SSP

SARAH Scan and Plot

Version 1.2.2

Florian Staub
florian.staub@cern.ch

July 31, 2015

Disclaimer
`SSP` 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@th.physik.uni-bonn.de
- If you have any suggestions, what is missing or can be improved, please let us know.
- If you use this software, please cite `arXiv:1109.5147`

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

Contents

Chapter 1

Introduction

1.1 Purpose of SSP

SSP is supposed to be a handy tool doing parameter studies using the public tools SPheno [1, 2], WHIZARD [3, 4], HiggsBounds [5, 6], CalcHep [7, 8] and micrOMEGAs [9]. A description how to use all these programs together also for extensions of the MSSM based on model implementations in SARAH [10, 11, 12] is given in [13].

1.2 Download and installation

The package can be downloaded from

```
http://projects.hepforge.org/sarah/SSP.html
```

The package should be extracted to the application directory of Mathematica,

```
$HOME/.Mathematica/Applications/
```

on a Unix system and

```
[Mathematica-Directory]\AddOns\Applications\
```

in a Windows environment ($HOME and [Mathematica-Directory] should be substituted with the home and Mathematica installation directories respectively).

1.3 Running SSP

After SSP is loaded via

```
In[1]: <<"SSP/SSP.m"
```

a scan is started by

```
In[2]: Start["Inputfile"];
```

The input file contains all information about the parameter scan like what programs should be included in the scan, the range of the input values, the number of points, what should be plotted and so on. The information needed to run the different programs like their location or the name of the binaries, is saved separately in a second file.

Chapter 2

Inputfile

2.1 Setting the information for the different programs

In a first file, which has to be located in the root directory of `SSP`, the location and some basic information about the different tools have to be defined:

a) `SPheno`:

(a) Location of the `SPheno` binary

```
DEFAULT[SPheno] = "[SPheno Directory]/bin/SPheno[Model]";
```

(b) The name of the in- and output file

```
DEFAULT[SPhenoInputFile] = "LesHouches.in.[Model]";
DEFAULT[SPhenoSpectrumFile] = "SPheno.spc.[Model]";
```

b) `micrOMEGAs`

(a) The working directory of `micrOMEGAs`

```
DEFAULT[MicroOmegasDirectory] = "[Micromegas Directory]/[Model]";
```

(b) The name of the executable

```
DEFAULT[MicroOmegasBin] = "CalcOmega";
```

(c) The name of the in- and output file

```
DEFAULT[MicroOmegasBin] = "CalcOmega";
DEFAULT[MicroOmegasOutputFile] = "omg.out";
```

c) `HiggsBounds`:

(a) The command including the full path to run `HiggsBounds`, e.g

```
DEFAULT[HiggsBounds] = "[HB Directory]/HiggsBounds LandH effC 3 1";
```

Note, it is not necessary to define any path were the input files for `HiggsBounds` are expected to be.

d) `MadGraph`

(a) The installation directory of `MadGraph`

```
DEFAULT[MadGraph] = "[MG Directory]";
```

(b) The `MadGraph` model file

```
DEFAULT[MadGraphModel] = "[Model]";
```

(c) The output file

```
DEFAULT[MadGraphOutputFile] = "SubProcesses/results.dat";
```

e) `WHIZARD`

(a) The command to run `WHIZARD`

```
DEFAULT[WHIZARD]="/[WHIZARD Directory]/bin/whizard";
```

(b) The name of the parameter file which `SPheno` writes for `WHIZARD`

```
DEFAULT[WHIZARDparameterFile]="WHIZARD.par.[Model]";
```

f) `Vevacious`

g) The command to run the `Vevacious` binary

```
DEFAULT[VevaciousBin]="/[Vevacious Directory]/bin/Vevacious.exe";
```

h) The name of the initialization file used for the run

```
DEFAULT[VevaciousInit]="/[Vevacious Directory]/bin/VevaciousInitialization.xml";
```

2.2 Defining the scans

The input file to define the different scans contains the following information

a) The name of the file which contains the setup of the different tools

b) Identifiers for the different scans

c) What programs should be included

d) The input values or ranges for all parameter

e) Optionally, a list of constraints

f) A list of what should be plotted

Scans and output First, it is possible to define several scans within one input file. For that reason, a list with names for all scans must be given

```
RunScans = {ScanM0,ScanM12,ScanM0M12};
```

The output is organized as follows

- By default, all results and plots for a specify scan are saved in the directory

  `> [SSP Directory]/Output/[Name of Scan]`

- All calculated spectrum files are glued together to the files `SpectrumFiles.spc` and `SpectrumFiles_all.spc`. The difference between these two files is that the first one includes only points which fulfill a defined constraint, while in the second on the results for any point is saved.

- In addition, the results are also saved in `Mathematica` format in the file `data.m`

It is also possible to overwrite these default values and define the output directory as well as the name for the data and spectrum files

```
DEFINITION[a_][OutputDirectory]="Example1";
DEFINITION[ScanM0][SpectrumFile]="Spectra_Scan1_m0";
DEFINITION[ScanM0][OutputFile]="Data_Scan1_m0.m";
```

Note, to the defined name of the spectrum file the ending `.spc` respectively `_all.spc` are added.

Setting up the programs

- `SPheno` has always to be used, so there is no flag to disbable. However, for the other programs, it can be decedid if they should be included or not. By default, all other programs are disabled.

- To include `HiggsBounds`, just the corresponding flag has to be set to `True`, no further information is needed:

  ```
  DEFINITION[Scan][IncludeHiggsBounds] = True;
  ```

  That will automatically run `HiggsBounds` using the output of `SPheno`. Make sure, to set the corresponding flag (`SPhenoInfo 75`) to `1` in the `SPheno` input. The output file of `HiggsBounds` is read by `SSP` and the information is added the spectrum file. For instance, the following line in `HiggsBounds_result.dat`

  ```
  #cols:  n    Mh(1)    Mh(2)    Mh(3) Mhplus(1) HBresult chan obsratio ncomb
  #
          1  108.680  529.352  529.231  535.888     0       1  1.39906    1
  ```

  leads to the entry

  ```
  Block HIGGSBOUNDS #
   1    108.680 #
   2    529.352 #
   3    529.231 #
   4    535.888 #
  ```

```
5          0 #
6          1 #
7    1.39906 #
8          1 #
```

- To include dark matter scans, the corresponding flag has to be set to `True`. In addition, the PDG of the dark matter particle has to be given.

```
DEFINITION[ScanM0][IncludeMicrOmegas] = True;
DEFINITION[a_][DarkMatterCandidate] = 1000022;
```

If the relic density for all possible LSPs should be calculated, also `ALL` can be used instead of a specific number or it is possible to define several candidates by

```
DEFINITION[a_][DarkMatterCandidate] = 1000022 | 1000012 | 40000012;
```

Note, it is assumed that `micrOMEGAs` uses an SLHA input. Therefore, the `SPheno` spectrum file is copied to the `micrOMEGAs` directory and renamed to the given file name. The output file created by `micrOMEGAs` should just contain lines with numerical values (e.g. for the relic density and direct detection cross sections).

Micromegas Output format Each line in the file written by `micrOMEGAs` must have the following format

```
Integer Float # Comment
```

These lines are added afterwards to the spectrum file created by `SPheno` using a new block called `DARKMATTER`. Hence, the user has access to this information by `DARKMATTER[N]`, where `N` corresponds to the integer at the beginning of each line in the `micrOMEGAs` output file. For example, using the main file created by `SARAH` for `micrOMEGAs`, the content of output file `omg.out` reads

```
1   6.248656  # relic density
100 0.676687  # ~C01 ~C01 -> h1 Z
101 0.070703  # ~C01 ~C01 -> e3 E3
102 0.048270  # ~C01 ~C01 -> e2 E2
103 0.048204  # ~C01 ~C01 -> e1 E1
104 0.045455  # ~C01 ~C01 -> u3 U3
105 0.038574  # ~C01 ~C01 -> d3 D3
106 0.020219  # ~C01 ~C01 -> Wm Wp
107 0.016671  # ~C01 ~C01 -> h1 h1
```

Main annihilation channels `SSP` checks automatically if also a file `channels.out` is written by `micrOMEGAs`. This file is supposed to contain a list of the main annihilation channels given in the format

```
IN1 IN2 OUT1 OUT2 BR # Comment
```

where INx and OUTx are the PDGs of the particles in the inital respectively final states. If this is the case, the lines are added also to the spectrum file using the block DMCHANNELS. In this way the user can check for the size if specific channels contributing to the dark matter annihilation using DMCHANNESL[IN1,IN2,OUT1,OUT2]. For instance, the content of channles.out might be

```
1000022 1000022  25  23 0.676687 # ~C01 ~C01 -> h1 Z
1000022 1000022  15 -15 0.070703 # ~C01 ~C01 -> e3 E3
1000022 1000022  13 -13 0.048270 # ~C01 ~C01 -> e2 E2
1000022 1000022  11 -11 0.048204 # ~C01 ~C01 -> e1 E1
1000022 1000022   6  -6 0.045455 # ~C01 ~C01 -> u3 U3
1000022 1000022   5  -5 0.038574 # ~C01 ~C01 -> d3 D3
1000022 1000022 -24  24 0.020219 # ~C01 ~C01 -> Wm Wp
1000022 1000022  25  25 0.016671 # ~C01 ~C01 -> h1 h1
```

- To calculate cross sections with MadGraph, the corresponding flag has to be set to True

```
DEFINITION[a_][IncludeMadGraph]=True;
```

In addition, for each process the location of the directory containing all MadEvent files and the process card have to be given in an array

```
DEFINITION[a_][MadGraphRuns]= {
 {"/home/USER/[MG Directory]/[Process 1]","/home/USER/[MG Directory]/[Process 1]/pro
 {"/home/USER/[MG Directory]/[Process 2]","/home/USER/[MG Directory]/[Process 2]/pro
 ...
  };
```

- In order to perform a Monte Carlo study at each point in parameter space by WHIZARD, the flag has to be set to True.

```
DEFINITION[a_][IncludeWHIZARD]=True;
```

In addition, SSP needs a list with the different input files for WHIZARD. These files are afterwards evaluated at each point in parameter space.

```
DEFINITION[a_][WHIZARDruns]={
  "~/SUSY_Frame_Test/WO_runs/Input1.sin",
  "~/SUSY_Frame_Test/WO_runs/Input2.sin"
};
```

When SSP runs WHIZARD, it creates for each point a sub-directory in the current output directory which is called

```
 WHIZARD_point_N
```

N is the number of the evaluated point. Afterwards, it copies the parameters file from SPheno to that directory and runs the different input files. All of the output of WHIZARD is kept in these directories.

- To include the calculation of decay widths or cross sections using `CalcHep`, the corresponding flag has to be set to `True`

```
DEFINITION[a_][IncludeCalcHep]=True;
```

  Afterwards, a list has to be given which contains for each process the following information

  - Directory of `CalcHep` binary (by default called `n_calchep`
  - Syntax to perform the calculations. This is normally a line of the form

    ```
    > ./n_calchep -blind "[{]][{{[]]"
    ```

  - Name of the output file. The file is normally called `prt_1` and it is necessary that the last line of that file has the form

    ```
     < >      9.5390E-04        9.45E-04     50000         1
    ```

    `SSP` takes the first number after `< >` as final result and saves it.

  Using these conventions, the content of `DEFINITION[name][CalcHepRuns]` might look like

```
DEFINITION[a_][CalcHepRuns]={
{"[CHep Processes Directory]/ee-h1h1_MSSM",
   "./n_calchep -blind \"[[[[[[[[[[{}]]{[[[[[{}}\"", "prt_1"},
{"[CHep Processes Directory]/ee-CC_MSSM",
   "./n_calchep -blind \"[[[[[[[[[[{}]]{[[[[[{}}\"", "prt_1"}
};
```

  The results obtained by `CalcHep` in that way are added to the Block `CalcHep`. The entry in that block corresponds to the position in `DEFINITION[name][CalcHepRuns]`.

- To include `Vevacious`, the flag

```
DEFINITION[a_][IncludeVevacious]=True;
```

  has to be set. The output of `Vevacious` is added to the spectrum file in the block

```
BLOCK VEVACIOUSRESULTS #
```

  The main important entries are

```
BLOCK VEVACIOUSRESULTS #
    0    0    ...    # stability of input
    0    1    ...    # tunneling time in Universe ages
                         / calculation type
    0    2    ...    # estimated best tunneling temperature
                         / survival probability at this temperature
```

Setting time constraints for the different programs It can happen that a program doesn't terminate for some points. This stops the scan until the corresponding task of the program is killed by user interaction. Therefore, the open source program `timeout` is delivered with `SSP`. Just the makefile has been changed so that the compiled executable is called `timeoutSSP` in order not to create a conflict with an already existing installation of `timeout`. Using `timeout` it is possible to start a process and give as argument the maximal number of seconds the process is allowed to run. Exactly this will `SSP` do if the following variables receive values in the input file:

- `MaximalTimeSPheno`
- `MaximalTimeCalcHep`
- `MaximalTimeWHIZARD`
- `MaximalTimeMO`

Note, the number of seconds must be given as an integer. Also `timeout` which is located in the package directory has to be extracted and compiled by the user before it can be used.

2.2.1 Scans

First, all appearing blocks in the SLHA input file for `SPheno` are given

```
DEFINITION[a_][Blocks]={MODSEL,SMINPUTS,MINPAR,SPhenoInput};
```

Afterwards, for each block the entries are defined. The general syntax is

```
DEFINITION[SCAN][BLOCK]={
  {{nr}, {value}},  ...  };
```

Of course, also matrix entries like

```
DEFINITION[SCAN][BLOCK]={
  {{nr1,nr2}, {value}},  ...  };
```

are possible.
`value` can be

- **A fixed value**, e.g.

  ```
  Value->500.
  ```

- **An interval**, e.g.

  ```
  Min->0,Max->1000, Steps->10,Distribution->LINEAR
  ```

  The possible distributions are `LINEAR, LOG` and `RANDOM`

- **A relation to other parameters**, e.g.

  ```
  Value->2*MINPAR[2]
  ```

- **An fit value to fulfill a constrain**, e.g.

```
Value->TBFIT
```

see sec. 2.2.2 for more information.

- **A free variable for a contour plot**, e.g.

```
Value->CONTOURSCANPARAMTER[1]
```

see sec. 2.2.3 for more information.

Example To perform a grid scan with 2500 points in the $(m_0, M_{1/2})$ plane with fixed values for $\tan\beta$, A_0 and signμ, the input reads

```
DEFINITION[ScanM0M12][MINPAR]={
  {{1}, {Min->0,Max->1000, Steps->50,Distribution->LINEAR}},
  {{2}, {Min->0,Max->1000, Steps->50,Distribution->LINEAR}},
  {{3}, {Value->10}},
  {{4}, {Value->1}},
  {{5}, {Value->0}}  };
```

Scatter plot If you want to have a scatter plot where all parameters are independently and randomly varied for the given ranges and a given total number of points, set

```
DEFINITION[name][ScatterPlot] = True;
DEFINITION[name][ScatterPoints] = Number;
```

In that way, only the entries `Value` respectively `Min/Max` of each parameter are taken into account, but the number of steps as well as the chosen distribution is ignored.
For large scatter plots the number of valid points can be huge. Hence, it might be necessary to define a condition for those points which should be saved. This is possible by setting the corresponding variable `UseCheckSavingPoints` to `True` and define a Mathematica function `CheckSavingPoints` which uses as argument the information from the spectrum file.

```
DEFINITION[a_][UseCheckSavingPoints]=True;
DEFINITION[a_][CheckSavingPoints][spc_]:=
         If[(MASS[25]/.spc)<128 &&(MASS[25]/.spc)>122
            && (DARKMATTER[1]/.spc)<0.2,True,False];
```

This function will check that the light Higgs masses is between 122 and 128 GeV and the relic density Ωh^2 calculated by `micrOMEGAs` is smaller than 0.2.

Including old runs Normally, `SSP` deletes existing files in the output directory in order not to mix up scans and results. However, if you want to keep the results of the former runs and to add the new scan points, use

```
DEFINITION[name][AttendPoints] = True;
```

This might especially for scatter plots be a good idea to increase the data set.

2.2.2 Fit to constraints

It is possible to define constraints for each scan separately. The general syntax is

```
DEFINITION[SCAN][FitValues]={
 {Parameter,Value,Error},...  };
```

So, it is demanded that the value of `Parameter` lies at each point in the range [`Value - Error`,`Value + Error`]. Of course, it is also necessary to define the free parameters. That's done by

```
DEFINITION[FITHIGGS][FreeParameters]={
 {Parameter,{Min,Max}}  };
```

In that way, `Parameter` is varied in the range [`Min`,`Max`] to fulfill the constraint.
Some examples are in order here. With

```
DEFINITION[FITHIGGS][FitValues]={
 {MASS[25],115,0.1}  };
```

the Higgs mass at each point is fitted to be 115 ± 0.1 GeV. The parameters which are adjusted to reach that aim are set by

```
DEFINITION[FITHIGGS][FreeParameters]={
 {TANBFIT,{5,15}},
 {A0FIT,{0,100}} };

DEFINITION[FITHIGGS][MINPAR]={
  {{1}, {Min->0.,Max->1000, Steps->10,Distribution->LINEAR}},
  {{2}, {Value->500.}},
  {{3}, {Value->TANBFIT}},
  {{4}, {Value->1.}},
  {{5}, {Value->A0FIT}}   };
```

That means, that $\tan\beta$ can be varied in the range $[0, 15]$ and $A_0 \in [0, 100]$ GeV.
Also functions of output parameters and relations between output parameters can be given as constraints e.g.

```
DEFINITION[FITHIGGSINOCOMPONENT][FitValues]={
 {ZN[1,4]^2+ZN[1,3]^2,1,0.5}  };

DEFINITION[FITcoannihilation][FitValues]={
 {MASS[1000022],MASS[1000011],1}  };
```

In the first example, the Higgsino component of the lightest neutralino must be larger than 0.5 and in the second one, the mass difference between the lightest neutralino and lightest stau must be smaller than 1 GeV.
Similarly, it is also possible to adjust the range of the free parameter dynamically, e.g.

```
DEFINITION[FITcoannihilation][FreeParameters]={
 {M12FIT,{0.5*MINPAR[1],2*MINPAR[1]}} };
```

All fits are performed by using the function `NMinimize` of `Mathematica`. Therefore, it is also possible to change the options of that function, e.g. to choose another fit algorithm.

```
DEFINITION[FITHIGGS][FitOptions]={Method->"NelderMead"};
```

Since most fit algorithm cannot be used with a finite parameter range $[a, b]$, `SSP` transforms the boundaries to problem with infinite boundaries. That's done by the stretching

$$P_{ext} = a + \frac{b-a}{2} + \frac{\text{atan}(P_{int})}{\pi}(b-a) \tag{2.1}$$

with $P_{int} \in [-\infty, +\infty]$ and $P_{ext} \in [a, b]$. For more details of `NMinimize` and the different options, we refer to the `Mathematica` manual.
To get an insight what the fit routine does, it is possible to set `ShowAllSteps` to `True` before starting the run, i.e.

```
In[1]: <<"SSP/SSP.m"
In[2]: ShowAllSteps = True;
In[3]: Start["Input file"]
```

In that way, for each step the following information is printed:

- The (unstrechted) input points $\in [-\infty, +\infty]$
- The stretched points ($\in [a, b]$)
- The calculated value of the constraints
- The demanded values of the constraints
- The calculated χ^2 value

2.2.3 2D parameter space sampling

A common problem is the sampling of a two dimensional parameter space, e.g. when checking the dark matter relic density in the $(m_0, M_{1/2})$. In many cases, a grid or random scan might not be the best choice because distinct areas should be sampled more precisely than others. For that reason it might be interesting to use the `ContourPlot` function of `Mathematica` which was developed exactly for such purposes. The input for using `ContourPlot` with `SSP` is

```
DEFINITION[DM][CountourScan]=
  {DARKMATTER[1],
  {CONTOURSCANPARAMTER[1],0,1500},
  {CONTOURSCANPARAMTER[2],0,1500},
   CPOptions,"DM_A0.eps"};
```

First, the z value is defined, afterwards the ranges for the x and y values are given. As options, the usually options of `ContourPlot` can be used. Finally, the name of the created plot is set. A set of options for a relic density scan might be

```
CPOptions={PlotPoints->30, PrecisionGoal->"Quality",
             ContourShading ->False, Contours->{0.09,0.10,0.11}};
```

See again the Mathematica manual for more details of the different options. Of course, it is also necessary to tell SSP which input parameter should be varied in that way. That can be done by setting the value of two parameters to CONTOURSCANPARAMTER[1] and CONTOURSCANPARAMTER[2]. The MINPAR block corresponding to a relic density scan reads

```
DEFINITION[A0][MINPAR]={
  {{1}, {Value->CONTOURSCANPARAMTER[1]}},
  {{2}, {Value->CONTOURSCANPARAMTER[2]}},
  {{3}, {Value->10}},
  {{4}, {Value->1}},
  {{5}, {Min->1,Max->1000,Steps->3,Distribution->LOG}}    };
```

In that way, contour plots for the relic density in the $(m_0, M_{1/2})$ plane are created for $A_0 = 1, 10, 1000$ GeV and $\tan\beta = 10, \text{sign}\mu > 0$.
The most important option for contour scans is the number of plot points. As example, we have created several scans in the $(m_0, M_{1/2}$ plan for a mSugra MSSM scenario with fixed$A_0 = 0$ GeV, $\tan\beta = 10$ and $\mu > 0$. We used PrecisionGoal->"Quality" and varied the PlotPoints between 5 and 50. The result is given in fig. 2.1. We give in addition in tab. 2.1 the number of valid parameter points which have been evaluated.

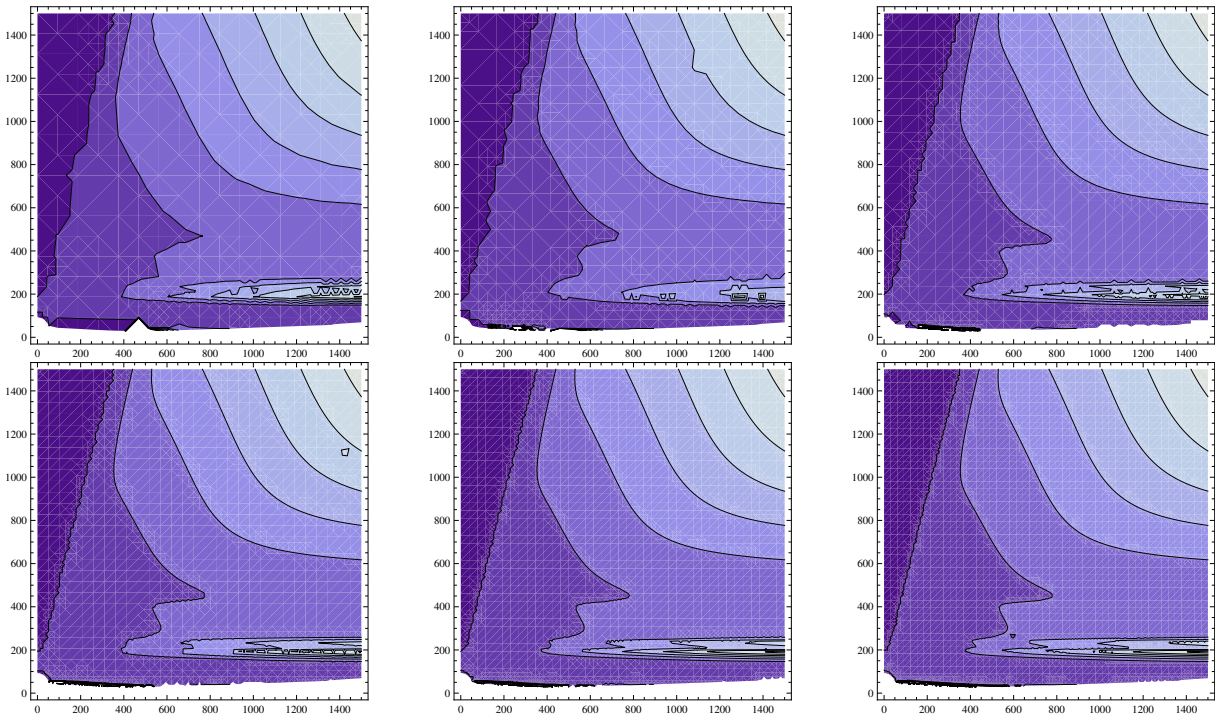


Figure 2.1: $(m_0, M_{1/2})$ plane for different values of PlotPoints. First row (from left to right): 5, 10, 20. Second row (from left to right): 30, 40, 50. The mesh gives an insight, where Mathematica has decreased the distance between the interpolation points.

PlotPoints	5	10	20	30	40	50
Calculated points	990	2077	4589	9242	13324	17416

Table 2.1: Number of evaluated parameter space points in the $(m_0, M_{1/2})$-plane for a given value of `PlotPoints`.

2.2.4 Marcov Chain Monte Carlo

`SSP` provides the basic functionality to run Marcov Chain Monte Carlo (MCMC) based on the Metropolis algorithm. The performed steps are:

1. A randomly chosen point is used as starting
2. The likelyhood of this point is calculated
3. A new point is choose with some specified propability
4. The likelyhood of the new point is calculated
5. The chain jumps with some probability function to the new point
6. steps 2-5 are iterated until some convergence criteria are fulfilled

To run a MCMC use

```
DEFINITION[a_][MCMC] = True;
```

at the beginning of the `SSP` input file.
The variable parameters including the range of the initially choose random value and a probability to get the next step is defined in the body of the input file. For instance,

```
DEFINITION[a_][MINPAR]={
 {{1}, {Min->0, Max->2500,
         NextStep->Hold[RandomReal[NormalDistribution[MINPAR[1], 50]]]},
 {{2}, {Min->0, Max->2500,
         NextStep->Hold[RandomReal[NormalDistribution[MINPAR[2], 50]]]},
 {{3}, {Min->5, Max->50,
         NextStep->Hold[RandomReal[NormalDistribution[MINPAR[3], 1]]]},
 {{4}, {Value->1}},
 {{5}, {Min->-5000, Max->5000,
         NextStep->Hold[RandomReal[NormalDistribution[MINPAR[5], 100]]]}}
  };
```

This means that the probability for the next step is given by some normal distribution arround the current point with a given variance defined for each parameter. Note, in addition the global parameters `MCMC‘Steps` (only accepted steps) and `MCMC‘AllSteps` (all steps) could be used if some heat-bath algorithm should be used.

To calcululate the likelyhood, constraints can be defined in the input file in `SSP`. The overall likelyhood is the product of all different likelyhoods. For instance, to use noraml distibutions for the light Higgs mass, the dark matter relic density and the branching ratio $b \to s\gamma$ as well as a step function for the chargino mass, the constraints might read

```
DEFINITION[a_][ConstraintsMCMC]={
 If[DARKMATTER[1]<0,0,Exp[-(0.11 - DARKMATTER[1])^2/(2*0.02^2)]],
 Exp[-(MASS[35]-126)^2/(2*2^2)],
 Exp[-(FINETUNING[99])^2/(2*60^2)],
 Exp[-(SPHENOLOWENERGY[1]*10^4-2.95)^2/(2*0.05^2)],
 1./(1 + Exp[+(120 - MASS[1000024])/5])
 };
```

The step propability is a function of the old and new likelyhood. And defined by

```
DEFINITION[a_][MCMCjumpingQ][new_, old_] =
  If[new > old, True,
   If[(new/old) > Random[], True, False]
    ];
```

This means, the new point is always accepted if the likelyhoods is larger, otherwise it is just accepted with some probability. Also here, some people might use the current number of steps (`MCMC‘Steps`, `MCMC‘AllSteps`) to simulate some temperature dependence.

Finally, criteria can be defined after which the chain should stop:

```
DEFINITION[a_][MCMCconvergenceCheck][like_] :=
  If[like> 0.9 || MCMC‘Steps > 5000, True,False];
```

Here, the chain stops if a likelyhood better than 0.9 is reached or after 5000 accepted jumps.
During the run of the MCMC `SSP` saves two different files containing the spectra: `SpectrumFiles.spc` with all calculated points and `SpectrumFiles_accepted.spc` with the accepted points. In addition, to each point a new block `MCMCINFO` is added. It contains the number of accepted points, the number of total points and the calculated likelyhood. For instance:

```
Block MCMCMINFO #
1 14 # accepted point
2 29 # all points
3 1.27889061901735e-27024 #  Likelyhood
```

2.3 Plots

It is also possible to give a list of plots which should be automatically created by `SSP` after the scans are finished

```
DEFINITION[SCAN][Plots]={
{type, {x,y,(z)}, style, filename},
...};
```

`SSP` recognizes `P2D` for two-dimensional and `P3D` for three-dimensional plots. For example, we want to plot the dependence of the Higgs masses on m_0, the gluino mass in the $(\tan\beta, A_0)$ plane and the Higgsino fraction of the lightest neutralino depending on μ. The input is

```
DEFINITION[ScanM0M12][Plots]={
  {P2D, {MINPAR[1],{MASS[25],MASS[35],MASS[36]}},Style1,
                                                  "m0_Higgs.eps"},
  {P3D, {MINPAR[2], MINPAR[3], MASS[1000021]}, Style2,
                                                "Glu_m0_m12.eps"},
  {P2D, {HMIX[2],{Log[10,ZN[1,3]^2+ZN[1,4]^2] }, Style2,
                                          "mu_HiggsionoComp.eps"}};

BasicStyle= {Frame->True, Axes->False,PlotJoined->True};
Style1 = Map[Join[BasicStyle,{PlotStyle->#}]&,{Red,Green,Blue}];
Style1 = Map[Join[BasicStyle,{PlotStyle->#}]&,{Blue}];
```

Using LaTeXfor labels It is also possible to use LaTeXcommands in the plot labels. For this purpose, a new command `UseLaTeX` is used, e.g.

```
FrameLabels->{UseLaTeX["$m_0$"], UseLaTeX["$m_{\\tilde{\\chi}^0_i}$"]
```

When this is done, temporarily used strings in the plots are replaced in a second step by the LaTeXtext using the public script `fragmaster` which is based on `psfrag`. The script `fragmaster.sh` is included in the package directory of `SSP`. It might be necessary to make it executable via

```
 > chmod -x fragmaster.sh
```

before it can be used by `SSP`.

2.4 Using results of already performed scans

Of course, it is also possible to use the results of older scans done by `SSP`. A very simple way to do that is to use the function `MakeSubNum` which translates the content of a SLHA spectrum file to `Mathematica` replacement rules. For example,

```
In[1]: <<SSP.m
In[2]: data=Get[Filename];
In[3]: ReplacementRules = MakeSubNum/@ data;
```

While the content of `Filename` for each point in parameter space was of the form

```
{...,{MASS,{25,115.2},{35,360.2},..},};
```

`ReplacementRules` is now of the form

```
{...,{MASS[25] -> 115.2, MASS[35] -> 360.2,..},};
```

So, it is quite easy to create list plots by

```
ListPlot[Table[{MINPAR[1],MASS[25]} /.
ReplacementRules[[k]],{k,1,Length[ReplacmenteRules]}],Options];
```

Chapter 3

Examples

We give here an overview about the four examples for making scans and plots in the MSSM which are already included in `SSP`.

3.1 Settings file

The content of `DefaultSettings.m` should look like

```
DEFAULT[SPhenoDirectory] =
"~/Documents/SUSY-Framework/SPheno3.1.3/bin/";
DEFAULT[SPhenoBin] = "./SPhenoMSSM";
DEFAULT[SPhenoInputFile] = "LesHouches.in.MSSM";
DEFAULT[SPhenoSpectrumFile] = "SPheno.spc.MSSM";

DEFAULT[MicroOmegasDirectory] =
"~/Documents/SUSY-Framework/micromegas_2.4.1/MSSM/";
DEFAULT[MicroOmegasInputFile] = "SPheno.spc.MSSM";
DEFAULT[MicroOmegasBin] = "CalcOmega";
DEFAULT[MicroOmegasOutputFile] = "omg.out";
DEFAULT[DarkMatterCandidate] = 1000022;

DEFAULT[WHIZARDexecution] =
        "~/Documents/SUSY-Framework/whizard/bin/whizard";
DEFAULT[WHIZARDparFile] = "WHIZARD.par.MSSM";
```

3.2 $m_0/M_{1/2}$ grid scan

In that example, we perform three different scans:

- `ScanM0`: variation of m_0 between 0 and 1000 GeV
- `ScanM12`: variation of $M_{1/2}$ between 0 and 1000 GeV
- `ScanM0M12`: variation of both, m_0 and $M_{1/2}$, between 0 and 1000 GeV

The first step is to name the three different scans

```
RunScans = {ScanM0,ScanM12,ScanM0M12};
```

Afterwards, all necessary information for the LesHouches input has to be defined: all appearing block names as well as the numerical values for each parameter. Since most blocks have the same content for all three scans, we can use a pattern as block name (e.g. `a_`) and just have to define them once.

```
DEFINITION[a_][Blocks]={MODSEL,SMINPUTS,MINPAR,SPhenoInput};

DEFINITION[a_][MODSEL]={
  {{1},{Value->1}},
  {{6},{Value->1}}
};

DEFINITION[a_][SMINPUTS]={
  {{2},{Value->1.166390*10^-5}},
  {{3},{Value->0.1172}},
  {{4},{Value->91.18760}},
  {{5},{Value->4.2}},
  {{6},{Value->172.9}},
  {{7},{Value->1.777}}
};

DEFINITION[ScanM0][MINPAR]={
  {{1}, {Min->0,Max->1000, Steps->10,Distribution->LINEAR}},
  {{2}, {Value->500}},
  {{3}, {Value->10}},
  {{4}, {Value->1}},
  {{5}, {Value->0}}
  };

DEFINITION[ScanM12][MINPAR]={
  {{1}, {Value->2*MINPAR[2]}},
  {{2}, {Min->0,Max->1000, Steps->10,Distribution->LINEAR}},
  {{3}, {Value->10}},
  {{4}, {Value->1}},
  {{5}, {Value->0}}
  };

DEFINITION[ScanM0M12][MINPAR]={
  {{1}, {Min->0,Max->1000, Steps->10,Distribution->LINEAR}},
  {{2}, {Min->0,Max->1000, Steps->10,Distribution->LINEAR}},
  {{3}, {Value->10}},
  {{4}, {Value->1}},
  {{5}, {Value->0}}
  };


DEFINITION[a_][SPhenoInput]={
```

```
  {{1},{Value->-1}},
  {{2},{Value->1}},
  {{11},{Value->0}},
  {{12},{Value->12}},
  {{21},{Value->0}}
};
```

Finally, we can define what should be plotted. Here, we demand the following plots

- For `ScanM0`:
  - The dependence of all three Higgs masses on m_0
  - The dependence of all six slepton masses on m_0
- For `ScanM12`:
  - The dependence of all three Higgs masses on $M_{1/2}$
  - The dependence of all four neutralino masses on $M_{1/2}$
- For `ScanM0M12`:
  - The variation of all three neutral Higgs masses depending on m_0 caused by the different values of $M_{1/2}$
  - The variation of all six selectron masses depending on m_0 caused by the different values of $M_{1/2}$
  - The variation of all four neutralino masses depending on $M_{1/2}$ caused by the different values of m_0
  - The variation of all three neutral Higgs masses depending on $M_{1/2}$ caused by the different values of m_0
  - Contour plot of the light Higgs mass in the $(m_0, M_{1/2})$-plane
  - Contour plot of the heavy Higgs mass in the $(m_0, M_{1/2})$-plane

```
DEFINITION[ScanM0][Plots]={
  {P2D, {MINPAR[1],{MASS[25],MASS[35],MASS[36]}},Style1,"m0_Higgs.eps"},
  {P2D, {MINPAR[1],{MASS[1000011],MASS[2000011],MASS[1000013],
           MASS[2000013],MASS[1000015 ],MASS[2000015]}},
           Style2,"m0_Selectrons.eps"}
};

DEFINITION[ScanM12][Plots]={
  {P2D, {MINPAR[2],{MASS[25],MASS[35],MASS[36]}},Style3,"m12_Higgs.eps"},
  {P2D, {MINPAR[2],{MASS[1000022],MASS[1000023],MASS[1000034],
           MASS[1000035]}}, Style4, "m12_Neutralinos.eps"}
};

DEFINITION[ScanM0M12][Plots]={
  {P2D, {MINPAR[1],{MASS[25],MASS[35],MASS[36]}},
            Style1,"m0_Higgs_different_m12.eps"},
```

```
  {P2D, {MINPAR[1],{MASS[1000011],MASS[2000011],MASS[1000013],
            MASS[2000013],MASS[1000015],MASS[2000015]}},
            Style2, "m0_Selectrons_different_m12.eps"},
   {P2D, {MINPAR[2],{MASS[25],MASS[35],MASS[36]}},
            Style3,"m12_Higgs_different_m0.eps"},
  {P2D, {MINPAR[2],{MASS[1000022],MASS[1000023],
            MASS[1000034],MASS[1000035]}},
            Style4, "m12_Neutralinos_different_m0.eps"},
  {P3D, {MINPAR[1],MINPAR[2],MASS[25]},Style5,"m0_m12_Mass25.eps"},
  {P3D, {MINPAR[1],MINPAR[2],MASS[35]},Style5,"m0_m12_Mass35.eps"}
};
```

Some cosmetic, namely the options for the different plots which can be set as follows:

```
BasicStyle= {Frame->True, Axes->False,
      FrameLabel->{Style[Subscript["m","0"],16],Style["Masses",16]},
      PlotJoined->False, FrameTicksStyle -> Directive[Black, 14]};
Style1 = Map[Join[BasicStyle,{PlotStyle->#}]&,{Red,Green,Blue}];
Style2 = Map[Join[BasicStyle,{PlotStyle->#}]&,
      {Red,Green,Blue,Black,Orange,Pink}];

BasicStyle2= {Frame->True,Axes->False,
      FrameLabel->{Style[Subscript["M","1/2"],16],Style["Masses",16]},
      PlotJoined->False, FrameTicksStyle -> Directive[Black, 14]};
Style3 = Map[Join[BasicStyle2,{PlotStyle->#}]&,{Red,Green,Blue}];
Style4 = Map[Join[BasicStyle2,{PlotStyle->#}]&,{Red,Green,Blue,Black}];

Style5= {Frame->True,Axes->False,
      FrameLabel->{Style[Subscript["m","0"],16],
      Style[Subscript["M","1/2"],16]},
      FrameTicksStyle -> Directive[Black, 14],ContourLabels->True};
```

Results

- `ScanM0`

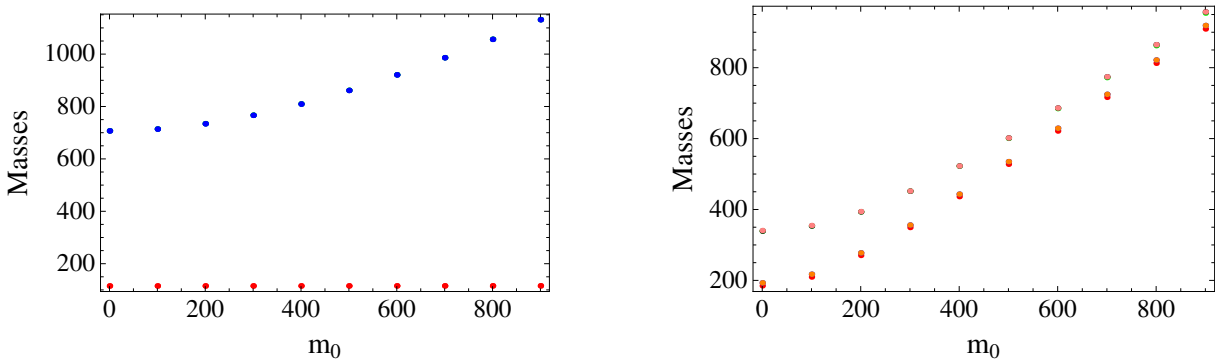


Figure 3.1: Left: mass of the lightest Higgs vs. m_0. Right: mass of the charged sleptons vs. m_0

- `ScanM12`

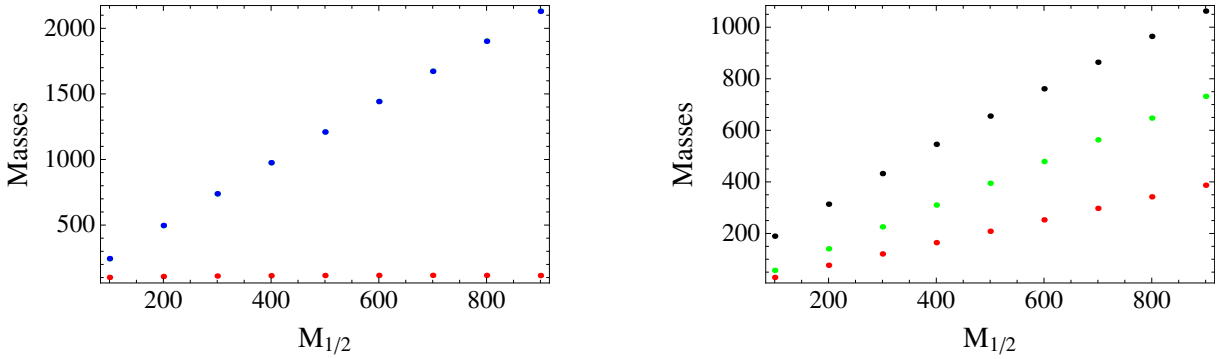


Figure 3.2: Left: mass of the lightest Higgs vs. $M_{1/2}$. Right: mass of the neutralinos vs. $M_{1/2}$

- `ScanM0M12`

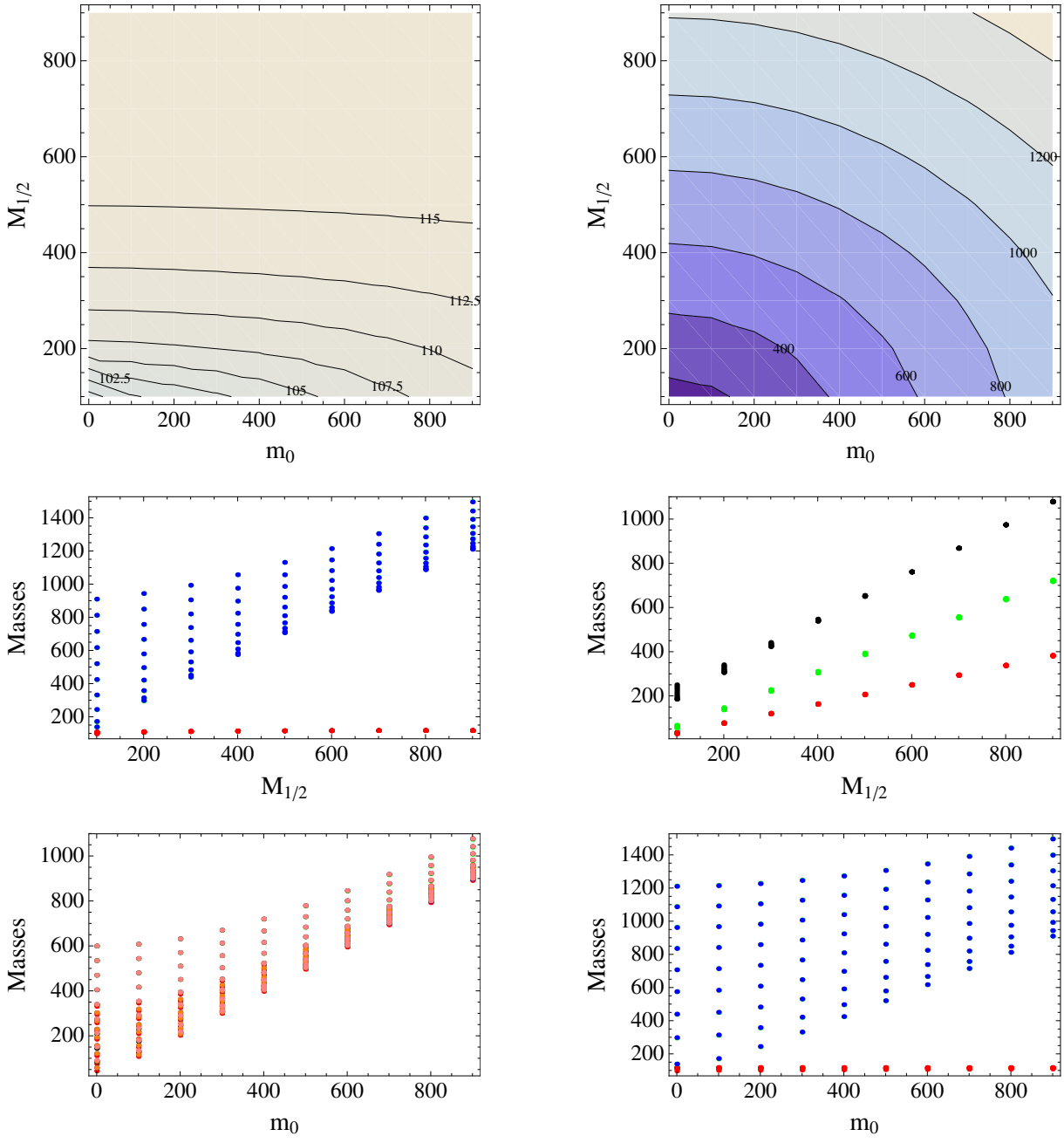


Figure 3.3: From top to bottom, left to right: mass of the light Higgs in the $(m_0, M_{1/2})$-plane, mass of the heavy Higgs in the $(m_0, M_{1/2})$-plane, masses of the scalar Higgs fields depending on $M_{1/2}$ for different values of m_0, masses of the neutralinos depending on $M_{1/2}$ for different values of m_0, masses of the heavy Higgs in the $(m_0, M_{1/2})$, masses of the scalar Higgs fields depending on m_0 for different values of $M_{1/2}$,masses of the charged sleptons depending on m_0 for different values of $M_{1/2}$.

3.3 Sampling the Dark Matter relic density in the (m_0,$M_{1/2}$)-plane for different values of A_0, $\tan\beta$ and m_{top}

In that example, we perform scans of the relic density in the (m_0,$M_{1/2}$)-plane using the `Mathematica` function `ContourPlot`. That's done for two different values of A_0, $\tan\beta$ as well as m_{top}. Again, the first step is to define names for all three scans.

```
RunScans = {A0,TANB,MTOP};
```

We use the option to include the calculation of the relic density in all scans

```
DEFINITION[a_][IncludeDarkMatter]=True;
```

We set for each run the options for the contour scan. We choose m_0 and $M_{1/2}$ between 0 and 1500 GeV each time. Note, a contour scan is performed for each parameter space point: since we have two values for A_0, $\tan\beta$ and m_{top}, respectively, we will end up with 6 contour plots.

```
DEFINITION[a_][MakeCountourScan]=True;

DEFINITION[A0][CountourScan]={DARKMATTER[1],
  {CONTOURSCANPARAMTER[1],0,1500},{CONTOURSCANPARAMTER[2],0,1500},
  CStyle,"DM_A0.eps"};
DEFINITION[TANB][CountourScan]={DARKMATTER[1],
  {CONTOURSCANPARAMTER[1],0,1500},{CONTOURSCANPARAMTER[2],0,1500},
  CStyle,"DM_TanB.eps"};
DEFINITION[MTOP][CountourScan]={DARKMATTER[1],
  {CONTOURSCANPARAMTER[1],0,1500},{CONTOURSCANPARAMTER[2],0,1500},
  CStyle,"DM_mTop.eps"};

CStyle1={PlotPoints->30, PrecisionGoal->"Quality"};
```

Defining the other parameter ranges and the options of the LesHouches input is very similar to the first example.

```
DEFINITION[a_][Blocks]={MODSEL,SMINPUTS,MINPAR,SPhenoInput};

DEFINITION[a_][MODSEL]={
  {{1},{Value->1}},
  {{6},{Value->1}}
};

DEFINITION[A0][SMINPUTS]={
  {{2},{Value->1.166390*10^-5}},
  {{3},{Value->0.1172}},
  {{4},{Value->91.18760}},
  {{5},{Value->4.2}},
  {{6},{Value->172.9}},
  {{7},{Value->1.777}}
};
```

```
DEFINITION[TANB][SMINPUTS]=DEFINITION[A0][SMINPUTS];

DEFINITION[MTOP][SMINPUTS]={
  {{2},{Value->1.166390*10^-5}},
  {{3},{Value->0.1172}},
  {{4},{Value->91.18760}},
  {{5},{Value->4.2}},
  {{6},{Min->170,Max->175,Steps->2,Distribution->LINEAR}},
  {{7},{Value->1.777}}
};

DEFINITION[A0][MINPAR]={
  {{1}, {Value->CONTOURSCANPARAMTER[1]}},
  {{2}, {Value->CONTOURSCANPARAMTER[2]}},
  {{3}, {Value->10}},
  {{4}, {Value->1}},
  {{5}, {Min->1,Max->100,Steps->2,Distribution->LOG}}
  };

DEFINITION[TANB][MINPAR]={
  {{1}, {Value->CONTOURSCANPARAMTER[1]}},
  {{2}, {Value->CONTOURSCANPARAMTER[2]}},
  {{3}, {Min->5,Max->10,Steps->2,Distribution->LINEAR}},
  {{4}, {Value->1}},
  {{5}, {Value->0}}
  };


DEFINITION[MTOP][MINPAR]={
  {{1}, {Value->CONTOURSCANPARAMTER[1]}},
  {{2}, {Value->CONTOURSCANPARAMTER[2]}},
  {{3}, {Value->10}},
  {{4}, {Value->1}},
  {{5}, {Value->0}}
  };



DEFINITION[a_][SPhenoInput]={
  {{1},{Value->-1}},
  {{2},{Value->1}},
  {{11},{Value->0}},
  {{12},{Value->12}},
  {{21},{Value->0}}
};
```

Since the contour plots are created automatically and we are not interested in other plots for that example, we are already finished.

Results

Example 2

In that example, we want to get an impression how the relic density in the $(m_0, M_{1/2})$-plane shifts, when varying $A_0, \tan\beta$ or m_{top}. Therefore, we did six contour scans using a small number of `PlotPoints` which are sufficient for that purpose.

- `A0`

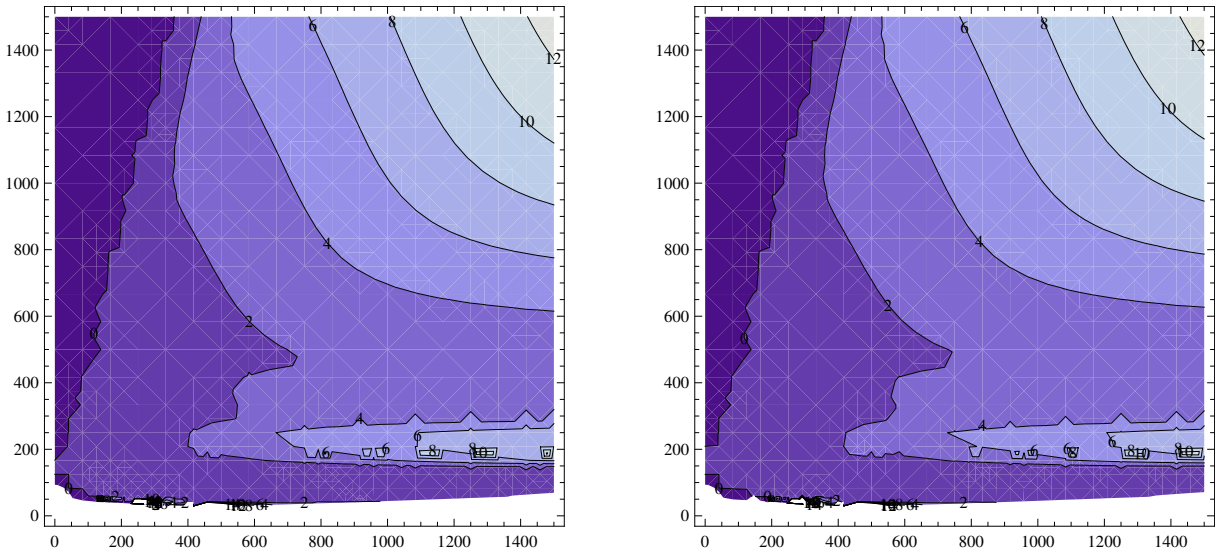


Figure 3.4: Relic density in the $(m_0, M_{1/2})$-plane for two different values of A_0: 0 GeV (left) and 100 GeV (right)

- `mTop`

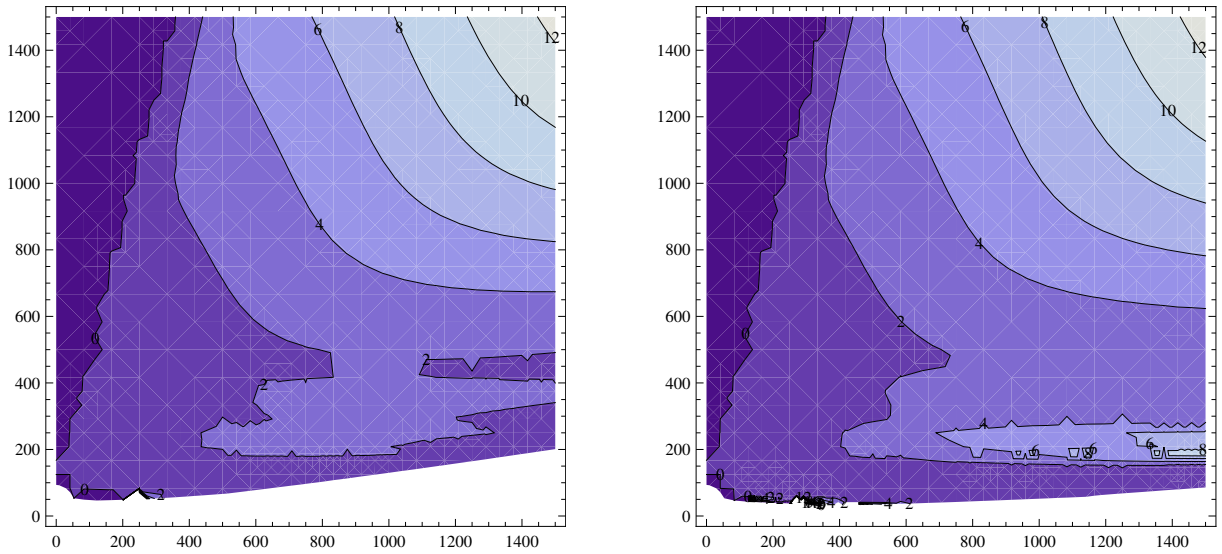


Figure 3.5: Relic density in the $(m_0, M_{1/2})$-plane for two different values of m_{top}: 170 GeV (left) and 175 GeV (right)

- `TanB`

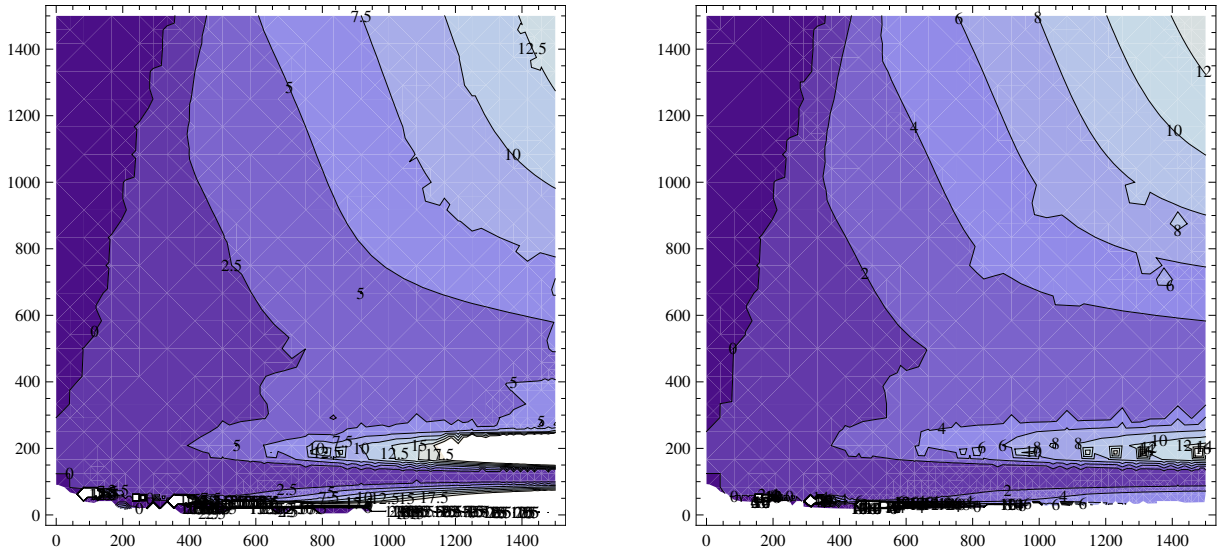


Figure 3.6: Relic density in the $(m_0, M_{1/2})$-plane for two different values of $\tan_\beta$: 5 (left) and 10 (right)

3.4 Fitting the Higgs mass m_h and mass of the lightest neutralino

We perform to scans in that example

- `FITHIGGS`: we vary m_0 between 0 and 1000 GeV. It is used as constraint that the light Higgs mass must be 115 ± 0.1 GeV. To reach that aim, $\tan\beta$ and A_0 should be varied.
- `FITNeutralino`: we vary m_0 between 250 and 1000 GeV and $\tan\beta$ between 10 and 20. As constrained is used that the difference between the mass of the lightest neutralino and the lightest stau should be smaller than 1 GeV.

First, the names of the runs are given

```
RunScans = {FITHIGGS,FITNeutralino};
```

Afterwards the constraints and well as the option for `NMinimize` of `Mathematica` are set. For `FITHIGGS` we want to use the `NelderMead` method will we keep for `FITNeutralino` the default options of `Mathematica`. Note, that we have used in `FITNeutralino` boundaries for $M_{1/2}$ which depend on m_0. Choosing dynamically adjusted boundaries might increase the speed of convergence.

```
DEFINITION[FITHIGGS][FitOptions]={Method->"NelderMead"};
DEFINITION[FITHIGGS][FitValues]={
 {MASS[25],115,0.1}
 };
DEFINITION[FITHIGGS][FreeParameters]={
 {TANBFIT,{5,15}},
 {A0FIT,{0,100}}
};


DEFINITION[FITNeutralino][FitOptions]={};
DEFINITION[FITNeutralino][FitValues]={
 {MASS[1000022],MASS[1000011],1}
 };
DEFINITION[FITNeutralino][FreeParameters]={
 {M12FIT,{2*MINPAR[1],8*MINPAR[1]}}
};
```

Again, it remains to define the other parameter ranges

```
DEFINITION[a_][MODSEL]={
  {{1},{Value->1}},
  {{6},{Value->1}}
};

DEFINITION[a_][SMINPUTS]={
  {{2},{Value->1.166390*10^-5}},
  {{3},{Value->0.1172}},
  {{4},{Value->91.18760}},
  {{5},{Value->4.2}},
  {{6},{Value->172.9}},
  {{7},{Value->1.777}}
};

DEFINITION[FITHIGGS][MINPAR]={
```

```
  {{1}, {Min->0.,Max->1000, Steps->10,Distribution->LINEAR}},
  {{2}, {Value->500.}},
  {{3}, {Value->TANBFIT}},
  {{4}, {Value->1.}},
  {{5}, {Value->A0FIT}}
  };

DEFINITION[FITNeutralino][MINPAR]={
  {{1}, {Min->50.,Max->500, Steps->10,Distribution->LINEAR}},
  {{2}, {Value->M12FIT}},
  {{3}, {Min->10.,Max->20, Steps->10,Distribution->LINEAR}},
  {{4}, {Value->1.}},
  {{5}, {Value->0.}}
  };

DEFINITION[a_][SPhenoInput]={
  {{1},{Value->-1}},
  {{2},{Value->1}},
  {{11},{Value->0}},
  {{12},{Value->12}},
  {{21},{Value->0}}
};
```

The following information should be plotted this time:

- `FITHIGGS`
  - The lightest Higgs mass depending on m_0 in order to check, if the fit worked
  - All other Higgs masses vs. m_0
  - The fit value of $\tan\beta$ depending on m_0
  - The fit value of A_0 depending on m_0
- `FITNeutralino`
  - The difference between the lightest neutralino and stau mass depending on m_0 in order to check, if the fit worked
  - $M_{1/2}$ vs. m_0
  - Two Contours plots of the masses of the second and third neutralino in the $(m_0, M_{1/2})$ plane

At the end, we define the plot styles.

```
DEFINITION[FITHIGGS][Plots]={
  {P2D, {MINPAR[1],{MASS[25]}},Style1a,"m0_25-Higgs-Fit.eps"},
  {P2D, {MINPAR[1],{MASS[35],MASS[36],MASS[37]}},
          Style1b,"m0_35-37-Higgs-Fit.eps"},
  {P2D, {MINPAR[1],{MINPAR[3]}},Style2a,"m0_tb-Higgs-Fit.eps"},
  {P2D, {MINPAR[1],{MINPAR[5]}},Style2b,"m0_A0-Higgs-Fit.eps"}
};
```

```
DEFINITION[FITNeutralino][Plots]={
  {P2D, {MINPAR[1],{MASS[1000022]-MASS[1000011]}},
           Style3,"m0_1000022_1000011-N-Fit.eps"} ,
  {P2D, {MINPAR[1],{MINPAR[2]}},Style1,"m0_m12-N-Fit.eps"},
  {P3D, {MINPAR[1],MINPAR[3],MASS[1000023]},
           Style4,"m0_m12_1000023-N-Fit.eps"},
  {P3D, {MINPAR[1],MINPAR[3],MASS[1000025]},
           Style4,"m0_m12_1000025-N-Fit.eps"},
  {P3D, {MINPAR[1],MINPAR[3],MASS[1000035]},
           Style4,"m0_m12_1000035-N-Fit.eps"}
};


BasicStyle= {Frame->True, Axes->False,
     FrameLabel->{Style[Subscript["m","0"],16],Style["Mass",16]},
     PlotJoined->True, FrameTicksStyle -> Directive[Black, 14]};
Style1a = Map[Join[BasicStyle,{PlotStyle->#}]&,{Blue}];
Style1b = Map[Join[BasicStyle,{PlotStyle->#}]&,{Blue,Green,Red}];
Style3 = Map[Join[BasicStyle,{PlotStyle->#}]&,{Green,Blue}];

Style2a = {{Frame->True,Axes->False,
     FrameLabel->{Style[Subscript["m","0"],16],
%       Style["tan\[Beta]",16]},
     PlotJoined->True, FrameTicksStyle -> Directive[Black, 14]}}
Style2b = {{Frame->True, Axes->False,
     FrameLabel->{Style[Subscript["m","0"],16],
     Style[Subscript["A",0],16]},
     PlotJoined->True, FrameTicksStyle -> Directive[Black, 14]}}
Style4={Frame->True, Axes->False,
     FrameLabel->{Style[Subscript["m","0"],16],
     Style[Subscript["M","1/2"],16]},
     ContourLabels->True};
```

Results

- FITHIGGS

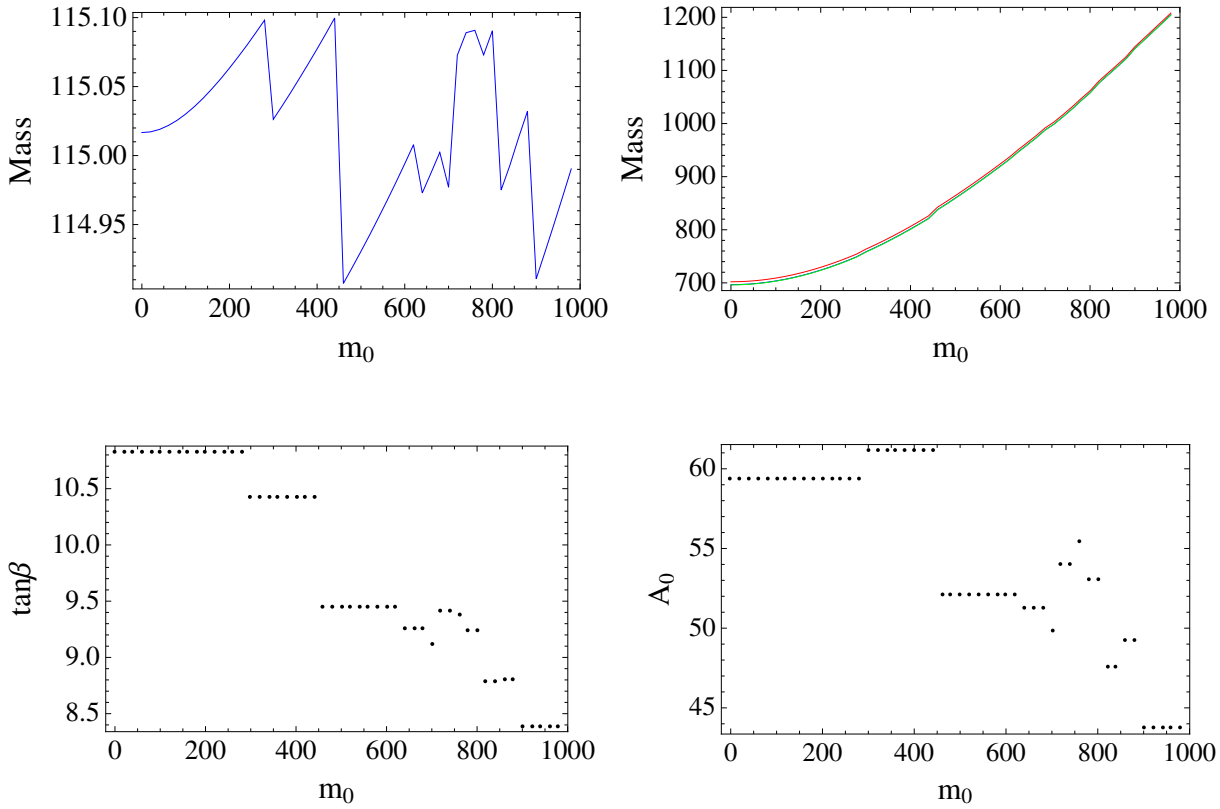


Figure 3.7: Results for a variation of m_0 including the constraint that the light Higgs mass is 115 ± 0.1 GeV. First row: light Higgs mass (left), other Higgs masses (right). Second row: fitted values of A_0 (left) and $\tan\beta$ (right)

- `FITNeutralino`

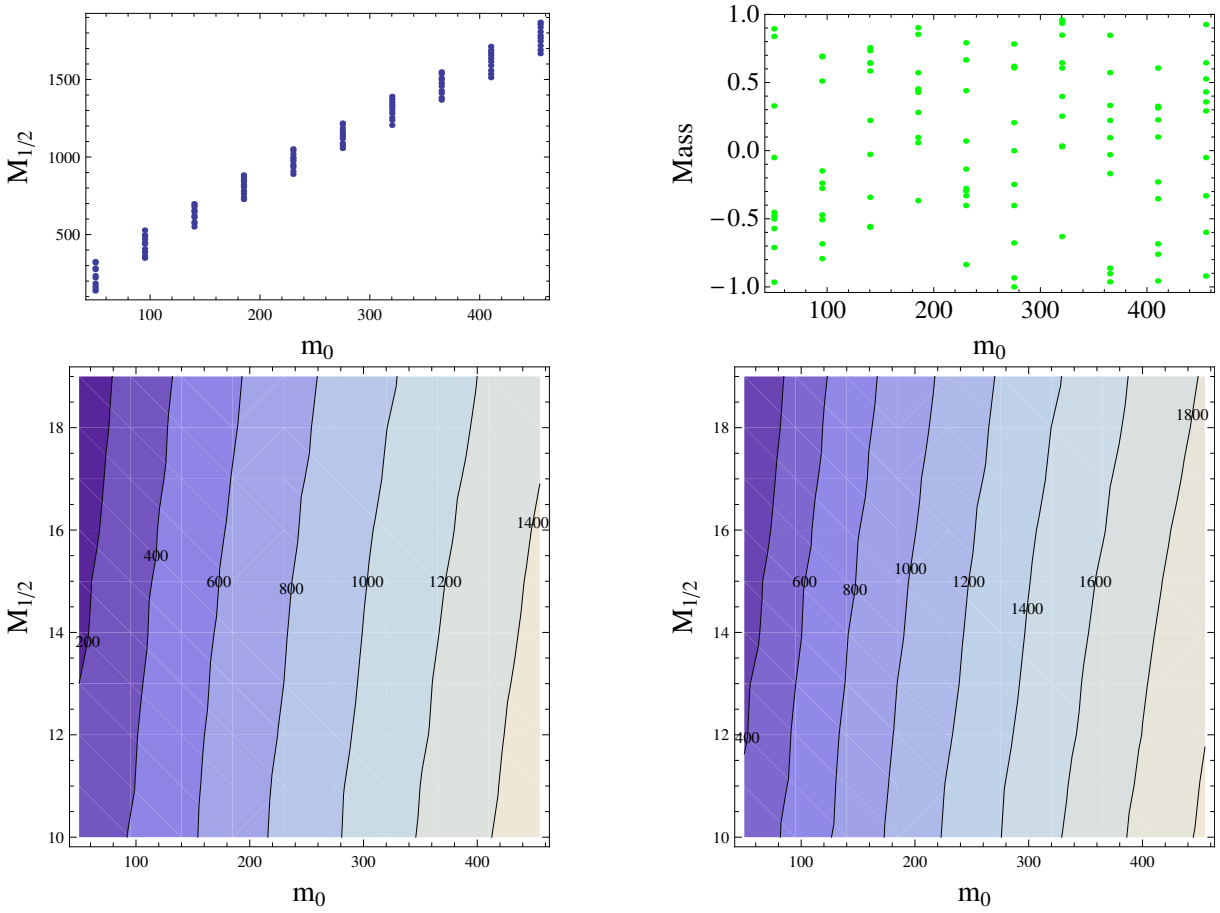


Figure 3.8: First row: m_0 vs. $M_{1/2}$ (left) and the mass difference between the lightest stau and lightest neutralino. Second row: masses of the second (left) and third (right) lightest neutralinos in the $(m_0, M_{1/2})$-plane.

3.5 Checking chargino and Higgs production for e^+e^--collisions

In that example, for vary A_0 and calculate not only the masses and branching ratios of the particle, but also the cross sections for $e^-e^+ \to hZ$ and $e^-e^+ \to \tilde{\chi}_1^+\tilde{\chi}_1^-$. For that purpose, we have created first `n_calchep` version for both processes and stored the in subdirectory of `CH-Processes`. In addition, we did a manual test run with the option `+blind` to get the correct option for calling `n_calchep` in the blind mode.
We do just one scan, which is named by

```
RunScans = {ScanA0};
```

To call the `n_calchep` version during the parameter scan, we set the directories as well as the command to execute them and the output file (here `prt_1`).

```
DEFINITION[a_][CalcHepRuns]={
{"~/CH-Processes/ee-h1Z_MSSM","./n_calchep -blind
```

```
        \"[[[[[[[[[[{}]]{[[[[[{}}\"","prt_1"},
{"~/CH-Processes/ee-CC_MSSM","./n_calchep -blind
        \"[[[[[[[[[[{}]]{[[[[[{}}\"","prt_1"}
};
```

The options for the LesHouches input are

```
DEFINITION[a_][Blocks]={MODSEL,SMINPUTS,MINPAR,SPhenoInput};

DEFINITION[a_][MODSEL]={
  {{1},{Value->1}},
  {{6},{Value->1}}
};

DEFINITION[a_][SMINPUTS]={
  {{2},{Value->1.166390*10^-5}},
  {{3},{Value->0.1172}},
  {{4},{Value->91.18760}},
  {{5},{Value->4.2}},
  {{6},{Value->172.9}},
  {{7},{Value->1.777}}
};

DEFINITION[a_][MINPAR]={
  {{1}, {Value->500}},
  {{2}, {Value->500}},
  {{3}, {Value->10}},
  {{4}, {Value->1}},
  {{5}, {Min->0,Max->1000, Steps->10,Distribution->LINEAR}}
  };

DEFINITION[a_][SPhenoInput]={
  {{1},{Value->-1}},
  {{2},{Value->1}},
  {{11},{Value->0}},
  {{12},{Value->12}},
  {{21},{Value->0}}
};
```

We plot the following

- The light and heavy Higgs mass depending on A_0
- The two cross sections calculated by `CalcHep` depending on A_0
- The selectron masses depending on A_0

```
DEFINITION[a_][Plots]={
  {P2D, {MINPAR[5],{MASS[25],MASS[35]}},Style1,"A0_Higgs.eps"},
  {P2D, {MINPAR[5],{CHep[1],CHep[2]}},Style1,"A0_CS.eps"},
```

```
  {P2D, {MINPAR[5],{MASS[1000011],MASS[2000011], MASS[1000013],
           MASS[2000013],MASS[1000015],MASS[2000015]}},
           Style2, "A0_Selectrons.eps"}
};

BasicStyle= {Frame->True, Axes->False,
              FrameLabel->{"m0[GeV]","m[GeV]"},PlotJoined->True};
BasicStyle2= {Frame->True, Axes->False,
              FrameLabel->{"m0[GeV]","m[GeV]"},PlotJoined->False};
Style1 = Map[Join[BasicStyle,{PlotStyle->#}]&,{Red,Green}];
Style2 = Map[Join[BasicStyle,{PlotStyle->#}]&,
             {Red,Green,Blue,Black,Orange,Pink}];
```

3.6 MC study of monojet production at LHC with WHIZARD

```
DEFINITION[a_][IncludeWHIZARD]=True;


DEFINITION[a_][WHIZARDruns]={
"~/Documents/SUSY_Frame_Test/W0_runs/Input_for_Example5.sin"
};


DEFINITION[a_][Blocks]={MODSEL,SMINPUTS,MINPAR,SPhenoInput};

DEFINITION[a_][MODSEL]={
  {{1},{Value->1}},
  {{6},{Value->1}}
};

DEFINITION[a_][SMINPUTS]={
  {{2},{Value->1.166390*10^-5}},
  {{3},{Value->0.1172}},
  {{4},{Value->91.18760}},
  {{5},{Value->4.2}},
  {{6},{Value->172.9}},
  {{7},{Value->1.777}}
};

DEFINITION[a_][MINPAR]={
  {{1}, {Min->500,Max->1000, Steps->3,Distribution->LINEAR}},
  {{2}, {Value->500}},
  {{3}, {Value->10}},
  {{4}, {Value->1}},
  {{5}, {Value->0}}
  };
```

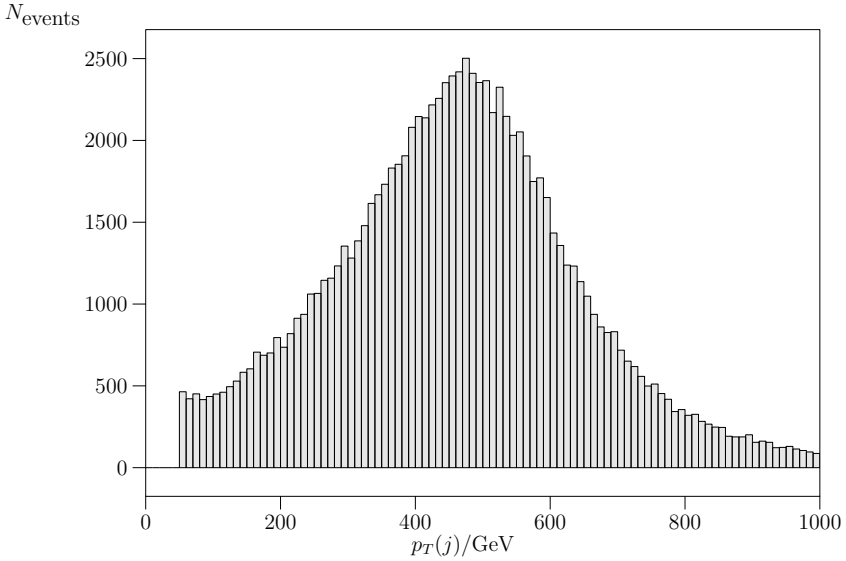
```
DEFINITION[a_][SPhenoInput]={
  {{1},{Value->-1}},
  {{2},{Value->1}},
  {{11},{Value->0}},
  {{12},{Value->12}},
  {{21},{Value->0}},
  {{78},{Value->1}}
};

DEFINITION[a_][Plots]={ };
```

```
model = mssm_sarah

include("WHIZARD.par.MSSM")

# read_slha("sps1a.slha")
alias parton = u1:u1bar:d1:d1bar:G
alias jet = parton
alias neutralino = C01

process monojet = parton, parton => jet, neutralino, neutralino

compile

sqrts = 14 TeV

beams = p, p => lhapdf
cuts = all Pt >= 50 GeV [jet]
integrate (monojet) { iterations = 5:20000 }

$description = "Monojets"
$y_label = "$N_{\textrm{events}}$"
$title = "Jet-$p_T$ in $pp\to j\tilde\chi^0\tilde\chi^0$"
$x_label = "$p_T(j)$/GeV"
histogram pt_jet (0 GeV, 1000 GeV, 10 GeV)
$title = "Jet rapidity in $pp\to j\tilde\chi^0\tilde\chi^0$"
$x_label = "$\eta(j)$"
histogram eta_jet (-5, 5, 0.1)
analysis = record pt_jet (eval Pt [extract index 1 [jet]]);
           record eta_jet (eval Eta [extract index 1 [jet]])

simulate (monojet) { n_events = 100000 }
compile_analysis { $out_file = "monojet.dat" }
```

Results

1 Jet-p_T in $pp \to j\tilde{\chi}^0\tilde{\chi}^0$

Monojets

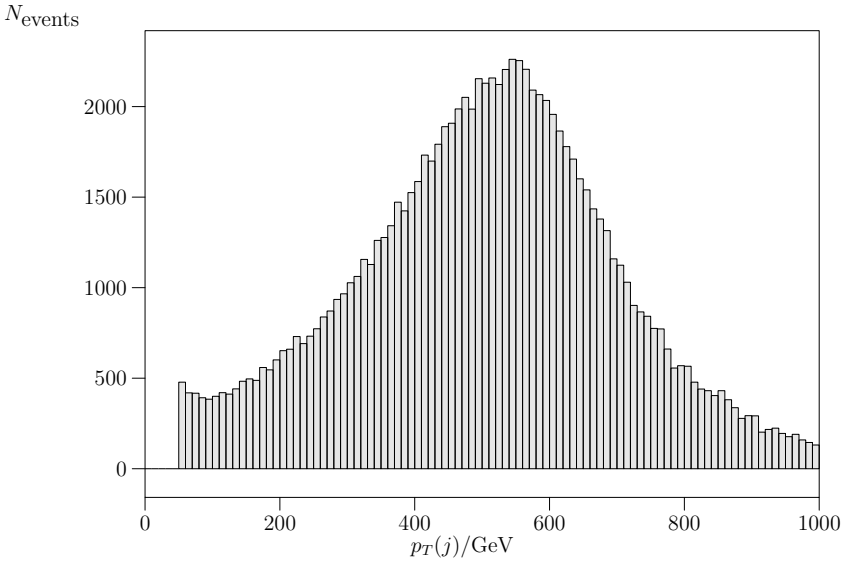


Data within bounds:
$\langle \text{Observable} \rangle = 459.7 \pm 0.57 \quad [n_{\text{entries}} = 98540]$

All data:
$\langle \text{Observable} \rangle = 470.3 \pm 0.63 \quad [n_{\text{entries}} = 100000]$

1 Jet-p_T in $pp \to j\tilde{\chi}^0\tilde{\chi}^0$

Monojets

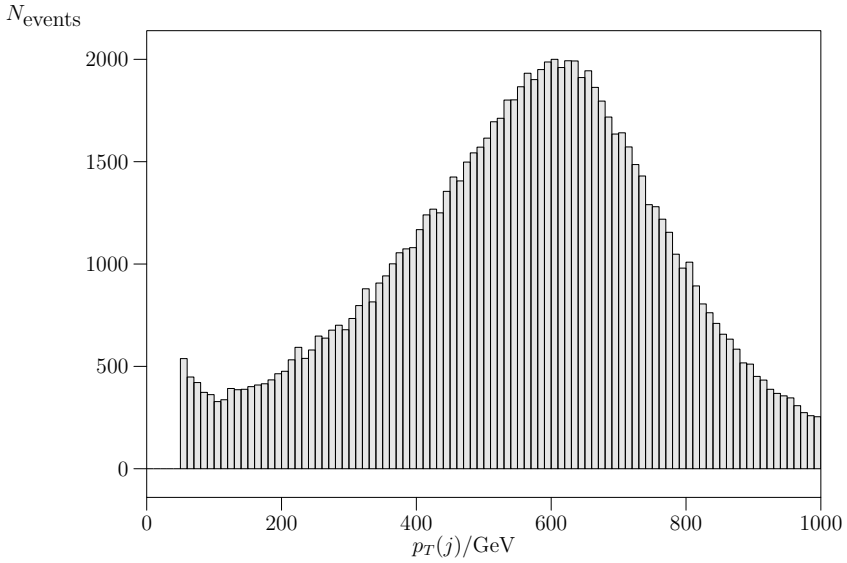


Data within bounds:
$\langle \text{Observable} \rangle = 503.4 \pm 0.61 \quad [n_{\text{entries}} = 97576]$

All data:
$\langle \text{Observable} \rangle = 520.0 \pm 0.69 \quad [n_{\text{entries}} = 100000]$

1 Jet-p_T in $pp \to j\tilde{\chi}^0\tilde{\chi}^0$

Monojets



Data within bounds:
$\langle \text{Observable} \rangle = 553.4 \pm 0.66 \quad [n_{\text{entries}} = 95859]$

All data:
$\langle \text{Observable} \rangle = 579.5 \pm 0.76 \quad [n_{\text{entries}} = 100000]$

1

3.7 Scatter plot: Higgs masses above 118 GeV

In this example 2500 points a randomly choosen for the parameter ranges $m_0 \in [0, 2500]$ GeV, $M_{1/2} \in [0, 2500]$ GeV, $\tan(\beta) \in [5, 50]$ and $A_0 \in [-5000, 5000]$ GeV. Only points are saved for which the lightest Higgs mass is larger than 118 GeV.
First, the default settings are loaded and the name of the scan is defined.

```
LoadSettings="DefaultSettings.m.MSSM";
RunScans = {ScatterMSSM};
```

As next step, the scatter plot is initialized and the condition for the Higgs mass is set.

```
(* Use the scatter plot function *)
 DEFINITION[a_][ScatterPlot] = True;
 DEFINITION[a_][ScatterPoints] = 2500;

(* save only points with Higgs masses larger than 118 GeV *)

DEFINITION[a_][UseCheckSavingPoints]=True;
DEFINITION[a_][CheckSavingPoints][spc_]:=
                   If[(MASS[25]/.spc)>118,True,False];
```

The input values for the LesHouches are set

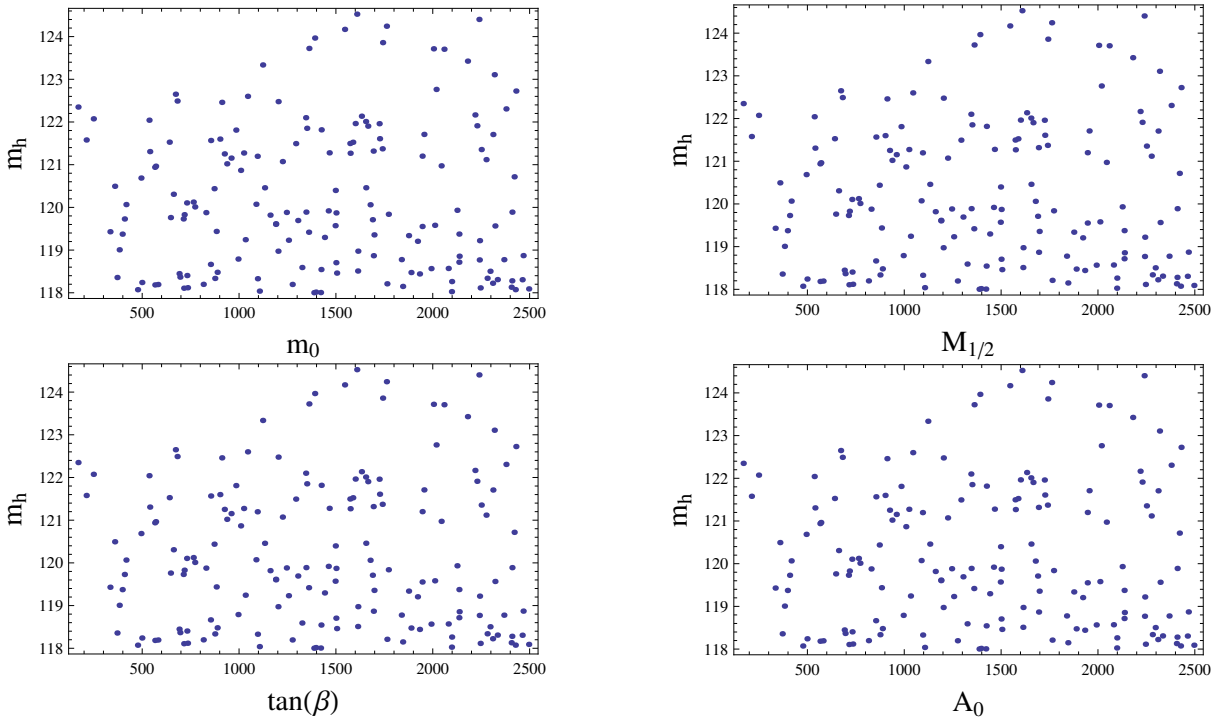
```
DEFINITION[a_][Blocks]={MODSEL,SMINPUTS,MINPAR,SPhenoInput};

DEFINITION[a_][MODSEL]={
  {{1},{Value->1}},
  {{6},{Value->1}}
};

DEFINITION[a_][SMINPUTS]={
  {{2},{Value->1.166390*10^-5}},
  {{3},{Value->0.1172}},
  {{4},{Value->91.18760}},
  {{5},{Value->4.2}},
  {{6},{Value->172.9}},
  {{7},{Value->1.777}}
};

DEFINITION[a_][MINPAR]={
  {{1}, {Min->0, Max->2500}},
  {{2}, {Min->0, Max->2500}},
  {{3}, {Min->5, Max->50}},
  {{4}, {Value->1}},
  {{5}, {Min->-5000, Max->5000}}
  };


DEFINITION[a_][SPhenoInput]={
```

```
  {{1},{Value->-1}},
  {{2},{Value->1}},
  {{11},{Value->1}},
  {{12},{Value->0}},
  {{13},{Value->0}},
  {{21},{Value->0}},
  {{75},{Value->1}},
  {{76},{Value->1}}
};
```

Finally, the light Higgs mass is plotted.

```
DEFINITION[a_][Plots]={
  {P2D, {MINPAR[1],{MASS[25]}},
      {Frame->True, Axes->False,
       FrameLabel->{Style[Subscript["m","0"],16],
          Style[Subscript["m","h"],16]}},"m0_mH.eps"},
  {P2D, {MINPAR[1],{MASS[25]}},
      {Frame->True, Axes->False,
       FrameLabel->{Style[Subscript["M","1/2"],16],
          Style[Subscript["m","h"],16]}},"m12_mH.eps"},
  {P2D, {MINPAR[1],{MASS[25]}},
      {Frame->True, Axes->False,
       FrameLabel->{Style["tan(beta)",16],
          Style[Subscript["m","h"],16]}},"tb_mH.eps"},
  {P2D, {MINPAR[1],{MASS[25]}},
      {Frame->True, Axes->False,
       FrameLabel->{Style[Subscript["A","0"],16],
          Style[Subscript["m","h"],16]}},"A0_mH.eps"}
};
```

Results


m_h
m_0
M_1/2
tan(β)
A_0
118
119
120
121
122
123
124
500
1000
1500
2000
2500

Bibliography

[1] 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.

[2] W. Porod and F. Staub. SPheno 3.1: Extensions including flavour, CP-phases and models beyond the MSSM. 2011, 1104.1573.

[3] Mauro Moretti, Thorsten Ohl, and Jurgen Reuter. O'Mega: An Optimizing matrix element generator. 2001, hep-ph/0102195.

[4] Wolfgang Kilian, Thorsten Ohl, and Jurgen Reuter. WHIZARD: Simulating Multi-Particle Processes at LHC and ILC. 2007, 0708.4233.

[5] 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.

[6] 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.

[7] A. Pukhov. Calchep 2.3: MSSM, structure functions, event generation, 1, and generation of matrix elements for other packages. 2004, hep-ph/0412191.

[8] E.E. Boos, M.N. Dubinin, V.A. Ilyin, A.E. Pukhov, and V.I. Savrin. CompHEP: Specialized package for automatic calculations of elementary particle decays and collisions. 1994, hep-ph/9503280.

[9] G. Belanger, F. Boudjema, A. Pukhov, and A. Semenov. MicrOMEGAs 2.0: A Program to calculate the relic density of dark matter in a generic model. *Comput.Phys.Commun.*, 176:367–382, 2007, hep-ph/0607059.

[10] F. Staub. SARAH. 2008, 0806.0538.

[11] Florian Staub. From Superpotential to Model Files for FeynArts and CalcHep/CompHep. *Comput.Phys.Commun.*, 181:1077–1086, 2010, 0909.2863.

[12] Florian Staub. Automatic Calculation of supersymmetric Renormalization Group Equations and Self Energies. *Comput.Phys.Commun.*, 182:808–833, 2011, 1002.0840.

[13] Florian Staub, Thorsten Ohl, Werner Porod, and Christian Speckner. A tool box for implementing supersymmetric models. 2011, 1109.5147.