Implementation of the Split-MSSM into FlexibleSUSY

Alexander Voigt

FeynHiggs meeting
DESY Hamburg, Germany

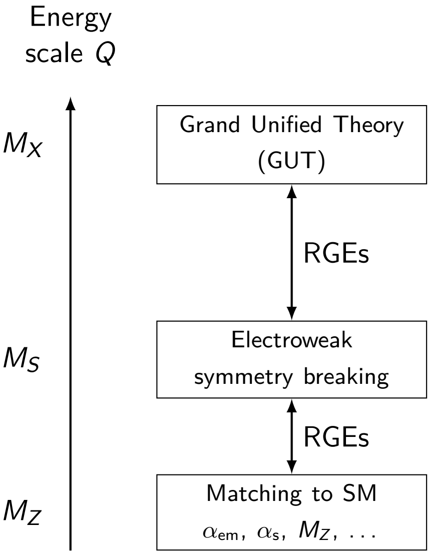
31.03.2015

Contents

FlexibleSUSY = spectrum generator generator

FlexibleSUSY

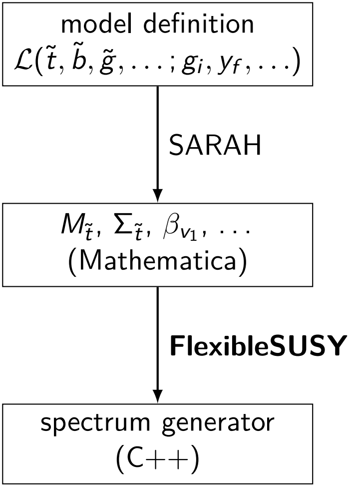
FlexibleSUSY's physical problem statement



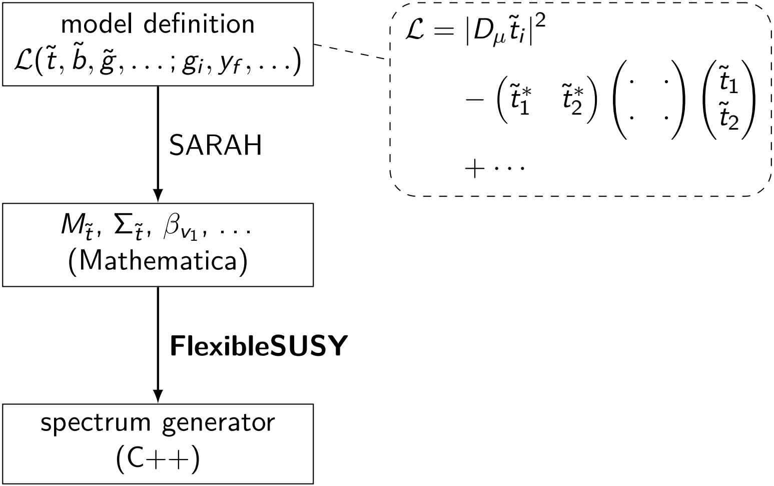
FlexibleSUSY's Weltanschauung

- Model is defined in terms of Lagrangian parameters:
  g_i, y_{ij}, v, … in the $\overline{\text{MS}}/\overline{\text{DR}}$ scheme
- Input parameters:
  $\alpha_{\text{em,SM}}^{(5),\overline{\text{MS}}}(M_Z)$, $\alpha_{\text{s,SM}}^{(5),\overline{\text{MS}}}(M_Z)$, M_Z, M_t, G_F, …
- Output parameters:
  m_h, M_h, Z_h, …

Generating a spectrum generator



Generating a spectrum generator



Generating a spectrum generator

model definition
$\mathcal{L}(\tilde{t}, \tilde{b}, \tilde{g}, \ldots; g_i, y_f, \ldots)$

↓ SARAH

$M_{\tilde{t}}$, $\Sigma_{\tilde{t}}$, β_{v_1}, ...
(Mathematica)

↓ **FlexibleSUSY**

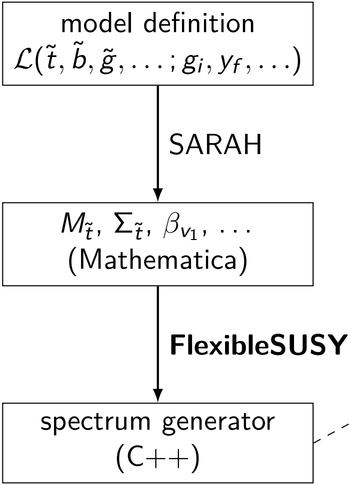
spectrum generator
(C++)

$$M_{\tilde{t}} = \begin{pmatrix} \cdot & \cdot \\ \cdot & \cdot \end{pmatrix}$$

$\Sigma_{\tilde{t}} =$  $+ \cdots$

$$\beta_{v_1} = \frac{v_1}{16\pi^2}\left(\frac{3}{10}g_1^2 + \frac{3}{2}g_2^2\right) + \cdots$$

Generating a spectrum generator




```
Matrix<2,2> get_mass_matrix_St() {
  Matrix<2,2> mass_matrix;

  mass_matrix(0,0) = ...;
  mass_matrix(0,1) = ...;
  mass_matrix(1,0) = ...;
  mass_matrix(1,1) = ...;

  return mass_matrix;
}

complex<double> self_energy_St() {
  complex<double> self_energy;

  self_energy += ...;
  self_energy += ...;
  self_energy += ...;

  return self_energy;
}

double beta_v1() {
  double beta_v1;

  beta_v1 = v1*(0.3*Sqr(g1)
     + 1.5*Sqrt(g2))/(16.*Sqr(Pi) + ...;

  return beta_v1;
}
```

FlexibleSUSY features

- **Precision**
  $\beta_i^{(1L)}$, $\beta_i^{(2L)}$, $\Sigma_i^{(1L)}$, $t_{h_i}^{(1)}$
  rMSSM: $\Sigma_{h_i}^{(2L,\alpha_t\alpha_s+(\alpha_t+\alpha_b)^2)}$, $t_{h_i}^{(2L,\alpha_t\alpha_s+(\alpha_t+\alpha_b)^2)}$
  rNMSSM: $\Sigma_{h_i}^{(2L,\alpha_t\alpha_s+(\alpha_t+\alpha_b)^2)}$, $t_{h_i}^{(2L,\alpha_t\alpha_s+(\alpha_t+\alpha_b)^2)}$
- **Tower of EFTs**
- Modular, well readable C++ code
- High speed
- Multiple BVP solvers

Contents

Split-MSSM Lagrangian

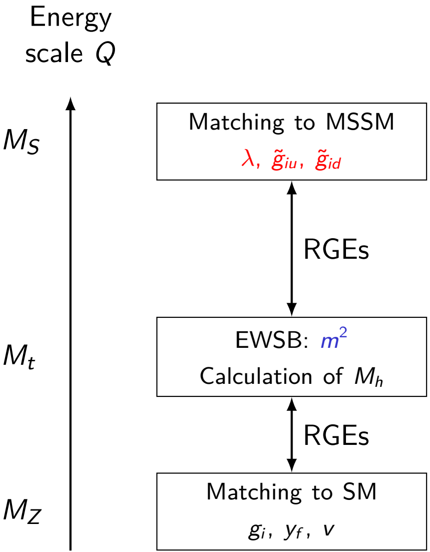
$$\begin{aligned}
\mathcal{L}_{\text{split}} &= \mathcal{L}_{\text{kin}} + \mathcal{L}_{\text{yuk}} + \mathcal{L}_{\text{Higgs}} + \mathcal{L}_{\text{Higgsino}} \\
\mathcal{L}_{\text{Higgs}} &= m^2 H^\dagger H - \frac{1}{2}\lambda (H^\dagger H)^2 \\
\mathcal{L}_{\text{Higgsino}} &= -\mu \tilde{H}_u^T \epsilon \tilde{H}_d \\
&\quad - \frac{M_3}{2}\tilde{g}^A \tilde{g}^A - \frac{M_2}{2}\tilde{W}^a \tilde{W}^a - \frac{M_1}{2}\tilde{B}\tilde{B} \\
&\quad - \frac{H^\dagger}{\sqrt{2}}\left(\tilde{g}_{2u}\sigma^a \tilde{W}^a + \tilde{g}_{1u}\tilde{B}\right)\tilde{H}_u \\
&\quad - \frac{H^T \epsilon}{\sqrt{2}}\left(-\tilde{g}_{2d}\sigma^a \tilde{W}^a + \tilde{g}_{1d}\tilde{B}\right)\tilde{H}_d + \text{h.c.}
\end{aligned}$$

Fixed by matching conditions: λ, $\tilde{g}_{1u}$, $\tilde{g}_{2u}$, $\tilde{g}_{1d}$, $\tilde{g}_{2d}$

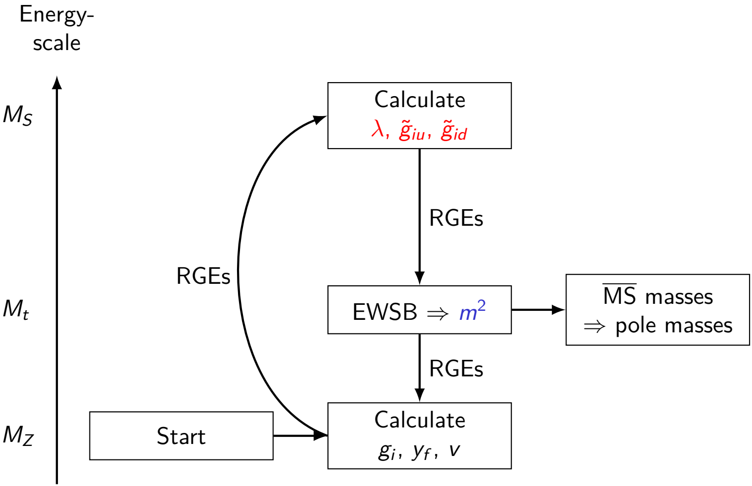
Fixed by EWSB: m^2

Input at M_Z: μ, M_1, M_2, M_2

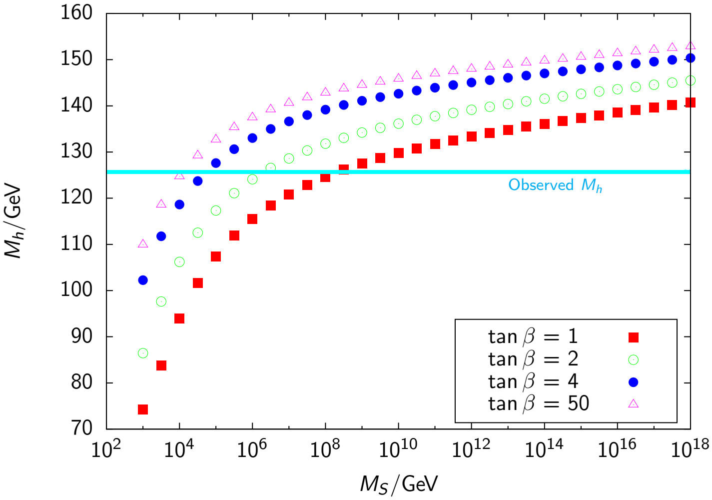
Physical problem statement



Algorithm to calculate the pole mass spectrum



Results [preliminary]



$$\mu = M_i = m_q^2 = m_u^2 = m_d^2 = m_l^2 = m_e^2 = 1\,\text{TeV},\ m_A = M_S,\ A_t = 0$$

Future plans

- cross-checking the implementation with Split-Suspect
- add leading $\Sigma_h^{(2)}$
- study CMSSM-like scenarions
  $\rightarrow$ requires complete matching to the MSSM for
  μ, M_i, A_t, $m_q^2, m_u^2, m_d^2, m_l^2, m_e^2$

? implementation into FeynHiggs

? generalization of the threshold corrections to general SUSY models

Thank you!

Backup

Calculation of λ, $\tilde{g}_{1u}$, $\tilde{g}_{2u}$, $\tilde{g}_{1d}$, $\tilde{g}_{2d}$

Matching conditions Split-MSSM $\leftrightarrow$ MSSM lead to

$$\tilde{g}_{1u} = \frac{3}{5} g_1 \sin\beta + \Delta\tilde{g}_{1u}$$

$$\tilde{g}_{1d} = \frac{3}{5} g_1 \cos\beta + \Delta\tilde{g}_{1d}$$

$$\tilde{g}_{2u} = g_2 \sin\beta + \Delta\tilde{g}_{2u}$$

$$\tilde{g}_{2d} = g_2 \cos\beta + \Delta\tilde{g}_{2d}$$

$$\lambda = \frac{1}{4}\left[g_2^2 + \frac{3}{5} g_1^2\right] \cos^2 2\beta + \Delta\lambda$$

The corrections $\Delta\tilde{g}_{iu}$, $\Delta\tilde{g}_{id}$, $\Delta\lambda$ are functions of

$$g_i, y_t, A_t, \mu, m_A, \tan\beta, m_q^2, m_u^2, m_d^2, m_l^2, m_e^2$$

Calculation of $g_3^{\overline{\text{DR}}}(M_Z)$

Input: $\alpha_{\text{s,SM}}^{(5),\overline{\text{MS}}}(M_Z) = 0.1185$

$\rightarrow$

$$\alpha_{\text{s}}^{\overline{\text{DR}}}(M_Z) = \frac{\alpha_{\text{s,SM}}^{(5),\overline{\text{MS}}}(M_Z)}{1 - \Delta\alpha_{\text{s,SM}}(M_Z) - \Delta\alpha_{\text{s}}(M_Z)}$$

with

$$\Delta\alpha_{\text{s,SM}}(\mu) = \frac{\alpha_{\text{s}}}{2\pi}\left[-\frac{2}{3}\log\frac{m_t}{\mu}\right]$$

$$\Delta\alpha_{\text{s}}(\mu) = \frac{\alpha_{\text{s}}}{2\pi}\left[\frac{1}{2} - \sum_{\text{SUSY particle } f} T_f \log\frac{m_f}{\mu}\right]$$

$\Rightarrow$

$$g_3^{\overline{\text{DR}}}(M_Z) = \sqrt{4\pi\alpha_{\text{s}}^{\overline{\text{DR}}}(M_Z)}$$

Calculation of $y_t^{\overline{\mathrm{DR}}}(M_Z)$

Input: m_t^{pole}

$\rightarrow$

$$
\begin{aligned}
m_{t,\text{susy}}^{\overline{\mathrm{DR}}}(\mu) &= m_t^{\text{pole}} + \mathrm{Re}\,\Sigma_t^S(m_t^{\text{pole}}) \\
&\quad + m_t^{\text{pole}}\Big[\mathrm{Re}\,\Sigma_t^L(m_t^{\text{pole}}) + \mathrm{Re}\,\Sigma_t^R(m_t^{\text{pole}}) \\
&\qquad + \Delta m_t^{(1),\text{qcd}} + \Delta m_t^{(2),\text{qcd}}\Big] \\
\Delta m_t^{(1),\text{qcd}} &= -\frac{g_3^2}{12\pi^2}\left[5 - 3\log\left(\frac{m_t^2}{\mu^2}\right)\right] \\
\Delta m_t^{(2),\text{qcd}} &= \left(\Delta m_t^{(1),\text{qcd}}\right)^2 \\
&\quad - \frac{g_3^4}{4608\pi^4}\Bigg[396\log^2\left(\frac{m_t^2}{\mu^2}\right) - 1476\log\left(\frac{m_t^2}{\mu^2}\right) \\
&\qquad - 48\zeta(3) + 2011 + 16\pi^2(1+\log 4)\Bigg]
\end{aligned}
$$

Split-MSSM EWSB condition

$$\begin{aligned}
0 &= \frac{\partial V_{\text{eff}}^{(1)}}{\partial v} \\
&= \frac{\partial V_{\text{tree}}}{\partial v} - t_h^{(1)}(m_i) \\
&= \sqrt{2} v \left(-m^2 + v^2 \lambda\right) - t_h^{(1)}(m_i)
\end{aligned}$$

$\rightarrow$ solve iteratively for m^2

Calculation of the Higgs pole mass

Solve for M_h^2 at $p^2 = M_h^2$:

$$0 = p^2 - m_h + \hat{\Sigma}_h(p^2)$$

where

$$m_h = -m^2 + 3\lambda v^2$$

and

$$\begin{aligned}
\hat{\Sigma}_h(p^2) &= \Sigma_h(p^2) - \delta m_h^2 + (p^2 - m_h^2)\delta Z_h \\
\delta m_h^2 &= \Sigma_h(p^2)\Big|_\Delta \\
\delta Z_h &= -\Sigma_h'(p^2)\Big|_\Delta
\end{aligned}$$

NMSSM-Spektrumgenerator in FlexibleSUSY

1. Get the source code from https://flexiblesusy.hepforge.org
2. Create a NMSSM spectrum generator:

```
$ ./install-sarah # if not already installed
$ ./createmodel --name=NMSSM
$ ./configure --with-models=NMSSM
$ make
```

3. Calculate spectrum for given parameter point (SLHA format):

```
$ ./models/NMSSM/run_NMSSM.x \
   --slha-input-file=models/NMSSM/LesHouches.in.NMSSM

Block MASS
   1000021     5.05906233E+02   # Glu
   1000024     1.46609728E+02   # Cha_1
   1000037     3.99399367E+02   # Cha_2
        37     4.33363816E+02   # Hpm_2
...
```

Definition of SplitMSSM boundary conditions

```
FSModelName = "SplitMSSM";

MINPAR = { {3, TanBeta} };

EXTPAR = { {0, MS},
           {1, MT},
           {2, MuInput},
           {3, M1Input},
           {4, M2Input},
           {5, M3Input},
           {6, mAInput} };

EWSBOutputParameters = { m2 };

HighScale = MS;
SUSYScale = MT;
LowScale  = SM[MZ];

LowScaleInput = { {v, Sqrt[2] MZDRbar / Sqrt[gY^2 + g2^2]},
                  {\[Mu], MuInput},
                  {M1, M1Input},
                  {M2, M2Input},
                  {M3, M3Input} };
```

Generated SplitMSSM spectrum generator C++ code

```
typedef Two_scale T; // or Lattice
SplitMSSM<T> split;
SplitMSSM_input_parameters input;
QedQcd qedqcd;

// create BCs
std::vector<Constraint<T>*> constraints = {
   new SplitMSSM_low_scale_constraint<T>(input,qedqcd),
   new SplitMSSM_susy_scale_constraint<T>(input),
   new SplitMSSM_high_scale_constraint<T>(input)
};

// solve RG eqs. with the above BCs
RGFlow<T> solver;
solver.add_model(&split, constraints);
solver.solve();

split.calculate_spectrum();
```