

Status of MadGOLEM

David López-Val

in collaboration with D. Gonçalves Netto, F. Gross, T. Plehn (Heidelberg U.), I. Wigmore (Edinburgh U.), K. Mawatari (Vrije U.)

Institut für Theoretische Physik
Ruprecht-Karls Universität Heidelberg



MadGraph Spring 2011 Workshop – May 4th 2011

Fermilab

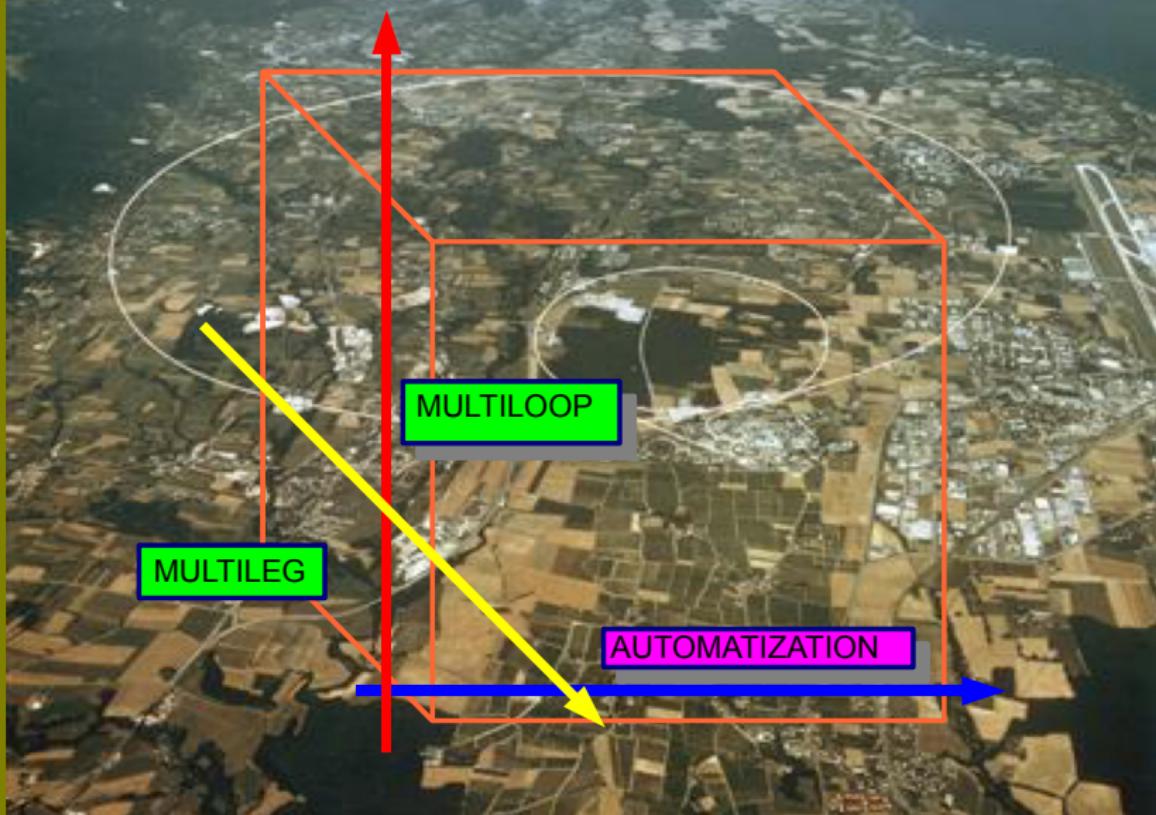
Outline

- 1 Motivating MadGOLEM
- 2 MadGOLEM from inside: architecture of the code
- 3 MadGOLEM from the very inside: finer details
 - The MadGOLEM one-loop Calculator
 - Handling the divergences
- 4 MadGOLEM from outside: running MadGOLEM
- 5 MadGOLEM underway: performance and cross-checks
- 6 Beyond cross-checks: New Physics studies with MadGOLEM
- 7 Conclusions & Outlook

Outline

- 1 Motivating MadGOLEM
- 2 MadGOLEM from inside: architecture of the code
- 3 MadGOLEM from the very inside: finer details
 - The MadGOLEM one-loop Calculator
 - Handling the divergences
- 4 MadGOLEM from outside: running MadGOLEM
- 5 MadGOLEM underway: performance and cross-checks
- 6 Beyond cross-checks: New Physics studies with MadGOLEM
- 7 Conclusions & Outlook

Tools for New Physics Searches at Hadron Colliders



Whys and wherefores

Why Multileg ?

- high rates of multiparticle final states.
- signal and background for New Physics searches

Whys and wheresores

Why Multileg ?

- high rates of multiparticle final states.
- signal and background for New Physics searches

Why Multiloop ?

- QCD corrections (virtual, real) numerically large ($K \sim 1.3$)
- QCD corrections of intrinsic phenomenological relevance
 - They stabilize the **renormalization/factorization scale** dependence
 - They provide **reliable normalizations** for collider observables
 - They account for **ISR** , allow **matching** with resummed calculations,...

Whys and wheresores

Why Multileg ?

- high rates of multiparticle final states.
- signal and background for New Physics searches

Why Multiloop ?

- QCD corrections (virtual, real) numerically large ($K \sim 1.3$)
- QCD corrections of intrinsic phenomenological relevance
 - They stabilize the **renormalization/factorization scale** dependence
 - They provide **reliable normalizations** for collider observables
 - They account for **ISR** , allow **matching** with resummed calculations,...

Why Automatization ?

- Huge (obviously far from hand-doable) complexity
- Many analogue structures \Rightarrow suggestive to **generalize** in a **process&model-independent** way

Challenges

An awkward business !! ...

- **complexity:** number of diagrams (combinatorics !), number of scales
- **stability:** linear dependence within the external momenta, reduction of one loop integrals (Gram determinants)
- **multidimensional** phase space integration
- gauge-invariant treatment of **unstable** particles
- ... a give and take between **speed** & **stability**

Challenges

An awkward business !! ...

- **complexity:** number of diagrams (combinatorics !), number of scales
- **stability:** linear dependence within the external momenta, reduction of one loop integrals (Gram determinants)
- **multidimensional** phase space integration
- gauge-invariant treatment of **unstable** particles
- ... a give and take between **speed** & **stability**

Towards Automation @ NLO

- Automated LO Event Generators: MadGraph/MadEvent, CalcHEP/CompHEP, Whizard, ...
- Virtual 1-loop Calculators: HELAC/CutTools, SAMURAI, BlackHat, **GOLEM**, ...
- Real emission calculators MadGraph/MadFKS, MadGraph/MadDipole

MadNLO aMC@NLO MadGOLEM

MadGOLEM in context

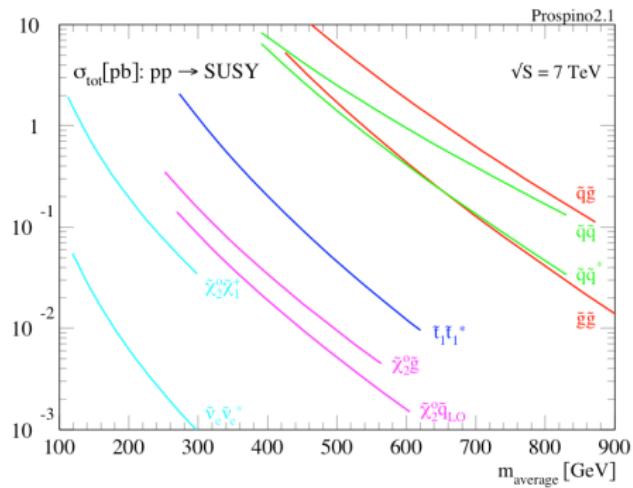
- ♠ NLO crucial for New Physics searches (e.g. in SUSY)

MadGOLEM in context

♠ NLO crucial for New Physics searches (e.g. in SUSY)

PROSPINO

Benakker, Höper,
Krämer, Plehn, Spira,
Zerwas

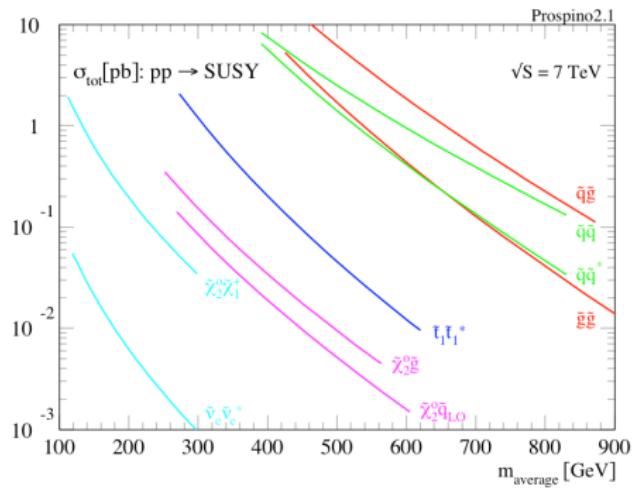


MadGOLEM in context

♠ NLO crucial for New Physics searches (e.g. in SUSY)

PROSPINO

Benakker, Höper,
Krämer, Plehn, Spira,
Zerwas



MadGOLEM

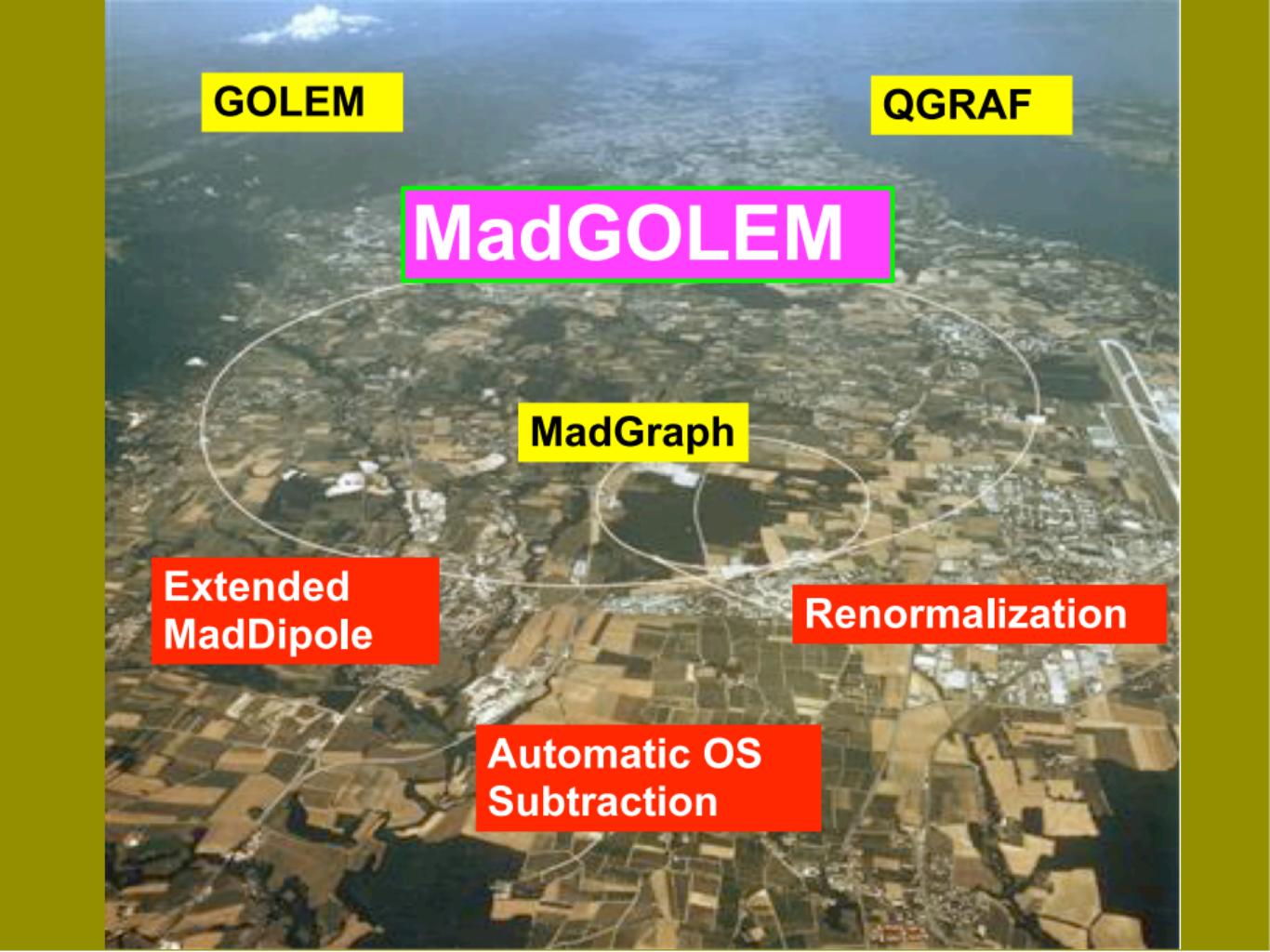
AUTOMATIZING



Prospino



GENERALIZING



GOLEM

QGRAF

MadGOLEM

MadGraph

**Extended
MadDipole**

Renormalization

**Automatic OS
Subtraction**

Outline

- 1 Motivating MadGOLEM
- 2 MadGOLEM from inside: architecture of the code
- 3 MadGOLEM from the very inside: finer details
 - The MadGOLEM one-loop Calculator
 - Handling the divergences
- 4 MadGOLEM from outside: running MadGOLEM
- 5 MadGOLEM underway: performance and cross-checks
- 6 Beyond cross-checks: New Physics studies with MadGOLEM
- 7 Conclusions & Outlook

MadGOLEM from inside: an automatic NLO calculator

$2 \rightarrow 2(+j)$ cross-section at a hadron collider

$$\sigma_{\text{pp}(p_1, p_2) \rightarrow X}^{\textcolor{red}{NLO}} = \sum_{a,b} \int_0^1 dx_1 \int_0^1 dx_2 f_a(x_1, \mu_F^2) f_b(x_2, \mu_F^2) \times \\ \underbrace{\sigma_{p_1 p_2 \rightarrow X}^{\textcolor{red}{NLO}}(x_1, x_2, \alpha_s(\mu_R^2), Q^2/\mu_F^2, Q^2/\mu_R^2)}$$

MadGOLEM from inside: an automatic NLO calculator

$2 \rightarrow 2(+j)$ cross-section at a hadron collider

$$\sigma_{\text{pp}(p_1, p_2) \rightarrow X}^{\text{NLO}} = \sum_{a,b} \int_0^1 dx_1 \int_0^1 dx_2 f_a(x_1, \mu_F^2) f_b(x_2, \mu_F^2) \times \\ \underbrace{\sigma_{p_1 p_2 \rightarrow X}^{\text{NLO}}(x_1, x_2, \alpha_s(\mu_R^2), Q^2/\mu_F^2, Q^2/\mu_R^2)}$$

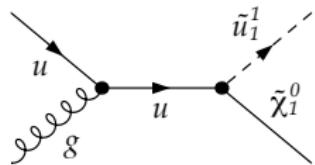
$$d\sigma^{NLO} = [\mathcal{B}(\Phi_{p1p2}) + \alpha_s \mathcal{V}(\Phi_{p1p2})] d\Phi_{p1p2} + \alpha_s \mathcal{R}(\Phi_{p1p2j}) d\Phi_{p1p2j}$$

MadGOLEM from inside: an automatic NLO calculator

$2 \rightarrow 2(+j)$ hadronic cross-section at a hadron collider

$$\sigma_{\text{pp}(p_1, p_2) \rightarrow X}^{\textcolor{red}{NLO}} = \sum_{a,b} \int_0^1 dx_1 \int_0^1 dx_2 f_a(x_1, \mu_F^2) f_b(x_2, \mu_F^2) \times \\ \underbrace{\sigma_{p_1 p_2 \rightarrow X}^{\textcolor{red}{NLO}}(x_1, x_2, \alpha_s(\mu_R^2), Q^2/\mu_F^2, Q^2/\mu_R^2)}_{}$$

$$d\sigma^{NLO} = \left[\underbrace{\mathcal{B}(\Phi_{p1p2})}_{\mathcal{B}(\Phi_{p1p2})} + \alpha_s \mathcal{V}(\Phi_{p1p2}) \right] d\Phi_{p1p2} + \alpha_s \mathcal{R}(\Phi_{p1p2j}) d\Phi_{p1p2j}$$



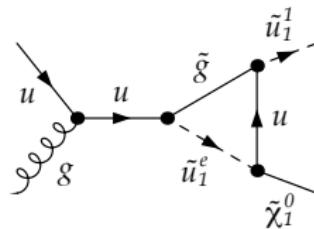
MadGOLEM from inside: an automatic NLO calculator

$2 \rightarrow 2(+j)$ cross-section at a hadron collider

$$\sigma_{\text{pp}(p_1, p_2) \rightarrow X}^{\text{NLO}} = \sum_{a,b} \int_0^1 dx_1 \int_0^1 dx_2 f_a(x_1, \mu_F^2) f_b(x_2, \mu_F^2) \times \\ \underbrace{\sigma_{p_1 p_2 \rightarrow X}^{\text{NLO}}(x_1, x_2, \alpha_s(\mu_R^2), Q^2/\mu_F^2, Q^2/\mu_R^2)}$$

$$d\sigma^{NLO} = \left[\mathcal{B}(\Phi_{p1p2}) + \underbrace{\alpha_s \mathcal{V}(\Phi_{p1p2})}_{\mathcal{R}(\Phi_{p1p2j})} \right] d\Phi_{p1p2} + \alpha_s \mathcal{R}(\Phi_{p1p2j}) d\Phi_{p1p2j}$$

- ♠ **QGRAF**
- ♠ **GOLEM**
- ♠ **QGRAF-GOLEM interface**
- ♠ **CT generator& library**



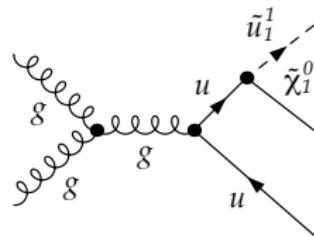
MadGOLEM from inside: an automatic NLO calculator

$2 \rightarrow 2(+j)$ hadronic cross-section at a hadron collider

$$\sigma_{\text{pp}(p_1, p_2) \rightarrow X}^{\text{NLO}} = \sum_{a,b} \int_0^1 dx_1 \int_0^1 dx_2 f_a(x_1, \mu_F^2) f_b(x_2, \mu_F^2) \times \underbrace{\sigma_{p_1 p_2 \rightarrow X}^{\text{NLO}}(x_1, x_2, \alpha_s(\mu_R^2), Q^2/\mu_F^2, Q^2/\mu_R^2)}$$

$$d\sigma^{NLO} = [\mathcal{B}(\Phi_{p1p2}) + \alpha_s \mathcal{V}(\Phi_{p1p2})] d\Phi_{p1p2} + \underbrace{\alpha_s \mathcal{R}(\Phi_{p1p2j}) d\Phi_{p1p2j}}$$

- ♣ MadGraph
- ♣ (extended) MadDipole
- ♣ Automatic OS subtraction



MadGOLEM from inside: an automatic NLO calculator

$2 \rightarrow 2(+j)$ hadronic cross-section at a hadron collider

$$\sigma_{\text{pp}(p_1, p_2) \rightarrow X}^{\textcolor{red}{NLO}} = \sum_{a,b} \int_0^1 dx_1 \int_0^1 dx_2 f_a(x_1, \mu_F^2) f_b(x_2, \mu_F^2) \times \\ \underbrace{\sigma_{p_1 p_2 \rightarrow X}^{\textcolor{red}{NLO}}(x_1, x_2, \alpha_s(\mu_R^2), Q^2/\mu_F^2, Q^2/\mu_R^2)}$$

$$d\sigma^{NLO} = [\mathcal{B}(\Phi_{p1p2}) + \alpha_s \mathcal{V}(\Phi_{p1p2})] \underbrace{d\Phi_{p1p2}}_{+ \alpha_s \mathcal{R}(\Phi_{p1p2j}) d\Phi_{p1p2j}}$$



MadEvent

$$d\Phi_{p1p2} = \prod_{i=1}^2 \frac{d^3 p_i}{(2\pi)^3 2 E_i} (2\pi)^4 \delta^{(4)}(p_1 + p_2 - \sum_i p_i)$$

Structure overview: MadGOLEM modules

Step	Tool	Strategy
Multileg ME generation	Modified MadGraph	Feynman diagrams & hel. ampl. (HELAS-based)
One-loop ME generation	QGRAF	graph theory
Reduction of 1-loop tensors to a scalar integrals	GOLEM	Dim. Reg. (UV & IR) Passarino, Veltman ['79]
Subtraction of IR divergences	Modified MadGraph/MadDipole	Catani, Seymour ['97]
Renormalization of UV divergences	own routines	OS / $\overline{\text{MS}}$ schemes SUSY restoration
Subtraction of OS divergences	own routines	Beenakker et al. ['96]

Structure overview: MadGOLEM modules

Step	Tool	Strategy
Multileg ME generation	Modified MadGraph	Feynman diagrams & hel. ampl. (HELAS-based)
One-loop ME generation	QGRAF	graph theory
Reduction of one-loop ME to basic scalar integrals	GOLEM	Dim. Reg. (UV & IR) Passarino, Veltman ['79]
Subtraction of IR divergences	Modified MadGraph/MadDipole	Catani, Seymour ['97]
Renormalization of UV divergences	own routines	OS / $\overline{\text{MS}}$ schemes SUSY restoration
Subtraction of OS divergences	own routines	Beenakker et al. ['96]

Structure overview: MadGOLEM modules

Step	Tool	Strategy
Multileg ME generation	Modified MadGraph	Feynman diagrams & hel. ampl. (HELAS-based)
One-loop ME generation	QGRAF	graph theory
Reduction of one-loop ME to basic scalar integrals	GOLEM	Dim. Reg. (UV & IR) Passarino, Veltman ['79]
Subtraction of IR divergences	Modified MadGraph/MadDipole	Catani, Seymour ['97]
Renormalization of UV divergences	own routines	OS / $\overline{\text{MS}}$ schemes SUSY restoration
Subtraction of OS divergences	own routines	Beenakker et al.['96]

Structure overview: MadGOLEM modules

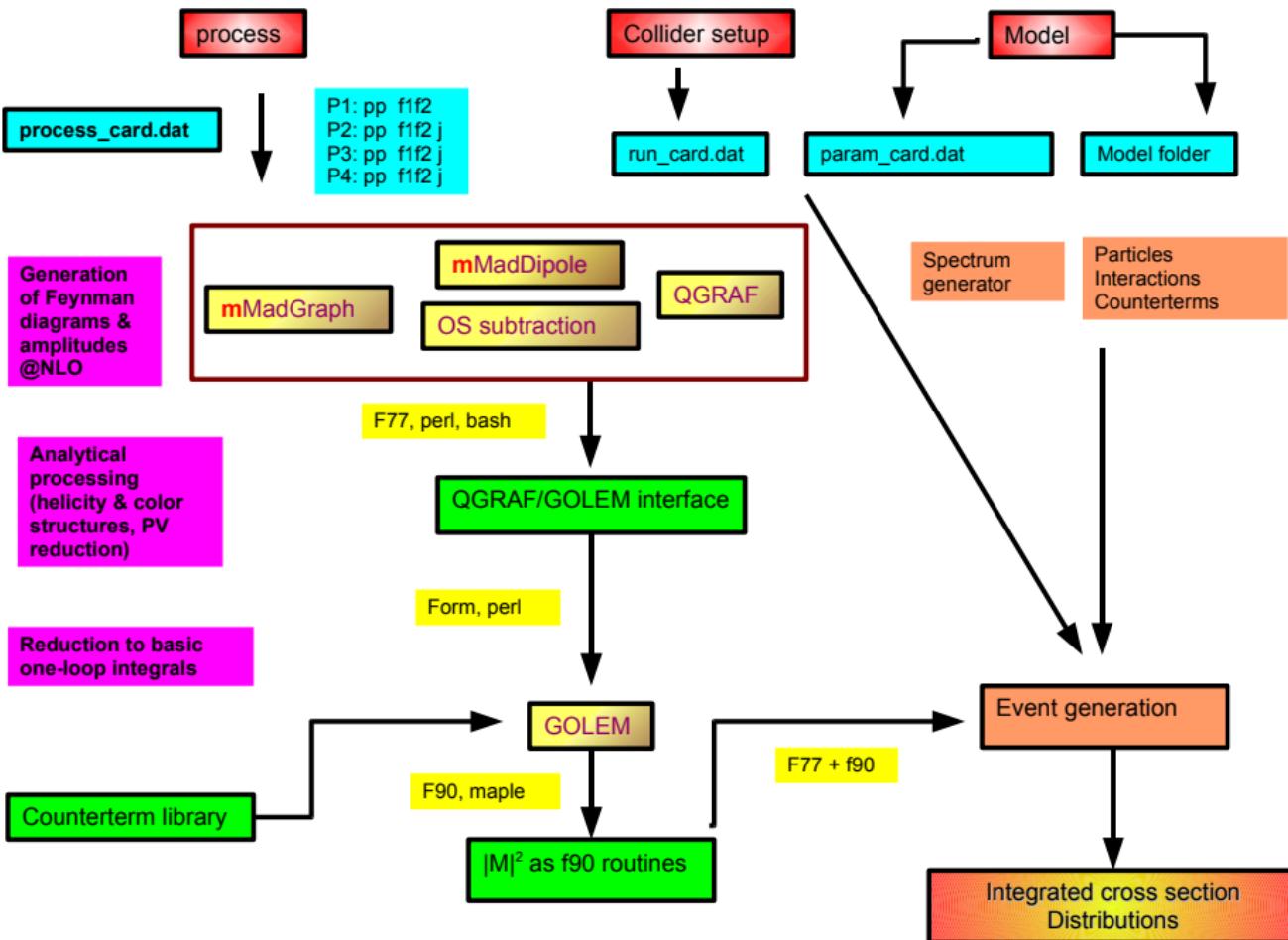
Step	Tool	Strategy
Multileg ME generation	Modified MadGraph	Feynman diagrams & hel. ampl. (HELAS-based)
One-loop ME generation	QGRAF	graph theory
Reduction of one-loop ME to basic scalar integrals	GOLEM	Dim. Reg. (UV & IR) Passarino, Veltman ['79]
Subtraction of IR divergences	Modified MadGraph/MadDipole	Catani, Seymour ['97]
Renormalization of UV divergences	own routines	OS / $\overline{\text{MS}}$ schemes SUSY restoration
Subtraction of OS divergences	own routines	Beenakker et al. ['96]

Structure overview: MadGOLEM modules

Step	Tool	Strategy
Multileg ME generation	Modified MadGraph	Feynman diagrams & hel. ampl. (HELAS-based)
One-loop ME generation	QGRAF	graph theory
Reduction of one-loop ME to basic scalar integrals	GOLEM	Dim. Reg. (UV & IR) Passarino, Veltman ['79]
Subtraction of IR divergences	Modified MadGraph/MadDipole	Catani, Seymour ['97]
Renormalization of UV divergences	own routines	OS / \overline{MS} schemes SUSY restoration
Subtraction of OS divergences	own routines	Beenakker et al. ['96]

Structure overview: MadGOLEM modules

Step	Tool	Strategy
Multileg ME generation	Modified MadGraph	Feynman diagrams & hel. ampl. (HELAS-based)
One-loop ME generation	QGRAF	graph theory
Reduction of one-loop ME to basic scalar integrals	GOLEM	Dim. Reg. (UV & IR) Passarino, Veltman ['79]
Subtraction of IR divergences	Modified MadGraph/MadDipole	Catani, Seymour ['97]
Renormalization of UV divergences	own routines	OS / \overline{MS} schemes SUSY restoration
Subtraction of OS divergences	own routines	Beenakker et al. ['96]



Surveying the tools (1): MadGraph/MadEvent

The MadGraph/MadEvent framework [Stelzer, Long], [Maltoni, Stelzer]

- Identifies the partonic subchannels contributing to a given process
- Provides the tree-level Feynman diagrams & matrix elements
- Computes the square of the matrix element and sums over spin & color
- Integrates over the phase space
- Returns cross-sections and parton-level events

⇒ MadGraph/MadEvent @ MadGOLEM

- MadGOLEM is built upon MadGraph v.4.5
- Modified `Cards/` and `bin/scripts` to generate input for QGRAF-GOLEM & dipole-OS subtraction
- Modified `MadGraphII/*.f` files to include/link the QGRAF-GOLEM & dipole-OS subtraction output
- In-house Phase Space generator

Surveying the tools (2): QGRAF

QGRAF

- Fortran 77 code for the automatic generation of Feynman diagrams
- **Input:** model file (vertices, propagators) & topology constraints
- **Method:** Graph-Theory-based algorithm (systematic connection of *colored* nodes of a certain valency)
- **Output:** symbolic description of the diagrams:
 - vertices/couplings
 - propagators/fields
 - symmetry factors, overall signs(*)

P. Nogueira, J. Comp. Phys. 105 (1993) 279

Surveying the tools (3): GOLEM

General Online Loop Evaluator For Matrix Elements

$$\begin{aligned}
 \mathcal{M}^{virtual} &= \\
 &= \frac{(2\pi\mu)^{4-d}}{i\pi^2} \int d^D q \frac{q_{\mu_1} \dots q_{\mu_n}}{(q^2 - m_1^2 [(q+k_1)^2 - m_1^2] \dots [(q+k_{N+1})^2 - m_{N+1}^2])} \\
 &= \sum_n \sum_i C_{\mu_1 \dots \mu_n}^i I_i^{\mu_1 \dots \mu_n}
 \end{aligned}$$

- ♣ Modified Passarino-Veltman Reduction of tensor form factors to a certain set of basis integrals \Rightarrow Inverse Gram determinants circumvented [Binoth, Guillet, Heinrich]

Binoth, Cullen, Guffanti, Guillet, Heinrich, Karg, Kauer, Kleinschmidt, Pilon, Reiter, Rodgers, Wigmore, arXiv:0810.0992 [hep-ph], arXiv:1101.5595 [hep-ph].

Surveying the tools (4): MadDipole

MadDipole @ MadGraph/MadEvent

Automatic generation of the **dipole subtraction terms** for **real radiation** and
virtual NLO corrections within MadGraph/MadEvent

[Frederix, Gehrmann, Greiner] arXiv:0808.2128, arXiv:1004.2903 [hep-ph]

Surveying the tools (4): MadDipole

MadDipole @ MadGraph/MadEvent

Automatic generation of the **dipole subtraction terms** for **real radiation** and
virtual NLO corrections within MadGraph/MadEvent

[Frederix, Gehrmann, Greiner] arXiv:0808.2128, arXiv:1004.2903 [hep-ph]

$$\sigma = \int_m d\sigma^B + \int_{m+1} (d\sigma^R - d\sigma^A) + \int_m \left[\int_1 d\sigma^A + d\sigma^V \right]$$

[Catani, Seymour]

Surveying the tools (4): MadDipole

MadDipole @ MadGraph/MadEvent

Automatic generation of the **dipole subtraction terms** for **real radiation** and
virtual NLO corrections within MadGraph/MadEvent

[Frederix, Gehrmann, Greiner] arXiv:0808.2128, arXiv:1004.2903 [hep-ph]

$$\sigma = \int_m d\sigma^B + \int_{m+1} (d\sigma^R - d\sigma^A) + \int_m \left[\int_1 d\sigma^A + d\sigma^V \right]$$

[Catani, Seymour]

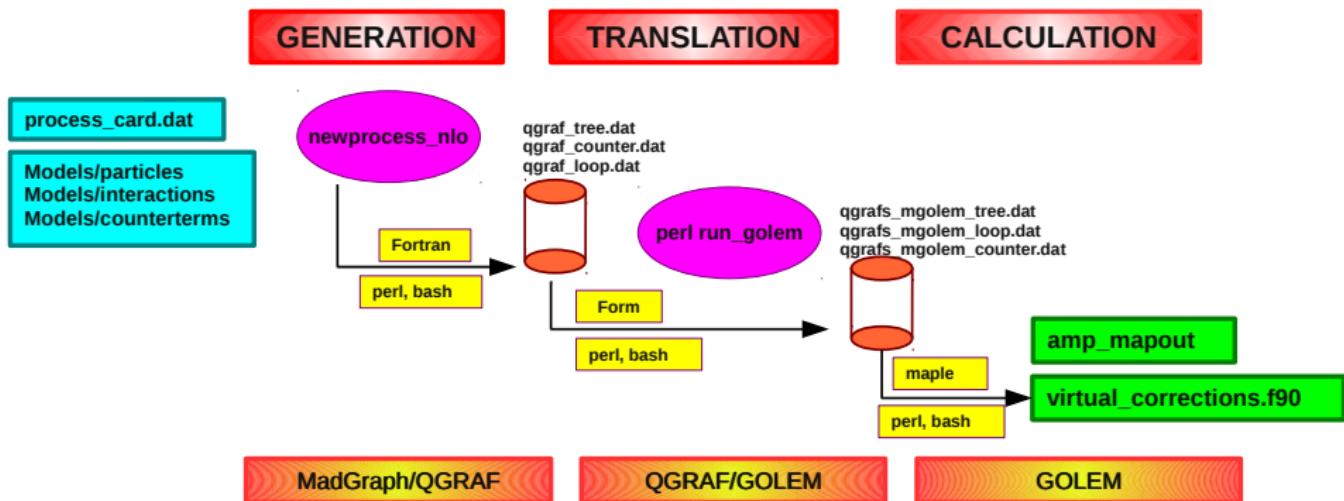
Extended MadDipole @ MadGOLEM:

- **Integrated SUSY dipoles** (including α -dependence) [Nagy, Trocsanyi]

Outline

- 1 Motivating MadGOLEM
- 2 MadGOLEM from inside: architecture of the code
- 3 MadGOLEM from the very inside: finer details
 - The MadGOLEM one-loop Calculator
 - Handling the divergences
- 4 MadGOLEM from outside: running MadGOLEM
- 5 MadGOLEM underway: performance and cross-checks
- 6 Beyond cross-checks: New Physics studies with MadGOLEM
- 7 Conclusions & Outlook

Step 1: Generation of the one-loop amplitude



Step 1: Generation of the one-loop amplitude

Generation of the one-loop amplitude

- User command: `newprocess_nlo`
- Language: bash, Fortran
- Input: `Model file & process_card` (which model, which process, which type of NLO corrections – SM-QCD, SUSY-QCD.)
- Modules involved: `QGRAF`
- Output: `qgraf_lo.dat`, `qgraf_nlo.dat`, `qgraf_counter.dat` :

Step 1: Generation of the one-loop amplitude

Generation of the one-loop amplitude

- **User command:** `newprocess_nlo`
- Language: bash, Fortran
- Input: Model file & process_card (which model, which process, which type of NLO corrections – SM-QCD, SUSY-QCD.)
- Modules involved: QGRAF
- Output: `qgraf_lo.dat`, `qgraf_nlo.dat`, `qgraf_counter.dat`:

Step 1: Generation of the one-loop amplitude

Generation of the one-loop amplitude

- **User command:** `newprocess_nlo`
- **Language:** bash, Fortran
- **Input:** Model file & process_card (which model, which process, which type of NLO corrections – SM-QCD, SUSY-QCD.)
- **Modules involved:** QGRAF
- **Output:** `qgraf_lo.dat`, `qgraf_nlo.dat`, `qgraf_counter.dat` :

Step 1: Generation of the one-loop amplitude

Generation of the one-loop amplitude

- **User command:** `newprocess_nlo`
- **Language:** bash, Fortran
- **Input:** `Model file & process_card` (which model, which process, which type of NLO corrections – SM-QCD, SUSY-QCD.)
- Modules involved: `QGRAF`
- Output: `qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat`:

Step 1: Generation of the one-loop amplitude

Generation of the one-loop amplitude

- **User command:** `newprocess_nlo`
- **Language:** bash, Fortran
- **Input:** `Model file & process_card` (which model, which process, which type of NLO corrections – SM-QCD, SUSY-QCD.)
- **Modules involved:** `QGRAF`
- **Output:** `qgraf_lo.dat`, `qgraf_nlo.dat`, `qgraf_counter.dat` :

Step 1: Generation of the one-loop amplitude

Generation of the one-loop amplitude

- **User command:** `newprocess_nlo`
- **Language:** bash, Fortran
- **Input:** `Model file & process_card` (which model, which process, which type of NLO corrections – SM-QCD, SUSY-QCD.)
- **Modules involved:** `QGRAF`
- **Output:** `qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat`:

Step 1: Generation of the one-loop amplitude

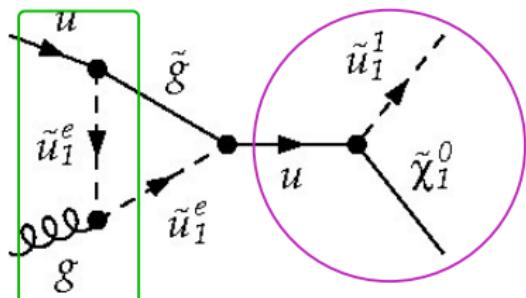
Generation of the one-loop amplitude

- **User command:** `newprocess_nlo`
 - **Language:** bash, Fortran
 - **Input:** Model file & process_card (which model, which process, which type of NLO corrections – SM-QCD, SUSY-QCD.)
 - **Modules involved:** QGRAF
 - **Output:** `qgraf_lo.dat`, `qgraf_nlo.dat`, `qgraf_counter.dat` :
- ♠ Extremely **fast** \iff few seconds for $\mathcal{O}(100)$ diagrams.

Step 1: Generation of the one-loop amplitude

One-loop amplitude from QGRAF: **qgraf_nlo.dat**

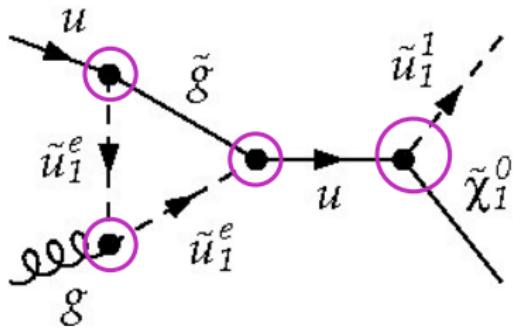
```
+ 1 *
inp([field.u], idx2r2, p1) *
inplorentz(+1, iv2r2L1, p1, ZERO ) *
inpcolor(1, iv2r2C3) *
inp([field.g], idx3r1, p2) *
inplorentz(+2, iv3r1L2, p2, ZERO ) *
inpcolor(2, iv3r1C8) *
out([field.ul], idx1r3, p3) *
outlorentz(+0, iv1r3L0, p3, MUL ) *
outcolor(1, iv1r3C3) *
out([field.n1], idx1r1, p4) *
outlorentz(+1, iv1r1L1, p4, MN1 ) *
outcolor(2, iv1r1C1) *
vertex(iv1,GULNIP ,ONE,
[field.n1], idx1r1, +1, -p4, iv1r1L1, +1, iv1r1C1,
[field.u], idx1r2, +1, p3+p4, iv1r2L1, +3, iv1r2C3,
[field.ulx], idx1r3, -0, -p3, iv1r3L0, -3, iv1r3C3) *
vertex(iv2,GQLGOP ,ONE,
[field.go], idx2r1, +1, k1-p1, iv2r1L1, +8, iv2r1C8,
[field.u], idx2r2, +1, p1, iv2r2L1, +3, iv2r2C3,
[field.ulx], idx2r3, -0, -k1, iv2r3L0, -3, iv2r3C3) *
vertex(iv3|,GC ,ONE,
[field.g], idx3r1, +2, p2, iv3r1L2, +8, iv3r1C8,
[field.ul], idx3r2, +0, k1, iv3r2L0, +3, iv3r2C3,
[field.ulx], idx3r3, -0, -k1-p2, iv3r3L0, -3, iv3r3C3) *
```



Step 1: Generation of the one-loop amplitude

One-loop amplitude from QGRAF: **qgraf_nlo.dat**

```
+ 1 *
inp([field.u], idx2r2, p1) *
inplorentz(+1, iv2r2L1, p1, ZERO ) *
inpcolor(1, iv2r2C3) *
inp([field.g], idx3r1, p2) *
inplorentz(+2, iv3r1L2, p2, ZERO ) *
inpcolor(2, iv3r1C8) *
out([field.ul], idx1r3, p3) *
outlorentz(+0, iv1r3L0, p3, MUL ) *
outcolor(1, iv1r3C3) *
out([field.n1], idx1r1, p4) *
outlorentz(+1, iv1r1L1, p4, MN1 ) *
outcolor(2, iv1r1C1) *
vertex(iv1,GULN1P ,ONE,
[field.n1], idx1r1, +1, -p4, iv1r1L1, +1, iv1r1C1,
[field.u], idx1r2, +1, p3+p4, iv1r2L1, +3, iv1r2C3,
[field.ulx], idx1r3, -0, -p3, iv1r3L0, -3, iv1r3C3) *
vertex(iv2,GQLGOP ,ONE,
[field.go], idx2r1, +1, k1-p1, iv2r1L1, +8, iv2r1C8,
[field.u], idx2r2, +1, p1, iv2r2L1, +3, iv2r2C3,
[field.ulx], idx2r3, -0, -k1, iv2r3L0, -3, iv2r3C3) *
vertex(iv3,GC ,ONE,
[field.g], idx3r1, +2, p2, iv3r1L2, +8, iv3r1C8,
[field.ul], idx3r2, +0, k1, iv3r2L0, +3, iv3r2C3,
[field.ulx], idx3r3, -0, -k1-p2, iv3r3L0, -3, iv3r3C3) *
```



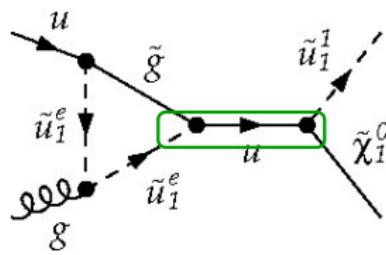
Step 1: Generation of the one-loop amplitude

One-loop amplitude from QGRAF: **qgraf_nlo.dat**

```

vertex(iv4,GQLGOM ,ONE,
[field.ux], idx4r1, -1, -p3-p4, iv4r1L1, -3, iv4r1C3,
[field.go], idx4r2, +1, -k1+p1, iv4r2L1, +8, iv4r2C8,
[field.ul], idx4r3, +0, k1+p2, iv4r3L0, +3, iv4r3C3) *
prop([field.u], idx4r1, idx1r2) *
propcolor(+3, iv4r1C3, iv1r2C3) *
proplorentz(+1, p3+p4, ZERO , iv4r1L1, iv1r2L1) *
prop([field.ul], idx2r3, idx3r2) *
propcolor(+3, iv2r3C3, iv3r2C3) *
proplorentz(+0, k1, MUL , iv2r3L0, iv3r2L0) *
prop([field.go], idx4r2, idx2r1) *
propcolor(+8, iv4r2C8, iv2r1C8) *
proplorentz(+1, k1-p1, MG0 , iv4r2L1, iv2r1L1) *
prop([field.ul], idx3r3, idx4r3) *
propcolor(+3, iv3r3C3, iv4r3C3) *
proplorentz(+0, k1+p2, MUL , iv3r3L0, iv4r3L0)

```



Step 2: Translation of the one-loop amplitude

QGRAF-GOLEM Interface

♠ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: **QGRAF output \Rightarrow Feynman rules**

- User command: `perl run_golem.pl`
- Language: bash, perl, FORM
- Input: `qgraf_lo.dat`, `qgraf_nlo.dat`, `qgraf_counter.dat`. Pre-processing options available (e.g. `topology selection`)
- Modules involved: **QGRAF-GOLEM Interface**
- Output: `graphs_golem_tree/loop/counter.h` :

Step 2: Translation of the one-loop amplitude

QGRAF-GOLEM Interface

- ♠ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: **QGRAF output \Rightarrow Feynman rules**

- **User command:** perl run_golem.pl
- Language: bash, perl, FORM
- Input: qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat . Pre-processing options available (e.g. topology selection)
- Modules involved: QGRAF-GOLEM Interface
- Output: graphs_golem_tree/loop/counter.h :

Step 2: Translation of the one-loop amplitude

QGRAF-GOLEM Interface

- ♠ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: QGRAF output \Rightarrow Feynman rules

- **User command:** perl run_golem.pl
- **Language:** bash, perl, FORM
- **Input:** qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat . Pre-processing options available (e.g. topology selection)
- **Modules involved:** QGRAF-GOLEM Interface
- **Output:** graphs_golem_tree/loop/counter.h :

Step 2: Translation of the one-loop amplitude

QGRAF-GOLEM Interface

♠ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: **QGRAF output \Rightarrow Feynman rules**

- **User command:** perl run_golem.pl
- **Language:** bash, perl, FORM
- **Input:** qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat . Pre-processing options available (e.g. topology selection)
- Modules involved: QGRAF-GOLEM Interface
- Output: graphs_golem_tree/loop/counter.h :

Step 2: Translation of the one-loop amplitude

QGRAF-GOLEM Interface

♠ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: **QGRAF output \Rightarrow Feynman rules**

- **User command:** perl run_golem.pl
- **Language:** bash, perl, FORM
- **Input:** qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat . Pre-processing options available (e.g. topology selection)
- **Modules involved:** QGRAF-GOLEM Interface
- **Output:** graphs_golem_tree/loop/counter.h :

Step 2: Translation of the one-loop amplitude

QGRAF-GOLEM Interface

- ♠ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: QGRAF output \Rightarrow Feynman rules

- **User command:** perl run_golem.pl
- **Language:** bash, perl, FORM
- **Input:** qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat . Pre-processing options available (e.g. topology selection)
- **Modules involved:** QGRAF-GOLEM Interface
- **Output:** graphs_golem_tree/loop/counter.h :

Step 2: Translation of the one-loop amplitude

QGRAF-GOLEM Interface

- ♠ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: QGRAF output \Rightarrow Feynman rules

- **User command:** perl run_golem.pl
- **Language:** bash, perl, FORM
- **Input:** qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat . Pre-processing options available (e.g. topology selection)
- **Modules involved:** QGRAF-GOLEM Interface
- **Output:** graphs_golem_tree/loop/counter.h :

- ♠ Different modules for different pieces of the amplitude:

vertices.h

Step 2: Translation of the one-loop amplitude

QGRAF-GOLEM Interface

- ♠ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: QGRAF output → Feynman rules

- **User command:** perl run_golem.pl
 - **Language:** bash, perl, FORM
 - **Input:** qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat . Pre-processing options available (e.g. topology selection)
 - **Modules involved:** QGRAF-GOLEM Interface
 - **Output:** graphs_golem_tree/loop/counter.h :
- ♠ Different modules for different pieces of the amplitude:

vertices.h

propagators.h

Step 2: Translation of the one-loop amplitude

QGRAF-GOLEM Interface

- ♠ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: QGRAF output → Feynman rules

- **User command:** perl run_golem.pl
- **Language:** bash, perl, FORM
- **Input:** qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat . Pre-processing options available (e.g. topology selection)
- **Modules involved:** QGRAF-GOLEM Interface
- **Output:** graphs_golem_tree/loop/counter.h :

- ♠ Different modules for different pieces of the amplitude:

vertices.h

propagators.h

color.h

Step 2: Translation of the one-loop amplitude

QGRAF-GOLEM Interface

- ♠ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: QGRAF output → Feynman rules

- **User command:** perl run_golem.pl
- **Language:** bash, perl, FORM
- **Input:** qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat . Pre-processing options available (e.g. topology selection)
- **Modules involved:** QGRAF-GOLEM Interface
- **Output:** graphs_golem_tree/loop/counter.h :

- ♠ Different modules for different pieces of the amplitude:

vertices.h

propagators.h

color.h

stitch.h

Step : Translation of the one-loop amplitude

QGRAF-GOLEM Interface

- ♠ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: QGRAF output → Feynman rules

- **User command:** perl run_golem.pl
- **Language:** bash, perl, FORM
- **Input:** qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat . Pre-processing options available (e.g. topology selection)
- **Modules involved:** QGRAF-GOLEM Interface
- **Output:** graphs_golem_tree/loop/counter.h :

- ♠ Different modules for different pieces of the amplitude:

vertices.h

propagators.h

color.h

stitch.h

dirac.h

Step 2: Translation of the one-loop amplitude

QGRAF-GOLEM Interface

- ♣ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: QGRAF output \Rightarrow Feynman rules

- User command: `perl run_golem.pl`
 - Language: bash, perl, FORM
 - Input: `qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat`. Pre-processing options available (e.g. [topology selection](#))
 - Modules involved: [QGRAF-GOLEM Interface](#)
 - Output: `graphs_golem_tree/loop/counter.h`:
- ♣ Different modules for different pieces of the amplitude:

`vertices.h``propagators.h``color.h``stitch.h``dirac.h``sign`

Step 2: Translation of the one-loop amplitude

QGRAF-GOLEM Interface

- ♠ Translates the symbolic QGRAF output into an algebraic expression and is further (pre)processed by FORM: QGRAF output \Rightarrow Feynman rules

- User command: `perl run_golem.pl`
- Language: bash, perl, FORM
- Input: `qgraf_lo.dat, qgraf_nlo.dat, qgraf_counter.dat`. Pre-processing options available (e.g. topology selection)
- Modules involved: QGRAF-GOLEM Interface
- Output: `graphs_golem_tree/loop/counter.h`:

- ♠ Different modules for different pieces of the amplitude:

`vertices.h`

`propagators.h`

`color.h`

`stitch.h`

`dirac.h`

`sign`

- ♠ Particular care devoted to consistently handle Majorana fermions [Denner et al.]

Step 2: Translation of the one-loop amplitude

One-loop amplitude upon translation: **GRAPH_MGOLEM_LOOP.h**

```
diagram1 = + Den(k1 + k2,0)*Den( - k3 - k4,0)*intM(Den(q1,0),Den(k1
+ k2 + q1,0))*SUNSum(Col14,3)*SUNSum(Glu15,8)*SUNSum(Col18,3)*
SUNT(Glu2,Col14,Col1)*SUNT(Glu15,Col3,Col18)*SUNT(Glu15,Col18,
Col14)*GG1^3*scalar3*Pi^(-2) * ( 1/32*Spinor(k4,MN1,-1)*g_(2,7,
k3,Lor5,k1,Lor5,k1,Lorhx2)*Spinor(k1,0,1)*e2(Lorhx2)*GULN1P2
+ 1/32*Spinor(k4,MN1,-1)*g_(2,7_,k3,Lor5,k1,Lor5,k2,Lorhx2)*
Spinor(k1,0,1)*e2(Lorhx2)*GULN1P2 + 1/32*Spinor(k4,MN1,-1)*g_(
2,7_,k3,Lor5,k2,Lor5,k1,Lorhx2)*Spinor(k1,0,1)*e2(Lorhx2)*
GULN1P2 + 1/32*Spinor(k4,MN1,-1)*g_(2,7_,k3,Lor5,k2,Lor5,k2,
Lorhx2)*Spinor(k1,0,1)*e2(Lorhx2)*GULN1P2 + 1/32*Spinor(k4,MN1
,-1)*g_(2,7_,k3,Lor5,q1,Lor5,k1,Lorhx2)*Spinor(k1,0,1)*
e2(Lorhx2)*GULN1P2 + 1/32*Spinor(k4,MN1,-1)*g_(2,7_,k3,Lor5,q1
,Lor5,k2,Lorhx2)*Spinor(k1,0,1)*e2(Lorhx2)*GULN1P2 + 1/32*
Spinor(k4,MN1,-1)*g_(2,7_,k4,Lor5,k1,Lor5,k1,Lorhx2)*Spinor(k1
,0,1)*e2(Lorhx2)*GULN1P2 + 1/32*Spinor(k4,MN1,-1)*g_(2,7_,k4,
Lor5,k1,Lor5,k2,Lorhx2)*Spinor(k1,0,1)*e2(Lorhx2)*GULN1P2 + 1/
32*Spinor(k4,MN1,-1)*g_(2,7_,k4,Lor5,k2,Lor5,k1,Lorhx2)*
Spinor(k1,0,1)*e2(Lorhx2)*GULN1P2 + 1/32*Spinor(k4,MN1,-1)*g_(
2,7_,k4,Lor5,k2,Lor5,k2,Lorhx2)*Spinor(k1,0,1)*e2(Lorhx2)*
GULN1P2 + 1/32*Spinor(k4,MN1,-1)*g_(2,7_,k4,Lor5,q1,Lor5,k1
,Lorhx2)*Spinor(k1,0,1)*e2(Lorhx2)*GULN1P2 + 1/32*Spinor(k4,MN1
,-1)*g_(2,7_,k4,Lor5,q1,Lor5,k2,Lorhx2)*Spinor(k1,0,1)*
```

Step 3: Calculation of the one-loop amplitude

FORM/Maple processing

- ♣ Works out the color, helicity & kinematic structure of the NLO amplitude, performs one-loop integral reduction and provides algebraic&numerical code
 - User command: `perl run_golem.pl`
 - Language: bash, perl, FORM, Maple
 - Input: `graphs_golem_tree/loop/counter.h`
 - Modules involved: GOLEM
 - Output: `amp_tree/loop/counter.mapout, virtual_corrections.f90` :

Step 3: Calculation of the one-loop amplitude

FORM/Maple processing

♣ Works out the color, helicity & kinematic structure of the NLO amplitude, performs one-loop integral reduction and provides algebraic&numerical code

- **User command:** perl run_golem.pl
- Language: bash, perl, FORM, Maple
- Input: graphs_golem_tree/loop/counter.h .
- Modules involved: GOLEM
- Output: amp_tree/loop/counter.mapout, virtual_corrections.f90 :

Step 3: Calculation of the one-loop amplitude

FORM/Maple processing

♣ Works out the color, helicity & kinematic structure of the NLO amplitude, performs one-loop integral reduction and provides algebraic&numerical code

- **User command:** perl run_golem.pl
- **Language:** bash, perl, FORM, Maple
- **Input:** graphs_golem_tree/loop/counter.h .
- **Modules involved:** GOLEM
- **Output:** amp_tree/loop/counter.mapout, virtual_corrections.f90 :

Step 3: Calculation of the one-loop amplitude

FORM/Maple processing

- ♣ Works out the color, helicity & kinematic structure of the NLO amplitude, performs one-loop integral reduction and provides algebraic&numerical code
 - **User command:** perl run_golem.pl
 - **Language:** bash, perl, FORM, Maple
 - **Input:** graphs_golem_tree/loop/counter.h .
 - **Modules involved:** GOLEM
 - **Output:** amp_tree/loop/counter.mapout, virtual_corrections.f90 :

Step 3: Calculation of the one-loop amplitude

FORM/Maple processing

- ♣ Works out the color, helicity & kinematic structure of the NLO amplitude, performs one-loop integral reduction and provides algebraic&numerical code
 - **User command:** perl run_golem.pl
 - **Language:** bash, perl, FORM, Maple
 - **Input:** graphs_golem_tree/loop/counter.h .
 - **Modules involved:** GOLEM
 - **Output:** amp_tree/loop/counter.mapout, virtual_corrections.f90 :

Step 3: Calculation of the one-loop amplitude

FORM/Maple processing

- ♣ Works out the color, helicity & kinematic structure of the NLO amplitude, performs one-loop integral reduction and provides algebraic&numerical code
 - **User command:** perl run_golem.pl
 - **Language:** bash, perl, FORM, Maple
 - **Input:** graphs_golem_tree/loop/counter.h .
 - **Modules involved:** GOLEM
 - **Output:** amp_tree/loop/counter.mapout, virtual_corrections.f90 :

Step 3: Calculation of the one-loop amplitude

FORM/Maple processing

◆ Works out the color, helicity & kinematic structure of the NLO amplitude, performs one-loop integral reduction and provides algebraic&numerical code

- User command: perl run_golem.pl
- Language: bash, perl, FORM, Maple
- Input: graphs_golem_tree/loop/counter.h .
- Modules involved: GOLEM
- Output: amp_tree/loop/counter.mapout, virtual_corrections.f90 :

Main modules

- color structure \Rightarrow color decomposition & saturation
- helicity structure \Rightarrow spinor-helicity formalism
- loop functions \Rightarrow GOLEM reduction algorithm

Step 3: Calculation of the one-loop amplitude

FORM/Maple processing

◆ Works out the color, helicity & kinematic structure of the NLO amplitude, performs one-loop integral reduction and provides algebraic&numerical code

- User command: perl run_golem.pl
- Language: bash, perl, FORM, Maple
- Input: graphs_golem_tree/loop/counter.h .
- Modules involved: GOLEM
- Output: amp_tree/loop/counter.mapout, virtual_corrections.f90 :

Main modules

- color structure \Rightarrow color decomposition & saturation
- helicity structure \Rightarrow spinor-helicity formalism
- loop functions \Rightarrow GOLEM reduction algorithm

Step 3: Calculation of the one-loop amplitude

FORM/Maple processing

◆ Works out the color, helicity & kinematic structure of the NLO amplitude, performs one-loop integral reduction and provides algebraic&numerical code

- User command: perl run_golem.pl
- Language: bash, perl, FORM, Maple
- Input: graphs_golem_tree/loop/counter.h .
- Modules involved: GOLEM
- Output: amp_tree/loop/counter.mapout, virtual_corrections.f90 :

Main modules

- color structure \Rightarrow color decomposition & saturation
- helicity structure \Rightarrow spinor-helicity formalism
- loop functions \Rightarrow GOLEM reduction algorithm

Step 3: Calculation of the one-loop amplitude

FORM/Maple processing

◆ Works out the color, helicity & kinematic structure of the NLO amplitude, performs one-loop integral reduction and provides algebraic&numerical code

- User command: `perl run_golem.pl`
- Language: bash, perl, FORM, Maple
- Input: `graphs_golem_tree/loop/counter.h`
- Modules involved: `GOLEM`
- Output: `amp_tree/loop/counter.mapout, virtual_corrections.f90` :

Main modules

- **color structure** ⇒ color decomposition & saturation
- **helicity structure** ⇒ spinor-helicity formalism
- **loop functions** ⇒ GOLEM reduction algorithm

`collect_tree/loop/counter.map`

`create_golem.map`

Detailed layout for the NLO amplitude

Decomposition of NLO contributions

Detailed layout for the NLO amplitude

Decomposition of NLO contributions

$$\mathcal{M}^{\text{NLO}} = \underbrace{\mathcal{M}_{\text{[color}}}^{\text{partial amplitudes}}_{\text{amplitudes}} \times \underbrace{\mathcal{B}_{\text{color}}}_{\text{basis}}$$

Detailed layout for the NLO amplitude

Decomposition of NLO contributions

$$\mathcal{M}^{\text{NLO}} = \underbrace{\mathcal{M}_{\text{[color/helicity}}}^{\text{partial amplitudes}} \times \underbrace{\mathcal{B}_{\text{color}} \otimes \mathcal{B}_{\text{hel}}}^{\text{basis}}$$

Detailed layout for the NLO amplitude

Decomposition of NLO contributions

$$\mathcal{M}^{\text{NLO}} = \underbrace{\mathcal{M}_{[\text{color}/\text{helicity}/\text{1L-function}]}^{\text{NLO}}}_{\text{partial amplitudes}} \times \underbrace{\mathcal{B}_{\text{color}} \otimes \mathcal{B}_{\text{hel}} \otimes \mathcal{B}_{\text{1Lfunction}}}_{\text{basis}}$$

Detailed layout for the NLO amplitude

Decomposition of NLO contributions

$$\mathcal{M}^{\text{NLO}} = \underbrace{\mathcal{M}_{[\text{color/helicity/1L-function}]}^{\text{NLO}}}_{\text{partial amplitudes}} \times \underbrace{\mathcal{B}_{\text{color}} \otimes \mathcal{B}_{\text{hel}} \otimes \mathcal{B}_{\text{1Lfunction}}}_{\text{basis}}$$

Including the Counterterms

$$\mathcal{L}_0 \rightarrow \mathcal{L}(Z_\phi^{1/2} \phi, Z_g g) = \mathcal{L}(\phi, g) + \delta \mathcal{L}(\phi, g, \delta Z_\phi, \delta g)$$

Detailed layout for the NLO amplitude

Decomposition of NLO contributions

$$\mathcal{M}^{\text{NLO}} = \underbrace{\mathcal{M}_{[\text{color/helicity/1L-function}]}^{\text{NLO}}}_{\text{partial amplitudes}} \times \underbrace{\mathcal{B}_{\text{color}} \otimes \mathcal{B}_{\text{hel}} \otimes \mathcal{B}_{\text{1Lfunction}}}_{\text{basis}}$$

Including the Counterterms

$$\mathcal{L}_0 \rightarrow \mathcal{L}(Z_\phi^{1/2} \phi, Z_g g) = \mathcal{L}(\phi, g) + \delta \mathcal{L}(\phi, g, \delta Z_\phi, \delta g)$$

$$\underbrace{\delta \mathcal{L}(\delta Z_\phi, \delta g)}_{\text{Models/vertex_ct.dat}} \Leftrightarrow \underbrace{\Sigma_q, \Sigma_{\tilde{q}}, \Sigma_g, \Sigma_{\tilde{g}}}_{\text{GOLEMproc/CT_list.map}} @ \mathcal{O}(\alpha_s)$$

Step 3: Calculation of the one-loop amplitude

One-loop amplitude upon translation: **GRAPH_MGOLEM_LOOP.h**

```
# Colour basis has 2 elements.  
#  
NCOLS := 2:  
COL[ 1] := dd(Col022,Col021)*dd(Col1,Col3):  
COL[ 2] := dd(Col022,Col3)*dd(Col1,Col021):  
NC := 3:  
TR := 1/2:  
NF := 5:  
#. # epsilon_tensor basis has 1 elements.  
#. NEPS := 1:  
EPSTEN[ 1] := 1:  
#. # Function basis has 4 elements.  
#. NUM_LOC_FUNS := 4:  
FUN[ 1] := BUBd4(S12,0,0):  
FUN[ 2] := BUBd4(S12,MUL2,MG02):  
FUN[ 3] := BUBd4EPS(S12,0,0):  
FUN[ 4] := BUBd4EPS(S12,MUL2,MG02):  
. # 12 helicity amplitudes found  
. #  
NUM_HELIS := 12:  
HELI[ 1]:=[1, 1, 5, 1]:  
HELI[ 2]:=[1, 1, 5, -1]:  
HELI[ 3]:=[1, 1, 5, 1]:  
HELI[ 4]:=[1, 1, 5, -1]:  
HELI[ 5]:=[1, 1, 5, 1]:  
HELI[ 6]:=[1, 1, 5, -1]:  
HELI[ 7]:=[1, 1, 5, 1]:  
HELI[ 8]:=[1, 1, 5, -1]:  
HELI[ 9]:=[1, 1, 5, 1]:  
HELI[10]:=[1, 1, 5, -1]:  
HELI[11]:=[1, 1, 5, 1]:  
HELI[12]:=[1, 1, 5, -1]:
```

Step 3: Calculation of the one-loop amplitude

One-loop amplitude upon translation: **GRAPH_MGOLEM_LOOP.h**

GRAPH_COEFF has indices: NGRAPH,NHELI,NCOL,NEPSTEN,NFUN

```

GRAPH_COEFF[ 1, 8, 1, 1, 1]:=-1/36*(-MUL2*S23+MUL2*MN12+S23^2+S12*S23-MN12*S23)*GULN1P2*GG1^3/Pi^2;
GRAPH_COEFF[ 1, 8, 2, 1, 1]:=1/12*(-MUL2*S23+MUL2*MN12+S23^2+S12*S23-MN12*S23)*GULN1P2*GG1^3/Pi^2;
GRAPH_COEFF[ 1, 8, 1, 1, 2]:=0;
GRAPH_COEFF[ 1, 8, 2, 1, 2]:=0;
GRAPH_COEFF[ 1, 8, 1, 1, 3]:=0;
GRAPH_COEFF[ 1, 8, 2, 1, 3]:=0;
GRAPH_COEFF[ 1, 8, 1, 1, 4]:=0;
GRAPH_COEFF[ 1, 8, 2, 1, 4]:=0;
GRAPH_COEFF[ 1, 8, 1, 1, 5]:=0;
GRAPH_COEFF[ 1, 8, 2, 1, 5]:=0;
GRAPH_COEFF[ 1, 8, 1, 1, 6]:=0;
GRAPH_COEFF[ 1, 8, 2, 1, 6]:=0;
GRAPH_COEFF[ 1, 8, 1, 1, 7]:=0;
GRAPH_COEFF[ 1, 8, 2, 1, 7]:=0;
GRAPH_COEFF[ 1, 8, 1, 1, 8]:=0;
GRAPH_COEFF[ 1, 8, 2, 1, 8]:=0;
GRAPH_COEFF[ 1, 8, 1, 1, 9]:=1/36*(-MUL2*S23+MUL2*MN12+S23^2+S12*S23-MN12*S23)*GULN1P2*GG1^3/Pi^2;
GRAPH_COEFF[ 1, 8, 2, 1, 9]:=-1/12*(-MUL2*S23+MUL2*MN12+S23^2+S12*S23-MN12*S23)*GULN1P2*GG1^3/Pi^2;
GRAPH_COEFF[ 1, 8, 1, 1, 10]:=0;
GRAPH_COEFF[ 1, 8, 2, 1, 10]:=0;
GRAPH_COEFF[ 1, 8, 1, 1, 11]:=0;
GRAPH_COEFF[ 1, 8, 2, 1, 11]:=0;
SPINOR_FAC[ 1, 8 ] := InvSpaa(k1,k2)*InvSpaa(k1,k4)*InvSpaa(k2,k3)*InvSpbb(k1,k3);

```

Step 3: Calculation of the one-loop amplitude

One-loop amplitude upon translation: **GRAPH_MGOLEM_LOOP.h**

GRAPH_COEFF has indices: NGRAPH,NHELI,NCOL,NEPSTEN,NFUN

```

GRAPH_COEFF[ 1, 8, 1, 1]:= -1/36*(-MUL2*S23+MUL2*MN12+S23^2+S12*S23-MN12*S23)*GULN1P2*GG1^3/Pi^2;
GRAPH_COEFF[ 1, 8, 2, 1]:= 1/12*(-MUL2*S23+MUL2*MN12+S23^2+S12*S23-MN12*S23)*GULN1P2*GG1^3/Pi^2;
GRAPH_COEFF[ 1, 8, 1, 2]:= 0;
GRAPH_COEFF[ 1, 8, 2, 2]:= 0;
GRAPH_COEFF[ 1, 8, 1, 3]:= 0;
GRAPH_COEFF[ 1, 8, 2, 3]:= 0;
GRAPH_COEFF[ 1, 8, 1, 4]:= 0;
GRAPH_COEFF[ 1, 8, 2, 4]:= 0;
GRAPH_COEFF[ 1, 8, 1, 5]:= 0;
GRAPH_COEFF[ 1, 8, 2, 5]:= 0;
GRAPH_COEFF[ 1, 8, 1, 6]:= 0;
GRAPH_COEFF[ 1, 8, 2, 6]:= 0;
GRAPH_COEFF[ 1, 8, 1, 7]:= 0;
GRAPH_COEFF[ 1, 8, 2, 7]:= 0;
GRAPH_COEFF[ 1, 8, 1, 8]:= 0;
GRAPH_COEFF[ 1, 8, 2, 8]:= 0;
GRAPH_COEFF[ 1, 8, 1, 9]:= 1/36*(-MUL2*S23+MUL2*MN12+S23^2+S12*S23-MN12*S23)*GULN1P2*GG1^3/Pi^2;
GRAPH_COEFF[ 1, 8, 2, 9]:= -1/12*(-MUL2*S23+MUL2*MN12+S23^2+S12*S23-MN12*S23)*GULN1P2*GG1^3/Pi^2;
GRAPH_COEFF[ 1, 8, 1, 10]:= 0;
GRAPH_COEFF[ 1, 8, 2, 10]:= 0;
GRAPH_COEFF[ 1, 8, 1, 11]:= 0;
GRAPH_COEFF[ 1, 8, 2, 11]:= 0;
SPINOR_FAC[ 1, 8, 1]:= InvSpaa(k1,k2)*InvSpaa(k1,k4)*InvSpaa(k2,k3)*InvSpbb(k1,k3);

```

Step 3: Calculation of the one-loop amplitude

One-loop amplitude upon translation: **GRAPH_MGOLEM_LOOP.h**

GRAPH_COEFF has indices: NGRAPH,NHELI,NCOL,NEPSTEN,NFUN

```

GRAPH_COEFF[ 1, 8, 1, 1, 1] = -1/36*(-MUL2*S23+MUL2*MN12+S23^2+S12*S23-MN12*S23)*GULN1P2*GG1^3/Pi^2;
GRAPH_COEFF[ 1, 8, 2, 1, 1] = 1/12*(-MUL2*S23+MUL2*MN12+S23^2+S12*S23-MN12*S23)*GULN1P2*GG1^3/Pi^2;
GRAPH_COEFF[ 1, 8, 1, 1, 2] = 0;
GRAPH_COEFF[ 1, 8, 2, 1, 2] = 0;
GRAPH_COEFF[ 1, 8, 1, 1, 3] = 0;
GRAPH_COEFF[ 1, 8, 2, 1, 3] = 0;
GRAPH_COEFF[ 1, 8, 1, 1, 4] = 0;
GRAPH_COEFF[ 1, 8, 2, 1, 4] = 0;
GRAPH_COEFF[ 1, 8, 1, 1, 5] = 0;
GRAPH_COEFF[ 1, 8, 2, 1, 5] = 0;
GRAPH_COEFF[ 1, 8, 1, 1, 6] = 0;
GRAPH_COEFF[ 1, 8, 2, 1, 6] = 0;
GRAPH_COEFF[ 1, 8, 1, 1, 7] = 0;
GRAPH_COEFF[ 1, 8, 2, 1, 7] = 0;
GRAPH_COEFF[ 1, 8, 1, 1, 8] = 0;
GRAPH_COEFF[ 1, 8, 2, 1, 8] = 0;
GRAPH_COEFF[ 1, 8, 1, 1, 9] = 1/36*(-MUL2*S23+MUL2*MN12+S23^2+S12*S23-MN12*S23)*GULN1P2*GG1^3/Pi^2;
GRAPH_COEFF[ 1, 8, 2, 1, 9] = -1/12*(-MUL2*S23+MUL2*MN12+S23^2+S12*S23-MN12*S23)*GULN1P2*GG1^3/Pi^2;
GRAPH_COEFF[ 1, 8, 1, 1, 10] = 0;
GRAPH_COEFF[ 1, 8, 2, 1, 10] = 0;
GRAPH_COEFF[ 1, 8, 1, 1, 11] = 0;
GRAPH_COEFF[ 1, 8, 2, 1, 11] = 0;
SPINOR_FAC[ 1, 8, 1] := InvSpaa(k1,k2)*InvSpaa(k1,k4)*InvSpaa(k2,k3)*InvSpbb(k1,k3);

```

Step 3: Calculation of the one-loop amplitude

One-loop amplitude upon translation: **GRAPH_MGOLEM_LOOP.h**

```
#####
# FIELD RENORMALIZATION CONSTANTS
#####
##### fieldRC_ur_qcd := [-4/3*GG2^2/(16*Pi^2),BUBd4(0,0,0),0]:
fieldRC_ur_qcd := [-4/3*GG2^2/(16*Pi^2),BUBd4(0,0,0),0];
##### fieldRC_ul_sqcd = [-4/3*GC^2/(16*Pi^2) BUBd4(0,MG02,MG02)
fieldRC_ul_sqcd = [-4/3*GC^2/(16*Pi^2) BUBd4(0,MG02,MG02);
##### fin1_uv_su:
fin1_uv_su:
##### fin2_uv_su:
fin2_uv_su:

#####
# COUPLING CONSTANT RENORMALIZATION.
#####
#####

# GS (MSbar scheme)
GSRC_qcd1 := [-BETA0/2, SINGLEPOLEd4,0];
GSRC_qcd2 := [-GC^2/(16*Pi^2)/3, 0, Log(MT^2/musq)];
GSRC_sqcd1 := [-GC^2/(16*Pi^2)/12,
0, Log((MUL2/musq))+Log((MCL2/musq))+Log((MLT2/musq))+Log((MUR2/musq))+Log((MCR2/musq))+Log((MTR2/musq))];
GSRC_sqcd2 := [-GC^2/(16*Pi^2)/12,
0, Log((MDL2/musq))+Log((MSL2/musq))+Log((MDR2/musq))+Log((MSR2/musq))+Log((MBR2/musq))];
GSRC_sqcd3 := [-GC^2/(16*Pi^2), 0, Log((MG0^2/musq))];

#####
# VERTEX AND SELF-ENERGY COUNTERTERMS - Models included: SM and MSSM
#####
#####

[GULN1PCT2_qcd1, GULN1P2*fieldRC_sul_qcd1[1]/2,fieldRC_sul_qcd1[2],fieldRC_sul_qcd1[3],1],
[GULN1PCT2_qcd2, GULN1P2*fieldRC_sul_qcd2[1]/2,fieldRC_sul_qcd2[2],fieldRC_sul_qcd2[3],1],
[GULN1PCT2_qcd3, GULN1P2*fieldRC_ul_qcd1[1]/2,fieldRC_ul_qcd2[2],fieldRC_ul_qcd3[1],
[GULN1PCT1_qcd1, GULN1PCT1_qcd1, GURN1M2*fieldRC_sur_qcd1[1]/2,fieldRC_sur_qcd1[2],fieldRC_sur_qcd1[3],1,
[GULN1PCT1_qcd2, GURN1M2*fieldRC_sur_qcd2[1]/2,fieldRC_sur_qcd2[2],fieldRC_sur_qcd2[3],1],
[GULN1PCT2_sqcd1, GULN1P2*fieldRC_sul_sqcd1[1]/4,fieldRC_sul_sqcd2[2],fieldRC_sul_sqcd3[2],1],
[GULN1PCT2_sqcd2, GULN1P2*fieldRC_ul_sqcd1[1]/2,fieldRC_ul_sqcd2[2],fieldRC_ul_sqcd3[2],1],
[GULN1PCT2_sqcd3, GULN1P2*SUSYRESTORATIONG2[1],SUSYRESTORATIONG2[2],SUSYRESTORATIONG2[3],2],
```

Handling the UV divergences

Renormalization scheme

- $\overline{\text{MS}}$, for the field-strength RCs of the massless particles
- $\overline{\text{OS}}$, for the field-strength RCs of the massive particles
- $\overline{\text{MS}}$ /zero-momentum , for g_s [Beenakker et al, Berge et al]
- SUSY breaking from Dimensional Regularization restored through additional finite CTs [Martin, Vaughn; Beenakker et al].

Handling the UV divergences

Renormalization scheme

- $\overline{\text{MS}}$, for the field-strength RCs of the massless particles
 - $\overline{\text{OS}}$, for the field-strength RCs of the massive particles
 - $\overline{\text{MS}}$ /zero-momentum , for g_s [Beenakker et al, Berge et al]
 - SUSY breaking from Dimensional Regularization restored through additional finite CTs [Martin, Vaughn; Beenakker et al].
- ♠ Only field-strength RCs of colored fields and g_s renormalization matter
- Highly generic structures \Rightarrow Lorentz & color representations
 - Currently implemented for the SM & MSSM \Leftrightarrow straightforward to extend !

Handling the IR divergences

- ♣ Dipole Subtraction: [Catani, Seymour; Catani, Dittmaier, Seymour, Trocsanyi]

$$\sigma = \int_m d\sigma^B + \int_{m+1} d\sigma^R + \int_m \left[\int_1 d\sigma^V \right]$$

Handling the IR divergences

♠ Dipole Subtraction: [Catani, Seymour; Catani, Dittmaier, Seymour, Trocsanyi]

$$\sigma = \int_m d\sigma^B + \underbrace{\int_{m+1} d\sigma^R}_{\int \frac{dk}{k^{1+\epsilon}} \sim \frac{1}{\epsilon_{IR}}, \int \frac{d\theta}{\theta^{1+\epsilon}} \sim \frac{1}{\epsilon_{IR}}} + \int_m \left[\int_1 \underbrace{d\sigma^V}_{\int \frac{dk}{k^{1+\epsilon}} \sim \frac{1}{\epsilon_{IR}}} \right]$$

Handling the IR divergences

♠ Dipole Subtraction: [Catani, Seymour; Catani, Dittmaier, Seymour, Trocsanyi]

$$\sigma = \int_m d\sigma^B + \underbrace{\int_{m+1} d\sigma^R - d\sigma^A}_{\frac{1}{\epsilon_{IR}} - \frac{1}{\epsilon_{IR}}} + \int_m \left[\int_1 d\sigma^V + \underbrace{d\sigma^A}_{\frac{1}{\epsilon_{IR}} - \frac{1}{\epsilon_{IR}}} \right]$$

Handling the IR divergences

♠ Dipole Subtraction: [Catani, Seymour; Catani, Dittmaier, Seymour, Trocsanyi]

$$\sigma = \int_m d\sigma^B + \underbrace{\int_{m+1} d\sigma^R - d\sigma^A}_{\frac{1}{\epsilon_{IR}} - \frac{1}{\epsilon_{IR}}} + \int_m \left[\int_1 d\sigma^V + \underbrace{d\sigma^A}_{\frac{1}{\epsilon_{IR}} - \frac{1}{\epsilon_{IR}}} \right]$$

Local (pointwise) subtraction of the IR poles

- Based on factorization of collinear&soft singularities
- Process-independent
- Analytically integrable over the single-parton phase-space containing the divergences

Handling the IR divergences

♠ Dipole Subtraction: [Catani, Seymour; Catani, Dittmaier, Seymour, Trocsanyi]

$$\sigma = \int_m d\sigma^B + \underbrace{\int_{m+1} d\sigma^R - d\sigma^A}_{\frac{1}{\epsilon_{IR}} - \frac{1}{\epsilon_{IR}}} + \int_m \left[\int_1 \underbrace{d\sigma^V + d\sigma^A}_{\frac{1}{\epsilon_{IR}} - \frac{1}{\epsilon_{IR}}} \right]$$

Local (pointwise) subtraction of the IR poles

- Based on factorization of collinear&soft singularities
- Process-independent
- Analytically integrable over the single-parton phase-space containing the divergences

$$\begin{aligned} d\sigma^A &= \sum_l f(\epsilon_{IR}) \times d\sigma_l^B \otimes \textcolor{green}{V}_l \\ \int_{m+1} d\sigma^A &= \sum_l \int_m f(\epsilon_{IR}) \times d\sigma_l^B \otimes \int_1 \textcolor{green}{V}_l = f(\epsilon_{IR}) \times \int_m d\sigma_l^B \otimes \textcolor{orange}{I} \end{aligned}$$

ISUSY (including α -dependence [Nagy, Trocsanyi]) available @ MadGOLEM

Handling the OS Divergences

Automatized OS Subtraction available @ MadGOLEM [Beenakker, Höpker, Spira, Zerwas]

$$\begin{aligned} d\sigma^R &\longrightarrow d\sigma^R \Big]_{\text{regular}} + d\sigma^{R*} \Big]_{\mathcal{O}(1/(p^2 - m^2))} \\ ug \rightarrow \tilde{u}_L \tilde{\chi}_1 j &+ uu \rightarrow \tilde{u}_L \tilde{u}_L^* \rightarrow \tilde{u}_L \tilde{\chi}_1 j \end{aligned}$$

Handling the OS Divergences

Automatized OS Subtraction available @ MadGOLEM [Beenakker, Höpker, Spira, Zerwas]

$$\begin{aligned} d\sigma^R &\longrightarrow d\sigma^R \Big]_{\text{regular}} + d\sigma^{R*} \Big]_{\mathcal{O}(1/(p^2 - m^2))} \\ ug \rightarrow \tilde{u}_L \tilde{\chi}_1 j &+ uu \rightarrow \tilde{u}_L \tilde{u}_L^* \rightarrow \tilde{u}_L \tilde{\chi}_1 j \end{aligned}$$

$$\sigma = \int_{m+1} d\sigma^R$$

Handling the OS Divergences

Automatized OS Subtraction available @ MadGOLEM [Beenakker, Höpker, Spira, Zerwas]

$$\begin{aligned} d\sigma^R &\longrightarrow d\sigma^R \Big]_{\text{regular}} + d\sigma^{R*} \Big]_{\mathcal{O}(1/(p^2 - m^2))} \\ ug \rightarrow \tilde{u}_L \tilde{\chi}_1 j &+ uu \rightarrow \tilde{u}_L \tilde{u}_L^* \rightarrow \tilde{u}_L \tilde{\chi}_1 j \end{aligned}$$

$$\sigma = \int_{m+1} d\sigma^R \longrightarrow \int_{m+1} \left[d\sigma^R + d\sigma^{R*}(\Gamma_{\tilde{u}_L}) - d\sigma^{OS}(\Gamma_{\tilde{u}_L}) \right]$$

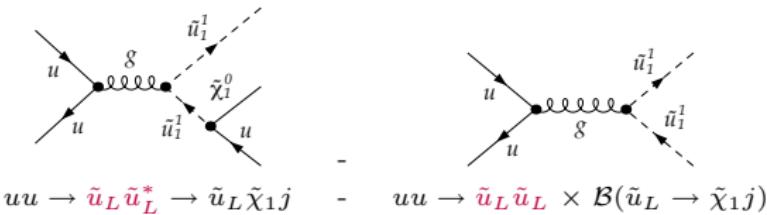
Handling the OS Divergences

Automatized OS Subtraction available @ MadGOLEM [Beenakker, Höpker, Spira, Zerwas]

$$d\sigma^R \longrightarrow d\sigma^R \Big]_{\text{regular}} + d\sigma^{R*} \Big]_{\mathcal{O}(1/(p^2 - m^2))}$$

$$ug \rightarrow \tilde{u}_L \tilde{\chi}_1 j + uu \rightarrow \tilde{u}_L \tilde{u}_L^* \rightarrow \tilde{u}_L \tilde{\chi}_1 j$$

$$\sigma = \int_{m+1} d\sigma^R \longrightarrow \int_{m+1} \left[d\sigma^R + d\sigma^{R*}(\Gamma_{\tilde{u}_L}) - d\sigma^{OS}(\Gamma_{\tilde{u}_L}) \right]$$



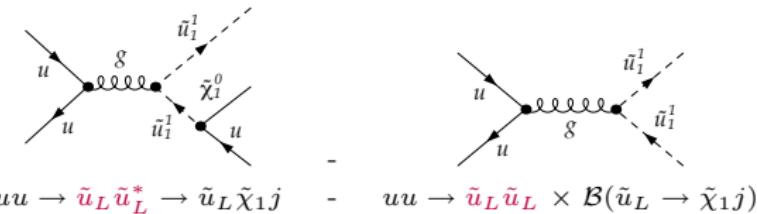
Handling the OS Divergences

Automatized OS Subtraction available @ MadGOLEM [Beenakker, Höpker, Spira, Zerwas]

$$d\sigma^R \longrightarrow d\sigma^R \Big]_{\text{regular}} + d\sigma^{R*} \Big]_{\mathcal{O}(1/(p^2 - m^2))}$$

$$ug \rightarrow \tilde{u}_L \tilde{\chi}_1 j + uu \rightarrow \tilde{u}_L \tilde{u}_L^* \rightarrow \tilde{u}_L \tilde{\chi}_1 j$$

$$\sigma = \int_{m+1} d\sigma^R \longrightarrow \int_{m+1} \left[d\sigma^R + d\sigma^{R*}(\Gamma_{\tilde{u}_L}) - d\sigma^{OS}(\Gamma_{\tilde{u}_L}) \right]$$



$$\frac{d\sigma^{OS}}{dM^2} = \sigma^{Born} \frac{m_{\tilde{u}_L} \Gamma_{\tilde{u}_L}/\pi}{(M^2 - m_{\tilde{u}_L}^2) + m^2 \Gamma_{\tilde{u}_L}^2} + \mathcal{O}\left(\frac{1}{(M^2 - m_{\tilde{u}_L}^2)}\right)$$

- Pointwise subtraction of the OS poles – analogue to CS dipoles
- Avoids double-counting & preserves gauge invariance
- $\Gamma_{\tilde{u}_L}$ as regulator \Rightarrow dependence cancels in the final results

Outline

- 1 Motivating MadGOLEM
- 2 MadGOLEM from inside: architecture of the code
- 3 MadGOLEM from the very inside: finer details
 - The MadGOLEM one-loop Calculator
 - Handling the divergences
- 4 MadGOLEM from outside: running MadGOLEM
- 5 MadGOLEM underway: performance and cross-checks
- 6 Beyond cross-checks: New Physics studies with MadGOLEM
- 7 Conclusions & Outlook

The user's perspective: running MadGOLEM

3-stage procedure – 3 interfaces \leftrightarrow 3 executables

The user's perspective: running MadGOLEM

3-stage procedure – 3 interfaces \leftrightarrow 3 executables

Stage 1: DEFINING THE PROCESS

process_card \leftrightarrow ./newprocess_nlo

- ♠ To be provided by the user:
 - P1: $pp \rightarrow ab$, LO process
 - P2: $pp \rightarrow abj$, NLO process (**virtual** contributions)
 - P3: $pp \rightarrow abj$, NLO process (**real emission** contributions)
 - P(k): $pp \rightarrow ab^{'*} \rightarrow abj$, $k = 4, \dots n$, NLO processes which subtract the OS divergences arising in P3
 - Type of corrections to be included: pure QCD, SUSY-QCD ...

The user's perspective: running MadGOLEM

Stage 2: COMPUTING THE AMPLITUDE

```
./run_golem_pl
```

- ◆ At this point the user is able to:

- Select diagram topologies \Rightarrow detailed analysis of the virtual corrections
- Access the **analytical output** in several stages \Rightarrow very useful for cross-checking (and to dig out some physics!)
- Employ a handful of debugging options \Rightarrow eases the implementation of in-house arrangements

The user's perspective: running MadGOLEM

Stage 2: COMPUTING THE AMPLITUDE

```
./run_golem_pl
```

- ♣ At this point the user is able to:

- Select diagram topologies \Rightarrow detailed analysis of the virtual corrections
- Access the analytical output in several stages \Rightarrow very useful for cross-checking (and to dig out some physics!)
- Employ a handful of debugging options \Rightarrow eases the implementation of in-house arrangements

The user's perspective: running MadGOLEM

Stage 2: COMPUTING THE AMPLITUDE

```
./run_golem_pl
```

- ◆ At this point the user is able to:

- Select diagram topologies \Rightarrow detailed analysis of the virtual corrections
- Access the **analytical output** in several stages \Rightarrow very useful for cross-checking (and to dig out some physics!)
- Employ a handful of debugging options \Rightarrow eases the implementation of in-house arrangements

The user's perspective: running MadGOLEM

Stage 2: COMPUTING THE AMPLITUDE

```
./run_golem_pl
```

- ◆ At this point the user is able to:

- Select diagram topologies \Rightarrow detailed analysis of the virtual corrections
- Access the **analytical output** in several stages \Rightarrow very useful for cross-checking (and to dig out some physics!)
- Employ a handful of debugging options \Rightarrow eases the implementation of in-house arrangements

The user's perspective: running MadGOLEM

Stage 2: COMPUTING THE AMPLITUDE

```
./run_golem_pl
```

♣ At this point the user is able to:

- Select diagram topologies \Rightarrow detailed analysis of the virtual corrections
- Access the **analytical output** in several stages \Rightarrow very useful for cross-checking (and to dig out some physics!)
- Employ a handful of debugging options \Rightarrow eases the implementation of in-house arrangements

Stage 3: GETTING THE NUMBERS (cross-sections, parton-level events)

```
run_card.dat ↔ ./generate_events_nlo 2 2 myrun
```

The user's perspective: running MadGOLEM

Stage 2: COMPUTING THE AMPLITUDE

```
./run_golem_pl
```

♣ At this point the user is able to:

- Select diagram topologies \Rightarrow detailed analysis of the virtual corrections
- Access the **analytical output** in several stages \Rightarrow very useful for cross-checking (and to dig out some physics!)
- Employ a handful of debugging options \Rightarrow eases the implementation of in-house arrangements

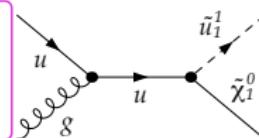
Stage 3: GETTING THE NUMBERS (cross-sections, parton-level events)

```
run_card.dat ↔ ./generate_events_nlo 2 2 myrun
```

♣ Model+ Process \rightarrow numbers in 3 steps! ... and yet the code is largely flexible/extendable

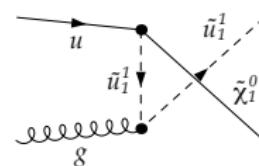
Setting up a $2 \rightarrow 2$ process NLO

```
pp>uln1 @1 # for L0
QCD=99          # Max QCD couplings
QED=2          # Max QED couplings
end_coup       # End the couplings input
```



```
pp>uln1j @2 # for NLO (virtual+integrated-dipole)
QCD=99          # Max QCD couplings
QED=2          # Max QED couplings
end_coup       # End the couplings input
```

```
pp>uln1j @3 # for NLO (unintegrated-dipole)
QCD=99          # Max QCD couplings
QED=2          # Max QED couplings
end_coup       # End the couplings input
```



$pp \rightarrow X_1 X_2$

P1: Leading-Order cross section

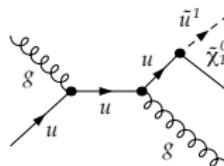
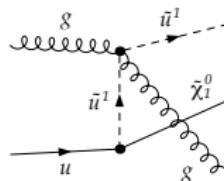
$$\sigma = \int_m d\sigma^B + \int_{m+1} (d\sigma^R - d\sigma^A - d\sigma^{OS}) + \int_m \left[\int_1 d\sigma^A + d\sigma^V \right]$$

Setting up a $2 \rightarrow 2$ process NLO

```
pp>uln1      @1 # for LO
QCD=99        # Max QCD couplings
QED=2         # Max QED couplings
end_coup      # End the couplings input
```

```
pp>uln1j     @2 # for NLO (virtual+integrated-dipole)
QCD=99        # Max QCD couplings
QED=2         # Max QED couplings
end_coup      # End the couplings input
```

```
pp>uln1j     @3 # for NLO (unintegrated-dipole)
QCD=99        # Max QCD couplings
QED=2         # Max QED couplings
end_coup      # End the couplings input
```



$$pp \rightarrow X_1 X_2 + j$$

P2: NLO cross section - dipole subtraction

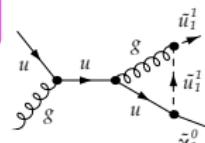
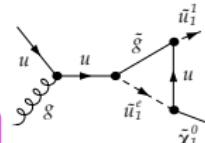
$$\sigma = \int_m d\sigma^B + \int_{m+1} (d\sigma^R - \textcolor{red}{d\sigma^A} - d\sigma^{\text{OS}}) + \int_m \left[\int_1 \textcolor{red}{d\sigma^A} + d\sigma^V \right]$$

Setting up a $2 \rightarrow 2$ process NLO

```
pp>uln1      @1 # for L0
QCD=99          # Max QCD couplings
QED=2           # Max QED couplings
end_coup        # End the couplings input
```

```
pp>uln1j     @2 # for NLO (virtual+integrated-dipole)
QCD=99          # Max QCD couplings
QED=2           # Max QED couplings
end_coup        # End the couplings input
```

```
pp>uln1j     @3 # for NLO (unintegrated-dipole)
QCD=99          # Max QCD couplings
QED=2           # Max QED couplings
end_coup        # End the couplings input
```



$$pp \rightarrow X_1 X_2 + j$$

P2: NLO cross section - virtual corrections

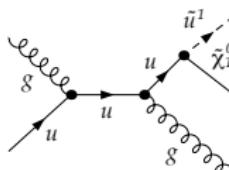
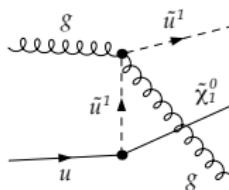
$$\sigma = \int_m d\sigma^B + \int_{m+1} (d\sigma^R - d\sigma^A) - d\sigma^{\text{OS}} + \int_m \left[\int_1 d\sigma^A + \color{red} d\sigma^V \right]$$

Setting up a $2 \rightarrow 2$ process NLO

```
pp>uln1      @1 # for L0
QCD=99          # Max QCD couplings
QED=2           # Max QED couplings
end_coup        # End the couplings input
```

```
pp>uln1j     @2 # for NLO (virtual+integrated-dipole)
QCD=99          # Max QCD couplings
QED=2           # Max QED couplings
end_coup        # End the couplings input
```

```
pp>uln1j     @3 # for NLO (unintegrated-dipole)
QCD=99          # Max QCD couplings
QED=2           # Max QED couplings
end_coup        # End the couplings input
```



$$pp \rightarrow X_1 X_2 + j$$

P3: NLO cross section - contributions from real emission

$$\sigma = \int_m d\sigma^B + \int_{m+1} (d\sigma^R - d\sigma^A - d\sigma^{OS}) + \int_m \left[\int_1 d\sigma^A + d\sigma^V \right]$$

Setting up a $2 \rightarrow 2$ process NLO

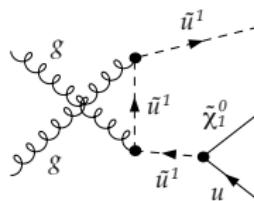
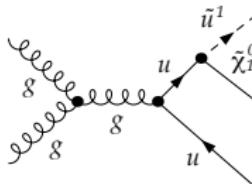
```

pp>ul~>uln1q {4} @5 # for NLO (unintegrated-dipole)
QCD=99           # Max QCD couplings
QED=2           # Max QED couplings
end_coup        # End the couplings input

pp>ur>uln1q {4} @6 # for NLO (unintegrated-dipole)
QCD=99           # Max QCD couplings
QED=2           # Max QED couplings
end_coup        # End the couplings input

pp>go>uln1q {3} @7 # for NLO (unintegrated-dipole)
QCD=99           # Max QCD couplings
QED=2           # Max QED couplings
end_coup        # End the couplings input

```



$$pp \rightarrow X_1 X_2^* \rightarrow X_1 X_2 + j$$

P4: Subtraction of On-Shell Divergences

$$\sigma = \int_m d\sigma^B + \int_{m+1} (\textcolor{red}{d\sigma^R} - d\sigma^A - \textcolor{red}{d\sigma^{OS}}) + \int_m \left[\int_1 d\sigma^A + d\sigma^V \right]$$

Running a $2 \rightarrow 2$ process NLO

(Loading screencasts_talk/newprocessnlo.mp4)

Running a $2 \rightarrow 2$ process NLO

(Loading screencasts_talk/rungolem.mp4)

Reading out the results for $2 \rightarrow 2$ NLO

Process results

$$s = 434.644 \pm 0.860(\text{ab})$$

Graph	Cross Sect(ab)	Error(ab)	Events (K)	Eff	Unwgt	Luminosity
Sum	434.644	0.860		1	0.1	
Sub Group 1						
P1_gu_uln1	292.660	0.238	0	0.0		0.00
P1_ug_uln1	291.890	0.240	0	0.0		0.00
Sub Group total = 584.54999999999995						
Sub Group 3						
P3_gu_uln1g	0.600	0.002	0	0.0		0.00
P3_ug_uln1g	0.600	0.002	0	0.0		0.00
P3_gg_uln1ux	0.146	0.002	0	0.1		0.00
P3_uu_uln1u	0.079	0.230	0	29.1		0.00
P3_uux_uln1ux	0.027	0.007	0	2.5		0.00
P3_uxu_uln1ux	0.027	0.005	0	2.0		0.00
Sub Group total = 1.4775809999999998						
Sub Group 2						
P2_ug_uln1g	0.000	0.000	0	0.0		*****
P2_gu_uln1g	0.000	0.000	0	0.0		*****
P2_gg_uln1ux	-2.965	0.002	0	0.0		*****
P2_uxu_uln1ux	-4.530	0.004	0	0.0		*****
P2_uux_uln1ux	-4.538	0.004	0	0.0		*****
P2_uu_uln1u	-139.350	0.122	0	0.0		*****
Sub Group total = -151.383500000000000						

Outline

- 1 Motivating MadGOLEM
- 2 MadGOLEM from inside: architecture of the code
- 3 MadGOLEM from the very inside: finer details
 - The MadGOLEM one-loop Calculator
 - Handling the divergences
- 4 MadGOLEM from outside: running MadGOLEM
- 5 MadGOLEM underway: performance and cross-checks
- 6 Beyond cross-checks: New Physics studies with MadGOLEM
- 7 Conclusions & Outlook

Validation strategies

♣ ONE-LOOP CORRECTIONS & COUNTERTERMS

- gauge invariance
- cancellation of UV/IR divergences (analytically & numerically)
- Finite parts – numerical comparison with FeynArts/FormCalc/LoopTools

[Hahn]

Validation strategies

♣ ONE-LOOP CORRECTIONS & COUNTERTERMS

- gauge invariance
- cancellation of UV/IR divergences (analytically & numerically)
- Finite parts – numerical comparison with [FeynArts](#)/[FormCalc](#)/[LoopTools](#)

[Hahn]

Validation strategies

♣ ONE-LOOP CORRECTIONS & COUNTERTERMS

- gauge invariance
- cancellation of UV/IR divergences (analytically & numerically)
- Finite parts – numerical comparison with **FeynArts/FormCalc/LoopTools**

[Hahn]

Validation strategies

♣ ONE-LOOP CORRECTIONS & COUNTERTERMS

- gauge invariance
- cancellation of UV/IR divergences (analytically & numerically)
- Finite parts – numerical comparison with **FeynArts/FormCalc/LoopTools**

[Hahn]

♣ REAL EMISSION, Dipoles AND OS SUBTRACTION

- $\alpha(\alpha_{OS})$ dependence & behavior in the soft and collinear limits, for all dipoles [Nagy, Trocsanyi] and OS subtraction terms
- cancellation of IR/OS divergences – numerical stability and convergence

Validation strategies

♣ ONE-LOOP CORRECTIONS & COUNTERTERMS

- gauge invariance
- cancellation of UV/IR divergences (analytically & numerically)
- Finite parts – numerical comparison with **FeynArts/FormCalc/LoopTools**

[Hahn]

♣ REAL EMISSION, Dipoles AND OS SUBTRACTION

- $\alpha(\alpha_{OS})$ dependence & behavior in the soft and collinear limits, for all dipoles [Nagy, Trocsanyi] and OS subtraction terms
- cancellation of IR/OS divergences – numerical stability and convergence

Validation strategies

♣ ONE-LOOP CORRECTIONS & COUNTERTERMS

- gauge invariance
- cancellation of UV/IR divergences (analytically & numerically)
- Finite parts – numerical comparison with **FeynArts/FormCalc/LoopTools**

[Hahn]

♣ REAL EMISSION, Dipoles AND OS SUBTRACTION

- $\alpha(\alpha_{OS})$ dependence & behavior in the soft and collinear limits, for all dipoles [Nagy, Trocsanyi] and OS subtraction terms
- cancellation of IR/OS divergences – numerical stability and convergence

Validation strategies

♣ ONE-LOOP CORRECTIONS & COUNTERTERMS

- gauge invariance
- cancellation of UV/IR divergences (analytically & numerically)
- Finite parts – numerical comparison with **FeynArts/FormCalc/LoopTools**

[Hahn]

♣ REAL EMISSION, Dipoles AND OS SUBTRACTION

- $\alpha(\alpha_{OS})$ dependence & behavior in the soft and collinear limits, for all dipoles [Nagy, Trocsanyi] and OS subtraction terms
- cancellation of IR/OS divergences – numerical stability and convergence

Validation strategies

♣ ONE-LOOP CORRECTIONS & COUNTERTERMS

- gauge invariance
- cancellation of UV/IR divergences (analytically & numerically)
- Finite parts – numerical comparison with [FeynArts](#)/[FormCalc](#)/[LoopTools](#)

[Hahn]

♣ REAL EMISSION, Dipoles AND OS SUBTRACTION

- $\alpha(\alpha_{OS})$ dependence & behavior in the soft and collinear limits, for all dipoles [\[Nagy, Trocsanyi\]](#) and [OS subtraction terms](#)
- cancellation of IR/OS divergences – numerical stability and convergence

♣ DIRECT COMPARISON

when available, with independent NLO predictions in a process-by-process basis:

- $e^+e^- \rightarrow \tilde{u}_L \tilde{u}_L^*$: [Drees, Hikasa; Beenakker et al. \['90\]](#)
- $pp \rightarrow \tilde{e}_L \tilde{e}_L$: [Beenakker, Klasen, Krämer, Plehn, Spira, Zerwas](#)

Performance of the loop calculator

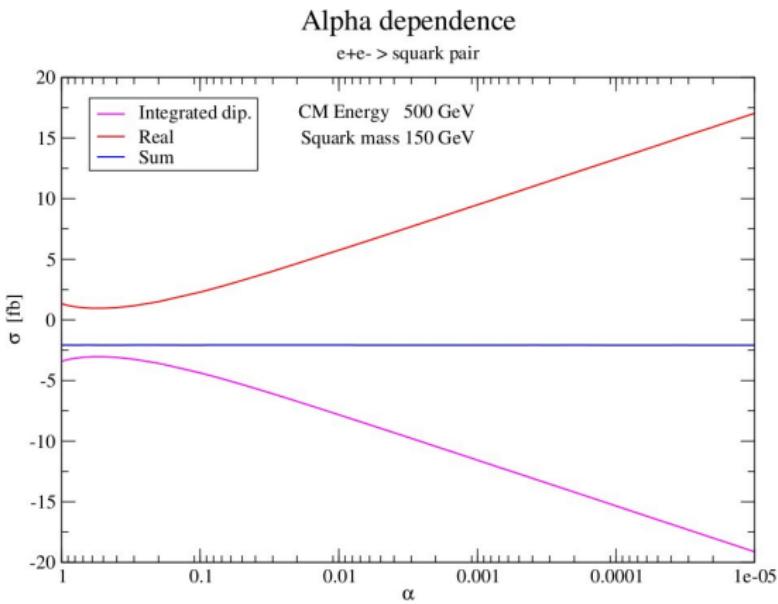
- ♣ An explicit example: SUSY-QCD 1-loop **virtual** corrections to $pp \rightarrow \tilde{u}_L \tilde{\chi}_1$

SPS1a

	σ^{NLO} (MadGOLEM)	σ^{NLO} (FormCalc)
$g - \tilde{u}_L - \tilde{u}_L^*$ vertex	$-1.173(2) \times 10^{-5}$	$-1.17(1) \times 10^{-5}$
$g - u - \bar{u}$ vertex	$8.202(7) \times 10^{-5}$	$8.20(8) \times 10^{-5}$
$\tilde{\chi}_1 - u - \tilde{u}_L$ vertex	$1.155(1) \times 10^{-4}$	$1.15(2) \times 10^{-4}$
u self-energy	$-5.006(4) \times 10^{-5}$	$-5.01(3) \times 10^{-5}$
\tilde{u}_L self-energy	$1.173(2) \times 10^{-5}$	$1.17(1) \times 10^{-5}$
boxes	$9.576(8) \times 10^{-5}$	$9.57(1) \times 10^{-5}$

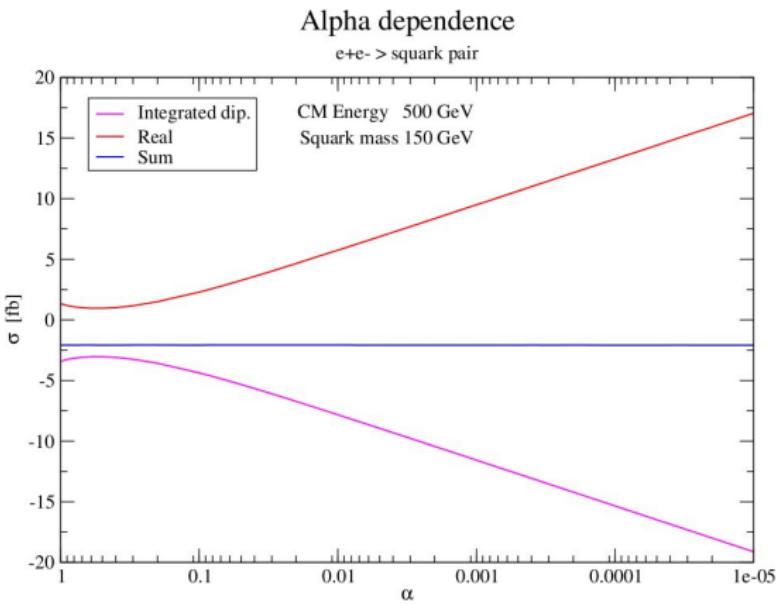
Performance of the SUSY dipoles

$$e^+ e^- \rightarrow \tilde{u}_L^* \tilde{u}_L$$



Performance of the SUSY dipoles

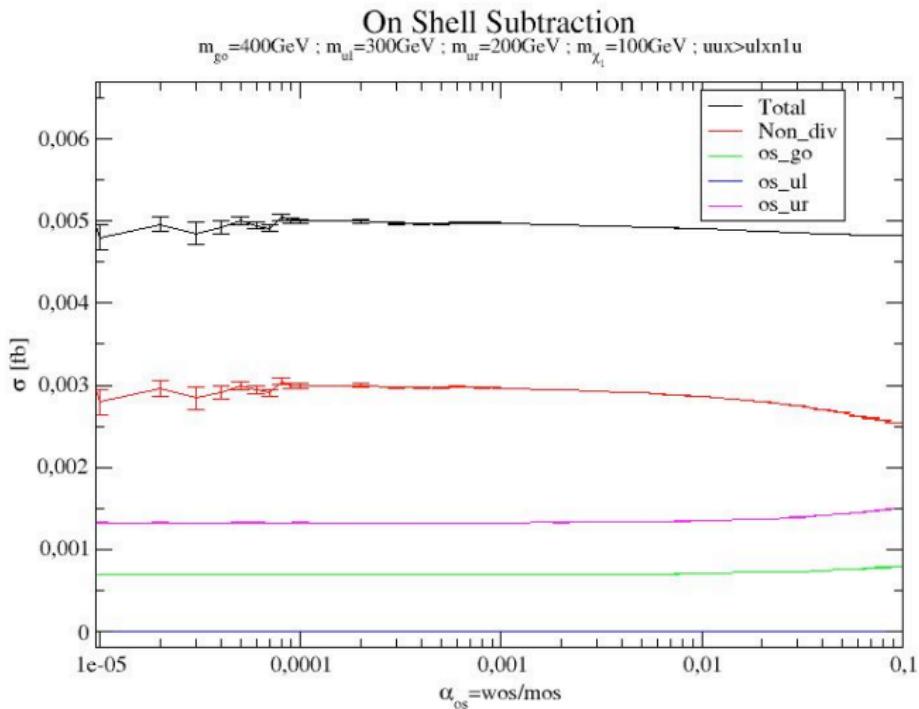
$$e^+ e^- \rightarrow \tilde{u}_L^* \tilde{u}_L$$



- ♠ For details cf. Dorival Gonçalves Netto's talk in the afternoon sessions & Gonçalves Netto, DLV, Mawatari, Plehn, Wigmore, to appear.

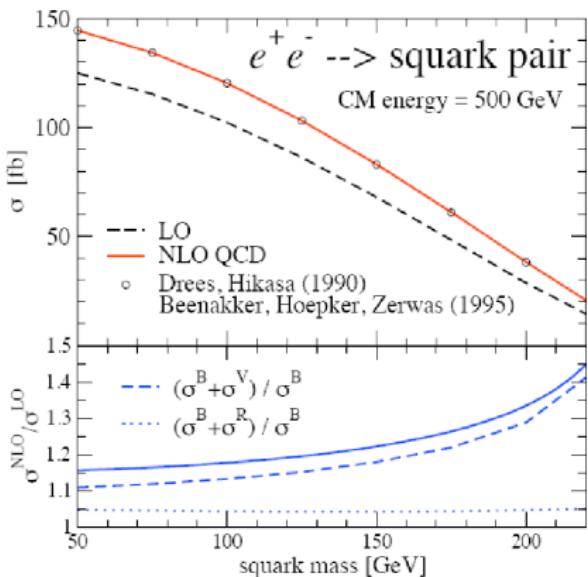
Performance of the OS subtraction

$$pp \rightarrow \tilde{u}\tilde{\chi}_1$$



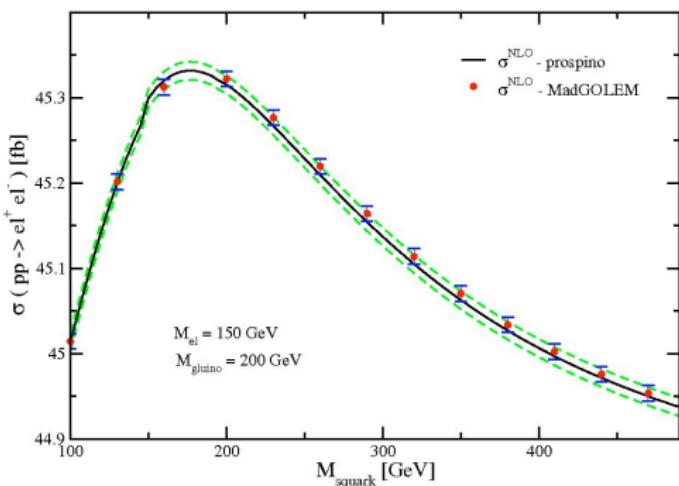
Process-by-process cross-check: $e^+e^- \rightarrow \tilde{u}_L^*\tilde{u}_L$

$$e^+e^- \rightarrow \tilde{u}_L^*\tilde{u}_L$$



Process-by-process cross-check: $\text{pp} \rightarrow \tilde{e}_L \tilde{e}_L^*$

$\text{pp} \rightarrow \tilde{e}_L \tilde{e}_L^*$

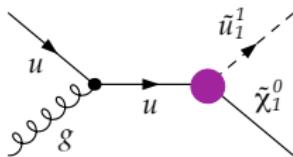


- Event rate as a function of the Renormalization Scale
- **MadGOLEM vs Prospino – Excellent agreement within integration errors**

Outline

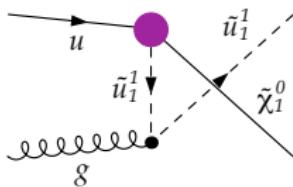
- 1 Motivating MadGOLEM
- 2 MadGOLEM from inside: architecture of the code
- 3 MadGOLEM from the very inside: finer details
 - The MadGOLEM one-loop Calculator
 - Handling the divergences
- 4 MadGOLEM from outside: running MadGOLEM
- 5 MadGOLEM underway: performance and cross-checks
- 6 Beyond cross-checks: New Physics studies with MadGOLEM
- 7 Conclusions & Outlook

MadGOLEM taking off: $\text{pp} \rightarrow \tilde{u}\tilde{\chi}^0$ at NLO

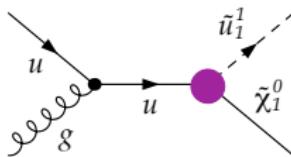


$$\text{pp} \rightarrow \tilde{u}_L \tilde{\chi}_1^0 \rightarrow u \tilde{\chi}_1^0 \tilde{\chi}_1^0$$

- Potential source of **Monojet signatures**
- Characteristic p_T spectrum
- Sensitivity to $q\bar{q}\tilde{\chi}_1^0$ couplings

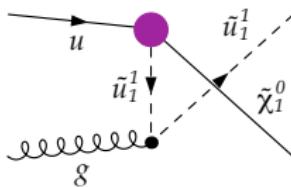


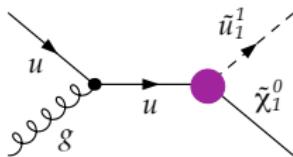
MadGOLEM taking off: $pp \rightarrow \tilde{u}\tilde{\chi}^0$ at NLO



$$pp \rightarrow \tilde{u}_L \tilde{\chi}_1^0 \rightarrow u \tilde{\chi}_1^0 \tilde{\chi}_1^0$$

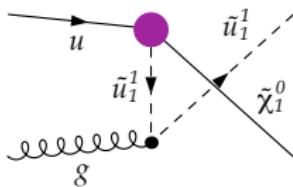
- Potential source of **Monojet signatures**
- Characteristic p_T spectrum
- Sensitivity to $q\bar{q}\tilde{\chi}_1^0$ couplings



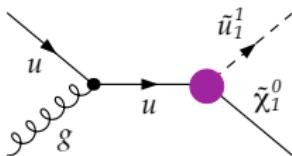
MadGOLEM taking off: $\text{pp} \rightarrow \tilde{u}\tilde{\chi}^0$ at NLO

$$\text{pp} \rightarrow \tilde{u}_L \tilde{\chi}_1^0 \rightarrow u \tilde{\chi}_1^0 \tilde{\chi}_1^0$$

- Potential source of **Monojet signatures**
- Characteristic p_T spectrum
- Sensitivity to $q\bar{q}\tilde{\chi}_1^0$ couplings

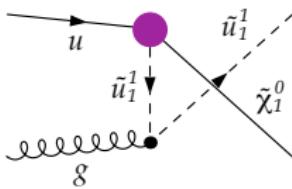


MadGOLEM taking off: $pp \rightarrow \tilde{u}\tilde{\chi}^0$ at NLO



$$pp \rightarrow \tilde{u}_L \tilde{\chi}_1^0 \rightarrow u \tilde{\chi}_1^0 \tilde{\chi}_1^0$$

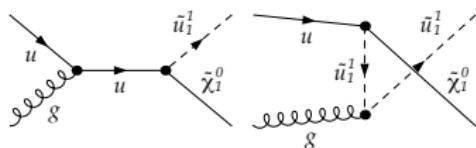
- Potential source of **Monojet signatures**
- Characteristic p_T spectrum
- Sensitivity to $q\bar{q}\tilde{\chi}_1^0$ couplings



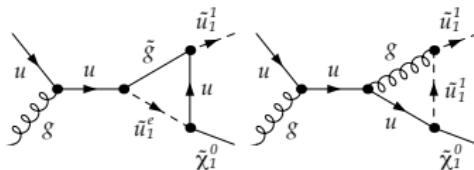
- ♠ Nature of the LSP (bino, wino-like ?)
- ♠ Constraints on DM detection
- ♠ SUSY relations between couplings
- ♠ Pattern of SUSY breaking \Rightarrow handle on the fundamental SUSY breaking mechanism !

For a recent analysis of the collider signatures (@ LO), cf. Allanach, Grab & Haber
[arXiv:1010.4261](https://arxiv.org/abs/1010.4261)

MadGOLEM taking off: $pp \rightarrow \tilde{u}\tilde{\chi}^0$ at NLO

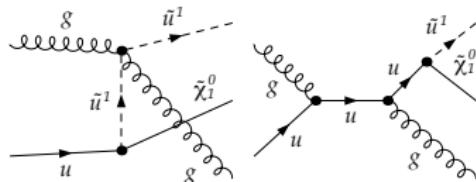
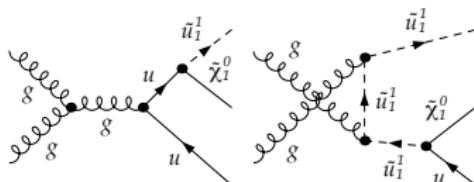


tree-level

virtual corrections $\mathcal{O}(\alpha_s^2)$ 

MSSM renormalization

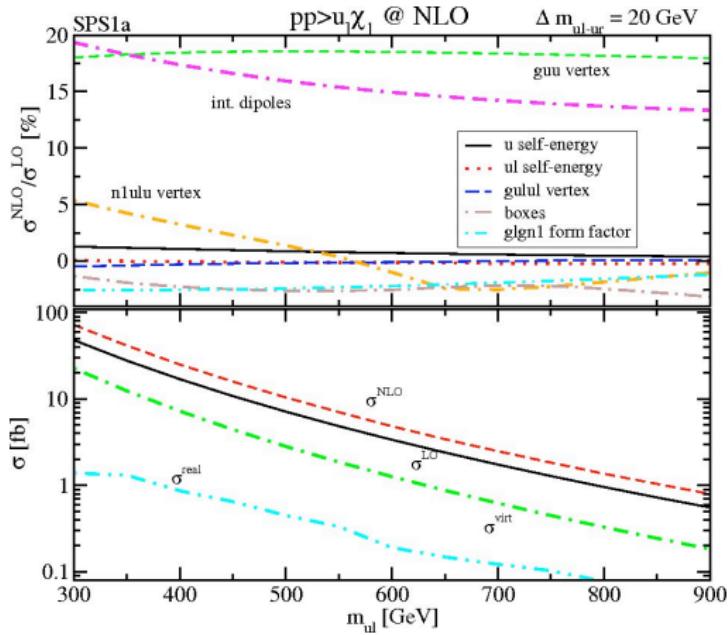
& SUSY-restoration

real corrections $\mathcal{O}(\alpha_s^2)$ OS divergent real corrections $\mathcal{O}(\alpha_s^2)$ 

ii , fi & if dipoles

OS subtraction

MadGOLEM taking off: $pp \rightarrow \tilde{u}\tilde{\chi}^0$ at NLO



♣ For details cf. Dorival Gonçalves Netto's talk at [Pheno'11](#) & Gonçalves Netto, DLV, Mawatari, Plehn, Wigmore, to appear.

MadGOLEM taking off: $pp \rightarrow \tilde{u}\tilde{\chi}^0$ at NLO

	σ^{LO} [fb]	σ^{NLO} [fb]	K	$M_{\tilde{u}_L}$ [GeV]	$M_{\tilde{u}_R}$ [GeV]	$M_{\tilde{\chi}_1}$ [GeV]
SPS1a	160.339	216.647	1.351	561.119	549.259	181.696
SPS1b	22.178	29.743	1.341	871.658	850.474	161.764
SPS2	1.081	1.440	1.332	1554.313	1553.824	237.406
SPS3	24.475	32.868	1.343	853.684	831.809	160.546
SPS4	41.920	55.305	1.319	760.030	747.952	227.720
SPS5	74.398	100.282	1.348	674.953	657.290	231.186
SPS6	56.474	76.636	1.357	669.564	660.214	224.299
SPS7	18.063	24.261	1.343	895.986	874.580	268.970
SPS8	7.371	9.733	1.320	1113.372	1077.385	138.947
SPS9	5.246	6.930	1.321	1276.308	1281.497	187.173

♠ For details cf. Dorival Gonçalves Netto's talk at Pheno'11 & Gonçalves Netto, DLV, Mawatari, Plehn, Wigmore, to appear.

Outline

- 1 Motivating MadGOLEM
- 2 MadGOLEM from inside: architecture of the code
- 3 MadGOLEM from the very inside: finer details
 - The MadGOLEM one-loop Calculator
 - Handling the divergences
- 4 MadGOLEM from outside: running MadGOLEM
- 5 MadGOLEM underway: performance and cross-checks
- 6 Beyond cross-checks: New Physics studies with MadGOLEM
- 7 Conclusions & Outlook

Conclusions & Outlook

What MadGOLEM provides

- Automates the calculation of NLO QCD corrections for a generic $2 \rightarrow 2$ process within the MadGraph/Golem framework
- Allows to obtain the NLO cross-sections and parton-level events in 3 steps :
 - Defining the process & generating the amplitude: `./newprocess_nlo`
 - Processing the amplitude `perl run_golem.pl` (helicity & color structures, loop-integral reduction, inclusion of CTs)
 - Getting the numbers: `./bin/generate_events_nlo 2 2 myprocess`
- Able to cope with generic BSM scenarios – being the MSSM our testing ground: Generalized & automatized PROSPINO
- Highly modular structure, handful of debugging options, multi-step analytical output \Rightarrow much more than a black box !

Conclusions & Outlook

What MadGOLEM provides

- Automates the calculation of NLO QCD corrections for a generic $2 \rightarrow 2$ process within the MadGraph/Golem framework
- Allows to obtain the NLO cross-sections and parton-level events in 3 steps :
 - Defining the process & generating the amplitude: `./newprocess_nlo`
 - Processing the amplitude `perl run_golem.pl` (helicity & color structures, loop-integral reduction, inclusion of CTs)
 - Getting the numbers: `./bin/generate_events_nlo 2 2 myprocess`
- Able to cope with generic BSM scenarios – being the MSSM our testing ground: Generalized & automatized PROSPINO
- Highly modular structure, handful of debugging options, multi-step analytical output \Rightarrow much more than a black box !

Conclusions & Outlook

What MadGOLEM provides

- Automates the calculation of NLO QCD corrections for a generic $2 \rightarrow 2$ process within the MadGraph/Golem framework
- Allows to obtain the NLO cross-sections and parton-level events in 3 steps :
 - Defining the process & generating the amplitude: `./newprocess_nlo`
 - Processing the amplitude `perl run_golem.pl` (helicity & color structures, loop-integral reduction, inclusion of CTs)
 - Getting the numbers: `./bin/generate_events_nlo 2 2 myprocess`
- Able to cope with generic BSM scenarios – being the MSSM our testing ground: Generalized & automatized PROSPINO
- Highly modular structure, handful of debugging options, multi-step analytical output \Rightarrow much more than a black box !

Conclusions & Outlook

What MadGOLEM provides

- Automates the calculation of NLO QCD corrections for a generic $2 \rightarrow 2$ process within the MadGraph/Golem framework
- Allows to obtain the NLO cross-sections and parton-level events in 3 steps :
 - Defining the process & generating the amplitude: `./newprocess_nlo`
 - Processing the amplitude `perl run_golem.pl` (helicity & color structures, loop-integral reduction, inclusion of CTs)
 - Getting the numbers: `./bin/generate_events_nlo 2 2 myprocess`
- Able to cope with generic BSM scenarios – being the MSSM our testing ground: Generalized & automatized PROSPINO
- Highly modular structure, handful of debugging options, multi-step analytical output \Rightarrow much more than a black box !

Conclusions & Outlook

MadGOLEM: status

- Operative for the SM/MSSM: SUSY dipoles & counterterms, OS subtraction, consistent handle on Majorana fermions ...
- Dedicated analysis of the performance of the dipole & OS subtraction terms \Rightarrow numerical stability and convergence in the presence of IR/OS divergences.
- Thoroughly checked for SM/MSSM processes – analytically & numerically
- Cross-check with independent codes underway (e.g. Prospino)
- Already able to tackle original calculations – cf. $\tilde{q}\tilde{\chi}^0$ to appear (talk by Dorival Gonçalves Netto @ Pheno11)

Conclusions & Outlook

Current limitations

- Limited to $2 \rightarrow 2$ processes
- Several types of processes yet to be tested
- Some degree of model-dependence: counterterms others than SM/MSSM are to be introduced by the user

Conclusions & Outlook

Current limitations

- Limited to $2 \rightarrow 2$ processes
- Several types of processes yet to be tested
- Some degree of model-dependence: counterterms others than SM/MSSM are to be introduced by the user

MadGOLEM: Current and near-future directions

- EXTEND & GENERALIZE : to make the code more flexible and adaptable to a completely generic BSM setup
- CLEAN-UP the code & REFINE the user interfaces: aiming at a truly user-friendly, handy layout – and a transparent internal structure.
- CROSS-CHECK : new processes against the available NLO calculators – e.g. all SUSY production channels in Prospino, H^\pm with MC@NLO ...
- MIGRATE : our loop calculator to the current GOLEM95 framework

Conclusions & Outlook

Current limitations

- Limited to $2 \rightarrow 2$ processes
- Several types of processes yet to be tested
- Some degree of model-dependence: counterterms others than SM/MSSM are to be introduced by the user

MadGOLEM: Current and near-future directions

- EXTEND & GENERALIZE : to make the code more flexible and adaptable to a completely generic BSM setup
- CLEAN-UP the code & REFINE the user interfaces: aiming at a truly user-friendly, handy layout – and a transparent internal structure.
- CROSS-CHECK : new processes against the available NLO calculators – e.g. all SUSY production channels in Prospino, H^\pm with MC@NLO ...
- MIGRATE : our loop calculator to the current GOLEM95 framework

Conclusions & Outlook

Current limitations

- Limited to $2 \rightarrow 2$ processes
- Several types of processes yet to be tested
- Some degree of model-dependence: counterterms others than SM/MSSM are to be introduced by the user

MadGOLEM: Current and near-future directions

- EXTEND & GENERALIZE : to make the code more flexible and adaptable to a completely generic BSM setup
- CLEAN-UP the code & REFINE the user interfaces: aiming at a truly user-friendly, handy layout – and a transparent internal structure.
- CROSS-CHECK : new processes against the available NLO calculators – e.g. all SUSY production channels in Prospino, H^\pm with MC@NLO ...
- MIGRATE : our loop calculator to the current GOLEM95 framework

Conclusions & Outlook

Current limitations

- Limited to $2 \rightarrow 2$ processes
- Several types of processes yet to be tested
- Some degree of model-dependence: counterterms others than SM/MSSM are to be introduced by the user

MadGOLEM: Current and near-future directions

- EXTEND & GENERALIZE : to make the code more flexible and adaptable to a completely generic BSM setup
- CLEAN-UP the code & REFINE the user interfaces: aiming at a truly user-friendly, handy layout – and a transparent internal structure.
- CROSS-CHECK : new processes against the available NLO calculators – e.g. all SUSY production channels in Prospino, H^\pm with MC@NLO ...
- MIGRATE : our loop calculator to the current GOLEM95 framework

Conclusions & Outlook

Messages to take home

- The distance between BSM models and predicted event rates @ NLO is being shortened to just typing three commands !! – (and hopefully not forgetting what these three commands encompass . . .)
- The code is still under development, though it already enables to perform original research
- First public release of the code in *O(few months)* ⇒ STAY TUNED !!

Conclusions & Outlook

Messages to take home

- The distance between BSM models and predicted event rates @ NLO is being shortened to just typing three commands !! – (and hopefully not forgetting what these three commands encompass . . .)
- The code is still under development, though it already enables to perform original research
- First public release of the code in $\mathcal{O}(\text{few months}) \Rightarrow$ STAY TUNED !!

Conclusions & Outlook

Messages to take home

- The distance between BSM models and predicted event rates @ NLO is being shortened to just typing three commands !! – (and hopefully not forgetting what these three commands encompass . . .)
- The code is still under development, though it already enables to perform original research
- First public release of the code in *O(few months)* ⇒ STAY TUNED !!

Conclusions & Outlook

MadGOLEM contributes to our common effort of bringing together



Conclusions & Outlook

MadGOLEM contributes to our common effort of bringing together



Theory

Experiment

Conclusions & Outlook

MadGOLEM contributes to our common effort of bringing together



Theory

THANKS A LOT !!

Experiment