DESY Summer Student Programme
Project Report

Gfitter
Automation & SUSY Extension

Kieran O'Mahony
Martin Goebel
Johannes Haller

15.09.2008

Abstract

Gfitter, a statistical framework for model testing, has been used to produce detailed results of the global electroweak fit of the Standard Model. We present a solution to fully automate the production and web publication of results from a Standard Model fit. The first steps of implementing a supersymmetric module in Gfitter (GSUSY) have also been performed and preliminary results are presented for the supersymmetric m_h-max scenario.

Contents

1 Introduction

Gfitter[1] is a collection of C++ libraries built using ROOT[2] functionality which provides comprehensive statistical capabilities for evaluating particular HEP models. The core Gfitter package provides data handling, statistical analysis and the handling of errors and theoretical uncertainties. Gfitter is designed in a modular fashion and is easily extensible. In a particular run Gfitter could perform a multi dimensional scan, perform Monte Carlo sampling or simply evaluate a fit result. Theoretical models, such as the Standard Model, are implemented as plugins to the Gfitter core.

The analysis of a particular theory can be quite thorough and therefore can require a lot of effort if it needs to be redone. Section 2 outlines a tool for automating the fitting process. Section 3 outlines the first results obtained from an initial implementation of a supersymmetry plugin for Gfitter.

2 Standard Model Fit

The results of the global electroweak Standard Model fit attained using the Gfitter tool are yet to be published. However the results are presently available via the Gfitter website[3] and are in excellent agreement with those produced by the ZFITTER[4] package.

2.1 Automation of the fit

A particular run in Gfitter is performed using an XML input file or *steering card*. Performing a full statistical analysis of the Standard Model fit requires many individual runs of the Gfitter program, each with it's own independent input file. Apart from being time consuming and laborious to rerun the whole, or even partial, analysis procedure when, for example, a new experimental value is published it is also necessary to update every input file by hand.

The automation of the fit was achieved using a single Python script. A single 'master' steering card is used as a reference point and the script takes a simple input file specifying how the master steering card needs to change to achieve the desired run. The input file can specify multiple jobs and is structured as a Python list (named *st_cards*). Listing 1 shows an example of such a file specifying a single job. Detailed usage information on the automation script is given in appendix A.1.

```
st_cards = [
 { 'out' : 'HiggsScanToy', 'spec' : {
    'Actions' : { 'ToyScan1D' : 'T:MH:Nbins=50 : Ntoys=1000' },
    'Parameters' : {
        'DeltaMW_Scale' : { 'Active' : 'F' },
        'DeltaSin2ThetaF_Scale' : { 'Active' : 'F' },
        'DeltaOMS' : { 'Active' : 'F' },
        'MH' : { 'ScanRange' : '[20:250]' }
       } } }
];
```

Listing 1: Automation job specifying a 1D Monte Carlo scan of M_H

Having a single master steering card ensures that it is a quick and painless process to rerun the analysis when a new value is published. As Gfitter is still under development it is also probable that future changes to the Gfitter source code will impact the structure of the steering card and/or require updating the automation process. However the single steering card, single automation script should ensure that this is also a relatively easy process.

After automating the generation of results from Gfitter it was also desirable to automate their publication. Therefore another script, which is completely independent from the automation script, was created to fully update the Gfitter website. An archive of previous results is also maintained. Detailed usage information is given in appendix A.2.

3 Supersymmetric module

An initial implementation of a supersymmetric module for Gfitter (GSUSY) was created as a direct wrapper around the FeynHiggs[5, 6, 7, 8] package. All results presented in this report are preliminary and were generated using FeynHiggs version 2.6.2.

3.1 A brief overview of Supersymmetry

Although the Standard Model (SM) of particle physics has proven to be incredibly successful at describing the fundamental particles and interactions it still has a large number of unsolved problems. The most famous perhaps being the issue of particle mass which is generally expected to be solved by the existence of a Higgs field and an associated Higgs boson[9, 10, 11]. From the fit of the Standard Model (c.f. Section 2) it is known that the Higgs mass should be of order 100GeV. However the Higgs mass receives enormous quantum corrections from every particle that couples to the Higgs field. This leads to an inconsistency of some 30 orders of magnitude at the expected scale of new physics and is known as the *"hierarchy problem"*.

Supersymmetry (SUSY) is a proposed extension of the Standard Model in which every SM particle must have a superpartner differing in spin by 1/2 (forming a *supermultiplet*). Therefore each SM fermion has a corresponding SUSY guage boson (collectively the *sfermions*, composed of *squarks* and *sleptons*) and each SM guage boson has a corresponding SUSY fermion (collectively the *guaginos*, e.g. the gluon's superpartner is the gluino). The Minimal Supersymmetric Standard Model (MSSM) is the minimal extension necessary to realize supersymmetry. One of the nice features of SUSY is that it has the potential to fix the hierarchy problem, due to cancellations from the all-but-identical superpartners.

For the electroweak theory to remain consistent it is necessary to not only have one Higgs superpartner but to have two Higgs doublets, therefore predicting the existence of five (three neutral and two charged) Higgs particles (plus their corresponding superpartners, the *higgsinos*). This report focuses on the lightest of these particles, m_{h^0}, which is electrically neutral. m_{h^0} is predicted to have a mass of less than about 135GeV/c^2[12].

m_h-max	
$\tan\beta$	0.4-40
M_A (GeV)	4-1000
M_{SUSY} (GeV)	1000
M_2 (GeV)	200
μ (GeV)	200
$M_{\tilde{g}}$ (GeV)	800
X_t (GeV)	$-2M_{SUSY}$

Table 1: Parameters for the m_h-max scenario.

3.2 Preliminary results for the m_h-*max* scenario

To fully describe the Higgs sector in the MSSM a handful of parameters are sufficient. The main parameters are $\tan\beta = \nu_1/\nu_2$, which is the ratio of the vacuum expectation values of the two Higgs doublets, and M_A, the mass of the neutral CP-odd Higgs particle. Other parameters which enter at the level of radiative corrections are the energy scale of SUSY breaking, M_{SUSY}, a gaugino mass at the electroweak scale, M_2, the strength of the supersymmetric Higgs mixing, μ, the gluino mass, $M_{\tilde{g}}$ and the stop mixing parameter, X_t. Table 3.2 shows the particular values used in this study, which specify the so-called m_h-max scenario. m_h-max has the nice property of maximising the upper limit on m_{h^0} and thus has a large theoretically accessible parameter space.

The supersymmetric module for Gfitter (GSUSY) was implemented by incorporating the FeynHiggs package as a Gfitter plugin. FeynHiggs provides extensive one-loop and two-loop calculations and predictions of the Higgs masses, their branching ratios, as well as a value for the anomalous magnetic moment of the muon, $(g-2)_\mu$, and a SUSY prediction of the W mass, among others.

Preliminary results from LEP[13] have already extensively constrained the parameter space for the m_h-max scenario being considered here (referred to as m_h-max*(b)* in the cited paper). Using the individual constraints from $(g-2)_\mu$, the branching ratio $B \to X_s\gamma$ and M_W figure 1 shows the results of several scans in $\tan\beta$, m_{h^0} parameter space. It is evident that m_{h^0} is significantly constrained by these parameters, most notably by $B \to X_s\gamma$ and $(g-2)_\mu$. Figure 2 shows the 68%, 95% and 99% confidence level regions for a full fit combining the results of all parameters. When taken in combination with the LEP results a large part of the SUSY parameter space for the m_h-max scenario is excluded at 95% confidence level.

However, just prior to this report a programming error was found in the GSUSY code, therefore these results should be taken as a 'proof of concept' for a GSUSY module rather than providing any physical result.

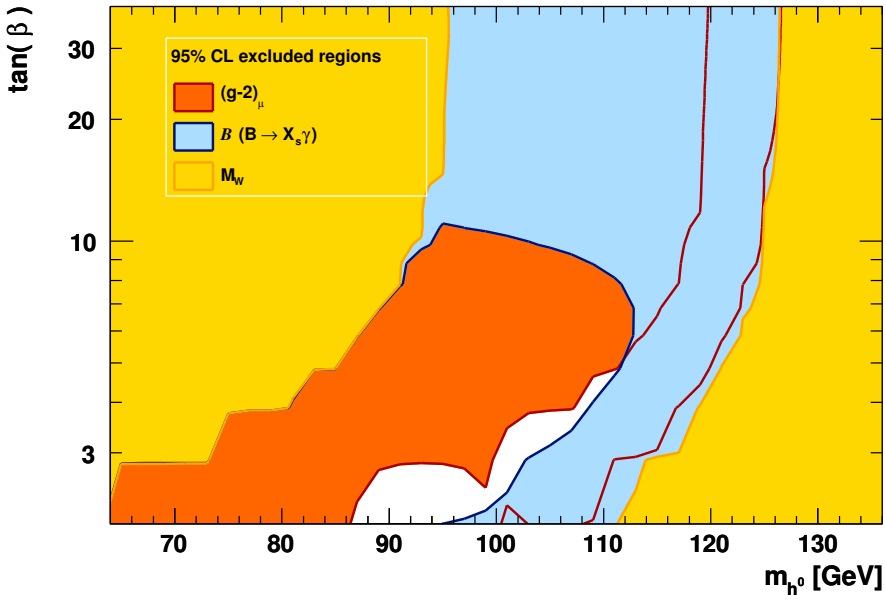

Figure 1: 95% excluded regions in the $\tan\beta$, m_{h^0} plane for $B \to X_s\gamma$, M_W and $(g-2)_\mu$ for the m_h-max scenario.

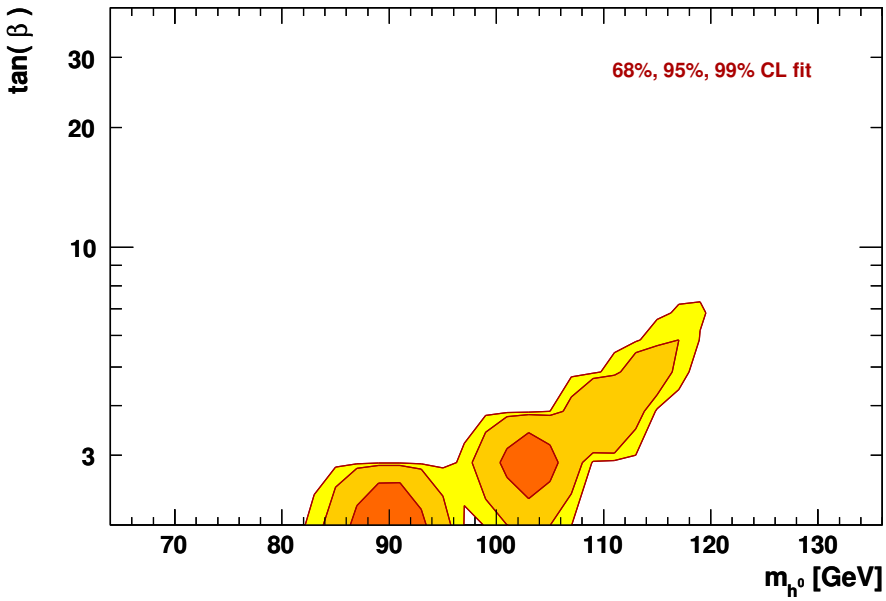

Figure 2: 68%, 95% and 99% CL allowed regions for the full fit for the m_h-max scenario.

4 Conclusion

Gfitter is a powerful tool for statistical analysis of HEP models. However pre-existing results can be time consuming to reproduce in the occasion of new experimental values becoming available. This report presented a tool for completely automating the production of results, as well as their publication to the Gfitter website.

Preliminary results from the supersymmetric module for Gfitter (GSUSY), which was implemented using the FeynHiggs package, show promising capabilities to constrain the supersymmetric parameter space. When considering the m_h-max scenario the $\tan\beta$, m_{h^0} parameter space was significantly constrained, especially when combined with results of direct searches at LEP. Future work should provide a more extensive investigation of the parameter space and many other possibilities exist, for example the investigation of an mSUGRA model.

References

[1] *H. Flächer, M. Goebel, J. Haller, A. Höcker, K. Mönig, J. Stelzer. Gfitter - Revisiting the Global Electroweak Fit of the Standard Model and Beyond.* to be submitted to Eur. Phys. J. C.

[2] F. Rademaker R. Brun. Root - an object oriented data. analysis framework. *Nucl. Instrum. Meth. A*, 389:81–86, 1997.

[3] *http://www.desy.de/∼gfitter/.*

[4] A. B. Arbuzov *et. al.* Zfitter: a semi-analytical program for fermion pair production in e+e- annihilation, from version 6.21 to version 6.42. *Comput. Phys. Commun.*, (174):728–758, 2006.

[5] S. Heinemeyer W. Hollik H. Rzehak G. Weiglein M. Frank, T. Hahn. The higgs boson masses and mixings of the complex mssm in the feynman-diagrammatic approach. *JHEP*, (47), 02 2007.

[6] W. Hollik P. Slavich G. Weiglein G. Degrassi, S. Heinemeyer. Towards high-precision predictions for the mssm higgs sector. *Eur. Phys. J. C.*, (28):133–143, 2003.

[7] G. Weiglein S. Heinemeyer, W. Hollik. The masses of the neutral cp-even higgs bosons in the mssm: Accurate analysis at the two-loop level. *Eur. Phys. J. C.*, (9):343–366, 1999.

[8] G. Weiglein S. Heinemeyer, W. Hollik. Feynhiggs: a program for the calculation of the masses of the neutral cp-even higgs bosons in the mssm. *Comput. Phys. Commun.*, (124):76–89, 2000.

[9] Peter W. Higgs. Broken symmetries, massless particles and gauge fields. *Phys. Lett.*, 12:132–133, 1964.

[10] F. Englert and R. Brout. Broken symmetry and the mass of guage vector mesons. *Phys. Rev. Lett.*, 13:321–322, 1964.

[11] G. S. Guralnik, C. R. Hagen, and T. W. B. Kibble. Global conservation laws and massless particles. *Phys. Rev. Lett.*, 13:585–587, 1964.

[12] Stephen P. Martin. A supersymmetry primer, version 4. [arXiv:hep-ph/9709356v4], 2006.

[13] L3 OPAL Collaborations. The LEP Working Group for Higgs Boson Searches ALEPH, DELPHI. Search for neutral mssm higgs bosons at lep. *Eur. Phys. J. C*, (47):547–587, 2006.

A Gfitter Automation : A User Guide

A.1 Fit Automation

Basic usage:

GenCards.py ⟨[arguments]⟩ [INPUT FILE]

Where [arguments] can be one or more of

–**help** Print a help message and exit

-s Safe mode, don't overwrite any pre-existing files.

-r Don't execute root

-m=MASTER Use the specified master file instead of the default DataCard.xml

-x=COMMAND Execute command before doing anything

Several default values are defined at the top of the script itself, namely the default master file and the output directory. The input file needs to define the list *st_cards* and should be structured as follows:

```
st_cards = {
    { 'out' : '[OUTPUT_FILENAME]',
      ⟨'master' : '[MASTER_FILE]'⟩
      'spec' : {
        ⟨'[TAGNAME_TO_ALTER]' : {
                                  ⟨'_st_mod_matchattrib' : { '[MATCH_ATTR]' : '[MATCH_ATTR_VALUE]' }, ⟩
                                  '[ATTRIBUTE_NAME_TO_ALTER]' : '[NEW_ATTRIBUTE_VALUE]',
                                  ...
                                },
        ⟩
        ⟨ 'Parameters' : {
             '[PARAM_TO_ALTER]' : { '[ATTRIBUTE_NAME_TO_ALTER]' : ( '[NEW_ATTRIBUTE_VALUE]' |
                                                                    '_st_mod_remove' |
                                                                    '_st_mod_range' |
                                                                    { 'value' : '[REPLACE_WITH_STRING]',
                                                                      '_st_mod_match' : '[REGULAR_EXPRESSION]'
                                                                    } ),
                                    ...,
                                    ⟨'_st_mod_type' : ('replace' | 'append' )⟩
                                    ⟨'_st_mod_match_alias' : (True|False) ⟩
                                },
             ...
        ⟩
        ⟨'postexec' : '[SHELL_COMMAND]',⟩
        ⟨'spec_range' : { 'range' :
                                  { 'Parameters' :
                                     { '[PARAM]' : { '[ATTR]' : [VALUE_LIST] },
                                                   ...
                                     },
                                     ...
                                  },
                                  'iterations' :  [NUMBER_OF_ITERATIONS] },
                                     ...
        ⟩
    },
    ...
}
```

where

[ ] specifies an item that needs to be specified (e.g. [OUTPUT_FILENAME] should read something like 'MyBigFile').

⟨ ⟩ specifies an optional entry.

... specifies that the previous entry can be repeated as many times as necessary.

(one | two) specifies a range where one of the entries should be specified (i.e. in this case either *one* or *two*).

NOTE Parameter tags are handled separately (as shown above) and so should only appear under the 'Parameters' entry.

_st_mod_* entries change the behaviour of the script.

Description of each entry:

out Specifies the filename to which output should be written, this will be the name of both the xml steering card and the resulting root file (NOTE a file extension should not be specified!).

master *Optional* Use the specified master file (do not include file extension) instead of the default DataCard.xml.

spec The specification of what needs to be modified in the master file.

- [**TAGNAME_TO_ALTER**] *Optional* Modify the xml tag with the given name
  - **_st_mod_matchattrib** *Optional* As well as matching on the tag name perform a secondary match on the given attribute name/-value pair. Useful for changing a tag which appears multiple times e.g. correlations.
  - [**ATTRIBUTE_NAME_TO_ALTER**] Change this attribute to the given [NEW_ATTRIBUTE_VALUE].

Parameters *Optional* Modify the specified parameters

- [**PARAM_TO_ALTER**] Tag name of the parameter to modify
  - [**ATTRIBUTE_NAME_TO_ALTER**] Alter this attribute by performing one of the following
    - [**NEW_ATTRIBUTE_VALUE** Simply set the attribute to a new value.
    - **_st_mod_remove** Delete this attribute.
    - **_st_mod_range** Set this attribute to a range, which must be specified under the 'spec_range' entry. The parameter will be iteratively set to every entry in the sequence with a Gfitter run being performed at each step.
    - **_st_mod_match** Alter the current value of the attribute by replacing any occurrences that match [REGULAR_EXPRESSION] with [REPLACE_WITH_STRING].
  - **_st_mod_type** *Optional* One of
    - **replace** Replace the current parameter (default)
    - **append** Append a new parameter tag to the file, leaving the current parameter tag intact (if it exists).
  - **_st_mod_match_alias** Match on the value of the 'Alias' attribute instead of the tag name.

postexec *Optional* Execute the specified shell command after generating the xml card and running root. Note that this has the nice feature that any thing between graves (‘) will be preprocessed as Python commands, so values from script variables (or any Python statement) may be used here.

spec_range *Optional* Define ranges for parameters which will be iteratively modified

range Specify values which change at each iteration.

Parameters Only varying parameters is supported at present.

[**PARAM**] A parameter which will be updated at every step

[**ATTR**] An attribute which will be updated at every step

[**VALUE_LIST**] A Python list which contains a value for each iteration.

iterations The number of iterations that will be performed.

Notes & tips

- a very useful example of the confusing entries 'postexec' and 'spec_range' can be found in the file "MH_PValue.py".
- the shell script 'GenAllCards.sh' will produce the full Standard Model fit.

A.2 Website Publication

The website update script takes as input a single file, processes any ⟨FILLME ... /⟩ tags within the script, generates the required plots and tables using ROOT macros and performs archiving if necessary.
Basic usage:

UpdateWeb.py ⟨[arguments]⟩ [INPUT FILE].in

Where [arguments] can be one or more of

–**help** Print a help message and exit

-A Suppress archiving, just overwrite the previous file

-d debug, show root output

Configuration variables (such as the web directory, Gfitter directory etc.) and all available plots and results are defined within the script. Each plot is defined using an *id*, the name of the input file, the input file type, the macro which will generate the plot and optionally conversion options as well as a post macro may be specified. For example:

```
'full_fit_table' : { 'input' : 'ShowFullFitTable',
                     'input_type' : '.eps',
                     'macro' : 'FullFitTable.C',
                     'post_macro' : GFDIR + 'scripts/table2gif.sh ShowFullFitTable',
                     'convert_options' : '-density 288 -geometry 45% -trim'
                   }
```

Each available result is defined in a similar way, but the necessary information this time is the ROOT file, the branch and the leaf (histogram) in which the result may be found. An optional post processing of the result may also be specified. For example:

```
'standard_higgs_pvalue' : { 'file' : 'FullFit_Standard.root',
                            'path' : 'Result_MH|MH_Chi2',
                            'postprocess' : 'TMath.Prob(val,13)' }
```

A particular page on the website which is desired to be auto generated should be created as a '.in' file. For example to auto-generate the 'index.html' page an 'index.html.in' file should be created (note that any 'index.html' file which exists will be overwritten by the script!). *FILLME* tags should then be placed within the '.in' file to specify the placement of plots and tables and to control the behaviour of the script.

The available *FILLME* tags are defined as follows:

```
<FILLME type="option"
        option="[OPTION_NAME]"
        value="[OPTION_VALUE]"
        />
```

where [OPTION_NAME] specifies **any global variable** in the script but the most obvious choices would be *archive_date*, *archive_files_dir* and the boolean *archive* which toggles archiving (default is False or archiving switched off).

```
<FILLME type="archive_toc"
        />
```

places the archive table of contents.

```
<FILLME type="plot"
        id="[ID]"
        size="([WIDTH]x[HEIGHT] | x[HEIGHT] | [WIDTH]x)"
        link="(True | False)"
        />
```

places a plot identified by [ID], optionally with a link to a full size version and optionally resized to the specified dimensions.

```
<FILLME type='result'
        id='[ID]"
        format="[FORMAT]"
        />
```

places a result identified by [ID] which will be formatted using the (sprintf like) formating string e.g. "%.0f".

```
<FILLME type="link"
        id="[ID]"
        ext=".eps"
        text=".ps"
        />
```

places a link to a version of a file specified by [ID] (probably a plot). The file type to link to is specified by the *ext* attribute while the text of the link is specified by *text*.