COLLIER

A Complex One-Loop LIbrary with Extended Regularizations

Lars Hofer
UB Barcelona

UNIVERSITAT DE BARCELONA

EXCELENCIA MARÍA DE MAEZTU

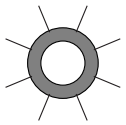
Institute of Cosmos Sciences

in collaboration with
A. Denner and S. Dittmaier

Loopfest, Buffalo, August 2015

One-loop amplitudes

general structure of one-loop amplitudes:

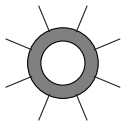


$$= \int d^D q \, \frac{N(q)}{D_0 \cdots D_{N-1}} = \sum_r c_{\mu_1 \ldots \mu_r} \underbrace{\int d^D q \, \frac{q^{\mu_1} \cdots q^{\mu_r}}{D_0 \cdots D_{N-1}}}_{\text{tensor integral } T^{\mu_1 \cdots \mu_r}}$$

with $D_i = (q + p_i)^2 - m_i^2$

One-loop amplitudes

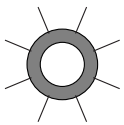
general structure of one-loop amplitudes:



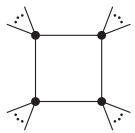
$$= \int d^D q \, \frac{N(q)}{D_0 \cdots D_{N-1}} = \sum_r c_{\mu_1 \ldots \mu_r} \underbrace{\int d^D q \, \frac{q^{\mu_1} \cdots q^{\mu_r}}{D_0 \cdots D_{N-1}}}_{\text{tensor integral } T^{\mu_1 \cdots \mu_r}}$$

with $D_i = (q + p_i)^2 - m_i^2$

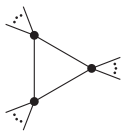
can be decomposed in terms of scalar integrals:



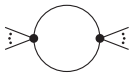
$$= \sum_l d_l$$



$$+ \sum_k c_k$$



$$+ \sum_j b_j$$



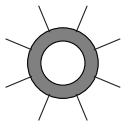
$$+ \sum_i a_i$$

$$+ R$$

$$= \sum_l d_l D_0(l) + \sum_k c_k C_0(k) + \sum_j b_j B_0(j) + \sum_i a_i A_0(i) + R$$

One-loop amplitudes

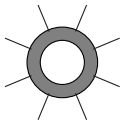
general structure of one-loop amplitudes:



$$= \int d^D q \, \frac{N(q)}{D_0 \cdots D_{N-1}} = \sum_r c_{\mu_1 \ldots \mu_r} \underbrace{\int d^D q \, \frac{q^{\mu_1} \cdots q^{\mu_r}}{D_0 \cdots D_{N-1}}}_{\text{tensor integral } T^{\mu_1 \cdots \mu_r}}$$

with $D_i = (q + p_i)^2 - m_i^2$

can be decomposed in terms of scalar integrals:



$$= \sum_l d_l \; + \sum_k c_k \; + \sum_j b_j \; + \sum_i a_i \; + R$$

$$= \sum_l d_l D_0(l) + \sum_k c_k C_0(k) + \sum_j b_j B_0(j) + \sum_i a_i A_0(i) + R$$

different approaches for calculation:

- conventional method (Feynman diagrams) $\rightarrow$ TI's needed
- generalised unitarity $\rightarrow$ SI's needed

  [Ossola, Papadopoulos, Pittau '07, Bern, Dixon, Kosower, Britto, Cachazo, Feng,Ellis, Giele, Melnikov, . . .]
- recursive methods using tensor integrals $\rightarrow$ TI's needed

  [van Hameren'09; Cascioli,Maierhöfer,Pozzorini'11; Actis,Denner,LH,Scharf,Uccirati'12]

Tools for NLO

- Many tools for NLO calculatios, e.g.
  `FeynArts`, `FeynCalc`, `FormCalc`, `Blackhat`, `NGluon`, `HELAC-NLO`, `GoSam`, `CutTools`, `HELAC-1LOOP`, `Samurai`, `MadGraph5_aMC@NLO`, `OpenLoops`, `Recola`

- Libraries for scalar and tensor integrals, e.g.
  `FF` [van Oldenborgh], `LoopTools` [Hahn,Perez-Victoria], `QCDLoop` [R.K.Ellis,Zanderighi], `OneLOop` [van Hameren], `Golem95C` [Cullen,Guillet,Heinrich,Kleinschmidt,Pilon,...], `PJFry` [Fleischer,Riemann]

- This talk:

  COLLIER **A Complex One-Loop LIbrary with Extended Regularizations**

  fortran-library for fast and stable numerical evaluation of scalar and tensor integrals [Denner,Dittmaier,LH]

  Released on April 25:
  `http://collier.hepforge.org`, arXiv:1604.06792

Collier: Applications

successfully used in many calculations of

- NLO QCD corrections, e.g.
  $pp \to t\bar{t}b\bar{b}$ [Bredenstein,Denner,Dittmaier,Pozzorini '09]
  $pp \to WWb\bar{b}$ [Denner,Dittmaier,Kallweit,Pozzorini '11]
  $pp \to WWb\bar{b}H$ [Denner,Feger '15] (incl. 7-point integrals)

- NLO EW corrections, e.g.
  $e^+e^- \to$ 4 fermions [Denner,Dittmaier,Roth,Wieders '05]
  $pp \to Hjj$ via VBF [Ciccolini,Denner,Dittmaier '07]
  $pp \to l^+l^-jj$ [Denner,LH,Scharf,Uccirati '14]
  $pp \to \mu^+\mu^-e^+e^-, \mu^+e^-\nu_\mu\bar{\nu}_e$ [Biedermann et al. '16]
  $pp \to e^+\bar{\nu}_e\mu^-\bar{\nu}_\mu b\bar{b}$ [Denner,Pellen '16] (incl. 8-point integrals)

Collier: Applications

successfully used in many calculations of

- NLO QCD corrections, e.g.
  $pp \to t\bar{t}b\bar{b}$ [Bredenstein,Denner,Dittmaier,Pozzorini '09]
  $pp \to WWb\bar{b}$ [Denner,Dittmaier,Kallweit,Pozzorini '11]
  $pp \to WWb\bar{b}H$ [Denner,Feger '15] (incl. 7-point integrals)

- NLO EW corrections, e.g.
  $e^+e^- \to$ 4 fermions [Denner,Dittmaier,Roth,Wieders '05]
  $pp \to Hjj$ via VBF [Ciccolini,Denner,Dittmaier '07]
  $pp \to l^+l^-jj$ [Denner,LH,Scharf,Uccirati '14]
  $pp \to \mu^+\mu^-e^+e^-, \mu^+e^-\nu_\mu\bar{\nu}_e$ [Biedermann et al. '16]
  $pp \to e^+\bar{\nu}_e\mu^-\bar{\nu}_\mu b\bar{b}$ [Denner,Pellen '16] (incl. 8-point integrals)

- NNLO QCD corrections:
  real-virtual corrections require calculation of one-loop integrals
  for configurations with unresolved external particles
  $\to$ kinematics involves collinear/soft external momenta
  $pp \to Z\gamma$ [Grazini,Kallweit,Ratlev,Torre '13]
  $pp \to ZZ$ [Cascioli et al. '14], $pp \to WW$ [Gehrmann et al. '14]

Features of Collier (1)

- complete set of one-loop scalar integrals
- implementation of tensor integrals for (in principle) arbitrary number of external momenta N
  (tested in many physical processes up to $N = 8$)
- various expansion methods implemented for exceptional phase-space points
  (to arbitrary order in expansion parameter)
- mass- and dimensional regularisation supported for IR-singularities
- complex masses supported (unstable particles)
- output: coefficients $T^N_{0\cdots 0 i_1 \cdots i_k}$ or tensors $(T^N)^{\mu_1 \cdots \mu_P}$

Features of Collier (2)

- two independent implementations: COLI+DD
- cache-system to avoid recalculation of identical integrals
- uncertainty estimates are performed and returned with the results for the integrals
- problematic integrals can be reported to log-files
- demo programs illustrating the usage of the library

Features of Collier (2)

- two independent implementations: COLI+DD
- cache-system to avoid recalculation of identical integrals
- uncertainty estimates are performed and returned with the results for the integrals
- problematic integrals can be reported to log-files
- demo programs illustrating the usage of the library

integrated in automated NLO generators

- OpenLoops [Cascioli,Maierhöfer,Pozzorini]
- Recola [Actis,Denner,LH,Lang,Scharf,Uccirati]

  Released on May 3:
  `http://recola.hepforge.org`, arXiv:1605.01090

Tensor integrals in dim-reg

$$T^{N,\mu_1\dots\mu_P} = \frac{(2\pi\mu)^{2\epsilon}}{\mathrm{i}\pi^2}\int \mathrm{d}^D q\, \frac{q^{\mu_1}\cdots q^{\mu_P}}{N_0 N_1 \dots N_{N-1}}, \qquad N_i = (q+p_i)^2 - m_i^2$$

Structure UV- or IR-singular integrals in $D = 4 - 2\epsilon$ dimensions:

$$T^N = \tilde{T}^N_{\text{fin}} + \frac{a^{\text{UV}}}{\epsilon_{\text{UV}}}\mu_{\text{UV}}^{2\epsilon_{\text{UV}}} + \left(\frac{a^{\text{IR}}}{\epsilon_{\text{IR}}^2} + \frac{a^{\text{IR}}}{\epsilon_{\text{IR}}}\right)\mu_{\text{IR}}^{2\epsilon_{\text{IR}}}$$

Tensor integrals in dim-reg

$$T^{N,\mu_1\dots\mu_P} = \frac{(2\pi\mu)^{2\epsilon}}{\mathrm{i}\pi^2}\int \mathrm{d}^D q\, \frac{q^{\mu_1}\cdots q^{\mu_P}}{N_0 N_1 \dots N_{N-1}}, \qquad N_i = (q+p_i)^2 - m_i^2$$

Structure UV- or IR-singular integrals in $D = 4 - 2\epsilon$ dimensions:

$$T^N = \tilde{T}^N_{\text{fin}} + \frac{a^{\text{UV}}}{\epsilon_{\text{UV}}}\mu_{\text{UV}}^{2\epsilon_{\text{UV}}} + \left(\frac{a^{\text{IR}}}{\epsilon_{\text{IR}}^2} + \frac{a^{\text{IR}}}{\epsilon_{\text{IR}}}\right)\mu_{\text{IR}}^{2\epsilon_{\text{IR}}}$$

- change of normalization of tensor integral

  $$T^N \to c(\epsilon) T^N \qquad \text{with} \qquad c(\epsilon) = 1 + \mathcal{O}(\epsilon)$$

  is equivalent to a redefinition

  $$\frac{1}{\epsilon_{\text{UV}}} \to \Delta_{\text{UV}} = \frac{c(\epsilon)}{\epsilon_{\text{UV}}}, \qquad \frac{1}{\epsilon_{\text{IR}}} \to \Delta^{(1)}_{\text{IR}} = \frac{c(\epsilon)}{\epsilon_{\text{IR}}}, \qquad \frac{1}{\epsilon^2_{\text{IR}}} \to \Delta^{(2)}_{\text{IR}} = \frac{c(\epsilon)}{\epsilon^2_{\text{IR}}}$$

- convention in Collier: $c(\epsilon) = \Gamma(1+\epsilon)(4\pi)^\epsilon$

Output of Collier

$$T^N = T^N_{\text{fin}}(\mu^2_{\text{UV}}, \mu^2_{\text{IR}}) + a^{\text{UV}}\,\Delta_{\text{UV}} + a_2^{\text{IR}}\left(\Delta^{(2)}_{\text{IR}} + \Delta^{(1)}_{\text{IR}}\,\ln\mu^2_{\text{IR}}\right) + a_1^{\text{IR}}\Delta^{(1)}_{\text{IR}}$$

- scales $\mu^2_{\text{UV}}, \mu^2_{\text{IR}}$ and poles $\Delta_{\text{UV}}, \Delta^{(1)}_{\text{IR}}, \Delta^{(2)}_{\text{IR}}$ can be set to arbitrary real values

  $\Rightarrow$ output of Collier: numerical value for full T^N

Output of Collier

$$T^N = T^N_{\text{fin}}(\mu^2_{\text{UV}}, \mu^2_{\text{IR}}) + a^{\text{UV}}\Delta_{\text{UV}} + a_2^{\text{IR}}\left(\Delta^{(2)}_{\text{IR}} + \Delta^{(1)}_{\text{IR}} \ln \mu^2_{\text{IR}}\right) + a_1^{\text{IR}}\Delta^{(1)}_{\text{IR}}$$

- scales $\mu^2_{\text{UV}}, \mu^2_{\text{IR}}$ and poles $\Delta_{\text{UV}}, \Delta^{(1)}_{\text{IR}}, \Delta^{(2)}_{\text{IR}}$ can be set to arbitrary real values

  $\Rightarrow$ output of Collier: numerical value for full T^N

- cancellation of poles can be checked numerically by varying $\Delta_{\text{UV}},\ \Delta^{(1)}_{\text{IR}},\ \Delta^{(2)}_{\text{IR}}$

- convention for prefactor $c(\epsilon)$ can be changed by shifting $\Delta_{\text{UV}},\ \Delta^{(1)}_{\text{IR}},\ \Delta^{(2)}_{\text{IR}}$ accordingly

- coefficient a^{UV} of $1/\epsilon_{\text{UV}}$ - pole returned also as separate output

Methods implemented in Collier

applied method depends on number N of propagators

- $N = 1, 2$: explicit analytical expressions
- $N = 3, 4$: exploit Lorentz-covariance

  standard PV reduction [Passarino,Veltman '79]

  $\rightarrow$ spurious $1/\det(Z)$ divergences in exceptional phase-space regions ($\det(Z)$: Gram determinant)

  $\Rightarrow$ stable expansions in $(\det(Z))^g$ [Denner,Dittmaier '05]
- $N \geq 5$: exploit 4-dimensionality of space-time

  [Melrose '65; Denner,Dittmaier '02,'05; Binoth et al. '05]

Basic scalar integrals from analytic expressions
['t Hooft,Veltman'79; Beenaker,Denner'90; Denner,Nierste,Scharf'91; Ellis,Zanderighi'08; Denner,Dittmaier'11]

$\Rightarrow$ fast and stable numerical reduction algorithm

Calculation of 3- and 4-point coefficients

1. Perform PV reduction:

   accuracy ΔT^N better than target precision η_{req}?

   yes → return result — done

Calculation of 3- and 4-point coefficients

1. Perform PV reduction:

   accuracy ΔT^N better than target precision η_{req}?

   yes → return result → done

   no ↓

2. Switch to expansions:

   select method with best a-priori error estimate $\Delta T^N_{\mathrm{prelim}}$

Calculation of 3- and 4-point coefficients

1. Perform PV reduction:

   accuracy ΔT^N better than target precision η_{req}?

   yes → return result — done

   no ↓

2. Switch to expansions:

   select method with best a-priori error estimate $\Delta T^N_{\mathrm{prelim}}$

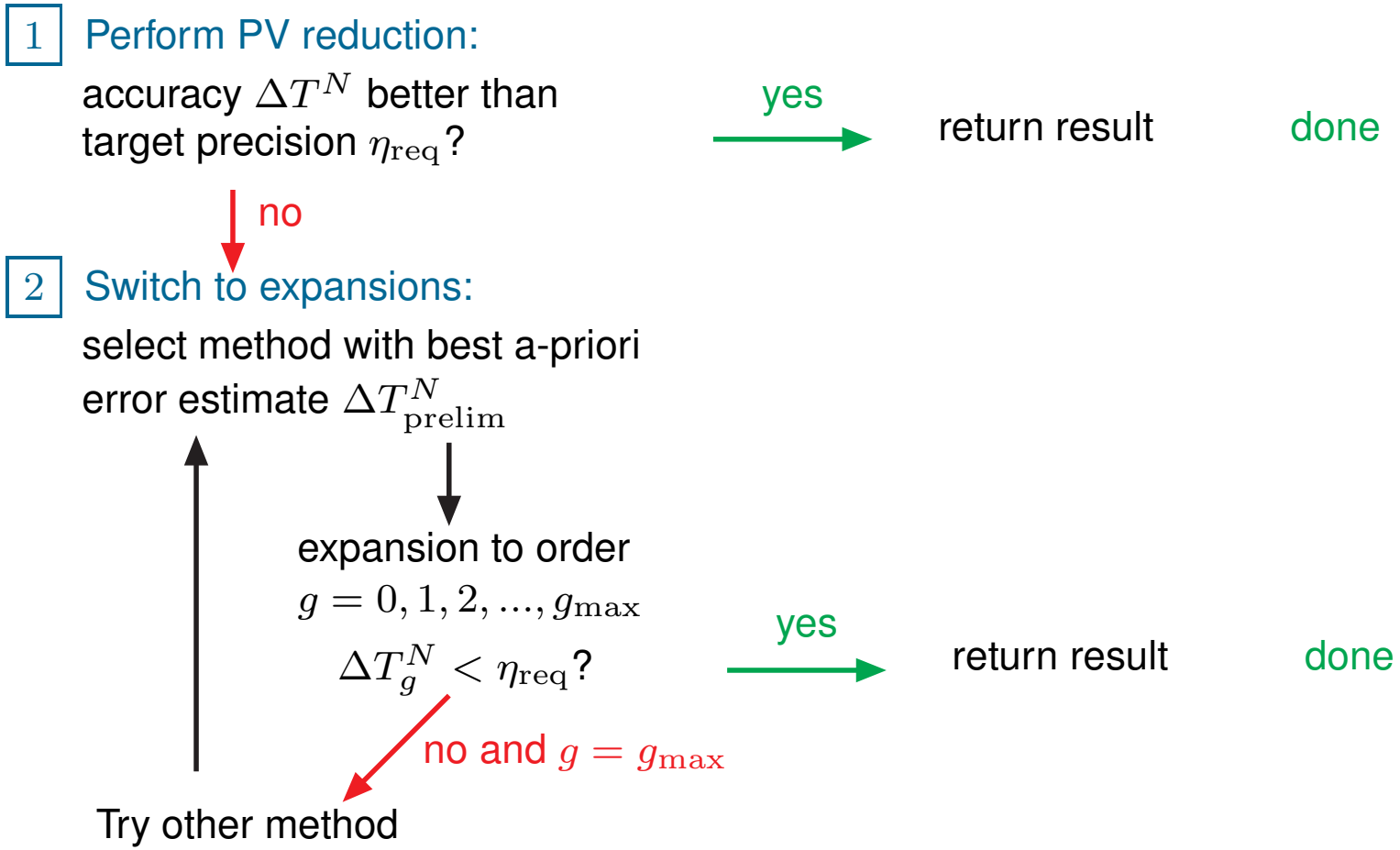
   ↓

   expansion to order $g = 0, 1, 2, \ldots, g_{\max}$

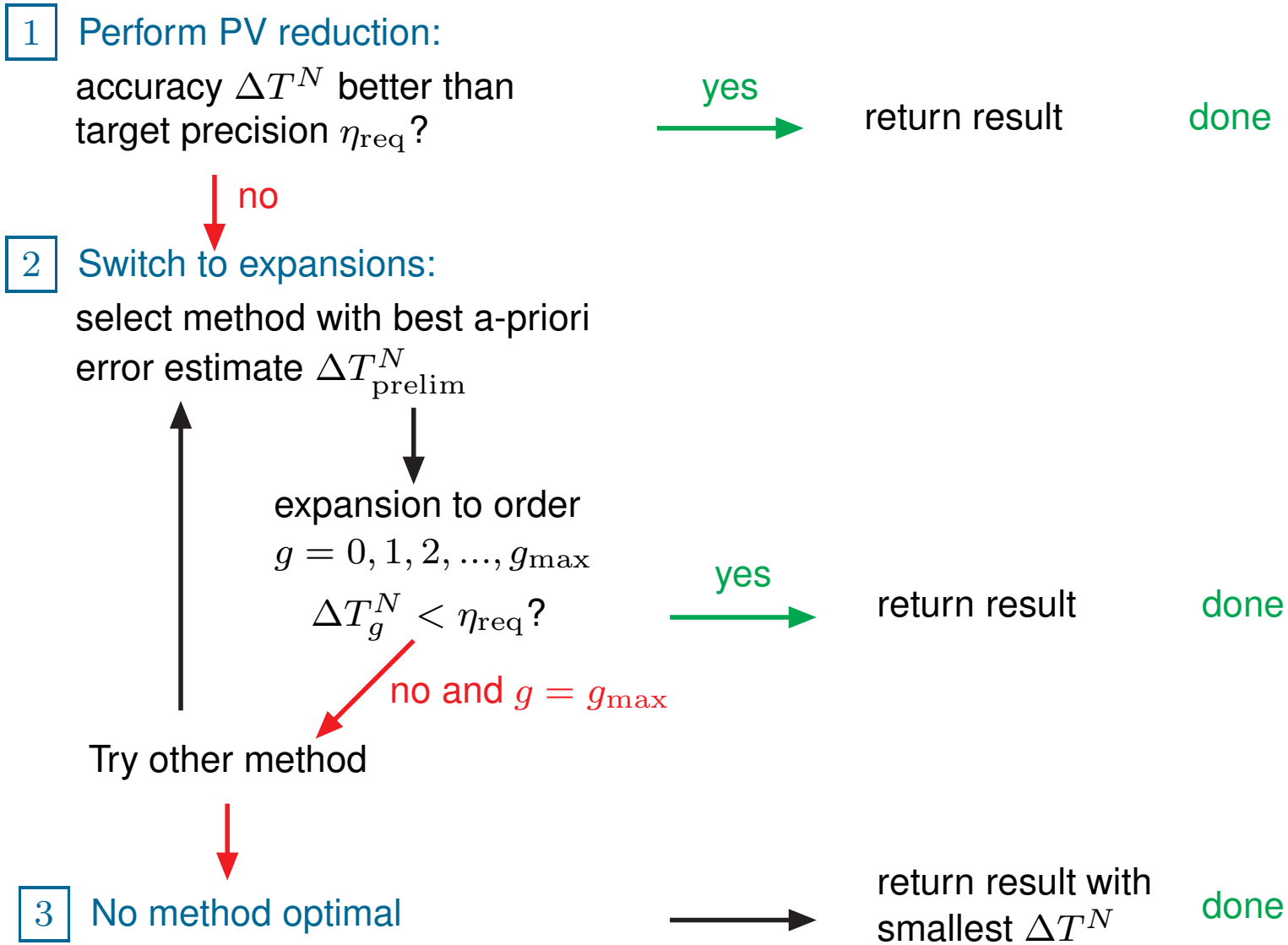
   $\Delta T^N_g < \eta_{\mathrm{req}}$?

   yes → return result — done

Calculation of 3- and 4-point coefficients



Calculation of 3- and 4-point coefficients



Error estimates

1 PV-reduction

- error propagation during reduction:
  $\delta D_0 = 10 \times \text{machine precision}$
  $\delta D_r \sim a_{\max}\, \delta D_{r-1} \qquad (a_{\max} \sim 1/\det(Z))$
  (+error propagation from 3-point functions)
- symmetry of coefficients
  $$\delta D_r \sim |D_{i_1 i_2 \ldots i_r} - D_{i_2 i_1 \ldots i_r}|, \qquad (0 \neq i_1 \neq i_2 \neq 0)$$

Error estimates

1 PV-reduction

- error propagation during reduction:
  $\delta D_0 = 10 \times$ machine precision
  $\delta D_r \sim a_{\max}\, \delta D_{r-1} \qquad (a_{\max} \sim 1/\det(Z))$
  (+error propagation from 3-point functions)
- symmetry of coefficients
  $$\delta D_r \sim |D_{i_1 i_2 \ldots i_r} - D_{i_2 i_1 \ldots i_r}|, \qquad (0 \neq i_1 \neq i_2 \neq 0)$$

2 Expansions: $D_r = D_r^{(0)} + \ldots + D_r^{(g)}$

- a-prori estimate:
  neglected higher orders: $\delta D_r \sim (\det(Z))^{g+1}$
  (+error propagation from 3-point functions)
- extrapolation after calculation: $\delta D_r = D_r^{(g)} \times \dfrac{D_r^{(g)}}{D_r^{(g-1)}}$

Error estimates

1 PV-reduction

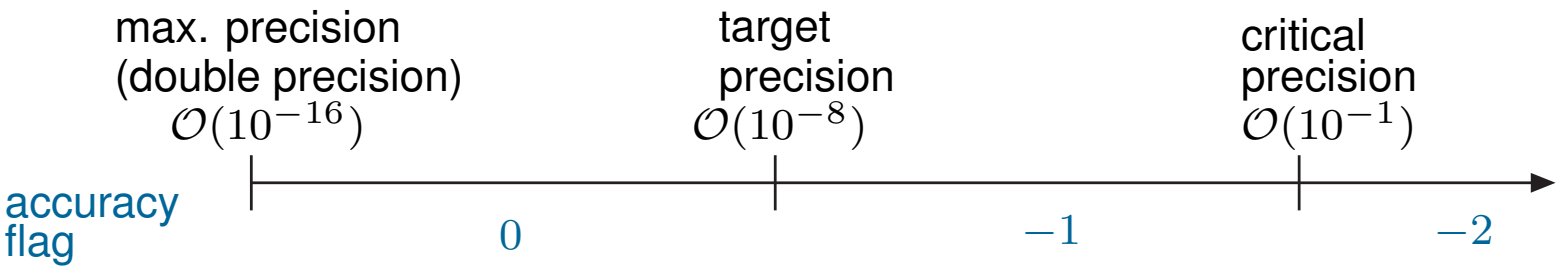
- error propagation during reduction:
  $\delta D_0 = 10 \times$ machine precision
  $\delta D_r \sim a_{\max}\, \delta D_{r-1} \qquad (a_{\max} \sim 1/\det(Z))$
  (+error propagation from 3-point functions)
- symmetry of coefficients
  $$\delta D_r \sim |D_{i_1 i_2 \ldots i_r} - D_{i_2 i_1 \ldots i_r}|, \qquad (0 \neq i_1 \neq i_2 \neq 0)$$

2 Expansions: $D_r = D_r^{(0)} + \ldots + D_r^{(g)}$

- a-prori estimate:
  neglected higher orders: $\delta D_r \sim (\det(Z))^{g+1}$
  (+error propagation from 3-point functions)
- extrapolation after calculation: $\delta D_r = D_r^{(g)} \times \dfrac{D_r^{(g)}}{D_r^{(g-1)}}$

+ error estimate at almost zero cost in run-time
– only rough order-of-magnitude estimate, typically accurate within a factor of 10-100 (conservative in most cases)

Precision handling



- target precision:
  governs selection of expansion method and expansion depth
  $\rightarrow$ balancing between precision and run-time

- critical precision:
  arguments and results of function calls are reported to an output file if estimated accuracy is worse than critical precision

- accuracy flag:
  stores status of worst integral within all function calls of the same phase space point (reinitialized for new phase-space point)

Coefficients vs. tensors

$$(T^N)^{\mu_1\cdots\mu_P} = \sum_k \sum_{i_1,\ldots,i_k} T^{N,P}_{\underbrace{0\cdots0}_{P-k}\, i_1\cdots i_k} \{ \underbrace{g\cdots g}_{(P-k)/2}\, p_{i_1}\cdots p_{i_k}\}^{\mu_1\cdots\mu_P}$$

# of tensor coefficients (TC) vs. # of tensor elements (TE)

	$r=0$	$r=1$	$r=2$	$r=3$	$r=4$	$r=5$	$r=6$
$N=3$	1	3	7	13	22	34	50
$N=4$	1	4	11	24	46	80	130
$N=5$	1	5	16	40	86	166	296
$N=6$	1	6	22	62	148	314	610
$N=7$	1	7	29	91	239	553	1163
tensor	1	5	15	35	70	126	210

#TC $<$ #TE

#TC $>$ #TE

Coefficients vs. tensors

$$(T^N)^{\mu_1\cdots\mu_P} = \sum_k \sum_{i_1,\ldots,i_k} T^{N,P}_{\underbrace{0\cdots0}_{P-k}\, i_1\cdots i_k} \{ \underbrace{g\cdots g}_{(P-k)/2}\, p_{i_1}\cdots p_{i_k}\}^{\mu_1\cdots\mu_P}$$

# of tensor coefficients (TC) vs. # of tensor elements (TE)

	$r=0$	$r=1$	$r=2$	$r=3$	$r=4$	$r=5$	$r=6$
$N=3$	1	3	7	13	22	34	50
$N=4$	1	4	11	24	46	80	130
$N=5$	1	5	16	40	86	166	296
$N=6$	1	6	22	62	148	314	610
$N=7$	1	7	29	91	239	553	1163
tensor	1	5	15	35	70	126	210

#TC $<$ #TE

#TC $>$ #TE

NLO generators OpenLoops and Recola:
parametrisation of one-loop amplitude in terms of tensor integrals:

calculated by OpenLoops/Recola

$$\mathcal{M} = \sum_j c^{(j)}_{\mu_1\ldots\mu_{n_j}} T^{\mu_1\ldots\mu_{n_j}}_{(j)} \rightarrow \text{Tensor Integrals}$$

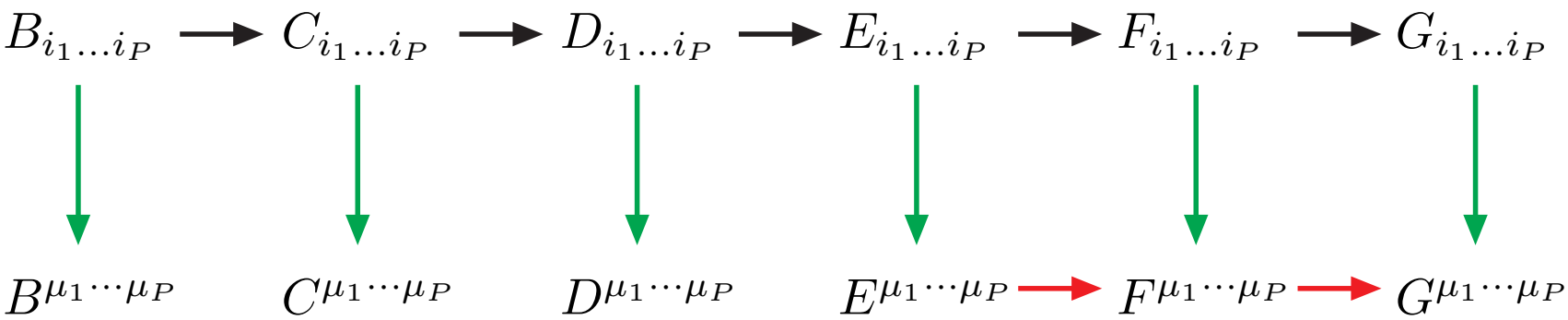
$\Rightarrow$ need full tensors!

From coefficients to tensors

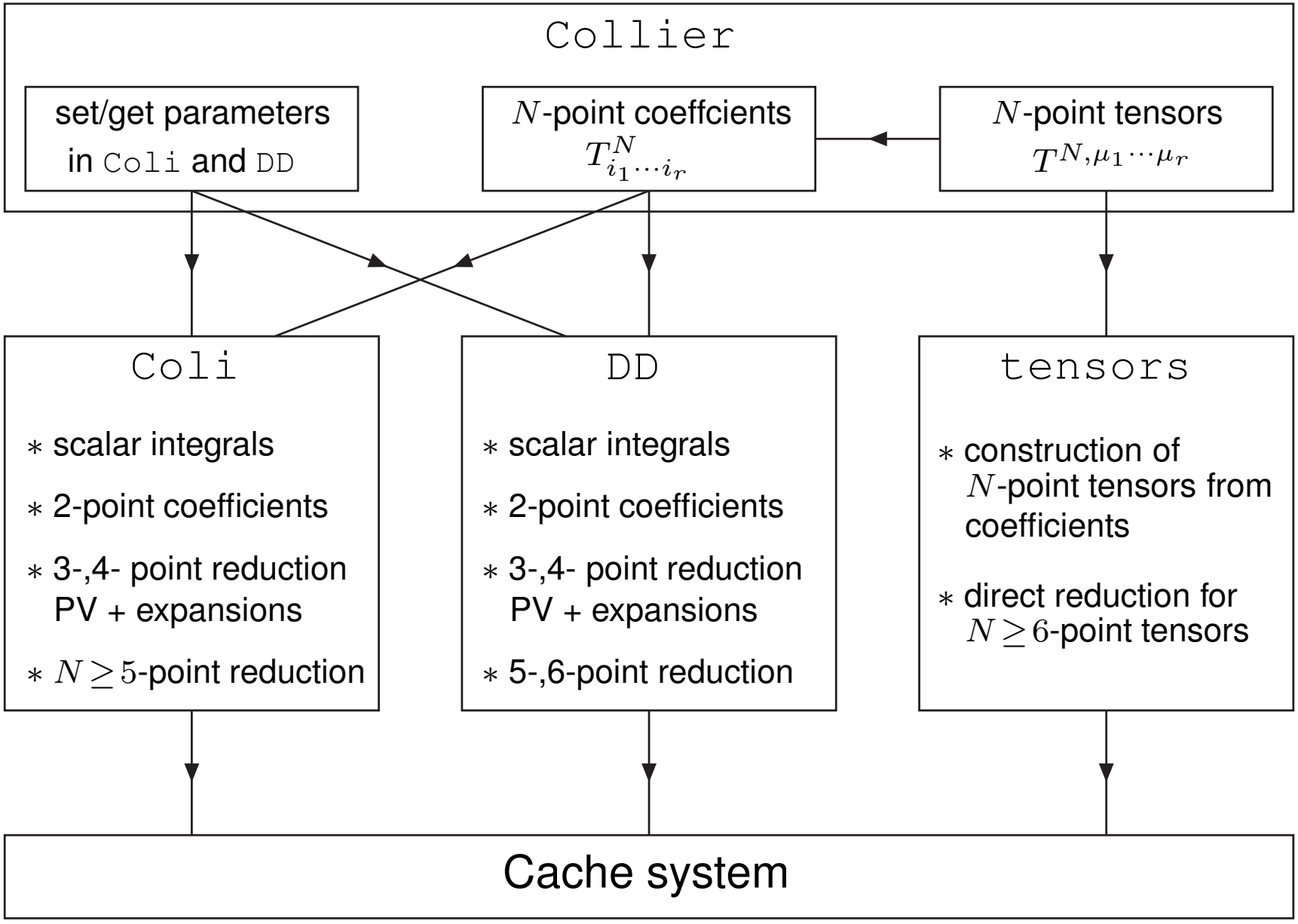
$$(T^N)^{\mu_1\cdots\mu_P} = \sum_k \sum_{i_1,\ldots,i_k} T^{N,P}_{\underbrace{0\cdots0}_{P-k} i_1\cdots i_k} \{ \underbrace{g\cdots g}_{(P-k)/2}\, p_{i_1}\cdots p_{i_k}\}^{\mu_1\cdots\mu_P}$$

In Collier:

- output: coefficients $T^N_{0\cdots0 i_1\cdots i_k}$ or tensors $(T^N)^{\mu_1\cdots\mu_P}$
- efficient algorithm to construct tensors from invariant coefficients for arbitrary N, P via recursive calculation of tensor structures
- for $N \geq 6$: Direct reduction at tensor level

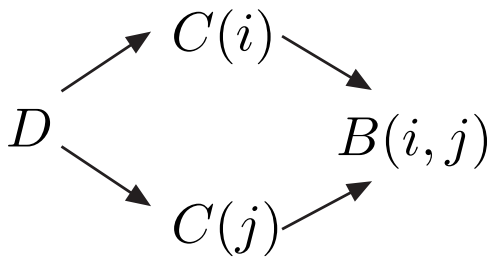


Structure of Collier



Cache system

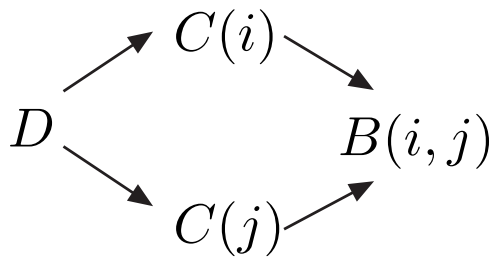
Evaluation of one-loop amplitude leads to multiple calls for the same tensor integral (TI):



- within one master-call:
  same TI appears several times
  in reduction tree
- different master calls and their reductions lead to same TI

Cache system

Evaluation of one-loop amplitude leads to multiple calls for the same tensor integral (TI):



- within one master-call:
  same TI appears several times
  in reduction tree
- different master calls and their reductions lead to same TI

Cache system in Collier:

- Identify each TI-call via index pair (N, i):
  N= number of external master call
  i = binary index for internal calls (propagated in reduction)
- pointers for each pair (N, i) point to same address in cache if arguments of TI's are identical
  first call: write cache further calls: read cache
- internal calls: always cached
  external calls: only cached if global cache is switched on by user
  $\rightarrow$ requires same sequence of integral calls for each event!

Download & installation

- download `collier-X.Y.Z.tar.gz` from
  `http://collier.hepforge.org`
  and unpack the archive:
  `tar -zxvf collier-X.Y.Z.tar.gz`

- easy installation via the CMAKE build system:
  ```
  cd COLLIER-X.Y.Z/build
  cmake [options] ..
  make
  ```

- options:

  `-Dstatic=ON`

  → create static instead of dynamic library

  `-DCMAKE_Fortran_COMPILER=comp`

  → use comp as fortran compiler (comp can be gfortran, ifort, pgf95, ... or the full path to a compiler)

Initialization

```
! include module COLLIER
use COLLIER
```

Initialization

```
! include module COLLIER
use COLLIER

! initialization:
! example for up to 6-point integrals of rank up to 4
! with file output directed to folder "output_cll"
! (empty string suppresses file output)
call Init_cll(Nmax=6,rmax=4,"output_cll")
! --> all parameters are set to default values
!     (can be modified later)
```

Initialization

```
! include module COLLIER
use COLLIER

! initialization:
! example for up to 6-point integrals of rank up to 4
! with file output directed to folder "output_cll"
! (empty string suppresses file output)
call Init_cll(Nmax=6,rmax=4,"output_cll")
! --> all parameters are set to default values
!     (can be modified later)

! set UV renormalization scale to 100GeV (default=1GeV)
call SetMuUV2_cll(1d4)

! choose mode (default=1)
! mode=1: use COLI
! mode=2: use DD
! mode=3: use COLI+DD and report differences
call setMode_cll(3)
```

Initialization

```
! include module COLLIER
use COLLIER

! initialization:
! example for up to 6-point integrals of rank up to 4
! with file output directed to folder "output_cll"
! (empty string suppresses file output)
call Init_cll(Nmax=6,rmax=4,"output_cll")
! --> all parameters are set to default values
!     (can be modified later)

! set UV renormalization scale to 100GeV (default=1GeV)
call SetMuUV2_cll(1d4)

! choose mode (default=1)
! mode=1: use COLI
! mode=2: use DD
! mode=3: use COLI+DD and report differences
call setMode_cll(3)

! initialize global cache system:
! use 1 cache, integrals up to 6-point functions cached
call InitCacheSystem_cll(1,Nmax=6)
```

Collier parameters

parameter	type	set with	default
mode	integer $\in \{1,2,3\}$	SetMode_cll	1
η_{req}	double precision	SetReqAcc_cll	1d-8
η_{crit}	double precision	SetCritAcc_cll	1d-1
η_{check}	double precision	SetCheckAcc_cll	1d-4
μ^2_{UV}	double precision	SetMuUV2_cll	1d0
μ^2_{IR}	double precision	SetMuIR2_cll	1d0
Δ_{UV}	double precision	SetDeltaUV_cll	0d0
$\Delta^{(1)}_{\text{IR}}, \Delta^{(2)}_{\text{IR}}$	double precision	SetDeltaIR_cll	0d0,0d0
$\{\overline{m}^2_1, \ldots, \overline{m}^2_{n_{\text{reg}}}\}$	double complex (n_{reg})	SetMinf2_cll	{}
σ_{stop}	integer < 0	SetErrStop_cll	-8
N_{tenred}	integer ≥ 6	SetTenRed_cll	6
n_{cache}	integer ≥ 0	InitCacheSystem_cll	0
$N^{\text{max}}_{\text{cache}}$	integer (n_{cache}) ≥ 1	SetCacheLevel_cll	–
n_{err}	integer ≥ 0	SetMaxErrOut_cll	100
$n^{N,\text{max}}_{\text{check}}$	integer (N_{max}) ≥ 0	SetMaxCheck_cll	$\{50, \ldots, 50\}$
$n^{N,\text{max}}_{\text{crit}}$	integer (N_{max}) ≥ 0	SetMaxCrit_cll	$\{50, \ldots, 50\}$
$\widehat{P}^{\text{max}}$	integer ≥ 6	SetRitmax_cll	14

Usage of Collier in a MC generator

```
! loop over MC events
do nevent=1,nevent_tot

  ! loop over channels
  do nchan=1,nchan_tot

    ! initialize new event in Collier
    call InitEvent(ncache(nchan))
    ! --> reinitializes corresponding cache
    ! --> resets error and accuracy flags

    ! reset parameters if needed
    call SetMuUV2_cll(Q2run)

    ! sequence of integral calls
    ! SAME ORDER FOR EACH EVENT IF CACHE USED!!!
    call TN_cll(TN1,TN1uv,MomVec1,MomInv1,masses1,N1,r1,TN1err)
    ...

    ! read out error and accuracy flags
    call GetErrFlag(eflag)
    call GetAccFlag(aflag)

  end do
end do
```

Features of RECOLA + COLLIER

- tree-level and one-loop amplitudes in the full SM
- recursive calculation employing Dyson-Schwinger-like recursion relations for computation of tensor coefficients
- complete set of SM counterterms and rational terms included
- complex mass scheme supported $\Rightarrow$ application to processes involving unstable particles possible
- efficient treatment of colour using colour-structures
- colour- and spin-correlated matrix elements for dipole subtraction

$\Rightarrow$ download combined Recola + Collier package from

`http://recola.hepforge.org`

Conclusions

- Collier= fortran library for numerical calculation of scalar and tensor integrals
- numerical stable results thanks to expansion methods for 3-,4-point integrals
- dimensional and mass regularization supported, as well as complex masses for unstable particles
- two independent implementations: Collier = Coli + DD
- the combination OpenLoops + Recola allows a fast and numerically stable evaluation of QCD and EW one-loop amplitudes to arbitrary SM processes
- Collier and Recola have been published recently and are available from hepforge