Automatic NLO calculations with **GoSam**

Gionata Luisoni

gionata.luisoni@durham.ac.uk

Institute for Particle Physics Phenomenology
University of Durham

In collaboration with:

G.Cullen, N. Greiner, G.Heinrich, P.Mastrolia, G.Ossola, T.Reiter, F. Tramontano

GoSam release: arXiv:1111.2034 [hep-ph]

IPPP, University of Durham

LHCPhenoNet Workshop

NLO multi-leg highlights

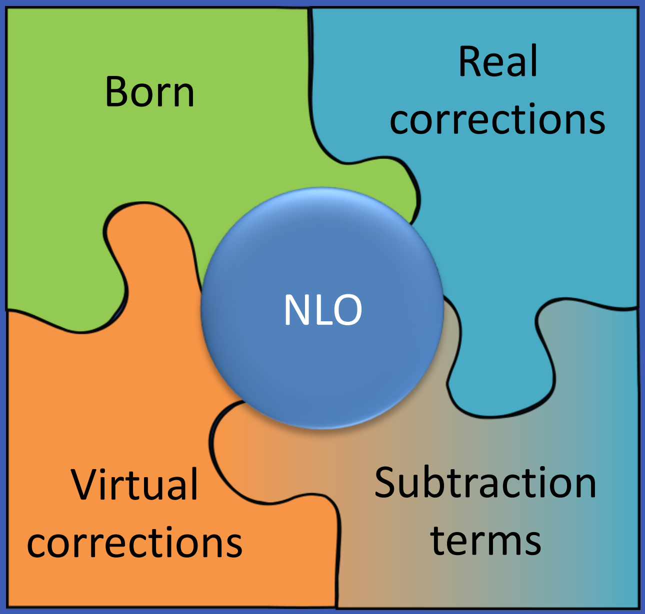
- Progresses in NLO calculation:
  - $\mathrm{pp} \longrightarrow W + 3\ \mathrm{jets}$ Blackhat (09) / Rocket (09)
  - $\mathrm{pp} \longrightarrow t\bar{t}b\bar{b}$ Denner-Dittmaier (09) / HELAC-NLO (09)
  - $\mathrm{pp} \longrightarrow Z(\gamma) + 3\ \mathrm{jets}$ Blackhat (10)
  - $\mathrm{pp} \longrightarrow t\bar{t}jj$ HELAC-NLO (10)
  - $\mathrm{pp} \longrightarrow W^+W^-b\bar{b}$ Denner-Dittmaier (10) / HELAC-NLO (10)
  - $e^+e^- \longrightarrow 5\ \mathrm{jets}$ Rocket (10)
  - $\mathrm{pp} \longrightarrow W^+W^+jj$ Rocket (10)
  - $\mathrm{pp} \longrightarrow Z(\gamma)\,/\,W + 4\ \mathrm{jets}$ Blackhat (11)
  - $\mathrm{pp} \longrightarrow b\bar{b}b\bar{b}$ Golem / Samurai (11)
  - $\mathrm{pp} \longrightarrow W^+W^-jj$ Rocket (11)
  - $\mathrm{pp} \longrightarrow 4\ \mathrm{jets}$ Blackhat (11)

Key Concept

AUTOMATION

Automation in NLO calculations

- Different ingredients of a NLO calculation have also different levels of automation according to their complexity:



- Virtual corrections
  - Automatized recently:
    - FEYNARTS/FORMCALC/LOOPTOOLS (public) [Hahn et al.]
    - HELAC-NLO (public) [Bevilacqua, Czakon, van Hameren, Papadopoulos, Pittau, Worek 11]
    - MadLoop [Hirschi,Frederix,Frixione,Garzelli, Maltoni,Pittau 11]
    - NGLUON (public) [Badger, Biedermann, Uwer 10]
    - GoSam (public) [Cullen, Greiner, Heinrich, GL, Mastrolia, Ossola, Reiter, Tramontano 11]
    - Dedicated programs also involve high level of automation:

Denner, Dittmaier, Pozzorini et al., VBFNLO (public), MCFM (public), BLACKHAT, ROCKET

Progresses in NLO calculation

- Evolution from collection of pre-coded processes…

… to generation of full NLO processes by the user "on the fly"!

- Possible thanks to pioneering works:
  - improvements on the computation of tensor integrals,
    [Binoth et al. GOLEM95; Denner, Dittmaier]
  - application of unitarity to the computation of the one loop amplitudes,
    [Bern, Dixon, Kosower; Britto, Cachazo, Feng]
  - reduction at the integrand level.
    [Ossola, Papadopoulos, Pittau; Ellis, Giele, Kunszt, Melnikov]

➡ FOCUS MORE ON PHENOMENOLOGY

The GoSam Project: phylosophy

GoSam

Golem (General One Loop Evaluator of Matrix elements)

Samurai (Scattering Amplitudes from Unitarity based Reduction At Integrand level)

An automated amplitude generation based on Feynman diagrams

- Based upon:
  - Algebraic generation of D-dimensional integrands via Feynman diagrams
  - Reduction at the integrand level via D-dimensional extension of the OPP method
  - Generation on the fly of the full rational term

The GoSam Project: goals

- Main targets:
  - Provide an automated tool for stable evaluation of one-loop matrix elements
  - Be general and model independent (QCD, EW, MSSM, …)
  - Interface with existing tools (MadEvent, Sherpa, POWHEG-BOX, …)
  - Build upon open source tools only (next slide)
  - Support open standards (for interfacing)

The GoSam Project: the codes

GoSam

Python package to write code (fortran95)

Code generation

- Diagram generation:

  QGRAF [Nogueira 92]
- Algebra:

  FORM [Vermaseren 91]

  SPINNEY [Cullen, Koch-Janusz, Reiter 10]
- Code generator:

  HAGGIES [Reiter 09]

Generated code execution

- Loop integral reduction:

  SAMURAI [Mastrolia, Ossola, Reiter, Tramontano 10]

  GOLEM95 [Binoth, Cullen, Guillet, Heinrich, Pilon, Reiter 08]

  PJFRY [Yundin]
- Scalar integral evaluation:

  AVHOLO [van Hameren]

  QCDLOOP [Ellis, Zanderighi]

  GOLEM95C [Cullen, Guillet, Heinrich, Kleinschmidt, Pilon, Reiter, Rodgers 11]

Reduction methods

- SAMURAI
  [Mastrolia, Ossola, Reiter, Tramontano 10]

Reduction method can be choosen at runtime

- Tensorial integrand-level reconstruction
  [Heinrich, Ossola, Reiter, Tramontano 10]

  with

  - GOLEM95C [Binoth, Cullen, Guillet, Heinrich, Kleinschmidt, Pilon, Reiter, Rodgers 11]
  - SAMURAI [Mastrolia, Ossola, Reiter, Tramontano 10]
  - PJFry [Yundin]

A Walk through GoSam…

- GoSam as a standalone code

- Interfacing with an external Monte Carlo program:
  - The BLHA-interface
  - An example with Sherpa

GoSam as standalone code

- Preparation of the input card “myprocess.rc“:

```
#!/bin/env /mt/home/luisonig/bin/gosam.py
process_name=ttH
process_path=./ttH_virtual
#### physics options  #####
in=g,g                 # accepts also PDG codes
out=H,t,t~
order=gs, 2, 4
model=smdiag
model.options= masses: mT mH, width: none, \
    alpha: 0.0072973525376, mZ: 92.0584, mW: 80.376, \
    mT: 172.6, mH: 130.0, Nf:5, Nfgen:1
zero=mU,mD,mC,mS,mB,wT,wB,wW,wZ,wH
one=gs,e
symmetries=family,generation
helicities=[+-][+-]0[+-][+-]
qgraf.options=onshell,notadpole,nosnail
qgraf.verbatim=true=iprop[D,S,C,B, 0, 0]
```

Look for example at Higgs-top-antitop production

Models can be imported from FeynRules or LanHEP

Specify which parameters should be set ALGEBRAICALLY to zero or one

Options on helicity and loop diagrams

GoSam as standalone code

- Preparation of the input card "myprocess.rc" (continued):

```
#### program options ####

extensions=samurai, golem95, dred

# abbrev.level=helicity  # group , diagram
# abbrev.limit=0

form.bin=tform
form.tempdir=/tmp
fc.bin=gfortran -O2

samurai.fcflags=-I${HOME}/include/samurai
samurai.ldflags=-L${HOME}/lib/ -lqcdloop -lavh_olo \
                -lsamurai
samurai.version=2.1.1

golem95.fcflags=-I${HOME}/include/golem95
golem95.ldflags=-L${HOME}/lib/ -lgolem
```

Several other extension and options available.
For further details check our user manual:
http://www.hepforge.org/archive/gosam/gosam-1.0.pdf

GoSam as standalone code

- Generate code and compile

`$ gosam.py myprocess.rc` python code generates fortran95 code

`$ make source` Form & Haggies process diagrams to write code

`$ make compile` fortran95 code in compiled

```
ggttH : makevirt
File  Edit  View  Scrollback  Bookmarks  Settings  Help
luisonig@D22:ggttH$ ./makevirt
---> Creating code for virtual part...
---> Generating code for amplitudes...
make -f Makefile.source source
make[1]: Entering directory `/scratch/luisonig/GoSam_Processes/ttH/ggttH/ttH_virtual'
make[2]: Entering directory `/scratch/luisonig/GoSam_Processes/ttH/ggttH/ttH_virtual/doc'
make[2]: Nothing to be done for `source'.
make[2]: Leaving directory `/scratch/luisonig/GoSam_Processes/ttH/ggttH/ttH_virtual/doc'
make[2]: Entering directory `/scratch/luisonig/GoSam_Processes/ttH/ggttH/ttH_virtual/common'
FORM is generating color.txt
  0.01 sec + 0.13 sec: 0.15 sec out of 0.10 sec
haggies is generating color.f90
haggies is generating model.f90
FORM is generating version.out
  0.00 sec + 0.00 sec: 0.00 sec out of 0.00 sec
haggies is generating version.f90
make[2]: Leaving directory `/scratch/luisonig/GoSam_Processes/ttH/ggttH/ttH_virtual/common'
make[2]: Entering directory `/scratch/luisonig/GoSam_Processes/ttH/ggttH/ttH_virtual/helicity0'
Form is processing tree diagram 1 @ Helicity 0
  0.02 sec + 0.11 sec: 0.13 sec out of 0.08 sec
Form is processing tree diagram 2 @ Helicity 0
  0.01 sec + 0.11 sec: 0.13 sec out of 0.08 sec
Form is processing tree diagram 3 @ Helicity 0
  0.02 sec + 0.06 sec: 0.09 sec out of 0.06 sec
Form is processing tree diagram 4 @ Helicity 0
```

GoSam as standalone code

- Check produced code with automatic generated documentation before the full generation/run

Index	1	2	3	4	5
0	−	−	0	−	−
1	−	−	0	−	+
2	−	−	0	+	−
3	−	−	0	+	+
4	−	+	0	−	−
5	−	+	0	−	+
6	−	+	0	+	−
7	−	+	0	+	+
$8 \to 4$	+	−	0	−	−
$9 \to 5$	+	−	0	−	+
$10 \to 6$	+	−	0	+	−
$11 \to 7$	+	−	0	+	+
12	+	+	0	−	−
13	+	+	0	−	+
14	+	+	0	+	−
15	+	+	0	+	+

GoSam 1.0: $gg \to Ht\bar{t}$

luisonig

2012-02-19 (22:10:59)

Abstract

This process consists of 8 tree-level diagrams and 160 NLO diagrams. Golem has identified 15 groups of NLO diagrams by analyzing their one-loop integrals.

- Documentation contains information about
  - the generated helicities
  - the colour basis

GoSam as standalone code

- Loop diagrams are grouped into sets of diagrams which share loop propagators

5.4 Group 3 (5-Point)

General Information

The maximum effective rank in this group is 4.

$$r_1 = -k_2 + k_5, \quad m_1 = m_t$$
$$r_2 = -k_2$$
$$r_3 = 0$$
$$r_4 = -k_4, \quad m_4 = m_t$$
$$r_5 = -k_3 - k_4, \quad m_5 = m_t$$

$$S = \begin{pmatrix} S_{1,1} & 0 & S_{1,3} & S_{1,4} & S_{1,5} \\ 0 & 0 & 0 & S_{2,4} & S_{2,5} \\ S_{3,1} & 0 & 0 & 0 & S_{3,5} \\ S_{4,1} & S_{4,2} & 0 & S_{4,4} & S_{4,5} \\ S_{5,1} & S_{5,2} & S_{5,3} & S_{5,4} & S_{5,5} \end{pmatrix}$$

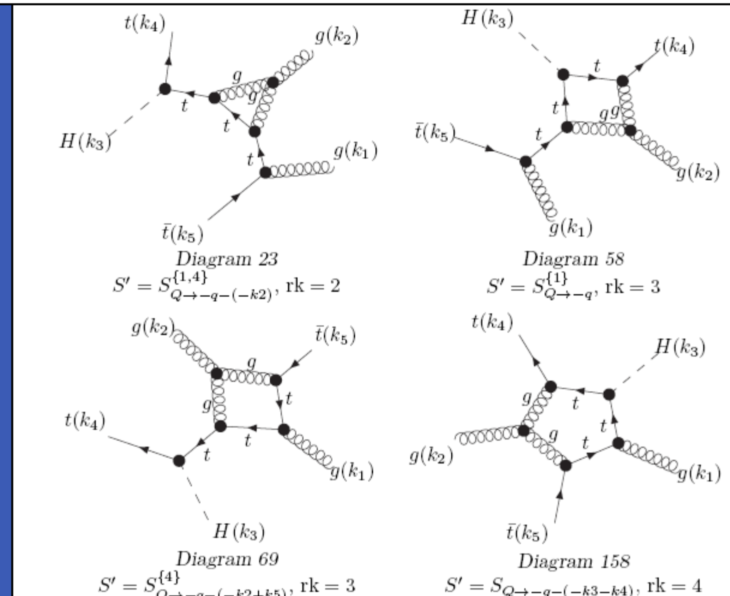
$$S_{1,1} = -2m_t^2$$
$$S_{1,3} = -s_{51} + s_{34} - s_{12}$$
$$S_{1,4} = -2m_t^2 + s_{45} + m_H^2 - s_{23} - s_{12}$$
$$S_{1,5} = -2m_t^2$$
$$S_{2,4} = s_{51} - s_{23} - s_{34} + m_H^2$$
$$S_{2,5} = s_{51} - m_t^2$$
$$S_{3,5} = -m_t^2 + s_{34}$$
$$S_{4,4} = -2m_t^2$$
$$S_{4,5} = -2m_t^2 + m_H^2$$
$$S_{5,5} = -2m_t^2$$

Loop diagrams are grouped into sets of diagrams which share loop-propagators. A loop integral can be written as

$$\int \frac{\mathrm{d}^n k}{i\pi^{\frac{n}{2}}} \frac{\mathcal{N}(q)}{\prod_{j=1}^{N} \left[(k+r_j)^2 - m_j^2 - i m_j \Gamma_j + i\delta\right]} \tag{16}$$

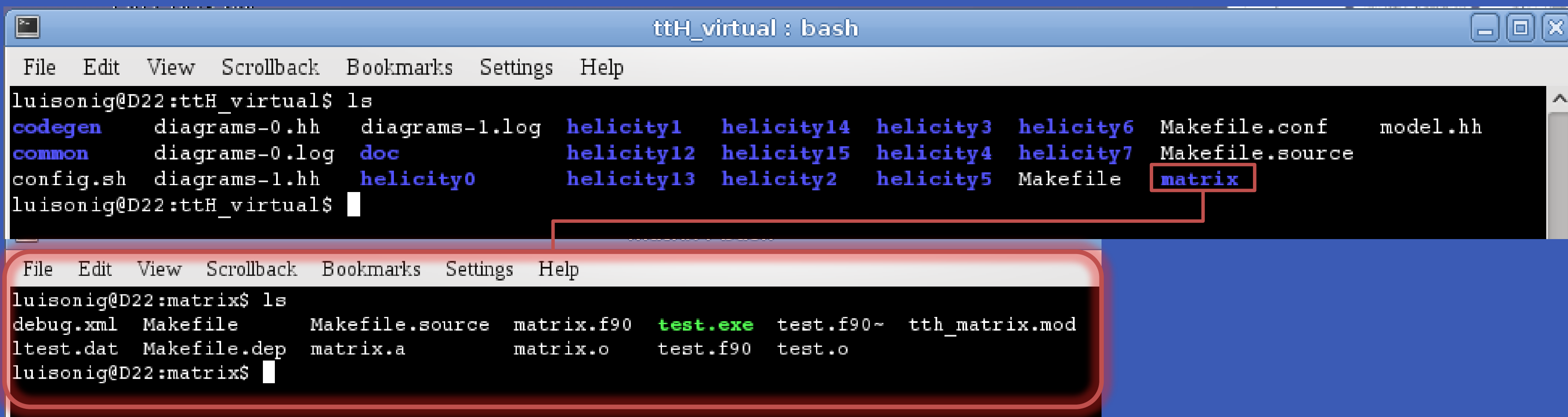
For each group we list r_j, m_j and Γ_j. For m_j and Γ_j only non-vanishing symbols are listed. Furthermore, we give the matrix S which is defined as

$$S_{\alpha\beta} = (r_\alpha - r_\beta)^2 - m_\alpha^2 - i m_\alpha \Gamma_\alpha - m_\beta^2 - i m_\beta \Gamma_\beta. \tag{17}$$



GoSam as standalone code

- When the full code is ready:
  - Contributions divided in directories by helicity



➤Output of *test.exe*:

```
#                   LO:   0.69180834378626 26E-04
# NLO, finite part:   17.78339386183546
# NLO, single pole:  -9.484483151742308
# NLO, double pole:  -6.000000000000000
# IR,  single pole:  -9.484483151741490
# IR,  double pole:  -5.999999999999999
#  Time/Event [ms]:                 280.000
```

Generated code comes with **`test.exe`** routine which allows to check cancellation of poles with built-in integrated dipoles: **make test.exe**

Example: pp → H t $\bar{t}$

Generation time: 1h 20min   Compilation time: 3h 6min   Time for 1 PS point: 280 ms

Machine: Intel Core Quad CPU Q6600 @ 2.4 GHz / 6 GB RAM

Process generated in DRED and converted to CDR at runtime

	E	p_x	p_y	p_z
u/g	250.0	0.0	0.0	250.0
$\bar{u}/g$	250.0	0.0	0.0	-250.0
H	136.35582793693018	15.133871809486299	27.986733991031045	26.088703626953386
t	181.47665951104506	20.889486679044587	-50.105625289561424	14.002628607367491
$\bar{t}$	182.16751255202476	-36.023358488530903	22.118891298530357	-40.091332234320859

parameters			
$\sqrt{s}$	500.0	N_f	5
μ	m_t	$N_{f,h}$	1
m_t	172.6	α_s	0.1076395107858145
m_H	130	v	246.21835258713082

result $gg \to t\bar{t}H$	GoSam	Ref. [39]
$a_0 \cdot 10^5$	6.127399805961155	6.127400074872043
c_0/a_0	9.006680638719660	9.006680836410272
c_{-1}/a_0	2.986347664537282	2.9863477301662056
c_{-2}/a_0	−6.000000000000004	−6.000000131659877

Comparison with MadLoop [Hirschi,Frederix,Frixione,Garzelli, Maltoni,Pittau 11]

GoSam: further tested calculations

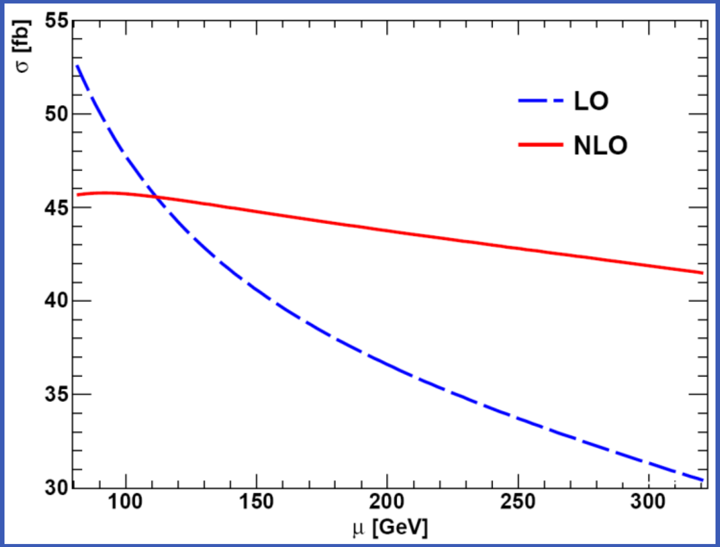
- $q\bar{q} \longrightarrow b\bar{b}b\bar{b}$
- $g\bar{g} \longrightarrow b\bar{b}b\bar{b}$
- $q\bar{q} \longrightarrow t\bar{t}b\bar{b}$
- $g\bar{g} \longrightarrow t\bar{t}b\bar{b}$
- $u\bar{d} \longrightarrow W^+ ggg$
- $u\bar{u} \longrightarrow H\, t\bar{t}$
- $g\bar{g} \longrightarrow H\, t\bar{t}$
- $u\bar{d} \longrightarrow W + s\bar{s} \longrightarrow e^+\nu_e s\bar{s}$
- $u\bar{d} \longrightarrow W + gg \longrightarrow e^+\nu_e gg$
- $d\bar{d} \longrightarrow Z\, gg \longrightarrow e^+e^- gg$
- $u\bar{d} \longrightarrow W + b\bar{b} \longrightarrow e^+\nu_e b\bar{b}$ with massive b's
- $u\bar{d} \longrightarrow W + g \longrightarrow e^+\nu_e g$ EW corrections
- $e^+e^- \longrightarrow Z \longrightarrow d\bar{d}g$
- $e^+e^- \longrightarrow Z \longrightarrow b\bar{b}g$ with massive b's
- $\gamma\gamma \longrightarrow \gamma\gamma\gamma\gamma$
- $u\bar{d} \longrightarrow W^+W^+ s\bar{c} \longrightarrow e^+\nu_e\mu^+\nu_\mu s\bar{c}$
- $u\bar{u} \longrightarrow W^+W^+ c\bar{c} \longrightarrow e^+\nu_e\mu^+\nu_\mu c\bar{c}$
- $u\bar{d} \longrightarrow W^+W^+ \bar{s}c \longrightarrow e^+\nu_e\mu^+\nu_\mu \bar{s}c$
- plus a large number of 2 to 2 processes

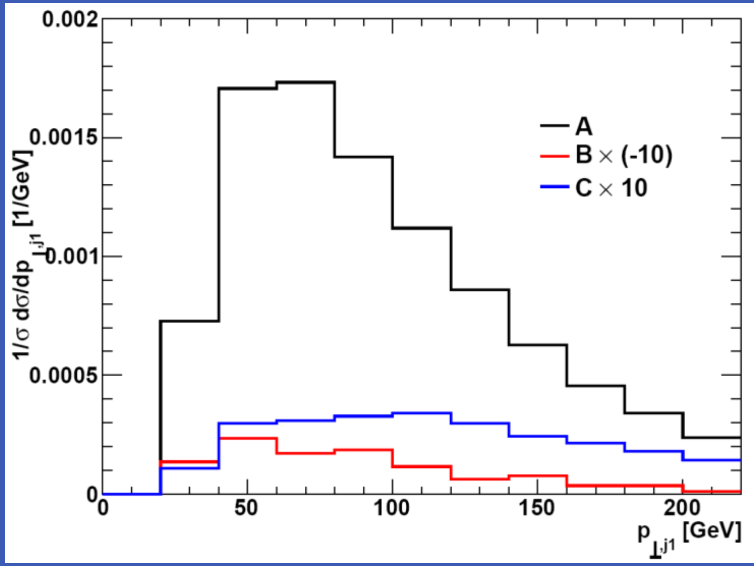
W^+W^- + 2 jets @ NLO with GoSam

[Greiner, Heinrich, Mastrolia, Ossola, Reiter, Tramontano 12]

- First application of GoSam for the calculation of previously unknown NLO contributions:
  - Part A: no 3rd gen. quarks in fermion loops and VB attached to closed fermion loops, [Melia, Melnikov, Rontsch, Zanderighi 11]
  - Part B: VB attached to closed fermion loops,
  - Part C: 3rd gen. quarks in the loops.

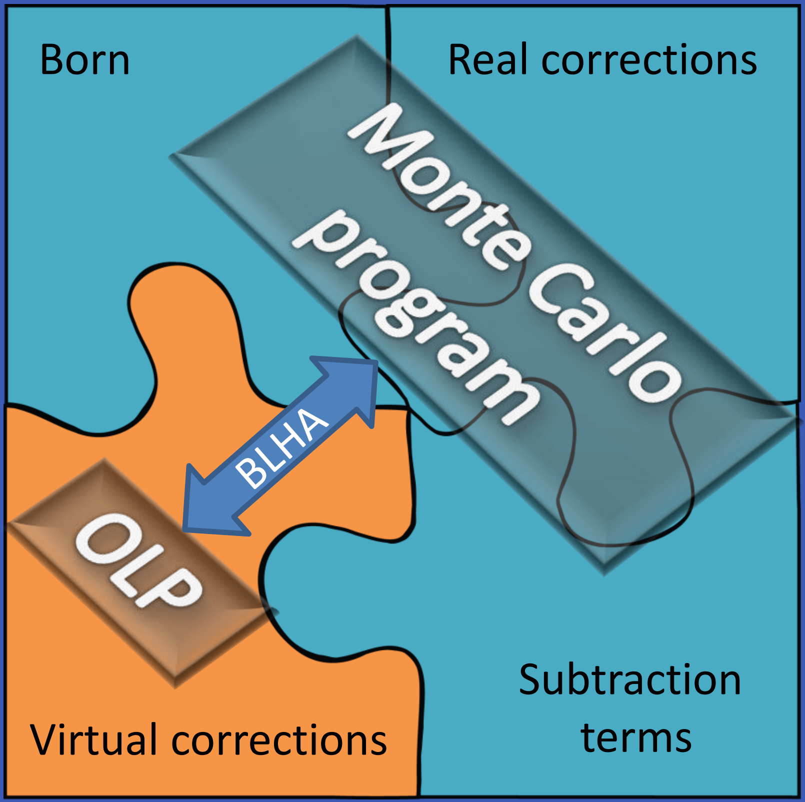
  → previously unknown (Parts B and C)



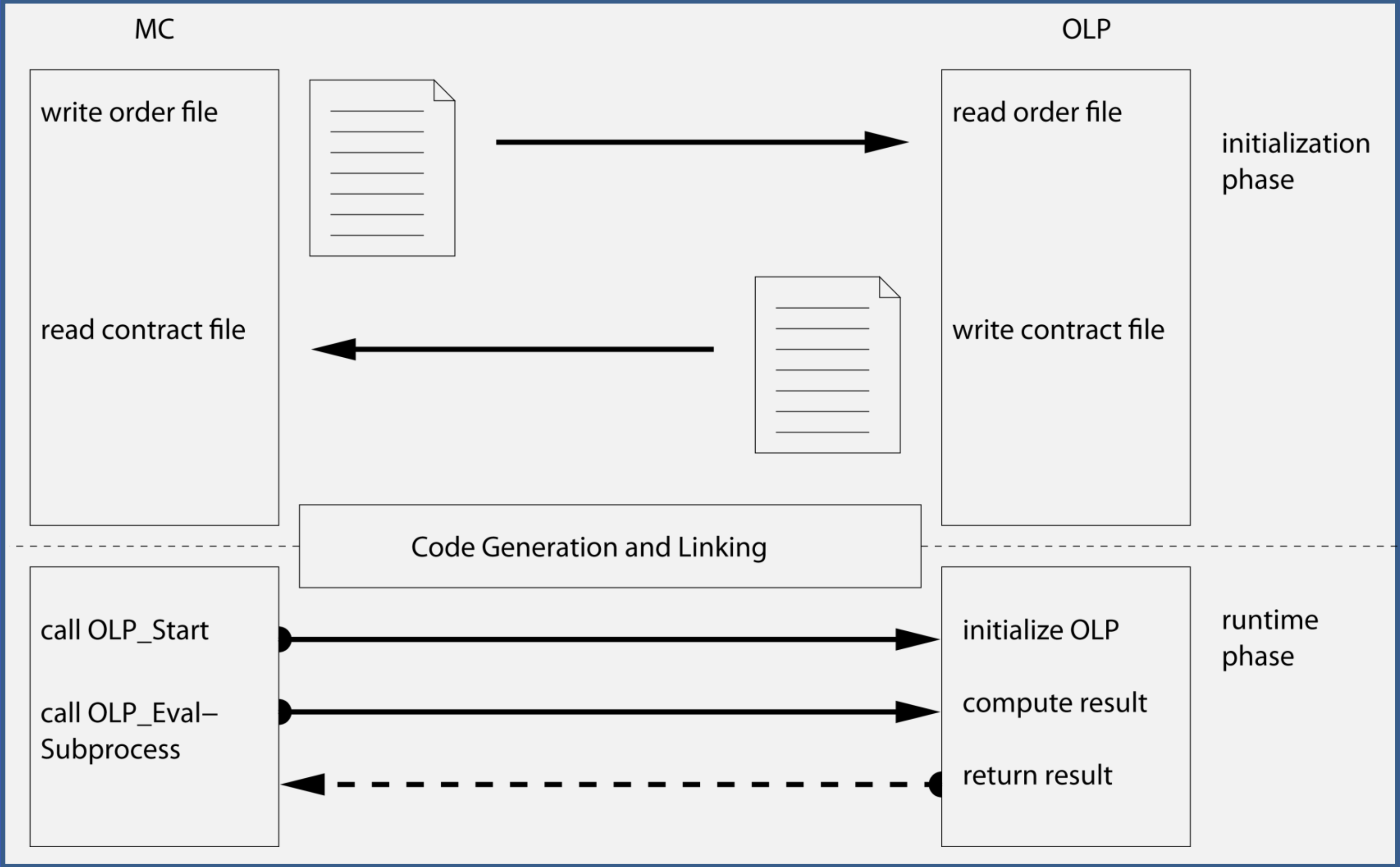


GoSam: interface with MC

- GoSam supports the Binoth-Les-Houches-Accord (BLHA) standards to interface with Monte Carlo generators:
  - Monte Carlo program:
    Born / real corr. / sub. terms
  - One-loop Program (OLP):
    virtual corr.
  - Pre-runtime comunication via “order“ and “contract“ files
  - At runtime:
    - `OLP_Start()`
    - `OLP_EvalSubProcess()`



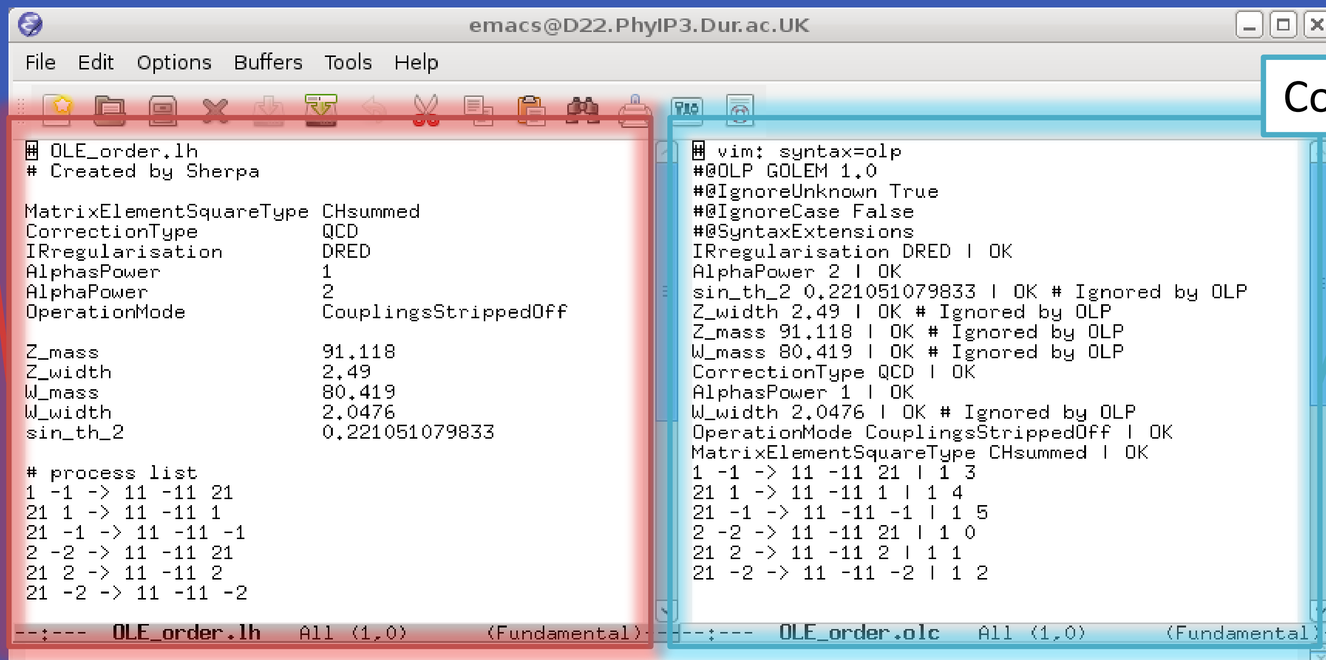
BLHA-interface: order & contract

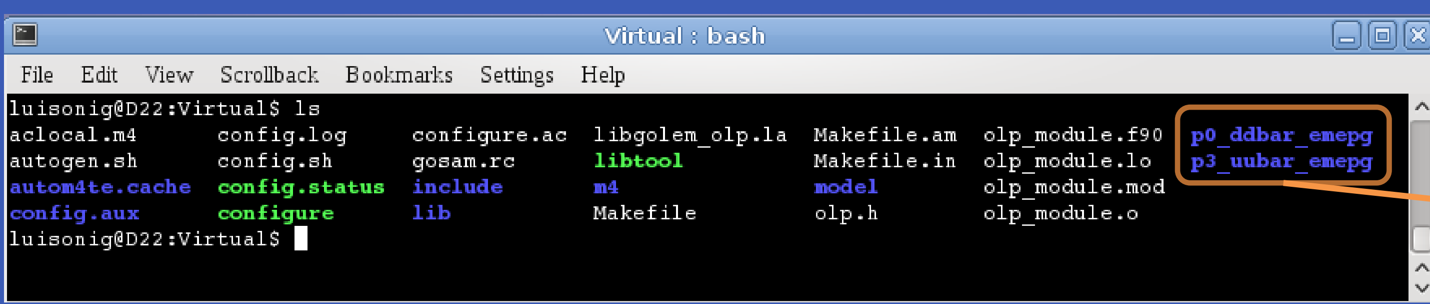


In practice: GoSam+ Sherpa

Order file

Contract file





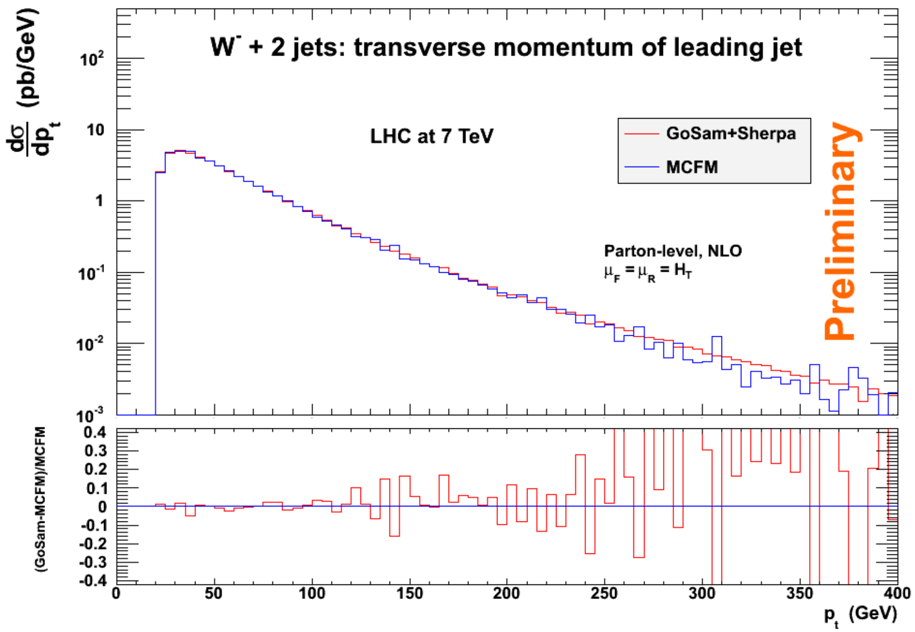
GoSam produces only the code strictly needed avoiding redundancies and exploiting crossing-symmetry

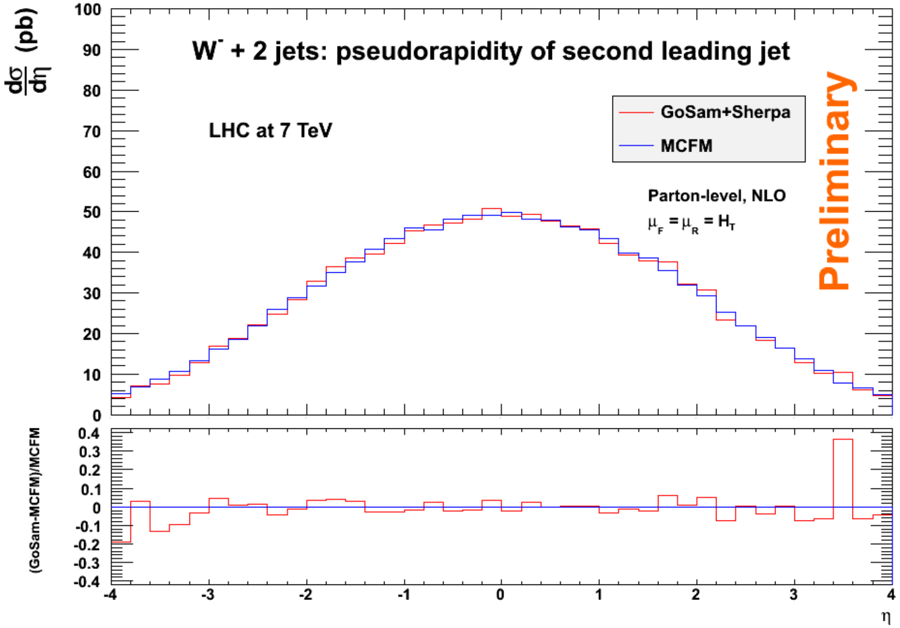
In practice: GoSam+ Sherpa

- Few steps needed to compute e.g. Z+1 jet @NLO:
  - Prepare Sherpa card according to your need and run it once
    - The “order“ file and the necessary tree-level code is generated
  - Run GoSam feeding the “order“ file and a configuration file with further needed inputs (paths / filtering options / …)
    - Run GoSam with “autotools“ extension will make it easy to produce a dynamical library which can be linked to Sherpa
  - After the virtual code is set up, generate and compile it with `configure / make / make install`
  - The produced library `libgolem_olp.so` must be added to the `SHERPA_LDADD` option in the Sherpa card

➡ HAVE FUN WITH PHENOMENOLOGY

GoSam+Sherpa vs MCFM: W^- + 2 jets





TIMINGS:

Set-up

Virtual: < 1 min 30 sec

Generation & Compilation

Virtual: ~ 2h 30 min

Running

Born : ~20 min

Real : ~ 18h 30 min

Virtual: ~ 13h

Machine Intel(R) Core(TM)2 Quad CPU Q6600 @ 2.40GHz

NUMBER OF EVENTS:

Born : 5'000'000 x 2

Real : 50'000'000 x 20

Virtual: 1'000'000 x 10

PHYSICS:

LHC 7 TeV

Cuts

pt_jet > 20 GeV

eta_jet < 4.0

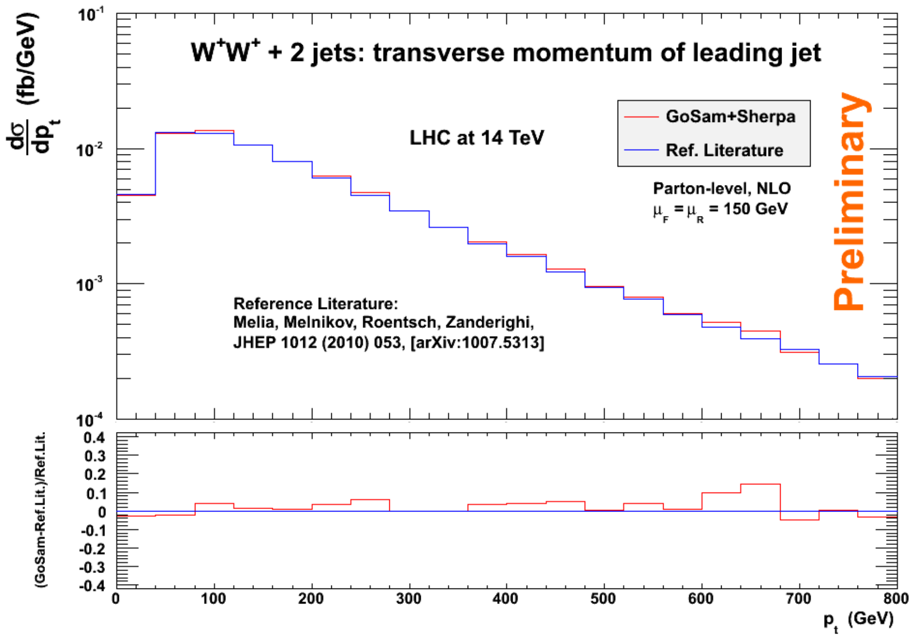
kt_alg, R=0.7

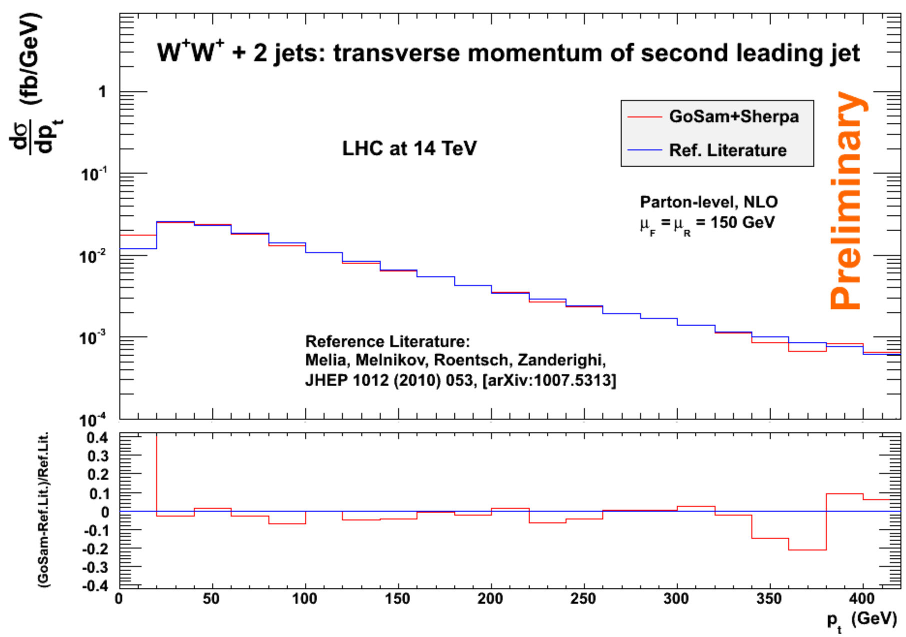
Scale H_T

PDFs CT10.LHgrid

IPPP, University of Durham

GoSam+Sherpa vs Melia et al.: W^+W^+ + 2 jets





TIMINGS:

Set-up
Virtual: < 1 min 30 sec

Generation & Compilation
Virtual: ~ 7h 25 min

Running
Born : ~ 30 min
Real : ~ 13h 30 min
Virtual: ~ 7h

Machine Intel(R) Core(TM)2 Quad CPU Q6600 @ 2.40GHz

NUMBER OF EVENTS:

Born : 1'000'000 x 10
Real : 50'000'000 x 10
Virtual: 1'000'000 x 10

PHYSICS:

LHC 7 TeV

Cuts
pt_lep > 20 GeV
|eta_lep| < 2.4
pt_miss > 30 GeV
antikt_alg, R=0.4

Scale 150 GeV

PDFs MSTW2008nlo.LHgrid

IPPP, University of Durham

GoSam+ Sherpa: comments

- Automatized NLO calculations with GoSam+Sherpa
  - Two codes can be linked with very few commands
  - High level of automation and optimization in the generated code
  - Sherpa 1.4.0 (released yesterday!) has the BLHA-interface active and is fully equiped to interface with GoSam
  - Some ready-to-use packages for 2->2, 2->3, 2->4 processes to be used with Sherpa will soon become available (no need to run GoSam locally) [http://projects.hepforge.org/gosam/proc/]

Conclusions and Outlook

- GoSam is a code for the computation of one-loop multi-leg amplitudes
  - Based on Feynman diagrams
  - Uses D-dimensional reduction tecniques
  - Flexible and broadly applicable tool
  - Public
  - Easy to interface with MC event generator to perform full NLO calculations

http://projects.hepforge.org/gosam/

- Possibilities for precision studies using NLO parton-level matched with parton-shower and with hadronization effects just around the corner
  - Possible to steer everything by just editing a single input card

- We look forward to interfacing with other tools and performing NLO analyses for the LHC

Backup slides…

Reduction methods: Samurai [default]

[Mastrolia, Ossola, Reiter, Tramontano 10]

- OPP reduction algorithm [Ossola, Papadopoulos, Pittau 07]
- D-dimensional extension [Ellis, Giele, Kunszt, Melnikov 08]
- Coefficient of polynomials via DFT [Mastrolia et al. 08]
- Computation of the full rational term in one go [Internal GoSam algebraic handling]

For any one-loop amplitude:

$$\mathcal{A}_n = \int d^d\bar{q} \frac{\mathcal{N}(\bar{q},\epsilon)}{\bar{D}_0\bar{D}_1\cdots\bar{D}_{n-1}} \quad ; \quad \mathcal{N}(\bar{q},\epsilon) = N_0(\bar{q}) + \epsilon N_1(\bar{q}) + \epsilon^2 N_2(\bar{q})$$

$$\bar{D}_i = (\bar{q}+p_i)^2 - m_i^2 = (q+p_i)^2 - m_i^2 - \mu^2 \quad ; \quad \bar{\not{q}} = \not{q} + \not{\mu} \; ; \quad \bar{q}^2 = q^2 - \mu^2$$

Result of integration can be expressed as linear combination of scalar integrals: boxes, triangles, bubbles, tadpoles and rational terms

Integrals with μ^2 in the numerator

$$\mathcal{A}_n = \sum_{i_0<i_1<i_2<i_3}^{m-1} d(i_0i_1i_2i_3)D_0(i_0i_1i_2i_3) + \sum_{i_0<i_1<i_2}^{m-1} c(i_0i_1i_2)C_0(i_0i_1i_2) + \sum_{i_0<i_1}^{m-1} b(i_0i_1)B_0(i_0i_1) + \sum_{i_0}^{m-1} a(i_0)A_0(i_0) + \mathcal{R}$$

Reduction methods: Tensorial Reconstr.

[Heinrich, Ossola, Reiter, Tramontano 10]

- Tensorial reconstruction convoluted with tensor integrals:

Rewrite numerator function as linear combination of tensors

$$\mathcal{N}(q) = \sum_{r=0}^{R} C_{\mu_1 \ldots \mu_r} q_{\mu_1} \ldots q_{\mu_r}$$

$$C_{\mu_1 \ldots \mu_r} q_{\mu_1} \cdots q_{\mu_r} = \sum_{(i_1,i_2,i_3,i_4) \vdash r} \hat{C}^{(r)}_{i_1\, i_2\, i_3\, i_4} \cdot (q_1)^{i_1} (q_2)^{i_2} (q_3)^{i_3} (q_4)^{i_4}$$

Determine the coefficients by sampling in q_μ in a bottom-up approach

if $q_\mu = (x, y, z, w)$ then $\mathcal{N}(q) = \mathcal{N}(x, y, z, w)$

Level-0 $q = (0,0,0,0)$; $\mathcal{N}(0,0,0,0) \equiv \mathcal{N}^{(0)} = C_0$

Level-1 4 systems, each sampling a monomial depending on one component of q_μ only

$$\mathcal{N}^{(1)}(q) \equiv \mathcal{N}(q) - \mathcal{N}^{(0)}$$

$$q = (x,0,0,0) \Rightarrow \mathcal{N}^{(1)}(x,0,0,0) \equiv x\, C_1 + x^2\, C_{11} + \ldots + x^R\, C_{\underbrace{11\ldots1}_{R \text{ times}}}$$

$$q = (0,y,0,0) \Rightarrow \mathcal{N}^{(1)}(0,y,0,0) \equiv y\, C_2 + y^2\, C_{22} + \ldots + y^R\, C_{\underbrace{22\ldots2}_{R \text{ times}}}$$

Allows to avoid numerical instabilities due to vanishing Gram determinants

Derive & Numpolvec

- The latest version of GoSam also implements two new features to improve speed and precision:
  - **`derive`**: computes the numerator by expanding it in a Tayor series

$$\mathcal{N}(\hat{q}) = \mathcal{N}(0) + \hat{q}^{\mu} \frac{\partial}{\partial \hat{q}_{\mu}} \mathcal{N}(\hat{q})|_{q=0} + \frac{1}{2!} \hat{q}^{\mu} \hat{q}^{\nu} \frac{\partial}{\partial \hat{q}_{\mu}} \frac{\partial}{\partial \hat{q}_{\nu}} \mathcal{N}(\hat{q})|_{q=0} + \dots$$

    one-to-one correspondence between derivatives at $\hat{q} = 0$ and the coefficients of the tensor integrals

  - **`numpolvec`**: uses numerical polarization vectors for external massless gauge bosons
    - this allows to reduce the code by generating only few helicities

3-Steps to the Loop Amplitude

GOSAM

Code set-up

- Python program writes code files from template
  - QGRAPH writes structure of diagrams

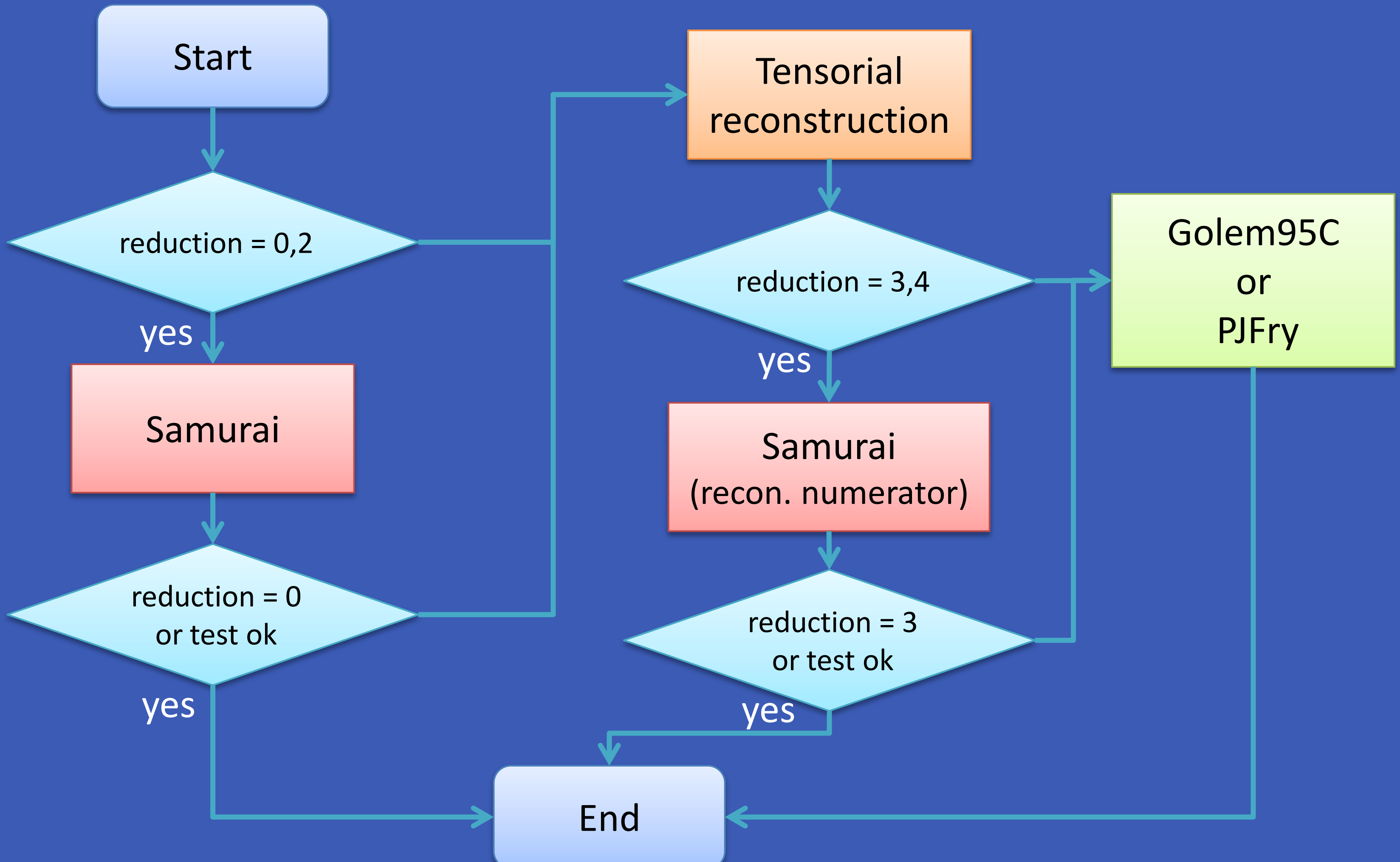
Code creation and compilation

- Fortran code is compiled and amplitudes are written
  - FORM & SPINNEY read and manipulates amplitudes
  - HAGGIES optimizes expressions and writes fortran code

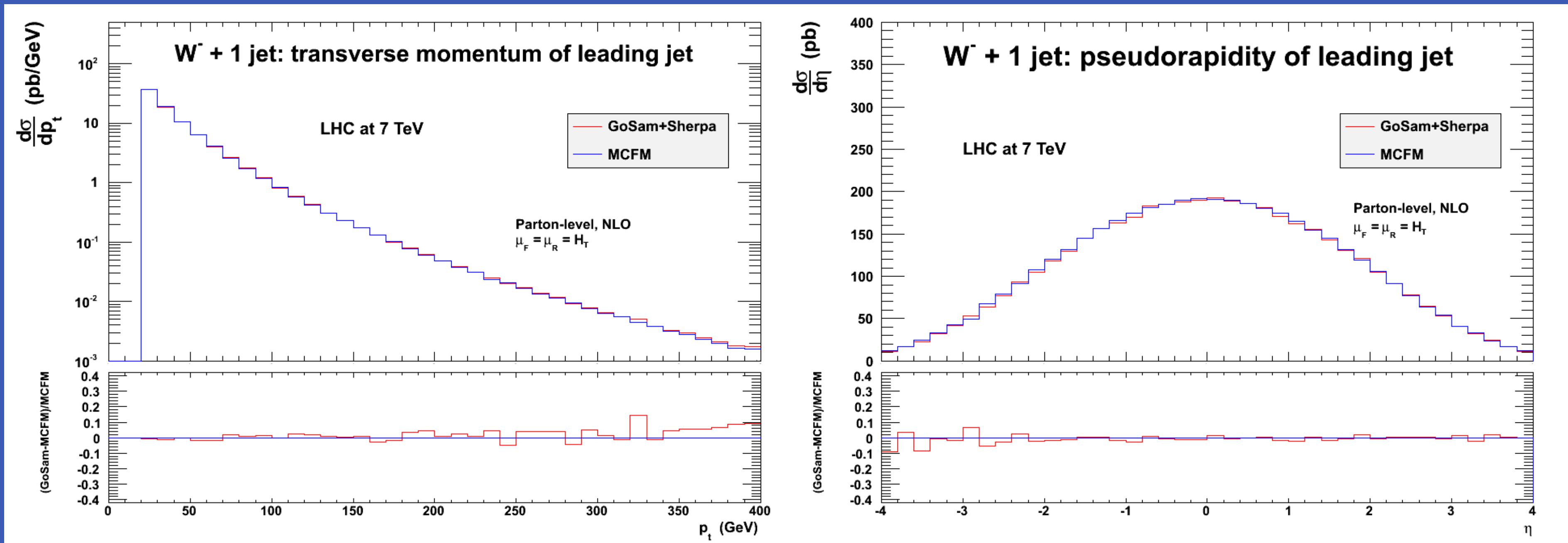
Code execution

- Fortran code computes amplitudes
  - SAMURAI/GOLEM95/… called to reduce integrals
  - AVHOLO/QCDLOOP/… called to evaluate scalar integrals

Reduction: strategies



GoSam+Sherpa vs MCFM: W+1 jet



TIMINGS:
Set-up
Virtual: < 10 sec
Generation & Compilation
Virtual: < 2 min
Running
Born : < 2h
Real : ~ 4h 20 min
Virtual: ~ 1h 10 min

Machine Intel(R) Core(TM)2 Quad CPU Q6600 @ 2.40GHz

NUMBER OF EVENTS:
Born : 50'000'000
Real : 50'000'000 x 5
Virtual: 5'000'000 x 5

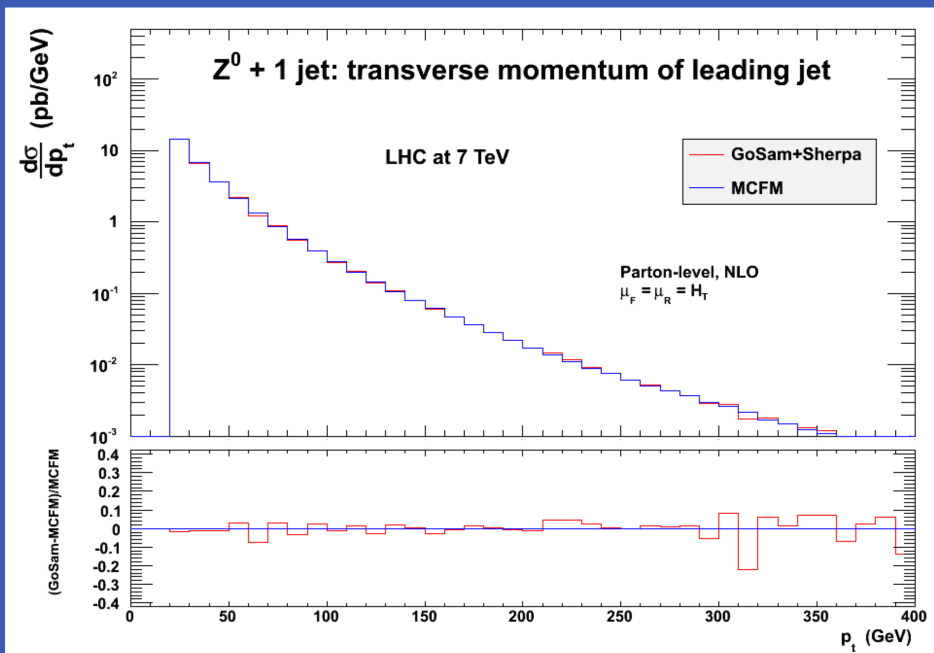
PHYSICS:
LHC 7 TeV

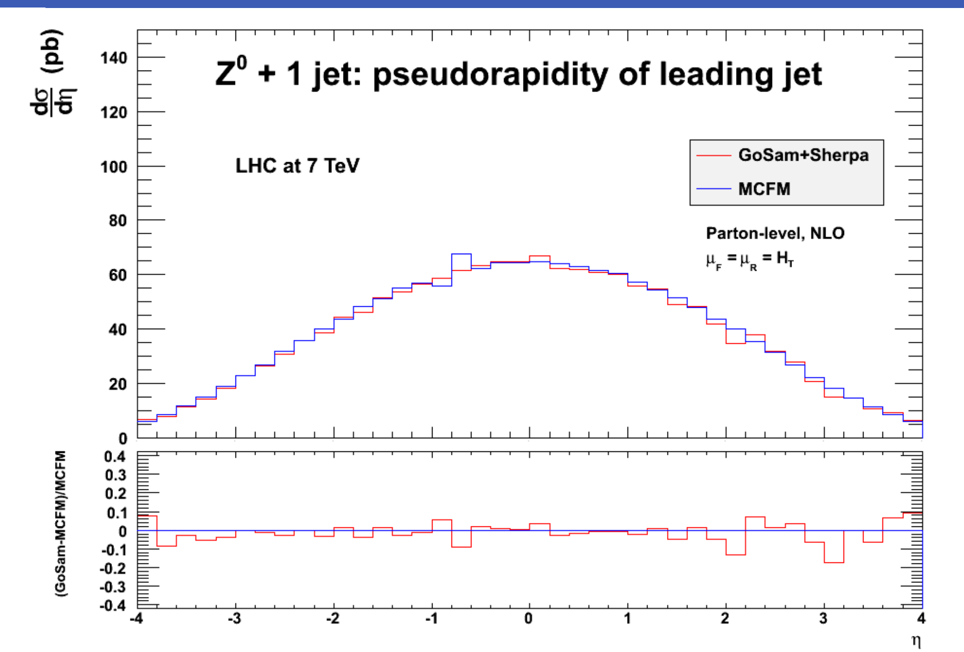
Cuts
pt_jet > 20 GeV
eta_jet < 4.0
kt_alg, R=0.7

Scale H_T

PDFs CT10.LHgrid

GoSam+Sherpa vs MCFM: Z+1 jet





TIMINGS:

Set-up

Virtual: < 20 sec

Generation & Compilation

Virtual: < 25 min

Running

Born : < 2h

Real : ~ 5h

Virtual: ~ 15h 30 min

Machine Intel(R) Core(TM)2 Quad CPU Q6600 @ 2.40GHz

NUMBER OF EVENTS:

Born : 5'000'000

Real : 50'000'000 x 6

Virtual: 5'000'000 x 6

PHYSICS:

LHC 7 TeV

Cuts

pt_jet > 20 GeV

eta_jet < 4.0

kt_alg, R=0.7

m(e+e-) > 15 GeV

Scale H_T

PDFs CT10.LHgrid

BSM physics with GoSam

- New models can be added via

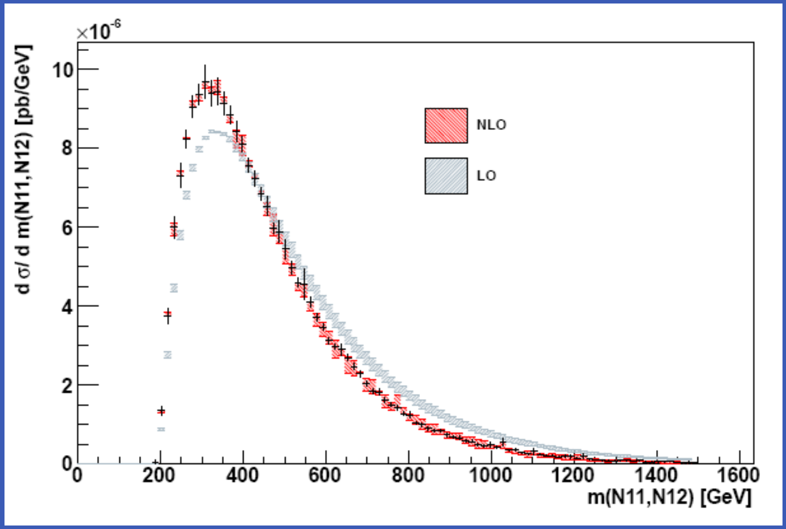
  FeynRules [Christensen, Duhr] LanHEP [Semenov]

- Allows to compute one-loop corrections also for BSM phenomenology

  - Example: $pp \longrightarrow \chi_0^1 \chi_0^1$ in MSSM

Veto: pt_jet > 20 GeV
eta_jet < 4.5

Scale Mz



[Figure by G.Cullen and N.Greiner]