Recent Progress in the GOLEM Project

T. Reiter (Nikhef)

in collaboration with
G. Cullen, A. Guffanti, J.P. Guillet, G. Heinrich, S. Karg,
N. Kauer, T. Kleinschmidt, E. Pilon, M. Rodgers, I. Wigmore

Loops & Legs 2010, Wörlitz, 25–30 April 2010

Overview

A General Setup for NLO Calculations

Tensor Integral Reduction with Golem95

An Automated 1-Loop Matrix Element Generator

GOLEM @ Work

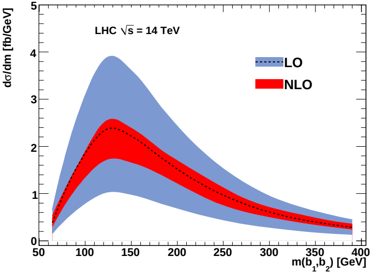
Conclusions

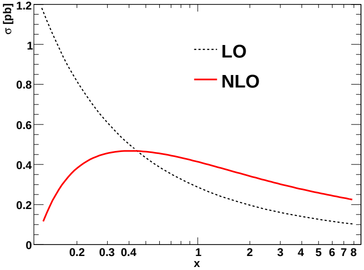
(Instead of a) Motivation

Three examples of NLO calculations — three things in common:

- Calculations have to deal with a large number of diagrams, cuts, subprocesses, . . .
- Computations make use of many different available and customized programs.
- NLO corrections are important
  - improvement of the theoretical error
  - for taking effects into account which are not present at LO

Example 1: $q\bar{q} \to b\bar{b}b\bar{b}$

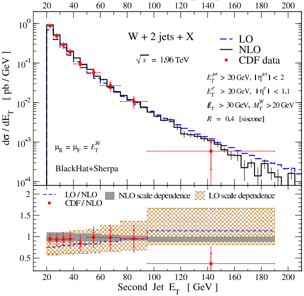


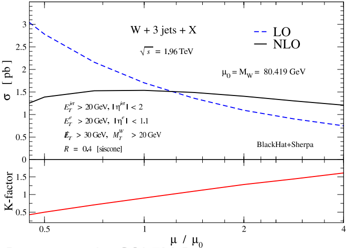


. . . by the GOLEM collaboration

- importance: BSM Higgs searches
- complexity: 2 → 4, massless QCD, ≈ 250 loop diagrams, 128 real emission diagrams
- tools: golem-2.0, golem95, MadGraph/MadEvent, MadDipole, Whizard [Binoth, Cullen, Greiner, Guffanti, Guillet, Heinrich, Pilon, TR, Reuter; Maltoni, Stelzer; Frederix, Gehrmann, Greiner; Kilian, Ohl, Reuter, Moretti]

Example 2: $pp \to W^\pm + jjj$





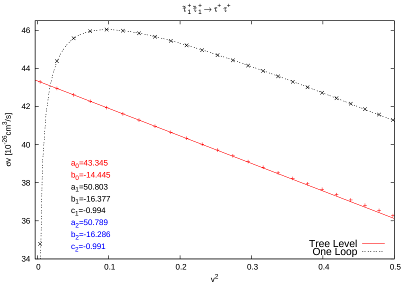
. . . by the BlackHat collaboration

- importance: background for SUSY searches
- complexity: 2 → 4, massless QCD, ≈ 2000 loop diagrams (+ crossings), ≈ 500 real emission diagrams (+ crossings),
- tools: BlackHat, Sherpa

[Berger, Bern, Dixon, Febres Cordero, Forde, Gleisberg, Ita, Kosower, Maitre; Archibald, Gleisberg, Hoche, Krauss, Schonherr, Schumann, Siegert, Winter]

→ see D. Maitre's talk

Example 3: relic density of DM in the MSSM



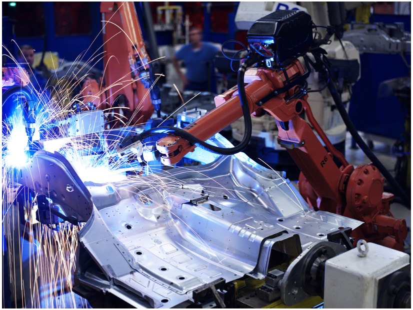
. . . by the SloopS collaboration

- importance: DM searches, precision cosmology
- complexity: Several 2 → 2, 2 → 3 and some 2 → 4 processes, SM and MSSM
- tools: SloopS, FeynArts, FormCalc, GRACE-loop, GRACE-susy, LanHEP, CompHEP [Baro, Boudjema, Semenov; Hahn, Perez-Victoria, Schappacher; Fujimoto et al.; Pukhov et al.]

A General Setup for NLO Calculations

The examples show

- high complexity of calculation
- automated approach needed
- challenges:
    - numerical stability
    - speed
    - resource usage
    - bookkeeping



Exploit modularity to split up the problem.

A General Setup for NLO Calculations

$$\sigma = \alpha_i^n \left(\sigma_0 + \alpha_i \sigma_1 + \mathcal{O}(\alpha_i^2)\right)$$

$$\sigma_0 = \int_{2\to N} \mathrm{d}\Phi(\{p\}) \sum_{\vec{\lambda},i,j} \langle c_i | \, C_{ij} \, | c_j \rangle \, \mathcal{A}^\dagger_{i,\mathcal{B}}(\{p\},\{\lambda\}) \mathcal{A}_{j,\mathcal{B}}(\{p\},\{\lambda\})$$

$$\sigma_1 = \int_{2\to N} \mathrm{d}\Phi(\{p\}) \sum_{\vec{\lambda},i,j} \langle c_i | \, C_{ij} \, | c_j \rangle \, \mathcal{A}^\dagger_{i,\mathcal{B}}(\{p\},\{\lambda\}) \mathcal{A}_{j,\mathcal{V}}(\{p\},\{\lambda\}) + h.c.$$

$$+ \int_{2\to N+1} \mathrm{d}\Phi(\{p'\}) \sum_{\vec{\lambda},i,j} \langle c'_i | \, C'_{ij} \, | c'_j \rangle \, \mathcal{A}^\dagger_{i,\mathcal{R}}(\{p'\},\{\lambda\}) \mathcal{A}_{j,\mathcal{R}}(\{p'\},\{\lambda\})$$

- renormalisation removes UV poles in $\mathcal{A}_\mathcal{V}$
- $\mathcal{A}_\mathcal{V}$ also contains IR poles in $1/(d-4)$
- $\mathcal{A}_\mathcal{R}$ contains singularities for $p_i \propto p_j$ and $p_i \to 0$
- singularities cancel between both contributions [Kinoshita,Lee,Nauenberg]

A General Setup for NLO Calculations

$$\sigma = \alpha_i^n \left(\sigma_0 + \alpha_i \sigma_1 + \mathcal{O}(\alpha_i^2)\right)$$

$$\sigma_0 = \int_{2\to N} \mathrm{d}\Phi(\{p\}) \sum_{\vec{\lambda},i,j} \langle c_i|\, C_{ij}\, |c_j\rangle\, \mathcal{A}^\dagger_{i,\mathcal{B}}(\{p\},\{\lambda\}) \mathcal{A}_{j,\mathcal{B}}(\{p\},\{\lambda\})$$

$$\sigma_1 = \int_{2\to N} \mathrm{d}\Phi(\{p\}) \sum_{\vec{\lambda},i,j} \langle c_i|\, C_{ij}\, |c_j\rangle\, \mathcal{A}^\dagger_{i,\mathcal{B}}(\{p\},\{\lambda\}) \mathcal{A}_{j,\mathcal{V}}(\{p\},\{\lambda\}) + h.c.$$

$$+ \int_{2\to N+1} \mathrm{d}\Phi(\{p'\}) \sum_{\vec{\lambda},i,j} \langle c'_i|\, C'_{ij}\, |c'_j\rangle\, \mathcal{A}^\dagger_{i,\mathcal{R}}(\{p'\},\{\lambda\}) \mathcal{A}_{j,\mathcal{R}}(\{p'\},\{\lambda\})$$

- renormalisation removes UV poles in $\mathcal{A}_\mathcal{V}$
- $\mathcal{A}_\mathcal{V}$ also contains IR poles in $1/(d-4)$
- $\mathcal{A}_\mathcal{R}$ contains singularities for $p_i \propto p_j$ and $p_i \to 0$
- singularities cancel between both contributions [Kinoshita,Lee,Nauenberg]

A General Setup for NLO Calculations (cont'd)

Appropriate subtraction terms render both parts finite:

$$
\begin{aligned}
\sigma_1 = \int_{2\to N} \mathrm{d}\Phi(\{p\}) \Bigg[& \sum_{\vec{\lambda},i,j} \left\langle c_i \right| C_{ij} \left| c_j \right\rangle \mathcal{A}^{\dagger}_{i,\mathcal{B}}(\{p\},\{\lambda\}) \mathcal{A}_{j,\mathcal{V}}(\{p\},\{\lambda\}) \\
& + h.c. + \int_1 \mathrm{d}\Phi^{(d)}(\{p\};q) \mathcal{M}_{\mathrm{sub}}(\{p\},q,\{\lambda\}) \Bigg] \\
+ \int_{2\to N+1} \mathrm{d}\Phi(\{p'\}) \Bigg[& \\
& \sum_{\vec{\lambda}',i,j} \left\langle c'_i \right| C'_{ij} \left| c'_j \right\rangle \mathcal{A}^{\dagger}_{i,\mathcal{R}}(\{p'\},\{\lambda'\}) \mathcal{A}_{j,\mathcal{R}}(\{p'\},\{\lambda'\}) \\
& - \mathcal{M}_{\mathrm{sub}}(\{p'\},\{\lambda'\}) \Bigg]
\end{aligned}
$$

A General Setup for NLO Calculations (cont'd)

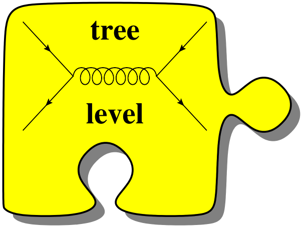
Modularity should be reflected in the setup of NLO calculations by independent software modules:

- Born (LO) cross section
- virtual corrections
- real emission
- IR subtraction

A General Setup for NLO Calculations (cont'd)

Modularity should be reflected in the setup of NLO calculations by independent software modules:

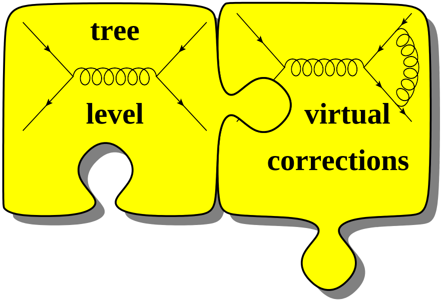
- Born (LO) cross section
- virtual corrections
- real emission
- IR subtraction



A General Setup for NLO Calculations (cont'd)

Modularity should be reflected in the setup of NLO calculations by independent software modules:

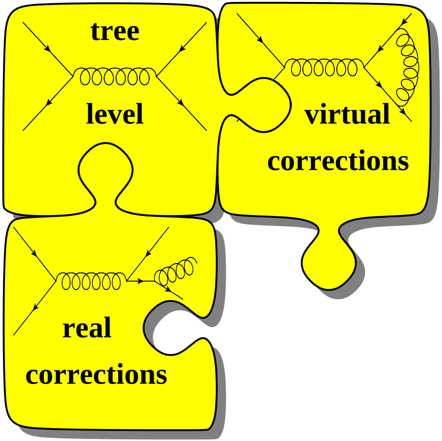
- Born (LO) cross section
- virtual corrections
- real emission
- IR subtraction



A General Setup for NLO Calculations (cont'd)

Modularity should be reflected in the setup of NLO calculations by independent software modules:

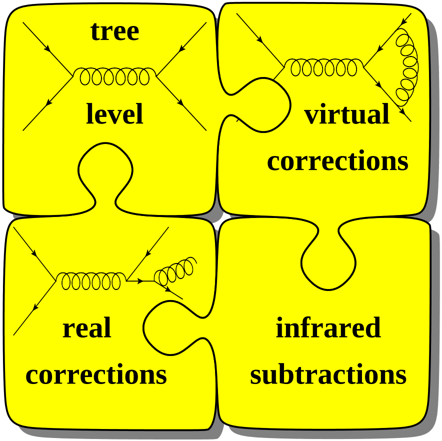
- Born (LO) cross section
- virtual corrections
- real emission
- IR subtraction



A General Setup for NLO Calculations (cont'd)

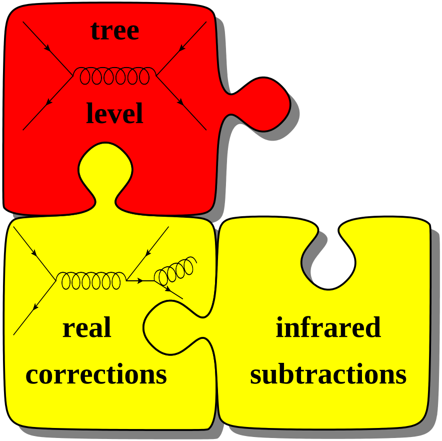
Modularity should be reflected in the setup of NLO calculations by independent software modules:

- Born (LO) cross section
- virtual corrections
- real emission
- IR subtraction



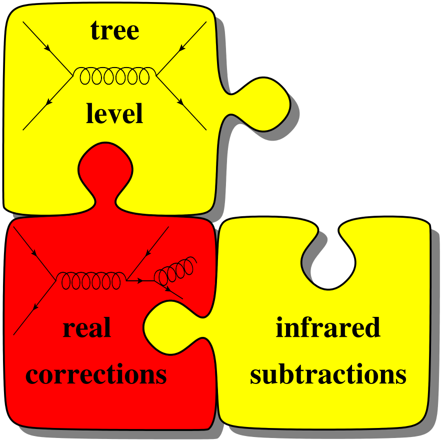
Related Work: Tree Level parts

- Many highly automated tools available
  - ALPGEN [Mangano et al.]
  - CalcHEP [Pukhov,Belyaev,Christensen]
  - CompHEP [Boos et al.]
  - FeynArts/FormCalc [Hahn,Perez-Victoria]
  - Grace [Yuasa et al.]
  - HELAC-PHEGAS [Papadopoulos et al.]
  - MadGraph [Maltoni,Stelzer]
  - Sherpa (AMEGIC) [Krauss,Kuhn,Soff]
  - Whizard (O'Mega) [Kilian,Moretti,Ohl,Reuter]
- Applicable to real part



Related Work: Tree Level parts

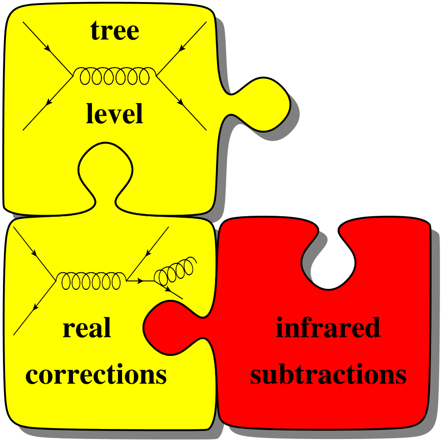
- Many highly automated tools available
    - ALPGEN [Mangano et al.]
    - CalcHEP [Pukhov,Belyaev,Christensen]
    - CompHEP [Boos et al.]
    - FeynArts/FormCalc [Hahn,Perez-Victoria]
    - Grace [Yuasa et al.]
    - HELAC-PHEGAS [Papadopoulos et al.]
    - MadGraph [Maltoni,Stelzer]
    - Sherpa (AMEGIC) [Krauss,Kuhn,Soff]
    - Whizard (O'Mega) [Kilian,Moretti,Ohl,Reuter]
- Applicable to real part



Related Work: IR Subtraction

Implementations for IR subtraction:

- Catani-Seymour dipoles
  - AutoDipole
    [Hasegawa,Moch,Uwer]
  - HELAC-DIPOLE
    [Czakon,Papadopoulos,Worek]
  - MadDipole
    [Frederix,Gehrmann,Greiner]
  - Sherpa [Gleisberg,Krauss]
  - TevJet [Seymour,Tevlin]
- FKS dipoles
  - MadFKS
    [Frederix,Frixione,Maltoni,Stelzer]
  - ...

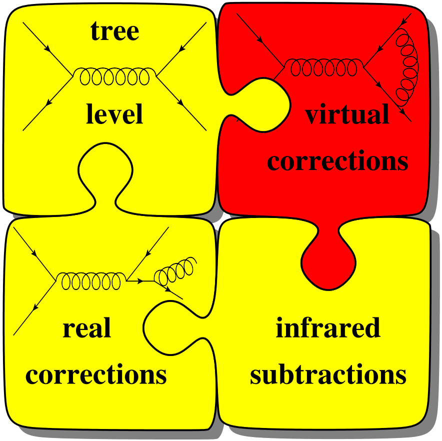


Virtual Corrections

- several approaches are discussed
- most of them are implemented
- nearly no public code
- not as automated as other parts of the calculation

GOLEM

- is Feynman diagram based
- is automatized
- will be public

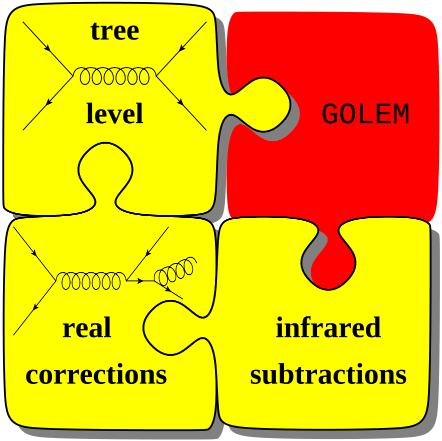


Virtual Corrections

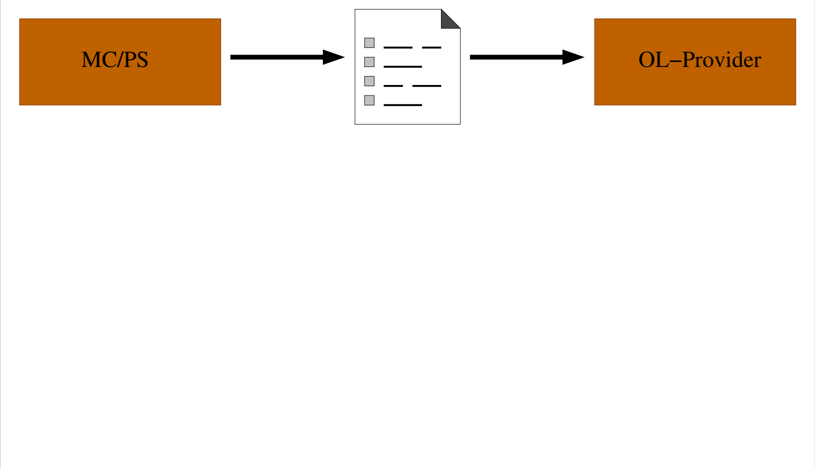
- several approaches are discussed
- most of them are implemented
- nearly no public code
- not as automated as other parts of the calculation

GOLEM

- is Feynman diagram based
- is automated
- will be public

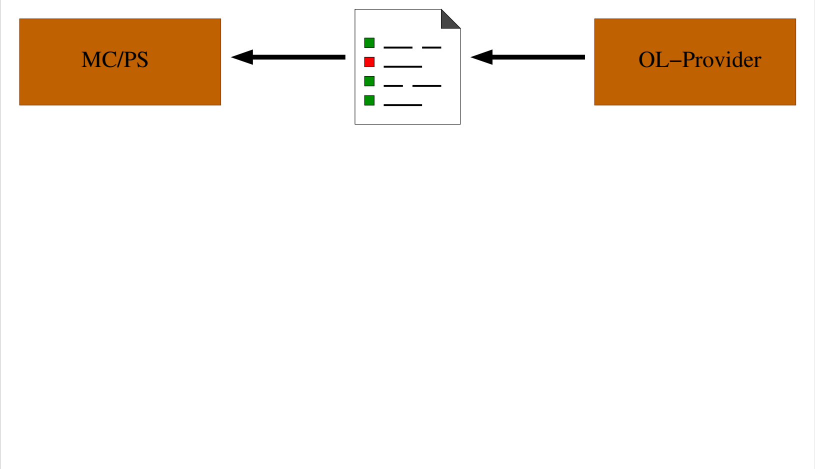


A General Interface for NLO Calculations



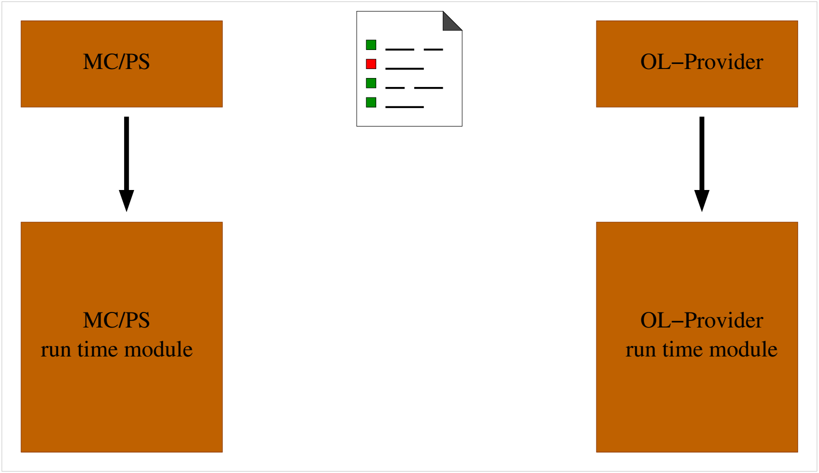
initialization: request for process (order file)

A General Interface for NLO Calculations



initialization: confirmation/rejection of request (contract file)

A General Interface for NLO Calculations



MC/PS

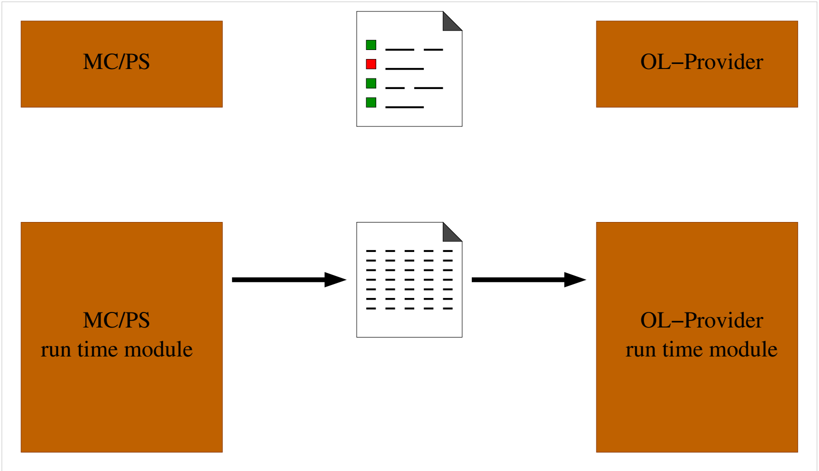
OL–Provider

MC/PS
run time module

OL–Provider
run time module

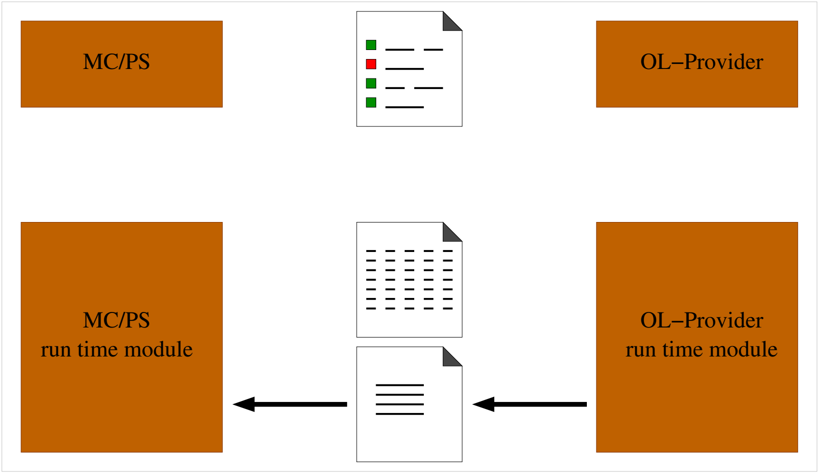
initialization: code generation (optional)

A General Interface for NLO Calculations



runtime phase: sending event

A General Interface for NLO Calculations



runtime phase: receiving results

A General Interface for NLO Calculations (cont'd)

- Specification available since Jan '10 [[1001.1307] Binoth et al.]
- Initialization: communication through files
  ⇒ language independent
- Runtime: API (at least C/C++ and Fortran)
- MC/PS
  - choose parameters, PDFs, approximations, . . .
  - choose amplitude format: colour and/or helicity summed, leading colour, . . .
  - sends events $(E, \vec{p}, m)_i, \alpha_s(\mu), \ldots$
- One-Loop Provider
  - evaluates interference term virtual × LO and LO × LO
  - performs UV-renormalisation
  - sends $1/\epsilon$ expansion (virtual part) and LO result of amplitude

A General Interface for NLO Calculations (cont'd)

- Specification available since Jan '10 [[1001.1307] Binoth et al.]
- Initialization: communication through files
  ⇒ language independent
- Runtime: API (at least C/C++ and Fortran)
- MC/PS
  - choose parameters, PDFs, approximations, ...
  - choose amplitude format: colour and/or helicity summed, leading colour, ...
  - sends events $(E, \vec{p}, m)_i, \alpha_s(\mu), \ldots$
- One-Loop Provider
  - evaluates interference term virtual × LO and LO × LO
  - performs UV-renormalisation
  - sends $1/\epsilon$ expansion (virtual part) and LO result of amplitude

A General Interface for NLO Calculations (cont'd)

- Specification available since Jan '10 [[1001.1307] Binoth et al.]
- Initialization: communication through files
  $\Rightarrow$ language independent
- Runtime: API (at least C/C++ and Fortran)
- MC/PS
  - choose parameters, PDFs, approximations, ...
  - choose amplitude format: colour and/or helicity summed, leading colour, ...
  - sends events $(E, \vec{p}, m)_i, \alpha_s(\mu), \ldots$
- One-Loop Provider
  - evaluates interference term virtual × LO and LO × LO
  - performs UV-renormalisation
  - sends $1/\epsilon$ expansion (virtual part) and LO result of amplitude

A General Interface for NLO Calculations (cont'd)

- Specification available since Jan '10 [[1001.1307] Binoth et al.]
- Initialization: communication through files
  $\Rightarrow$ language independent
- Runtime: API (at least C/C++ and Fortran)
- MC/PS
  - choose parameters, PDFs, approximations, ...
  - choose amplitude format: colour and/or helicity summed, leading colour, ...
  - sends events $(E, \vec{p}, m)_i, \alpha_s(\mu), \ldots$
- One-Loop Provider
  - evaluates interference term virtual × LO and LO × LO
  - performs UV-renormalisation
  - sends $1/\epsilon$ expansion (virtual part) and LO result of amplitude

Tensor Integral Reduction with Golem95

Structure of a One-Loop Amplitude

$$\mathcal{A}^{\{\lambda\}}(\{p_j\};\{m_j\}) = \sum_{\{c_i\},\alpha} f^{\{c_i\}}\mathcal{G}_\alpha^{\{\lambda\}}(\{p_j\};\{m_j\})$$

$$\begin{aligned}\mathcal{G}_\alpha^{\{\lambda\}}(\{p_j\};\{m_j\}) &= \int\frac{\mathrm{d}^n k}{i\pi^{n/2}}\frac{\mathcal{N}^{\{\lambda\}}(k)}{D_1\cdots D_N}\\ &= \sum_r \mathcal{N}^{\{\lambda\}}_{\mu_1\ldots\mu_r}(\{p_j\};\{m_j\})\cdot I_N^{n\,\mu_1\ldots\mu_r}(\{p_j\};\{m_j\})\end{aligned}$$

Idea: Split implementation into two steps

1. Numerically stable reduction of tensor integrals
2. Matrix element generator for one-loop amplitudes

Structure of a One-Loop Amplitude

$$\mathcal{A}^{\{\lambda\}}(\{p_j\};\{m_j\}) = \sum_{\{c_i\},\alpha} f^{\{c_i\}}\mathcal{G}_\alpha^{\{\lambda\}}(\{p_j\};\{m_j\})$$

$$\begin{aligned}\mathcal{G}_\alpha^{\{\lambda\}}(\{p_j\};\{m_j\}) &= \int\frac{\mathrm{d}^n k}{i\pi^{n/2}}\frac{\mathcal{N}^{\{\lambda\}}(k)}{D_1\cdots D_N}\\ &= \sum_r \mathcal{N}^{\{\lambda\}}_{\mu_1\ldots\mu_r}(\{p_j\};\{m_j\})\cdot I_N^{n\,\mu_1\ldots\mu_r}(\{p_j\};\{m_j\})\end{aligned}$$

Idea: Split implementation into two steps

1. Numerically stable reduction of tensor integrals
2. Matrix element generator for one-loop amplitudes

Structure of a One-Loop Amplitude

$$\mathcal{A}^{\{\lambda\}}(\{p_j\};\{m_j\}) = \sum_{\{c_i\},\alpha} f^{\{c_i\}}\mathcal{G}_\alpha^{\{\lambda\}}(\{p_j\};\{m_j\})$$

$$\begin{aligned}\mathcal{G}_\alpha^{\{\lambda\}}(\{p_j\};\{m_j\}) &= \int\frac{\mathrm{d}^n k}{i\pi^{n/2}}\frac{\mathcal{N}^{\{\lambda\}}(k)}{D_1\cdots D_N} \\ &= \sum_r \mathcal{N}^{\{\lambda\}}_{\mu_1\ldots\mu_r}(\{p_j\};\{m_j\})\cdot I_N^{n\,\mu_1\ldots\mu_r}(\{p_j\};\{m_j\})\end{aligned}$$

Idea: Split implementation into two steps

1. Numerically stable reduction of tensor integrals
2. Matrix element generator for one-loop amplitudes

Structure of a One-Loop Amplitude

$$\mathcal{A}^{\{\lambda\}}(\{p_j\};\{m_j\}) = \sum_{\{c_i\},\alpha} f^{\{c_i\}}\mathcal{G}_\alpha^{\{\lambda\}}(\{p_j\};\{m_j\})$$

$$\begin{aligned}\mathcal{G}_\alpha^{\{\lambda\}}(\{p_j\};\{m_j\}) &= \int\frac{\mathrm{d}^n k}{i\pi^{n/2}}\frac{\mathcal{N}^{\{\lambda\}}(k)}{D_1\cdots D_N}\\ &= \sum_r \mathcal{N}^{\{\lambda\}}_{\mu_1\ldots\mu_r}(\{p_j\};\{m_j\})\cdot I_N^{n\,\mu_1\ldots\mu_r}(\{p_j\};\{m_j\})\end{aligned}$$

Idea: Split implementation into two steps

1. Numerically stable reduction of tensor integrals
2. Matrix element generator for one-loop amplitudes

golem95 One-Loop Integral Library

$$I_N^{n\,\mu_1\ldots\mu_r}(a_1,\ldots,a_r;S) = \int \frac{\mathrm{d}^n k}{i\pi^{n/2}} \frac{q_{a_1}^{\mu_1}\cdots q_{a_r}^{\mu_r}}{\prod_{j\in S}(q_j^2 - m_j^2 + i\delta)}$$

$$S_{ij} = (q_i - q_j)^2 - m_i^2 - m_j^2, \quad q_j = k + r_j$$

Decomposition into Lorentz invariant integrals:

$$I_N^{n\,\mu_1\ldots\mu_r} = (-1)^N \Gamma(N - n/2) \sum_{p,j_1,\ldots,j_p} T^{\mu_1\ldots\mu_r}(\{r_j\}, g^{\cdot\cdot})\times$$

$$\int \mathrm{d}z_1\cdots\mathrm{d}z_N \delta(1 - z_1 - \ldots z_N) \frac{z_{l_1}\ldots z_{l_p}}{(-1/2z^T Sz - i\delta)^{N-n/2}}$$

- Can be reduced further → $I_{N-1}^n + I_N^{n+2}$
- Can be evaluated numerically (degenerated kinematics)

⇒ Gram determinants can be avoided

golem95 One-Loop Integral Library

$$I_N^{n\,\mu_1\ldots\mu_r}(a_1,\ldots,a_r;S) = \int \frac{\mathrm{d}^n k}{i\pi^{n/2}} \frac{q_{a_1}^{\mu_1}\cdots q_{a_r}^{\mu_r}}{\prod_{j\in S}(q_j^2 - m_j^2 + i\delta)}$$

$$S_{ij} = (q_i - q_j)^2 - m_i^2 - m_j^2, \quad q_j = k + r_j$$

Decomposition into Lorentz invariant integrals:

$$I_N^{n\,\mu_1\ldots\mu_r} = (-1)^N \Gamma(N - n/2) \sum_{p,j_1,\ldots,j_p} T^{\mu_1\ldots\mu_r}(\{r_j\}, g^{\cdot\cdot})\times$$

$$\int \mathrm{d}z_1 \cdots \mathrm{d}z_N \delta(1 - z_1 - \ldots z_N) \frac{z_{l_1}\ldots z_{l_p}}{(-1/2z^T S z - i\delta)^{N-n/2}}$$

- Can be reduced further → $I_{N-1}^n + I_N^{n+2}$
- Can be evaluated numerically (degenerated kinematics)

⇒ Gram determinants can be avoided

golem95 One-Loop Integral Library

$$I_N^{d;\mu_1\ldots\mu_r}(S) = \int\frac{\mathrm{d}^d k}{i\pi^d/2}\frac{k^{\mu_1}k^{\mu_2}\cdots k^{\mu_r}}{[(k+r_1)^2-m_1^2]\cdots[(k+r_N)^2-m_N^2]}$$

$$\begin{aligned} S_{ij} &= (r_i-r_j)^2-m_i^2-m_j^2 \\ G_{ij} &= 2r_i\cdot r_j \end{aligned}$$

- Tensor Integrals from loop momentum

golem95 One-Loop Integral Library

$$I_N^{d;\mu_1\ldots\mu_r}(S) = \int \frac{\mathrm{d}^d k}{i\pi^d/2} \frac{k^{\mu_1}k^{\mu_2}\cdots k^{\mu_r}}{[(k+r_1)^2-m_1^2]\cdots[(k+r_N)^2-m_N^2]}$$

$$I_N^d(S) = (-1)^N\Gamma(N{-}d/2)\int_0 \mathrm{d}^N z \frac{\delta(1-\sum z_i)}{\left(-\frac{1}{2}z^{\mathsf{T}}Sz - i\delta\right)^{N-d/2}}$$

- Reduction to scalar basis $\Rightarrow (\det G)^{-1}$

golem95 One-Loop Integral Library

$$I_N^{d;\mu_1\ldots\mu_r}(S) = \int\frac{\mathrm{d}^d k}{i\pi^d/2}\frac{k^{\mu_1}k^{\mu_2}\cdots k^{\mu_r}}{[(k+r_1)^2-m_1^2]\cdots[(k+r_N)^2-m_N^2]}$$

$$I_N^d(l_1,\ldots,l_p;S) = (-1)^N\Gamma(N-d/2)\int_0\mathrm{d}^N z\frac{\delta(1-\sum z_i)z_{l_1}\cdots z_{l_p}}{\left(-\frac{1}{2}z^\intercal S z - i\delta\right)^{N-d/2}}$$

$$I_N^d(S) = (-1)^N\Gamma(N-d/2)\int_0\mathrm{d}^N z\frac{\delta(1-\sum z_i)}{\left(-\frac{1}{2}z^\intercal S z - i\delta\right)^{N-d/2}}$$

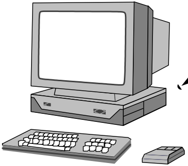
- `GOLEM` basis: integrals with numerator

golem95 One-Loop Integral Library

$$I_N^{d;\mu_1\ldots\mu_r}(S) = \int \frac{\mathrm{d}^d k}{i\pi^d/2} \frac{k^{\mu_1}k^{\mu_2}\cdots k^{\mu_r}}{[(k+r_1)^2-m_1^2]\cdots[(k+r_N)^2-m_N^2]}$$

$$I_N^d(l_1,\ldots,l_p;S) = (-1)^N\Gamma(N-d/2)\int_0 \mathrm{d}^N z \frac{\delta(1-\sum z_i)z_{l_1}\cdots z_{l_p}}{\left(-\frac{1}{2}z^\intercal S z - i\delta\right)^{N-d/2}}$$

$$I_N^d(S) = (-1)^N\Gamma(N-d/2)\int_0 \mathrm{d}^N z \frac{\delta(1-\sum z_i)}{\left(-\frac{1}{2}z^\intercal S z - i\delta\right)^{N-d/2}}$$



- numerical evaluation or algebraic reduction

golem95 One-Loop Integral Library

Current version of golem95

- http://lappweb.in2p3.fr/lapth/Golem/golem95.html
- algebraic separation of IR poles
- cache, avoiding multiple evaluation
- all required integrals for $N \leq 6$, massless
- documentation, examples available

Partially finished

- version with propagator masses
- currently: finite box (D_0) by call to LoopTools [T. Hahn] (ready)
- numerical branch under development

Early stage of development

- Complex propagator masses

golem95 One-Loop Integral Library

Current version of golem95

- http://lappweb.in2p3.fr/lapth/Golem/golem95.html
- algebraic separation of IR poles
- cache, avoiding multiple evaluation
- all required integrals for $N \leq 6$, massless
- documentation, examples available

Partially finished

- version with propagator masses
- currently: finite box (D_0) by call to LoopTools [T. Hahn] (ready)
- numerical branch under development

Early stage of development

- Complex propagator masses

golem95 One-Loop Integral Library

Current version of golem95

- http://lappweb.in2p3.fr/lapth/Golem/golem95.html
- algebraic separation of IR poles
- cache, avoiding multiple evaluation
- all required integrals for $N \leq 6$, massless
- documentation, examples available

Partially finished

- version with propagator masses
- currently: finite box (D_0) by call to LoopTools [T. Hahn] (ready)
- numerical branch under development

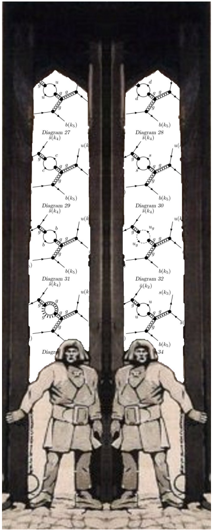
Early stage of development

- Complex propagator masses

An Automated 1-Loop Matrix Element Generator

golem-2.0

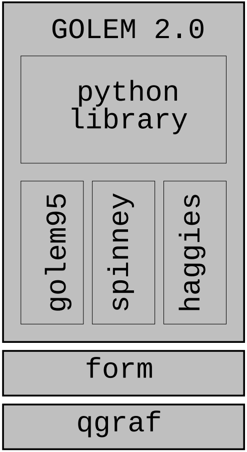
Matrix elements made easy...



GOLEM-2.0: One-Loop Matrix Element Generator

Overview

- implementation of the Golem method
- very modular
  - python library (command line tools)
  - spinney: helicity spinors in Form
  - haggies: optimizing code generator
  - golem95: integral library
- based on Form and QGraf [Vermaseren;Nogueira]
- Fortran 95 matrix element code



GOLEM 2.0

python library

golem95

spinney

haggies

form

qgraf

GOLEM-2.0: Matrix Elements Made Easy

- create configuration file
- enter process, here: $gg \longrightarrow s\bar{s}b\bar{b}$ @ NLO in QCD
- set up process directory
- generate code and draw diagrams

shell

```
$ golem-main.py --template process.in
$
```

GOLEM-2.0: Matrix Elements Made Easy

- create configuration file
- enter process, here: $gg \longrightarrow s\bar{s}b\bar{b}$ @ NLO in QCD
- set up process directory
- generate code and draw diagrams

editor: process.in

```
process_path=<a directory>
in=g,g
out=s,s~,b,b~
order=gs,4,6
model=sm

# more settings optional
...
```

GOLEM-2.0: Matrix Elements Made Easy

- create configuration file
- enter process, here: $gg \longrightarrow s\bar{s}b\bar{b}$ @ NLO in QCD
- set up process directory
- generate code and draw diagrams

shell

```
$ golem-main.py --template process.in
$ edit process.in
$ golem-main.py process.in
$
```

GOLEM-2.0: Matrix Elements Made Easy

- create configuration file
- enter process, here: $gg \longrightarrow s\bar{s}b\bar{b}$ @ NLO in QCD
- set up process directory
- generate code and draw diagrams

shell

```
$ golem-main.py --template process.in
$ edit process.in
$ golem-main.py process.in
$ make dist # -> matrix.tar.gz
$
```

GOLEM-2.0: Matrix Elements Made Easy

- create configuration file
- enter process, here: $gg \longrightarrow s\bar{s}b\bar{b}$ @ NLO in QCD
- set up process directory
- generate code and draw diagrams

shell

```
$ golem-main.py --template process.in
$ edit process.in
$ golem-main.py process.in
$ make dist # -> matrix.tar.gz
$ make doc # -> process.ps
$
```

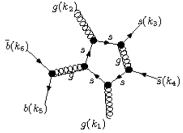


Diagram 1109

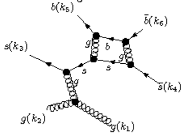


Diagram 1111

GOLEM-2.0: Work in Progress

Features not fully implemented (but planned/in progress):

- Les Houches interface
- FeynRules import [C. Duhr]
- Renormalisation of massive theories

Implemented but not fully tested:

- Majorana fermions and higher spins
- massive processes

+ improvements considering size, speed and interface

GOLEM at Work (Some Results)

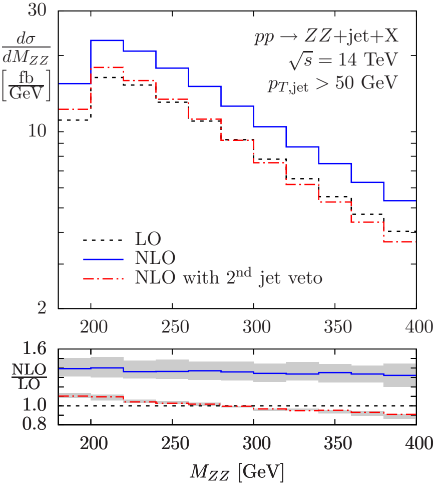
`GOLEM` method has been used for

- $gg \longrightarrow W^*W^* \longrightarrow l\nu l'\nu'$ [Binoth,Ciccolini,Kauer,Krämer]
- $gg \longrightarrow HH, HHH$ [Binoth,Karg,Kauer,Rückl]
- $pp \longrightarrow Hjj$ (VBF/GF) [Andersen,Binoth,Heinrich,Smillie]
- $pp \longrightarrow VVj$ [Binoth,Gleisberg,Karg,Kauer,Sanguinetti]
- $q\bar{q} \longrightarrow b\bar{b}b\bar{b}$ [Binoth,Greiner,Guffanti,Guillet,TR,Reuter]
- $pp \longrightarrow \text{Graviton} + j$ [Karg et al.]
- $gg \longrightarrow b\bar{b}b\bar{b}$ (in progress)
- $\ldots$

$pp \to VVj$ [Binoth,Gleisberg,Karg,Kauer,Sanguinetti]

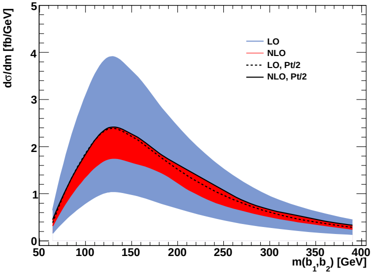
- high priority "wishlist" process
- algebraic reduction of tensor integrals
- at most pentagon diagrams
- successful comparison with

  [Dittmaier,Kallweit,Uwer;Campbell,Ellis,Zanderighi]



$q\bar{q} \to b\bar{b}b\bar{b}$ [Binoth,Greiner,Guffanti,Guillet,TR,Reuter]

- added to “wishlist” 2007
- background to BSM Higgs search
- calculation using `golem-2.0` and `golem95`
- $gg \to b\bar{b}b\bar{b}$ missing
  $\Rightarrow$ to be completed $\approx$ June



Outlook

Advertisment. . .

- golem-2.0 is an open, extensible framework
- fully operational interface to Samurai added

What is Samurai? [Ossola, Mastrolia, TR, Tramontano]

- new library implementing *d*-dimensional reduction at the integrand level
- publication plus code public soon

. . . stay tuned!

Outlook

Advertisment. . .

- `golem-2.0` is an open, extensible framework
- fully operational interface to Samurai added

What is Samurai? [Ossola, Mastrolia, TR, Tramontano]

- new library implementing *d*-dimensional reduction at the integrand level
- publication plus code public soon

. . . stay tuned!

Conclusions

- Automatization of one-loop matrix element required
- GOLEM method: Feynman diagram based approach
- Implementation in two steps:
  - Tensor integral form factors (`golem95`)
  - Matrix element generator (`golem-2.0`)
- `golem95`:
  - massless version publicly available
  - propagator masses coming soon
  - complex masses in preparation
- `golem-2.0`
  - main features are implemented
  - well prepared for QCD/EW/BSM
  - already in use for some calculations
  - more tests required
  - first release planned for summer 2010