Description Of The Signals Implemented In LHC-FASER

January 9, 2012

Contents

1 Introduction

The C++ code LHC-FASER reads in a spectrum for the Minimal Supersymmetric Standard Model (MSSM) from a file in the SUSY Les Houches Accords (SLHA) format and returns approximate event rates for various signals. The code and accompanying lookup tables were created by Michael Krämer, Jonas Lindert, Ben O'Leary, and Carsten Robens.

2 LHC-FASER Flow

There are two stages in the intended workflow of LHC-FASER: the setup stage, and the calculation of rates for each point.

2.1 Setup stage

1: The `LHC-FASER.hpp` header file is included.

2: An `LHC_FASER::lhcFaser` object is constructed.

3: The required signals are added by the `LHC_FASER::lhcFaser::addSignal( std::string const )` function. The given string is interpretted by the `LHC_FASER::lhcFaser` object as one of the signals described in section 3. This function returns a pointer to the `LHC_FASER::signalHandler` object which calculates the event rate for this particular signal. In adding the signal, certain objects are constructed and lookup grids are loaded:

   a: If the cross-section grids for the production of the colored sparticles at the signal's energy have not yet been loaded, they are loaded.

   b: If the objects responsible for working out the acceptances for the various Standard Model (SM) particles that are produced by the cascade decays of the colored sparticles at the signal's energy have not yet been constructed, they are constructed.

   c: If the jet and Missing Transverse Energy (MET) cut acceptance grids for the signal's energy have not yet been loaded, they are loaded.

   Signals share grids, and only one instance of any grid is loaded into memory within a given `LHC_FASER::lhcFaser` object.

2.2 Workflow for a new point

Once the `LHC_FASER::signalHandler` objects have been constructed, they are used to calculate rates for different MSSM spectra as given in SLHA files. However, since `LHC-FASER` is not responsible for the generation of the SLHA files, the `LHC_FASER::lhcFaser` object needs to be informed when to read in a new SLHA file.

1: The `LHC_FASER::lhcFaser` object is told to read in the SLHA file with its `LHC_FASER::lhcFaser::updateForNewSlha( std::string const )` function. The string provides the name of the SLHA file. It does not matter if the name of the SLHA file is different for the new point or not, though the function may be used without an argument, and in this case the last name used is used again. When either version of this function is called, all the internal objects that are used in the calculation of the rates erase their cached values and prepare to re-calculate their values if required for this new point.

2: The `LHC_FASER::signalHandler` pointers are then used to call their `LHC_FASER::signalHandler::getValue()` functions. These functions collect various cached values from internal objects and combine them to obtain a final approximate event rate. Each cached value is re-calculated the first time it is sought after the `LHC_FASER::lhcFaser` object performs its `LHC_FASER::lhcFaser::updateForNewSlha( std::string const )` function. The flow of calculating the event rate is follows:

a: Each pair of colored sparticles that can be directly produced is iterated through, and the (cached) interpolated cross-section from the appropriate grid is used to determine whether this channel is worth calculating fully: if the cross-section is too low, the subsequent cascades are not calculated.

b: If cross-section for a channel is significant, all the (cached) possible yet non-negligible cascades for each of the sparticles are considered. This is to say that as the cascades are built from following direct decays of sparticles, at each stage of the cascade, any decays of the last sparticle in the cascade which result in a total branching ratio that is below a threshold are ignored when building the next stage of the cascade.

c: If branching ratios of the pair of cascades combined with the cross-section of the channel, *e.g.* the partial cross-section into the final state defined by the cascades, is not below a threshold, the (cached) acceptances for the appropriate numbers of leptons and jets from the cascades are combined with the (cached) interpolated value of the acceptance for the jet plus MET cut to obtain a final value to be returned.

3: Other signals from the same `LHC_FASER::lhcFaser` object benefit from using the cached values which are calculated for the new point when requested by the first `LHC_FASER::signalHandler` pointer's `LHC_FASER::signalHandler::getValue()` function. However, some cached values may only be relevant to some signals, and they are only be calculated when first required (in the context of a single SLHA file's data).

3 `LHC-FASER` Signals

The `LHC-FASER` signals are essentially a combination of which jet+MET cut set to use with which combination of leptons to use.

3.1 Jet+MET cut sets

No cut: No jet or MET cuts are set. The total cross-section is only reduced by the lepton cuts.

CMS 2-jet: The signal is multiplied by the acceptance for at least two hard jets with a cut on α_T as described by [1]. The both jets have to have $p_T > 100$ GeV. The MET comes indirectly from $\alpha_T > 0.55$. If the event has more than two hard jets (any jets with $p_T > 50$ GeV are counted), they are clustered into two pseudo-jets according to the reference.

Atlas 3-jet: The signal is multiplied by the acceptance for three hard jets plus MET with a cut on M_{eff} as described by [2]. The leading jet has to have $p_T > 120$ GeV, the others $p_T > 40$ GeV. The MET must be > 100 GeV and also $> 0.25 M_{\text{eff}}$. There are also cuts on the angular separation of the jets. This is not quite what is described in [2].

Atlas 4-jet: As above for the Atlas 3-jet signal, but now *four* hard jets are required, again with $p_T > 120, 40, 40, 40$ GeV. Also, the MET is now required to be only $> 0.2 M_{\text{eff}}$ in addition to > 100 GeV. Again, this is not quite what is described in [2].

3.2 Lepton cut sets

Any combination of any number of any charge of either electrons or muons may be specified, though only a single acceptance cut and a single vetoing cut may be specified per signal. The acceptance and vetoing cuts are to say that for a signal requiring n positrons for example, only events with exactly n positrons with $p_T >$ acceptance cut *and* exactly zero positrons with $p_T <$ acceptance cut but $>$ vetoing cut, and *any* number of positrons with $p_T <$ vetoing cut are counted. For example, if the acceptance cut is 20 GeV and the vetoing cut is 10 GeV, and exactly 1 lepton of either charge and either flavor is required, an event with 1 μ^+ with $p_T = 25$ GeV and no other leptons will count; an event with 1 e^- with $p_T = 122$ GeV and 18 other leptons all with $p_T < 10$ GeV will count; an event with 1 μ^+ with $p_T = 15$ GeV and no other leptons will not count; an event with 1 μ^- with $p_T = 32$ GeV, 1 μ^+ with $p_T = 12$ GeV and no other leptons will not count.

However, the signals are restricted to a finite number of combinations of jet+MET cuts with lepton cuts, so the current lepton cuts used are:

0: Exactly zero leptons may pass the vetoing cut.

1: Exactly one $(\mu^\pm, e^\pm)$ must pass the acceptance cut and exactly zero further leptons may pass the vetoing cut.

2: Exactly two $(\mu^\pm, e^\pm)$ must pass the acceptance cut and exactly zero further leptons may pass the vetoing cut.

OSSF-OSDF: Exactly two leptons, both of the opposite electric charge, must pass the acceptance cut and exactly zero further leptons may pass the vetoing cut. The total with different flavor (OSDF) is subtracted from the total with same flavor (OSSF).

same-sign dilepton: Exactly two leptons, both of the same electric charge, must pass the acceptance cut and exactly zero further leptons may pass the vetoing cut.

3.3 Currently-implemented signals

Atlas4jMET0l: Atlas 4-jet cut + 0-lepton cut; reference [2]

Atlas4jMET1l: Atlas 4-jet cut + 1-lepton cut; reference [2]

Atlas4jMET2l: Atlas 4-jet cut + 2-lepton cut; reference [2]

Atlas3jMET0l: Atlas 3-jet cut + 0-lepton cut; reference [2]

Atlas3jMET1l: Atlas 3-jet cut + 1-lepton cut; reference [2]

Atlas3jMET2l: Atlas 3-jet cut + 2-lepton cut; reference [2]

sameSignDilepton: generic same-sign dilepton cut with a 20 GeV cut for both acceptance and vetoing, no jet or MET cut; based on a paper I cannot find any more, probably should change it to be similar to reference [3], or maybe that can become "CMSsamesigndilepton".

The strings to passwhen adding these signals are now in a format for parsing the number of leptons, beam energy, and lepton cuts "easily". For example, `Atlas4jMET_0l_7TeV_pTl12.5GeV` is the signal with the Atlas 4-jet cut + 0-lepton cut for LHC beam energy of 7 TeV with a user-defined lepton p_T cut of 12.5 GeV, and `Atlas3jMET_1l_7TeV_pTl27.0GeV_pTl12.5GeV` is the signal with the Atlas 3-jet cut + 1-lepton cut for LHC beam energy of 7 TeV with a user-defined lepton acceptance p_T cut of 27.0 GeV and lepton veto p_T cut of 12.5 GeV. These signals with default lepton p_T cuts are specified with the same strings but without the "`_pTl12.5GeV`" and so on.

3.4 Soon-to-be-implemented signals

Atlas4jMETOSSFOSDF: Atlas 4-jet cut + OSSF-OSDF cut.

Atlas3jMETOSSFOSDF: Atlas 3-jet cut + OSSF-OSDF cut.

CMS2jMETanyl: CMS 2-jet cut with no lepton cut.

CMS2jMETOSSFOSDF: CMS 2-jet cut + OSSF-OSDF cut.

CMSsamesigndilepton: reference [3].

4 Approximations used

I should get around to writing this soon.

5 Conclusions

A Improvements still to come

References

[1] Vardan Khachatryan et al. Search for Supersymmetry in pp Collisions at 7 TeV in Events with Jets and Missing Transverse Energy. *Phys.Lett.*, B698:196–218, 2011.

[2] Coll ATLAS. Prospects for supersymmetry discovery based on inclusive searches at a 7 tev centre-of-mass energy with the atlas detector. Technical Report ATL-PHYS-PUB-2010-010, CERN, Geneva, Jul 2010.

[3] Serguei Chatrchyan et al. Search for new physics with same-sign isolated dilepton events with jets and missing transverse energy at the LHC. *JHEP*, 1106:077, 2011.