

Sherpa+OpenLoops: Experience & Suggestions

Philipp Maierhöfer

Physik-Institut
Universität Zürich

NLO Users of Sherpa Meeting
München, 10 January 2014

In collaboration with
F. Cascioli, S. Kallweit, N. Moretti and S. Pozzorini

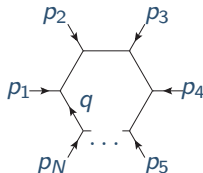
Outline

1 Where are we?

2 What could be improved?

OpenLoops

To calculate a one-loop amplitude, we start from Feynman diagrams, factorised into **colour factors**, **tensor coefficients**, and **tensor integrals**.



$$= \mathcal{C} \cdot \sum_{r=0}^R \mathcal{N}_r^{\mu_1 \dots \mu_r} \cdot \int d^d q \frac{D_i = (q + \sum_{\ell=0}^i p_\ell)^2 - m_i^2}{D_0 D_1 \dots D_{N-1}}$$

Open loops is an algorithm for the numerical recursive construction of the coefficients $\mathcal{N}_r^{\mu_1 \dots \mu_r}$ in 4 dimensions, combined with tensor integral reduction (Collier [Denner, Dittmaier, Hofer]) or, alternatively, OPP reduction (CutTools [Ossola, Papadopoulos, Pittau], Samurai [Mastrolia, Ossola, Reiter, Tramontano]).

Universal building blocks: connect vertices and propagators around the loop, factorising the loop momenta.

Implemented in OpenLoops (and now also independently in MadLoop).

Status

Sherpa+OpenLoops is working smoothly and producing output

- 4-leptons+jet, merged [Cascioli, Höche, Krauss, PM, Pozzorini, Siegert]
- $t\bar{t}b\bar{b}$ ($m_b > 0$), showered [Cascioli, PM, Moretti, Pozzorini, Siegert]

And also OpenLoops within other frameworks

- $W^+W^-b\bar{b}$ ($m_b > 0$) [Cascioli, Kallweit, PM, Pozzorini]
- HHj , merged [PM, Papaefstathiou]
- $Z\gamma j$ real-virtual for NNLO [Grazzini, Kallweit, Rathlev, Torre]

Automation

Using Sherpa+OpenLoops boils down to writing a Sherpa run card.

Huge step in beyond fixed order NLO simulations in Sherpa 2.0:
MC@NLO matching & MEPS@NLO merging [Höche, Krauss, Schönherr, Siegert]

OpenLoops process libraries are available to the ATLAS and CMS Monte Carlo working groups.

OpenLoops Performance

process	diags	size/MB	time/ms
$u\bar{u} \rightarrow t\bar{t}$	11	0.1	0.27(0.16)
$u\bar{u} \rightarrow W^+ W^-$	12	0.1	0.14
$u\bar{d} \rightarrow W^+ g$	11	0.1	0.24
$u\bar{d} \rightarrow Zg$	34		0.75
$gg \rightarrow t\bar{t}$	44	0.2	1.6(0.7)
$u\bar{u} \rightarrow t\bar{t}g$	114	0.4	4.8(2.4)
$u\bar{u} \rightarrow W^+ W^- g$	198	0.4	3.4
$u\bar{d} \rightarrow W^+ gg$	144	0.5	4.0
$u\bar{d} \rightarrow Zgg$	408		17
$gg \rightarrow t\bar{t}g$	585	1.2	40(14)
$u\bar{u} \rightarrow t\bar{t}gg$	1507	3.6	134(101)
$u\bar{u} \rightarrow W^+ W^- gg$	2129	2.5	89
$u\bar{d} \rightarrow W^+ ggg$	1935	4.2	120
$u\bar{d} \rightarrow Zggg$	5274		524
$gg \rightarrow t\bar{t}gg$	8739	16	1460(530)

Measured on an i7-3770K (single thread) with gfortran 4.8 -O0, dynamic (ifort static $\sim 30\%$ faster), tensor integral reduction with Collier.

Colour and helicity summed.

W production includes leptonic decays and non-resonant contributions.

$t\bar{t}$ production numbers in brackets are for massless decays.

$\mathcal{O}(10^{-5})$ bad points in real life applications with collinear (decaying) particles, practically no bad points for well separated particles.

Decay Treatment

Decay afterburner

Sherpa provides the possibility to let on-shell particles decay.
Makes life simpler, but comes at a price.

- Sum over all helicities required.
- LO spin correlations only.
- No radiation from decay products.

Proposal: include decays in matrix elements

- Common for leptonic decays + QCD, *here*: resonant diagrams
→ hadronic decays and EW corrections possible.
- Often less helicities required.
- NLO spin correlations.
- Radiation from decay products.
- Corrections to decays (still) possible.

But **requires an appropriate phase space.**

Tree Matrix Element Performance

OpenLoops vs. MadGraph5 benchmark

(i5-750, gfortran 4.7 -O2, dynamic, timings in ms)

process	MG	OL	OL/MG
$gg \rightarrow t\bar{t}g$	0.070	0.021	0.3
$gg \rightarrow t\bar{t}gg$	0.91	0.32	0.35
$gg \rightarrow t\bar{t}ggg$	21	16	0.76
$ud \rightarrow e^+ \nu_e gg$	0.0060	0.0048	0.8
$u\bar{d} \rightarrow e^+ \nu_e ggg$	0.052	0.023	0.44
$u\bar{d} \rightarrow e^+ \nu_e gggg$	0.68	0.31	0.45
$u\bar{u} \rightarrow e^+ \nu_e \mu^- \bar{\nu}_\mu g$	0.010	0.0066	0.66
$u\bar{u} \rightarrow e^+ \nu_e \mu^- \bar{\nu}_\mu gg$	0.064	0.023	0.36
$u\bar{u} \rightarrow e^+ \nu_e \mu^- \bar{\nu}_\mu ggg$	0.66	0.21	0.32

OpenLoops is typically faster by a factor ~ 2 .

Particularly important if the integration time is dominated by real corrections.

Still potential for improvements.

What about AMEGIC/COMIX?

COMIX: effect of colour/helicity sampling (requires integration)?

Provide possibility to use OpenLoops tree matrix elements in Sherpa?

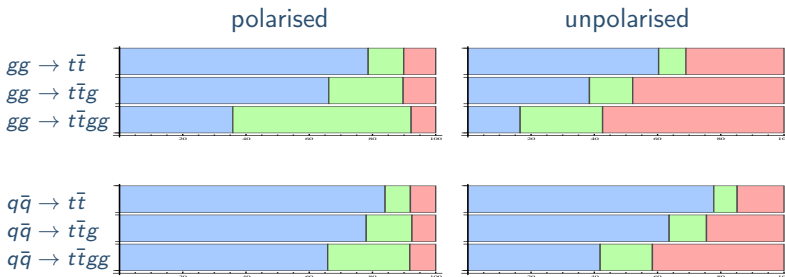
Needs convention/interface to pass colour information for the Shower.

Helicity Sums

For “not too complicated” processes up to $2 \rightarrow 4$

single helicity: time for tensor reduction \gg time for coefficients

full helicity sum: time for tensor reduction \approx time for coefficients



fractions of total runtime for **scalar integrals**, **tensor reduction**, **coefficients**

full helicity sums cost only a factor ~ 2

For higher multiplicities, esp. with many gluons, this is no more true.

When does helicity sampling start to make sense?

Integration Speed

Integrate $t\bar{t}b\bar{b}$ and $WWt\bar{t}$ with Sherpa and S. Kallweit's integrator (SK) and measure runtimes (hours) to achieve 0.2% accuracy (extrapolated).
 very preliminary (difficult to get strictly comparable numbers)

	α	Sherpa	SK	Sherpa/SK
$WWb\bar{b}$	1	4930	108	46
	0.01	2640	–	24
	0.001	1690	–	16
$t\bar{t}b\bar{b}$	1	542(252)	31	18(10)

Optimisation phase is not counted;
 $t\bar{t}b\bar{b}$ uses extended optimisation phase in Sherpa.

SK is an order of magnitude faster than Sherpa
 → lots of room for improvements.

Integration Optimisation

Exploit freedom in parameter choices to optimise the Integration

- CS α parameter (shift weights between VI and RS contributions).
- Choose individual target accuracies for different contributions with given (fixed) total accuracy such that the integration is fastest.

Monitor contributions and accuracies, **extrapolate accuracies in order to find optimal target accuracies for individual contributions.**

Is there a non-trivial interplay with the choice of α ?

Different story: easy to use optimisation of the integrator for loop induced processes (e.g. let OLP provide diagrams with effective vertices).

Scale variations

- SK **uses counterterms at different scales** (from OpenLoops)
+ a single evaluation of the loop amplitude.

MC+OLP interfacing

Sherpa uses a dedicated interface to OpenLoops.

- Tailored to fit the needs, avoids bloat due to unused features and peculiarities of the BLHA.
- But: interface changes affect both sides; no recyclability wrt. other MCs/OLPs

In the process of interfacing OpenLoops with Herwig++, a **BLHA interface** was **implemented in OpenLoops**.

- Interfacing on MC still requires a non-negligible amount of work.
- Still painful process of getting conventions consistent.

Try to **formulate demands on a future standard**: easy to use vendor independent API (no contract files) + universal mechanism to pass vendor specific parameters, names and conventions, imposing restrictions before processes are loaded, exception handling, process mappings, . . .

Replacing programs should not require changes in the interface code.

Conclusions

OpenLoops generator for one-loop amplitudes

- Numerical recursion for loop momentum polynomials
- Automatic, fast, stable (thanks to Collier)

Sherpa+OpenLoops

- Full automation of NLO simulations
- Write a run card and it works

Now that it's working it's time for optimisations

- Also think about tree matrix elements
- Treatment of decays
- Integrator optimisation, scale variations
- ...

To make decisions, one needs reliable and comparable benchmarks.