New Developments in FEYNRULES

A. Alloul

FEYNRULES: AA, N.D. Christensen, C. Degrande, C. Duhr, B. Fuks

IPHC - Université de Strasbourg

May 16-21, 2013

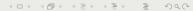
《曰》 《圖》 《注》 《注》 [] []



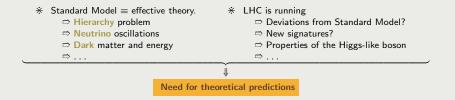
1 Basic features of FEYNRULES

2 New developments in version 1.8





Need for automatization



- * Develop **new** models
 - ⇔ Write Lagrangian
 - $\, rac > \,$ Compute several quantities (decay widths, mass and mixing matrices \ldots)
- * Implement in Monte Carlo (MC) tools
 - ➡ Many available tools (CALCHEP, MADGRAPH, PYTHIA, SHERPA...)
 - Every tool has its dedicated file format
 - Every tool has its advantages / disadvantages

Starting from your model, FEYNRULES provides all the necessary ingredients to generate the model file for you beloved MC tool(s)

How it works

* Mathematica package

- Christensen, Duhr (CPC'09), AA, Christensen, Degrande, Duhr, Fuks (prep)
- Version 1.6 and 1.8 beta downloadable from: feynrules.irmp.ucl.ac.be

* Derives Feynman rules from a Lagrangian.

- * Exports them to MC tools through interfaces.
 - ⇒ Available interfaces to
 - CALCHEP FEYNARTS MADGRAPH SHERPA UFO WHIZARD

How it works

* Mathematica package

- ⇒ Christensen, Duhr (CPC'09), AA, Christensen, Degrande, Duhr, Fuks (prep)
- Version 1.6 and 1.8 beta downloadable from: feynrules.irmp.ucl.ac.be

《曰》 《圖》 《臣》 《臣》

* Derives Feynman rules from a Lagrangian.

- Implement your model \Rightarrow
 - Gauge group and parameter declarations ⇒ Lorentz and gauge local invariance required.
 - Field definitions \Rightarrow scalar, fermion, vector, tensor, ghost, superfield, spin-3/2.
- Existing database (many models already available)
 - SM & simple extensions
 - Supersymmetry
 - Extra-dimensions
 - Strongly couopled and effective theories
- * Exports them to MC tools through interfaces.
 - Available interfaces to
 - CALCHEP FEYNARTS MADGRAPH SHERPA UFO WHIZARD

From your idea to the Monte Carlo events:

* Define the model

Gauge symmetries	Particles	Parameters
M\$GaugeGroups = { SU3C == { Abelian -> False.		
CouplingConstant -> gs, GaugeBoson -> G,		
StructureConstant -> f, Representations -> {T,Co SymmetricTensor -> dSUN		

Lagrangian

◆ロ → ◆回 → ◆ 三 → ◆ 三 → ● ◆ ● ◆ ●

From your idea to the Monte Carlo events:

* Define the model

Gauge symmetries	Particles	Parameters	Lagrangian
<pre>M\$ClasseSDescription == { V[1] == { ClassName -> G, SelfConjugate -> True, Indices -> {Index[Gluon]}, Mass -> 0, Width -> 0, ParticleName -> "g", PDG -> 21, PropagatorLabel -> "G", PropagatorType -> C, Plil == { ClassName -> q, ClassNembers -> {d,u,s,c,b,t}, FlavourIndex -> Flavour, SelfConjugate -> False, Indices -> {Index[Flavour], Int Mass -> {MQ, MD, MU, MS, MC, MI Width -> {WQ, 0, 0, 0, 0, 0, 0, W ParticleName -> ("d", "u", "s" AntiParticleName -> ("d", "u", "s" AntiParticleName -> {Id=", ""u", "s" AntiParticleName -> {Id=", "s" Self Conjugator -> {1, 2, 3, 4, 5, 6}, ClassName -> {1, 2, 3, 4, 5, 6}, Self Conjugator -> {1, 2, 3, 4, 5, 6}, Self Conj</pre>	<pre>lex[Colour]}, 3, MT}, "}, "c", "b", "t"}, ", "s", "c", "b", "t"}</pre>	Other ⇒ ⇒ ⇒ , "t~"},	Lagrangian classes are available: Weyl fermions W[] Scalars S[] Chiral superfields CSF[] Vector superfields VSF[]
PropagatorLabel -> {"d", "u", ' PropagatorType -> Straight,	's", "c", "b", "t"}	5	
PropagatorArrow -> Forward}}			그 > 《圊 > 《글 > 《글 > 글 · ·

From your idea to the Monte Carlo events:

* Define the model

Gauge symmetries	Particles	Parameters	Lagrangian
$g_s = \sqrt{4\pi\alpha_s}$	→	<pre>M\$Parameters == { aS == { ParameterType -> External, BlockName -> SMINPUTS, OrderBlock -> 3, Value -> 0.1184, InteractionOrder -> {QCD,2}, TeX -> Subscript[\[Alpha],s], Description -> "Strong coupling constant at the Z pole" }, gs == { ParameterType -> Internal, Value -> Sqrt[4*Pi*aS], InteractionOrder -> {QCD,1}, TeX -> Subscript[g,s], ParameterName -> G, Description -> "Strong coupling constant" }}</pre>	

From your idea to the Monte Carlo events:

* Define the model

Gauge symmetries

Particles

$$\begin{split} \mathcal{L}_{QCD} &= \\ -\frac{1}{4} G^{a}_{\mu\nu} G^{\mu\nu}_{a} + \\ \sum_{f} \left[\bar{q}_{f} (i \phi + g_{s} \phi^{a} T^{a} - m_{f}) q_{f} \right] \end{split}$$

LQCD := -1/4 FS[G,mu,nu,aa] FS[G,mu,nu,aa] +	Parameters	Lagrangian
I*qbar.Ga[mu].DC[q,mu] - MQ[f]*qbar[s,f,c].q[s,f,c];		

◆ロ → ◆回 → ◆ 三 → ◆ 三 → ● ◆ ● ◆ ●

From your idea to the Monte Carlo events:

- ✤ Define the model
- * Load the model

<<FeynRules`; LoadModel["modelfile.fr"];

() < </p>

How it works: the QCD Lagrangian

From your idea to the Monte Carlo events:

- * Define the model 🗸
- * Load the model \checkmark
- * Several commands available
 - ⇒ General commands
 - Hermitian conjugate: HC[q]
 - Flavor expansion: ExpandIndices[LQCD, FlavorExpand -> Generation]
 - Check hermiticity CheckHermiticity[LQCD]
 - Feynman Rules: FeynmanRules[LQCD]

• ...

(日) (同) (目) (日) (日) (日)

How it works: the QCD Lagrangian

From your idea to the Monte Carlo events:

- * Define the model 🗸
- ✤ Load the model
- * Several commands available
 - 🗢 General commands 🖌
 - ⇒ Supersymmetry dedicated commands:
 - Get components of a superfield SF2Components[superfield]
 - Kinetic terms of a vector superfield VSFKineticTerms[superfield]
 - ...

◆ロト ◆母 ト ◆臣 ト ◆臣 ト ◆ 国 ト ◆ の へ ()

How it works: the QCD Lagrangian

From your idea to the Monte Carlo events:

- * Define the model 🗸
- * Load the model 🗸
- * Several commands available
 - 🗢 General commands 🗸
 - Supersymmetry dedicated commands

* Interfaces

- \Rightarrow UFO: WriteUF0[LQCD]
- ⇒ FEYNARTS: WriteFeynArtsOutput[LQCD]
- ➡ CALCHEP: WriteCHOutput[LQCD]
- \Rightarrow WHIZARD: WriteWOOutput[LQCD]
- ⇒ ...

《口》 《圖》 《문》 《문》

 \equiv

How it works: the QCD Lagrangian

From your idea to the Monte Carlo events:

- ★ Define the model
- ★ Load the model
- Several commands available ×
 - ⇒ General commands
 - Supersymmetry dedicated commands
- The UNIVERSAL FEYNBULES OUTPUT interface ×
 - ⇒ A python module to be linked to any code
 - ⇒ All model information is included
 - ⇒ No restriction on the vertices (e.g., Lorentz and color structure)
 ⇒ Used by MADGRAPH 5, MADANALYSIS 5, GOSAM and HERWIG++

(日) (同) (目) (日) (日) (日)

How it works: the QCD Lagrangian

From your idea to the Monte Carlo events:

- ✤ Define the model
- ✤ Load the model
- * Several commands available
 - ⇒ General commands
 - Supersymmetry dedicated commands
- * Interfaces 🗸

For more details see:

- 🐼 FeynRules Feynman rules made easy, Christensen, Duhr, CPC'09
- Main A superspace module for the FeynRules package, Duhr, Fuks, CPC'11
- A Comprehensive approach to new physics simulations, Christensen, de Aquino, Degrande, Duhr, Fuks, Herquet, Maltoni, Schumann, E.P.J C71 '11
 - 📪 UFO The Universal FeynRules Output, Degrande, Duhr, Fuks, Grellscheid, Mattelaer, Reiter, CPC '12
- Introducing an interface between WHIZARD and FeynRules, Christensen, Duhr, Fuks, Reuter, Speckner, E.P.J. C72 '12
- Beyond the Minimal Supersymmetric Standard Model: from theory to phenomenology, Fuks, Int.J.Mod.Phys. A27 '12







2 New developments in version 1.8





Spin-3/2 Rarita-Schwinger fields

- * Spin-3/2 fields
 - ⇒ Described first by Rarita & Schwinger in 1941.
 - ⇒ Appears in some BSM theories (Gravitino in SUSY, e.g)
- * In FEYNRULES { Lagrangian's dimensionality unrestricted. Complete superspace module. } Everything is there !
- * Two new classes for field decalaration
 - \Rightarrow For two-component fermions RW
 - ⇒ For four-component fermions R
- * Goldstino / Gravitino
 - \Rightarrow Spontaneous supersymmetry breaking \Rightarrow massless fermion.
 - \Rightarrow Its interactions given by supercurrent $\epsilon \cdot J^{\mu} + \bar{\epsilon} \cdot \bar{J}^{\mu} = \frac{\partial \mathcal{L}}{\partial (\partial_{\mu} X)} \delta_{\epsilon} X K^{\mu}$
 - ➡ FEYNRULES dedicated command Supercurrent[lc,lv,lw,sp,mu]
 - 1c: Chiral Lagrangian, 1v: Vector Lagrangian, 1w: Superpotential, sp,mu: Spin & Lorentz indices.

* UFO and CALCHEP interfaces adapted

Simulating spin-3/2 particle production at colliders, Christensen, de Aquino, Deutschmann, Duhr, Fuks, Garcia-Cely, Mattelaer, Mawatari, Oexl, Takaesu, In preparation

Beta version available

Decays package

- * Tree-Level two-body decay widths
 - ⇒ Are now automatically computed for any model

```
verts = FeynmanRules[LQCD];
vertsexp = FlavorExpansion[verts];
results = ComputeWidths[vertsexp];
```

- ⇒ Everything is analytical
- * Phase-space closed channels included
 - \Rightarrow No information on the (numerical values of the) spectrum at this level
 - Benchmark scenario independent
- * Information passed to the UFO WriteUFO[LQCD, AddDecays -> True]
 - ⇒ Flexible: closed formulas (NLO, n-body, BSM) can be included.
- ✤ MadGraph 5¹
 - ⇒ checks numerically open channels.

 ${\tt IS}$ Computing decay rates for new physics theories with ${\rm FeynRules}$ and ${\rm MadGraph}$,

Duhr, Fuks, Mattelaer, Oeztürk,

In preparation

Beta version available

《日》 《圖》 《王》 《王》

 $^{1}\mathrm{CALCHEP}$ calculates also automatically widths

Spectrum generator

Problem

- * After generating the model file for your MC-generator, you need
 - \Rightarrow To calculate the mass matrices: Lengthy and error-prone

イロト イヨト イヨト イヨト

Ξ

Problem

- * After generating the model file for your MC-generator, you need

 - $\stackrel{risk}{\Rightarrow}$ To calculate the mass matrices: Lengthy and error-prone $\stackrel{risk}{\Rightarrow}$ Diagonalize them: No analytical solution for matrices bigger than 4 × 4

《口》 《圖》 《문》 《문》

Ξ

Spectrum generator

Problem

- * After generating the model file for your MC-generator, you need
 - ➡ To calculate the mass matrices: Lengthy and error-prone
 - \Rightarrow Diagonalize them: No analytical solution for matrices bigger than 4 \times 4
 - Dupdate the model file: Lengthy (several scenarios) and error-prone

《口》 《圖》 《臣》 《臣》

Spectrum generator

Problem

- * After generating the model file for your MC-generator, you need
 - To calculate the mass matrices: Lengthy and error-prone
 - Diagonalize them: No analytical solution for matrices bigger than 4 × 4
 - Update the model file: Lengthy (several scenarios) and error-prone

Solution

- ✤ FeynRules will
 - ⇒ extract automatically analytical expressions for mass matrices
 - ⇒ generate automatically a numerical code for the diagonalization
- * The numerical code:
 - ⇒ produces a SLHA-like output

イロト イヨト イヨト イヨト ヨー わらぐ

Mass matrices generation with $\operatorname{FeynRules}$

Model file simplified

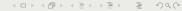
		M\$MixingsDescription = {
		$Mix["1u"] == \{MassBasis \rightarrow \{A, Z\},\$
×	Field mixing declaration	<pre>GaugeBasis -> {B, Wi[3]},</pre>
		MixingMatrix -> UG,
		BlockName -> WEAKMIX}}

⇒ Various options for different fields

Mass matrices generation with $\operatorname{FeynRules}$

Model file simplified

- * Field mixing declaration M\$MixingDescription == {Mix["Id"] == {options }}
- * Vacuum expectation values: M\$vevs == {{phi[2],vev}}



Mass matrices generation with $\operatorname{FeynRules}$

Model file simplified

- * Field mixing declaration M\$MixingDescription == {Mix["Id"] == {options }}
- * Vacuum expectation values: M\$vevs
- * Mass matrices, mixing matrices declared automatically as complex

() < </p>

Mass matrices generation with $\operatorname{FeynRules}$

Model file simplified

- * Field mixing declaration M\$MixingDescription == {Mix["Id"] == {options }}
- * Vacuum expectation values: M\$vevs
- * Mass matrices, mixing matrices declared automatically as complex

Commands available in $\operatorname{FeynRules}$

- * Compute mass matrices: ComputeMassMatrix[lagr]
 - \Rightarrow Possible to only compute a set of mixings with option Mix -> {"Id"}
- * Access specific informations with the Id

➡ MassMatrix["Id"], GaugeBasis["Id"], MassBasis["Id"]

- * Summary of all results
 - MixingSummary[]

() < </p>

Mass matrices generation with FEYNRULES

Model file simplified

- * Field mixing declaration M\$MixingDescription == {Mix["Id"] == {options }}
- * Vacuum expectation values: M\$vevs
- * Mass matrices, mixing matrices declared automatically as complex

Commands available in $\operatorname{FeynRules}$

- * Compute mass matrices: ComputeMassMatrix[lagr]
 - ⇒ Possible to only compute a set of mixings with option Mix -> {"Id"}
- * Access specific informations with the Id

➡ MassMatrix["Id"], GaugeBasis["Id"], MassBasis["Id"]

* Summary of all results



Towards a numerical code

- * Write the C++ package: WriteASperGe[Lagrangian,Mix -> {"Id"}]
- * Run the package and import results: RunASperGe[]

ASperGe: Automated Spectrum Generation

- * ASperGe is a C++ package
 - ➡ Generated automatically by FEYNRULES directory ModelName_MD
 - ⇒ Contains all the necessary routines to
 - Define the mass matrices
 - Diagonalize them
 - Generate an SLHA-compliant output file
 - ⇒ **Standalone** package
 - Only need GSL and a C++ compiler

```
ACAT2013% cd MyTestModel
ACAT2013% make
ACAT2013% ./ASperGe input.dat output.dat
Block MASS
# pdg code mass particle
```

```
22 0.000000e+00 # A
```

```
23 9.180401e+01 # Z
```

Automated mass spectrum generation for new physics,

AA, D'Hondt, De Causmaecker, Fuks, Rausch de Traubenberg, E.P.J C73 '13

NLO computations

Motivations

- * Leading Order (LO) calculations good for prospections
 - \Rightarrow Need for more precise theoretical predictions.
 - ▷ Next to Leading Order (NLO) is the next step
 - $NLO = LO + R_2$ vertices + UV counterterms + real emissions + virtual emissions.

Can be evaluated once for all from a Lagrangian

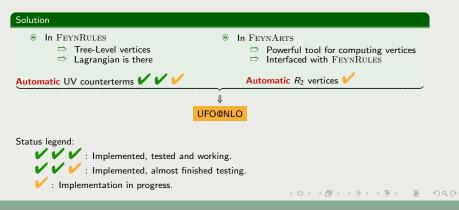
Solution	
 ★ In FEYNRULES ⇒ Tree-Level vertices ⇒ Lagrangian is there 	 ★ In FEYNARTS ⇒ Powerful tool for computing vertices ⇒ Interfaced with FEYNRULES
Automatic UV counterterms	Automatic R ₂ vertices
	UFO@NLO

NLO computations

Motivations

- * Leading Order (LO) calculations good for prospections
 - ⇒ Need for more precise theoretical predictions.
 - Next to Leading Order (NLO) is the next step
 - $NLO = LO + R_2$ vertices + UV counterterms + real emissions + virtual emissions.

Can be evaluated once for all from a Lagrangian



Speed improvements: Multicore

- * Multicore is fully supported
 - ▷ Need to set the boolean FR\$Parallel = True;
 - ▷ Set the number of cores to use FR\$KernelNumber
 - ⇒ Second master kernel free
 - ⇒ Speed improvement!

Model MSSM loaded.
Starting Feynman rule calculation.
Expanding the Lagrangian
Starting Feynman rule calculation.
Expanding the Lagrangian
Expansion of indices distributed over 8 kernels.
Collecting the different structures that enter the vertex.
417 possible non-zero vertices have been found -> starting the computation: 417 / 417.
417 vertices obtained.
[81.117552, Null]

() < </p>

 \longrightarrow On 1 core 351 s \Rightarrow gain of factor 4!

Conclusion

- * FEYNRULES is a MATHEMATICA package
- ★ In the last stable version of FEYNRULES:
 - ⇒ Model implementation minimal
 - Superspace package
 - ⇒ Many interfaces to Monte Carlo tools
 - ➡ UNIVERSAL FEYNRULES OUTPUT
- * The next version's features
 - ⇒ Spectrum generator (with ASPERGE). ✔
 - \Rightarrow Decay package for $1 \rightarrow 2$ processes.
 - ⇒ Rarita-Schwinger spin-3/2 field implemented. 🖌
- * NLO computations and automatic renormalization of the Lagrangian. \checkmark next version

Thank you for your attention

《口》 《圖》 《臣》 《臣》

 \equiv

◆ロト ◆母 ト ◆臣 ト ◆臣 ト ◆ 国 ト ◆ の へ ()

NLO in FEYNRULES: Technical details

* Assumptions

- \Rightarrow Maximum dimension of the operators = 4
- $\begin{array}{ll} \rightleftharpoons & \text{Feynman Gauge} \\ \rightleftharpoons & \{\gamma_5, \gamma_\mu\} = 0 \end{array}$
- 't Hooft-Veltman scheme
- ⇒ On-shell scheme for the masses and wave functions of massive states
- \Rightarrow MS otherwise
- * Outlook
 - ⇒ Effective theories
 - ⇒ Other gauge (not Feynman)
 - ⇒ ... any suggestion?