Getting Started: Mixings

Mixings

- So far our model has the following form:

$$\mathcal{L} = D_\mu \phi_i^\dagger D^\mu \phi_i - m^2 \phi_i^\dagger \phi_i + \lambda (\phi_i^\dagger \phi_i)^2$$

Mixings

- So far our model has the following form:

$$\mathcal{L} = D_\mu \phi_i^\dagger D^\mu \phi_i - m^2 \phi_i^\dagger \phi_i + \lambda (\phi_i^\dagger \phi_i)^2$$

- In many BSM models the new fields are not mass eigenstates, but they mix, e.g.

$$\mathcal{L} = D_\mu \phi_i^\dagger D^\mu \phi_i - m^2 \phi_i^\dagger \phi_i - m_{12}^2 (\phi_1^\dagger \phi_2 + \phi_2^\dagger \phi_1) + \lambda (\phi_i^\dagger \phi_i)^2$$

- The gauge and mass eigenstates are then related via some unitary rotation,

$$\begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix} = U \begin{pmatrix} \Phi_1 \\ \Phi_2 \end{pmatrix}$$

Mixings

- FeynRules offers the possibility to write the Lagrangian in terms of the gauge eigenstates, and let Mathematica perform the rotation.
- N.B.: Right now FeynRules **does not** diagoanlize the mass matrix for you! The diagonalization has to be performed by the user.
- For small mixing matrices, this can simply be done in Mathematica.
- For larger matrices, need to use some external numerical code.
  - ➡ See tomorrow's lecture.

Mixings

- The mixing matrix is declared as a parameter:

```
M$Parameter = {
  ...

   UU == {
          ComplexParameter -> True,
          Unitary -> True
          Indices  -> {Index[Scalar], Index[Scalar]},
          Value    -> { UU[1,1]  ->  ...,
                        UU[1,2]  ->  ...,
                         ...}
     }
   ...
};
```

Mixings

- The mass eigenstates are declared as normal particles

```
M$ClassesDescription = {
   ....
   S[11] == {
          ClassName       -> PP,
          ClassMembers -> {PP1,PP1},
          SelfConjugate   -> False,
          Indices            -> {Index[Scalar], Index[Gluon]},
          FlavorIndex      -> Scalar,
          Mass                -> {{MP1, ...},  {MP2, ...}}
     }
    ...
};
```

Mixings

- The gauge eigenstates are declared in a similar way

```
M$ClassesDescription = {
   S[1] == {
        ClassName         -> phi,
        ClassMembers   -> {phi1,phi2},
        SelfConjugate    -> False,
        Indices             -> {Index[Scalar], Index[Gluon]},
        FlavorIndex      -> Scalar,
        ~~Mass               -> {MS, 100}~~
        Unphysical         -> True,
        Definitions -> {phi[i_, a_] :> Module[{j}, UU[i,j] PP[j,a]]}
     }
};
```

Extending existing implementations

Extending the SM

- So far we have only considered our model standalone.
- For LHC phenomenology, one usually wants a BSM model that is an extension of the SM.
- FeynRules offers the possibility to start form the SM model, and to add/change/remove particles and operators.
- For this, it is enough to load our new model together with the SM implementation:

```
LoadModel[ “SM.fr“, “Phi_4_Gauged“ ];
```

- Note that the ‘parent model’ should always be loaded first in order to ensure that everything is set up correctly.

N.B.: In the SM implementation, the gluon and the QCD gauge group are already defined, so no need to redefine them.

Other available models

- The same procedure can be used to extend any other models.
- Many models can be downloaded from the FeynRules web page, and can serve as a start to implement new models (http://feynrules.irmp.ucl.ac.be/).

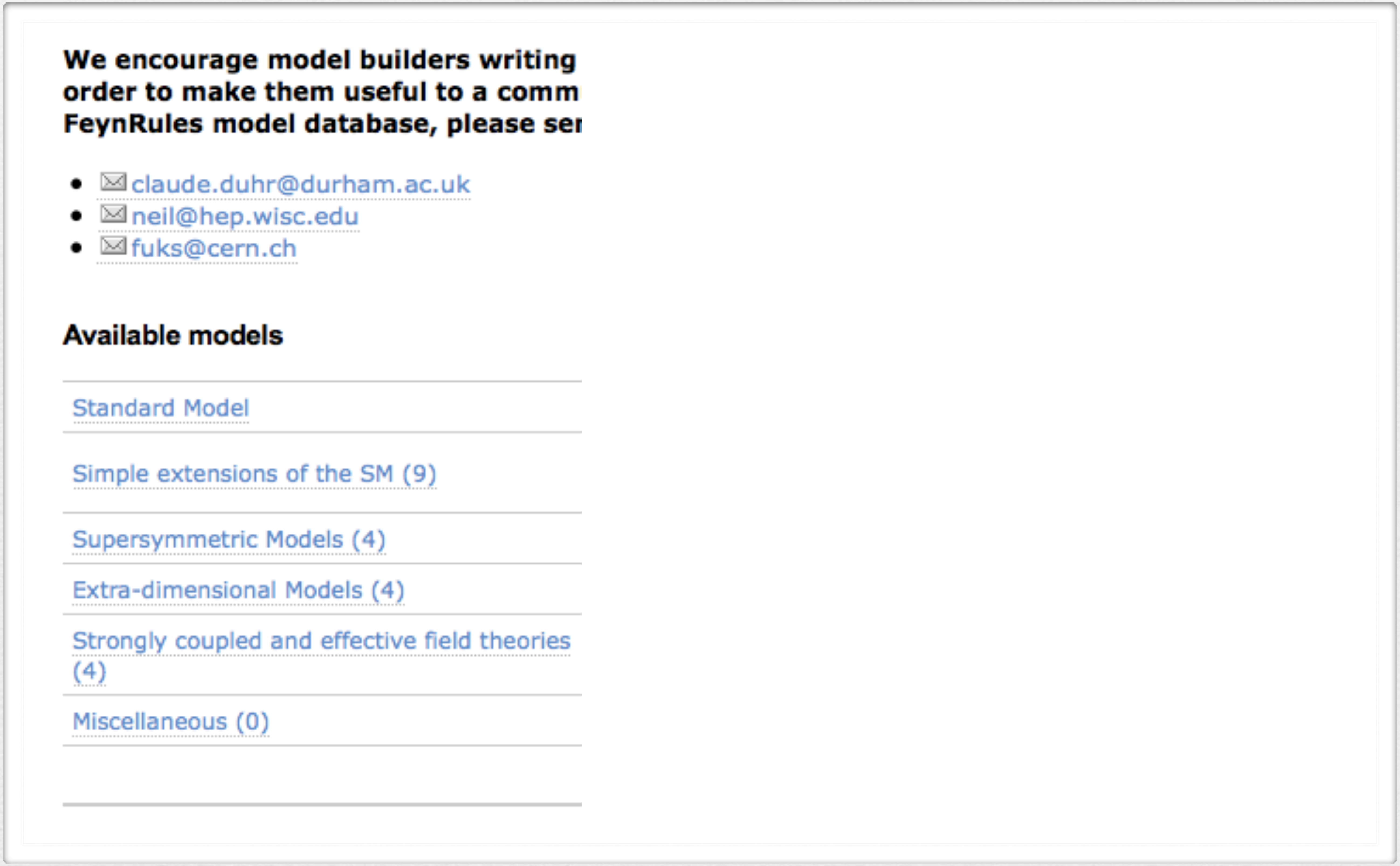
  ➡ SM (+ extensions: 4th generation, diquarks, See-saw...).

  ➡ MSSM, NMSSM, RPV-MSSM.

  ➡ Extra dimensions: UED, LED, Higgsless, HEIDI.

  ➡ Minimal walking Technicolor.

Model database



We encourage model builders writing order to make them useful to a comm FeynRules model database, please ser

- claude.duhr@durham.ac.uk
- neil@hep.wisc.edu
- fuks@cern.ch

Available models

Standard Model

Simple extensions of the SM (9)

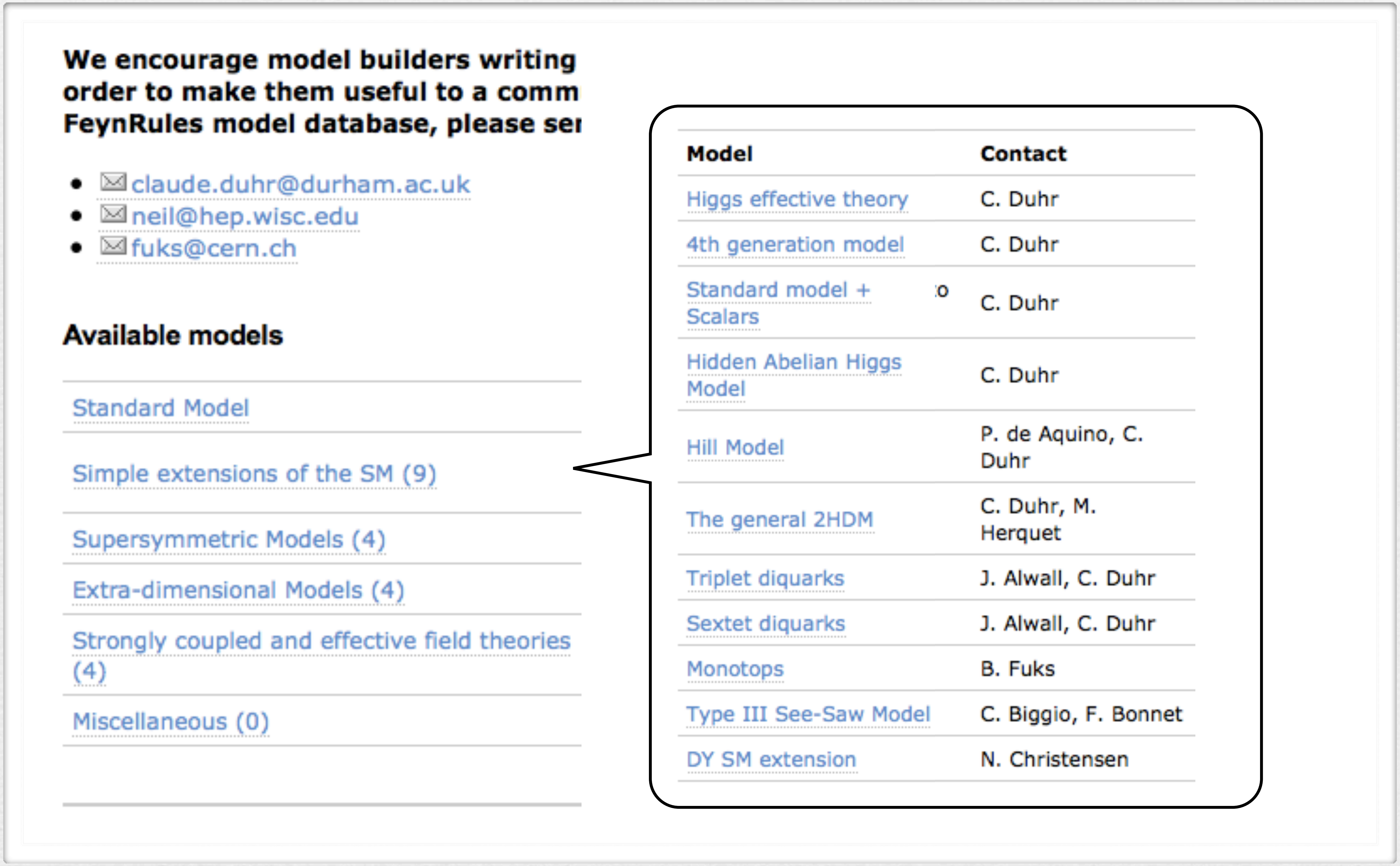
Supersymmetric Models (4)

Extra-dimensional Models (4)

Strongly coupled and effective field theories (4)

Miscellaneous (0)

Model database



We encourage model builders writing
order to make them useful to a comm
FeynRules model database, please se

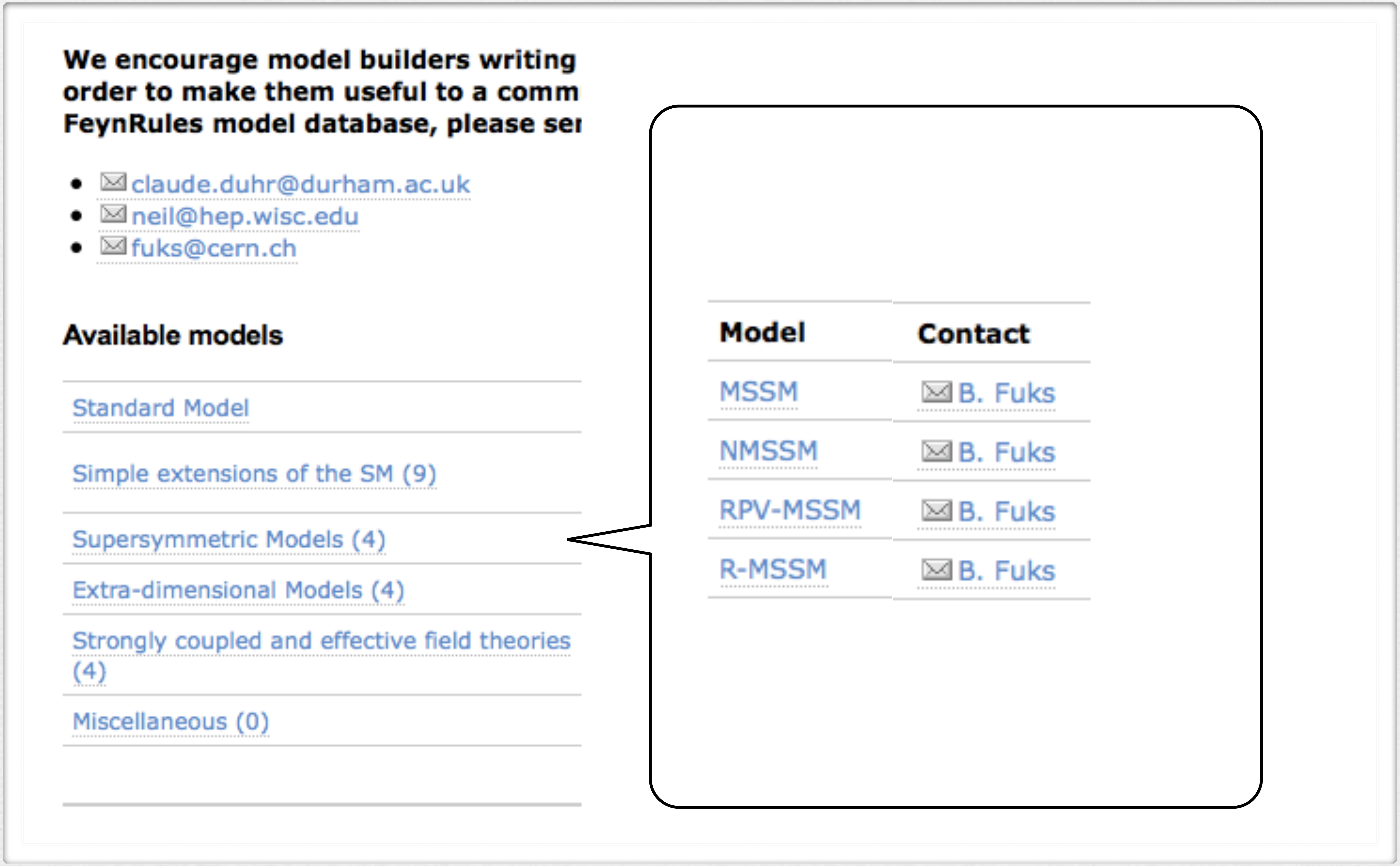
- claude.duhr@durham.ac.uk
- neil@hep.wisc.edu
- fuks@cern.ch

Available models

- Standard Model
- Simple extensions of the SM (9)
- Supersymmetric Models (4)
- Extra-dimensional Models (4)
- Strongly coupled and effective field theories (4)
- Miscellaneous (0)

Model	Contact
Higgs effective theory	C. Duhr
4th generation model	C. Duhr
Standard model + Scalars	C. Duhr
Hidden Abelian Higgs Model	C. Duhr
Hill Model	P. de Aquino, C. Duhr
The general 2HDM	C. Duhr, M. Herquet
Triplet diquarks	J. Alwall, C. Duhr
Sextet diquarks	J. Alwall, C. Duhr
Monotops	B. Fuks
Type III See-Saw Model	C. Biggio, F. Bonnet
DY SM extension	N. Christensen

Model database



We encourage model builders writing
order to make them useful to a comm
FeynRules model database, please se

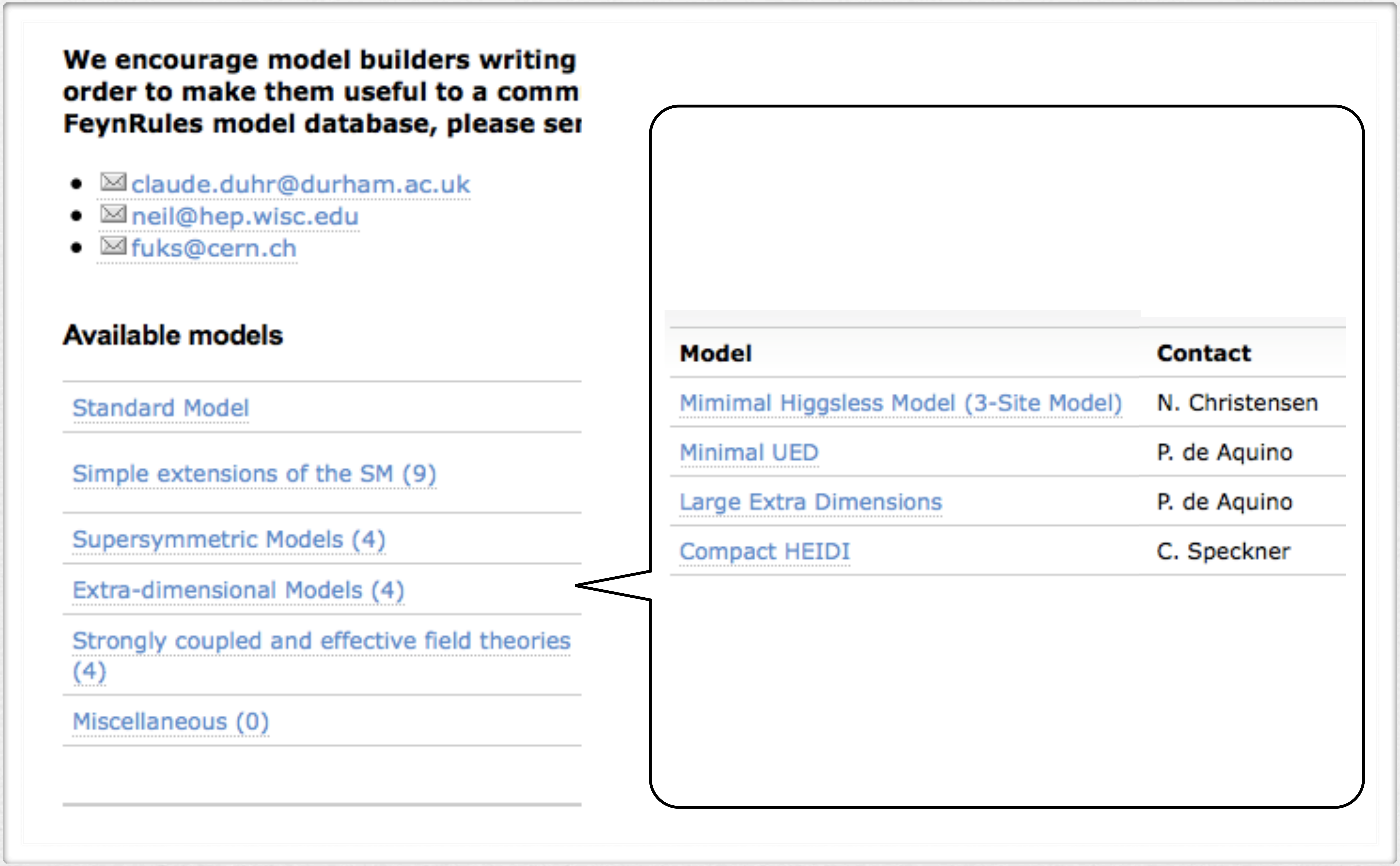
- ✉claude.duhr@durham.ac.uk
- ✉neil@hep.wisc.edu
- ✉fuks@cern.ch

Available models

Standard Model
Simple extensions of the SM (9)
Supersymmetric Models (4)
Extra-dimensional Models (4)
Strongly coupled and effective field theories (4)
Miscellaneous (0)

Model	Contact
MSSM	✉B. Fuks
NMSSM	✉B. Fuks
RPV-MSSM	✉B. Fuks
R-MSSM	✉B. Fuks

Model database



We encourage model builders writing order to make them useful to a comm FeynRules model database, please se

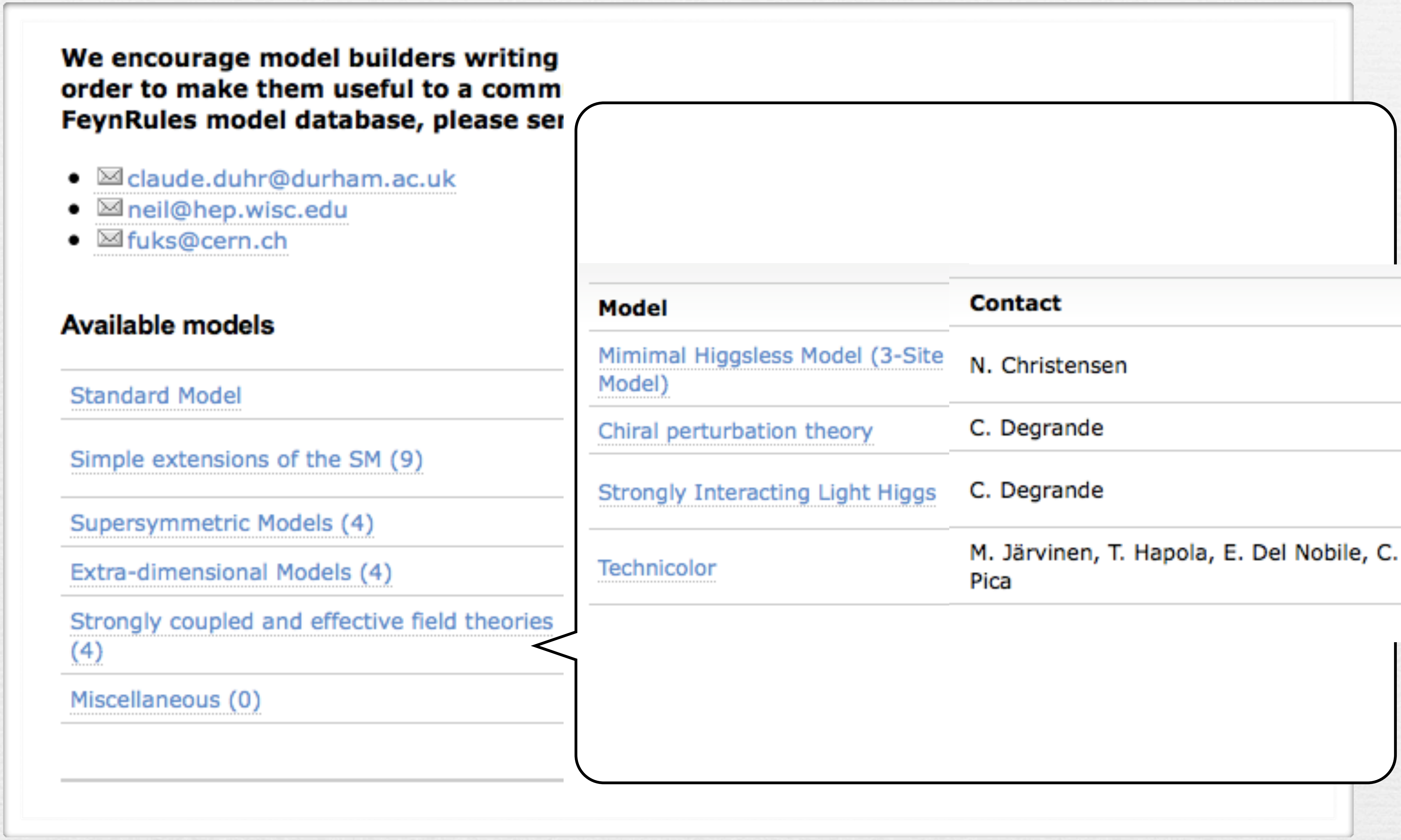
- claude.duhr@durham.ac.uk
- neil@hep.wisc.edu
- fuks@cern.ch

Available models

Standard Model
Simple extensions of the SM (9)
Supersymmetric Models (4)
Extra-dimensional Models (4)
Strongly coupled and effective field theories (4)
Miscellaneous (0)

Model	Contact
Mimimal Higgsless Model (3-Site Model)	N. Christensen
Minimal UED	P. de Aquino
Large Extra Dimensions	P. de Aquino
Compact HEIDI	C. Speckner

Model database



We encourage model builders writing
order to make them useful to a comm
FeynRules model database, please se

- claude.duhr@durham.ac.uk
- neil@hep.wisc.edu
- fuks@cern.ch

Available models

- Standard Model
- Simple extensions of the SM (9)
- Supersymmetric Models (4)
- Extra-dimensional Models (4)
- Strongly coupled and effective field theories (4)
- Miscellaneous (0)

Model	Contact
Mimimal Higgsless Model (3-Site Model)	N. Christensen
Chiral perturbation theory	C. Degrande
Strongly Interacting Light Higgs	C. Degrande
Technicolor	M. Järvinen, T. Hapola, E. Del Nobile, C. Pica

Complicated models

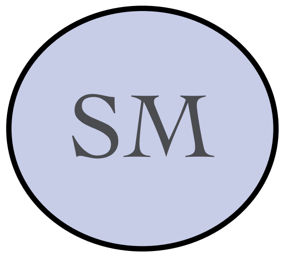
- So far we have only discussed how to implement very general models, and slight variations thereof by starting from a ‘parent model’.
- For many phenomenological applications this is an overkill, as one is very often only interested in a very restricted scenario that serves as a benchmark for phenomenological studies.
    - ➡ Example: We are only interested in models with massless leptons, and without flavor mixing.
- In pratise, these restricted models have the advantage that the number of vertices is much less, which can reduce considerably the run time of the MC codes.

Model Restrictions

- A *model restriction* is a model that is obtained from a bigger model by putting some of its parameters to zero (or 1, etc.).
- Example:

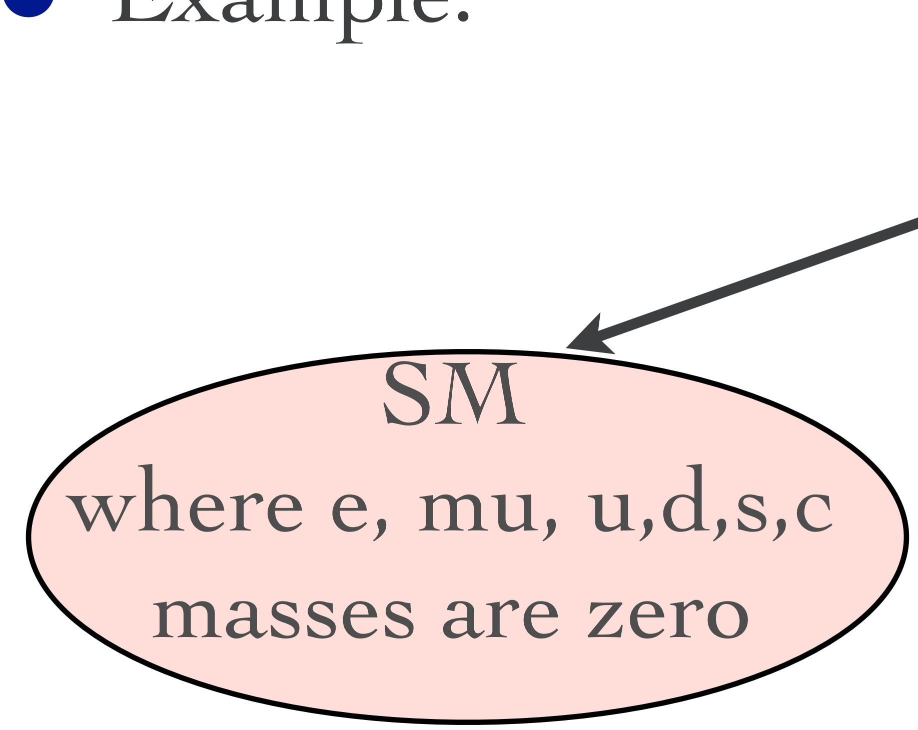
Model Restrictions

- A *model restriction* is a model that is obtained from a bigger model by putting some of its parameters to zero (or 1, etc.).
- Example:



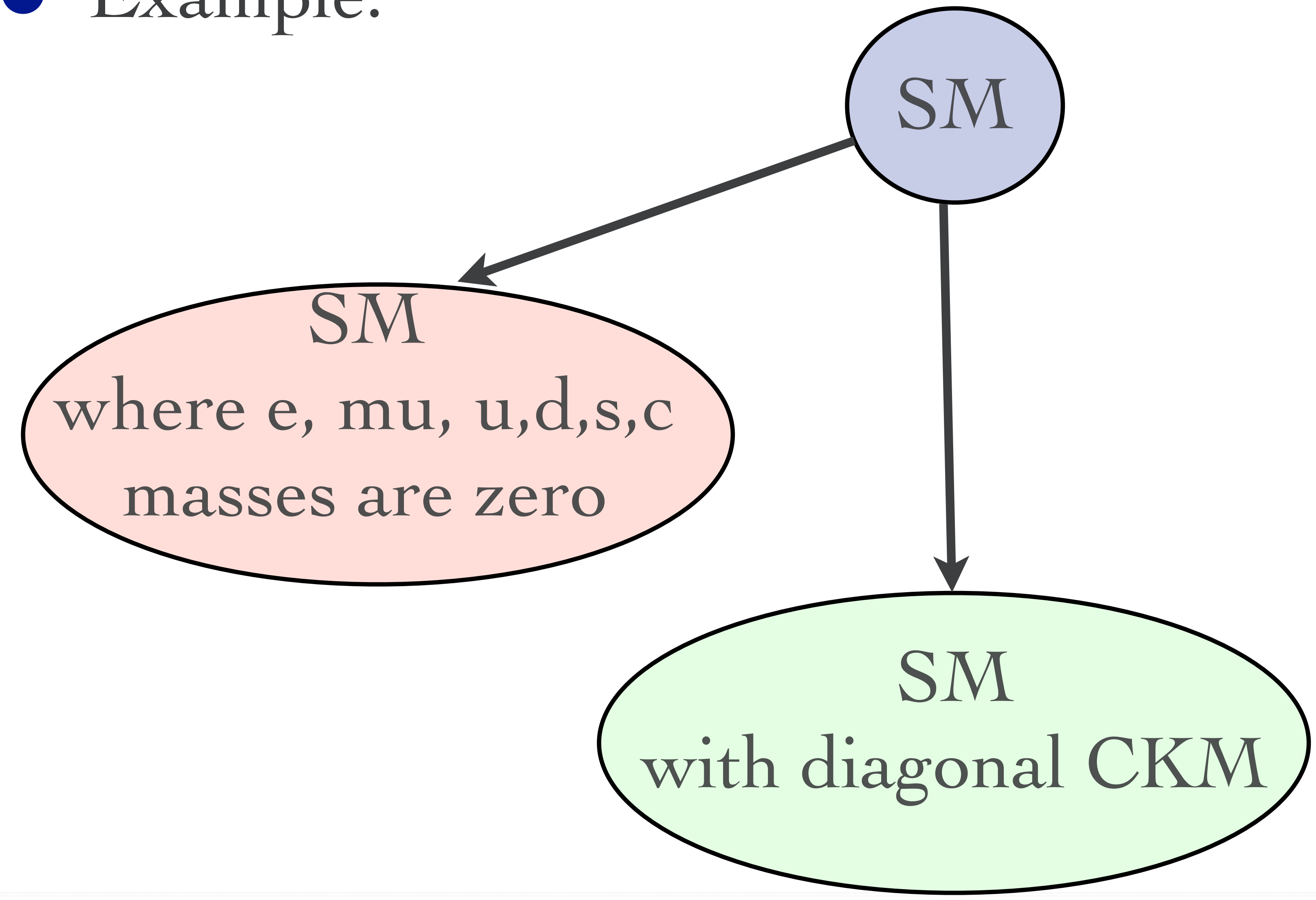
Model Restrictions

- A *model restriction* is a model that is obtained from a bigger model by putting some of its parameters to zero (or 1, etc.).
- Example:



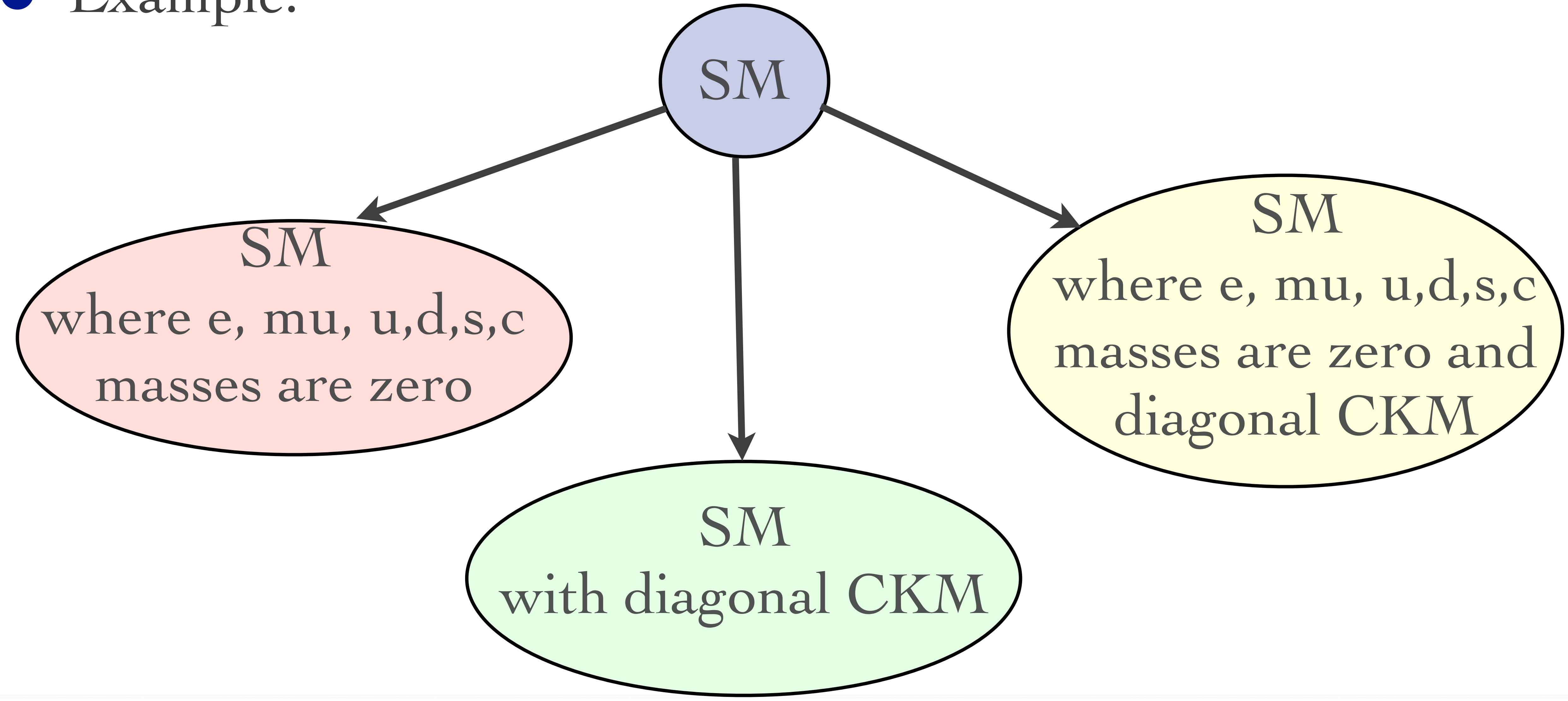
Model Restrictions

- A *model restriction* is a model that is obtained from a bigger model by putting some of its parameters to zero (or 1, etc.).
- Example:



Model Restrictions

- A *model restriction* is a model that is obtained from a bigger model by putting some of its parameters to zero (or 1, etc.).
- Example:

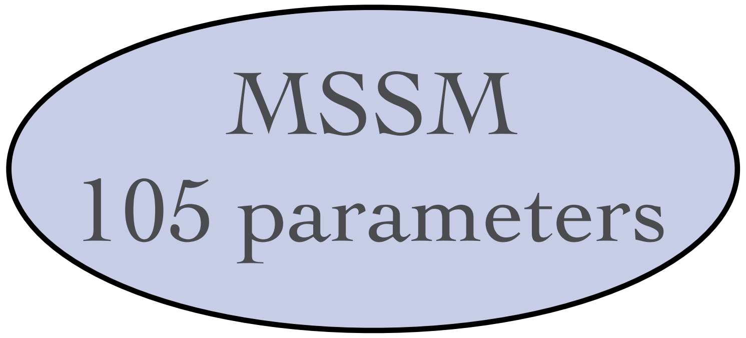

SM
SM where e, mu, u,d,s,c masses are zero
SM with diagonal CKM
SM where e, mu, u,d,s,c masses are zero and diagonal CKM

Model Restrictions

- A *model restriction* is a model that is obtained from a bigger model by putting some of its parameters to zero (or 1, etc.).
- Example:

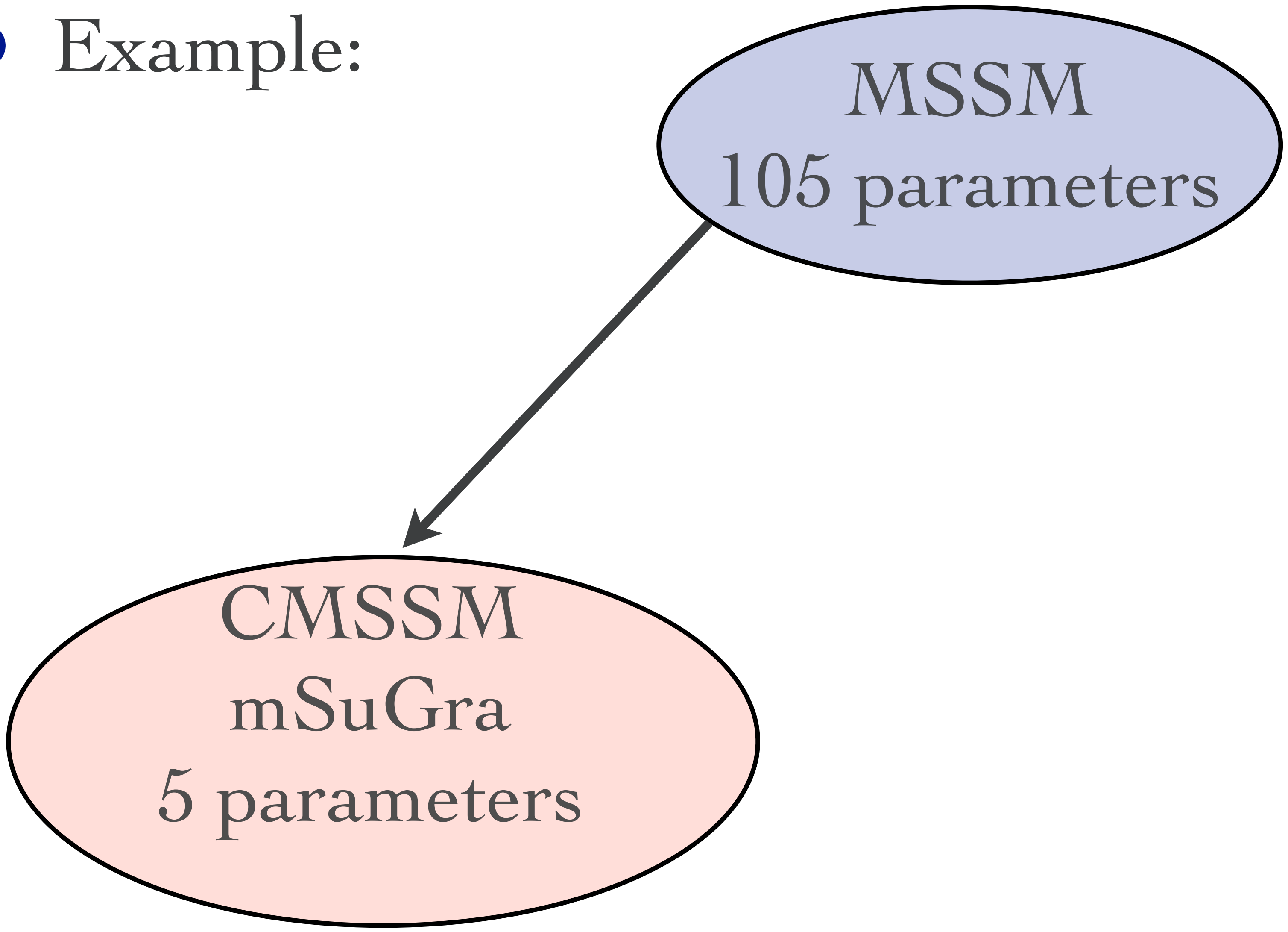
Model Restrictions

- A *model restriction* is a model that is obtained from a bigger model by putting some of its parameters to zero (or 1, etc.).
- Example:



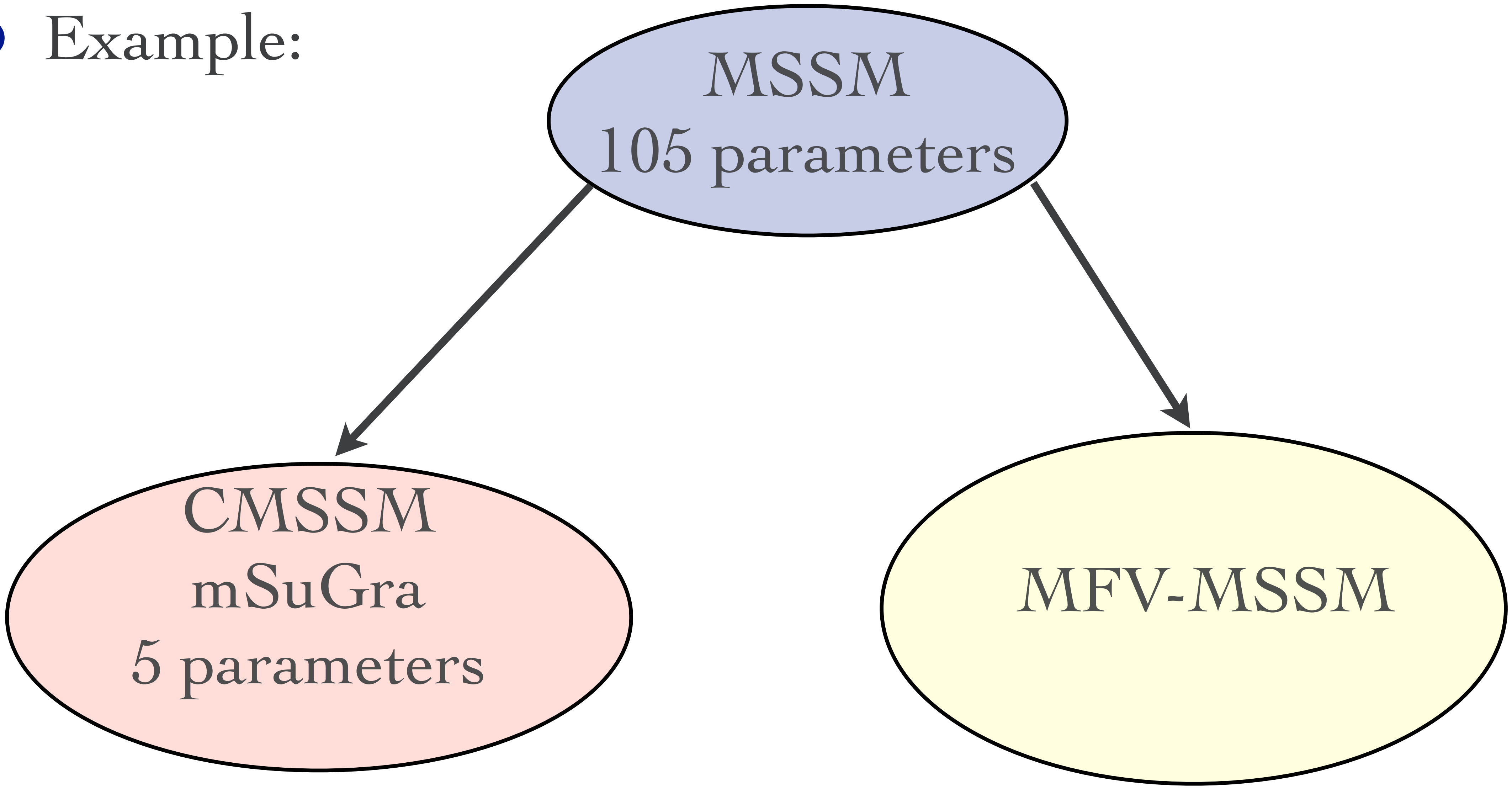
Model Restrictions

- A *model restriction* is a model that is obtained from a bigger model by putting some of its parameters to zero (or 1, etc.).
- Example:



Model Restrictions

- A *model restriction* is a model that is obtained from a bigger model by putting some of its parameters to zero (or 1, etc.).
- Example:



Model Restrictions

- In phenomenological applications one generally does not need the *full* model, but only a subset.

- Keeping the full model is ok, but it might make the MC unnecessarily slow.
  - ➡ Example: for generic CKM, lots of flavor-violating vertices, that lead to diagrams that are numerically subleading.

- We want a way to get rid of the 'undesired' vertices!

Model Restrictions

- Restriction files allow to achieve this by using simple Mathematica replacement rules.

```
M$Restrictions = {
        CKM[i_,i_] -> 1,
        CKM[i_?NumericQ, j_?NumericQ] :> 0 /; (i =!= j),
}
```

- If one or more restrictions are loaded after loading a model file, the corresponding replacement rules are applied at runtime when computing the vertices.

```
LoadRestriction[ “DiagonalCKM.rst“ ];
```

Towards LHC phenomenology: The FeynRules interfaces

The Interfaces

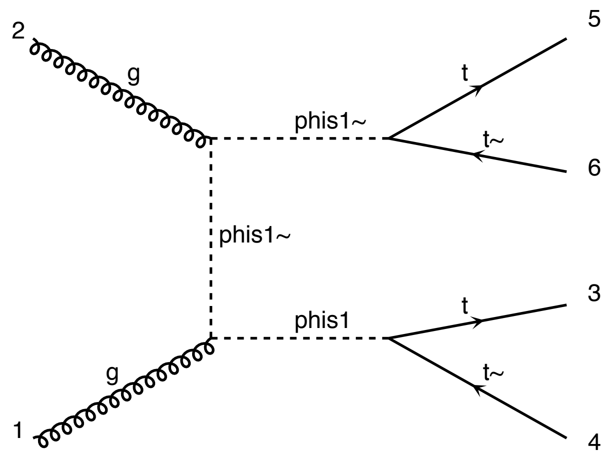
- So far we have only discussed how to implement a model into FeynRules and how to obtained the vertices.
- Next we want to do phenomenology!
- FeynRules contains interfaces to the following codes:
  - ➡ CalcHep / CompHep
  - ➡ FeynArts / FormCalc
  - ➡ MadGraph
  - ➡ Sherpa
  - ➡ Whizard / Omega
- Each interface produces a set of text files that can be read into the existing generators.

Running Interfaces

- The interfaces are called via the Mathematica commands

```
WriteCHOutput[ LSM, L ];              (* CalcHep *)
WriteFeynArtsOutput[ LSM, L ];        (* FeynArts/FormCalc *)
WriteMGOutput[ LSM, L ];              (*MadGraph 4 *)
WriteUFO[ LSM, L ];                   (* UFO / MadGraph 5 *)
WriteSHOutput[ LSM, L ];              (* Sherpa *)
WriteWOOutput[ LSM, L];               (* Whizard / Omega *)
```

- The files produced by FeynRules can then be processed by the matrix element generators.



Running Interfaces

- It is important to note that, even though FeynRules can obtain the vertices of very large classes of models, not every model can be output to every MC generator!
- Some interfaces to some generators have the color and / or Lorentz structures hardwired.

	Spins	Lorentz	Color
CalcHep	0,1/2,1,2	~all	1,3,8 (limited)
FeynArts	0,1/2,1	all	all
MadGraph	0,1/2,1,3/2,2	all	1,3,6,8
Sherpa	0,1/2,1	SM - like	1,3,8
Whizard	0,1/2,1,2	MSSM - like	1,3,8

N.B.: These limitations apply to the FeynRules interfaces. Some generators allow for more general structures that are however not implemented into the interface.

FeynRules MC conventions

- FeynRules itself does not make any assumption on the model, but its core is completely agnostic of any structure, like QCD, QED, etc.
- In order for the MC generator to function properly, they must be able to identify in each new model some standard information, like for example
  - ➡ Color and electric charges of particles.
  - ➡ Color structures of vertices.
  - ➡ Strong and weak coupling constant.
  - ➡ etc.
- Roughly speaking, each MC code needs the information on the SM parameters to be provided in a specific format.

FeynRules MC conventions

- As a consequence, even though the FeynRules core is completely agnostic, the SM parameters must be entered following specific conventions.

FeynRules MC conventions

- As a consequence, even though the FeynRules core is completely agnostic, the SM parameters must be entered following specific conventions.
- The SM gauge groups must be defined in the same way as in the SM implementation, e.g., for QCD,
  - ➡ Fundamental representation matrices: `T`
  - ➡ Structure constants: `f`
  - ➡ Strong coupling: `gs`

FeynRules MC conventions

- As a consequence, even though the FeynRules core is completely agnostic, the SM parameters must be entered following specific conventions.
- The SM gauge groups must be defined in the same way as in the SM implementation, e.g., for QCD,
  - ➡ Fundamental representation matrices: T
  - ➡ Structure constants: f
  - ➡ Strong coupling: gs
- The SM input parameters should correspond to the SMINPUTS of the SUSY Les Houches Accord:

$$M_Z, \alpha_s, \alpha_{EW}^{-1}, G_F$$

From FeynRules to MadGraph

- The FeynRules interface for MadGraph is the so called UFO interface. [See tomorrow's lecture]
- The UFO interface can be called via

```
WriteUFO[ L ];
```

From FeynRules to MadGraph

- The FeynRules interface for MadGraph is the so called UFO interface. [See tomorrow's lecture]
- The UFO interface can be called via

```
WriteUFO[ L ];
```

- The interface has a certain number of options, in particular all the options of `FeynmanRules[ ]` are allowed.
  - ➡ `Input`: a list of vertices. Allows to input vertices directly rather than computing them from a Lagrangian.
  - ➡ `Output`: name of the output directory, which is at the same time the name by which the model will be called in MadGraph.

From FeynRules to MadGraph

- Running the interface produces a set of text files that collectively go under name UFO (= Universal FeynRules Output).
- The content of the individual files will be explained in tomorrows lecture.
- For now, we consider a UFO a black box that contains the information about a BSM model in some specific format that can be understood by MadGraph.
- UFO is the default MadGraph model format, and so every FeynRules model can be dealt with in exactly the same way as all the built-in MadGraph models.

From FeynRules to MadGraph

- The output of FeynRules is a directory.
- It is enough to copy this directory into the /models/ subdirectory of MadGraph, and the new model is ready to use!

```
$
```

From FeynRules to MadGraph

- The output of FeynRules is a directory.
- It is enough to copy this directory into the /models/ subdirectory of MadGraph, and the new model is ready to use!

```
$./bin/mg5
```

From FeynRules to MadGraph

- The output of FeynRules is a directory.
- It is enough to copy this directory into the /models/ subdirectory of MadGraph, and the new model is ready to use!

```
mg5>
```

From FeynRules to MadGraph

- The output of FeynRules is a directory.
- It is enough to copy this directory into the /models/ subdirectory of MadGraph, and the new model is ready to use!

```
mg5>import model Phi_4_Theory
```

From FeynRules to MadGraph

- The output of FeynRules is a directory.
- It is enough to copy this directory into the /models/ subdirectory of MadGraph, and the new model is ready to use!

```
mg5>import model Phi_4_Theory

INFO: Change particles name to pass to MG5 convention
Kept definitions of multiparticles l- / j / vl / l+ / p / vl~
unchanged
```

From FeynRules to MadGraph

- The output of FeynRules is a directory.
- It is enough to copy this directory into the /models/ subdirectory of MadGraph, and the new model is ready to use!

```
mg5>import model Phi_4_Theory
INFO: Change particles name to pass to MG5 convention
Kept definitions of multiparticles l- / j / vl / l+ / p / vl~
unchanged
```

From FeynRules to MadGraph

- The output of FeynRules is a directory.
- It is enough to copy this directory into the /models/ subdirectory of MadGraph, and the new model is ready to use!

```
mg5>import model Phi_4_Theory
INFO: Change particles name to pass to MG5 convention
Kept definitions of multiparticles l- / j / vl / l+ / p / vl~
unchanged
```

- MadGraph overwrites some of the SM particle names to its own nomenclature.
- To prevent this, use the modelname option.

From FeynRules to MadGraph

- The output of FeynRules is a directory.
- It is enough to copy this directory into the /models/ subdirectory of MadGraph, and the new model is ready to use!

```
mg5>import model Phi_4_Theory
```

- MadGraph overwrites some of the SM particle names to its own nomenclature.
- To prevent this, use the modelname option.

From FeynRules to MadGraph

- The output of FeynRules is a directory.
- It is enough to copy this directory into the /models/ subdirectory of MadGraph, and the new model is ready to use!

```
mg5>import model Phi_4_Theory  --modelname
```

- MadGraph overwrites some of the SM particle names to its own nomenclature.
- To prevent this, use the modelname option.

From FeynRules to MadGraph

- The output of FeynRules is a directory.
- It is enough to copy this directory into the /models/ subdirectory of MadGraph, and the new model is ready to use!

```
mg5>import model Phi_4_Theory  --modelname


Kept definitions of multiparticles l- / j / vl / l+ / p / vl~
unchanged
```

- MadGraph overwrites some of the SM particle names to its own nomenclature.
- To prevent this, use the modelname option.

Checking a model

Checking a model

- While FeynRules provides a high level of automation, it is very important to make sure that a model implementation is correct!
- There are various checks that can be made at various levels of the chain FR > UFO > MG5.

Checking a model

- While FeynRules provides a high level of automation, it is very important to make sure that a model implementation is correct!
- There are various checks that can be made at various levels of the chain FR > UFO > MG5.
- As a first check, FeynRules allows to check that a Lagrangian is hermitian

```
CheckHermiticity[ L ]
```

- FeynRules then computes the vertices of $\mathcal{L} - \mathcal{L}^\dagger$.
- The list of non-zero vertices is printed to screen.

  N.B.: Some of these vertices might still be zero, because Mathematica is unable to detect that expressions like $T^aT^a - T^bT^b$ vanish.

Checking a model

- Gauge invariance cannot be checked directly in FeynRules.
- MadGraph 5 offers the possibility to check that a matrix element vanishes if an external gluon is replaced by its momentum.

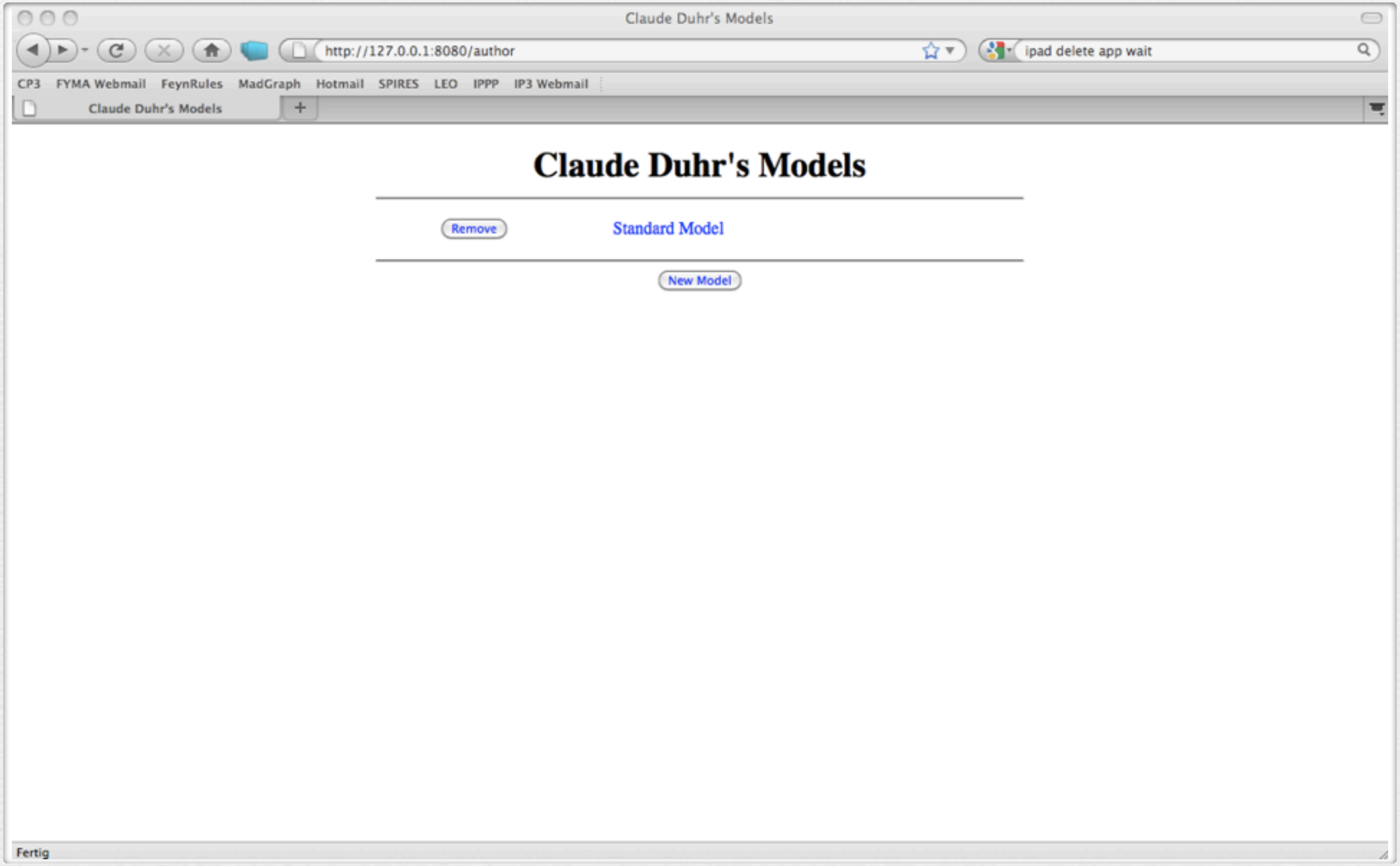
```
mg5> check gauge <process>
```

- In addition MadGraph allows to implement a model both in unitary and in Feynman gauge, so gauge invariance can be checked by comparing results in different gauges.

Checking a model

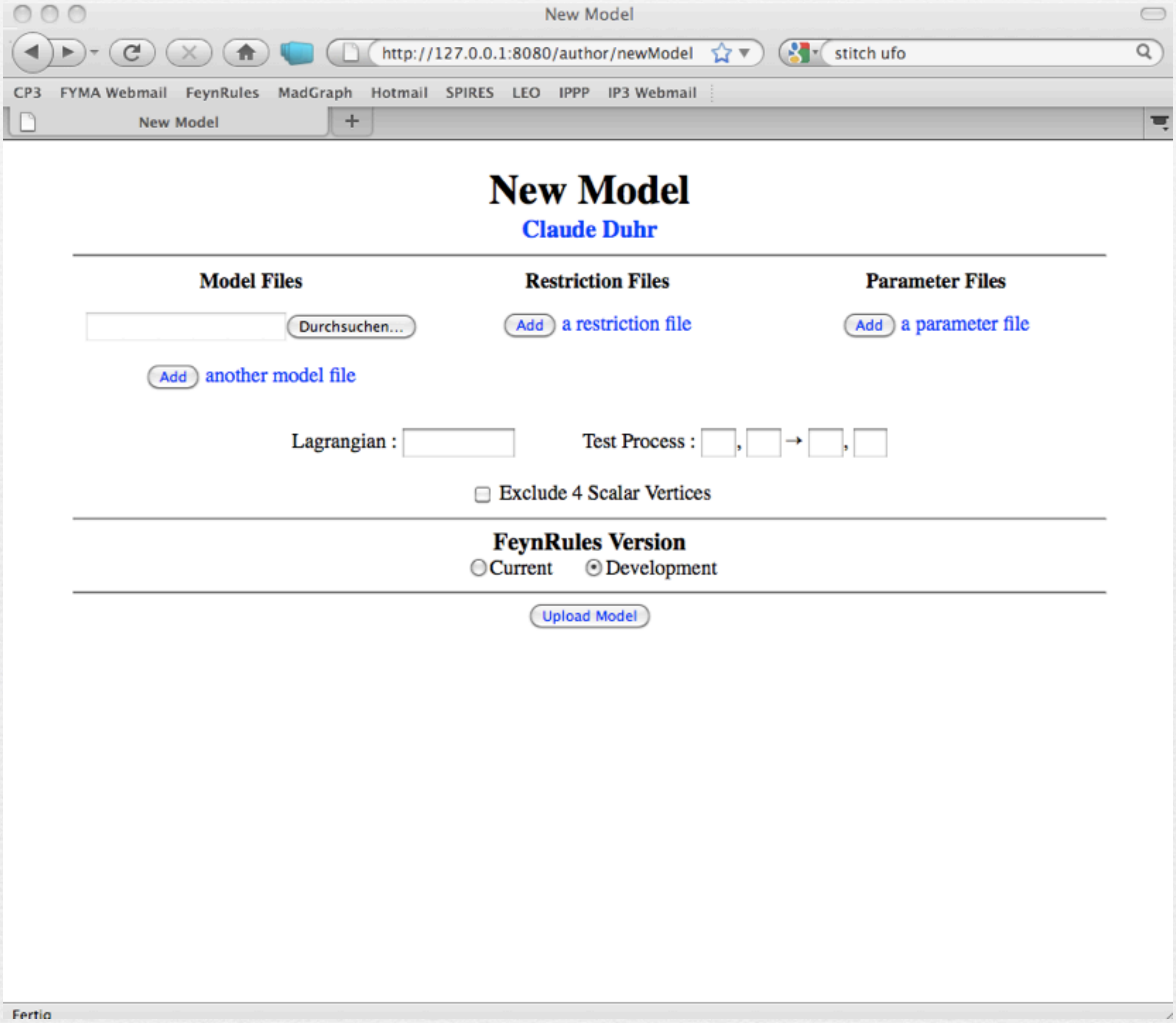
- However, we can even do better!
- As FeynRules allows to output a model to various different matrix element generators, we can compare results across different generators:
  - ➡ Gauge invariance.
  - ➡ Factors of i.
  - ➡ Different ways of dealing with unitarity cancellations.
  - ➡ Different ways of dealing with color.
- This comparison process can easily be automatized!

Web validation


Claude Duhr's Models
http://127.0.0.1:8080/author
ipad delete app wait
CP3 FYMA Webmail FeynRules MadGraph Hotmail SPIRES LEO IPPP IP3 Webmail
Claude Duhr's Models
Fertig

Claude Duhr's Models

Standard Model



New Model

http://127.0.0.1:8080/author/newModel

stitch ufo

CP3 FYMA Webmail FeynRules MadGraph Hotmail SPIRES LEO IPPP IP3 Webmail

New Model

Claude Duhr

Model Files — Durchsuchen... — Add another model file

Restriction Files — Add a restriction file

Parameter Files — Add a parameter file

Lagrangian :

Test Process : , → ,

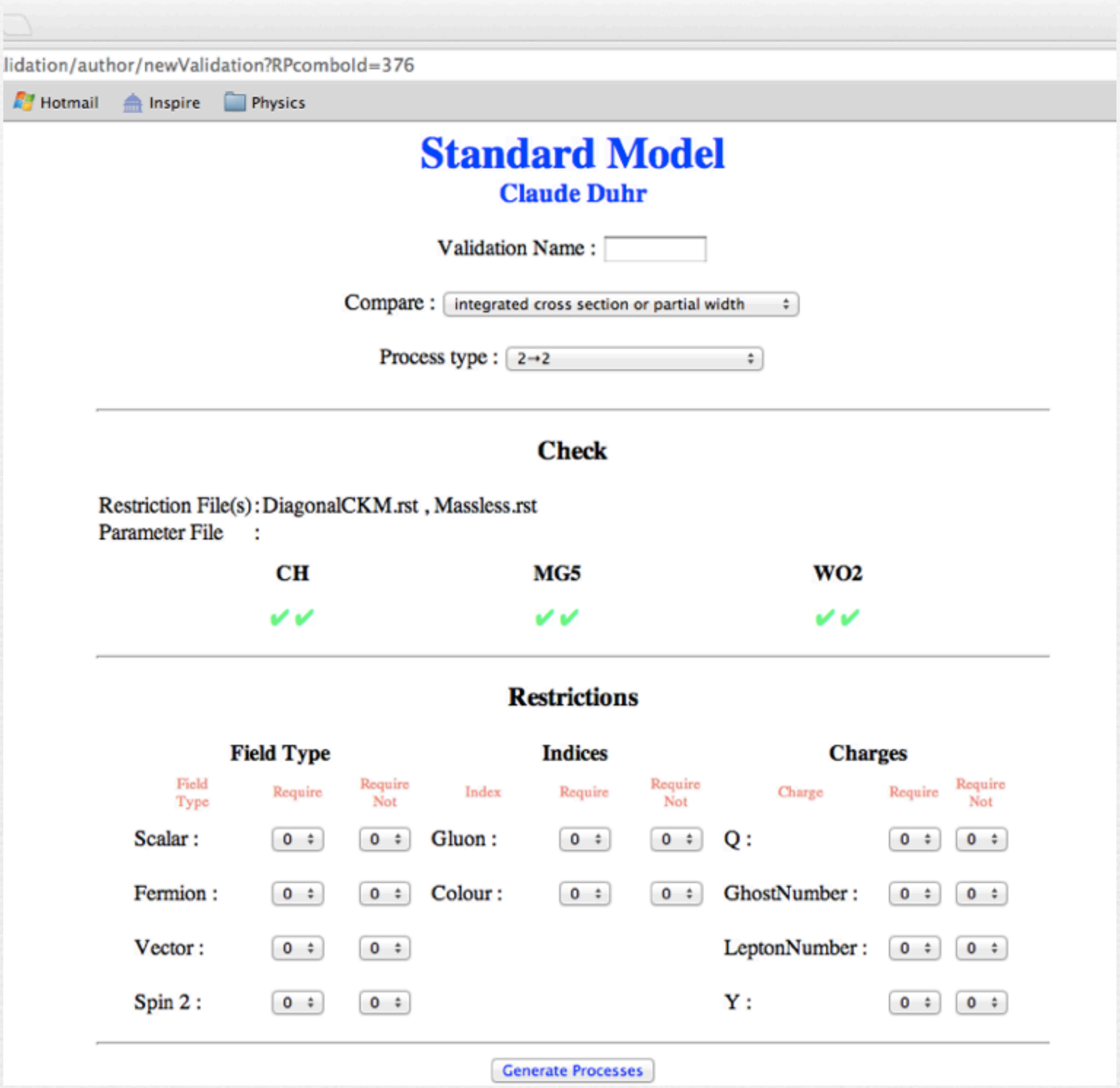
☐ Exclude 4 Scalar Vertices

FeynRules Version

Current [✓ Development]

Upload Model

Fertig


lidation/author/newValidation?RPcomboId=376

Hotmail Inspire Physics

Standard Model

Claude Duhr

Validation Name :

Compare : integrated cross section or partial width

Process type : 2→2

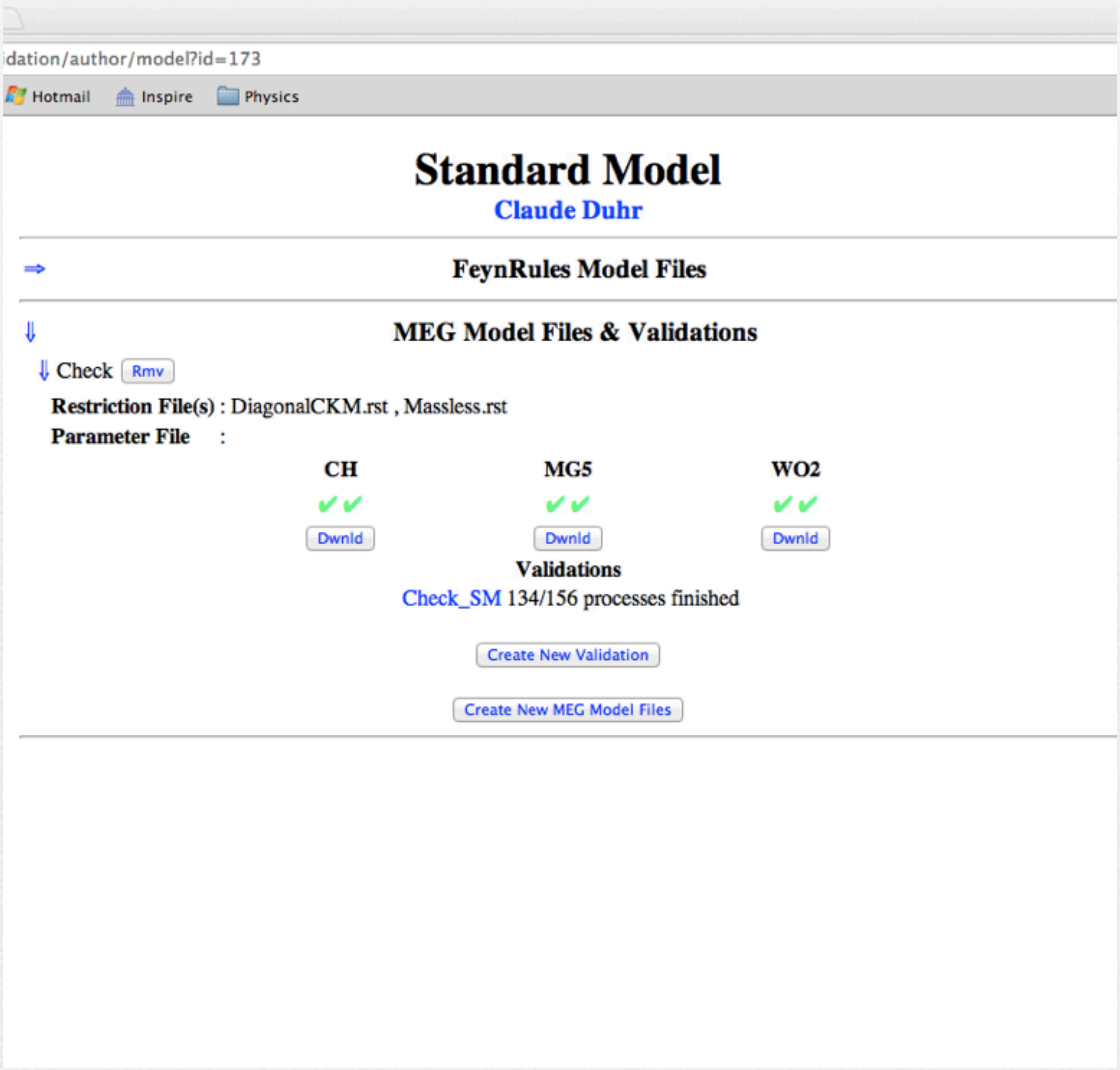
Check

Restriction File(s) : DiagonalCKM.rst , Massless.rst
Parameter File :

CH	MG5	WO2
✓✓	✓✓	✓✓

Restrictions

Field Type			Indices			Charges		
Field Type	Require	Require Not	Index	Require	Require Not	Charge	Require	Require Not
Scalar :	0	0	Gluon :	0	0	Q :	0	0
Fermion :	0	0	Colour :	0	0	GhostNumber :	0	0
Vector :	0	0				LeptonNumber :	0	0
Spin 2 :	0	0				Y :	0	0



Standard Model

Claude Duhr

⇒ FeynRules Model Files

⇓ MEG Model Files & Validations

⇓ Check Rmv

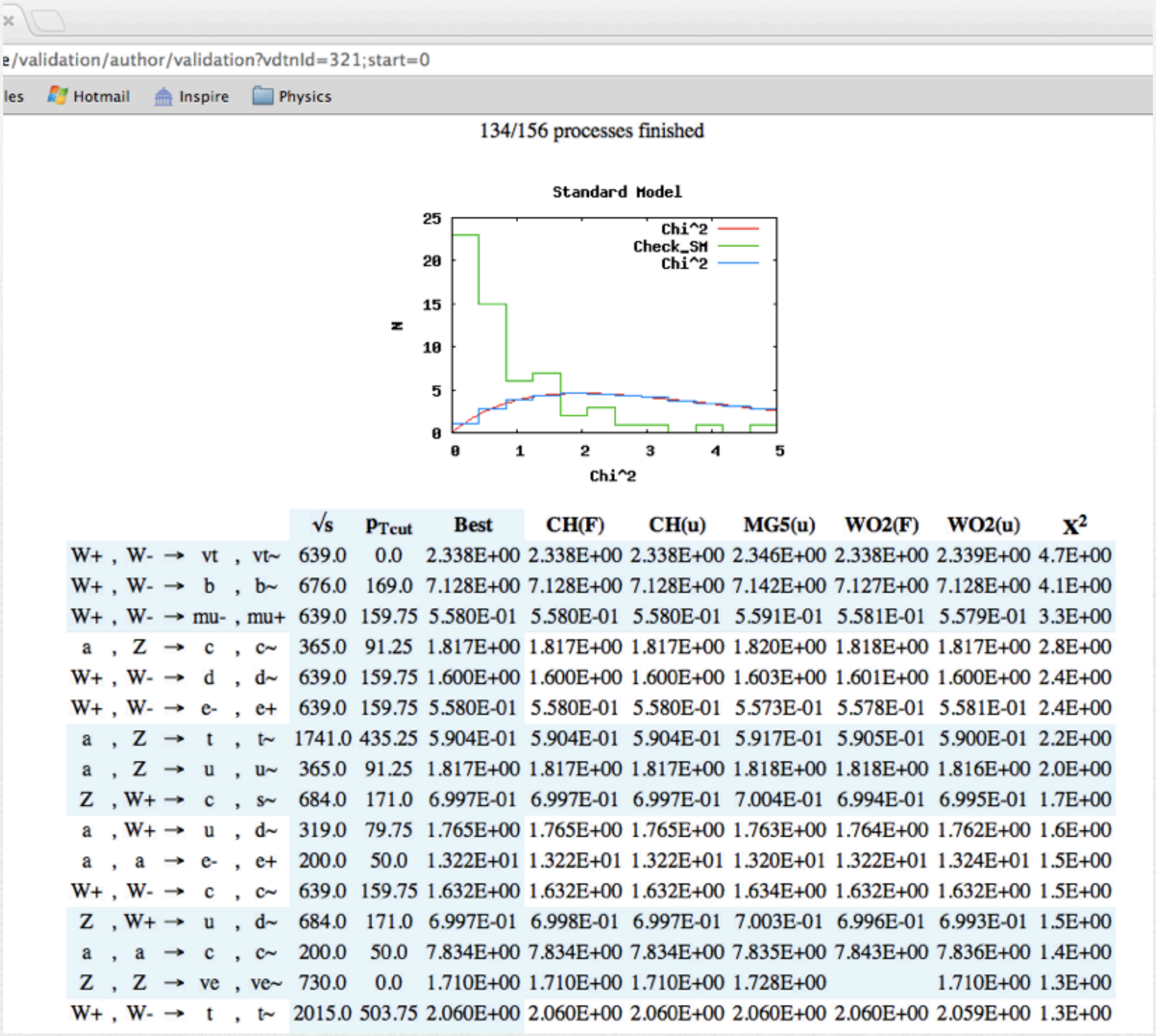
Restriction File(s) : DiagonalCKM.rst , Massless.rst

Parameter File :

CH	MG5	WO2
✓✓	✓✓	✓✓
Dwnld	Dwnld	Dwnld

Validations

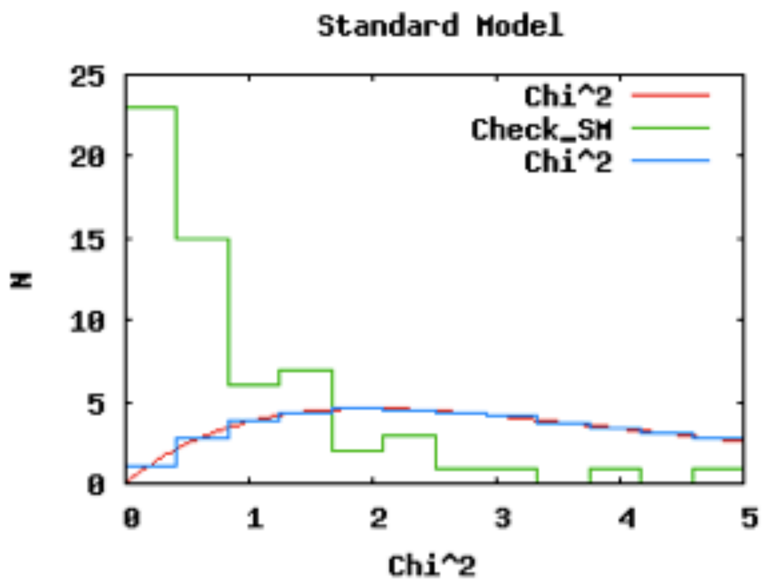
Check_SM 134/156 processes finished



e/validation/author/validation?vdtnId=321;start=0

les Hotmail Inspire Physics

134/156 processes finished



	$\sqrt{s}$	$p_{T\mathrm{cut}}$	Best	CH(F)	CH(u)	MG5(u)	WO2(F)	WO2(u)	X^2
W+ , W- → vt , vt~	639.0	0.0	2.338E+00	2.338E+00	2.338E+00	2.346E+00	2.338E+00	2.339E+00	4.7E+00
W+ , W- → b , b~	676.0	169.0	7.128E+00	7.128E+00	7.128E+00	7.142E+00	7.127E+00	7.128E+00	4.1E+00
W+ , W- → mu- , mu+	639.0	159.75	5.580E-01	5.580E-01	5.580E-01	5.591E-01	5.581E-01	5.579E-01	3.3E+00
a , Z → c , c~	365.0	91.25	1.817E+00	1.817E+00	1.817E+00	1.820E+00	1.818E+00	1.817E+00	2.8E+00
W+ , W- → d , d~	639.0	159.75	1.600E+00	1.600E+00	1.600E+00	1.603E+00	1.601E+00	1.600E+00	2.4E+00
W+ , W- → e- , e+	639.0	159.75	5.580E-01	5.580E-01	5.580E-01	5.573E-01	5.578E-01	5.581E-01	2.4E+00
a , Z → t , t~	1741.0	435.25	5.904E-01	5.904E-01	5.904E-01	5.917E-01	5.905E-01	5.900E-01	2.2E+00
a , Z → u , u~	365.0	91.25	1.817E+00	1.817E+00	1.817E+00	1.818E+00	1.818E+00	1.816E+00	2.0E+00
Z , W+ → c , s~	684.0	171.0	6.997E-01	6.997E-01	6.997E-01	7.004E-01	6.994E-01	6.995E-01	1.7E+00
a , W+ → u , d~	319.0	79.75	1.765E+00	1.765E+00	1.765E+00	1.763E+00	1.764E+00	1.762E+00	1.6E+00
a , a → e- , e+	200.0	50.0	1.322E+01	1.322E+01	1.322E+01	1.320E+01	1.322E+01	1.324E+01	1.5E+00
W+ , W- → c , c~	639.0	159.75	1.632E+00	1.632E+00	1.632E+00	1.634E+00	1.632E+00	1.632E+00	1.5E+00
Z , W+ → u , d~	684.0	171.0	6.997E-01	6.998E-01	6.997E-01	7.003E-01	6.996E-01	6.993E-01	1.5E+00
a , a → c , c~	200.0	50.0	7.834E+00	7.834E+00	7.834E+00	7.835E+00	7.843E+00	7.836E+00	1.4E+00
Z , Z → ve , ve~	730.0	0.0	1.710E+00	1.710E+00	1.710E+00	1.728E+00		1.710E+00	1.3E+00
W+ , W- → t , t~	2015.0	503.75	2.060E+00	2.060E+00	2.060E+00	2.060E+00	2.060E+00	2.059E+00	1.3E+00

Summary

- Implementing a New Physics into a matrix element generator can be a tedious and error-prone task.
- FeynRules tries to remedy this situation by providing a Mathematica framework where a new model can be implemented starting directly from the Lagrangian.
- There are no restrictions on the model, except
  - ➡ Lorentz and gauge invariance
  - ➡ Locality
  - ➡ Spins: 0, 1/2, 1, 2, ghosts (3/2 to come in the future)
- Try it out on your favorite model!
  http://feynrules.irmp.ucl.ac.be/