HidSecSOFTSUSY: Incorporating effects from hidden sectors in the numerical computation of the MSSM spectrum

Boaz Keren-Zur

Abstract

SOFTSUSY is a software designed to solve the RG equations of the MSSM and compute its low energy spectrum. HidSecSOFTSUSY is an extension of the SOFTSUSY package which modifies the beta functions to include contributions from light dynamic fields in the hidden sector.

Keywords: MSSM, hidden sector

1. Program Summary

Program title: HidSecSOFTSUSY
Program obtainable from: http://www.tau.ac.il/~kerenzu
Distribution format: tar.gz
Programming language: C++, fortran
Computer: Personal computer
Operating system: Tested on GNU/Linux
Word size: 32 bits
External routines: requires an installed version of SOFTSUSY.
Typical running time: a few seconds per parameter point.
Nature of problem: Calculating supersymmetric particle spectrum and mixing parameters while incorporating dynamic modes from the hidden sector into the renormalization group equations. The solution to the equations must be consistent with a high-scale boundary condition on supersymmetry breaking parameters, as well as a weak-scale boundary condition on gauge couplings, Yukawa couplings and the Higgs potential parameters.
Solution method: Nested iterative algorithm.

2. Introduction

Supersymmetry is an appealing framework for physics beyond the standard model as it solves the hierarchy problem, leads to gauge coupling unification and provides a basis for a UV completion of the standard model. Supersymmetry is not observed in nature (at least not below the 100 GeV scale) and is assumed to be broken by the dynamics of an additional hidden sector. In the minimal supersymmetric standard model (MSSM) the hidden sector is integrated out at a scale M and soft SUSY breaking ($\not{\text{SUSY}}$) terms are introduced into the effective Lagrangian. The value of these effective couplings in the IR can be obtained by evolving the renormalization group equations down from the scale M. The package `SOFTSUSY`[1] solves this set of coupled differential equations by an iterative process for which the boundaries are the Z boson mass, where the SM parameters are known, and the scale M, where the soft $\not{\text{SUSY}}$ terms are provided by the user of the software.

This, however, is not necessarily the entire picture. The hidden sector might contain degrees of freedom which are lighter than the scale M and have non-trivial contributions to the beta functions of the MSSM parameters. For example, in models of direct gauge mediation the hidden sector contains pseudomoduli which have masses of the order of the soft $\not{\text{SUSY}}$ terms, and might be charged under the standard model (e.g. [2]). The contributions of these fields to the gauge coupling beta function is a leading order effect. Another example is referred to as extra-ordinary gauge mediation (EOGM)[3], where the theory has multiple messenger scales. In such theories the various messengers are integrated out at different scales and at the intermediate scales the effective theory contains both soft $\not{\text{SUSY}}$ terms (generated by the integration out of heavy messengers) and dynamical messengers which contribute at leading order to the RG flow of the gauge couplings and MSSM scalar masses.

In order to incorporate the contributions from the hidden sector into `SOFTSUSY` an extension of the package named `HidSecSOFTSUSY` was written. The input is a list of the relevant superfields in the hidden sector including their masses and representations under the MSSM gauge group. The running of the RG flow is split into several stages defined by the masses of the dynamical hidden sector particles. The beta functions are modified according to the interactions of the dynamical particles at each stage, and between the stages the software takes into account threshold effects. The extended package includes the following features:

- The user provides information regarding the hidden sector fields. Their contribution to the beta function of the MSSM gauge coupling is computed automatically.
- EOGM models can be implemented by marking hidden sector fields as messengers. Such fields have additional contributions to the beta functions and important threshold effects which are also computed automatically. The magnitude of these effects depends on the pattern of SUSY breaking and there are several parameters that should be provided by the user.

  Since doublet and triplet messengers are introduced separately, models of doublet-triplet splitting are naturally accommodated in the software.
- Advanced users may define parameters in the hidden sector and use the `SOFTSUSY` engine to compute their RG flow. This requires some basic knowledge of C++.
- More advanced threshold effect can also be introduced using user defined functions.

3. The MSSM RG flow and contributions from hidden fields

The contribution of a (non-messenger) charged chiral multiplet Ψ to the beta function of the gauge coupling g_a is given by

$$\Delta\beta_a = N_f \frac{g_a^3}{16\pi^2} I_a(\Psi) \,, \tag{1}$$

where N_f is the number of flavors (or the degeneracy of the field Ψ), and $I_a(\Psi)$ is the Dynkin index of the chiral multiplet Ψ. The beta function for the scalar mass $m^2_{\phi_i}$ is modified by

$$\Delta\beta_{m^2_{\phi_i}} = \frac{6}{5} Y_i g_1^2 \mathrm{Tr}[\mathrm{Y m}^2] \,, \tag{2}$$

where Y_i is the hypercharge, and the trace is over all the charged scalars in the hidden sector. There are no leading order threshold effects which are relevant to the MSSM parameters as the Ψ multiplet is integrated out.

If the charged chiral multiplet can be considered as a messenger, namely the splitting of the mass squared between the fermions and bosons is of the order of the SUSY breaking scale, $\sqrt{F}$, then additional contributions become important. First, the following 2-loop contribution to the beta function of the scalar masses becomes relevant

$$\Delta\beta_{m_i^2} = \sum_a \frac{g_a^4}{(16\pi^2)^2} C_a(i) \mathrm{Str}[\mathrm{I_a}(\Psi)\mathcal{M}^2] \,, \tag{3}$$

where $C_a(i)$ is the casimir of the MSSM scalar field i, $\mathcal{M}^2$ is the messenger mass matrix, and Str[...] is a supertrace. In the case of a messenger, the supertrace of the mass matrix is large enough to make this 2-loop effect important[3].

When a messenger Ψ is integrated out, there are important threshold effects. The MSSM gauginos and scalar masses receive the following contributions

$$\begin{aligned} \Delta M_a &= N_f \frac{g_a^2}{16\pi^2} I_a(\Psi) \Lambda_G[\Psi] \\ \Delta m_i^2 &= 2 \sum_a N_f \left(\frac{g_a^2}{16\pi^2}\right)^2 C_a(i) I_a(\Psi) \Lambda_S^2[\Psi] \end{aligned} \tag{4}$$

$\Lambda_G[\Psi]$ and $\Lambda_S^2[\Psi]$ are model dependent. In the limit $\frac{F}{M^2} << 1$ they are given by

$$\begin{aligned} \Lambda_G[\Psi] &= F_X \frac{\partial_X \mathcal{M}_\Psi}{\mathcal{M}_\Psi} \\ \Lambda_S^2[\Psi] &= |F_X|^2 \left|\frac{\partial_X \mathcal{M}_\Psi}{\mathcal{M}_\Psi}\right|^2 \,, \end{aligned} \tag{5}$$

where X is the SUSY breaking spurion, F_X is its F component, and M_Ψ is the fermion mass of the messenger Ψ.

Three comments are in order. The current version of `HidSecSOFTSUSY` does not support running hidden field masses. The masses are fixed, for the technical reason that they determine the different threshold scales. Second, note that although there might be large splittings between the boson and fermion masses, the whole messenger multiplet is integrated out at the fermion mass scale. The corrections due to this mass splitting were not implemented in this version of `HidSecSOFTSUSY`. The last comment is that users of `HidSecSOFTSUSY` who do not wish to discuss multiple messenger scales and doublet-triplet splittings are not required to specify all the details regarding the messenger sector, and are only required to provide the regular `SOFTSUSY` boundary conditions.

4. Calculation algorithm

The RG flow is determined by a complicated set of coupled differential equations. Part of the boundary conditions (the measured standard model parameters) are given at the low scale boundary, and the others (user defined soft ~~SUSY~~ terms) are given at the high energy boundary. In addition to that, the system is constrained by the conditions of EW symmetry breaking at an intermediate scale (with a user defined $\tan\beta$). In order to match between these constraints `SOFTSUSY` implements an iterative procedure in which the set of differential equations is evolved back and forth between the different scales. The details regarding the algorithm can be found in [1]. All the data and computation methods are held in a class named `MSSMSoftsusy`.

In this extension package a new class `HidSecSoftsusy` is defined. It inherits all the members and methods of `MSSMSoftsusy`, and introduces the contributions of the hidden sector through the following intervention points:

- The iterative running between the different boundaries is divided into steps which are determined by the masses of the hidden fields.
- At each step hidden fields are integrated out (or in), and threshold effects are taken into account. The only threshold effects which are already implemented in `HidSecSOFTSUSY` are due to integration out of messengers (eq. (4)). Additional threshold effects can be supplied by the user via user defined functions.

- The contribution to the beta functions (eqs. (1,2,3)) from dynamical hidden fields (fields whose mass are smaller than the current energy scale) are taken into account automatically. Additional contributions to beta functions can be supplied by the user via user defined functions.
- In the RGE engine of SOFTSUSY all the MSSM parameters are held in one array which is modified step by step along the RG flow according to the beta functions. If the hidden sector contains additional parameters which are scale dependent it is easy to include the new parameters in the RG evolution. This is done in the software by increasing the length of the array and holding the new parameters in the last cells. The beta functions for these parameters should be supplied by the user via user defined functions.

5. User defined hidden sector

HidSecSOFTSUSY allows two levels of intervention by the user – defining the hidden sector fields and parameters in the input file, and by writing user defined C++ functions which modify the beta functions and threshold effects. The former does not require any programming, while the latter requires some basic understanding of the data structure in the software and several lines of additional code.

5.1. Input file format

The input for HidSecSOFTSUSY can be provided in a file written in what is known as the Les Houches Accord format [4, 5]. In order to add a hidden field, a block in the following format should be added

```
Block HIDFIELD               # SU2 doublet messenger
    1   1.000000000e+10      # Mass
    2   1.000000000e+08      # Super Trace of the mass squared (only for messengers)
    3   1.000000000e+00      # Number of flavors
    4   0.500000000e+00      # Hypercharge
    5   2.000000000e+00      # SU2 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    6   1.000000000e+00      # SU3 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    7   2.000000000e+05      # LambdaG
    8   4.000000000e+10      # LambdaS squared
    0   0.0                  # End of HidField Block
```

The data in the HIDFIELD block is given according to the following:

- 1 - The fermionic mass of the superfield (in GeV).
- 2 - Str$[\mathcal{M}^2] = \sum m_b^2 - 2m_f^2$ (in GeV^2). This quantity is relevant for the 2-loop contribution to the beta function of the scalar masses (eq. (3)). If the hidden field is not a messenger, the value of this quantity can be set to zero.
- 3 - The number of flavors, which is originally defined for QCD, will be used here to denote the degeneracy, or the number of copies of this field, for any type of field.
- 4 - The hypercharge is given by its numerical value.
- 5,6 - The representations under SU(2) and SU(3) which are supported in HidSecSOFTSUSY appear in table A.2 and are represented by the numerical value specified there.
- 7,8 - The soft ~~SUSY~~ scales defined in eq. (4) are given in GeV (Λ_G in GeV, and Λ_S^2 in GeV^2). If the hidden field is not a messenger, the value of these quantities should be set to zero.
- 0 - Each HIDFIELD block must end with a line with index 0.

The hidden sector running parameters are defined in an optional block in the following format

```
Block HIDSECPARS             # Input parameters
    1   3.000000000e+00      # Number of HidSecPars
    2   1.000000000e-01      #
    2   1.000000000e-02      #
    2   1.000000000e-03      #
```

The first integer input number indicates the number of running parameters in the hidden sector, and the next items in the list give their value at the boundary[1].

5.2. User defined beta-functions

The user defined beta-function accepts a const pointer for the `HidSecSoftsusy` class, and an array of numbers containing the value of the beta functions at the current energy scale. The list of the physical parameters in this array is given in table A.3. The values of the beta function can be modified according to the new physics introduced in the model. The beta functions for the hidden sector parameters appear at the end of the array, at the same order as defined in the input file[2].

```
void (*userDefinedHidSecBeta)    (const HidSecSoftsusy *, DoubleVector &);
```

5.3. User defined threshold effect

As we go down the RG flow, and the scale defined by the mass of a hidden field is reached, the theory is replaced by an effective theory where the hidden field is integrated out. Due to the integration out of the heavy modes, terms in the Lagrangian are generated or modified. In a similar manner, when we go up the RG flow, and cross the mass threshold, the hidden field has to be integrated in. In `HidSecSOFTSUSY` the threshold effects must be written explicitly in a user defined function. These functions (one for integrating out and one for integrating in) accept as input a pointer to the `HidSecSoftsusy` class, and the index of the field being integrated in/out[3]. Unlike the user-defined-beta-functions, in the threshold-effect-function the user can modify the MSSM parameters themselves, not only the beta functions, and therefore extra care is necessary when introducing changes.

```
void (*userDefinedIntegrateOutHidField)(HidSecSoftsusy *, int);
void (*userDefinedIntegrateInHidField) (HidSecSoftsusy *, int);
```

6. Running `HidSecSOFTSUSY`

In order to use `HidSecSOFTSUSY` the user must have a version of `SOFTSUSY` installed. The instructions for the installation can be found on the project's homepage. The `HidSecSOFTSUSY` files should be copied into the project's directory[4], and the project should be recompiled:

```
> ./configure
> make
```

`HidSecSOFTSUSY` produces an executable called `hidsecsoftsusy.x` which accepts input in the SUSY Les Houches Accord 2 (SLHA2) [5] input/output option. The user must provide a file (*e.g.* the example files included in the `HidSecSOFTSUSY` distribution `HidSecInputSU5_adjoint` and `HidSecInput_EOGM`), that specifies the model dependent input parameters. The code may then be run with

```
> ./hidsecsoftsusy.x < input_file_name
```

The output of the provided executable is based on the `SOFTSUSY` output in the SLHA2 format, followed by a list of the hidden fields with their mass and the hidden sector parameters with their values at the low energy boundary.

[1] The line which contains the number of hidden sector parameters is marked by the index `1` and lines listing the boundary values of these parameters should be marked by `2`.

[2] In order to read the MSSM parameters from the `HidSecSoftsusy` class one can use the output member functions of `MSSMSoftsusy`, or use the member function `display()` which returns an array of parameters which are organized at the same order appearing in table A.3.

[3] Note that in `HidSecSoftsusy` the hidden fields are held in an increasing mass order.

[4] The files `Makefile.am` and `Makefile.in` should be overwritten.

7. HidSecSOFTSUSY usage examples

In this section we describe three scenarios where HidSecSOFTSUSY can be used, and the required modification in the input file and software.

7.1. Charged pseudo-moduli

In the model described in [2] the hidden sector contains a TeV scale superfield in the adjoint representation of $SU(5)$, which decomposes in the following way under $SU(3) \times SU(2)_L \times U(1)_Y$

$$\mathbf{24} = (\mathbf{8},\mathbf{1})_0 \oplus (\mathbf{1},\mathbf{3})_0 \oplus (\mathbf{3},\mathbf{2})_{-5/6} \oplus (\bar{\mathbf{3}},\mathbf{2})_{5/6} \oplus (\mathbf{1},\mathbf{1})_0 \, . \tag{6}$$

In order to add these fields to the HidSecSOFTSUSY running, the input file should contain the following blocks:

```
 ######################################################################
 Block HIDFIELD              # SU3 Octet superfield
    1   5.000000000e+04      # Mass
    2   0.000000000e+00      # Mass2STrace
    3   1.000000000e+00      # N Flavors
    4   0.000000000e+00      # Hypercharge
    5   1.000000000e+00      # SU2 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    6   4.000000000e+00      # SU3 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    0   0.0                  # End of HidField Block
######################################################################
Block HIDFIELD              # SU2 Triplet superfield
    1   5.000000000e+04      # Mass
    2   0.000000000e+00      # Mass2STrace
    3   1.000000000e+00      # N Flavors
    4   0.000000000e+00      # Hypercharge
    5   4.000000000e+00      # SU2 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    6   1.000000000e+00      # SU3 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    0   0.0                  # End of HidField Block
######################################################################
 Block HIDFIELD              # SU3xSU2 bifundamental superfield
    1   5.000000000e+04      # Mass
    2   0.000000000e+00      # Mass2STrace
    3   1.000000000e+00      # N Flavors
    4  -0.833333333e+00      # Hypercharge
    5   2.000000000e+00      # SU2 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    6   2.000000000e+00      # SU3 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    0   0.0                  # End of HidField Block
######################################################################
Block HIDFIELD              # SU3xSU2 bifundamental superfield
    1   5.000000000e+04      # Mass
    2   0.000000000e+00      # Mass2STrace
    3   1.000000000e+00      # N Flavors
    4   0.833333333e+00      # Hypercharge
    5   2.000000000e+00      # SU2 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    6   3.000000000e+00      # SU3 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    0   0.0                  # End of HidField Block
######################################################################
Block HIDFIELD              # singlet superfield (part of SU(5) adjoint)
    1   5.000000000e+04      # Mass
    2   0.000000000e+00      # Mass2STrace
    3   1.000000000e+00      # N Flavors
    4   0.000000000e+00      # Hypercharge
    5   1.000000000e+00      # SU2 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    6   1.000000000e+00      # SU3 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    0   0.0                  # End of HidField Block
```

7.2. EOGM

In models with multiple messenger scales the MODSEL block in the input file should be set to 4

```
Block MODSEL      # Select model
    1    4      # sugra=1, gmsb=2, amsb=3, eogm=4
```

and the MINPAR should not contain data regarding the messenger and SUSY breaking scales:

```
Block MINPAR      # Input parameters
    1   0.000000000e+00      # m gravitino
    3   1.000000000e+01      # tanb
    4   1.000000000e+00      # sign(mu)
```

The blocks defining the various messengers as hidden fields should contain the information regarding the scale of soft SUSY terms generated when they are integrated out, and the supertrace of their mass squared:

```
Block HIDFIELD               # SU2 doublet messenger
    1   1.000000000e+10      # Mass
    2   1.000000000e+08      # Mass2STrace
    3   1.000000000e+00      # N Flavors
    4  -0.500000000e+00      # Hypercharge
    5   2.000000000e+00      # SU2 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    6   1.000000000e+00      # SU3 rep (SINGLET=1, FUND=2, ANTIFUND=3, ADJOINT=4)
    7   2.000000000e+05      # LambdaG
    8   4.000000000e+10      # LambdaS2
    0   0.0                  # End of HidField Block
```

7.3. Running hidden sector parameters

Consider a model where all the light hidden sector fields are charged under a $U(1)_D$ gauge group, which is weakly mixed with the $U(1)_{em}$ (like in theories of hidden dark sector e.g. [7]) and assume the following beta function for the coupling constant:

$$\beta_{g_D} = \sum_i (N_f)_i \frac{g_D^3}{16\pi^2} \tag{7}$$

where the sum goes over the dynamical fields. The coupling constant and its value at the UV boundary should be introduced into the input file by adding the following block:

```
Block HIDSECPARS             # Input parameters
    1   1.000000000e+00      # Number of HidSecPars
    2   1.000000000e-03      #
```

In order to use the RGE engine of SOFTSUSY in determining the low energy value of the gauge coupling, the following function can be added to the software

```
void MYuserDefinedHidSecBeta    (const HidSecSoftsusy * h, DoubleVector & d)
{

   //The current energy scale
   double ThisMu = h->displayMu();

   //The current value of the U(1)_D coupling in the beta function array
   double gD =  h->displayHidSecPar(1);

   double gDbetaValue = 0;
   const double oneO16Pisq = 1.0 / (16.0 * sqr(PI));

   //Adding the contributions of all the dynamical hidden sector fields
   for (int i = 1 ; i <= h->displayNHidFields() ; i++)
```

```
   {
     if (ThisMu > h->displayHidFieldMass(i))
     {
        gDbetaValue += oneO16Pisq * (gD*gD*gD)
                         * h->displayHidFieldRepDimension(i,SU2)
                         * h->displayHidFieldRepDimension(i,SU3)
                         * h->displayHidFieldNFlavors(i);
     }
   }

   //The index of the U(1)_D coupling in the beta function array
   int    gDIndex = numSoftParsMssm + 1;

   //Modifying the beta function array accordingly
   d.set(gDIndex , gDbetaValue);
}
```

Acknowledgments

This software was written during research work with Y. Oz and L. Mazucatto.

Appendix A. Object Structure

The class in the SOFTSUSY package in which the RG flow computation is implemented is named MSSMSoftsusy. From the class RGE it inherits the engine which performs the iterative algorithm for solving the coupled differential equations, and it also contains all the data regarding the MSSM parameters. In HidSecSOFTSUSY we defined two new classes. The first, named HidField, holds the relevant information regarding a single field in the hidden sector. The second class, HidSecSoftsusy, inherits all the properties of MSSMSoftsusy, with an additional array of HidField objects, and a vector of numbers which can contain the hidden sector parameters.

Appendix A.1. Class HidField

This class contains the physical parameters which define the field in the hidden sector. The list of parameters appears in table A.1. The gauge group and representation are defined using enum types MSSMGaugeGroup and HidFieldRep which appear in table A.2. In addition to the IO methods, it contains methods which compute the Casimir, the Dynkin index and the dimension of the representation under the 3 standard model gauge groups.

Appendix A.2. Class HidSecSoftsusy

HidSecSoftsusy is a derived class of MSSMSoftsusy, the class which implements the MSSM spectrum calculation in SOFTSUSY. In addition to the MSSM information it contains arrays of hidden fields and hidden sector parameters, and pointers to user defined functions. The class contains the following member functions:

Appendix A.2.1. `void addHidField (const HidField * newHidField)`

This method accepts a pointer to HidField and copies its content into the HidFields array, maintaining an order by increasing mass.

Appendix A.2.2. `int run (double xi, double xf, double eps)`

The run method of MSSMSoftsusy which implements the RG flow between two energy scales is overloaded. In the HidSecSoftsusy version of the method the running is divided into steps which are determined by the masses of the hidden fields. At each step the method IntegrateOutHidField (or IntegrateInHidField) is called.

data variable		**methods**
`Double Mass` M [GeV]	Fermionic mass	`setMass`, `displayMass`
`Double HyperCharge` `HidFieldRep SU2Rep, SU3Rep`	Hypercharge and representation under SU(2) and SU(3)	`setSU2Rep`, `setSU3Rep`, `setHyperCharge`, `displayHyperCharge`, `displaySU2Rep`, `displaySU3Rep`, `displayCasimir`, `displayDynkinIndex`, `displayRepDimension`
`int Nflavors` N_f	Number of flavors	`setNflavors`, `displayNflavors`
`Double LambdaG, LambdaS2` Λ_G [GeV], Λ_S^2 [GeV^2]	Gaugino and scalar mass scales generated by integrating out this field.	`setLambdaG`, `setLambdaS2` `displayLambdaG`, `displayLambdaS2`
`Double StrM2` Str$\{\mathcal{M}^2\}$ [GeV^2]	Supertrace of the mass matrix.	`setStrM2`, `displayStrM2`

Table A.1: `HidField` class data and I/O methods.

enum type		**Identifier**	**Group**	**enum type**		**Identifier**	**Representation**
`MSSMGaugeGroup`	1	`U1`	$U(1)$	`HidFieldRep`	1	`SINGLET`	Singlet
	2	`SU2`	$SU(2)$		2	`FUND`	Fundamental
	3	`SU3`	$SU(3)$		3	`ANTIFUND`	Anti-fundamental
					4	`ADJOINT`	Adjoint

Table A.2: `MSSMGaugeGroup` and `HidFieldRep` enum types.

#	**Physical parameter**	#	**Physical parameter**
1 – 9	u quark Yukawa couplings	37 – 45	u-Higgs quark trililnear couplings
10 – 18	d quark Yukawa couplings	46 – 54	d-Higgs quark trililnear couplings
19 – 27	lepton Yukawa couplings	55 – 63	lepton-Higgs trililnear couplings
28	$U(1)$ gauge coupling	64 – 72	quark doublet scalar mass squared
29	$SU(2)$ gauge coupling	73 – 81	$\bar{u}$ quark scalar mass squared
30	$SU(3)$ gauge coupling	82 – 90	$\bar{d}$ quark scalar mass squared
31	μ	91 – 99	lepton doublet scalar mass squared
32	$\tan\beta$	100 – 108	$\bar{e}$ doublet scalar mass squared
33	The Higgs VEV	109	$B\mu$
34	$U(1)$ gaugino mass	110	m^2_{Hd}
35	$SU(2)$ gaugino mass	111	m^2_{Hu}
36	$SU(3)$ gaugino mass	112 – ...	hidden sector parameters

Table A.3: The order of physical parameters in the beta function array.

Appendix A.2.3. `void IntegrateOutHidField (int i), IntegrateInHidField (int i)`

This method calls the user defined function for integrating out/in a hidden sector field. If the hidden sector field is a messenger (it has non-zero Λ_G or Λ_S), it also calls `AddOrRemoveMessenger` which implements eq. (4). Users who wish to provide the user defined function should note that the serial number of the hidden sector, `i`, is determined by its mass.

Appendix A.2.4. `DoubleVector beta() const`

The `beta` method of `MSSMSoftsusy` which computes the derivative of the MSSM couplings is modified to include the hidden sector contributions discussed in sec. 3. It also calls the user defined beta function. The output is held in a `DoubleVector` array. Users who wish to provide the user defined beta function can use table A.3 which lists the location of the various physical parameters in this array.

Appendix A.2.5. `bool AnomalyCancellation()`

This method checks whether the hidden fields satisfy the anomaly cancellation conditions.

References

[1] B. C. Allanach, SOFTSUSY: A C++ program for calculating supersymmetric spectra, Comput. Phys. Commun. 143 (2002) 305–331. arXiv:hep-ph/0104145, doi:10.1016/S0010-4655(01)00460-X.

[2] B. Keren-Zur, L. Mazzucato and Y. Oz, JHEP **0810**, 099 (2008) [arXiv:0807.4543 [hep-ph]].

[3] C. Cheung, A. L. Fitzpatrick and D. Shih, JHEP **0807**, 054 (2008) [arXiv:0710.3585 [hep-ph]].

[4] P. Skands, et al., SUSY Les Houches accord: Interfacing SUSY spectrum calculators, decay packages, and event generators, JHEP 07 (2004) 036. arXiv:hep-ph/0311123.

[5] B. Allanach, et al., SUSY Les Houches Accord 2, Comp. Phys. Commun. 180 (2009) 8–25. arXiv:0801.0045, doi:10.1016/j.cpc.2008.08.004.

[6] K. A. Intriligator, N. Seiberg and D. Shih, JHEP **0604**, 021 (2006) [arXiv:hep-th/0602239].

[7] N. Arkani-Hamed, D. P. Finkbeiner, T. R. Slatyer and N. Weiner, Phys. Rev. D **79** (2009) 015014 [arXiv:0810.0713 [hep-ph]].