Integrand reduction techniques at one and higher loops

Tiziano Peraro

Max-Planck-Institut für Physik, München, Germany

Higgs Centre for Theoretical Physics – Edinburgh
9 May 2014

Outline

Introduction and motivation

Motivation

- Theoretical understanding of scattering amplitudes
  - basic analytic/algebraic structure of loop integrands and integrals
- Need of theoretical predictions for colliders (LHC)
  - probing large phase space $\Rightarrow$ several external legs
  - need of NLO or higher accuracy $\Rightarrow$ computations at the loop level
- Automation of methods for predictions in perturbative QFT

We developed a coherent framework for the integrand decomposition of Feynman integrals

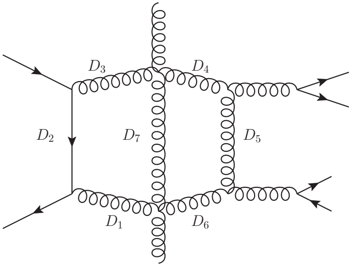
- based on simple concepts of algebraic geometry
- applicable at all loops

Integrand reduction

- The integrand of a generic ℓ-loop integral:
  - is a rational function in the components of the loop momenta $\bar{q}_i$
  - polynomial numerator $\mathcal{N}_{i_1\cdots i_n}$

$$\mathcal{M}_n = \int d^d\bar{q}_1 \cdots d^d\bar{q}_\ell \;\; \mathcal{I}_{i_1\cdots i_n}, \qquad \mathcal{I}_{i_1\cdots i_n} \equiv \frac{\mathcal{N}_{i_1\cdots i_n}}{D_{i_1}\cdots D_{i_n}}$$

  - quadratic polynomial denominators D_i
    - they correspond to Feynman loop propagators



$$D_i = \Big(\sum_j (-)^{s_{ij}}\bar{q}_j + p_i\Big)^2 - m_i^2$$

$$\underbrace{\bar{q}_i}_{d\text{-dimensional}} = \underbrace{q_i}_{4\text{-dimensional}} + \underbrace{\vec{\mu}_i}_{(-2\epsilon)\text{-dimensional}}$$

$$\bar{q}_i \cdot \bar{q}_j = (q_i \cdot q_j) - \mu_{ij}$$

Integrand reduction

The idea

Manipulate the integrand and reduce it to a linear combination of "simpler" integrands.

- The integrand-reduction algorithm leads to

$$\mathcal{I}_{i_1\cdots i_n} \equiv \frac{\mathcal{N}_{i_1\cdots i_n}}{D_{i_1}\cdots D_{i_n}} = \frac{\Delta_{i_1\cdots i_n}}{D_{i_1}\cdots D_{i_n}} + \cdots + \sum_{k=1}^{n} \frac{\Delta_{i_k}}{D_{i_k}} + \Delta_{\emptyset}$$

- The residues $\Delta_{i_1\cdots i_k}$ are irreducible polynomials in $\bar{q}_i$
  - can't be written as a combination of denominators $D_{i_1}, \ldots, D_{i_k}$
  - universal topology-dependent parametric form
  - the coefficients of the parametrization are process-dependent

From integrands to integrals

- By integrating the integrand decomposition

$$\mathcal{M}_n = \int d^d\bar{q}_1 \cdots d^d\bar{q}_\ell \left(\frac{\Delta_{i_1\cdots i_n}}{D_{i_1} \cdots D_{i_n}} + \cdots + \sum_{k=1}^{n} \frac{\Delta_{i_k}}{D_{i_k}} + \Delta_\emptyset \right)$$

  - some terms vanish and do not contribute to the amplitude $\Rightarrow$ spurious terms
  - non-vanishing terms give Master Integrals (MIs)
- The amplitude is a linear combination of MIs
- The coefficients of this linear combination can be identified with some of the coefficients which parametrize the polynomial residues

From integrands to integrals

- By integrating the integrand decomposition

$$\mathcal{M}_n = \int d^d\bar{q}_1 \cdots d^d\bar{q}_\ell \left(\frac{\Delta_{i_1\cdots i_n}}{D_{i_1}\cdots D_{i_n}} + \cdots + \sum_{k=1}^{n} \frac{\Delta_{i_k}}{D_{i_k}} + \Delta_\emptyset \right)$$

  - some terms vanish and do not contribute to the amplitude $\Rightarrow$ spurious terms
  - non-vanishing terms give Master Integrals (MIs)

- The amplitude is a linear combination of MIs
- The coefficients of this linear combination can be identified with some of the coefficients which parametrize the polynomial residues

$\Rightarrow$ reduction to MIs $\equiv$ polynomial fit of the residues

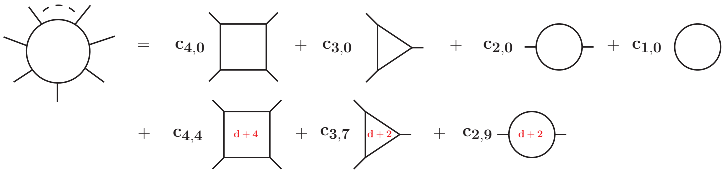
The one-loop decomposition

At one loop the result is well known:

- the integrand decomposition
  [Ossola, Papadopoulos, Pittau (2007); Ellis, Giele, Kunszt, Melnikov (2008)]

$$\mathcal{I}_{i_1\cdots i_n} = \frac{\mathcal{N}_{i_1\cdots i_n}}{D_{i_1}\cdots D_{i_n}} = \sum_{j_1\cdots j_5} \frac{\Delta_{j_1 j_2 j_3 j_4 j_5}}{D_{j_1} D_{j_2} D_{j_3} D_{j_4} D_{j_5}} + \sum_{j_1 j_2 j_3 j_4} \frac{\Delta_{j_1 j_2 j_3 j_4}}{D_{j_1} D_{j_2} D_{j_3} D_{j_4}} + \sum_{j_1 j_2 j_3} \frac{\Delta_{j_1 j_2 j_3}}{D_{j_1} D_{j_2} D_{j_3}} + \sum_{j_1 j_2} \frac{\Delta_{j_1 j_2}}{D_{j_1} D_{j_2}} + \sum_{j_1} \frac{\Delta_{j_1}}{D_{j_1}}$$

- the integral decomposition



- all the Mater Integrals are known!

Integrand reduction and polynomials

- At ℓ-loops we want to achieve the integrand decomposition:

$$\mathcal{I}_{i_1\cdots i_n}(\bar{q}_1, \cdots, \bar{q}_\ell) \equiv \frac{\mathcal{N}_{i_1\cdots i_n}}{D_{i_1}\cdots D_{i_n}} = \underbrace{\frac{\Delta_{i_1\cdots i_n}}{D_{i_1}\cdots D_{i_n}} + \cdots + \sum_{k=1}^{n} \frac{\Delta_{i_k}}{D_{i_k}}}_{\text{they must be irreducible}} + \Delta_\emptyset$$

- We trade $(\bar{q}_1, \ldots, \bar{q}_\ell)$ with their coordinates $\mathbf{z} \equiv (z_1, \ldots, z_m)$
  - $\Rightarrow$ numerator and denominators $\equiv$ polynomials in $\mathbf{z}$

$$\mathcal{I}_{i_1\cdots i_n}(\mathbf{z}) \equiv \frac{\mathcal{N}_{i_1\cdots i_n}(\mathbf{z})}{D_{i_1}(\mathbf{z})\cdots D_{i_n}(\mathbf{z})}$$

$\Rightarrow$ Integrand reduction $\equiv$ problem of multivariate polynomial division

The problem of the determination of the residues of a generic diagram has been solved. [Y. Zhang (2012), P. Mastrolia, E. Mirabella, G. Ossola, T.P. (2012-14)]

Residues via polynomial division

Y. Zhang (2012), P. Mastrolia, E. Mirabella, G. Ossola, T.P. (2012)

- Define the Ideal of polynomials

$$\mathcal{J}_{i_1\cdots i_n} \equiv \langle D_{i_1}, \ldots, D_{i_n} \rangle = \left\{ p(\mathbf{z}) \; : \; p(\mathbf{z}) = \sum_j h_j(\mathbf{z}) D_j(\mathbf{z}), \; h_j \in P[z] \right\}$$

- Take a Gröbner basis $G_{\mathcal{J}_{i_1\cdots i_n}}$ of $\mathcal{J}_{i_1\cdots i_n}$

$$G_{\mathcal{J}_{i_1\cdots i_n}} = \{g_1, \ldots, g_s\} \quad \text{such that} \quad \mathcal{J}_{i_1\cdots i_n} = \langle g_1, \ldots, g_s \rangle$$

- Perform the multivariate polynomial division $\mathcal{N}_{i_1\cdots i_n}/G_{\mathcal{J}_{i_1\cdots i_n}}$

$$\mathcal{N}_{i_1\cdots i_n}(z) = \underbrace{\sum_{k=1}^{n} \mathcal{N}_{i_1\cdots i_{k-1} i_{k+1}\cdots i_n}(z)\, D_{i_k}(z)}_{\text{quotient} \, \in \, \mathcal{J}_{i_1\cdots i_n}} + \underbrace{\Delta_{i_1\cdots i_n}(z)}_{\text{remainder}}$$

- The remainder $\Delta_{i_1\cdots i_n}$ is irreducible $\Rightarrow$ can be identified with the residue

Recursive Relation for the integrand decomposition

P. Mastrolia, E. Mirabella, G. Ossola, T.P. (2012)

The recursive formula

$$\mathcal{N}_{i_1\cdots i_n} = \sum_{k=1}^{n} \mathcal{N}_{i_1\cdots i_{k-1} i_{k+1}\cdots i_n}\, D_{i_k} + \Delta_{i_1\cdots i_n}$$

$$\mathcal{I}_{i_1\cdots i_n} \equiv \frac{\mathcal{N}_{i_1\cdots i_n}}{D_{i_1}\cdots D_{i_n}} = \sum_{k} \mathcal{I}_{i_1\cdots i_{k-1} i_{k+1}\cdots i_n} + \frac{\Delta_{i_1\cdots i_n}}{D_{i_1}\cdots D_{i_n}}$$

- Fit-on-the-cut approach
  - from a generic $\mathcal{N}$, get the parametric form of the residues Δ
  - determine the coefficients sampling on the cuts (impose $D_i = 0$)
- Divide-and-Conquer approach
  - generate the $\mathcal{N}$ of the process
  - compute the residues by iterating the polynomial division algorithm

Fit-on-the-cut approach

[Ossola, Papadopoulos, Pittau (2007)]

The decomposition of the numerator

$$\mathcal{N}_{i_1\cdots i_n} = \sum_{k=0}^{n} \sum_{\{j_1\cdots j_k\}} \Delta_{j_1\cdots j_k} \prod_{h\in\{i_1\cdots i_n\}\setminus\{j_1\cdots j_k\}} D_h.$$

- Fit the coefficients of the residues sampling on the multiple cuts
- First step: n-ple cut
  - impose $D_{i_1} = \cdots = D_{i_n} = 0$

$$\Delta_{i_1\cdots i_n} = \mathcal{N}_{i_1\cdots i_n}$$

- Further steps: k-ple cut
  - impose $D_{i_1} = \cdots = D_{i_k} = 0$ for any subset $\{i_1 \ldots i_k\}$

$$\Delta_{i_1\cdots i_k} = \frac{\mathcal{N}_{i_1\cdots i_n} - \text{higher-point contibrutions}}{\prod_{h\neq i_1,\ldots,i_k} D_h}$$

Fit-on-the-cut approach: The reducibility criterion

What happens if a cut has no solution?

The reducibility criterion

- If a cut $D_{i_1} = \cdots = D_{i_k} = 0$ has no solutions, the associated residue vanishes. In other words, any numerator is completely reducible.
- This generally happens with overdetermined systems i.e. when the number of cut denominators is higher than the one of loop coordinates.

- When $D_{i_1} = \cdots = D_{i_k} = 0$ has no solution:

$$\Delta_{i_1 \cdots i_k} = 0 \qquad \Rightarrow \text{no need to perform the fit}$$

$$\mathcal{N}_{i_1 \cdots i_n} = \sum_{k=1}^{n} \mathcal{N}_{i_1 \cdots i_{k-1} i_{k+1} \cdots i_n} D_{i_k}$$

$$\mathcal{I}_{i_1 \cdots i_n} = \sum_{k} \mathcal{I}_{i_1 \cdots i_{k-1} i_{k+1} \cdots i_n}$$

Fit-on-the-cut approach: The maximum-cut theorem

The maximum-cut theorem

- We define maximum-cut, a cut where

$$\#(\text{cut-denominators}) \equiv \#(\text{components-of-loop-momenta})$$

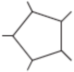
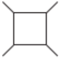
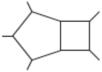
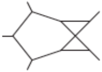
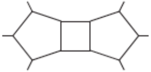
- In non-special kinematic configurations it has a finite number of solutions

$$\#(\text{coefficients-of-the-residue}) = \#(\text{solutions-of-the-cut})$$

- The fit-on-the-cut approach therefore gives a number of equations which is equal to the number of unknown coefficients.

Fit-on-the-cut approach: The maximum-cut theorem

Examples:

diagram	Δ	n_s	diagram	Δ	n_s
	c_0	1		$c_0 + c_1 z$	2
	$\sum_{i=0}^{3} c_i z^i$	4		$\sum_{i=0}^{3} c_i z^{\,i}$	4
	$\sum_{i=0}^{7} c_i z^{\,i}$	8		$\sum_{i=0}^{7} c_i z^{\,i}$	8

Fit-on-the-cut approach

Pros:

- each multiple cut projects out the corresponding residue
  - $\Rightarrow$ the systems of equations for the coefficients are much smaller
- can be implemented either analytically or numerically
- very successful application at one-loop

Cons:

- at higher-loops the solutions of the cuts can be difficult to find
- it cannot be applied in to all integrands/topologies
  - if we have e.g. quadratic propagators the formula yields

$$\frac{\mathcal{N}_{i_1 \cdots i_n} - \text{higher-point contibrutions}}{\prod_{h \neq i_1, \ldots, i_k} D_h} = \frac{0}{0}$$

Fit-on-the-cut approach

Pros:

- each multiple cut projects out the corresponding residue
  - $\Rightarrow$ the systems of equations for the coefficients are much smaller
- can be implemented either analytically or numerically
- very successful application at one-loop

Cons:

- at higher-l ... cult to find
- it cannot b ...
  - if we h ... elds

$$\prod_{h\neq i_1,\dots,i_k} \quad \frac{0}{0}$$

OBSERVATION:
these issues are not present
in the divide-and-conquer approach
which instead can be applied to
any integrand

One-loop decomposition from polynomial division

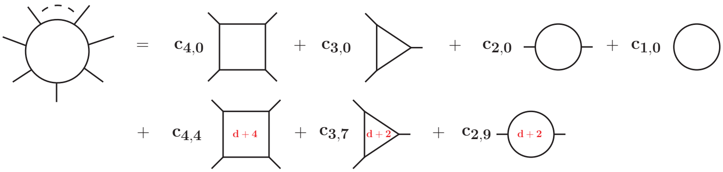
P. Mastrolia, E. Mirabella, G. Ossola, T.P. (2012)

- Start from the most general one-loop amplitude in $d = 4 - 2\epsilon$
- Apply the recursive formula for the integrand decomposition

  $\Rightarrow$ it reproduces the OPP result

  [Ossola, Papadopoulos, Pittau (2007); Ellis, Giele, Kunszt, Melnikov (2008)]
- Drop the spurious terms

$\Rightarrow$ Get the most general integral decomposition (well knwon result)



One-loop decomposition from polynomial division

At one loop in $4-2\epsilon$ dimensions:

- 5 coordinates $\mathbf{z} = (z_1, z_2, z_3, z_4, z_5)$
  - 4 components (z_1, z_2, z_3, z_4) of q w.r.t. a 4-dimensional basis
  - $z_5 = \mu^2$ encodes the (-2ϵ)-dependence on the loop momentum
- we start with

$$\mathcal{I}_n \equiv \mathcal{I}_{1\cdots n} = \frac{\mathcal{N}_{1\cdots n}(\mathbf{z})}{D_1(\mathbf{z})\cdots D_n(\mathbf{z})}$$

most general 1-loop numerator

generic 1-loop denominators

- if $m>5$ any integrand $\mathcal{I}_{i_1\cdots i_m}$ is reducible (reducibility criterion)

$$\mathcal{I}_{i_1\cdots i_m} = \sum_k \mathcal{I}_{i_1\cdots i_{k-1} i_{k+1}\cdots i_m}, \quad \Rightarrow \quad \Delta_{i_1\cdots i_m} = 0 \quad \text{for } m>5$$

- for $m \leq 5$ the polynomial-division algorithm gives the already-known parametric form of the residues $\Delta_{ijk\cdots}$

- Choice of 4-dimensional basis for an m-point residue

$$e_1^2 = e_2^2 = 0\,, \qquad e_1 \cdot e_2 = 1\,, \qquad e_3^2 = e_4^2 = \delta_{m4}\,, \qquad e_3 \cdot e_4 = -(1-\delta_{m4})$$

- Coordinates: $\mathbf{z} = (z_1, z_2, z_3, z_4, z_5) \equiv (x_1, x_2, x_3, x_4, \mu^2)$

$$q^\mu = -p_{i_1}^\mu + x_1\, e_1^\mu + x_2\, e_2^\mu + x_3\, e_3^\mu + x_4\, e_4^\mu\,, \qquad \bar{q}^2 = q^2 - \mu^2$$

- Generic numerator

$$\mathcal{N}_{i_1\cdots i_m} = \sum_{j_1,\ldots,j_5} \alpha_{\vec{j}}\, z_1^{j_1}\, z_2^{j_2}\, z_3^{j_3}\, z_4^{j_4}\, z_5^{j_5}\,, \qquad (j_1 \ldots j_5) \quad \text{such that} \quad \mathrm{rank}(\mathcal{N}_{i_1\cdots i_m}) \leq m$$

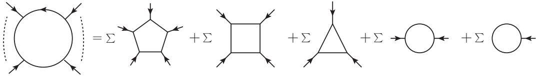
- Residues

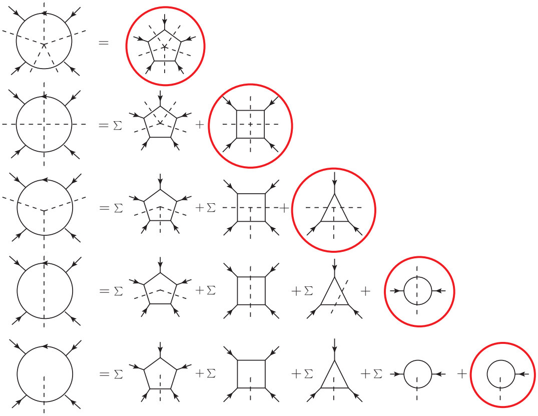
$$\begin{aligned}
\Delta_{i_1 i_2 i_3 i_4 i_5} &= c_0 \\
\Delta_{i_1 i_2 i_3 i_4} &= c_0 + c_1 x_4 + \mu^2 (c_2 + c_3 x_4 + \mu^2 c_4) \\
\Delta_{i_1 i_2 i_3} &= c_0 + c_1 x_3 + c_2 x_3^2 + c_3 x_3^3 + c_4 x_4 + c_5 x_4^2 + c_6 x_4^3 + \mu^2 (c_7 + c_8 x_3 + c_9 x_4) \\
\Delta_{i_1 i_2} &= c_0 + c_1 x_2 + c_2 x_3 + c_3 x_4 + c_4 x_2^2 + c_5 x_3^2 + c_6 x_4^2 + c_7 x_2 x_3 + c_9 x_2 x_4 + c_9 \mu^2 \\
\Delta_{i_1} &= c_0 + c_1 x_1 + c_2 x_2 + c_3 x_3 + c_4 x_4
\end{aligned}$$

- It can be easily extended to higher-rank numerators

Fit-on-the-cut at 1-loop

Integrand decomposition:





Fit-on-the cut

fit m-point residues on m-ple cuts

Integrand reduction via Laurent expansion (NINJA)

The integrand reduction via Laurent expansion:
[P. Mastrolia, E. Mirabella, T.P. (2012)]

- fits residues by taking their asymptotic expansions on the cuts
- yields diagonal systems of equations for the coefficients
- requires the computation of fewer coefficients
- subtractions of higher point residues is simplified
  - implemented as corrections at the coefficient level

Integrand reduction via Laurent expansion (NINJA)

The integrand reduction via Laurent expansion:
[P. Mastrolia, E. Mirabella, T.P. (2012)]

- fits residues by taking their asymptotic expansions on the cuts
- yields diagonal systems of equations for the coefficients
- requires the computation of fewer coefficients
- subtractions of higher point residues is simplified
  - implemented as corrections at the coefficient level

★ Implemented in the semi-numerical C++ library NINJA [T.P. (2014)]

  - Laurent expansions via a simplified polynomial-division algorithm
  - interfaced with the package GOSAM
  - interface with FORMCALC [T. Hahn et al.] under development
  - is a faster and more stable integrand-reduction algorithm

Integrand reduction via Laurent expansion (NINJA)

The integrand reduction via Laurent expansion:
[P. Mastrolia, E. Mirabella, T.P. (2012)]

- fits residues by taking their asymptotic expansions on the cuts
- yields diagonal systems of equations for the coefficients
- requires the computation of fewer coefficients
- subtractions of higher point residues is simplified
  - implemented as corrections at the coefficient level

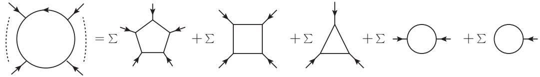
★ Implemented in the semi-numerical C++ library NINJA [T.P. (2014)]

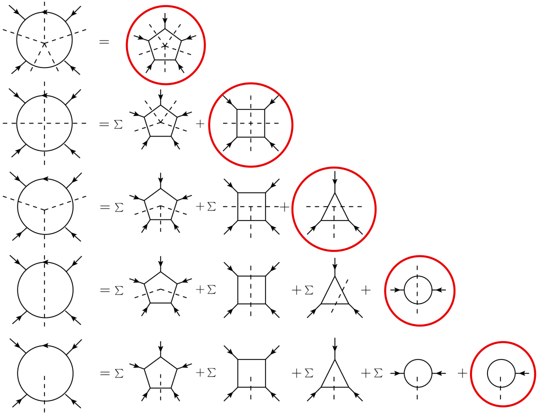
- Laurent expansions via a simplified polynomial-division algorithm
- interfaced with the package GOSAM
- interface with FORMCALC [T. Hahn et al.] under development
- is a faster and more stable integrand-reduction algorithm

★ NINJA is public ⇒ `ninja.hepforge.org`

Integrand reduction via Laurent expansion (NINJA)

Integrand decomposition:

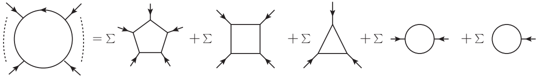


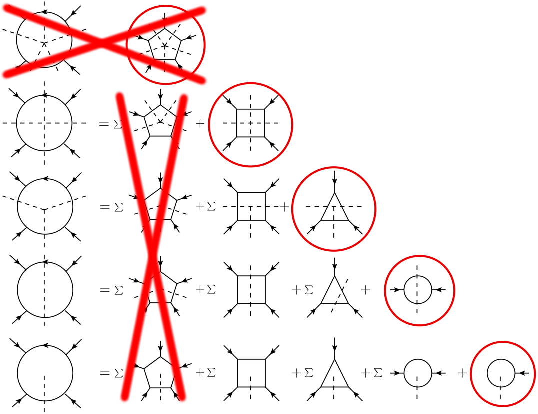


Laurent-expansion method

Integrand reduction via Laurent expansion (NINJA)

Integrand decomposition:



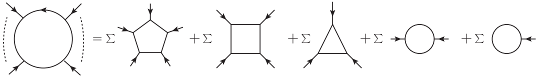


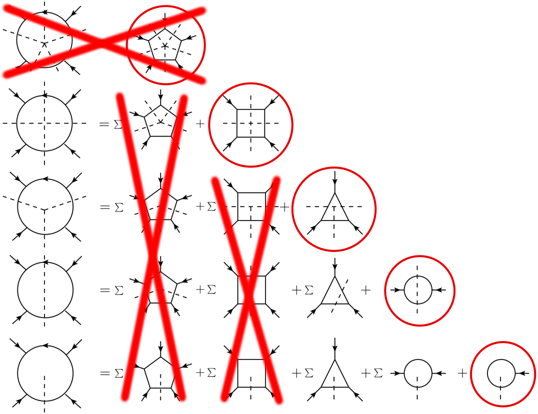
Laurent-expansion method

- pentagons not needed

Integrand reduction via Laurent expansion (NINJA)

Integrand decomposition:



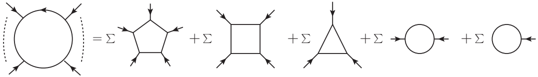


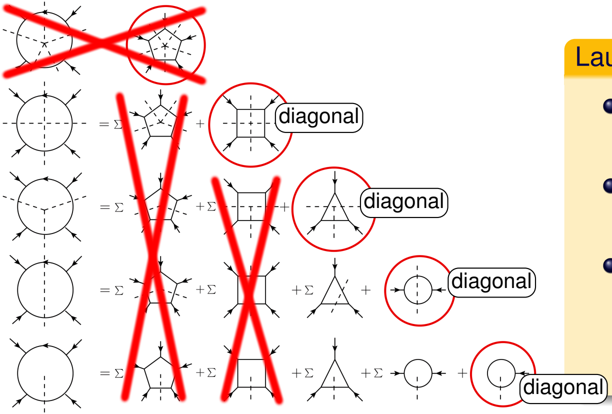
Laurent-expansion method

- pentagons not needed
- boxes never subtracted

Integrand reduction via Laurent expansion (NINJA)

Integrand decomposition:



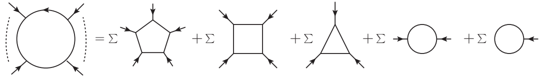


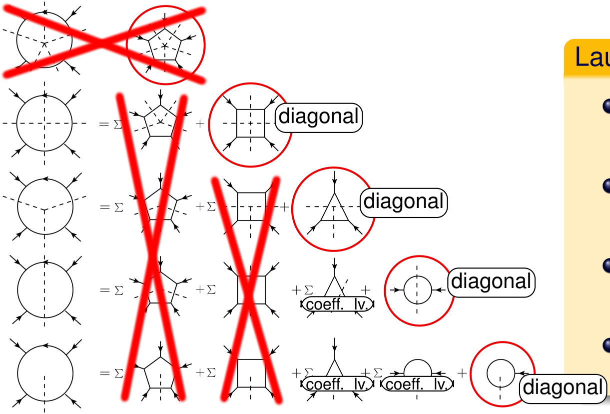
Laurent-expansion method

- pentagons not needed
- boxes never subtracted
- diagonal systems of equations

Integrand reduction via Laurent expansion (NINJA)

Integrand decomposition:





Laurent-expansion method

- pentagons not needed
- boxes never subtracted
- diagonal systems of equations
- subtractions at coefficient level

One-loop boxes via Laurent expansion

- The residue of a box reads

$$\Delta_{ijkl}(q,\mu^2) = d_0 + d_2\mu^2 + d_4\,\mu^4 + (d_1 + d_3\mu^2)(q\cdot v_\perp)$$

- d_0 via 4-dimensional 4ple cuts [Britto, Cachazo, Feng (2004)]
- d_4 from d-dimensional 4-ple cuts in the limit $\mu^2 \to \infty$ [S. Badger (2008)]
  - d-dimensional solutions of a 4-ple cut

$$q_\pm = a^\mu \pm \sqrt{\alpha + \frac{\mu^2}{\beta^2}}\, v_\perp^\mu = \pm\frac{\sqrt{\mu^2}}{\beta}\, v_\perp^\mu + \mathcal{O}(1)$$

  - the integrand in the asymptotic limit $\mu^2 \to \infty$ of the cut-solutions

$$\left.\frac{\mathcal{N}(q,\mu^2)}{\prod_{m\neq i,j,k,l} D_m}\right|_{\text{cut}} = d_4\,\mu^4 + \mathcal{O}(\mu^3)$$

- d_1, d_2, d_3 are spurious and do not need to be computed

One-loop triangles via Laurent expansion

- The residue of a triangle

$$\Delta_{ijk}(q) = c_0 + c_7\,\mu^2 + (c_1 + c_8\mu^2)\,(q\cdot e_3) + c_2\,(q\cdot e_3)^2 + c_3\,(q\cdot e_3)^3 \\ + (c_4 + c_9\mu^2)\,(q\cdot e_4) + c_5\,(q\cdot e_4)^2 + c_6\,(q\cdot e_4)^3$$

- solutions of a triple cut $D_i = D_j = D_k = 0$ parametrized by the free variables t and μ^2

$$q_+^\mu = a^\mu + t\,e_3^\mu + \frac{\alpha+\mu^2}{2\,t}\,e_4^\mu, \qquad q_-^\mu = a^\mu + \frac{\alpha+\mu^2}{2\,t}\,e_3^\mu + t\,e_4^\mu$$

- in the limit $t \to \infty$ [Forde (2007)]

$$\left.\frac{\mathcal{N}(q_\pm)}{\prod_{m\neq i,j,k} D_m}\right|_{\text{cut}} = \Delta_{ijk} + \sum_l \frac{\Delta_{ijkl}}{D_l} + \sum_{lm}\frac{\Delta_{ijklm}}{D_l D_m} \\ = \Delta_{ijk} + d_1^\pm + d_2^\pm\,\mu^2 + \mathcal{O}(1/t)$$

with $d_i^+ + d_i^- = 0$

One-loop triangles via Laurent expansion

- In the asymptotic limit $t \to \infty$

$$\left. \frac{\mathcal{N}(q_\pm)}{\prod_{m \neq i,j,k} D_m} \right|_{\text{cut}} = (d_1^\pm + d_2^\pm\, \mu^2) + \Delta_{ijk} + \mathcal{O}(1/t) \qquad \text{with } d_i^+ + d_i^- = 0$$

  - the integrand

$$\left. \frac{\mathcal{N}(q_\pm)}{\prod_{m \neq i,j,k} D_m} \right|_{\text{cut}} = n_0^\pm + n_4^\pm\, \mu^2 + (n_1^\pm + n_5^\pm\, \mu^2)\, t + n_2^\pm\, t^2 + n_3^\pm\, t^3 + \mathcal{O}(1/t)$$

  - the residue

$$\Delta_{ijk}(q_+) = c_0 + c_7\, \mu^2 - (c_4 + c_9\, \mu^2)\, t + c_5\, t^2 - c_6\, t^3 + \mathcal{O}(1/t)$$
$$\Delta_{ijk}(q_-) = c_0 + c_7\, \mu^2 - (c_1 + c_8\, \mu^2)\, t + c_2\, t^2 - c_3\, t^3 + \mathcal{O}(1/t)$$

  - by comparison we get

$$c_0 = \frac{n_0^+ + n_0^-}{2}, \quad c_1 = -n_1^-, \quad c_2 = n_2^-, \quad c_3 = -n_3^-, \quad \ldots$$

One-loop bubbles via Laurent expansion

- The residue of a bubble

$$\Delta_{ij}(q) = b_0 + b_1\,(q\cdot e_2) + b_2\,(q\cdot e_2)^2 + b_3\,(q\cdot e_3) + b_4\,(q\cdot e_3)^2 + b_5\,(q\cdot e_4) + b_6\,(q\cdot e_4)^2 + b_7\,(q\cdot e_2)(q\cdot e_3) + b_8\,(q\cdot e_2)(q\cdot e_4) + b_9\,\mu^2$$

- solutions of a double cut $D_i = D_j = 0$, parametrized by the free variables t, x and μ^2

$$q_+ = x\,e_1 + (\alpha_0 + x\,\alpha_1)e_2 + t\,e_3 + \frac{\beta_0 + \beta_1 x + \beta_2 x^2 + \mu^2}{2\,t}\,e_4$$
$$q_- = x\,e_1 + (\alpha_0 + x\,\alpha_1)e_2 + \frac{\beta_0 + \beta_1 x + \beta_2 x^2 + \mu^2}{2\,t}\,e_3 + t\,e_4$$

- in the limit $t \to \infty$

$$\left.\frac{\mathcal{N}(q_\pm)}{\prod_{m\neq i,j} D_m}\right|_{\text{cut}} = \Delta_{ij} + \sum_k \frac{\Delta_{ijk}}{D_k} + \sum_{kl} \frac{\Delta_{ijkl}}{D_k D_l} + \sum_{klm} \frac{\Delta_{ijklm}}{D_k D_l D_m}$$
$$= \Delta_{ij} + \sum_k \frac{\Delta_{ijk}}{D_k} + \mathcal{O}(1/t)$$

One-loop bubbles via Laurent expansion

- In the asymptotic limit $t \to \infty$
  - the integrand

$$\left.\frac{\mathcal{N}(q_\pm)}{\prod_{m\neq i,j,k} D_m}\right|_{\text{cut}} = n_0^\pm + n_6^\pm\,\mu^2 + n_1^\pm\,x + n_2^\pm\,x^2 + \left(n_3^\pm + n_4^\pm x\right)t + n_5^\pm\,t^2 + \mathcal{O}(1/t)$$

  - the subtraction term

$$\frac{\Delta_{ijk}(q_\pm)}{D_k} = \tilde{b}_0^{k,\pm} + \tilde{b}_6^{k,\pm}\,\mu^2 + \tilde{b}_1^{k,\pm}\,x + \tilde{b}_2^{k,\pm}\,x^2 + \left(\tilde{b}_3^{k,\pm} + \tilde{b}_4^{k,\pm}x\right)t + \tilde{b}_5^{k,\pm}\,t^2 + \mathcal{O}(1/t)$$

    - $\tilde{b}_i^{k,\pm}$ are known functions of the triangle coefficients
  - the residue

$$\Delta_{ij}(q_+) = b_0 + b_9\,\mu^2 + b_1\,x + b_2 x^2 - \left(b_5 + b_8 x\right)t + b_6\,t^2 + \mathcal{O}(1/t)$$

$$\Delta_{ij}(q_-) = b_0 + b_9\,\mu^2 + b_1\,x + b_2 x^2 - \left(b_3 + b_7 x\right)t + b_4\,t^2 + \mathcal{O}(1/t)$$

  - by comparison, applying subtractions at the coefficient level

$$b_0 = n_0^\pm - \sum_k \tilde{b}_0^{k,\pm}, \quad b_1 = n_1^\pm - \sum_k \tilde{b}_1^{k,\pm}, \quad b_3 = -n_3^- + \sum_k \tilde{b}_3^{k,-}, \quad \ldots$$

Semi-numerical implementation in NINJA

- The input is the numerator $\mathcal{N}$ cast in (three or) four different forms
  - leading terms of parametric expansions of the numerator
  - coefficients of the expansion written to an array $\mathcal{N}$ `[]`
  - all easily obtained from its analytic expression
- The PYTHON script NINJANUMGEN uses FORM-4 to
  - automatically compute expansions from a FORM expression of $\mathcal{N}$
  - generate optimized source code needed as input for NINJA
- NINJA at run-time
  - computes parametric on-shell solutions
  - performs Laurent expansions via pol. div.
  - implements subtractions at coefficient level
  - multiplies the obtained coefficients with the MI's
- Semi-numeric Laurent expansion via polynomial division
  - expansion of numerator $\mathcal{N}$ `[]` / denominators D_i

Semi-numerical implementation in NINJA

```
// Numerator: can be generated using the script ninjanumgen
class MyNumerator : public ninja::Numerator {
public:

  // evaluates the numerator N(q, μ^2) - same as Samurai
  virtual Complex evaluate(q, μ^2, ...);

  // (optional) expansion for 4-ple cut rational term q^μ → t v_⊥^μ + O(1)
  virtual void muExpansion(v_⊥, ..., Complex N);

  // expansion for triangles and tadpoles q^μ → v_0^μ + t v_3^μ + (β + μ^2)/(2 t) v_4^μ
  virtual void t3Expansion(v_0, v_3, v_4, β, ..., Complex N[]);

  // expansion for bubbles q^μ → v_1^μ + x v_2^μ + t v_3^μ + (β_0 + β_1 x + β_2 x^2 + μ^2)/(2 t) v_4^μ
  virtual void t2Expansion(v_1, v_2, v_3, v_4, β_i, ..., Complex N[]);
};
```

—

note: `t2Expansion` is `t3Expansion` with: $v_0 \to v_1^\mu + x\, v_2^\mu,\ \beta \to \beta_0 + \beta_1 x + \beta_2 x^2$

Semi-numerical implementation in NINJA

Master Integrals:

- are called via a generic interface
  - ⇒ any user-defined library of Master Integrals can be used
- the library of MI's to be used can be specified at run time
- NINJA provides the interface for two default libraries
  - ONELOOP library [A. van Hameren] wrapper + caching
    - computed MI's are cached by NINJA
    - constant-time lookup from their arguments
  - LOOPTOOLS library [T. Hahn]
    - an internal cache is already present ⇒ interface is a simple wrapper

Higher-rank:

- support for higher-rank $r = n + 1$
- higher-rank MI's (can but) do not need to be provided

Automation of one-loop computation

In several one-loop packages we can distinguish three phases:

1. Generation
   - generate the integrand
   - cast it in a suitable form for reduction
   - write it in a piece of source code (e.g. FORTRAN or C/C++)
2. Compilation
   - compile the code
3. Run-time
   - use a reduction library in order to compute the integrals

Automation of one-loop computation in GOSAM

GOSAM is a PYTHON package which:

- generates analytic integrands
  - using QGRAF [P. Nogueira] and FORM [J. Vermaseren et al.]
- writes them into FORTRAN90 code
- can use different reduction algorithms at run-time
  - SAMURAI (d-dim. integrand reduction)
    - faster than GOLEM95 but numerically less stable
    - former default in GOSAM-1.0
  - GOLEM95 (tensor reduction)
    - slower than SAMURAI but more stable
    - default rescue-system for unstable points
  - NINJA
    - fast (2 to 5 times faster than SAMURAI)
    - stable (in worst cases $\mathcal{O}(1/1000)$ unstable points)
    - current default in GOSAM-2.0 $\leftarrow$ just released

Benchmarks of GOSAM + NINJA

H. van Deurzen, G. Luisoni, P. Mastrolia, E. Mirabella, G. Ossola and T.P. (2013)

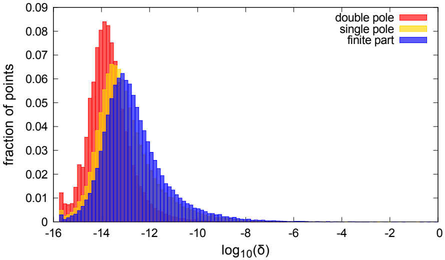
Benchmarks: GOSAM + NINJA			
Process		# NLO diagrams	ms/event[a]
$W+3j$	$d\bar{u} \to \bar{\nu}_e e^- ggg$	1 411	226
$Z+3j$	$d\bar{d} \to e^+ e^- ggg$	2 928	1 911
$t\bar{t}b\bar{b}\ (m_b \neq 0)$	$d\bar{d} \to t\bar{t}b\bar{b}$	275	178
	$gg \to t\bar{t}b\bar{b}$	1 530	5 685
$t\bar{t}+2j$	$gg \to t\bar{t}gg$	4 700	13 827
$W\,b\,\bar{b}+1j\,(m_b \neq 0)$	$u\bar{d} \to e^+ \nu_e b\bar{b}g$	312	67
$W\,b\,\bar{b}+2j\,(m_b \neq 0)$	$u\bar{d} \to e^+ \nu_e b\bar{b}s\bar{s}$	648	181
	$u\bar{d} \to e^+ \nu_e b\bar{b}d\bar{d}$	1 220	895
	$u\bar{d} \to e^+ \nu_e b\bar{b}gg$	3 923	5 387
$H+3j$ in GF	$gg \to Hggg$	9 325	8 961
$t\bar{t}H+1j$	$gg \to t\bar{t}Hg$	1 517	1 505
$H+3j$ in VBF	$u\bar{u} \to Hgu\bar{u}$	432	101
$H+4j$ in VBF	$u\bar{u} \to Hggu\bar{u}$	1 176	669
$H+5j$ in VBF	$u\bar{u} \to Hgggu\bar{u}$	15 036	29 200

more processes in arXiv:1312.6678

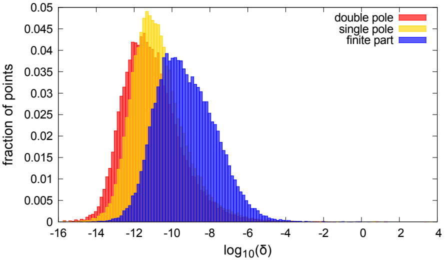
[a] Timings refer to full color- and helicity-summed amplitudes, using an Intel Core i7 CPU @ 3.40GHz, compiled with `ifort`.

Stability of NINJA

- $H + 4j$ in VBF ($u\bar{u} \to Hggu\bar{u}$)



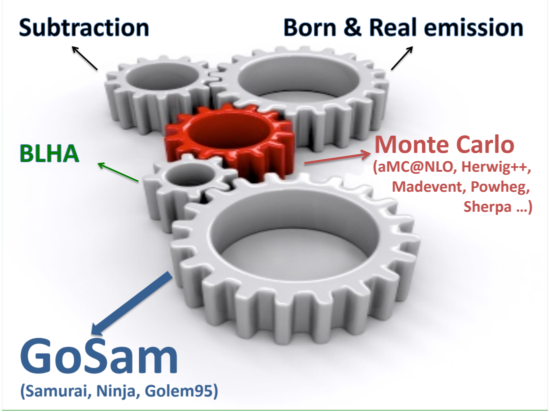
- $t\,\bar{t}\,H + 1\,j$ ($gg \to t\bar{t}Hg$)



Rate of unstable points, i.e. with error $\delta > \delta_{\text{threshold}}$ on the finite part:

$\delta_{\text{threshold}}$	$u\bar{u} \to Hggu\bar{u}$	$gg \to t\bar{t}Hg$
10^{-3}	0.02%	0.06%
10^{-4}	0.04%	0.16%
10^{-5}	0.08%	0.56%

From amplitudes to observables with GOSAM



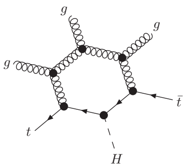
The GOSAM collaboration:

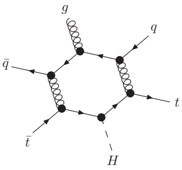
G. Cullen, H. van Deurzen, N. Greiner, G. Heinrich, G. Luisoni, P. Mastrolia, E. Mirabella, G. Ossola, J. Reichel , J. Schlenk, J. F. von Soden-Fraunhofen, T. Reiter, F. Tramontano, T.P.

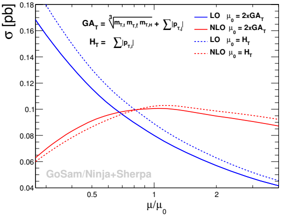
Application: $pp \to t\bar{t}H + jet$ with GOSAM + NINJA

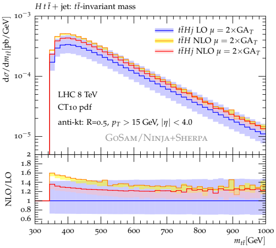
H. van Deurzen, G. Luisoni, P. Mastrolia, E. Mirabella, G. Ossola, T.P. (2013)

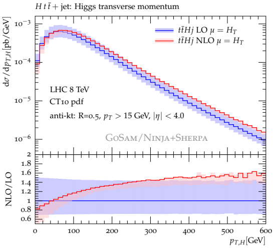
- Interfaced with the Monte Carlo SHERPA

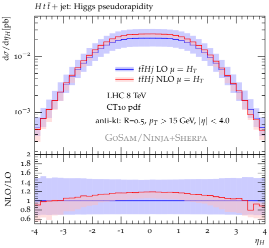






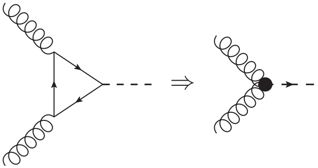






Application: $pp \to H + jets$ in GF with GOSAM + NINJA

- $m_t \to \infty$ approximation



- effective couplings $H + (2,3,4)gl.$
- higher-rank integrands ⇒ extension of int. red. methods
  [P. Mastrolia, E. Mirabella,T.P.(2012), H. van Deurzen (2013)]

- $H + 2j$ (GOSAM+SAMURAI+SHERPA)
  [H. van Deurzen, N. Greiner, G. Luisoni, P. Mastrolia, E. Mirabella, G. Ossola, J. F. von Soden-Fraunhofen, F. Tramontano, T.P.(2013)]
- $H + 3j$ (GOSAM+SAMURAI+SHERPA+MADGRAPH4/MADEVENT)
  [G. Cullen, H. van Deurzen, N. Greiner, G. Luisoni, P. Mastrolia, E. Mirabella, G. Ossola, F. Tramontano, T.P.(2013)]
- new analysis with ATLAS-like cuts, using NINJA for the reduction
  [G. Cullen, H. van Deurzen, N. Greiner, J. Huston, G. Luisoni, P. Mastrolia, E. Mirabella, G. Ossola, F. Tramontano, J. Winter, V. Yundin, T.P. (preliminary, 2014)]

Application: $pp \to H + jets$ in GF with GOSAM + NINJA

- new distributions using NINJA (preliminary)
  - better accuracy
  - better performance

$$\mu_F = \mu_R = \frac{\hat{H}_T}{2} = \frac{1}{2}\left(\sqrt{m_H^2 + p_{t,H}^2} + \sum_{jets} |p_{t,jet}|^2\right)$$

- ATLAS-like cuts

$$R = 0.4, \qquad p_{t,jet} > 30\text{GeV}, \qquad |\eta_{jet}| < 4.4$$

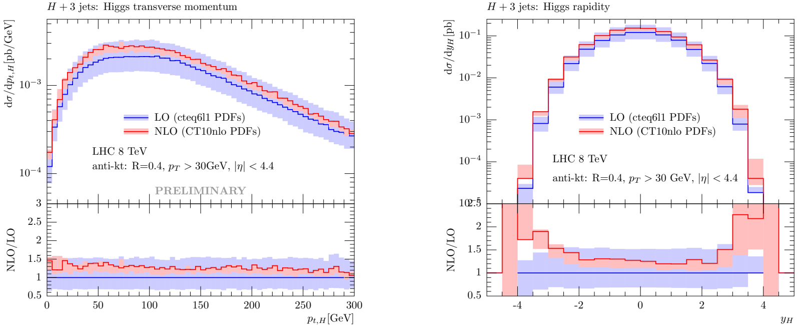
- total cross section

$$\sigma_{LO}^{(H+2j)}([\text{pb}]) = 1.23^{+37\%}_{-24\%}, \qquad \sigma_{LO}^{(H+3j)}([\text{pb}]) = 0.381^{+53\%}_{-32\%}$$
$$\sigma_{NLO}^{(H+2j)}([\text{pb}]) = 1.590^{-4\%}_{-7\%}, \qquad \sigma_{NLO}^{(H+3j)}([\text{pb}]) = 0.485^{-3\%}_{-13\%}$$

Application: $pp \to H + jets$ in GF with GOSAM + NINJA

- new distributions using NINJA (preliminary)
  - better accuracy
  - better performance

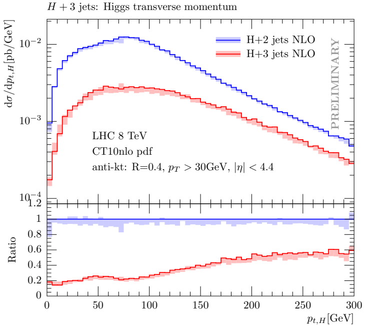
$$\mu_F = \mu_R = \frac{\hat{H}_T}{2} = \frac{1}{2}\left(\sqrt{m_H^2 + p_{t,H}^2} + \sum_{jets} |p_{t,jet}|^2\right)$$

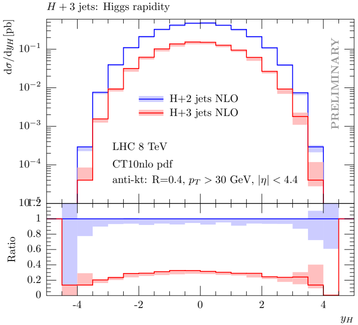


Application: $pp \to H + jets$ in GF with GOSAM + NINJA

- new distributions using NINJA (preliminary)
  - better accuracy
  - better performance

$$\mu_F = \mu_R = \frac{\hat{H}_T}{2} = \frac{1}{2}\left(\sqrt{m_H^2 + p_{t,H}^2} + \sum_{jets} |p_{t,jet}|^2\right)$$





Extension to higher loops

- The integrand-level approach to scattering amplitudes at one-loop
  - can be used to compute any amplitude in any QFT
  - has been implemented in several codes, some of which public [SAMURAI, CUTTOOLS, NINJA]
  - has produced (and is still producing) results for LHC [GOSAM, FORMCALC, BLACKHAT, MADLOOP, NJETS, OPENLOOP . . .]
- At two or higher loops
  - no general recipe is available
  - the standard and most successful approach is the Integration By Parts (IBP) method, but it becomes difficult for high multiplicities

Extension to higher loops

- The integrand-level approach to scattering amplitudes at one-loop
  - can be used to compute any amplitude in any QFT
  - has been implemented in several codes, some of which public [SAMURAI, CUTTOOLS, NINJA]
  - has produced (and is still producing) results for LHC [GOSAM, FORMCALC, BLACKHAT, MADLOOP, NJETS, OPENLOOP ...]
- At two or higher loops
  - no general recipe is available
  - the standard and most successful approach is the Integration By Parts (IBP) method, but it becomes difficult for high multiplicities

The integrand-level approach might be a tool for understanding the structure of multi-loop scattering amplitudes and a method for their evaluation.

Extension to higher loops

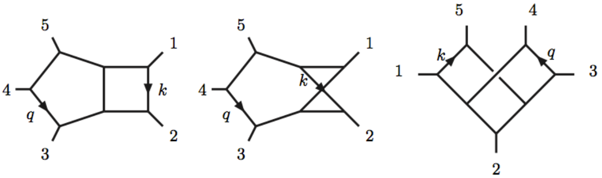
- The integrand-level approach to scattering amplitudes at one-loop
  - can be used to compute any amplitude in any QFT
  - has been implemented in several codes, some of which public [SAMURAI, CUTTOOLS, NINJA]
  - has produced (and is still producing) results for LHC [GOSAM, FORMCALC, BLACKHAT, MADLOOP, NJETS, OPENLOOP . . .]
- At two or higher loops
  - no general recipe is available
  - the standard and most successful approach is the Integration By Parts (IBP) method, but it becomes difficult for high multiplicities

The integrand-level approach might be a tool for understanding the structure of multi-loop scattering amplitudes and a method for their evaluation.

- . . . we are moving the first steps in this direction

$\mathcal{N} = 4$ SYM and $\mathcal{N} = 8$ SUGRA amplitudes

P. Mastrolia, G. Ossola (2011); P. Mastrolia, E. Mirabella, G. Ossola, T.P. (2012)



- Examples in $\mathcal{N} = 4$ SYM and $\mathcal{N} = 8$ SUGRA amplitudes ($d = 4$)
  - generation of the integrand
    - graph based [Carrasco, Johansson (2011)]
    - unitarity based [U. Schubert (Diplomarbeit)]
  - fit-on-the-cut approach for the reduction
- Results:

$\mathcal{N} = 4$ linear combination of 8 and 7-denominators MIs

$\mathcal{N} = 8$ linear combination of 8, 7 and 6-denominators MIs

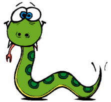
Divide-and-Conquer approach

P. Mastrolia, E. Mirabella, G. Ossola, T.P. (2013)

The divide-and-conquer approach to the integrand reduction

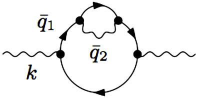
- does not require the knowledge of the solutions of the cut
- can always be used to perform the reduction in a finite number of purely algebraic operations
- has been automated in a PYTHON package which uses MACAULAY2 and FORM for algebraic operations

MACAULAY2 $\Leftarrow$ PYTHON $\Rightarrow$ FORM



- also works in special cases where the fit-on-the-cut approach is not applicable (e.g. in presence of double denominators)

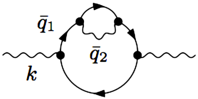
Divide-and-Conquer approach: a simple example



$$\mathcal{I}_{11234} = \frac{\mathcal{N}_{11234}}{D_1^2 D_2 D_3 D_4}$$

$$\begin{aligned} D_1 &= \bar{q}_1^2 - m^2 , \\ D_2 &= (\bar{q}_1 - k)^2 - m^2 , \\ D_3 &= \bar{q}_2^2 , \\ D_4 &= (\bar{q}_1 + \bar{q}_2)^2 - m^2 \end{aligned}$$

Divide-and-Conquer approach: a simple example



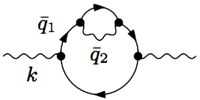
$$\mathcal{I}_{11234} = \frac{\mathcal{N}_{11234}}{D_1^2 D_2 D_3 D_4}$$

$$\begin{aligned} D_1 &= \bar{q}_1^2 - m^2\,, \\ D_2 &= (\bar{q}_1 - k)^2 - m^2\,, \\ D_3 &= \bar{q}_2^2\,, \\ D_4 &= (\bar{q}_1 + \bar{q}_2)^2 - m^2 \end{aligned}$$

- Basis $\{e_i\} \equiv \{k, k_\perp, e_3, e_4\}$ and coordinates $\mathbf{z} = (x_1, x_2, x_3, x_4, y_1, y_2, y_3, y_4, \mu_{11}, \mu_{12}, \mu_{22})$

$$q_1 = \sum_i x_i\, e_i, \qquad q_2 = \sum_i y_i\, e_i, \qquad (\bar{q}_i \cdot \bar{q}_j) = (q_i \cdot q_j) - \mu_{ij}$$

Divide-and-Conquer approach: a simple example



$$\mathcal{I}_{11234} = \frac{\mathcal{N}_{11234}}{D_1^2 D_2 D_3 D_4}$$

$$\begin{aligned} D_1 &= \bar{q}_1^2 - m^2\,, \\ D_2 &= (\bar{q}_1 - k)^2 - m^2\,, \\ D_3 &= \bar{q}_2^2\,, \\ D_4 &= (\bar{q}_1 + \bar{q}_2)^2 - m^2 \end{aligned}$$

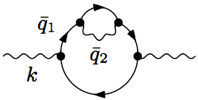
- Basis $\{e_i\} \equiv \{k, k_\perp, e_3, e_4\}$ and coordinates $\mathbf{z} = (x_1, x_2, x_3, x_4, y_1, y_2, y_3, y_4, \mu_{11}, \mu_{12}, \mu_{22})$

$$q_1 = \sum_i x_i\, e_i, \qquad q_2 = \sum_i y_i\, e_i, \qquad (\bar{q}_i \cdot \bar{q}_j) = (q_i \cdot q_j) - \mu_{ij}$$

- division of $\mathcal{N}_{11234}$ modulo $\mathcal{G}_{\mathcal{J}_{11234}} (= \mathcal{G}_{\mathcal{J}_{1234}})$

$$\mathcal{N}_{11234} = \underbrace{\mathcal{N}_{1234} D_1 + \mathcal{N}_{1134} D_2 + \mathcal{N}_{1124} D_3 + \mathcal{N}_{1123} D_4}_{\text{quotients}} + \underbrace{\Delta_{11234}}_{\text{remainder}}$$

Divide-and-Conquer approach: a simple example



$$\mathcal{I}_{11234} = \frac{\mathcal{N}_{11234}}{D_1^2 D_2 D_3 D_4} \qquad \begin{aligned} D_1 &= \bar{q}_1^2 - m^2\,, \\ D_2 &= (\bar{q}_1 - k)^2 - m^2\,, \\ D_3 &= \bar{q}_2^2\,, \\ D_4 &= (\bar{q}_1 + \bar{q}_2)^2 - m^2 \end{aligned}$$

- Basis $\{e_i\} \equiv \{k, k_\perp, e_3, e_4\}$ and coordinates $\mathbf{z} = (x_1, x_2, x_3, x_4, y_1, y_2, y_3, y_4, \mu_{11}, \mu_{12}, \mu_{22})$

$$q_1 = \sum_i x_i\, e_i, \qquad q_2 = \sum_i y_i\, e_i, \qquad (\bar{q}_i \cdot \bar{q}_j) = (q_i \cdot q_j) - \mu_{ij}$$

- division of $\mathcal{N}_{11234}$ modulo $\mathcal{G}_{\mathcal{J}_{11234}} (= \mathcal{G}_{\mathcal{J}_{1234}})$

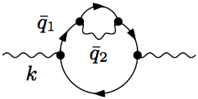
$$\mathcal{N}_{11234} = \underbrace{\mathcal{N}_{1234} D_1 + \mathcal{N}_{1134} D_2 + \mathcal{N}_{1124} D_3 + \mathcal{N}_{1123} D_4}_{\text{quotients}} + \underbrace{\Delta_{11234}}_{\text{remainder}}$$

- division of $\mathcal{N}_{i_1 i_2 i_3 i_4}$ modulo $\mathcal{G}_{\mathcal{J}_{i_1 i_2 i_3 i_4}}$, e.g.

$$\mathcal{N}_{1234}/\mathcal{G}_{\mathcal{J}_{1234}} \quad \Rightarrow \quad \mathcal{N}_{1234} = \underbrace{\mathcal{Q}_{234}^{(1234)} D_1 + \mathcal{Q}_{134}^{(1234)} D_2 + \mathcal{Q}_{124}^{(1234)} D_3 + \mathcal{Q}_{123}^{(1234)} D_4}_{\text{quotients}} + \underbrace{\Delta_{1234}}_{\text{remainder}}$$

$$\mathcal{N}_{1134}/\mathcal{G}_{\mathcal{J}_{1134}} \quad \Rightarrow \quad \mathcal{N}_{1134} = \underbrace{\mathcal{Q}_{134}^{(1134)} D_1 + \mathcal{Q}_{114}^{(1134)} D_3 + \mathcal{Q}_{113}^{(1134)} D_4}_{\text{quotients}} + \underbrace{\Delta_{1134}}_{\text{remainder}}$$

Divide-and-Conquer approach: a simple example



$$\mathcal{I}_{11234} = \frac{\mathcal{N}_{11234}}{D_1^2 D_2 D_3 D_4} \qquad \begin{aligned} D_1 &= \bar{q}_1^2 - m^2\,, \\ D_2 &= (\bar{q}_1 - k)^2 - m^2\,, \\ D_3 &= \bar{q}_2^2\,, \\ D_4 &= (\bar{q}_1 + \bar{q}_2)^2 - m^2 \end{aligned}$$

- Basis $\{e_i\} \equiv \{k, k_\perp, e_3, e_4\}$ and coordinates $\mathbf{z} = (x_1, x_2, x_3, x_4, y_1, y_2, y_3, y_4, \mu_{11}, \mu_{12}, \mu_{22})$

$$q_1 = \sum_i x_i\, e_i, \qquad q_2 = \sum_i y_i\, e_i, \qquad (\bar{q}_i \cdot \bar{q}_j) = (q_i \cdot q_j) - \mu_{ij}$$

- division of $\mathcal{N}_{11234}$ modulo $\mathcal{G}_{\mathcal{J}_{11234}} (= \mathcal{G}_{\mathcal{J}_{1234}})$

$$\mathcal{N}_{11234} = \underbrace{\mathcal{N}_{1234} D_1 + \mathcal{N}_{1134} D_2 + \mathcal{N}_{1124} D_3 + \mathcal{N}_{1123} D_4}_{\text{quotients}} + \underbrace{\Delta_{11234}}_{\text{remainder}}$$

- division of $\mathcal{N}_{i_1 i_2 i_3 i_4}$ modulo $\mathcal{G}_{\mathcal{J}_{i_1 i_2 i_3 i_4}}$

$$\mathcal{N}_{11234} = \underbrace{\mathcal{N}_{234} D_1^2 + \mathcal{N}_{134} D_1 D_2 + \mathcal{N}_{124} D_1 D_3 + \mathcal{N}_{123} D_1 D_4 + \mathcal{N}_{114} D_2 D_3 + \mathcal{N}_{113} D_2 D_4}_{\text{(sums of) quotients}}$$
$$+ \underbrace{\Delta_{1234} D_1 + \Delta_{1134} D_2 + \Delta_{1124} D_3 + \Delta_{1123} D_4}_{\text{remainders}} + \Delta_{11234}$$

Divide-and-Conquer approach: a simple example

- after a further step (division $\mathcal{N}_{i_1 i_2 i_3}/\mathcal{G}_{\mathcal{J}_{i_1 i_2 i_3}}$) no quotient remains

$$\mathcal{N}_{11234} = \Delta_{11234} + \Delta_{1234} D_1 + \Delta_{1134} D_2 + \Delta_{1124} D_3 + \Delta_{1123} D_4 + \Delta_{234} D_1^2 + \Delta_{114} D_2 D_3 + \Delta_{113} D_2 D_4$$

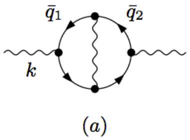
- the integrand decomposition becomes

$$\begin{aligned}
\mathcal{I}_{11234} = \frac{\mathcal{N}_{11234}}{D_1^2 D_2 D_3 D_4} &= \frac{\Delta_{11234}}{D_1^2 D_2 D_3 D_4} + \frac{\Delta_{1234}}{D_1 D_2 D_3 D_4} + \frac{\Delta_{1134}}{D_1^2 D_3 D_4} + \frac{\Delta_{1124}}{D_1^2 D_2 D_4} \\
&+ \frac{\Delta_{1123}}{D_1^2 D_2 D_3} + \frac{\Delta_{234}}{D_2 D_3 D_4} + \frac{\Delta_{114}}{D_1^2 D_4} + \frac{\Delta_{113}}{D_1^2 D_3}
\end{aligned}$$

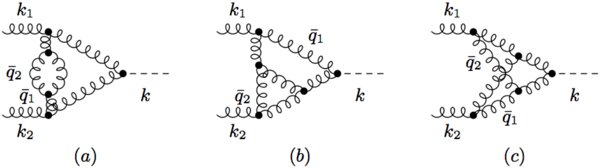
$$\begin{aligned}
\Delta_{11234} &= 16 m^2 \left(k^2 + 2\, m^2 - k^2 \epsilon\right) , \\
\Delta_{1234} &= 16 \left[(q_2 \cdot k)(1-\epsilon)^2 + m^2\right] , \\
\Delta_{1124} &= -\Delta_{1123} = 8\,(1-\epsilon) \left[k^2 (1-\epsilon) + 2\, m^2\right] , \\
\Delta_{1134} &= -16 m^2 \ (1-\epsilon) , \\
\Delta_{113} &= -\Delta_{114} = \Delta_{234} = 8\ (1-\epsilon)^2 .
\end{aligned}$$

Examples of divide-and-conquer approach

- Photon self-energy in massive QED, $(4-2\epsilon)$-dimensions



- Diagrams entering $gg \to H$, in $(4-2\epsilon)$-dimensions



Additional relations between integrals

P. Mastrolia, G. Ossola, T.P. (work in progress)

- The integrals given by the integrand reduction can be further reduced with additional identities
  - traditional approach: Integration by Part (IBP)

$$\int \frac{\partial}{\partial \bar{q}_i^\mu} \frac{\mathcal{N}(\bar{q}_i)^\mu}{D_{i_1} \cdots D_{i_n}} = 0$$

- A 2-step strategy
  1. use integrand reduction first
     $\Rightarrow$ integrals with higher multiplicity should be reduced
  2. then apply IBP
     $\Rightarrow$ could be easier after integrand reduction
- Can we instead see IBPs from Integrand Reduction?
  - Can we recover IBPs from int. red. relations computed in step 1?

Dimensionally shifted integrals

One-loop case:

- with $v_\perp^\mu = \epsilon^\mu{}_{\mu_1 \cdots \mu_{n-1}} k_1^{\mu_1} \cdots k_{n-1}^{\mu_{n-1}}$, we can prove

$$\mathcal{I}_{1\ldots n}[\mu^2] = -\epsilon\, \mathcal{I}_{1\cdots n}^{(d+2)},$$

$$\mathcal{I}_{1\ldots n}[(q \cdot v_\perp)^2] = \mathcal{I}_{1\ldots n}[\epsilon(q, k_1, \ldots, k_{n-1})^2] = -\frac{v_\perp^2}{2} \mathcal{I}_{1\cdots n}^{(d+2)}$$

- perform integrand reduction of $\mathcal{I}_{1\ldots n}[(q \cdot v_\perp)^2]$ from $n = 1$ to higher-points
    - we can reuse the same pol. divisions of integrand reduction
- if $\mathcal{I}_{1\ldots n}[(q \cdot v_\perp)^2]$ reducible at integrand level $\Rightarrow$ then $\mathcal{I}_{1\ldots n}$ reducible at integral level
    - we get an homogeneous equation with integrals in $d + 2$
    - after $d \to d - 2$ we get an IBP relation

Example: one-loop tadpole

Tadpoles: $\mathcal{N} = \epsilon(q)^2 = q^2$:

$$\mathcal{I}_0 = \frac{q^2}{D_0}, \qquad D_0 = \bar{q}^2 - m^2$$

- No external vectors $\Rightarrow$ use special case

$$\mathcal{I}_0[q^2] = -2\,\mathcal{I}_0^{(d+2)}$$

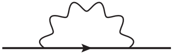
- After integrand reduction

$$\begin{aligned} -2\,\mathcal{I}_0^{(d+2)} &= \mathcal{I}_0[q^2] = \mathcal{I}_0[\mu^2] + m^2\,\mathcal{I}_0 \\ &= \frac{d-4}{2}\,\mathcal{I}_0^{(d+2)} + m^2\,\mathcal{I}_0 \end{aligned}$$

Dimensional shift for tadpoles

$$d\,\mathcal{I}_0^{(d+2)} = 2\,m^2\,\mathcal{I}_0$$

Example: one-loop bubble



$$\mathcal{I}_{01}[(q\cdot v_\perp)^2] = \frac{\mathcal{N}}{D_0 D_1}$$

$$\mathcal{N} = \epsilon(q,k)^2 = q^2\,k^2 - (q\cdot k)^2$$
$$D_0 = \bar{q}^2$$
$$D_1 = \bar{q}^2 + 2(q\cdot k)$$

- Integrand reduction

$$\mathcal{N} = m^2\,\mu^2 + D_0\,\left(\frac{1}{2}m^2 + \frac{1}{2}((q+k)\cdot k)\right) + D_1\,\left(-\frac{1}{2}(q\cdot k)\right)$$

$$-\frac{3m^2}{2}\,\mathcal{I}_{01}^{(d+2)} = \mathcal{I}_{01}[\mathcal{N}] = m^2\mathcal{I}_{01}[\mu^2] + \frac{m^2}{2}\,\mathcal{I}_1 = \frac{d-4}{2}\,m^2\,\mathcal{I}_{01}^{(d+2)} - \frac{d}{4}\mathcal{I}_1^{(d+2)}$$

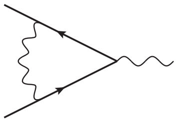
- The result in $d+2$ dimensions

$$(d-1)\,\mathcal{I}_{01}^{(d+2)} = \frac{1}{2\,m^2}\,d\,\mathcal{I}_1^{(d+2)}$$

Result in d dimensions

$$(d-3)\,\mathcal{I}_{01} = \frac{1}{2\,m^2}\,(d-2)\,\mathcal{I}_1$$

Example: one-loop triangle



$$\mathcal{I}_{012}[\mathcal{N}] = \frac{\mathcal{N}}{D_0 D_1 D_2}$$

$$D_0 = \bar{q}^2$$
$$D_1 = \bar{q}^2 + 2(q \cdot k_1)$$
$$D_2 = \bar{q}^2 - 2(q \cdot k_2)$$

- With a similar procedure, from $\mathcal{I}_{012}[\epsilon(q,k_1,k_2)^2]$ and $\mathcal{I}_{12}[\epsilon(q,k_1+k_2)^2]$, we get

$$(2-d)\,\mathcal{I}_{012}^{(d+2)} = \mathcal{I}_{12}$$
$$(1-d)\,\mathcal{I}_{12}^{(d+2)} = \frac{4m^2-s}{2}\mathcal{I}_{12} + \mathcal{I}_1$$

- The result in $d+2$ dimensions

$$(2-d)\,\mathcal{I}_{012}^{(d+2)} = \frac{2}{4m^2-s}\left((1-d)\,\mathcal{I}_{12}^{(d+2)} + \frac{d}{2\,m^2}\,\mathcal{I}_1^{(d+2)}\right)$$

Result in d dimensions

$$(4-d)\,\mathcal{I}_{012} = \frac{2}{4m^2-s}\left((3-d)\,\mathcal{I}_{12} + \frac{d-2}{2\,m^2}\,\mathcal{I}_1\right)$$

Higher loops

- At one-loop we used

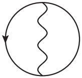
$$\mathcal{I}[\epsilon(q, k_1, \ldots, k_{n-1})^2], \qquad \mathcal{I}[\mu^2]$$

- At higher loops we should use

$$\mathcal{I}[\epsilon(q_1, \ldots, q_\ell, k_1, \ldots, k_{n-1})^2], \qquad \mathcal{I}[\epsilon(\vec{\mu}_1, \ldots, \vec{\mu}_\ell)].$$

- Relations for integrals in μ^2 can be easily found at any loop using Schwinger parametrization

Example: two-loop



$$\mathcal{I}_{123}[\mathcal{N}] = \frac{\mathcal{N}}{D_1 D_2 D_3}$$

$$\begin{aligned} D_1 &= \bar{q}_1^2 - m^2 = q_1^2 - m^2 - \mu_{11} \\ D_2 &= \bar{q}_2^2 - m^2 = q_2^2 - m^2 - \mu_{22} \\ D_3 &= (\bar{q}_1 - \bar{q}_2)^2 = (q_1 - q_2)^2 - \mu_{11} - \mu_{22} + 2\,\mu_{12} \end{aligned}$$

- The integrand reduction gives

$$\begin{aligned} -3\mathcal{I}_{123}^{(d+2)} &= \mathcal{I}_{123}[\epsilon(q_1, q_2)^2] = \mathcal{I}_{123}[(q_1 \cdot q_2)^2 - q_1^2 q_2^2] \\ &= \frac{1}{4}\mathcal{I}_{123}[4\mu_{12}^2 - 4\mu_{11}\mu_{22}] + m^2 \mathcal{I}_{123}[2\mu_{12} - \mu_{11} - \mu_{22}] - \frac{m^2}{2}\,\mathcal{I}_{12}. \end{aligned}$$

- Integrals in μ_{ij}

$$\mathcal{I}_{123}[4\mu_{12}^2 - 4\mu_{11}\mu_{22}] = -2\epsilon(1+2\epsilon)\mathcal{I}^{(d+2)}, \qquad \mathcal{I}_{123}[2\mu_{12} - \mu_{11} - \mu_{22}] = -\frac{4-d}{d}\mathcal{I}_{12}$$

Final result in d dimensions

$$\mathcal{I}_{123} = \frac{d-2}{2m^2(d-3)}\mathcal{I}_{12}$$

Summary and Outlook

- Summary
  - we have a framework for the all-loop reduction at the integrand level
  - the integrand is decomposed via multivariate polynomial division
  - at one loop it reproduces well knwon results (OPP)
  - one-loop reduction is improved by Laurent expansion (NINJA)
  - algebraic reduction at any loop via divide-and-conquer approach
  - IBPs via integrand reduction and d-shifts

- Outlook
  - improve one-loop generation (recursion, global abbreviations,. . .)
  - treatment of (few) remaining unstable points within NINJA
  - application of int. red. + d-shifts a full two-loop QED/QCD process
  - fully automated analytic one-loop via divide-and-conquer

THANK YOU
FOR YOUR ATTENTION

BACKUP SLIDES

Rotation method for error estimation

H. van Deurzen, G. Luisoni, P. Mastrolia, E. Mirabella, G. Ossola, T.P. (2013)

- Definitions

$$\begin{aligned} A &: \quad \text{numerical result for the amplitude} \\ A_{rot} &: \quad \text{numerical result for the amplitude with rotated kinematics} \\ A_{ex} &: \quad \text{exact result for the amplitude} \sim \text{amplitude in quad. prec.} \end{aligned}$$

- the exact error is defined as

$$\delta_{ex} = \left| \frac{A_{ex} - A}{A_{ex}} \right|$$

- the estimated error is defined as

$$\delta_{rot} = 2 \left| \frac{A_{rot} - A}{A_{rot} + A} \right|$$

- one can check that $\delta_{rot} \sim \delta_{ex}$

Rotation method for error estimation

A validation of the rotation method

- example: $W\,b\,\bar{b} + 1\,j\ \ (u\bar{d} \to e^{+}\nu_e b\bar{b}g)$, with $m_b \neq 0$