SARAH

A tool for (not only SUSY) model builders

Version 4.0

Florian Staub,
fnstaub@th.physik.uni-bonn.de

September 26, 2013

Disclaimer
SARAH is published under the GNU library public license [1], this means that you can use it for free. We have tested this software and its result, but we can't guarantee that this software works correctly or that the physical results derived using this software are correct.

We have the following requests:

- If you find any bug, please inform us by eMail: fnstaub@physik.uni-bonn.de
- If you have any suggestions, what is missing or can be improved, please let us know.
- If you use SARAH, please cite the appropriate references:
  - **"Linking SARAH and MadGraph using the UFO format"**
    F. Staub.
    arXiv:1207.0906 [hep-ph]
  - **"A Tool Box for Implementing Supersymmetric Models"**
    F. Staub, T. Ohl, W. Porod and C. Speckner.
    arXiv:1109.5147 [hep-ph]
    Comput. Phys. Commun. **183**, 2165 (2012)
  - **"The Electroweak sector of the NMSSM at the one-loop level"**
    F. Staub, W. Porod and B. Herrmann.
    arXiv:1007.4049 [hep-ph]
    JHEP **1010**, 040 (2010)
  - **"Automatic Calculation of supersymmetric Renormalization Group Equations and Self Energies"**
    F. Staub.
    arXiv:1002.0840 [hep-ph]
    Comput. Phys. Commun. **182**, 808 (2011)
  - **"Sarah"**
    F. Staub.
    arXiv:0806.0538 [hep-ph]

[1] http://www.fsf.org/copyleft/lgpl.html

Contents

Chapter 1

Introduction

Supersymmetry (SUSY) is one of the most popular extensions of the standard model (SM) of particle physics [1, 2, 3, 4, 5, 6]: it solves the hierarchy problem [7, 8], leads to unification of the three gauge couplings [9, 10, 11, 12] and offers often a candidate for dark matter [13].

The minimal supersymmetric standard model (MSSM) is nowadays well studied. Every event generator or diagram calculator can handle the MSSM out of the box. Unfortunately, there remains a lot of work if somebody wants to change the supersymmetric model, e.g. extend the particle content, add new gauge groups or add new interactions to the superpotential. First, it must be checked that the new model is free from gauge anomalies. As second step, the full Lagrangian must be derived and all interactions have to be extracted. This is complicated by the fact that the fields in gauge eigenstates have to be rotated to new mass eigenstates: the rotations must be incorporated, mass matrices have to be calculated and diagonalized. Finally, the tadpole equations are needed to find the minimum of the potential. All these steps are needed just to get a rough impression of the new model. If also phenomenological studies should be made by using one of the existing programs, model files must be created. Moreover, for the embedding of the model in a GUT theory, the Renormalization Group Equations (RGEs) are needed. Furthermore, often loop corrections to the masses are demanded.

This is exactly that kind of work `SARAH` was written for. `SARAH` just needs the gauge structure, particle content and superpotential to produce all information about the gauge eigenstates. As gauge groups, all $SU(N)$ groups can be handled and the superfields can transform as any arbitrary, irreducible representation of these groups. Breaking of gauge symmetries and mixings of particles can easily be added. Also the gauge fixing are automatically derived, and the corresponding ghost interactions are calculated. The two-Loop RGEs for the superpotential parameters, the gauge couplings and the soft-breaking parameters are derived. In addition, the self energies are calculated at one-loop level. `SARAH` can write all information about the model to LaTeXfiles, or create model files for `FeynArts` [14], `WHIZARD` [15] and `CalcHep`/`CompHep` [16, 17], which can also be used for dark matter studies using `MicrOmegas` [18]. In addition, also the UFO format is supported which can be used for instance with `MadGraph 5` [19].

Starting with the third version, `SARAH` is also supposed to be the first 'spectrum-generator-generator': it uses all analytical expressions to generate source code for `SPheno` [20]. The source code can be used to calculate the mass spectrum with `SPheno` for a new model using 2-loop RGEs and 1-loop corrections to the masses. In addition, the necessary routines for two- and three-body decays are written.

The intention by the development of `SARAH` was to make it very flexible: there is a big freedom for the matter and gauge sector which can be handled. The work with `SARAH` should be easy: every information `SARAH` needs are specified in an easy to modify model file. Nevertheless, `SARAH` is also fast: an existing model can be changed within minutes, and the needed time for doing all necessary calculations and writing a model file is normally less than 10 minutes.

Chapter 2

Quick start

2.1 Download and installation

SARAH is a package for Mathematica [1] and was tested with versions 5.2, 7 and 8.
SARAH can be downloaded from

http://projects.hepforge.org/sarah/

The package archive contains the following directories:

a) Models: Definition of the different models

b) Package: All package files.

c) LaTeX-Packages: LaTeXpackages, which might be needed for the output of SARAH

During the work, also the directory

Output

is created. It will contain all files written by SARAH.
In addition, the root directory of SARAH contains this manual (sarah.pdf), an overview of all models included in the package (models.pdf), a short introduction (Readme.txt) as well as an example for the evaluation of the MSSM (Example.nb), an example for creating model files for WHIZARDExample_WHIZARD.nb or in the UFO format Example_UFO.nb as well as an example how to obtain the SPheno source code Example_SPheno.nb.
The package should be extracted to the application directory of Mathematica. This directory is Linux

home/user/.Mathematica/Applications/

and

Mathematica-Directory\AddOns\Applications\

in Windows.

[1] Mathematica is a protected product by Wolfram Research

2.2 Run SARAH

After the installation, the package is loaded in Mathematica via

```
<<"[SARAH Directory]/SARAH.m"
```

and a supersymmetric model is initialized by

```
Start["Modelname"];
```

Here, `Modelname` is the name of the corresponding model file, e.g. for the minimal supersymmetric standard model the command would read

```
Start["MSSM"];
```

or for the next-to-minimal supersymmetric standard model in CKM basis

```
Start["NMSSM","CKM"];
```

is used. In the following, we refer for all given examples the model file of the MSSM. Our conventions concerning the fields definitions and rotations in the MSSM are given in app. G.

2.3 What happens automatically

When a model is initialized using the `Start` command, this model is first checked for gauge anomalies and charge conservation. If not all checks are fulfilled, a warning is printed. More information about the different checks is given in app. D. Afterwards, the calculation of the complete Lagrangian at tree level starts. The performed steps are presented in app. B.1.

The next steps are to accomplish all necessary rotations and redefinition of fields: if a gauge symmetry is broken, the fields responsible for the symmetry breaking are getting a vacuum expectation value (VEV) and the gauge fields are rotated. Afterwards, the matter particles are rotated to the new mass eigenbasis and the tadpole equations are derived. These steps can be repeated if more rotations or symmetry breakings are necessary.

During this evaluation some more things might done automatically: particles are integrated out to get an effective theory, the ghost interactions are derived, the mass matrices and tadpole equations are calculated at tree level. At the end, SARAH splits the Lagrangian in different pieces to increase the speed of following calculations. Additionally, if numerical values for all parameters are provided, e.g. in the input files or in a LesHouches spectrum file [21], SARAH calculates the eigenvalues of the mass matrices and the rotation matrices.

2.4 Commands

The most important commands to work with SARAH are:

a) `ShowModels`: Shows a list with all installed models

b) `SARAH‘FirsSteps`: Shows a short introduction to SARAH

c) `CheckModel`: Performs several checks of the implementation of the current model

d) `Vertex[Fields, Options]`: Calculates a vertex for given fields

e) `MassMatrices[$EIGENSTATES]`: Shows all mass matrices for given eigenstates `$EIGENSTATES`

f) `TadpoleEquations[$EIGENSTATES]`: Shows all tadpole equations for given eigenstates `$EIGENSTATES`

g) `MassMatrix[Field]`: Shows the mass matrix of the field `Field`

h) `TadpoleEquation[X]`: Shows the tadpole equation corresponding to a vev or a scalar particle `X` to the VEV `VEV`

i) `CalcRGEs[Options]`: Calculates the RGEs

j) `CalcLoopCorrections[Options]`: Calculates one-loop one and two-points functions for given eigenstates `$EIGENSTATES`

k) `ModelOutput[$EIGENSTATES,Options]`: Create output defined by options for given eigenstates `$EIGENSTATES`.

l) `MakeVertexList[$EIGENSTATES,Options]`: Calculates all vertices for given eigenstates `$EIGENSTATES`

m) `MakeSPheno[Options]`: Writes source code for `SPheno`

n) `MakeTeX[Options]`: Writes LaTeXfiles

o) `MakeCHep[Options]`: Writes `CalcHep`/`CompHep` model files

p) `MakeFeynArts[Options]`: Writes `FeynArts` model file

q) `MakeWHIZARD[Options]`: Writes model files for `WHIZARD` and `Omega`

r) `MakeUFO[Options]`: Writes model files in the UFO format

s) `MakeLHPCstyle[$EIGENSTATES]`: Writes steering files for the LHPC spectrum plotter

t) `MakeAll[Options]`: Generates the output for `SPheno`, `WHIZARD`, `CalcHep`, `FeynArts` and in the UFO format as well as the LaTeX files

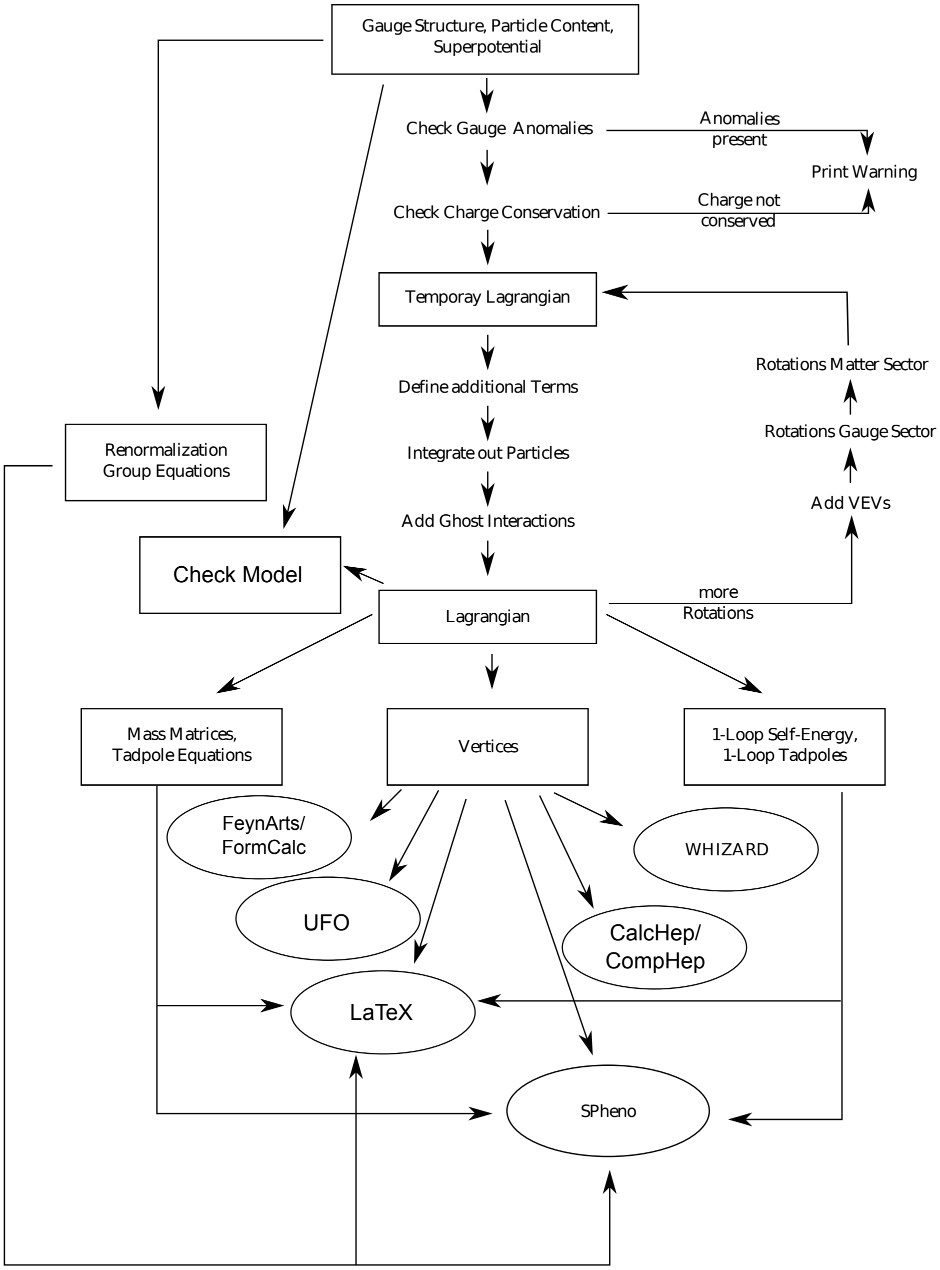

Figure 2.1: Setup of SARAH

Chapter 3

Working with SARAH

3.1 Definition of particles

Before we explain the different functions of SARAH, we have to clarify the nomenclature. All gauge eigenstates are named as follows:

```
ParticleType <> Basis <> [Indices]
```

Type Here, ParticleType is just one letter and indicates the type of a field. The convention is as follows:

a) F for fermionic component of chiral superfield

b) S for scalar component of chiral superfield

c) f for fermionic component of vector superfield

d) V for bosonic component of vector superfield

e) g for ghost field

In addition, there are two types of auxiliary fields. This auxiliary field are not related in any way to the auxiliary components of the superfields in SUSY theories, but they are needed only for writing a CalcHep/CompHep model file (see sec. 8.1.2). The type indicating letters are:

a) A for an auxiliary scalar

b) a for an auxiliary vector boson

> After rotating gauge fields to new mass eigenstates, there are no longer constraints concerning the names of fermions and scalars. However, vector bosons must still begin with V because the corresponding ghosts are automatically added!

Basis The Basis of a particle is the name of the underlying superfield. In the MSSM for example, this can be dL for the superfield of the left down quarks and squarks or G for the vector superfield transforming under the strong interaction.

Indices `Indices` is the list of the indices which the particle carries. There are three different kinds of indices:

a) `generation`: For all particles which appear in more than one generation

b) `lorentz:` For all particles carrying a Lorentz index

c) `charge:` For all components of a chiral superfield charged under a non-Abelian gauge group if this indices are not implicit

d) `adjoint:` For all all components of a vector field for a non-Abelian gauge group if this indices are not implicit

Examples To clarify the above definitions, here some examples:

a) `VB[{lorentz}]`: B-Boson with one Lorentz index

b) `fB`: Bino

c) `SHd0`: Neutral down Higgs

d) `FHd0`: Neutral down Higgsino

e) `VG[{adjcolor,Lorentz}]`: Gluon with one index for the adjoint of the color group and one Lorentz index

f) `fG[{adjcolor}]`: Gluino with one adjoint index

g) `gG[{adjcolor}]`: Gluon ghost with one adjoint index

h) `SdL[{generation,color}]`: Left handed d-squark with one generation and one color index

i) `FdL[{generation,color}]`: Left handed d-quark with one generation and one color index

j) `hh[{generation}]`: Neutral, CP-even Higgs (light and heavy Higgs) after EWSB

A comprehensive overview about all models defined in
SARAH is given in the file `Models.pdf`. Furthermore, the MSSM is discussed in app. G.

3.1.1 Antiparticles

There are two functions to assign antiparticles: depending on the type of the particle `conj` or `bar` are used.

a) Scalar, vector boson and Weyl spinor: `conj`, e.g. `conj[SdR]` or `conj[VWm]`.

b) Dirac fermion and ghost: `bar`, e.g `bar[Fd]` or `bar[gG]`.

`SARAH` checks if a particle is a real scalar or vector bosons respectively a Majorana fermion. In these cases it simplifies the expressions by using

```
conj[RP] := RP
bar[MF] := MF
```

for a real particle RP or also real parameter, and for a Majorana fermion MF. The names of all Majorana fermions of the current model are saved in the list MajoranaPart, and all real parameters and particles are listed in realVar.

Note, that the head bar is overloaded: it is either interpreted as hermitian or complex conjugated depending on the position of the fermions in a Dirac chains in order to build up Lorentz scalars.

Some words about the necessity of conj: The function conj is very similar to the existing function Conjugate of Mathematica. But at least with Mathematica 5.2 there are some problems concerning this function: it is not possible to calculate the derivative with respect to a complex conjugate variable, and Conjugate is a numerical function. Hence, it is in some cases too slow in handling big analytical expressions. This was improved in Mathematica 7.0 but we want to stay compatible also with version 5.2 and keep therefore conj.

3.2 Different eigenstates of one model

While calculating the Lagrangian of a model, SARAH saves the information of the different eigenstates of the model. The name of the eigenstates can be defined by the user in the model file, see sec. 5.6. For the model file included in the official package, the eigenstates are

a) GaugeES: gauge eigenstates without any rotation

b) EWSB: eigenstates after electroweak symmetry breaking

c) SCKM: eigenstates in Super-CKM basis

d) TEMP: auxiliary eigenstates with no physical meaning

3.3 Model information

There is a lot of information automatically calculated and saved by SARAH when initializing a model. In this section is shown, how this information can be accessed and used.

3.3.1 Particle content

To get an overview of all particles of the different eigenstates, use

```
Particles[Eigenstates]
```

e.g. Particles[GaugeES] or Particles[EWSB] for the gauge eigenstates or the eigenstates after electroweak symmetry breaking (EWSB), respectively. The output is a list with the following information about each particle:

a) Name of the particle

b) Type of the particle (F for fermion, S for scalar, G for ghosts, A for auxiliary field)

c) Number of first generation (can be different from 1 in effective theories)

d) Number of last generation

e) Indices of the particle

Example For instance, the entry for the gauge eigenstates of the left-down quark reads

```
{FdL, 1, 3, F, {generation, color}}
```

3.3.2 Masses of particles

SARAH automatically calculates the tree level masses of all particles. This information is saved in

```
Masses[$EIGENSTATES]
```

It returns are a list with replacements rules. First, a statement `Mass[]` with the name of the particle is given followed by the value. There are three possibilities for the value

a) Expression: if an expression is used for a mass, SARAH has calculated the tree level mass depending on other parameters of the model.

b) `MassGiven`: this means that a numerical value for the mass was given in the particle file (see 4.3).

c) `MassRead`: this means that the value of the mass was read in from a LesHouches input file.

More information of defining masses is given in section 4.3.

Example The mass of the Z-Boson (`VZ`) after EWSB is saved in `Masses[EWSB]`. The corresponding entry is

```
Mass[VZ] -> ((vd^2 + vu^2)*(g2*Cos[ThetaW] + g1*Sin[ThetaW])^2/4
```

3.3.3 Mass Matrices

SARAH calculates automatically the mass matrices before rotating the fields to the new eigenstates and saves the information in arrays. The basis of the rotations can be seen by using

```
MixBasis[$EIGENSTATES]
```

The matrices itself are saved in two arrays:

```
MassMatrices[$EIGENSTATES]
```

and

```
MassMatricesFull[$EIGENSTATES]
```

The difference between this two arrays is that in the first one, the different generations are written as indices, while in the second on the generation indices are explicitly inserted. This means, in the first case the basis for the mass matrix in the down squark sector is just

```
(SdL[{gn,cn}],SdL[{gm,cm}])
```

while in the second case the basis vector is

```
(SdL[{1,cn1}],SdL[{2,cn2}],SdL[{3,cn3}],SdL[{1,cm1}],SdL[{2,cm2}],SdL[{3,cm2}])
```

It is also possible to use the command

```
MassMatrix[Field]
```

to obtain the mass matrix for a specific particle.

Example Let's have a look at the down-squark sector after EWSB.

```
MixBasis[EWSB][[1]]
```

returns the basis of the mass matrix:

```
{{SdL, SdR}, {conj[SdL], conj[SdR]}}
```

The (1,1) element of the mass matrix is saved in

```
MassMatricesFull[EWSB][[1,1,1]]
```

and looks like

```
-(g1^2*vd^2)/24 - (g2^2*vd^2)/8 + (g1^2*vu^2)/24 +
   (g2^2*vu^2)/8 + mq2[1, 1] + (vd^2*Yd[1, 1]^2)/2
```

The same result is obtained by

```
MassMatrix[Sd][[1,1]]
```

3.3.4 Tadpole Equations

Gauge symmetries are broken, if particles receive a VEVs. These VEVs v_i (or equivalently the real components ϕ of the scalar field) should minimize the potential and therefore fulfill the equations

$$\frac{\partial V}{\partial v} = \frac{\partial V}{\partial \phi} = 0 \tag{3.1}$$

V is the scalar potential of the model. If CP is violated in the Higgs sector, similar conditions arise for the pseudo-scalar components of the fields

$$\frac{\partial V}{\partial \sigma} = 0 \tag{3.2}$$

Eq. (3.1) are the so called tadpole equations which are cubic equations in the VEVs. SARAH saves the left hand side of (3.1) in arrays named

```
TadpoleEquations[$EIGENSTATES]
```

The order of the tadpole equations in this array corresponds to the order of the definition of VEVs in the model file, see (5.6.1.1).
There is also the shorter command

```
 TadpoleEquation[X]
```

to obtain the tadpole equations corresponding to a specific VEV or state.

Example The tadpole equation for v_d after EWSB is saved in

```
TadpoleEquations[EWSB][[1]]
```

and reads

```
mHd2*vd + (g1^2*vd^3)/8 + (g2^2*vd^3)/8 - (g1^2*vd*vu^2)/8 -
  (g2^2*vd*vu^2)/8 + vd*\[Mu]^2 - vu*B[\[Mu]]
```

The same result can be obtained by

```
 TadpoleEquation[vd]
```

or

```
 TadpoleEquation[phid]
```

3.3.5 Parts of the Lagrangian

All information about a model and it's interactions are encoded in the full Lagrangian. `SARAH` calculates the full Lagrangian from the superpotential and the gauge sector by using the method explained in appendix B.1. The final results, the Lagrangians for the different eigenstates are saved as

```
Lagrangian[$EIGENSTATES]
```

For a realistic SUSY model, the Lagrangian is generally very lengthy. Therefore, `SARAH` splits it in different parts in order to speed up some calculations. This splitting might be also helpful for analyzing the structure of interactions involving different kinds of fields. An overview of all names for the different parts of the Lagrangian is given in app. C.
Also the results of the different steps during then calculation of the Lagrangian in gauge eigenstates (e.g. F-Terms, D-Terms, kinetic parts) are saved. Thus, it is possible to have a detailed look at specific parts of the Lagrangian. The names of the parts are also given in app. B.1.

3.3.6 Writing all information about particles and parameters in an external file

All information about particles and parameters of the considered model can by written in two files `ParticleInfo.m` and `ParameterInfo`, which are saved in

```
../\SARAH/Output/"Model Name"/
```

by using

```
ExportModelInformation;
```

The file `ParticleInfo.m` contains the following information for all eigenstates of the model

a) R-Parity

b) PDG, PDG.IX

c) LaTeX name

d) Output name

e) `FeynArts` number

f) Type

g) Self-conjugated or not

h) Number of generations

i) Indices

j) Electric charge

k) Description

l) Mass, Width

while `ParameterInfo.m` contains the following information about all parameters

a) Dependence on other parameters

b) Real or complex

c) Numerical value

d) Position in LesHouches input

e) LaTeX name

3.4 Calculating Vertices

One of the main functions of `SARAH` is to calculate the vertices for a model. In contrast to the most other calculations, vertices are not calculated automatically when initializing a model: it can last several minutes to calculate all vertices of a model and sometimes these calculations are not necessary. Of course, it is also possible to tell `SARAH` that all vertices should automatically calculated, see section 3.4.2. In this section, we want to focus on calculating vertices 'by hand'.

3.4.1 Calculating specific vertices

Vertices are calculated by

```
Vertex[ParticleList,Options]
```

`ParticleList` is a list containing the involved fields. This list can consist of up to 6 particles if an effective theory is analyzed (see sec. 5.7).
The following `Options` are supported by the `Vertex` command:

a) `Eigenstates`, Value: `$EIGENSTATES`, Default: Last entry in `NameOfState`
Fixes the considered eigenstates

b) `UseDependences`: Value `True` or `False`, Default: `False`
Optional relations between the parameters (see section 4.2) will be used, if `UseDependences` is set to `True`.

The output of `Vertex` is an array:

```
{{ParticleList},{{Coefficient 1, Lorentz 1},{Coefficient 2, Lorentz 2},...}
```

First, the list of the involved particles is given and the indices are inserted. The second part consists of the value of the vertex and can be also a list, if different Lorentz structures are possible. In the part independent of any Lorentz index the following symbols can appear

a) `Delta[a,b]`: Kronecker delta $\delta_{\alpha\beta}$

b) `ThetaStep[i,j]`: Step function Θ_{ij}

c) `Lam[t,a,b]`: Gell-Mann matrix $\lambda^t_{\alpha\beta}$

d) `LambdaProd[x,y][a,b]`: Matrix product of two Gell-Mann matrices $(\lambda^x\lambda^y)_{\alpha\beta}$

e) `Sig[t,a,b]`: Pauli matrix $\sigma^t_{\alpha\beta}$

f) `SigmaProd[x,y][a,b]`: Matrix product of two Pauli matrices $(\sigma^x\sigma^y)_{\alpha\beta}$

g) `fSU3[i,j,k]`: Structure constants of $SU(3)$: f^{ijk}

h) `fSU2[i,j,k]`: Structure constants of $SU(2)$: ϵ^{ijk}

i) `FST[SU[N]][i,j,k]`: Structure constants of $SU(N)$: f_N^{ijk}

j) `TA[SU[N]][a,i,j]`: Generator of $SU(N)$: T^a_{ij}

k) Couplings, e.g. `g1`, `g2`, `g3`, `Ye[a,b]`, `Yd[a,b]`, `Yu[a,b]`, ...

l) Mixing matrices, e.g. `ZD[a,b]`, ...

The part transforming under the Lorentz group can consist of

a) `gamma[lor]`: Gamma matrix γ_μ

b) `g[lor1,lor2]`: Metric tensor $g_{\mu\nu}$

c) `Mom[particle,lor]`: Momentum p^μ_P of particle P

d) `PL`, `PR`: Polarization operators $P_L = \frac{1-\gamma_5}{2}$, $P_R = \frac{1+\gamma_5}{2}$

e) `1`: If the vertex is a Lorentz scalar.

f) `LorentzProduct[_,_]`: A non commutative product of terms transforming under the Lorentz group

Examples Some examples to clarify the usage and output of `Vertex`:

a) **One possible Lorentz structure:** `Vertex[hh,Ah,Z]` leads to the vertex of scalar and a pseudo scalar Higgs with a Z-boson

```
{{hh[{gt1}], Ah[{gt2}], VZ[{lt3}]},
 {((MA[gt2,1]*MH[gt1,1] - MA[gt2,2]*MH[gt1,2])*(g2*Cos[ThetaW]+g1*Sin[ThetaW]))/2,
                Mom[Ah[{gt2}], lt3] - Mom[hh[{gt1}],lt3]}}
```

The output is divided in two parts. First, the involved particles are given, second, the value of the vertex is given. This second part is again split in two parts: one is the Lorentz independent part and the second part defines the transformation under the Lorentz group.

b) **Several possible Lorentz structures** `Vertex[bar[Fd],Fd,hh]` is the interaction between d-quarks and a Higgs:

```
{{bar[Fd[{gt1, ct1}]], Fd[{gt2, ct2}], hh[{gt3}]},
 {((-I)*Delta[ct1,ct2]*Delta[gt1,gt2]*MH[gt3,2]*Yd[gt2,gt1])/Sqrt[2],PL},
 {((-I)*Delta[ct1,ct2]*Delta[gt1,gt2]*MH[gt3,2]*Yd[gt1,gt2])/Sqrt[2],PR}}
```

Obviously, there are three parts: one for the involved particles and two for the different Lorentz structures. `PL` and `PR` are the polarization projectors $P_L = \frac{1}{2}(1-\gamma_5), P_R = \frac{1}{2}(1+\gamma_5)$.

c) **Changing the considered eigenstates and using Weyl fermions** It is also possible to calculate the vertices for Weyl fermions and/or to consider the gauge eigenstates. For instance,

```
Vertex[{fB, FdL, conj[SdL]}, Eigenstates -> GaugeES]
```

returns

```
{{fB, FdL[{gt2, ct2}], conj[SdL[{gt3, ct3}]]},
 {((-I/3)*g1*Delta[ct2, ct3]*Delta[gt2, gt3])/Sqrt[2],1}}
```

d) Using dependences With `Vertex[{conj[Se], Se, VP}, UseDependences -> True]` g_1 and g_2 are replaced by the electric charge e. This and similar relations can be defined in the parameters file (see sec. 4.2.3).

```
{{conj[Se[{gt1}]], Se[{gt2}], VP[{lt3}]},
{(-I)*e*Delta[gt1,gt2],-Mom[conj[Se[{gt1}]],lt3]+Mom[Se[{gt2}],lt3]}}
```

e) Fixing the generations It is possible to give the indices of the involved particles already as input

```
Vertex[{hh[{1}], hh[{1}], Ah[{2}], Ah[{2}]}]
```

leads to

```
{{hh[{1}], hh[{1}], Ah[{2}], Ah[{2}]},
 {(-I/4)*(g1^2 + g2^2)*Cos[2*\[Alpha]]*Cos[2*\[Beta]], 1}}
```

Obviously, the given definition of the mixing matrices for the Higgs fields were automatically inserted. If the indices are fixed by a replacement, the definition of the mixing matrix wouldn't be used

```
Vertex[{hh, hh, Ah, Ah}] /. {gt1->1, gt2->1,gt3->2, gt3->2}
```

returns

```
{{hh[{1}], hh[{1}], Ah[{2}], Ah[{gt4}]},
 {(-I/4)*(g1^2 + g2^2)*(conj[ZA[2, 1]]*conj[ZA[gt4, 1]] -
   conj[ZA[2, 2]]*conj[ZA[gt4, 2]])*(conj[ZH[1, 1]]^2 - conj[ZH[1, 2]]^2), 1}}
```

However,

```
Vertex[{hh, hh, Ah, Ah}] /. {gt1->1, gt2->1,gt3->2, gt3->2} /.subAlways
```

leads to the former expression using the mixing angle β.

f) Effective operators In effective theories also interactions between two fermions and two scalars are possible. As example an effective vertex for a model in which the gluino was integrated out:

```
Vertex[{Fd, Fd, conj[Sd], conj[Sd]}]
```

Returns

```
{{Fd[{gt1, ct1}], Fd[{gt2, ct2}], conj[Sd[{gt3, ct3}]], conj[Sd[{gt4, ct4}]]},
{-(g3^2*(sum[j1, 1, 8, (Lam[j1, ct3, ct2]*Lam[j1, ct4, ct1])/Mass[fG][j1]]*
          ZD[gt3, gt2]*ZD[gt4, gt1] +
         sum[j1, 1, 8, (Lam[j1, ct3, ct1]*Lam[j1, ct4, ct2])/Mass[fG][j1]]*
          ZD[gt3, gt1]*ZD[gt4, gt2])),
    LorentzProduct[PL, PL]}, {0, LorentzProduct[PR, PL]},
{g3^2*(sum[j1, 1, 8, (Lam[j1, ct2, ct3]*Lam[j1, ct4, ct1])/Mass[fG][j1]]*
       ZD[gt3, 3 + gt2]*ZD[gt4, gt1] +
       sum[j1, 1, 8, (Lam[j1, ct2, ct4]*Lam[j1, ct3, ct1])/Mass[fG][j1]]*
       ZD[gt3, gt1]*ZD[gt4, 3 + gt2]),
    LorentzProduct[PL, PR]}, {0, LorentzProduct[PR, PR]},
{0, LorentzProduct[gamma,PL, PL]}, {0, LorentzProduct[gamma, PR, PL]},
{0, LorentzProduct[gamma, PL, PR]}, {0, LorentzProduct[gamma,PR, PR]}}
```

Obviously, SARAH checks the eight possible operators (4 different combination of polarization operators with and without a γ matrix) and returns the result for each operator.

3.4.2 Calculating all vertices

To calculate all vertices at once for a given model, use

`MakeVertexList[Eigenstates,Options]`

First, the name of the eigenstates has to be given. The possible options are:

a) `effectiveOperators`, Values: `True` or `False`, Default: `False`
If also higher dimensional operators should be calculated. By default, this concerns only four point interactions.

b) `SixParticleInteractions`, Values: `True` or `False`, Default: `False`
If also the six-point interactions should be calculated.

c) `GenericClasses`, Values: `All` or a list of generic types, Default: `All`
Calculates the vertices only for the given types of interaction

The results are saved in list named

`SA‘VertexList[Type]`

with `Type` = `SSS,SSSS,SSVV,SSV,SVV,FFS,FFV,VVV,VVVV,GGS,GGV,ASS`.

3.5 Renormalization Group Equations

SARAH calculates the renormalization group equations (RGEs) for the parameters of the superpotential, the soft-breaking terms and the gauge couplings at one and two loop level. This is done by using the generic formulas of [22] extended by the results for several Abelian gauge groups [23] and Dirac mass terms for gauginos [24]. In addition, the gauge dependence in the running of the VEVs is included [25]

For non-SUSY models the RGEs of a general quantum field theory are calculated by using the results of Refs. [26, 27, 28, 29]. In addition, also the impact of kinetic mixing is coverd by using the rules of Ref. [30] and also the gauge dependence in the running VEVs is included [25].
The calculation is started via

`CalcRGEs[Options]`

Options The different options are

a) `TwoLoop`, Value: `True` or `False`, Default: `True`
If also the two loop RGEs should be calculated.

b) `ReadLists`, Value: `True` or `False`, Default: `False`
If the RGEs have already be calculated, the results are saved in the output directory. The RGEs can be read from these files instead of doing the complete calculation again.

c) `VariableGenerations`, Value: List of particles, Default: `{}`
Some theories contain heavy superfields which should be integrated out above the SUSY scale. Therefore, it is possible to calculate the RGEs assuming the number of generations of specific superfields as free variable to make the dependence on these fields obvious. The new variable is named `NumberGenertions[X]`, where `X` is the name of the superfield.

d) `NoMatrixMultiplication`, Values: `True` or `False`, Default: `False`
Normally, the β-functions are simplified by writing the sums over generation indices as matrix multiplication. This can be switched off using this option.

e) `IgnoreAt2Loop`, Values: a list of parameters, Default: `{}`
The calculation of 2-loop RGEs for models with many new interactions can be very time-consuming. However, often one is only interested in the dominant effects of the new contributions at the 1-loop level. Therefore, `IgnoreAt2Loop -> $LIST` can be used to neglect parameters at the two-loop level The entries of $LIST can be superpotential or soft SUSY-breaking parameters as well as gauge couplings.

f) `WriteFunctionsToRun`, `True` or `False`, Default: `True`
Defines, if a file should be written to evaluate the RGEs numerically in `Mathematica`

The β-functions for SUSY will be stored in the following arrays:

a) `Gij`: Anomalous dimensions of all chiral superfields

b) `BetaWijkl`: Quartic superpotential parameters

c) `BetaYijk`: Trilinear superpotential parameters

d) `BetaMuij`: Bilinear superpotential parameters

e) `BetaLi`: Linear superpotential parameters

f) `BetaQijkl`: Quartic soft-breaking parameters

g) `BetaTijk`: Trilinear soft-breaking parameters

h) `BetaBij`: Bilinear soft-breaking parameters

i) `BetaSLi`: Linear soft-breaking parameters

j) `Betam2ij`: Scalar squared masses

k) `BetaMi`: Majorana Gaugino masses

l) `BetaGauge`: Gauge couplings

m) `BetaVEVs`: VEVs

n) `BetaDGi`: Dirac gaugino mass terms

and for non-SUSY models in

- `Gij`: Anomalous dimensions of all fermions and scalars
- `BetaGauge`: Gauge couplings
- `BetaLijkl`: Quartic scalar couplings
- `BetaYijk`: Interactions between two fermions and one scalar
- `BetaTijk`: Cubic scalar interactions
- `BetaMuij`: Bilinear fermion term
- `BetaBij`: Bilinear scalar term
- `BetaVEVs`: Vacuum expectation values

These arrays are also saved in the directory

```
../\SARAH/Output/"ModelName"/RGE
```

All entries of this arrays are three dimensional: The first entry is the name of the parameter, the second the one-loop β-function and the third one the two loop β-function.

GUT normalization The gauge couplings of $U(1)$ gauge groups are often normalized at the GUT scale with respect to a specific GUT group. Therefore, it is possible to define for each gauge coupling the GUT-normalization by the corresponding entry in the parameters file. See sec. 4.2.3 for more information.

Generally, the results contain sums over the generation indices of the particles in the loop. `SARAH` always tries to write them as matrix multiplications, in order to shorten the expressions. Therefore, new symbols are introduced:

a) `MatMul[A,B,C,...][i,j]`: $(ABC\dots)_{i,j}$. Matrix multiplication, also used for vector-matrix and vector-vector multiplication.

b) `trace[A,B,C,...]`: $\mathrm{Tr}(ABC\dots)$. Trace of a matrix or of a product of matrices.

c) `Adj[M]`: $M^\dagger$. Adjoint of a matrix

d) `Tp[M]`: M^T. Transposed of a matrix

Remarks Some remarks about the output:

a) To differ between generation and other indices during the calculation, `Kronecker[i,j]` is used for generation indices instead of `Delta[i,j]`.

b) The results for the scalar masses are simplified by using abbreviations for often appearing traces, see also Ref. [22]. The definition of the traces are saved in the array `TraceAbbr`.

c) If the model contains parameters with three indices, matrix multiplication is automatically switched off and the results are given as sum over the involved indices. In addition, these expressions are simplified by replacing a parameter with three indices by a sum of parameters with two indices. The β function in this form a saved in `NAME <> 3I` with `NAME` stands for the standard array containing the RGEs.

Examples

a) **β-function of Yukawa coupling** The Yukawa couplings of the MSSM are saved in BetaYijk. The first entry consists of

```
BetaYijk[[1,1]]:  Ye[i1,i2] ,
```

i.e. this entry contains the β-functions for the electron Yukawa coupling. The corresponding one-loop β-function is

```
BetaYijk[[1,2]]:
(-9*g1^2*Ye[i1,i2])/5-3*g2^2*Ye[i1,i2]+3*trace[Yd,Adj[Yd]]*Ye[i1,i2]+
  trace[Ye,Adj[Ye]]*Ye[i1, i2]+3*MatMul[Ye,Adj[Ye],Ye][i1, i2]
```

The two-loop β-function is saved in BetaYijk[[1,3]] but we skip it here because of its length.

b) **β-function of soft-breaking masses and abbreviations for traces** The soft-breaking mass of the selectron is the first entry of Betam2ij

```
 Betam2ij[[1,1]]:           me2[i1,i2]
```

and the one-loop β-function is saved in Betam2ij[[1,2]]:

```
(-24*g1^2*MassB*conj[MassB]+10*g1^2*Tr1[1])*Kronecker[i1,i2]/5 +
 4*mHd2*MatMul[Ye,Adj[Ye]][i1,i2]+4*MatMul[T[Ye],Adj[T[Ye]]][i1,i2] +
  2*MatMul[me2,Ye,Adj[Ye]][i1,i2]+4*MatMul[Ye, ml2, Adj[Ye]][i1,i2] +
  2*MatMul[Ye,Adj[Ye],me2][i1,i2]
```

The definition of the element Tr1[1] is saved in TraceAbbr[[1,1]]:

```
{Tr1[1], -mHd2 + mHu2 + trace[md2] + trace[me2] - trace[ml2] +
          trace[mq2] - 2*trace[mu2]}
```

c) **Number of generations as variable**: With

```
CalcRGEs[VariableGenerations -> {q}]
```

the number of generations of the left-quark superfield is handled as variable. Therefore, the one-loop β-function of the hypercharge couplings reads

```
 (63*g1^3)/10 + (g1^3*NumberGenerations[q])/10
```

d) **No matrix multiplication** Using matrix multiplication can be switched off by

```
CalcRGEs[NoMatrixMultiplication -> True]
```

The one-loop β-function for the electron Yukawa coupling is now written as

```
  sum[j2,1,3,sum[j1,1,3,conj[Yd[j2,j1]]*Yu[i1,j1]]*Yd[j2,i2]] +
2*sum[j2,1,3,sum[j1,1,3,conj[Yu[j1,j2]]*Yu[j1,i2]]*Yu[i1,j2]] +
  sum[j2,1,3,sum[j1,1,3,conj[Yu[j2,j1]]*Yu[i1,j1]]*Yu[j2,i2]] +
(3*sum[j2,1,3,sum[j1,1,3,conj[Yu[j1,j2]]*Yu[j1,j2]]]*Yu[i1,i2])/2 +
(3*sum[j2,1,3,sum[j1,1,3,conj[Yu[j2,j1]]*Yu[j2,j1]]]*Yu[i1,i2])/2 -
 (13*g1^2*Yu[i1,i2])/15-3*g2^2*Yu[i1,i2]-(16*g3^2*Yu[i1,i2])/3
```

e) Ignoring parameters at two-loop Using

```
CalcRGEs[IgnoreAt2Loop -> {T[L1],T[L2],L1,L2}]
```

in the MSSM with trilinear $\not{R}$pV would ignore the λ and λ' coupling as well as their soft-breaking equivalents in the calculation of the 2-loop RGEs.

Running RGEs in Mathematica `SARAH` writes the RGEs into a file which can be used for a numerical evaluation of the RGEs in `Mathematica`. This file is stored in the same directory and called `RunRGEs.m` and provides a function called `RunRGEs`. The syntax of this function is

```
RunRGEs[log(start), log(end), initialization, Options];
```

First, the logarithm of the scale where the running starts is given, the second entry is the logarithm of the scale where the running should end, and as third entry a list with the non-zero values of parameters at the starting scale is given. As option, it can be defined, if two-loop contributions should be included `TwoLoop -> True/False`. By default, `True` is used. For instance, a one-loop running of the gauge couplings from 1 TeV to 10^{16} GeV is performed by

```
<< "[\$SARAH-Directory]/Output/[\$MODEL]/RGEs/RunRGEs.m";
solution = RunRGEs[3,16,{g1->0.45, g2->0.63, g3->1.04}, TwoLoop->False];
```

Here, we saved the output of `RunRGEs` as variable, which can be used as follows

```
{g1[16], g2[16], g3[16]} /. solution[[1]];
Plot[g1[x]/. solution[[1]], {x,3,16}];
```

3.6 Loop Corrections

`SARAH` calculates the analytical expressions for the one-loop corrections to the tadpoles and the self energy of all particles. These calculations are performed in $\overline{\text{DR}}$-scheme and in the 't Hooft gauge. To command to start the calculation is

```
CalcLoopCorrections[Eigenstates,Options];
```

As usual, `Eigenstates` can be for instance in the case of the MSSM either `GaugeES` for the gauge eigenstates or `EWSB` for the eigenstates after EWSB. If the vertices for the given set of eigenstates were not calculated before, this is done before the calculation of the loop contributions begins. As option a list with fields can be given (`OnlyWith -> Particle1,Particle2,...`). Only corrections involving these fields as internal particles are included.

Conventions Using the conventions of Ref. [31], the results will contain the Passarino Veltman integrals listed in app. E.2. The involved couplings are abbreviated by

a) `Cp[p1,p2,p3]` and `Cp[p1,p2,p3,p4]` for non-chiral, three and four point interactions involving the particles `p1 - p4`.

b) `Cp[p1,p2,p3][PL]` and `Cp[p1,p2,p3][PR]` for chiral, three-point interactions involving the fields `p1 - p3`.

The self energies can be used for calculating the radiative corrections to masses and mass matrices, respectively. We have summarized the needed formulas for this purpose in app. E.2.3. For calculating the loop corrections to a mass matrix, it is convenient to use unrotated, external fields, while the fields in the loop are rotated. Therefore, SARAH adds to the symbols of the external particle in the interaction an `U` for 'unrotated', e.g. `Sd` → `USd`. The mixing matrix associated to this field in the vertex has to be replaced by the identity matrix when calculating the correction to the mass matrix.

Results The results for the loop corrections are saved in two different ways. First as list containing the different loop contribution for each particle. Every entry reads

```
{Particles, Vertices, Type, Charge Factor, Symmetry Factor}
```

and includes the following information

a) `Particles`: The particles in the loop.

b) `Vertices`: The needed Vertex for the correction is given.

c) `Charge Factor`: If several gauge charges of one particle are allowed in the loop, this factor will be unequal to one. In the case of the MSSM, only the a factor of 3 can appear because of the different colors.

d) `Symmetry Factor`: If the particles in the loop indistinguishable, the weight of the contribution is only half of the case of distinguishable particles. If two different charge flows are possible in the loop, the weight of the diagram is doubled, e.g. loop with charged Higgs and W-boson. The absolute value of the factor depends on the type of the diagram.

The results differ in general between the $\overline{\text{MS}}$ and $\overline{\text{DR}}$ renormalization scheme by a constant term which is reflected in the variable `rMS`. `rMS = 0` gives to the results in $\overline{\text{DR}}$ scheme and `rMS = 1` corresponds to $\overline{\text{MS}}$ scheme.

The information about the loop correction are also saved in the directory

```
../\SARAH/Output/"ModelName"/$EIGENSTATES/Loop
```

One Loop Tadpoles The complete results as sums of the different contributions are saved in the two dimensional array

```
Tadpoles1LoopSums[$EIGENSTATES]
```

The first column gives the name of the corresponding VEV, the second entry the one-loop correction.
A list of the different contributions, including symmetry and charge factors, is

```
Tadpoles1LoopList[$EIGENSTATES];
```

One Loop Self Energies The results are saved in the following two dimensional array

```
SelfEnergy1LoopSum[$EIGENSTATES]
```

The first column gives the name of the particle, the entry in the second column depends on the type of the field

a) Scalars: one-loop self energy $\Pi(p^2)$

b) Fermions: one-loop self energies for the different polarizations ($\Sigma^L(p^2)$,$\Sigma^R(p^2)$, $\Sigma^S(p^2)$)

c) Vector bosons: one-loop, transversal self energy $\Pi^T(p^2)$

Also a list with the different contributions does exist:

```
SelfEnergy1LoopList[$EIGENSTATES]
```

Examples

a) One-loop tadpoles The correction of the tadpoles due to a chargino loop is saved in

```
Tadpoles1LoopList[EWSB][[1]];
```

and reads

```
 {bar[Cha],Cp[Uhh[{gO1}],bar[Cha[{gI1}]],Cha[{gI1}]],FFS,1,1/2}
```

The meaning of the different entries is: (i) a chargino (`Cha`) is in the loop, (ii) the vertex with an external, unrotated Higgs (`Uhh`) with generation index `gO1` and two charginos with index `gI1` is needed, (iii) the generic type of the diagram is `FFS`, (iv) the charge factor is 1, (v) the diagram is weighted by a factor $\frac{1}{2}$ with respect to the generic expression (see app. E.2).
The corresponding term in `Tadpoles1LoopSum[EWSB]` is

```
4*sum[gI1,1,2, A0[Mass[bar[Cha[{gI1}]]]^2]*
   Cp[phid,bar[Cha[{gI1}]],Cha[{gI1}]]*Mass[Cha[{gI1}]]]
```

b) One-loop self-energies

(a) The correction to the down squark matrix due to a four point interaction with a pseudo scalar Higgs is saved in `SelfEnergy1LoopList[EWSB][[1, 12]]` and reads

```
{Ah,Cp[conj[USd[{gO1}]],USd[{gO2}],Ah[{gI1}],Ah[{gI1}]],SSSS,1,1/2}
```

This has the same meaning as the term

```
-sum[gI1,1,2,A0[Mass[Ah[{gI1}]]^2]*
   Cp[conj[USd[{gO1}]],USd[{gO2}],Ah[{gI1}],Ah[{gI1}]]]/2
```

in `SelfEnergy1LoopSum[EWSB]`.

(b) Corrections to the Z boson are saved in `SelfEnergy1LoopList[EWSB][[15]]`. An arbitrary entry looks like

```
{bar[Fd], Fd, Cp[VZ, bar[Fd[{gI1}]], Fd[{gI2}]], FFV, 3, 1/2}
```

and corresponds to

```
(3*sum[gI1, 1, 3, sum[gI2, 1, 3,
     H0[p^2, Mass[bar[Fd[{gI1}]]]^2, Mass[Fd[{gI2}]]^2]*
 (conj[Cp[VZ,bar[Fd[{gI1}]],Fd[{gI2}]][PL]]*
     Cp[VZ,bar[Fd[{gI1}]],Fd[{gI2}]][PL] +
  conj[Cp[VZ,bar[Fd[{gI1}]],Fd[{gI2}]][PR]]*
     Cp[VZ,bar[Fd[{gI1}]],Fd[{gI2}]][PR]) +
 2*B0[p^2,Mass[bar[Fd[{gI1}]]]^2,Mass[Fd[{gI2}]]^2]*
     Mass[bar[Fd[{gI1}]]]*Mass[Fd[{gI2}]]*
  Re[Cp[VZ,bar[Fd[{gI1}]],Fd[{gI2}]][PL]*
     Cp[VZ,bar[Fd[{gI1}]],Fd[{gI2}]][PR])]])/2
```

in SelfEnergy1LoopListSum[EWSB].

3.7 Output for LaTeXand diagram calculators

With

ModelOutput[Eigenstates,Options]

LaTeX-files and model files for FeynArts and CompHep/CalcHep can be generated. Here, Eigenstates specifies the eigenstates which should be used, e.g. GaugeES or EWSB. The options are the following

a) WriteTeX, values: True or False, Default: False
If a LaTeXfile containing all information about the model should be written.

b) WriteFeynArts, values: True or False, Default: False
If a model file for FeynArts should be written.

c) WriteCHep, values: True or False, Default: False
If a model file for CompHep/CalcHep should be written.

d) WriteWHIZARD, values: True or False, Default: False
If a model file for WHIZARD should be written.

e) WritUFO, values: True or False, Default: False
If model files in the UFO format should be written.

f) effectiveOperators, Values: True or False, Default: False
If also higher dimensional operators should be calculated. By default, this concerns only four point interactions.

g) SixParticleInteractions, Values: True or False, Default: False
If also the six-point interactions should be calculated.

h) FeynmanDiagrams, Values: True or False, Default: True
If Feynman diagrams for each vertex should be drawn in the LaTeXfile.

i) ReadLists, Values: True or False, Default: False
If the results of former calculations should be used to save time.

j) IncludeRGEs, Values: True or False, Default: True
If the RGEs should be calculated.

k) `TwoLoopRGEs`, Values: `True` or `False`, Default: `True`
If the two loop RGEs should be calculated. (`IncludeRGEs` must be set to `True`)

l) `IncludeLoopCorrections`, Values: `True` or `False`, Default: `True`
If the one-loop corrections to the self-energy and the tadpoles should be calculated.

The generated output will be saved in

```
../\SARAH/Output/$MODEL/$EIGENSTATES/TeX/"
../\SARAH/Output/$MODEL/$EIGENSTATES/\FeynArts/"
../\SARAH/Output/$MODEL/$EIGENSTATES/CHep/"
../\SARAH/Output/$MODEL/$EIGENSTATES/UFO/"
../\SARAH/Output/$MODEL/$EIGENSTATES/WHIZARD/"
```

Chapter 4

The model files

All information of the different models are saved in three different files which have to be in one directory

`.../\SARAH/Models/"ModelName"/`

> The directory name is equal to the name of the model and must contain a model file with the same name!

The three files are: one model file with the same name as the model directory (`ModelName.m`), a file containing additional information about the particles of the model (`particles.m`) and a file containing additional information about the parameters of the model (`parameters.m`). Only the first file is really necessary for calculating the Lagrangian and to get a first impression of a model. However, for defining properties of parameters and particles and for producing an appropriate output the other two files are needed.

4.1 The model file

The model file is the heart of `SARAH`: the complete model is specified by the model file. Thus, we will explain the general structure of the model file here, and have a look at the different functions and its physical meaning in detail in the next chapter.

4.1.1 Description

The model file contains the following parts: First the gauge structure and the particle content are given, and the matter interactions are defined by the superpotential. These are general information needed for all eigenstates of the model and must always be present.
This part is followed by the definition of the names for all eigenstates (`NameOfStates`). For these eigenstates can afterwards several properties be defined using the corresponding `DEFINITION` statements: decomposition of scalars in scalar, pseudo scalar and VEV (`DEFINITION[$EIGENSTATES][VEVs]`), rotations in the matter (`DEFINITION[$EIGENSTATES][MatterSector]`) and gauge sector (`DEFINITION[$EIGENSTATES][GaugeSector`
New couplings can be added and existing couplings can be changed by hand (`DEFINITION[$EIGENSTATES][Additional]`) and particles can be decomposed in the different flavors (`DEFINITION[$EIGENSTATES][Flavors]`)
Afterwards, the particles are defined, which should be integrated out or deleted. At the end, the Dirac spinors have to be built out of Weyl spinors, a spectrum file can be defined and a choice for automatically output can be made.

4.1.2 Schematic Structure

The model file is structured as follows

General information

a) Global symmetrie are defined in an array `Global`, see sec. 5.1

```
Global[[1]] = {...
```

b) Gauge structure of the model given by the vector superfields, see sec. 5.2.1

```
Gauge[[1]] = {...
```

c) Matter content given by the chiral superfields see sec. 5.2.2

```
SuperFields[[1]] = { ...
```

or fermion and scalar fields for non-SUSY models, see sec. 5.2.3

```
FermionFields[[1]] = { ...
ScalarFields[[1]] = { ...
```

d) Superpotential, see sec. 5.3

```
SuperPotential { ...
```

or the Lagrangian, see sec. 5.5

```
DEFINITION[$EIGENSTATES][LagrangianInput] = { ...
```

Eigenstates

Names for the different eigenstates, see sec. 5.6

```
NameOfStates = { ...
```

Definition of properties for the different eigenstates

a) Vacuum expectation values, see sec. 5.6.1.1

```
DEFINITION[$EIGENSTATES][VEVs]= { ...
```

b) Rotations in the gauge sector, see sec. 5.6.3

```
DEFINITION[$EIGENSTATES][GaugeSector]= { ...
```

c) Expansion of flavors, see sec. 5.6.2

```
DEFINITION[$EIGENSTATES][Flavors]= { ...
```

d) Rotations in the matter sector, see sec. 5.6.4

```
DEFINITION[$EIGENSTATES][MatterSector]= { ...
```

e) Gauge fixing terms: since version 3.1 no longer needed as input, see sec. B.1

f) Additional couplings or redefinition of existing couplings, see sec. 5.6.6.2

```
DEFINITION[$EIGENSTATES][Additional]= { ...
```

g) Definition of Dirac spinors, see sec. 5.8

```
DEFINITION[$EIGENSTATES][DiracSpinors]= { ...
```

Additional, general Information

a) Integrating out or deleting particles, see sec. 5.7

```
IntegrateOut[[1]] = { ...
DeleteParticles[[1]] = { ...
```

b) Assigning a spectrum file, see sec. 14.1

```
SpectrumFile = ...
```

4.2 Parameter file

4.2.1 General

The information of the parameter file are needed for some in- and output routines. Also, it is possible to define simplifying assumptions for the parameters. The parameter file consists of a list called `ParameterDefinition`. This is an array with two columns: the first column gives the name of the parameter, the second column defines the properties of the parameter. These properties can be:

a) `Description`: Value: a string
A string for defining the parameter, see sec. 4.4

b) `Real`, Value: `True` or `False`, Default: `False`
Defines, if a parameter is always taken to be real.

c) `Form`, Value: `None`, `Diagonal` or `Scalar`, Default: `None`

d) `LaTeX`, Value: `None` or LaTeX name
The name of the parameter used in the LaTeX output. Standard LaTeX language should be used (\ has to be replaced by \\).

e) `Dependence`, Value: `None` or a function
The parameter is always replaced by this definition, see sec. 4.2.3

f) `DependenceOptional`, Value: `None` or a function
It can be chosen during the work, if the parameter is replaced by the made definition, see sec. 4.2.3

g) `DependenceNum`, Value: `None` or a function
This definition is used in numerical calculations, see sec. 4.2.3

h) `MatrixProduct`, Value: `None` or a list of two matrices
The parameter is defined as a product of two matrices, see sec. 4.2.3

i) `LesHouches`, Value: `None` or the position in LesHouches input file
The numerical value of the parameter can be given by a LesHouches file, see sec. 4.2.3

j) `Value`, Value: `None` or a number
A numerical value for the parameter can be chosen, see sec. 4.2.3

4.2.2 Simplifying assumptions

`SARAH` normally handles parameters in the most general way: most parameters assumed to be complex and all tensors can have off diagonal values. This can be changed by certain statements in the parameter file. First, it is possible to define a parameter as real by setting

```
Real -> True
```

The gauge couplings are by default assumed to be real.
The degrees of freedom for a tensor valued parameter T can be reduced by using the `Form` statement with the following options

a) `Diagonal`: only diagonal entries are assumed to be unequal from zero: $T_{ij} \to \delta_{ij} T_{ij}$

b) `Scalar` the tensor is replaced by a scalar: $T_{ij} \to T$

Examples

a) CP and Flavor conserving Yukawa matrices are defined by

```
{Yu,  { Real -> True,
        Form -> Diagonal}}
```

4.2.3 Defining values and dependences for parameters

4.2.3.1 Dependences

There are different possibilities to define dependences between parameters by using the `Dependence` statements. The difference between the three dependence statements is the time at which the relations are used

a) `Dependence`: The relations are always used. The corresponding substitutions are saved in `subAlways`.

b) `DependenceOptional`: The relations are only used if the option `UseDependence` is set to `True` as option for specific commands, e.g. when calculating vertices. The substitutions are saved in `subDependences`.

c) `DependenceNum`: The dependences are only used when a numerical value for the parameter is calculated. The substitutions are saved in `subNum`.

The indices of vectors or tensors are implicitly assumed to be `index1`, `index2`, ... This in combination with `sum[index,start,final]` can be used in the following way

```
{X, {Dependences -> sum[n1,1,3] sum[n2,1,3] A[index1,n1] Y[n1,n2] B[n2,index2]}}
```

and is interpreted as

$$X_{i_1 i_2} \to \sum_{n_1=1}^{3} \sum_{n_2=1}^{3} A_{i_1 n_1} Y_{n_1 n_2} B_{n_2 i_2} \tag{4.1}$$

Matrix valued parameters can also defined as matrix product of other matrices by using `MatrixProduct`. The argument must be a list consisting of two matrices of same dimension:

```
{X, {MatrixProduct-> {A,B} }}
```

Using this definition, every appearance of a matrix product of $A^\dagger B$ is replaced by X and $B^\dagger A$ by $X^\dagger$ and can be used, for instance, to define the CKM matrix.

Examples

a) One dependence, which might be used always, is the parametrization of a mixing matrix matrix by a mixing angle: the mixing of the charged Higgs (`ZP`) in the MSSM can be parametrized by the mixing angle β. This is defined in `SARAH` via

```
{ZP,  { Dependence -> {{-Cos[\[Beta]],Sin[\[Beta]]},
                        {Sin[\[Beta]],Cos[\[Beta]]}}
```

b) The relation between the gauge couplings g_1 and g_2 and the electric charge e is an example for an optional dependence. This relation is defined by

```
{g1,  {DependenceOptional -> e/Cos[ThetaW]}}
```

Now, the result for vertices can be expressed in terms of the electric charge by using

```
Vertex[List of Particles, UseDependences -> True]
```

c) A relation, which might only be used for numerical calculations, is the relation between the Weinberg angle and the gauge couplings:

```
{ThetaW, {DependenceNum ->  ArcCos[g2/Sqrt[g1^2+g2^2]]}
```

d) The CKM matrix is defined as the product of two rotation matrices:

```
{CKM, {MatrixProduct -> {Vd,Vu} }}
```

4.2.3.2 Numerical values

If the considered parameter does not depend on other parameters, there are two ways to assign a numerical value to this parameter:

a) `Value`: Adds directly a numerical value to the parameter definition

b) `LesHouches`: Defines the position of the numerical value for the parameter in a LesHouches spectrum file (see also [21]). The statement has to has the following form:

(a) If the dimension of the parameters is known by SARAH, it is sufficient to give just the name of the block, e.g.

```
{Yu, { LesHouches -> Yu } };
```

(b) If the block appears several times in the LesHouches file, it is possible to give the number of appearance as optional argument:

```
{g3,  {LesHouches -> {gauge,3}}};
```

This reads the first block which are normally the GUT-scale values. However,

```
{g3,  {LesHouches -> {{gauge,3},2}}};
```

reads the second block which are the values at the SUSY-scale.

Example

a) The CKM matrix can be defined in the Wolfenstein parametrization as

```
{{Description ->"CKM Matrix",
 LaTeX -> "V^{CKM}",
             MatrixProduct -> {Vd,Vu},
             Dependence ->  None,
             DependenceNum -> {{1-1/2*lWolf^2,lWolf,aWolf*lWolf^3*Sqrt[rWolf^2+nWolf^2]},
                               {-lWolf,1-1/2*lWolf^2, aWolf*lWolf^2},
                               {aWolf*lWolf^3*Sqrt[(1-rWolf)^2 +nWolf^2],-aWolf*lWolf^2,1}},
             LesHouches -> VCKM,
             DependenceSPheno -> Matmul[Transpose[conj[Vd]],Vu],
             OutputName-> VCKM      }},
```

b) The Wolfenstein parameters are real and the experimental values are known

```
{{Description->"Wolfenstein Parameter eta",
             Value -> 0.341,
             Real -> True,
             OutputName-> nWolf,
             LesHouches -> {WOLFENSTEIN,4}              }},
```

4.3 Particles File

The particle file contains information about all fields of the model. This information is usually needed needed for an appropriate output.

a) `Description`: A string for defining the particle

b) `PDG`: The PDG number. Needed, if the written model files should be readable by event generators or if masses are given by a LesHouches file.

c) `PDG.IX`: Defines a nine-digit number of a particle supporting the proposal Ref. [32, 33]. By default, the entries of `PDG` are used. To switch to the new scheme, either at the beginning of a SARAH session or in the model files, the following statement has to be added:

```
UsePDGIX = True;
```

d) `ElectricCharge`: Defines in the electric charge of a particle in units of e. This information is exported to the `CalcHep` and UFO model files.

e) `Width`: The width of the particle. If not defined, 0 is used.

f) `Mass`: The options of defining the mass of a particle are:

(a) Numerical Value: A numerical value for the mass of the particle is given.

(b) `Automatic`: `SARAH` derives the tree level expression for the mass from the Lagrangian and calculates the value by using the values of the other parameters.

(c) `LesHouches:` `SARAH` reads the mass from a LesHouches file.

g) `LaTeX`: The name of the particle in LaTeX files in standard LaTeX language. If not defined, the Mathematica `InputForm` of the particle name is used. It is also possible to give a list with two entries corresponding to the name of the particle and anti-particle.

h) `FeynArtsNr`: The number of the particle in a `FeynArts` model file. If not defined, the number will be generated automatically: the smallest free number is taken.

i) `Output`: A short form the particle name consisting of two letter with no no-standard signs. Needed, to make sure that all programs outside `SARAH` can read the name correctly. If not defined, the Mathematica `InputForm` is used and potentially truncated.

j) `LHPC`: Defines the column and color used for the particle in the steering file for the `LHPC Spectrum Plotter`. All colors defined in `gnuplot` can be used.

This information must be given for all eigenstates in arrays named

```
ParticleDefinition[$EIGENSTATES]
```

Only for the LaTeXoutput also the names of the Weyl spinors and intermediate states (like scalar and pseudo scalar components of Higgs) should be given to improve the look and readability of the produced pdf file.

Example For the eigenstates after EWSB, an entry might look like

```
ParticleDefinitions[EWSB} = {
   ...
   {Sd ,  {  Description -> "Down Squark",
             PDG ->  {1000001,2000001,1000003,2000003,1000005,2000005},
             PDG.IX -> {-200890201,-200890202,-200890203,-200890204,-200890205,-200890206}
             Mass -> Automatic,
             FeynArtsNr -> 14,
             LaTeX -> "\\tilde{d}",,
             ElectricCharge -> -1/3,
             LHPC -> {7, "cyan"},
             OutputName -> "sd" }},
   ... }
```

4.4 Global definitions

It is also possible to define global properties for parameters or particles which are present in more than one model file. These properties are afterwards used for all models. The global definitions are saved in the files `particles.m` and `parametes.m` directly in the main model directory. For each parameter or particle, an entry like

```
{{        Descriptions -> "Down Squark",
          PDG ->  {1000002,2000002,1000004,2000004,1000006,2000006},
          Width -> Automatic,
          Mass -> Automatic,
          FeynArtsNr -> 13,
          LaTeX -> "\\tilde{u}",
          OutputName -> "um" }},
```

can be added. In particular, the entry `Description` is important. This should be an unique identifier for each particle or parameter. This identifier can later on be used in the different files of the different models, e.g.

```
{Su ,  {  Descriptions -> "Down Squark"}},
```

Of course, it is also possible to overwrite some of the global definitions by defining them locally, too. For instance, to use `u` instead of `um` as output name in a specific model, the entry should be changed to

```
{Su ,  {  Descriptions -> "Down Squark",
          OutputName -> "u" }},
```

in the corresponding particle file of the model.

Chapter 5

Definition of models

5.1 Global symmetries

Z_N and $U(1)$ global symmetries can be defined. For this purpose, a new array `Global` has been introduced:

```
Global[[1]] = {Z[2], RParity};
Global[[2]] = {U[1], PecceiQuinn};
```

First, the kind of the symmetry is defined and afterwards a name is given to the symmetry. In principle, up to 99 different global symmetries can be defined for one model. By convention, Z_N symmetries are always taken to be a multiplicative symmetry. I.e., the charges Q of additive symmetries have to be given as $\exp(2i\pi Q)$.

5.2 Particle content

5.2.1 Vector superfields

The vector superfields are defined by the array `Gauge`. An entry reads

```
Gauge[[i]]={Superfield Name, Dimension, Name of Gauge Group, Coupling, Expand};
```

The different parts have the following meaning:

a) `Superfield name`:
This is the name for the vector superfield and also the basis of the names for vector bosons and gauginos as explained in sec. 3.1

b) `Dimension`:
This defines the dimension of the $SU(N)$ gauge group: `U[1]` for an Abelian gauge group or `SU[N]` with integer N for a non-Abelian gauge group.

c) `Name of Gauge Group`:
This is the name of the gauge group, e.g. hypercharge, color or left. This choice is import because all matter particles charged under a non-Abelian gauge group carry an corresponding index. The name of the index consists of the first three letter of the name plus a number. Hence, it must be taken care that the first three letters of different gauge group names are not identical. Also the name for the indices in the adjoint representation are derived from this entry.

d) `Coupling`: The name of the coupling constant, e.g. `g1`

e) `Expand`: Values can be `True` or `False`. If it is set to `True`, all sums over the corresponding indices are evaluated during the calculation of the Lagrangian. This is normally done non-Abelian gauge groups which get broken like the $SU(2)_L$ in the MSSM.

`SARAH` adds for every vector superfield a soft-breaking gaugino mass

```
Mass<>"Superfield Name"
```

Example: Standard model color group

```
Gauge[[3]] = {G,  SU[3],  color,  g3, False};
```

The consequence of this entry is

a) Gluon and gluino are named `VG` respectively `fG`

b) The $SU(3)$ generators, the Gell-Mann matrices, are used

c) The color index is abbreviated `colX` (for `X` = 1,2, ...)

d) The adjoint color index is abbreviated `acolX` (for `X` = 1,2, ...)

e) The strong coupling constant is named `g3`

f) The sums over the color indices are not evaluated

Models with several $U(1)$ gauge groups The general, gauge invariant Lagrangian for a theory including two $U(1)$ gauge groups reads

$$L = -\frac{1}{4}\hat{F}^{a,\mu\nu}\hat{F}^{a}_{\mu\nu} - \frac{1}{4}\hat{F}^{b,\mu\nu}\hat{F}^{b}_{\mu\nu} - \frac{\sin\chi}{2}\hat{F}^{a,\mu\nu}\hat{F}^{b}_{\mu\nu}|\hat{D}_\mu\phi|^2 + i\bar{\Psi}\gamma^\mu\hat{D}_\mu\Psi \tag{5.1}$$

with the covariant derivative

$$\hat{D} = \partial_\mu - ig^0_a Q_a \hat{A}^a_\mu - ig^0_b Q_b \hat{A}^b_\mu. \tag{5.2}$$

It is possible to perform a field rotation to remove the mixing term between the field strength tensors. In that case, the covariant derivate changes to

$$D_\mu = \partial_\mu - i(\bar{g}_a Q_a + \bar{g}_{ba})\bar{A}^a_\mu - i(\bar{g}_{ab}Q_a + \bar{g}_b Q_b)\bar{A}^b_\mu. \tag{5.3}$$

`SARAH` uses eq. (5.3) to write the Lagrangian for models containing several Abelian gauge groups. For that purposes, it generates new gauge couplings

```
g<>A<>B
```

for the off-diagonal couplings. Here `gA` and `gB` are the names for the diagonal gauge couplings defined in `Gauge`, i.e the first letter is always dropped. In addition, the gaugino mass terms are written as

$$\sum_i\sum_j M_{ij}\lambda_i\lambda_j + \text{h.c.}. \tag{5.4}$$

The sum i and j runs over all Abelian gauge groups. The names for the off-diagonal gaugino mass are

```
Mass<>A<>B
```

Here, `A` and `B` are the names of the vector superfields defined in `Gauge`.
Note, to include these mixing effects in the RGEs, there have been done some changes with respect to [22]. See section E.1.3.

Example In the case of a gauge sector containing

```
Gauge[[1]] = {R,  U[1],  right,  gR, False};
Gauge[[2]] = {BL,  U[1],  bminusl,  gBL, False};
```

the off-diagonal gauge couplings are called

```
gRBL
gBLR
```

and the off-diagonal gaugino masses are

```
MassRBL
MassBLR
```

5.2.2 Chiral Superfields

Chiral superfields are defined as follows:

```
SuperField[[i]] = {SuperField Name,  Generations, Components, Transformation Gauge 1,
Transformation Gauge 2..., Transformation Global 1, Transformation Global 2 };
```

a)

b) `Superfield Name`: The name for the superfield

c) `Generations`: The number of generations

d) `Components`: The basis of the name for the components. Two cases are possible:

   (a) The field transforms only trivially under the gauge groups with expanded indices. In this case, the entry is one dimensional.

   (b) The field transforms non-trivially under gauge groups with expanded indices. In this case, the entry is a vector or higher dimensional tensor fitting to the dimension of the field. Note, representations larger than the fundamental one are written as tensor products

e) `Transformation Gauge X`: Transformation under the different gauge groups defined before. For $U(1)$ this is the charge, for non-Abelian gauge groups the dimensions is given as integer respectively negative integer. The dimension D of an irreducible representation is not necessarily unique. Therefore, to make sure, `SARAH` uses the demanded representation, also the corresponding Dynkin labels have to be added.

f) `Transformation Global X`: Transformation under the different global symmetries. If only one quantum number is given per superfield per global symmetry, this number is used for the superfield itself but also for the scalar and fermionic component. To define a R symmetry, a list with three entries has to be given. For chiral superfields, the first entry is the charge for the superfield, the second for the scalar component, the third for the fermionic component. For vector superfield, the second entries refers to the gaugino, the third to the gauge boson.

`SARAH` adds automatically for all chiral superfields soft-breaking squared masses named

```
m <> "Name of Superfield" <> 2
```

In addition, there are mixed soft-breaking parameters added which are allowed by gauge invariance and R-parity. The last check can be disabled by setting

```
RParityConservation = False;
```

in the model file.

Examples

a) Fields with expanded indices The definition of the left quark superfield in the MSSM is

```
SuperField[[1]] = {q, 3, {uL,  dL},   1/6, 2, 3, {-1,-1,1}};
```

The consequence of this definition is

(a) Left up-squarks and quarks are called `SuL` / `FuL`
(b) Left down-squarks and quarks are called `SdL` / `FdL`
(c) There are three generations
(d) The superfield is named `q`
(e) The soft-breaking mass is named `mq2`
(f) The hypercharge is $\frac{1}{6}$
(g) The superfield transforms as **2** under $SU(2)$
(h) The superfield transforms as **3** under $SU(3)$
(i) The superfield and scalar have R-parity -1, the fermion +1.

b) Fields with no expanded indices The right down-quark superfield is defined in the MSSM as

```
SuperField[[3]] = {d, 3, {conj[dR]},  1/3, 1, -3,  {-1,-1,1}};
```

The meaning is

(a) The right squarks and quarks are called `SdR` and `FdR`
(b) There are three generations
(c) The Superfield name is `d`
(d) The soft-breaking mass is named `md2`
(e) The hypercharge is $\frac{1}{3}$
(f) It does not transform under $SU(2)$
(g) It does transform as $\mathbf{\overline{3}}$ under $SU(3)$
(h) The superfield and scalar have R-parity -1, the fermion +1.

c) Specification of representation Since the **10** under $SU(5)$ is not unique, it is necessary to add the appropriate Dynkin labels, i.e.

```
SuperField[[1]] = {Ten, 1, t, {10,{0,1,0,0}},...};
```

or

```
SuperField[[1]] = {Ten, 1, t, {10,{0,0,1,0}},...};
```

d) Mixed soft-breaking terms In models which contain fields with the same quantum numbers under gauge and global symmetries mixed soft-breaking terms are added. For instance, in models with heavy squarks

```
SuperField[[3]] = {d, 3, {conj[dR]},  1/3, 1, -3};
...
SuperField[[10]] = {DH, 3, {conj[dRH]},  1/3, 1, -3};
```

the term of the form

```
mdDH (conj[SdR] SdRH + SdR conj[SdRH])
```

is automatically added. For the MSSM with R-parity violation, also the term

```
mlHd (conj[Sl] SHd + Sl conj[SHd])
```

is created, when the flag `RParityConservation = False` is used in the model file.

5.2.3 Scalars and Fermions

The syntax for scalar and fermion fields in the case of non-SUSY models looks exactly the same:

```
FermionFields[[1]] = {q, 3, {uL, qL}, 1/6, 2, 3};
...
ScalarFields[[1]] =  {H, 1, {H0, Hm}, 1/2, 2, 1};
```

Here, we have not assumed any global symmetry. Note, as in all previous versions the fermionic components of superfields start with `F` and the scalars with `S`, i.e. the quark doublets stemming from the superfield definition above are called `FuL, FdL` and the squarks are called `SuL, SdL`. For fermion and scalar fields defined separately, no renaming takes place but the fields are used exactly as they are defined. By default all scalars are taken to be complex. To define them as real the name of the field has to be added to the list `RealScalars`. For instance, a real singlet is added to the model by

```
ScalarFields[[2]] = {S, 1, s,  0, 1, 1};
RealScalars = {S};
```

It is not possible to use `SuperFields` and `FermionFields` or `ScalarFields` at the same time. If the user wants to define scalars or fermions, all superfields have to be written as components.

5.3 Superpotential

The definition of the superpotential is straight forward

```
SuperPotential = {{{Coefficient,Parameter,(Contraction)},
                      {Particle 1, Particle 2, ...} }, ...}
```

Each term of the superpotential is defined due to two list:

a) The first list is two dimensional and defines a numerical coefficient (`Coefficient`) and the name of the parameter (`Parameter`)

b) The second list gives the involved superfields

c) The contraction of the indices can be given optionally.

Up to four superfields are possible, i.e. also effective operators can be considered.
Since version 4 also a short form exists, where the entire superpotential can be given as sum of terms:

```
SuperPotential = Coefficient*Parameter Particl1.Particle2 + ...
```

Contraction of indices The indices of the involved particles are automatically contracted by SARAH. Sometimes, there are more possibilities to contract all indices. Therefore, it is also possible to fix the contraction of each term. The contraction used by SARAH can be seen by

```
ShowSuperpotentialContractions;
```

Properties of couplings and soft-breaking terms If the particles involved in the different interactions have more than one generation, the couplings are in general complex tensors carrying up to three generation indices. Assumptions like diagonality or no CP violation can added by using the parameter file (see sec. 4.2).
The corresponding soft-breaking term to each superfield coupling is automatically added to the Lagrangian. The soft-breaking couplings carry the same indices as the superpotential coupling. They are named as

a) Quartic terms: `Q["Name of Coupling"]`

b) Trilinear terms: `T["Name of Coupling"]`

c) Bilinear terms: `B["Name of Coupling"]`

d) Linear terms: `L["Name of Coupling"]`

Simplifying assumptions for the soft-breaking terms can be made independently of the assumptions for the superpotential parameters in the parameter file.

Example The term involving the up Yukawa coupling is

$$Y^u_{n_1,n_2}\hat{q}^i_{\alpha,n_1}\epsilon^{ij}\hat{H}^j_u\hat{u}_{\beta,n_2}\delta_{\alpha,\beta} \tag{5.5}$$

This can be defined in SARAH with

```
 {{1, Yu},{q,Hu,u}}
```

The explicit contraction of the indices would read as

```
 {{1, Yu,Delta[col1,col3] Eps[lef1,lef2] },{q,Hu,u}}
```

while the short hand input form is just

```
Yu q.Hu.u
```

There is a soft-breaking term automatically added to Lagrangian, which has the same meaning as

$$T(Y^u)_{n_1,n_2}\tilde{q}^i_{\alpha,n_1}\epsilon^{ij}H^j_u\tilde{u}^*_{\beta,n_2}\delta_{\alpha,\beta} \tag{5.6}$$

5.4 Excluding terms or adding Dirac gaugino mass terms

As shown in app. B.1 SARAH uses the information given so far about the vector and chiral superfields to calculate the entire Lagrangian for the gauge eigenstates. This includes in general the kinetic terms, the matter interactions from the superpotential, D- and F-terms as well as soft-breaking terms. However, it is also possible to exclude parts by the following statements:

- `AddTterms = True/False;`, default: `True`, includes/excludes trilinear softbreaking couplings
- `AddBterms = True/False;`, default: `True`, includes/excludes bilinear softbreaking couplings
- `AddLterms = True/False;`, default: `True`, includes/excludes linear softbreaking couplings
- `AddSoftScalarMasses = True/False;`, default: `True`, includes/excludes soft-breaking scalar masses
- `AddSoftGauginoMasses = True/False;`, default: `True`, includes/excludes Majorana masses for gauginos
- `AddSoftTerms = True/False;`, default: `True`, includes/excludes all soft-breaking terms
- `AddDterms = True/False;`, default: `True`, includes/excludes all D-terms
- `AddFterms = True/False;`, default: `True`, includes/excludes all F-terms

On the other hand, Dirac mass terms for gauginos are not written automatically if chiral superfields in the adjoint representation are present. The reason for this is just that models without these terms are still more common, e.g. the NMSSM is usually studied without a bino-singlino Dirac mass term. Therefore, to include Dirac mass terms for gauginos, one has to add explicitly

```
AddDiracGauginos = True;
```

to the model file. In this case `SARAH` writes down all possible mass terms between chiral and vector superfields and the corresponding D-terms for the model. In this context, it uses `MD<>v<>c` as name for the new mass parameters where `v` is the name of the vector and `c` the name of the chiral superfield. If several fields in the adjoint representation of one gauge group are present, `SARAH` will generate the corresponding terms for all of them. To remove some of them, the parameters can be put to zero in the parameters file of the model definition. Furthermore, if several Abelian gauge groups are present, the impact of kinetic mixing is also respected.

Example One can add to the particle content of the MSSM, fields in the adjoint representations of the different gauge groups:

```
SuperFields[[8]] = {S, 1, s, 0, 1, 1};
SuperFields[[9]] = {T, 1, {{T0/Sqrt[2],Tp},{Tm, -T0/Sqrt[2]}}, 0, 3, 1};
SuperFields[[10]] = {oc, 1, Oc, 0, 1, 8};
```

Here, the triplet superfield is defined as

$$\hat{T} = \begin{pmatrix} \hat{T}^0/\sqrt{2} & \hat{T}^+ \\ \hat{T}^- & -\hat{T}^0/\sqrt{2} \end{pmatrix} \tag{5.7}$$

and to include the Dirac mass terms, use

```
AddDiracGauginos = True;
```

This leads to the corresponding mass term and the D-terms. `SARAH` names the new parameters using the corresponding superfield names as `MDBS` (bino-singlet mass term), `MDWBT` (wino-triplet mass term), `MDGoc` (gluino-octet mass term).

5.5 Lagrangian

For non-SUSY models the user can define terms in the Lagrangian for the different eigenstates

```
LagNoHC = -Mu2 conj[H].H + 1/2 \[Lambda] conj[H].H.conj[H].H;
LagHC =  Yd conj[H].d.q + Ye conj[H].e.l + Yu H.u.q;

DEFINITION[GaugeES][LagrangianInput]= {
         {LagHC,   {AddHC->True}},
         {LagNoHC,{AddHC->False}}
};
```

Here, we defined all interactions in terms of gauge eigenstates (GaugeES) and all rotations to the mass eigenstates are performed by SARAH. In addition, there exists the option that SARAH adds automatically the hermitian conjugated for the given part of the Lagrangian (AddHC->True).

5.6 Properties of different eigenstates

For defining the properties of the different sets of eigenstates, the DEFINITION statement is used:

```
DEFINITION[$EIGENSTATES]["Property"]= {...};
```

The possible properties are VEVs, GaugeSector, MatterSector, Phase, Flavors and Additional. Note, Additional for SUSY models has the same conventions as LagrangianInput for non-SUSY models. But, before defining this properties, the names for all eigenstates must be fixed in the correct order. This is done due to

```
NameOfStates={List of Name};
```

This list can, in principle, be arbitrary long. Common entries are, e.g.

```
NameOfStates={GaugeES, EWSB}
NameOfStates={GaugeES, SCKM, EWSB}
```

5.6.1 Vacuum expectation values

5.6.1.1 Introduction

The particles responsible for breaking a gauge symmetry receive a VEV. After the symmetry breaking, these particles are parametrized by a scalar ϕ and a pseudo scalar σ part and the VEV v:

$$S = \frac{1}{\sqrt{2}} \left(\phi_S + i\sigma_S + v_S\right) \tag{5.8}$$

5.6.1.2 Implementation in SARAH

This is in SARAH done by

```
DEFINITION[$EIGENSTATES][VEVs] =
{Particle Name, {{VEV, Coefficient 1},
      {Pseudoscalar, Coefficient 2},{Scalar, Coefficient 3},({Phase})};
```

a) Name: The name of the particle receiving a VEV

b) `VEV`: Name of the VEV

c) `Scalar`: Name of the scalar component

d) `Pseudoscalar`: Name of the pseudo scalar component

e) `Coefficient 1,2,3`: The different (numerical) coefficients.

f) `Phase`: Optional phase

All indices carried by the particle receiving the VEV are automatically added to the scalar and pseudo scalar part. The scalar, pseudo scalar and the VEV are handled as real parameters in SARAH. The phase is only an optional argument and can be skipped for Higgs sectors without CP violation.

Example In the MSSM, the Higgs H_d^0 and H_u^0 get VEVs v_d and v_u:

$$H_u^0 = \frac{1}{\sqrt{2}}\left(v_u + i\sigma_u + \phi_u\right) , \qquad H_d^0 = \frac{1}{\sqrt{2}}\left(v_d + i\sigma_d + \phi_d\right) \tag{5.9}$$

This is done in SARAH by using

```
DEFINITION[EWSB][VEVs]=
{{SHd0, {vd, 1/Sqrt[2]}, {sigmad, I/Sqrt[2]},{phid,1/Sqrt[2]}},
 {SHu0, {vu, 1/Sqrt[2]}, {sigmau, I/Sqrt[2]},{phiu,1/Sqrt[2]}},
};
```

To add a relative phase, use

```
DEFINITION[EWSB][VEVs]=
{{SHd0, {vd, 1/Sqrt[2]}, {sigmad, I/Sqrt[2]},{phid,1/Sqrt[2]}},
 {SHu0, {vu, 1/Sqrt[2]}, {sigmau, I/Sqrt[2]},{phiu,1/Sqrt[2]},{eta}},
};
```

This is interpreted as

$$H_u^0 = \frac{e^{i\eta}}{\sqrt{2}}\left(v_u + i\sigma_u + \phi_u\right) , \qquad H_d^0 = \frac{1}{\sqrt{2}}\left(v_d + i\sigma_d + \phi_d\right) \tag{5.10}$$

5.6.1.3 Aligned VEVs

The standard definition of a model with broken electric charge due to VEVs charged slepton VEVs looks like

```
DEFINITION[EWSB][VEVs]=
   {...
    {SeL,  {vL,  1/Sqrt[2]},  {sigmaL,I/Sqrt[2]},{phiL,1/Sqrt[2]}},
    {SeR,  {vR,  1/Sqrt[2]},  {sigmaR,I/Sqrt[2]},{phiR,1/Sqrt[2]}},
   };
```

With this definition, all three generations of left and right sleptons would get a VEV. However, usually one is only interested in the case that staus receive VEVs. This can now be defined by

```
DEFINITION[EWSB][VEVs]=
  {..,
   {SeL, {vL[3], 1/Sqrt[2]}, {sigmaL,I/Sqrt[2]},{phiL,1/Sqrt[2]}},
   {SeR, {vR[3], 1/Sqrt[2]}, {sigmaR,I/Sqrt[2]},{phiR,1/Sqrt[2]}}};
```

If one wants to consider smuon and stau VEVs, `vL[2,3]`, `vR[2,3]` can be used.

5.6.1.4 Complex VEVs

To define complex VEVs, it is possible to give the phase as last argument:

```
DEFINITION[EWSB][VEVs]=
  {{SHd0, {vd, 1/Sqrt[2]}, {sigmad,I/Sqrt[2]},{phid,1/Sqrt[2]}},
   {SHu0, {vu, 1/Sqrt[2]}, {sigmau,I/Sqrt[2]},{phiu,1/Sqrt[2]},{eta}}};
```

This is understood as $H_u^0 \to \frac{\exp(i\eta)}{\sqrt{2}}(v_u + i\sigma_u + \phi_u)$. Another possibility to define complex VEVs is to define

```
DEFINITION[EWSB][VEVs]=
 {{SHd0, {vdR, 1/Sqrt[2]}, {vdI, I/Sqrt[2]},
                            {sigmad,I/Sqrt[2]},{phid,1/Sqrt[2]}},
  {SHu0, {vuR, 1/Sqrt[2]}, {vuI, I/Sqrt[2]},
                            {sigmau,I/Sqrt[2]},{phiu,1/Sqrt[2]}}
   };
```

which is understood as

$$H_d^0 \to \frac{1}{\sqrt{2}}\left(v_d^R + i v_d^I + i\sigma_d + \phi_d\right) , \qquad H_u^0 \to \frac{1}{\sqrt{2}}\left(v_u^R + i v_u^I + i\sigma_u + \phi_u\right) . \tag{5.11}$$

This format has the advantage that the tree-level tadpole equations are also in the complex case are purely polynomials and can be used numerically with dedicated codes like `HOM4PS2` [34] which is used by `Vevacious`.

5.6.2 Decomposition of Flavors

5.6.2.1 Introduction

If a model without flavor violation is considered, it might be demanded to give each generation of a family an own name, e.g. use different symbols for electron, muon and tau.

5.6.2.2 Implementation in SARAH

The expansion of flavors is done by

```
DEFINITION[EWSB][Flavors]= { ...
 {Field, {Name 1, Name 2, ... }},
 ... }
```

There must be as many names as generations of `Field` exist.

Example The down-type quarks can be expanded as follows:

```
DEFINITION[EWSB][Flavors]= {
 {Fd0L, {FdL,FsL,FbL}},
 {Fd0R, {FdR,FsR,FbR}} }
```

5.6.3 Mixings in the Gauge Sector

5.6.3.1 Introduction

After breaking a gauge symmetry, the vector bosons mix among each other as well as the gauginos do. In general, the different generations of a vector boson or gaugino rotate to different mass eigenstates. Hence, it is not possible to use the same parametrization as in the matter sector, shown in sec. 5.6.4.

5.6.3.2 Implementation in SARAH

It is possible to define the mixing in the gauge sector in the following way:

```
DEFINITION[$EIGENSTATES][GaugeSector]=
{ {{Old 1a, Old 2a,...},{New 1a, New 2b,..},MixingMatrix 1},
  {{Old 1b, Old 2b,...},{New 1b, New 2b,..},MixingMatrix 2},
  ...
```

Here, `Old Nx` is the name of the old and `New Nx` of rotated eigenstates. `MixingMatrix X` is the rotatin matrix relating the old and new basis. SARAH interprets this definition as matrix multiplication:

$$(N_1, N_2, \dots)^T = M(O_1, O_2, \dots)^2 \tag{5.12}$$

Example We consider the electroweak symmetry breaking of the MSSM

$$\begin{aligned} W_1 &= \frac{1}{\sqrt{2}}\left(W^- + W^{-*}\right) \ , && (5.13)\\ W_2 &= i\frac{1}{\sqrt{2}}\left(W^{-*} - W^-\right) \ , && (5.14)\\ W_3 &= \sin\Theta_W \gamma + \cos\Theta_W Z \ , && (5.15)\\ B &= \cos\Theta_W \gamma - \sin\Theta_W Z \ , && (5.16) \end{aligned}$$

This is given in SARAH by

```
DEFINITION[EWSB][GaugeSector]=
{ {{VB,VWB[3]},{VP,VZ},ZZ},
  {{VWB[1],VWB[2]},{VWm,conj[VWm]},ZW},
  {{fWB[1],fWB[2],fWB[3]},{fWm,fWp,fW0},ZfW}
};
```

The rotation matrices $Z^{\gamma Z}$ (`ZZ`), Z^W (`ZW`) and $Z^{\tilde{W}}$ (`ZfW`) are defined in the parameter file of the corresponding model as

$$Z^{\gamma Z} = \begin{pmatrix} \cos\Theta_W & -\sin\Theta_W \\ \sin\Theta_W & \cos\Theta_W \end{pmatrix} , \qquad Z^W = Z^{\tilde{W}} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ -i & i \end{pmatrix} \tag{5.17}$$

In the definition of EWSB W^+ (`VWp`) can not be used! This would introduce a new complex field not related to W^- (`VWm`) and therefore change the degrees of freedom of the theory. Of course, the winos $\tilde{W}^+$ and $\tilde{W}^-$ are not related by complex conjugation, so two new fields `fWm` and `fWp` are used after mixing.

SARAH always handles the vector bosons for unbroken gauge theories as real particles. To define the mass eigenstates as real or complex SARAH checks to number of degrees of freedom.

5.6.4 Particle Mixings

Symmetry breaking or also bilinear terms in the superpotential lead to a rotation of the former (gauge) eigenstates to new mass eigenstates. The definition of these rotations depends on the fact, if the corresponding mass matrix is hermitian or not.

5.6.4.1 Introduction

Properties of hermitian mass matrices In the hermitian case, the Lagrangian has the form

$$\mathcal{L}_{Mass} = \phi^\dagger M \phi. \tag{5.18}$$

The matrix M can be diagonalized by a matrix U:

$$M_{Dia} = U^{-1} M U. \tag{5.19}$$

The eigenvalues of the mass matrix M are the masses of the new mass eigenstates ψ, which are related to the former eigenstates by

$$\psi^i = U^{ij} \phi_j \tag{5.20}$$

Properties of non-hermitian mass matrices If the mass term in the Lagrangian is built by two vectors ϕ_1, ϕ_2, i.e.

$$\mathcal{L} = \phi_1^T M \phi_2 \ , \tag{5.21}$$

two mixing matrices are needed to diagonalize the mass matrix:

$$M_{Dia} = V^{-1} M U. \tag{5.22}$$

The two mixing matrices V and U diagonalize the matrices MM^T and $M^T M$

$$M^2_{Dia} = V^{-1} M M^T V, \qquad M^2_{Dia} = U^{-1} M^T M U \tag{5.23}$$

and connect the new eigenstates $\vec{\psi}_1$ and $\vec{\psi}_2$ with the old ones by

$$\psi_1^i = V_{ij} \phi_1^j, \qquad \psi_2^i = U_{ij} \phi_2^j \tag{5.24}$$

5.6.4.2 Implementation in SARAH

Both type of mixings are defined due to

```
DEFINITION[$EIGENSTATES][MatterSector] = { ... }
```

SARAH differs between hermitian and non-hermitian mixings by the form used for the definition. Note, that first all generations of one particle in the basis are inserted, before the next particle follows.

5.6.4.2.1 Hermitian mixings in SARAH The form for defining hermitian rotations is

```
{{List of Old Eigenstates},{Name of New Eigenstates, Name of Mixing Matrix}}
```

a) First, a list of the names of old eigenstates is given.

b) The name of the new eigenstates must be given.

c) The name of the mixing matrix must be given.

Examples The mixing in the down-squark sector is given by

```
{{SdL, SdR}, {Sd, ZD}}
```

This means, the three left d-squarks `SdL` and the three right d-squarks `SdR` mix to new eigenstates called `Sd` with generation index running from 1 to 6. The mixing matrix is called `MD`. The above statement leads to the following relation between the states

$$\tilde{d}^i = \sum_{j=1}^{3} Z^D_{i,j} \tilde{d}^j_L \tag{5.25}$$

$$\tilde{d}^{i+3} = \sum_{j=1}^{3} Z^D_{i+3,j} \tilde{d}^j_R \tag{5.26}$$

In the flavor conserving case, this matrix is reducible. Therefore, `SARAH` checks all matrices for reducibility and sets the non-block elements automatically to zero.

5.6.4.2.2 Non-hermitian mixings in `SARAH`

Non-hermitian rotations are defined in `SARAH` by

```
{{{First Basis},{Second Basis}},{{First States,First Matrix},{Second
States,Second Matrix}}}
```

Let us clarify this convention by an example.

Example We consider the chargino sector in the MSSM. This mixing is specified by

```
{{{fWm, FHdm}, {fWp, FHup}}, {{Lm,Um}, {Lp,Up}}}
```

This means that the gauge eigenstates $\tilde{W}^-$ (`fWm`) and $\tilde{H}_d^-$ (`FHdm`) mix to the negative charged mass eigenstates λ^- (`Lm`), while $\tilde{W}^+$ (`fWp`) and $\tilde{H}_u^+$ (`FHup`) form the new eigenstates λ^+ (`Lp`). The new and old eigenstates are connected by the mixing matrices U^- (`Um`) and U^+ (`Up`).

$$\begin{pmatrix} \lambda_1^- \\ \lambda_2^- \end{pmatrix} = U^- \begin{pmatrix} \tilde{W}^- \\ \tilde{H}_d^- \end{pmatrix}, \qquad \begin{pmatrix} \lambda_1^+ \\ \lambda_2^+ \end{pmatrix} = U^+ \begin{pmatrix} \tilde{W}^+ \\ \tilde{H}_u^+ \end{pmatrix} \tag{5.27}$$

5.6.5 Gauge fixing terms and ghost interactions

5.6.5.1 Introduction

As explained in app. B.1, the general form of a gauge fixing term in R_ξ-gauge is

$$\mathcal{L}_{GF} = -\frac{1}{2R_\xi} \sum_a |f^a|^2 \tag{5.28}$$

with some gauge fixing functions f^a.

5.6.5.2 Implementation in SARAH

If ghost vertices were to be calculated by SARAH 3.1 or earlier versions, it has been necessary to define the gauge fixing terms in R_ξ gauge. However, since version 3.2 SARAH derives these terms automatically using the calculated kinetic terms of the Lagrangian. To this end, the condition is applied that the mixing between scalar particles and vector bosons vanishes. Afterwards, the derived gauge fixing terms are used to calculate the ghost interactions.
Since it can happen in models with an extended gauge sector that several Goldstone bosons are a mixture of the same gauge eigenstate, for each massive vector boson, the corresponding Goldstone boson has to be defined

```
{{   Description -> "Z-Boson",
        ...
        Goldstone -> Ah[{1}]}},
...
{{   Description -> "Z'-Boson",
        ...
        Goldstone -> Ah[{2}]}},
```

The user can check the gauge fixing terms derived by SARAH be looking at

```
DEFINITION[$EIGENSTATES][GeneratedGaugeFixing]
```

The general form of the gauge fixing term is:

```
DEFINITION[$EIGENSTATES][GeneratedGaugeFixing] = {{Function, Prefactor}, ... };
```

Here, `Function` is the f of eq. (5.28), and the corresponding factor is `Prefactor`. If the gauge fixing functions involve derivatives of gauge bosons,

```
Der["Gauge Boson"]
```

is used.

Examples

a) The gauge fixing term for the color group in R_ξ gauge is:

$$\mathscr{L}_{GF} = -\frac{1}{2\xi_g}|\partial_\mu g|^2 \tag{5.29}$$

The corresponding expression in SARAH reads

```
DEFINITION[GaugeES][GeneratedGaugeFixing]=
{{Der[VG], -1/(2 RXi[G])},...};
```

b) The gauge fixing term corresponding to the Z-Boson after EWSB is (see app. B.20).

$$\mathscr{L}_{GF} = -\frac{1}{2\xi_Z}\left(\partial^\mu Z_\mu + \xi_Z M_Z G^0\right)^2 \tag{5.30}$$

The corresponding Goldstone boson is in SARAH the first generation of the CP-Odd Higgs (see app. G). Therefore, the gauge fixing term obtained by SARAH are

```
DEFINITION[EWSB][GeneratedGaugeFixing]=
{(2*Der[VZ] - (sigmad*vd - sigmau*vu)*RXi[Z]*
    (g1*ZZ[1, 2] - g2*ZZ[2, 2]))/2, -1/(2*RXi[Z])}
```

Here, ZZ is the $\gamma - Z$ mixing matrices which can be expressed by the Weinberg angle. Therefore, this expression is equivalent to the input used in older version of SARAH

```
DEFINITION[EWSB][GaugeFixing]=
{{Der[VZ] + Mass[VZ] RXi[Z] Ah[{1}],  - 1/(2 RXi[Z])},...};
```

5.6.6 Additional couplings or redefinition of existing couplings

5.6.6.1 Introduction

Sometimes, it might be necessary to define interactions in the Lagrangian, which can not be derived from the superpotential or the kinetic interaction. Furthermore, it might be necessary to change the properties of some vertices by hand. For example, integrating out particles might most likely spoil SUSY. Therefore, supersymmetric relations are not longer valid: the standard model Higgs self couplings are free parameters, while they are fixed in SUSY by the gauge and Yukawa couplings.

5.6.6.2 Implementation in SARAH

Both, defining new interactions and changing existing ones, is done in SARAH with one statement. For each set of eigenstates, new or changed terms can be defined separately by declaring them in the model file via

```
DEFINITION[$EIGENSTATES][Additonal] = {
{Lag, {Options}},
  ... };
```

Lag is a Lagrangian which is added to the complete Lagrangian, so it must has mass dimension 4. A rather short notation can be used, only some points have to be considered

- Fields are separated by dots
- Weyl fermions are used
- All indices are automatically added and contracted

The new couplings are handled in a similar way as the couplings of the superpotential: tensor indices are added automatically and they are assumed to be complex. Further assumptions about the coupling can be made in parameters.m.
The manually defined terms will be handled like every other term of the Lagrangian, i.e. they are affected by rotations and replacements if the eigenstates are changed.
The two possible options are:

a) AddHC -> True/False: Defines if the hermitian conjugated of this term is not added to the Lagrangian

b) Overwrite -> True/False: Defines if existing couplings involving the same fields are overwritten.

Example

a) Define new terms A mixed soft-breaking term of the form

$$m^2_{lH_d}(\tilde{l}^* H_d + \tilde{l} H_d^*) \tag{5.31}$$

is added to the Lagrangian of the gauge eigenstates by

```
DEFINITION[GaugeES][Additional] =
{{mlHd2 conj[Sl].SHd, {Overwrite->False, AddHC->True}}};
```

b) Adding additional terms to existing couplings With

```
DEFINITION[EWSB][Additional] =
{{1/24 Kappa hh.hh.hh.hh, {Overwrite->False, AddHC->False}}};
```

the Higgs self couplings receive an additional contribution:

$$\Gamma_{h^4}\ (a_i g_i^2 + b_i Y_i^2) \quad \rightarrow \quad \Gamma_{h^4}\ (a_i g_i^2 + b_i Y_i^2 + \kappa) \tag{5.32}$$

c) Overwriting existing terms To overwrite the former expressions for the Higgs self interactions,

```
DEFINITION[EWSB][Additional] =
{{LagNew, {Overwrite->True, AddHC->False}}};
LagNew = 1/24 Kappa hh.hh.hh.hh + 1/24 Lambda hh.hh.Ah.Ah;
```

is used. This has the following effect:

$$\Gamma_{h^4}\ (a_i g_i^2 + b_i Y_i^2) \quad \rightarrow \quad \Gamma_{h^4}\ \kappa \tag{5.33}$$

d) Interactions involving derivatives For interactions involving derivatives, Der is used

```
DEFINITION[EWSB][Additional] =
{{Kappa Der[SHd,lor3].conj[SHd].VB,{Overwrite->False, AddHC->True}}};
```

e) Matter interactions of the standard model This method can be used to define the matter interactions of the standard model

```
SuperPotential = {};
DEFINITION[GaugeES][Additional]= {
        {LagHC, {Overwrite->True, AddHC->True}},
        {LagNoHC,{Overwrite->True, AddHC->False}}
};
LagNoHC = Mu conj[SH].SH + 1/24 Lambda1 conj[SH].SH.conj[SH].SH;
LagHC = - Yd conj[SH].Fq.conj[FdR] - Ye conj[SH].Fl.conj[FeR] + Yu SH.Fq.conj[FuR];
```

5.7 Effective or non-supersymmetric theories

It is easy in SARAH to integrate particles out of the spectrum to get an effective theory, or just to delete this particle to get a non-supersymmetric limit of the model.

5.7.1 Integrating out particles

5.7.1.1 General

If in a theory very heavy particles Φ exist, they are no physical degree of freedom if the energy scale is below the mass of the particle. In this case, an effective theory for the lighter fields ϕ is derived by integrating out the heavy states

$$L(\phi)_{eff} = \int L(\phi, \Phi) d\Phi \tag{5.34}$$

This procedure will lead to higher dimensional operators like the four fermion interaction in Fermi's theory.

5.7.1.2 Implementation in SARAH

Integrating out particles in SARAH is easy: the heavy particles, which should be integrated by, are added to IntegrateOut:

```
IntegrateOut = {Particle 1, Particle 2,...}
```

Here, Particle 1 can be the component name of a particle, e.g SdR for all right d-squarks. Moreover, if the superfield name is used, all corresponding component fields are integrated out. If only specific generations of one field should be integrated out, the assignment is

```
IntegrateOut = {Particle[{first,last}],...}
```

Here, first is the first generation which should be integrated out and last is the last one.

> The list IntegrateOut can consist of particles belonging to different eigenstates. All particles are integrated out at the first time they do appear.

Example

a) To get an effective theory by integrating out the gluino, use

```
IntegrateOut = { fG };
```

b) RGE running of soft-breaking masses leads to the effect that the first two generations of sleptons are heavier than the third one. Therefore, building an effective theory with only the third generation of squarks is done by

```
IntegrateOut = {SdR[{1,2}],SuR[{1,2}],SdL[{1,2}],SuL[{1,2}]};
```

5.7.2 Deleting particles

Deleting particles can be done in the same way as integrating them out. The difference is that there are no effective operators are calculated. Deleting is therefore faster and should be used, if the higher dimensional operators are not needed.
Deleting particles is done by

```
DeleteParticle={List Of Particles};
```

The usage of `DeleteParticles` Is the same as of `IntegrateOut`. By deleting particles it is easy get a non supersymmetric limit of a model: it is possible, i.e. one can get the standard model by deleting all SUSY particles and the non-standard Higgs field. Since `SARAH` supports are also a second $SU(3)$ it might also be possible to implement Technicolor (-like) theories with this procedure.
Caveat! Of course, one must be careful: Some remaining interactions are still based on SUSY relation, which are not longer valid for non-supersymmetric theories. For example, the four point self-interaction of the light Higgs is in SUSY defined by the gauge interactions, but a free parameter in the standard model. To delete such relations the option for redefining couplings can be used, see sec. 5.6.6.2.

5.8 Definition of Dirac spinors

Event generators and programs for calculating Feynman Diagrams are normally written for Dirac spinors, but `SARAH` does all internal calculations for Weyl spinors. Therefore, it is necessary to define, how the Weyl spinors combine to Dirac spinors. That can be done for each all eigenstates separately

```
DEFINITION[$EIGENSTATES][DiracSpinors]={
 Dirac Spinor -> {Weyl 1, Weyl 2},
 ...
 }
```

`Dirac Spinor` is the new name of the Dirac spinor, while `Weyl 1` and `Weyl 2` the names of the Weyl spinors building the left respectively right component of the Dirac spinor.

Example The electron `Fe` is built from the components `FeL` and `FeR` by

```
DEFINITION[EWSB][DiracSpinors]={
 Fe -> {FeL, FeR},
 ...
  }
```

while the neutralinos `Chi`, which are Majorana particles, consists only of the Weyl spinor `L0`

```
Chi -> {L0, conj[L0]}
```

5.9 Checking the model files

After the initialization of a model via `Start["MODEL"]` it can be checked for (self-) consistency using the command

```
CheckModel;
```

The following checks are performed:

- Leads the particle content to gauge anomalies?
- Leads the particle content to the Witten anomaly?
- Are all terms in the superpotential in agreement with charge, and if defined, R-parity conservation?
- Are there other terms allowed in the superpotential by gauge invariance and possibly R-parity conservation beyond those defined.

- Are all mixings consistent with unbroken gauge groups?
- Are all definitions of Dirac spinors consistent with unbroken gauge groups?
- Are there terms in the Lagrangian of the mass eigenstates which can cause additional mixing between fields?
- Are all mass matrices irreducible?
- Are the properties of all particles and parameters defined correctly?

Note, the check for anomalies works so far only for supersymmetric models.

Chapter 6

LaTeXoutput

6.1 Writing a LaTeX file

It is possible to write a LaTeX file with all information about the mode by using

```
ModelOutput[Eigenstates, WriteTeX->True];
```

This calculates first all interactions for the eigenstates. If this was already done before, it is also possible to use

```
MakeTeX[Options];
```

There are different Tex-files produced containing the following information:

a) List of the fields

b) Important parts of the Lagrangian (soft-breaking terms, gauge fixing terms)

c) Mass Matrices and tadpole equations

d) Renormalization Group Equations

e) One-loop self energies and tadpole equations

f) All interactions

g) Details about the conventions used in `SARAH`

6.2 Options

The options are

a) `FeynmanDiagrams`, Values: `True` or `False`, Default: `True`
Defines, if the Feynman diagrams for all interactions should be drawn.

b) `effectiveOperators`, Values: `True` or `False`, Default: `True`
Defines, if the higher dimensional operators should be included in the LaTeXfile. By default, there are only the vertices involving up to four particles. For switching on six particle interactions `SixParticleInteractions` is used.

c) `SixParticleInteractions`, Values: `True` or `False`, Default: `False`
Defines, if also the six-particle interactions should be added to the LaTeX output

d) `ShortForm`, Values: `True` or `False`, Default: `False`
Defines, if a shorter notation for the vertices should be used

e) `WriteSARAH`, Values: `True` or `False`, Default: `False`
Defines, if the names and parameters used in `SARAH` should be written

6.3 Creating the pdf File

The LaTeX files are saved in the directory

```
../Output/$MODEL/$EIGENSTATES/TeX
```

and the main file is `$MODEL <> $EIGENSTATES <>.Tex`. All other files are included in this file by using the `input`-command of LaTeX. If `Diagrams->True` is used, the following steps must be done for generating an pdf document including the diagrams:

a) First, compile the Tex file, e.g. `pdflatex model.tex`

b) Go to the directory `Diagrams` and compile every `.mp` file with `mpost`. This is done under Linux and under Windows with

```
mpost FeynmanDiaX.mp
```

It is also possible to apply the `mpost` command on all `.mp`-files at once by using

```
find . -name "*.mp" -exec mpost {} \;
```

c) After generating all diagrams, go back and compile the `.tex`-file again by using `pdflatex`.

To simplify this procedure, `SARAH` will write a shell script in the Tex-output directory which does exactly these three steps. It can be started under Linux with

```
./MakePDF.sh
```

or under Windows with

```
MakePDF.bat
```

It is possible that the script must be first declared is executable in Linux via

```
chmod 755 MakePDF.sh
```

Chapter 7

Output for FeynArts

7.1 Generate model files for FeynArts

A model file for "FeynArts " is created by

```
ModelOutput[$EIGENSTATES, WriteFeynArts->True]
```

or, if `ModelOutput` was used before and the vertices are already calculated, by

```
MakeFeynArts[Options]
```

As options can `AddCounterTerms -> True` be used to add counter terms to each parameter: in that case all appearing parameters are replaced by $x \to x + \delta x$. Note, this option has to be used carefully because the routines are not yet very sophisticate. For instance, the shift is applied for mixing angles like the Weinberg angle itself and not for trigonometric functions as it is usually done. This will be improved in the future.
The following things are done:

a) A list of all particles present in the models is generated. The particles in `FeynArts` are named

   (a) `S[X]`: For scalars, with some integer `X`

   (b) `F[X]`: For fermions, with some integer `X`

   (c) `V[X]`: For vector bosons, with some integer `X`

   (d) `U[X]`: For ghosts, with some integer `X`

   `X` can be defined in the particle definitions file of `SARAH` or is chosen automatically. `FeynArts` also supports labels for particles which are easier to read for humans eyes by using a TeX-like output. The label for each particle is generated from the defined LaTeX name by `SARAH`.

b) A list with all appearing indices is written.

c) The list with interactions is written. If the theory contains several non-Abelian, unbroken gauge groups, the generators of these gauge groups will appear in the vertices. By default, the generators of $SU(3)$ are associated to `SUNT` and automatically simplified when using `FormCalc` [35].

7.2 Dependences, numerical values and special abbreviations for FormCalc

A second file is generated by SARAH together with the model file for FeynArts. This file is called `Substitutions-<> $EIGENSTATES <>.m` and contains additional information which might be useful for calculating diagrams:

a) Replacement rules with the defined dependences in SARAH, see sec. 4.2.3:

  (a) `Dependences`

  (b) `DependencesOptional`

  (c) `DependencesNum`

b) The definitions of the masses: `Masses` in SARAH

c) The numerical values for the parameters in SARAH: `NumericalValues`

d) Special abbreviations for FormCalc like those are also defined for the MSSM and SM in the FormCalc package:

  (a) A complex conjugation is replaced by C: `Conjugate[X] ->XC`

  (b) A square is replaced by 2: `X^2 -> X2`

  (c) The names of soft-breaking couplings are merged: `T[X] -> TX`

It is recommended to use this definitions to speed up the calculations with FormCalc.

Chapter 8

Output for CalcHep/CompHep

8.1 Generate model files for CalcHep and CompHep

To generate model files for CalcHep and CompHep,

```
ModelOutput[$EIGENSTATES, WriteCHep->True]
```

or

```
MakeCHep[options]
```

is used. The second command offers more options to control the output:

a) FeynmanGauge, Values: True or False, Default: True
By setting to True, the interactions of the Goldstone bosons are written in the interaction file.

b) CPViolation, Values True or False, Default: False
By setting to True, the possibility of CP violation is included in the model files, see sec. 8.1.4.

c) ModelNr, Values Integer, Default: 1
The number added to names of the files, see next section.

d) CompHep, Values: True or False, Default: False
By setting to True the model files are written in the CompHep format.

e) NoSplittingWith, Values: Particles List, Default: {}
If one of the given particles appears in a four-point interaction, the interaction is not split using auxiliary fields

f) NoSplittingOnly, Values: Particles List, Default: {}
If all particles of a four-point interaction appear in the given list, the interaction is not split using auxiliary fields

g) UseRunningCoupling, Values: True or False, Default: False
Defines, if the standard running of the strong coupling should be included in the model file.

h) SLHAinput, Values: True or False, Default: False
Defines, if parameter values should be read from a LesHouches input file, see sec. 8.1.5.

i) `CalculateMasses`, Values: `True` or `False`, Default: `False`
The tree level mass matrices are diagonalized by `CalcHep` to calculate the masses and rotation matrices. Note, `SLHAinput -> False` has to be used in addition

j) `RunSPhenoViaCalcHep`, Values: `True` or `False`, Default: `False`
Writes C code to run `SPheno` from the graphical interface of `CalcHep` to calculate the input parameters on the fly. Note, the path to the `SPheno` executable for the considered model has to be set in the file `SPhenoViaCalcHep.c` written by `SARAH`

k) `IncludeEffectiveHiggsVertices`, Values: `True` or `False`, Default: `False`
Includes the loop induced vertices of the CP even and odd Higgs to two photons or gluons. The numerical values for these couplings can be calculated with `SPheno` and they are given in the spectrum file and also included by `SLHAinput->True`.

8.1.1 Model Files

The `CalcHep/CompHep` output of `SARAH` generates the following four files

a) `prtclX.mdl`: Contains all particles

b) `lgrngX.mdl`: Contains the interactions

c) `varsX.mdl`: Contains the numerical values of the variables

d) `funcX.mdl`: Contains dependences between the parameters

`X` is a number, which can be chosen by the option `ModelNr`.

8.1.1.1 Particles

First, there are stringent constraints on the naming of particles in `CalcHep`: only names up to four letters are allowed and also indices aren't supported. Therefore, it is not possible to use the `SARAH` internal definitions of particles. Thus, the names used for the model files are based on the defined `OutputName` of each particle in the following way

a) The basis of each name is the entry in `OutputName` in the particle file, see sec. 4.3

b) If the considered particle is not self-conjugated, for the anti particle the first letter is changed from upper to lower case or vice versa.

c) If there are more generations for one particle, the number of the generation is appended at the end of the name

d) If the defined R-parity is -1, a `~` is added to the beginning of the name to assign SUSY particles. In this way, it is possible to use the model files in `MicrOmegas` without the need of an additional list of all SUSY particles

The steps above are the standard procedure for all vector bosons, fermions and most scalars. Ghosts, Goldstone bosons and auxiliary fields handled in a different way. There are three different kind of ghosts. These are not written in the particle file, but appear in the Lagrangian file:

a) Faddeev-Popov Ghost: these are the well known Ghost derived from the gauge transformations of the gauge fixing term. The name in the model file is

```
"Name of Vector Boson".C
```

b) Goldstone Ghosts: these are just the Goldstone bosons 'eaten up' by the gauge particles. Their name is

```
"Name of Vector Boson".f
```

c) Tensor Ghosts: Is needed to express the four gluon interaction. The name is

```
"Name of Gluon".t
```

SARAH derives the name of Goldstone and Faddeev-Popov ghost automatically from the underlying vector boson, but the tensor ghost and its one interaction with two gluons is hard-coded.

The last kind of fields known by CalcHep/CompHep are auxiliary fields. Their purpose is explained in the next section, but their names are as follows

```
~OX
```

Here, X is an integer. The antiparticle, if it is not the same, is counted as X+1.

8.1.2 Auxiliary fields in CalcHep/CompHep

We mentioned in the last section that CalcHep and CompHep needs special auxiliary fields. The reason is that the color structure is implicit. Hence, interactions of four colored particles or two colored and two gluons suffer from an ambiguity. Therefore, these interactions are split in two three particle interactions by inserting auxiliary fields.
SARAH does a similar splitting for all interactions between four scalars by inserting auxiliary fields when calculating the F- and D-Terms. Also the interactions between two squarks and two gluons are split in two three particle interactions. The splitting can be suppressed for specific vertices by using NoSplittingWith or NoSplittingOnly.

8.1.3 Vertex functions

All interactions are parametrized by a variable in the Lagrange file. The values of these variables are defined by using the results of SARAH for the corresponding vertices. The following renaming had to be done:

a) Tensor indices are just added to the name, therefore all sums in the vertices had do be evaluated:

```
sum[i1,1,3,MD[1,i1]]      ->       MD11 + MD12 + MD13
```

Some values, which are known to be zero like in the flavor conserving case, are set to zero at this point.

b) Variables names, which are longer then six letters, are truncated.

c) All parameters are assumed to be real, i.e. complex conjugation is removed (see sec. 8.1.4).

d) The generators and structure constants of the strong interaction are removed because they are defined implicitly.

possible combinations of generation indices. This can lead for some models to a very long time for writing the model file, e.g. in the $\mu\nu SSM$ with 8 charged Higgs: All 8^3 combinations of the self interactions must be written.

8.1.4 CP Violation

CalcHep/CompHep can't handle complex values in the function or vars file, but only in the Lagrange file. Therefore, all variables are by default assumed to be real, when SARAH writes the model file with default options. This can be changed by setting CPViolation to True. In that case, SARAH splits all parameters, which are not explicitly defined as real, in real and imaginary part:

```
X -> RX + i*IX
```

The real and imaginary part for every interaction is calculated using that splitting, and both parts are written separately in the Lagrange file:

```
v0001 -> Rv0001 + i*Iv0001
```

SARAH writes in this file the output name of every needed variable. Since CalcHep/CompHep does only support variable names with a length of maximal 6 letters, SARAH cuts the name of all variables, which are longer then that. If a numerical value for the variable is available, e.g. if it is defined in the parameter file or it was read in from LesHouches file, it will also be added in the vars file, otherwise NaN appears. func1.mdl.

8.1.5 SLHA input

CalcHep supports the possibility to read LesHouches input files [36]. SARAH can write the corresponding definitions in the functions file of CalcHep. In this context, it is assumed that a LesHouches file called SPheno.spc.[MODEL] is located in the same directory. However, this can easily be adjusted manually.

8.1.6 What can be a problem...

We have made the following experiences by testing model files with CalcHep/CompHep:

a) A PDG number of 0 is not allowed for other particles than auxiliary particles.

b) In the vars file is no discrimination between small and capital letters. This must be taken into account by naming the mixing matrices and couplings in SARAH.

c) Higher dimensional operators are not supported

d) The color structure is implicit and indices are not supported in CalcHep/CompHep. Therefore, it is difficult to implement models with other unbroken non-Abelian gauge groups than the color group.

e) Writing the output for models with particles appearing in a large number of generations and non reducible mixing matrices like in the flavor conserving MSSM, last very long, because all possible combinations of indices have to be written separately.

8.2 micrOMEGAs

micrOMEGAs [37] is a well known tool for the calculation of the relic density of a dark matter candidate. micrOMEGAs uses CalcHep to calculate the annihilation and co-annihilation processes. Therefore, it is necessary to generate first a model file for CalcHep to implement new models in micrOMEGAs as described in above.
SARAH writes two files for micrOMEGAs which can serve as so called main files, i.e. they can be compiled with micrOMEGAs and executed to perform calculations. While CalcOmega.cpp calculates only Ωh^2 and

writes the result to the file omg.out. CalcOmega_with_DDetection.cpp computes also direct detection signals. In that case, the numbers in the different lines in omg.out correspond to

- Relic density Ωh^2
- Spin independent cross section with proton in pb
- Spin dependent cross section with proton in pb
- Spin independent cross section with neutron in pb
- Spin dependent cross section with neutron in pb
- Recoil at ^{73}Ge: number of events in 10 - 50 keV region
- Recoil at ^{131}Xe: number of events in 10 - 50 keV region
- Recoil at ^{23}Na: number of events in 10 - 50 keV region
- Recoil at ^{127}I: number of events in 10 - 50 keV region

Using the SLHA+ functionality of CalcHep is also possible with micrOMEGAs. Therefore, it is sufficient to copy the spectrum file written by SPheno to the directory of micrOMEGAs and start the calculation.

Chapter 9

Output for WHIZARD

(in collaboration with C. Speckner)

To generate model files for WHIZARD

```
ModelOutput[$EIGENSTATES, WriteWHIZARD->True]
```

is used.If the vertices have been calculated already by MakeVertexList,

```
MakeWHIZARD[options]
```

can be used. The possible options are

a) WriteOmega, Values: True or False, Default: True
Defines, if the model files for Omega should be written

b) WriteWHIZARD, Values: True or False, Default: True
Defines, if the model files for WHIZARD should be written

c) Exclude, Values: list of generic type, Default: {SSSS}
Defines, which generic diagrams are excluded when writing the model file

d) WOModelName, Values: string, Default: defined model name
Gives the possibility to change the model name

e) MaximalCouplingsPerFile, Values: Number, Default: 500
Defines the maximal number of couplings written in one file

f) Version, Values: Number, Default: 2.0.3
Defines the version of WHIZARD for which the model file is generated

g) ReadLists, Values: True or False, Default: False
Defines, if the information from a former evaluation should be used

Using the generated model files with WHIZARD

After the interface has completed, the generated files can be found in the WHIZARD_Omega subdirectory of SARAHs output directory. In order to use the model with WHIZARD 2.x, the generated code must be compiled and installed. For most applications, this is done by simply issuing (inside the output directory)

```
./configure
make
make install
```

By default, the third command installs the compiled model into .whizard in current user's home directory where it is automatically picked up by WHIZARD. Alternative installation paths can be specified using the --prefix option to WHIZARD.

```
./configure --prefix=/path/to/installation/prefix
```

If the files are installed into the WHIZARD installation prefix, the program will also pick them up automatically, while WHIZARD's --localprefix option must be used to communicate any other choice to WHIZARD. In case WHIZARD is not available in the binary search path, the WO_CONFIG environment variable can be used to point configure to the binaries

```
./configure WO_CONFIG=/path/to/whizard/binaries
```

More information on the available options and their syntax can be obtained with the --help option.
In the case of WHIZARD 1.x output, the generated files must be patched into the WHIZARD source tree. To this end, the interface creates a script called inject. In most cases, it is sufficient to simply call the script as

```
./inject /path/to/whizard
```

(from within the output directory). Issuing ./inject --help will display a list of options which can be used to adapt the script to more complicated usage scenarios.

Chapter 10

Output in UFO format

To generate model files in the UFO [19] format which can be used e.g. with Madgraph,

```
ModelOutput[$EIGENSTATES, WriteUFO->True]
```

is used.If the vertices have been calculated already by MakeVertexList or ModelOutput,

```
MakeUFO[options]
```

can be used. The possible options are

a) Exclude, Values: list of generic type, Default: {SSSS,GGS,GGV}
Defines, which generic diagrams are excluded when writing the model file

The output written by SARAH consists of the files

- particles.py: contains the particles present in the model
- parameters.py: contains all parameters present in the model
- lorentz.py: defines the Lorentz structures needed for the vertices
- vertices.py: defines the vertices
- couplings.py: expressions to calculate the couplings
- coupling_orders.py: defines hierarchies for the coupling orders

These files are saved in the directory

```
$SARAH/Output/$MODEL/$EIGENSTATES/UFO/
```

This directory contains also additional files which are model independent and can therefore be used with all models: function_library.py, object_library.py, __init__.py and write_param_card.py. These files were kindly provided by Olivier Mattelaer.

Using the UFO model files of SARAH with Madgraph

To use the model files with Madgraph, it is sufficient to copy all files to $MADGRAPH/models/$NAME. Here, $MADGRAPH is the directory containing the local Madgraph installation and $NAME is a freely-chosen name of a new subdirectory. This directory name is used afterwards to load the model in Madgraph via

```
> import model $NAME
```

Madgraph has a list of pre-defined names for the particles of the SM and MSSM which are used by default. However, it could be that there are conflicts between these names and the names used by SARAH in an extension of the MSSM. For instance, h3 is defined in Madgraph as the pseudoscalar Higgs in the MSSM, but SARAH uses it in the NMSSM for the third-heaviest scalar Higgs. In order to avoid such clashes, a model can be loaded using only the names defined by the UFO files via

```
> import model $NAME -modelname
```

To use these model files for the calculation of cross sections, it is, of course, necessary to provide the numerical values of all masses and parameters. The necessary input can be obtained by using a SPheno module created by SARAH for the given model. The SLHA spectrum files written by this SPheno module can directly be used with Madgraph.

Chapter 11

Output for SPheno

11.1 Introduction

SARAH is based on Mathematica and therefore it is usually not sensible to do exhaustive numerical calculations in SARAH's native environment. As opposed to that, there is SPheno [20, 38], a well tested spectrum calculator written in Fortran. SPheno provides fast numerically routines for the evaluation of the RGEs, calculating the phase space of 2- and 3-body decays as well as Passarino Veltman integrals and much more. Since these routines are model independent, they can be used in principle for all SUSY models implemented in SARAH.
The generation of the source code for SPheno is started via

```
MakeSPheno[Options]
```

The different options are:

- ReadLists->True can be used if all vertices and RGEs have already been calculated for the model and the former results should be used to save time.
- InputFile. The name of the SPheno input file. If not defined, SPheno.m is used.
- StandardCompiler->Compile. The compiler which should be set as standard in the created Makefile. Default is gfortran.

The generated source code is located in

```
/Directory of SARAH/Output/$MODEL/$EIGENSTATES/SPheno/
```

To compile the code, a new sub-directory called $MODEL in local installation of SPheno has to be created and the code has to be copied into that directory. Afterwards,

```
make Model=$MODEL
```

has the be executed in the root directory of SPheno. This compiles the code and a new binary

```
SPheno <> $MODEL
```

is generated in bin/.

11.2 Input file

For the SPheno output, a new file SPheno.m is needed to define the properties of the generated SPheno version. The file SPheno.m must be located in the same directory as the other input files of the current model. The content of this can be the following:

a) MINPAR: A list of parameters which should be read from a LesHouches file by SPheno. First, the number in the block is defined, afterwards the variable. For example:

```
 MINPAR = {{1,m0},
           {2,m12},
           {3,TanBeta},
           {4,SignMu},
           {5,Azero}};
```

Now, all information about these parameters can later on given to SPheno by using an input file containing the part

```
 Block MINPAR #
    1  7.000000E+01 #  m_0
    2  2.500000E+02 #  M_1/2
    3  1.000000E+01 #  Tan(beta)
    4. 1.000000E+00 #  Sign(mu)
    5. 0.000000E+00 #  A_0
```

In case that several sets of parameters are demanded, e.g. to support mSugra- and GMSB-like boundary conditions, MINPAR can also be a nested like, e.g.

```
MINPAR=Table[{},{2}];

MINPAR[[1]]={{1,m0},
             {2,m12},
             {3,TanBeta},
             {4,SignumMu},
             {5,Azero}};

(* GMSB input parameters *)

MINPAR[[2]]={{1,LambdaInput},
             {2,MessengerScale},
             {3,TanBeta},
             {4,SignumMu},
             {6,cGrav},
             {7,n5plets},
             {8,n10plets}};
```

In that case MINPAR[[X]] is associated with the boundary conditions BoundarySUSYScale[[x]], BoundaryHighScale[[x]], BoundaryEWSBScale[[x]] described below.

b) EXTPAR: It is also possible to define additional parameters for the block EXTPAR of the LesHouches input file by

```
EXTPAR = {{Nr1,  Var1},
          {Nr2,  Var2},
          ...};
```

For instance, in order to give three additional VEVs as input, we can use

```
EXTPAR = {{100, v1},
          {101, v2},
          {102, v3}};
```

and set the values later on in the input file by

```
 Block EXTPAR #
    100  1.000000E-04 #  v_1
    101  1.500000E-04 #  v_2
    103  2.000000E-04 #  v_3
```

Note, there are no hardcoded entries for MINPAR or EXTPAR. That makes it necessary to define these blocks also for models for which in principle SLHA conventions exist. However, that provides also more freedom in varying the model and the free parameters.

c) ParametersToSolveTadpoles: SARAH derives for each VEV the corresponding minimum condition for the vacuum. These equations give constraints to the same number of parameters as VEV are in the models. ParametersToSolveTadpoles is used to set the parameter which fixed by the tadpole equations.
For example, to use the standard choice in the MSSM μ, B_μ, the entry reads:

```
 ParametersToSolveTadpoles = {\[mu], B[\mu]};
```

SARAH uses the Solve command of Mathematica to solve the tadpole equations for the given set of parameters. If the solution is not unique because a parameter X appears squared, SARAH solves the equations for the absolute squared. The phase is defined by the automatically generated variable SignumX, which is expected to be given as input.
The expressions for the constrained parameters are automatically used during the numerical analysis at the SUSY as well at the electroweak scale. That's possible, because also the RGE evaluation of all VEVs is included in the generated SPheno version.
If models with CP violation in the Higgs sector are studied, it is often necessary to solve the tadpole equations for complex parameters. This can be done by demanding that Mathematica should solve the tadpole equations for the real and imaginary part of the corresponding parameter, e.g.

```
 ParametersToSolveTadpoles = {\[mu], re[B[\mu]], im[B[\mu]]};
```

d) AssumptionsTadpoleEquations It is possible to define a list with replacements which are by SARAH when it tries to solve the tadpole equations. For instance, to approximate some matrices as diagonal and assume that all parameters are real, use

```
AssumptionsTadpoleEquations = {Ye[a__]->Delta[a] Ye[a],
    T[Ye][a__]->Delta[a] T[Ye][a], conj[x_]->x};
```

e) `UseGivenTapdoleSolution`: In cases, in which `Mathematica` won't find an analytical solution for the tadpole equations for the given set of parameters, this variable has to be set to `True` and an approximated solution can be given. These solutions are defined by

- `SubSolutionsTadpolesTree`: For the solution at tree level

  ```
  SubSolutionsTadpolesTree = {x1 -> sol1, x2 -> sol2,...};
  ```

  Here, `x1`, `x2` are the names of the parameters which are fixed by the tadpole equations and `sol1`, `sol2` are the approximated expressions for them.

- `SubSolutionsTadpolesLoop`: The solutions of the one loop corrected tadpole equations. The one loop corrections to the different VEVs have to be named `Tad1Loop[i]`.

f) `RenormalizationScale`: It is possible to use a dynamical adjusted renormalization scale, e.g. a function of the stop masses

```
RenormalizationScale = MSu[1]*MSu[6];
```

g) `RenormalizationScaleFirstGuess`: For a first evaluation of the RGEs, before any mass has been calculated and `SPheno` hasn't any glue about the renormalization scale, an approximated scale can be used as 'first guess'. For example, for a mSugra scenario the common choice is

```
  RenormalizationScaleFirstGuess = m0^2 + 4 m12^2;
```

This affects the running only if the SUSY scale is not fixed and SPA conventions are disabled in the LesHouches input file.

h) **Two loop contributions to the Higgs masses**: if the Higgs sector of the model is the same as for the MSSM, the original SPheno routines for calculating the two loop tadpole equations and two loop self energies to the the scalar and pseudo scalar Higgs can be activated by setting

```
 UseHiggs2LoopMSSM = True;
```

i) **Boundary Conditions**: It is possible to define boundary conditions at three different scales:

- Electroweak scale: `BoundaryEWSBScale`
- SUSY scale: `BoundarySUSYScale`
- GUT scale: `BoundaryHighScale`

All these conditions are applied when running up and down with RGEs. In contrast, there is also the possibility to define a boundary condition at the EW scale which is only applied when running down from the SUSY scale:

```
BoundaryEWSBScaleRunningDown = ...
```

In addition, if thresholds are involved, boundary conditions can be set at the threshold scale. See section 11.2.1. It is also possible to use a low scale input without any RGE running. In that case special boundary conditions can be defined by the array `BoundaryLowScaleInput`.
All boundaries are defined by a two dimensional array. The first entry is the name of the parameter, the second entry is the used condition at the considered scale. The condition can be ...

- . . . an input parameter from MODSEL or EXTPAR, e.g.

  ```
  {MassB, m12};
  ```

- . . . a block in the SLHA input file, e.g.

  ```
  {Yv, LHInput[Yv]};
  ```

- . . . a function of different parameters, e.g.

  ```
  {TYd, Azero*Yd};
  ```

- . . . a diagonal matrix, e.g.

  ```
  {md2, DIAGONAL m0^2};
  ```

- . . . matrix multiplications or the inverse of a matrix, e.g.

  ```
  {X, MatMul2[A,InverseMatrix[B], FortranFalse]};
  ```

- . . . a self defined function

  ```
  {X, Func[A,B,C]};
  ```

  It is also possible to use some self defined function. The Fortran code of that function has to included in the array `SelfDefinedFunctions` in `SPheno.m`. It will later on be written to `Model_Data.f90`. Note, that the standard functions needed for GMSB are already included [39]:

  - fGMSB[X]:

    $$\begin{aligned} f(x) &= \frac{1+x}{x^2}\left(\ln(1+x) - 2\mathrm{Li}_2(\frac{x}{1+x}) + \frac{1}{2}\mathrm{Li}_2(2\frac{x}{1+x})\right) + \\ & \frac{1-x}{x^2}\left(\ln(1-x) - 2\mathrm{Li}_2(\frac{x}{x-1}) + \frac{1}{2}\mathrm{Li}_2(2\frac{x}{x-1})\right) \end{aligned} \tag{11.1}$$

  - gGMSB[X]:

    $$g(x) = \frac{1+x}{x^2}\ln(1+x) + \frac{1-x}{x^2}\ln(1-x) \tag{11.2}$$

For the matrix multiplication `MatMul2` has to be used. The third argument controls whether if only diagonal elements (`FortranTrue`) should be considered or not (`FortranFalse`).

Boundary conditions can be overridden by assigning a value to a parameter in the LesHouches input file. For example, the Higgs soft breaking masses at the GUT scale can be forced to have specific values instead of m_0^2 by declaring

```
Block MSOFTIN    #
 21  10000.000   # mHd2
 22  20000.00    # mHu2
```

in the SLHA file.

Several sets of boundary conditions In order to implement different versions of a single model which differ only by the used boundary conditions, `BoundaryEWSBScale`, `BoundarySUSYScale`, `BoundaryHighScale` can be also a nested list, e.g.

```
BoundarySUSYScale = Table[{},{2}];
BoundaryGUTScale = Table[{},{2}];

BoundarySUSYScale[[1]] = {{KappaNMSSM, KappaInput},
                          {LambdaNMSSM, LambdaInput}};
BoundaryGUTScale[[1]]  = {};

BoundarySUSYScale[[2]] = {};
BoundaryGUTScale[[2]]  = {{KappaNMSSM, KappaInput},
                          {LambdaNMSSM, LambdaInput}};
```

In the first case, the input values for λ and κ are used at the SUSY scale, in the second on at the GUT scale. To communicate to `SPheno` which set of boundary conditions should be used for a run, flag 2 in `MODSEL` is used:

```
Block MODSEL #
 2  X  # This uses the X. set of boundary conditions.
```

The default value is 1.

j) Lists for calculating decay widths:

- `ListDecayParticles`: List of particles for which the two-body decays are to be calculated. This can be a list of particles using the names inside `SARAH`, e.g.

  ```
  ListDecayParticles = {Sd,Su,Se,hh,Ah,Hpm,Chi};
  ```

  or just `Automatic`. If `Automatic` is used, the widths of all particles not defined as standard model particles as well as for the top quark are calculated.
- `ListDecayParticles3B`; Three body decays of fermions. This can be a list with the names of the particles and the corresponding files names, e.g.

  ```
  ListDecayParticles3B =  {{Chi,"Neutralino.f90"},
                            {Cha,"Chargino.f90"},
                            {Glu,"Gluino.f90"}};
  ```

  or just `Automatic`. If `Automatic` is used, the widths of all fermions not defined as standard model particles are calculated. The auto generated file names are `ParticleName.f90`.

This checks whether two massless pseudo scalars are present in the spectrum and, if this is the case, it uses as Goldstone boson the not singlet-like particle.

k) Flag to switch off loop-corrections: if

```
FlagLoopContributions = True;
```

is used, flags are created which can be used to disable loops which contain specific fields. This applies to the 1-loop mass corrections as well as the calculation of low-energy observables. The necessary entries to are added to the block `SPhenoInput` in the LesHouches input file and also listed in the template created by `SARAH`.

l) Low scale input: it is also possible to define the free parameters of the model at a specific scale without RGE running. These parameters are afterward used to calculate the loop corrected mass spectrum and the decays. The corresponding flag is

```
Block MODSEL #
 1  0     #  Low scale input
12  1000. #  Renormalization scale
```

which in this example would declare the parameters to be renormalized at 1 TeV if no explicit renormalization scale is defined.

Information about particles and parameters

SARAH needs some information about the physical meaning of some particles and parameters. These information are used to calculate the gauge and Yukawa couplings at the electroweak scale, calculate the CKM matrix, use the correct on-shell masses, etc. All definitions are done by the description statement in the parameters and particles file.

a) The following particles are needed:

- Leptons
- Down-Quarks
- Up-Quarks
- Photon
- Gluon
- W-Boson
- Z-Boson
- Up-Squarks
- Higgs
- Pseudo-Scalar Higgs
- Charged Higgs
- Neutrinos
- Sneutrinos

b) The following parameters have to be defined:

- Up-Yukawa-Coupling
- Down-Yukawa-Coupling
- Lepton-Yukawa-Coupling
- Hypercharge-Coupling
- Left-Coupling
- Strong-Coupling
- Up-Squark-Mixing-Matrix
- Down-Squark-Mixing-Matrix

- `Left-Lepton-Mixing-Matrix`
- `Right-Lepton-Mixing-Matrix`
- `Left-Down-Mixing-Matrix`
- `Right-Down-Mixing-Matrix`
- `Left-Up-Mixing-Matrix`
- `Right-Up-Mixing-Matrix`
- `Weinberg-Angle`
- `Down-VEV`
- `Up-VEV`
- `Scalar-Mixing-Matrix`
- `Pseudo-Scalar-Mixing-Matrix`
- `Softbreaking right Down-Squark Mass`
- `Softbreaking right Up-Squark Mass`
- `Softbreaking left Slepton Mass`
- `Softbreaking right Slepton Mass`
- `Softbreaking left Squark Mass`
- `Trilinear-Up-Coupling`
- `Trilinear-Down-Coupling`
- `Trilinear-Lepton-Coupling`

Mixing angles in gauge sector

The mixings in the gauge sector are normally expressed by some angles, e.g, Θ_W in SM/MSSM. `SPheno` diagonalizes the mass matrix for the gauge bosons and calculates also the values of the rotation matrices. Those can be used to calculate the corresponding mixing angle. For this purpose, the relation between the mixing angle and the mixing matrix have to be defined in the `parameter.m` file using an additional dependence called `DependenceSPheno` For instance,

```
{{ Description -> "Weinberg-Angle",
            ...
            DependenceSPheno -> ArcCos[Abs[ZZ[1,1]]] }},
```

11.2.1 Including Thresholds

Using `SARAH` it is possible to include thresholds in the RGE evaluation.

11.2.1.1 Thresholds without gauge symmetry breaking

If all scales have the same gauge structure, it is possible for `SARAH` to derive the RGEs for all scales from the RGEs for the highest scale by performing the following steps:

- For those fields which should be integrated out during the run new variables $n_{gen}(\Phi_i)$ are introduced, which define the number of generation of the heavy field Φ_i. All gauge group constants like the the Dynkin index summed over chiral superfields, $S(R)$, are expressed as function of $n_{gen}(\Phi_i)$. These $n_{gen}(\Phi_i)$ are dynamically adjusted, when the energy scale crosses a threshold.

- It is also necessary to set the couplings which involve heavy fields to zero when a threshold is crossed. For example, the Yukawa type coupling of the form $Y^{ij}\Phi_i\phi_j H$ involves three generations of the heavy field Φ. At the threshold of Φ_k, the k-th row of Y is set to zero. That happens similarly for all other superpotential and soft-breaking parameters.
- It is assumed, that the masses of the scalar and fermionic component of a heavy superfield are the same, i.e. the masses are much larger than the soft-breaking masses. Furthermore, it is assumed that the masses are given by a bilinear superpotential term.

To include thresholds without gauge symmetry breaking, the following steps have to be performed:

a) The heavy fields must be deleted in the SARAH model definition:

```
DeleteFields = {...};
```

This ensures, that the these particle are not take into account for the calculation of decays or loop corrections at the SUSY scale.

b) The thresholds have to be defined in SPheno.m :

```
Thresholds = {{Scale1, {HeavyFields1}},
              {Scale2, ... }};
```

For all scales an entry in the array Thresholds has to be added. Each entry defines first the threshold scale, at second position a list with the heavy superfields is given. Also specific generations for a superfield can be given.

It is possible to define boundary conditions at each threshold scale for running up and down separately:

```
BoundaryConditionsUp[[x]] = { ...};
BoundaryConditionsDown[[x]] = { ...};
```

Threshold corrections Using 2-loop RGEs demands 1-loop boundary condition. Therefore, at each threshold scale the one loop threshold corrections to gauge couplings and gaugino masses are calculated. The general expressions are [40]

$$g_i \quad \rightarrow \quad g_i\left(1 \pm \frac{1}{16\pi^2} g_i^2 I_2^i(r) \ln\left(\frac{M^2}{M_T^2}\right)\right) \ , \tag{11.3}$$

$$M_i \quad \rightarrow \quad M_i\left(1 \pm \frac{1}{16\pi^2} g_i^2 I_2^i(r) \ln\left(\frac{M^2}{M_T^2}\right)\right) \ . \tag{11.4}$$

$I_2^i(r)$ is the Dynkin index of a field transforming as representation r with respect to the gauge group belonging to the gauge coupling g_i, M is the mass of this particle and M_T is the threshold scale. When evaluating the RGEs from the low to the high scale the contribution is positive, when running down, it is negative.

Example As an example, a version of SPheno implementing the seesaw type II and type III models can be generated by adding the following entries to Spheno.m

a) Seesaw II:

```
Thresholds={
  {Abs[MTMIN],{s,sb,t,tb,z,zb}}
};
```

b) Seesaw III:

```
Thresholds={
  {Abs[MWM3IN[1,1]],{Hx3[1],Hxb3[1],Hg3[1],Hb3[1],Hw3[1]}},
  {Abs[MWM3IN[2,2]],{Hx3[2],Hxb3[2],Hg3[2],Hb3[2],Hw3[2]}},
  {Abs[MWM3IN[3,3]],{Hx3[3],Hxb3[3],Hg3[3],Hb3[3],Hw3[3]}}
};
```

11.2.1.2 Thresholds with gauge symmetry breaking

If the gauge structure at the different scales are different, each set of RGEs is calculated separately and this information is then combined into one consistent version of SPheno. This code includes routines for calculating finite shifts in the gauge couplings and gaugino mass parameters. As an example, the implementation of a left-right supersymmetric model with two symmetry breaking scales is shown in app. F.6. In order to implement such a model, the following steps are necessary:

a) For each regime, a separate model file for SARAH has to be created. These model file have to be saved in the subdirectories Regime-1, Regime-2, . . . of

```
[SARAH Directory]/Models/[Model]/
```

They are numbered beginning with the highest scale.

b) The SPheno input file for the higher scales must provide the following information:

- IntermediateScale = True
- RegimeNr = X
- A list of the heavy fields, which should be integrated out, the gauge sector below the threshold as well as the corresponding quantum numbers of the fields integrated. That's needed to calculate the finite shifts at the threshold scale. For instance, the entries might read

```
HeavyFields = {Field_1, Field_2,..};
NextGauge =   {U[1],   SU[2],   SU[2],    SU[3]};
NextQN = {
       {Field_1, 0,       2,       1,         1},
       {Field_2, 1/3,     1,       2,         4},
 ...
};
```

c) All necessary information for combining the regimes to one SPheno is given in SPheno.m of the lowest scale.

- IntermediateScale = False
- RegimeNr = X
- The threshold scales: ThresholdScales = ...

- The boundary conditions for running up and down at each threshold scale:

  ```
  BoundaryConditionsUp[[x]] = {...};
  BoundaryConditionsDown[[x]] = {...};
  ```

  In the boundary conditions `index1`, `index2`, ... can be used for defining sums over indices.
- The usual information for SPheno, defined in the sec. 11.2.

When starting the SPheno output of the lowest scale, automatically all other scales are evolved. Note, to calculate the RGEs of the different regimes requires SARAH to start one additional Mathematica kernel. For passing the information between the different Mathematica kernels a directory Meta in the model directory is created by SARAH. Also the screen output of Mathematica during the evaluation of the higher regimes is written to that directory (Output-Regime-X.m). So, the user can supervise the progress and see potential error messages. The necessary information of each regime for writing the combined source code for SPheno at the end is saved by SARAH in the files Regime-X.m.

11.2.2 Supported models and known issues

While SARAH can create valid SPheno code for many different models, there are some requirements on the model and some minor restrictions on the functionality of the resulting SPheno module. At the moment, those are

- **Fit to low energy data**: in order to perform a fit to low energy data (e.g. for fermion masses, m_Z, G_F and α_{em}) as starting point of the RGE evaluation, the following parameters must be present in the model: Yukawa couplings for lepton and quarks, two Higgs VEVs and, of course, the three SM gauge couplings and the SM particle content. However, it is still possible to use at least some features of the SPheno output of SARAH by manually supplying model parameters for SPheno. In that way, the RGE evaluation and the fit the electroweak data is skipped, but the one-loop corrected masses as well as the decay widths and branching ratios are calculated.

- **Flavor decomposition**: with SARAH it is possible to assign a unique name to each generation of a particular field and this way treat the individual generations as independent fields. That is not yet supported in the SPheno output. Furthermore, mixing matrices generated with the option NoFlavorMixing can not yet be handled by the numerical code.

11.2.3 Low energy SPheno version

It is also possible to create a SPheno version with much less features which only accepts low energy input. That means, the RGEs are not written out and also the fit to the electroweak data is not performed in the numerical evaluation of one point. It just solves the tadpole equations, calculates the tree- and one-loop masses as well as the decay widths and branching ratios. The advantage of such a SPheno version is that it works with a larger set of models, e.g. also non-SUSY models or other models not supported by a full evaluation as explained in sec. 11.2.2. To get a SPheno version without RGE evolution, insert

```
OnlyLowEnergySPheno = True;
```

in SPheno.m. The remaining information needed by SARAH is only a small subset of the settings discussed above and consists of

- MINPAR

- ParametersToSolveTadpoles

- BoundaryLowScaleInput
- ListDecayParticles and ListDecayParticles3B. Note that the Automatic statement for automatically deriving the decays of all non-SM particles does not work in this case as SARAH doesn't differ between SUSY or Non-SUSY particle in order to make the output as generic as possible. Therefore, the lists of the decaying particles have to be supplied manually.

If all SM couplings (gauge and Yukawa couplings) as well as VEVs for up- and down Higgs are present in the considered model, SARAH calculates the running gauge and Yukawa couplings at the SUSY scale. For this purpose, it uses the $\overline{\text{DR}}$ masses from [41] and 2-loop SM RGEs.

11.2.4 Models with another gauge group at the SUSY scale

Some SUSY models have the distinct feature that they gauge group at the SUSY scale doesn't consist of $SU(3)_C \times SU(2)_L \times U(1)_Y$. This is for instance the case in left-right models in which $U(1)_R \times U(1)_{B-L}$ is just broken around the SUSY scale (see e.g. Ref. [42] and references therein). This special feature has to be taken into account in some calculations. For instance, the calculation of the running couplings at the EW scale assumes that the hypercharge is present.
Therefore, SARAH has to create in this kind of models an auxiliary variable for the hypercharge coupling called gYaux. This is done by adding

```
AuxiliaryHyperchargeCoupling = True;
```

in SPheno.m. In addition, the user has to define a relation between the existing gauge couplings and the hypercharge coupling. For instance, in the model mentioned above, this relation reads

```
ExpressionAuxHypercharge =Sqrt[(gBL*gR - gBLgR*gRgBL)^2/((gBLgR - gR)^2 + (gBL - gRgBL)^2)];
```

Note, if kinetic mixing is neglected, this reduces to the more familiar form of $\sqrt{g_{BL}^2 g_R/(g_R^2+g_{BL}^2)}$. When setting the boundary conditions to relate the gauge couplings, one has to make sure that always the relations for the not GUT-normalized values are used. For instance,

```
BoundaryEWSBScale = {
  {gYauxt, Sqrt[5/3]*gYaux},
  {gR, g1RBLFactor*gYauxt},
  {gRgBLt, 1*gRgBL},
  {gBLgRt, Sqrt[2/3]*gBLgR},
  {gBLt, (5 gBLgRt gR gRgBLt - Sqrt[6] gRgBLt gYauxt^2
      + Sqrt[(3 gBLgRt^2 - 2 Sqrt[6] gBLgRt gR + 2 gR^2) *
        (5 (gR^2 + gRgBLt^2) - 3 gYauxt^2) gYauxt^2])/(5 gR^2 - 3 gYauxt^2)},
  {gBL, Sqrt[3/2]*gBLt},
  {TanBetaR, TanBetaRinput},
  {vChiR, vR*TanBetaR/Sqrt[1 + TanBetaR^2]},
  {vChiRb, vR*1/Sqrt[1 + TanBetaR^2]}};
```

Here, g1RBLFactor is the ratio of g_R/g_Y^{aux} which has been calculated in the iteration before

```
BoundaryEWSBScaleRunningDown = {
  {gBLt, gBL*Sqrt[2/3]},
  {gRgBLt, 1*gRgBL},
  {gBLgRt, Sqrt[2/3]*gBLgR},
  {gYaux, Sqrt[5*(gBLt*gR - gBLgRt*gRgBLt)^2/(3*(gBLt^2 + gBLgRt^2)
    + 2*(gR^2 + gRgBLt^2) - 2*Sqrt[6]*(gR*gBLgRt + gBLt*gRgBLt))]},
  {g1RBLFactor, gR/gYaux},
  {gYaux, Sqrt[3/5]*gYaux}
 };
```

11.3 Writing input files for HiggsBounds with SPheno

HiggsBounds [43, 44] is a tool to test the neutral and charged Higgs sectors against the current exclusion bounds from the Higgs searches at the LEP, Tevatron and LHC experiments. The required input consists of the masses, width and branching ratios of the Higgs fields. In addition, it is either possible to provide full information about production cross sections in e^+e^- and pp collisions, or to work with a set of effective couplings.
HiggsBounds can be downloaded from

http://projects.hepforge.org/higgsbound

Although HiggsBounds supports a LesHouches interface, this functionality is restricted so far to at most 5 neutral Higgs fields, and therefore, we don't use it. Instead, a SPheno module generated by SARAH can create all necessary input files needed for a run of HiggsBounds with effective couplings (option whichinput=effC). To write these file, the flag 76 in the block SPhenoInfo in the LesHouches input file has to be set to 1.

```
[frame=shadowbox]
 Block SPhenoInput #
 76 1 # Write files for HiggsBounds
```

Unfortunately, we can not provide all information which can be used by HiggsBounds to check the constraints. So far, the effective couplings $H \to \gamma Z$ and $H \to ggZ$ are not calculated by SPheno and therefore they are set to zero in the output. In addition, as already mentioned, the SPheno version created by SARAH does not include the calculations of e^+e^- cross sections. For this reason, also the LEP production cross section of charged Higgs fields is not available for SPheno and it sets this value also to 0 in the output. However, it is of course possible to calculate this cross section as well as all other cross sections needed for the options (whichinput=hadr or whichinput=part) of HiggsBounds using CalcHep, Madgraph or WHIZARD.
The following files are created by SPheno

- MH_GammaTot.dat:
  Masses and widths of all neutral Higgs fields
- MHplus_GammaTot.dat:
  Masses and widths of all charged Higgs fields
- BR_H_NP.dat:
  Branching ratios for neutral Higgs fields into invisible and other neutral Higgs fields.
- BR_Hplus.dat:
  Branching ratios of charged Higgs fields into $c\bar{s}$, $c\bar{b}$ and $\tau\bar{\nu}$ final states
- BR_t.dat:
  Branching ratios of top quark into bottom quark and a W boson or charged Higgs

- effC.dat: effective couplings of neutral Higgs fields to $s\bar{s}$, $c\bar{c}$, $b\bar{b}$, $t\bar{t}$, $\mu\bar{\mu}$, $\tau\bar{\tau}$, $\gamma\gamma$, gg, γZ, ggZ as well as to all other neutral Higgs fields.
- LEP_HpHm_CS_ratios.dat:
  LEP production cross section of charged Higgs. (set to zero, see above).

To run HiggsBounds, use in the HiggsBounds directory

```
[frame=none]
> ./HiggsBounds LandH effC [NN] [NC] '[SPheno Directory]'
```

where [NN] has to be replaced by the number of neutral Higgs particle, and [NC] by the number of the charged ones. For more information, see also the HiggsBounds manual. The results of the check are written to the file HiggsBounds_results.dat in the directory from which HiggsBounds has been called.

Chapter 12

Output for `Vevacious`

`Vevacious` [45] provides the possibility to check the one-loop effective potential of a given model and parameter point for the global minimum. In these checks the possibility of dangerous vacuum expectation values (VEVs) of charged or colored scalars can be taken into account and the life-time of meta-stable vacua can be calculated using `Cosmotransitions` [46]. For details about the input for of `Vevacious` we refer the interested reader to Ref. [45].

`MakeVevacious[Options];`

The possible options are:

- `ComplexParameters`, Value: list of parameters, Default: {}:
  By default, all parameters are assumed to be real when writing the `Vevacious` input files. However, the user can define those parameters which should be treated as complex.
- `IgnoreParameters`, Value: list of parameters, Default: {}:
  The user can define a list of parameters which should be set to zero when writing the `Vevacious` input.
- `OutputFile`, Value: String, Default `MyModel.vin`, where `MyModel` here is the same name as is given in `Start["MyModel"];` above:
  The name used for the output file.
- `Scheme`, Value: `DR` or `MS`, Default: `DR` for SUSY models, `MS` for non-SUSY models:
  Defines if as renormalization scheme $\overline{\mathrm{DR}}'$ or $\overline{\mathrm{MS}}$ should be used.

The first two options allow to treat parameters differently in the `Vevacious` output as defined in the `SARAH` model file.

Example: MSSM with stau VEVs In general, three changes are always necessary to include stau VEVs in a model.

a) Defining the particles which can get a VEV

```
DEFINITION[EWSB][VEVs]=
{{SHd0, {vdR, 1/Sqrt[2]}, {sigmad, I/Sqrt[2]},{phid,1/Sqrt[2]}},
 {SHu0, {vuR, 1/Sqrt[2]}, {sigmau, I/Sqrt[2]},{phiu,1/Sqrt[2]}},
 {SeL,  {vLR[3], 1/Sqrt[2]}, {vLI[3], I/Sqrt[2]},
                                       {sigmaL, I/Sqrt[2]},{phiL,1/Sqrt[2]}},
   {SeR,   {vER[3], 1/Sqrt[2]}, {vEI[3], I/Sqrt[2]},
                                       {sigmaR, I/Sqrt[2]},{phiR,1/Sqrt[2]}},
   {SHdm,  {0, 0}, {sigmaM, I/Sqrt[2]},{phiM,1/Sqrt[2]}},
   {SHup,  {0, 0}, {sigmaP, I/Sqrt[2]},{phiP,1/Sqrt[2]}},
   {SvL,   {0, 0}, {sigmaV, I/Sqrt[2]},{phiV,1/Sqrt[2]}}
};
```

where it is important that the VEVs have names that are at least two characters long.

b) Changing the rotation of the vector bosons:

```
DEFINITION[EWSB][GaugeSector] =
{
  {{VB,VWB[1],VWB[2],VWB[3]},{VB1,VB2,VB3,VB4},ZZ},
  {{fWB[1],fWB[2],fWB[3]},{fWm,fWp,fW0},ZfW}  };
```

c) Changing the rotation of matter fields:

```
DEFINITION[EWSB][MatterSector]=
{     ...
      {{phid, phiu,phiM,phiP,phiV,phiL,phiR}, {hh, ZH}},
      {{sigmad, sigmau,sigmaM,sigmaP,sigmaV,sigmaL,sigmaR}, {Ah, ZA}},
      {{fB, fW0, FHd0, FHu0, FvL, FeL, conj[FeR],fWm, FHdm, fWp, FHup}, {L0, ZN}},
      ...          };
```

Chapter 13

Other output

13.1 LHPC spectrum plotter

The LHPC spectrum plotter is a small but handy tool to produce plots of the SUSY mass spectrum based on the information given in a SLHA output file [47].

For the output it is necessary to provide a second control file in addition to the SLHA spectrum file. The control file includes information about the paths to the necessary shell tools (`gnuplot`, `latex`, `dvips`, `ps2eps`, `rm`, `mv`) and the LaTeX name associated with a PDG number. In addition, the color and column used for the different particles are defined in that file. `SARAH` can provide such a file which works nicely together with the spectrum file written by a `SPheno` module also created by `SARAH`. By default it assumes the standard paths under Linux, while the color and column of each particle can be defined in `particles.m` using the new option `LHPC`. For instance, to put the gluino in the fourth column and to use purple for the lines, the entry reads

```
 {{   Description -> "Gluino",
             ...
             LaTeX -> "\\tilde{g}",
             LHPC -> {4, "purple"},
             ... }},
```

As name for the colors all available colors in `gnuplot` can be used. The control file for a given set of eigenstates of the initialized model is written via

```
MakeLHPCstyle[$EIGENSTATES];
```

and saved in the directory

```
$SARAH/Output/$MODEL/$EIGENSTATES/LHPC/
```

It is used together with a spectrum file to create the figure by the shell command

```
./LhpcSpectrumPlotter.exe SPheno.spc.$MODEL LHPC_$MODEL_Control.txt
```

13.2 All at once

To generate the entire output for `SPheno`, `CalcHep`, `WHIZARD`, `FeynArts` as well as model files in the UFO format and the LaTeX file, use

`MakeAll[Options];`

The options are

- `ReadLists`, Values: `True/False`, Default: `False`: Should results for earlier runs are used.
- `IncludeSPheno`, Values: `True/False`, Default: `True`: Includes/excludes the `SPheno` output
- `IncludeFeynArts`, Values: `True/False`, Default: `True`: Includes/excludes the `FeynArts` output
- `IncludeCalcHep`, Values: `True/False`, Default: `True`: Includes/excludes the `CalcHep` output
- `IncludeWHIZARD`, Values: `True/False`, Default: `True`: Includes/excludes the `WHIZARD` output
- `IncludeUFO`, Values: `True/False`, Default: `True`: Includes/excludes the UFO output
- `IncludeTeX`, Values: `True/False`, Default: `True`: Includes/excludes the LaTeX output

Chapter 14

Numerical values

SARAH offers some basic routines for working with numerical values of parameters and calculating mixing matrices and masses.

14.1 Adding numerical values

There are different ways to define numerical values for a parameter: they can just be added in the parameter file as described in sec. 4.2 or the values from a LesHouches spectrum file can be used. To read automatically a LesHouches file when evaluating the command `Start`, add

```
SpectrumFile="Name of File"
```

to the model file. The spectrum file must be in the same directory as the model file.
It is also possible to read a spectrum file afterwards via the command

```
ReadSpectrum["Spectrum File"]
```

An additional possibility to add a numerical value during the work with SARAH is to use

```
SetParameterValue[parameter,value];
```

`parameter` is the name of the parameter and `value` the numerical Value. Of course, it is also possible to remove easily a value. For this purpose the command

```
DeleteParameterValue[parameter];
```

is used.

14.2 Calculate Mixing Matrices

After numerical values for all free parameters are defined, the mass eigenstates and entries of mixing matrices can be calculated by

```
CalcMatrices;
```

The eigenvalues and mixing matrices of each mass matrix are saved in two additional variables:

a) Eigenvalues: the eigenvalues of the mass matrix are saved in

```
Mass <> $PARTICLE
```

b) The numerical value of the mixing matrix are saved in

```
"Name of Mixing Matrix" <> Num
```

Negative Squared Masses It could happen that for some parameter points some eigenvalues of the mass matrices for scalars are getting negative. If this happens, there will appear a warning in the output and the variable `warning` is set to `True`.

Example After using `CalcMatrices`, the numerical value for the down-squark mixing matrix are saved in `MDNum` and the corresponding squark masses are saved in `MassSd`.

14.3 Calculate Numerical Values

To get the numerical value for a term

```
NumericalValue[x];
```

is used. `x` can be e.g. the entry of a mass matrix or a vertex

Examples

a) The numerical value for the Higgs mass matrix is calculated with

```
NumericalValue[MassMatricesFullEWSB[[4]]];
```

b) The numerical value for the interaction between the photon and down squarks is calculated with

```
NumericalValue[Vertex[{VP,Sd,conj[Sd]}][[2,1]]];
```

Appendix A

Calculation of Group Factors

SARAH supports not only chiral superfields in the fundamental representation but in any irreducible representation of $SU(N)$. In most cases, it is possible to fix the transformation properties of the chiral superfield by stating the corresponding dimension D. If the dimension is not unique, also the Dynkin labels are needed. For calculating kinetic terms and D-terms, it is necessary to derive from representation the corresponding generators. Furthermore, the eigenvalues C_2 of the quadratic Casimir for any irreducible representation r

$$T^aT^a\phi(r) = C_2(r)\phi(r) \tag{A.1}$$

as well as the Dynkin index I

$$Tr(T^aT^b)\phi(r) = I\delta_{ab}\phi(r) \tag{A.2}$$

are needed for the calculation of the RGEs. All of that is derived by SARAH due to the technique of Young tableaux. The following steps are evolved:

a) The corresponding Young tableaux fitting to the dimension D is calculated using the hook formula:

$$D = \Pi_i \frac{N + d_i}{h_i} \tag{A.3}$$

d_i is the distance of the i. box to the left upper corner and h_i is the hook of that box.

b) The vector for the highest weight Λ in Dynkin basis is extracted from the tableaux.

c) The fundamental weights for the given gauge group are calculated.

d) The value of $C_2(r)$ is calculated using the Weyl formula

$$C_2(r) = (\Lambda, \Lambda + \rho) \ . \tag{A.4}$$

ρ is the Weyl vector.

e) The Dynkin index $I(r)$ is calculated from $C_2(r)$. For this step, the value for the fundamental representation is normalized to $\frac{1}{2}$.

$$I(r) = C_2(r)\frac{D(r)}{D(G)} \tag{A.5}$$

With $D(G)$ as dimension of the adjoint representation.

f) The number of co- and contra-variant indices is extracted from the Young tableaux. With this information, the generators are written as tensor product.

The user can calculate this information independently from the model using the new command

```
CheckIrrepSUN[Dim,N]
```

`Dim` is the dimension of the irreducible representation and `N` is the dimension of the $SU(N)$ gauge group. The result is a vector containing the following information: (i) repeating the dimension of the field, (ii) number of covariant indices, (iii) number of contra-variant indices, (iv) value of the quadratic Casimir $C_2(r)$, (v) value of the Dynkin index $I(r)$, (vi) Dynkin labels for the highest weight.

Examples

a) Fundamental representation The properties of a particle, transforming under the fundamental representation of $SU(3)$ are calculated via `CheckIrrepSUN[3,3]`. The output is the well known result

```
{3, 1, 0, 4/3, 1/2, {1, 0}}
```

b) Adjoint representation The properties of a field transforming as **24** of $SU(5)$ are calculated by `CheckIrrepSUN[24,5]` . The output will be

```
{24, 1, 1, 5, 5, {1, 0, 0, 1}}
```

c) Different representations with the same dimension The **70** under $SU(5)$ is not unique. Therefore, `CheckIrrepSUN[{70, {0, 0, 0, 4}}, 5]` returns

```
{70, 0, 4, 72/5, 42, {0, 0, 0, 4}}
```

while `CheckIrrepSUN[{70, {2, 0, 0, 1}}, 5]` leads to

```
{70, 2, 1, 42/5, 49/2, {2, 0, 0, 1}}
```

Appendix B

Calculation of the Lagrangian of supersymmetric models and deriving the vertices

B.1 The supersymmetric Lagrangian

We describe in this section the calculation of the complete Lagrangian for a supersymmetric model based on the superpotential and the gauge structure.

Interactions of chiral superfields If we call the superpotential for a given theory W and use ϕ_i for the scalar and ψ_i for the fermionic component of a chiral supermultiplet, the matter interactions can by derived by

$$\mathcal{L}_Y = -\frac{1}{2} W^{ij} \psi_i \psi_j + \text{h.c.}\,, \qquad \mathcal{L}_F = F^{*i} F_i + \text{h.c.} \tag{B.1}$$

with

$$W^{ij} = \frac{\delta^2}{\delta\phi_i \delta\phi_j} W \qquad \text{and} \qquad F^i = -W^{*i} = \frac{\delta W}{\delta \phi_i}\,. \tag{B.2}$$

The first term of eq. (B.1) describes the interaction of two fermions with one scalar, while the second term forms the so called F-terms which describe four-scalar interactions.

Interactions of vector superfields We name the spin-$\frac{1}{2}$ component of a vector supermultiplet λ and the spin-1 component A^μ. The most general Lagrangian only involving these fields is

$$\mathcal{L} = -\frac{1}{4} F^a_{\mu\nu} F^{\mu\nu a} - i \lambda^{\dagger a} \bar{\sigma}^\mu D_\mu \lambda^a \tag{B.3}$$

with the field strength

$$F^a_{\mu\nu} = \partial_\mu A^a_\nu - \partial_\nu A^a_\mu + g f^{abc} A^b_\mu A^c_\nu\,, \tag{B.4}$$

and the covariant derivative

$$D_\mu \lambda^a = \partial_\mu \lambda^a + g f^{abc} A^b_\mu \lambda^c\,. \tag{B.5}$$

Here, f^{abc} is the structure constant of the gauge group. Plugging eq. (B.4) in the first term of eq. (B.3) leads to self-interactions of three and four gauge bosons

$$\mathcal{L}_V = -\frac{1}{4}(\partial_\mu A^a_\nu - \partial_\nu A^a_\mu) g f^{abc} A^{\mu,b} A^{\nu,c} - \frac{1}{4} g^2 (f_{abc} A^b_\mu A^c_\nu)(f^{ade} A^{\mu,e} A^{\nu,e}) \; . \tag{B.6}$$

The second term of eq. (B.3) describes the interactions between vector bosons and gauginos.

Supersymmetric gauge interactions The parts of the Lagrangian with both chiral and vector superfields are the kinetic terms for the fermions and scalars

$$\mathcal{L}_{kin} = -D^\mu \phi^{*i} D_\mu \phi_i - i\psi^{\dagger i} \bar{\sigma}^\mu D_\mu \psi_i \tag{B.7}$$

as well as the interaction between a gaugino and a matter fermion and scalar

$$\mathcal{L}_{GFS} = -\sqrt{2} g (\phi^* T^a \psi) \lambda^a + \text{h.c.} \; . \tag{B.8}$$

Here, T^a are the fundamental generators of the gauge group. Furthermore, the covariant derivatives are

$$D_\mu \phi_i = \partial_\mu \phi_i - i g A^a_\mu (T^a \phi)_i \; , \tag{B.9}$$

$$D_\mu \phi^{*i} = \partial_\mu \phi^{*i} + i g A^a_\mu (\phi^* T^a)^i \; , \tag{B.10}$$

$$D_\mu \psi_i = \partial_\mu \psi_i - i g A^a_\mu (T^a \psi)_i \; , \tag{B.11}$$

In addition, the D-Terms are defined by

$$\mathcal{L}_D = \frac{1}{2} D^a D^a \; . \tag{B.12}$$

The solution of the equations of motion for the auxiliary fields leads to

$$D^a = -g(\phi^* T^a \phi) \; . \tag{B.13}$$

Soft-breaking terms SUSY must be a broken. This can be parametrized by adding soft-breaking terms to the Lagrangian. The possible terms are the mass terms for all scalar matter fields and gauginos

$$\mathcal{L}_{SB} = -m^2_{\phi_i} \phi_i \phi^*_i - \frac{1}{2} M_{\lambda_i} \lambda_i \lambda_i \tag{B.14}$$

as well as soft-breaking interaction corresponding to the superpotential terms

$$\mathcal{L}_{Soft,W} = T \phi_i \phi_j \phi_k + B \phi_i \phi_j + S \phi_i \; . \tag{B.15}$$

Dirac gauginos Using the just described method and input it was possible to implement the MSSM and many extensions of it in SARAH. However, in the last years another structure had been become popular in SUSY model building: Dirac mass terms for gauginos. If one demands for instance a continuous R-symmetry this forbids to write down Majorana mass terms of the gauginos. However, Dirac mass terms between a gaugino and a chiral superfields in the adjoint representation might be allowed. Such a mass term between a vector and a chiral superfield expanded in component fields leads to two physical relevant terms in the Lagrangian [48]:

$$\mathcal{L}_{GF} = -m_D \lambda_a \Psi_a + \sqrt{2} m_D \phi_a D_a \tag{B.16}$$

The first term is the Dirac mass term of the gauginos, the second term is an additional D-term contribution.
Models with Dirac mass terms are fully supported since version 3.2.0 of SARAH. Also the corresponding RGEs are calculated at two-loop level.

Gauge fixing terms and ghost interactions The Lagrangian of a theory without further restrictions is invariant under a general gauge transformation. This invariance leads to severe problems in the quantization of the theory as can be seen in the divergence of functional integrals. Therefore, it is in necessary to add gauge fixing terms to break this gauge invariance.
The general form of the gauge fixing Lagrangian is

$$\mathcal{L}_{GF} = -\frac{1}{2}\sum_a |f(x)^a|^2 \; . \tag{B.17}$$

f_a can be a function of partial derivatives of a gauge boson and a Goldstone boson. The corresponding ghost terms of the ghost fields $\bar{\eta}$ and η are

$$\mathcal{L}_{Ghost} = -\bar{\eta}_a(\delta f^a) \; . \tag{B.18}$$

Here, δ assigns the operator for a BRST transformation. For an unbroken gauge symmetry, the gauge fixing terms in the often chosen R_ξ-gauge are

$$\mathcal{L}_{GF} = -\frac{1}{2R_\xi}\sum_a \left(\partial^\mu V^a_\mu\right)^2 \; . \tag{B.19}$$

Here, V_μ are the gauge boson of the unbroken gauge group. It is often common to choose a distinct value for R_ξ. The most popular gauges are the unitary gauge $R_\xi \to \infty$ and the Feynman-'t Hooft-gauge $R_\xi = 1$. For broken symmetries, the gauge fixings terms are chosen in a way that the mixing terms between vector bosons and scalars disappear from the Lagrangian. Therefore, the common choice for the gauge fixing Lagrangian for theories with the standard model gauge sector after EWSB is

$$\mathcal{L}_{GF,R_\xi} = -\frac{1}{2\xi_\gamma}\left(\partial^\mu \gamma_\mu\right)^2 - \frac{1}{2\xi_Z}\left(\partial^\mu Z_\mu + \xi_Z M_Z G^0\right)^2 + -\frac{1}{\xi_{W^+}}\left(\partial^\mu W^+_\mu + \xi_{W^+} M_W G^+\right)^2 \; . \tag{B.20}$$

Here, G^0 and G^+ are the Goldstone bosons, which build the longitudinal component of the massive vector bosons.

B.2 Deriving the vertices

SARAH calculates the vertices as partial derivatives with respect to the external fields and applies afterwards the vacuum conditions

$$c\langle\frac{\partial^n L}{\partial\phi_i \ldots \partial\phi_j}\rangle \tag{B.21}$$

The numerical coefficient c depends on the generic type of the interactions and takes the value i if one or more scalars are involved and is 1 in the other cases. Furthermore, we define a momentum flow by the replacement

$$\partial_\mu \phi \to -i\mathrm{p}_\mu(\phi) \tag{B.22}$$

Appendix C

Parts of the Lagrangian in `SARAH`

By default, `SARAH` writes down the most general Lagrangian derived by the rules shown in app. B.1. However, there are also flags to suppress specific terms in the Lagrangian. This might especially interesting for models with a R-symmetric which forbids Majorana mass terms for gauginos as well as trilinear soft-breaking terms, or it can be used for the implementation of non-SUSY models. The following options exist:

- `AddTterms = True/False;`, default: `True`, includes/excludes trilinear softbreaking couplings
- `AddBterms = True/False;`, default: `True`, includes/excludes bilinear softbreaking couplings
- `AddLterms = True/False;`, default: `True`, includes/excludes linear softbreaking couplings
- `AddSoftScalarMasses = True/False;`, default: `True`, includes/excludes soft-breaking scalar masses
- `AddSoftGauginoMasses = True/False;`, default: `True`, includes/excludes Majorana masses for gauginos
- `AddDiracGauginos = True/False;`, default: `False`, includes/excludes Dirac masses for gauginos
- `AddSoftTerms = True/False;`, default: `True`, includes/excludes all soft-breaking terms
- `AddDterms = True/False;`, default: `True`, includes/excludes all D-terms
- `AddFterms = True/False;`, default: `True`, includes/excludes all F-terms

`SARAH` saves the different parts of the Lagrangian for the different eigenstates in the following variables:

- `LagSV[$EIGENSTATES]`: Parts with scalars and vector bosons (= kinetic terms for scalars)
- `LagFFV[$EIGENSTATES]`: Parts with fermions and vector bosons (= kinetic terms of scalars)
- `LagSSSS[$EIGENSTATES]`: Parts with only scalars (= scalar potential)
- `LagFFS[$EIGENSTATES]`: Parts with fermions and scalars
- `LagVVV[$EIGENSTATES]`: Parts with three vector bosons
- `LagVVVV[$EIGENSTATES]`: Parts with four vector bosons
- `LagGGS[$EIGENSTATES]`: Parts with ghosts and scalars

- LagGGV[$EIGENSTATES]: Parts with ghosts and vector bosons
- LagSSA[$EIGENSTATES]: Parts with scalars and auxiliary fields (only needed for CalcHep output)

In addition, for an effective theory, there might exist

- LagSSSSSS[$EIGENSTATES]: Dimension 6 operators with only scalars
- LagSSSVVV[$EIGENSTATES]: Dimension 6 operators with scalars and fermions
- LagFFFF[$EIGENSTATES]: Dimension 6 operators with only fermions
- LagFFSS[$EIGENSTATES]: Dimension 5 operators with fermions and scalars
- LagFFVV[$EIGENSTATES]: Dimension 6 operators with fermions and vector bosons

Moreover, the different results of the calculation of the Lagrangian in gauge eigenstates are also saved separately. The variable names are:

- Superpotential: Superpotential
- Fermion - scalar interactions coming from the superpotential: Wij
- F-Terms: FTerms
- Scalar soft-breaking masses: SoftScalarMass
- Gaugino masses: SoftGauginoMass
- Soft-breaking couplings: SoftW
- Kinetic terms for scalars: KinScalar
- Kinetic terms for fermions: KinFermion
- D-Terms: DTerms
- Interactions between gauginos and a scalar and a fermion: FSGaugino
- Trilinear self-interactions of gauge bosons: GaugeTri
- Quartic self-interactions of gauge bosons: GaugeQuad
- Interactions between vector bosons and gauginos: BosonGaugino

Furthermore, the additional interactions and the redefinition of existing interactions are saved in LagRedefinition.

Appendix D

Gauge anomalies

Before SARAH starts the calculation of the Lagrangian it checks the model for the different triangle anomalies. These anomalies can involve diagrams with three external gauge bosons belonging to the same $U(1)$ or $SU(N)$ gauge group. To be anomaly free the sum over all internal fermions has to vanish

$$U(1)_i^3 \quad : \quad \sum_n {Y_n^i}^3 = 0 \; , \tag{D.1}$$

$$SU(N)_i^3 \quad : \quad \sum_n \mathrm{Tr}(T_n^i T_n^i T_n^i) = 0 \; . \tag{D.2}$$

We label the different gauge groups with the indices i, j, k. Y_n^i is the charge of particle n under the abelian gauge group i while T_n^i is the generator with respect to a non-abelian gauge group.
Combinations of two different gauge groups are possible, if one group is an $U(1)$. Hence, another condition for the absence of anomalies is

$$U(1)_i \times SU(N)_j^2 : \sum_n Y_n^i \, \mathrm{Tr}(T_n^j T_n^j) = 0 \; . \tag{D.3}$$

If more than one $U(1)$ gauge group are present, anomalies can be generated by two or three different $U(1)$ gauge bosons as external fields, too. Therefore, it has to be checked, that

$$U(1)_i \times U(1)_j^2 \quad : \quad \sum_n Y_n^i {Y_n^j}^2 = 0 \; , \tag{D.4}$$

$$U(1)_i \times U(1)_j \times U(1)_k \quad : \quad \sum_n Y_n^i Y_n^j Y_n^k = 0 \tag{D.5}$$

holds. In addition, it has to be checked that there is an even number of $SU(2)$ doublets. This is the necessary for a model in order to be free of the Witten anomaly [49]. If one condition is not fulfilled, a warning is given by SARAH but the model can be evaluated anyway.

Appendix E

Conventions and generic expressions

E.1 Renormalization group equations

E.1.1 Generic form of β functions

We summarize in this section the used equations for the calculation of the one- and two-loop RGEs in SARAH. These equations are extensively discussed in literature, see e.g. [22, 50, 51, 52, 53, 54, 55, 56, 50, 57]. For a general $N = 1$ supersymmetric gauge theory with superpotential

$$W(\phi) = L_i \phi_i + \frac{1}{2}\mu^{ij}\phi_i\phi_j + \frac{1}{6}Y^{ijk}\phi_i\phi_j\phi_k \ , \tag{E.1}$$

the soft SUSY-breaking scalar terms are given by

$$V_{\text{soft}} = \left(S^i\phi_i + \frac{1}{2}b^{ij}\phi_i\phi_j + \frac{1}{6}h^{ijk}\phi_i\phi_j\phi_k + \text{c.c.}\right) + (m^2)^i{}_j\phi_i\phi_j^* + \frac{1}{2}M_\lambda\lambda_a\lambda_a \ . \tag{E.2}$$

The anomalous dimensions are given by

$$\gamma_i^{(1)j} = \frac{1}{2}Y_{ipq}Y^{jpq} - 2\delta_i^j g^2 C(i) \ , \tag{E.3}$$

$$\begin{aligned}\gamma_i^{(2)j} =& -\frac{1}{2}Y_{imn}Y^{npq}Y_{pqr}Y^{mrj} + g^2 Y_{ipq}Y^{jpq}[2C(p) - C(i)] \\ &+ 2\delta_i^j g^4[C(i)S(R) + 2C(i)^2 - 3C(G)C(i)] \ ,\end{aligned} \tag{E.4}$$

and the β-functions for the gauge couplings are given by

$$\beta_g^{(1)} = g^3\left[S(R) - 3C(G)\right] \ , \tag{E.5}$$

$$\beta_g^{(2)} = g^5\left\{-6[C(G)]^2 + 2C(G)S(R) + 4S(R)C(R)\right\} - g^3 Y^{ijk}Y_{ijk}C(k)/D(G) \ . \tag{E.6}$$

Here, $C(i)$ is the quadratic Casimir for a specific superfield and $C(R), C(G)$ are the quadratic Casimirs for the matter and adjoint representations, respectively. $D(G)$ is the dimension of the adjoint representation. The β-functions for the superpotential parameters can be obtained by using superfield technique. The obtained expressions are

$$\beta_W^{ijkl} = W^{ijkp}\left[\frac{1}{16\pi^2}\gamma_p^{(1)l}+\frac{1}{(16\pi^2)^2}\gamma_p^{(2)l}\right]+(l\leftrightarrow i)+(l\leftrightarrow j)+(l\leftrightarrow k)\ , \tag{E.7}$$

$$\beta_Y^{ijk} = Y^{ijp}\left[\frac{1}{16\pi^2}\gamma_p^{(1)k}+\frac{1}{(16\pi^2)^2}\gamma_p^{(2)k}\right]+(k\leftrightarrow i)+(k\leftrightarrow j)\ , \tag{E.8}$$

$$\beta_\mu^{ij} = \mu^{ip}\left[\frac{1}{16\pi^2}\gamma_p^{(1)j}+\frac{1}{(16\pi^2)^2}\gamma_p^{(2)j}\right]+(j\leftrightarrow i)\ , \tag{E.9}$$

$$\beta_L^{i} = L^{p}\left[\frac{1}{16\pi^2}\gamma_p^{(1)i}+\frac{1}{(16\pi^2)^2}\gamma_p^{(2)i}\right]\ . \tag{E.10}$$

The expressions for trilinear, soft-breaking terms are

$$\frac{d}{dt}h^{ijk} =\frac{1}{16\pi^2}\left[\beta_h^{(1)}\right]^{ijk}+\frac{1}{(16\pi^2)^2}\left[\beta_h^{(2)}\right]^{ijk}\ , \tag{E.11}$$

with

$$\begin{aligned}\left[\beta_h^{(1)}\right]^{ijk} =&\frac{1}{2}h^{ijl}Y_{lmn}Y^{mnk}+Y^{ijl}Y_{lmn}h^{mnk}-2\left(h^{ijk}-2MY^{ijk}\right)g^2C(k)\\ &+(k\leftrightarrow i)+(k\leftrightarrow j)\ ,\end{aligned} \tag{E.12}$$

$$\begin{aligned}\left[\beta_h^{(2)}\right]^{ijk} =&-\frac{1}{2}h^{ijl}Y_{lmn}Y^{npq}Y_{pqr}Y^{mrk}\\ &-Y^{ijl}Y_{lmn}Y^{npq}Y_{pqr}h^{mrk}-Y^{ijl}Y_{lmn}h^{npq}Y_{pqr}Y^{mrk}\\ &+\left(h^{ijl}Y_{lpq}Y^{pqk}+2Y^{ijl}Y_{lpq}h^{pqk}-2MY^{ijl}Y_{lpq}Y^{pqk}\right)g^2\left[2C(p)-C(k)\right]\\ &+\left(2h^{ijk}-8MY^{ijk}\right)g^4\left[C(k)S(R)+2C(k)^2-3C(G)C(k)\right]\\ &+(k\leftrightarrow i)+(k\leftrightarrow j)\ .\end{aligned} \tag{E.13}$$

For the bilinear soft-breaking parameters, the expressions read

$$\frac{d}{dt}b^{ij} =\frac{1}{16\pi^2}\left[\beta_b^{(1)}\right]^{ij}+\frac{1}{(16\pi^2)^2}\left[\beta_b^{(2)}\right]^{ij}\ , \tag{E.14}$$

with

$$\begin{aligned}\left[\beta_b^{(1)}\right]^{ij} =&\frac{1}{2}b^{il}Y_{lmn}Y^{mnj}+\frac{1}{2}Y^{ijl}Y_{lmn}b^{mn}+\mu^{il}Y_{lmn}h^{mnj}-2\left(b^{ij}-2M\mu^{ij}\right)g^2C(i)\\ &+(i\leftrightarrow j)\ ,\end{aligned} \tag{E.15}$$

$$\begin{aligned}\left[\beta_b^{(2)}\right]^{ij} =&-\frac{1}{2}b^{il}Y_{lmn}Y^{pqn}Y_{pqr}Y^{mrj}-\frac{1}{2}Y^{ijl}Y_{lmn}b^{mr}Y_{pqr}Y^{pqn}\\ &-\frac{1}{2}Y^{ijl}Y_{lmn}\mu^{mr}Y_{pqr}h^{pqn}-\mu^{il}Y_{lmn}h^{npq}Y_{pqr}Y^{mrj}\\ &-\mu^{il}Y_{lmn}Y^{npq}Y_{pqr}h^{mrj}+2Y^{ijl}Y_{lpq}\left(b^{pq}-\mu^{pq}M\right)g^2C(p)\\ &+\left(b^{il}Y_{lpq}Y^{pqj}+2\mu^{il}Y_{lpq}h^{pqj}-2\mu^{il}Y_{lpq}Y^{pqj}M\right)g^2\left[2C(p)-C(i)\right]\\ &+\left(2b^{ij}-8\mu^{ij}M\right)g^4\left[C(i)S(R)+2C(i)^2-3C(G)C(i)\right]\\ &+(i\leftrightarrow j)\ ,\end{aligned} \tag{E.16}$$

Finally, the RGEs for the linear soft-breaking parameters are

$$\frac{d}{dt}S^i = \frac{1}{16\pi^2}\left[\beta_S^{(1)} + \beta_{SD}^{(1)}\right]^i + \frac{1}{(16\pi^2)^2}\left[\beta_S^{(2)} + \beta_{SD}^{(2)}\right]^i \;, \tag{E.17}$$

with

$$\begin{aligned}
\left[\beta_S^{(1)}\right]^i =& \frac{1}{2}Y^{iln}Y_{pln}S^p + L^pY_{pln}h^{iln} + \mu^{ik}Y_{kln}B^{ln} + 2Y^{ikp}(m^2)_p^l\mu_{kl} + h^{ikl}B_{kl} \;, && \text{(E.18)}\\
\left[\beta_S^{(2)}\right]^i =& 2g^2C(l)Y^{ikl}Y_{pkl}S^p - \frac{1}{2}Y^{ikq}Y_{qst}Y^{lst}Y_{pkl}S^p - 4g^2C(l)(Y^{ikl}M - h^{ikl})Y_{pkl}L^p \\
&- \left[Y^{ikq}Y_{qst}h^{lst}Y_{pkl} + h^{ikq}Y_{qst}Y^{lst}Y_{pkl}\right]L^p - 4g^2C(l)Y_{jnl}(\mu^{nl}M - B^{nl})\mu^{ij} \\
&- \left[Y_{jnq}h^{qst}Y_{lst}\mu^{nl} + Y_{jnq}Y^{qst}Y_{lst}B^{nl}\right]\mu^{ij} + 4g^2C(l)(2Y^{ikl}\mu_{kl}|M|^2 - Y^{ikl}B_{kl}M \\
&- h^{ikl}\mu_{kl}M^* + h^{ikl}B_{kl} + Y^{ipl}(m^2)_p^k\mu_{kl} + Y^{ikp}(m^2)_p^l\mu_{kl}) \\
&- \Big[Y^{ikq}Y_{qst}h^{lst}B_{kl} + h^{ikq}Y_{qst}Y^{lst}B_{kl} + Y^{ikq}h_{qst}h^{lst}\mu_{kl} + h^{ikq}h_{qst}Y^{lst}\mu_{kl} \\
&+ Y^{ipq}(m^2)_p^kY_{qst}Y^{lst}\mu_{kl} + Y^{ikq}Y_{qst}Y^{pst}(m^2)_p^l\mu_{kl} + Y^{ikp}(m^2)_p^qY_{qst}Y^{lst}\mu_{kl} \\
&+ 2Y^{ikq}Y_{qsp}(m^2)_t^pY^{lst}\mu_{kl}\Big] \;. && \text{(E.19)}
\end{aligned}$$

as well as additional contributions in the presence of Dirac gaugino mass terms [24]. The new terms are

$$\begin{aligned}
\beta_{SD}^{(1)} =& 2\sqrt{2}g_Ym_D^{aY}\mathrm{Tr}(\mathcal{Y}m^2) + \left[(m_D^2)_{ef}(A^{aef} + MY^{aef}) + Y_{efk}\mu^{ka}(m_D^2)^{ef}\right] && \text{(E.20)}\\
\beta_{SD}^{(2)} =& 2\sqrt{2}g_Ym_D^{aY}\mathrm{Tr}(\mathcal{Y}m^2(4g^2C_2 - Y_2)) + \\
&+ 4(\beta_{m_D}^{(1)}/m_D)_g^f\left[(m_D^2)_{ef}(A^{aeg} + MY^{aeg}) + Y_{efk}\mu^{ka}(m_D^2)^{eg}\right] && \text{(E.21)}
\end{aligned}$$

with

$$(\beta_{m_D}^{(1)}/m_D)_g^f = \frac{1}{2}(Y_2)_g^f + g^2(S_2 - 5C_2(G))\delta_g^f \tag{E.22}$$

With this results, the list of the β-functions for all couplings is complete. Now, we turn to the RGEs for the gaugino masses, squared masses of scalars and vacuum expectation values. The result for the gaugino masses is

$$\frac{d}{dt}M = \frac{1}{16\pi^2}\beta_M^{(1)} + \frac{1}{(16\pi^2)^2}\beta_M^{(2)} \;, \tag{E.23}$$

with

$$\begin{aligned}
\beta_M^{(1)} =& g^2\left[2S(R) - 6C(G)\right]M \;, && \text{(E.24)}\\
\beta_M^{(2)} =& g^4\left\{-24[C(G)]^2 + 8C(G)S(R) + 16S(R)C(R)\right\}M \\
&+ 2g^2\left[h^{ijk} - MY^{ijk}\right]Y_{ijk}C(k)/D(G) \;. && \text{(E.25)}
\end{aligned}$$

The results for the Dirac masses can be expressed in the short from [24]

$$\beta_{m_D^{iA}} = \gamma_j^im_D^{jA} + \frac{\beta_g}{g}m_D^{iA} \tag{E.26}$$

using the anomalous dimension of the chiral superfield in the adjoint representation and the β function of the corresponding gauge coupling.
The one- and two-loop RGEs for the scalar mass parameters read

$$\frac{d}{dt}(m^2)^j_i = \frac{1}{16\pi^2}\left[\beta^{(1)}_{m^2}\right]^j_i + \frac{1}{(16\pi^2)^2}\left[\beta^{(2)}_{m^2}\right]^j_i \ , \tag{E.27}$$

$$\tag{E.28}$$

with

$$\begin{aligned}\left[\beta^{(1)}_{m^2}\right]^j_i =& \frac{1}{2}Y_{ipq}Y^{pqn}(m^2)^j_n + \frac{1}{2}Y^{jpq}Y_{pqn}(m^2)^n_i + 2Y_{ipq}Y^{jpr}(m^2)^q_r \\ &+ h_{ipq}h^{jpq} - 8\delta^j_i MM^\dagger g^2 C(i) + 2g^2 \mathbf{t}^{Aj}_i \mathrm{Tr}[\mathbf{t}^A m^2] \ ,\end{aligned} \tag{E.29}$$

$$\begin{aligned}\left[\beta^{(2)}_{m^2}\right]^j_i =& -\frac{1}{2}(m^2)^l_i Y_{lmn}Y^{mrj}Y_{pqr}Y^{pqn} - \frac{1}{2}(m^2)^j_l Y^{lmn}Y_{mri}Y^{pqr}Y_{pqn} \\ &- Y_{ilm}Y^{jnm}(m^2)^l_r Y_{npq}Y^{rpq} - Y_{ilm}Y^{jnm}(m^2)^r_n Y_{rpq}Y^{lpq} \\ &- Y_{ilm}Y^{jnr}(m^2)^l_n Y_{pqr}Y^{pqm} - 2Y_{ilm}Y^{jln}Y_{npq}Y^{mpr}(m^2)^q_r \\ &- Y_{ilm}Y^{jln}h_{npq}h^{mpq} - h_{ilm}h^{jln}Y_{npq}Y^{mpq} \\ &- h_{ilm}Y^{jln}Y_{npq}h^{mpq} - Y_{ilm}h^{jln}h_{npq}Y^{mpq} \\ &+ \Big[(m^2)^l_i Y_{lpq}Y^{jpq} + Y_{ipq}Y^{lpq}(m^2)^j_l + 4Y_{ipq}Y^{jpl}(m^2)^q_l + 2h_{ipq}h^{jpq} \\ &- 2h_{ipq}Y^{jpq}M - 2Y_{ipq}h^{jpq}M^\dagger + 4Y_{ipq}Y^{jpq}MM^\dagger\Big] g^2\left[C(p) + C(q) - C(i)\right] \\ &- 2g^2\mathbf{t}^{Aj}_i(\mathbf{t}^A m^2)^l_r Y_{lpq}Y^{rpq} + 8g^4\mathbf{t}^{Aj}_i\mathrm{Tr}[\mathbf{t}^A C(r) m^2] \\ &+ \delta^j_i g^4 MM^\dagger\left[24C(i)S(R) + 48C(i)^2 - 72C(G)C(i)\right] \\ &+ 8\delta^j_i g^4 C(i)(\mathrm{Tr}[S(r)m^2] - C(G)MM^\dagger) \ .\end{aligned} \tag{E.30}$$

The RGEs for a VEV v^i is proportional to the anomalous dimension of the chiral superfield whose scalar component receives the VEV

$$\frac{d}{dt}v^i = v^p\left[\frac{1}{16\pi^2}\gamma^{(1)i}_p + \frac{1}{(16\pi^2)^2}\gamma^{(2)i}_p\right] \tag{E.31}$$

E.1.2 Direct product of gauge groups

To calculate the RGEs for a direct product of gauge group, the following substitution rules are needed [22]. Note, these replacements are not sufficient in the case of several $U(1)$ gauge groups as discussed in app. E.1.3. For the β functions of gauge couplings and gauginos the rules are

$$g^3C(G) \to g_a^3C(G_a) \;, \tag{E.32}$$

$$g^3S(R) \to g_a^3S_a(R) \;, \tag{E.33}$$

$$g^5C(G)^2 \to g_a^5C(G_a)^2 \;, \tag{E.34}$$

$$g^5C(G)S(R) \to g_a^5C(G_a)S_a(R) \;, \tag{E.35}$$

$$g^5S(R)C(R) \to \sum_b g_a^3g_b^2S_a(R)C_b(R) \;, \tag{E.36}$$

$$16g^4S(R)C(R)M \to 8\sum_b g_a^2g_b^2S_a(R)C_b(R)(M_a+M_b) \;, \tag{E.37}$$

$$g^3C(k)/d(G) \to g_a^3C_a(k)d(G_a) \;. \tag{E.38}$$

For the other β functions, we need

$$g^2C(r) \to \sum_a g_a^2C_a(r) \;, \tag{E.39}$$

$$g^4C(r)S(R) \to \sum_a g_a^4C_a(r)S_a(R) \;, \tag{E.40}$$

$$g^4C(r)C(G) \to \sum_a g_a^4C_a(r)C(G_a) \;, \tag{E.41}$$

$$g^4C(r)^2 \to \sum_a\sum_b g_a^2g_b^2C_a(r)C_b(r) \;, \tag{E.42}$$

$$48g^4MM^\dagger C(i)^2 \to \sum_a\sum_b g_a^2g_b^2C_a(i)C_b(i)(32M_aM_a^\dagger+8M_aM_b^\dagger+8M_bM_a^\dagger) \;, \tag{E.43}$$

$$g^2t_i^{Aj}Tr(t^Am^2) \to \sum_a g_a^2(t_a^A)_i^jTr(t_a^Am^2) \;, \tag{E.44}$$

$$g^2t_i^{Aj}(t^Am^2)_r^lY_{lpq}Y^{rpq} \to \sum_a g_a^2(t_a^A)_i^j(t_a^Am^2)_r^lY_{lpq}Y^{rpq} \;, \tag{E.45}$$

$$g^4t_i^{Aj}Tr(t^4C(r)m^2) \to \sum_a\sum_b g_a^2g_b^2(t_a^A)_i^jTr(t_a^AC_b(r)m^2) \;, \tag{E.46}$$

$$g^4C(i)Tr(S(r)m^2) \to \sum_a g_a^4C_a(i)Tr(S_a(r)m^2) \;. \tag{E.47}$$

E.1.3 Several $U(1)$ gauge groups

In the case of several Abelian gauge groups, it is necessary to use a generalized form of the some terms given in the last section. The complete set of two-loop RGEs has been published in [23], but we use here a notation closer to [58]. First, we define a generalized charge $\mathscr{Y}$ for each chiral superfield r

$$\mathscr{Y}(r)_\alpha = \sum_\beta Y(r)_\beta g_{\beta\alpha}. \tag{E.48}$$

$Y(r)_\beta$ is the charge of the chiral superfield r with respect to the gauge group β. The sum runs over all Abelian gauge groups. Here and in the following we use Greek letters for $U(1)$ gauge groups and Latin ones for $SU(N)$. Using that definition, we can define a generalized Dynkin index for Abelian gauge groups as

$$Q_{\alpha\beta} = \sum_r \mathcal{Y}(r)_\alpha \mathcal{Y}(r)_\beta . \tag{E.49}$$

In addition, we define

$$\tilde{t}^{\alpha i}_j = \delta_{ij}\mathcal{Y}(i)_\alpha \qquad \text{for} \quad U(1)\ , \tag{E.50}$$

$$\tilde{t}^{a i}_j = t^{a i}_j \qquad \text{for} \quad SU(N)\ . \tag{E.51}$$

as well as

$$\tilde{C}_\alpha(r) = \mathcal{Y}(r)_\alpha \mathcal{Y}(r)_\alpha \qquad \text{for} \quad U(1)\ , \tag{E.52}$$

$$\tilde{C}_a(r) = g_a^2 C_a(r) \qquad \text{for} \quad SU(N)\ . \tag{E.53}$$

Now, to write the RGEs of the last section in the case of several $U(1)$ gauge groups, we have to perform the following replacements for gauge couplings and gauginos. Note, we give only the parts involving Abelian gauge groups. The expressions without Abelian gauge groups keep unchanged.

$$g^3 S(R) \to \sum_\gamma g_{\alpha\gamma} Q_{\gamma\beta}\ , \tag{E.54}$$

$$g^5 S(R) C(R) \to \sum_\gamma g_{\alpha\gamma} \sum_r \sum_a \tilde{C}_a(r) \mathcal{Y}(r)_\gamma \mathcal{Y}(r)_\beta \qquad \text{for} \quad U(1)\ , \tag{E.55}$$

$$g^3 C(k) \to \sum_\gamma g_{\alpha\gamma} \mathcal{Y}(k)_\gamma \mathcal{Y}(k)_\beta\ , \tag{E.56}$$

$$g^2 S(R) M \to \sum_\gamma \frac{1}{2}\left(M_{\alpha\gamma} Q_{\gamma\alpha} + Q_{\alpha\gamma} M_{\gamma\beta}\right)\ , \tag{E.57}$$

$$16 g^4 S(R) C(R) M \to 8\left(\sum_r \sum_\gamma \frac{1}{2}\left(M_{\alpha\gamma}\mathcal{Y}(r)_\gamma \mathcal{Y}(r)_\beta + M_{\gamma\beta}\mathcal{Y}(r)_\gamma \mathcal{Y}(r)_\alpha\right)\bar{C}(r) + \sum_r \mathcal{Y}(r)_\alpha \mathcal{Y}(r)_\beta \bar{C}^M(r)\right)\ , \tag{E.58}$$

$$g^2 C(k) \to \mathcal{Y}(k)_\alpha \mathcal{Y}(k)_\beta\ , \tag{E.59}$$

$$g^2 C(k) M \to \sum_\gamma \frac{1}{2}\left(M_{\alpha\gamma}\mathcal{Y}(k)_\gamma \mathcal{Y}(k)_\beta + \mathcal{Y}(k)_\gamma \mathcal{Y}(k)_\alpha M_{\gamma\beta}\right)\ . \tag{E.60}$$

For all other couplings we need

$$g^2 C(r) \to \sum_\alpha \mathcal{Y}(r)_\alpha \mathcal{Y}(r)_\alpha + \sum_a g_a^2 C_a(r) \equiv \bar{C}(r)\ , \tag{E.61}$$

$$M g^2 C(r) \to \sum_\alpha \sum_\beta M_{\alpha\beta}\mathcal{Y}(r)_\alpha \mathcal{Y}(r)_\beta + \sum_a g_a^2 M_a C_a(r) \equiv \bar{C}^M(r)\ , \tag{E.62}$$

$$M^* g^2 C(r) \to \sum_\alpha \sum_\beta M^*_{\beta\alpha}\mathcal{Y}(r)_\alpha \mathcal{Y}(r)_\beta + \sum_a g_a^2 M_a C_a(r)\ , \tag{E.63}$$

$$M M^* g^2 C(r) \to \sum_\alpha \sum_\beta (M M^*)_{\alpha\beta}\mathcal{Y}(r)_\alpha \mathcal{Y}(r)_\beta + \sum_a g_a^2 M_a M_a^* C_a(r)\ , \tag{E.64}$$

$$g^4 C(r) S(R) \to \sum_\alpha \sum_\beta \mathcal{Y}(r)_\alpha Q_{\alpha\beta} \mathcal{Y}(r)_\beta + \sum_a g_a^4 C_a(r) S_a(R)\ , \tag{E.65}$$

$$Mg^4C(r)S(R) \to \sum_\alpha\sum_\beta\sum_\gamma \mathcal{Y}(r)_\alpha Q_{\alpha\beta}M_{\beta\gamma}\mathcal{Y}(r)_\gamma + \sum_a g_a^4 M_a C_a(r)S_a(R)\ , \tag{E.66}$$

$$g^4C^2(r) \to \bar{C}(r)^2\ , \tag{E.67}$$

$$Mg^4C^2(r) \to \bar{C}^M(r)\bar{C}(r)\ , \tag{E.68}$$

$$g^2t_i^{Aj}\mathrm{Tr}(t^4m^2) \to \tilde{t}_i^{Aj}\sum_r (m^2)_r^r\tilde{t}_r^{Ar}\ , \tag{E.69}$$

$$g^2t_i^{Aj}(t^Am^2)_r^l \to \tilde{t}_i^{Aj}\sum_q \tilde{t}_q^{Al}(m^2)_r^q\ , \tag{E.70}$$

$$g^4t_i^{Aj}\mathrm{Tr}(t^4C(r)m^2) \to \tilde{t}_j^{Ai}\sum_r\sum_\beta \tilde{t}_r^{\alpha r}\mathcal{Y}(r)_\beta\mathcal{Y}(r)_\beta(m^2)_r^r\ , \tag{E.71}$$

$$g^4C(i)\mathrm{Tr}(S(r)m^2) \to \sum_\alpha\sum_\beta \mathcal{Y}(i)_\alpha\mathcal{Y}(i)_\beta\sum_r \mathcal{Y}(r)_\alpha\mathcal{Y}(r)_\beta(m^2)_r^r + \sum_a g_a^4C_a(i)\sum_r S_a(r)(m^2)_r^r\ , \tag{E.72}$$

$$\begin{aligned}24g^4MM^*C(r)S(R) \to\ & 24\sum_a g_a^4M_aM_a^2C_a(r)S_a(R) + 8\Big(\sum_{\alpha,\beta,\gamma,\delta}\mathcal{Y}(r)_\alpha M_{\alpha\beta}M^*_{\delta\beta}Q_{\delta\gamma}\mathcal{Y}(r)_\gamma + \\ & + \sum_{\alpha,\beta,\gamma,\delta}\mathcal{Y}(r)_\alpha M^*_{\beta\alpha}M_{\beta\delta}Q_{\delta\gamma}\mathcal{Y}(r)_\gamma + \sum_{\alpha,\beta,\gamma,\delta}\mathcal{Y}(r)_\alpha M_{\alpha\beta}Q_{\beta\gamma}M^*_{\delta\gamma}\mathcal{Y}(r)_\alpha\Big)\ ,\end{aligned} \tag{E.73}$$

$$\begin{aligned}48g^4MM^*C(r)^2 \to\ & \sum_a\sum_b g_a^2g_b^2C_a(r)C_b(r)(32M_aM_a^* + 8M_aM_b^* + 8M_bM_a^*) \\ & + \sum_a g_a^2C_a(r)\Big(32M_aM_a^*\sum_\alpha \mathcal{Y}(r)_\alpha\mathcal{Y}(r)_\alpha + 16M_a\sum_{\alpha,\beta}\mathcal{Y}(r)_\alpha M^*_{\beta\alpha}\mathcal{Y}(r)_\beta + \\ & \qquad 16M_a^*\sum_{\alpha,\beta}\mathcal{Y}(r)_\alpha M_{\alpha\beta}\mathcal{Y}(r)_\beta + 32\sum_{\alpha,\beta,\gamma}\mathcal{Y}(r)_\alpha M_{\alpha\beta}M^*_{\gamma\beta}\mathcal{Y}(r)_\gamma\Big) \\ & + 32\sum_{\alpha,\beta,\gamma}\mathcal{Y}(r)_\alpha M_{\alpha\beta}M^*_{\gamma\beta}\mathcal{Y}(r)_\gamma\sum_\alpha \mathcal{Y}(r)_\alpha\mathcal{Y}(r)_\alpha + \\ & \qquad 16\sum_{\alpha,\beta}\mathcal{Y}(r)_\alpha M_{\alpha\beta}\mathcal{Y}(r)_\beta\sum_{\alpha,\beta}\mathcal{Y}(r)_\alpha M^*_{\beta\alpha}\mathcal{Y}(r)_\beta\ .\end{aligned} \tag{E.74}$$

E.2 One-loop amplitudes for one- and two-point functions

We used for the calculation of the one-loop self energies and the one-loop corrections to the tadpoles in $\overline{\mathrm{DR}}$-scheme the scalar functions defined in [31]. The basic integrals are

$$A_0(m) = 16\pi^2Q^{4-n}\int\frac{d^nq}{i\,(2\pi)^n}\frac{1}{q^2-m^2+i\varepsilon}\ , \tag{E.75}$$

$$B_0(p,m_1,m_2) = 16\pi^2Q^{4-n}\int\frac{d^nq}{i\,(2\pi)^n}\frac{1}{\Big[q^2-m_1^2+i\varepsilon\Big]\Big[(q-p)^2-m_2^2+i\varepsilon\Big]}\ , \tag{E.76}$$

with the renormalization scale Q. The integrals are regularized by integrating in $n = 4 - 2\epsilon$ dimensions. The result for A_0 is

$$A_0(m) \;=\; m^2\left(\frac{1}{\hat{\epsilon}} + 1 - \ln\frac{m^2}{Q^2}\right) \;, \tag{E.77}$$

where $1/\hat{\epsilon} = 1/\epsilon - \gamma_E + \ln 4\pi$. The function B_0 has the analytic expression

$$B_0(p, m_1, m_2) \;=\; \frac{1}{\hat{\epsilon}} - \ln\left(\frac{p^2}{Q^2}\right) - f_B(x_+) - f_B(x_-) \;, \tag{E.78}$$

with

$$x_\pm \;=\; \frac{s \pm \sqrt{s^2 - 4p^2(m_1^2 - i\varepsilon)}}{2p^2} \;, \qquad f_B(x) \;=\; \ln(1-x) - x\ln(1-x^{-1}) - 1 \;, \tag{E.79}$$

and $s = p^2 - m_2^2 + m_1^2$. All the other, necessary functions can be expressed by A_0 and B_0. For instance,

$$B_1(p, m_1, m_2) \;=\; \frac{1}{2p^2}\Big[A_0(m_2) - A_0(m_1) + (p^2 + m_1^2 - m_2^2)B_0(p, m_1, m_2)\Big] \;, \tag{E.80}$$

and

$$\begin{aligned} B_{22}(p, m_1, m_2) \;=\; & \frac{1}{6}\Bigg\{\frac{1}{2}\Big(A_0(m_1) + A_0(m_2)\Big) + \left(m_1^2 + m_2^2 - \frac{1}{2}p^2\right)B_0(p, m_1, m_2) \\ & + \frac{m_2^2 - m_1^2}{2p^2}\,\Big[A_0(m_2) - A_0(m_1) - (m_2^2 - m_1^2)B_0(p, m_1, m_2)\Big] \\ & + m_1^2 + m_2^2 - \frac{1}{3}p^2\Bigg\} \;. \end{aligned} \tag{E.81}$$

Furthermore, for the vector boson self-energies it is useful to define

$$F_0(p, m_1, m_2) = A_0(m_1) - 2A_0(m_2) - (2p^2 + 2m_1^2 - m_2^2)B_0(p, m_1, m_2) \;, \tag{E.82}$$

$$G_0(p, m_1, m_2) = (p^2 - m_1^2 - m_2^2)B_0(p, m_1, m_2) - A_0(m_1) - A_0(m_2) \;, \tag{E.83}$$

$$H_0(p, m_1, m_2) = 4B_{22}(p, m_1, m_2) + G(p, m_1, m_2) \;, \tag{E.84}$$

$$\tilde{B}_{22}(p, m_1, m_2) = B_{22}(p, m_1, m_2) - \frac{1}{4}A_0(m_1) - \frac{1}{4}A_0(m_2) \tag{E.85}$$

In all calculations, specific coefficient are involved:

- c_S is the symmetry factor: if the particles in the loop are indistinguishable, the weight of the contribution is only half of the weight in the case of distinguishable particles. If two different charge flows are possible in the loop, the weight of the diagram is doubled.
- c_C is a charge factor: for corrections due to vector bosons in the adjoint representation this is the Casimir of the corresponding group. For corrections due to matter fields this can be, for instance, a color factor for quarks/squarks. For corrections of vector bosons in the adjoint representation this is normally the Dynkin index of the gauge group.
- c_R is 2 for real fields and Majorana fermions in the loop and 1 otherwise.

We use in the following Γ for non-chiral interactions and Γ_L/Γ_R for chiral interactions. If two vertices are involved, the interaction of the incoming particle has an upper index 1 and for the outgoing field an upper index 2 is used.

E.2.1 One-loop tadpoles

a) Fermion loop (generic name in SARAH: FFS):

$$T = 8 c_S c_C m_F \Gamma A_0(m_F^2) \tag{E.86}$$

b) Scalar loop (generic name in SARAH: SSS):

$$T = -2 c_S c_C \Gamma A_0(m_S^2) \tag{E.87}$$

c) Vector boson loop (generic name in SARAH: SVV):

$$T = 6 c_S c_C \Gamma A_0(m_V^2) \tag{E.88}$$

E.2.2 One-loop self-energies

E.2.2.0.1 Corrections to fermion

a) Fermion-scalar loop (generic name in SARAH: FFS):

$$\begin{aligned}
\Sigma^S(p^2) &= m_F c_S c_C c_R \Gamma_R^1 \Gamma_L^{2,*} B_0(p^2, m_F^2, m_S^2) \\
\Sigma^R(p^2) &= -c_S c_C c_R \frac{1}{2} \Gamma_R^1 \Gamma_R^{2,*} B_1(p^2, m_F^2, m_S^2) \\
\Sigma^L(p^2) &= -c_S c_C c_R \frac{1}{2} \Gamma_L^1 \Gamma_L^{2,*} B_1(p^2, m_F^2, m_S^2)
\end{aligned}$$

b) Fermion-vector boson loop (generic name in SARAH: FFV):

$$\begin{aligned}
\Sigma^S(p^2) &= -4 c_S c_C c_R m_F \Gamma_L^1 \Gamma_R^{2,*} B_0(p^2, m_F^2, m_S^2) \\
\Sigma^R(p^2) &= -c_S c_C c_R \Gamma_L^1 \Gamma_L^{2,*} B_1(p^2, m_F^2, m_S^2) \\
\Sigma^L(p^2) &= -c_S c_C c_R \Gamma_R^1 \Gamma_R^{2,*} B_1(p^2, m_F^2, m_S^2)
\end{aligned}$$

E.2.2.0.2 Corrections to scalar

a) Fermion loop (generic name in SARAH: FFS):

$$\Pi(p^2) = c_S c_C c_R \Big((\Gamma_L^1 \Gamma_L^{2,*} + \Gamma_R^1 \Gamma_R^{2,*}) G_0(p^2, m_F^2, m_S^2) + (\Gamma_L^1 \Gamma_R^{2,*} + \Gamma_R^1 \Gamma_L^{2,*}) B_0(p^2, m_F^2, m_S^2) \Big) \tag{E.89}$$

b) Scalar loop (two 3-point interactions, generic name in SARAH: SSS):

$$\Pi(p^2) = c_S c_C c_R \Gamma^1 \Gamma^{2,*} B_0(p^2, m_F^2, m_S^2) \tag{E.90}$$

c) Scalar loop (4-point interaction, generic name in SARAH: SSSS):

$$\Pi(p^2) = -c_S c_C \Gamma A_0(m_S^2) \tag{E.91}$$

d) Vector boson-scalar loop (generic name in SARAH: SSV):

$$\Pi(p^2) = c_S c_C c_R \Gamma^1 \Gamma^{2,*} F_0(p^2, m_F^2, m_S^2) \tag{E.92}$$

e) Vector boson loop (two 3-point interactions, generic name in SARAH: SVV):

$$\Pi(p^2) = c_S c_C c_R \frac{7}{2} \Gamma^1 \Gamma^{2,*} B_0(p^2, m_F^2, m_S^2) \tag{E.93}$$

f) Vector boson loop (4-point interaction, generic name in SARAH: SSVV):

$$\Pi(p^2) = c_S c_C \Gamma A_0(m_V^2) \tag{E.94}$$

E.2.2.0.3 Corrections to vector boson

a) Fermion loop (generic name in SARAH: FFV):

$$\Pi^T(p^2) = c_S c_C c_R \left((|\Gamma^1_L|^2 + |\Gamma^1_R|^2) H_0(p^2, m_V^2, m_F^2) + 4Re(\Gamma^1_L \Gamma^2_R) B_0(p^2, m_V^2, m_F^2) \right) \tag{E.95}$$

b) Scalar loop (generic name in SARAH: SSV):

$$\Pi^T(p^2) = -4 c_S c_C c_R |\Gamma|^2 B_{22}(p^2, m_{S_1}^2, m_{S_2}^2) \tag{E.96}$$

c) Vector boson loop (generic name in SARAH: VVV):

$$\Pi^T(p^2) = |\Gamma|^2 c_S c_C c_R \left(-(4p^2 + m_{V_1}^2 + m_{V_2}^2) B_0(p^2, m_{V_1}^2, m_{V_1}^2) - 8 B_{22}(p^2, m_{S_1}^2, m_{S_2}^2) \right) \tag{E.97}$$

d) Vector-Scalar-Loop (generic name in SARAH: SVV):

$$\Pi^T(p^2) = |\Gamma|^2 c_S c_C c_R B_0(p^2, m_V^2, m_S^2) \tag{E.98}$$

We need here only the diagrams involving three point interactions because the 4-point interactions are related to them due to gauge invariance.

E.2.3 One-loop corrections to masses

The one-loop self-energies can be used to calculate the one-loop masses and mass matrices.

a) Real scalars: for a real scalar ϕ, the one-loop corrections are included by calculating the real part of the poles of the corresponding propagator matrices [31]

$$\text{Det}\left[p_i^2 \mathbf{1} - m^2_{\phi,1L}(p^2)\right] = 0, \tag{E.99}$$

where

$$m^2_{\phi,1L}(p^2) = \tilde{m}^2_{\phi,T} - \Pi_\phi(p^2). \tag{E.100}$$

Equation (E.99) has to be solved for each eigenvalue $p^2 = m_i^2$ which can be achieved in an iterative procedure. This has to be done also for charged scalars as well as the fermions. Note, $\tilde{m}^2_T$ is the tree-level mass matrix but for the parameters fixed by the tadpole equations the one-loop corrected values $X^{(1)}$ are used.

b) Complex scalars: for a complex scalar η field we use at one-loop level

$$m^{2,\eta}_{1L}(p_i^2) = \tilde{m}^{2,\eta}_T - \Pi_\eta(p_i^2), \tag{E.101}$$

While in case of sfermions $\tilde{m}^{2,\eta}_T$ agrees exactly with the tree-level mass matrix, for charged Higgs bosons $\mu^{(1)}$ and $B_\mu^{(1)}$ or $m^{(1)}_{H_d}$ and $m^{(1)}_{H_d}$ has to be used depending on the set of parameters the tadpole equations are solved for.

c) Majorana fermions: the one-loop mass matrix of a Majorana χ fermion is related to the tree-level mass matrix by

$$\begin{aligned} M^\chi_{1L}(p_i^2) \quad &= \quad M^\chi_T - \frac{1}{2}\Big[\Sigma^0_S(p_i^2) + \Sigma^{0,T}_S(p_i^2) + \left(\Sigma^{0,T}_L(p_i^2) + \Sigma^0_R(p_i^2)\right) M^\chi_T \\ &\qquad + M^{\tilde{\chi}^0}_T \left(\Sigma^{0,T}_R(p_i^2) + \Sigma^0_L(p_i^2)\right)\Big], \end{aligned} \tag{E.102}$$

where we have denoted the wave-function corrections by Σ^0_R, Σ^0_L and the direct one-loop contribution to the mass by Σ^0_S.

d) Dirac fermions: for a Dirac fermion Ψ one has to add the self-energies as

$$M_{1L}^{\Psi}(p_i^2) = M_T^{\Psi} - \Sigma_S^{+}(p_i^2) - \Sigma_R^{+}(p_i^2) M_T^{\Psi} - M_T^{\Psi} \Sigma_L^{+}(p_i^2). \tag{E.103}$$

Appendix F

More information about the SPheno output

F.1 Generated Files for SPheno

All routines generated by
SARAH are strongly inspired by the intrinsic routines of SPheno to ensure that they interact nicely with all other SPheno functions. We give here some details of the different files which are written by
SARAH as well as of the main functions for the case that the user wants to do some changes.

F.1.1 BranchingRatios_[Model].f90

Contains the routines for calculating the branching ratios for all particles. Checks are done, if three body decays for fermions are necessary or not.

F.1.2 Couplings_[Model].f90

All vertices calculated by
SARAH are written to this files. Also functions for calculating different subsets of these couplings are generated:

- AllCouplings: all couplings for the mass eigenstates (used for the calculation of decays)
- CouplingsForSMfermion: couplings involved in the one loop self energy of SM fermions
- CouplingsForVectorBosons: couplings involved in the one loop self energy of SM vector bosons
- CouplingsForTadpoles: couplings involved in the calculation of the one loop tadpoles
- CouplingsForLoop: couplings needed for the one loop self energies of all particles

For all couplings involved in loop calculation, the mixing matrices of external particles are replaced by the identity matrix.

F.1.3 CouplingsForDecays_[Model].f90

Calculates the running couplings at the mass scale of each decaying particle. The SUSY parameters are obtained by running the RGEs from the SUSY scale to the desired energy scale. The calculation of the SM gauge and Yukawa coupling is different for particles with masses above or below the SUSY scale. For lighter particles, the SM RGEs are used to run from M_Z to the mass scale, for heavier particles the SUSY RGEs are taken to run from M_{SUSY}.
In addition, the running tree-level masses are calculated. These are used to get effective couplings of the Higgs bosons to two gluons or photons. Furthermore, also the couplings of a light Higgs to a real and virtual vector boson (W, Z) are calculated.

F.1.4 InputOutput_[Model].f90

This file contains the routines for reading the `LesHouches.in.[Model]` input file and writing the output `SPheno.spc.[Model]`.

Input The following blocks are changed in comparison to the standard LesHouches/`SPheno` input, see also sec. F.2.

- `Block MODSEL`
- `Block MINPAR`
- `Block EXTPAR`
- `Block MODSEL`
- `Block SPHENOINPUT`

Output Routines for writing the following information to the LesHouches spectrum file are generated:

- Parameters at the SUSY scale:
  As block names for the output, the entries in `LesHouches` in the parameter file of
  SARAH are used. If this entry is missing, the name for the block is generated automatically.
- Masses at the SUSY scale:
  The PDG of the particles file of `SARAH` is used. If this is missing or 0, the mass is not written in order to exclude unphysical states.
- Optionally, decay widths and branching ratios for the particle in `ListDecayParticles` and `ListDecayParticles3B`
- Optionally, results for low energy constraints
- Optionally, effective couplings of the Higgs to SM particles which can be used for instance by `HiggsBounds`
- Optionally, the GUT values of all parameters
- Optionally, the loop contributions of all particles to the effective coupling of the Higgs to two photons or gluons

In addition, that file contains a routine to write the parameters to a `WHIZARD` specific output file

F.1.5 LoopCouplings_[Model].f90

Routines for calculating the following couplings:

- Corrections to SM gauge couplings due to all particles heavier than M_Z, i.e. normally SUSY particles and top quark (AlphaSDR, AlphaEwDR)
- Effective couplings of scalar/pseudoscalar Higgs to SM fermions (e.g. CouphhtoFe).
- DeltaVB: one-loop correction to G_F calculated from $\mu \to e \sum_{ij} \nu_i \bar{\nu}_j$

F.1.6 LoopMasses_[Model].f90

This file contains the routines for calculating the one loop contributions to the tadpoles and the self energies:

- OneLoopMasses: main routine for calling all other functions
- TadpolesOneLoop: calculates the one loop tadpoles.
- Pi1LoopX: one loop self energy for particle X (scalars or vector bosons (transverse part))
- Sigma1LoopX: calculates $\Sigma_L, \Sigma_R, \Sigma_S$ for fermions
- OneLoopX: calculates the one loop corrected mass for particle X in an iterative way in order to solve

$$\text{Det}\left[p_i^2 \mathbf{1} - m_{1L}^{2,X}(p^2)\right] = 0, \tag{F.1}$$

for external masses on-shell, i.e. $p^2 = m_{Tree}^{2,X}$.

All these calculation are done in 't Hooft gauge. Normally, the results are one-loop $\overline{\text{DR}}$ masses and mixing matrices. For further calculations, the mixing matrices corresponding to an external momentum equal to the heaviest mass eigenstate is taken. However, for SM particles the pole masses are used in the decays as well as in the output.

F.1.7 LowEnergy_[Model].f90

Contains the routines for the calculation of precision observables:

- $b \to s\gamma$ [59, 60, 61] (BToQGamma)
- $l_i \to l_j \gamma$ [62] (BrLgammaLp)
- $l_i \to 3l_j$ (with $l = (e, \mu, \tau)$), [62] (BR1LeptonTo3Leptons)
- the anomalous magnetic moment of leptons [63] (Gminus2)
- electric dipole moments of the charged leptons [64, 65] (LeptonEDM)
- $\delta\rho = 1 - \rho = \frac{\Pi_{WW}(0)}{m_W^2} - \frac{\Pi_{ZZ}(0)}{m_Z^2}$ (Π_{ZZ}, Π_{WW} are the self-energies of the massive vector bosons). [66] (DeltaRho)
- $\mu - e$ conversion in nuclei (Al, Ti, Sr, Sb, Au, Pb) based on the results of [67] (BrLLpHadron)

- $\tau \to lP^0$ with a pseudoscalar meson P^0 (BrLLpHadron) (π^0, η, η') based on the results of [68]
- $Z \to l_i l_j$ calculated and implemented by Kilian Nickel (based on generic results obtained by FeynArts/FormCalc) [69] (BrZLLp)
- $B_{s,d} \to l_i l_j$ calculated and implemented by Kilian Nickel (based on generic results obtained by FeynArts/FormCalc) (BrB0LLp)

Note, the calculation is based on the approach and conventions given in the references. However, these expressions are generalized and
SARAH includes new contributions possible in an extended SUSY model. This is done be calculating and implementing the amplitudes for each generic possible diagram for a given process in a general form. The necessity of this is for instance pronounced in Ref. [70] where it has been shown that the Z-penguins to $l_i \to 3l_j$ often small in the MSSM can dominate in other models like inverse seesaw.

F.1.8 ModelData_[Model].f90

Contains the declaration of global variables and a function for initializing all variables with 0. Only masses of particles, which are integrated out, are initialized with 10^{16} GeV. That's used for the threshold corrections in the first iteration.
In addition, functions for the calculation of GMSB boundary conditions as well as the content of SelfDefinedFunctions is written to that file, see also sec. 11.2.

F.1.9 RGEs_[Model].f90

That file includes all information about the RGEs. In principle,
SARAH generates three different sets of RGEs for ...

- ... running of all parameters (gauge couplings, superpotential parameter, soft breaking parameters) from the GUT scale to the SUSY scale
- ... running of all parameters and VEVs from SUSY scale to electroweak scale and back
- ... running of a minimal set of parameters from electroweak scale to GUT scale: that includes all SM gauge and Yukawa couplings as well as the other parameter the running depends on. Above a threshold scale, the set of parameters consists of all parameter related to the SM gauge and Yukawa couplings.

For each set of RGEs the following function are generated:

- ParametersToGXX: saves the parameters in a vector of length XX
- GToParametersXX: extracts the parameters from a vector of length XX
- rgeXX: definition of all β-functions.

SARAH does the following simplification/modifications of the β-functions before exporting it to Fortran code, in order to increase the speed of the numerical calculation

- All matrix multiplications are replaced by constants, which are calculated at the beginning of each rgeXX routine
- All powers of numbers/parameters are replaced by a constants, which are also calculated first

F.1.10 `Shifts_[Model].f90`

That file is only created for models including at least one threshold scale with gauge symmetry breaking. It contains the necessary routines to calculate the finite shifts for gauge couplings and gaugino masses at the threshold.

F.1.11 `SPheno_[Model].f90`

The main program. Calls routines for reading the input, calculating the spectrum, decay widths and low energy observables and writing the output.

F.1.12 `SugraRuns_[Model].f90`

Main routines for calculating the (s)particle spectrum:

- `FirstGuess`: calculates a first, approximate spectrum as starting point
- `SugraRuns`: calculates in an iterative way the spectrum
- `BoundaryEW`: calculates the starting point of the running for gauge and Yukawa couplings from electroweak data
- `BoundaryHS, BoundarySUSY`: sets the boundary conditions at the GUT and SUSY scale.
- `BoundaryConditionsUp,BoundaryConditionsDown`: applies the boundary conditions at the threshold scales when running up/down

F.1.13 `SusyDecays_[Model].f90`

Calculates all two body decays by using the phase space functions of `SPheno`. If scalar and pseudo scalar Higgs as well as the quarks are defined in the particles file by the corresponding `Description` statement, the one loop corrections to the decays $H, A \to q\bar{q}$ from gluons are added, see app. F.3 for more information.

F.1.14 `SusyMasses_[Model].f90`

This file contains functions to calculate all running masses at tree level. The mass matrices involving Goldstone bosons are taken in 't Hooft gauge. Note, that works only, if the Goldstone bosons have been associated with the corresponding vector boson in the `particles.m` file, see sec. 5.6.5. The main routines in that file are

- `TreeMasses`: main function to call the other subroutines
- `CalculateX`: derives the tree level mass and mixing matrix of particle `X` by solving the eigensystem of the corresponding mass matrix.

F.1.15 Three Body Decays, e.g. `Glu_[Model].f90`

Contains all necessary routines to calculate the branching ratios and decay width of a fermion to three other fermions. All possible processes and diagrams are generated by
SARAH and mapped to the phase space function of `SPheno`.

F.1.16 Additional Files

- A model specific Makefile
- Template for an input file of the current model: LesHouches.in.[Model], see app. F.2.
- Template for an input file assuming a SUSY scale input: LesHouches.in.[Model]_low, see also app. F.2.

F.2 LesHouches input file

The LesHouches input file for each model is by default named

```
LesHouches.in.[Model]
```

However, it is also possible to give the name of the in- and output file as option when running SPheno, see also Ref. [38]:

```
./[SPheno Directory]/bin/SPheno[Model] [Input file] [Output file]
```

While the entries of the common block MINPAR and EXTPAR are completely defined by the SPheno input file for
SARAH, some other blocks have distinct values.

SMINPUTS

The entries of this block accordingly to the SLHA conventions are:

1 : $\alpha_{\rm em}^{-1}(m_Z)^{\overline{\rm MS}}$. Inverse electromagnetic coupling at the Z pole in the $\overline{\rm MS}$ scheme (with 5 active flavours).

2 : G_F. Fermi constant (in units of GeV^{-2}).

3 : $\alpha_s(m_Z)^{\overline{\rm MS}}$. Strong coupling at the Z pole in the $\overline{\rm MS}$ scheme (with 5 active flavours).

4 : m_Z, pole mass.

5 : $m_b(m_b)^{\overline{\rm MS}}$. b quark running mass in the $\overline{\rm MS}$ scheme.

6 : m_t, pole mass.

7 : m_τ, pole mass.

8 : m_{ν_3}, pole mass.

11 : $m_{\rm e}$, pole mass.

12 : m_{ν_1}, pole mass.

13 : m_μ, pole mass.

14 : m_{ν_2}, pole mass.

21 : $m_d(2\text{ GeV})^{\overline{\rm MS}}$. d quark running mass in the $\overline{\rm MS}$ scheme.

22 : $m_u(2\text{ GeV})^{\overline{\rm MS}}$. u quark running mass in the $\overline{\rm MS}$ scheme.

23 : $m_s(2\text{ GeV})^{\overline{\rm MS}}$. s quark running mass in the $\overline{\rm MS}$ scheme.

24 : $m_c(m_c)^{\overline{\mathrm{MS}}}$. c quark running mass in the $\overline{\mathrm{MS}}$ scheme.

MODSEL

The block MODSEL is not totally equivalent to the SLHA conventions: since each SPheno module generated by
SARAH handles one specify model, the flags 3 and 4 have no effect. In addition, we added the flag 2 to choose between the different boundary conditions, and flag 1 is used to choose between a GUT scale and SUSY scale input.

1 : (Default=1) Choice of input type
  0 : low scale input
  1 : Gut scale input

2 : (Default=1) Choice of boundary conditions.
  X : GUT scale input. X gives the number of the set of boundary conditions

3 : No effect

4 : No effect

5 : (Default=0) CP violation. Switches defined are:
  0 : CP is conserved. No information even on the CKM phase is used. This corresponds to the SLHA1.
  1 : CP is violated, but only by the standard CKM phase. All other phases are assumed zero.
  2 : CP is violated. Completely general CP phases allowed.

6 : (Default=0) Flavour violation. Switches defined are:
  0 : No (SUSY) flavour violation.
  1-3 : Flavour is violated.

SPHENOINPUT

The block with SPheno specific commands is similar to the conventions used by the standard SPheno version 3.1 and above. However, some switches have no effect and there are also some new entries which we highlight with in bold.

1 : sets the error level

2 : if 1, the SPA conventions are used

3 : No effect

4 : No effect

7 : Skip two loop Higgs masses

11 : if 1 then the branching ratios of the SUSY and Higgs particles are calculated, if 0 then this calculation is omitted.

12 : sets minimum value for a branching ratios, so that it appears in the output

13 : Include possible three-body decays (Note, these calculations can be time consuming!)

21-26 : No effect

31 : sets the value of M_{GUT}, otherwise M_{GUT} is determined by the condition $g_1 = g_2$

32 : sets strict unification, i.e. $g_1 = g_2 = g_3$

33 : Set fixed renormalization scale (Note, SPA conventions have to be switched off)

34 : sets the relative precision with which the masses are calculated, default is 10^{-4}

35 : sets the maximal number of iterations in the calculation of the masses, default is 40

36 : whether to write out debug information for the loop calculations

38 : this entry sets the loop order of the RGEs: either 1 or 2, default is 2, i.e. using 2-loop RGEs

41 : sets the width of the Z-boson Γ_Z, default is 2.49 GeV

42 : sets the width of the W-boson Γ_W, default is 2.06 GeV

50 : if 1, negative fermion masses are rotated to real ones by using complex mixing matrices; if 0, all mixing matrices for Majorana fermions are real, but masses can be negative

51 : If set to 0, the parameters $Y_u, Y_d, T_u, T_d, m_q^2, m_d^2, m_u^2$ are not rotated in SCKM basis in the output file

52 : if 1, a negative mass squared is always ignored and set 0

53 : if 1, a negative mass squared at M_Z is always ignored and set 0

54 : if 1, the output is written even if there has been a problem during the run

55 : if 0, the loop corrections to the masses are skipped

57 : if 0, the calculation of the low energy observables is skipped

58 : if 0, the calculation of δ_{VB} in the boundary conditions at the SUSY scale is skipped

60 : if 0, possible effects from kinetic mixing are neglected

62 : if 1, a sign flip in μ (or other quadratic terms obtained by the tadpole equations) is ignored

65 : X, defined the solution of the tadpole equations if several, independent solutions exist

72 : if 1, the running values of all parameters at the GUT scale are written

75 : if 1, a file containing all parameters in WHIZARD format is created

76 : if 1, input files for HiggsBounds are written

80 : if not set 0 the program exists with a non-zero value if a problem has occurred

90-92 : No effect

F.3 Decays calculated by SPheno

In general, SPheno version produced with SARAH calculate the two-body decays of all SUSY scalars as well as the Higgs fields. In addition, for SUSY fermions also the three-body decays are included at tree-level. In addition, the calculation of all decays has been improved by performing an RGE evaluation of all couplings from the SUSY scale to the mass scale of the decaying particle. Furthermore, the calculation of the loop-induced decays of a Higgs particle into two photons and two gluons include also the dominant QCD corrections based on the results given in Ref. [71]. This leads to the following precision precision in the calculations:

- for all fermionic SUSY particle the two- and three body decays are calculated at tree-level
- for squarks, sleptons and additional heavy vector bosons the two-body decays are calculated at tree level
- In the Higgs sector, possible decays into two SUSY or SM particles are calculated at tree-level. In the case of two quarks in the final state the dominant QCD corrections due to gluons are included. The loop induced decays into two photons and gluons are fully calculated at LO with the dominant NLO corrections as just mentioned. In addition, in the Higgs decays also final states with off-shell gauge bosons (ZZ^*, WW^*) are also take into account.

F.4 Input files to generate a SPheno version for the MSSM and NMSSM

F.4.1 MSSM

MSSM.m

We have already discussed the main parts of the MSSM input file for SARAH in sec. G.2. Therefore, we just show the file here for completeness again.

```
ModelName = "MSSM";
ModelNameLaTeX ="MSSM";

(* ------------------------------ Vector Superfields ------------------------------ *)

Gauge[[1]]={B,   U[1], hypercharge, g1,False};
Gauge[[2]]={WB, SU[2], left,        g2,True};
Gauge[[3]]={G,  SU[3], color,       g3,False};

(* ------------------------------ Chiral Superfields ------------------------------ *)

Fields[[1]] = {{uL,  dL},  3, q,   1/6, 2, 3};
Fields[[2]] = {{vL,  eL},  3, l,  -1/2, 2, 1};
Fields[[3]] = {{Hd0, Hdm}, 1, Hd, -1/2, 2, 1};
Fields[[4]] = {{Hup, Hu0}, 1, Hu,  1/2, 2, 1};

Fields[[5]] = {conj[dR], 3, d,  1/3, 1, -3};
Fields[[6]] = {conj[uR], 3, u, -2/3, 1, -3};
```

```
Fields[[7]] = {conj[eR], 3, e,     1, 1,  1};

(* ------------------------------ Superpotential ------------------------------ *)

SuperPotential = { {{1, Yu},{u,q,Hu}}, {{-1,Yd},{d,q,Hd}},
                   {{-1,Ye},{e,l,Hd}}, {{1,\[Mu]},{Hu,Hd}}};


(* -------------------------- Different eigenstates -------------------------- *)

NameOfStates={GaugeES, EWSB};

(* ------------------------ Rotations in gauge sector ------------------------ *)

DEFINITION[EWSB][GaugeSector]=
{ {{VB,VWB[3]},{VP,VZ},ZZ},
  {{VWB[1],VWB[2]},{VWm,conj[VWm]},ZW},
  {{fWB[1],fWB[2],fWB[3]},{fWm,fWp,fW0},ZfW}
};

(* ------------------------ Decomposition of Scalars ------------------------ *)

DEFINITION[EWSB][VEVs]=
  {{SHd0, {vd, 1/Sqrt[2]}, {sigmad, \[ImaginaryI]/Sqrt[2]},{phid,1/Sqrt[2]}},
   {SHu0, {vu, 1/Sqrt[2]}, {sigmau, \[ImaginaryI]/Sqrt[2]},{phiu,1/Sqrt[2]}}};


(* ------------------------ Rotations in matter sector ------------------------ *)

DEFINITION[EWSB][MatterSector]=
{     {{SdL, SdR},                    {Sd, ZD}},
      {{SvL},                         {Sv, ZV}},
      {{SuL, SuR},                    {Su, ZU}},
      {{SeL, SeR},                    {Se, ZE}},
      {{phid, phiu},                  {hh, ZH}},
      {{sigmad, sigmau},              {Ah, ZA}},
      {{SHdm,conj[SHup]},             {Hpm,ZP}},
      {{fB, fW0, FHd0, FHu0},         {L0, ZN}},
      {{{fWm, FHdm}, {fWp, FHup}},    {{Lm,UM}, {Lp,UP}}},
      {{{FeL},{conj[FeR]}},           {{FEL,ZEL},{FER,ZER}}},
      {{{FdL},{conj[FdR]}},           {{FDL,ZDL},{FDR,ZDR}}},
      {{{FuL},{conj[FuR]}},           {{FUL,ZUL},{FUR,ZUR}}}
        };

(* ---------------------------------- Phases ---------------------------------- *)

DEFINITION[EWSB][Phases]=  {     {fG, PhaseGlu}      };
```

```
(* ---------------------------------- Dirac Spinors ---------------------------------- *)

DEFINITION[EWSB][DiracSpinors]={
 Fd  -> {FDL, conj[FDR]},
 Fe  -> {FEL, conj[FER]},
 Fu  -> {FUL, conj[FUR]},
 Fv  -> {FvL, 0},
 Chi -> {L0, conj[L0]},
 Cha -> {Lm, conj[Lp]},
 Glu -> {fG, conj[fG]}
};

DEFINITION[GaugeES][DiracSpinors]={
   Bino -> {fB, conj[fB]},
   Wino -> {fWB, conj[fWB]},
   Glu  -> {fG, conj[fG]},
   H0   -> {FHd0, conj[FHu0]},
   HC   -> {FHdm, conj[FHup]},
   Fd1  -> {FdL, 0},
   Fd2  -> {0, FdR},
   Fu1  -> {FuL, 0},
   Fu2  -> {0, FuR},
   Fe1  -> {FeL, 0},
   Fe2  -> {0, FeR},
   Fv   -> {FvL,0}
};
```

`SPheno.m`

To generate the `SPheno` output for the MSSM, `SPheno.m` should provide the following information:
We want to have mSugra like boundary conditions. Therefore, we define as minimal set of parameters for the model $m_0, M_{1/2}, A_0, \text{sign}\mu$ and $\tan\beta$. These will later on be read from the `MINPAR` block of a LesHouches input file

```
MINPAR={{1,m0},
           {2,m12},
           {3,TanBeta},
           {4,SignumMu},
           {5,Azero}};
```

In general, these parameters are assumed to be complex, i.e. it is possible to use also the block `IMMINPAR` to define the imaginary part. However, some Fortran functions like `sin` can't be used with complex numbers, therefore we have to define $\tan\beta$ explicitly as real. Also m_0 is a real parameter.

```
RealParameters = {TanBeta, m0};
```

As usual in the MSSM, the tadpole equations should be solved with respect to μ and B_μ. That's defined by

```
ParametersToSolveTadpoles = {\[Mu],B[\[Mu]]};
```

To study models with a dynamically adjusted SUSY scale, the expressions for the definition of the SUSY scale can be given. The first expression is used only before the mass spectrum has been calculated the first time. Note, that these definitions can easily disabled in the LesHouches input file by flag `MODSEL 12` and a fixed scale can be used. Also, when SPA conventions are switched on in the LesHouches input file by the flag `2` in the block `SPhenoInput`, a fixed scale of 1 TeV is used.

```
RenormalizationScaleFirstGuess = m0^2 + 4 m12^2;
RenormalizationScale = MSu[1]*MSu[6];
```

As said, we want to use mSugra like boundary conditions. These are defined by

```
BoundaryHighScale={
  {T[Ye],     Azero*Ye},
  {T[Yd],     Azero*Yd},
  {T[Yu],     Azero*Yu},
  {mq2,       DIAGONAL m0^2},
  {ml2,       DIAGONAL m0^2},
  {md2,       DIAGONAL m0^2},
  {mu2,       DIAGONAL m0^2},
  {me2,       DIAGONAL m0^2},
  {mHd2,      m0^2},
  {mHu2,      m0^2},
  {MassB,     m12},
  {MassWB,    m12},
  {MassG,     m12}
};
```

It is also possible to use the generated `SPheno` version with a low scale input. This is enabled by setting `MODSEL 1` to `0` in the LesHouches input file. In that case, input values for all free parameters of the model are expected. However, it is possible to define also some boundary conditions to calculate for example the SUSY VEVs dynamically.

```
BoundaryLowScaleInput={
 {vd,Sqrt[2 mz2/(g1^2+g2^2)]*Sin[ArcTan[TanBeta]]},
 {vu,Sqrt[2 mz2/(g1^2+g2^2)]*Cos[ArcTan[TanBeta]]}
};
```

Finally, we define that the code for the calculation of the two and three body decays is generated for all SUSY and Higgs particles. That's done by using the flag `Automatic`.

```
ListDecayParticles = Automatic;
ListDecayParticles3B = Automatic;
```

F.4.2 NMSSM

`NMSSM.m`

```
ModelName = "NMSSM";
ModelNameLaTeX ="NMSSM";

(* ------------------------------ Vector Superfields ------------------------------ *)

Gauge[[1]]={B,    U[1], hypercharge, g1,False};
Gauge[[2]]={WB,  SU[2], left,        g2,True};
Gauge[[3]]={G,   SU[3], color,       g3,False};

(* ------------------------------ Chiral Superfields ------------------------------ *)

Fields[[1]] = {{uL,  dL},  3, q,   1/6, 2, 3};
Fields[[2]] = {{vL,  eL},  3, l,  -1/2, 2, 1};
Fields[[3]] = {{Hd0, Hdm}, 1, Hd, -1/2, 2, 1};
Fields[[4]] = {{Hup, Hu0}, 1, Hu,  1/2, 2, 1};

Fields[[5]] = {conj[dR], 3, d,  1/3, 1, -3};
Fields[[6]] = {conj[uR], 3, u, -2/3, 1, -3};
Fields[[7]] = {conj[eR], 3, e,    1, 1,  1};
Fields[[8]] = {sR,       1, s,    0, 1,  1};


(* -------------------------------- Superpotential -------------------------------- *)

SuperPotential = { {{1, Yu},{q,Hu,u}}, {{-1,Yd},{q,Hd,d}},
  {{-1,Ye},{l,Hd,e}}, {{1,\[Lambda]},{Hu,Hd,s}}, {{1/3,\[Kappa]},{s,s,s}}};


(* ---------------------------- Different eigenstates ---------------------------- *)

NameOfStates={GaugeES, EWSB};


(* --------------------------- Rotations in gauge sector --------------------------- *)

DEFINITION[EWSB][GaugeSector]=
{ {{VB,VWB[3]},{VP,VZ},ZZ},
  {{VWB[1],VWB[2]},{VWm,conj[VWm]},ZW},
  {{fWB[1],fWB[2],fWB[3]},{fWm,fWp,fW0},ZfW}
};

(* --------------------------- Decomposition of Scalars --------------------------- *)


DEFINITION[EWSB][VEVs]=
{     {SHd0, {vd, 1/Sqrt[2]}, {sigmad, I/Sqrt[2]}, {phid,1/Sqrt[2]}},
      {SHu0, {vu, 1/Sqrt[2]}, {sigmau, I/Sqrt[2]}, {phiu,1/Sqrt[2]}},
      {SsR, {vS, 1/Sqrt[2]},  {sigmaS, I/Sqrt[2]}, {phiS, 1/Sqrt[2]}}
  };


(* ------------------------ Rotations in matter sector ------------------------ *)

DEFINITION[EWSB][MatterSector]=
{     {{SdL, SdR},                     {Sd, ZD}},
      {{SvL},                          {Sv, ZV}},
      {{SuL, SuR},                     {Su, ZU}},
      {{SeL, SeR},                     {Se, ZE}},
      {{phid, phiu, phiS},             {hh, ZH}},
      {{sigmad, sigmau,sigmaS},        {Ah, ZA}},
      {{SHdm,conj[SHup]},              {Hpm,ZP}},
      {{fB, fW0, FHd0, FHu0,FsR},      {L0, ZN}},
      {{{fWm, FHdm}, {fWp, FHup}},     {{Lm,UM}, {Lp,UP}}},
      {{{FeL},{conj[FeR]}},            {{FEL,ZEL},{FER,ZER}}},
      {{{FdL},{conj[FdR]}},            {{FDL,ZDL},{FDR,ZDR}}},
      {{{FuL},{conj[FuR]}},            {{FUL,ZUL},{FUR,ZUR}}}
   };

(* ------------------------------------ Phases ------------------------------------ *)

DEFINITION[EWSB][Phases]=   {     {fG, PhaseGlu}       };

(* --------------------------------- Dirac Spinors --------------------------------- *)

DEFINITION[EWSB][DiracSpinors]={
 Fd  -> {FDL, conj[FDR]},
 Fe  -> {FEL, conj[FER]},
 Fu  -> {FUL, conj[FUR]},
 Fv  -> {FvL, 0},
 Chi -> {L0, conj[L0]},
 Cha -> {Lm, conj[Lp]},
 Glu -> {fG, conj[fG]}
};

DEFINITION[GaugeES][DiracSpinors]={
   Bino -> {fB, conj[fB]},
   Wino -> {fWB, conj[fWB]},
   Glu  -> {fG, conj[fG]},
   H0   -> {FHd0, conj[FHu0]},
   HC   -> {FHdm, conj[FHup]},
   Fd1  -> {FdL, 0},
   Fd2  -> {0, FdR},
   Fu1  -> {FuL, 0},
   Fu2  -> {0, FuR},
   Fe1  -> {FeL, 0},
  Fe2  -> {0, FeR},
  Fv   -> {FvL,0},
  S    -> {FsR, conj[FsR]}
};
```

`SPheno.m`

We define as free parameters $\kappa, A_\kappa, \lambda, A_\lambda$ as well as the singlet VEV v_S in addition to the mSugra parameters of the MSSM. To be SLHA conform, these parameters are added the the `EXTPAR` block.

```
MINPAR={{1,m0},
          {2,m12},
          {3,TanBeta},
          {5,Azero}};

EXTPAR = {
              {61,LambdaInput},
              {62,KappaInput},
              {63,ALambdaInput},
              {64,AKappaInput},
              {65,vSInput}
           };
```

Again, $\tan\beta$ must be defined as real an we choose the same expressions for the SUSY scale as in the MSSM.

```
RealParameters = {TanBeta, m0};

RenormalizationScaleFirstGuess = m0^2 + 4 m12^2;
RenormalizationScale = MSu[1]*MSu[6];
```

We need new parameters to solve the tadpole equations. The easiest choice is to use the soft breaking masses for the Higgs and the gauge singlet.

```
ParametersToSolveTadpoles = {mHd2,mHu2,ms2};
```

This time, we want to specify two different boundary conditions:

a) Input for κ and λ are used at the SUSY scale, while A_κ and A_λ are used at the GUT scale.

b) κ, λ, A_κ and A_λ are used at the GUT scale.

For all other parameters, the boundaries are always set at the GUT scale. Only the singlet VEV is taken at the SUSY scale. For that reason, we initialize the arrays

```
BoundarySUSYScale = Table[{},{2}];
BoundaryHighScale = Table[{},{2}];
```

First, the two sets of conditions at the SUSY scale.

```
BoundarySUSYScale[[1]] = {
  {vS,             vSInput},
  {\[Kappa],   KappaInput},
  {\[Lambda],  LambdaInput}
};

BoundarySUSYScale[[2]] = {
  {vS,             vSInput}
};
```

Second, the GUT scale conditions

```
BoundaryHighScale[[1]]={
  {T[Ye],               Azero*Ye},
  {T[Yd],               Azero*Yd},
  {T[Yu],               Azero*Yu},
  {mq2,                DIAGONAL m0^2},
  {ml2,                DIAGONAL m0^2},
  {md2,                DIAGONAL m0^2},
  {mu2,                DIAGONAL m0^2},
  {me2,                DIAGONAL m0^2},
  {T[\[Kappa]],     AKappaInput*\[Kappa]},
  {T[\[Lambda]],    ALambdaInput*\[Lambda]},
  {MassB,              m12},
  {MassWB,             m12},
  {MassG,              m12}
};

BoundaryHighScale[[2]]={
  {T[Ye],               Azero*Ye},
  {T[Yd],               Azero*Yd},
  {T[Yu],               Azero*Yu},
  {mq2,                DIAGONAL m0^2},
  {ml2,                DIAGONAL m0^2},
  {md2,                DIAGONAL m0^2},
  {mu2,                DIAGONAL m0^2},
  {me2,                DIAGONAL m0^2},
  {\[Kappa],         KappaInput},
  {\[Lambda],        LambdaInput},
  {T[\[Kappa]],     AKappaInput*KappaInput},
  {T[\[Lambda]],    ALambdaInput*LambdaInput},
  {MassB,              m12},
  {MassWB,             m12},
  {MassG,              m12}
};
```

As for the MSSM, we calculate the VEVs dynamically when a low energy input is chosen.

```
BoundaryLowScaleInput={
 {vd,  Sqrt[2 mz2/(g1^2+g2^2)]*Sin[ArcTan[TanBeta]]},
 {vu,  Sqrt[2 mz2/(g1^2+g2^2)]*Cos[ArcTan[TanBeta]]}
};
```

Also for the calculation of the decay widths and branching ratios we make the same choice as in the MSSM.

```
ListDecayParticles = Automatic;
ListDecayParticles3B = Automatic;
```

F.5 Model files for Seesaw type I – III

For the model files, we show only the difference in comparison to the MSSM. For a discussion of seesaw I – III see Ref. [72] and references therein.

F.5.1 Seesaw I

Seesaw1.m

In the case of seesaw I, the particle content is extended by three generations of a gauge singlet $\hat{\nu}_R$.

```
Fields[[8]] = {vR,          3, v,     0,1,   1};
```

The new field causes a Yukawa like interaction and a mass term. In addition, there is an effective operator which gets initialized after integrating out the right handed neutrino.

```
SuperPotential =
   {...,{{1,Yv},{v,l,Hu}},{{1/2,Mv},{v,v}},{{1,MNuL},{l,Hu,l,Hu}}};
```

Since the field is heavy, it should not be included in the calculation of the vertices and masses at the SUSY scale. Therefore, we 'delete' it. Note, the low energy results like masses and vertices are independent of deleted particles. However, the RGEs are not since they have to be also valid at the GUT scale.

```
DeleteParticles={v};
```

SPheno.m

We choose a unification of the soft-breaking mass of the scalar singlet with the other soft-breaking masses at the GUT scale. In addition, we want to define the value of the superpotential parameters at the GUT scale as input values in the LesHouches file. Furthermore, we have also a mSugra like condition for the trilinear soft-breaking coupling. Since the bilinear soft-breaking term does not influence the RGE running of the other parameters, we can safely set it to zero. Now, the additional boundary conditions at the GUT scale are

```
BoundaryHighScale={
   ...,
   {mv2,      DIAGONAL m0^2},
   {Mv,       LHInput[Mv]},
   {Yv,       LHInput[Yv]}
```

```
  {B[Mv],    0},
  {T[Yv],    Azero*LHInput[Yv]}
};
```

We want to include three threshold scales: each generation of the gauge singlet should be integrated out at energies similar to their mass. Therefore, a good choice is

```
Thresholds={
  {Abs[MvIN[1,1]],{v[1]}},
  {Abs[MvIN[2,2]],{v[2]}},
  {Abs[MvIN[3,3]],{v[3]}}
};
```

When thresholds are included, it is possible to define for each threshold scale the boundary conditions separately for running up and down the RGEs. First, it is necessary to initialize the corresponding arrays

```
BoundaryConditionsUp=Table[{},{Length[Thresholds]}];
BoundaryConditionsDown=Table[{},{Length[Thresholds]}];
```

When a gauge singlet is integrated out, the effective operator receives a contribution of the form

$$\kappa = -Y_\nu^T\, M_N^{-1}\, Y_\nu \ . \tag{F.2}$$

When heavy superfields are integrated, the mass splitting between the fermionic and scalar component is neglected. The masses are calculated individually at each threshold scale and saved in arrays with the name

```
 MassOf <> Name of Superfield
```

Therefore, the contributions to the effective operator at the different scales are given by

```
BoundaryConditionsDown[[1]]={
  {MNuL[index1,index2], - Yv[3,index1] Yv[index2,3]/MassOfv[3]}};
BoundaryConditionsDown[[2]]={
  {MNuL[index1,index2], - Yv[2,index1] Yv[index2,2]/MassOfv[2]}};
BoundaryConditionsDown[[3]]={
  {MNuL[index1,index2], - Yv[1,index1] Yv[index2,1]/MassOfv[1]}};
```

F.5.2 Seesaw II

Seesaw2.m

For the seesaw II, it is necessary to add a scalar $SU(2)_L$ triplet which also carries hypercharge Y_i. Such a particle is part of the 15-plet of $SU(5)$. Therefore, we add to the $SU(5)$ invariant superpotential the interactions of a pair of **15** and $\overline{\mathbf{15}}$. After $SU(5)$ breaking, the **15** splits into irreducible representations of $SU(3)_C \times SU(2)_L \times U(1)_Y$

$$\mathbf{15} = \hat{S} + \hat{T} + \hat{Z} \ . \tag{F.3}$$

The corresponding quantum numbers are

$$\hat{S} : (\mathbf{6},\mathbf{1})_{-2/3} \ , \qquad \hat{T} : (\mathbf{1},\mathbf{3})_{1} \ , \qquad \hat{Z} : (\mathbf{3},\mathbf{2})_{1/6} \ . \tag{F.4}$$

These fields are implemented in
SARAH by

```
Fields[[8]] = {{{Tpp,1/Sqrt[2] Tp},{1/Sqrt[2] Tp, T0}}, 1, t, 1, 3, 1};
Fields[[9]] = {{{T0b,1/Sqrt[2] Tm},{1/Sqrt[2] Tm, Tmm}}, 1, tb, -1, 3, 1};
Fields[[10]] = {S, 1, s, -2/3, 1, 6};
Fields[[11]] = {conj[Sc], 1, sb, 2/3, 1, -6};
Fields[[12]] = {{z1,z2}, 1, z, 1/6, 2, 3};
Fields[[13]] = {{z1b,z2b}, 1, zb, -1/6, 2, -3};
```

The new terms in the superpotential after $SU(5)$ breaking are Again, after integrating out the colored Higgs fields, we end up with the superpotential terms

$$
\begin{aligned}
W^{II} &= \frac{1}{\sqrt{2}}\left(Y_T\,\hat{l}\,\hat{T}\,\hat{l} + Y_S\,\hat{d}\,\hat{S}\,\hat{d}\right) + Y_Z\,\hat{d}\,\hat{Z}\,\hat{l} + \frac{1}{\sqrt{2}}\lambda_1 \hat{H}_d \hat{T} \hat{H}_d + \frac{1}{\sqrt{2}}\lambda_2 \hat{H}_u \hat{\bar{T}} \hat{H}_u \\
&\quad + M_T \hat{T}\hat{\bar{T}} + M_Z \hat{Z}\hat{\bar{Z}} + M_S \hat{S}\hat{\bar{S}}
\end{aligned} \tag{F.5}
$$

This reads in
SARAH

```
SuperPotential = { ...,
 {{1/Sqrt[2],Yt},{l,t,l}},{{1/Sqrt[2],Ys},{d,s,d}},{{1,Yz},{d,z,l}},
 {{1/Sqrt[2],L1},{Hd,t,Hd}},{{1/Sqrt[2],L2},{Hu,tb,Hu}},
 {{1,MT},{t,tb}}, {{1,MZ},{z,zb}},{{1,MS},{s,sb}}, {{1,MNuL},{l,Hu,l,Hu}}};
```

All components of the 15-plet should be removed at the low scale, therefore we add

```
DeleteParticles={t,tb,s,sb,z,zb};
```

SPheno.m

We define as new parameters which can be adjusted by the LesHouches input file the values for λ_1, λ_2 and M_T

```
EXTPAR={{200,Lambda1IN},
        {201,Lambda2IN},
        {210,MTScaleIN}};
```

In addition, we use again mSugra boundary conditions which are also $SU(5)$ invariant with respect to the new parameters. It is again possible to neglect the bilinear soft-breaking terms corresponding to the mass terms in the superpotential.

```
BoundaryHighScale={
  ...,
  {mt2,    m0^2},
  {mtb2,   m0^2},
  {ms2,    m0^2},
  {msb2,   m0^2},
  {mz2,    m0^2},
  {mzb2,   m0^2},
  {MT,      MTScaleIN},
  {MZ,      MT},
  {MS,      MT},
  {B[MZ],  0},
  {B[MS],  0},
  {B[MT],  0},
  {Ys,      Yt},
  {Yz,      Yt},
  {T[Yt],  Azero*Yt},
  {T[Ys],  Azero*Yt},
  {T[Yz],  Azero*Yt},
  {T[L1],  Azero*L1},
  {T[L2],  Azero*L2}
};
```

Since, we have only one 15-plet, we need only one threshold scale.

```
Thresholds={
  {Abs[MTScaleIN],{s,sb,t,tb,z,zb}}
};
```

As boundary conditions, we choose this time that the input values of Y_t, λ_1 and λ_2 are used at the threshold scale. Of course, we initialize again the Weinberg operator when crossing the threshold. This time, the analytical expression is

$$\kappa = \frac{1}{2}\lambda_2 M_T^{-1} Y_T \tag{F.6}$$

The corresponding lines in the input file are

```
BoundaryConditionsUp=Table[{},{Length[Thresholds]}];
BoundaryConditionsDown=Table[{},{Length[Thresholds]}];

BoundaryConditionsUp[[1]] = {
  {Yt, LHInput[Yt]},
  {L1, Lambda1IN},
  {L2, Lambda2IN}
};

BoundaryConditionsDown[[1]] = {
  {MNuL, -L2 Yt/MT }
};
```

F.5.3 Seesaw III

Seesaw3.m

The type III seesaw is based on additional fields belonging to the adjoint representation of $SU(2)_L$. Hence, we add particles sitting in the 24-plet, the adjoint representation of $SU(5)$, to the spectrum. It is not sufficient to add just one generation of 24-plets to explain all neutrino data if we assume $SU(5)$ invariant

boundary conditions at the GUT scale: the induced mass splitting between the different generations of neutrinos won't be large enough. Therefore, we will add three generations of 24-plets.
The **24** has the same gauge quantum numbers as the gauge bosons of $SU(5)$ and can be decomposed in SM representations by

$$24_M = \hat{G}_M + \hat{W}_M + \hat{B}_M + \hat{X}_M + \hat{\bar{X}}_M \tag{F.7}$$

with

$$\hat{G}_M : (\mathbf{8},\mathbf{1})_0\,,\ \hat{W}_M : (\mathbf{1},\mathbf{3})_0\,,\ \hat{B}_M : (\mathbf{1},\mathbf{1})_0\,, \hat{X}_M : (\mathbf{3},\mathbf{2})_{-5/6}\,,\ \hat{\bar{X}}_M : (\bar{\mathbf{3}},\mathbf{2})_{5/6}\ . \tag{F.8}$$

Hence, we have to add to the
SARAH input file the following field definitions

```
Fields[[8]]  = {{{HW0/Sqrt[2],HWp},{HWm, -HW0/Sqrt[2]}}, 3, Hw3, 0, 3, 1};
Fields[[9]]  =  {HG, 3, Hg3, 0, 1, 8};
Fields[[10]] = {HB, 3, Hb3, 0, 1, 1};
Fields[[11]] = {{HXu,HXd}, 3, Hx3, 5/6, 2, -3};
Fields[[12]] = {{HXub,HXdb}, 3, Hxb3, -5/6, 2, 3};
```

The new terms in the superpotential are

$$\begin{aligned} W^{III} \quad &= \quad Y_W \hat{H}_u \hat{W}_M \hat{l} - \sqrt{\frac{3}{10}} Y_B \hat{H}_u \hat{B}_M \hat{l} + Y_X \hat{H}_u \hat{\bar{X}}_M \hat{d} + \\ & \quad\ \frac{1}{2} M_B \hat{B}_M \hat{B}_M + \frac{1}{2} M_G \hat{G}_M \hat{G}_M + \frac{1}{2} M_W \hat{W}_M \hat{W}_M + M_X \hat{X}_M \hat{\bar{X}}_M\ . \end{aligned} \tag{F.9}$$

These terms and the Weinberg operator read in
SARAH

```
SuperPotential = { ..,
 {{Sqrt[6/20],Yb3},{Hu,Hb3,l}},{{1,Yw3},{Hu,Hw3,l}},{{1,Yx3},{Hu,Hxb3,d}},
  {{1,MXM3},{Hx3,Hxb3}},{{1/2,MWM3},{Hw3,Hw3}},{{1/2,MGM3},{Hg3,Hg3}},
  {{1/2,MBM3},{Hb3,Hb3}},  {{1,MNuL},{l,Hu,l,Hu}}};
```

Again, all additional fields are removed at the low scale

```
DeleteParticles={Hw3,Hg3,Hb3,Hx3,Hxb3};
```

SPheno.m

We use here the same kind of boundary conditions at the GUT scale as for type I and II: mSugra-like and $SU(5)$ invariant.

```
BoundaryHighScale={
...,
   {mHw32,     DIAGONAL m0^2},
   {mHx32,     DIAGONAL m0^2},
   {mHxb32,    DIAGONAL m0^2},
   {mHg32,     DIAGONAL m0^2},
   {MWM3,       LHInput[MWM3]},
  {MXM3,      MWM3},
  {MBM3,      MWM3},
  {MGM3,      MWM3},
  {B[MWM3],  0},
  {B[MXM3],  0},
  {B[MBM3],  0},
  {B[MGM3],  0},
  {Yb3,       LHInput[Yb3]},
  {Yw3,       Yb3},
  {Yx3,       Yb3},
  {T[Yw3],   Azero*Yb3},
  {T[Yx3],   Azero*Yb3},
  {T[Yb3],   Azero*Yb3}
};
```

These boundary conditions have the effect that the mass splitting between the different members of the 24-plet is not very large. Therefore, we can integrate one generation of 24-plets at each scale:

```
Thresholds={
  {Abs[MWM3IN[1,1]],{Hx3[1],Hxb3[1],Hg3[1],Hb3[1],Hw3[1]}},
  {Abs[MWM3IN[2,2]],{Hx3[2],Hxb3[2],Hg3[2],Hb3[2],Hw3[2]}},
  {Abs[MWM3IN[3,3]],{Hx3[3],Hxb3[3],Hg3[3],Hb3[3],Hw3[3]}}
};
```

Finally, we need the boundary conditions. Since the value of the Weinberg operator at the threshold scale is the sum of seesaw I and III contributions, we have

$$\kappa = -\left(\frac{1}{2}Y_W^T M_W^{-1} Y_W + \frac{3}{10} Y_B^T M_B^{-1} Y_B\right) \ . \tag{F.10}$$

That's equivalent to the the following definitions in SPheno.m

```
BoundaryConditionsUp=Table[{},{Length[Thresholds]}];
BoundaryConditionsDown=Table[{},{Length[Thresholds]}];

BoundaryConditionsDown[[1]]={
  {MNuL[index1,index2],MNuL[index1,index2] +1/2 Yw3[3,index1]
     Yw3[index2,3]/MassOfHw3[3] +3/10 Yb3[3,index1] Yb3[index2,3]/MassOfHb3[3]}
};

BoundaryConditionsDown[[2]]={
  {MNuL[index1,index2],MNuL[index1,index2] + 1/2 Yw3[2,index1]
     Yw3[index2,2]/MassOfHw3[2] +3/10 Yb3[2,index1] Yb3[index2,2]/MassOfHb3[2]}
};

BoundaryConditionsDown[[3]]={
  {MNuL[index1,index2], 1/2 Yw3[1,index1] Yw3[index2,1]/MassOfHw3[1] +
     3/10 Yb3[1,index1] Yb3[index2,1]/MassOfHb3[1]}
};
```

F.6 Implementation of a model with a gauge symmetry breaking scale in SARAH and SPheno

We show here the implementation of a left-right supersymmetric model in SPheno and SARAH in detail. This discussion is based on the model presented in [73]. Since we are here mainly interested in the implementation in SARAH, it is sufficient to simplify the model a bit by just choosing one threshold scale and not two as discussed in [73]. Adding the second threshold scale is straightforward, but would lead to some redundancy in the following.

F.6.1 Summary of the model

We give here only a short summary about the model and refer for more details to [73] and references therein. As mentioned, we use in the following only one threshold scale at which $\times SU(2)_R \times U(1)_{B-L}$ gets broken

$$SU(2)_L \times SU(2)_R \times U(1)_{B-L} \to SU(2)_L \times U(1)_Y \tag{F.11}$$

From GUT scale to $SU(2)_R \times U(1)_{B-L}$ breaking scale The MSSM particle content above the threshold is extended by the presence of four fields which are triplets under $SU(2)_L$ or $SU(2)_R$ and which carry a $B-L$ charge. In addition, there are two triplets which are uncharged under $B-L$. Furthermore, the right handed neutrino are part of the spectrum and the Higgs fields are arranged in so called bi-doublets Φ. To get a non-trivial CKM matrix, we need at least two generations of Φ fields. The particle content is summarized as follows

Superfield	generations	$SU(3)_c$	$SU(2)_L$	$SU(2)_R$	$U(1)_{B-L}$
Q	3	3	2	1	$\frac{1}{3}$
Q^c	3	$\bar{3}$	1	2	$-\frac{1}{3}$
L	3	1	2	1	-1
L^c	3	1	1	2	1
Φ	2	1	2	2	0
Δ	1	1	3	1	2
$\bar{\Delta}$	1	1	3	1	-2
Δ^c	1	1	1	3	-2
$\bar{\Delta}^c$	1	1	1	3	2
Ω	1	1	3	1	0
Ω^c	1	1	1	3	0

The superpotential for the model reads

$$\begin{aligned}\mathcal{W} &= Y_Q Q\Phi Q^c + Y_L L\Phi L^c - \frac{\mu}{2}\Phi\Phi + fL\Delta L + f^* L^c\Delta^c L^c \\ &+ a\Delta\Omega\bar{\Delta} + a^*\Delta^c\Omega^c\bar{\Delta}^c + \alpha\Omega\Phi\Phi + \alpha^*\Omega^c\Phi\Phi \\ &+ M_\Delta\Delta\bar{\Delta} + M_\Delta^*\Delta^c\bar{\Delta}^c + M_\Omega\Omega\Omega + M_\Omega^*\Omega^c\Omega^c\,.\end{aligned} \tag{F.12}$$

Below $SU(2)_R \times U(1)_{B-L}$ breaking scale Here, we are left with the MSSM plus the effective Weinberg operator which causes neutrino masses after EWSB.

Boundary conditions To link both scales, we need the following set of boundary conditions:

$$Y_d = Y_Q^1 \cos\theta_1 - Y_Q^2 \sin\theta_1 \ , \qquad Y_u = -Y_Q^1 \cos\theta_2 + Y_Q^2 \sin\theta_2 \ , \tag{F.13}$$

$$Y_e = Y_L^1 \cos\theta_1 - Y_L^2 \sin\theta_1 \ , \qquad Y_\nu = -Y_L^1 \cos\theta_2 + Y_L^2 \sin\theta_2 \ , \tag{F.14}$$

where $R = \sin(\theta_1 - \theta_2)$. For the soft-trilinear couplings, we must just replace Y by T in the expressions. For the sfermionic soft masses, we have

$$m_q^2 = m_{u^c}^2 = m_{d^c}^2 \ = \ m_{Q^c}^2 \ , \tag{F.15}$$

$$m_l^2 = m_{e^c}^2 \ = \ m_{L^c}^2 \ , \tag{F.16}$$

$$M_L = M_R \ = \ M_2 \ . \tag{F.17}$$

while we need in the Higgs sector the relations

$$m_{H_d}^2 \ = \ \cos^2\theta_1 (m_\Phi^2)_{11} + \sin^2\theta_1 (m_\Phi^2)_{22} - \sin\theta_1 \cos\theta_1 \left[(m_\Phi^2)_{12} + (m_\Phi^2)_{21} \right] \ , \tag{F.18}$$

$$m_{H_u}^2 \ = \ \cos^2\theta_2 (m_\Phi^2)_{11} + \sin^2\theta_2 (m_\Phi^2)_{22} - \sin\theta_2 \cos\theta_2 \left[(m_\Phi^2)_{12} + (m_\Phi^2)_{21} \right] \ , \tag{F.19}$$

In the gauge sector, we have to express the hypercharge coupling and the corresponding gaugino by

$$g_1 \ = \ \frac{\sqrt{5} g_2 g_{BL}}{\sqrt{2g_2^2 + 3g_{BL}^2}} \ , \tag{F.20}$$

$$M_1 \ = \ \frac{2g_2^2 M_1 + 3g_{BL}^2 M_R}{2g_2^2 + 3g_{BL}^2} \ . \tag{F.21}$$

F.6.2 Model files for SARAH

We present in the following the input files for SARAH to define the model at the different scales. For shortness, we concentrate here on the parts necessary for the SPheno output and skip the gauge fixing terms and definition of Dirac spinors above the threshold scale.

From GUT scale to $SU(2)_R \times U(1)_{B-L}$ breaking scale

The vector and chiral superfields of the highest scale define the gauge sector and particle content are

```
Gauge[[1]]={B,    U[1], bminl,         gBL,False};
Gauge[[2]]={WL,  SU[2], left ,         g2,True};
Gauge[[3]]={WR,  SU[2], right ,        g2,True};
Gauge[[4]]={G,   SU[3], color ,        g3,False};

Fields[[1]] = {{uL,  dL},                              3, qL,         1/6, 2, 1, 3};
Fields[[2]] = {{conj[dR], - conj[uR]},                 3, qR,        -1/6, 1, 2,-3};
Fields[[3]] = {{vL,  eL},                              3, lL,        -1/2, 2, 1, 1};
Fields[[4]] = {{conj[eR],  - conj[vR]},                3, lR,         1/2, 1, 2, 1};
Fields[[5]] = {{{Hd0, Hup},{Hdm, Hu0}},                2, Phi,          0, 2,-2, 1};
Fields[[6]] = {{{deltaLp/Sqrt[2], deltaLpp},
                 {deltaL0, - deltaLp/Sqrt[2]}},        1, deltaL,       1, 3, 1, 1};
Fields[[7]] = {{{deltaLbarm/Sqrt[2], deltaLbar0},
               {deltaLbarmm, - deltaLbarm/Sqrt[2]}}, 1, deltaLbar, -1, 3, 1, 1};
Fields[[8]] = {{{deltaRm/Sqrt[2], deltaR0},
               {deltaRmm, - deltaRm/Sqrt[2]}},         1, deltaR,      -1, 1, 3, 1};
```

```
Fields[[9]]  = {{{deltaRbarp/Sqrt[2], deltaRbarpp},
               {deltaRbar0, - deltaRbarp/Sqrt[2]}},  1, deltaRbar,  1, 1, 3, 1};
Fields[[10]] = {{{omegaL0/Sqrt[2], omegaLp},
               {omegaLm, - omegaL0/Sqrt[2]}},        1, omegaL,     0, 3, 1, 1};
Fields[[11]] = {{{omegaR0/Sqrt[2], omegaRp},
               {omegaRm, - omegaR0/Sqrt[2]}},        1, omegaR,     0, 1, 3, 1};
```

The superpotential reads

```
SuperPotential = { {{1, YQ},              {qL,qR,Phi}},
                   {{1, YL},              {lL,lR,Phi}},
                   {{1,  f},              {lL,deltaL,lL}},
                   {{1,conj[f]},          {lR,deltaR,lR}},
                   {{1,Mdelta},           {deltaL,deltaLbar}},
                   {{1,conj[Mdelta]},     {deltaR,deltaRbar}},
                   {{-1/2,Mu3},           {Phi,Phi}},
                   {{1,Momega},           {omegaL,omegaL}},
                   {{1,conj[Momega]},     {omegaR,omegaR}},
                   {{1,a},                {deltaL,omegaL,deltaLbar}},
                   {{1,conj[a]},          {deltaR,omegaR,deltaRbar}},
                   {{1,AlphaOm},          {omegaL,Phi,Phi}},
                   {{1,conj[AlphaOm]},    {omegaR,Phi,Phi}}    };
```

The gauge bosons and gauginos of the right sector decompose into

```
DEFINITION[RSB][GaugeSector]=
{ {{VWR[1],VWR[2]},{VWRm,conj[VWRm]},ZW},
  {{VB,VWR[3]},{VBY,VZ2},ZZ},
  {{fWR[1],fWR[2],fWR[3]},{fWRm,fWRp,fWR0},ZfW}
};
```

after the omega and delta fields have received their VEV

```
DEFINITION[RBLSB][VEVs]=
{ {SomegaR0,     {vR,1/Sqrt[2]},  {sigmaOmR,I/Sqrt[2]}, {phiOmR,1/Sqrt[2]}},
  {SdeltaR0,     {vBL,1/Sqrt[2]}, {sigmaR,I/Sqrt[2]},    {phiR,1/Sqrt[2]}},
  {SdeltaRbar0,  {vBL,1/Sqrt[2]}, {sigmaRbar,I/Sqrt[2]},{phiRbar,1/Sqrt[2]}}};
```

Finally, we need the rotations in the matter sector to the new mass eigenstates

```
DEFINITION[RSB][MatterSector]=
{     {{SdeltaRm, conj[SdeltaRbarp]},                {Hpm1R1,ZC1}},
      {{SomegaRm, conj[SomegaRp]},                   {Hpm2R1,ZC2}},
      {{fB,fWR0,FdeltaR0, FdeltaRbar0, FomegaR0},    {L0, ZN}},
      {{{fWRm, FomegaRm}, {fWRp, FomegaRp}},         {{Lm,UM}, {Lp,UP}}},
      {{phiR, phiRbar, phiOmR},                      {hhR2, ZH}},
      {{sigmaR, sigmaRbar, sigmaOmR},                {AhR2, ZP}},
      {{FvL, conj[FvR]},                             {N0, Znu}},
      {{SHd0,conj[SHu0]},                            {SH0r1,UH0}},
      {{SHdm,conj[SHup]},                            {SHCr1,UHC}},
      {{SomegaLm,conj[SomegaLp]},                    {SO1r1,UO1}},
      {{SdeltaLp,conj[SdeltaLbarm]},                 {SDLpR1,UDLp}},
      {{SdeltaLpp,conj[SdeltaLbarmm]},               {SDLppR1,UDLpp}},
      {{SdeltaL0,conj[SdeltaLbar0]},                 {SDL0r1,UDL0}},
      {{SdeltaRmm,conj[SdeltaRbarpp]},               {SDRmmR1,UDRmm}},
      {{SdeltaR0,conj[SdeltaRbar0]},                 {SDR0r1,UDR0}}
       };
```

Below $SU(2)_R \times U(1)_{B-L}$ breaking scale

Here, we have just the particles and gauge groups of the MSSM. Therefore, the model file nearly the same in the one presented in sec. F.4. We point only out the difference:

- The superpotential contains the Weinberg operator

```
SuperPotential = {...,{{1,WOp},{1,Hu,1,Hu}}  };
```

- The neutrinos are massive and mix among each other

```
DEFINITION[EWSB][MatterSector]= { ...,{{FvL}, {FV, UV}}};
```

- These states form Majorana spinors

```
dirac[[4]] = {Fv,  FV, conj[FV]};
```

F.6.3 Model files for SPheno output

We need for both scales the corresponding SPheno.m files for
SARAH to create the Fortran source code. While the first one is rather short, the second one includes all necessary boundary conditions.

Above $SU(2)_R \times U(1)_{B-L}$ breaking scale

First, it is necessary to tell
SARAH the number of the regime (counted from GUT to low scale) and that it is an intermediate scale.

```
RegimeNr = 1;
IntermediateScale = True;
```

Afterwards, we give a list with all particles which are integrated out at the threshold scale after gauge symmetry breaking.

```
HeavyFields = {Hpm1R1, ChiR1, Cha1r1, hhR1, AhR1,
               FvR1, SVRr1, SH0r1[3], SHCr1[3],
               SO1r1, SDLpR1, SDLppR1,
               SDL0r1, SDRmmR1, DR3r1,
               DL1r1, DL2r1, DL3r1, H0r1, HCr1};
```

The number in bracket coming with SH0r1 and SHCr1 means that only the third generation and above is integrated out.
To calculate the finite shifts of the gauge couplings and gaugino masses, it is necessary to define the gauge sector of the next scale NextGauge as well as the quantum number of the fields which are integrated out with respect to that gauge groups NextQN.

```
NextGauge=          {U[1], SU[2], SU[3]};
NextQN = { {Hpm1R1,     -1,     1,     1},
           {ChiR1,       0,     1,     1},
           {Cha1r1,     -1,     1,     1},
           {hhR1,        0,     1,     1},
           {AhR1,        0,     1,     1},
           {FvR1,        0,     1,     1},
           {SVRr1,       0,     1,     1},
           {SH0r1,    -1/2,     1,     1},
           {SHCr1,     1/2,     2,     1},
           {SO1r1,       0,     1,     1},
           {SDLpR1,      1,     1,     1},
           {SDLppR1,     2,     1,     1},
           {SDL0r1,      1,     3,     1},
           {SDRmmR1,    -2,     1,     1},
           {DR3r1,      -2,     1,     1},
           {DL1r1,       1,     1,     1},
           {DL2r1,       1,     1,     1},
           {DL3r1,       1,     3,     1},
            {H0r1,      -1/2,     1,     1},
            {HCr1,       1/2,     2,     1}
};
```

Finally, it is necessary to define the information about the vacuum conditions. We have here two different VEVs and therefore need parameter which should be fixed by the tadpole equations. Since, there is no closed analytical solution for that, we give an approximated expression by neglecting the soft-breaking terms.

```
ParametersToSolveTadpoles = {Mdelta, Momega};
UseGivenTadpoleSolution = True;

SubSolutionsTadpolesTree={
  Mdelta -> - SignumMdelta ac1 vR/Sqrt[2],
  Momega -> - SignumMomega ac1 vBL^2/(2 Sqrt[2] vR)
};
SubSolutionsTadpolesLoop={};
```

Below $SU(2)_R \times U(1)_{B-L}$ breaking scale

The second scale is not an intermediate scale. Hence,

```
RegimeNr = 2;
IntermediateScale = False;
```

We make the following choice of free parameters of that model: in addition to the standard mSugra parameters $(m_0, M_{1/2}, A_0, \tan\beta, \text{sign}\mu)$, we need input values for B_0, the superpotential parameter a, the signs of M_Ω and M_Δ, the two VEVs v_R and v_{BL} as well as the threshold scale. These are defined in the blocks `MINPAR` and `EXTPAR`.

```
MINPAR= {
          {1,  m0},
          {2,  m12},
          {3,  TanBeta},
          {4,  SignumMu},
          {5,  Azero},
          {6,  Bzero},
          {7,  SignumMomega},
          {8,  SignumMdelta},
          {9,  aInput}};

EXTPAR = {
           {100, vRinput},
           {101, vBLinput},
           {200, TScale}};
```

As in the usual MSSM we solve the tadpole equations for v_d and v_u with respect to μ and B_μ. Furthermore, we use also the some definition for the SUSY scale.

```
ParametersToSolveTadpoles = {\[Mu],B[\[Mu]]};

RenormalizationScaleFirstGuess = m0^2 + 4 m12^2;
RenormalizationScale = MSu[1]*MSu[6];
```

At the GUT scale we use boundary conditions motivated by minimal supergravity: all scalar soft-breaking masses are proportional to m_0, the gaugino masses are proportional to $M_{1/2}$, the trilinear soft-breaking couplings are given by the corresponding superpotential parameter times A_0 and the bilinear soft-breaking couplings are B_0 times the superpotential parameter. In addition, the values for the coupling matrices f, α as well as for μ are also read from the LesHouches input file.

```
BoundaryHighScale={
    {T[YQ],            Azero*YQ},
    {T[YL],            Azero*YL},
    {f,                LHInput[f]},
    {T[f],             Azero*LHInput[fm]},
    {AlphaOm,          LHInput[AlphaOm]},
    {T[AlphaOm],       Azero*LHInput[AlphaOm]},
    {T[a],             Azero*aInput},
    {B[Mdelta],        Bzero*Mdelta},
    {B[Momega],        Bzero*Momega},
    {B[Mu3],           Bzero*LHInput[Mu3]},
    {mqL2,             DIAGONAL m0^2},
    {mqR2,             DIAGONAL m0^2},
    {mlL2,             DIAGONAL m0^2},
    {mlR2,             DIAGONAL m0^2},
    {mPhi2,            DIAGONAL m0^2},
    {mdeltaL2,         m0^2},
    {mdeltaLbar2,      m0^2},
    {mdeltaR2,         m0^2},
    {mdeltaRbar2,      m0^2},
    {momegaL2,         m0^2},
    {momegaR2,         m0^2},
    {MassB,            m12},
    {MassWL,           m12},
    {MassG,            m12}
};
```

To glue the both regimes, we need to define the appropriate boundary conditions. First, we initialize the arrays

```
ThresholdScales = {TSCALE};

BoundaryConditionsUp = Table[{},{Length[ThresholdScales]}];
BoundaryConditionsDown = Table[{},{Length[ThresholdScales]}];
```

and use afterwards the equations eqs. (F.13)-(F.21). In order to keep the code short, we define

```
  ST1  = Sin[Theta1];
  CT1  = Cos[Theta1];
  ST2  = Sin[Theta2];
  CT2  = Cos[Theta2];
  ST21 = Sin[Theta2-Theta1];
  CT21 = Cos[Theta2-Theta1];
```

Using these abbreviation, the boundary conditions can be written as

```
BoundaryConditionsUp[[1]] = {
 {YQ[index1,index2,1],     (Yu[index1,index2] ST1 + Yd[index1,index2]ST2)/ST21 },
 {YQ[index1,index2,2],     (Yu[index1,index2] CT1 + Yd[index1,index2]CT2)/ST21 },
 {YL[index1,index2,1],     (Yv[index1,index2] ST1 + Ye[index1,index2]ST2)/ST21 },
 {YL[index1,index2,2],     (Yv[index1,index2] CT1 + Ye[index1,index2]CT2)/ST21 },
 {gBL,                      Sqrt[2] g1 g2 /Sqrt[5 g2^2 -3 g1^2]},
 {Yv,                        LHInput[Yv]}
};

BoundaryConditionsDown[[1]] = {
 {vR,               vRinput},
 {vBL,              vBLinput},
 {a,                aInput},
 {Theta1,           ArcTan[RealPart[((vR*AlphaOm[1,2])/2 + Mu3[1,2])/Mu3[2,2]]]},
 {Theta2,           ArcTan[RealPart[(-(vR*AlphaOm[1,2])/2 + Mu3[1,2])/Mu3[2,2]]]},
 {g1,               Sqrt[5] g2 gBL/Sqrt[2 g2^2 + 3 gBL^2]},
 {MassB,            (2 g2^2 MassB + 3 gBL^2 MassWL)/(2 g2^2 + 3 gBL^2)},
 {MassWB,           MassWL},
 {Yd[index1,index2], YQ[index1,index2,1] CT1 - YQ[index1,index2,2] ST1},
 {Yu[index1,index2], - YQ[index1,index2,1] CT2 + YQ[index1,index2,2] ST2},
 {Ye[index1,index2], YL[index1,index2,1] CT1 - YL[index1,index2,2] ST1},
 {Yv[index1,index2], - YL[index1,index2,1] CT2 + YL[index1,index2,2] ST2},
 {T[Yd][index1,index2], T[YQ][index1,index2,1] CT1 -T[YQ][index1,index2,2]ST1},
 {T[Yu][index1,index2], -T[YQ][index1,index2,1] CT2+T[YQ][index1,index2,2] ST2},
 {T[Ye][index1,index2], T[YL][index1,index2,1] CT1 -T[YL][index1,index2,2] ST1},
 {mu2,          mqR2},
 {md2,          mqR2},
 {mq2,          mqR2},
 {me2,          mlR2},
 {ml2,          mlR2},
 {mHd2, CT1^2 mPhi2[1,1] + ST1^2 mPhi2[2,2] - ST1 CT1(mPhi2[1,2] + mPhi2[2,1])},
 {mHu2, CT2^2 mPhi2[1,1] + ST2^2 mPhi2[2,2] - ST2 CT2(mPhi2[1,2] + mPhi2[2,1])},
 {WOp,          MatMul2[MatMul2[Yv,InverseMatrix[f],FortranFalse],Transpose[Yv],
                  FortranFalse]/vR}
 };
```

Note, Y_ν does not appear in one of the model files. Therefore, it is necessary to define the dimension of that matrix by hand

```
AdditionalVariablesSPheno={Yv[3,3]};
```

There are some parameters involved which must be real values. It might be also helpful to choose some initialization values for some parameters to stabilize the numerics in the first iteration

```
RealParameters = {TanBeta, vRinput,vBLinput,Theta1,Theta2,TScale};

InitializationValues = {
 {Mu3IN[1,1], (Mu3IN[1,2]^2 - AlphaOmIN[1,2]^2 vRInput^2/4)/Mu3IN[2,2]},
 {Theta1,    ArcTan[RealPart[-(Mu3IN[1,2]+AlphaOmIN[1,2]vRInput/2)/Mu3IN[2,2]]]},
 {Theta2,    ArcTan[RealPart[(Mu3IN[1,2]-AlphaOmIN[1,2]vRInput/2)/Mu3IN[2,2]]]},
 {Mdelta,    aInput*SignumMdelta*vRinput/2 },
 {Momega,    SignumMomega*(aInput^2*vBLinput^2)/(8 Mdelta)}
};
```

Appendix G

The minimal supersymmetric standard model

G.1 Conventions

G.1.1 Vector Superfields

SF	Spin $\frac{1}{2}$	Spin 1	$SU(N)$	Coupling	Name
$\hat{B}$	$\lambda_{\tilde{B}}$	B	$U(1)$	g_1	hypercharge
$\hat{W}$	$\lambda_{\tilde{W}}$	W	SU(2)	g_2	left
$\hat{g}$	$\lambda_{\tilde{g}}$	g	SU(3)	g_3	color

G.1.2 Chiral Superfields

SF	Spin 0	Spin $\frac{1}{2}$	Generations	$(U(1)\otimes \mathrm{SU}(2)\otimes \mathrm{SU}(3))$
$\hat{q}$	$\tilde{q}$	q	3	$(\frac{1}{6},\mathbf{2},\mathbf{3})$
$\hat{l}$	$\tilde{l}$	l	3	$(-\frac{1}{2},\mathbf{2},\mathbf{1})$
$\hat{H}_d$	H_d	$\tilde{H}_d$	1	$(-\frac{1}{2},\mathbf{2},\mathbf{1})$
$\hat{H}_u$	H_u	$\tilde{H}_u$	1	$(\frac{1}{2},\mathbf{2},\mathbf{1})$
$\hat{d}$	$\tilde{d}_R^*$	d_R^*	3	$(\frac{1}{3},\mathbf{1},\overline{\mathbf{3}})$
$\hat{u}$	$\tilde{u}_R^*$	u_R^*	3	$(-\frac{2}{3},\mathbf{1},\overline{\mathbf{3}})$
$\hat{e}$	$\tilde{e}_R^*$	e_R^*	3	$(1,\mathbf{1},\mathbf{1})$

G.1.3 Superpotential and Lagrangian

G.1.3.0.4 Superpotential

$$W = Y_u\,\hat{u}\,\hat{q}\,\hat{H}_u\, - Y_d\,\hat{d}\,\hat{q}\,\hat{H}_d\, - Y_e\,\hat{e}\,\hat{l}\,\hat{H}_d\, + \mu\,\hat{H}_u\,\hat{H}_d \tag{G.1}$$

G.1.3.0.5 Softbreaking terms

$$\begin{aligned} -L_{SB,W} = & - H_d^0 H_u^0 B_\mu + H_d^- H_u^+ B_\mu + H_d^0 \tilde{d}^*_{R,i\alpha} \delta_{\alpha\beta} \tilde{d}_{L,j\beta} T_{d,ij} - H_d^- \tilde{d}^*_{R,i\alpha} \delta_{\alpha\beta} \tilde{u}_{L,j\beta} T_{d,ij} \\ & + H_d^0 \tilde{e}^*_{R,i} \tilde{e}_{L,j} T_{e,ij} - H_d^- \tilde{e}^*_{R,i} \tilde{\nu}_{L,j} T_{e,ij} - H_u^+ \tilde{u}^*_{R,i\alpha} \delta_{\alpha\beta} \tilde{d}_{L,j\beta} T_{u,ij} + H_u^0 \tilde{u}^*_{R,i\alpha} \delta_{\alpha\beta} \tilde{u}_{L,j\beta} T_{u,ij} + \text{h.c.} \end{aligned} \tag{G.2}$$

$$\begin{aligned} -L_{SB,\phi} = & + m_{H_d}^2 |H_d^0|^2 + m_{H_d}^2 |H_d^-|^2 + m_{H_u}^2 |H_u^0|^2 + m_{H_u}^2 |H_u^+|^2 + \tilde{d}^*_{L,j\beta} \delta_{\alpha\beta} m^2_{q,ij} \tilde{d}_{L,i\alpha} \\ & + \tilde{d}^*_{R,i\alpha} \delta_{\alpha\beta} m^2_{d,ij} \tilde{d}_{R,j\beta} + \tilde{e}^*_{L,j} m^2_{l,ij} \tilde{e}_{L,i} + \tilde{e}^*_{R,i} m^2_{e,ij} \tilde{e}_{R,j} + \tilde{u}^*_{L,j\beta} \delta_{\alpha\beta} m^2_{q,ij} \tilde{u}_{L,i\alpha} \\ & + \tilde{u}^*_{R,i\alpha} \delta_{\alpha\beta} m^2_{u,ij} \tilde{u}_{R,j\beta} + \tilde{\nu}^*_{L,j} m^2_{l,ij} \tilde{\nu}_{L,i} \end{aligned} \tag{G.3}$$

$$-L_{SB,\lambda} = \frac{1}{2} \Big(\lambda^2_{\tilde{B}} M_1 + M_2 \lambda^2_{\tilde{W},i} + M_3 \lambda^2_{\tilde{g},i} + \text{h.c.} \Big) \tag{G.4}$$

G.1.4 Gauge fixing terms

G.1.4.0.6 Gauge fixing terms for gauge eigenstates

$$L_{GF} = - \frac{1}{2\xi_G} \partial_\mu g_\alpha - \frac{1}{2\xi_W} \partial_\mu W^i \tag{G.5}$$

G.1.4.0.7 Gauge fixing terms for mass eigenstates after EWSB

$$L_{GF} = - \frac{1}{2\xi_P} \partial_\mu \gamma - \frac{1}{2\xi_G} \partial_\mu g_\alpha - \frac{1}{2\xi_Z} \Big(- A_1^0 m_Z \xi_Z + \partial_\mu Z \Big) - \frac{1}{\xi_W} \Big(i H_1^- m_{W^-} \xi_W + \partial_\mu W^- \Big) \tag{G.6}$$

G.1.5 Vacuum expectation values

$$H_d^0 = \frac{1}{\sqrt{2}} \left(\phi_d + i\sigma_d + v_d \right) , \qquad H_u^0 = \frac{1}{\sqrt{2}} \left(\phi_u + i\sigma_u + v_u \right) \tag{G.7}$$

G.1.6 Rotations of vector bosons and gauginos after EWSB

$$W^-_{1\rho} = \frac{1}{\sqrt{2}} W^-_\rho + \frac{1}{\sqrt{2}} W^+_\rho , \qquad W^-_{2\rho} = -i \frac{1}{\sqrt{2}} W^-_\rho + i \frac{1}{\sqrt{2}} W^+_\rho \tag{G.8}$$

$$W^-_{3\rho} = \cos\Theta_W Z_\rho + \sin\Theta_W \gamma_\rho , \qquad B_\rho = \cos\Theta_W \gamma_\rho - \sin\Theta_W Z_\rho \tag{G.9}$$

$$\lambda_{\tilde{W},1} = \frac{1}{\sqrt{2}} \tilde{W}^- + \frac{1}{\sqrt{2}} \tilde{W}^+ , \qquad \lambda_{\tilde{W},2} = -i \frac{1}{\sqrt{2}} \tilde{W}^- + i \frac{1}{\sqrt{2}} \tilde{W}^+ , \qquad \lambda_{\tilde{W},3} = \tilde{W}^0 \tag{G.10}$$

G.1.7 Rotations in matter sector to mass eigenstates after EWSB

In the following, Greek letters α_i, β_i refer to color indices and o_i, p_i to generations indices.

a) **Mass matrix for neutralinos**, basis: $\Big(\lambda_{\tilde{B}}, \tilde{W}^0, \tilde{H}_d^0, \tilde{H}_u^0 \Big)$

$$m_{\tilde{\chi}^0} = \begin{pmatrix} M_1 & 0 & -\frac{1}{2} g_1 v_d & \frac{1}{2} g_1 v_u \\ 0 & M_2 & \frac{1}{2} g_2 v_d & -\frac{1}{2} g_2 v_u \\ -\frac{1}{2} g_1 v_d & \frac{1}{2} g_2 v_d & 0 & -\mu \\ \frac{1}{2} g_1 v_u & -\frac{1}{2} g_2 v_u & -\mu & 0 \end{pmatrix} \tag{G.11}$$

This matrix is diagonalized by N:

$$N m_{\tilde{\chi}^0} N^\dagger = m^{dia}_{\tilde{\chi}^0} \tag{G.12}$$

with

$$\lambda_{\tilde{B}} = \sum_j N^*_{j1}\lambda^0_j\,, \qquad \tilde{W}^0 = \sum_j N^*_{j2}\lambda^0_j\,, \qquad \tilde{H}^0_d = \sum_j N^*_{j3}\lambda^0_j\,, \qquad \tilde{H}^0_u = \sum_j N^*_{j4}\lambda^0_j \tag{G.13}$$

b) Mass matrix for charginos, basis: $\left(\tilde{W}^-, \tilde{H}^-_d\right) / \left(\tilde{W}^+, \tilde{H}^+_u\right)$

$$m_{\tilde{\chi}^-} = \begin{pmatrix} M_2 & \frac{1}{\sqrt{2}} g_2 v_u \\ \frac{1}{\sqrt{2}} g_2 v_d & \mu \end{pmatrix} \tag{G.14}$$

This matrix is diagonalized by U and V

$$U^* m_{\tilde{\chi}^-} V^\dagger = m^{dia}_{\tilde{\chi}^-} \tag{G.15}$$

with

$$\tilde{W}^- = \sum_j U^*_{j1}\lambda^-_j\,, \qquad \tilde{H}^-_d = \sum_j U^*_{j2}\lambda^-_j\,, \qquad \tilde{W}^+ = \sum_j V^*_{1j}\lambda^+_j\,, \qquad \tilde{H}^+_u = \sum_j V^*_{2j}\lambda^+_j \tag{G.16}$$

c) Mass matrix for leptons, basis: $\left(e_{L,o_1}\right) / \left(e^*_{R,p_1}\right)$

$$m_e = \begin{pmatrix} \frac{1}{\sqrt{2}} v_d Y_{e,p_1 o_1} \end{pmatrix} \tag{G.17}$$

This matrix is diagonalized by U^e_L and U^e_R

$$U^{e,*}_L m_e U^{e,\dagger}_R = m^{dia}_e \tag{G.18}$$

with

$$e_{L,i} = \sum_j U^{e,*}_{L,ji} E_{L,j}\,, \qquad e_{R,i} = \sum_j U^e_{R,ij} E^*_{R,j} \tag{G.19}$$

d) Mass matrix for down-quarks, basis: $\left(d_{L,o_1\alpha_1}\right) / \left(d^*_{R,p_1\beta_1}\right)$

$$m_d = \begin{pmatrix} \frac{1}{\sqrt{2}} v_d \delta_{\alpha_1\beta_1} Y_{d,p_1 o_1} \end{pmatrix} \tag{G.20}$$

This matrix is diagonalized by U^d_L and U^d_R

$$U^{d,*}_L m_d U^{d,\dagger}_R = m^{dia}_d \tag{G.21}$$

with

$$d_{L,i\alpha} = \sum_{t_2} U^{d,*}_{L,ji} D_{L,j\alpha}\,, \qquad d_{R,i\alpha} = \sum_{t_2} U^d_{R,ij} D^*_{R,j\alpha} \tag{G.22}$$

e) Mass matrix for up-quarks, basis: $\left(u_{L,o_1\alpha_1}\right) / \left(u^*_{R,p_1\beta_1}\right)$

$$m_u = \left(\begin{array}{c} \frac{1}{\sqrt{2}} v_u \delta_{\alpha_1\beta_1} Y_{u,p_1 o_1} \end{array} \right) \tag{G.23}$$

This matrix is diagonalized by U^u_L and U^u_R

$$U^{u,*}_L m_u U^{u,\dagger}_R = m^{dia}_u \tag{G.24}$$

with

$$u_{L,i\alpha} = \sum_{t_2} U^{u,*}_{L,ji} U_{L,j\alpha}\,, \qquad u_{R,i\alpha} = \sum_{t_2} U^{u}_{R,ij} U^*_{R,j\alpha} \tag{G.25}$$

f) Mass matrix for down-squarks, basis: $\left(\tilde{d}_{L,o_1\alpha_1}/\tilde{d}_{R,o_2\alpha_2}\right), \left(\tilde{d}^*_{L,p_1\beta_1}, \tilde{d}^*_{R,p_2\beta_2}\right)$

$$m_{11} = \frac{1}{24}\delta_{\alpha_1\beta_1}\Big(12\Big(2m^2_{q,o_1p_1} + v_d^2 \sum_{a=1}^{3} Y^*_{d,ap_1} Y_{d,ao_1}\Big) - \Big(3g_2^2 + g_1^2\Big)\Big(- v_u^2 + v_d^2\Big)\delta_{o_1p_1}\Big) \tag{G.26}$$

$$m_{12} = \frac{1}{\sqrt{2}}\delta_{\alpha_1\beta_2}\Big(v_d T_{d,p_2o_1} - v_u \mu^* Y_{d,p_2o_1}\Big) \tag{G.27}$$

$$m_{22} = \frac{1}{12}\delta_{\alpha_2\beta_2}\Big(6\Big(2m^2_{d,p_2o_2} + v_d^2 \sum_{a=1}^{3} Y^*_{d,o_2a} Y_{d,p_2a}\Big) + g_1^2\Big(- v_d^2 + v_u^2\Big)\delta_{o_2p_2}\Big) \tag{G.28}$$

This matrix is diagonalized by Z^D:

$$Z^D m^2_{\tilde{d}} Z^{D,\dagger} = m^{dia}_{2,\tilde{d}} \tag{G.29}$$

with

$$\tilde{d}_{L,i\alpha} = \sum_{t_2} Z^{D,*}_{ji} \tilde{d}_{j\alpha}\,, \qquad \tilde{d}_{R,i\alpha} = \sum_{t_2} Z^{D,*}_{ji} \tilde{d}_{j\alpha} \tag{G.30}$$

g) Mass matrix for sneutrinos, basis: $\left(\tilde{\nu}_{L,o_1}\right) / \left(\tilde{\nu}^*_{L,p_1}\right)$

$$m^2_{\tilde{\nu}} = \left(\begin{array}{c} \frac{1}{8}\Big(8m^2_{l,o_1p_1} + \Big(g_1^2 + g_2^2\Big)\Big(- v_u^2 + v_d^2\Big)\delta_{o_1p_1}\Big) \end{array} \right) \tag{G.31}$$

This matrix is diagonalized by Z^V:

$$Z^V m^2_{\tilde{\nu}} Z^{V,\dagger} = m^{dia}_{2,\tilde{\nu}} \tag{G.32}$$

with

$$\tilde{\nu}_{L,i} = \sum_{t_2} Z^{V,*}_{ji} \tilde{\nu}_j \tag{G.33}$$

h) Mass matrix for up-squarks, basis: $(\tilde{u}_{L,o_1\alpha_1}, \tilde{u}_{R,o_2\alpha_2}) \,/\, \Big(\tilde{u}^*_{L,p_1\beta_1}, \tilde{u}^*_{R,p_2\beta_2}\Big)$

$$m_{11} = \frac{1}{24}\delta_{\alpha_1\beta_1}\Big(12\Big(2m^2_{q,o_1p_1} + v_u^2\sum_{a=1}^{3}Y^*_{u,ap_1}Y_{u,ao_1}\Big) - \Big(-3g_2^2 + g_1^2\Big)\Big(-v_u^2 + v_d^2\Big)\delta_{o_1p_1}\Big) \tag{G.34}$$

$$m_{12} = \frac{1}{\sqrt{2}}\delta_{\alpha_1\beta_2}\Big(-v_d\mu^*Y_{u,p_2o_1} + v_uT_{u,p_2o_1}\Big) \tag{G.35}$$

$$m_{22} = \frac{1}{6}\delta_{\alpha_2\beta_2}\Big(3v_u^2\sum_{a=1}^{3}Y^*_{u,o_2a}Y_{u,p_2a} + 6m^2_{u,p_2o_2} + g_1^2\Big(-v_u^2 + v_d^2\Big)\delta_{o_2p_2}\Big) \tag{G.36}$$

This matrix is diagonalized by Z^U:

$$Z^U m^2_{\tilde{u}} Z^{U,\dagger} = m^{dia}_{2,\tilde{u}} \tag{G.37}$$

with

$$\tilde{u}_{L,i\alpha} = \sum_{t_2} Z^{U,*}_{ji}\tilde{u}_{j\alpha}\,, \qquad \tilde{u}_{R,i\alpha} = \sum_{t_2} Z^{U,*}_{ji}\tilde{u}_{j\alpha} \tag{G.38}$$

i) Mass matrix for sleptons, basis: $(\tilde{e}_{L,o_1}, \tilde{e}_{R,o_2}) \,/\, (\tilde{e}^*_{L,p_1}, \tilde{e}^*_{R,p_2})$

$$m_{11} = \frac{1}{8}\Big(4v_d^2\sum_{a=1}^{3}Y^*_{e,ap_1}Y_{e,ao_1} + 8m^2_{l,o_1p_1} + \Big(-g_2^2 + g_1^2\Big)\Big(-v_u^2 + v_d^2\Big)\delta_{o_1p_1}\Big) \tag{G.39}$$

$$m_{12} = \frac{1}{\sqrt{2}}\Big(v_dT_{e,p_2o_1} - v_u\mu^*Y_{e,p_2o_1}\Big) \tag{G.40}$$

$$m_{22} = \frac{1}{4}\Big(2v_d^2\sum_{a=1}^{3}Y^*_{e,o_2a}Y_{e,p_2a} + 4m^2_{e,p_2o_2} + g_1^2\Big(-v_d^2 + v_u^2\Big)\delta_{o_2p_2}\Big) \tag{G.41}$$

This matrix is diagonalized by Z^E:

$$Z^E m^2_{\tilde{e}} Z^{E,\dagger} = m^{dia}_{2,\tilde{e}} \tag{G.42}$$

with

$$\tilde{e}_{L,i} = \sum_{t_2} Z^{E,*}_{ji}\tilde{e}_j\,, \qquad \tilde{e}_{R,i} = \sum_{t_2} Z^{E,*}_{ji}\tilde{e}_j \tag{G.43}$$

j) Mass matrix for scalar Higgs, basis: (ϕ_d, ϕ_u)

$$m_h^2 = \begin{pmatrix} m^2_{H_d} + |\mu|^2 + \frac{1}{8}\Big(g_1^2 + g_2^2\Big)\Big(3v_d^2 - v_u^2\Big) & -\Re\Big(B_\mu\Big) - \frac{1}{4}\Big(g_1^2 + g_2^2\Big)v_dv_u \\ -\Re\Big(B_\mu\Big) - \frac{1}{4}\Big(g_1^2 + g_2^2\Big)v_dv_u & m^2_{H_u} + |\mu|^2 - \frac{1}{8}\Big(g_1^2 + g_2^2\Big)\Big(-3v_u^2 + v_d^2\Big) \end{pmatrix} \tag{G.44}$$

This matrix is diagonalized by Z^H:

$$Z^H m_h^2 Z^{H,\dagger} = m^{dia}_{2,h} \tag{G.45}$$

with

$$\phi_d = \sum_{t_2} Z^H_{j1} h_j \,, \qquad \phi_u = \sum_{t_2} Z^H_{j2} h_j \tag{G.46}$$

The mixing matrix can be parametrized by

$$Z^H = \left(\begin{array}{cc} - \sin\alpha & \cos\alpha \\ \cos\alpha & \sin\alpha \end{array}\right) \tag{G.47}$$

k) Mass matrix for pseudo scalar Higgs, basis: (σ_d, σ_u)

$$m^2_{A^0} = \left(\begin{array}{cc} m_{H_d}^2 + |\mu|^2 + \frac{1}{8}\Big(g_1^2 + g_2^2\Big)\Big(- v_u^2 + v_d^2\Big) & \Re\Big(B_\mu\Big) \\ \Re\Big(B_\mu\Big) & m_{H_u}^2 + |\mu|^2 - \frac{1}{8}\Big(g_1^2 + g_2^2\Big)\Big(- v_u^2 + v_d^2\Big) \end{array}\right) \tag{G.48}$$

This matrix is diagonalized by Z^A:

$$Z^A m^2_{A^0} Z^{A,\dagger} = m^{dia}_{2,A^0} \tag{G.49}$$

with

$$\sigma_d = \sum_{t_2} Z^A_{j1} A^0_j \,, \qquad \sigma_u = \sum_{t_2} Z^A_{j2} A^0_j \tag{G.50}$$

The mixing matrix can be parametrized by

$$Z^A = \left(\begin{array}{cc} - \cos\beta & \sin\beta \\ \sin\beta & \cos\beta \end{array}\right) \tag{G.51}$$

l) Mass matrix for charged Higgs, basis: $\Big(H_d^-, H_u^{+,*}\Big)$

$$m^2_{H^-} = \left(\begin{array}{cc} m_{H_d}^2 + |\mu|^2 + \frac{1}{8}\Big(g_1^2 + g_2^2\Big)\Big(v_d^2 - v_u^2\Big) & \frac{1}{4} g_2^2 v_d v_u + B_\mu \\ \frac{1}{4} g_2^2 v_d v_u + B_\mu^* & m_{H_u}^2 + |\mu|^2 + \frac{1}{8}\Big(g_2^2 - g_1^2\Big)\Big(v_d^2 - v_u^2\Big) \end{array}\right) \tag{G.52}$$

This matrix is diagonalized by Z^+:

$$Z^+ m^2_{H^-} Z^{+,\dagger} = m^{dia}_{2,H^-} \tag{G.53}$$

with

$$H_d^- = \sum_{t_2} Z^{+,*}_{j1} H_j^- \,, \qquad H_u^+ = \sum_{t_2} Z^+_{j2} H_j^+ \tag{G.54}$$

The mixing matrix can be parametrized by

$$Z^+ = \left(\begin{array}{cc} - \cos\beta & \sin\beta \\ \sin\beta & \cos\beta \end{array}\right) \tag{G.55}$$

G.1.8 Tadpole equations

$$\frac{\partial V}{\partial v_d} = \frac{1}{8}\Big(8v_d|\mu|^2 - 8v_u\Re\Big(B_\mu\Big) + v_d\Big(8m^2_{H_d} + g_1^2v_d^2 - g_1^2v_u^2 + g_2^2v_d^2 - g_2^2v_u^2\Big)\Big) \tag{G.56}$$

$$\frac{\partial V}{\partial v_u} = \frac{1}{8}\Big(- 8v_d\Re\Big(B_\mu\Big) + 8v_u|\mu|^2 + v_u\Big(8m^2_{H_u} - g_1^2v_d^2 + g_1^2v_u^2 - g_2^2v_d^2 + g_2^2v_u^2\Big)\Big) \tag{G.57}$$

G.2 Implementation in SARAH

In the following, we showcase the different components of the MSSM implementation in SARAH (see [74] for a summary of our conventions).

a) The gauge sector is $U(1) \times SU(2) \times SU(3)$ and is defined by declaring the corresponding vector superfields.

```
Gauge[[1]]={B,   U[1], hypercharge, g1, False};
Gauge[[2]]={WB, SU[2], left,        g2, True};
Gauge[[3]]={G,  SU[3], color,       g3, False};
```

First, the name of the vector superfield is given. The second entry defines the dimension of the group, the third one is the name of the gauge group and the forth one the name of the corresponding gauge coupling. If the last entry is set to True, the sum over the charge induces is processed, otherwise the charges are used as variable. In that case, the color charges are written as indices, while the sum over isospins is expanded.
Note, SARAH adds for every vector superfield automatically a soft-breaking gaugino mass.

b) The next step is to define the matter sector. That's done by the array Fields. The conventions are the following. First, the root of the names for the component fields is given (e.g. X): the derived names of the fermionic components start with F in front (i.e. FX), while for scalars a S is used (i.e. SX). At second position the number of generations is defined and the third entry is the name of the entire superfield. The remaining entries are the transformation properties with respect to the different gauge groups.
Using these conventions, the doublet superfields $\hat{q}, \hat{l}, \hat{H}_d, \hat{H}_u$ are added by

```
Fields[[1]] = {{uL,  dL},  3, q,   1/6, 2, 3};
Fields[[2]] = {{vL,  eL},  3, l,  -1/2, 2, 1};
Fields[[3]] = {{Hd0, Hdm}, 1, Hd, -1/2, 2, 1};
Fields[[4]] = {{Hup, Hu0}, 1, Hu,  1/2, 2, 1};
```

While for the singlet superfields $\hat{d}, \hat{u}, \hat{e}$

```
Fields[[5]] = {conj[dR], 3, d,  1/3, 1, -3};
Fields[[6]] = {conj[uR], 3, u, -2/3, 1, -3};
Fields[[7]] = {conj[eR], 3, e,     1, 1,  1};
```

is used.
Note, SARAH adds also for scalars automatically the soft masses.

c) The superpotential of the MSSM is

$$W = \hat{q}Y_u\hat{u}\hat{H}_u - \hat{q}Y_d\hat{d}\hat{H}_d - \hat{l}Y_e\hat{e}\hat{H}_d + \mu\hat{H}_u\hat{H}_d \tag{G.58}$$

and represented in SARAH by

```
SuperPotential = { {{1, Yu},{u,q,Hu}}, {{-1,Yd},{d,q,Hd}},
                   {{-1,Ye},{e,l,Hd}}, {{1,\[Mu]},{Hu,Hd}}  };
```

d) There are two different sets of eigenstates: the gauge eigenstates before EWSB and the mass eigenstates after EWSB. The internal names are

```
NameOfStates={GaugeES, EWSB};
```

e) The vector bosons and gauginos rotate after EWSB as follows

```
DEFINITION[EWSB][GaugeSector]=
{ {{VB,VWB[3]},{VP,VZ},ZZ},
  {{VWB[1],VWB[2]},{VWm,conj[VWm]},ZW},
  {{fWB[1],fWB[2],fWB[3]},{fWm,fWp,fW0},ZfW}
};
```

The rotation matrices $Z^{\gamma Z}$ (`ZZ`), Z^{W} (`ZW`) and $Z^{\tilde{W}}$ (`ZfW`) are defined in the parameter file of the corresponding model as

$$Z^{\gamma Z} = \begin{pmatrix} \cos\Theta_W & -\sin\Theta_W \\ \sin\Theta_W & \cos\Theta_W \end{pmatrix}, \qquad Z^{W} = Z^{\tilde{W}} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -i & i \end{pmatrix} \tag{G.59}$$

This encodes the common mixing of vector bosons and gauginos after EWSB

$$W_{1\rho} = \frac{1}{\sqrt{2}} W^-_\rho + \frac{1}{\sqrt{2}} W^+_\rho , \qquad W_{2\rho} = -i\frac{1}{\sqrt{2}} W^-_\rho + i\frac{1}{\sqrt{2}} W^+_\rho \tag{G.60}$$

$$W_{3\rho} = \cos\Theta_W Z_\rho + \sin\Theta_W \gamma_\rho , \qquad B_\rho = \cos\Theta_W \gamma_\rho - \sin\Theta_W Z_\rho \tag{G.61}$$

$$\lambda_{\tilde{W},1} = \frac{1}{\sqrt{2}} \tilde{W}^- + \frac{1}{\sqrt{2}} \tilde{W}^+ , \qquad \lambda_{\tilde{W},2} = -i\frac{1}{\sqrt{2}} \tilde{W}^- + i\frac{1}{\sqrt{2}} \tilde{W}^+ , \qquad \lambda_{\tilde{W},3} = \tilde{W}^0 \tag{G.62}$$

f) The neutral components of the scalar Higgs receive vacuum expectation values (VEVs) v_d/v_u and split into scalar and pseudo scalar components

$$H^0_d = \frac{1}{\sqrt{2}} (\phi_d + i\sigma_d + v_d) , \qquad H^0_u = \frac{1}{\sqrt{2}} (\phi_u + i\sigma_u + v_u) \tag{G.63}$$

This is encoded in SARAH by

```
DEFINITION[EWSB][VEVs]=
{{SHd0,{vd,1/Sqrt[2]},{sigmad,I/Sqrt[2]},{phid,1/Sqrt[2]}},
 {SHu0,{vu,1/Sqrt[2]},{sigmau,I/Sqrt[2]},{phiu,1/Sqrt[2]}}};
```

g) The particles mix after EWSB to new mass eigenstates

```
DEFINITION[EWSB][MatterSector]=
{{{SdL, SdR                 }, {Sd, ZD}},
 {{SuL, SuR                 }, {Su, ZU}},
 {{SeL, SeR                 }, {Se, ZE}},
 {{SvL                      }, {Sv, ZV}},
 {{phid, phiu               }, {hh, ZH}},
 {{sigmad, sigmau           }, {Ah, ZA}},
 {{SHdm, conj[SHup]         }, {Hpm,ZP}},
 {{fB, fW0, FHd0, FHu0}, {L0, ZN}},
 {{{fWm, FHdm}, {fWp, FHup}}, {{Lm,U},  {Lp,V}}},
 {{{FeL},         {conj[FeR]}}, {{FEL,ZEL},{FER,ZER}}},
 {{{FdL},         {conj[FdR]}}, {{FDL,ZDL},{FDR,ZDR}}},
 {{{FuL},         {conj[FuR]}}, {{FUL,ZUL},{FUR,ZUR}}}  };
```

This defines the mixings to the mass eigenstates: first, a list with gauge eigenstates is given, then the name of the new mass eigenstates and the mixing matrix follows. Hence, the first line is interpreted as

$$d_{L,i\alpha} = \sum_j U^{d,*}_{L,ji} D_{L,j\alpha}\,, \qquad d_{R,i\alpha} = \sum_j U^{d}_{R,ij} D^*_{R,j\alpha}\,, \tag{G.64}$$

while the 8th line defines the mixing in the chargino sector

$$\tilde{W}^- = \sum_j U^*_{j1}\lambda^-_j\,, \qquad \tilde{H}^-_d = \sum_j U^*_{j2}\lambda^-_j\,, \qquad \tilde{W}^+ = \sum_j V^*_{1j}\lambda^+_j\,, \qquad \tilde{H}^+_u = \sum_j V^*_{2j}\lambda^+_j \tag{G.65}$$

h) No particles should be integrated out or deleted

```
IntegrateOut={};
DeleteParticles={};
```

i) The Dirac spinors for the mass eigenstates are

```
DEFINITION[EWSB][DiracSpinors]={
 Fd  - > {FDL, conj[FDR]},
 Fe   -> {FEL, conj[FER]},
 Fu   -> {FUL, conj[FUR]},
 Fv   -> {FvL, 0},
 Chi -> {L0, conj[L0]},
 Cha -> {Lm, conj[Lp]},
 Glu -> {fG, conj[fG]}
};
```

That leads to the replacements

$$d \to \begin{pmatrix} d_L \\ d_R \end{pmatrix}, \ldots, \tilde{\chi}^- \to \begin{pmatrix} \lambda^- \\ (\lambda^+)^* \end{pmatrix}, \tilde{g} \to \begin{pmatrix} \lambda_g \\ \lambda^*_g \end{pmatrix} \tag{G.66}$$

when going from four- to two-component formalism.

Appendix H

Verification of Output

For details about our checks to verify and validate the output of SARAH we refer to the following references:

- Renormalization group equations and self-energies see Ref. [74, 75]
- FeynArts/FormCalc as well as CalcHep/CompHep see Ref. [76]
- WHIZARD and SPheno, output see Ref. [77, 75]
- UFO output, see Ref. [78]

Appendix I

Evaluation Time

To give an impression for the needed evaluation time for different routines and models, we collected some values in Table I.1. This times were measured under Mathematica 5.2 running on an a Lenovo Thinkpad X220 with a Intel i7-2620M (2.70GHz) core.

Command	MSSM	NMSSM	B-L-SSM	Seesaw 3
`Start`	8.9	10.3	34.0	18.5
`ModelOutput[EWSB]`	49.7	62.6	445.3	83.3
`CalcRGEs[]`	11.2	16.9	82.5	60.3
`CalcLoopCorrections[EWSB]`	5.9	6.9	55.0	9.8
`MakeFeynArts[]`	0.5	0.6	5.5	1.7
`MakeCHep[]`	10.8	12.7	98.7	19.3
`MakeUFO[]`	38.5	62.8	887.5	80.6
`MakeWHIZARD[]`	274.8	538.9	4326.5	486.8
`MakeSPheno[]`	134.1	196.5	1024.7	292.3
`MakeTeX[]`	3.9	6.3	61.6	7.1

Table I.1: Time needed in seconds to evaluate several commands of `SARAH` in Mathematica 5.2

Appendix J

Models included in the public version

- Minimal supersymmetric standard model (see Ref. [79] and references therein):
  - With general flavor and CP structure (`MSSM`)
  - Without flavor violation (`MSSM/NoFV`)
  - With explicit CP violation in the Higgs sector (`MSSM/CPV`)
  - In SCKM basis (`MSSM/CKM`)
- Singlet extensions:
  - Next-to-minimal supersymmetric standard model (`NMSSM`, `NMSSM/NoFV`, `NMSSM/CPV`, `NMSSM/CKM`) (see Refs. [80, 81] and references therein)
  - near-to-minimal supersymmetric standard model (`near-MSSM`) [82]
  - General singlet extended, supersymmetric standard model (`SMSSM`) [82, 83]
- Triplet extensions
  - Triplet extended MSSM (`TMSSM`) [84]
  - Triplet extended NMSSM (`TNMSSM`) [85]
- Models with R-parity violation [86, 87, 88, 89, 90, 91, 92, 93]
  - bilinear RpV (`MSSM-RpV/Bi`)
  - Lepton number violation (`MSSM-RpV/LnV`)
  - Only trilinear lepton number violation (`MSSM-RpV/TriLnV`)
  - Baryon number violation (`MSSM-RpV/BnV`)
  - $\mu\nu$SSM (`munuSSM`) [94, 95]
- Additional $U(1)'s$
  - $U(1)$-extended MSSM (`UMSSM`) [82]
  - secluded MSSM (`secluded-MSSM`) [96]
  - minimal $B-L$ model (`B-L-SSM`) [97, 98, 99, 100]
  - minimal singlet-extended $B-L$ model (`N-B-L-SSM`)

- SUSY-scale seesaw extensions
  - inverse seesaw (`inverse-Seesaw`) [101, 102, 103]
  - linear seesaw (`LinSeesaw`) [101, 104]
  - singlet-extended inverse seesaw (`inverse-Seesaw-NMSSM`)
  - inverse seesaw with $B-L$ gauge group (`B-L-SSM-IS`)
  - minimal $U(1)_R \times U(1)_{B-L}$ model with inverse seesaw (`BLRinvSeesaw`) [105, 42]
- Models with Dirac Gauginos
  - MSSM/NMSSM with Dirac Gauginos (`DiracGauginos`) [106, 107, 108]
  - minimal R-Symmstric SSM (`MRSSM`) [109]
- High-scale extensions
  - Seesaw 1 - 3 ($SU(5)$ version) , (`Seesaw1`,`Seesaw2`,`Seesaw3`) [110, 111, 112, 113, 72]
  - Left/right model (ΩLR) (`Omega`) [73, 114]
- Non-SUSY models:
  - SM (`SM`, `SM/CKM`) (see for instance Ref. [115] and references therein)
  - inert doublet model (`Inert`) [116]

Bibliography

[1] P. Ramond. Dual theory for free fermions. *Physical Review D*, 3(10):2415–2418, May 1971.

[2] J. Wess and B. Zumino. Supergauge transformations in four-dimensions. *Nuclear Physics*, B70:39–50, 1974.

[3] D. V. Volkov and V. P. Akulov. Is the neutrino a goldstone particle? *Physical Letters*, B46:109–110, 1973.

[4] M. H. Friedman and Y. Srivastava. Allowed supersymmetric vacuums. *Physical Review D*, 16(2):304–306, Jul 1977.

[5] S. Ferrara, J. Wess, and B. Zumino. Supergauge multiplets and superfields. *Physical Letters*, B51:239, 1974.

[6] L. Girardello and Marcus T. Grisaru. Soft breaking of supersymmetry. *Nuclear Physics*, B194:65, 1982.

[7] Steven Weinberg. Implications of dynamical symmetry breaking. *Physical Review D*, 13(4):974–996, Feb 1976.

[8] S. Weinberg. Implications of dynamical symmetry breaking: An addendum. *Physical Review D*, 19(4):1277–1280, Feb 1979.

[9] Paul Langacker and Mingxing Luo. Implications of precision electroweak experiments for m_t, ρ_0, $sin^2\theta_W$, and grand unification. *Physical Review D*, 44(3):817–822, Aug 1991.

[10] John R. Ellis, S. Kelley, and Dimitri V. Nanopoulos. Probing the desert using gauge coupling unification. *Physical Letters*, B260:131–137, 1991.

[11] C. Giunti, C. W. Kim, and U. W. Lee. Running coupling constants and grand unification models. *Mod. Physical Letters*, A6:1745–1755, 1991.

[12] L. Hall. Grand unification of effective gauge theories. *Nuclear Physics B*, 178:75–124, February 1981.

[13] John R. Ellis, J. S. Hagelin, Dimitri V. Nanopoulos, Keith A. Olive, and M. Srednicki. Supersymmetric relics from the big bang. *Nucl. Phys.*, B238:453–476, 1984.

[14] T. Hahn. Generating feynman diagrams and amplitudes with feynarts 3. *Computer Physics Communications*, 140:418, 2001, hep-ph/0012260.

[15] Wolfgang Kilian, Thorsten Ohl, and Jurgen Reuter. WHIZARD: Simulating Multi-Particle Processes at LHC and ILC. 2007, 0708.4233.

[16] A. Pukhov. CalcHEP 3.2: MSSM, structure functions, event generation, batchs, and generation of matrix elements for other packages. 2004, hep-ph/0412191.

[17] A. S. Belyaev et al. CompHEP-PYTHIA interface: Integrated package for the collision events generation based on exact matrix elements. 2000, hep-ph/0101232.

[18] G. Belanger, F. Boudjema, A. Pukhov, and A. Semenov. Dark matter direct detection rate in a generic model with micrOMEGAs2.1. *Comput. Phys. Commun.*, 180:747–767, 2009, 0803.2360.

[19] Celine Degrande et al. UFO - The Universal FeynRules Output. 2011, 1108.2040.

[20] Werner Porod. Spheno, a program for calculating supersymmetric spectra, susy particle decays and susy particle production at e+ e- colliders. *Computer Physics Communications*, 153:275, 2003, hep-ph/0301101.

[21] B. Allanach et al. SUSY Les Houches Accord 2. *Comp. Phys. Commun.*, 180:8–25, 2009, 0801.0045.

[22] Stephen P. Martin and Michael T. Vaughn. Two loop renormalization group equations for soft supersymmetry breaking couplings. *Phys. Rev.*, D50:2282, 1994, hep-ph/9311340.

[23] Renato Fonseca, Michal Malinsky, Werner Porod, and Florian Staub. Running soft parameters in SUSY models with multiple U(1) gauge factors. 2011, 1107.2670. * Temporary entry *.

[24] Mark D. Goodsell. Two-loop RGEs with Dirac gaugino masses. 2012, 1206.6697.

[25] Marcus Sperling, Dominik StÃ¶ckinger, and Alexander Voigt. Renormalization of vacuum expectation values in spontaneously broken gauge theories. *JHEP*, 1307:132, 2013, 1305.1548.

[26] Marie E. Machacek and Michael T. Vaughn. Two Loop Renormalization Group Equations in a General Quantum Field Theory. 1. Wave Function Renormalization. *Nucl. Phys.*, B222:83, 1983.

[27] Marie E. Machacek and Michael T. Vaughn. Two Loop Renormalization Group Equations in a General Quantum Field Theory. 2. Yukawa Couplings. *Nucl. Phys.*, B236:221, 1984.

[28] Marie E. Machacek and Michael T. Vaughn. Two Loop Renormalization Group Equations in a General Quantum Field Theory. 3. Scalar Quartic Couplings. *Nucl. Phys.*, B249:70, 1985.

[29] Ming-xing Luo, Hua-wen Wang, and Yong Xiao. Two-loop renormalization group equations in general gauge field theories. *Phys. Rev.*, D67:065019, 2003, hep-ph/0211440.

[30] Renato M. Fonseca, Michal Malinsky, and Florian Staub. Renormalization group equations and matching in a general quantum field theory with kinetic mixing. 2013, 1308.1674.

[31] Damien M. Pierce, Jonathan A. Bagger, Konstantin T. Matchev, and Ren-jie Zhang. Precision corrections in the minimal supersymmetric standard model. *Nucl. Phys.*, B491:3–67, 1997, hep-ph/9606211.

[32] G. Brooijmans, B. Gripaios, F. Moortgat, Jose Santiago, P. Skands, et al. Les Houches 2011: Physics at TeV Colliders New Physics Working Group Report. 2012, 1203.1488.

[33] L. Basso, A. Belyaev, D. Chowdhury, M. Hirsch, S. Khalil, et al. Supersymmetry Les Houches Accord: Seesaw models and generalised PDG numbering scheme. 2012, 1206.4563.

[34] T.L. Lee, T.Y. Li, and C.H. Tsai. Hom4ps-2.0: a software package for solving polynomial systems by the polyhedral homotopy continuation method. *Computing*, 83(2):109–133, 2008.

[35] T. Hahn and M. Perez-Victoria. Automatized one-loop calculations in 4 and d dimensions. *Computer Physics Communications*, 118:153, 1999, hep-ph/9807565.

[36] G. Belanger, Neil D. Christensen, A. Pukhov, and A. Semenov. SLHAplus: a library for implementing extensions of the standard model. 2010, 1008.0181.

[37] G. Belanger, F. Boudjema, A. Pukhov, and A. Semenov. micrOMEGAs2.0: A program to calculate the relic density of dark matter in a generic model. *Comput. Phys. Commun.*, 176:367–382, 2007, hep-ph/0607059.

[38] W. Porod and F. Staub. SPheno 3.1: Extensions including flavour, CP-phases and models beyond the MSSM. 2011, 1104.1573.

[39] G. F. Giudice and R. Rattazzi. Theories with gauge-mediated supersymmetry breaking. *Phys. Rept.*, 322:419–499, 1999, hep-ph/9801271.

[40] Lawrence J. Hall. Grand Unification of Effective Gauge Theories. *Nucl. Phys.*, B178:75, 1981.

[41] Zhi-zhong Xing, He Zhang, and Shun Zhou. Updated Values of Running Quark and Lepton Masses. *Phys.Rev.*, D77:113016, 2008, 0712.1419.

[42] Martin Hirsch, Werner Porod, Laslo Reichert, and Florian Staub. Phenomenology of the minimal supersymmetric $U(1)_{B-L} \times U(1)_R$ extension of the standard model. 2012, 1206.3516.

[43] Philip Bechtle, Oliver Brein, Sven Heinemeyer, Georg Weiglein, and Karina E. Williams. HiggsBounds: Confronting Arbitrary Higgs Sectors with Exclusion Bounds from LEP and the Tevatron. *Comput.Phys.Commun.*, 181:138–167, 2010, 0811.4169.

[44] Philip Bechtle, Oliver Brein, Sven Heinemeyer, Georg Weiglein, and Karina E. Williams. HiggsBounds 2.0.0: Confronting Neutral and Charged Higgs Sector Predictions with Exclusion Bounds from LEP and the Tevatron. 2011, 1102.1898.

[45] J.E. Camargo-Molina, B. O'Leary, W. Porod, and F. Staub. Vevacious: A Tool For Finding The Global Minima Of One-Loop Effective Potentials With Many Scalars. 2013, 1307.1477.

[46] Carroll L. Wainwright. CosmoTransitions: Computing Cosmological Phase Transition Temperatures and Bubble Profiles with Multiple Fields. *Comput.Phys.Commun.*, 183:2006–2013, 2012, 1109.4189.

[47] Ben O'Leary. LHPC Spectrum Plotter. 2012, github.com/benoleary/LesHouchesParserClasses_CPP.

[48] Karim Benakli. Dirac Gauginos: A User Manual. *Fortsch.Phys.*, 59:1079–1082, 2011, 1106.1649.

[49] Edward Witten. An SU(2) anomaly. *Phys. Lett.*, B117:324–328, 1982.

[50] Youichi Yamada. Two loop renormalization of gaugino masses in general supersymmetric gauge models. *Phys. Rev. Lett.*, 72:25–27, 1994, hep-ph/9308304.

[51] I. Jack, D. R. T. Jones, and A. Pickering. The soft scalar mass beta-function. *Phys. Lett.*, B432:114–119, 1998, hep-ph/9803405.

[52] Ian Jack, D. R. Timothy Jones, Stephen P. Martin, Michael T. Vaughn, and Youichi Yamada. Decoupling of the epsilon scalar mass in softly broken supersymmetry. *Phys. Rev.*, D50:5481–5483, 1994, hep-ph/9407291.

[53] D. R. T. Jones and L. Mezincescu. The Chiral Anomaly and a Class of Two Loop Finite Supersymmetric Gauge Theories. *Phys. Lett.*, B138:293, 1984.

[54] Peter C. West. The Yukawa beta Function in N=1 Rigid Supersymmetric Theories. *Phys. Lett.*, B137:371, 1984.

[55] I. Jack, D. R. T. Jones, and A. Pickering. Renormalisation invariance and the soft beta functions. *Phys. Lett.*, B426:73–77, 1998, hep-ph/9712542.

[56] Youichi Yamada. Two loop renormalization of gaugino mass in supersymmetric gauge model. *Phys. Lett.*, B316:109–111, 1993, hep-ph/9307217.

[57] Youichi Yamada. Two loop renormalization group equations for soft SUSY breaking scalar interactions: Supergraph method. *Phys. Rev.*, D50:3537–3545, 1994, hep-ph/9401241.

[58] I. Jack and D.R.T. Jones. The Fayet-Iliopoulos D term and its renormalization in the MSSM. *Phys.Rev.*, D63:075010, 2001, hep-ph/0010301.

[59] Christoph Bobeth, Andrzej J. Buras, Frank Kruger, and Jorg Urban. QCD corrections to $\bar{B} \to X_{d,s}\nu\bar{\nu}$, $\bar{B}_{d,s} \to \ell^+\ell^-$, $K \to \pi\nu\bar{\nu}$ and $K_L \to \mu^+\mu^-$ in the MSSM. *Nucl.Phys.*, B630:87–131, 2002, hep-ph/0112305.

[60] Seungwon Baek, Toru Goto, Yasuhiro Okada, and Ken-ichi Okumura. Muon anomalous magnetic moment, lepton flavor violation, and flavor changing neutral current processes in SUSY GUT with right-handed neutrino. *Phys.Rev.*, D64:095001, 2001, hep-ph/0104146.

[61] Tobias Huber, Enrico Lunghi, Mikolaj Misiak, and Daniel Wyler. Electromagnetic logarithms in anti-B -> X(s) l+ l-. *Nucl.Phys.*, B740:105–137, 2006, hep-ph/0512066.

[62] Ernesto Arganda and Maria J. Herrero. Testing supersymmetry with lepton flavor violating tau and mu decays. *Phys.Rev.*, D73:055003, 2006, hep-ph/0510405.

[63] Tarek Ibrahim and Pran Nath. CP violation and the muon anomaly in N=1 supergravity. *Phys.Rev.*, D61:095008, 2000, hep-ph/9907555.

[64] A. Bartl, T. Gajdosik, W. Porod, P. Stockinger, and H. Stremnitzer. Electron and neutron electric dipole moments in the constrained MSSM. *Phys.Rev.*, D60:073003, 1999, hep-ph/9903402.

[65] A. Bartl, W. Majerotto, W. Porod, and D. Wyler. Effect of supersymmetric phases on lepton dipole moments and rare lepton decays. *Phys.Rev.*, D68:053005, 2003, hep-ph/0306050.

[66] Manuel Drees and Kaoru Hagiwara. SUPERSYMMETRIC CONTRIBUTION TO THE ELECTROWEAK ρ PARAMETER. *Phys.Rev.*, D42:1709–1725, 1990.

[67] E. Arganda, M.J. Herrero, and A.M. Teixeira. mu-e conversion in nuclei within the CMSSM seesaw: Universality versus non-universality. *JHEP*, 0710:104, 2007, 0707.2955.

[68] E. Arganda, M.J. Herrero, and J. Portoles. Lepton flavour violating semileptonic tau decays in constrained MSSM-seesaw scenarios. *JHEP*, 0806:079, 2008, 0803.2039.

[69] H.K. Dreiner, K. Nickel, F. Staub, and A. Vicente. New bounds on trilinear R-parity violation from lepton flavor violating observables. *Phys.Rev.*, D86:015003, 2012, 1204.5925.

[70] M. Hirsch, F. Staub, and A. Vicente. Enhancing $l_i \to 3l_j$ with the Z^0-penguin. *Phys.Rev.*, D85:113013, 2012, 1202.1825.

[71] M. Spira, A. Djouadi, D. Graudenz, and P.M. Zerwas. Higgs boson production at the LHC. *Nucl.Phys.*, B453:17–82, 1995, hep-ph/9504378.

[72] J. N. Esteves, M. Hirsch, W. Porod, J. C. Romao, and F. Staub. Supersymmetric type-III seesaw: lepton flavour violating decays and dark matter. 2010, 1010.6000.

[73] J. N. Esteves et al. LHC and lepton flavour violation phenomenology of a left- right extension of the MSSM. 2010, 1011.0348.

[74] Florian Staub. Automatic Calculation of supersymmetric Renormalization Group Equations and Self Energies. 2010, 1002.0840.

[75] Florian Staub, Werner Porod, and Bjorn Herrmann. The Electroweak sector of the NMSSM at the one-loop level. *JHEP*, 1010:040, 2010, 1007.4049.

[76] Florian Staub. From Superpotential to Model Files for FeynArts and CalcHep/CompHep. *Comput.Phys.Commun.*, 181:1077–1086, 2010, 0909.2863.

[77] Florian Staub, Thorsten Ohl, Werner Porod, and Christian Speckner. A Tool Box for Implementing Supersymmetric Models. *Comput.Phys.Commun.*, 183:2165–2206, 2012, 1109.5147.

[78] Florian Staub. Linking SARAH and MadGraph using the UFO format. 2012, 1207.0906.

[79] Stephen P. Martin. A Supersymmetry primer. 1997, hep-ph/9709356.

[80] M. Maniatis. The Next-to-Minimal Supersymmetric extension of the Standard Model reviewed. *Int.J.Mod.Phys.*, A25:3505–3602, 2010, 0906.0777.

[81] Ulrich Ellwanger, Cyril Hugonie, and Ana M. Teixeira. The Next-to-Minimal Supersymmetric Standard Model. *Phys.Rept.*, 496:1–77, 2010, 0910.1785.

[82] Vernon Barger, Paul Langacker, Hye-Sung Lee, and Gabe Shaughnessy. Higgs Sector in Extensions of the MSSM. *Phys.Rev.*, D73:115010, 2006, hep-ph/0603247.

[83] Graham G. Ross, Kai Schmidt-Hoberg, and Florian Staub. The Generalised NMSSM at One Loop: Fine Tuning and Phenomenology. *JHEP*, 1208:074, 2012, 1205.1509.

[84] Stefano Di Chiara and Ken Hsieh. Triplet Extended Supersymmetric Standard Model. *Phys.Rev.*, D78:055016, 2008, 0805.2623.

[85] Kaustubh Agashe, Aleksandr Azatov, Andrey Katz, and Doojin Kim. Improving the tunings of the MSSM by adding triplets and singlet. *Phys.Rev.*, D84:115024, 2011, 1109.2842.

[86] Lawrence J. Hall and Mahiko Suzuki. Explicit R-Parity Breaking in Supersymmetric Models. *Nucl.Phys.*, B231:419, 1984.

[87] Herbert K. Dreiner. An Introduction to explicit R-parity violation. 1997, hep-ph/9707435.

[88] B.C. Allanach, A. Dedes, and H.K. Dreiner. R parity violating minimal supergravity model. *Phys.Rev.*, D69:115002, 2004, hep-ph/0309196.

[89] Gautam Bhattacharyya. A Brief review of R-parity violating couplings. 1997, hep-ph/9709395.

[90] Vernon D. Barger, G.F. Giudice, and Tao Han. Some New Aspects of Supersymmetry R-Parity Violating Interactions. *Phys.Rev.*, D40:2987, 1989.

[91] B.C. Allanach, A. Dedes, and Herbert K. Dreiner. Bounds on R-parity violating couplings at the weak scale and at the GUT scale. *Phys.Rev.*, D60:075014, 1999, hep-ph/9906209.

[92] M. Hirsch, M.A. Diaz, W. Porod, J.C. Romao, and J.W.F. Valle. Neutrino masses and mixings from supersymmetry with bilinear R parity violation: A Theory for solar and atmospheric neutrino oscillations. *Phys.Rev.*, D62:113008, 2000, hep-ph/0004115.

[93] R. Barbier, C. Berat, M. Besancon, M. Chemtob, A. Deandrea, et al. R-parity violating supersymmetry. *Phys.Rept.*, 420:1–202, 2005, hep-ph/0406039.

[94] D.E. Lopez-Fogliani and C. Munoz. Proposal for a new minimal supersymmetric standard model. *Phys.Rev.Lett.*, 97:041801, 2006, hep-ph/0508297.

[95] A. Bartl, M. Hirsch, A. Vicente, S. Liebler, and W. Porod. LHC phenomenology of the mu nu SSM. *JHEP*, 0905:120, 2009, 0903.3596.

[96] Cheng-Wei Chiang and Eibun Senaha. Electroweak phase transitions in the secluded U(1)-prime-extended MSSM. *JHEP*, 1006:030, 2010, 0912.5069.

[97] S. Khalil and A. Masiero. Radiative B-L symmetry breaking in supersymmetric models. *Phys.Lett.*, B665:374–377, 2008, 0710.3525.

[98] Pavel Fileviez Perez and Sogee Spinner. The Fate of R-Parity. *Phys.Rev.*, D83:035004, 2011, 1005.4930.

[99] Ben O'Leary, Werner Porod, and Florian Staub. Mass spectrum of the minimal SUSY B-L model. *JHEP*, 1205:042, 2012, 1112.4600.

[100] Lorenzo Basso, Ben O'Leary, Werner Porod, and Florian Staub. Dark matter scenarios in the minimal SUSY B-L model. *JHEP*, 1209:054, 2012, 1207.0507.

[101] Michal Malinsky, J.C. Romao, and J.W.F. Valle. Novel supersymmetric SO(10) seesaw mechanism. *Phys.Rev.Lett.*, 95:161801, 2005, hep-ph/0506296.

[102] Asmaa Abada, Debottam Das, Avelino Vicente, and Cedric Weiland. Enhancing lepton flavour violation in the supersymmetric inverse seesaw beyond the dipole contribution. *JHEP*, 1209:015, 2012, 1206.6497.

[103] P.S. Bhupal Dev, Subhadeep Mondal, Biswarup Mukhopadhyaya, and Sourov Roy. Phenomenology of Light Sneutrino Dark Matter in cMSSM/mSUGRA with Inverse Seesaw. 2012, 1207.6542.

[104] Valentina De Romeri and Martin Hirsch. Sneutrino Dark Matter in Low-scale Seesaw Scenarios. 2012, 1209.3891.

[105] M. Hirsch, M. Malinsky, W. Porod, L. Reichert, and F. Staub. Hefty MSSM-like light Higgs in extended gauge models. *JHEP*, 1202:084, 2012, 1110.3037.

[106] G. Belanger, K. Benakli, M. Goodsell, C. Moura, and A. Pukhov. Dark Matter with Dirac and Majorana Gaugino Masses. *JCAP*, 0908:027, 2009, 0905.1043.

[107] K. Benakli and M.D. Goodsell. Dirac Gauginos, Gauge Mediation and Unification. *Nucl.Phys.*, B840:1–28, 2010, 1003.4957.

[108] Karim Benakli, Mark D. Goodsell, and Florian Staub. Dirac Gauginos and the 125 GeV Higgs. 2012, 1211.0552.

[109] Graham D. Kribs, Erich Poppitz, and Neal Weiner. Flavor in supersymmetry with an extended R-symmetry. *Phys.Rev.*, D78:055010, 2008, 0712.2039.

[110] Francesca Borzumati and Toshifumi Yamashita. Minimal supersymmetric SU(5) model with non-renormalizable operators: Seesaw mechanism and violation of flavour and CP. *Prog.Theor.Phys.*, 124:761–868, 2010, 0903.2793.

[111] Anna Rossi. Supersymmetric seesaw without singlet neutrinos: Neutrino masses and lepton flavor violation. *Phys.Rev.*, D66:075003, 2002, hep-ph/0207006.

[112] M. Hirsch, J.W.F. Valle, W. Porod, J.C. Romao, and Albert Villanova del Moral. Probing minimal supergravity in type I seesaw with lepton flavour violation at the LHC. *Phys.Rev.*, D78:013006, 2008, 0804.4072.

[113] J.N. Esteves, J.C. Romao, Albert Villanova del Moral, M. Hirsch, J.W.F. Valle, et al. Flavour violation at the LHC: type-I versus type-II seesaw in minimal supergravity. *JHEP*, 0905:003, 2009, 0903.1408.

[114] J.N. Esteves, J.C. Romao, M. Hirsch, W. Porod, F. Staub, et al. Dark matter and LHC phenomenology in a left-right supersymmetric model. *JHEP*, 1201:095, 2012, 1109.6478.

[115] W. Hollik. Quantum field theory and the Standard Model. 2010, 1012.3883.

[116] Laura Lopez Honorez, Emmanuel Nezri, Josep F. Oliver, and Michel H.G. Tytgat. The Inert Doublet Model: An Archetype for Dark Matter. *JCAP*, 0702:028, 2007, hep-ph/0612275.