Die approbierte Originalversion dieser Diplom-/Masterarbeit ist in der Hauptbibliothek der Technischen Universität Wien aufgestellt und zugänglich.
http://www.ub.tuwien.ac.at

The approved original version of this diploma or master thesis is available at the main library of the Vienna University of Technology.
http://www.ub.tuwien.ac.at/eng

TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

DIPLOMARBEIT

Development of a Framework for Interpretation of LHC Data Based on Simplified Models

Ausgeführt am
Atominstitut der Technischen Universität Wien
in Zusammenarbeit mit dem
Institut für Hochenergiephysik (HEPHY)
der Österreichischen Akademie der Wissenschaften (ÖAW)

unter der Anleitung von
Univ.Prof. DI Dr. Christian Fabjan
und DI Dr. Wolfgang Waltenberger

durch

Doris Proschofsky-Spindler

Wien, 9.1.2014

Abstract

Within the scope of this thesis a framework for interpretation of searches for new physics at the Large Hadron Collider (LHC) was developed. This framework is called `SModelS` and is based on the idea of simplified model spectra (SMS). It uses the SMS as an interface between results from LHC searches and theories for new physics Beyond the Standard Model (BSM). A database was installed to store the data from supersymmetry (SUSY) searches from the experiments CMS and ATLAS. The `SModelS` framework decomposes a BSM model into its SMS components and calculates theoretical predictions for the production cross section of each SMS. To apply these predictions, the computed values are compared to the experimental data from the database.

The idea behind `SModelS` is to have a framework which generalizes the existing results for SUSY searches in order to make statements about other BSM theories as well. The `SModelS` framework also can be used to identify blind spots in the experimental procedure, i.e., regions in the parameter space of SUSY or other BSM theories, which are not probed by the current analyses. In case of a positive result in one analysis, `SModelS` can help find the best fitting parameter regions compatible with all other analyses.

In this thesis the `SModelS` framework is explained and its validation is shown. A first scan over a simple pMSSM model is performed and analyzed.

Kurzfassung

Im Rahmen dieser Diplomarbeit wurde ein Softwarepaket zur Interpretation der Suchen nach neuer Physik am Large Hadron Collider (LHC) entwickelt. Dieses Softwarepaket heißt `SModelS`. `SModelS` basiert auf dem Konzept von Simplified Model Spectra (SMS). Es verwendet die SMS als Verbindung zwischen den Ergebnissen der Suchen am LHC und den Theorien über Physik jenseits des Standard Models (BSM). Um die Ergebnisse der Suchen nach Supersymmetrie (SUSY) der Experimente CMS und ATLAS zu speichern, wurde eine Datenbank erstellt. `SModelS` spaltet ein BSM Model in seine SMS Komponenten auf und berechnet deren Produktionswirkungsquerschnitt, welcher von der entsprechenden Theorie vorhergesagt wird. Um diese Vorhersagen interpretieren zu können, werden die berechneten Werte mit den experimentellen Ergebnissen aus der Datenbank verglichen.

Die Idee hinter `SModelS` ist ein Softwarepaket zur Verfügung zu haben, welches die existierenden Ergebnisse der SUSY Suchen verallgemeinert um mit ihrer Hilfe auch Aussagen über andere BSM Theorien machen zu können. `SModelS` kann auch verwendet werden, um blinde Punkte in den experimentellen Analysetechniken zu identifizieren, zum Beispiel Regionen im Parameterraum von SUSY oder anderen BSM Theorien, die durch die bisherigen Analysen noch nicht getestet wurden. Falls bei einer Analyse ein Signal auftritt, kann `SModelS` dazu verwendet werden, die Region im Parameterraum zu finden, welche diesen neuen Effekt am besten beschreibt.

In dieser Diplomarbeit wird `SModelS` erklärt und seine Validierung gezeigt. Außerdem wurde ein erster Scan über ein einfaches pMSSM Model ausgeführt und analysiert.

Acknowledgements

First, I would like to express my sincere gratitude to Prof. Fabjan for giving me the opportunity to do my thesis in such a nice and productive working environment and for his special support during the last month of my thesis, which allowed me to finally finish my studies. I would like to express the deepest appreciation to my supervisor Wolfgang Waltenberger for his guidance, his continuous support and his endless patience answering all my questions even the most stupid ones. Furthermore, I would like to express my special thanks to Wolfgang Adam and Sabine Kraml for their experienced advice, their much appreciated help, especially during the final period, and for proofreading this thesis. I also want to thank Suchita Kulkarni and Andre Lessa for many fruitful discussions and their enjoyable company during the stay in Grenoble and the meeting in Vienna. My very special thanks belong to Ursula Laa for the many many hours of working together and the endless discussions we had about programming, particle physics, . . . I am really grateful for your company. I also want to thank Wolfgang Magerl for his professional advice regarding programming and his patience while explaining it to me. Actually I want to thank all my colleagues from the HEPHY CMS analysis team for their help and the pleasant company. Furthermore, I am grateful for the financial support by the FEMtech initiative of the BMVIT of Austria.

Finally, my sincere thanks belong to my parents and my sister who always supported and encouraged me unconditionally and my beloved boyfriend Andreas who stood by me during all ups and downs.

Contents

1 Introduction

The Large Hadron Collider (LHC) at CERN (Conseil Européen pour la Recherche Nucléaire) was built to search for new physics at the TeV energy scale. ATLAS (A Toroidal LHC ApparatuS) and CMS (Compact Muon Solenoid) are the two general purpose experiments at the LHC. After the first phase of LHC operations, the results from the experiments agree very well with the predictions of the Standard Model of particle physics (SM). On July 4^{th}, 2012 CMS and ATLAS conjointly announced the discovery of a new boson with a mass of about 125 GeV [1, 2]. Ensuing measurements showed that this boson is a SM(-like) Higgs boson. The discovery of the Higgs boson completes the SM. Therefore, the Nobel Prize in physics was awarded to Francois Englert and Peter W. Higgs on October 8^{th}, 2013. However, the SM cannot answer many fundamental questions. Hence, new physics beyond the SM (BSM) is expected at the TeV energy scale, but until now no signs for new physics are observed.

One elegant solution to many deficiencies of the SM would be the existence of supersymmetry (SUSY). Supersymmetry is an extension to the SM which predicts a supersymmetric partner for each SM particle. Due to the complexity of the theory, SUSY searches at ATLAS and CMS interpret their results within so-called simplified model spectra (SMS). Simplified model spectra are effective models, which introduce only a few new particles and branching ratios. The fact that the SUSY searches at CMS and ATLAS use SMS is the basis for the interpretation tool, called `SModelS` [3], which was developed within the scope of this thesis in cooperation with a group of experimental and theoretical physicists.

The idea behind `SModelS` is to use the published SUSY results and generalize them in order to make more general statements about SUSY and other BSM models. The `SModelS` framework can identify possible regions in the parameter space which are not tested by the experimental analyses. In case of a positive result in one analysis, `SModelS` can help find the best fitting parameter regions compatible with all other analyses.

This document first gives an overview of the Large Hadron Collider at CERN and its two main experiments CMS and ATLAS (Chap. 2).

In the next chapter (Chap. 3) the Standard Model of particle physics is discussed in short. Supersymmetry is introduced to show a possible solution to different problems and open questions in the SM. As a consequence of the huge number of new parameters introduced by SUSY, the concept of simplified model spectra is presented at the end of this chapter.

An overview of all additional software used in the `SModelS` framework is given in Chap. 4. The software is needed for simulations of different SUSY scenarios. The typically used file formats SLHA and LHE are described as well.

The main part of this thesis describes the `SModelS` framework itself. In Chap. 5 the basic ideas behind the framework and its working principles are explained. At the end of this chapter the validation of the `SModelS` framework is described and corresponding plots are shown.

After the `SModelS` framework is explained, two examples for its application are given (Chap. 6). The first example is a scan over a simple pMSSM model. This scan shows the main purpose of the `SModelS` framework. The second example is the generation of the summary plots for SUSY searches at CMS. To create these plots the database and some functions of the framework to read out the database were used.

The last chapter (Chap. 7) gives a short summary of this thesis and an outlook at plans for further development and use of the `SModelS` framework.

2 Large Hadron Collider

The Large Hadron Collider (LHC) [4] at CERN (Conseil Européen pour la Recherche Nucléaire) is the follow-up project of the Large Electron-Positron Collider (LEP), installed in the same 26.7 km tunnel. The LHC consists of a ring of superconducting magnets, which accelerate and focus the hadrons. Two proton beams travel inside the accelerator in opposite directions close to the speed of light. Therefore, two evacuated beam pipes are needed. A solution was found in the adoption of the twin-bore magnet design also known as "two-in-one" superconducting magnet design proposed by John Blewett. There, the two beam channels are accommodated in a common cold mass and cryostat with magnetic flux circulating in opposite direction in the two pipes. The disadvantage of this solution is that the beams are not only mechanically but also magnetically coupled. Using superfluid helium, the superconducting magnets are cooled down to temperatures below 2 K and magnetic fields above 8 T are obtained. In order to insulate the cryomagnets and the helium distribution, to minimize the background at the experiments and to maximize the life time of the beam, a high vacuum is needed. Therefore, three different vacuum systems are installed: the insulation vacuum for cryomagnets, the insulation vacuum for helium distribution and the beam vacuum. For the insulation vacua a pressure of 10^{-1} mbar at room temperature and 10^{-6} mbar at cryogenic temperatures is sufficient, while for the beam vacuum the requirements are much stricter. For the beam regions at room temperature a pressure of $10^{-10} - 10^{-11}$ mbar is necessary. The vacuum in the beam is $10^{15}\,H_2\ m^{-3}$, given by gas densities normalized to hydrogen and taking into account the ionisation cross sections for each gas species. In contrast, at the intersection regions in the experiments the gas density is $10^{13}\,H_2\ m^{-3}$. To realize these low pressures in such large dimensions, the three vacuum systems are subdivided into manageable sectors by vacuum barriers for the insulation vacua and sector valves in case of the beam vacuum.

The LHC is linked to the CERN accelerator complex (Fig. 2.1), which injects the LHC with two 450 GeV proton beams. At the beginning proton bunches of 1.1×10^{11} protons each are extracted from hydrogen atoms by stripping off their electrons. To accelerate the protons up to 450 GeV a chain of different accelerators

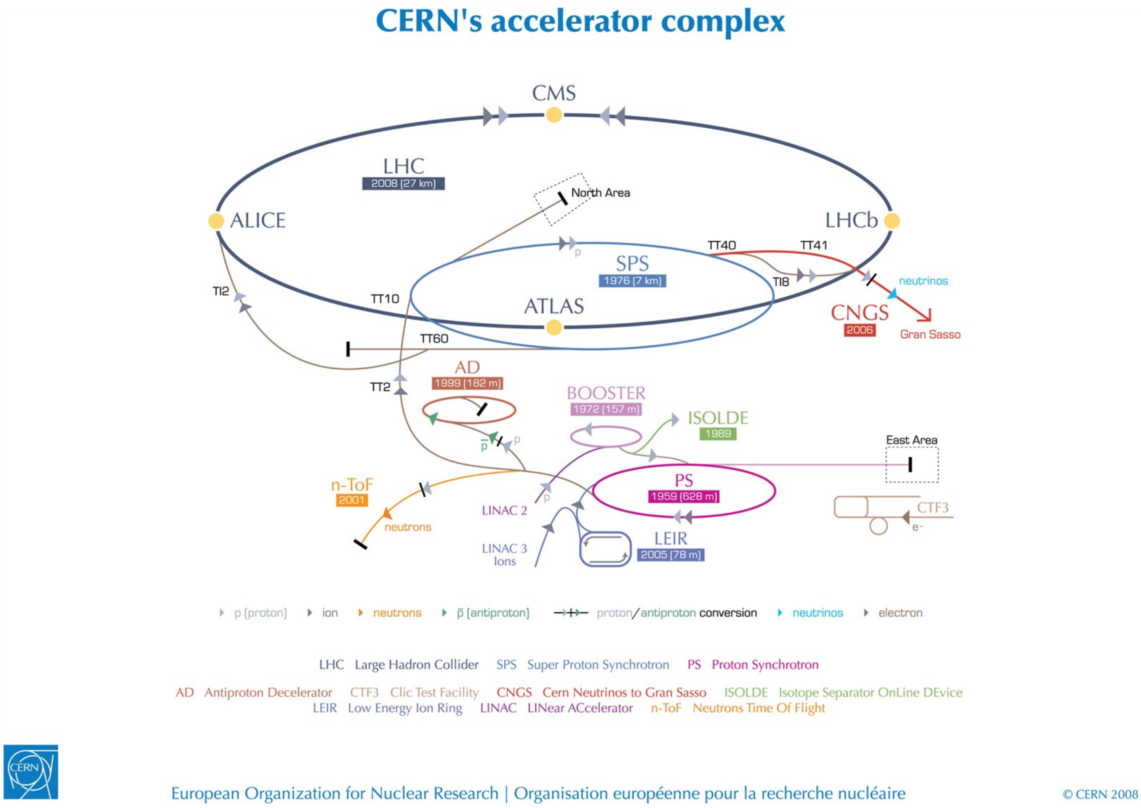


Figure 2.1: Scheme of the CERN accelerator complex [5].

is required. These bunches get first accelerated and focused in the Linac 2 (Linear Accelerator 2) until they reach a kinetic energy of 50 MeV. From Linac 2 the protons are transferred to the Proton Synchrotron Booster (PSB) where the bunches reach an energy of 1.4 GeV. Afterwards, the bunches change into the Proton Synchrotron (PS) to reach an energy of 25 GeV and are filled subsequently into the Super Proton Synchrotron (SPS). It takes about 3 to 4 cycles of the PS, with a cycling time of 3.6 s, to fill the SPS. In the SPS the protons are accelerated to 450 GeV with a cycling time of 21.6 s. From the SPS the protons are finally transferred to the LHC where they are further accelerated to the maximum energy. To fill the LHC it takes 12 cycles of the SPS. According to design values, each of the two beams at the LHC contains 2808 bunches circulating at a distance of 25 ns. These bunches are brought to collision at the four intersection points where the four main experiments ALICE (A Large Ion Collider Experiment), ATLAS (A Toroidal LHC ApparatuS), CMS (Compact Muon Solenoid) and LHCb (Large Hadron Collider beauty) are installed (Fig. 2.2). Other LHC experiments are TOTEM (Total elastic and diffractive cross-section measurement), LHCf (Large Hadron Collider forward) and MoEDAL (Monopole and Exotics Detector at the LHC).

The main goal of the LHC is finding new physics. The LHC started operation in

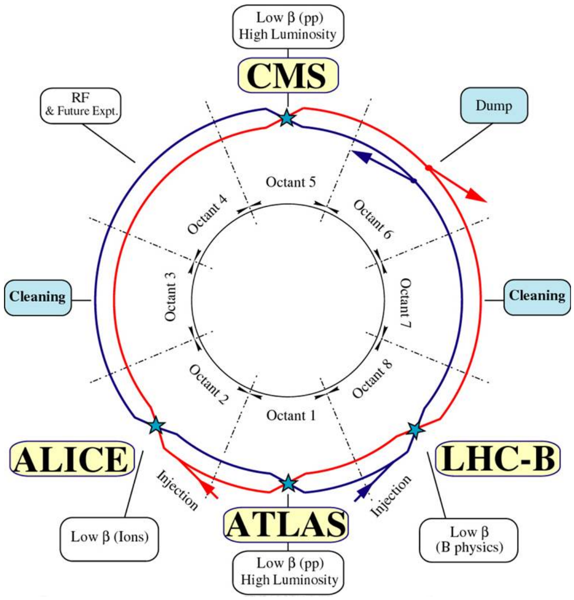


Figure 2.2: Scheme of the Large Hadron Collider with its experiments [6].

November 2009. The first two runs in 2010 and 2011 were at a center of mass energy $\sqrt{s} = 7\,\text{TeV}$ with increasing integrated luminosity. The third run in 2012 was already at $\sqrt{s} = 8\,\text{TeV}$. To upgrade and improve the accelerator and the experiments in order to reach soon the design value for the center of mass energy 14 TeV, the LHC is shut down at the moment. In 2015 operations will resume at higher energy, first at $\sqrt{s} = 13\,\text{TeV}$ with the aim to further increase the energy to the design value at $\sqrt{s} = 14\,\text{TeV}$ later on. One important parameter to characterize the performance of the LHC is the luminosity $\mathcal{L}$. The luminosity $\mathcal{L}$ only depends on beam parameters

$$\mathcal{L} = \frac{N_b^2 n_b f_{rev} \gamma_r}{4\pi \epsilon_n \beta^*} F \tag{2.1}$$

where N_b denotes the number of particles per bunch, n_b the number of bunches per beam, f_{rev} the revolution frequency, γ_r the relativistic gamma factor, ϵ_n the normalized transverse beam emittance, β^* the beta function at the collision point and F the geometric luminosity reduction factor. The design luminosity $\mathcal{L}$ of the LHC is $10^{34}\,\text{cm}^{-2}\text{s}^{-1}$. The event production rate $\frac{dN}{dt}$ is given by:

$$\frac{dN}{dt} = \mathcal{L}\sigma \tag{2.2}$$

where σ denotes the cross section. The integral of the luminosity $\mathcal{L}$ with respect to time is called integrated luminosity L. The number of collisions N can be obtained from the integrated luminosity

$$L = \int \mathcal{L} dt = \frac{N}{\sigma} \quad (2.3)$$

Knowing the integrated luminosity L, it is possible to calculate the actual expectation for the event count of any process from its cross section. The total inelastic cross section for proton-proton collisions was measured by TOTEM [7, 8]. It is determined to be $72.9 \pm 1.5\,\text{mb}$ at $\sqrt{s} = 7\,\text{TeV}$ and $74.7 \pm 1.7\,\text{mb}$ at $\sqrt{s} = 8\,\text{TeV}$.

Besides protons, the LHC is also designed for heavy ion operation.

The physics program of the LHC can be split into two major parts: Standard Model (SM) and beyond the Standard Model (BSM) physics. Some of the main goals are:

- **The Higgs boson:** One of the main reasons for building the LHC was to find the Standard Model Higgs boson. On July 4^{th}, 2012, ATLAS and CMS conjointly announced the discovery of a Higgs boson with a mass of about 125 GeV [1, 2].

- **Supersymmetric particles:** One theory which could solve some problems of the SM is supersymmetry (SUSY, see Chap. 3.2). If SUSY is natural, supersymmetric particles should be found at the LHC.

- **Dark matter:** One dark matter candidate is the lightest supersymmetric particle (LSP). Assuming that R-parity is conserved, the LSP must appear at the LHC as missing energy.

- **Extra spatial dimensions:** Another approach for physics beyond the SM are extra spatial dimensions. Some analyses search for different manifestations of extra dimensions at the LHC.

- **Standard Model:** To study the SM in more detail, quantum chromo dynamics (QCD), electroweak- and flavour-physics studies are part of the physics program of the LHC as well. Flavour-physics studies at the LHCb experiment search also for deviations from the SM.

2.1 CMS

CMS (Compact Muon Solenoid, Fig. 2.3) is one of the two general purpose detectors at LHC [10]. To achieve the LHC physics goals the requirements for the CMS detector can be defined as follows [11]:

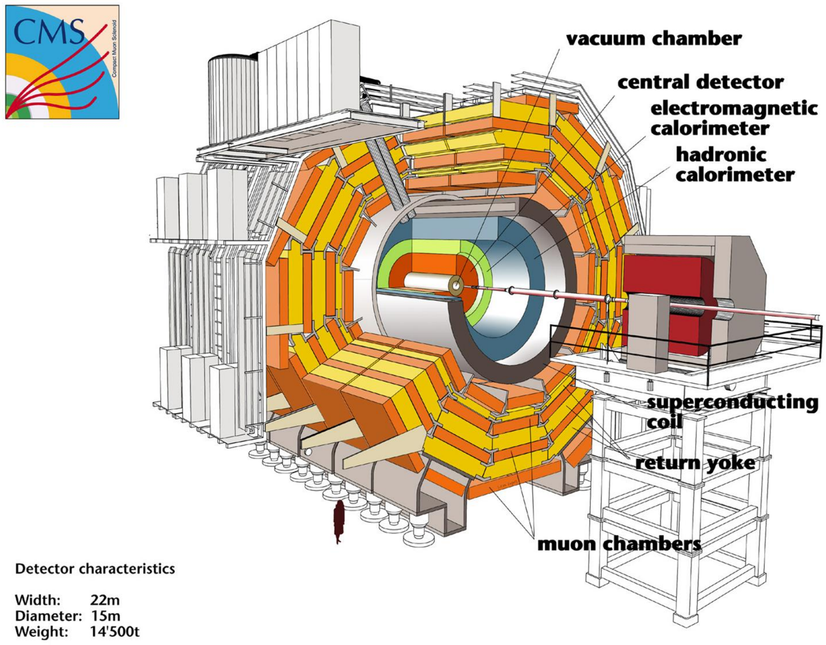


Figure 2.3: Scheme of the CMS detector [9].

- The muon system must have a good muon identification and momentum resolution. It also must have a good dimuon mass resolution of about 1 % at 100 GeV/c^2 and be able to determine the charge of muons with $p < 1$ TeV/c unambiguously.

- The requirements for the inner tracker are a good charged-particle momentum resolution and reconstruction efficiency. A pixel detector close to the interaction region is required for efficient triggering and offline tagging of τ's and b-jets.

- The electromagnetic calorimeter needs a good electromagnetic energy resolution and a good diphoton and dielectron mass resolution of about 1 % at 100 GeV/c^2. Furthermore, it must cover a wide geometry and measure the direction of the photons and localize the primary interaction vertex correctly. Also the measurement of the π^0 rejection and the efficient photon and lepton isolation at high luminosities is obligatory.

- For the hadron calorimeter a good missing-transverse-energy $\not{E}_T^{miss}$ and dijet-mass resolution is essential. Therefore, it must have a large hermetic geometric coverage with a fine lateral segmentation.

Hence, the main characteristics of CMS are a high-field solenoid, a full silicon-based inner tracking system, and a fully active scintillating crystals-based electromagnetic

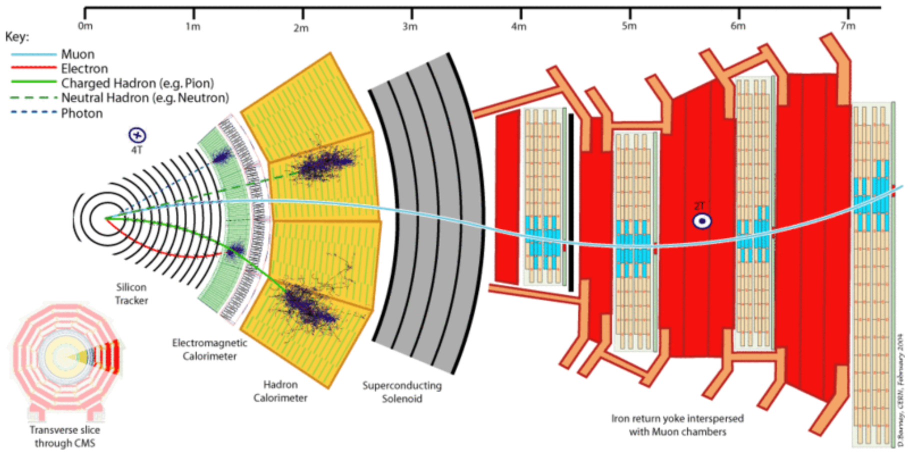


Figure 2.4: Slice plane of a vertical cut through the CMS detector including tracks of different particles [12].

calorimeter. The overall detector design is mostly determined by the choice of the magnetic field for muon momentum detection, which needs to have a large bending power for precision measurement. In numbers, CMS is 21.6 m long, has a diameter of 14.6 m and weights 12 500 tons. CMS is divided in eight major parts: five coils, two endcaps and the inner tracking part, which were assembled at the surface and afterwards let down to the cavern.

The magnet system consists of a large superconducting solenoid and iron return yokes (Fig. 2.4) and provides a magnetic field of 3.8 T. This high magnetic field is necessary for a good momentum resolution. The solenoid is a high-purity aluminum-stabilized conductor with an indirect cooling and full epoxy impregnation. The iron return yokes consist of 4 wound layers, which make each of the 5 coil modules. The return field of the solenoid is large enough to saturate the yokes so that four muon 'stations' can be integrated between to ensure robustness and full geometric coverage.

In the barrel region each muon 'station' consists of multiple layers of drift tube chambers (DT) and resistiive plate chambers (RPC). In comparison, the endcaps are equipped with cathode strip chambers (CSC) instead of DT. The reasons for the choice of DTs in the barrel region are the small neutron induced background, the low muon rate and the low residual magnetic field in the chambers, while in the endcaps the muon rate, the neutron background and the magnetic field are high. Hence, for the endcaps CSCs are chosen. RPCs are added in the barrel as well as in the endcaps because of their fast response and good time resolution. The disadvantage of RPCs is that their position resolution is much coarser. The total muon system consists of 25 000 m^2 of active detection planes and about 1 million electronic channels. To ensure precision measurement a centrally produced muon is measured three times: first in the inner tracker, then after the coil and the last time in the return flux.

Within the solenoid the outermost detector is the hadron calorimeter (HCAL) to measure hadronic showers. Because of its position, the choice is strongly influenced by the magnetic parameters. There is just a minimum amount of space left for the active medium to measure most of the hadrons before they reach the magnet. Therefore, tile/fibre technology was chosen. Tile/fibre technology means plastic scintillator tiles which are read out by embedded wavelength-shift fibres. In order to fulfill the requirements, like the reduction of the non-Gaussian tails in energy resolution and hermeticity for the missing transverse energy $\not{E}_T^{miss}$, the material in terms of interaction length must be maximized. Thus, brass is chosen as the absorber material. Other advantages of brass, besides its short interaction length, are that it is easy to machine and it is non-magnetic. The HCAL is complemented by an additional layer of scintillators, the hadron outer detector.

The HCAL surrounds the electromagnetic calorimeter (ECAL), which measures electrons and photons (Fig. 2.4). The ECAL is a hermetic homogeneous calorimeter comprising 61 200 lead tungstate ($PbWO_4$) crystals in the barrel region and 7 324 crystals in each of the 2 endcaps. The advantages of lead tungstate crystals are their short radiation and Moliere lengths, their fast response, their radiation hardness and their fine granularity. Disadvantages of lead tungstate crystals are the relatively low light yield and their sensitivity to changes in temperature. Hence, photodetectors with intrinsic grain, which can operate within a magnetic field, and a stable temperature within 0.1 °C are required. A very compact design of the ECAL is possible due to the crystals.

The innermost part of the CMS detector is the full silicon-based inner tracking system. As the name implies, the main function of the inner tracking system is the tracking of the charged particles produced at the primary vertex. It is divided into three regions because of the high luminosity of the charged particle flux at different radii. In the outermost part ($r > 55$ cm) where the particle flux is already low enough, larger-pitch silicon microstrips with a maximum cell size of $25\,\text{cm}\times180\,\mu\text{m}$ are used. Smaller silicon strip detectors ($10\,\text{cm}\times80\,\mu\text{m}$) are used in the intermediate region ($20 < r < 55$ cm). In the innermost region, closest to the interaction vertex, pixel detectors are used because of the extremely high particle flux in this region ($10^7\,\text{s}^{-1}\text{cm}^{-2}$ at $r \approx 10$ cm). The size of one pixel is $100\times150\,\mu\text{m}^2$. The inner tracker consists of 66 million pixels and 9.6 million silicon strips.

The LHC has a bunch crossing rate of 40 MHz, resulting in about 4×10^7 bunch-crossings/s at design luminosity, corresponding to almost 10^9 collisions per second.

Only data from a few hundred collisions per second can be written to permanent storage. Therefore, the trigger system needs a rejection factor of about 10^6. The CMS trigger system is comprised of four parts: the detector electronics, the Level-1 trigger processors, the readout network and the High-Level trigger (HLT) executed by the online event filter system (processor farm). The Level-1 trigger processors include the calorimeter-, the muon- and the global trigger. The maximum transit time from front end electronics to Level-1 trigger and the time for a first decision is altogether $3.2\,\mu$s. During that time the data is held in buffer while new data from the following events is collected in the detector. In this step 1 out of approximately 10 000 events is kept as output, which corresponds to $\approx$ 100 kHz. This decision making process involves the muon system, the calorimeters and some correlation information between these two systems. The decision itself is based on 'trigger primitive' objects like photons, electrons, muons and jets above a set of E_T and p_T thresholds and the global sums of E_T and $\not{E}_T^{miss}$. After the Level-1 trigger, the data is transferred to front-end readout buffers. Each event has a size of 1.5 MB on average. These events are transferred to processors, which run the HLT code. The Level-1 trigger output of 100 kHz is reduced by the HLT to a few 100 Hz for mass storage. The selected events are stored in the LHC computing grid, which consists of many worldwide spread computing centers. These computing centers are arranged in a hierarchical structure. The primary center is Tier-0 at CERN supplemented by Tier-1 and Tier-2 centers at different laboratories and universities.

2.2 ATLAS

ATLAS (A Toroidal LHC ApparatuS, Fig. 2.5) [14] is the second general purpose detector at the LHC.

The basic requirements for the ATLAS detector to fit the LHC physics program can be summarized as follows [15]:

- For electron and photon identification and measurements a very good electromagnetic calorimeter is required, which must be accompanied by a full-coverage hadron calorimeter for accurate jet and $\not{E}_T^{miss}$ measurements.
- The muon system must be able to perform high-precision muon momentum measurements also at high luminosity using only the external muon spectrometer.
- The requirements for the tracker are efficient tracking for high-p_T lepton momentum measurements at high luminosities, electron, photon, τ and heavy-

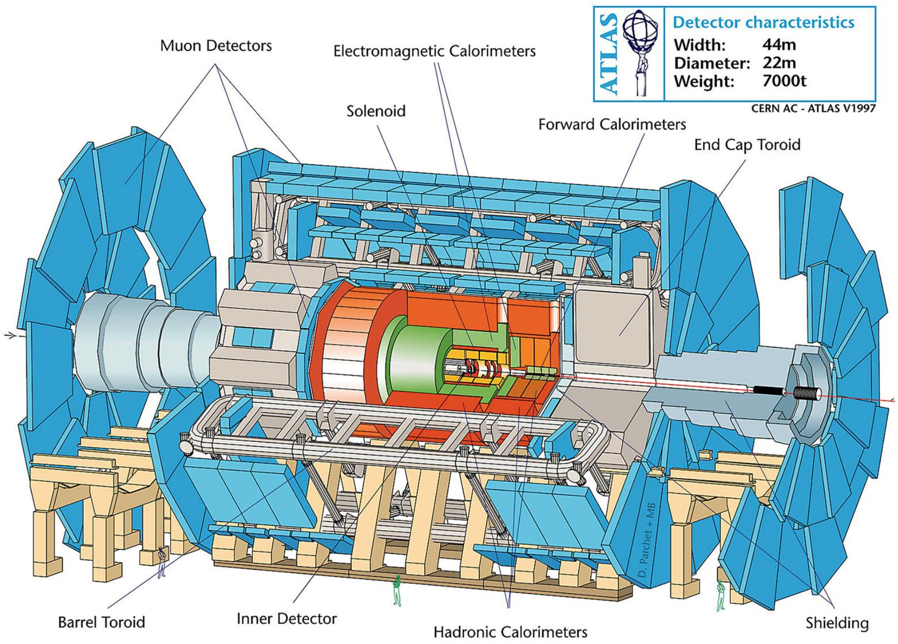


Figure 2.5: Layout of the ATLAS detector [13].

flavor identification and the capability of full event reconstruction at lower luminosities.

- For high efficiencies for most physics processes of interest at the LHC it is necessary to trigger and measure particles at low p_T thresholds.
- To miss no events of interest the detector needs maximum coverage in all directions.

The primary goal of the ATLAS detector is to provide as many signatures as possible operating at high luminosities. The overall dimensions of ATLAS are defined by the muon spectrometer. The detector is about 46 m long, has a diameter of about 22 m and weighs 7000 tons.

The main difference to the CMS detector is the magnet system. It consists of a central superconducting solenoid surrounded by large superconducting air-core toroids. The central superconducting solenoid provides a magnetic field of about 2 T for the inner detector and a peak field of 2.6 T at the inner solenoid. The toroids are independent coils arranged with an eight-fold symmetry around the calorimeters (barrel toroid) and two toroids are located at the endcaps (endcap toroids). The peak magnetic field reached at the barrel toroid is 3.9 T and at one endcap toroid 4.1 T. For cooling down the superconducting system to 4.5 K liquid helium is used.

The air-core toroids provide the magnetic field for the muon spectrometer which measures muons using the deflection of their tracks by the magnetic field. This magnet configuration has one main advantage, namely it provides a magnetic field that is mostly orthogonal to the muon trajectories and it minimizes the degradation of resolution due to multiple scattering. For muon measurement the muon spectrometer is equipped with a separate trigger and high precision tracking chambers. The choice of the spectrometer instrumentation is mostly driven by the high particle flux. Altogether four different muon chamber technologies are used for the muon spectrometer: Monitored Drift Tubes, Cathode Strip Chambers, Thin Gap Chambers and Resistive Plate Chambers. Monitored Drift Tubes are the outermost muon chambers in the barrel region and at the endcaps. They provide precision measurement of the muon tracks in the principal bending direction of the magnetic field. The Cathode Strip Chambers are closest to the primary interaction vertex because of their high granularity. Additional Resistive Plate Chambers are used in the barrel region and Thin Gap Chambers at the endcaps. To match the stringent requirements on the mechanical accuracy and the survey of the precision chambers an optical alignment is installed.

Inside the muon spectrometer, the outermost part is the hadron calorimeter. The hadron calorimeter in the barrel is divided into three parts: the central barrel and two identical extended barrels. For the barrel HCAL basically plastic scintillator tiles embedded in an iron absorber are used, while for the hadronic endcap calorimeter and the forward calorimeter lead/liquid-argon (LAr) detectors are utilized. The main advantage of LAr is their radiation hardness.

The hadron calorimeter encases the electromagnetic calorimeter, which measures electrons and photons. For the electromagnetic calorimeter LAr detectors are used as well, but this time with accordion geometry. Before the particles enter the electromagnetic calorimeter they need to pass a presampler detector, which is placed directly behind the solenoid. This presampler is used to correct for the energy lost in the material of the tracker and the solenoid. The electromagnetic calorimeter and the presampler detector are contained in the barrel cryostat, which surrounds the inner detector cavity.

The innermost part of the ATLAS detector is the inner detector mainly used for tracking. It is located inside the central solenoid and is comprised of high resolution detectors at the inner radii and continuous tracking elements at outer radii. Around the primary vertex the highest granularity is necessary. Therefore, semiconductor pixel detectors and strip detectors made of silicon are used. Because of the effort to minimize the material introduced in the detector and the costs, the total number of precision layers is limited to three pixel and eight strip layers. Therefrom, seven tracking points are obtained. In order to increase the number of tracking points a straw tube tracker is used. An other reason for the straw tube tracker is the fact, that the detector is a transition radiation detector, which is able to identify electrons. The advantages of a straw tube tracker are the large number of tracking points (36 at the ATLAS detector), that there is much less material introduced at each point and the lower costs. Because of the large number of tracking points, a continuous track following is possible. This combination gives a very robust pattern recognition and high precision.

The so achieved data needs to be selected and stored. From the initial bunch-crossing rate of 40 MHz the data must be reduced to about 100 Hz for permanent storage. This task is done by the online event selection. It consists of three levels each refining the decision from the previous and adding more selection criteria.

3 The Standard Model and Physics Beyond It

The Standard Model (SM) of particle physics is a very successful theory to describe the fundamental constituents of matter and their interactions. Many predictions of the SM are already experimentally detected. The most current example is the detection of the Higgs boson. Despite the success of the SM still some questions remain open. Therefore, it is necessary to think about possible extensions of the SM and about possible new theories beyond the SM (BSM).

3.1 Standard Model

The Standard Model of particle physics is based on quantum field theories (QFT). Mathematically it can be described by a spontaneously broken gauge theory with the gauge group structure $SU(3)_C \otimes SU(2)_L \otimes U(1)_Y$. The excitations of the quantum fields are described by elementary particles. The SM contains three types of elementary particles: matter particles with spin $\frac{1}{2}$, gauge bosons with spin 1 and the Higgs boson with spin 0 (Fig. 3.1). The fermions are grouped into three generations of left-handed quarks and leptons (isospin doublets) as well as right-handed quarks and charged leptons (isospin singulet). In physics four fundamental forces are known: electromagnetic, weak, strong interaction and gravitation. The electromagnetic and the weak force can be combined to the electroweak force. The SM describes these fundamental interactions except for the gravitation. The fundamental interactions between fermions are mediated by the exchange of gauge bosons. Leptons interact only via the electroweak force exchanging photons, $W^{\pm}$ and Z bosons while quarks also interact via the strong force exchanging gluons since they carry color charge as well. To introduce the masses of the elementary particles into the SM a scalar field, the Higgs field, has to be added to the theory. The Higgs field acquires a non zero vacuum expectation value (VEV) $v \approx 246\,\text{GeV}$. This non zero ground state energy of the Higgs field breaks the electroweak symmetry spontaneously, which means that the Lagrangian is invariant under symmetry transformations while the ground state

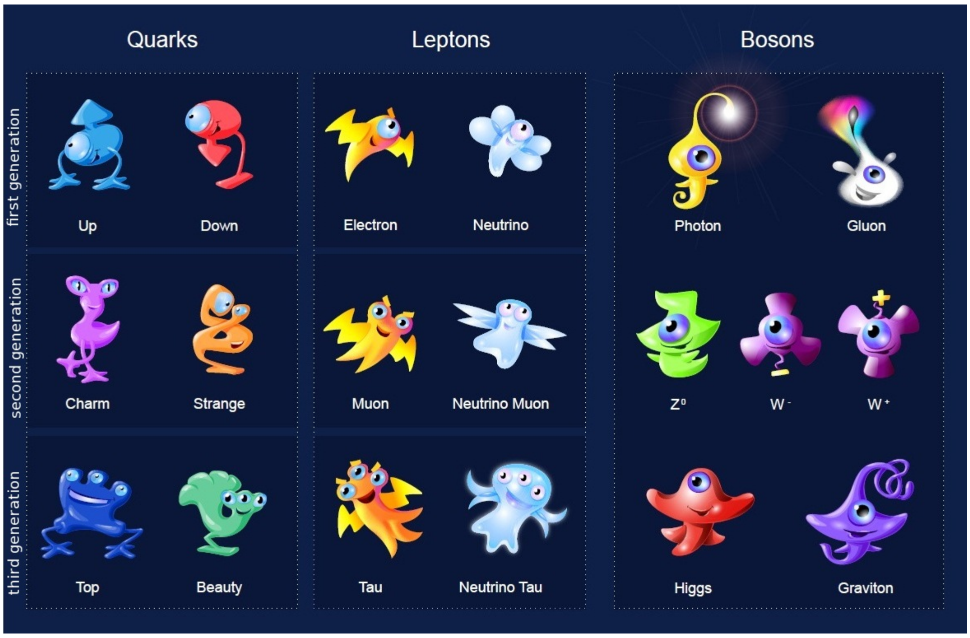


Figure 3.1: Standard Model particles (Artist's view) [16].

is not. Due to electroweak symmetry breaking, the massless gauge fields B and $W_{1,2,3}$ mix to the observed mass eigenstates $W^{\pm}$ and Z^0 and the photon field A. Hence, the Higgs field gives mass to the gauge bosons. The fermions gain their mass by adding Yukawa terms, which couple the fermions to the Higgs field.

The SM is a very succesfull theory, which is confirmed by experiments to a high level of precision up to the electroweak energy scale. Nevertheless, it has to be an effective field theory since it does not include gravity and general relativity. Therefore, it is not valid up to arbitrary high energies. Some questions remain open like:

- **Gravity:** One of the main problems of the SM is that gravity cannot be included. At low energies, where gravity is very weak compared to other forces, it can be neglected. A problem arises as soon as the energy approches the Planck scale ($m_P \approx 1.22 \times 10^{19}$ GeV) since at these energies gravity should be comparable to the other three elementary forces.

- **Unification of gauge couplings:** The strength of interactions is described by the gauge couplings, which depend on the energy scale. By extrapolating the gauge couplings to higher energy scales, their unification is expected. The extrapolation in the SM to the GUT (Grand Unified Theory) scale ($m_{GUT} \approx 10^{15}$ GeV) does not result in their exact unification. Another problem that arises from the SM GUT scale is a predicted proton decay. So far, a proton decay has never been observed. Assuming supersymmetry (SUSY) the unification can be realized at a GUT scale of $m_{GUT} \approx 2 \times 10^{16}$ GeV since the additional supersymmetric particles contribute to the evolution as well. Another benefit of the higher GUT scale is that protons are allowed to be stable.

- **Origin of electroweak symmetry breaking:** The electroweak symmetry breaking is established by introducing a potential term for the Higgs field to the SM Lagrangian. This term does not follow directly from theory and it is not understood why it is realized in our universe in that specific form.

- **Hierarchy problem:** The hierarchy problem arises from the fact that all particles, which couple to the Higgs field, add quadratically divergent loop corrections to the bare Higgs boson mass. The loop corrections to m_H^2 in first order from a fermion coupling to a Higgs boson are described by

$$\Delta m_H^2 = -\frac{|\lambda_f|^2}{8\pi^2}\Lambda_{UV}^2 \ldots \tag{3.1}$$

where λ_f is the Yukawa coupling of Higgs field to fermion fields and Λ_{UV}^2 denotes the ultraviolet momentum cut-off to regulate the loop integral. Assuming that the SM is valid up to Planck scale ($\Lambda_{UV}^2 = m_P^2$) the first order loop correction Δm_H^2 would be many orders of magnitude larger than the observed Higgs mass. To achieve the cancellation between the quadratically divergent corrections and the observed Higgs boson mass, an unnatural level of fine tuning is required. One possible solution is provided by SUSY where for each fermion a bosonic counterpart is predicted and vice versa. In case of unbroken SUSY the sensitivity to Λ_{UV}^2 between fermionic and bosonic partners would cancel. For broken SUSY Δm_H^2 can be approximated by

$$\Delta m_H^2 \approx |m_f^2 - m_S^2| \tag{3.2}$$

where m_f denotes the mass of the SM fermions and m_S the mass of the superpartner. In order to keep this correction reasonably small, the SUSY particles should not be to heavy.

- **Dark matter and dark energy:** Cosmic observations, e.g., of the cosmic microwave background, lead to the insight that just a small part of our universe ($\approx 5\,\%$) consists of baryonic matter. The rest splits into dark matter ($\approx 27\,\%$) and dark energy ($\approx 68\,\%$). Dark matter is commonly assumed to consist of massive particles, which interact weakly and via the gravitational force. Such dark matter candidates are provided by several BSM theories. Dark energy is responsible for the accelerated expansion of the universe. It is an hypothetical form of energy, which interacts only via the gravitational force.

- **Matter/Antimatter asymmetry:** Cosmology states that during the Big Bang matter and antimatter were produced in equal amounts. This assumption is in conflict with the obviously observed matter dominance in our universe. The SM could only explain a small asymmetry.

To find answers to these questions it is necessary to think of new theories beyond the Standard Model, e.g., supersymmetry or universal extra dimensions.

3.2 Supersymmetry

Supersymmetry is one theoretical candidate to solve some of the outstanding problems of the SM. SUSY is a symmetry between fermions and bosons, which predicts a supersymmetric partner for each SM particle [17]. These supersymmetric particles, commonly referred to as sparticles or superpartners, arise from a symmetry between fermionic and bosonic fields. This transformation can be written as

$$Q|\text{Fermion}\rangle = |\text{Boson}\rangle, \quad Q|\text{Boson}\rangle = |\text{Fermion}\rangle \tag{3.3}$$

where Q is an anti-commuting fermionic operator.

The SM particles and their superpartners can be arranged in supermultiplets which are irreducible representations of the SUSY algebra. The naming convention for the SM fermion superpartners is to add a prefix ‘s’ to the fermion name (e.g., squark, slepton) while for the SM boson superpartners the suffix ‘ino’ is added to the SM name (e.g., gaugino, higgsino). The SM particles and their superpartners have identical quantum numbers, except for the spin. If SUSY were an exact symmetry it would predict also identical masses for SM particles and their superpartners, but if this prediction were realized in nature the sparticles would have been already detected by the experiments. Therefore, SUSY must be a broken symmetry which allows considerably heavier superpartners.

The most basic realisation of Supersymmetry is the Minimal Supersymmetric Standard Model (MSSM) [18].

The MSSM is based on the Standard Model gauge group $SU(3)_C \times SU(2)_L \times U(1)_Y$. The particle content and the interactions of the MSSM are kept as small as possible. It consists of three families for each quark and each lepton supermultiplet (Tab. 3.1). Like in the SM, there are no right-handed neutrinos. To get rid of possible gauge

Names	mass eigenstates ($P_R = +1$)	spin	mass eigenstates ($P_R = -1$)	spin
quarks, squarks	$u_{L,R}$, $d_{L,R}$, $c_{L,R}$, $s_{L,R}$, $t_{L,R}$, $b_{L,R}$	$\frac{1}{2}$	$\tilde{u}_{L,R}$, $\tilde{d}_{L,R}$, $\tilde{c}_{L,R}$, $\tilde{s}_{L,R}$, $\tilde{t}_{1,2}$, $\tilde{b}_{1,2}$	0
leptons, sleptons	$e_{L,R}$, ν_{e_L}, $\mu_{L,R}$, ν_{μ_L}, $\tau_{L,R}$, ν_{τ_L}	$\frac{1}{2}$	$\tilde{e}_{L,R}$, $\tilde{\nu}_{e_L}$, $\tilde{\mu}_{L,R}$, $\tilde{\nu}_{\mu_L}$, $\tilde{\tau}_{L,R}$, $\tilde{\nu}_{\tau_L}$	0
Gluons, gluinos	G^A	1	$\tilde{g}^A$	$\frac{1}{2}$
bosons (H, W, B)	h^0, H^0, A^0, $H^{\pm}$, $W^{\pm}$, Z^0	$0, 1$	$\tilde{\chi}_1^0$, $\tilde{\chi}_2^0$, $\tilde{\chi}_3^0$, $\tilde{\chi}_4^0$, $\tilde{\chi}_1^{\pm}$, $\tilde{\chi}_2^{\pm}$	$\frac{1}{2}$

Table 3.1: MSSM particle spectrum.

anomalies of the chiral superfields, it is necessary to introduce two scalar Higgs fields $H_d = \binom{H_d^0}{H_d^-}$ and $H_u = \binom{H_u^+}{H_u^0}$ and their supersymmetric partners the higgsinos. The two scalar Higgs fields yield five physical Higgs bosons (h^0, H^0, A^0, $H^{\pm}$) after electroweak symmetry breaking. Deduced from the MSSM Lagrangian [17], the H_u gives mass to up-type quarks while H_d gives mass to down-type quarks and charged leptons. The vector bosons of the SM have fermionic superpartners, the gauginos (i.e. gluino, winos, bino). The winos and bino will mix with the higgsinos to charginos and neutralinos. The superpartners for the gluons are gluinos, which are color octet fermions. Therefore, the gluinos cannot mix with other MSSM particles, since there are no other color octet fermions.

3.2.1 R-Parity

In the MSSM, the lepton and baryon number is not conserved self-consistently by the Lagrangian as it is in the SM. To assure lepton and baryon number conservation, R-parity is introduced. It is defined as

$$P_R = (-1)^{3(B-L)+2s} \tag{3.4}$$

where B is the baryon number, L the total lepton number and s the spin quantum number of the particle. SM particles have even R-parity ($P_R = +1$), while their superpartners have odd R-parity ($P_R = -1$). Therefore, the SUSY particles are always produced in pairs and their decay products involve always an odd number of SUSY particles. The LSP (lightest supersymmetric particle) cannot decay further, hence it is stable. In order to make the LSP a dark matter candidate, it is assumed that the LSP is electrically neutral and therefore interacts only weakly with ordinary matter.

3.2.2 Soft-SUSY Breaking

The effective Lagrangian of the MSSM can be written generally as

$$\mathcal{L} = \mathcal{L}_{SUSY} + \mathcal{L}_{soft} \tag{3.5}$$

where $\mathcal{L}_{SUSY}$ denotes the part which is invariant under SUSY transformations and $\mathcal{L}_{soft}$ comprises all contributions that break SUSY. $\mathcal{L}_{SUSY}$ contains all gauge and Yukawa interactions among the supermultiplets. This part has 19 parameters, like the SM Lagrangian. It is assumed that SUSY is broken spontaneously like the electroweak symmetry. However, the nature of SUSY breaking is unknown. It is assumed that SUSY breaking is an effect of physics at high energy scale. As mentioned before, the effects for SUSY breaking are parametrized in $\mathcal{L}_{soft}$. Soft symmetry breaking means that the symmetry is broken, but the reappearance of the quadratic divergences is prevented. A total of 105 parameters is introduced by soft SUSY breaking in form of terms in the $\mathcal{L}_{soft}$. These terms are mass terms for gluinos, winos, binos and scalar fermions, mass and bilinear terms for the Higgs bosons and terms for the trilinear coupling between sfermions and Higgs bosons. Concrete models of SUSY breaking provide relations for the soft-breaking terms, thus it is possible to construct models with just a few parameters, e.g., the constraint MSSM (compare Sec. 3.2.4) and the phenomenological MSSM (compare Sec. 3.2.5).

3.2.3 Naturalness

One of the main reasons why it is expected to find SUSY signatures at the LHC is that SUSY provides an elegant solution for the hierarchy problem. The Higgs mass is stabilized if quadratically loop corrections among the superpartners almost cancel. Therefore, the masses of the superpartners are required to be relatively small in order to prevent any fine tuning of the parameters. Natural SUSY [19], for $\Delta m_H^2 \lesssim m_H^2$, implicates that no additional fine tuning is necessary. The requirements for naturalness can be summarized by following tree-level relation in the MSSM,

$$-\frac{m_Z^2}{2} = |\mu|^2 + m_{H_u}^2. \tag{3.6}$$

The left-hand side of this equation only depends on the mass of the Z-boson m_Z. The right-hand side is defined by two SUSY mass parameters μ and m_{H_u}. The squared Higgs boson mass at tree level mainly depends on $|\mu|^2$. Therefore, all superpartners which have the strongest coupling to the Higgs boson must not be to far above the electroweak energy scale. In particular, the higgsinos should not be too heavy because they mainly depend on $|\mu|^2$. Similarly, $m^2_{H_u}$ depends on the stop and gluino masses, which are also not allowed to be too heavy in case of 'natural' SUSY. All other superpartners, including the first two generations of squarks, do not receive any limitations on their masses. Rough requirements for a natural SUSY spectrum are the existence of two stops and one left-handed sbottom below 700 GeV, two higgsino like charginos and/or neutralinos below 350 GeV and a gluino below 1.5 TeV.

3.2.4 Constrained MSSM

The MSSM Lagrangian has in total 124 parameters, where 105 parameters are introduced by the soft SUSY breaking mechanism. This large number of parameters makes it almost impossible to make phenomenological predictions for this theory. Therefore, simplifying assumptions about the SUSY breaking are introduced to the MSSM. One popular simplification is the constrained MSSM (cMSSM) [17]. This model assumes that the SUSY breaking occurs because of flavor blind gravitational interactions from a hidden sector at the Planck scale. The 105 new parameters in $\mathcal{L}_{soft}$ are reduced to 4 parameters and the choice of one sign:

- $m_{1/2}$, a common gaugino mass
- m_0, a common scalar mass
- $\tan\beta = \frac{v_u}{v_d}$, the ratio between the Higgs field VEVs
- A_0, a trilinear coupling parameter
- the sign of the Higgs boson mass parameter μ

The Higgs boson mass parameter μ is determined, except for its sign, by the requirement that the electroweak symmetry is broken correctly. These boundary conditions for the soft SUSY terms are given at the GUT scale. The masses at the electroweak scale are computed from renormalization group (RG) equations (Fig. 3.2).

The cMSSM was intensively used for first SUSY searches at the LHC. The main advantage of the cMSSM is its simplicity, which as well is its main disadvantage. It might includes too strict constraints, thus it is blind to big part of parameter space.

One extension of the cMSSM is the NUHM (Non Universal Higgs Mass) model.

3.2.5 Phenomenological MSSM

An other approach is to constrain the MSSM with phenomenological constraints without assuming any particular SUSY breaking meachanism. The resulting model

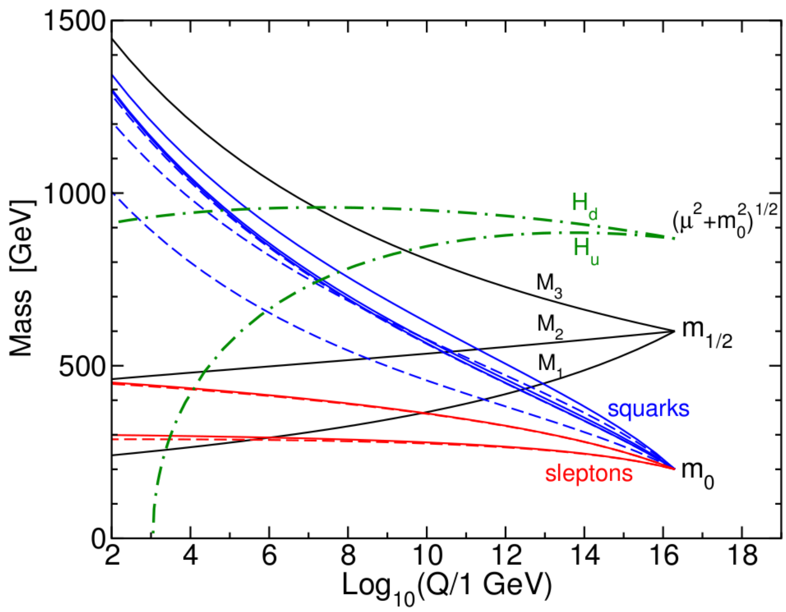


Figure 3.2: Evolution of the RG equations of scalar and gaugino mass parameters in the cMSSM [17]. The $\mu^2 + m_{H_u}^2$ runs negative at the electroweak energy scale resulting in electroweak symmetry breaking. The parameter values are $m_0 = 200\,\text{GeV}$, $m_{\frac{1}{2}} = -A_0 = 600\,\text{GeV}$, $\tan\beta = 10$ and $\mu > 0$. Q denotes the inverse gauge couplings α^{-1}.

is called phenomenological MSSM (pMSSM) [18]. The pMSSM is mainly based on three assumptions.

The first requirement is to prevent new sources of CP-violation. Experimental results constrain new possible sources of CP-violation. These experimental limits result especially from the electron and neutron electric moments and K system measurements. New sources of CP-violation are eliminated by assuming that all phases in the soft-SUSY potential are zero.

The second assumption is that no flavor changing neutral currents (FCNC) are allowed. The FCNCs are constrained by experiments. In $\mathcal{L}_{soft}$ they are introduced by the non-diagonal matrix elements in the sfermion mass matrices and the trilinear coupling matrices. Hence, the sfermion matrices and trilinear coupling matrices are assumed to be diagonal.

The third requirement for the pMSSM is first and second generation universality for squarks and sleptons. Experimental data predict a limited mass splitting between first and second generation squarks unless the squarks are significantly heavier than 1 TeV. Therefore, the masses of the first and second generation squarks can be assumed to be equal. The third generation squark masses are much less constrained by experiments. Furthermore, the trilinear couplings for the first and second generation squarks can be set to zero, since the trilinear couplings affect only the third generation squarks.

These assumptions lead to the following 19 real, weak-scale SUSY Lagrangian paramaters in addition to the SM parameters:

- the gaugino mass parameters M_1, M_2, and M_3
- the ratio of the Higgs vacuum expectation values (VEV) $\tan\beta = v_2/v_1$
- the higgsino mass parameter μ and the pseudo-scalar Higgs mass m_A
- 10 sfermion mass parameters $m_{\tilde{F}}$, where $\tilde{F} = \tilde{Q}_1, \tilde{U}_1, \tilde{D}_1, \tilde{L}_1, \tilde{E}_1, \tilde{Q}_3, \tilde{U}_3, \tilde{D}_3,$ $\tilde{L}_3, \tilde{E}_3$ (imposing $m_{\tilde{Q}_1} \equiv m_{\tilde{Q}_2}$, $m_{\tilde{L}_1} \equiv m_{\tilde{L}_2}$, etc.), and
- 3 trilinear couplings A_t, A_b and A_τ

3.3 SUSY Searches at the LHC

It is very likely that the experiments at the LHC discover SUSY, if 'natural' SUSY is realized in Nature. Assuming that gluinos and squarks have masses around 1 TeV they should be frequently produced. The most important production modes for this scenario would be gluino pair production ($pp \to \tilde{g}\tilde{g}$), squark pair production ($pp \to \tilde{q}\tilde{q}$) and gluino squark associated production ($pp \to \tilde{g}\tilde{q}$).

If squarks and gluinos are too heavy, the dominant production would be the electroweak production of charginos and neutralinos. The main production modes would be $pp \to \tilde{\chi}_i^+\tilde{\chi}_j^-$ and $pp \to \tilde{\chi}_i^\pm\tilde{\chi}_k^0$ with $i, j = 1, 2$ and $k = 1, \ldots, 4$.

To know which signatures one has to expect at the LHC, it is essential to know how the sparticles decay. At tree level, gluinos only decay into squarks, because gluinos only interact via the strong force. These squarks can either be on- or off-shell squarks. The dominating mode depends on the mass of the squarks. If the squarks are lighter than the gluinos, the main decay mode is the two-body decay $\tilde{g} \to q\tilde{q}$. Stops and sbottoms are often lighter than the other squarks. Hence, the gluino decay into sbottoms or stops is significant. In case the squarks are heavier than the gluinos the three-body decays via off-shell squarks ($\tilde{g} \to q\bar{q}'\tilde{\chi}_i^\pm$ and $\tilde{g} \to q\bar{q}\tilde{\chi}_k^0$) dominate.

Squarks decay into gluinos plus jets ($\tilde{q} \to q\tilde{g}$), if gluinos are lighter than squarks. Otherwise, squarks decay into quarks plus neutralinos or charginos ($\tilde{q} \to q\tilde{\chi}$). Since the light quark flavours only interact via gauge bosons, the decays into gauginos are preferred. Therefore, the dominant decay modes are decays into $\tilde{\chi}_1^0$, $\tilde{\chi}_2^0$ and $\tilde{\chi}_1^\pm$. Stops and sbottoms couple as well to the higgsino components of the neutralinos and the charginos, because of their additional strong Yukawa coupling. Hence, stops and sbottoms can also decay into each other, if the mass splitting is large enough.

Sleptons decay into a lepton plus neutralino or chargino ($\tilde{l}^{\pm} \to l^{\pm}\tilde{\chi}_k^0$, $\tilde{l}^{\pm} \to \nu\tilde{\chi}_i^{\pm}$, $\tilde{\nu} \to \nu\tilde{\chi}_k^0$, $\tilde{\nu} \to l^{\pm}\tilde{\chi}_i^{\mp}$). The direct decays into the LSP are practically always open.

Charginos and neutralinos have various decay modes. These decay modes depend on the mass difference between the charginos and neutralinos, the gaugino-higgsino mixing and the mass pattern of the other sparticles. Neutralinos and charginos can decay into lighter neutralinos or charginos paired with an SM boson or into a slepton-lepton pair. Decays into squark-quark pairs are also allowed, but less likely.

The SUSY signatures expected in the detector result directly from the possible decay chains of the sparticles. Since the $\tilde{\chi}_1^0$ is considered to be the LSP, SUSY searches look for large missing transverse energy $\not{E}_T$. Besides the large $\not{E}_T$ multiple hard jets and maybe some leptons are expected. In case of lepton pairs they either are same-sign or opposite-sign di-leptons. Many SUSY searches can be classified according to their lepton multiplicity.

3.4 Simplified Model Spectra

The main problem searching for BSM models is that they usually have too many free parameters making the analysis computationally infeasible. Constraining the BSM parameters to a small set might constrain the theory too much. As a solution simplified model spectra (SMS) were proposed. The idea of simplified model spectra is to decompose the full theoretical model into smaller generic descriptions, so-called topologies, to make data analysis manageable [20, 21, 22]. Simplifed model spectra constitute an intermediate stage between LHC results and the Lagrangian (Fig. 3.3). They are limits of more general new physics scenarios. An effective Lagrangian defines the SMS describing the interactions of a small number of new particles. Hence, an equivalent description of new physics processes is given by the appearing masses, cross sections and branching ratios. These parameters are directly related to physics observables. Therefore, SMS are a particular effective framework for evaluating searches and a good starting point for characterizing positive signals of new physics.

The main applications for SMS are the identification of the boundaries of search sensitivity, to derive limits on more general models and the characterization of new physics signals if any appear.

In this thesis the following CMS naming convention for the simplified models [23] is used. Each name start with T for topology. The next character defines the

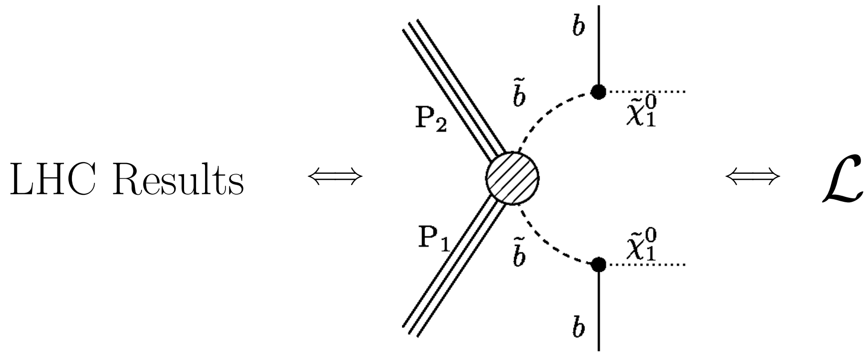


Figure 3.3: Simplified model spectra introduce an interface between the results obtained by different searches at the LHC and BSM theories.

mother sparticles produced in the pp-collision. In the `SModelS` code only sparticle pair production is possible, since it originally is written for interpretation of R-parity conserving SUSY analyses and a production of two sparticles is the simplest realization. The following characters are possible after the T:

- **Odd numbers:** Odd numbers stand for initially produced gluinos. The number itself specifies how many decays follow afterwards. The most simple case is *1*, which means each gluino decays directly into two quarks and an LSP. *3* would mean that one gluino decays directly into quarks and an LSP while the other one has two decay vertices meaning it decays into some SM particles and any other lighter sparticle, which afterwards decays into the LSP and some other SM particle. If both gluinos have two decay vertices, it is marked by the number *5*. Every decay chain which is more complicated, simply is marked by *7*.

- **Even numbers:** Even numbers stand for initial squark pair production. The most simple realization is denoted by *2* where both squarks decay directly into LSP and SM particles. Like for odd numbers more complex even numbers mean a more complex simplified model. So *4* stands for one squark decaying directly into an LSP while the other one decays first into a lighter sparticle which decays into an LSP. Also *6* means the same for squarks like *5* for gluinos and all more complex squark decays are denoted by *8*.

- **Chi. . . :** If the pp-collision initially produces charginos or neutralinos it is marked by *Chi. . .*. In this convention neutralinos are named simply *Chi* while for charginos a *p* for positive charge or/and an *m* for negative charge is added. Therefore, if it is a positive charged chargino it is named *Chip*, a negative one *Chim*. In case the sign does not matter it is called *Chipm*. In contrast to

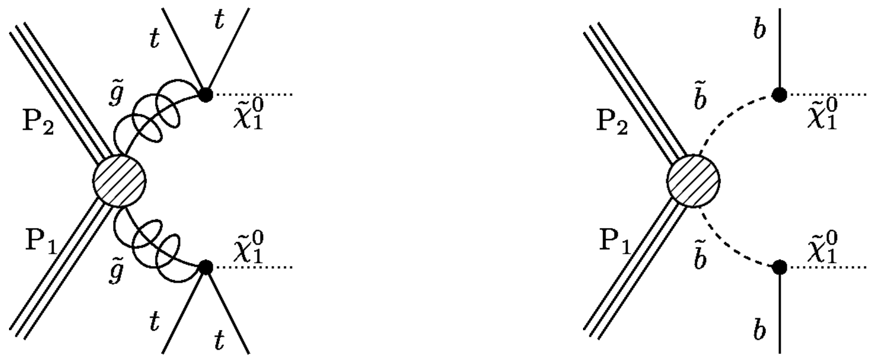


Figure 3.4: Feynman like diagram of two different simplified models. left: An initially produced gluino pair decays via top quarks into LSP (*T1tttt*). right: An initially produced sbottom pair decays via bottom quarks into LSP (*T2bb*).

squark or gluino production, which are marked by one number, for chargino or neutralino production each of the two initial particles has to be denoted separately. So if, e.g., one chargino and one neutralino is produced the simplified model will be named *TChiChipm*. . . .

- **Slep/Snu:** For initial slepton production the prefex *Slep* is used and for a sneutralino *Snu*. Like for neutralino and chargino production, each of the two sleptons of a slepton pair production is written down separately *TSlepSlep*. . . . The same holds for sneutralinos or a sneutralino slepton pair.

The following letters denote the produced SM particles, e.g., *T1tttt* means a gluino pair production where each gluino decays into two top quarks and one LSP (Fig. 3.4 left). Another simple example is *T2bb*, which denotes two initially produced sbottoms where each decays into a bottom quark and an LSP (Fig. 3.4 right). A full list of all simplified models which are currently available in the database and their corresponding Feynman-like diagrams is given in Appendix A.

The `SModelS` framework is based on the fact that the SUSY analyses at CMS and ATLAS use SMS to interpret their results. Using the SMS interpretation, these results can also be mapped to other BSM theories, which give rise to the same SMS topologies. Hence, results inspired by specific SUSY scenarios can be used to make statements on other SUSY scenarios or other BSM theories. This is done by `SModelS`. `SModelS` is a tool that decomposes a full BSM spectrum into its SMS topologies and compares these results to the experimental results.

4 Software Tools

To simulate a full SUSY event many different calculations need to be done. In Fig. 4.1 a typical program flow for the simulation of SUSY events is shown.

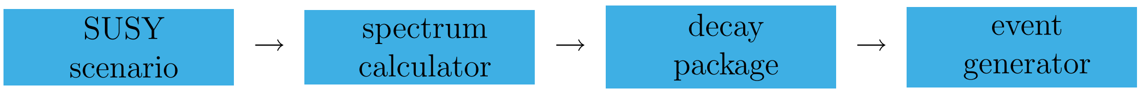


Figure 4.1: Program flow for the simulation of a SUSY event.

For each of these steps many public software packages exist currently. For the `SModelS` decomposition scheme only informations about the mass spectrum, branching ratios and production cross section are necessary. Therefore, in the `SModelS` framework and for the preparation of the scan some already existing software tools are used.

For the work presented in this thesis the following public software packages are used:

- SOFTSUSY [24] to calculate the sparticle spectra,
- SDECAY [25] to compute sparticle decays,
- PYTHIA [26] to generate events,
- PROSPINO [27, 28] and NLLfast [29] to calculate higher order corrections for the production cross section.

4.1 SLHA and LHE Files

In the `SModelS` framework two file formats are used as input and output files for the different decomposition steps: SLHA [30] and LHE [31]. These file standards were designed in order to have a standardized interface between different programs for different simulation steps of a SUSY event.

The standard SLHA file consists of the input parameters, the mass spectrum, the mixing matrices and the decay tables. In the decay tables all allowed particle decays are stored including the decay width and branching ratios. In the `SModelS` framework a new section XSECTION was introduced, which will be added soon to the official SLHA standard. The XSECTION block contains the production cross section calculated up to different orders for all produced sparticle pairs.

In contrast to an SLHA file, an LHE file contains full events computed, e.g., by PYTHIA from the information provided by an SLHA file. Each event lists its decay chain and for each particle in this chain the energy, the momentum and other parameters are listed as well.

The different information contained in SLHA and LHE files motivates different decomposition strategies (Chap. 5.3.2). At the moment SLHA files are used to run the `SModelS` code. In the current version of `SModelS` only the SLHA decomposition is validated, the LHE decomposition code exists as a beta version. As soon as the LHE decomposition is validated as well, the user will have opportunity to choose between LHE and SLHA decomposition.

4.2 SOFTSUSY

Given an MSSM model, SOFTSUSY [24] calculates the sparticle spectra with full flavor mixing structure within CP-conserving MSSM by solving the renormalization group equations. The theoretical constraints on soft SUSY breaking to be used are provided by the user. Boundary conditions are weak-scale mass coupling, fermion mass data and electroweak symmetry breaking.

For the work presented in this thesis it was used as a first step in order to create SLHA files for randomly selected pMSSM parameters points for the scan. The obtained sparticle spectra and elements of the mixing matrices are passed to SDECAY.

4.3 SDECAY

SDECAY [25] is a FORTRAN code, which calculates decay widths and branching ratios of SUSY particles within the MSSM by evaluating the various couplings of the sparticles and the MSSM Higgs bosons. The calculations include higher order effects as well. For these calculations SDECAY needs the particle spectrum and the soft SUSY breaking parameters in SLHA format.

The output from SDECAY are complete pMSSM SLHA files with decay tables for each mass spectrum calculated by SOFTSUSY.

4.4 PYTHIA

PYTHIA [26] is an event generator for high energy physics specialized on multiparticle production in collisions between particles, e.g., e^+e^-, pp, pe. The generated events should as closely as possible resemble the ones observed by a 'perfect' detector. This is done by factorizing the problem into small manageable components. First, PYTHIA calculates the production process in leading order (LO), e.g., sparticle pair production. The next step is to simulate the decay processes and thereby completing the description of the hard scatter. This typically includes the decay of heavy unstable particles as the top quark and the heavy vector bosons. The resulting decay products (light quarks and leptons) are decayed further through showering and usually this step is followed by a detector simulation. To obtain the same behavior and fluctuations as in real data, in generators Monte Carlo techniques are used.

In the `SModelS` framework PYTHIA is used to simulate pp collisions to get the LO production cross sections, which are needed to calculate the weight for each event.

4.5 PROSPINO

PROSPINO (a programm for the PROduction of Supersymmetric Particles In Next-to-leading Order QCD) [27] is a program to calculate the production cross sections of squarks and gluinos at hadron colliders in LO and NLO (next-to-leading order). The actual version of PROSPINO, PROSPINO2 [28], includes also stop, neturalino/chargino and slepton pair production and associated production of squarks with gluinos and of neutralinos/charginos with gluinos or squarks. PROSPINO is able to calculate the differential cross section in the transverse momentum p_T and the rapidity as well as the total cross sections with possible cuts in p_T and rapidity. The user can choose the squark, gluino and stop masses, the renormalization/factorization scale, the collider type and the set of parton density functions. These parameters are passed to PROSPINO in an SLHA file.

4.6 NLL-fast

NLL-fast [29] calculates production cross sections including the next-to-leading order supersymmetric QCD corrections. For squark and gluino production cross sections also the resummation of soft gluon emission at next-to-leading logarithmic (NLL) accuracy is included. The program uses a fast interpolation routine for these calculations. The values to interpolate from are read out from grid files providing NLO and NLO+NLL results, where the NLO results are taken from PROSPINO.

In the `SModelS` framework NLL-fast is used to extend the production cross sections given by PYTHIA to NLO+NLL, so that more accurate weights can be obtained.

5 SModelS

`SModelS` is a software package with the basic idea to make maximum use of existing SMS results of ATLAS and CMS SUSY searches. In this framework BSM model points have to be decomposed in simplified model topologies and their predicted weights (production cross section times branching ratio $\sigma \times \mathcal{B}$) to be compared against the experimental SMS results. Therefore, a database was created, which contains all the available experimental results of ATLAS and CMS SUSY searches. The experimental results, which are used for the comparison, are upper limits on the production cross section (assuming a branching ratio of 1) for one SMS topology considering different sparticle masses.

5.1 General Concepts and Objects

The `SModelS` decomposition is possible because most of the CMS and ATLAS searches for BSM physics are interpreted in the context of simplified models. In the `SModelS` approach, we assume these SMS are insensitive to various specific details of the BSM model. The only requirement is that the BSM analyses give rise to the same signal topology and feature similar event kinematics. Therefore, in a first approximation the properties of the BSM model can be reduced to their mass spectrum, production cross section and decay branching ratios. Using these assumptions the full model can be decomposed into series of independent SMS with corresponding specific weights. The `SModelS` framework is divided into two parts: the experimental and the theoretical side. On the experimental side a key issue is the database, which contains the analyses results from ATLAS and CMS, and its querying. The theoretical part has many tasks to fulfill: the computation of cross sections and branching ratios, the decomposition of a full BSM model into its SMS, parsing of the `SModelS` description of the experimental results and merging the SMS to match the description.

The working principle of the `SModelS` framework is sketched in Fig. 5.1. Starting at the theory side, in order to cast the theory in a largely model-independent way, the full BSM spectrum has to be decomposed into SMS topologies. That means

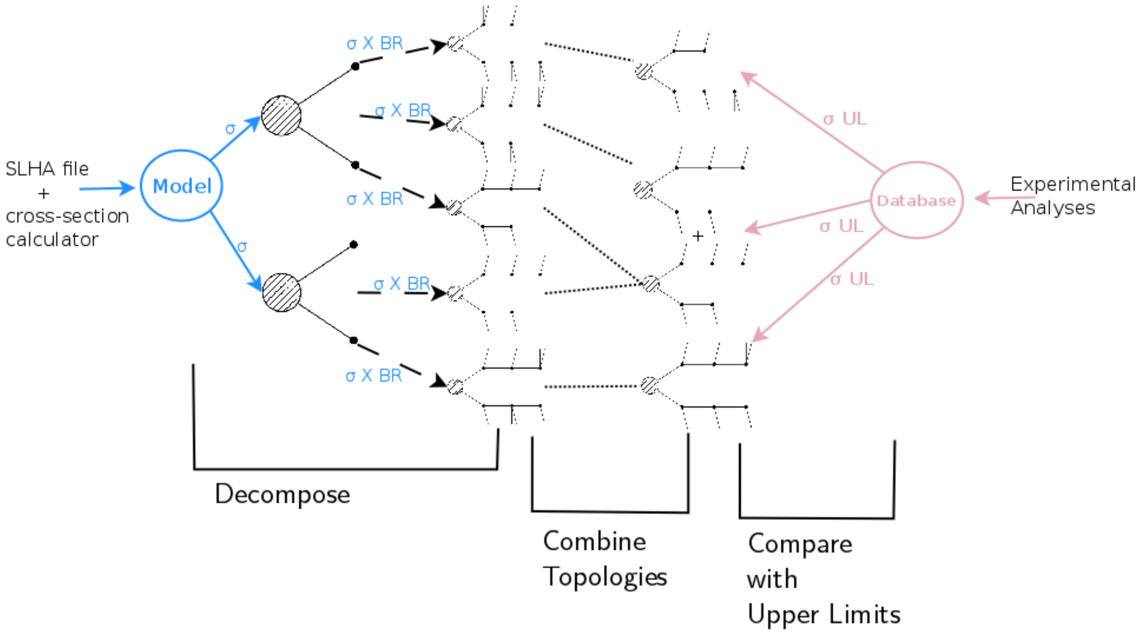


Figure 5.1: Schematic view of the `SModelS` working principle [3].

that all signal topologies appearing in the full model and their respective weights have to be computed. In the `SModelS` framework only models with $\mathbb{Z}_2$-symmetry are considered, therefore, all SMS topologies will always originate from pair production of new $\mathbb{Z}_2$-odd particles. These new BSM particles decay as $P \to P' + SM$ where P and P' are mother and daughter BSM particles and SM stands for the SM decay products. `SModelS` only handles on-shell particles, so all particles appearing are on-shell. Off-shell decays are introduced to `SModelS` as 3-body decays with no mention of the off-shell states. Hence, the information given by an SMS diagram can be reduced to three main objects:

- the topology of the diagram (vertices and SM particles, Fig. 5.2),
- the masses of the BSM particles,
- the weight of diagram.

After the decomposition is done, the SMS comprises no longer any reference to any specific details of the full model. Next, the SMS and their computed weights need to be mapped to the experimental analyses. In case the analysis constrains only one SMS and provides the upper limit on the production cross section as a function of the BSM mass vector, the mapping is straightforward, e.g., searches for *T1* and *T2* topologies. It is, however, more common for analyses to constrain a sum of several topologies, assuming a specific relative contribution from each of them. One example is slepton pair production, where the search sums over final states with electrons or

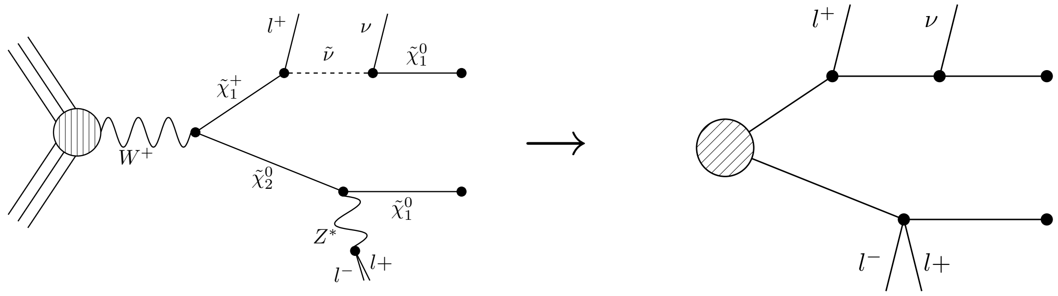


Figure 5.2: left: SMS, right: topology of SMS used in the `SModelS` framework [3].

muons considering both flavors contributing equally. In such cases it is necessary to combine all SMS from the decomposition with a single lepton being emitted at each branch and which have the same mass vector, in order to correctly compare the theoretical prediction to the experimental results. It is also required to verify the conditions of the analysis. For the example with the slepton pair production, this means that one must check that e and μ contribute equally. Once all SMS are combined according to the assumptions of the experimental analyses, the theoretical prediction of the topology weight ($\sigma \times \mathcal{B}$) is compared to the experimental upper limit to decide whether the particular parameter point is allowed or excluded.

5.1.1 SModelS Formalism

To be able to fully describe the experimental results within the `SModelS` framework, a formal language has been introduced. In this language it takes only two terms to characterize a result completely: constraints and conditions. A constraint, as the name implicates, constrains a part of a fundamental theory to which the experimental result is applicable. A condition describes the additional assumptions made by an analysis in order to make the experimental result applicable to a specific constraint.

Constraints

Starting from the assumption that only mass and weight of the BSM particles describe the type and the kinematics of the observed SM particles produced by an BSM event leads to the concept of constraints. Taking a simplified model, only the SM particles, the masses of the sparticles and the position of the SM particles and the sparticles in the decay chain is of interest. Therefore, the simplified model can be written as a string containing two branches b1 and b2 [[b1],[b2]] and at each branch different vertices v1, v2,. . . where the BSM particles decay into SM particles and BSM particles [[[v1],[v2]],[[v3]]] (Fig 5.3). The type of the BSM particle in the decay chain is irrelevant, hence, if two topologies have the same SM particles at the same places in the decay chain, they are considered as identical in the `SModelS` language.

Conditions

The `SModelS` framework has to ensure that all additional assumptions made by the experimental analyses also are made by the fundamental theory. This is necessary for the experimental result to be applicable in the `SModelS` framework. One example is flavor-democracy in the dilepton channel. In case a certain fundamental theory predicts more τ leptons than electrons and muons the experimental limit is not applicable for this theory because flavor-democracy is violated. In `SModelS` language the condition for flavor-democracy in the second branch is written as [[[nu],[tau]],[[L],[L]]] $>$ 3[[[nu],[tau]],[[tau],[tau]]].

These conditions in addition to the constraint fully describe a SMS result in the `SModelS` language.

5.1.2 Objects used in SModelS

During a full `SModelS` run, different tasks, like decomposition, mass clustering, invisible and mass compression, must be done. The `SModelS` program package is written in an object-oriented way using Python 2.7. The `SModelS` framework consists of three packages (*Experiment*, *Theory*, *Tools*) containing many modules. A list of all modules and their functions is given in Appendix B.

How the concepts of `SModelS` are transfered into Python code is shown in Fig. 5.4. First, *XSecComputer.compute()* calculates the weights of all possible SMS for the specific BSM model point and creates a *CrossSection* object to store this information. As soon as the $\sigma \times \mathcal{B}$ are known, a list containing all signal topologies is created by the decomposition. In parallel, a list of all SMS topologies from the experimental results is created using *SMSAnalysisFactory.load()*. This list consists of partly

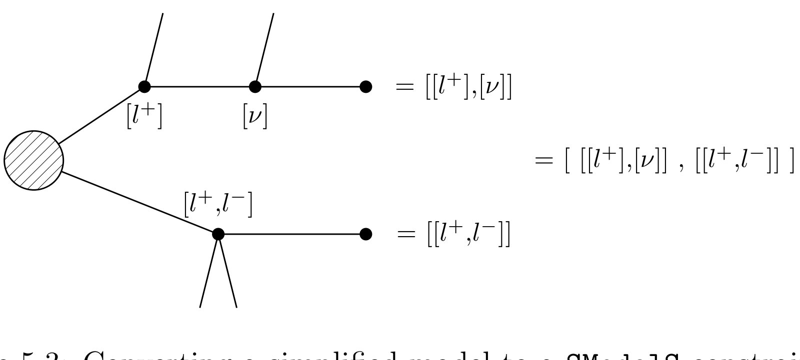


Figure 5.3: Converting a simplified model to a `SModelS` constraint [3].

empty *EAnalysis* objects. Using *EAnalysis.add()* the SMS topologies created by the decomposition are added to the list of *EAnalysis* objects, which hold the experimental SMS. Finally, the function *EAnalysis.computeTheoryPredictions()* first performs the mass clustering and afterwards computes the theoretical predictions for $\sigma \times \mathcal{B}$.

Listing 5.1 shows a simple Python script doing the decomposition, loading the experimental data, doing the clustering and calculating the theoretical predictions. Hence, there are four main methods in the `SModelS` framework: *SMSAnalysisFactory.load()*, *SLHADecomposer.decompose()*, *Analysis.add()* and *Analysis.computeTheoryPredictions()*.

Listing 5.1: A simple program running the whole `SModelS` framework

```
ListOfAnalyses = SMSAnalysisFactory.load()
ListOfTopologies = SLHADecomposer.decompose()
for EAnalysis in ListOfAnalyses:
    EAnalysis.add(ListOfTopologies)
    EAnalysis.computeTheoryPredictions()
```

To translate the `SModelS` concepts to Python code, following objects are used to handle all the information.

SMSAnalysis

The most fundamental object of the `SModelS` framework is the *EAnalysis* class of the *SMSAnalysis* module. It uses the classes from the *SMSDataObjects* and the *TheoryPrediction* module. The dependencies are sketched in Fig. 5.5. The *EAnalysis* class contains the basic information obtained from the database like the label, the center of mass energy, the luminosity, the run and the factor of mass compression, but also more complex objects which are not created from the beginning but constructed during a `SModelS` run like *ResultList*. A variety of complex functions is defined for the *EAnalysis* object (App. B.2.1).

At the beginning of a `SModelS` run the experimental results are loaded from the database into an *EAnalysis* object and the decomposed theory objects are added. Almost every step of comparing, compressing, clustering and finally calculating the theoretical prediction is done in the *EAnalysis* object.

SMSDataObjects

The *SMSDataObjects* module contains the main information about a SMS. It describes the topology using the *GTop* (general topology), *ATop* (analysis topology) or *CTop* (cluster topology) class. These classes contain three different lists: *vertnumb*, *vertparts* and *ElList*. Only the *CTop* has an additional entry for the cluster

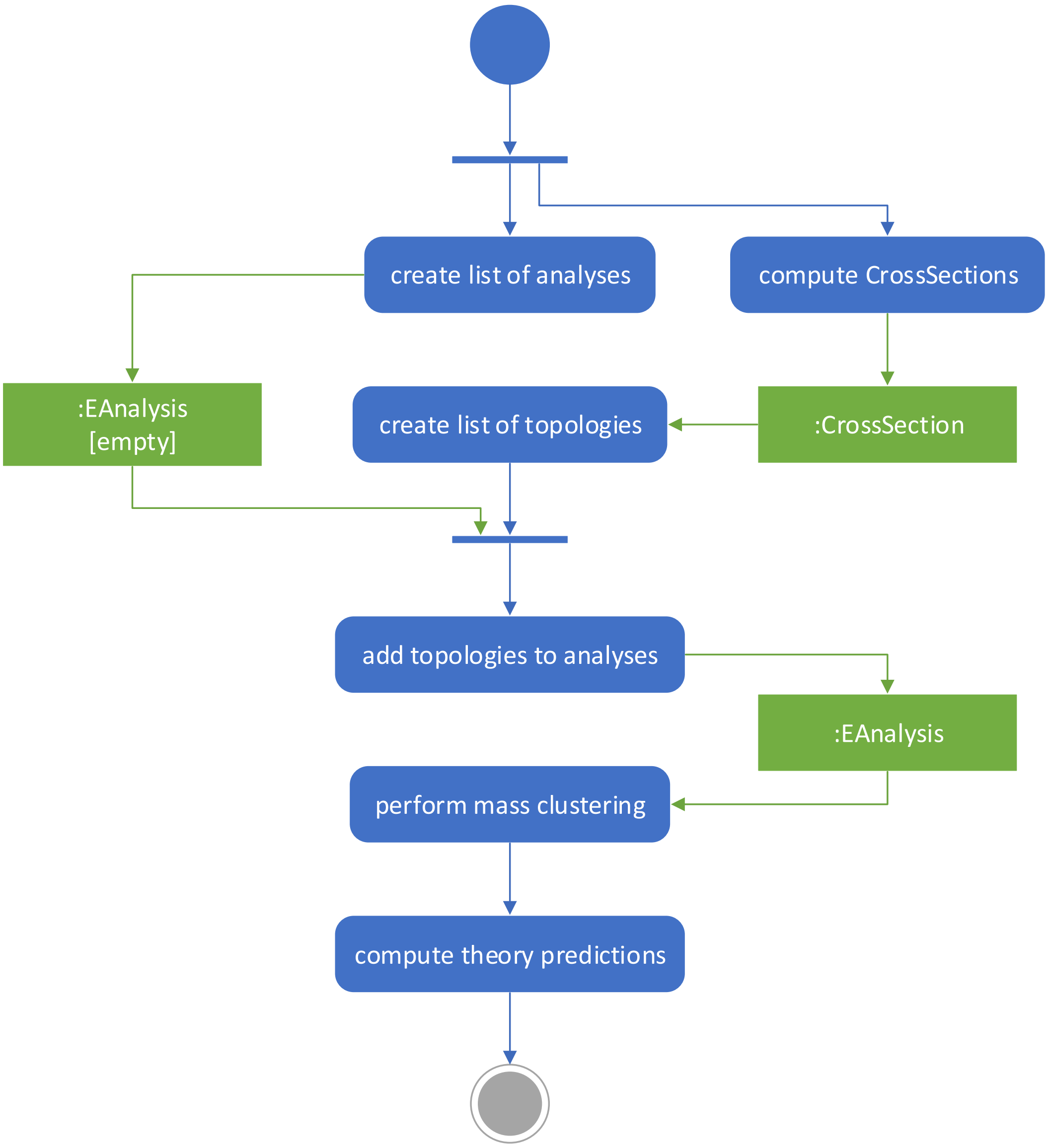


Figure 5.4: Activity diagram of the `SModelS` framework [32].

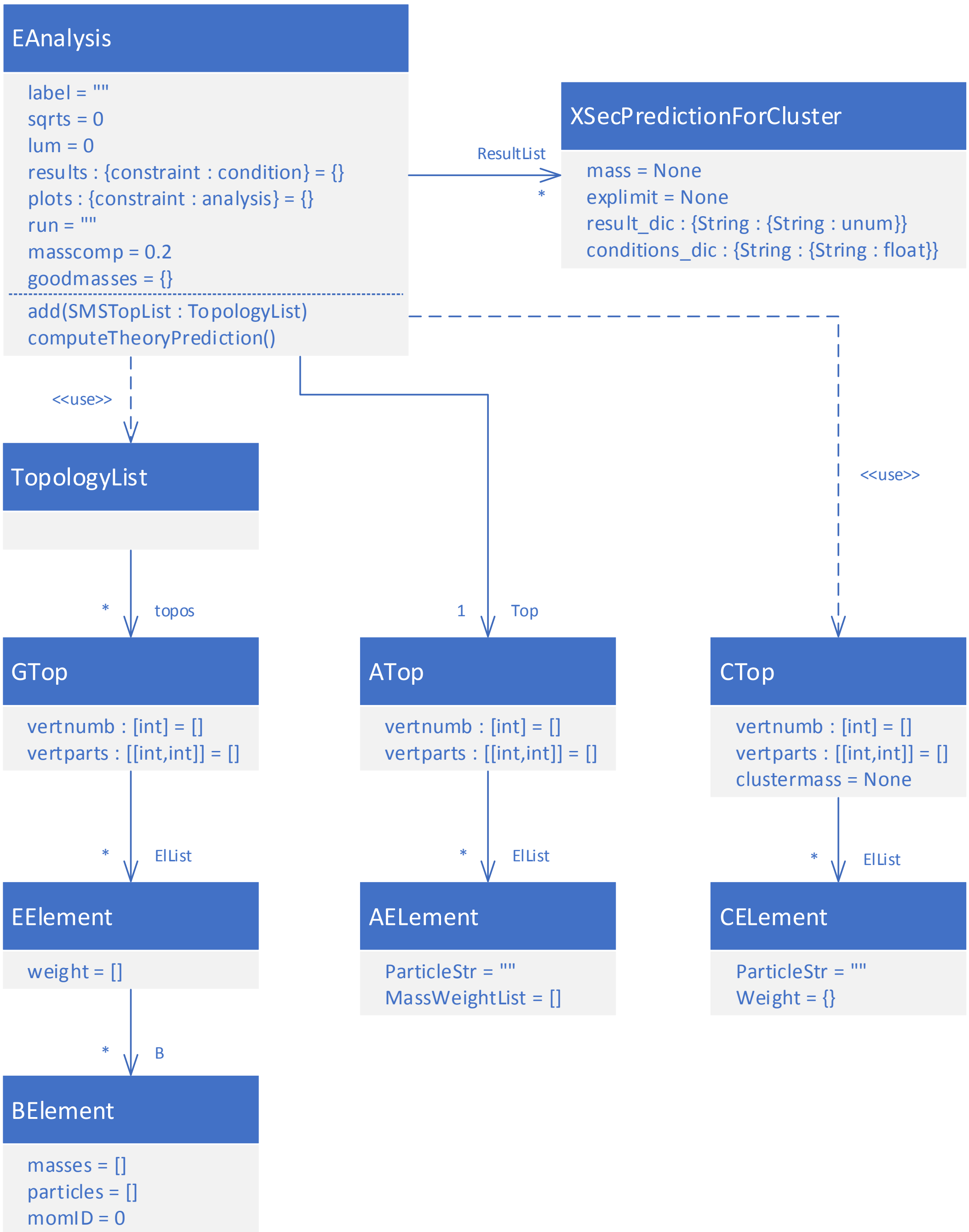


Figure 5.5: Class diagram of EAnalysis [32].

mass. The first list *vertnumb* lists the number of vertices at each branch while *vertparts* lists the number of SM particles at each vertex. The *ElList* comprises all *EElement* (event element), *AElement* (analysis element) or *CElement* (cluster element) belonging to the topology, where the different elements are classes as well. An *EElement* consists of a list of the weights of the SMS and a list, called *B*, where all the *BElement* (branch element) are listed. *BElement* is also a class having a list for the masses of all $\mathbb{Z}_2$-odd particles appearing in the SMS (= mass vector), a list containing all the SM particles at each vertex and an entry for the particle code according to the Particle Data Group (PDG) [33] convention (PDG id) of the mother partice. The *B* list of an *EElement* must have two entries if pair production is assumed. An *AElement* consists of the string *ParticleStr* listing all the particles using the same convention as the constraints and the *MassWeightList* list containing *MassWeight* objects. The *MassWeight* class has two entries: the mass vector and the weight of the SMS. Finally, the *CElement* contains the string *ParticleStr* and a dictionary with the weights of the SMS.

The class *TopologyList* contains a list of all available *GTop* called *topos*. Each of the different classes has many functions defined, which are listed in Appendix B.2.1.

The *SMSDataObjects* module is used by the *EAnalysis* class.

CrossSection

The *CrossSection* module consists of three different classes, which are all interdependent: *CrossSection*, *SingleXSecInfo* and *XSecInfoList*. These classes can be initialized at two different points of the code: Either, if the production cross section has to be computed, in the *XSecComputer.compute()* function or, if the SLHA files contain an XSECTION block, during the *SLHADecomposer.decompose()*.

The *CrossSection* object holds the production cross section of produced sparticle pairs at different orders (in the current version of `SModelS` only LO and NLO+NLL) and also their weight.

In a *SingleXSecInfo* object additional information like the center of mass energy and the order is stored. These objects are listed in a *XSecInfoList* object ordered by the center of mass energy in descending order.

TheoryPrediction

The final results are contained in the *XSecPredictionForCluster* class, which is added to *EAnalysis.ResultList*. It comprises the mass and the expected upper limit on the production cross section as well as two dictionaries: *results_dic* and *conditions_dic*.

In the *results_dic* for each constraint, order and center of mass energy the corresponding weight in fb is stored. The *conditions_dic* is similar to the *results_dic*, but instead of the constraint it contains the conditions.

5.2 Database of Experimental Results

5.2.1 Experimental Results from CMS and ATLAS

The ATLAS and CMS SUSY searches are interpreted in SMS. The `SModelS` database contains analyses, which use the following search strategies [23]:

- **All-hadronic:** This kind of analyses is interested in events with multiple jets and high missing transverse momentum ($\not{E}_T = ||p_T^{miss}||$), which is expected from gluino and squark production. To suppress the multiple-jet background different methods and observables are used. To reduce backgrounds from SM top and W boson production, isolated lepton events are vetoed.

- **All-hadronic events with b-jets:** These searches are motivated by the idea that sbottom or stop squarks are possibly lighter than other squarks. Hence, it is searched for events with multiple jets, from which at least one must be b-tagged, and large $\not{E}_T$. In order to suppress the background, isolated leptons are vetoed.

- **Single lepton:** Events with a single isolated lepton are selected. The observed lepton transverse momentum (p_T) spectrum and other control samples are used to predict the $\not{E}_T$ spectrum. In the $\not{E}_T$ distribution it is searched for deviations from the SM background. These analyses are sensitive to sparticle decays, in which the $\not{E}_T$ spectrum is decoupled from the p_T distribution.

- **Single lepton events with b-jets:** These searches look for a single isolated lepton, large $\not{E}_T$ and at least one b-tagged jet. Such events are expected from pair-produced stop quarks or gluinos.

- **Opposite-sign di-leptons:** Events with opposite-sign leptons, jets and $\not{E}_T$ are selected. These sleptons are expected to be produced in chargino and neutralino decays, which emerge from squark or gluino cascade decays. For these analyses two complementary search strategies were developed. The first one uses different signal regions to require cuts on $\not{E}_T$ and the scalar sum of the transverse jet energies (H_T). In the other case, the characteristic kinematic edge in the di-lepton mass distribution is studied.

- **Same-sign di-leptons:** These analyses search for same-sign di-leptons and large $\not{E}_T$. Such signatures are expected from gluinos and neutralinos. To suppress the background a combination of $\not{E}_T$ and H_T cuts is used.

- **Same-sign di-leptons and b-jets:** For such analyses events with a pair of same-sign di-leptons, large $\not{E}_T$ and at least one b-tagged jet are of interest. These signatures are expected from gluinos decaying into stops or sbottoms.

- **Di-leptons from Z boson decay:** These searches look for events with a pair of opposite-charge and same-flavor leptons from a Z boson decay, jets and $\not{E}_T$.

- **Di-leptons:** Events containing a pair of isolated leptons are selected. Using a set of kinematic variables (RAZOR variables) these searches are sensitive to signatures from strongly produced sparticles which decay through charginos and neutralinos.

- **Multi-leptons:** These analyses search for events with at least three leptons. Such signatures are expected from chargino and neutralino decays. Cuts on several event variables, the invariant mass of the lepton pair and the hadronic activity are used to suppress the background.

- **Hadronic taus:** This kind of analyses is interested in events with at least two opposite-sign taus which decay hadronically and high $\not{E}_T$. Such signatures are expected in direct chargino or neutralino producing events decaying via tau sleptons.

- **Higgs boson:** These analyses search for events containing a Higgs boson in the final state. Decays via a Higgs boson are expected for wino-like $\tilde{\chi}_1^{\pm}$, $\tilde{\chi}_2^0$ and bino-like $\tilde{\chi}_1^0$.

Generally the results of the searches are published in two different maps (Fig. 5.6): One is a map showing acceptance times efficiency ($A \times \varepsilon$) as a function of the mass parameters for the assumed simplified model. The other plot shows the 95 % confidence level upper limit on production cross section of the SMS mapped as function of the mass parameters ($\sigma \times \mathcal{B}$ UL maps).

The simplest possible analysis is a "cut and count" analysis of one signal region. Therefore, a $A \times \varepsilon$ map is created to describe the probability of signal events to be reconstructed and accepted in the signal region for a set of masses. The calculation of the upper limits for a single signal region can be based on the Poisson probability for an observed event count in the case of the background only or signal plus background hypothesis. The expected number of events for the second hypothesis depends on

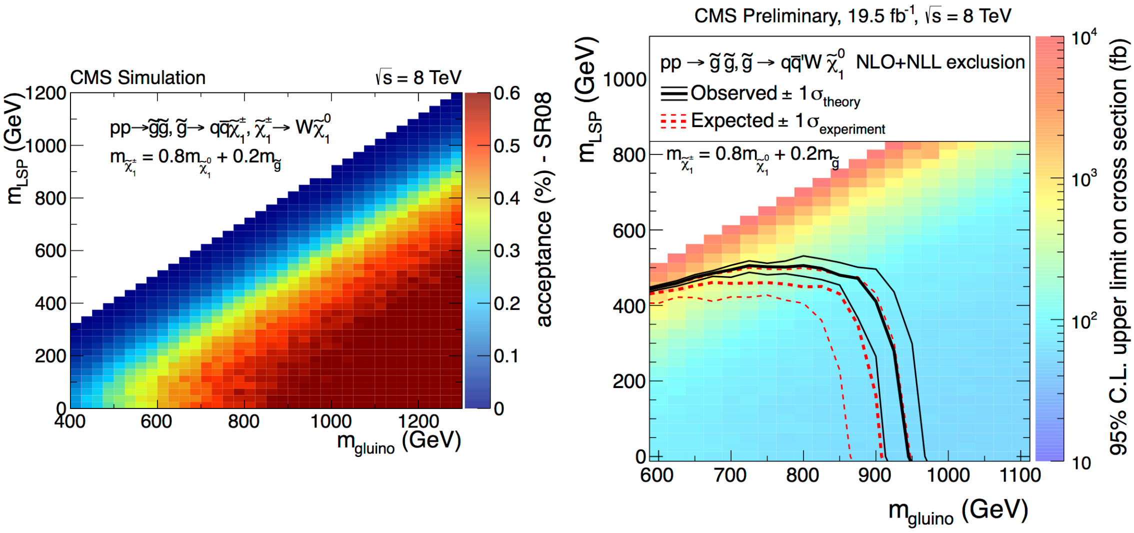


Figure 5.6: Acceptance (left) and upper limit on cross section (right) maps for *T5WW* created by the CMS analysis CMS-SUS-13-013 [34].

the signal cross section, $A \times \varepsilon$ and the integrated luminosity. This concept can be generalized to include systematic uncertainties. The upper limit at 95 % confidence level is defined as the cross section value where the integral of the tail of the signal plus background distribution below the observed value is equal to 5 %.

To obtain an exclusion line to interpret the results, it is necessary to use a reference cross section. The reference cross section for a SMS is calculated from the production cross section of mother sparticles and the branching ratio into the certain SMS topology. In general, the analyses assume a branching ratio of 100 %. Therefore, the reference cross section is equal to the production cross section of the sparticles. The production cross section depends mainly on the sparticle masses appearing in the SMS, all other masses are decoupled. The exclusion line is determined by comparing the reference cross section to the upper limit. Points on the mass plane for which the reference cross section exceeds the upper limit are excluded. The boundary line between excluded and not excluded points is the exclusion line. To mark the uncertainties the theory uncertainty is added to and subtracted from the reference cross section and these values are again compared against the upper limit. To obtain the expected results, event numbers are generated for the expected background and signal plus background hypothesis.

5.2.2 Database

The SUSY analyses results published by ATLAS and CMS are standardized and stored in the database. The database consists of different directories all having the same substructure. Basically the results are sorted by their center of mass energy $\sqrt{s}$ and the experiment. Currently three main directories exist: *2012*, *8TeV* and *ATLAS8TeV*. There is no naming convention for these directories, they grew with the project and in the following will be referred to as runs like in the `SModelS` framework. The CMS results with $\sqrt{s} = 7\,\text{TeV}$ are stored in *2012*, *8TeV* contains the CMS results with $\sqrt{s} = 8\,\text{TeV}$ and *ATLAS8TeV*, as the name implicates, contains the ATLAS results at $\sqrt{s} = 8\,\text{TeV}$.

Each run comprises one directory for each analysis and each analysis consists of one or more different simplified models. All analyses, which are currently available in the database, are listed in Tab. 5.1- 5.3 including all simplified models and the luminosity for each analysis.

Analysis	Simplified Models (Tx Names)	Luminosity [fb^{-1}]
SUS-12-005 SUS-11-024 [35, 36]	T2	4.7
SUS-12-011 [37]	T2	5.0
SUS-11-013 [38]	TChiWZ	5.0
SUS-11-022 [39]	T1tttt, T2tt, T1bbbb, T2, T1, T2bb	5.0
SUS-12-006 [40]	TChiWZ	5.0

Table 5.1: CMS results at $\sqrt{s} = 7\,\text{TeV}$ currently available in the database.

Analysis	Simplified Models (Tx Names)	Luminosity [fb^{-1}]
SUS-12-022 [41]	TChiWZ, TChiChipmStauStau, TSlepSlep, TChiChipmSlepStau, TChiChipmSlepL, TChipChimSlep-Snu	9.2
SUS-12-026 [42]	T1tttt	9.2
SUS-13-008 [43]	T6ttWW, T1tttt, T6bbZZ, T7btbtWW, T5tttt	19.5
SUS-13-004 [44]	T2tt, T1bbbb, T1tttt	19.3
SUS-12-024 [45]	T1bbbb, T1tttt	19.4
SUS-13-006 [46]	TChiWZ, TChiChipmStauStau, TSlepSlep, TChiChipmSlepStau, TChiChipmSlepL, TChipChimSlep-Snu	19.5
SUS-13-016 [47]	T1tttt	19.7

SUS-13-017 [48]	TChiChipmHW	19.5
SUS-12-028 [49]	T1tttt, T2tt, T1bbbb, T2, T1, T2bb	11.7
SUS-13-011 [50]	T2tt, T6bbWW	19.5
SUS-13-012 [51]	T2, T1	19.5
SUS-13-013 [34]	T5WW, T6ttWW, T1tttt, T7btbtWW, T5tttt	19.5
SUS-13-002 [52]	T6ttWW, T1tttt	19.5
SUS-13-007 [53]	T1tttt	19.4

Table 5.2: CMS results at $\sqrt{s} = 8\,\text{TeV}$ currently available in the database.

Analysis	Simplified Models (Tx Names)	Luminosity [fb^{-1}]
ATLAS-CONF-2013-035 [54]	TChiWZoff, TChiWZon, TChiChipmSlepL	20.7
ATLAS-CONF-2013-036 [55]	TChiChiSlepSlep	20.7
ATLAS-CONF-2013-037 [56]	T2tt, T6bbWW	20.7
ATLAS-CONF-2012-166 [57]	T2tt, T6bbWW	13.0
ATLAS-CONF-2013-053 [58]	T2bb	20.1
ATLAS-CONF-2013-001 [59]	T6bbWWoff	12.8
ATLAS-CONF-2013-025 [60]	T6ttZZ	20.7
ATLAS-CONF-2013-007 [61]	T8ChiSlep, T1tbtb, T6ttWW, T1tttt, T5tttt, T7ChiSlep, T5WW	20.7
ATLAS-CONF-2013-028 [62]	TChiChipmStauL, TChipChimStauSnu	20.7
SUSY-2013-04 [63]	T1tttt	20.3
SUSY-2013-05 [64]	T6bbWWon, T2bb, T6bbWWoff	20.1
ATLAS-CONF-2013-024 [65]	T2tt	20.5
ATLAS-CONF-2012-105 [66]	T1tttt	5.8
ATLAS-CONF-2013-048 [67]	T2bbWW, T6bbWW	20.3
ATLAS-CONF-2013-062 [68]	T6WW, T6bbWWoff, T5WW	20.0
ATLAS-CONF-2013-089 [69]	T6WW, T5WW	20.3
ATLAS-CONF-2013-049 [70]	TSlepSlep	20.3
ATLAS-CONF-2013-047 [71]	TGQ, T6WW, T2, T1, T5WW, T5tctc	20.3

ATLAS-CONF-2013-061 [72]	T1bbbb, T1tbtb, T1tttt	20.1
ATLAS-CONF-2013-065 [73]	T2tt, T6bbWW	20.3
ATLAS-CONF-2013-093 [74]	TChiChipmHW	20.3

Table 5.3: ATLAS results at $\sqrt{s} = 8\,\text{TeV}$ currently available in the database.

Each analysis directory has the same structure. It contains a file *convert.py* for standardizing the results, a Python script called *Standardizer.py* containing functions to standardize the data, a directory *orig* for storing the original data published by the experiments and, of course, the data itself. The data storage is divided into two parts: the results themselves like $\sigma \times \mathcal{B}$ UL maps and exclusion lines and additional information like luminosity, document number, constraints, conditions and all information that could be of interest. Originally the upper limit on cross section maps and the exclusion lines were stored in ROOT files named *sms.root*. The upper limit on cross section maps are 2-dimensional ROOT-histograms (T2HF) where the x- and y-coordinate correspond to the masses of two different sparticles and the z-coordinate gives the upper limit on the cross section in fb. In the same file, the exclusion lines are stored in TGraph format. For practical reasons, now the upper limit on cross section maps are stored in Python dictionaries called *sms.py*, but the ROOT files still exist especially for the exclusion lines. The naming convention used in *sms.root* for the upper limit on cross section maps is *limit_* + Tx name of the simplified model for the observed upper limits. For the expected upper limits on cross section the term *expectedlimit_* is used instead of *limit_*. The naming convention for the exclusion lines is analogous, *exclusion_* + Tx name of the simplified model for the observed exclusion line, for the expected ones it is *expectedexclusion_* + Tx name, and for the observed or expected exclusions $\pm 1\sigma$ simply *p1* or *m1* is added to the Tx name.

For the upper limit on cross section maps stored in Python dictionaries, the naming is different. The observed upper limits are stored in a dictionary called *Dict* and the expected upper limit dictionaries are called *ExpectedDict*. These dictionaries are nested. The outermost one is using the Tx name as key for each simplified model. The next level of nesting uses the x-coordinate as key and the last level uses the y-coordinate. The value is the z-coordinate, i.e. the upper limit on cross section in fb at this specific point.

The experimental analyses publish also results for searches for SMS topologies which include intermediate sparticles. In these cases, the 2-dimensional $\sigma \times \mathcal{B}$ UL maps are not defined because of the three unknown sparticle masses instead of two. Therefore, one mass has to be constraint by the two others. To make proper use of these results, it is necessary to have results for different constraints on the third sparticle mass. Most of the time, the mass of the mother sparticle is given on the x-axis and the daughter sparticle mass is on the y-axis, like for SMS with only one decay. For constraining the mass of the intermediate sparticle usually one of the following options is used:

- the mass is set constant, e.g., $m_{\tilde{\chi}_1^\pm} = 50\,\mathrm{GeV}$
- the mass is set in a fixed ratio to the mother or the daughter sparticle, e.g., $\frac{m_{\tilde{g}}}{m_{\tilde{\chi}_1^\pm}} = 2$
- the mass is defined by the x-value solving the following equation:

$$m_{intermediate} = x \cdot m_{mother} + (1-x) \cdot m_{daughter} \tag{5.1}$$

Some analyses constrain the mass of the mother or the daughter sparticle and scan over the intermediate and the daughter or mother mass. These constraints on the masses and axes of the upper limit plots are stored in the field *axes* in *info.txt*. To read out these plots correctly, meaning to get a mass array containing three masses and the corresponding upper limit, a special nomenclature [75] and an interpolation between the upper limits of the different mass planes has been included. The interpolation routine is called *SMSInterpolation* and it is used by the function *SMSResults.getSmartUpperLimit()*. For interpolation linear griddata interpolation provided by the Python package *scipy* is used. In case one mass is fixed the upper limit map is nonapplicable to many cases. For the `SModelS` framework the most useful results are those published, e.g., for three different x-values, such that the interpolation can be used.

All the additional information is stored in a file called *info.txt*. One example database entry for Fig. 5.6 is shown in Tab. 5.4. To read out this information easily the module *SMSResults* was written. *SMSResults* is part of the package *Experiment* and contains many different functions. All functions of *SMSResults* are listed in Appendix B.

5.2.3 Using the Database in the SModelS Framework

To compare the computed cross section to the upper limit reported by the experiments, the `SModelS` framework needs to load the information from the database. This is done by the function *SMSAnalysisFactory.load()*. Its functionality is described as pseudocode in Listing 5.2.

```
sqrts: 8.00
lumi: 19.50
url: https://twiki.cern.ch/twiki/bin/view/CMSPublic/PhysicsResultsSUS13013
exclusions: T5WW020 700 945
exclusions: T5WW050 375 917
expectedexclusions: T5WW020 675 907
expectedexclusions: T5WW050 375 916
constraint: T5WW -> [[['jet','jet'],['W']],[['jet','jet'],['W']]]
condition: T5WW -> None
fuzzycondition: T5WW -> None
category: T5WW -> hadronic
axes: T5WW: M1 M0 020 - M1 M0 050
```

Table 5.4: Example for a database entry describing the SMS result of Fig. 5.6.

Listing 5.2: Pseudocode SMSAnalysisFactory.load()

```
ListOfAnalyses = []
for analysis in analyses:
    for topology in SMSResults.getTopologies:
        initialize EAnalysis
        set values in EAnalysis (label, sqrts, masscomp, run)
           with help of SMSResults
        constraint = SMSResults.getConstraint (e.g. '[[['l+']],[[
           'l-']]]')
        set values in EAnalysis.Top = ATop (vertparts, vernumb
           both reconstructed from constraint)
        conditions = SMSResults.getConditions (e.g. '[[['l+']],[[
           'l-']]] > 1.8* [[['e+']],[['e-']]]')
        EAnalysis.results = {constraint: condition}
        EAnalysis.plots = {constraint: [topology, [analysis]]}
        ListOfAnalyses.append(EAnalysis)

for EAnalysis in ListOfAnalysis:
    EAnalysis.generateElements: takes constraint and conditions
       and generates an AElement for each appearing element
    EAnalysis.getPlots: checks if for each constraint exists an
       entry in EAnalysis.plots and if for this model data is
       available

return ListOfAnalyses
```

The function *SMSAnalysisFactory.load()* creates the analysis objects *EAnalysis* from the information provided by the database and returns them in a list. For this task it uses the *SMSAnalysis.EAnalysis* class from the *Theory* package, the functions *PhysicsUnits* from the *Tools* package and *SMSResults* from the *Experiment* package. The user can specify the analyses, the SMS topologies and/or the center of mass energies for which an *EAnalyis* object is created. By default the function will use all analyses and SMS given in the database with $\sqrt{s} = 7$ and 8 TeV.

For each SMS topology in each analysis one *EAnalysis* object is initialized and the values are set using *SMSResults*. Each *EAnalysis* object contains one *ATop* object. From the constraint the information about the vertices for *vertpart* and *vertnumb* in the *ATop* is reconstructed. Using the function *EAnalysis.generateElement()* creates an *AElement* for each *EAnalysis* object from the constraint and conditions. To check if for each *EAnalysis* object upper limits are available in the database the function *EAnalysis.getPlots()* is used.

5.3 Calculation of Theoretical Predictions

Having the experimental data already converted into `SModelS` language the theoretical model needs to be decomposed into `SModelS` language as well, in order to do the comparison. Therefore, the full BSM model is passed through the `SModelS` framework.

5.3.1 Calculation of the Production Cross Section

The knowledge of the production cross section for all produced sparticle pairs is essential for the decomposition in the `SModelS` framework.

`SModelS` provides a function *writeXSecToSLHAFile()* for adding the XSECTION block to an arbitrary SLHA file. This function is part of *SLHATools*, which is a module of the *Theory* package. It needs an SLHA file without XSECTION block as input file. The user can provide information about the cross section to be computed (center of mass energy, order, label) in form of a *CrossSection* object. In case that the *CrossSection* object was created in an earlier step, this object can be used to obtain the information as well. If no *CrossSecction* object exists, default values (7 and 8 TeV at leading order and next-to-leading order plus next-to-leading-logarithmic order) are used to compute the production cross sections. The actual calculation of the production cross sections is done by the function *XSecComputer.compute()* from the *Theory* package. The function *XSecComputer.compute()* returns a *CrossSection* object, which includes the production cross sections and the weights according to the branching ratios for the different production modes occurring in the provided SLHA file. Some general information about this *CrossSection* object is stored in the *XSecInfoList* object. The XSECTION block is written to the SLHA file using the information from the classes in the *CrossSection* module.

The actual calculation of the production cross section is more complex. It is done by the function *XSecComputer.compute()*, which uses PYTHIA and NLLfast. To run *XSecComputer.compute()* one must define the number of events which should be calculated by PYTHIA and one SLHA file must be passed. If no additional information about the production cross section to be computed is provided and also the *XSecInfoList* object does not exist, the cross section is computed at 7 and 8 TeV at LO and NLO+NLL. PYTHIA is called by the function *XSecComputer.runPythia()*. This function needs the SLHA file, the number of events which should be generated and the center of mass energy in TeV to run PYTHIA. From the PYTHIA output the total production cross section of all included subprocesses in fb and the LHE file are returned to *XSecComputer.compute()*. Then the function *XSecComputer.loFromLHE()* computes the production cross section for all occurring SMS and the weight for a single event. For this task *XSecComputer.loFromLHE()* needs the LHE file, the total production cross section and the total number of produced events by PYTHIA. The LHE file is read using *LHEReader.LHEReader()*, which takes the LHE file and the total number of events produced by PYTHIA as maximum number of events to read to count the number of truly generated events. All produced pairs of mother sparticles are read out by using *getMom()* an internal function of the *LHEReader*. The function *XSecComputer.getprodcs()* is called to sort the produced mother sparticle pairs by their PDG id and to count the number of events with the same production mode in the LHE file. This information is returned to *XSecComputer.loFromLHE()* where for each of these production modes the weight w

$$w = \frac{\sigma_{tot}}{n_{true}} \tag{5.2}$$

and the production cross section σ

$$\sigma = \frac{n\sigma_{tot}}{n_{true}} \tag{5.3}$$

are calculated at leading order. The n_{true} denotes the total number of events in the LHE file, σ_{tot} stands for the total production cross section and n denotes the number of events corresponding to this specific production mode. The production modes, their number of appearance in the LHE file, σ and w are returned to *XSecComputer.compute()*. Simplified model spectra, which not appear in this LHE file, are not taken into account for further calculations and are deleted. For the remaining SMS the weight is recalculated as follows:

$$w = \frac{n\sigma_{tot}}{N_{tot}n_{true0}}, \tag{5.4}$$

where N_{tot} stands for the number of all events which should be generated by PYTHIA (by default 10000) and n_{true0} represents the number of events with the same production mode. In the following, the NLO and NLO+NLL cross sections are calculated, if requested by the user. For these higher-order calculations the function *NLLXSec.getNLLresult()* is called to compute the k-factor for each production mode using NLLfast. The gluino mass and the average mass of a light squark is read from the SLHA file using *pyslha*. To run NLLfast *NLLXSec.getNLLfast()* is called passing the production mode, the squark and gluino mass and the center of mass energy to it. From the NLLfast output the cross sections in LO, NLO and NLO+NLL and the k-factors for NLO and NLO+NLL are returned *XSecComputer.compute()*. There, for each production mode at each center of mass energy the k-factors are multiplied to the corresponding LO w and σ in order to obtain the weights and production cross sections at higher orders. Using all the computed information the *CrossSection* and the *XSecInfoList* objects are created. Finally, *XSecComputer.compute()* returns a *CrossSection* object containing the weights, cross sections, additional information from the *XSecInfoList* and the names of the corresponding LHE files.

5.3.2 Decomposition

As mentioned before, decomposition in the context of `SModelS` means to split a complex BSM model into simple SMS topologies taking only the resulting SM particles, the weights of the topologies and the masses of the sparticles into account.

In the `SModelS` framework the decomposition is done by the function *SLHADecomposer.decompose()* contained in the *Theory* package.

Currently the decomposition is mainly based on SLHA files. LHE file based decomposition is implemented, but just as a beta-version. Future plans are to provide fully tested SLHA and LHE file based decomposition, so that the user can choose. The LHE based decomposition requires the generation of parton level Monte Carlo (MC) events and their mapping to the corresponding SMS. For the SLHA based decomposition only the SLHA file with the sparticle spectrum, the decay table and the production cross sections is necessary.

LHE based Decomposition

The decomposition based on LHE files, containing parton level MC events, is the most general one. Each BSM model for which it is possible to simulate such MC events can be decomposed using the LHE file. Each event contained in the LHE file is mapped to the corresponding SMS topology. The $\sigma \times \mathcal{B}$ for each SMS is directly given by the sum of the MC weights of all events contributing to the same SMS. The main disadvantage of this decomposition method is that MC uncertainties are introduced in the decomposition results, but this problem can be curtailed by increasing the MC statistics.

SLHA based Decomposition

For this kind of decomposition the BSM model must be available in SLHA format. Considering SUSY, many software tools are available to calculate the sparticle spectrum and the decay table (see Chap. 4). The production cross section, also for higher orders, can be calculated using PYTHIA, PROSPINO and NLLfast or the implementation of PYTHIA and NLLfast in the `SModelS` framework (see Chap. 5.3.1). For other BSM models different software tools will be needed to generate an SLHA file.

The decomposition is done by using the production cross section for all produced BSM particle pairs and their branching ratios to generate all possible signal topologies. Compared to the LHE based decomposition the uncertainties for SLHA decomposition are much smaller, since the only contributing ones are the cross section uncertainties. To prevent the inclusion of a large number of irrelevant SMS a lower limit for the $\sigma \times \mathcal{B}$ is introduced, so that only topologies with a weight above the cut value are kept. By default this lower limit is 0.1 fb.

The translation of these concepts into the *SLHADecomposer.decompose()* function of the `SModelS` framework is shown in Listing 5.3 in form of a pseudocode. To decompose a BSM model *SLHADecomposer.decompose()* needs the BSM model in SLHA format. The user can provide the production cross sections in form of a *CrossSection* object. If no *CrossSection* object is passed, it is constructed from the XSECTION block in the SLHA file. To be able to create all final SMS topologies the relevant information is read from the SLHA file and the *CrossSection* object: the highest production cross section of each initially produced BSM particle for almost degenerate production modes, all allowed branchings for these particles and their masses. The next step is to generate all possible one-branch elements (*SMSDataObjects.BElement*) for each particle. The *BElement* is only created if its weight (calculated from the maximum production cross section times branching

Listing 5.3: Pseudocode SLHADecomposer.decompose()

```
if not XSection:
    create CrossSection.XSecInfoList from SLHA file
Pdic = {pdg id mother particle: maximum production cross section}
BRdic = {pdg id: [[branching ratio, [pdg ids of daughters]],...],
    -pdg id: [[branching ratio, [-pdg ids of daughters]],...]}
Massdic = {pdg id: addunit(mass, 'GeV'), -pdg id: addunit(mass, '
   GeV')}

#generate all possible BElements with sigmamax*BR > sigcut
for particle in Pdic.keys():
    create BElement
    ElList = List of BElements
    WList = List of weights (= maximum production cross section
       from Pdic)
#create BElements according to possible branchings
while newElement:
    newElement = False
    for BElement in ElList:
        if mother particle of BElement is already stable final
           state:
            FinalWeightList.append(WList[BElement])
            FinalList.append(BElement)
            continue
        for decay in BRdic[pdg id mother particle]:
            NewBElement = deepcopy(BElement)
            for id in decay.ids:
                if id is R-even: vertparts.append(ParticleNames.
                   Reven[id])
                if id is R-odd: mass.append(Massdic[id]),
                   NewBElement.momID.append(id)
            NewBElement.particles.append(vertparts)
            keep only NewBElements with weight*branching ratio >
               sigma cut: newElement = True
            calculate weight for NewBElement = WList[mother
               particle]*branching ratio
    if newElement:
        ElList = deepcopy(List of NewBElements)
        WList = deepcopy(List of computed weights for
           NewBElements)

SMSTopList = SMSDataObjects.TopologyList()
#combines BElements to possible EElements according to production
    cross section
find sparticles of initially produced pairs in BElements as
   mother particles
    SMSDataObjects.EElement()
    EElement.B = [BElement_1, BElement_2]
    recaluclate weight for each order and center of mass energy:
        EElement.weight = (pair production cross section *
           weight_mother1 * weight_mother2)/(Pdic[mother1]*Pdic[
           mother2])
    SMSDataObjects.GTop()
    set values in GTop using EElement.getEinfo()
    if user wishes: compress GTop using TopologyBuilder.
       compressTopology() invisible or masscompression
    TopologyList.addList([final GTop])

return TopologyList
```

ratio) is above the cut value. Therefore, for each particle a *BElement* is initialized and all parameters, like weight, mass and PDG id of the mother sparticle are set using the information from the SLHA file and the *CrossSection* object. Based on these *BElements*, new *BElements* are created according to the branchings allowed for each mother BSM particle. The mass vector is constructed from the masses of all $\mathbb{Z}_2$-odd daughter particles, while the $\mathbb{Z}_2$-even particles are added to a list of all particles identifying the signal topology. The weight for these new *BElements* is calculated by the maximum production cross section for the mother sparticle times the branching ratio for this specific topology. After all possible *BElements* have been created, an *SMSDataObjects.TopologyList* object is initialized. In order create all SMS which together describe the BSM model, the *BElement* objects are matched according to the initially produced BSM particle pairs. Each match is stored in an *SMSDataObjects.EElement* object. The weight for these *EElement* objects is calculated for each occurring center of mass energy and order according to following equation:

$$w_{final} = \frac{\sigma_{prod} w_{mom1} w_{mom2}}{\sigma_{max1}\sigma_{max2}} \tag{5.5}$$

The σ_{prod} denotes the cross section of the pair production, which was either initially computed or provided by the SLHA file. The weights calculated for each *BElement* object are named w_{mom1} and w_{mom2}. The σ_{max1} and σ_{max2} are the maximum production cross sections for each mother BSM particle. These recalculated weights are added to *EElement.weight*. The information about the vertices are read out from the *EElement* object by using *EElement.getEInfo()*. A general topology is initialized by calling *SMSDataObjects.GTop()*. There the information about vertex particles and vertex numbers is set as *GTop.vertnumb* and *GTop.vertparts* and the *EElement* is added to the *GTop.ElList*. In case the user has chosen any type of compression *TopologyBuilder.compressTopology()* is called on this list of topologies and the output is added to *SMSDataObjects.TopologyList()*. If none of these two options is selected, the list of *GTop* objects is added to the *SMSDataObjects.TopologyList()* object which finally is returned to the user.

Compression of Topologies

In the `SModelS` framework one has to distinguish between two types of topology compressions: invisible compression and mass compression.

The invisible compression is performed on SMS which include invisible decays at the end of their decay chain, like decays into ν (Fig. 5.7). Such SMS are equal to their compressed version for the experimental analyses, since the effective final BSM

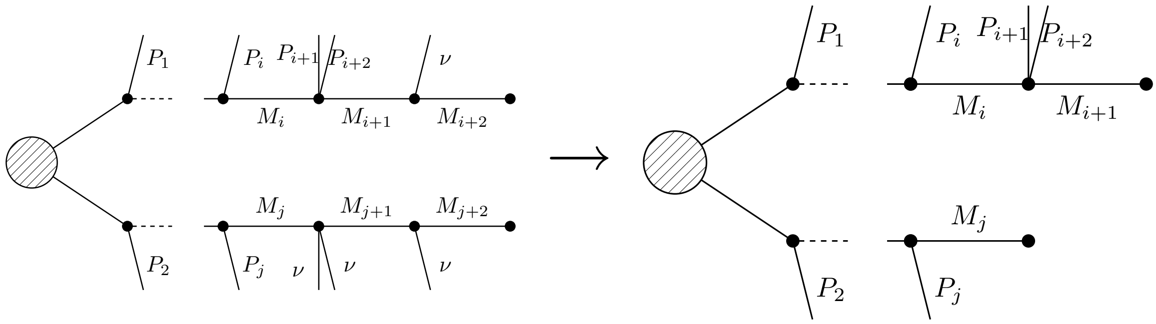


Figure 5.7: Invisible compression of an SMS topology.

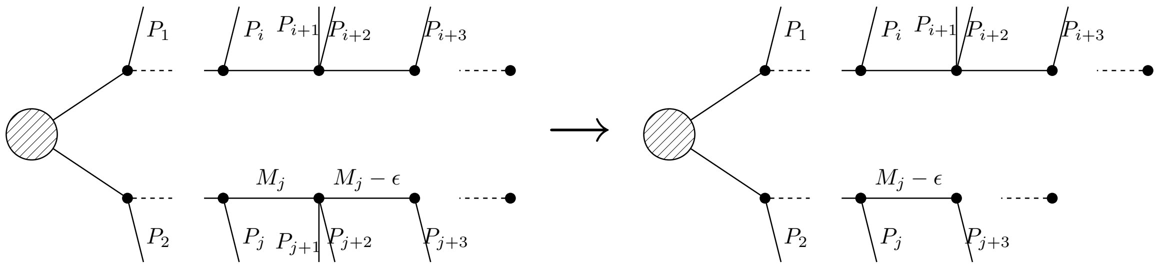


Figure 5.8: Mass compression of an SMS topology.

momentum is given by the sum of the neutrino and the BSM final state momenta. In `SModelS` language invisible compression can be notated as $[[V, [\nu, \nu]], [V', [\nu]]] \rightarrow [[V], [V']]$ where V and V' stand for other cascade decay vertices. When applying this compression, one has to remove the last entry of the mass vector for the BSM particle masses so that the LSP becomes heavier.

The mass compression means the compression of decays between quasi-degenerate states for topologies with an extremely compressed mass spectra (Fig. 5.8). This approximation is valid if the energy carried away by the SM particle during the decay of the quasi-degenerate states is insignificant for the experimental analyses. In this case the mass-compressed SMS is equivalent to the original one. In the `SModelS` framework mass compression is used for quasi-degenerated states with a mass difference below 5 GeV.

The concept of compressions allows to constrain complex SMS topologies using experimental constraints for simpler topologies. When performing compressions and both, the original and the compressed, topologies are kept, care must be taken to avoid double counting to prevent doubling the weight of the SMS.

5.3.3 Matching Theoretical Predictions with Experimental Results

Before the theoretical prediction can be compared to the experimental results, the SMS from the decomposition need to be combined according to the conditions of the analyses.

Listing 5.4: Pseudocode EAnalysis.add(ListOfTopologies)

```
for GTop in ListOfTopologies:
    if not GTop == self.Top: compares the number of vertices and
       particles at each vertex
        continue
    for EElement in Top.ElList:
        self.Top.addEventElement(): adds the weight (and the mass
            vector in case it is not equal to the mass vector of
            the AElement) of the EElement to the corresponding
           AElement.MassWeightList entries in self.Top.ElList

return True
```

In the `SModelS` framework this is done in two steps: First, the *TopologyList* object needs to be added to the *EAnalyis* object using the function *EAnalysis.add()*. Second, the function *EAnalysis.computeTheoryPredictions()* is called to do the clustering of the mass vectors.

The function *EAnalysis.add()* is described in Listing 5.4. The *EAnalysis* class contains an *ATop* object with *AElements*, which are constructed from the constraints and conditions of the analyses. The entries of the *TopologyList* are *GTop* objects obtained by the decomposition procedure. Each of these *GTop* objects is compared to the *ATop* of the *EAnalysis* object. Only if the number of vertices and the particles at each vertex match, the *EElement* of the *GTop* is added to the corresponding *AElement* of the *EAnalysis.Top* by using the function *EAnalysis.Top.addEventElement()*. The function *EAnalysis.Top.addEventElement()* checks if the signal topologies of the *EElement* and the *AElement* are equal. In this case, the mass vectors of the *EElement* and the *AElement* are compared. If both topologies have the same mass vectors, the *EElement* is added to the *AElement* by summing up their weights using the function *ClusterTools.sumweights()*, else a new *MassWeight* entry is added to the *AElement.MassWeightList* containing the weight and the mass vector of the *EElement*. The comparison of the mass vector is necessary, because the analyses implicitly assume that all combined SMS topologies have a common BSM mass vector. As soon as each *GTop* is checked and all matching *EElements* are added, the function returns ‘True’ to the user.

Listing 5.5: Pseudocode EAnalysis.computeTheoryPredictions()

```
for AElement in self.Top.ElList:
    check if mass array is good: ClusterTools.GoodMass()
    if GoodMass(): massweight.mass = GoodMass() and create list
       of good mass arrays

MCluster = DoCluster(Goodmasses,self.MassDist,dmin,MassAvg,self.
   MassPosition)

if DoCluster failed: use unclustered masses (=good masses)

#clear out ResultList
self.ResultList = []

for cluster in MCluster:
    create masscluster list by appending good masses ???
    initialize CTop(self.Top,masscluster)
    ClusterResult = CTop.evaluateCluster(self.results) creates
       and returns an XSecPredictionForCluster object and
       constructs the result_dic and conditions_dic
    ClusterResult.mass = CTop.clustermass
    ClusterResult.explimit = LimitGetter.GetPlotLimit(
       ClusterResult.mass,self,complain=False)
    check if average mass is inside cluster (expected limit for
       average mass ~ expected limit for individual mass):
        if not: continue
    self.ResultList.append(ClusterResult)

if not self.ResultList: return None
if DoCluster failed: return 'Cluster Failed'
return True
```

For finally evaluating the theoretical predictions for $\sigma \times \mathcal{B}$ the main method *EAnalysis.computeTheoryPredictions()* (Listing 5.5) is called. In the function *EAnalysis.add()* the SMS were only combined if the BSM mass vectors of the SMS from the database and from the decomposition of a BSM model are equal. Now the function *ClusterTools.GoodMass()* looks for similar mass arrays to combine more SMS. Mainly, the function *GoodMass()* decides if the mass array is 'good', which means that in case the branches have the same topology, the masses are similar or if the branches have different topologies, they at least have a similar mother and a similar LSP mass, but it also checks if there is an experimental limit available. The parameter to define if two mass arrays are similar is the sensitivity of an analysis in question to the difference between the two mass vectors. Therefore, the upper limits for both mass vectors are read out from the database and their relative difference d_{UL} is calculated by

$$d_{UL} = 2\frac{|ul_1 - ul_2|}{ul_1 + ul_2} \tag{5.6}$$

where $ul_{1,2}$ represents the upper limit returned from the function *MassPosition()* for each of the two mass arrays. In case the difference d is less than 20 % the mass vectors are considered as similar with respect to this particular analysis. To avoid cases where the upper limits are equal but the mass arrays are completely different, the distance between the mass vectors has to be below an absolute value. This is done by the function *MassDist()*, which computes the relative difference d_m between the mother masses of both mass arrays by

$$d_m = \frac{m_1 - m_2}{m_1 + m_2}. \tag{5.7}$$

Only mass arrays with $d_m < 0.5$ are considered. If the two mass arrays are considered as similar, they are passed to the function *ClusterTools.MassAvg()*, which computes the average masses and returns the new mass array to *EAnalysis.computeTheoryPrediction()* where the mass array in the *MassWeightList* object is substituted by the new one. The user can choose if the 'good' masses are stored in the *EAnalysis.goodmasses* object. The function *ClusterTools.DoCluster()* is used to cluster the 'good' mass arrays with similar upper limit values together and returns a list of clusters. For each of these clusters a *CTop* object is initialized. The function *CTop.evaluateCluster()* creates an *XSecPredictionForCluster* object and constructs the *result_dic* and the *conditions_dic* from the constraints and conditions of the particular experimental analysis. To set the value for the experimental upper limit the function *LimitGetter.GetPlotLimit()* is called. Finally, the *XSecPredictionForCluster* objects are added to the *EAnalyis.ResultList*, which contains all the computed information.

This procedure can best be illustrated by an example. Let us consider slepton pair production: In the MSSM, the masses for electrons and muons with the same handiness are usually degenerated, but $m_{\tilde{e}_R} \neq m_{\tilde{e}_L}$. The constraint from the experiment if e.g. the CMS analysis SUS-13-006 [46] is taken into account, is to combine topologies of the type [[[l+]],[[l-]]] for l = e,μ, which have similar mass arrays. In this case similar mass arrays means $(m_{\tilde{e}}, m_{\tilde{\chi}_1^0}) \simeq (m_{\tilde{\mu}}, m_{\tilde{\chi}_1^0})$. As mentioned before, similar mass arrays are defined by the difference in their upper limit values. Since left- and right-handed sleptons contribute to this topology, the sleptons need to be grouped by their masses before combining them to the experimental topology. For example, if the analysis is sensitive to right- and left-handed sleptons assuming a large mass splitting, the sleptons will be grouped by their handiness, else all sleptons will group together. Now the SMS with similar mass vectors are combined by adding their weights according to the constraints of the analysis. Before applying the experimental upper limit it is necessary to verify if the combined topology satisfies the conditions postulated by the analysis. Only if these assumptions are satisfied, the theoretical prediction for $\sigma \times \mathcal{B}$ can be compared to the experimental upper limit.

5.4 Validation of the SModelS Procedure

To test the functionality of `SModelS` as extensively as possible, so-called closure tests were performed. The idea of these closure tests is to reproduce the exclusion lines of the upper limit plots published by the experimental analyses using the `SModelS` framework. To this end, SLHA files are created for each of the simplified models pretending to be the full model for the different analyses. The first task is to decompose the full model from the SLHA file, which has to be equal to the simplified model to be tested, using *SLHADecomposer.decompose()*. Then *SMSAnalysisFactory.load()* is used to create the *EAnalysis* objects from the database for the specific analysis and topology, which is to be tested. Having the theoretical model decomposed and the database results loaded, the results from the decomposition are added to the *ListOfAnalyses* which already contains the experimental results by applying *EAnalysis.add()*. To compute the `SModelS` predictions for the cross section *EAnalysis.computeTheoryPredictions()* is used. Finally, the plots are created by testing the computed production cross section against the experimental upper limit on the production cross section. Scanning over the whole mass range covered by the experimental analysis in this manner, if the `SModelS` framework is working well the excluded points should fit into the exclusion line published by the analysis. Two

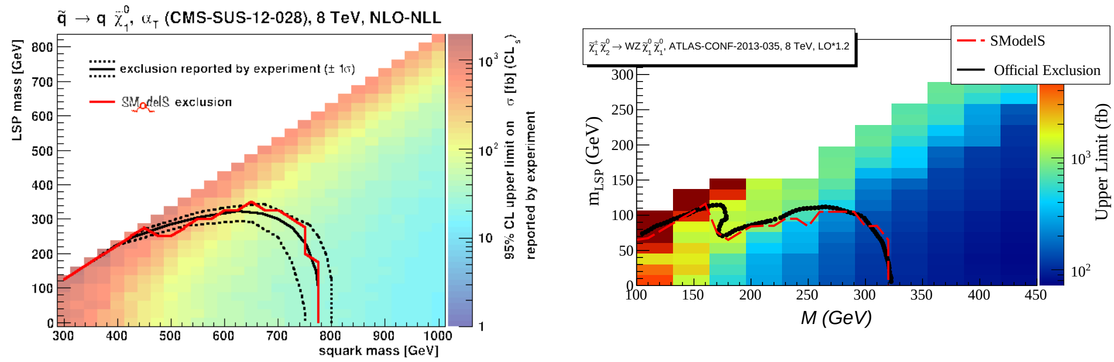


Figure 5.9: The black line shows the official exclusion line, the red line is the reproduced exclusion line using the `SModelS` framework. left: Closure test for *T2* topology documented in CMS-SUS-12-028 [49]. right: Closure test for *TChiWZ* topology [3] for on- and off-shell region documented in ATLAS-CONF-2013-035 [54].

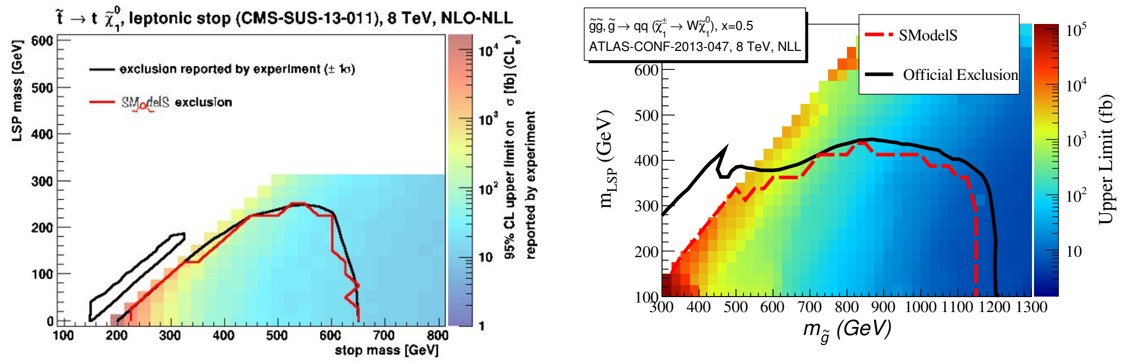


Figure 5.10: The black line shows the official exclusion line, the red line is the reproduced exclusion line using the `SModelS` framework. left: Closure test for *T2tt* topology documented in CMS-SUS-13-011 [50]. The output from the initial SLHA file were two SMS *T2tt* and *T2bb*. The island on the right is not reproduced by `SModelS` because in this test only the on-shell part of the plot was reproduced. right: Closure test for *T5WW* topology [3] documented in ATLAS-CONF-2013-047 [71].

examples are shown in Fig. 5.9. One can see that the reproduced exclusion line fits very well the published one.

These tests only validate the particular case where the output from the SLHA file is exactly one SMS. In this way also the predictions for more complex topologies, which need interpolation, are tested (Fig. 5.10 right). One test was already done, where the output of the SLHA file are two SMS and only one of them was taken into account to reproduce the exclusion line (Fig. 5.10 left). A future task is to do more complex tests, in order to test the reweighting by the branching ratio as well.

6 Applications of SModelS

In the context of this thesis the `SModelS` framework was used for two different tasks: to perform a first scan over a simple test model and to create the official summary plots for CMS. Both tasks will be described in this chapter.

6.1 pMSSM Scan

To test the `SModelS` framework on a complex SUSY model, it was applied in a large scan of a simplified version of the pMSSM assuming the GUT relationship between gaugino parameters $M_1 : M_2 : M_3 = 1 : 2 : 6$. In Tab. 6.1 the chosen parameter ranges for the test model are listed. The parameters are chosen to cover a region in parameter space which is not excluded by the experiments but for which experimental upper limits exist.

Parameter	Range	Description
M_1	100 – 1000 GeV	Gaugino mass
M_{01}	0 – 3000 GeV	common 1^{st} and 2^{nd} generation sfermion mass
M_{03}	0 – 1000 GeV	common 3^{rd} generation sfermion mass
M_A	100 – 2000 GeV	Pseudoscalar Higgs
μ	100 – 1000 GeV	
$\tan\beta$	2 – 50	
A_b	-1000 – 1000 GeV	Trilinear coupling, sbottom
A_{tau}	-1000 – 1000 GeV	Trilinear coupling, stau
A_t	$-3 \cdot M_{03} - 3 \cdot M_{03}$	Trilinear coupling, stop

Table 6.1: Parameters for the test model.

About 80 000 pMSSM points were created in a flat random scan and stored in SLHA files. This was done in two steps: First, the sparticle spectra and the mixing matrices were calculated by SOFTSUSY (Chap. 4.2) and afterwards SDECAY (Chap. 4.3) was used to compute the decay tables. Only points that satisfied the following conditions were considered:

- The LSP is the $\tilde{\chi}_1^0$, and all heavier sparticles decay promptly (hence $m_{\tilde{\chi}_1^\pm} - m_{\tilde{\chi}_1^0} > 300\,\text{MeV}$ is required to veto long-lived charginos),

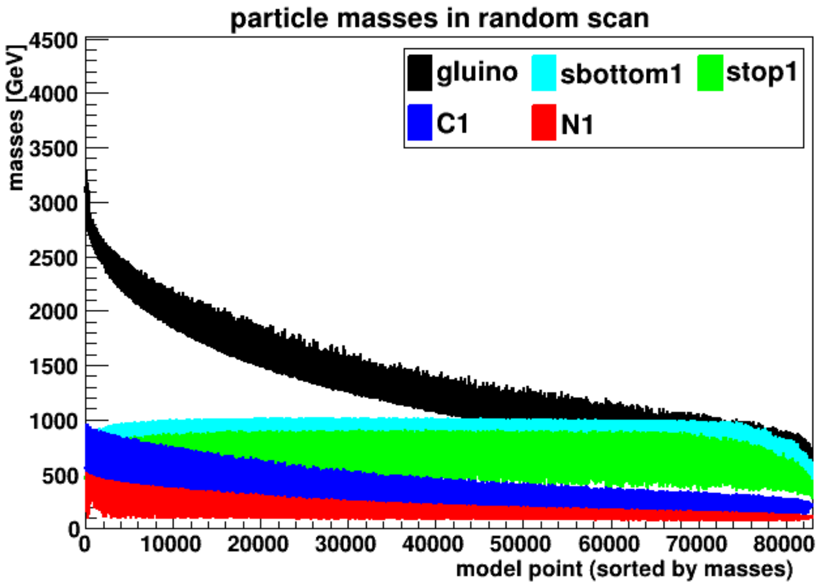


Figure 6.1: Distribution of sparticle masses in the random scan.

- LEP constraints on sparticle masses [76],
- a light Higgs within the mass window of $m_h = 123 - 128\,\text{GeV}$ [1, 2],
- compliance with low energy observables $b \to s\gamma$, $B_s \to \mu^+\mu^-$.

The distribution of the sparticle masses from this random scan is shown in Fig. 6.1. We see that the masses of the lighter chargino, $\tilde{\chi}_1^{\pm}$, and the lighter stop and sbottom mass eigenstates, $\tilde{t}_1$ and $\tilde{b}_1$, are below 1 TeV. The gluino on the other hand can be quite heavy, up to 3 TeV, like the light-flavor squarks and the sleptons. Therefore, we expect that the results from the CMS RAZOR [8, 36] and the CMS alphaT analysis [49] will give the most relevant constraints. Indeed the CMS RAZOR results exclude 5 000 of the points of this specific scan, while the alphaT analysis excludes about 1 000 points (Fig. 6.2).

It is also interesting to consider the sensitivity of the direct stop topology (*T2tt*). The points with the major contribution to *T2tt* are shown in Fig. 6.3. The red area marks the theory points where more than 50 % of all events are *T2tt* decays. For the exclusion plot (Fig. 6.4) all model points in which *T2tt* is more than 25 % of the total cross section are considered. The points excluded by `SModelS` are drawn in red, the not-excluded points are drawn in green. The lines drawn in the plot are the exclusion lines reported by different analyses. One can see that the ATLAS and CMS leptonic stop searches correspond very well to the exclusion area obtained by `SModelS`. The reason why the points with a stop mass between about 350 - 430 GeV are not found

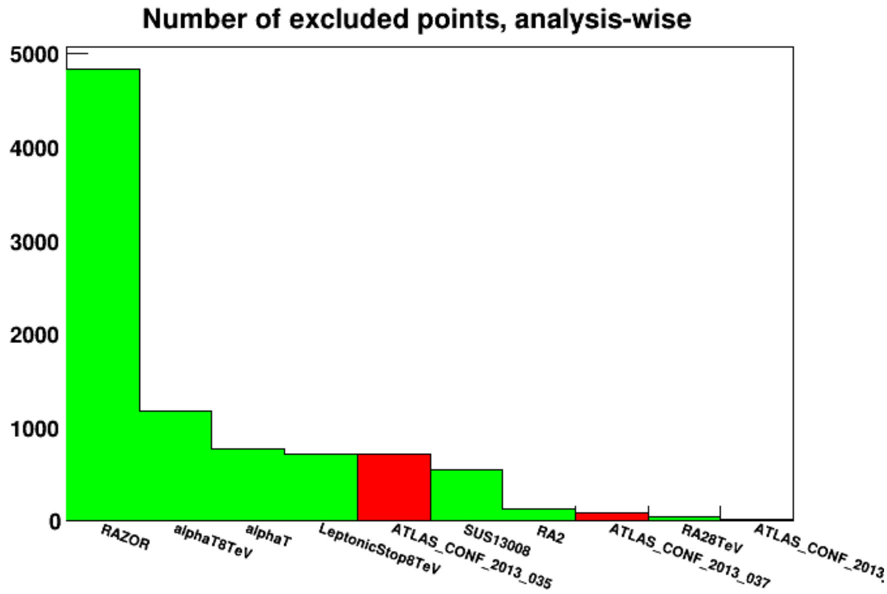


Figure 6.2: Number of excluded points per analysis of experimental data. CMS analyses are colored green, ATLAS analyses are colored red.

to be excluded by `SModelS` but by the experiment is that the branching ratio into *T2tt* becomes too low for these points while the experiments considered an ideal scenario with a branching ratio of 100 % into *T2tt*. There also is an explanation for the points with a stop mass between 460 - 540 GeV only found to be excluded by `SModelS`: In these cases the gluino mass of about 1500 GeV slightly increases the cross section relative to the decoupled limit. Therefore more points are excluded by comparing the cross section to the $\sigma \times \mathcal{B}$ UL obtained by the experimental analyses.

Two more detailed pMSSM scans were done, one focusing on the electroweak sector with all squarks assumed heavy and one with heavy leptons but light squarks including sbottoms and stop. The results of these scans are presented in the masters thesis of U. Laa [75] as well as in the recent publication [3].

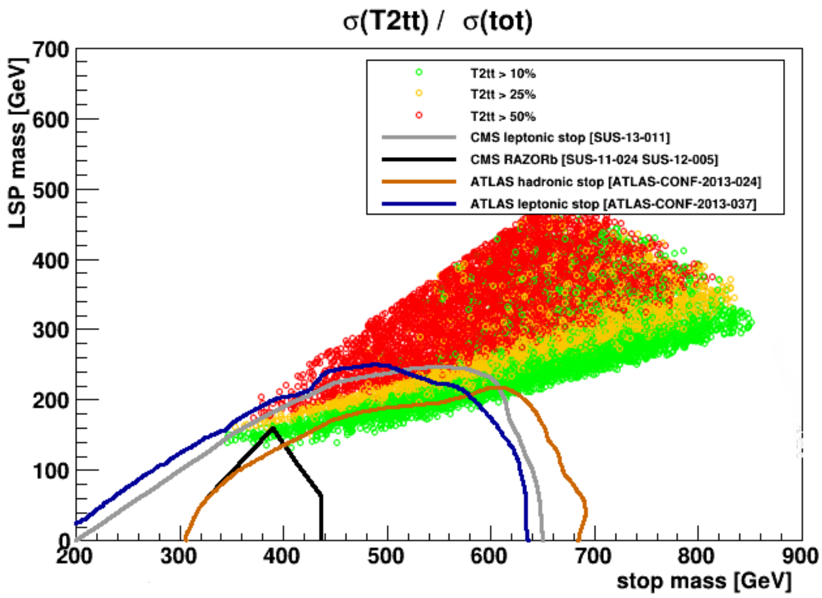


Figure 6.3: Fraction of *T2tt* for points in the scan in the $m_{\tilde{\chi}_1^0}$ vs. $m_{\tilde{t}}$ plane (only points $> 10\,\%$ are shown). The colored lines represent the exclusion lines reported by following analyses CMS-SUS-13-011 [50] (gray), CMS-SUS-12-005 [35] (black), ATLAS-CONF-2013-024 [65] (orange) and ATLAS-CONF-2013-037 [56] (blue).

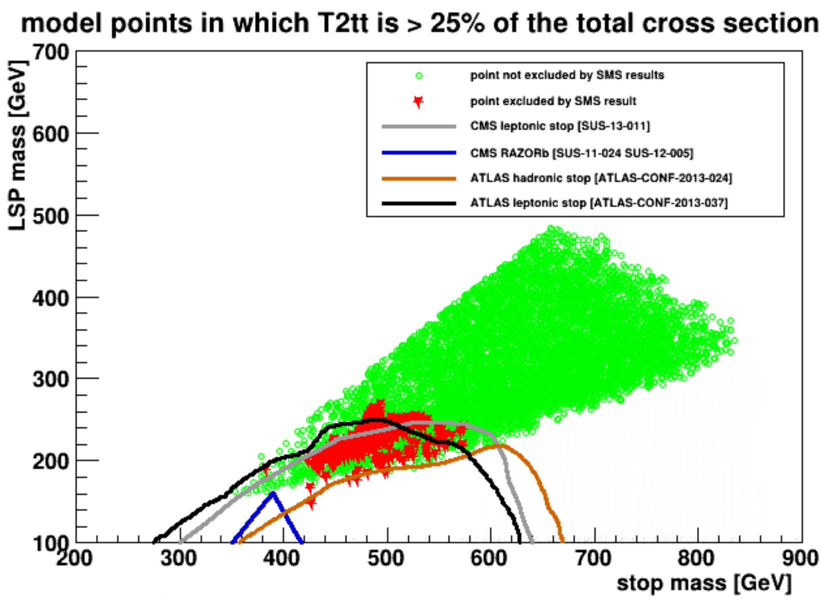


Figure 6.4: Points with a *T2tt* fraction $> 25\,\%$ found to be excluded by `SModelS`. The colored lines represent the exclusion lines reported by following analyses CMS-SUS-13-011 [50] (gray), CMS-SUS-12-005 [35] (blue), ATLAS-CONF-2013-024 [65] (orange) and ATLAS-CONF-2013-037 [56] (black).

6.2 CMS Summary Plots

Another application of the `SModelS` framework within the scope of this thesis is the creation of the CMS summary plots. To this aim the database and its Python interface *SMSResults* are used. Compared to the whole `SModelS` framework, the database can also be used for R-parity violating results. The bar plots summarize all CMS results of SUSY searches (Fig. 6.5 shows the R-parity conserving results and Fig. 6.6 shows the R-parity violating results). Their purpose is to give an overview of all available CMS SUSY results. The dark bars mark the masses of the mother sparticles up to which an analysis can exclude any new physics considering a massless LSP. The brighter bars mark the maximum exclusions at a mass splitting between mother sparticle and LSP of 200 GeV. The bars are sorted by the mother sparticles of the used model (from top to bottom: gluino, squark, stop, sbottom, EWK gaugino and slepton production). For further details on the different analyses see the publications.

Due to the fact that all public CMS results are available in the database, every information in the summary plots can be read out using *SMSResults*. One can ask for all results using *SMSResults.getAllResults()*, which returns a python dictionary containing all available analyses as key and a list of all topologies as value for the selected runs. Looping over all of the analysis-topology pairs and applying *SMSResults.getExclusion()* to each of them, returns three different limits on the mass of the mother sparticle for the exclusion assuming three different constraints for the LSP mass e.g. considering a mass splitting of 200 GeV between mother and LSP mass or considering a massless LSP.

More detailed are the summary plots for one specific topology, e.g. *T1tttt* in Fig. 6.7. These plots show all available exclusion lines published by the different analyses.

This type of summary plots are created similarly to the bar plot. The functions used are almost the same. The only difference is that instead of using *SMSResults.getExclusion()*, *SMSResults.getExclusionLine()* is used. *SMSResults.getExclusionLine()* returns a TGraph (a ROOT object), so it is favorable to use ROOT to create these plots.

More summary plots are published on the public CMS SUSY website [77].

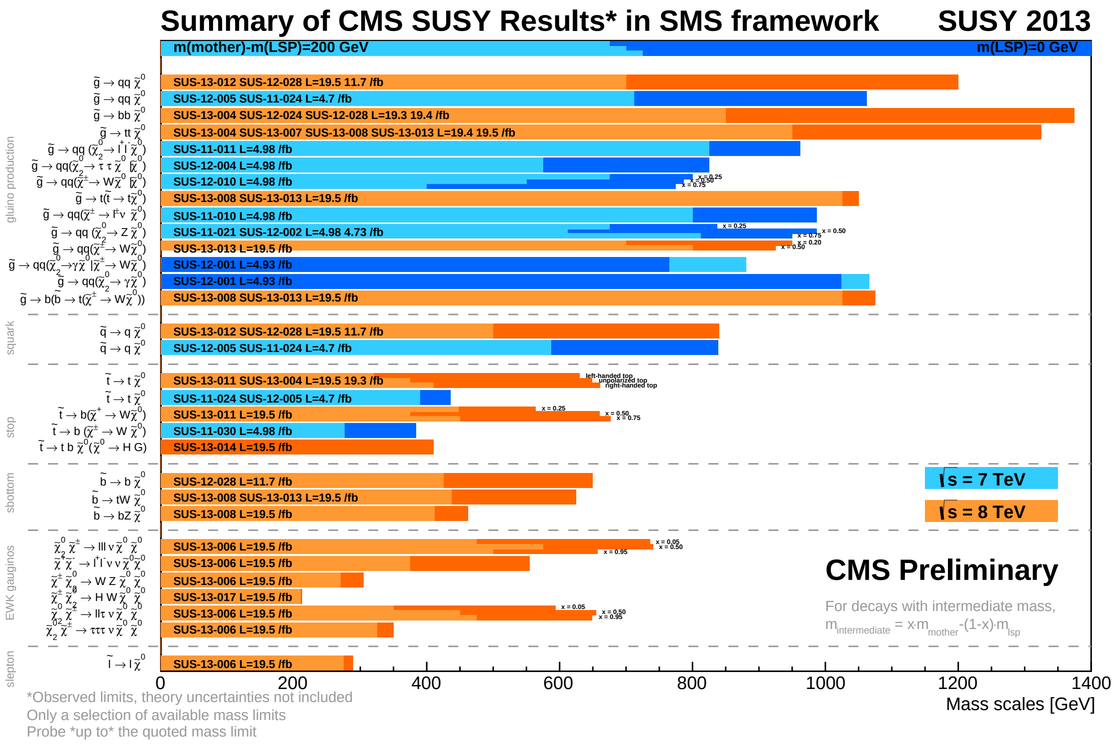


Figure 6.5: Summary plot of R-parity conserving CMS SUSY results [77].

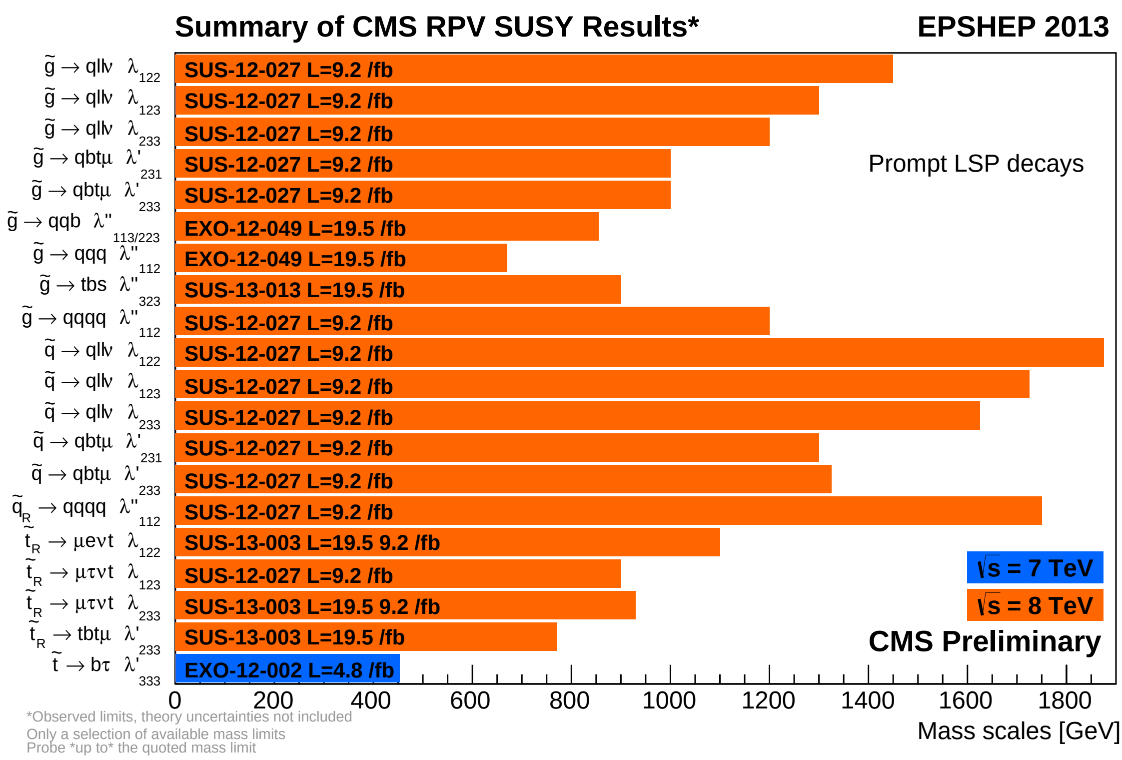


Figure 6.6: Summary plot of R-parity violating CMS SUSY results [77].

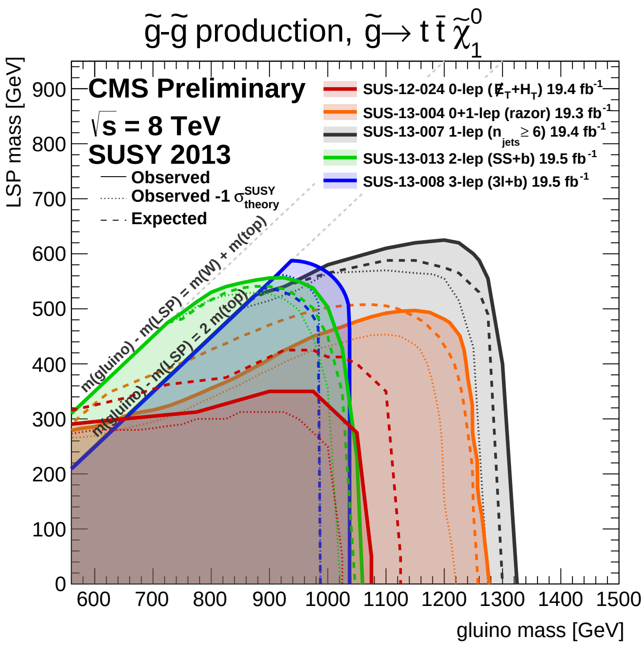


Figure 6.7: Summary plot of all CMS results for T1tttt [77].

7 Conclusion

Within the scope of this thesis a framework for the analysis of results of BSM searches was developed and tested. This framework is called `SModelS`. The `SModelS` framework decomposes a given BSM model point into its SMS and computes the weight ($\sigma \times \mathcal{B}$) for each SMS. These weights can be compared to the upper limits on the production cross section reported by the experimental searches. Therefore, a database is available containing all recent results from CMS and ATLAS SUSY searches. The `SModelS` framework was validated by some tests. The goal of these tests was to reproduce the exclusion lines which are reported by the experiment using the `SModelS` framework. Also, a first scan over a constrained pMSSM model was done using the `SModelS` framework. The CMS SUSY summary plots are created by using the `SModelS` database and its Python interface *SMSResults*. The application of `SModelS` to these two very different tasks demonstrates the versatility of the framework.

In the future the functionality of the `SModelS` framework will be increased. Soon, the LHE decomposition will be validated so that the user will have the choice between SLHA and LHE decomposition. Also the `SModelS` framework will be tested for other BSM models besides SUSY (e.g., UED) and if needed it will be extended. Another task is to validate the SMS approach in the presence of spin effects and effects of off-shell states. Currently, only the 95 % upper limits on the production cross section are used within `SModelS`. It is deliberated whether to substitute the 95 % upper limits by likelihoods, which would contain more information on the experimental result. Another idea is to implement efficiency maps. The efficiency maps could be used to combine several global topologies. At the moment, the `SModelS` framework uses only negative results from the experimental SUSY searches. In case of a positive experimental result, `SModelS` should be able to make predictions about best fitting parameter regions. Therefore, the code needs to be adapted.

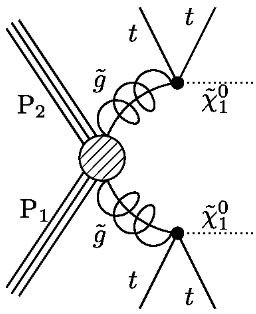
From the physics side, more extensive scans are planned, e.g., over a Natural SUSY model. Also a cooperation with the Fittino group is envisaged. The Fittino group [78] is extracting Largangian parameters from experimental results. Therefore, they combine likelihoods obtained by various experimental analyses.

A SMSDictionary

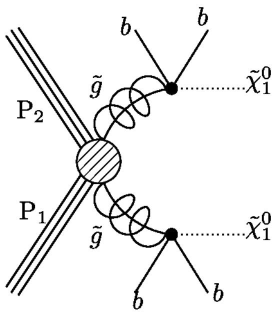
This chapter shows the Feynman-like diagrams and their Tx names for all simplified models, which are currently available in the `SModelS` database. The list of SMS is sorted by their production mode.

A.1 Gluino Production

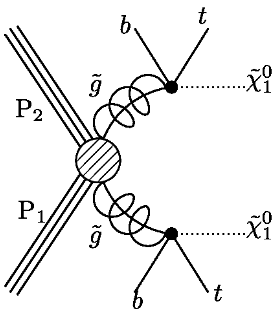
T1tttt



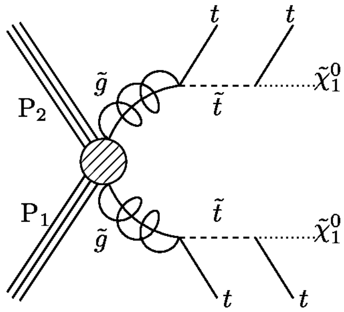
T1bbbb



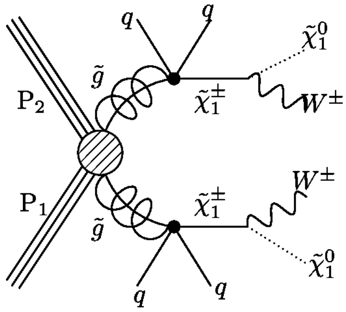
T1tbtb



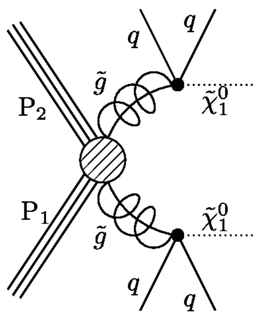
T5tttt



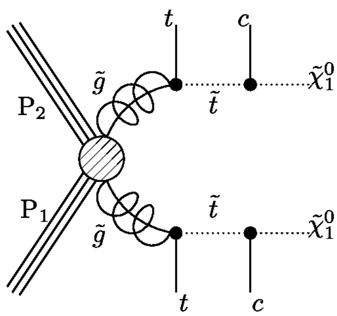
T5WW



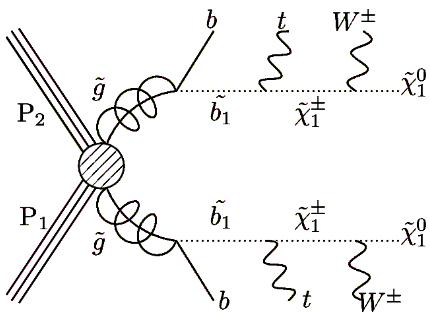
T1



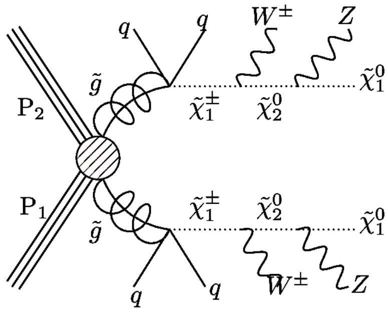
T5tctc



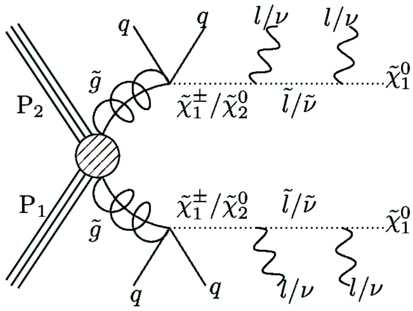
T7btbtWW



T7WWZZ

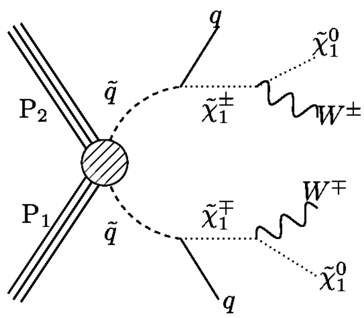


T7ChiChipmSlepL

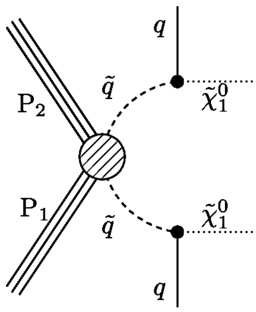


A.2 Light Squark Production

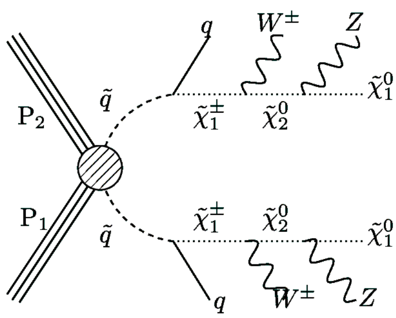
T6WW



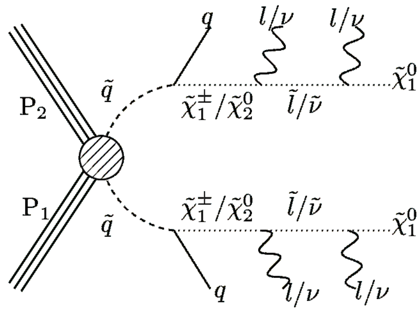
T2



T8WWZZ

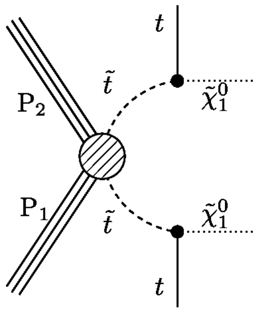


T8ChiChipmSlepL

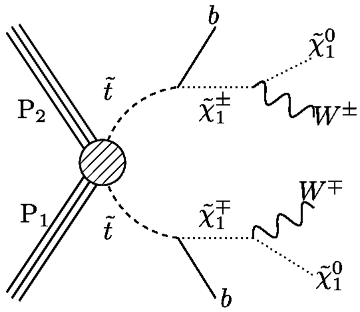


A.3 Stop Production

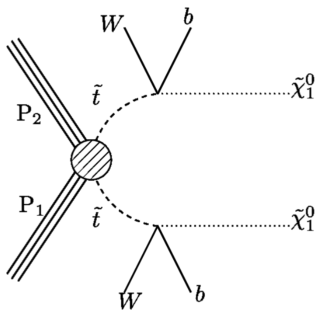
T2tt



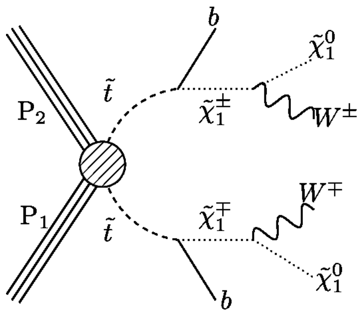
T6bbWW



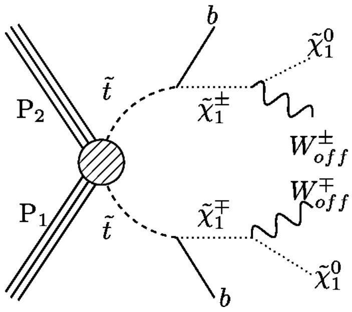
T2bbWW



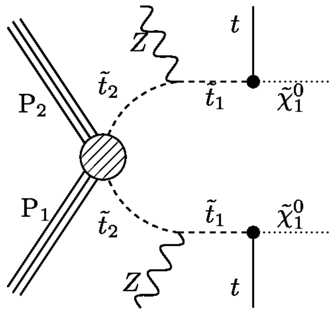
T6bbWWon



T6bbWWoff

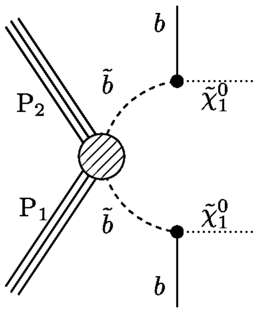


T6ttZZ

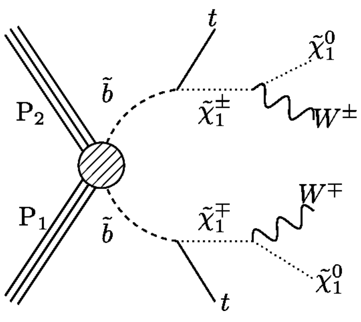


A.4 Sbottom Production

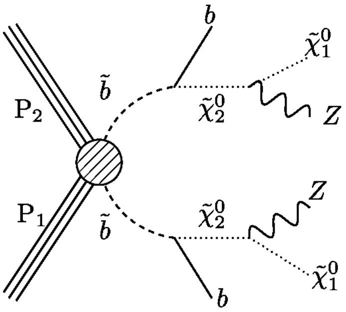
T2bb



T6ttWW

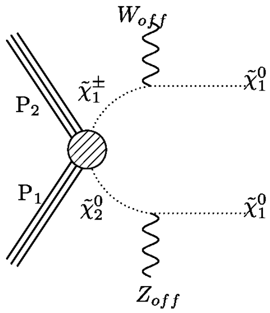


T6bbZZ

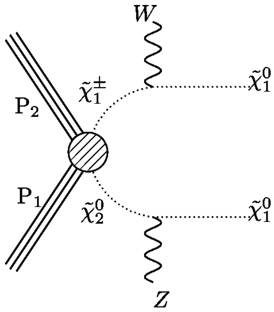


A.5 Electroweakino Production

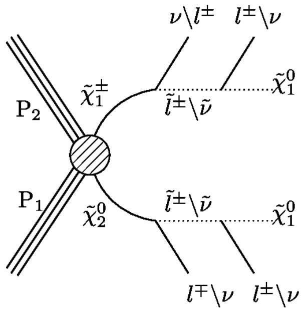
TChiWZoff



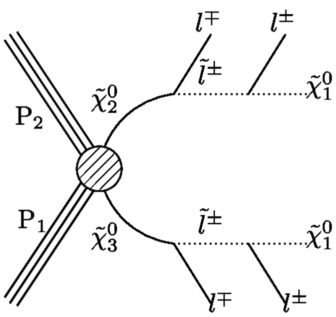
TChiWZon



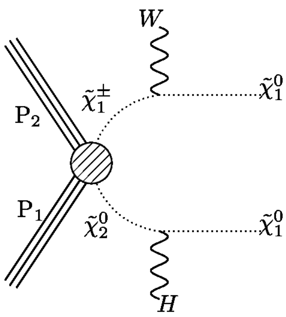
TChiChipmSlepL



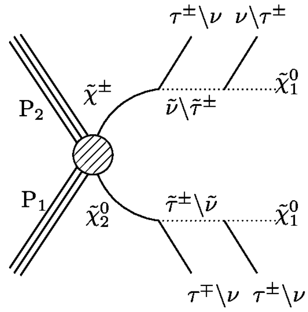
TChiChiSlepSlep



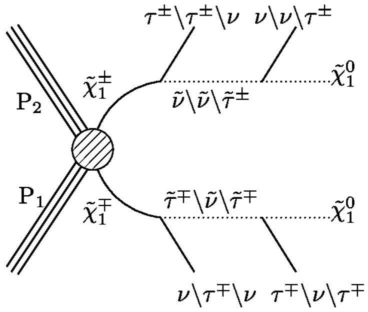
TChiChipmHW



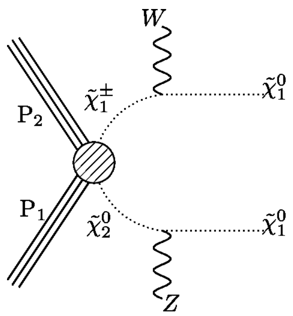
TChiChipmStauL



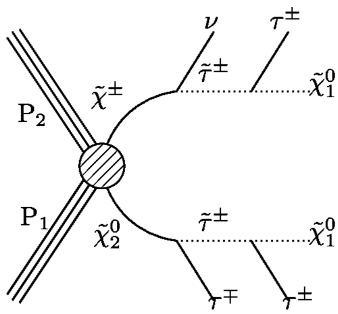
TChipChimStauSnu



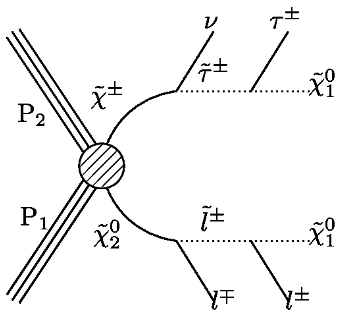
TChiWZ



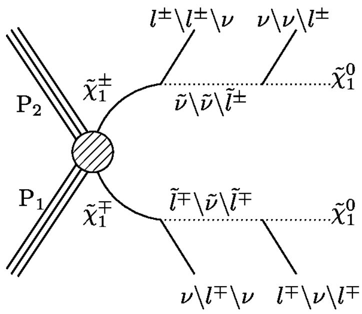
TChiChipmStauStau



TChiChipmSlepStau

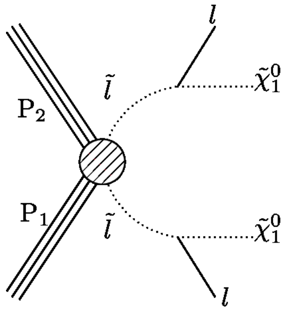


TChipChimSlepSnu



A.6 Slepton Production

TSlepSlep



B SModelS Functions

In this chapter all functions of the SModelS framework are listed. The functions are sorted by the three main packages and listed according to their modules. The SModelS functions which might be interesting for the user are shortly described.

B.1 Experiment

SMSInterpolation

gethistname (topo, mz)	
getxval (mx, my, mz, mass = False)	
getindex (ls, second = False)	
getaxis (w, a)	
compareM (masses, d)	
dogriddata (ana, topo, masses, d, debug = True, run = None)	Returns an interpolated $\sigma \times \mathcal{B}$ UL.
UpperLimit (ana, topo, masses, debug = True, run = None)	Checks the mass array and returns a $\sigma \times \mathcal{B}$ UL.

SMSAnalysisList_new

load ()	Initializes an EAnalysis object.

TxNames

getTx (element)	
ptsCount (b1, b2)	
getT1 (pts)	
getT3 (pts, b2)	
getT5 (pts)	
getT2 (pts)	
getT4 (pts)	
getT6 (pts)	

SMSResults

describeTx (topo, short = True)	Returns the decay chain of a given Tx name.
getAllHistNames (ana, topo, run = None)	Returns a list of all available $\sigma \times \mathcal{B}$ UL histograms.
setLogLevel (l = ["error"])	Sets the level for the logger.
setBase (base)	Defines the base directory of the database.
useUnits (b = True)	Turns the use of units on/off.
considerRuns (run = None)	Defines what runs are considered when asking for results.
verbosity (level = "error")	
getExclusion (analysis, topo, run = None)	Returns the exclusions on the mother particle.
getBinWidthX (analysis, topo, run = None)	Returns the bin width of a $\sigma \times \mathcal{B}$ UL histogram on the x-axis.
getLowX (analysis, topo, run = None)	Returns the lower edge of the x-axis of a $\sigma \times \mathcal{B}$ UL histogram.
getUpX (analysis, topo, run = None)	Returns the upper edge of the x-axis of a $\sigma \times \mathcal{B}$ UL histogram.
getLowY (analysis, topo, run = None)	Returns the lower edge of the y-axis of a $\sigma \times \mathcal{B}$ UL histogram.
getUpY (analysis, topo, run = None)	Returns the upper edge of the y-axis of a $\sigma \times \mathcal{B}$ UL histogram.
getBinWidthY (analysis, topo, run = None)	Returns the bin width of a $\sigma \times \mathcal{B}$ UL histogram on the y-axis.
getExclusionLine (topo, ana, expected = False, plusminussigma = 0, extendedinfo = False, xvalue = None, factor = 1.0)	Returns the exclusion line as TGraph.
getTopologies (analysis, run = None, allHistos = False)	Returns all topologies of a certain analysis.
getRun (analysis, run = None)	Returns the corresponding run for an analysis.
getExperiment (analysis, run = None)	Returns the name of the experiment which did the analysis.

getPrettyName (analysis, run = None, latex = False)	Returns a describing name for each analysis.
getAnalyses (topo, run = None)	Returns a list of all analyses which have results for a certain topology.
getAllResults (run = None, allHistos = False)	Returns all analyses and all topologies which are stored in the database.
getDatabaseResults (run = None, category = None)	Returns all public analyses and topologies stored in the database for which constraints are defined.
getClosestValue (Dict, mx, my)	Returns the $\sigma \times \mathcal{B}$ UL of the point closest to a given one.
inConvexHull (Dict, mx, my)	
getInterpolatedUpperLimitDelaunay (Dict, inmx, inmy)	Returns the interpolated $\sigma \times \mathcal{B}$ UL from a Python dictionary *sms.py*.
getInterpolatedUpperLimit (Dict, inmx, inmy)	Returns the interpolated $\sigma \times \mathcal{B}$ UL from a Python dictionary *sms.py*.
getUpperLimitFromDictionary (analysis, topo, mx = None, my = None, run = None, png = None, interpolate = False, expected = False)	
getSmartUpperLimit (analysis, topo, masses, massesbranch2 = None, debug = False)	
getUpperLimit (analysis, topo, mx = None, my = None, run = None, png = None, interpolate = False, expected = False)	Returns the $\sigma \times \mathcal{B}$ UL.
getEfficiency (analysis, topo, mx = None, my = None, run = None)	Returns the efficiency.
getExplanationForLackOfUpperLimit (analysis, topo, mx = None, my = None, run = None, number = False)	Returns an explanation if no $\sigma \times \mathcal{B}$ UL can be found.
isPrivate (analysis, run = None)	Checks if an analysis is flagged as private.
isPrivateTopology (analysis, topology, run = None)	Checks if a certain topology of an analysis is flagged as private.

isPublic (analysis, run = None)	Checks if an analysis is not flagged as private.
hasDataPublished (analysis, run = None)	Checks if the analysis has published their results in digital form.
getLumi (analysis, run = None)	Returns the integrated luminosity for a certain analysis.
getSqrts (analysis, run = None)	Returns the center of mass energy for a certain analysis.
getPAS (analysis, run = None)	Returns the PAS number of a certain analysis.
isSuperseded (analysis, run = None)	Checks if an analysis is already superseded by a newer one.
getOrder (analysis, run = None)	Returns the order in perturbation theory of the exclusion lines.
hasDictionary (analysis, run = None)	checks if the $\sigma \times \mathcal{B}$ UL information is available in form of a Python dictionary.
getx (analysis, topo = None, run = None)	Returns the x-values used in an analysis.
getFigures (analysis, run = None)	Returns the figure numbers for an analysis.
getComment (analysis, run = None)	Returns additional comments, if available.
getConditions (analysis, topo = ”all”, fuzzy = True, run = None)	Returns the conditions.
getConstraints (analysis, topo = ”all”, run = None)	Returns the constraints.
getCategories (analysis, topo = ”all”, run = None)	Returns the category of a certain topology.
getRequirement (analysis, run = None)	Returns additional requirements from the analysis, if available.
getCheckedBy (analysis, run = None)	Returns the name of the person who validated the database entry.
getJournal (analysis, run = None)	Returns the journal of an analysis.
getBibtex (analysis, run = None)	Returns the inspire page with the bibtex entry for an analysis.
getURL (analysis, run = None)	Returns the URL for an analysis.

hasURL (analysis, run = None)	Checks if the URL for an analysis is known.
getContact (analysis, run = None)	Returns the contact for a certain analysis.
getPerturbationOrder (analysis, run = None)	Returns the order in perturbation theory of the $\sigma \times \mathcal{B}$ UL.
particles (topo, plot = 'ROOT')	Returns the production mode of a given topology.
particleName (topo)	Returns the production mode for a given topology written out.
massDecoupling_ (topo)	
massDecoupling (topo, plot = 'ROOT', kerning = True)	Describes the assumed mass decoupling.
exists (analysis, topo, run = None)	Checks if all $\sigma \times \mathcal{B}$ information exists listed in the axes-information.
hasExclusionLine (ana, topo)	Checks if an exclusion line exists for a certain topology.
getaxes (analysis, topo = None, run = None)	Returns the information about the $\sigma\times\mathcal{B}$ UL histogram axes.

SMSHelpers

useROOT (b)	
close ()	
log (text, level = "error")	
getRun (analysis, run = None)	
parseMetaInfo (analysis, run)	
motherParticleExclusions (analysis, run)	
getLines (analysis, run, label = "condition")	
conditions (analysis, run)	
fuzzyconditions (analysis, run)	
constraints (analysis, run)	
categories (analysis, run)	
getPotentialNames (topo)	
getCanonicalName (topo)	

getRootFileName (analysis, run = None)	
getUpperLimitFromHisto (analysis, topo, run, complain = False, expected = False)	
getUpperLimitAtPoint (histo, mx, my, interpolate = False)	
getUpperLimitPng (analysis, topo, run)	
getEfficiencyHisto (analysis, topo, run)	
getEfficiencyAtPoint (histo, mx, my)	
getErrorMessage (histo, mx, my)	
hasMetaInfoField (analysis, field, run = None)	
getMetaInfoField (analysis, field, run = None)	
hasDictionary (analysis, run = None)	
hasHistogram (analysis, run = None)	
getUpperLimitDictionary (analysis, topo, run, expected = False)	

SMSAnalysisList

load ()	Initializes an EAnalysis object.

SMSAnalysisFactory

getRealTopo (Tx)	
getArray (constraint)	
load (anas = None, topos = None, sqrts = [7, 8])	Creates the EAnalysis objects from the information given in the database.

SMSResultsCollector

SMSInfo (obj, topo = None, ana = None, xvalue = '', plot = 'ROOT', kerning = True, year = None, name = None)	
analyses (topo = None, year = None)	

topos (ana = None, Masssplitting = False)
SMSObjects (obj, topo, ana, xvalue = ”, name = None)
AnalysisInfo (obj, ana)
pas (ana)
lumi (ana)
order (ana)
perturbationOrder (ana)
contact (ana)
isPublic (topo, ana, xvalue = ”)
analysisname (ana, plot)
particleName (topo)
particles (topo, plot = ’ROOT’)
production (topo, plot = ’ROOT’)
description (topo, plot = ’ROOT’, kerning = True, short = False)
decays (topo, plot = ’ROOT’, kerning = True, omitleft = False)
massDecoupling_ (topo)
massDecoupling (topo, plot = ’ROOT’, kerning = True)
exclusions (topo, ana, xvalue = ”, expected = False, plusminussigma = 0)
ul (topo, ana, xvalue = ”)
upperlimit (topo, ana, xvalue = ”)
exclusionline (topo, ana, xvalue = ”, factor = 1.0, extendedinfo = True, expected = False, plusminussigma = 0)
efficiency (topo, analysis, xfrac, histo = ’efficiency’)
getUpperLimit (topo, analysis, xfrac, mx = None, my = None)

LimitGetter

limit (analysis, addTheoryPredictions = [])	
GetPlotLimit (inmass, Analysis, complain = False)	Returns the $\sigma \times \mathcal{B}$ upper limit of a certain mass array.

B.2 Theory

NLLXSec

getNLLfast (process = "gg", pdf = 'cteq', squarkmass = 0., gluinomass = 0., Energy = 8, base = "./nllfast/")	Runs NLLfast.
getNLLresult (pdgid1, pdgid2, inputfile, base = "../nllfast", pdf = "cteq")	Returns the production cross section for a given pair production.

LHEDecomposer

decompose (lhefile, W = None, nevts = None, doCompress = False, doInvisible = False, minmassgap = None)	Does the LHE file based decomposition.

SLHADecomposer

decompose (slhafile, Xsec = None, sigcut = None, DoCompress = False, DoInvisible = False, minmassgap = -1, XsecsInfo = None)	Does the SLHA file based decomposition.

ClusterTools

DoCluster (objlist, Distfunc, dmin, AvgFunc = None, PosFunc = None)	Clusters the topologies and their elements and computes avarage masses.
GoodMass (mass, Distfunc, dmin)	Checks if two elements have similar branch masses if they have the same topology. In case they have different topologies, it checks if they have similar mother and LSP masses.
MassAvg (equivin, method = "harmonic")	Computes an average mass array for a list of equivalent masses.
sumweights (wlist)	Sums a list of weights.

ClusterDist (cluster1, cluster2, MD)	Defines the distance between two clusters.

TopologyBuilder

fromEvent (Event, weight = , DoCompress = False, DoInvisible = False,	Creates a topology from an event.
compressTopology (ETopList, DoCompress, DoInvisible, minmassgap)	Compresses the elements from a given list of topologies if possible.

SLHATools

createSLHAFile (topo, masses = None, filename = None, branching = None, totalwidth = None)	Creates an SLHA file for a certain SMS and certain masses.
writeXSecToSLHAFile (slhafile, nevts = 10000, basedir = None, XsecsInfo = None, printLHE = True)	Computes all occuring pair production cross sections and writes them in form of an XSECTION block to the SLHA file.
num_in_base (val, base = 62, min_digits = 1, complement = False,	
uniqueName (slhafile, blocks = "MINPAR":[3], "EXTPAR":[31, 32, 33, 34, 35, 36, 41, 42, 43, 44, 45, 46, 47, 48, 49, 23, 26])	Creates an unique name for an SLHA file.
xSecFromSLHAFile (slhafile)	Creates a CrossSection object from an SLHA file.

AuxiliaryFunctions

Csim (*els)	
Cgtr (a, b)	
similar (els)	
Ceval (instring, nEl)	
getelements (instring)	
eltonum (instring, dic)	

ParticleNames

getName (pdg)	Converts a PDG id number into the particle name.

getPdg (name)	Converts the particle name into the PDG id number.
simParticles (ptype1, ptype2, useDict = True)	Compares two particle names.

XSecComputer

loFromLHE (lhefile, totalxsec, nevts = None)	Computes the LO weights for each pair production process from an LHE file.
compute (nevts, slhafile, rpythia = True, basedir = None, datadir = None, XsecsInfo = None, tmpfiles = False, printLHE = True)	Computes the weights for each pair production process by running PYTHIA at 7 and 8 TeV.
getprodcs (pdgm1, pdgm2, sigma):	
runPythia (slhafile, n, sqrts = 7, datadir = "./data/", etcdir = "./etc/",	Runs PYTHIA.
clean (datadir)	Cleans up after having computed everything.

B.2.1 Class Modules

SMSAnalysis

EAnalysis

__init__ (self)	
__str__ (self)	
generateElements (self)	Creates a list of AElements available from the Database using the constraints and conditions.
getPlots (self, verbose = True)	
add (self, SMSTopList)	Adds the EElements to the list of AElements.
computeTheoryPredictions (self, keepMassInfo = False)	Evaluates the theoretical predictions of the production cross section and creates the ResultList.
split (self)	

MassDist (self, mass1, mass2)	Calculates the distance between two mass arrays by using the experimental limits.
MassPosition (self, mass, nounit = True)	Gives the position of the mass array in the upper limit space.

SMSDataObjects

BElement

__init__ (self, S = None)	
isEqual (ElA, ElB, order = True)	
__eq__ (self, other)	
isSimilar (self, elB, order = True, igmass = False)	Compares two BElements. Checks if particles are similar and masses are equal.
__str__ (self)	
describe (self)	Describes the BElement.

EElement

__init__ (self, S = None)	
getEinfo (self)	Returns the number of vertices and the SM particles at each vertex.
allParticles (self)	Returns all particles.
isSimilar (ElA, ElB, order = True, igmass = False)	Compares two EElements. Checks if particles are similar and masses are equal.
isEqual (ElA, ElB, order = True)	Compares two EElements. Checks if particles and masses are equal.
__eq__ (self, other)	
__str__ (self)	
describe (self)	Describes the EElement.
isInAnalysis (self, Analysis, igmass = False)	Checks if the EElement is present in the list of AElements.

AElement

__init__ (self, PartStr = "")	
getParticleList (self)	Returns a list containing all particles.

getEinfo (self)	Returns the number of vertices and the particles at each vertex.

CElement

__init__ (self, PartStr = None, weight = None)	
getParticleList (self)	Returns a list containing all particles.
getEinfo (self)	Returns the number of vertices and the particles at each vertex.

MassWeight

__init__ (self, mass = None, weight = None)	

CTop

__init__ (self, Top = None, masscluster = None)	
addAnalysisElement (self, NewElement, masscluster)	Adds an AElement to the identical CElement if it exists.
evaluateCluster (self, results)	Creates an XSecPredictionForCluster object using constraints, conditions and the prediction for the production cross section for each CElement.

ATop

__init__ (self)	
isEqual (self, Top2, order = False)	Checks if two topologies have the same number of vertices and the same particles.
__eq__ (self, other)	
leadingElement (self)	
addEventElement (self, NewElement, sqrts)	Adds an EElement to its corresponding AElement.

GTop

__init__ (self)	
leadingElement (self)	

elements (self)	Returns all available EElements.
describe (self)	Describes the GTop.
__str__ (self)	
checkConsistency (self, verbose = False)	Checks if all Elements of the topology have the same number of vertices and insertions per vertex.
isEqual (self, Top2, order = False)	Checks if two topologies have the same number of vertices and the same particles.
__eq__ (self, other)	
addElement (self, NewElement)	Adds a new element to the GTop.
massCompressedTopology (self, mingap)	Creates a compressed copy of this topology if two masses are degenerate.
invisibleCompressedTopology (self)	Creates a compressed copy of this topology if particles at the end of the decay chain are invisible for the detector.

TopologyList

__init__ (self, topos = [])	
__len__ (self):	
__getitem__ (self, n)	
addList (self, List)	
__str__ (self)	
describe (self)	Returns a list of all available topologies.
add (self, topo)	Adds a GTop to the list. If the topology exists already, it only adds the weights and new elements to the topology.

CrossSection

CrossSection

__init__ (self, data)	
__len__ (self)	
__getitem__ (self, i)	
items (self)	
__str__ (self)	
keys (self)	

weights (self)	Returns all available weights.
crossSections (self)	Returns all available production cross sections.
crossSectionsInfo (self)	Returns the available center of mass energies, orders and labels.
getCrossSection (self, pidmom1, pidmom2, order = "NLL", sqrts = 8)	Returns the production cross section for a given pair production.
getSumOfCrossSections (self, pidmoms = None, order = "NLL", sqrts = 8)	Returns the integrated production cross section of all given pdg ids.
crossSectionLightSquarks (self, order = "NLL", sqrts = 8)	Returns the integrated production cross section of all light squark productions.
lhefile (self, sqrts = 8, order = 0)	Returns the name of the corresponding LHE file.

SingleXSecInfo

__init__ (self, Str = None)	
__eq__ (self, other)	
__str__ (self)	

XSecInfoList

__init__ (self, Str = '7TeV (LO), 8TeV (LO), 7TeV (NLL), 8TeV (NLL)')	
__getitem__ (self, label = None)	
sort (self, reverse = False)	Sorts available production cross sections by their center of mass energy.
getSqrts (self)	Returns all used center of mass energies.

TheoryPrediction

XSecPredictionForCluster

__init__ (self)	
__str__ (self)	
describe (self)	Returns the basic informations about this XSecPredictionForCluster object.
oldformat (self)	
getConditionValues (self, wlabel = '')	Returns a list of condition values.

getMaxCondition (self, wlabel = None)	Returns the maximum condition value.
prediction (self, wlabel = None)	Returns the prediction for the production cross section.

B.3 Tools

PhysicsUnits

addunit (value, unitstring)	Creates an Unum object from a float by adding a given unit
rmvunit (value, unitstring)	Turns an Unum object into float by removing the unit

FeynmanGraphs

printParticle_ (label)	
segment (p1, p2, spin, Bend = None)	
zero ()	
connect (canvas, p1, p2, straight = True, label = None, spin = "fermion", bend = True,	
draw (element, filename = "bla.pdf", straight = False, inparts = True)	Draws a Feynman-like diagram.
drawBranch_ (branch, upwards, labels)	
asciidraw (element, labels = True)	Draws a simple ascii graph on the screen.

TexTable

GenAnalysisTable (ListOfAnalyses, texfile = None, wd = 0.15, fig_dir = None)	Generates a latex table with all analyses in ListOfAnalyses and their SMS
PrintAnalysisGraphs (Analysis, pdf_prefix = None)	Generates a pdf file with the graphs of the SMS topologies.

ROOTTools

getTGraphFromContour (exclhisto)	Returns the contour of an histogram as TGraph.

useNiceColorPalette (palette = ”temperature”, f = 0., ngradientcolors = 20)	Creates a fine-grained temperature color palette.

RCFile

yesno (B)	

SMSPrettyPrinter

wrap_onspace (text, width)	
wrap_always (text, width)	
wrap (text, width)	

VariousHelpers

getMaxLum (List)	Returns the maximum luminosity from a list of different analyses.
getInstallationBase ()	Returns the name of the base directory of the installation of the framework.
nCPUs ()	Returns the number of CPU cores on the machine.

List of Abbreviations

ALICE	A Large Ion Collider Experiment
ATLAS	A Toroidal LHC ApparatuS
CERN	Conseil Européen pour la Recherche Nucléaire
CMS	Compact Muon Solenoid
cMSSM	constraint Minimal Supersymmetric Standard Model
CSC	Cathode Strip Chambers
DT	Drift Tube Chambers
ECAL	Electromagnetic Calorimeter
FCNC	Flavor Changing Neutral Current
GUT	Grand Unified Theory
HCAL	Hadron Calorimeter
HLT	High-Level Trigger
LEP	Large Electron-Positron Collider
LHC	Large Hadron Collider
LHCb	Large Hadron Collider beauty
LHCf	Large Hadron Collider forward
Linac	Linear Accelerator
LO	leading order
LSP	lightest supersymmetric particle
MC	Monte Carlo
MoEDAL	Monopole and Exotics Detector at the LHC
MSSM	Minimal Supersymmetric Standard Model
NLL	next to leading logarithmic order
NLO	next to leading order
NUHM	Non Universal Higgs Mass
PDG id	The particle code according to the Particle Data Group (PDG) [33] convention
pMSSM	phenomenological Minimal Supersymmetric Standard Model
PS	Proton Synchrotron
PSB	Proton Synchrotron Booster

QCD	Quantum Chromo Dynamics
QFT	Quantum Field Theory
RG	Renormalization Group
RPC	Resistiive plate chamber
SM	Standard Model
SMS	Simplified Model Spectra
SPS	Super Proton Synchrotron
SUSY	supersymmetry
TOTEM	Total elastic and diffractive cross-section measurement
VEV	Vacuum Expectation Value

List of Figures

List of Tables

Bibliography

[1] **CMS** Collaboration, S. Chatrchyan *et al.*, "Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC," *Phys.Lett.* **B716** (2012) 30–61, arXiv:1207.7235 [hep-ex].

[2] **ATLAS** Collaboration, G. Aad *et al.*, "Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC," *Phys.Lett.* **B716** (2012) 1–29, arXiv:1207.7214 [hep-ex].

[3] S. Kraml, S. Kulkarni, U. Laa, A. Lessa, W. Magerl, D. Proschofsky-Spindler, and W. Waltenberger, "SModelS: a tool for interpreting simplified-model results from the LHC and its application to supersymmetry," arXiv:1312.4175 [hep-ph].

[4] L. Evans and P. Bryant, "LHC Machine," *JINST* **3** (2008) S08001.

[5] C. Lefèvre, "The CERN accelerator complex. Complexe des accélérateurs du CERN." Dec, 2008.

[6] Oct, 2013. http://lhc-machine-outreach.web.cern.ch/lhc-machine-outreach/images/lh%c-schematic.jpg.

[7] **TOTEM** Collaboration, "Luminosity-independent measurements of total, elastic and inelastic cross-sections at $\sqrt{s} = 7$ TeV," Tech. Rep. TOTEM-2012-004. CERN-PH-EP-2012-353, CERN, Geneva, Nov, 2012.

[8] **TOTEM** Collaboration, "Luminosity-Independent Measurement of the Proton-Proton Total Cross Section at $\sqrt{s} = 8$ TeV," *Phys. Rev. Lett.* **111** no. TOTEM-2012-005. CERN-PH-EP-2012-354, (Nov, 2012) 012001. 8 p.

[9] CERN AC, "Layout of CMS." Mar, 1998.

[10] **CMS** Collaboration, S. Chatrchyan *et al.*, "The CMS experiment at the CERN LHC," *JINST* **3** (2008) S08004.

[11] **CMS** Collaboration, G. L. Bayatian *et al.*, *CMS Physics: Technical Design Report Volume 1: Detector Performance and Software*. Technical Design Report CMS. CERN, Geneva, 2006.

[12] Oct, 2013. `https://cms-docdb.cern.ch/cgi-bin/PublicDocDB/ShowDocument?docid=5581`.

[13] CERN AC, "Layout of ATLAS. Dessin representant le detecteur ATLAS." Mar, 1998.

[14] **ATLAS** Collaboration, G. Aad *et al.*, "The ATLAS Experiment at the CERN Large Hadron Collider," *JINST* **3** (2008) S08003.

[15] **ATLAS** Collaboration, "ATLAS: Detector and physics performance technical design report. Volume 1,".

[16] A. Purcell, "Go on a particle quest at the first CERN webfest. Le premier webfest du CERN se lance à la conquête des particules,".

[17] S. P. Martin, "A Supersymmetry primer," `arXiv:hep-ph/9709356 [hep-ph]`.

[18] **MSSM Working Group** Collaboration, A. Djouadi *et al.*, "The Minimal Supersymmetric Standard Model: Group Summary Report," `arXiv:hep-ph/9901246 [hep-ph]`.

[19] M. Papucci, J. T. Ruderman, and A. Weiler, "Natural SUSY Endures," *JHEP* **1209** (2012) 035, `arXiv:1110.6926 [hep-ph]`.

[20] J. Alwall, P. Schuster, and N. Toro, "Simplified Models for a First Characterization of New Physics at the LHC," *Phys.Rev.* **D79** (2009) 075020, `arXiv:0810.3921 [hep-ph]`.

[21] N. Arkani-Hamed, P. Schuster, N. Toro, J. Thaler, L.-T. Wang, *et al.*, "MARMOSET: The Path from LHC Data to the New Standard Model via On-Shell Effective Theories," `arXiv:hep-ph/0703088 [HEP-PH]`.

[22] **LHC New Physics Working Group** Collaboration, D. Alves *et al.*, "Simplified Models for LHC New Physics Searches," *J.Phys.* **G39** (2012) 105005, `arXiv:1105.2838 [hep-ph]`.

[23] "Interpretation of Searches for Supersymmetry," Tech. Rep. CMS-PAS-SUS-11-016, CERN, Geneva, 2012.

[24] B. Allanach, "SOFTSUSY: a program for calculating supersymmetric spectra," *Comput.Phys.Commun.* **143** (2002) 305–331, arXiv:hep-ph/0104145 [hep-ph].

[25] M. Mühlleitner, "SDECAY: A Fortran code for SUSY particle decays in the MSSM," *Acta Phys.Polon.* **B35** (2004) 2753–2766, arXiv:hep-ph/0409200 [hep-ph].

[26] T. Sjostrand, S. Mrenna, and P. Z. Skands, "PYTHIA 6.4 Physics and Manual," *JHEP* **0605** (2006) 026, arXiv:hep-ph/0603175 [hep-ph].

[27] W. Beenakker, R. Hopker, and M. Spira, "PROSPINO: A Program for the production of supersymmetric particles in next-to-leading order QCD," arXiv:hep-ph/9611232 [hep-ph].

[28] Nov, 2013. http://www.thphys.uni-heidelberg.de/~plehn/index.php?show=prospino\&vi%sible=tools.

[29] M. Kramer, A. Kulesza, R. van der Leeuw, M. Mangano, S. Padhi, *et al.*, "Supersymmetry production cross sections in pp collisions at $\sqrt{s} = 7$ TeV," arXiv:1206.2892 [hep-ph].

[30] P. Z. Skands, B. Allanach, H. Baer, C. Balazs, G. Belanger, *et al.*, "SUSY Les Houches accord: Interfacing SUSY spectrum calculators, decay packages, and event generators," *JHEP* **0407** (2004) 036, arXiv:hep-ph/0311123 [hep-ph].

[31] J. Alwall, A. Ballestrero, P. Bartalini, S. Belov, E. Boos, *et al.*, "A Standard format for Les Houches event files," *Comput.Phys.Commun.* **176** (2007) 300–304, arXiv:hep-ph/0609017 [hep-ph].

[32] W. Magerl, "Restructuring a Software Framework for Interpretation of Searches for Physics Beyond the Standard Model," Bachelor's thesis, Technische Universität Wien, 2014. in progress.

[33] "Particle Data Group," Dec, 2013. http://pdg.lbl.gov.

[34] **CMS** Collaboration, "Search for new physics in events with same-sign dileptons and jets in pp collisions at 8 TeV," Tech. Rep. CMS-PAS-SUS-13-013, CERN, Geneva, 2013.

[35] **CMS** Collaboration, "Search for supersymmetry with the razor variables at CMS," Tech. Rep. CMS-PAS-SUS-12-005, CERN, Geneva, 2012.

[36] **CMS** Collaboration, "Search for supersymmetry with the razor variables at CMS," Tech. Rep. CMS-PAS-SUS-11-024, CERN, Geneva, 2012.

[37] **CMS** Collaboration, "Search for new physics in the multijet and missing transverse momentum final state in proton-proton collisions at $\sqrt{s} = 7$ TeV," *Phys. Rev. Lett.* **109** (Oct, 2012) 171803. `http://link.aps.org/doi/10.1103/PhysRevLett.109.171803`.

[38] **CMS** Collaboration, "Searches for Supersymmetry using Multilepton Signatures in pp Collisions at 7 TeV," Tech. Rep. CMS-PAS-SUS-12-013, CERN, Geneva, 2011.

[39] **CMS** Collaboration, S. Chatrchyan *et al.*, "Search for supersymmetry in final states with missing transverse energy and 0, 1, 2, or at least 3 b-quark jets in 7 TeV pp collisions using the variable alphaT," *JHEP* **1301** (2013) 077, `arXiv:1210.8115 [hep-ex]`.

[40] **CMS** Collaboration, S. Chatrchyan *et al.*, "Search for electroweak production of charginos and neutralinos using leptonic final states in *pp* collisions at $\sqrt{s} = 7$ TeV," *JHEP* **1211** (2012) 147, `arXiv:1209.6620 [hep-ex]`.

[41] **CMS** Collaboration, "Search for direct EWK production of SUSY particles in multilepton modes with 8TeV data," Tech. Rep. CMS-PAS-SUS-12-022, CERN, Geneva, 2013.

[42] **CMS** Collaboration, "A Search for Anomalous Production of Events with three or more leptons using 9.2 fb^{-1} of $\sqrt{s} = 8$ TeV CMS data," Tech. Rep. CMS-PAS-SUS-12-026, CERN, Geneva, 2013.

[43] **CMS** Collaboration, "Search for supersymmetry in pp collisions at sqrt(s) = 8 TeV in events with three leptons and at least one b-tagged jet," Tech. Rep. CMS-PAS-SUS-13-008, CERN, Geneva, 2013.

[44] **CMS** Collaboration, "Search for supersymmetry using razor variables in events with b-jets in pp collisions at 8 TeV," Tech. Rep. CMS-PAS-SUS-13-004, CERN, Geneva, 2013.

[45] **CMS** Collaboration, S. Chatrchyan *et al.*, "Search for gluino mediated bottom- and top-squark production in multijet final states in pp collisions at 8 TeV," *Phys.Lett.* **B725** (2013) 243–270, `arXiv:1305.2390 [hep-ex]`.

[46] **CMS** Collaboration, "Search for electroweak production of charginos, neutralinos, and sleptons using leptonic final states in pp collisions at 8 TeV," Tech. Rep. CMS-PAS-SUS-13-006, CERN, Geneva, 2013.

[47] **CMS** Collaboration, "Search for supersymmetry in pp collisions at sqrt(s) = 8 Tev in events with two opposite sign leptons, large number of jets, b-tagged jets, and large missing transverse energy.," Tech. Rep. CMS-PAS-SUS-13-016, CERN, Geneva, 2013.

[48] **CMS** Collaboration, "Search for electroweak production of charginos and neutralinos in final states with a Higgs boson in pp collisions at 8 TeV," Tech. Rep. CMS-PAS-SUS-13-017, CERN, Geneva, 2013.

[49] **CMS** Collaboration, S. Chatrchyan *et al.*, "Search for supersymmetry in hadronic final states with missing transverse energy using the variables AlphaT and b-quark multiplicity in pp collisions at 8 TeV," *Eur.Phys.J.* **C73** (2013) 2568, `arXiv:1303.2985 [hep-ex]`.

[50] **CMS** Collaboration, S. Chatrchyan *et al.*, "Search for top-squark pair production in the single-lepton final state in pp collisions at $\sqrt{s} = 8$ TeV," `arXiv:1308.1586 [hep-ex]`.

[51] **CMS** Collaboration, "Search for New Physics in the Multijets and Missing Momentum Final State in Proton-Proton Collisions at 8 TeV," Tech. Rep. CMS-PAS-SUS-13-012, CERN, Geneva, 2013.

[52] **CMS** Collaboration, "A search for anomalous production of events with three or more leptons using 19.5/fb of sqrt(s)=8 TeV LHC data," Tech. Rep. CMS-PAS-SUS-13-002, CERN, Geneva, 2013.

[53] **CMS** Collaboration, "Search for Supersymmetry in pp collisions at 8 TeV in events with a single lepton, multiple jets and b-tags," Tech. Rep. CMS-PAS-SUS-13-007, CERN, Geneva, 2013.

[54] **ATLAS** Collaboration, "Search for direct production of charginos and neutralinos in events with three leptons and missing transverse momentum in $21\,\text{fb}^{-1}$ of pp collisions at $\sqrt{s} = 8\,\text{TeV}$ with the ATLAS detector," Tech. Rep. ATLAS-CONF-2013-035, CERN, Geneva, Mar, 2013.

[55] **ATLAS** Collaboration, "Search for supersymmetry in events with four or more leptons in $21\,\text{fb}^{-1}$ of pp collisions at $\sqrt{s} = 8\,\text{TeV}$ with the ATLAS detector," Tech. Rep. ATLAS-CONF-2013-036, CERN, Geneva, Mar, 2013.

[56] **ATLAS** Collaboration, “Search for direct top squark pair production in final states with one isolated lepton, jets, and missing transverse momentum in $sqrts = 8\,\text{TeV}$ *pp* collisions using 21 fb^{-1} of ATLAS data,” Tech. Rep. ATLAS-CONF-2013-037, CERN, Geneva, Mar, 2013.

[57] **ATLAS** Collaboration, “Search for direct top squark pair production in final states with one isolated lepton, jets, and missing transverse momentum in sqrt(s) = 8 TeV pp collisions using 13.0 ifb of ATLAS data,” Tech. Rep. ATLAS-CONF-2012-166, CERN, Geneva, Dec, 2012.

[58] **ATLAS** Collaboration, “Search for direct third generation squark pair production in final states with missing transverse momentum and two *b*-jets in $\sqrt{s} = 8$ TeV *pp* collisions with the ATLAS detector.,” Tech. Rep. ATLAS-CONF-2013-053, CERN, Geneva, May, 2013.

[59] **ATLAS** Collaboration, “Search for direct stop production in events with missing transverse momentum and two b-jets using 12.8 fb^{-1} of pp collisions at $\sqrt{s} = 8$ TeV with the ATLAS detector,” Tech. Rep. ATLAS-CONF-2013-001, CERN, Geneva, Jan, 2013.

[60] **ATLAS** Collaboration, “Search for direct stop pair production in events with a Z boson, b-jets and missing transverse energy with the ATLAS detector using 21 fb^{-1} from proton-proton collision at $\sqrt{s} = 8$ TeV,” Tech. Rep. ATLAS-CONF-2013-025, CERN, Geneva, Mar, 2013.

[61] **ATLAS** Collaboration, “Search for strongly produced superpartners in final states with two same sign leptons with the ATLAS detector using 21 fb-1 of proton-proton collisions at sqrt(s)=8 TeV.,” Tech. Rep. ATLAS-CONF-2013-007, CERN, Geneva, Mar, 2013.

[62] **ATLAS** Collaboration, “Search for electroweak production of supersymmetric particles in final states with at least two hadronically decaying taus and missing transverse momentum with the ATLAS detector in proton-proton collisions at $\sqrt{s} = 8$ TeV,” Tech. Rep. ATLAS-CONF-2013-028, CERN, Geneva, Mar, 2013.

[63] **ATLAS** Collaboration, G. Aad *et al.*, “Search for new phenomena in final states with large jet multiplicities and missing transverse momentum at $\sqrt{s} = 8$ TeV proton-proton collisions using the ATLAS experiment,” *JHEP* **1310** (2013) 130, `arXiv:1308.1841 [hep-ex]`.

[64] **ATLAS** Collaboration, G. Aad *et al.*, "Search for direct third-generation squark pair production in final states with missing transverse momentum and two b-jets in $\sqrt{s}$=8 TeV pp collisions with the ATLAS detector," `arXiv:1308.2631 [hep-ex]`.

[65] **ATLAS** Collaboration, "Search for direct production of the top squark in the all-hadronic ttbar + etmiss final state in 21 fb-1 of p-pcollisions at sqrt(s)=8 TeV with the ATLAS detector," Tech. Rep. ATLAS-CONF-2013-024, CERN, Geneva, Mar, 2013.

[66] **ATLAS** Collaboration, "Search for Supersymmetry in final states with two same-sign leptons, jets and missing transverse momentum with the ATLAS detector in pp collisions at sqrt(s)=8 TeV," Tech. Rep. ATLAS-CONF-2012-105, CERN, Geneva, Aug, 2012.

[67] **ATLAS** Collaboration, "Search for direct top squark pair production in final states with two leptons in $\sqrt{s}$ = 8 TeV pp collisions using 20fb^{-1} of ATLAS data.," Tech. Rep. ATLAS-CONF-2013-048, CERN, Geneva, May, 2013.

[68] **ATLAS** Collaboration, "Search for squarks and gluinos in events with isolated leptons, jets and missing transverse momentum at $\sqrt{s}$ = 8 TeV with the ATLAS detector," Tech. Rep. ATLAS-CONF-2013-062, CERN, Geneva, Jun, 2013.

[69] **ATLAS** Collaboration, "Search for strongly produced supersymmetric particles in decays with two leptons at $\sqrt{s}$ = 8 TeV," Tech. Rep. ATLAS-CONF-2013-089, CERN, Geneva, Aug, 2013.

[70] **ATLAS** Collaboration, "Search for direct-slepton and direct-chargino production in final states with two opposite-sign leptons, missing transverse momentum and no jets in 20/fb of pp collisions at sqrt(s) = 8 TeV with the ATLAS detector," Tech. Rep. ATLAS-CONF-2013-049, CERN, Geneva, May, 2013.

[71] **ATLAS** Collaboration, "Search for squarks and gluinos with the ATLAS detector in final states with jets and missing transverse momentum and 20.3 fb^{-1} of $\sqrt{s}$ = 8 TeV proton-proton collision data," Tech. Rep. ATLAS-CONF-2013-047, CERN, Geneva, May, 2013.

[72] **ATLAS** Collaboration, "Search for strong production of supersymmetric particles in final states with missing transverse momentum and at least three

b-jets using 20.1 fb^{-1} of pp collisions at sqrt(s) = 8 TeV with the ATLAS Detector.," Tech. Rep. ATLAS-CONF-2013-061, CERN, Geneva, Jun, 2013.

[73] **ATLAS** Collaboration, "Searches for direct scalar top pair production in final states with two leptons using the stransverse mass variable and a multivariate analysis technique in $\sqrt{s} = 8$ TeV pp collisions using 20.3 fb^{-1} of ATLAS data," Tech. Rep. ATLAS-CONF-2013-065, CERN, Geneva, Jul, 2013.

[74] **ATLAS** Collaboration, "Search for chargino and neutralino production in final states with one lepton, two b-jets consistent with a Higgs boson, and missing transverse momentum with the ATLAS detector in 20.3 fb^{-1} of $\sqrt{s} =$ 8 TeV *pp* collisions," Tech. Rep. ATLAS-CONF-2013-093, CERN, Geneva, Aug, 2013.

[75] U. Laa, "Interpretation of CMS and ATLAS Simplified Models Results," Master's thesis, Universität Wien, 2014. in progress.

[76] "LEP SUSY Working Group," Dec, 2013. `http://lepsusy.web.cern.ch/lepsusy`.

[77] **CMS** Collaboration, "CMS Supersymmetry Physics Results," Oct, 2013. `https://twiki.cern.ch/twiki/bin/view/CMSPublic/PhysicsResultsSUS`.

[78] P. Bechtle, K. Desch, and P. Wienemann, "Fittino, a program for determining MSSM parameters from collider observables using an iterative method," *Comput.Phys.Commun.* **174** (2006) 47–70, `arXiv:hep-ph/0412012 [hep-ph]`.