SuSpect3

Michaël Ughetto
ughetto@lal.in2p3.fr

LAL/IN2P3

April 19, 2011

ID Card

Name: SuSpect
Job: Calculate SuSy spectrum using RGEs
Parents: Djouadi, Kneur & Moultaka
Birthdate: 2002 (updated regularly)

Find more here: `http://arxiv.org/abs/hep-ph/0211331`
Source and support:
`http://web.lupm.univ-montp2.fr/users/kneur/Suspect/`

SuSpect features

- It supports
  - mSUGRA
  - pMSSM (classical, highscale, bottom-up)
  - GMSB
  - AMSB
  - SUSY with heavy scalars (special version of SuSpect)
- It includes full two-loop RGEs, radiative corrections
- It has been widely tested and is a trustable spectrum calculator

SuSpect3

Goals:

- Keep precision of SuSpect Fortran
- Try to improve on flexibility/ease of use/implementation of new models by using:
  - C++
  - OOP
- Try mSUGRA as a test case

SuSpect

Step 1: Low energy input

$\alpha(M_Z)$,$\alpha_S(M_Z)$, $M_t^{\rm pole}$, $M_\tau^{\rm pole}$,$m_b^{\overline{\rm MS}}(m_b)$, $M_Z^{\rm pole}$,etc.
Translation to $\overline{\rm DR}$

Step 2: One– or two–loop RGEs running

RGEs with choice: $g_1 = g_2 \cdot \sqrt{3/5}$
$M_{\rm GUT} \sim 2 \cdot 10^{16}$ GeV

Step 3: Choice of SUSY-breaking model

mSUGRA, GMSB, AMSB, or pMSSM. Choice of high-energy input, eg:
mSUGRA: m_0, $m_{1/2}$, A_0, sign(μ) and $\tan\beta$

Step 4: EWSB

Run down all parameters to m_Z and $M_{\rm EWSB}$ scales
Calculate μ^2,
$\mu B = F(m_{H_u}, m_{H_d}, \tan\beta, V_{\rm loop})$

Step 5: Testing EWSB

Check of consistent EWSB (μ convergence, no tachyons, simple CCB/UFB, etc.)

Step 6: Masses and corrections

Diagonalization of mass matrices and calculation of masses/couplings
Radiative corrections to the physical Higgs, sfermions, gauginos masses

Overview

main.cxx

- `SUSPECT::suspect aSuspectCalculation;`
- `aSuspectCalculation.Initialize(SLHAstructure);`
  - Read inSLHAfile and fill a SLHA object
  - Initialize the model according to MODSEL
    - `m_model = new SUSPECT::ModelmSUGRA(m_SLHAblock);`
    - `m_model = new SUSPECT::ModelpMSSM(m_SLHAblock);`
    - `m_model = new SUSPECT::ModelGMSB(m_SLHAblock);`
    - `m_model = new SUSPECT::ModelAMSB(m_SLHAblock);`
    - ...
- `aSuspectCalculation.Execute();`
  - `m_model->Execute();`
    - `m_DRparam.Execute();`
    - `m_RGErunner.Initialize(log(m_scaleMZ),log(m_s...`
    - `m_RGErunner.Execute();`
    - ...
    - `FinalizeMasses(m_scaleEWSB);`
- `aSuspectCalculation.Finalize(verbose,outSLHAfile);`

SLHA

The common data storage structure

SUSPECT::SLHA4suspect, the SLHA type

All physical parameters are stored in an internal SLHA structure containing 4 kinds of SLHA substructures

Vector

EXTPAR, MASS, SU_ALGO, SMINPUT, MINPAR, MODSEL

VectorQ

HMIX, GAUGE, MSOFT, SU_RADCORR

Matrix

NMIX, UMIX, VMIX, STOPMIX, SBOTMIX, STAUMIX

MatrixQ

AE, AD, AU, AN, YE, YD, YU, YN

Of course, those come with accessors, mutators, etc.

the mSUGRA class

The `mSUGRA` class inherits of `SUSPECT::ModelBase::ModelBase(SUSPECT::SLHA4suspect *theSLHAblock)`, and contains:

- Common utilies for models (ODE integration object, DR translation, etc.)
- MassesEigenstates objects
- Radiative corrections objects

Furthermore, `SUSPECT::ModelmSUGRA::ModelmSUGRA(SUSPECT::SLHA4suspect *theSLHAblock)` adds model-specific features:

- Configures scales needed for the run: M_Z, M_{EWSB} and M_{GUT}
- Prepare SLHA block for high enrgy input

The **Execute()** method implements:

- The main loop, *ie*, the recursive bottom-up (coupling unification) and top-down (soft-breaking masses, tri-linear couplings, etc.)
- The EWSB convergence search

RgeEvolution::RgeEvolution(SUSPECT::SLHA4suspect *theSLHAblock)

Constructor:

- Read SU_ALGO in SLHA block to configure adaptive RK4 algorithm (stepsize, minimum accuracy, etc.)
- Initialize the RGEs with 1–loop or 2–loop depending on user choice. (This is done through polymorphism)

Initialize(start scale, stop scale, unification search: YES/NO):

- Read SLHA block and build the start parameter space out of it (y vector of fortran code)
- Fix the RGEs boundary scales
- Toggles ON/OFF the RGEs subgroup of gauge/yukawas

Execute():

- Run RGEs from start scale to stop scale using RK4 to solve RGE with the good number of loop
- Watch for unification, and exit integration loop when found

For example, GUT search is achieved executing the following code:

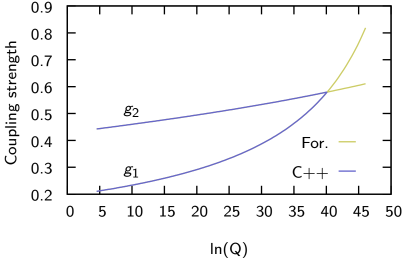
```
...
m_RGErunner(SUSPECT::SLHA4suspect *theSLHAblock);
...
m_RGErunner.Initialize(log(m_scaleMZ),log(m_scaleGUT),true);
m_RGErunner.Execute();
m_scaleGUT = m_RGErunner.UnificationScale();
...
```

MZ→GUT

$$M_{\mathrm{GUT}}(C++) = 2.67723\ 10^{17}\ GeV$$
$$M_{\mathrm{GUT}}(Fortran) = 2.67260\ 10^{17}\ GeV$$

$$\ln \frac{M_{\mathrm{GUT}}(Fortran)}{M_{\mathrm{GUT}}(C++)} = 0.01 =$$

1 step size!



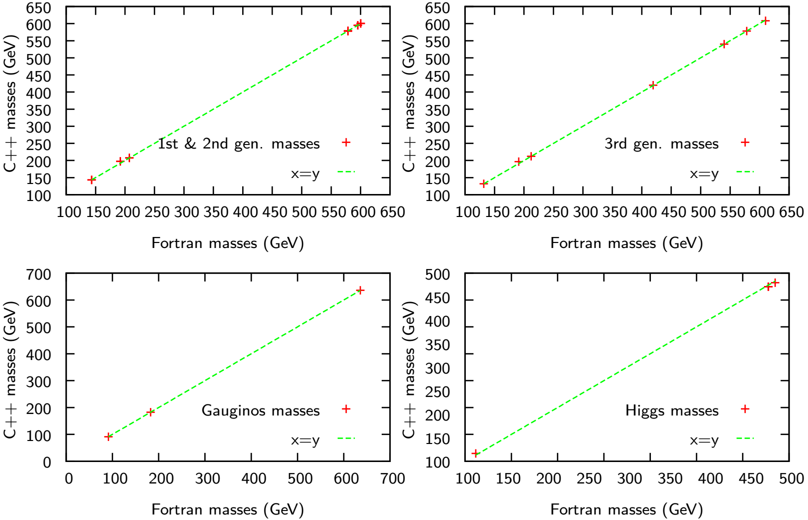
⇒ RGEs running and low energy input are **validated**
(tested for 1–loop and 2–loop RGEs)

Mass calculation

After running back to EWSB after GUT scale finding:

- The model calls `m_MassesEigenstates->Execute`
  - It calculates mass matrix elements, diagonalize them
  - It calculates mixing angles
  - If asked, this `Execute` method will Initialize and apply Radiative corrections (embedded object)
- Compares masses

After one loop: MZ→GUT→EWSB



A few comments on masses

To sum-up:

- Tests of masses calculus procedure
- Try to include the simplest radiative corrections

Result:

$$\frac{M_{Fortran} - M_{C++}}{M_{Fortran}} \text{ typically range from } 10^{-3} \text{ to } 10^{-5}$$

(including a difference due to 0.01 different GUT-scale)

$\Rightarrow$ Masses calculus is **OK**
(but yet to be perfected)

1st and 2nd generations

SOFTSUSY, SPHENO can separate light generations in MSSM, let's try it:

- Implement in C++
- Checks robustness of new code
- No change expected in mSUGRA
- Compared before/after with 11 digits: **OK**

But work is still to be done to take light generations in account within radiative corrections.

State for **mSuGra**

green is implemented and functional, red is currently being coded/tested

Step 1: Low energy input

$\alpha(M_Z)$,$\alpha_S(M_Z)$, M_t^{pole}, M_τ^{pole},$m_b^{\overline{\mathrm{MS}}}(m_b)$, M_Z^{pole},etc.
Translation to $\overline{\mathrm{DR}}$ *via* radiative corrections

Step 2: One– or two–loop RGEs running

RGEs with choice: $g_1 = g_2 \cdot \sqrt{3/5}$, $M_{\mathrm{GUT}} \sim 2 \cdot 10^{16}$ GeV

Step 3: Choice of SUSY-breaking model

mSUGRA, GMSB, AMSB, or pMSSM. Choice of high-energy input, eg:
mSUGRA: m_0, $m_{1/2}$, A_0, sign(μ) and $\tan\beta$

Step 4: EWSB

Run down all parameters to m_Z and M_{EWSB} scales
EWSB μ^2,
$\mu B = F_{\mathrm{non-linear}}(m_{H_u}, m_{H_d}, \tan\beta, V_{\mathrm{loop}})$

Step 5: Testing EWSB

Check of consistent EWSB (μ convergence, no tachyons, simple CCB/UFB, etc.)

Step 6: Masses and corrections

Diagonalization of mass matrices and calculation of masses/couplings
Radiative corrections to the physical Higgs, sfermions, gauginos masses

Bonus: A ROOT interface was added

```
SUSPECT::suspect2root suspectRoot(outRootFile,mode);
suspectRoot.Initialize(aSuspectCalculation.SLHAblock()->MODSEL(1));
suspectRoot.Fill(aSuspectCalculation);
```

To conclude

Reimplementation of Suspect in C++

- Project started 6 weeks ago
  - Low energy input: **OK**
  - RGEs: **OK**
  - Masses: **OK**
  - Radiative Corrextions Masses/SM implemented, fine comparison on-going
  - Main RGE-loop implemented
  - Short EWSB-loop implemented
- Short-terms goals
  - Finish mSUGRA complety
  - Harder task: will MSSM implementation be easier?