0 according to:

$$\lambda_a\widetilde{\lambda}_a \mapsto \lambda_{\widehat{a}}\widetilde{\lambda}_{\widehat{a}} = \lambda_a\widetilde{\lambda}_a - \lambda_I\,\widetilde{\lambda}_I \quad \text{and} \quad \lambda_b\widetilde{\lambda}_b \mapsto \lambda_{\widehat{b}}\widetilde{\lambda}_{\widehat{b}} = \lambda_b\widetilde{\lambda}_b + \lambda_I\,\widetilde{\lambda}_I\,,$$

Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):

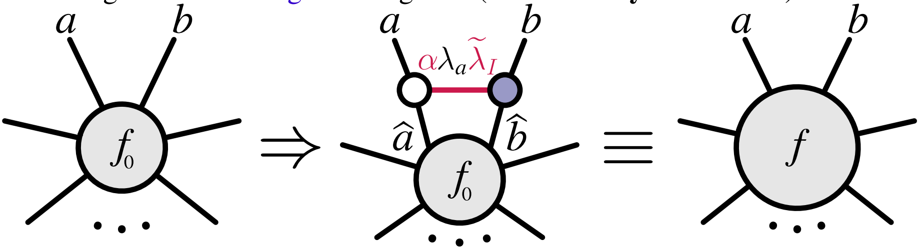


Adding the bridge has the effect of shifting the momenta p_a and p_b flowing into the diagram f_0 according to:

$$\lambda_a\widetilde{\lambda}_a \mapsto \lambda_{\hat{a}}\widetilde{\lambda}_{\hat{a}} = \lambda_a\widetilde{\lambda}_a - \lambda_I\widetilde{\lambda}_I \quad \text{and} \quad \lambda_b\widetilde{\lambda}_b \mapsto \lambda_{\hat{b}}\widetilde{\lambda}_{\hat{b}} = \lambda_b\widetilde{\lambda}_b + \lambda_I\widetilde{\lambda}_I\,,$$

Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):

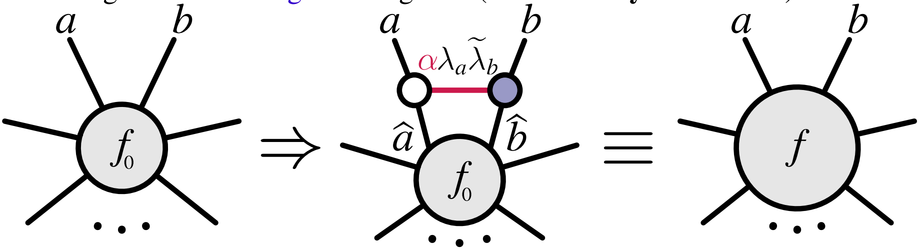


Adding the bridge has the effect of shifting the momenta p_a and p_b flowing into the diagram f_0 according to:

$$\lambda_a\widetilde{\lambda}_a \mapsto \lambda_{\hat{a}}\widetilde{\lambda}_{\hat{a}} = \lambda_a\widetilde{\lambda}_a - \alpha\lambda_a\widetilde{\lambda}_I \quad \text{and} \quad \lambda_b\widetilde{\lambda}_b \mapsto \lambda_{\hat{b}}\widetilde{\lambda}_{\hat{b}} = \lambda_b\widetilde{\lambda}_b + \alpha\lambda_a\widetilde{\lambda}_I\,,$$

Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):

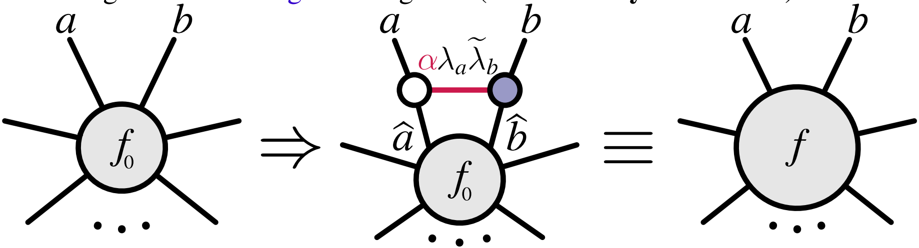


Adding the bridge has the effect of shifting the momenta p_a and p_b flowing into the diagram f_0 according to:

$$\lambda_a\widetilde{\lambda}_a \mapsto \lambda_{\hat{a}}\widetilde{\lambda}_{\hat{a}} = \lambda_a\widetilde{\lambda}_a - \alpha\lambda_a\widetilde{\lambda}_b \quad \text{and} \quad \lambda_b\widetilde{\lambda}_b \mapsto \lambda_{\hat{b}}\widetilde{\lambda}_{\hat{b}} = \lambda_b\widetilde{\lambda}_b + \alpha\lambda_a\widetilde{\lambda}_b,$$

Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):

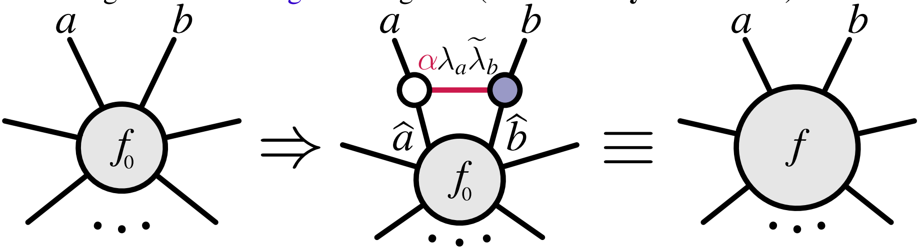


Adding the bridge has the effect of shifting the momenta p_a and p_b flowing into the diagram f_0 according to:

$$\lambda_a\widetilde{\lambda}_a \mapsto \lambda_{\widehat{a}}\widetilde{\lambda}_{\widehat{a}} = \lambda_a\big(\widetilde{\lambda}_a - \alpha\widetilde{\lambda}_b\big) \quad \text{and} \quad \lambda_b\widetilde{\lambda}_b \mapsto \lambda_{\widehat{b}}\widetilde{\lambda}_{\widehat{b}} = \lambda_b\widetilde{\lambda}_b + \alpha\lambda_a\widetilde{\lambda}_b,$$

Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):

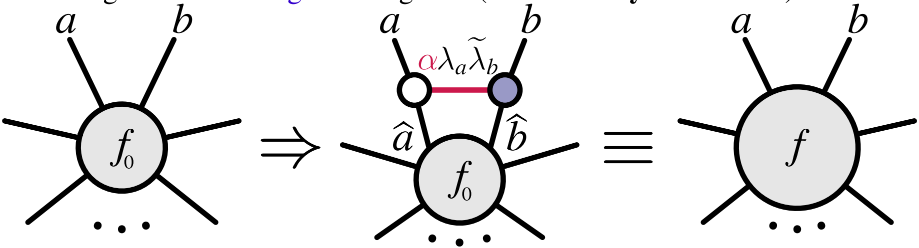


Adding the bridge has the effect of shifting the momenta p_a and p_b flowing into the diagram f_0 according to:

$$\lambda_a\widetilde{\lambda}_a \mapsto \lambda_{\hat{a}}\widetilde{\lambda}_{\hat{a}} = \lambda_a\left(\widetilde{\lambda}_a - \alpha\widetilde{\lambda}_b\right) \quad \text{and} \quad \lambda_b\widetilde{\lambda}_b \mapsto \lambda_{\hat{b}}\widetilde{\lambda}_{\hat{b}} = \left(\lambda_b + \alpha\lambda_a\right)\widetilde{\lambda}_b,$$

Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):



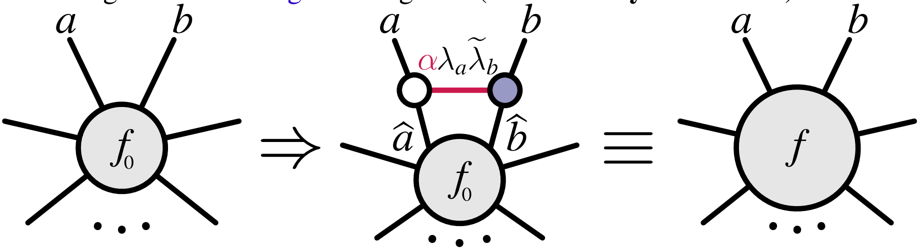
Adding the bridge has the effect of shifting the momenta p_a and p_b flowing into the diagram f_0 according to:

$$\lambda_a\widetilde{\lambda}_a \mapsto \lambda_{\hat{a}}\widetilde{\lambda}_{\hat{a}} = \lambda_a\big(\widetilde{\lambda}_a - \alpha\widetilde{\lambda}_b\big) \quad \text{and} \quad \lambda_b\widetilde{\lambda}_b \mapsto \lambda_{\hat{b}}\widetilde{\lambda}_{\hat{b}} = \big(\lambda_b + \alpha\lambda_a\big)\widetilde{\lambda}_b,$$

introducing a new parameter α, in terms of which we may write:

Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):



Adding the bridge has the effect of shifting the momenta p_a and p_b flowing into the diagram f_0 according to:

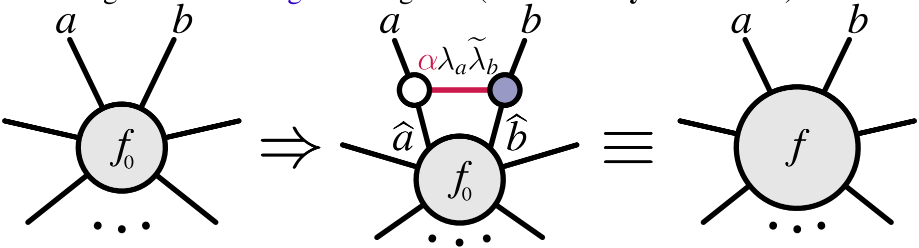
$$\lambda_a\widetilde{\lambda}_a \mapsto \lambda_{\widehat{a}}\widetilde{\lambda}_{\widehat{a}} = \lambda_a\big(\widetilde{\lambda}_a - \alpha\widetilde{\lambda}_b\big) \quad \text{and} \quad \lambda_b\widetilde{\lambda}_b \mapsto \lambda_{\widehat{b}}\widetilde{\lambda}_{\widehat{b}} = \big(\lambda_b + \alpha\lambda_a\big)\widetilde{\lambda}_b,$$

introducing a new parameter α, in terms of which we may write:

$$f(\ldots,a,b,\ldots) = \frac{d\alpha}{\alpha} f_0(\ldots,\widehat{a},\widehat{b},\ldots)$$

Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):



Adding the bridge has the effect of shifting the momenta p_a and p_b flowing into the diagram f_0 according to:

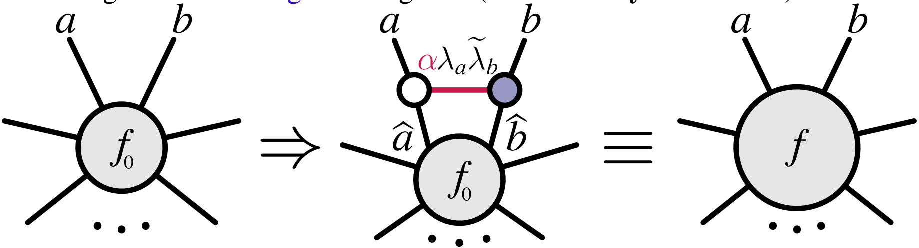
$$\lambda_a\widetilde{\lambda}_a \mapsto \lambda_{\widehat{a}}\widetilde{\lambda}_{\widehat{a}} = \lambda_a\big(\widetilde{\lambda}_a - \alpha\widetilde{\lambda}_b\big) \quad \text{and} \quad \lambda_b\widetilde{\lambda}_b \mapsto \lambda_{\widehat{b}}\widetilde{\lambda}_{\widehat{b}} = \big(\lambda_b + \alpha\lambda_a\big)\widetilde{\lambda}_b,$$

introducing a new parameter α, in terms of which we may write:

$$f(\ldots,a,b,\ldots) = \frac{d\alpha}{\alpha} f_0(\ldots,\widehat{a},\widehat{b},\ldots)$$

Building-Up On-Shell Diagrams with "BCFW" Bridges

Very complex on-shell diagrams can be constructed by successively adding "BCFW" bridges to diagrams (an **extremely** useful tool!):



Adding the bridge has the effect of shifting the momenta p_a and p_b flowing into the diagram f_0 according to:

$$\lambda_a\widetilde{\lambda}_a \mapsto \lambda_{\widehat{a}}\widetilde{\lambda}_{\widehat{a}} = \lambda_a\big(\widetilde{\lambda}_a - \alpha\widetilde{\lambda}_b\big) \quad \text{and} \quad \lambda_b\widetilde{\lambda}_b \mapsto \lambda_{\widehat{b}}\widetilde{\lambda}_{\widehat{b}} = \big(\lambda_b + \alpha\lambda_a\big)\widetilde{\lambda}_b,$$

introducing a new parameter α, in terms of which we may write:

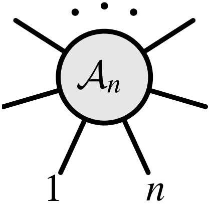
$$f(\ldots,a,b,\ldots) = \frac{d\alpha}{\alpha} f_0(\ldots,\widehat{a},\widehat{b},\ldots)$$

The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude

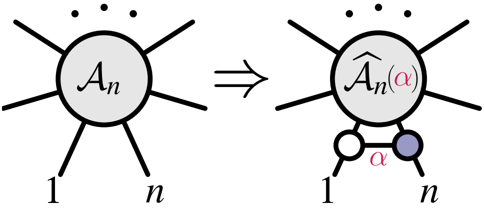
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude:



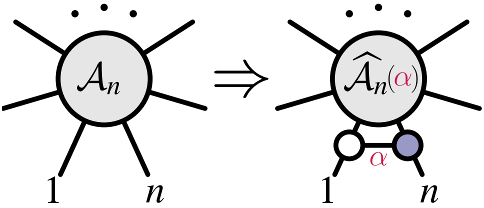
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude:



The Analytic Boot-Strap: All-Loop Recursion Relations

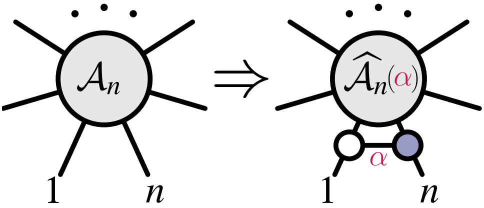
Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:



The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

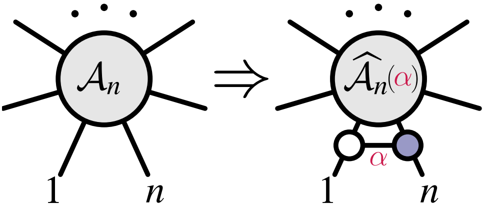


The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha = 0$ for (minus) the sum of residues away from the origin:

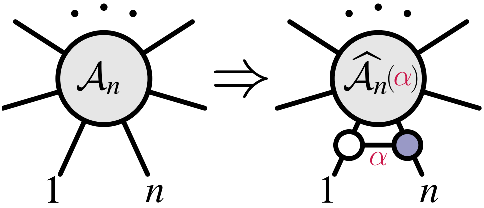


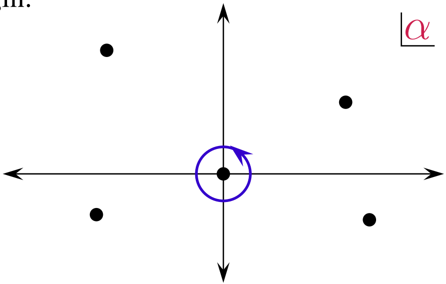
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



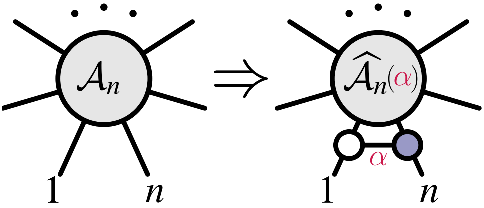


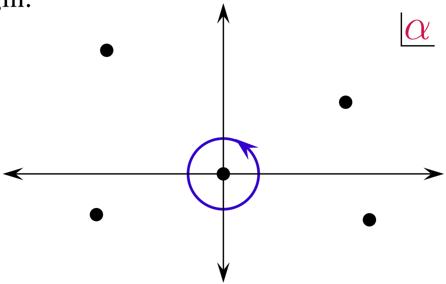
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha = 0$ for (minus) the sum of residues away from the origin:



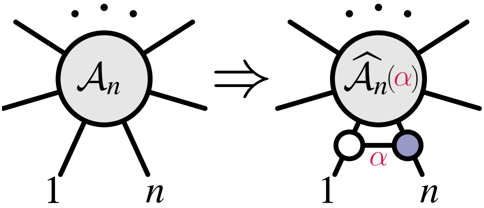


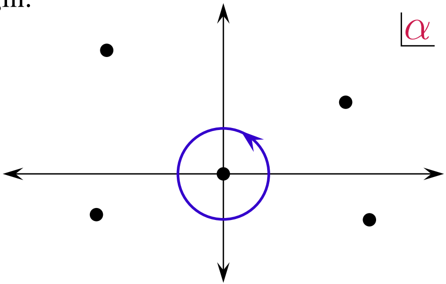
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha = 0$ for (minus) the sum of residues away from the origin:



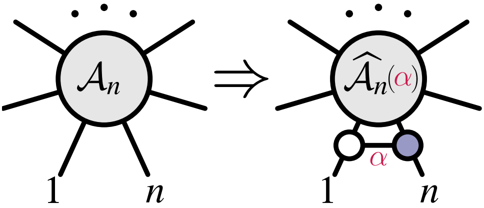


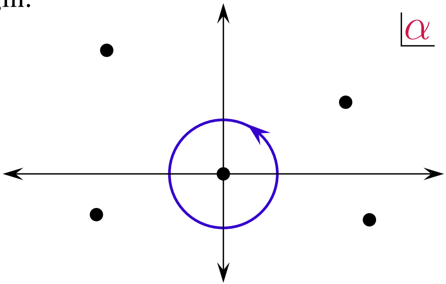
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha = 0$ for (minus) the sum of residues away from the origin:



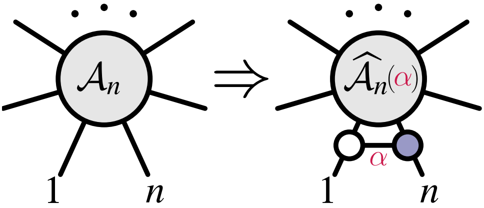


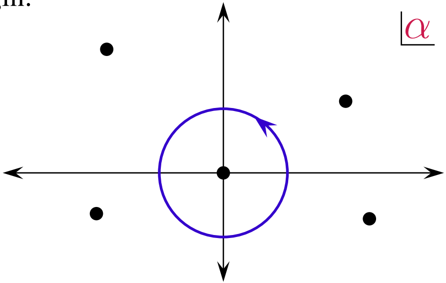
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n=\widehat{\mathcal{A}}_n(\alpha\to0)\propto\oint\limits_{\alpha=0}\frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



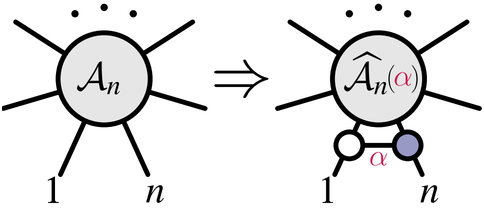


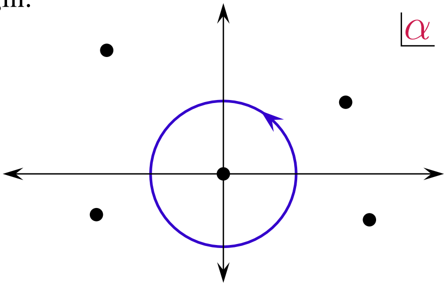
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha = 0$ for (minus) the sum of residues away from the origin:



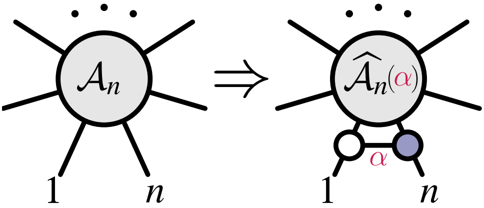


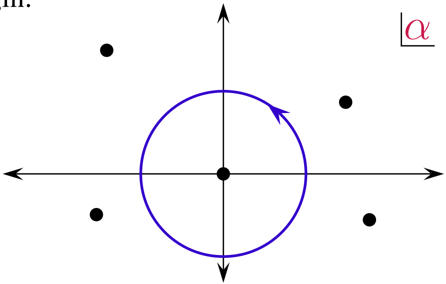
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha = 0$ for (minus) the sum of residues away from the origin:



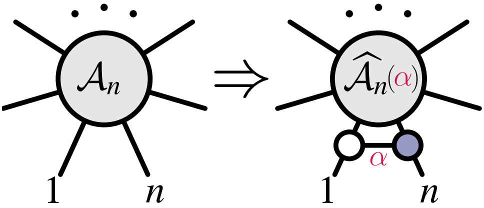


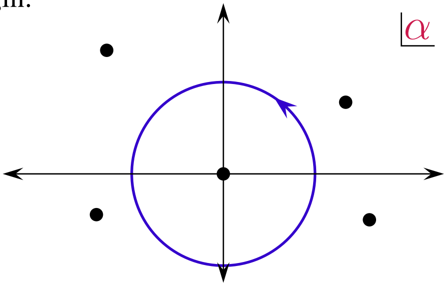
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\; \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



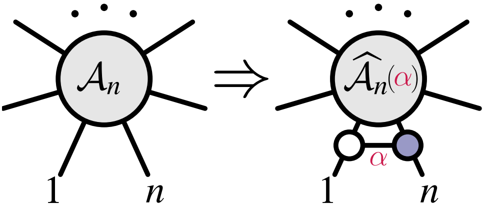


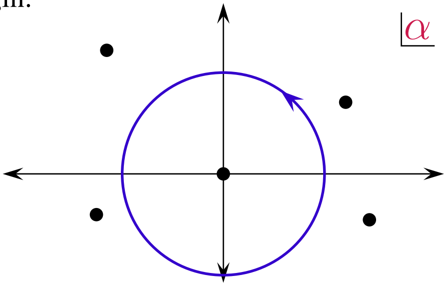
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint_{\alpha=0} \frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



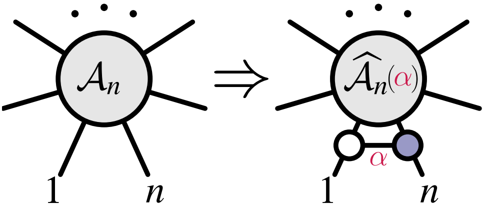


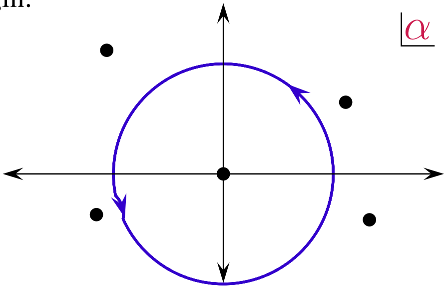
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



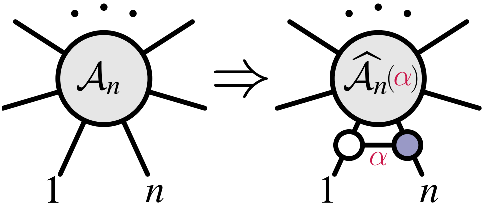


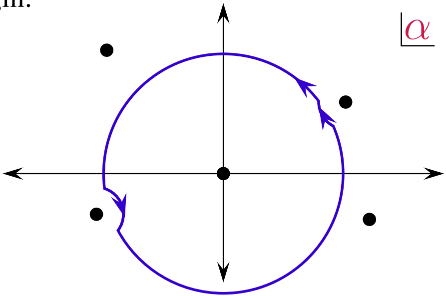
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint_{\alpha=0} \frac{d\alpha}{\alpha}\;\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



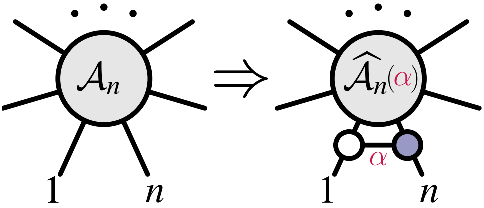


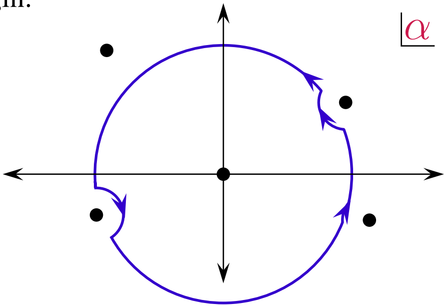
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



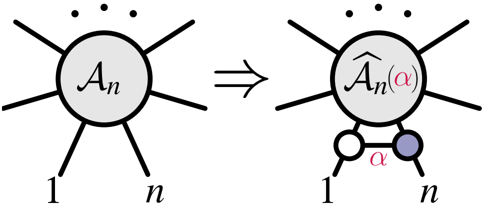


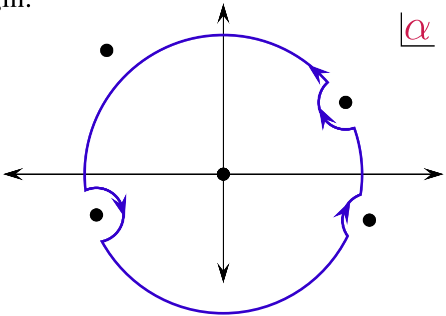
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha = 0$ for (minus) the sum of residues away from the origin:



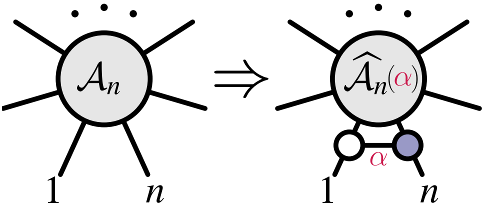


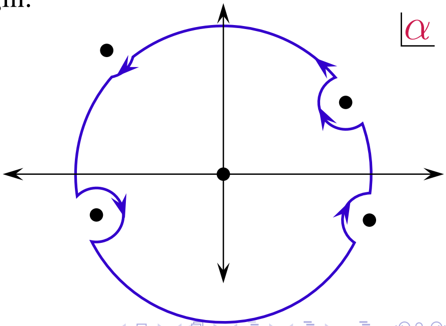
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha = 0$ for (minus) the sum of residues away from the origin:



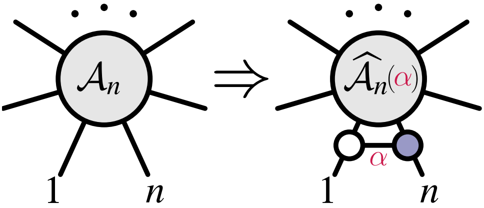


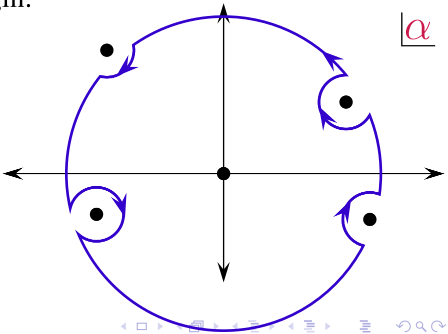
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude
the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha = 0$ for (minus)
the sum of residues away from the origin:



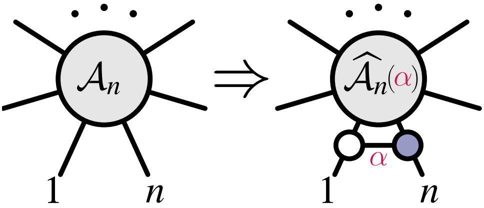


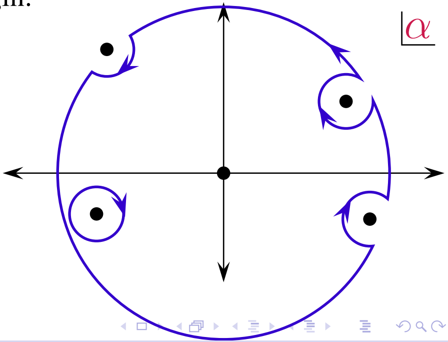
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint\limits_{\alpha=0}\frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



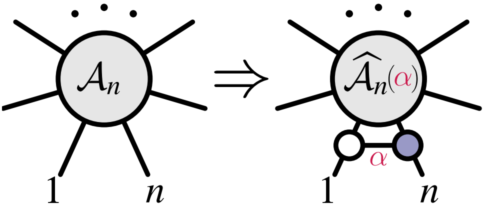


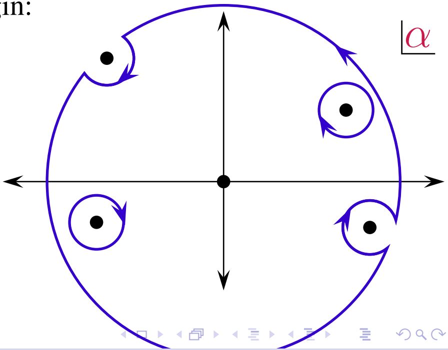
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint_{\alpha=0} \frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



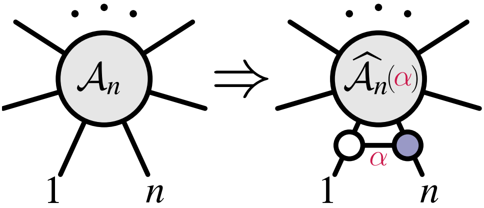


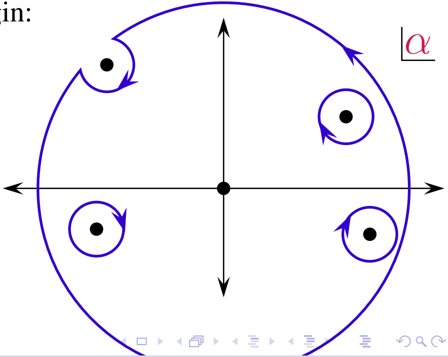
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint_{\alpha=0} \frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



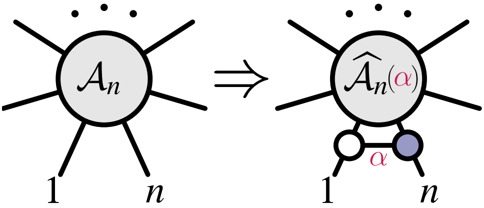


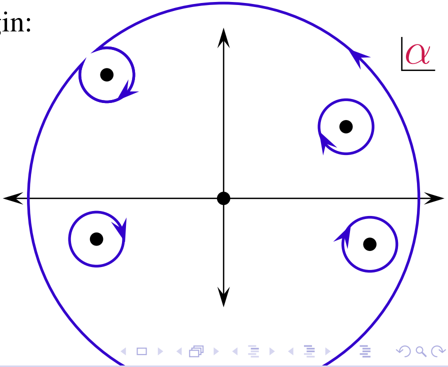
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha = 0$ for (minus) the sum of residues away from the origin:



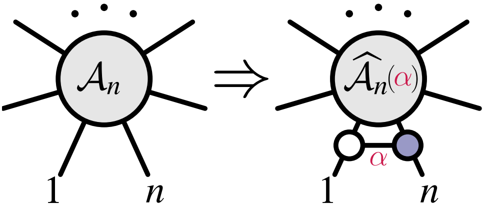


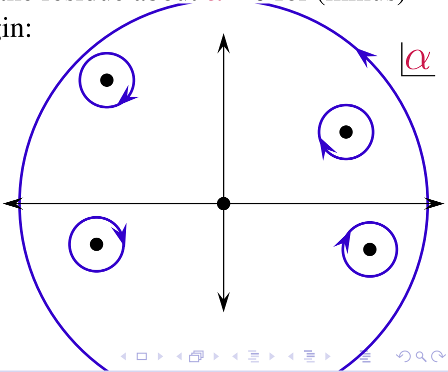
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



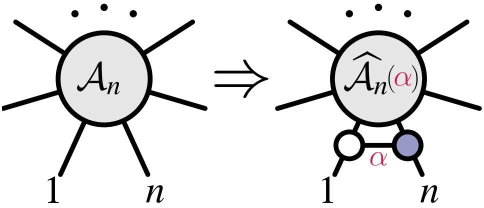


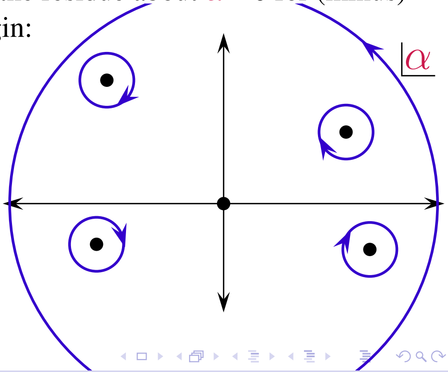
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



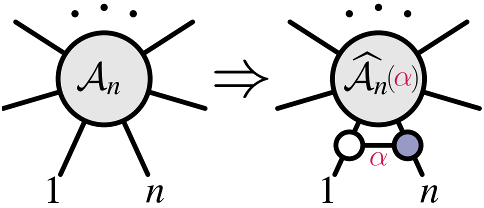


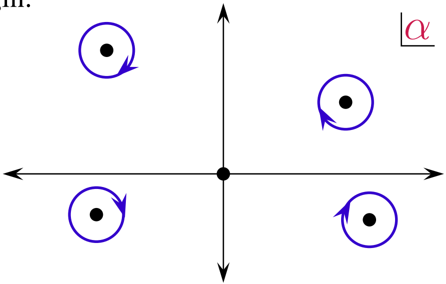
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



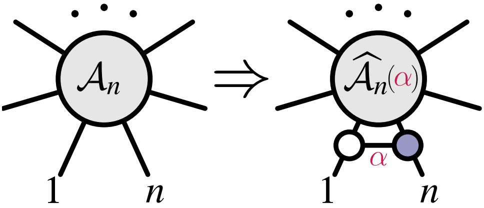


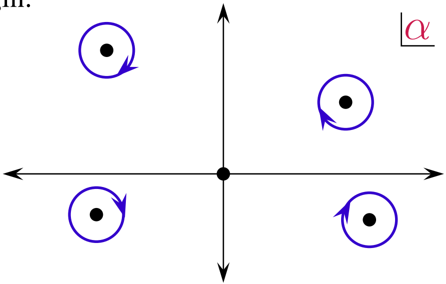
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha\!=\!0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\!\to\!0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha\!=\!0$ for (minus) the sum of residues away from the origin:



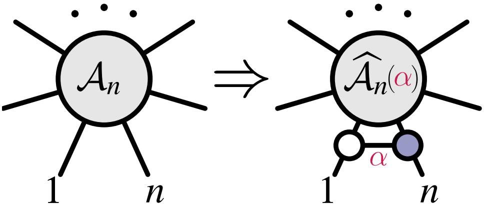


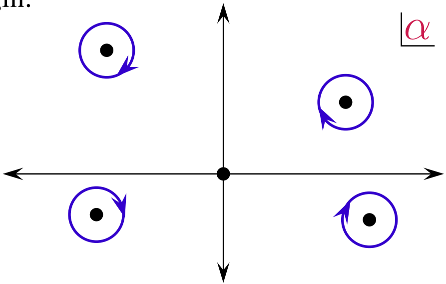
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude
the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint_{\alpha=0}\frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin:



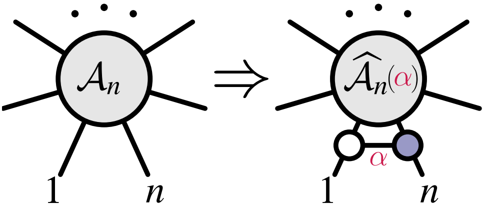


The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha\!=\!0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\!\to\!0) \propto \oint\limits_{\alpha=0}\frac{d\alpha}{\alpha}\;\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha\!=\!0$ for (minus) the sum of residues away from the origin—these come in two types:



The Analytic Boot-Strap: All-Loop Recursion Relations

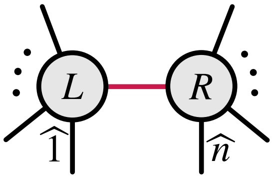
Consider adding a BCFW bridge to the full n-particle scattering amplitude

the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin—these come in two types:

factorization-channels

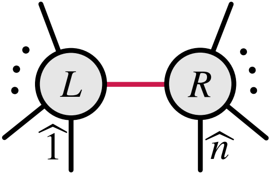


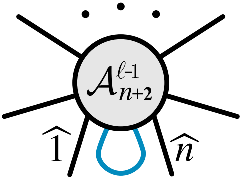
The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude
the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha=0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha\to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\,\widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha=0$ for (minus) the sum of residues away from the origin—these come in two types: factorization-channels and forward-limits



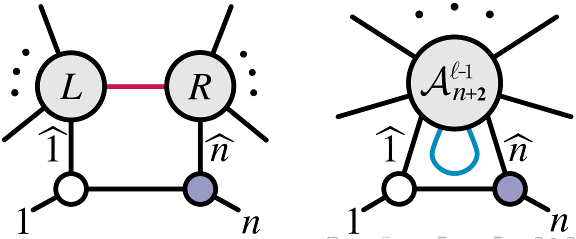


The Analytic Boot-Strap: All-Loop Recursion Relations

Consider adding a BCFW bridge to the full n-particle scattering amplitude
the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha = 0$ for (minus)
the sum of residues away from the origin—these come in two types:
factorization-channels and forward-limits

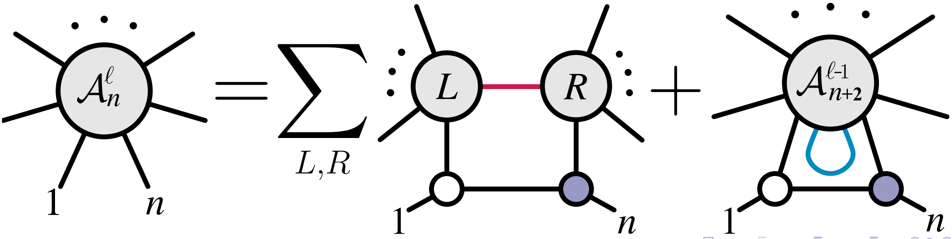


The Analytic Boot-Strap: All-Loop Recursion Relations

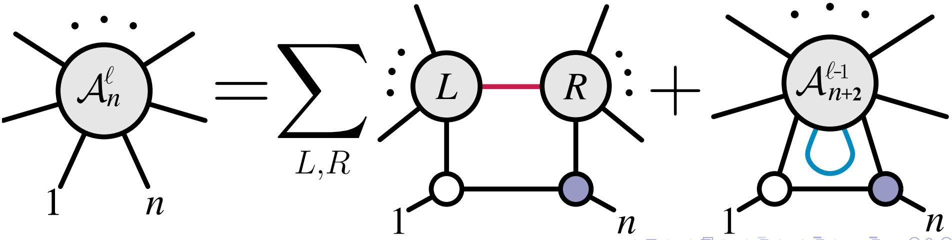
Consider adding a BCFW bridge to the full n-particle scattering amplitude the undeformed amplitude $\mathcal{A}_n$ is recovered as the **residue** about $\alpha = 0$:

$$\mathcal{A}_n = \widehat{\mathcal{A}}_n(\alpha \to 0) \propto \oint\limits_{\alpha=0} \frac{d\alpha}{\alpha}\, \widehat{\mathcal{A}}_n(\alpha)$$

We can use **Cauchy's theorem** to trade the residue about $\alpha = 0$ for (minus) the sum of residues away from the origin—these come in two types: factorization-channels and forward-limits

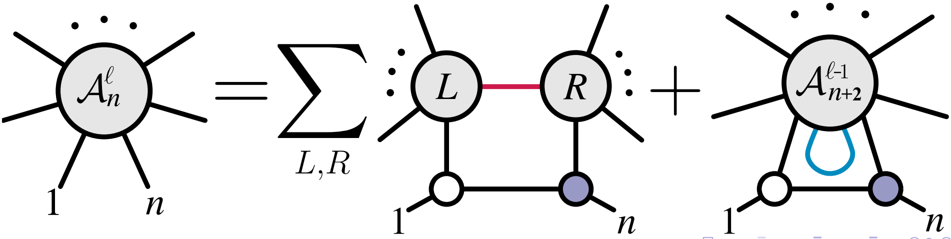


The Analytic Boot-Strap: All-Loop Recursion Relations



The Analytic Boot-Strap: All-Loop Recursion Relations

Forward-limits and loop-momenta:

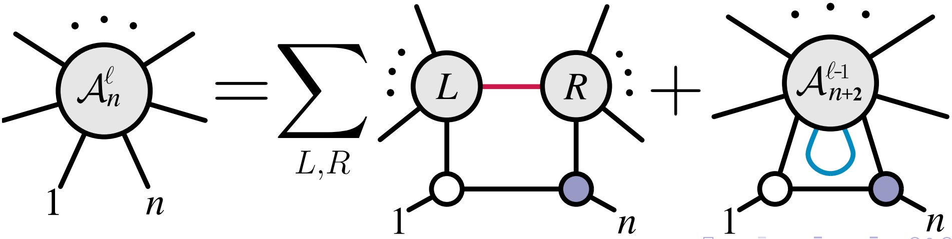


The Analytic Boot-Strap: All-Loop Recursion Relations

Forward-limits and loop-momenta:

the familiar "off-shell" loop-momentum is represented by on-shell data as:

$$\ell\equiv\lambda_I\widetilde{\lambda}_I+\alpha\lambda_1\widetilde{\lambda}_n\quad\text{with}\quad d^4\ell=\frac{d^2\lambda_I d^2\widetilde{\lambda}_I}{\mathrm{vol}(GL_1)}d\alpha\,\langle 1\,I\rangle\,[n\,I]$$

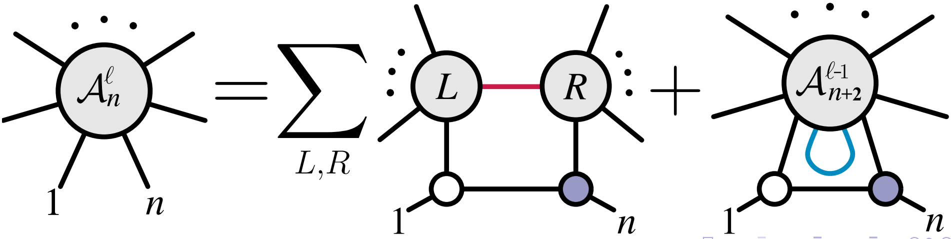


The Analytic Boot-Strap: All-Loop Recursion Relations

Forward-limits and loop-momenta:

the familiar "off-shell" loop-momentum is represented by on-shell data as:

$$\ell \equiv \lambda_I \widetilde{\lambda}_I + \alpha \lambda_1 \widetilde{\lambda}_n \quad \text{with} \quad d^4\ell = \frac{d^2\lambda_I d^2\widetilde{\lambda}_I}{\mathrm{vol}(GL_1)} d\alpha \, \langle 1\, I \rangle \, [n\, I]$$

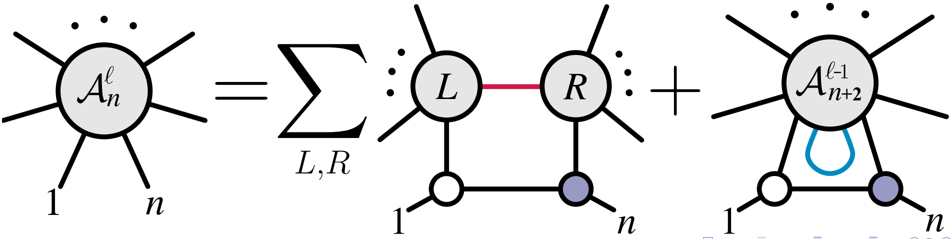


The Analytic Boot-Strap: All-Loop Recursion Relations

Forward-limits and loop-momenta:

the familiar "off-shell" loop-momentum is represented by on-shell data as:

$$\ell \equiv \lambda_I \widetilde{\lambda}_I + \alpha \lambda_1 \widetilde{\lambda}_n \quad \text{with} \quad d^4\ell = \frac{d^2\lambda_I d^2\widetilde{\lambda}_I}{\text{vol}(GL_1)} d\alpha \, \langle 1\, I\rangle \, [n\, I]$$

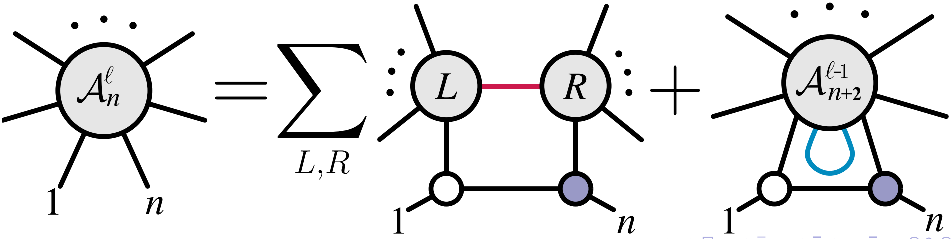


The Analytic Boot-Strap: All-Loop Recursion Relations

Forward-limits and loop-momenta:

the familiar "off-shell" loop-momentum is represented by on-shell data as:

$$\ell\equiv\lambda_I\widetilde{\lambda}_I+\alpha\lambda_1\widetilde{\lambda}_n\quad\text{with}\quad d^4\ell=\,d^3LIPS_I\;d\alpha\;\langle 1\,I\rangle\,[n\,I]$$

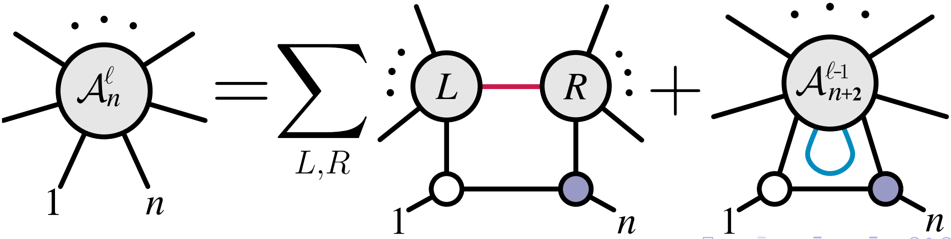


The Analytic Boot-Strap: All-Loop Recursion Relations

Forward-limits and loop-momenta:

the familiar "off-shell" loop-momentum is represented by on-shell data as:

$$\ell \equiv \lambda_I\widetilde{\lambda}_I + \alpha\lambda_1\widetilde{\lambda}_n \quad \text{with} \quad d^4\ell = d^3LIPS_I\; d\alpha\; \langle 1\, I\rangle\, [n\, I]$$

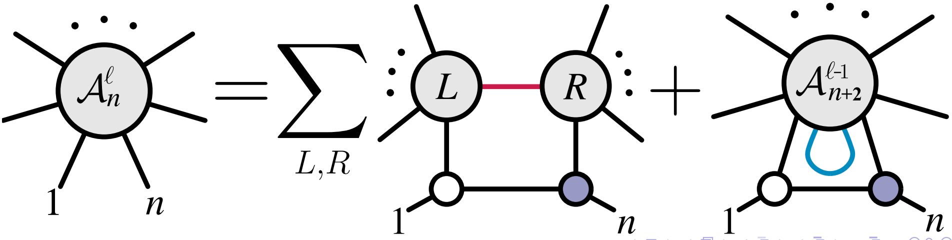


The Analytic Boot-Strap: All-Loop Recursion Relations

Forward-limits and loop-momenta:

the familiar "off-shell" loop-momentum is represented by on-shell data as:

$$\ell \equiv \lambda_I \widetilde{\lambda}_I + \alpha \lambda_1 \widetilde{\lambda}_n \quad \text{with} \quad d^4\ell = d^3 LIPS_I \; d\alpha \; \langle 1\, I\rangle \, [n\, I]$$



Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k\!=\!2$, $\mathcal{A}_n^{(2)}$!

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k\!=\!2$, $\mathcal{A}_n^{(2)}$! The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)}\bigotimes\mathcal{A}_3^{(1)}$:

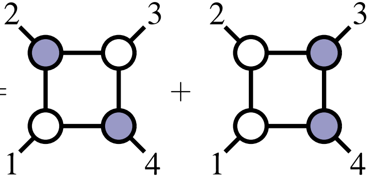
Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$! The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)} \bigotimes \mathcal{A}_3^{(1)}$:

$$\mathcal{A}_4^{(2)} =$$

Exempli Gratia: On-Shell Representations of Amplitudes

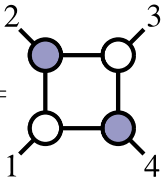
The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$! The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)}\bigotimes\mathcal{A}_3^{(1)}$:

$\mathcal{A}_4^{(2)} =$ 

Exempli Gratia: On-Shell Representations of Amplitudes

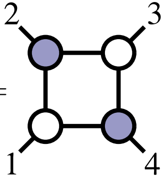
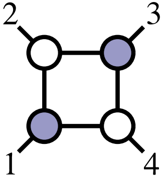
The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)} \otimes \mathcal{A}_3^{(1)}$:

$\mathcal{A}_4^{(2)} =$ 

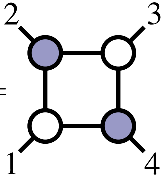
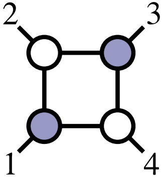
Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!
The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)} \bigotimes \mathcal{A}_3^{(1)}$:

$\mathcal{A}_4^{(2)} =$  $=$ 

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k\!=\!2$, $\mathcal{A}_n^{(2)}$!
The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)}\bigotimes\mathcal{A}_3^{(1)}$:

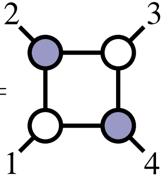
$\mathcal{A}_4^{(2)} =$  $=$ 

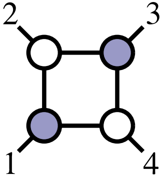
$\mathcal{A}_5^{(2)} =$

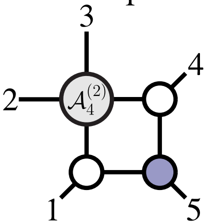
Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)} \otimes \mathcal{A}_3^{(1)}$:

$\mathcal{A}_4^{(2)} =$ 

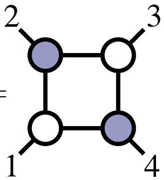
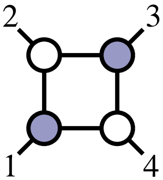
$=$ 

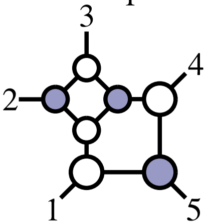
$\mathcal{A}_5^{(2)} =$ 

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)} \otimes \mathcal{A}_3^{(1)}$:

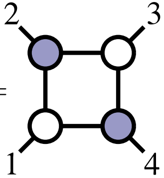
$\mathcal{A}_4^{(2)} =$  $=$ 

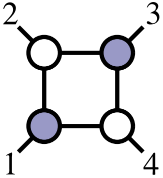
$\mathcal{A}_5^{(2)} =$ 

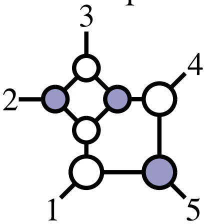
Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)} \otimes \mathcal{A}_3^{(1)}$:

$\mathcal{A}_4^{(2)} =$ 

$=$ 

$\mathcal{A}_5^{(2)} =$ 

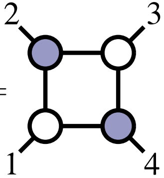
$\mathcal{A}_6^{(2)} =$

Exempli Gratia: On-Shell Representations of Amplitudes

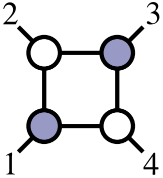
The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)} \otimes \mathcal{A}_3^{(1)}$:

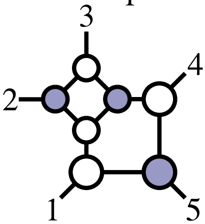
$\mathcal{A}_4^{(2)} =$



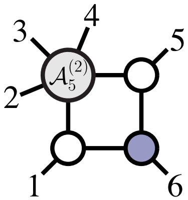
$=$



$\mathcal{A}_5^{(2)} =$



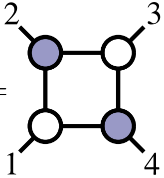
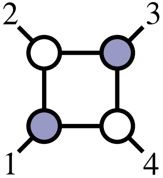
$\mathcal{A}_6^{(2)} =$

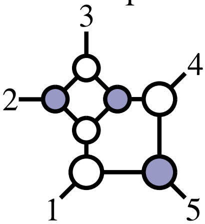


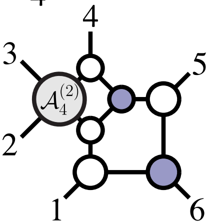
Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)} \bigotimes \mathcal{A}_3^{(1)}$:

$\mathcal{A}_4^{(2)} =$  $=$ 

$\mathcal{A}_5^{(2)} =$ 

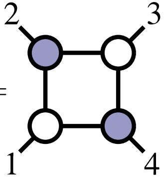
$\mathcal{A}_6^{(2)} =$ 

Exempli Gratia: On-Shell Representations of Amplitudes

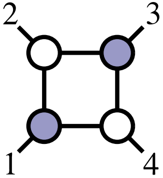
The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)} \bigotimes \mathcal{A}_3^{(1)}$:

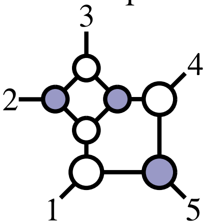
$\mathcal{A}_4^{(2)} =$



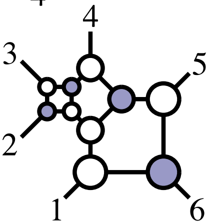
$=$



$\mathcal{A}_5^{(2)} =$

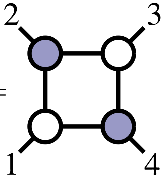


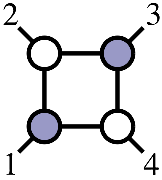
$\mathcal{A}_6^{(2)} =$

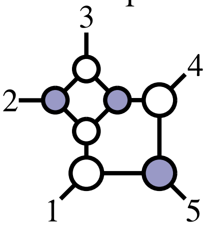


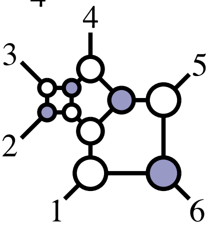
Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!
The **only** (non-vanishing) contribution to $\mathcal{A}_n^{(2)}$ is $\mathcal{A}_{n-1}^{(2)} \otimes \mathcal{A}_3^{(1)}$:

$\mathcal{A}_4^{(2)} =$ 

$=$ 

$\mathcal{A}_5^{(2)} =$ 

$\mathcal{A}_6^{(2)} =$ 

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

And it generates **very concise** formulae for all other amplitudes

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$! And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

Exempli Gratia: On-Shell Representations of Amplitudes

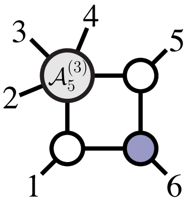
The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$! And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

$$\mathcal{A}_6^{(3)} =$$

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

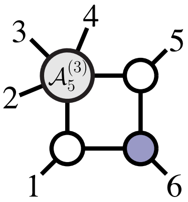
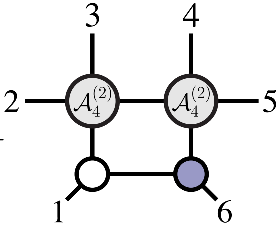
And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

$\mathcal{A}_6^{(3)} =$ 

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

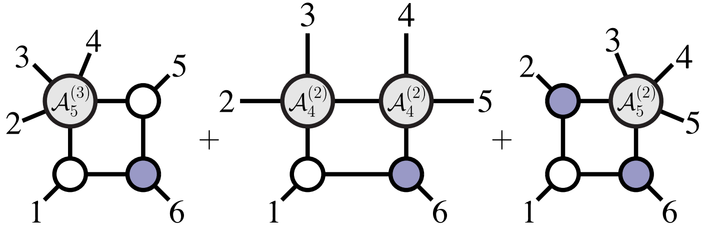
$\mathcal{A}_6^{(3)} =$  $+$ 

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

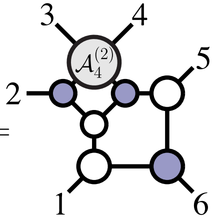
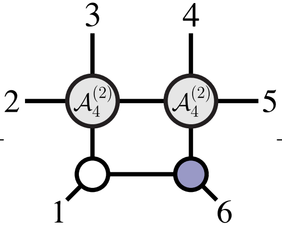
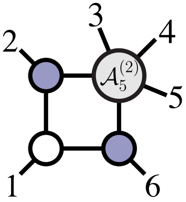
And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

$\mathcal{A}_6^{(3)} =$



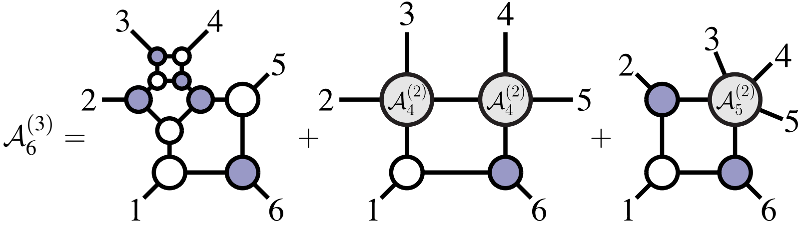
Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k\!=\!2$, $\mathcal{A}_n^{(2)}$! And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

$\mathcal{A}_6^{(3)} =$  $+$  $+$ 

Exempli Gratia: On-Shell Representations of Amplitudes

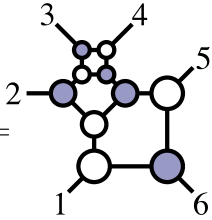
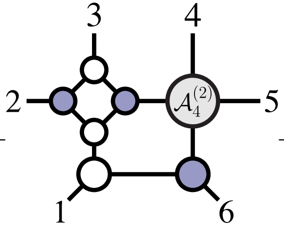
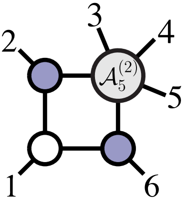
The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!
And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:



Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

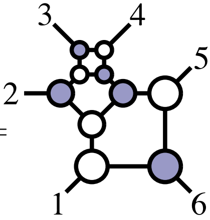
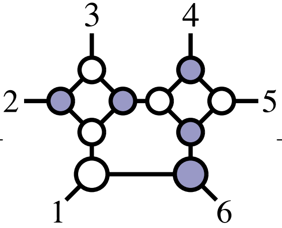
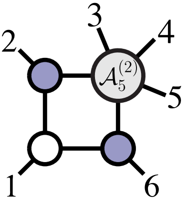
And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

$\mathcal{A}_6^{(3)} =$  $+$  $+$ 

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

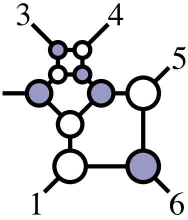
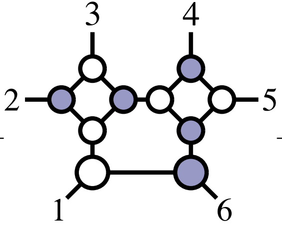
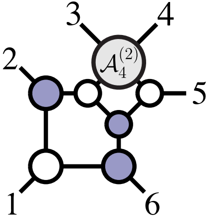
And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

$\mathcal{A}_6^{(3)} =$  $+$  $+$ 

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

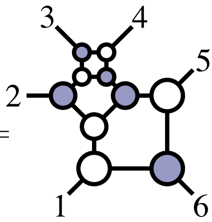
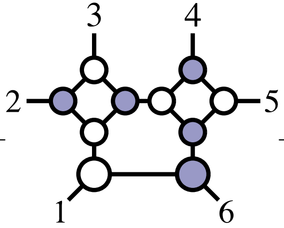
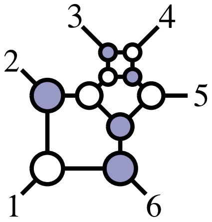
And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

$\mathcal{A}_6^{(3)} =$  $+$  $+$ 

Exempli Gratia: On-Shell Representations of Amplitudes

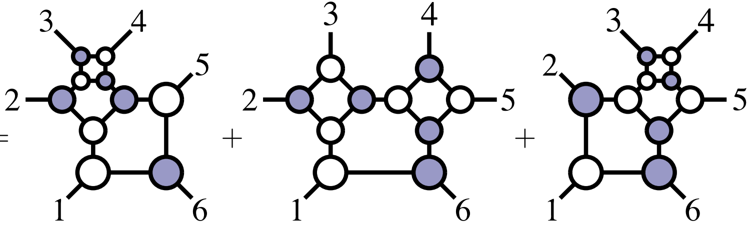
The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

$\mathcal{A}_6^{(3)} =$  $+$  $+$ 

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!
And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

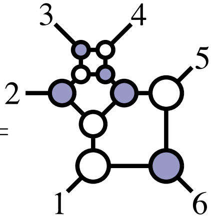
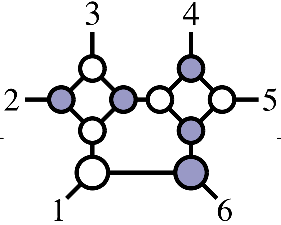
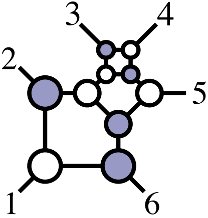
$\mathcal{A}_6^{(3)} =$ 

Observations regarding recursed representations of scattering amplitudes:

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!

And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

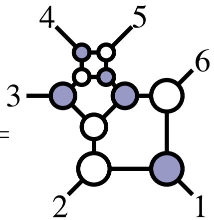
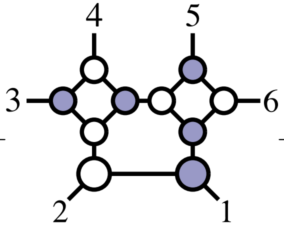
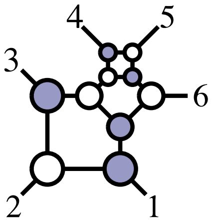
$\mathcal{A}_6^{(3)} =$  $+$  $+$ 

Observations regarding recursed representations of scattering amplitudes:

- varying recursion 'schema' can generate *many* 'BCFW formulae'

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$! And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

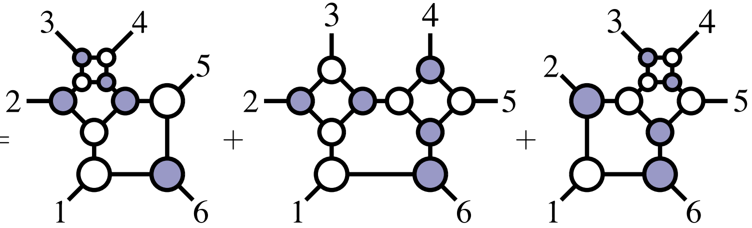
$\mathcal{A}_6^{(3)} =$  $+$  $+$ 

Observations regarding recursed representations of scattering amplitudes:

- varying recursion 'schema' can generate *many* 'BCFW formulae'

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k\!=\!2$, $\mathcal{A}_n^{(2)}$! And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

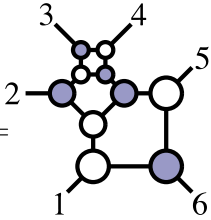
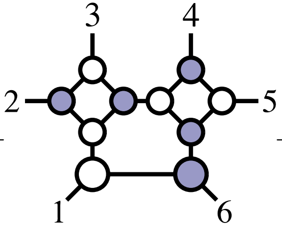
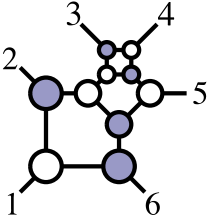
$\mathcal{A}_6^{(3)} =$ 

Observations regarding recursed representations of scattering amplitudes:

- varying recursion ‘schema’ can generate *many* ‘BCFW formulae’

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$! And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

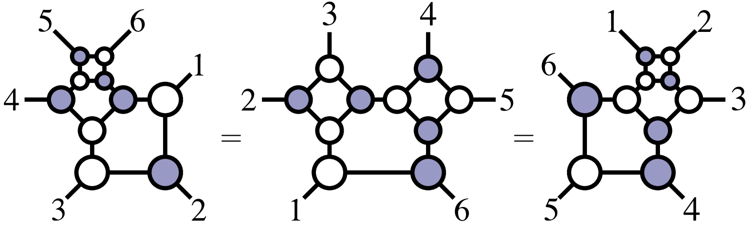
$\mathcal{A}_6^{(3)} =$  $+$  $+$ 

Observations regarding recursed representations of scattering amplitudes:

- varying recursion ‘schema’ can generate *many* ‘BCFW formulae’
- on-shell diagrams can often be related in surprising ways

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k\!=\!2$, $\mathcal{A}_n^{(2)}$!
And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:



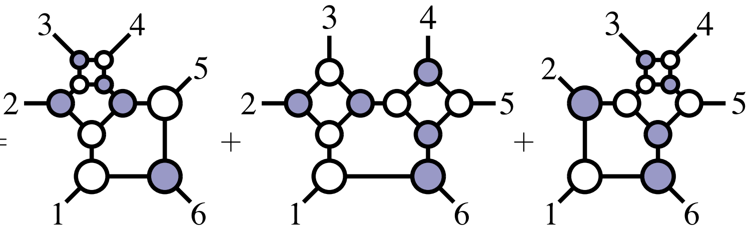
Observations regarding recursed representations of scattering amplitudes:

- varying recursion 'schema' can generate *many* 'BCFW formulae'
- on-shell diagrams can often be related in surprising ways

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$! And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

$\mathcal{A}_6^{(3)} =$

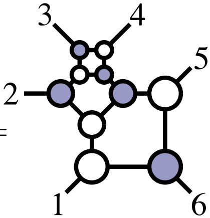
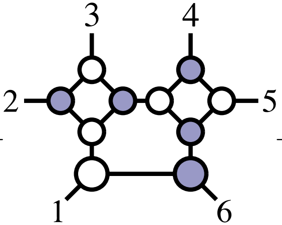
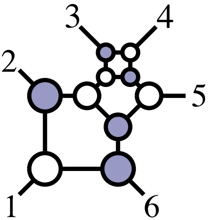


Observations regarding recursed representations of scattering amplitudes:

- varying recursion ‘schema’ can generate *many* ‘BCFW formulae’
- on-shell diagrams can often be related in surprising ways

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$! And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

$\mathcal{A}_6^{(3)} =$  $+$  $+$ 

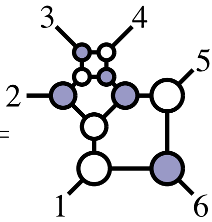
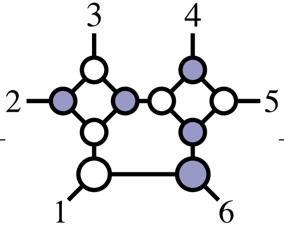
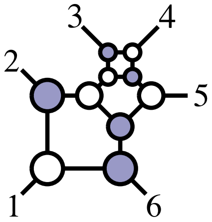
Observations regarding recursed representations of scattering amplitudes:

- varying recursion ‘schema’ can generate *many* ‘BCFW formulae’
- on-shell diagrams can often be related in surprising ways

Is there any way to invariantly characterize the on-shell functions associated with on-shell diagrams?

Exempli Gratia: On-Shell Representations of Amplitudes

The BCFW recursion relations realize an incredible fantasy: they **directly** produces the **Parke-Taylor** formula for all amplitudes with $k=2$, $\mathcal{A}_n^{(2)}$!
And it generates **very concise** formulae for all other amplitudes—*e.g.* $\mathcal{A}_6^{(3)}$:

$\mathcal{A}_6^{(3)} =$  $+$  $+$ 

Observations regarding recursed representations of scattering amplitudes:

- varying recursion ‘schema’ can generate *many* ‘BCFW formulae’
- on-shell diagrams can often be related in surprising ways

Is there any way to invariantly characterize the on-shell functions associated with on-shell diagrams?

Combinatorial Characterization of On-Shell Diagrams

On-shell diagrams can be altered without changing their associated functions

Combinatorial Characterization of On-Shell Diagrams

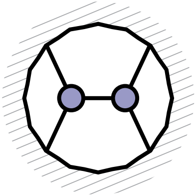
On-shell diagrams can be altered without changing their associated functions

- chains of equivalent three-particle vertices can be arbitrarily connected

Combinatorial Characterization of On-Shell Diagrams

On-shell diagrams can be altered without changing their associated functions

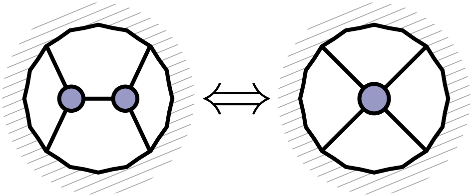
- chains of equivalent three-particle vertices can be arbitrarily connected



Combinatorial Characterization of On-Shell Diagrams

On-shell diagrams can be altered without changing their associated functions

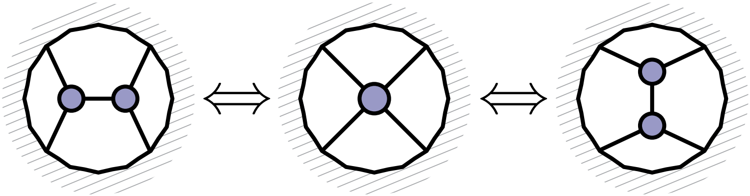
- chains of equivalent three-particle vertices can be arbitrarily connected



Combinatorial Characterization of On-Shell Diagrams

On-shell diagrams can be altered without changing their associated functions

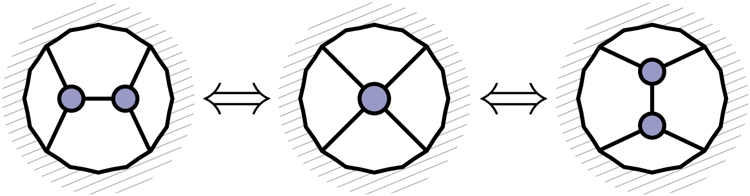
- chains of equivalent three-particle vertices can be arbitrarily connected



Combinatorial Characterization of On-Shell Diagrams

On-shell diagrams can be altered without changing their associated functions

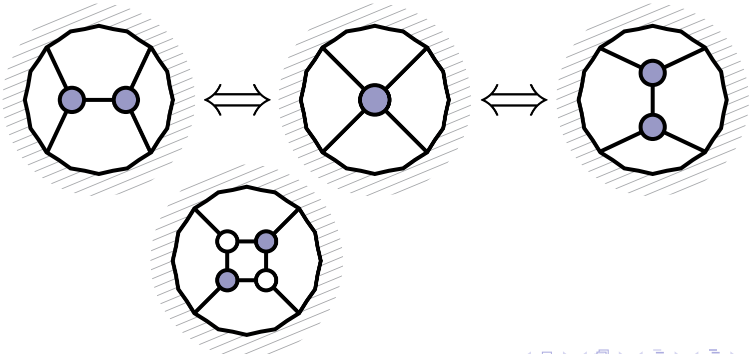
- chains of equivalent three-particle vertices can be arbitrarily connected
- any four-particle 'square' can be drawn in its two equivalent ways



Combinatorial Characterization of On-Shell Diagrams

On-shell diagrams can be altered without changing their associated functions

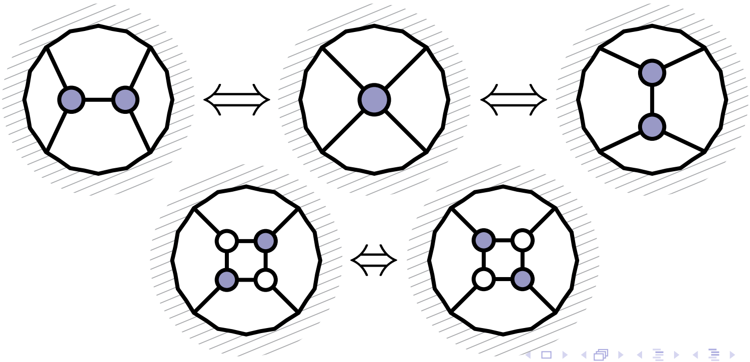
- chains of equivalent three-particle vertices can be arbitrarily connected
- any four-particle 'square' can be drawn in its two equivalent ways



Combinatorial Characterization of On-Shell Diagrams

On-shell diagrams can be altered without changing their associated functions

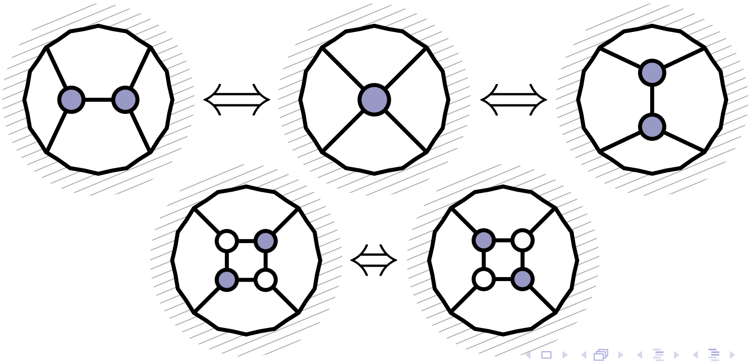
- chains of equivalent three-particle vertices can be arbitrarily connected
- any four-particle ‘square’ can be drawn in its two equivalent ways



Combinatorial Characterization of On-Shell Diagrams

On-shell diagrams can be altered without changing their associated functions

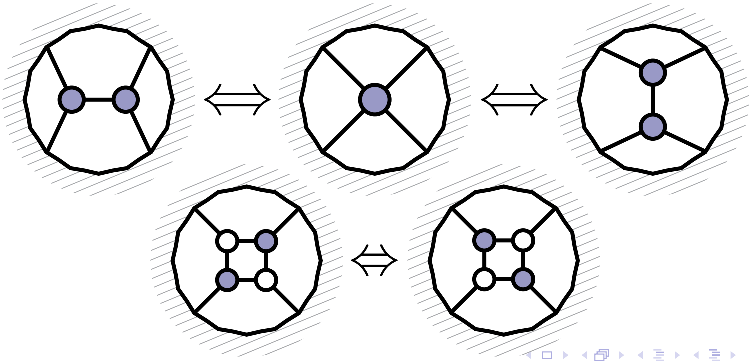
- chains of equivalent three-particle vertices can be arbitrarily connected
- any four-particle 'square' can be drawn in its two equivalent ways



Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths':

Starting from any leg a, turn:

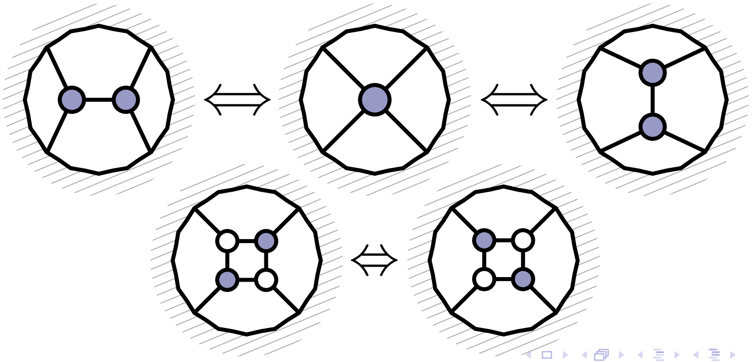


Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths':

Starting from any leg a, turn:

- *left* at each white vertex;

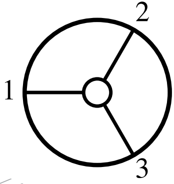


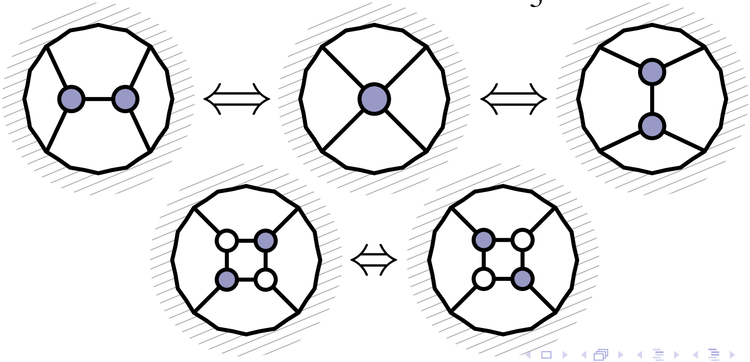
Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths':

Starting from any leg a, turn:

- *left* at each white vertex;



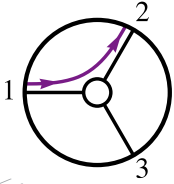


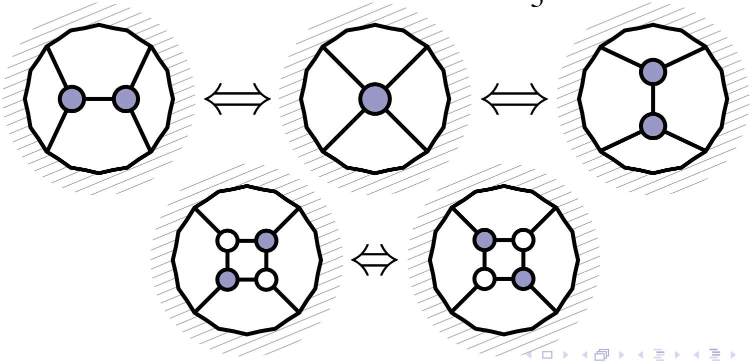
Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths':

Starting from any leg a, turn:

- *left* at each white vertex;



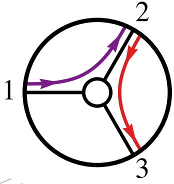


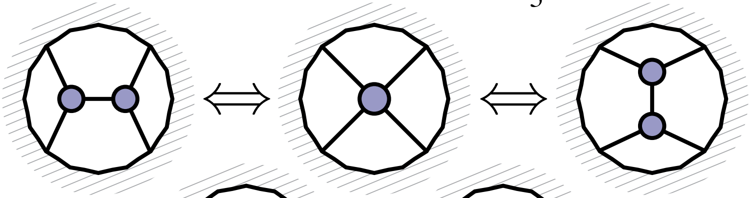
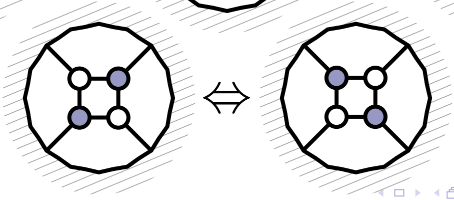
Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths':

Starting from any leg a, turn:

- *left* at each white vertex;



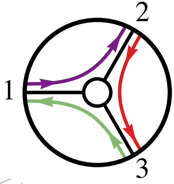



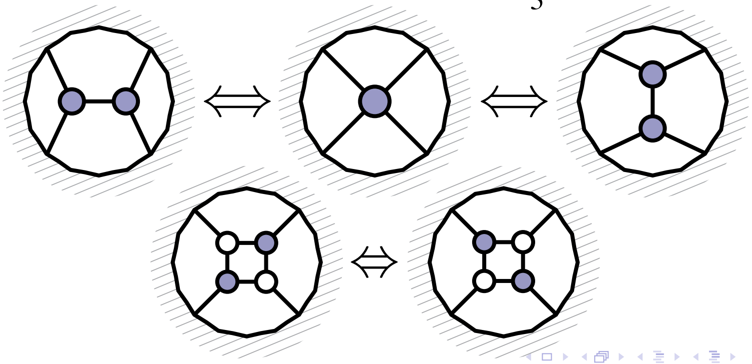
Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths':

Starting from any leg a, turn:

- *left* at each white vertex;



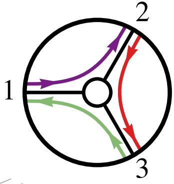


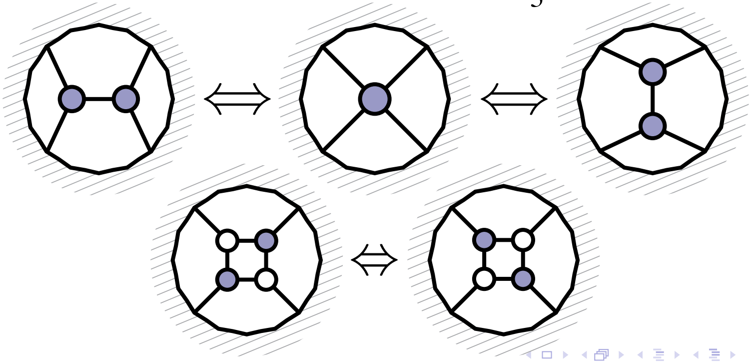
Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths':

Starting from any leg a, turn:

- *left* at each white vertex;
- *right* at each blue vertex.



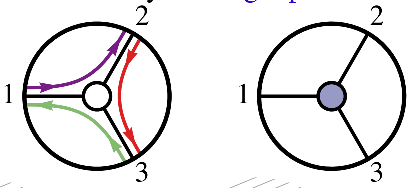


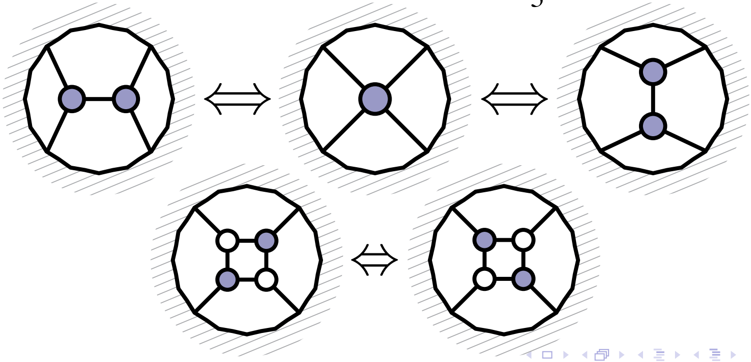
Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths':

Starting from any leg a, turn:

- *left* at each white vertex;
- *right* at each blue vertex.



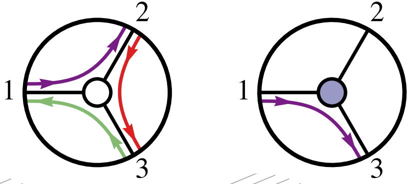


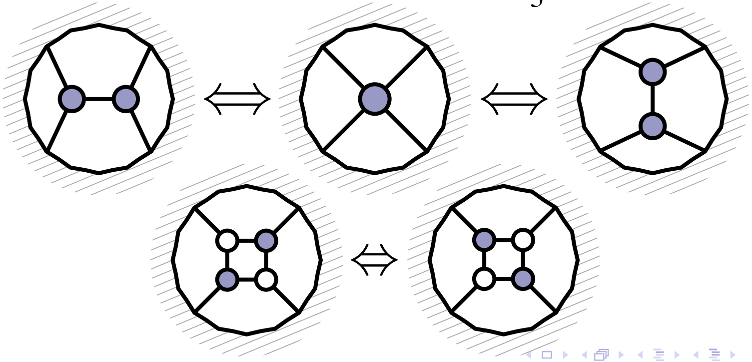
Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths':

Starting from any leg a, turn:

- *left* at each white vertex;
- *right* at each blue vertex.



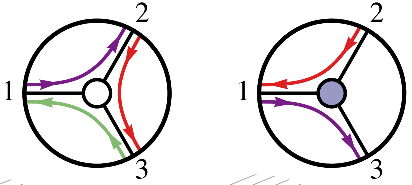


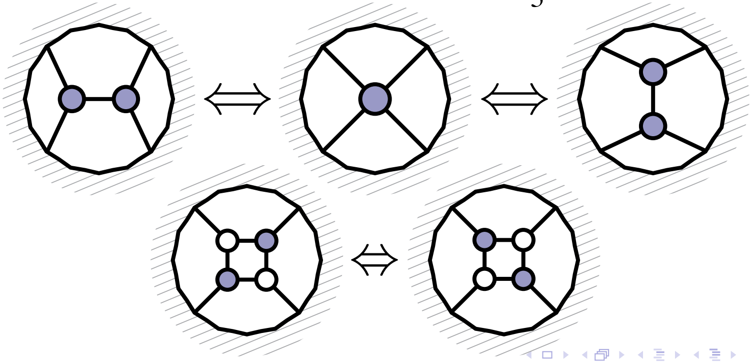
Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths':

Starting from any leg a, turn:

- *left* at each white vertex;
- *right* at each blue vertex.



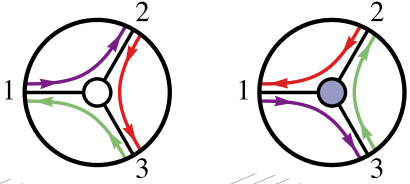


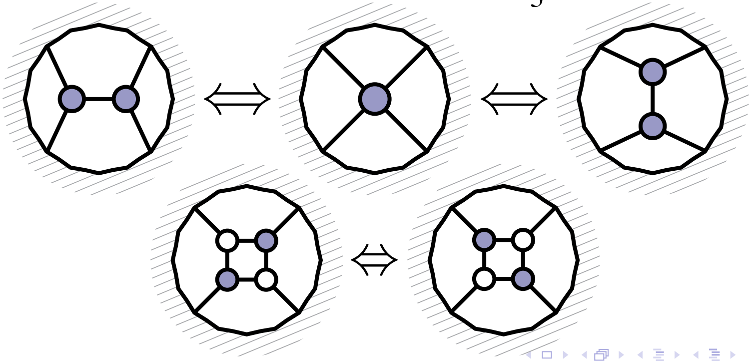
Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by ‘left-right paths’:

Starting from any leg a, turn:

- *left* at each white vertex;
- *right* at each blue vertex.





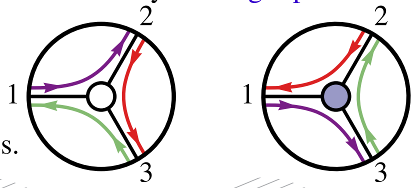
Combinatorial Characterization of On-Shell Diagrams

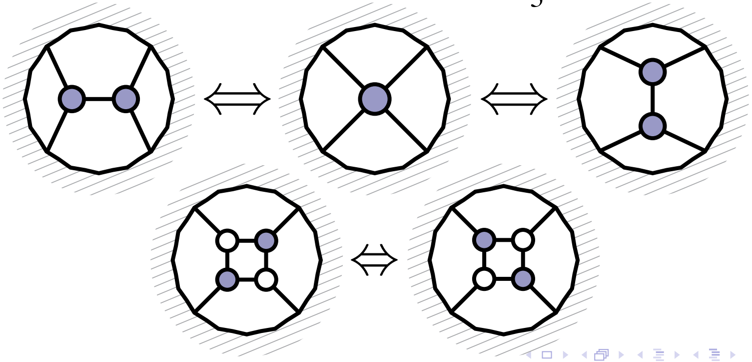
These moves leave invariant a permutation defined by 'left-right paths':

Starting from any leg a, turn:

- *left* at each white vertex;
- *right* at each blue vertex.

Let $\sigma(a)$ denote where path terminates.





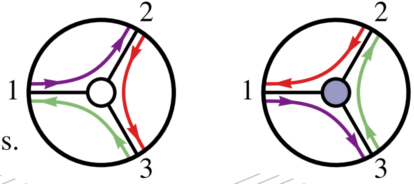
Combinatorial Characterization of On-Shell Diagrams

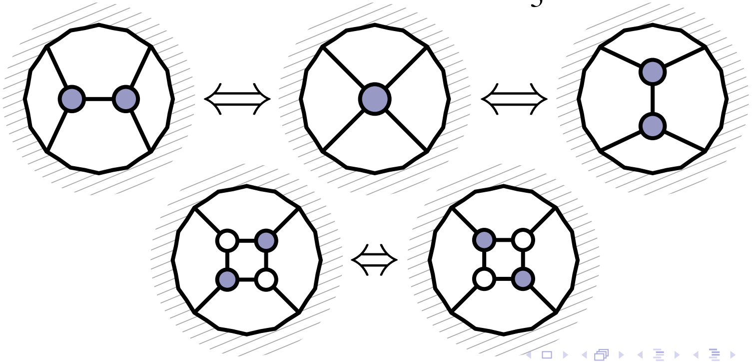
These moves leave invariant a permutation defined by 'left-right paths':

Starting from any leg a, turn:

- *left* at each white vertex;
- *right* at each blue vertex.

Let $\sigma(a)$ denote where path terminates.





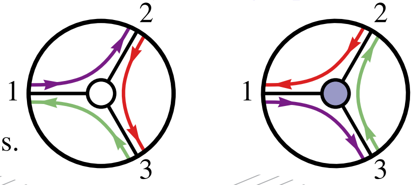
Combinatorial Characterization of On-Shell Diagrams

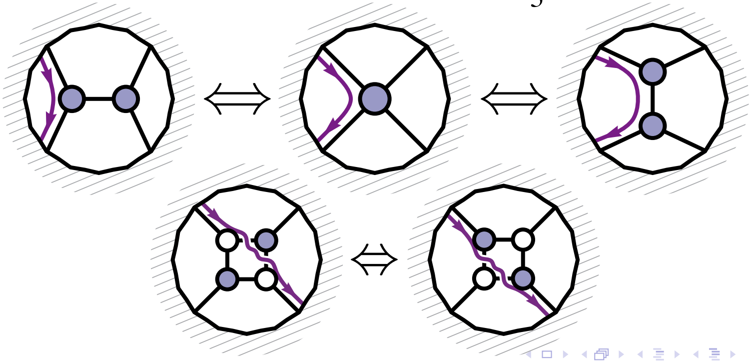
These moves leave invariant a permutation defined by 'left-right paths'

Starting from any leg a, turn:

- *left* at each white vertex;
- *right* at each blue vertex.

Let $\sigma(a)$ denote where path terminates.





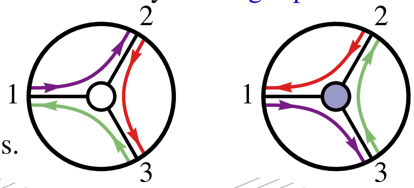
Combinatorial Characterization of On-Shell Diagrams

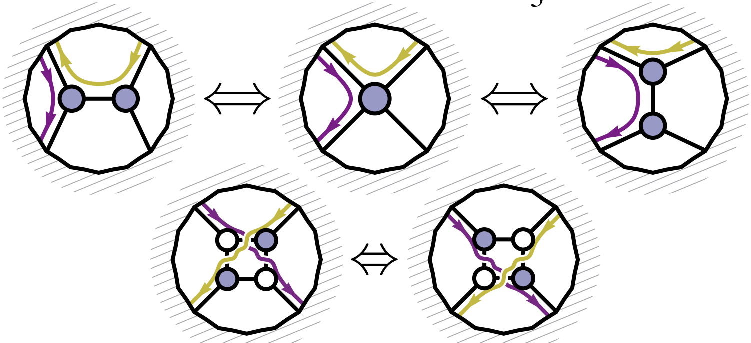
These moves leave invariant a permutation defined by 'left-right paths'

Starting from any leg a, turn:

- *left* at each white vertex;
- *right* at each blue vertex.

Let $\sigma(a)$ denote where path terminates.





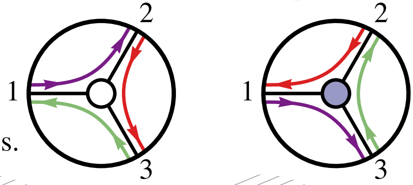
Combinatorial Characterization of On-Shell Diagrams

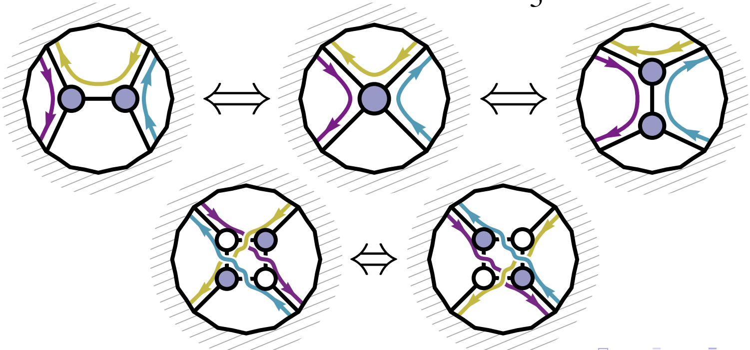
These moves leave invariant a permutation defined by 'left-right paths'

Starting from any leg a, turn:

- *left* at each white vertex;
- *right* at each blue vertex.

Let $\sigma(a)$ denote where path terminates.





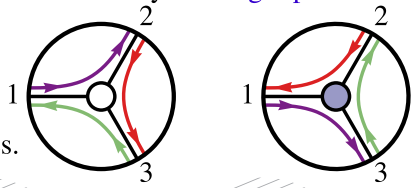
Combinatorial Characterization of On-Shell Diagrams

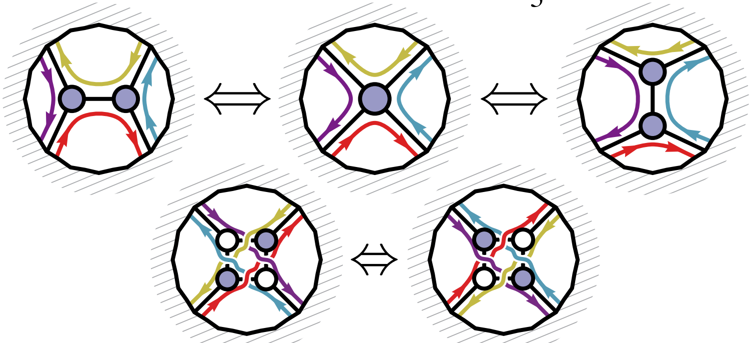
These moves leave invariant a permutation defined by 'left-right paths'

Starting from any leg a, turn:

- *left* at each white vertex;
- *right* at each blue vertex.

Let $\sigma(a)$ denote where path terminates.





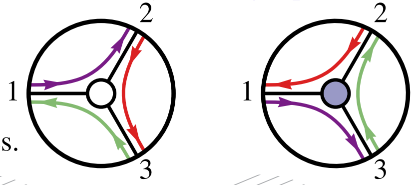
Combinatorial Characterization of On-Shell Diagrams

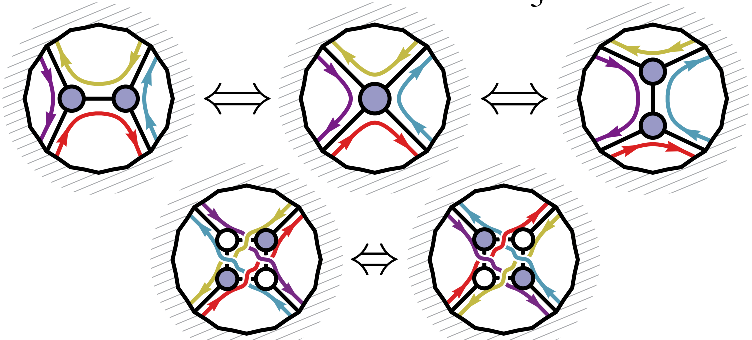
These moves leave invariant a permutation defined by 'left-right paths'

Starting from any leg a, turn:

- *left* at each white vertex;
- *right* at each blue vertex.

Let $\sigma(a)$ denote where path terminates.

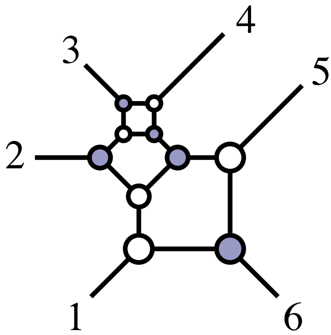


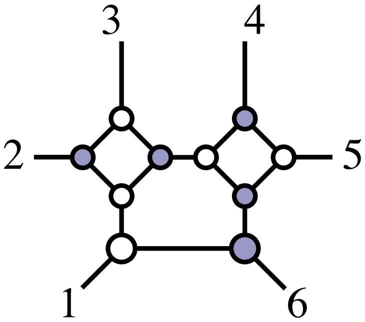


Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:

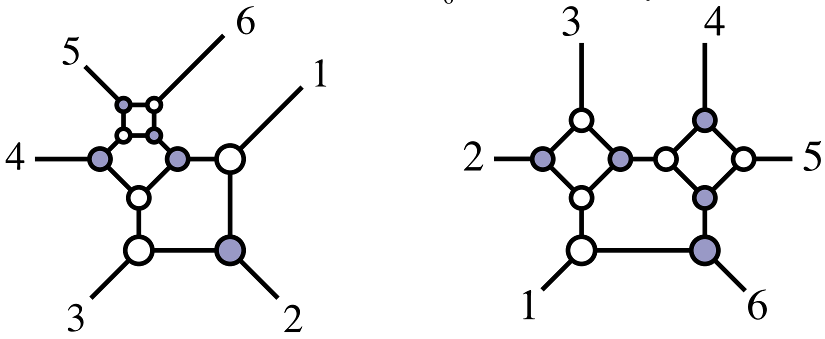




Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

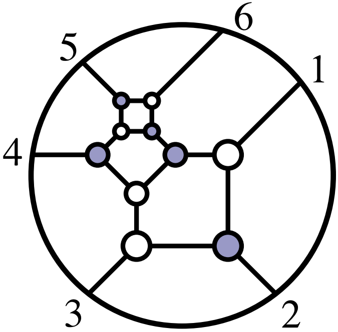
Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:

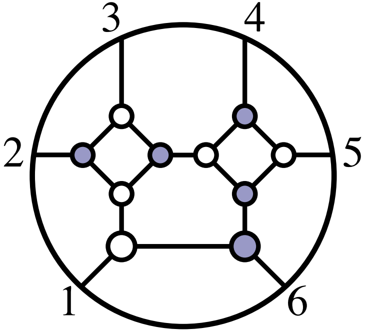


Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:

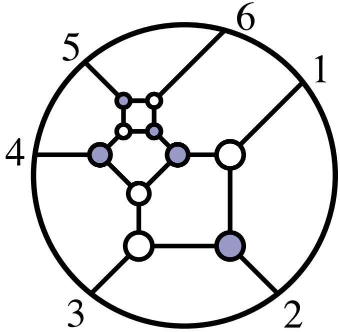


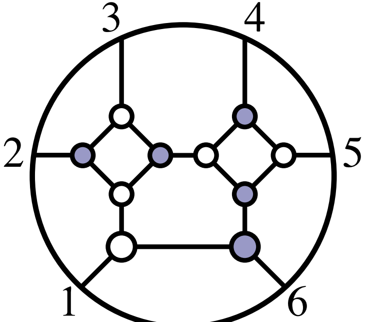


Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:





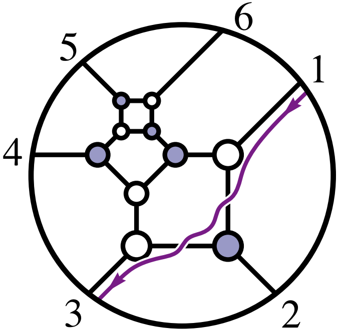
left-right permutation σ

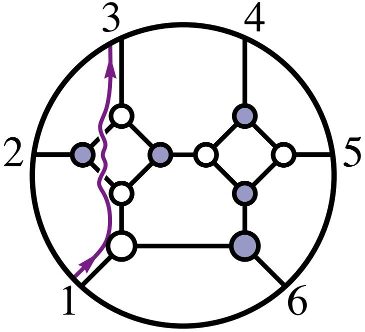
$$\sigma : \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ & & & & & \end{pmatrix}$$

Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:





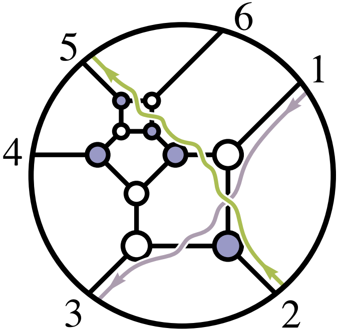
left-right permutation σ

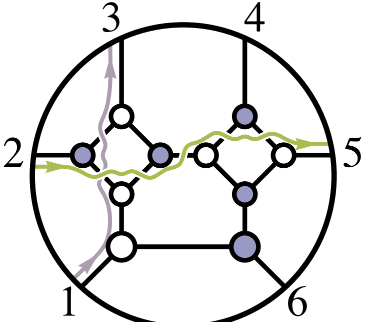
$$\sigma:\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6\\ \downarrow & & & & & \\ 3 & & & & & \end{pmatrix}$$

Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:





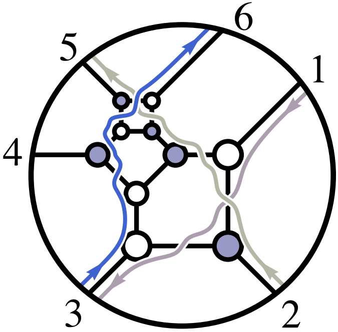
left-right permutation σ

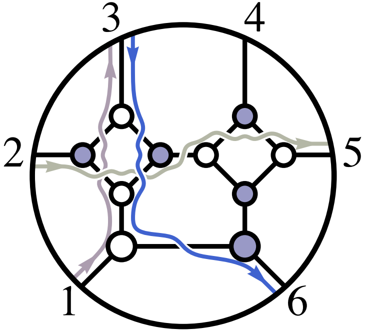
$$\sigma : \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ \downarrow & \downarrow & & & & \\ 3 & 5 & & & & \end{pmatrix}$$

Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:





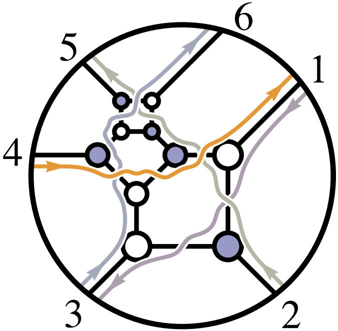
left-right permutation σ

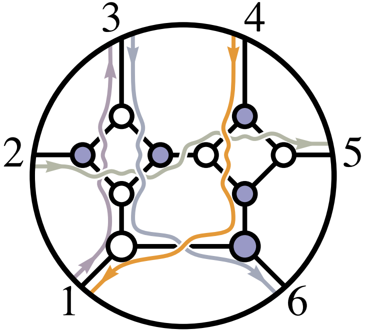
$$\sigma : \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ \downarrow & \downarrow & \downarrow & & & \\ 3 & 5 & 6 & & & \end{pmatrix}$$

Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:





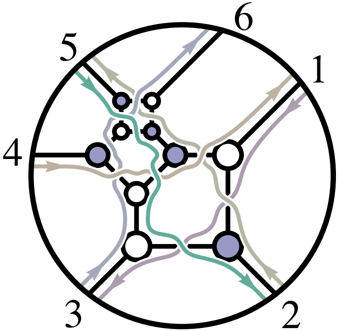
left-right permutation σ

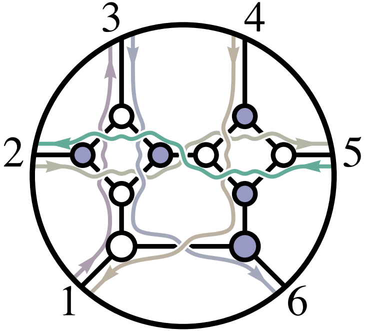
$$\sigma : \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ \downarrow & \downarrow & \downarrow & \downarrow & & \\ 3 & 5 & 6 & \mathbf{1} & & \end{pmatrix}$$

Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:





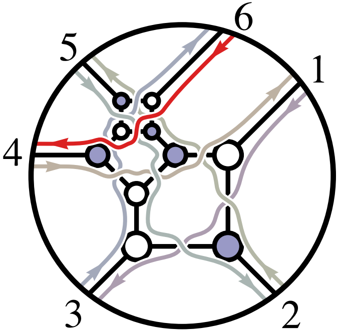
left-right permutation σ

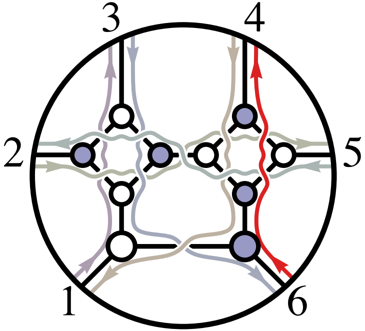
$$\sigma:\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \\ 3 & 5 & 6 & \mathbf{1} & \mathbf{2} & \end{pmatrix}$$

Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:





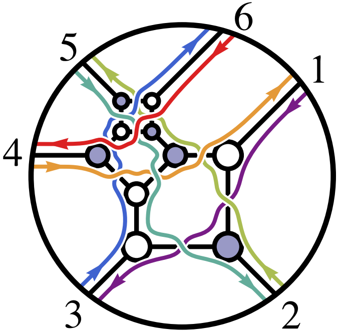
left-right permutation σ

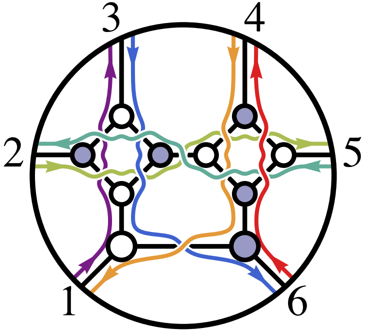
$$\sigma : \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 3 & 5 & 6 & \mathbf{1} & \mathbf{2} & \mathbf{4} \end{pmatrix}$$

Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:





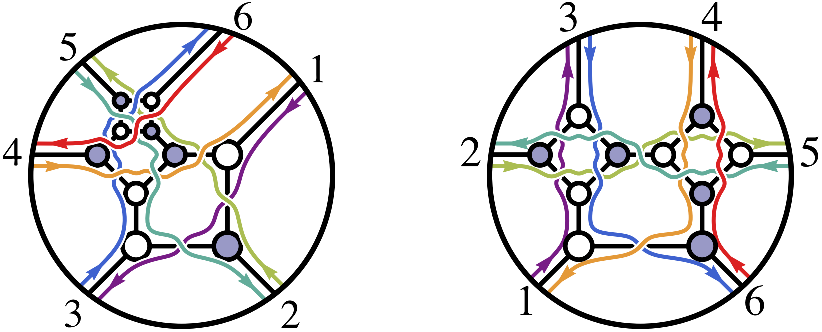
left-right permutation σ

$$\sigma : \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 3 & 5 & 6 & \mathbf{1} & \mathbf{2} & \mathbf{4} \end{pmatrix}$$

Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:



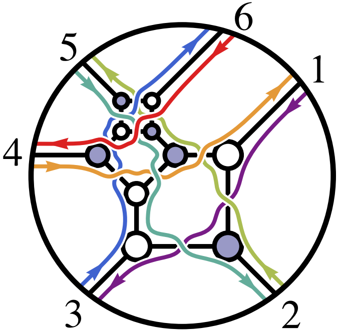
left-right permutation σ

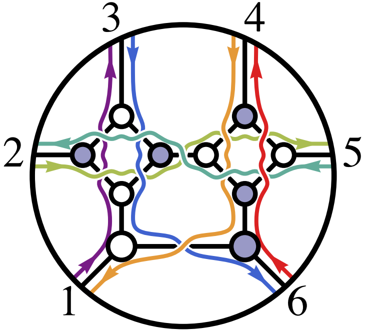
$$\sigma : \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 3 & 5 & 6 & \mathbf{1} & \mathbf{2} & \mathbf{4} \end{pmatrix}$$

Combinatorial Characterization of On-Shell Diagrams

These moves leave invariant a permutation defined by 'left-right paths'.

Recall that different contributions to $\mathcal{A}_6^{(3)}$ were related by rotation:





left-right permutation σ

$$\sigma : \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 3 & 5 & 6 & 7 & 8 & 10 \end{pmatrix}$$

Combinatorial Characterization of On-Shell Diagrams

Notice that the merge and square moves leave the number of 'faces' of an on-shell diagram invariant.

Combinatorial Characterization of On-Shell Diagrams

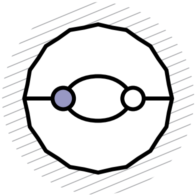
Notice that the merge and square moves leave the number of 'faces' of an on-shell diagram invariant. Diagrams with different numbers of faces can be related by 'reduction'

Combinatorial Characterization of On-Shell Diagrams

Notice that the merge and square moves leave the number of 'faces' of an on-shell diagram invariant. Diagrams with different numbers of faces can be related by 'reduction'—also known as 'bubble deletion':

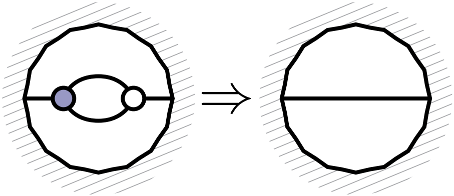
Combinatorial Characterization of On-Shell Diagrams

Notice that the merge and square moves leave the number of 'faces' of an on-shell diagram invariant. Diagrams with different numbers of faces can be related by 'reduction'—also known as 'bubble deletion':



Combinatorial Characterization of On-Shell Diagrams

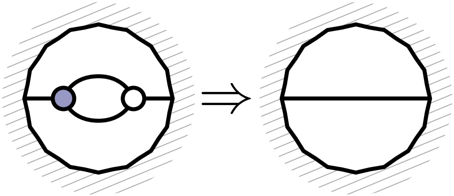
Notice that the merge and square moves leave the number of 'faces' of an on-shell diagram invariant. Diagrams with different numbers of faces can be related by 'reduction'—also known as 'bubble deletion':



Combinatorial Characterization of On-Shell Diagrams

Notice that the merge and square moves leave the number of 'faces' of an on-shell diagram invariant. Diagrams with different numbers of faces can be related by 'reduction'—also known as 'bubble deletion':

Bubble-deletion does not, however, relate 'identical' on-shell diagrams:

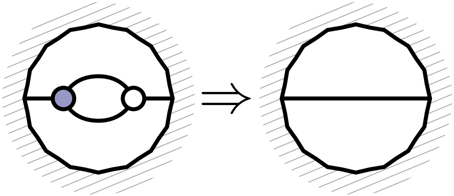


Combinatorial Characterization of On-Shell Diagrams

Notice that the merge and square moves leave the number of 'faces' of an on-shell diagram invariant. Diagrams with different numbers of faces can be related by 'reduction'—also known as 'bubble deletion':

Bubble-deletion does not, however, relate 'identical' on-shell diagrams:

- it leaves behind an overall factor of $d\alpha/\alpha$ in the on-shell function

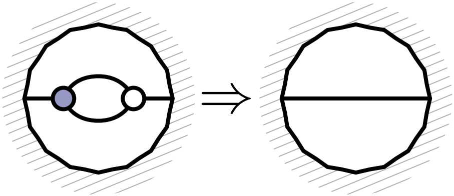


Combinatorial Characterization of On-Shell Diagrams

Notice that the merge and square moves leave the number of 'faces' of an on-shell diagram invariant. Diagrams with different numbers of faces can be related by 'reduction'—also known as 'bubble deletion':

Bubble-deletion does not, however, relate 'identical' on-shell diagrams:

- it leaves behind an overall factor of $d\alpha/\alpha$ in the on-shell function
- and it alters the corresponding left-right path permutation

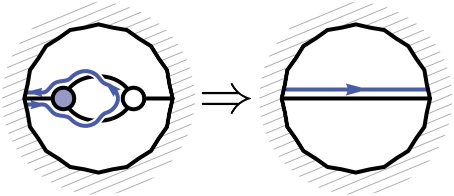


Combinatorial Characterization of On-Shell Diagrams

Notice that the merge and square moves leave the number of 'faces' of an on-shell diagram invariant. Diagrams with different numbers of faces can be related by 'reduction'—also known as 'bubble deletion':

Bubble-deletion does not, however, relate 'identical' on-shell diagrams:

- it leaves behind an overall factor of $d\alpha/\alpha$ in the on-shell function
- and it alters the corresponding left-right path permutation

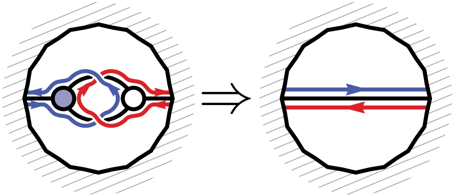


Combinatorial Characterization of On-Shell Diagrams

Notice that the merge and square moves leave the number of 'faces' of an on-shell diagram invariant. Diagrams with different numbers of faces can be related by 'reduction'—also known as 'bubble deletion':

Bubble-deletion does not, however, relate 'identical' on-shell diagrams:

- it leaves behind an overall factor of $d\alpha/\alpha$ in the on-shell function
- and it alters the corresponding left-right path permutation



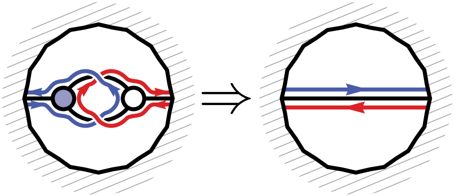
Combinatorial Characterization of On-Shell Diagrams

Notice that the merge and square moves leave the number of 'faces' of an on-shell diagram invariant. Diagrams with different numbers of faces can be related by 'reduction'—also known as 'bubble deletion':

Bubble-deletion does not, however, relate 'identical' on-shell diagrams:

- it leaves behind an overall factor of $d\alpha/\alpha$ in the on-shell function
- and it alters the corresponding left-right path permutation

Such factors of $d\alpha/\alpha$ arising from bubble deletion encode loop integrands!



Canonical Coordinates for Computing On-Shell Functions

Recall that attaching ‘BCFW bridges’ can lead to very rich on-shell diagrams.

Canonical Coordinates for Computing On-Shell Functions

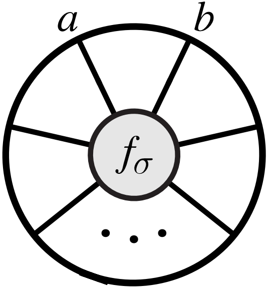
Recall that attaching 'BCFW bridges' can lead to very rich on-shell diagrams.

Conveniently, adding a BCFW bridge acts very nicely on permutations:

Canonical Coordinates for Computing On-Shell Functions

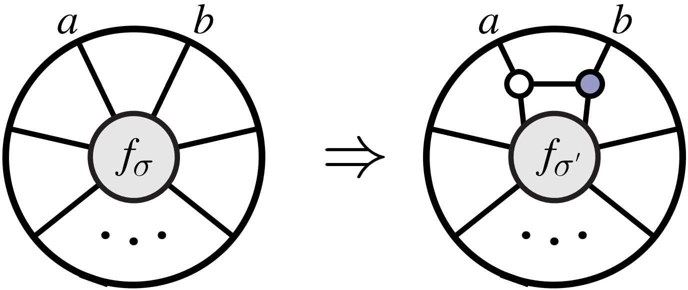
Recall that attaching 'BCFW bridges' can lead to very rich on-shell diagrams.

Conveniently, adding a BCFW bridge acts very nicely on permutations:



Canonical Coordinates for Computing On-Shell Functions

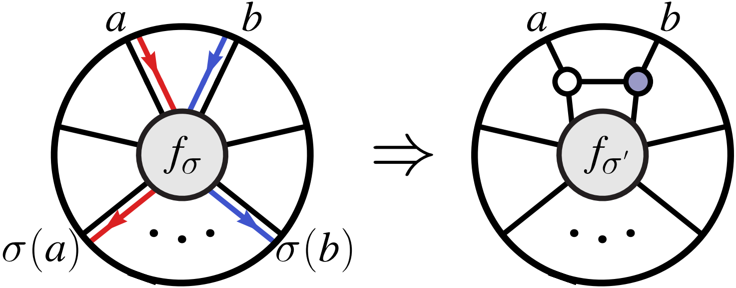
Recall that attaching 'BCFW bridges' can lead to very rich on-shell diagrams.
Conveniently, adding a BCFW bridge acts very nicely on permutations:



Canonical Coordinates for Computing On-Shell Functions

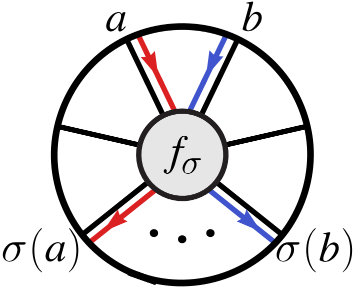
Recall that attaching 'BCFW bridges' can lead to very rich on-shell diagrams.

Conveniently, adding a BCFW bridge acts very nicely on permutations:



Canonical Coordinates for Computing On-Shell Functions

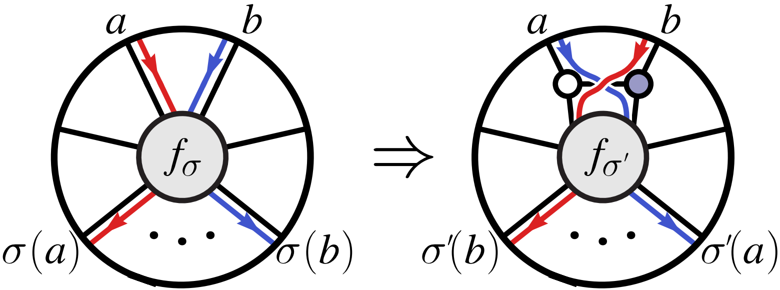
Recall that attaching ‘BCFW bridges’ can lead to very rich on-shell diagrams.
Conveniently, adding a BCFW bridge acts very nicely on permutations:



Canonical Coordinates for Computing On-Shell Functions

Recall that attaching 'BCFW bridges' can lead to very rich on-shell diagrams.

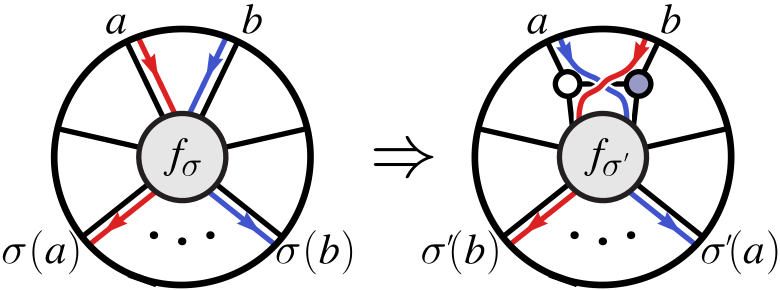
Conveniently, adding a BCFW bridge acts very nicely on permutations: it merely transposes the images of σ!



Canonical Coordinates for Computing On-Shell Functions

Recall that attaching 'BCFW bridges' can lead to very rich on-shell diagrams.

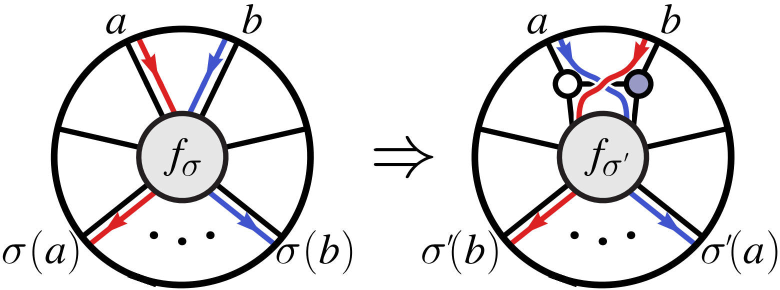
Conveniently, adding a BCFW bridge acts very nicely on permutations: it merely transposes the images of σ!



Canonical Coordinates for Computing On-Shell Functions

Recall that attaching 'BCFW bridges' can lead to very rich on-shell diagrams.

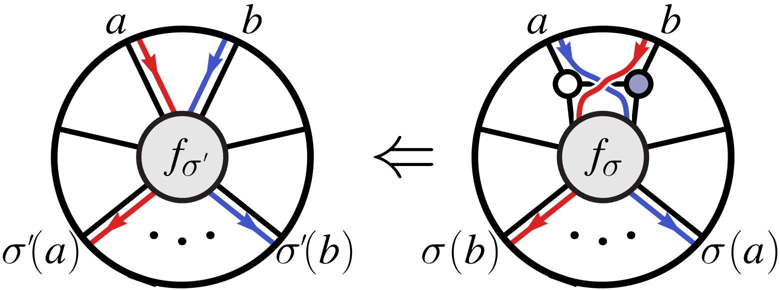
Read the other way,



Canonical Coordinates for Computing On-Shell Functions

Recall that attaching 'BCFW bridges' can lead to very rich on-shell diagrams.

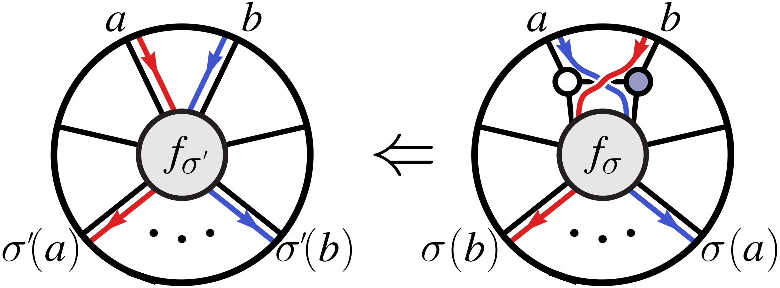
Read the other way,



Canonical Coordinates for Computing On-Shell Functions

Recall that attaching 'BCFW bridges' can lead to very rich on-shell diagrams.

Read the other way, we can 'peel-off' bridges and thereby decompose a permutation into transpositions according to $\sigma = (a\,b) \circ \sigma'$



Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions

'Bridge' Decomposition

$$\sigma : \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 3 & 5 & 6 & 7 & 8 & 10 \end{pmatrix}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions

'Bridge' Decomposition

$$\begin{array}{cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ f_\sigma & \{3 & 5 & 6 & 7 & 8 & 10\} \end{array}$$

Canonical Coordinates for Computing On-Shell Functions

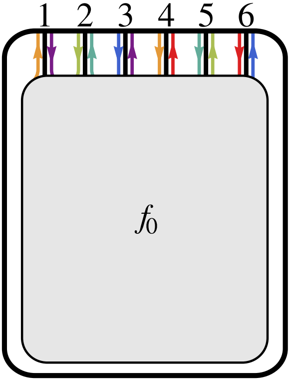
There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:

'Bridge' Decomposition

$$\begin{array}{cccccc c} & 1 & 2 & 3 & 4 & 5 & 6 \\ & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ f_\sigma & \{3 & 5 & 6 & 7 & 8 & 10\} \end{array}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

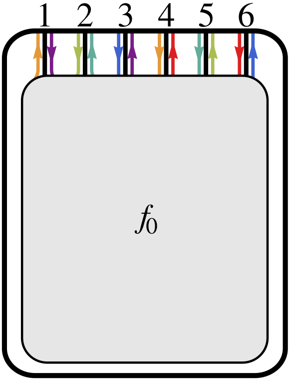


'Bridge' Decomposition

$$\begin{array}{cccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & \\ & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \tau \\ f_0 & \{3 & 5 & 6 & 7 & 8 & 10\} & \end{array}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



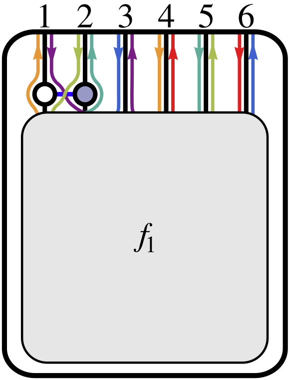
'Bridge' Decomposition

$$\begin{array}{c} \\ f_0 \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ \{3 & 5 & 6 & 7 & 8 & 10\} \end{array} \begin{array}{c} \tau \\ (1\,2) \end{array}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1} f_1$$



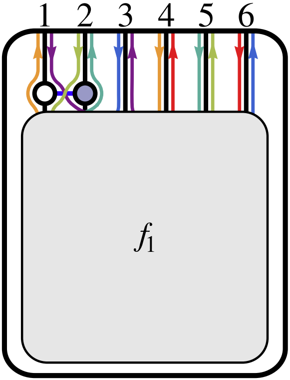
'Bridge' Decomposition

$$\begin{array}{lcccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & & \tau \\ & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & & \\ f_0 & \{3 & 5 & 6 & 7 & 8 & 10\} & & \\ & & & & & & & & (1\,2) \\ f_1 & \{5 & 3 & 6 & 7 & 8 & 10\} & & \end{array}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1} f_1$$



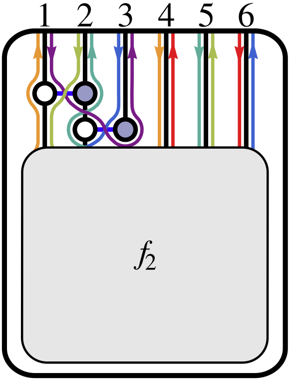
'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2} f_2$$



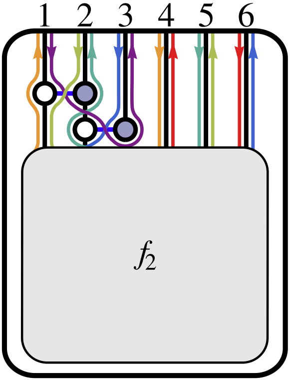
'Bridge' Decomposition

$$\begin{array}{cccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & \tau \\
 & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \\
f_0 & \{3 & 5 & 6 & 7 & 8 & 10\} & \\
 & & & & & & & (1\,2) \\
f_1 & \{5 & 3 & 6 & 7 & 8 & 10\} & \\
 & & & & & & & (2\,3) \\
f_2 & \{5 & 6 & 3 & 7 & 8 & 10\} &
\end{array}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2} f_2$$



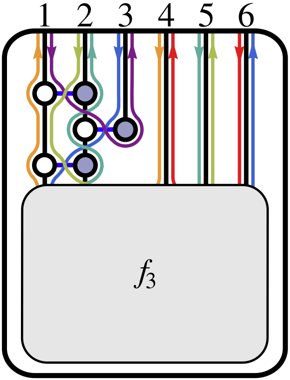
'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3} f_3$$



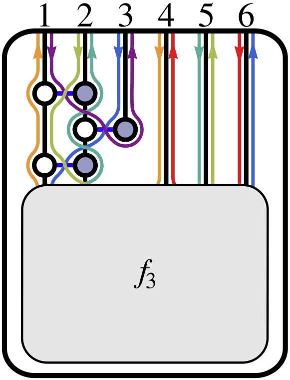
‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	$\{3$	5	6	7	8	$10\}$	
							$(1\,2)$
f_1	$\{5$	3	6	7	8	$10\}$	
							$(2\,3)$
f_2	$\{5$	6	3	7	8	$10\}$	
							$(1\,2)$
f_3	$\{6$	5	3	7	8	$10\}$	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau\equiv(a\,b)$ such that $\sigma(a)<\sigma(b)$:

$$f_0=\frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}f_3$$



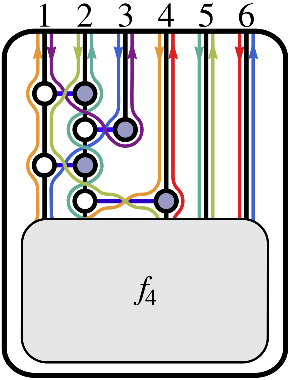
'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4} f_4$$



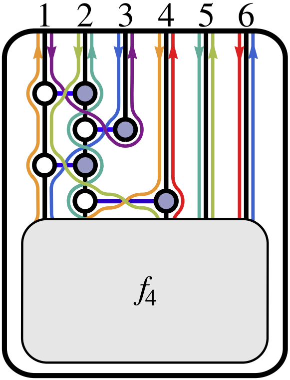
'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4} f_4$$



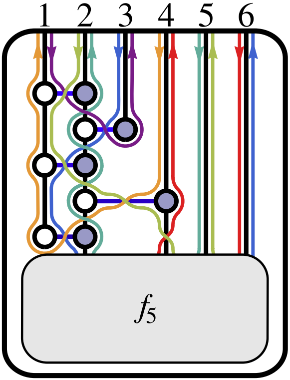
‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5} f_5$$



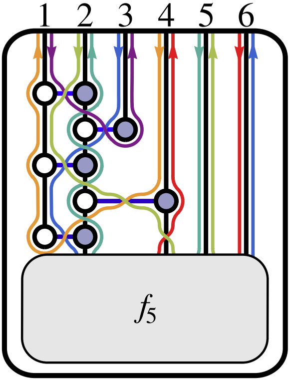
'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5} f_5$$



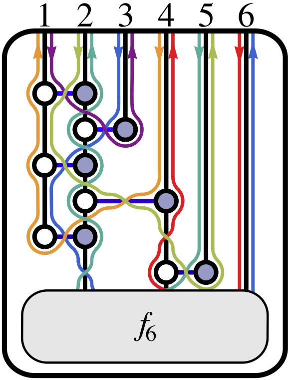
'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6} f_6$$



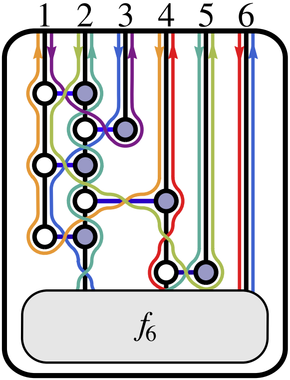
'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6} f_6$$



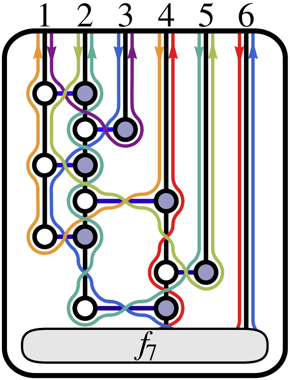
'Bridge' Decomposition

	$1 \downarrow$	$2 \downarrow$	$3 \downarrow$	$4 \downarrow$	$5 \downarrow$	$6 \downarrow$	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7} f_7$$



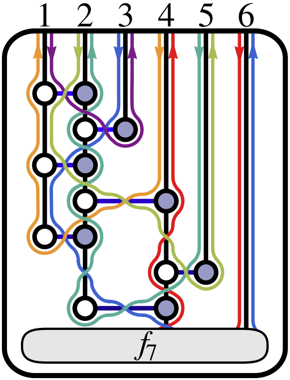
‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}f_7$$



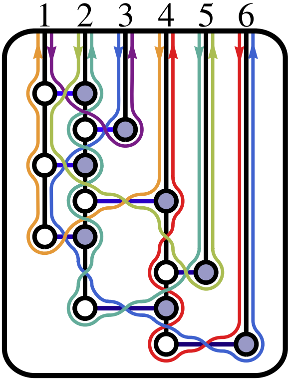
'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$



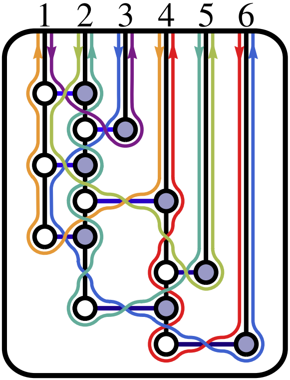
‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$



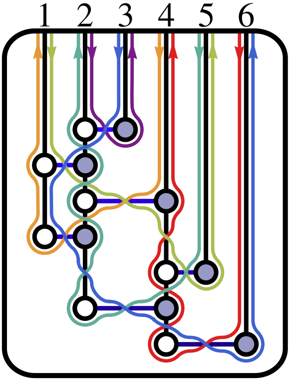
'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$



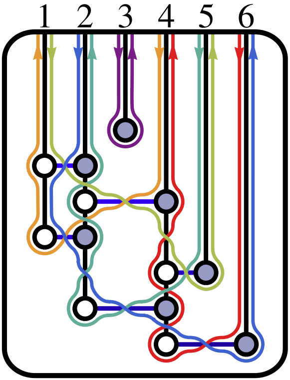
'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$



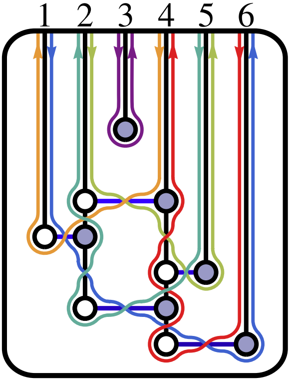
'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$



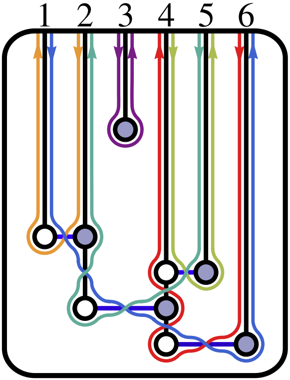
'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_3	{6	5	3	7	8	10}	
							(2 4)
f_4	{6	7	3	5	8	10}	
							(1 2)
f_5	{7	6	3	5	8	10}	
							(4 5)
f_6	{7	6	3	8	5	10}	
							(2 4)
f_7	{7	8	3	6	5	10}	
							(4 6)
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$



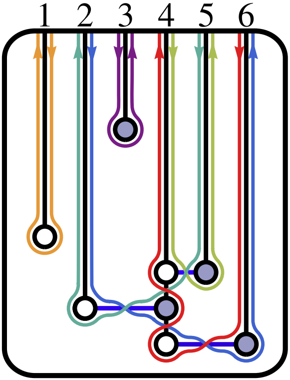
'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$



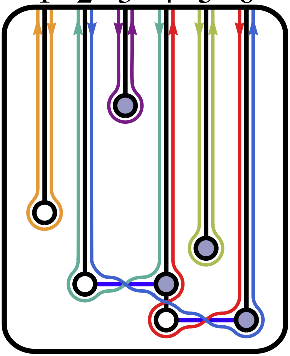
'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$



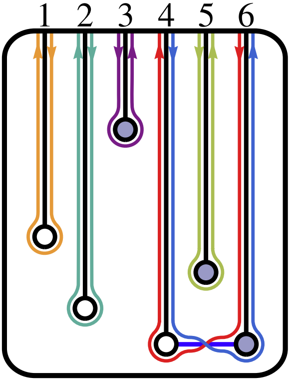
'Bridge' Decomposition

$$\begin{array}{lcccccccl}
 & 1 & 2 & 3 & 4 & 5 & 6 & & \tau \\
 & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & & \\
f_6 & \{7 & 6 & 3 & 8 & 5 & 10\} & & (2\,4) \\
f_7 & \{7 & 8 & 3 & 6 & 5 & 10\} & & (4\,6) \\
f_8 & \{7 & 8 & 3 & 10 & 5 & 6\} & &
\end{array}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8}f_8$$



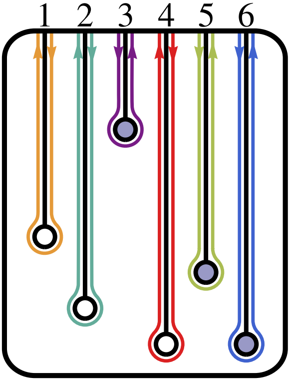
'Bridge' Decomposition

$$\begin{array}{ccccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & & \\
 & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & & \tau \\
f_7 & \{7 & 8 & 3 & 6 & 5 & 10\} & & \\
f_8 & \{7 & 8 & 3 & 10 & 5 & 6\} & & (4\,6)
\end{array}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8}f_8$$



'Bridge' Decomposition

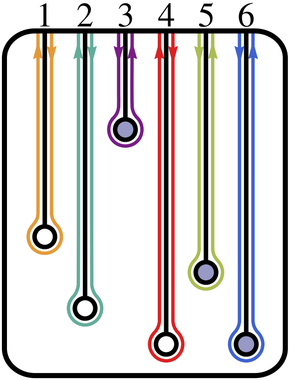
$$\begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \tau \end{array}$$

f_8 $\{7\;\;8\;\;3\;\;10\;\;5\;\;6\;\}$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$



'Bridge' Decomposition

1	2	3	4	5	6	
$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	τ

$f_8\ \{7\ \ 8\ \ 3\ \ 10\ \ 5\ \ 6\ \}$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$

$$f_8 = \prod_{a=\sigma(a)+n}\Big(\delta^4(\widetilde{\eta}_a)\,\delta^2(\widetilde{\lambda}_a)\Big)\prod_{b=\sigma(b)}\Big(\delta^2(\lambda_b)\Big)$$

'Bridge' Decomposition

$$\begin{array}{cccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & \\ & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \tau \\ f_8 & \{7 & 8 & 3 & 10 & 5 & 6\ \} & \end{array}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau\equiv(a\,b)$ such that $\sigma(a)<\sigma(b)$:

$$f_0=\frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8}f_8$$

$$f_8=\prod_{a=\sigma(a)+n}\Big(\delta^4\big(\widetilde{\eta}_a\big)\delta^2\big(\widetilde{\lambda}_a\big)\Big)\prod_{b=\sigma(b)}\Big(\delta^2\big(\lambda_b\big)\Big)$$

$$C\equiv\begin{array}{c}\begin{array}{cccccc}1&2&3&4&5&6\end{array}\\ \left(\begin{array}{cccccc}1&0&0&0&0&0\\0&1&0&0&0&0\\0&0&0&1&0&0\end{array}\right)\end{array}$$

'Bridge' Decomposition

$$\begin{array}{cccccccc}1&2&3&4&5&6&&\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow&\downarrow&&\tau\end{array}$$

$$f_8\;\{7\;\;8\;\;3\;\;10\;\;5\;\;6\;\}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$

$$f_8 = \delta^{3\times4}(C\cdot\widetilde{\eta})\,\delta^{3\times2}(C\cdot\widetilde{\lambda})\,\delta^{2\times3}(\lambda\cdot C^{\perp})$$

$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

'Bridge' Decomposition

$$\begin{array}{cccccc c} 1 & 2 & 3 & 4 & 5 & 6 & \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \tau \end{array}$$

$$f_8\;\{7\;\;8\;\;3\;\;10\;\;5\;\;6\}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8}f_8$$

$$f_8 = \delta^{3\times4}(C\cdot\widetilde{\eta})\,\delta^{3\times2}(C\cdot\widetilde{\lambda})\,\delta^{2\times3}(\lambda\cdot C^{\perp})$$

$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right) \end{array}$$

'Bridge' Decomposition

$$\begin{array}{cccccc c} 1 & 2 & 3 & 4 & 5 & 6 & \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \tau \end{array}$$

$$f_8\ \{7\ \ 8\ \ 3\ \ 10\ \ 5\ \ 6\ \}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$

$$f_7 = \frac{d\alpha_8}{\alpha_8}\delta^{3\times4}(C\cdot\widetilde{\eta})\,\delta^{3\times2}(C\cdot\widetilde{\lambda})\,\delta^{2\times3}(\lambda\cdot C^{\perp})$$

$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

$(4\,6)$: $\quad c_6 \mapsto c_6 + \alpha_8\, c_4$

'Bridge' Decomposition

$$\begin{array}{lcccccccl} & 1 & 2 & 3 & 4 & 5 & 6 & & \\ & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & & \tau \\ f_7 & \{7 & 8 & 3 & 6 & 5 & 10\} & & \\ f_8 & \{7 & 8 & 3 & 10 & 5 & 6\,\} & & (4\,6) \end{array}$$

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$

$$f_6 = \frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8}\delta^{3\times4}(C\cdot\widetilde{\eta})\,\delta^{3\times2}(C\cdot\widetilde{\lambda})\,\delta^{2\times3}(\lambda\cdot C^{\perp})$$

$$C \equiv \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}\\ C \equiv \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{pmatrix}$$

$(2\,4)$: $c_4 \mapsto c_4 + \alpha_7\, c_2$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$

$$f_5 = \frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8}\delta^{3\times4}(C\cdot\widetilde{\eta})\,\delta^{3\times2}(C\cdot\widetilde{\lambda})\,\delta^{2\times3}(\lambda\cdot C^{\perp})$$

$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

$$(4\,5):\quad c_5 \mapsto c_5 + \alpha_6\, c_4$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$

$$f_4 = \frac{d\alpha_5}{\alpha_5}\cdots\frac{d\alpha_8}{\alpha_8}\delta^{3\times4}(C\cdot\widetilde{\eta})\,\delta^{3\times2}(C\cdot\widetilde{\lambda})\,\delta^{2\times3}(\lambda\cdot C^{\perp})$$

$$C \equiv \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{pmatrix} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{pmatrix} \end{array}$$

$$(1\,2):\quad c_2 \mapsto c_2 + \alpha_5\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$

$$f_3 = \frac{d\alpha_4}{\alpha_4}\cdots\frac{d\alpha_8}{\alpha_8}\delta^{3\times4}(C\cdot\widetilde{\eta})\,\delta^{3\times2}(C\cdot\widetilde{\lambda})\,\delta^{2\times3}(\lambda\cdot C^{\perp})$$

$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

$$(2\,4):\quad c_4 \mapsto c_4 + \alpha_4\, c_2$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$

$$f_2 = \frac{d\alpha_3}{\alpha_3} \cdots \frac{d\alpha_8}{\alpha_8}\,\delta^{3\times4}(C\cdot\widetilde{\eta})\,\delta^{3\times2}(C\cdot\widetilde{\lambda})\,\delta^{2\times3}(\lambda\cdot C^{\perp})$$

$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

$$(1\,2):\quad c_2 \mapsto c_2 + \alpha_3\, c_1$$

‘Bridge’ Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$

$$f_1 = \frac{d\alpha_2}{\alpha_2}\cdots\frac{d\alpha_8}{\alpha_8}\,\delta^{3\times 4}(C\cdot\widetilde{\eta})\,\delta^{3\times 2}(C\cdot\widetilde{\lambda})\,\delta^{2\times 3}(\lambda\cdot C^{\perp})$$

$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

$$(2\,3):\quad c_3 \mapsto c_3 + \alpha_2\, c_2$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8} f_8$$

$$f_0 = \frac{d\alpha_1}{\alpha_1}\cdots\frac{d\alpha_8}{\alpha_8}\delta^{3\times4}(C\cdot\widetilde{\eta})\,\delta^{3\times2}(C\cdot\widetilde{\lambda})\,\delta^{2\times3}(\lambda\cdot C^{\perp})$$

$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{pmatrix} \end{array}$$

$$(1\,2):\quad c_2 \mapsto c_2 + \alpha_1\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	
							(1 2)
f_1	{5	3	6	7	8	10}	
							(2 3)
f_2	{5	6	3	7	8	10}	
							(1 2)
f_3	{6	5	3	7	8	10}	
							(2 4)
f_4	{6	7	3	5	8	10}	
							(1 2)
f_5	{7	6	3	5	8	10}	
							(4 5)
f_6	{7	6	3	8	5	10}	
							(2 4)
f_7	{7	8	3	6	5	10}	
							(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8}f_8$$

$$f_0 = \frac{d\alpha_1}{\alpha_1}\cdots\frac{d\alpha_8}{\alpha_8}\delta^{3\times4}\big(C\cdot\widetilde{\eta}\big)\delta^{3\times2}\big(C\cdot\widetilde{\lambda}\big)\delta^{2\times3}\big(\lambda\cdot C^{\perp}\big)$$

$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{pmatrix} \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$f_0 = \frac{d\alpha_1}{\alpha_1}\frac{d\alpha_2}{\alpha_2}\frac{d\alpha_3}{\alpha_3}\frac{d\alpha_4}{\alpha_4}\frac{d\alpha_5}{\alpha_5}\frac{d\alpha_6}{\alpha_6}\frac{d\alpha_7}{\alpha_7}\frac{d\alpha_8}{\alpha_8}f_8$$

$$f_0 = \frac{d\alpha_1}{\alpha_1}\cdots\frac{d\alpha_8}{\alpha_8}\delta^{3\times4}(C\cdot\widetilde{\eta})\delta^{3\times2}(C\cdot\widetilde{\lambda})\delta^{2\times3}(\lambda\cdot C^{\perp})$$

$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:

$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \hline \end{array}$$
$$\left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:

$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

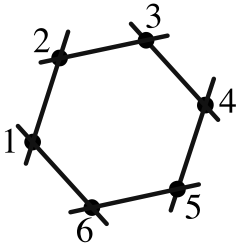
$$(6\,1):\quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

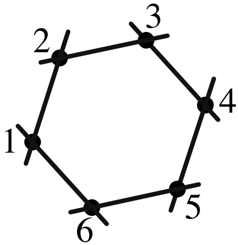
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

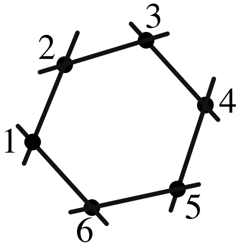
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau\equiv(a\,b)$ such that $\sigma(a)<\sigma(b)$:



$$C\equiv\begin{array}{c}\begin{array}{cccccc}1&2&3&4&5&6\end{array}\\ \left(\begin{array}{cccccc}1&(\alpha_1+\alpha_3+\alpha_5)&\alpha_2(\alpha_3+\alpha_5)&\alpha_4\alpha_5&0&0\\0&1&\alpha_2&(\alpha_4+\alpha_7)&\alpha_6\alpha_7&0\\\alpha_0\alpha_8&0&0&1&\alpha_6&\alpha_8\end{array}\right)\end{array}$$

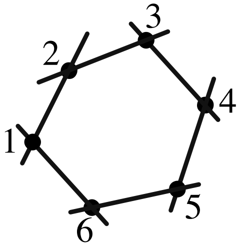
$$(6\,1):\quad c_1\mapsto c_1+\alpha_0\,c_6$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

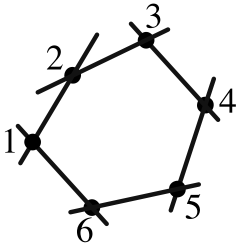
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau\equiv(a\,b)$ such that $\sigma(a)<\sigma(b)$:



$$C\equiv\begin{array}{c}\begin{array}{cccccc}1 & 2 & 3 & 4 & 5 & 6\end{array}\\ \left(\begin{array}{cccccc}1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0\\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0\\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8\end{array}\right)\end{array}$$

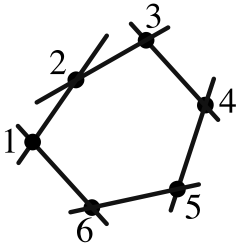
$$(6\,1):\quad c_1\mapsto c_1+\alpha_0\,c_6$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

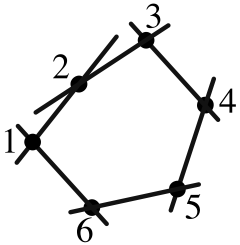
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

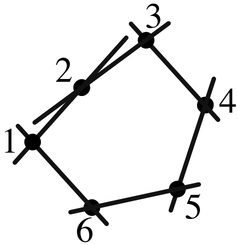
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

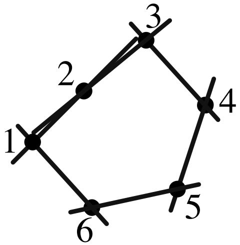
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

‘Bridge’ Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

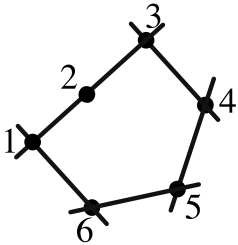
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

‘Bridge’ Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



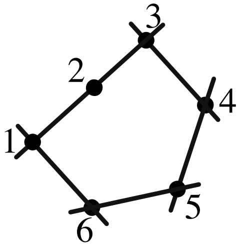
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

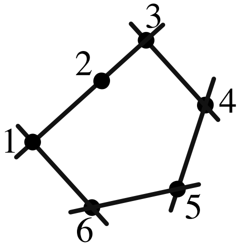
$$C \equiv \begin{pmatrix} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{pmatrix}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



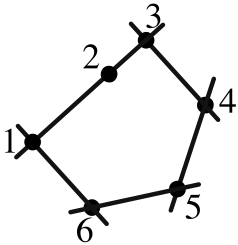
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau\equiv(a\,b)$ such that $\sigma(a)<\sigma(b)$:



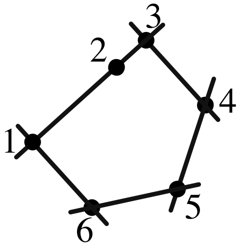
$$C\equiv\begin{array}{c}\begin{array}{cccccc}1&2&3&4&5&6\end{array}\\ \left(\begin{array}{cccccc}1&(\alpha_1+\alpha_3+\alpha_5)&\alpha_2(\alpha_3+\alpha_5)&\alpha_4\alpha_5&0&0\\0&1&\alpha_2&(\alpha_4+\alpha_7)&\alpha_6\alpha_7&0\\0&0&0&1&\alpha_6&\alpha_8\end{array}\right)\end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau\equiv(a\,b)$ such that $\sigma(a)<\sigma(b)$:



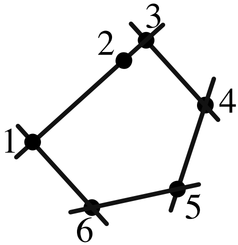
$$C\equiv\begin{array}{c}\begin{array}{cccccc}1 & 2 & 3 & 4 & 5 & 6\end{array}\\ \left(\begin{array}{cccccc}1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0\\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0\\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8\end{array}\right)\end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



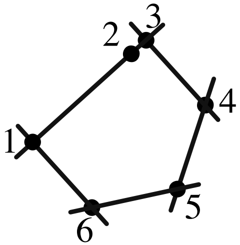
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



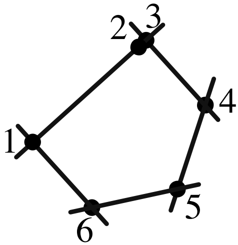
$$C \equiv \begin{array}{c}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}\\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)\end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



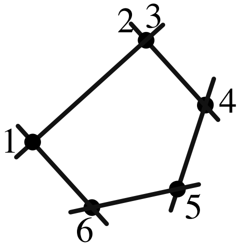
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



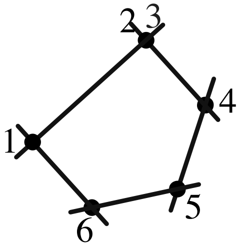
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

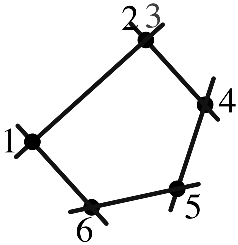
$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

‘Bridge’ Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau\equiv(a\,b)$ such that $\sigma(a)<\sigma(b)$:



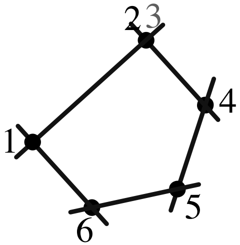
$$C\equiv\begin{array}{c}\begin{array}{cccccc}1&2&3&4&5&6\end{array}\\\left(\begin{array}{cccccc}1&(\alpha_3+\alpha_5)&\alpha_2(\alpha_3+\alpha_5)&\alpha_4\alpha_5&0&0\\0&1&\alpha_2&(\alpha_4+\alpha_7)&\alpha_6\alpha_7&0\\0&0&0&1&\alpha_6&\alpha_8\end{array}\right)\end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



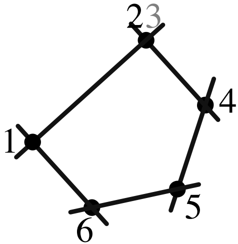
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



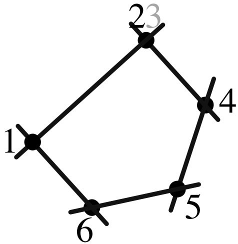
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

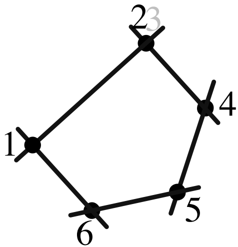
$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

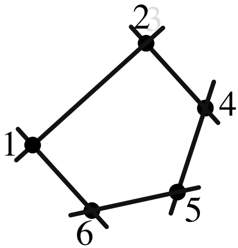
$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

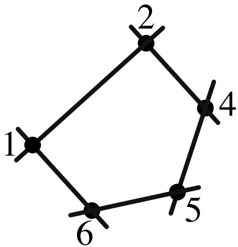
$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau\equiv(a\,b)$ such that $\sigma(a)<\sigma(b)$:



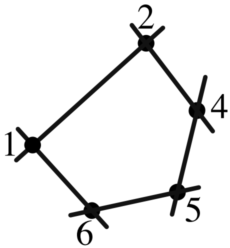
$$C\equiv\begin{array}{c}\begin{array}{cccccc}1&2&3&4&5&6\end{array}\\ \left(\begin{array}{cccccc}1&(\alpha_3+\alpha_5)&0&\alpha_4\alpha_5&0&0\\0&1&0&(\alpha_4+\alpha_7)&\alpha_6\alpha_7&0\\0&0&0&1&\alpha_6&\alpha_8\end{array}\right)\end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	
							(1 2)
f_1	{5	3	6	7	8	10}	
							(2 3)
f_2	{5	6	3	7	8	10}	
							(1 2)
f_3	{6	5	3	7	8	10}	
							(2 4)
f_4	{6	7	3	5	8	10}	
							(1 2)
f_5	{7	6	3	5	8	10}	
							(4 5)
f_6	{7	6	3	8	5	10}	
							(2 4)
f_7	{7	8	3	6	5	10}	
							(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



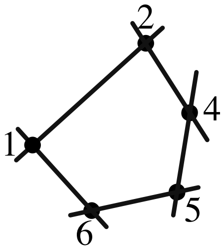
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



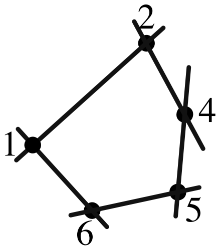
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



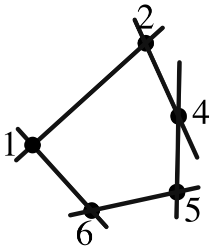
$$\begin{array}{cc} & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ C \equiv & \begin{pmatrix} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{pmatrix} \end{array}$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



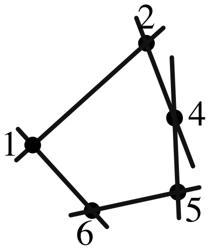
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



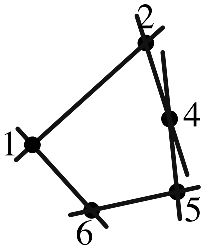
$$C \equiv \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \left(\begin{matrix} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{matrix}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau\equiv(a\,b)$ such that $\sigma(a)<\sigma(b)$:



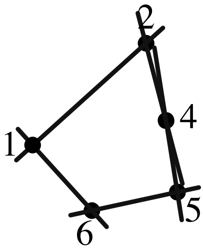
$$C\equiv\begin{array}{c}\begin{array}{cccccc}1&2&3&4&5&6\end{array}\\\left(\begin{array}{cccccc}1&(\alpha_3+\alpha_5)&0&\alpha_4\alpha_5&0&0\\0&1&0&(\alpha_4+\alpha_7)&\alpha_6\alpha_7&0\\0&0&0&1&\alpha_6&\alpha_8\end{array}\right)\end{array}$$

‘Bridge’ Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \hline \end{array}$$

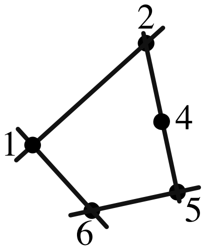
$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$
with columns labelled $1, 2, 3, 4, 5, 6$.

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



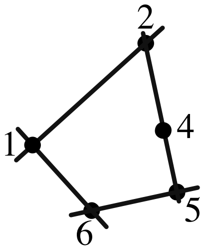
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



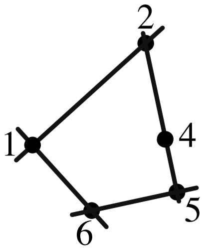
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

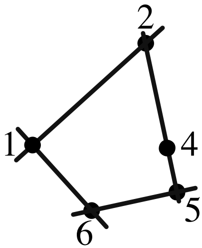
$$C \equiv \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



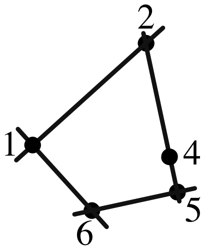
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



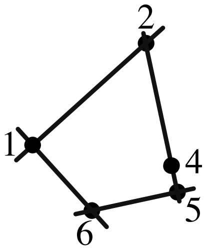
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

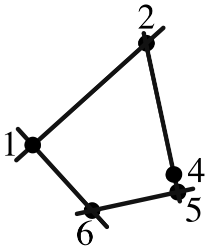
$$C \equiv \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

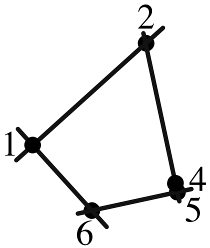
$$C \equiv \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



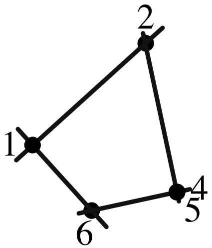
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \begin{pmatrix} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{pmatrix}$$

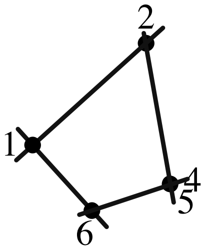
with columns labelled $1, 2, 3, 4, 5, 6$.

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



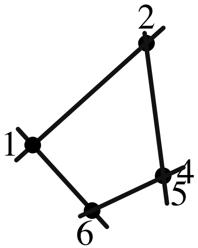
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



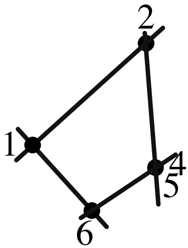
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



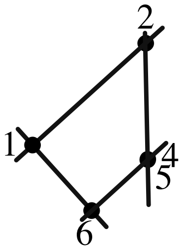
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



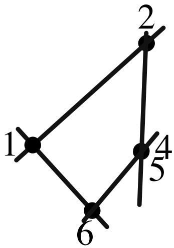
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



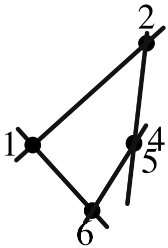
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



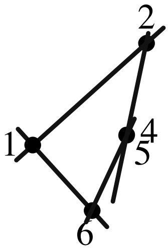
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



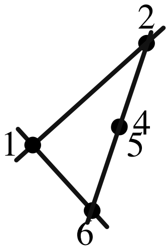
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



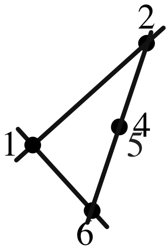
$$C \equiv \begin{array}{c} \\ \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	3	5	6	7	8	10	(1 2)
f_1	5	3	6	7	8	10	(2 3)
f_2	5	6	3	7	8	10	(1 2)
f_3	6	5	3	7	8	10	(2 4)
f_4	6	7	3	5	8	10	(1 2)
f_5	7	6	3	5	8	10	(4 5)
f_6	7	6	3	8	5	10	(2 4)
f_7	7	8	3	6	5	10	(4 6)
f_8	7	8	3	10	5	6	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



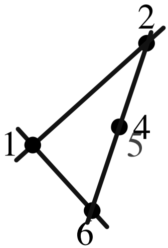
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



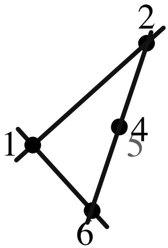
$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



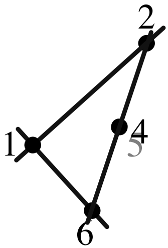
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau\equiv(a\,b)$ such that $\sigma(a)<\sigma(b)$:



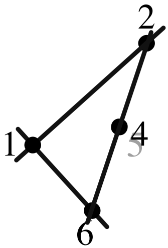
$$C\equiv\begin{array}{c}\begin{array}{cccccc}1&2&3&4&5&6\end{array}\\ \left(\begin{array}{cccccc}1&0&0&0&0&0\\0&1&0&\alpha_7&\alpha_6\alpha_7&0\\0&0&0&1&\alpha_6&\alpha_8\end{array}\right)\end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



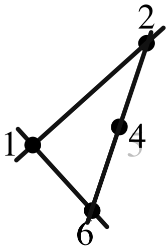
$$C \equiv \begin{array}{c} \\ \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



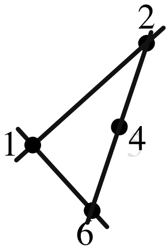
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



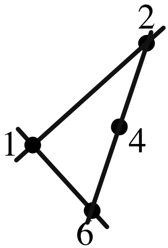
$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

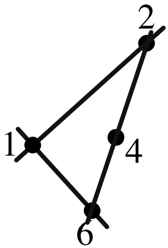
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	3	5	6	7	8	10	$(1\,2)$
f_1	5	3	6	7	8	10	$(2\,3)$
f_2	5	6	3	7	8	10	$(1\,2)$
f_3	6	5	3	7	8	10	$(2\,4)$
f_4	6	7	3	5	8	10	$(1\,2)$
f_5	7	6	3	5	8	10	$(4\,5)$
f_6	7	6	3	8	5	10	$(2\,4)$
f_7	7	8	3	6	5	10	$(4\,6)$
f_8	7	8	3	10	5	6	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



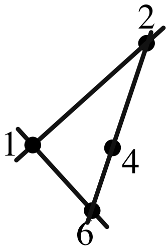
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



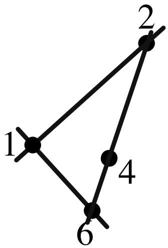
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



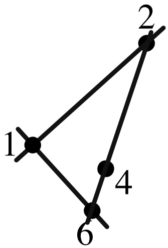
$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



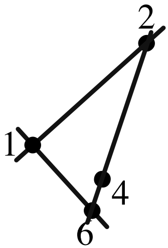
$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



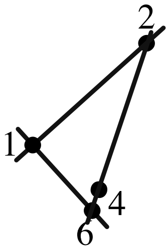
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



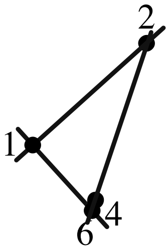
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



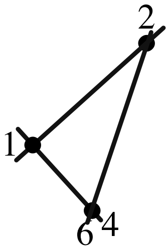
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



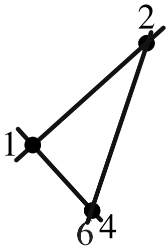
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



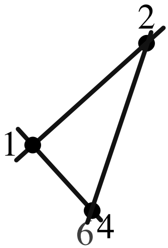
$$\begin{array}{c c} & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ C \equiv & \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



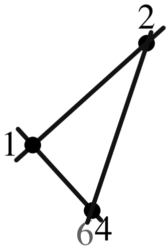
$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$C \equiv \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{pmatrix}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



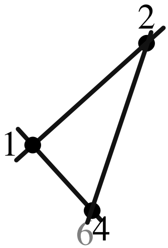
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

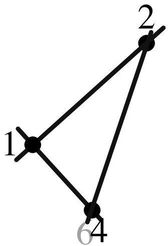
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



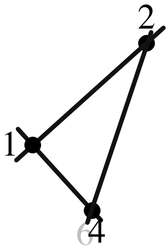
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



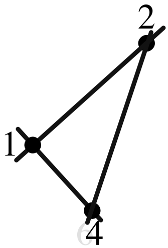
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



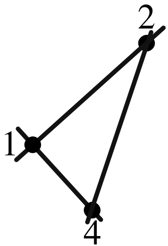
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \hline \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right)$$

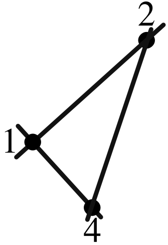
with columns labelled 1, 2, 3, 4, 5, 6.

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



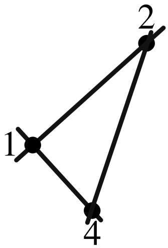
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$C \equiv \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

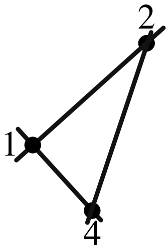
(columns labelled $1\;2\;3\;4\;5\;6$)

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



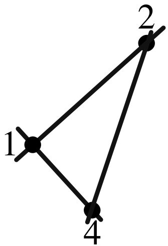
$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right)$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



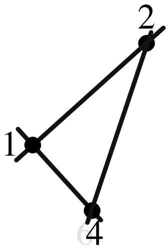
$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right)$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

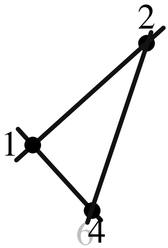
$$(4\,6): \quad c_6 \mapsto c_6 + \alpha_8\, c_4$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right)$$

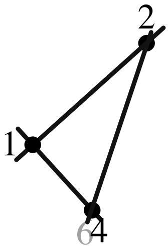
$$(4\,6):\quad c_6 \mapsto c_6 + \alpha_8\, c_4$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	3	5	6	7	8	10	$(1\,2)$
f_1	5	3	6	7	8	10	$(2\,3)$
f_2	5	6	3	7	8	10	$(1\,2)$
f_3	6	5	3	7	8	10	$(2\,4)$
f_4	6	7	3	5	8	10	$(1\,2)$
f_5	7	6	3	5	8	10	$(4\,5)$
f_6	7	6	3	8	5	10	$(2\,4)$
f_7	7	8	3	6	5	10	$(4\,6)$
f_8	7	8	3	10	5	6	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right)$$

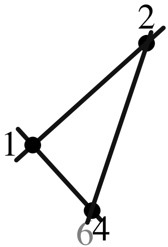
$(4\,6)$: $\quad c_6 \mapsto c_6 + \alpha_8\, c_4$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
f_0	3	5	6	7	8	10	$(1\,2)$
f_1	5	3	6	7	8	10	$(2\,3)$
f_2	5	6	3	7	8	10	$(1\,2)$
f_3	6	5	3	7	8	10	$(2\,4)$
f_4	6	7	3	5	8	10	$(1\,2)$
f_5	7	6	3	5	8	10	$(4\,5)$
f_6	7	6	3	8	5	10	$(2\,4)$
f_7	7	8	3	6	5	10	$(4\,6)$
f_8	7	8	3	10	5	6	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right)$$

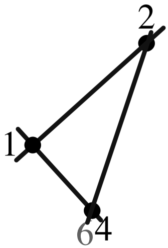
$(4\,6): \quad c_6 \mapsto c_6 + \alpha_8\, c_4$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

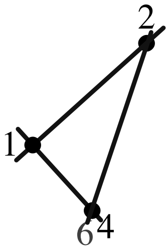
$$(4\,6):\quad c_6 \mapsto c_6 + \alpha_8\, c_4$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

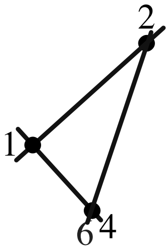
$$(4\,6): \quad c_6 \mapsto c_6 + \alpha_8\, c_4$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right)$$

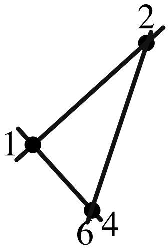
$$(4\,6):\quad c_6 \mapsto c_6 + \alpha_8\, c_4$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	3	5	6	7	8	10	$(1\,2)$
f_1	5	3	6	7	8	10	$(2\,3)$
f_2	5	6	3	7	8	10	$(1\,2)$
f_3	6	5	3	7	8	10	$(2\,4)$
f_4	6	7	3	5	8	10	$(1\,2)$
f_5	7	6	3	5	8	10	$(4\,5)$
f_6	7	6	3	8	5	10	$(2\,4)$
f_7	7	8	3	6	5	10	$(4\,6)$
f_8	7	8	3	10	5	6	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

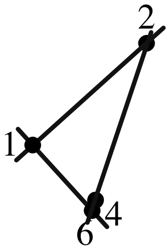
$$(4\,6): \quad c_6 \mapsto c_6 + \alpha_8\, c_4$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	3	5	6	7	8	10	$(1\,2)$
f_1	5	3	6	7	8	10	$(2\,3)$
f_2	5	6	3	7	8	10	$(1\,2)$
f_3	6	5	3	7	8	10	$(2\,4)$
f_4	6	7	3	5	8	10	$(1\,2)$
f_5	7	6	3	5	8	10	$(4\,5)$
f_6	7	6	3	8	5	10	$(2\,4)$
f_7	7	8	3	6	5	10	$(4\,6)$
f_8	7	8	3	10	5	6	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

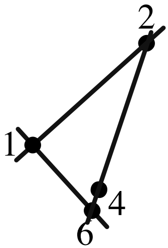
$$(2\,4): \quad c_4 \mapsto c_4 + \alpha_7\, c_2$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

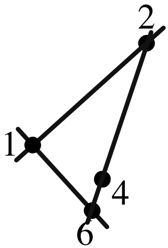
$(2\,4)$: $\quad c_4 \mapsto c_4 + \alpha_7\, c_2$

‘Bridge’ Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

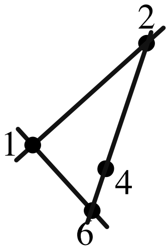
$$(2\,4): \quad c_4 \mapsto c_4 + \alpha_7\, c_2$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
f_0	3	5	6	7	8	10	$(1\,2)$
f_1	5	3	6	7	8	10	$(2\,3)$
f_2	5	6	3	7	8	10	$(1\,2)$
f_3	6	5	3	7	8	10	$(2\,4)$
f_4	6	7	3	5	8	10	$(1\,2)$
f_5	7	6	3	5	8	10	$(4\,5)$
f_6	7	6	3	8	5	10	$(2\,4)$
f_7	7	8	3	6	5	10	$(4\,6)$
f_8	7	8	3	10	5	6	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

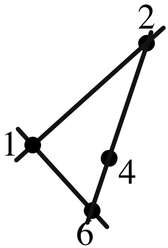
$$(2\,4): \quad c_4 \mapsto c_4 + \alpha_7\, c_2$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \hline \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right)$$

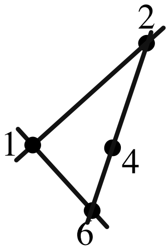
$$(2\,4):\quad c_4 \mapsto c_4 + \alpha_7\, c_2$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right)$$

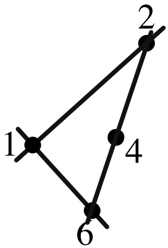
$$(2\,4): \quad c_4 \mapsto c_4 + \alpha_7\, c_2$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right)$$

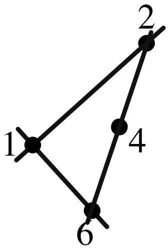
$$(2\,4): \quad c_4 \mapsto c_4 + \alpha_7\, c_2$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \alpha_8 \end{array}\right) \end{array}$$

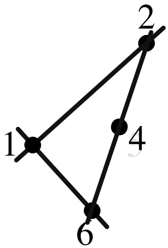
$$(2\,4):\quad c_4 \mapsto c_4 + \alpha_7\, c_2$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

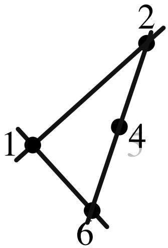
$$(4\,5): \quad c_5 \mapsto c_5 + \alpha_6\, c_4$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

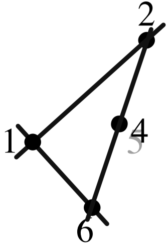
$$(4\,5): \quad c_5 \mapsto c_5 + \alpha_6\, c_4$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

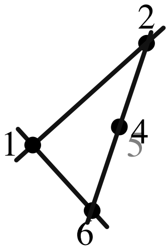
$$(4\,5): \quad c_5 \mapsto c_5 + \alpha_6\, c_4$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \hline \end{array}$$

$$C \equiv \begin{array}{c}\\ \\ \\ \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

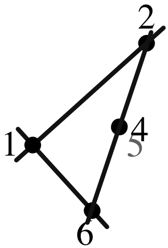
$$(4\,5):\quad c_5 \mapsto c_5 + \alpha_6\, c_4$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$
C \equiv \begin{array}{c} \\ \\ \\ \end{array}
\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}
$$

$$
C \equiv \left(\begin{array}{cccccc}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\
0 & 0 & 0 & 1 & \alpha_6 & \alpha_8
\end{array}\right)
\quad \text{(columns } 1,2,3,4,5,6\text{)}
$$

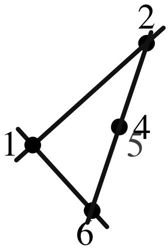
$(4\,5)$: $\; c_5 \mapsto c_5 + \alpha_6\, c_4$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

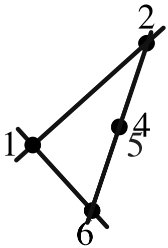
$$(4\,5): \quad c_5 \mapsto c_5 + \alpha_6\, c_4$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

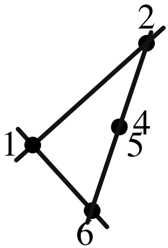
$$(4\,5):\quad c_5 \mapsto c_5 + \alpha_6\, c_4$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

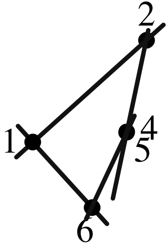
$(4\,5): \quad c_5 \mapsto c_5 + \alpha_6\, c_4$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

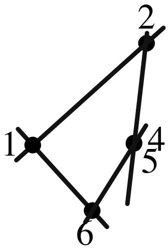
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_5\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

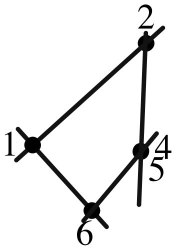
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_5\, c_1$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

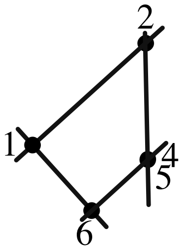
$(1\,2): \quad c_2 \mapsto c_2 + \alpha_5\, c_1$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

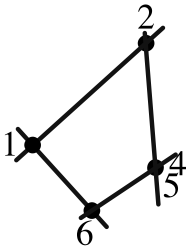
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_5\, c_1$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

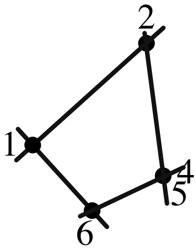
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_5\, c_1$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

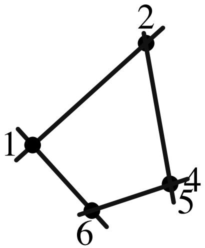
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_5\, c_1$$

‘Bridge’ Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

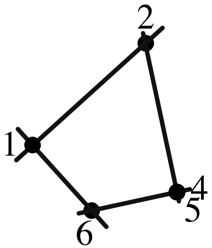
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_5\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_7 & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

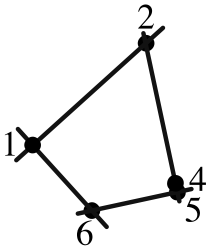
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_5\, c_1$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

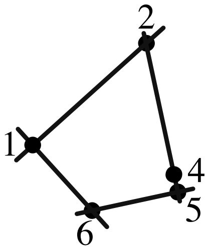
$$(2\,4): \quad c_4 \mapsto c_4 + \alpha_4\, c_2$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

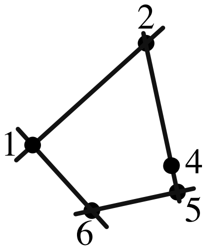
$$(2\,4): \quad c_4 \mapsto c_4 + \alpha_4\, c_2$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

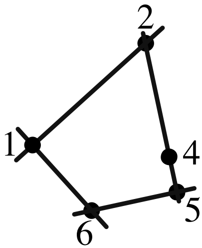
$$(2\,4): \quad c_4 \mapsto c_4 + \alpha_4\, c_2$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

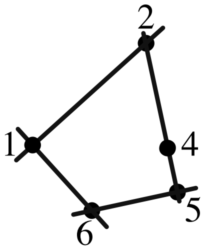
$$(2\,4):\quad c_4 \mapsto c_4 + \alpha_4\, c_2$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

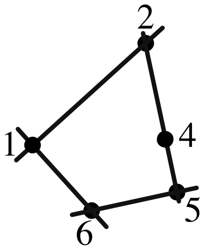
$$(2\,4): \quad c_4 \mapsto c_4 + \alpha_4\, c_2$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

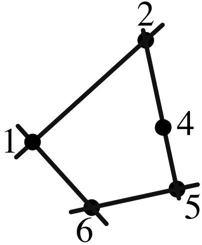
$$(2\,4): \quad c_4 \mapsto c_4 + \alpha_4\, c_2$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

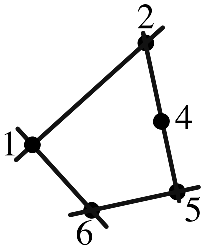
$$(2\,4):\quad c_4 \mapsto c_4 + \alpha_4\, c_2$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & \alpha_5 & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

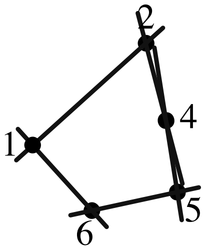
$$(2\,4): \quad c_4 \mapsto c_4 + \alpha_4\, c_2$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

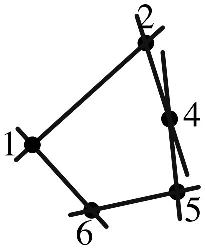
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_3\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{pmatrix} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{pmatrix} \end{array}$$

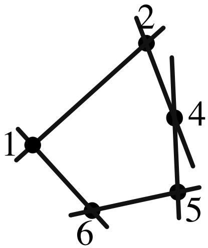
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_3\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau\equiv(a\,b)$ such that $\sigma(a)<\sigma(b)$:



$$C\equiv\begin{array}{c}\begin{array}{cccccc}1 & 2 & 3 & 4 & 5 & 6\end{array}\\ \left(\begin{array}{cccccc}1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0\\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0\\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8\end{array}\right)\end{array}$$

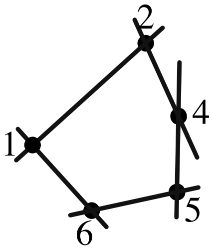
$$(1\,2):\quad c_2\mapsto c_2+\alpha_3\,c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	3	5	6	7	8	10	$(1\,2)$
f_1	5	3	6	7	8	10	$(2\,3)$
f_2	5	6	3	7	8	10	$(1\,2)$
f_3	6	5	3	7	8	10	$(2\,4)$
f_4	6	7	3	5	8	10	$(1\,2)$
f_5	7	6	3	5	8	10	$(4\,5)$
f_6	7	6	3	8	5	10	$(2\,4)$
f_7	7	8	3	6	5	10	$(4\,6)$
f_8	7	8	3	10	5	6	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

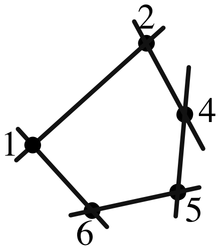
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_3\, c_1$$

‘Bridge’ Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

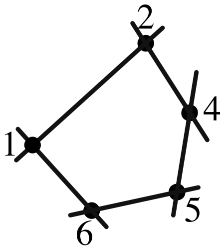
$$(1\,2):\quad c_2 \mapsto c_2 + \alpha_3\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

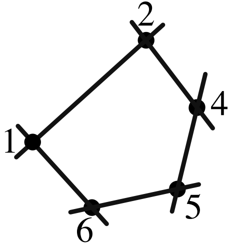
$$(1\,2):\quad c_2 \mapsto c_2 + \alpha_3\, c_1$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

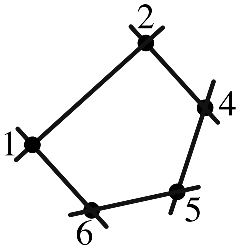
$$(1\,2):\quad c_2 \mapsto c_2 + \alpha_3\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & 0 & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & 0 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

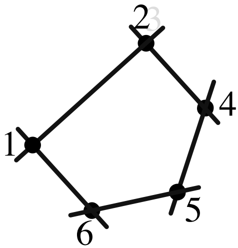
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_3\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	3	5	6	7	8	10	$(1\,2)$
f_1	5	3	6	7	8	10	$(2\,3)$
f_2	5	6	3	7	8	10	$(1\,2)$
f_3	6	5	3	7	8	10	$(2\,4)$
f_4	6	7	3	5	8	10	$(1\,2)$
f_5	7	6	3	5	8	10	$(4\,5)$
f_6	7	6	3	8	5	10	$(2\,4)$
f_7	7	8	3	6	5	10	$(4\,6)$
f_8	7	8	3	10	5	6	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \left(\begin{matrix} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{matrix}\right) \end{array}$$

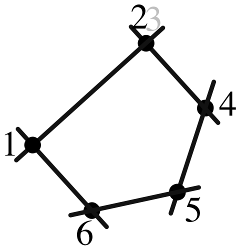
$$(2\,3): \quad c_3 \mapsto c_3 + \alpha_2\, c_2$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

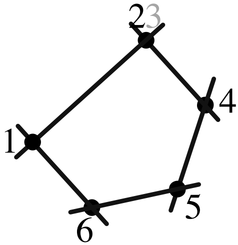
$$(2\,3): \quad c_3 \mapsto c_3 + \alpha_2\, c_2$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

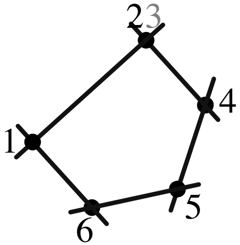
$$(2\,3): \quad c_3 \mapsto c_3 + \alpha_2\, c_2$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \begin{pmatrix} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{pmatrix}$$

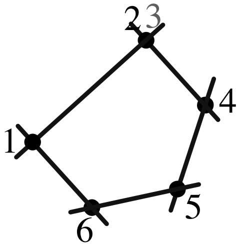
$$(2\,3): \quad c_3 \mapsto c_3 + \alpha_2\, c_2$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

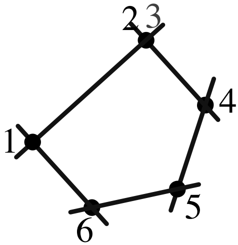
$$(2\,3): \quad c_3 \mapsto c_3 + \alpha_2\, c_2$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

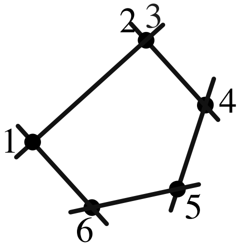
$$(2\,3): \quad c_3 \mapsto c_3 + \alpha_2\, c_2$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	3	5	6	7	8	10	(1 2)
f_1	5	3	6	7	8	10	(2 3)
f_2	5	6	3	7	8	10	(1 2)
f_3	6	5	3	7	8	10	(2 4)
f_4	6	7	3	5	8	10	(1 2)
f_5	7	6	3	5	8	10	(4 5)
f_6	7	6	3	8	5	10	(2 4)
f_7	7	8	3	6	5	10	(4 6)
f_8	7	8	3	10	5	6	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{matrix} 1 & \quad 2 & \quad\quad 3 & \quad\quad 4 & \quad 5 & 6 \end{matrix} \\ \begin{pmatrix} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{pmatrix} \end{array}$$

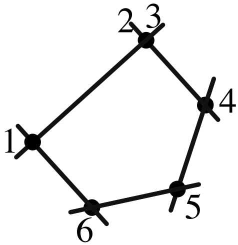
$$(2\,3):\quad c_3 \mapsto c_3 + \alpha_2\, c_2$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

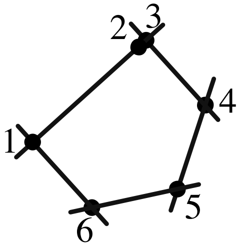
$$(2\,3): \quad c_3 \mapsto c_3 + \alpha_2\, c_2$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

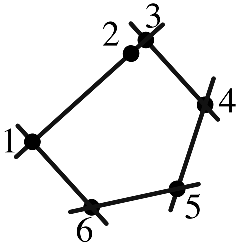
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_1\, c_1$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

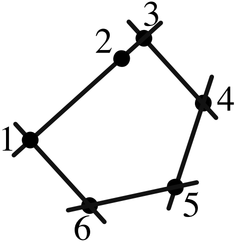
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_1\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

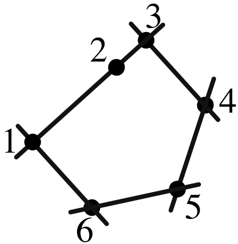
$$(1\,2):\quad c_2 \mapsto c_2 + \alpha_1\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\, b)$ such that $\sigma(a) < \sigma(b)$:



$$
\begin{array}{c c}
 & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\
C \equiv & \left(\begin{array}{cccccc}
1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\
0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\
0 & 0 & 0 & 1 & \alpha_6 & \alpha_8
\end{array}\right)
\end{array}
$$

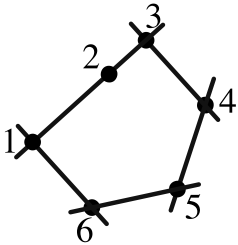
$$(1\,2):\quad c_2 \mapsto c_2 + \alpha_1\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \left(\begin{matrix} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{matrix}\right) \end{array}$$

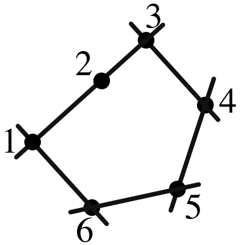
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_1\, c_1$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

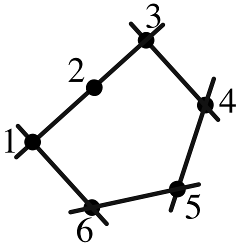
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_1\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

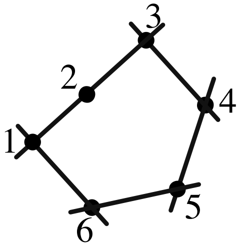
$$(1\,2): \quad c_2 \mapsto c_2 + \alpha_1\, c_1$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ 0 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

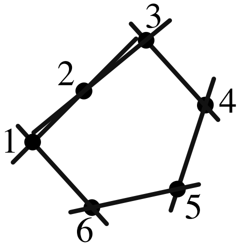
$$(1\,2):\quad c_2 \mapsto c_2 + \alpha_1\, c_1$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \hline \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

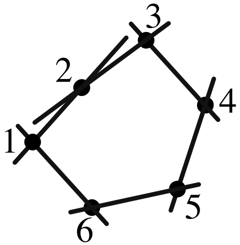
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	
							$(1\,2)$
f_1	{5	3	6	7	8	10}	
							$(2\,3)$
f_2	{5	6	3	7	8	10}	
							$(1\,2)$
f_3	{6	5	3	7	8	10}	
							$(2\,4)$
f_4	{6	7	3	5	8	10}	
							$(1\,2)$
f_5	{7	6	3	5	8	10}	
							$(4\,5)$
f_6	{7	6	3	8	5	10}	
							$(2\,4)$
f_7	{7	8	3	6	5	10}	
							$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

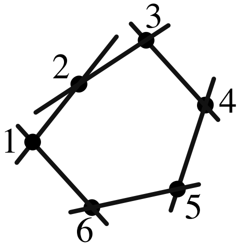
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau\equiv(a\,b)$ such that $\sigma(a)<\sigma(b)$:



$$C\equiv\begin{array}{c}\begin{array}{cccccc}1 & 2 & 3 & 4 & 5 & 6\end{array}\\ \left(\begin{array}{cccccc}1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0\\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0\\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8\end{array}\right)\end{array}$$

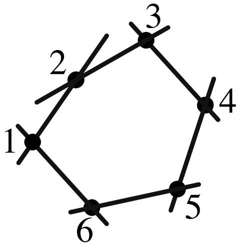
$$(6\,1):\quad c_1\mapsto c_1+\alpha_0\,c_6$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	↓	↓	↓	↓	↓	↓	
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

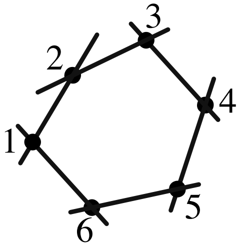
$$(6\,1):\quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

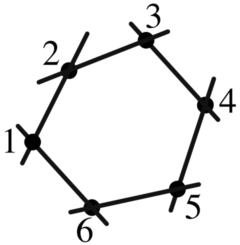
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	3	5	6	7	8	10	$(1\,2)$
f_1	5	3	6	7	8	10	$(2\,3)$
f_2	5	6	3	7	8	10	$(1\,2)$
f_3	6	5	3	7	8	10	$(2\,4)$
f_4	6	7	3	5	8	10	$(1\,2)$
f_5	7	6	3	5	8	10	$(4\,5)$
f_6	7	6	3	8	5	10	$(2\,4)$
f_7	7	8	3	6	5	10	$(4\,6)$
f_8	7	8	3	10	5	6	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

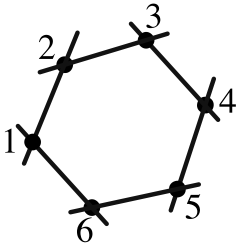
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

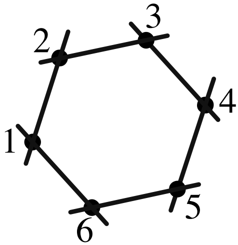
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

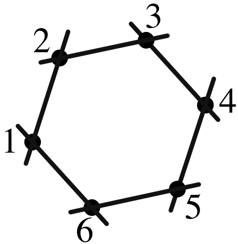
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

‘Bridge’ Decomposition

	1	2	3	4	5	6	τ
f_0	3	5	6	7	8	10	$(1\,2)$
f_1	5	3	6	7	8	10	$(2\,3)$
f_2	5	6	3	7	8	10	$(1\,2)$
f_3	6	5	3	7	8	10	$(2\,4)$
f_4	6	7	3	5	8	10	$(1\,2)$
f_5	7	6	3	5	8	10	$(4\,5)$
f_6	7	6	3	8	5	10	$(2\,4)$
f_7	7	8	3	6	5	10	$(4\,6)$
f_8	7	8	3	10	5	6	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \hline \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

(columns labelled $1\;2\;3\;4\;5\;6$)

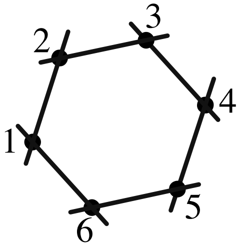
$$(6\,1): \quad c_1 \mapsto c_1 + \alpha_0\, c_6$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



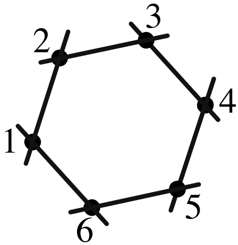
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{pmatrix} \end{array}$$

'Bridge' Decomposition

	1	2	3	4	5	6	τ
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

There are many ways to decompose a permutation into transpositions—*e.g.*, always choose the first transposition $\tau \equiv (a\,b)$ such that $\sigma(a) < \sigma(b)$:



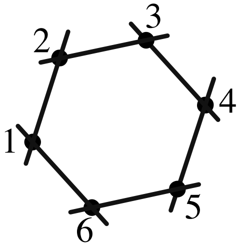
$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \end{array}$$

$$C \equiv \left(\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	$(1\,2)$
f_1	{5	3	6	7	8	10}	$(2\,3)$
f_2	{5	6	3	7	8	10}	$(1\,2)$
f_3	{6	5	3	7	8	10}	$(2\,4)$
f_4	{6	7	3	5	8	10}	$(1\,2)$
f_5	{7	6	3	5	8	10}	$(4\,5)$
f_6	{7	6	3	8	5	10}	$(2\,4)$
f_7	{7	8	3	6	5	10}	$(4\,6)$
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions



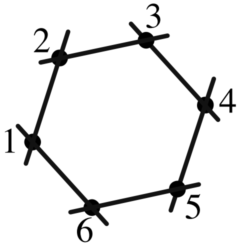
$$
C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}
$$

‘Bridge’ Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

$$\mathcal{L}_{6,3} \equiv \frac{d\alpha_0}{\alpha_0} \cdots \frac{d\alpha_8}{\alpha_8}$$



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \end{array}$$

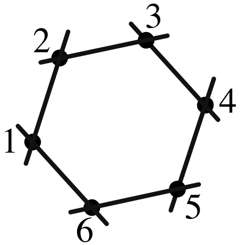
$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

$$\mathcal{L}_{6,3} \equiv \frac{d\alpha_0}{\alpha_0} \cdots \frac{d\alpha_8}{\alpha_8}$$



$$C \equiv \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \hline \end{array}$$

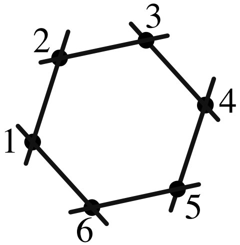
$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

‘Bridge’ Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

$$\mathcal{L}_{6,3} \equiv \frac{d\alpha_0}{\alpha_0} \cdots \frac{d\alpha_8}{\alpha_8} \quad = \frac{d^{3\times 6}C}{\mathrm{vol}(GL(3))}\frac{1}{(123)(234)(345)(456)(561)(612)}$$



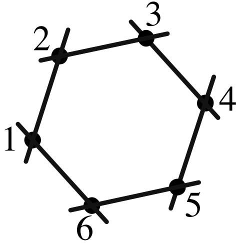
$$C \equiv \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ \hline \end{array}$$
$$C \equiv \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right)$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6 }	

Canonical Coordinates for Computing On-Shell Functions

$$\mathcal{L}_{n,k} \equiv \frac{d\alpha_1}{\alpha_1}\cdots\frac{d\alpha_{k(n-k)}}{\alpha_{k(n-k)}} = \frac{d^{k\times n}C}{\mathrm{vol}(GL(k))}\,\frac{1}{(1\cdots k)\,(2\cdots k{+}1)\cdots(n\cdots k{-}1)}$$



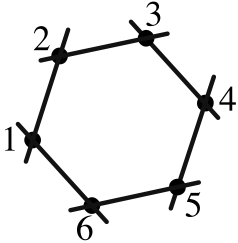
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

‘Bridge’ Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

$$\mathcal{L}_{n,k} \equiv \frac{d\alpha_1}{\alpha_1} \cdots \frac{d\alpha_{k(n-k)}}{\alpha_{k(n-k)}} = \frac{d^{k\times n}C}{\mathrm{vol}(GL(k))} \frac{1}{(1\cdots k)\,(2\cdots k{+}1)\cdots(n\cdots k{-}1)}$$



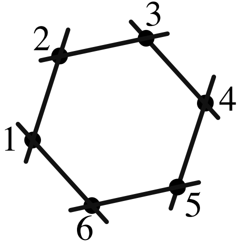
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

$$\mathcal{L}_{n,k} \equiv \frac{d\alpha_1}{\alpha_1} \cdots \frac{d\alpha_{k(n-k)}}{\alpha_{k(n-k)}} = \frac{d^{k\times n}C}{\mathrm{vol}(GL(k))} \frac{1}{(1\cdots k)\,(2\cdots k{+}1)\cdots(n\cdots k{-}1)}$$



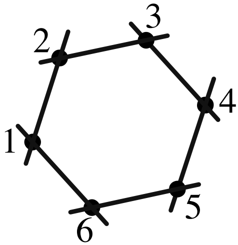
$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates for Computing On-Shell Functions

$$\mathcal{L}_{n,k} \equiv \frac{d\alpha_1}{\alpha_1} \cdots \frac{d\alpha_{k(n-k)}}{\alpha_{k(n-k)}} = \frac{d^{k\times n}C}{\mathrm{vol}(GL(k))} \frac{1}{(1\cdots k)\,(2\cdots k{+}1)\cdots(n\cdots k{-}1)}$$



$$C \equiv \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left(\begin{array}{cccccc} 1 & (\alpha_1+\alpha_3+\alpha_5) & \alpha_2(\alpha_3+\alpha_5) & \alpha_4\alpha_5 & 0 & 0 \\ 0 & 1 & \alpha_2 & (\alpha_4+\alpha_7) & \alpha_6\alpha_7 & 0 \\ \alpha_0\alpha_8 & 0 & 0 & 1 & \alpha_6 & \alpha_8 \end{array}\right) \end{array}$$

'Bridge' Decomposition

	1 ↓	2 ↓	3 ↓	4 ↓	5 ↓	6 ↓	τ
f_0	{3	5	6	7	8	10}	(1 2)
f_1	{5	3	6	7	8	10}	(2 3)
f_2	{5	6	3	7	8	10}	(1 2)
f_3	{6	5	3	7	8	10}	(2 4)
f_4	{6	7	3	5	8	10}	(1 2)
f_5	{7	6	3	5	8	10}	(4 5)
f_6	{7	6	3	8	5	10}	(2 4)
f_7	{7	8	3	6	5	10}	(4 6)
f_8	{7	8	3	10	5	6}	

Canonical Coordinates and the Manifestation of the *Yangian*

All on-shell diagrams, in terms of canonical coordinates, take the form:

Canonical Coordinates and the Manifestation of the *Yangian*

All on-shell diagrams, in terms of canonical coordinates, take the form:

$$f = \int \frac{d\alpha_1}{\alpha_1} \wedge \cdots \wedge \frac{d\alpha_d}{\alpha_d} \; \delta^{k\times 4}\big(C(\vec{\alpha})\cdot\widetilde{\eta}\big)\delta^{k\times 2}\big(C(\vec{\alpha})\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C(\vec{\alpha})^{\perp}\big)$$

Canonical Coordinates and the Manifestation of the *Yangian*

All on-shell diagrams, in terms of canonical coordinates, take the form:

$$f = \int \frac{d\alpha_1}{\alpha_1} \wedge \cdots \wedge \frac{d\alpha_d}{\alpha_d} \; \delta^{k\times 4}\big(C(\vec{\alpha})\cdot\widetilde{\eta}\big)\delta^{k\times 2}\big(C(\vec{\alpha})\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C(\vec{\alpha})^{\perp}\big)$$

Measure-preserving diffeomorphisms leave the function invariant

Canonical Coordinates and the Manifestation of the *Yangian*

All on-shell diagrams, in terms of canonical coordinates, take the form:

$$f = \int \frac{d\alpha_1}{\alpha_1} \wedge \cdots \wedge \frac{d\alpha_d}{\alpha_d} \; \delta^{k\times 4}\big(C(\vec{\alpha})\cdot\widetilde{\eta}\big)\delta^{k\times 2}\big(C(\vec{\alpha})\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C(\vec{\alpha})^{\perp}\big)$$

Measure-preserving diffeomorphisms leave the function invariant, but—via the δ-functions—can be recast variations of the kinematical data.

Canonical Coordinates and the Manifestation of the *Yangian*

All on-shell diagrams, in terms of canonical coordinates, take the form:

$$f = \int \frac{d\alpha_1}{\alpha_1} \wedge \cdots \wedge \frac{d\alpha_d}{\alpha_d} \; \delta^{k\times 4}\big(C(\vec{\alpha})\cdot\widetilde{\eta}\big)\delta^{k\times 2}\big(C(\vec{\alpha})\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C(\vec{\alpha})^{\perp}\big)$$

Measure-preserving diffeomorphisms leave the function invariant, but—via the δ-functions—can be recast variations of the kinematical data.

The *Yangian* corresponds to those diffeomorphisms that simultaneously preserve the measures of *all* on-shell diagrams.

Canonical Coordinates and the Manifestation of the *Yangian*

All on-shell diagrams, in terms of canonical coordinates, take the form:

$$f = \int \frac{d\alpha_1}{\alpha_1} \wedge \cdots \wedge \frac{d\alpha_d}{\alpha_d} \; \delta^{k\times4}\big(C(\vec{\alpha})\cdot\widetilde{\eta}\big)\delta^{k\times2}\big(C(\vec{\alpha})\cdot\widetilde{\lambda}\big)\delta^{2\times(n-k)}\big(\lambda\cdot C(\vec{\alpha})^{\perp}\big)$$

Measure-preserving diffeomorphisms leave the function invariant, but—via the δ-functions—can be recast variations of the kinematical data.

The *Yangian* corresponds to those diffeomorphisms that simultaneously preserve the measures of *all* on-shell diagrams.

On-Shell Recursion of Loop-Amplitude Integrands

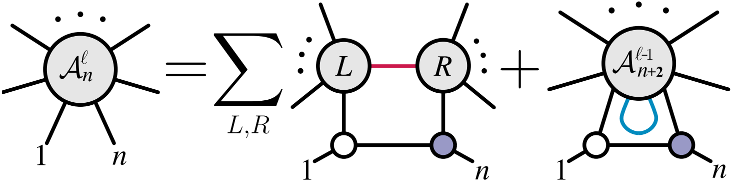
Let's look at an example of how loop amplitudes are represented by recursion.

On-Shell Recursion of Loop-Amplitude Integrands

Let's look at an example of how loop amplitudes are represented by recursion.

On-Shell Recursion of Loop-Amplitude Integrands

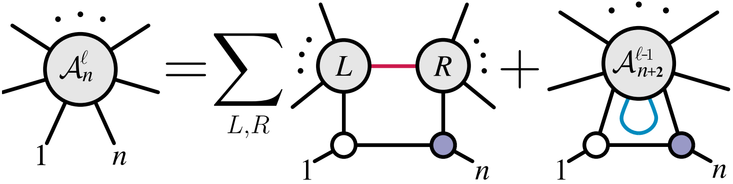
Let's look at an example of how loop amplitudes are represented by recursion.



On-Shell Recursion of Loop-Amplitude Integrands

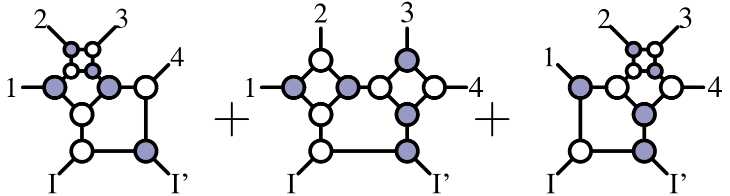
Let's look at an example of how loop amplitudes are represented by recursion.

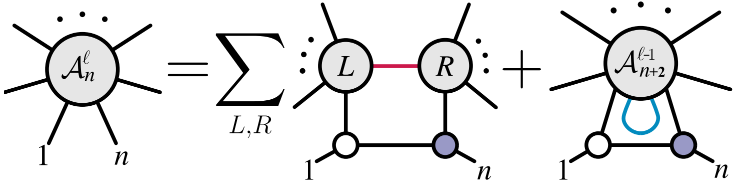
For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:



On-Shell Recursion of Loop-Amplitude Integrands

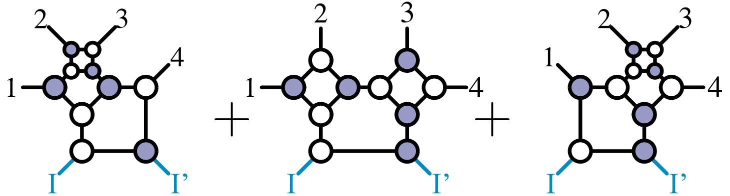
Let's look at an example of how loop amplitudes are represented by recursion.
For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:

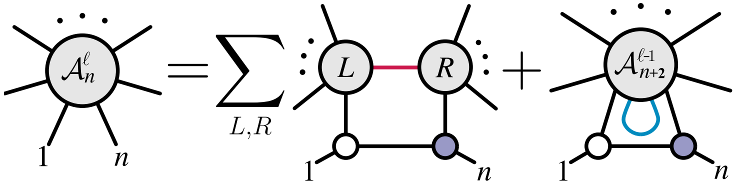




On-Shell Recursion of Loop-Amplitude Integrands

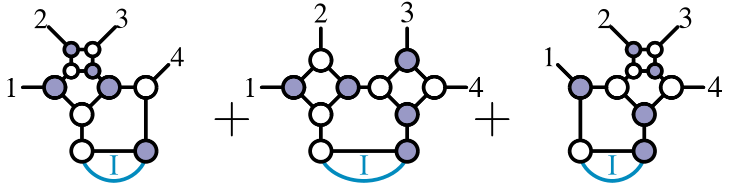
Let's look at an example of how loop amplitudes are represented by recursion. For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:

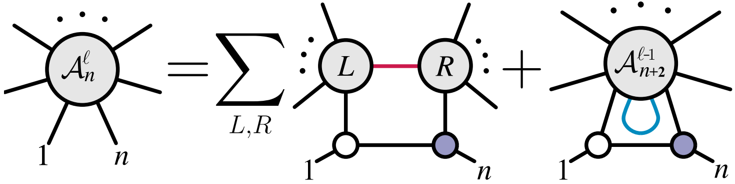




On-Shell Recursion of Loop-Amplitude Integrands

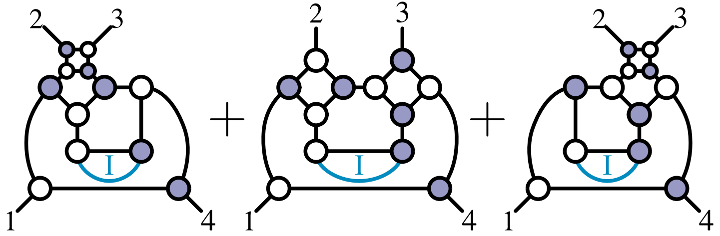
Let's look at an example of how loop amplitudes are represented by recursion. For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:

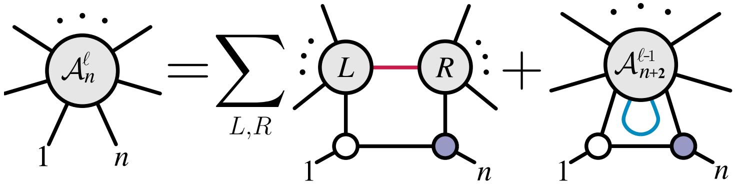




On-Shell Recursion of Loop-Amplitude Integrands

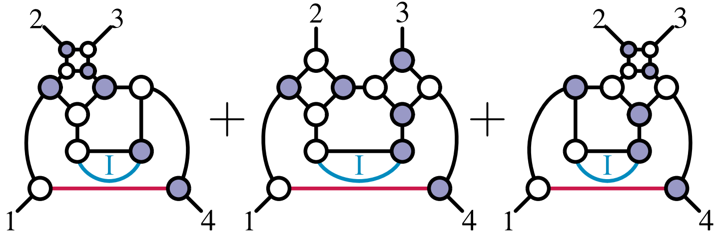
Let's look at an example of how loop amplitudes are represented by recursion. For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:

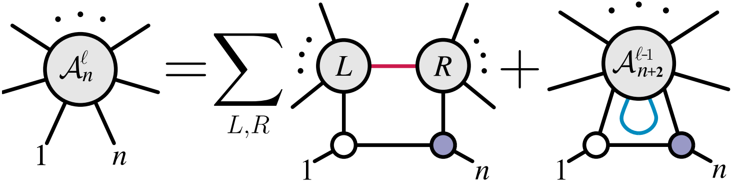




On-Shell Recursion of Loop-Amplitude Integrands

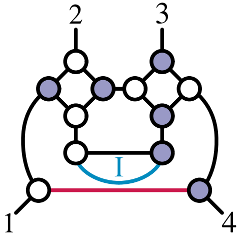
Let's look at an example of how loop amplitudes are represented by recursion. For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:

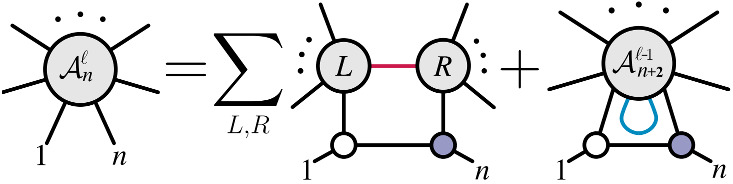




On-Shell Recursion of Loop-Amplitude Integrands

Let's look at an example of how loop amplitudes are represented by recursion. For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:

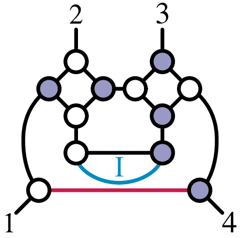


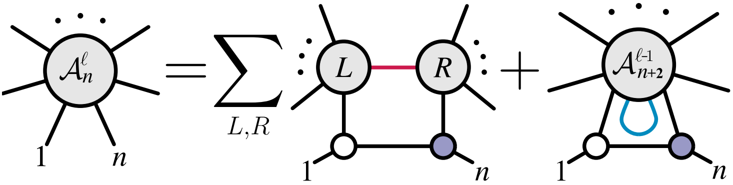


On-Shell Recursion of Loop-Amplitude Integrands

Let's look at an example of how loop amplitudes are represented by recursion.

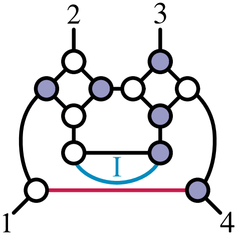
For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:



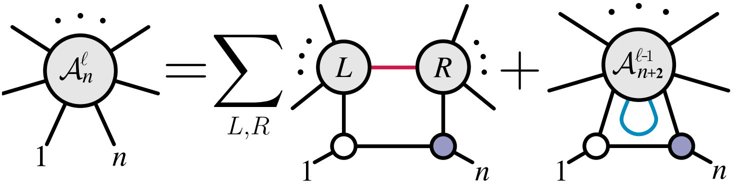


On-Shell Recursion of Loop-Amplitude Integrands

Let's look at an example of how loop amplitudes are represented by recursion. For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:

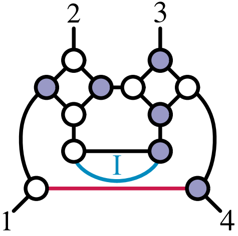


$$\int\limits_{\ell\in\mathbb{R}^{3,1}} d^4\ell$$

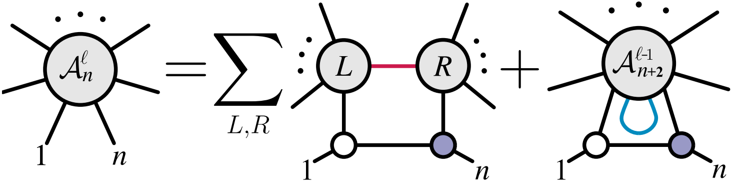


On-Shell Recursion of Loop-Amplitude Integrands

Let's look at an example of how loop amplitudes are represented by recursion.
For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:



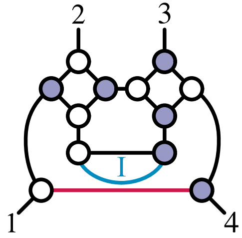
$$\int\limits_{\ell\in\mathbb{R}^{3,1}} d^4\ell \quad \Leftrightarrow$$



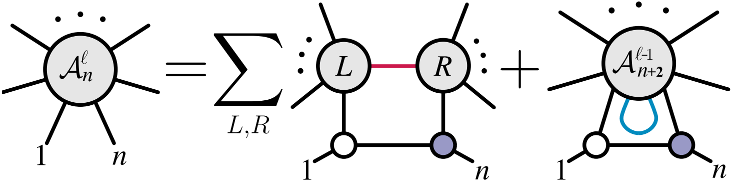
On-Shell Recursion of Loop-Amplitude Integrands

Let's look at an example of how loop amplitudes are represented by recursion.

For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:



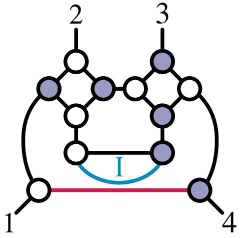
$$\int\limits_{\ell\in\mathbb{R}^{3,1}} d^4\ell \quad \Leftrightarrow \quad \int\limits_{\ell\equiv(\lambda_{\mathrm{I}}\widetilde{\lambda}_{\mathrm{I}}+\alpha\lambda_1\widetilde{\lambda}_4)\in\mathbb{R}^{3,1}} \frac{d^2\lambda_{\mathrm{I}}d^2\widetilde{\lambda}_{\mathrm{I}}}{\mathrm{vol}(GL_1)} d\alpha \langle \mathrm{I}1\rangle\,[n\mathrm{I}]$$



On-Shell Recursion of Loop-Amplitude Integrands

Let's look at an example of how loop amplitudes are represented by recursion.

For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:

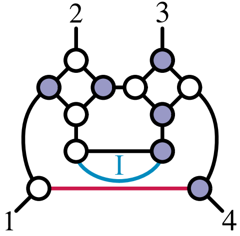


$$\int\limits_{\ell\in\mathbb{R}^{3,1}} d^4\ell \quad \Longleftrightarrow \quad \int\limits_{\ell\equiv(\lambda_{\mathrm{I}}\widetilde{\lambda}_{\mathrm{I}}+\alpha\lambda_1\widetilde{\lambda}_4)\in\mathbb{R}^{3,1}} \frac{d^2\lambda_{\mathrm{I}}d^2\widetilde{\lambda}_{\mathrm{I}}}{\mathrm{vol}(GL_1)} d\alpha \langle \mathrm{I}1\rangle\,[n\mathrm{I}]$$

$$\mathcal{A}_4^{(2),0}\times\int\limits_{\ell\in\mathbb{R}^{3,1}} d\log\left(\frac{\ell^2}{(\ell-\ell^*)^2}\right) d\log\left(\frac{(\ell+p_1)^2}{(\ell-\ell^*)^2}\right) d\log\left(\frac{(\ell+p_1+p_2)^2}{(\ell-\ell^*)^2}\right) d\log\left(\frac{(\ell-p_4)^2}{(\ell-\ell^*)^2}\right)$$

On-Shell Recursion of Loop-Amplitude Integrands

Let's look at an example of how loop amplitudes are represented by recursion.
For $\mathcal{A}_4^{(2),1}$, the only terms come from the 'forward limit' of the tree $\mathcal{A}_6^{(3),0}$:



$$\int\limits_{\ell\in\mathbb{R}^{3,1}} d^4\ell \quad\Leftrightarrow\quad \int\limits_{\ell\equiv(\lambda_{\mathrm{I}}\widetilde{\lambda}_{\mathrm{I}}+\alpha\lambda_1\widetilde{\lambda}_4)\in\mathbb{R}^{3,1}} \frac{d^2\lambda_{\mathrm{I}}d^2\widetilde{\lambda}_{\mathrm{I}}}{\mathrm{vol}(GL_1)}d\alpha\langle \mathrm{I}1\rangle\,[n\mathrm{I}]$$

$$\mathcal{A}_4^{(2),0}\times\int\limits_{\ell\in\mathbb{R}^{3,1}} d\log\left(\frac{\ell^2}{(\ell-\ell^*)^2}\right)d\log\left(\frac{(\ell+p_1)^2}{(\ell-\ell^*)^2}\right)d\log\left(\frac{(\ell+p_1+p_2)^2}{(\ell-\ell^*)^2}\right)d\log\left(\frac{(\ell-p_4)^2}{(\ell-\ell^*)^2}\right)$$

$$=\mathcal{A}_4^{(2),0}\times\int\limits_{\ell\in\mathbb{R}^{3,1}} d^4\ell\frac{(p_1+p_2)^2(p_3+p_4)^2}{\ell^2(\ell+p_1)^2(\ell+p_1+p_2)^2(\ell-p_4)^2}$$