Fakultät
für Physik und Astronomie

Master thesis

Charge and color breaking minima in models with extended Higgs sectors

November 2018

Author: Simon Geisler

Supervisor: Prof. Dr. W. Porod

I Abstract

The aim of this thesis is to investigate the stability of the **E**lectro**W**eak **S**ymmetry **B**reaking (**EWSB**) vacuum in models with extended Higgs sectors in order to constrain parameter spaces.

For this, `VevaciousPlusPlus` [1] is used to determine whether a parameter set is physical or not. An extension of `VevaciousPlusPlus` with a new homotopy continuation program `PHCpack` [2] is shown and the performance tests conclude, that it is a better option than the previous solver `HOM4PS2` [3]. With a C++ program, `VPPparamScanner` [4], to conveniently scan parameter ranges using the programs `SPheno` [5] and `VevaciousPlusPlus`, two different models are studied.

At first, a real scalar triplet extension of the Standard Model is investigated. For this model, neutral symmetry breaking and then charge breaking are both analyzed. The model brings some numerical issues of `VevaciousPlusPlus` into light, which need to be treated carefully in the future. The model fulfills the derived vacuum tree-level expectations well even at 1-loop level and shows, that further investigations could broaden the interesting parameter space for baryogenesis.

Finally, the Constrained Minimal Supersymmetric Standard Model is tested for **C**harge and **C**olor **B**reaking (**CCB**) minima. Two global fit parameter sets, which have been done by different collaborations and teams, are checked. It showed that there exists one global fit point from GAMBIT [6] "Stop co-annihilation", where the EWSB vacuum doesn't survive long enough towards a CCB minimum. All the other global fit parameter configurations turn out to be long-lived.

Kurzfassung

Das Ziel dieser Arbeit ist die Stabilitätsuntersuchung des Vakuums vom elektroschwachen Symmetriebruch für verschiedene Modelle mit erweiterten Higgs Sektoren, um dessen Parameterräume einzuschränken.

Dazu wird `VevaciousPlusPlus`[1] benutzt, um herauszufinden, ob eine Parameterkonfiguration physikalisch ist oder nicht. Ein neues Programm `PHCpack` [2] zur Berechnung von Minima über Homotopie Fortsetzung wurde in `VevaciousPlusPlus` integriert und es stellte sich als bessere Alternative zur vorherigen Option `HOM4PS2` [3] heraus. Mit einem C++ Programm, `VPPparamScanner`[4], welches Parameter überprüft und die Outputs von `SPheno` [5] und `VevaciousPlusPlus` verknüpft und auswertet, werden zwei verschiedene Modelle untersucht.

Zuerst wird ein Modell, bei dem das Standard Modell mit einem zusätzlichen reellen Triplet erweitert wird, betrachtet. Erst wird neutraler und danach geladener Symmetriebruch untersucht. Es zeigen sich vereinzelt numerische Probleme, die in zukünftigen Betrachtungen mit Sorgfalt behandelt werden müssen. Die tree-level Erwartungen für das Modell werden auf 1-loop Niveau immer noch gut erfüllt und es zeigt sich, dass der Parameterbereich des Modells, der interessant hinsichtlich des Baryogenese Problems ist, eventuell noch verbreitert werden könnte.

Danach wird noch das Constrained Minimal Supersymmetric Standard Model auf **C**harge und **C**olor **B**reaking (**CCB**) Minima untersucht. Bei der Betrachtung verschiedener Parameterkonfigurationen aus Global Fits stellt sich heraus, dass der Best Fit "Stop coannihilation" von GAMBIT [6] nicht stabil in Bezug auf ein tieferes CCB Minimum ist, und somit ausgeschlossen werden kann. Alle anderen überprüften Konfigurationen stellen sich als langlebig heraus.

II List of contents

III Listing index

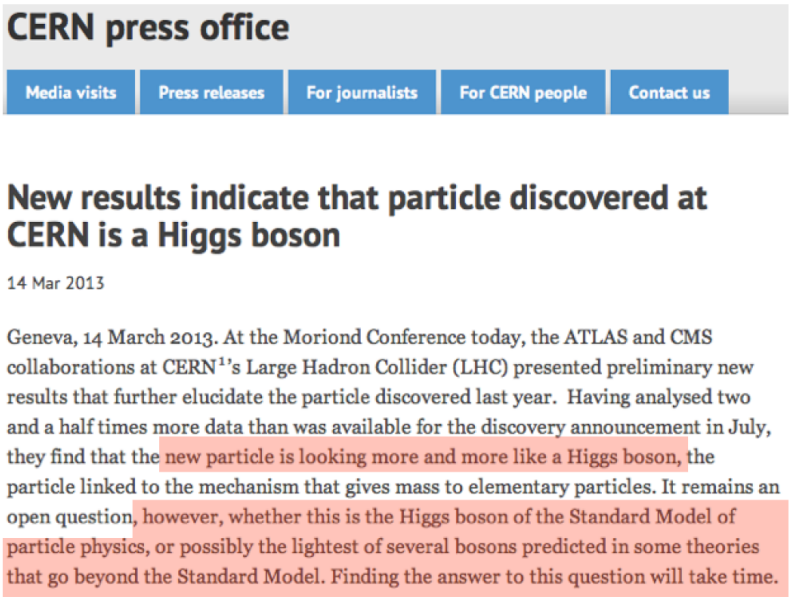
1 Introduction

Over a century of theoretical acquisitions, accompanied by technological and experimental progress, has led us to the current, up to high energies successful and sophisticated formulation of the fundamental theory of electroweak and strong interactions between the known elementary particles - the **S**tandard **M**odel (**SM**).

At first, the construction of a fully gauged theory brought problems with it, since experiments indicated the existence of massive gauge bosons, while the theoretical formulation doesn't allow naive mass terms for the vector bosons to show up in the Lagrangian due to the underlying symmetries.

1.1 The Higgs boson

In 1964, P.W. Higgs in his famous paper [7], and F. Englert with R. Brout in [8] independently showed that scalar fields, now named the Higgs fields, can yield mass terms by acquiring a non-vanishing vacuum expectation value and coupling with the gauge bosons. At first, there were no signs of such a particle until, almost 50 years of experimental studies in particle colliders later, finally a particle with the right properties to be a Higgs boson showed up in the Large Hadron Collider (LHC) in CERN.


CERN press office

Media visits | Press releases | For journalists | For CERN people | Contact us

New results indicate that particle discovered at CERN is a Higgs boson

14 Mar 2013

Geneva, 14 March 2013. At the Moriond Conference today, the ATLAS and CMS collaborations at CERN[1]'s Large Hadron Collider (LHC) presented preliminary new results that further elucidate the particle discovered last year. Having analysed two and a half times more data than was available for the discovery announcement in July, they find that the new particle is looking more and more like a Higgs boson, the particle linked to the mechanism that gives mass to elementary particles. It remains an open question, however, whether this is the Higgs boson of the Standard Model of particle physics, or possibly the lightest of several bosons predicted in some theories that go beyond the Standard Model. Finding the answer to this question will take time.

Fig. 1: Press article by the CERN press office in 2013, announcing that the observed resonance strongly points towards the Higgs boson. They also explain, that the possibility of extensions persists to exist, which points towards Beyond Standard Model theories. Credits [9].

The discovery of the Higgs boson led the mainstream media to exaggerate the Higgs boson as the "god particle", the mysterious object that gives rise to masses. This is

a vague explanation, since it solely produces masses for **elementary** particles and not for composite objects like nuclei, where most of the mass results from the interactions between the elementary particles.

While these findings at this time revealed the last remaining missing puzzle piece for the Standard Model, it isn't yet clear whether this particle is just the lightest one of plenty other scalar particles which can emerge in extended sectors, see Fig. 1. Beyond Standard Model theories are attractive to physicists, since they can answer questions which remain unanswered in the context of the Standard Model.

But of course, when formulating such new theories which extend the spectrum, new parameters, i.e. degrees of freedom, arise. These parameters are quasi arbitrary - they are constrained by symmetries, vacuum stability, renormalizability (if it's desired) and perturbativity, but nonetheless these new theories come with a huge volume of possible parameter configurations.

1.2 Motivation

It's welcoming to experimentalists when they know where to search for new physics and because of that it's important to constrain a new theory to the smallest parameter space possible. With that in mind, one can exclude theories, or their parameter spaces, from the huge pool of possibilities by comparing them with the observables in the measurements, for example conducted by particle accelerators. If one can for instance constrain the mass of a particle to a small range, experimentalists only need to scan in these regions to search for it.

There are many ways to constrain parameter spaces, and one of them is to investigate the vacuum stability of the scalar sector. Until today the universe remains in the state after **E**lectro**W**eak **S**ymmetry **B**reaking (**EWSB**) and didn't yet transition into some lower ground state, where for example charge or color symmetry is broken. This allows to constrain the parameter space, since certain deeper minimum configurations would shape the current ground state as unstable or metastable. Of course, if the EWSB vacuum is metastable for a certain parameter configuration, the lifetime of the state which would transition to the deeper configuration needs to exceed the lifetime of the universe to be reliable.

To understand this, the knowledge of the scalar spectrum and the structure of the potential is crucial, because this determines the particle mixings, the mass terms and also the stability of a given ground state. Unfortunately, the potential gets extremely complicated really fast, and for many degrees of freedom, even just on tree-level, the stability relations

can't be deduced analytically. Strong and computing-intensive numerical procedures are required to find minimum configurations, and when deeper states of the scalar fields are found, the tunneling time needs to be calculated too, like illustrated in Fig. 2.

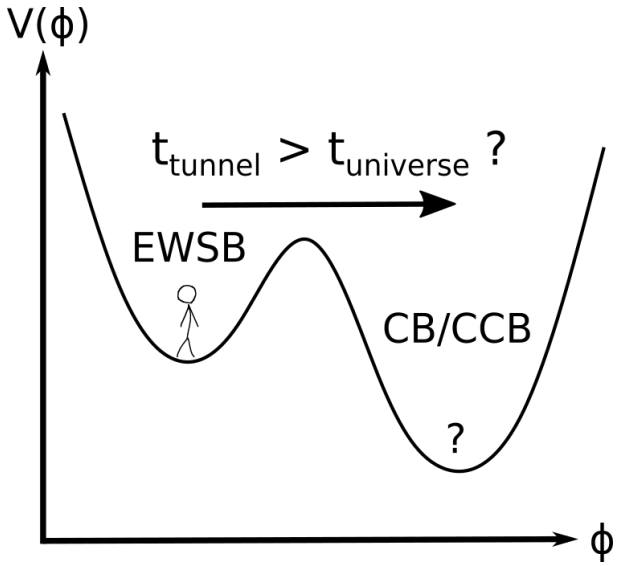


Fig. 2: Illustrative depiction of a potential function with a specific parameter configuration, which contains a deeper charge breaking (CB) or charge and color breaking (CCB) minimum than the state our universe is right now (EWSB). If such a state exists, in order to be a valid parameter configuration, the tunneling time needs to be larger than the lifetime of the universe.

In this thesis, using a program called `VevaciousPlusPlus`, the stability of the electroweak symmetry breaking vacuum will be investigated for different models. `VevaciousPlusPlus`, which combines several programs and libraries and uses their results, finds the global minimum at 1-loop level of a given scalar potential and calculates the tunneling time to this point. These calculations will be useful to constrain the parameter spaces which are valid in the sense of vacuum stability in the zero temperature low energy limit.

2 Theoretical framework

Before stepping into the developments and vacuum stability analyses, a small theoretical preliminary is presented in order to get an overview of the background knowledge of this thesis.

2.1 Spontaneous symmetry breaking

A main ingredient in gauge theories is the implementation of local spontaneous symmetry breaking, i.e. the Higgs mechanism, which provides masses for gauge bosons and fermions. The Higgs mechanism is necessary to construct high energy unification models, since in the low energy limit, they need to match the experimental observations. This means the presumed higher symmetries of the model need to break down to the Standard Model for low energies. The following sections are a short summary of the content given in [10].

2.1.1 Goldstone theorem

Before presenting the Higgs mechanism it's beneficial to examine the goldstone theorem first. The theorem describes the consequences of breaking continuous **global** symmetries in the ground state of bosonic scalar particles. The consideration of for example vectorial particles breaking a global symmetry would further break Lorentz invariance and due to a lack of experimental observation is forbidden.

In principle, the goldstone theorem states, that each broken symmetry, which is caused by the corresponding particles having a non-vanishing **V**acuum **E**xpectation **V**alue (**VEV**), excites massless modes: the Goldstone bosons.

To understand this, let's assume "a multiplet ϕ of scalar fields, transforming under a real orthogonal representation R of a global, i.e. not gauged, symmetry group G" [10] with a corresponding Lagrangian

$$\mathcal{L} = \frac{1}{2}\partial_\mu \phi^T \partial^\mu \phi - \frac{\lambda}{4}(\phi^T\phi - v^2)^2, \quad \lambda > 0. \tag{1}$$

Finding the equations of motion by varying the Lagrangian with respect to ϕ

$$\frac{\delta \mathcal{L}}{\delta \phi} = (\Box - \lambda v^2)\phi + \lambda(\phi^T\phi)\phi = 0 \tag{2}$$

and comparing this result with the operator $(\Box + m^2)$ of the Klein-Gordon equation concludes a tachyonic, i.e. unstable, ground state for $\phi = 0$ with $m^2 = -\lambda v^2 < 0$ and $v \neq 0$.

To solve this problem, one has to expand

$$\phi(x) = \phi_0 + \chi(x)$$

around the potential minimum $\phi_0^T \phi_0 = v^2$, and define $\chi(x)$ as the dynamical field. Inserting this into Eq. 1 leads to

$$\mathcal{L} = \frac{1}{2}\partial_\mu \chi^T \partial^\mu \chi - \frac{1}{2}\chi^T M \chi - \lambda \phi_0^T \chi \chi^T \chi - \frac{\lambda}{4}\left(\chi^T \chi\right)^2, \tag{3}$$

with a mass matrix $M = 2\lambda\phi_0 \otimes \phi_0^T$ obeying $M\phi_0 = 2\lambda v^2 \phi_0$. This result means, that there's "one mode with mass m $= 2\lambda v^2$ and all other modes massless" [10].

Let's define the subgroup $H \subset G$ with the group elements $h \in H$, which leave ϕ_0 invariant, i.e. $h\phi_0 = \phi_0$. In the sense of Lie groups, the generators of H, namely $\{\mathrm{T}_a\}_{a=1,\ldots,\dim(\mathrm{H})}$, correspond to residual symmetries of the ground state since the symmetry transformations with the generators T_a leave the ground state ϕ_0 invariant.

"The remaining generators $\{\mathrm{X}_i\}_{i=1,\ldots,\dim(\mathrm{G})-\dim(\mathrm{H})}$ generate the coset G/H" [10] and are related to the broken symmetries, which yield the massless Goldstone bosons, see Fig. 3. In the Higgs mechanism, the generators of X_i together with ϕ_0 will couple to the gauge bosons (or fermions) and, in a particular gauge, produce mass terms in the Lagrangian.

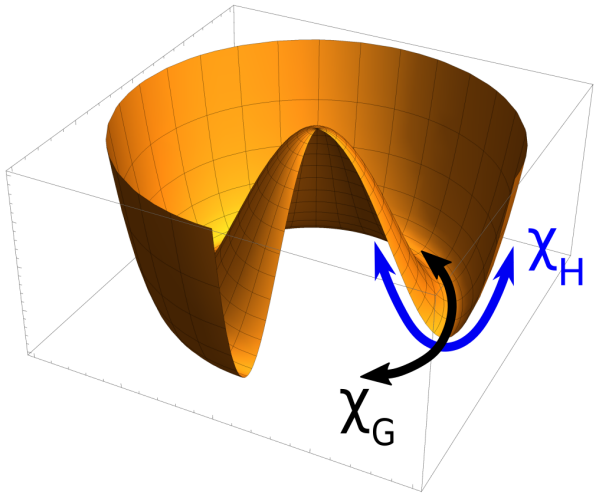


Fig. 3: Illustrative depiction as a Mathematica plot of the so called "mexican hat" potential shape. The remaining symmetries in the new ground state yields massless goldstone bosons χ_G. The curvature of the potential along these field configurations is zero, i.e. they don't have a mass. The remaining massive scalar bosons χ_H are the excitations along nonzero curvature directions. They will later be called the Higgs bosons.

2.1.2 Higgs mechanism

Subsequent to global symmetry breaking, let's study the effects on breaking gauged theories. Following the same conventions as before, consider a Lagrangian

$$\mathcal{L} = -\frac{1}{2} tr\left(F^{\mu\nu}F_{\mu\nu}\right) + \frac{1}{2}\left(D_\mu\phi\right)^\dagger D^\mu\phi - \frac{\lambda}{4}(\phi^T\phi - v^2)^2 \tag{4}$$

with the covariant derivative $D_\mu = \partial_\mu - i\,g\,A_\mu(x)$ and a gauge field $A_\mu(x)$. As we know from section 2.1 the residual symmetries after symmetry breaking don't change the ground state ϕ_0, which means, we can reach every other configuration

$$\phi(x) = e^{i\chi(x)/v}\phi_0, \quad \chi(x) = \sum_{i=1}^{\dim(G)-\dim(H)} \chi_i(x) X_i \tag{5}$$

with just the generators X_i corresponding to the broken symmetries (this holds while assuming that we can get every value by group transformations in G).

Now, for demonstration purposes, let's apply the unitarity gauge, in which we can eliminate the dynamical field $\chi(x)$ in the Lagrangian by the transformation

$$\begin{bmatrix} \phi(x) \\ D_\mu\phi(x) \\ A_\mu(x) \\ F_{\mu\nu}(x) \end{bmatrix} \rightarrow \begin{bmatrix} e^{-i\chi(x)/v}\phi(x) \\ e^{-i\chi(x)/v}D_\mu\phi(x) \\ e^{-i\chi(x)/v}A_\mu(x)e^{i\chi(x)/v} + \frac{i}{g}e^{-i\chi(x)/v}\left(\partial_\mu e^{i\chi(x)/v}\right) \\ e^{-i\chi(x)/v}F_{\mu\nu}(x)e^{i\chi(x)/v} \end{bmatrix}. \tag{6}$$

Again, inserting this into the Lagrangian given in equation 4, leads us to

$$\mathcal{L}_{ug} = -\frac{1}{2} tr\left(F^{\mu\nu}F_{\mu\nu}\right) + \frac{g^2}{2}\left(A_\mu\phi_0\right)^\dagger\left(A^\mu\phi_0\right) \tag{7}$$

which now has a mass term $M^2_{i,j}\left(A^i_\mu\right)^\dagger A^{\mu,j}$ and initially was not allowed by the restriction of gauge invariance. The mass matrix after representing the gauge fields in terms of the generators reads $M^2_{ij} = \frac{g^2}{2}\,\phi_0^\dagger\left(X_iX_j + X_jX_i\right)\phi_0$.

Concluding this procedure, we coupled gauge bosons to the scalar field multiplet ϕ, expanded around a stable ground state ϕ_0 and gauged away the dynamical part of $\phi(x)$. By doing this we were able to produce mass terms for the initially massless gauge bosons A_μ. All the generators X_i which change ϕ_0 finally correspond to massive gauge bosons.

2.2 Higgs sectors

The form of the potential and the exact procedure of the Higgs mechanism highly depends on the chosen model. There may be different multiplets of Higgs particles, either complex or real valued, with different quantum numbers which couple to different fields.

2.2.1 Standard Model

In 2012, a scalar particle with $M_H \approx 125$ GeV, obtaining a VEV $\approx 246GeV$ [11], was announced by experiments at the **L**arge **H**adron **C**ollider (**LHC**), and identified as the Higgs boson some time later in 2013, as already mentioned in the introduction section. With this detection, the last missing parameter of the SM was measured. Until now, the **S**tandard **M**odel (**SM**) remains to be an excellent theory to describe elementary particle physics at the current measured energy scales (see [9]).

The following is mainly based on [10] and can be reviewed for further information on this topic. The Standard Model gauge group G = $SU(3)_c \times SU(2)_L \times U(1)_Y$ breaks down to

$$G/H = (SU(3)_c \times SU(2)_L \times U(1)_Y) \,/\, (SU(3)_c \times U(1)_Q)\,. \tag{8}$$

This is called the **E**lectro**W**eak **S**ymmetry **B**reaking (**EWSB**) and can be mediated by an isospin Higgs doublet $\phi \in (\mathbf{1}_C, \mathbf{2}_T)_{Y=1}$, i.e. a singlet under SU(3) color, a doublet under SU(2) isospin and with hypercharge Y=1 (this is the minimal way to fulfill the desired symmetry breaking).

$$\phi = \begin{bmatrix} \phi^+ \\ \phi^0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} \phi_3 + i\phi_4 \\ \phi_1 + i\phi_2 \end{bmatrix} \tag{9}$$

The complex fields ϕ^+ and ϕ^0 can be parametrized by respectively two real degrees of freedom ϕ_3, ϕ_4 and ϕ_1, ϕ_2. After EWSB the charge reads $Q = T_3 + \frac{Y}{2}$, with T_3 as the third component of the weak isospin vector.

Alongside the kinetic part $|D_\mu \phi|^2$ for the Higgs bosons, let's demand a potential of the form

$$V(\phi) = \frac{\lambda}{2}\left(\phi^\dagger\phi - \frac{v^2}{2}\right)^2 \quad \text{with the minima} \quad \phi_0^\dagger\phi_0 = \frac{v^2}{2}. \tag{10}$$

We can now choose $\phi_0 = \frac{1}{\sqrt{2}}\begin{bmatrix} 0 \\ v \end{bmatrix}$ and notice that the charge Q isn't broken with this setup. Going through the Higgs mechanism procedure, we again expand ϕ around ϕ_0 with $\phi = \phi_0 + \begin{bmatrix} \chi^+ \\ \chi^0 \end{bmatrix}$ and use the gauge symmetries to rotate to $\phi = \frac{1}{\sqrt{2}} e^{i\vec{\alpha}\vec{T}} \begin{bmatrix} 0 \\ v+h \end{bmatrix}$, with

the basis vector $\vec{T}$ of the underlying Lie algebra.

As we did before, we can now use unitarity gauge $\phi \to e^{-i\vec{\alpha}\vec{T}}\phi$ to absorb the exponential factor. Now we have all the necessary ingredients and can insert them into the scalar part of the Lagrangian

$$\mathcal{L} = |D_\mu \phi|^2 - \frac{\lambda}{2}\left(\phi^\dagger\phi - \frac{v^2}{2}\right)^2 \tag{11}$$

while the covariant derivative in the representation of the Higgs doublet reads

$$D_\mu = \partial_\mu + ig\vec{W}_\mu\vec{T} + ig'B_\mu\frac{Y}{2}.$$

g is the SU(2) coupling, g' the U(1) coupling, $\vec{W_\mu}$ and B_μ are the gauge bosons.

After some calculations one can read off the masses or the mass matrices of the gauge bosons from the expansion of the kinetic terms of ϕ:

$$M^2_{W^\pm} = \frac{g^2v^2}{4}, \quad M^2_{W^3,B} = \frac{v^2}{4}\begin{pmatrix} g^2 & -gg' \\ -gg' & (g')^2 \end{pmatrix}. \tag{12}$$

$M^2_{W^3,B}$ has one nonzero eigenvalue $M^2_Z = \frac{v^2}{4}(g^2 + (g')^2)$, corresponding to the Z-Boson. The physical states consisting of the Z-Boson and the photon can be reached by a rotation with the Weinberg angle θ_w

$$\begin{bmatrix} Z_\mu \\ A_\mu \end{bmatrix} = \begin{pmatrix} cos(\theta_w) & -sin(\theta_w) \\ sin(\theta_w) & cos(\theta_w) \end{pmatrix}\begin{bmatrix} W^3_\mu \\ B_\mu \end{bmatrix}. \tag{13}$$

The Higgs mass, analogously derived like explained in section 2.1.2, is $m_H = \sqrt{\lambda}\cdot v$.

For now, we managed to generate mass terms for the gauge bosons. In the fermionic particle sector, due to the condition of gauge invariance, we also struggle to simply construct mass terms in the form $m(\overline{\Psi}_R\Psi_L + \overline{\Psi}_L\Psi_R)$.

In the same manner as with gauge bosons, the coupling of Higgs bosons to fermions, called Yukawa couplings, can yield the wanted mass terms. Due to the form $\overline{\Psi_R}\Psi_L \in (1,2)_{Y_L - Y_R}$, we need another Higgs doublet as we have $Y = \pm 1$ in the representations of the observed fermionic particle sector. The group structure of the Standard Model allows us to reach the representation $\tilde{\phi} \in (1,2)_{-1}$ by complex conjugating the Higgs boson from before as

$$\tilde{\phi} = i\sigma^2\overline{\phi} = \begin{pmatrix} \overline{\phi^0} \\ -\phi^- \end{pmatrix} \in (1,2)_{-1}. \tag{14}$$

Here, we won't discuss the exact form of the Yukawa couplings in detail, but the general form of Yukawa couplings are $\propto Y\overline{\Psi}\phi\Psi$ for the scalar, and $\propto Y\overline{\Psi}i\gamma^5\phi\Psi$ for the pseudoscalar part. Y are the Yukawa matrices which are not necessary diagonal. After applying the Higgs mechanism, the diagonalization of the mass matrices is not trivial due to the non-normality of the matrices.

This issue can be resolved but isn't reviewed here in detail. Briefly speaking, we have generation mixing in the quark sector (in the Standard Model there's no mixing in the leptonic sector) which is characterized by the **C**abibbo **K**obayashi **M**askawa matrix V_{CKM}.

As promising as the Standard Model may look, there remain several unanswered questions. There's for example no particle which has the properties to be a candidate for dark matter, which is to date an unsolved mystery in astrophysics. Baryogenesis, the mechanism to explain the matter-antimatter distribution in the universe, can't be explained by the CP violation in the SM alone. Also, the SM lacks naturalness, which means that the parameters need to be fine-tuned to match the experiments. This, and some unmentioned other open questions, motivates physicists to think outside the box and construct theories beyond of the Standard Model (see [9]).

2.2.2 Beyond Standard Model

With for example the discovery of neutrino oscillations, which leads to the conclusion that neutrinos need to have a non-vanishing mass (opposed to the SM prediction $m_\nu = 0$), hints of **B**eyond **S**tandard **M**odel (**BSM**) physics showed its first signs in experiments [12]. Many new experiments are planned and currently being conducted in order to measure observables in high precision to find clues of new physics and of course to rule out existing theories.

An important point when constructing extended theories is, that it needs to yield the physics of the SM at low energies, which means the higher assumed symmetries need to break down to $SU(3)_c \times SU(2)_L \times U(1)$ with the elaborated EWSB at the correct energy scale and associated scalar particles obtaining VEVs to explain our measured mass spectrum.

Furthermore severe constraints by the absence of **F**lavour **C**hanging **N**eutral **C**urrents (**FCNC**) or the $\rho = \frac{M_W^2}{M_Z^2 cos^2(\theta_W)}$ parameter, which measures "the ratio of the neutral current to charged current strength in the effective low-energy Lagrangian" [13], are imposed as constraints to the construction of extended Higgs sectors.

The ρ parameter for example reads [9]

$$\rho = \frac{\sum_i^n \left[I_i \left(I_i + 1 \right) - \frac{1}{4} Y_i^2 \right] v_i}{\sum_i^n \frac{1}{2} Y_i^2 v_i} \tag{15}$$

for a theory with n scalar multiplets and hypercharge Y_i, isospin I_i and corresponding VEV v_i. The constraint arising from the measured ρ parameter, which is very close to 1, $\rho = 1.00040 \pm 0.00024$ [11], enforces strong constraints on the possibilities on extensions of the quantum numbers and VEVs of the Higgs sector (see [9]).

Regarding the FCNC constraints, if there's more than one Higgs doublet, the tree level FCNC's could be sufficiently suppressed, if only one Higgs doublet couples to fermions, this is called the Glashow-Weinberg criterion. For the two Higgs doublet model "it is possible to have tree level FCNC completely fixed by the CKM matrix, as a result of an abelian symmetry" [11].

As strict as the constraints are, by choosing correct hypercharges and VEVs, expanding the Higgs sector of the SM is still possible and viable, since it can also solve for example problems with the SM Higgs boson like the hierarchy problem. In short words this is the issue of the quadratic mass corrections when running to high scales since the mass parameter isn't protected through gauge or chirality symmetries. Although this can be solved by specific fine-tuning, naturalness gets lost. For example, **SU**per**SY**mmetry (**SUSY**) extensions are able to explain the hierarchy in a much more natural way [11].

Until to date, all the measurements probing the properties of the measured Higgs boson with $m_H \approx 125\,\text{GeV}$ and relating precision measurements point towards the EWSB of the SM which are consistent with its' predictions [11]. The search for additional scalar particles continues and physicists hope to find clues for Beyond Standard Model physics to further complete the Standard Model up to high scales and to explain puzzling questions arising with it.

Specific models and extensions of their Higgs sectors will be explained and investigated in detail later in this work when their vacuum structure is analyzed.

2.3 `SPheno`

`SPheno` (**S**upersymmetric **Pheno**menology) [5] [14] [15] is a particle spectrum calculator written in Fortran by W. Porod. It's suitable for calculating spectra for supersymmetric extensions within theories in high scales. Although its name "**S**upersymmetric **Pheno**menology" suggests supersymmetry, also non-supersymmetric models are allowed. The program uses low energy observables and a given model as input for the several cal-

culations it conducts. Many of the functionalities SPheno offers are not used and thus are being omitted in the discussion. In order to keep this section simple and short, all the calculations and routines being done by SPheno aren't depicted here - only the relevant aspects to understand the usage of the program in the context of this thesis are shown.

Let's assume all the model data of the picked model are already present in the Fortran source code. This means, the Lagrangian with all the associated information is available for SPheno and it can calculate the spectrum with loop-corrections out of it.

To do so, an input file, conventionally named "LesHouches.in", is needed. This file, and also the output file, follows the conventions of **S**USY **L**es **H**ouches **A**ccord (**SLHA**) [16], where the parameters and flags are ordered in blocks. In "Block SPhenoInput" options for the execution of SPheno are given and in "Block MINPAR" all the input parameters are listed. In every block, each parameter or flag has an integer identifier, which helps the program to connect the desired parameter with the correct value.

In the output file of SPheno the calculated values for the model relevant parameters and coupling strengths at a given renormalization scale Q are printed in different blocks, which are defined in the source code of the chosen model. These output files are run point inputs for Vevacious/VevaciousPlusPlus.

2.4 Vevacious

In this work, Vevacious, or more precisely a newer version called VevaciousPlusPlus [1] by Ben O'Leary, is used to find the global minima of the "one-loop effective potential energy function" [17] and calculate the tunneling time to it, starting from a given parameter run point. Solving the minimization conditions of the potential function gets extremely complicated for enhanced Higgs sectors really fast and numerical methods are needed. Also, the calculation of the tunneling time is highly nontrivial too. The following is a summary of the most relevant points in [17].

2.4.1 Principle of operation

Before starting any calculation, of course an input, characterizing the model and its parameters, is needed. The model file, generated by the Mathematica package called SARAH [18], contains all the information about the potential function with the desired symmetry breaking conditions, together with the mass-squared matrices for the loop corrections. Furthermore, VevaciousPlusPlus needs parameter points, which need to be in the SLHA format. In this work, the SLHA input files are generated as output by SPheno, as already mentioned in section 2.3. In some rare cases, but as much avoided as

possible, `FlexibleSUSY` [19] is used. `VevaciousPlusPlus` undergoes three different steps until coming to its result.

At first, using homotopy continuation, all the solutions to the tadpole equations in tree-level are found with `HOM4PS2` [3] (*or* `PHCpack` [2]) and parsed by `VevaciousPlusPlus`. These solutions are "used as starting points for gradient-based minimization of the one-loop effective potential" [17], calculated with `MINUIT` [20] (`PyMinuit` or `iminuit` [21]). If now a global minimum other than the input vacuum, i.e. the given starting run point in the potential, is found, the tunneling time can be optionally calculated using `CosmoTransitions` [22].

Finally `VevaciousPlusPlus` writes an output, with whether stating the input to be stable or metastable, if desired together with the tunneling time.

2.4.2 Potential energy function structure

`VevaciousPlusPlus` determines the 1-loop effective potential minima, which are in general different to the tree-level ones. The resulting shift of VEVs by rolling to 1-loop leads to a different structure of the potential, i.e. the extremum configurations, and thus impacts the tunneling times between vacua.

Assuming a renormalizable model with N real degrees of freedom, i.e. $\phi_i, \quad i = 1, ..., N$ real scalar fields, the tree-level potential in general reads

$$V_{tree} = \lambda_{ijkl}\phi_i\phi_j\phi_k\phi_l + A_{ijk}\phi_i\phi_j\phi_k + \mu_{ij}^2\phi_i\phi_j, \tag{16}$$

where linear terms have been shifted away by redefinitions and constant terms are neglected, because they have no physical impact to the equations of motion. λ_{ijkl}, A_{ijk} and μ_{ij} are coefficients corresponding to the different combinations of fields. There are obvious symmetries for the coefficients since f.e. swapping two fields ϕ_i and ϕ_j yields the same terms, since they are scalar bosons, ergo symmetric in permutations, but this isn't relevant for the considerations right now.

Searching for extrema of the potential V_{tree}, one needs to vary the potential with respect to the fields $\frac{\delta V_{tree}}{\delta \phi_i}$, resulting to N polynomials with polynomial degree 3, called the tadpole equations. Unfortunately, as already mentioned before, powerful numerical methods are needed to find *all* the solutions to these equations. Homotopy continuation ensures to find **all** solutions and is a fast numerical method to do so. `HOM4PS2` is used in the official version of `Vevacious`, but it lacks performance when investigating high amounts of fields.

It gets a little more complicated when regarding the one-loop effective potential:

$$V_{1-loop} = V_{tree} + V_{counter} + V_{mass}, \tag{17}$$

with V_{tree} from Eq. 16 and $V_{counter}$ as the counterterm part due to renormalization and

$$V_{mass} = \frac{1}{64\pi^2} \sum_n (-1)^{2s_n} (2s_n + 1)(\tilde{M}_n^2(\Phi))^2 \left[log(\tilde{M}_n^2(\Phi)/Q^2) - c_n \right]. \tag{18}$$

The sum in equation 18 "runs over all real scalar, Weyl fermion, and vector degrees of freedom, with s_n being the spin of the degree of freedom" [17] and Φ is a field configuration. $\tilde{M}_n^2(\Phi)$ denote the eigenvalues of the mass-matrices in the Lagrangian and Q is the renormalization scale. c_n is a number which depends on the chosen regularization scheme such as $\overline{MS}$ or $\overline{DR}'$.

2.4.3 Vacuum tunneling

If the field configuration only resembles a metastable state, or even unstable state, the fields which attain VEVs will tunnel into the lower vacuum configurations. Of course, at some point it will reach the global minimum. The calculation of these phase transitions is highly non-trivial and requires powerful methods to do so.

The decay rate per volume, which is derived and explained in the famous papers of S. Coleman [23] and in the second paper together with C. G. Callan, Jr. [24], reads

$$\Gamma/vol = A \cdot e^{(-B/\hbar)} \left(1 + \mathcal{O}(\hbar)\right). \tag{19}$$

The values of A, which is given by a determinant term [24], and B, which is the bounce action [23], depend on the model and are both not easy to determine. Due to the fact, that the bounce action is in the exponent, the precision on the bounce action is more important than that of the coefficient A, which is estimated most of the time (Vevacious estimates this to be equal to the renormalization scale to the power of 4: Q^4).

To get an estimated time of tunneling, `CosmoTransitions` [22] uses path deformations along the connecting line of the extrema on the hypersurface of the potential at finite or zero temperature to find the bounce action B.

2.5 SARAH

`SARAH`, a complete description is given in [18], is a Mathematica package by F. Staub which allows analyzations of primarily supersymmetric extensions of the SM but also allows non-supersymmetric cases. `SARAH` can calculate the tadpole equations, all vertices

and mass matrices. Also, the 1-loop corrections to the tadpole equations and the self-energies are computed, together with the **R**enormalization **G**roup **E**quations (**RGE**) for the chosen model.

It creates compatible output files for various other interfaces and programs, but the relevant ones in this thesis are the Fortran source code generation for `SPheno`, which will calculate the spectrum file in the SLHA format, and also the model file generation for `VevaciousPlusPlus`.

`SARAH` is accompanied with many examples and templates for model examples, which contain a "particle.m" file, which defines all the particles, a "parameter.m" file, which describes all the parameters, and a "NameOfModel.m" file, which specifies the group structure, the symmetries and mixings of the particles and the potential of the chosen model. When using `SPheno` outputs, which we will do, there's also a "SPheno.m" file that provides settings for the calculations and source code generation for `SPheno`.

In this work, Mathematica 11 with `SARAH` version 4.12.3 and 4.13.0 was used, together with some small patches for the `VevaciousPlusPlus` (see section 3.1) model file creation part. The model file for `VevaciousPlusPlus`, which is generated by calling the command "MakeVevacious[Version->"++"]" in Mathematica, provides all the needed information for the potential function and its 1-loop corrections.

It contains the used names for the parameters which can get a VEV in the XML tag "FieldVariables", the SLHA Blocks for the desired symmetry breaking minima in "DsbMinimum", the tree-level potential in "TreeLevelPotential" and all the loop correction terms in the tag "LoopCorrections", which also names attributes that define the renormalization scheme and the gauge fixing. The loop corrections consist of an extra polynomial part in "ExtraPolynomialPart" and all the mass squared matrices (which are needed for the corrections, see Eq. 18).

3 Developments

3.1 VevaciousPlusPlus

VevaciousPlusPlus [1] (this will be sometimes abbreviated as V++ or Vevacious++) is a C++ overhaul of the older python version Vevacious. This program will be developed further in order to enhance the functionalities and will be used to scan parameter configuration ranges of different models.

3.1.1 PHCpack

As already mentioned, Vevacious utilizes HOM4PS2 to find the tree-level extrema of the potential. The performance decreases drastically with increasing degrees of freedom, i.e. number of fields acquiring VEVs, which is why PHCpack, another modern open-source homotopy continuation solver was added as an additional option to solve the tree-level tadpole equations. PHCpack, as opposed to HOM4PS2, offers the possibility of multi-tasking, which improves computation times for big systems of equations. At first, the idea was to also include bertini2 [25] into the VevaciousPlusPlus code as it provides a C++ interface for homotopy continuation. Unfortunately, the core of bertini2 is difficult to compile (i.e. it didn't work with all the needed libraries installed and with a clean Ubuntu OS) and implementing the bertini interface would have needed a much more complex setup and installation of VevaciousPlusPlus. Thus, since the blackbox solver of PHCpack already works fine for all the purposes, bertini2 isn't integrated.

3.1.1.1 Implementation in VevaciousPlusPlus

The PHCRunner (.hpp and .cpp files) in V++ resembles in many aspects the already existing HOM4PS2Runner, since in principle it needs to do the same procedures. It also inherits from the PotentialMinimizer header file, which contains several virtual functions for other routines.

The PHCRunner constructor needs three arguments: the path to the "phc" executable, the "resolutionsize", which basically defines the tolerance of the imaginary part to be neglectable small and subsequently regarding the found solution as purely real. The final argument is the number of tasks, which is needed for multitasking. An instantiation of the PHCRunner solves a system with the use of operator() in which a vector containing the system to solve and a vector filled with vectors of doubles, the solution sets, are used.

When solving the system, the input file for PHCpack is written by calling WritePHCInput, which isn't discussed here since, except some input syntax changes, is mostly a copy of the

HOM4PS2Runner method Writehom4ps2Input. WritePHCInput writes an input file with all the polynomial constraints, i.e. the tadpole equations, in the right equation format which is required by `PHCpack`, by calling WritePHCConstraint.

Afterwards, the phc executable is called in the given path with the flags -b, which means it calls the blackbox solver, and -t taskcount, which allows multitasking. The blackbox solver proved to be a good, and most of the time the fastest, method to find the tree-level extrema, since this routine always picks the right settings to solve the system. To the "system(...)" call, the input and output file are appended too.

`PHCpack` appends the final solutions to the input file, which is why the output file, that contains all the information about errors, elapsed time, number of paths and further information, is not interesting for the final result. For this reason, ParsePHCOutput takes the input file and parses all the real solutions with an absolute value of the imaginary parts lower than the given resolution size. All the relevant parts of the code relating to the described functionalities can be found in Appendix A.2.

3.1.1.2 Performance comparison with `HOM4PS2`

In order to get an idea of the performance of `PHCpack`, a small program was written to compare the time needed to solve a general polynomial system with a certain amount of fields with `HOM4PS2` and `PHCpack`. This program generated n polynomials for n fields, containing all possible combinations of fields with random coefficients.

The coupling strengths were mitigated from a MSSM example given in the github repository of `VevaciousPlusPlus`. Of course, this will not comprise all physical potential shapes and scenarios, since the potentials are usually much simpler due to symmetries and constraints. This simple performance check should suffice since the computation time for real examples would be smaller due to less complexity in the potential function shape. It has to be noted that this check is just a demonstrative example rather than a full analysis of performance.

The C++ program `PolyTesting` asks in the console for the starting and end amount of fields, the field variable name and the task count for `PHCpack`. Then it will generate a vector fvec of *field* objects, with the given field variable name, which are needed to execute *writeequationgeneral(fvec)*. This is needed for the `PHCpack` and `HOM4PS2` input files, which differ in their form a little bit but not by much. With the $<$ *chrono* $>$ header features the computation time needed for `PHCpack` and `HOM4PS2` is measured. The computation times are finally appended to a file, together with the amount of fields.

The class *field* is initiated with its fieldname as a string. *quadcoupling(field s, int pow), quadcoupling(field f1, int i1 ,field f2, int i2)* and *cubiccoupling(field f, int fpow)*, member functions of *field*, all generate strings in the form of a coefficient followed by the fields. The coefficients are generated with $random\_device$ and $uniform\_real\_distribution <>$ from the $< random >$ header. Also, a random sign is generated via $bernoulli\_distribution$ with probability p=0.5.

$writeequationgeneral(vector < field > fvec)$ runs over each field in fvec and couples it with every other field using the member functions of the *field* objects. Like that it writes a chain of strings which generate a polynomial for each field. Since we couple bosonic scalar fields, permutations of the fields are equivalent. This is considered and bypassed in the for-loops, which saves memory and redundancy in the input files for high amount of fields. The most important parts of the source code can be found in the appendix B.

Fieldcount	Time `PHCpack` (taskcount = 8)	Time `HOM4PS2`
4	122 ms	76 ms
5	807 ms	55689 ms
6	3985 ms	1612078 ms
7	20312 ms	>2:30 hr
8	136517 ms	
9	573101 ms	
10	2851994 ms	

Table 1: Computation with Intel Xeon E3-1230 v5 @ 3.40 GHz (8 CPUs)

Fieldcount	Time `PHCpack` (taskcount = 4)	Time `HOM4PS2`
4	303 ms	96 ms
5	2087 ms	74774 ms
6	12695 ms	2168893 ms
7	73726 ms	>2:30 hr
8	450346 ms	

Table 2: Computation with Intel Core i5-4200U CPU @ 1.60 GHz (4 CPUs)

Glancing at tables 1 and 2 one can definitely see the significantly better performance of `PHCpack` *for a field count* $\geq$ *5.* For a field count of 4, `HOM4PS2` seems to be faster, even versus multitasking. The computation time seems to grow exponentially for all scenarios, which happens because the programs need to track growing amounts of homotopy paths with growing amount of complexity. Even if one divides the time `HOM4PS2` needs to the task count `PHCpack` used, the latter one still outperforms for field counts $\geq$5. This means its overall a better choice to use `PHCpack` for systems including many degrees of freedom.

We will also see later, that HOM4PS2 sometimes doesn't find all the extrema, which can be fatal.

3.1.2 LagrangianParameterManager

As the name already suggests, the LagrangianParameterManager manages the parameters of the Lagrangian. This includes reading the SLHA file and assigning the correct values to the right parameters at the given renormalization scale. There are several other mechanisms of the parameter manager, which aren't discussed here, but the functionalities can be understood by studying the commentary in the source code in [1].

In cooperation with F. Staub the VevaciousPlusPlus model file output from his Mathematica package SARAH [18] was adjusted to the newly created parameter manager "SARAHManager". In SPheno, one can switch the calculation of the loop masses on or off which results to different relations between SLHA blocks and the parameter names. Ideally one should always calculate the **1-loop** corrections (and not the 2-loop corrections!) to the masses, since VevaciousPlusPlus is working with the 1-loop potential.

Considering the polynomial part of the potential, the Vevacious++ model file contains the "TreeLevelPotential" part and the "ExtraPolynomialPart", in which the latter one contains 1-loop corrections to the mass terms [26]. When finding the minima on tree-level with for example PHCpack it is necessary to evaluate the solutions with only the tree level potential in "TreeLevelPotential" and for this the non-loop corrected masses are needed as parameters. When rolling to 1-loop the corrections to the polynomial part of the potential are needed too. They are given in the "ExtraPolynomialPart" which has polynomials with a structure like $(\mu^2_{H,loop} - \mu^2_{H,tree})\phi^2$. Adding the "ExtraPolynomialPart" to "TreeLevelPotential" we end up with $(\mu^2_{H,tree} + (\mu^2_{H,loop} - \mu^2_{H,tree}))\phi^2 = \mu^2_{H,loop}\phi^2$.

In the SLHA file the distinction between tree and 1-loop values is represented in block names commencing with "Loop" or "Tree". If the flag 55 in the LesHouches.in file is set to 1, which means loop masses are calculated, the "Loop" blocks have a non-vanishing value, which is why the model files of SARAH are generated with a syntax for the derived parameters like for example "IFNONZERO[Parameter1,Parameter2]". Parameter1 is picked as value if it is not zero, else Parameter2 is picked as value. For this to be recognized from the program, a routine was added to add the given derived parameters to the newly derived "SARAHManager". The old "SlhaCompatibleWithSarahManager" defined specific aliases to be recognized and calculated with their definitions. In the new model files generated, these aliases were obsolete and could lead to parameter definition overlappings, which is why a new parameter manager was introduced.

Since Ben O'Leary decided to part the model inputs in two files "ModelFile.vin" and "ScaleAndBlock.xml", of which the latter contains the valid SLHA blocks to be parsed, the derived parameters are also parsed from the ScaleAndBlock file.

3.1.2.1 Implementation

The old version of `VevaciousPlusPlus` wasn't able to distinguish between the tree and loop cases in a generic way, as already mentioned it was done only with pre-defined aliases, leading to inflexibilities and incompatibilities. When the LesHouchesAccordBlockEntryManager reads and interprets the "ScaleAndBlock.xml" file and finds an XML tag with "⟨DerivedParameters⟩" it runs ParseDerivedParameters, which inputs the whole XML body as a string. ParseDerivedParameters basically sorts the parameters with its definitions and saves them in a vector of string pairs called derivedparameters.

The most important and relevant part for saving the parameters is found in the "SlhaCompatibleWithSarahManager" and "LesHouchesAccordBlockEntryManager" header and source files. There, a member function RegisterDerivedParameters, which inputs the vector of a pair of strings, given in the format of the "derivedparameters" vector from before, registers every single given parameter. The pairs of strings in "derivedparameters" have the parameter name in the first argument and the assignation as the second argument. As an example "MuLoop = IFNONZERO[MuSarahLoop, Mu]" means, that if the defined parameter MuSarahLoop, which also has a definition itself like for example a relation to the SLHA block "LOOPHMIX[1]", has a value non zero, this value is assigned by the ParameterManager, else the value of the parameter Mu, defined "Mu=HMIX[1]", is taken.

RegisterDerivedParameters checks every element of the input vector by using iterators with the valid options of parameter definitions and registers them in the LagrangianParameterManager using the given functions of V++. The relevant code can be found in the Appendix A.3.

3.1.3 Optimization with smart pointers

Due to the fact, that the usage of Regular Expressions (RegEx) required the C++11 standard, smart pointers can also be used to optimize the memory allocation of the created objects inside the code. Before, the absence of unique ownerships or smart pointers in general required multiple copies of the created objects, thus using more memory. When using V++ as a dynamic library it showed, that in some cases there were bugs with the deletion of the objects, which can be fixed by using unique and shared pointers. Smart pointers

facilitate many usages and can be implemented in extensions of `VevaciousPlusPlus`, since one doesn't need to care about the memory deallocation of an object. When the defined smart pointer goes out of scope, the object is deleted, which is equivalent to a smart garbage collector which is present in other programming languages like Java. For future extending developments this can be very useful, since the code is much less prone to bugs and due to unique ownerships, no modifications from unwanted instances is assured. When for example creating many V++ objects with multitasking in a cluster, the memory is secured and safely deallocated after leaving the usage scope.

A particular interesting pointer which is used in the code is the unique_ptr. These kind of pointers have the feature of being non copyable and only movable. This means that the ownership only belongs to one unique_ptr, which cannot be copied to another pointer. The only possibility is to move ownership to another unique_ptr, which leaves the old owner empty. Using this in the whole program when creating objects decreases the amount of memory needed and the unique ownership, as already mentioned, shapes the code to be much less vulnerable to bugs.

3.1.3.1 Implementation

All the changes which have been made related to the implementation of smart pointers aren't being documented in this thesis, since the whole program had to be changed to create unique_ptrs and not just raw pointers, which comprises a huge amount of code. For this reason, only the basic idea and form of the changes are given here. All the changes can be found on the public github of `VevaciousPlusPlus` [1].

In the source file VevaciousPlusPlus.cpp the objects are dynamically allocated via "new CreateObject(Arguments)". All these objects have a destructor "∼Object {...}", which deletes the pointers given to their constructors. Every "CreateObject(Arguments)" was modified in such a way, that every pointer created in these functions is a unique_ptr and when given to another method is moved there using std::move(unique_ptr), which makes the destructors redundant. To allocate memory using unique_ptr's the make_unique command is being used throughout the code. As this is only available in C++14, make_unique was manually added, analogously to its definition in C++14, given in Listing 1.

```
1 template<typename T, typename... Args>
2 std::unique_ptr<T> make_unique(Args&&... args)
3  {
4   return std::unique_ptr<T>(new T(std::forward<Args>(args)...));
5  }
```

Listing 1: Definition of make_unique

3.2 `VPPparamScanner`

In order to conveniently scan ranges of parameter configurations, check conditions, calculate parameters, read outputs and generate tables for plots, a program in C++, using the standard C++11, was written and used in this thesis. The program includes many small parts, i.e. tools for XML parsing or interpreting the mathematical expressions given as conditions with the library exprtk [27], but because of lengthiness only the main principle of operation and usage is shown in the following. The most important steps of the program will be discussed and also the ingredients for using the program. The source code is publicly available on github [4] which includes a README file to install it and template configs with explanatory comments.

3.2.1 Console arguments

On a Linux based system, one executes the program by using the console. This executable needs as console arguments **at least** one config file, which will be discussed later, and optionally one can append different settings for the iterations.

These settings encompass skipping of `SPheno`, `VevaciousPlusPlus` or the parsing and writing of output tables. This proves useful for example if the spectrum files of `SPheno` are already given and don't need to be calculated. For already given output files of `VevaciousPlusPlus` one can also just parse the files. If one only wants to produce spectrum files and V++ outputs, the parsing can also be skipped. There are more commands which can be given by console, but these are not relevant in the analyses of this thesis.

3.2.2 ReadConfiguration

This part is crucial for the usage of the program, since the configuration file needs to follow a given syntax for the XML bodies to define a configuration. The configuration is realized as a struct in the program and is passed through all main function calls in the program.

As a convention for XML tags "parent.child" relates to ⟨parent⟩ ⟨child⟩ ... ⟨/child⟩ ⟨/parent⟩. The config file starts with the V++ tag "VevaciousPlusPlusMainInput.InitializationFile" which points to the initialization files of the potential minimizer, the potential function and the tunneling calculator. In "VevaciousPlusPlusMainInput.RunPointInput" and "VevaciousPlusPlusMainInput.OutputFilename" one needs to just specify the directory where one wants to place the outputs of `SPheno` and `VevaciousPlusPlus`.

Afterwards, the `SPheno` executable is given in the tag "SPheno.executable" which consists of the path to the executable and its name. Also, the path and name of the LesHouches.in file needs to be given in "SPheno.inputfile".

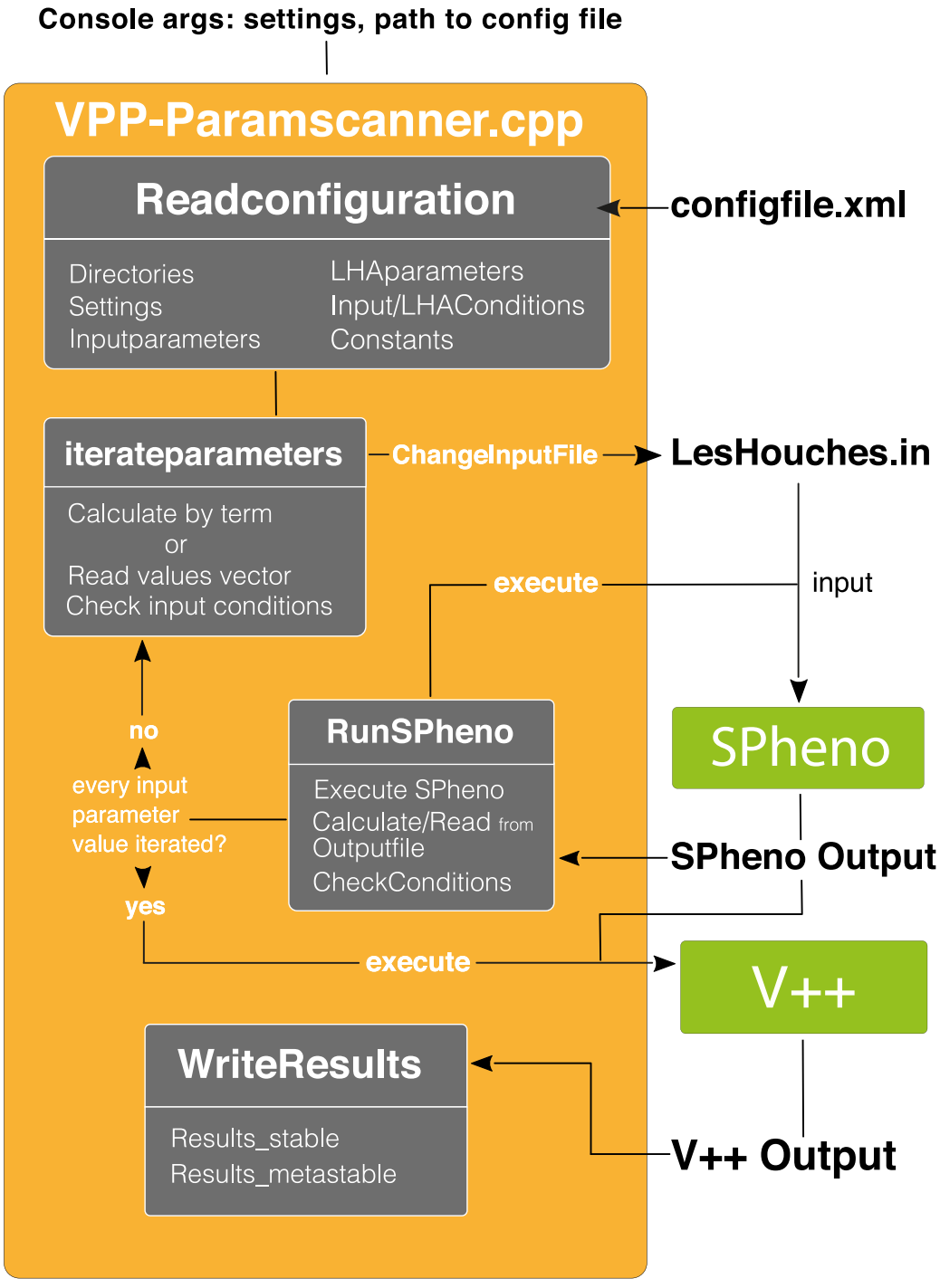


Fig. 4: The program structure of `VPPparamScanner`. The figure describes the main steps in the developed program as an overview, from input manipulation to output parsing with the console option -slhafirst.

The last big XML body "programspecific" contains settings for the execution of the program. In "programspecific.Results" the folder where the final results are printed is given. In this folder another folder with a timestamp is created after a successful run, where "Results_stable", "Results_metastable" and "Erroneousinputs" together with the given configuration file are placed.

If desired, one can change the delimiter in "programspecific.delimiter" used in the runtime of the program, which is just implemented for cosmetical reasons. The default setting for the delimiter is the underline character "_".

In the LesHouches.in file one can determine whether loop masses are calculated or not. In order to have this freedom in the program, an option to do so is included. By setting "programspecific.Loop" as 1, the flag 55 in the given `SPheno` input file is set to 1, and analogously, if set to 0, the flag 55 is set to 0.

"programspecific.VevaciousPlusPlusExec" needs to point to the `VevaciousPlusPlus` executable.

If desired, one can define constants in "programspecific.constant", which has a name "constant.name" and a double value "constant.value". These are useful for calculated parameters, which will be discussed soon.

An important part of the config file is the definition of the input parameters. These input parameters can either be just dummy variables defined to calculate input parameters or they can directly be a part of the "MINPAR" block in LesHouches.in (or optionally another block name defined in "inputparameter.blockname"). They need to at least be described by a name in "inputparameter.name".

There are now two types of such inputs. One has discrete values given by the user, which can be given in "inputparameter.values" or otherwise range-based declared in "inputparameter.beginval" , "inputparameter.endval" and "inputparameter.stepsize". The other type of input parameter doesn't include discrete values but a term given as a string in "inputparameter.term" which describes the functional dependence to other parameters. Here the defined constants come in handy and also the "inputparameter.name" of the other parameters can be used in the "inputparameter.term".

Then, **if** given a block as an integer in "inputparameter.block", the parameter refers to a MINPAR (or another block name) input. If there's no block given, it is assumed to be solely for the purpose of the calculation of other MINPAR parameters.

Additionally to input parameters, there are also LHA parameters, which are related to the output file `SPheno` produces. The definitions are set in "programspecific.lhablock"

which has a parameter name "lhablock.name", the block name it reads from the output "lhablock.blockname" and an integer "lhablock.block". This will read the value from the given SLHA blocks. Also, it's possible to calculate the corresponding value from all registered parameters, which consists of a "lhacalc.name" and a "lhacalc.term". The difference to input parameters is the time when they are calculated or read, since obviously at first a `SPheno` output needs to be created to read LHA parameters. This becomes useful if one wants for example to read loop corrected masses and calculate other quantities with them.

Lastly there are two types of conditions, realized as a "term". These terms should be of a Boolean type, obviously. There are "programspecific.condition.inputcondition" which are being checked before RunSPheno is started. These are useful when using range-based input parameters, but one wants them to fulfill some sort of inequality.

The other types are "programspecific.condition.lhacondition" which are being checked after the output file has been generated. It will remove the output file when the condition is not met. For some situations one can exclude spectrum files with phenomenologically excluded masses and save computation time with V++.

3.2.3 Iterateparameters, RunSPheno

After having registered all the given parameters and settings, the main part of the program comes into play. ReadConfiguration creates objects of InputParameter, which are saved in a vector<InputParameter>. These InputParameter instances contain a vector<double> *values*, where depending on the kind of input parameter the values are saved. When they are calculated by term, they only contain one value.

In order to get all the different combinations of values a recursive function iterateparameters(shared_ptr<Configuration> c, vector<InputParameter>::iterator iter) iterates through all single input parameters of which each also iterates over all doubles in the values vector. All the current values are saved in the vector<pair <string,double> > activeparameters, which connects a variable name with the value inputs.

In the beginning, the iterator points at the first parameter, and with every recursive call, the iterator is set to the next "InputParameter" until the end of the vector is reached.

When it generated one complete set of input parameters, after checking the input conditions, the LesHouches.in file for `SPheno` is changed. Inside the file, the MINPAR parameters are edited to the given values in activeparameters. Also, the loop mass calculation flag is changed, depending on the setting in the configuration file. Since `VevaciousPlusPlus` only needs the spectrum file, other flags, like for example the flag of Block 75 "Write

WHIZARD file" in "Block SPhenoInputs", are set to 0. Also, in order to just integrate the 1-loop corrections to the Higgs calculations into the SLHA output, namely the TREE and LOOP cases described in section 3.1.2, the flag 7, which skips the 2-loop mass calculations, is set to 1.

When activeparameters is filled with a parameter configuration it consists of all the MINPAR parameters, all the calculated input parameters and all the defined constants. The method RunSPheno(shared_ptr<Configuration>c) is called afterwards. It makes a system call to the `SPheno` executable together with the path to the LesHouches.in file.

Then the LHAparameters are being read from the output files and added to activeparameters. The LHAparameters containing a term are now calculated from all active parameters. Afterwards, when the lhaconditions are calculated and compliant, the output file is saved in the given folder in "VevaciousPlusPlusMainInput.RunPointInput".

This procedure repeats itself until every run point file is created and then runs V++ separately. Due to some errors, which were present at the time of development, every run point is being calculated independently by V++.

There is also an option to run a whole directory with V++ with one model file, but there were errors with the updating of the Lagrangian parameters. This problem will probably be solved soon and then also the whole directory can be taken as input, which can also be set in the config file.

3.2.4 WriteResults

After all the output files, which are in the XML format, have been created, the relevant information is being parsed with a XML parser using the boost library. All the parameter values of activeparameters, which have been appended to the Vevacious++ output file in the XML tag "parameters" are now parsed too.

If the parameter configuration is stable, all the relevant information is saved in "Results_stable" and analogously with "Results_metastable".

There's also a file "ErroneousInputs", which contains all the parameter configuration which either had problems with `SPheno`, i.e. if there are negative mass squared arguments it returns an error, or with `VevaciousPlusPlus`, i.e. the tunneling calculator has problems, or if the given conditions, either input or output, are not met.

To conclude, this program was written to conveniently generate tables which show all the information one needs to interpret the stability of the vacuum of certain parameter spaces. Since one doesn't need to be an expert with the code structure of `VevaciousPlusPlus`,

and one can easily edit this source code for special needs, it should be a more user-friendly alternative to generate result files without going deep into the interface of V++ itself. This program isn't really useful for cluster calculations and Message Passing Interfaces (MPI) accompanied by complicated systems with many symmetry breaking mechanisms, but in the range of this thesis and for fast scans of relatively simple needs, this tool should be useful.

4 Vacuum stability analyses

4.1 Real scalar triplet extension of the Standard Model

An extension of the SM Higgs sector with a real scalar triplet, **T**riplet extension of the **S**tandard **M**odel (**TSM**), transforming under $SU(3) \times SU(2)_L \times U(1)_Y$ as $(1_C, 3_L)_{Y=0}$ [28], promises interesting features like for example a dark matter candidate and also, in a specific setup of the model parameters, the possibility to generate a strongly first order two-step electroweak phase transition [29], which can be used to give answers to the baryogenesis problem of the SM.

4.1.1 Model specifications

To get a grasp of the model (this will be more or less a summary of the complete description in [28]) the most relevant aspects about the real triplet scalar extension are discussed in the following.

Let's define the triplet with the real fields Σ^0 and the two complex fields Σ^+ & Σ^-, that obey $\Sigma^+ = (\Sigma^-)^*$ and which are superpositions of two real scalar fields σ^1 and σ^2 with the form $\Sigma^\pm = \frac{1}{\sqrt{2}}(\sigma^1 \mp i\sigma^2)$, as

$$\Sigma = \frac{1}{2}\begin{pmatrix} \Sigma^0 & \sqrt{2}\Sigma^+ \\ \sqrt{2}\Sigma^- & -\Sigma^0 \end{pmatrix}. \tag{20}$$

Using the same conventions for the Standard Model Higgs doublet from section 2.2.1, the scalar part of the Lagrangian reads

$$\mathcal{L}_{scalar} = (D_\mu \phi)^\dagger (D^\mu \phi) + Tr\left((D_\mu \Sigma)^\dagger (D^\mu \Sigma)\right) - V(\phi, \Sigma), \tag{21}$$

where D_μ describes the covariant derivative under the Standard Model symmetry group. The potential part of the Lagrangian $V(\phi, \Sigma)$ will be interesting for the further considerations, since we want to examine the vacuum potential structure and its minimum configurations.

Setting $F = (\Sigma^0)^2 + 2\Sigma^+\Sigma^-$, which is equivalent to the negative Trace of Σ in Eq. 20, and requiring renormalizability, the potential in its most generic form can be written as

$$V(H, \Sigma) = -\mu^2 \phi^\dagger \phi + \lambda\left(\phi^\dagger \phi\right)^2 - \frac{1}{2}\mu_\Sigma^2 F + \frac{b_4}{4}F^2 + a_1 \phi^\dagger \Sigma \phi + \frac{a_2}{2}\phi^\dagger \phi F. \tag{22}$$

If the coefficient a_1 vanishes, which will be considered from now on, the scalar part of the

potential exhibits two global symmetries:

$$O(4)_H \times O(3)_\Sigma \quad \text{and} \quad (Z_2)_H \times (Z_2)_\Sigma, \tag{23}$$

where Z_2 invariance means reflection symmetry and $O(n)$ is the orthogonal group in n dimensions.

This model will be called **TSMZ2** (**T**riplet **S**tandard **M**odel with $\mathbf{Z_2}$ symmetry) from now on. The TSMZ2 model [29] may yield a realistic scenario for electroweak baryogenesis, as already mentioned before, and also a cold dark matter particle, namely Σ^0, which makes it interesting to investigate in detail.

4.1.2 Neutral Higgs symmetry breaking

Before analyzing charge breaking, it is beneficial to begin looking at symmetry breaking in the neutral components first, because the tree-level relations for vacuum stability can be easily derived and then compared with the 1-loop results of `VevaciousPlusPlus`. This will be a great check whether V++ is working correctly, since the program itself hasn't been tested much because it is still in the alpha development phase.

The tree-level scalar potential V at T=0 with a non-vanishing VEV for Σ^0 and ϕ^0, i.e. with symmetry breaking in the neutral components, reads [29]

$$V(h,\sigma) = -\frac{1}{2}\mu_H^2 h^2 - \frac{1}{2}\mu_\Sigma^2 \sigma^2 + \frac{1}{4}\lambda h^4 + \frac{1}{4} b_4 \sigma^4 + \frac{1}{4} a_2 h^2 \sigma^2. \tag{24}$$

The dynamical fields for the neutral Higgs doublet component are here denoted as h, the one for the triplet as σ. As we want to minimize the potential, the minimum conditions are given by the vanishing of the first derivative with respect to the variables h and σ

$$\begin{aligned} \frac{\partial V(h,\sigma)}{\partial h} &= h(-\mu^2 + \lambda h^2 + \frac{1}{2} a_h \sigma^2) = 0 \\ \frac{\partial V(h,\sigma)}{\partial \sigma} &= \sigma(-\mu_\Sigma + b_4 \sigma^2 + \frac{1}{2} a_2 h^2) = 0. \end{aligned} \tag{25}$$

The tadpole equations are solved by three different VEV configurations. There are solutions with following configurations: One with vanishing σ VEV: (h_{min},0), from now on called H vacuum, one with vanishing h VEV (0, σ_{min}), now called Σ vacuum and another with VEVs for both fields (h_{min}, σ_{min}), now called T vacuum.

If the EWSB vacuum H should be stable towards the Σ vacuum, we need to compare the

minimum configurations of $V(h,0)$ and $V(0,\sigma)$. H stays stable if

$$V\left(h_{min},0\right)_H < V\left(0,\sigma_{min}\right)_\Sigma \quad \text{with} \quad h_{min} = \frac{\mu_H}{\sqrt{\lambda}} \quad \text{and} \quad \sigma_{min} = \frac{\mu_\Sigma}{\sqrt{b_4}}, \tag{26}$$

in which h_{min} and σ_{min} can be easily derived from the extremum conditions, the tadpole equations in Eq. 25.

Looking at the inequality in Eq. 26 and using the tree-level identities at the electroweak vacuum $m_H^2 = 2\lambda v_0^2$ and $m_\Sigma^2 = -\mu_\Sigma^2 + \frac{1}{2}a_2 v_0^2$ with $m_H^2, m_\Sigma^2 > 0$ [29] yields the condition

$$\frac{1}{b_4}\left(m_\Sigma^2 - \frac{1}{2}a_2 v_H^2\right)^2 < \frac{1}{2}v_0^2 m_H^2. \tag{27}$$

Note that in the paper [29] there seems to be an error in their Eq. 7 since it seems that they forgot the square outside of the bracket in the left term. In the figure in which they use their inequality they seem to use the right formula of Eq. 27 so it's obviously just a typing mistake right there. Also, the next equation Eq. 8 in their paper has a typing error, the inequality operator needs to be reversed. This is demonstrated in the following.

If we assume the existence of the Σ minimum $(0,\sigma_{min})$, which is needed for the two-step phase transition, we need to require that the mass squared, which corresponds to the curvature of the potential at this point, of both Higgs are positive. So to speak, the second derivatives of the potential Eq. 24

$$\begin{aligned} M_H^2(h,\sigma) &= \frac{\partial^2 V(h,\sigma)}{\partial h^2} = -\mu^2 + 3\lambda h^2 + \frac{1}{2}a_2\sigma^2, \\ M_\Sigma^2(h,\sigma) &= \frac{\partial^2 V(h,\sigma)}{\partial \sigma^2} = -\mu_\Sigma^2 + 3b_4\sigma^2 + \frac{1}{2}a_2 h^2 \end{aligned} \tag{28}$$

at the point $(0,\sigma_{min})$ needs to be greater than 0.

This translates to the conditions

$$-\mu_H^2 + \frac{1}{2}a_2\sigma_{min} = -\mu_H^2 + \frac{1}{2}a_2\frac{\mu_\Sigma^2}{b_4} > 0 \quad \text{and} \quad -\mu_\Sigma^2 + 3b_4\sigma_{min}^2 = 2\mu_\Sigma^2 > 0. \tag{29}$$

Inserting the definition of the masses from before and after some equation rearrangements, we finally arrive at the inequality

$$\frac{a_2}{b_4}\left(\frac{1}{2}a_2 v_0^2 - m_\Sigma^2\right) > m_H^2. \tag{30}$$

In addition to these constraints, if we want to keep a stable EWSB minimum with VEV v_0, then the quartic coupling coefficients, namely λ and b_4, need to be greater than 0,

since these parameters determine the shape of the potential edges to be concave or convex. Obviously, a concave shape of the potential will make it unbounded from below, which is unphysical in itself.

Lastly, there are minima points which have a VEV for both h and σ. These points are now called $V(h_m, \sigma_m)_T$. In this case, the requirement that the potential is bounded from below and that the EWSB minimum H potential depth is lower, i.e. $V(h_m, \sigma_m)_T > V(v_0, 0)_H$, yields the inequality (given in [28])

$$\frac{a_2^2}{4\lambda} < b_4, \tag{31}$$

which holds for negative a_2 and positive λ & b_4.

4.1.2.1 `SARAH` model file

To generate the TSMZ2 model with the desired symmetry breaking schemes in the `VevaciousPlusPlus` model file and `SPheno` Fortran code, an already existing model file created by F. Staub was used and edited.

He already introduced a model "SM+Triplet-Real" which exhibits no Z2 symmetry. The term $a_1\,\phi^\dagger\,\Sigma\,\phi$ from Eq. 22 breaks Z2 symmetry and is present in the `SARAH` model file. Because we want to investigate the TSMZ2 model this term needs to be removed or a_1 needs to be set zero at all times.

`SARAH` here solves two tadpole equations with respect to h & σ and eliminates two degrees of freedom with it. For all cases the starting point with $\sigma = 0$ is chosen and thus Mathematica has problems to solve the second tadpole equation in Eq. 25, since the bracket content can be arbitrary and no parameter can be eliminated.

As EWSB means $v_0 \approx 246\,\text{GeV}$ and $\sigma = 0\,\text{GeV}$, this gives rise to the mentioned problem. A bypass to this is to simply generating the `SPheno` code with a TSMZ2 model file **without** symmetry breaking in Σ^0 and generate the `VevaciousPlusPlus` model file **with** symmetry breaking in Σ^0.

By doing this, the tadpole equations are solved just in one variable, namely μ in Eq. 22, and `SPheno` yields valid spectrum files for the EWSB case. `VevaciousPlusPlus` inputs the LHA spectrum file and reads the VEV of Σ^0 as zero.

4.1.2.2 `VevaciousPlusPlus` results

The `VPPparamScanner` config files to generate the results for the cases studied in the following can be found in the Appendix C in order to get an idea of the general setup.

Using `SPheno`, spectrum files with the run point at the EWSB vacuum H $(v_0, 0)$ were created, where the output files with the chosen parameter configurations are given in the LHA format. To compare the results with the paper [29], some of these parameter sets, like for example those in Figs. 6b and 8a, were chosen to be in the same ranges.

Since `VevaciousPlusPlus` computes the 1-loop contributions, the tree level inequalities Eqs. 27, 30 and 31 won't be reproduced completely. As far as 1-loop corrections are moderate and not too high, the V++ results should be comparable to the tree-level relations, meaning that the general structure looks about the same with small shifts. Also, in the following plots `SPheno` will always calculate the loop corrected Higgs masses, given by the flag "55" set to 1 in the LesHouches.in file.

It has to be noted, that the survival probability $P_{survival}$ of a metastable state most of the time is either 100% or 0%, but in some rare cases there are also values in between. For these scarce points, if $P_{survival}$ was above 68%, the run point was declared as "Metastable + H survives", and below as "Metastable + H unstable". This doesn't really affect the plots though, since there are not many points which need to be considered.

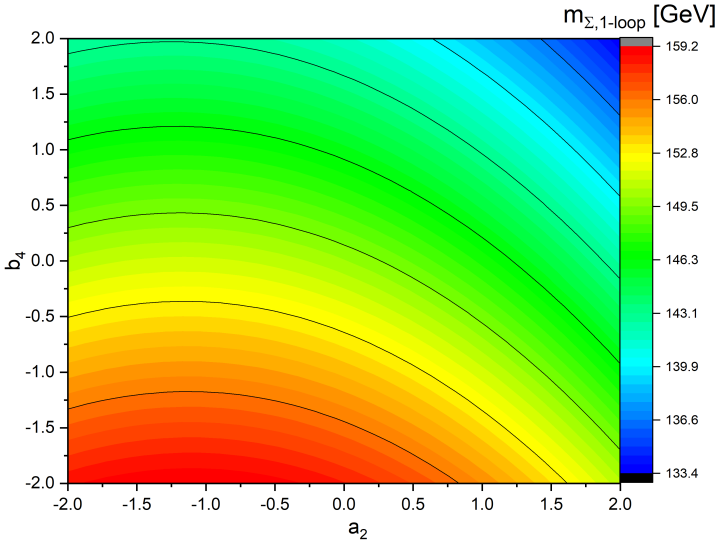


Fig. 5: In this plot, the variation of the loop corrected mass inside the parameter space $a_2 \in [-2, 2]$ and $b_4 \in [-2, 2]$ for a constant tree-level mass $m_{\Sigma,tree} = 150$ GeV is shown. The loop corrected mass is shifting the tree-level mass up to 15 GeV in the whole parameter space, which means up to 10%. The shifts get stronger for higher b_4 and a_2 and are small for vanishing b_4. Variation in a_2 for constant b_4 has a smaller effect on the corrections.

In Fig. 5, the variation of the loop corrected mass in the parameter space $a_2 \in [-2, 2]$ and $b_4 \in [-2, 2]$ for a tree-level mass $m_{\Sigma,tree} = 150$ GeV is shown.

The loop shifts are strong at the edges (except those with negative a_2 and high b_4 and vice versa) of the parameter space and low for small a_2 and b_4. Variation in b_4 changes

the amount of the loop shift significantly, reaching up to 10% difference from the tree-level value. Variation in a_2 for constant b_4 has a smaller effect to the corrections, but are non-negligible too.

We can conclude from this, that the following plots with constant tree-level mass Fig. 6 have different loop corrected masses in the range of up to 10% difference from the tree-level value, especially at the edges of the parameter ranges.

In Fig. 6 not the loop corrected neutral triplet mass can be taken as input parameter of the LesHouches.in file, since it is calculated by `SPheno`. The mass is held constant, but when taking a constant tree-level mass as input, the loop corrected mass will vary for different a_2 and b_4. In the plots given in this analysis, the negative b_4 points were just included for the reason of checking for numerical stabilities and also for aesthetic reasons.

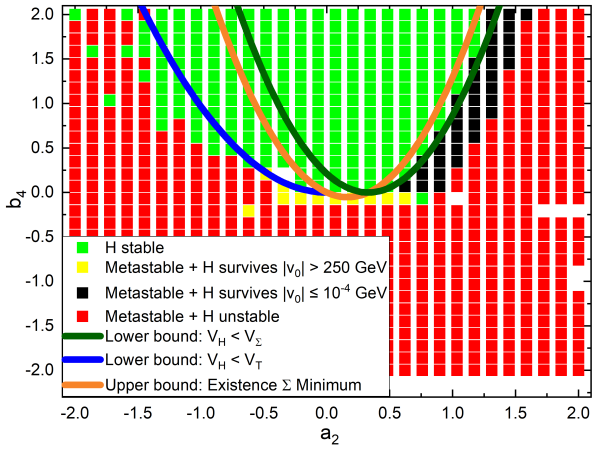
Let's now analyze the plots Figs. 6a, 6b and 6c. Constant tree-level triplet masses $m_{\Sigma,tree} = 100$, 150 and 200 GeV were tested, which all seem to fit nicely to the given inequalities. For $a_2 < 0$ the lower bound for the condition $V_H < V_T$ represents the tree-level expectations satisfactory, although there seem to be some exceptions for $a_2 < 0$, which will be discussed soon.

Furthermore, the lower bound for the condition $V_H < V_\Sigma$ seems to hold for the given parameter range of a_2, since the parameter configurations above the bounds all show a stable EWSB Vacuum.

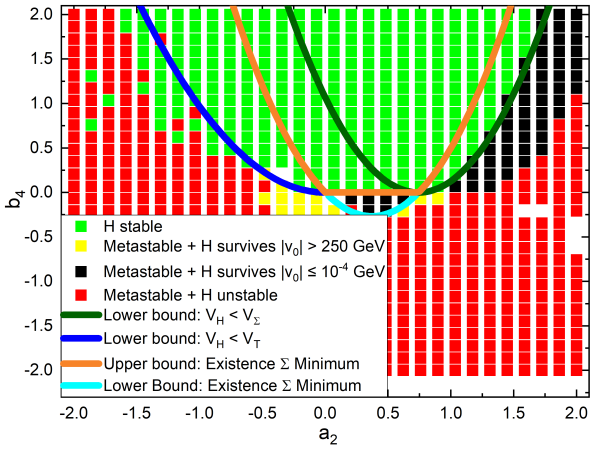
The bound for the Σ vacuum existence, Eq. 30, which is by the way an upper bound for positive b_4 and a lower bound for negative b_4 as depicted in the figures, also fulfills the tree level expectations in Figs. 6a, 6b and 6c. Except of some parameter configurations on the very edge inside the bound, which is probably due to the 1-loop corrections to the minima, the tree-level expectations are met. When $V_H < V_\Sigma$ and $V_H < V_T$ is fulfilled, the H starting point is stable as it is the deepest vacuum state, which is represented in the plots as well.

For negative a_2 the far-left points near the bound $V_T > V_H$ in Figs. 6a, 6b and 6c are not stable in independent `VevaciousPlusPlus` runs. Even when using the same LHA files, some of these critical parameter configurations return a very deep minimum with VEVs for both h and $\sigma \approx 10^9$ GeV and the point is declared unstable by `VevaciousPlusPlus`, due to the extremely fast tunneling. Also, sometimes not a deeper minimum is found for the same point.

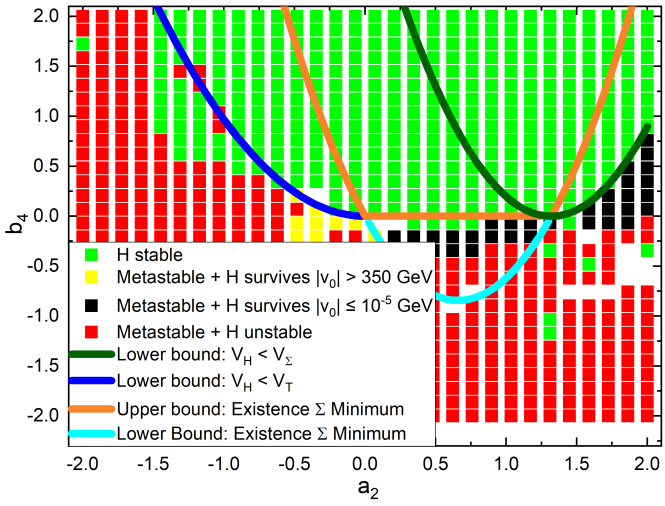
This seems to be a numerical instability with `MINUIT`, because extensive testing for some of the precarious parameter configurations revealed, that `MINUIT` sometimes rolls to different



(a) $m_{\Sigma,tree} = 100\,\text{GeV}$



(b) $m_{\Sigma,tree} = 150\,\text{GeV}$



(c) $m_{\Sigma,tree} = 200\,\text{GeV}$

Fig. 6: EWSB Vacuum H stability for tree-level neutral triplet masses of (a) $m_\Sigma = 100\,\text{GeV}$, (b) $m_\Sigma = 150\,\text{GeV}$ and (c) $m_\Sigma = 200\,\text{GeV}$ plotted with respect to the parameters a_2 and b_4 of the potential Eq. 22. The upper and lower bounds of Eqs. 27, 30 and 31 are also depicted in the plots. The derived tree-level bounds for the vacuum stability fit well into the stability results of the 1-loop potential calculated by `VevaciousPlusPlus`. Ergo the loop contributions seem to be sufficiently small to not change the stability behavior inside the parameter space. The metastable points with small neutral doublet Higgs VEV $v_0 \leq 10^{-4}$ GeV are depicted in contrast to v_0 many magnitudes higher. The points $v_0 \leq 10^{-4}$ GeV could be explained by numerical corrections to zero, which is demonstrated in Fig. 7. This means that these points are probably relating to a Σ vacuum with $v_0 = 0$ GeV. The tree-level relations also support these results in the plot, which means that only in the bounds of the existence of the Σ vacuum these points exist. For higher masses, the stability parameter space volume gets broader, and as expected, for $b_4 < 0$ there's always a deeper vacuum point due to the concave shape, which means the vacuum H is definitely metastable.

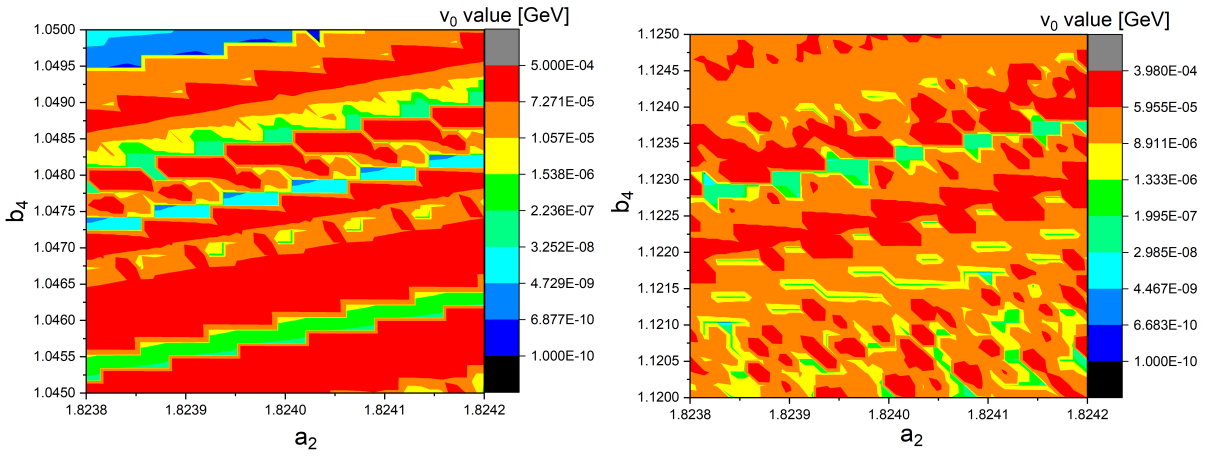
vacuum points, which sometimes yields deeper vacua and sometimes not.

The case of the result uncertainty happens when there are shiftings in the basin of attraction [1] so that a non **D**esired **S**ymmetry **B**reaking (**DSB**) minimum rolls to the DSB minimum due to loop corrections, which is handled by `VevaciousPlusPlus` with a scale transformation (default is a factor 4) of the tree-level minimum point. These scaling transformations result to high starting point field values which are then rolled to different very high 1-loop field values or back to the DSB values. This concludes two different outcomes: V++ finds a deep minimum and declares it as unstable or doesn't find a deeper minimum and declares the point as stable. This isn't really a problem in the broader picture of the full plot, since this only happens for some of the parameter points, but caution is required when jumping to conclusions in these parts of the results. The explanation for this phenomenon may be, that the gradient descent method is unstable for a potential shape which is very wiggly. Of course, the update rate of the method and the amount of function calls can then alter the directions it rolls to.

The points for positive a_2, especially these with positive b_4, produce in different independent runs constantly the same outcome, even with different `MINUIT` accuracies and settings, which makes them authentic towards these numerical instabilities.

Sometimes, also points with negative b_4 and positive a_2 seem to yield different results, As for example in Fig. 6c. There are parameter configurations which are declared stable although the potential should be unbounded from below. Detailed tests showed, that these points are also problematic due to the issue with the loop-corrections rolling back to the DSB values. Running these parameter configurations consecutively yields different results and changing the `MINUIT` strategies or precisions also changes the outcome. As this may sound problematic, in the broader picture this also doesn't really affect further studies, since these points can be easily extracted and thoroughly studied with different `MINUIT` settings for the gradient descent. For further V++ plots the "DSB rolling" will be treated with caution and tested with different settings for `MINUIT`.

When there exists a Σ vacuum and $V_H > V_\Sigma$, there's a deeper minimum with $(h, \sigma)_\Sigma = (\approx \mathbf{0}, \sigma_{min})$, but for some parameter ranges, depicted as "Metastable + H survives" in the plots, the tunneling time exceeds the time of the universe (yellow or black squares in the figures). This broadens the acceptable parameter space for the range given in [29], since they only concluded the space between the bound of Σ existence and $V_H < V_\Sigma$ to be interesting for their two-step electroweak phase transition. When allowing charge breaking though, these additional Σ vacuum points will be mostly replaced by deeper charge breaking minima, see section 4.1.3.



(a) (b)

Fig. 7: In these plots, the h field value in dependence of the parameters a_2 and b_4 for a triplet mass $m_{\Sigma,tree} = 150\,\text{GeV}$ is shown. In both cases, a sample parameter point out of the "Metastable + H stable" configuration of Fig. 6b with $v_0 \leq 10^{-4}$ was chosen. In both cases, there were multiple cases with $v_0 = 0$, which especially can be found in (a). Given that the parameter range is small and the deviations of the field value in many magnitudes from 10^{-4} down to 10^{-10} and lastly to 0, supports the assumption of numerical issues, which means that these points could relate to the Σ vacuum $(h, \sigma) = (0\ ,\ \sigma_{min})$.

The $(h, \sigma) = (\approx 0, \sigma_{min})\,\text{GeV}$ points are noted as such, because the field values of h are varying in the magnitude from 10^{-4} to 10^{-10} but aren't always exactly zero. There do exist some parameter points which have a field value of h exactly zero but this is rarely the case. In these configurations, the field values of σ are valued in the magnitude of 10^2.

In the contour plots Fig. 7 one can see the σ field values in a very small parameter range of a_2 and b_4 for $m_\sigma = 150$ GeV. The plots conclude, that small shifts in a_2 or b_4 can alter the value of v_0 in many magnitudes, like for example from 10^{-4} to 10^{-10}, but compared to the field value of σ stay low.

Furthermore, comparing all the different "Metastable + H stable" parameter configurations, there seems to be a huge change in v_0 from really small values $\leq 10^{-4}$ to $> 10^1$ with no values in between, as can be for example seen in Fig. 6b. Therefore, we can assume, that the small v_0 values can be regarded as numerical, irrelevant, corrections to $v_0 = 0$ and classify these points as the Σ vacuum.

The metastable points, where H survives for negative b_4 aren't to be fully trusted too. They are the same for independent runs but are not the same for different `MINUIT` accuracies, which defines the amount of function calls being made for gradient based minimization. The potential shape for negative b_4 per se is not well shaped, since the quartic term for the σ field shapes the potential to be unbounded from below (at least in the direction

with $v_0 = 0$). This means the parameter points for $b_4 < 0$ aren't physical in terms of stability of the vacuum H and can be disregarded in many cases of model study, and if the points with $b_4 < 0$ are declared stable, the numerical methods seem to not suffice to find the global minimum.

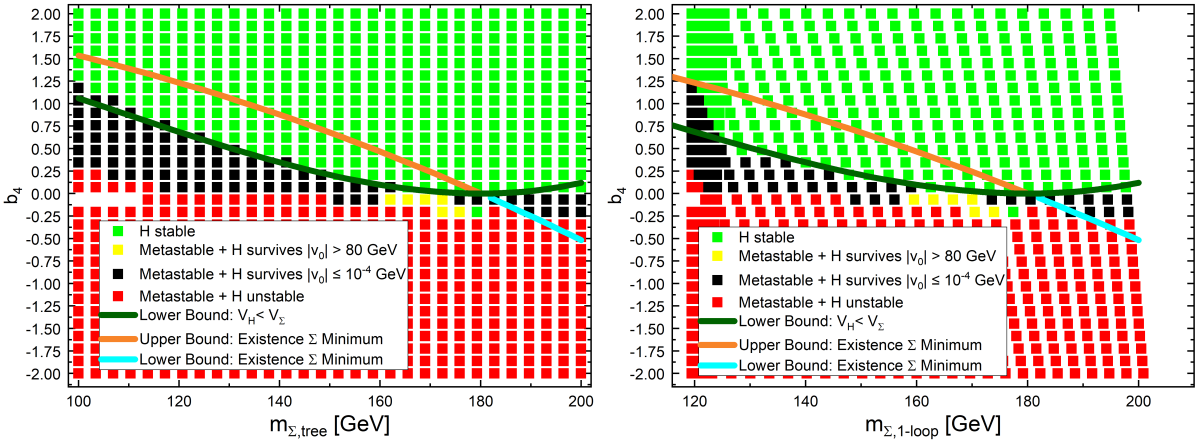
In the case, where we vary the tree-level triplet mass $m_{\Sigma,tree}$ in the LesHouches.in file, we can include the loop corrected mass in the plots, since we can read it from the `SPheno` output file. The tree-level mass in the following plots always runs between 100 and 200 GeV. Fig. 8 shows the vacuum stability for parameter configurations with constant $a_2 > 0$ and tree-level mass $m_{\Sigma,tree}$ or loop corrected neutral triplet mass $m_{\Sigma,1-loop}$. For demonstration purposes, we also compare the plots for tree-level and loop-corrected masses.

Figs. 8a and 8b show the same parameter configuration with different horizontal axes, first b_4 plotted against $m_{\Sigma,tree}$ and then b_4 plotted against $m_{\Sigma,1-loop}$. The loop corrections are not constant inside the parameter space, which explains the non-equidistant parameter points. In Fig. 8b, all the tree-level masses below 120 GeV seem to get shifted to ≈ 120 GeV and higher, which creates the squashed look of the points. For the other a_2 values, the mass shiftings are analogue. Also, when looking at for example Fig. 8b, the tree-level bound for $V_H < V_\Sigma$ with loop-mass smaller than 127 GeV seems to not hold anymore.

In Fig. 8b the "Metastable + H survives" points which correspond to the Σ vacuum are also shown. Here, just like discussed before, the v_0 value is not exactly zero, which can be traced back to numerical issues. Fig. 8b also has the same parameter configurations as the plots [29] and, just like before, the parameter space can be expanded to a broader area, since they only included points between the lower bound $V_H < V_\Sigma$ and the upper bound of the existence of the Σ vacuum as interesting points for their two-step EWSB phase transition theory. In future works, when it's needed, this can be easily investigated in higher resolutions, but for now this should suffice.

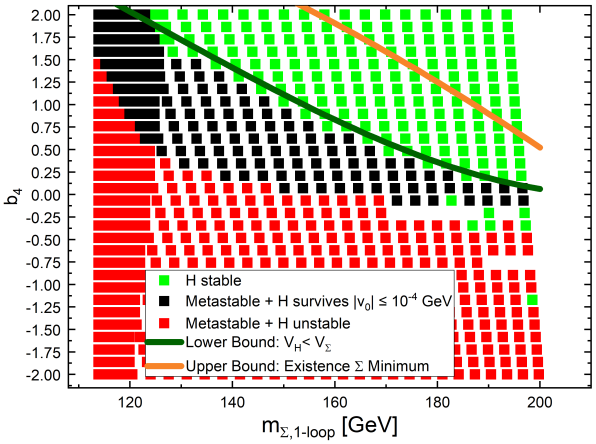
In general, the expectations are well met in Fig. 8, with small shifts of deep (h, σ) configurations towards higher b_4, which can be contributed to the 1-loop corrections. Also, the loop corrections in the cases $a_2 = 1.5$ and $a_2 = 2$ seem to fit well to the tree-level expectations. The loop corrections there seem to yield more stable configurations as expected in tree-level. For greater a_2 values, the stable points seem to move towards a higher mass and b_4, which is also predicted in the tree-level relations.

We also see here, that for a small amount of points with high mass, some stable points exist even for $b_4 < 0$. This is caused by the same problem with the DSB scale transformation as already discussed before. The blank spots in the plots are caused by problems in `SPheno` or `CosmoTransitions`.

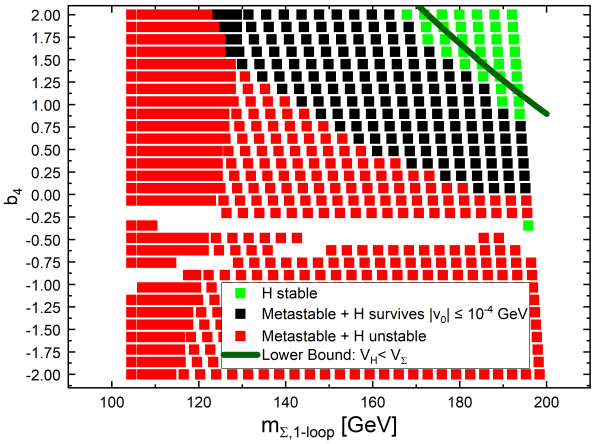


(a) $a_2 = 1.07$, tree-level mass

(b) $a_2 = 1.07$, loop corrected mass



(c) $a_2 = 1.5$, loop corrected mass



(d) $a_2 = 2$, loop corrected mass

Fig. 8: The EWSB Vacuum H stabilities for a_2 of Eq. 22 fixed at $a_2 = 1.07$, $a_2 = 1.5$ and $a_2 = 2$ are plotted with respect to the parameters $m_{\Sigma,tree}$ or $m_{\Sigma,1-loop}$ and b_4. The upper and lower bounds of Eqs. 27 and 30 are also depicted. Since a_2 is positive Eq. 31 isn't applicable in these parameter configurations. Glancing at (a) and (b) reveals, that due to the non-constant loop shift inside the parameter space, the points in (b) are not equidistant anymore. All the small tree-level masses seem to shift above ≥ 105 GeV and create the squashed look at the left edge. The tree-level bound for $V_H < V_\Sigma$ is not perfectly reflected in the plots since there are some deeper points than H even above the bound and the loop corrections for small masses seem to shift the metastable points far up. In these plots, like in Fig. 6, the Σ vacuum configurations have a small v_0 value, which could be attributed to numerical issues. In (c) and (d) there are some blank spots in the plot, which are caused by problems in `SPheno` or `CosmoTransitions`.

4.1.3 Charge Breaking

If we want to check **C**harge **B**reaking (**CB**) minima in the model, we need to allow VEVs for the charged scalar fields. With the 2-dimensional representations (see [28]) of the Higgs doublet and the real triplet from Eqs. 9 and 20, we now assign a VEV to every scalar field

$$\phi = \frac{1}{\sqrt{2}} \begin{pmatrix} v_p + h_p + i\chi_p \\ v_0 + h_0 + i\chi_0 \end{pmatrix} \quad \text{and} \quad \Sigma = \frac{1}{2} \begin{pmatrix} x_0 + \sigma^0 & x_p + \sigma^+ + i\psi^+ \\ x_m + \sigma^- + i\psi^- & -x_0 - \sigma^0 \end{pmatrix}. \tag{32}$$

x_0 and x_p denote the VEVs of the neutral and charged triplet fields. Since $\Sigma^+ = (\Sigma^-)^*$ it follows $x_p = (x_m)^*$.

Now we use the SU(2) gauge freedom to eliminate one charged VEV, namely v_p.

If we write $\langle\phi\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} v_p \\ v_0 \end{pmatrix}$, i.e. the vacuum expectation value is non-vanishing in both components of the doublet, we can now apply a general SU(2) transformation to eliminate the upper component. The matrices in a representation of SU(2) are defined [10] by unitarity $U^\dagger U = 1$ and $\det(U) = 1$ and in two dimensions can be written as [30]

$$U = \begin{pmatrix} z_1 & z_2 \\ -\overline{z}_2 & \overline{z}_1 \end{pmatrix}, \quad |z_1|^2 + |z_2|^2 = 1, \quad z_1, z_2 \in \mathbb{C}. \tag{33}$$

We can now fix the gauge in any desired way, since this doesn't alter the underlying physics and thus use SU(2) transformations to demand $U\langle\phi\rangle = \begin{pmatrix} 0 \\ v\,(v_0, v_p) \end{pmatrix}$.

It's easy to see that the matrix U can be chosen to have the form $U = \begin{pmatrix} z_1 & -\frac{z_1 v_p}{v_0} \\ \frac{z_1 v_p}{v_0} & \overline{z}_1 \end{pmatrix}$ and the condition in Eq. 33 yields $|z_1|^2 + \left(\frac{v_p}{v_0}\right)^2 |z_1|^2 = 1$ which fixes z_1 with

$$|z_1|^2 = \frac{1}{1 + \left(\frac{v_p}{v_0}\right)^2}. \tag{34}$$

Now we use the gauge invariance, that $U\langle\phi\rangle = \langle\phi\rangle$ and with that $v_p = 0$ follows. We now found a SU(2) gauge where we eliminated the charge breaking of the charged Standard Model Higgs. With this, we used two of the three real degrees of freedom of the SU(2).

Using $v_p = 0$ in Eq. 34 we get the constraint $|z_1|^2 = 1$ which is equivalent to $z_1\overline{z}_1 = |z_1|e^{i\theta} \cdot |z_1|e^{-i\theta} = 1$ and therefore we get $z_1 = e^{i\theta}$ with θ as an arbitrary phase. This is the remaining degree of freedom we have in the SU(2) matrix Eq. 33, since due to $v_p = 0$ the

complex number z_2 is equal to 0.

The matrix in Eq. 33 now has an extremely simple diagonal form and applying it to Σ, which transforms under SU(2) as $U\Sigma U^\dagger$, yields

$$U\Sigma U^\dagger = \frac{1}{2}\begin{pmatrix} x_0 & e^{2i\theta}x_p \\ e^{-2i\theta}x_p^* & -x_0 \end{pmatrix}. \tag{35}$$

Since x_p can in principle be a complex number it has the form $x_p = |x_p|e^{i\chi}$. Glancing at Eq. 35 we immediately see, that we can choose the phase θ to eliminate the phase factor χ in x_p and thus eliminate the imaginary part of x_p.

We are now left with a complex v_0 VEV for the Higgs doublet, where we can also eliminate the imaginary part. Since we used every gauge freedom of SU(2), only the $U(1)_Y$ remains. We have one parameter left, namely the phase factor of the U(1) transformation $e^{iY\theta(x)}$, to choose a convenient gauge. Fortunately, as the Σ field has a vanishing Hypercharge, applying the $U(1)_Y$ transformation doesn't change anything, since the Operator always yields $\Sigma' = e^0 \cdot \Sigma = \Sigma$, because Y Σ is always zero.

As we did before, we can parametrize v_0 in the exponential representation $v_0 = |v_0|e^{i\theta_H}$. Transforming the SU(2) gauged Higgs doublet with $\mathbb{1}_L \times e_Y^{-i\theta_H(x)}$ now eliminates the imaginary degrees of freedom in ϕ.

To conclude, we've basically used all the gauge freedom to simplify the symmetry breaking scheme to its maximum degree and are left with only real valued VEVs for H^0, Σ^0 and Σ^+ (or Σ^-). Since both Higgs fields are singlets under the $SU(3)_C$ group, there is no freedom we can use in that sector.

The tree-level tadpole equations with this chosen gauge now read

$$\begin{aligned} -\mu^2 v_0 + \lambda v_0^3 + \frac{1}{2}a_2 v_0 x_0^2 + a_2 v_0 x_p^2 &= 0 \\ -\mu_\Sigma^2 x_0 + \frac{1}{2}a_2 v_0^2 x_0 + b_4 x_0^3) + 2b_4 x_0 x_p^2 &= 0 \\ -2\mu_\Sigma^2 x_p + a_2 v_0^2 x_p + 2b_4 x_0^2 x_p + 4b_4 x_p^3 &= 0 \end{aligned} \tag{36}$$

where the VEVs v_0, x_0 and x_p are all real valued.

4.1.3.1 SARAH model file

To introduce the just derived VEV configurations in the TSMZ2 SARAH model file, it's necessary to define the relevant symmetry breaking in the main "SM+Triplet-Real-Z2.m"

file.

```
DEFINITION[EWSB][VEVs]=
{
{H0, {v, 1/Sqrt[2]}, {0,0}, {Ah, \[ImaginaryI]/Sqrt[2]}, {phiH, 1/Sqrt
    [2]}},
{Hp, {0, 0}, {0,0}, {Ahp, \[ImaginaryI]/Sqrt[2]},{phiHp, 1/Sqrt[2]}},
{T0, {vT, 1/Sqrt[2]}, {0, 0}, {phiT, 1/Sqrt[2]}},
{Tm, {vTm,1/Sqrt[2]}, {0, \[ImaginaryI]/Sqrt[2]}, {ATm, \[ImaginaryI]/
    Sqrt[2]}, {phiTm, 1/Sqrt[2]}}
};
```

Listing 2: Definition of the VEVs with charge breaking in the TSMZ2 model file

Listing 2 basically translates the charge breaking conditions from section 4.1.3 into the `SARAH` Mathematica syntax.

Since ϕ^+ (Hp in Listing 2) is a complex field, it needs to be split up in real and imaginary part, because of the mixing of the charged fields.

```
DEFINITION[EWSB][MatterSector]=
{
{{phiH, phiHp, phiT, phiTm},{hh,ZH}},
{{Ah,Ahp,ATm},{Hpm,ZP}},
...
};
```

Listing 3: Higgs sector mixings for charge breaking in the TSMZ2 model

Listing 3 shows the mixing of the scalar matter sector with the mixing matrix denoted as ZH for the scalar and ZP for the pseudoscalar part.

When charge is broken, all vector bosons of the electroweak gauge group can mix with each other:

```
DEFINITION[EWSB][GaugeSector] =
{
{{VB,VWB[3],VWB[1],VWB[2]},{VP,VZ,VWp,VWm},ZZ}
};
```

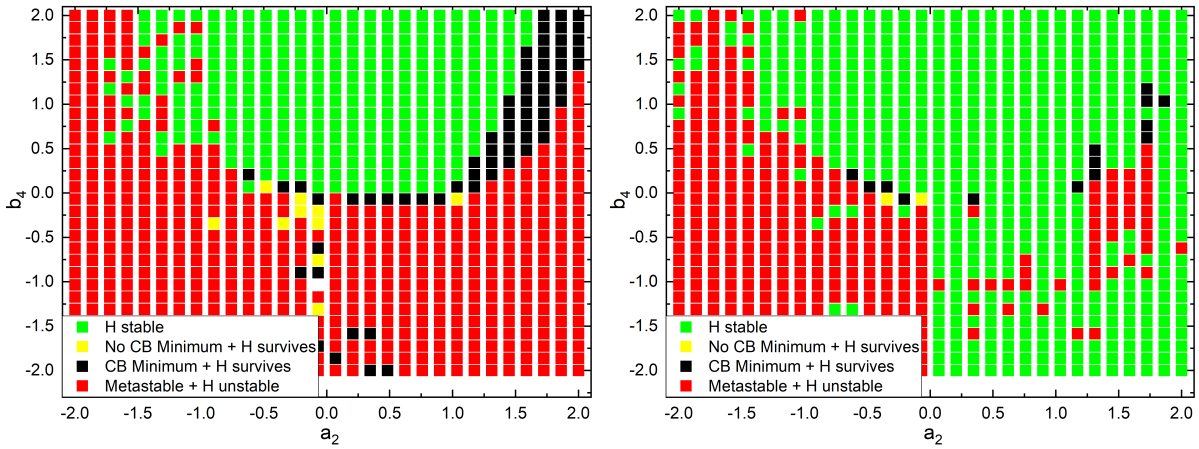
Listing 4: Gauge sector mixings for charge breaking in the TSMZ2 model

Because of the charge breaking, the vector bosons denoted as VP, VZ, VWp and VWm (the names are arbitrary and are not relating to the Standard Model names) do not have a charge in the new eigenbasis, which is a fact that needs to be considered in the "particles.m" file. With only these changes, and some further definitions of dummy variables due

to the tadpole solution problem discussed in section 4.1.2.1, a valid `VevaciousPlusPlus` file can be generated by `SARAH`. Of course, like before with neutral symmetry breaking, the `SPheno` spectrum file will be generated by a model file with only the EWSB with ϕ^0 as we want the EWSB vacuum H as starting point.

4.1.3.2 `VevaciousPlusPlus` results

Here, just like in section 4.1.2.2, the vacuum stability with respect to the parameter space with charge breaking will be analyzed with `VevaciousPlusPlus`. The parameter scope is chosen to be the same as in Figs. 6 and 8 to be able to compare the difference when allowing charge breaking minima to be present.



(a) `PHCpack` - blackbox solver (b) `HOM4PS2` - polyhedral homotopy continuation

Fig. 9: These plots show the different results yielded by `PHCpack` and `HOM4PS2`. A mass of $m_{\Sigma,tree} = 150$ GeV is chosen, but the erroneous results from `HOM4PS2` also emerge in every other parameter configuration. `HOM4PS2` doesn't seem to find deeper points than the EWSB vacuum even for $b_4 < 0$, which doesn't make sense at all, since the potential is unbounded from below. `PHCpack` doesn't seem to have problems and reveals the expected vacuum structure. For this reason, `PHCpack` is the preferred choice for further vacuum structure investigations.

Before presenting the results, an important observation has to be discussed. Trying both `HOM4PS2` (with polyhedral **and** linear homotopy continuation as setting) and `PHCpack`, which both calculate the tree-level minima, revealed that `HOM4PS2` wasn't able to find all the deeper vacuum configurations, while `PHCpack` was able to. Of course, to determine whether the parsing, or the input file generation by V++ or the algorithms of `HOM4PS2` were the problem, this was tested by completely rewriting the parsing algorithms into the RegEx style and investigating problematic parameter configurations outside of V++.

This procedure exposed, that `HOM4PS2` indeed isn't able to find all minimum configurations

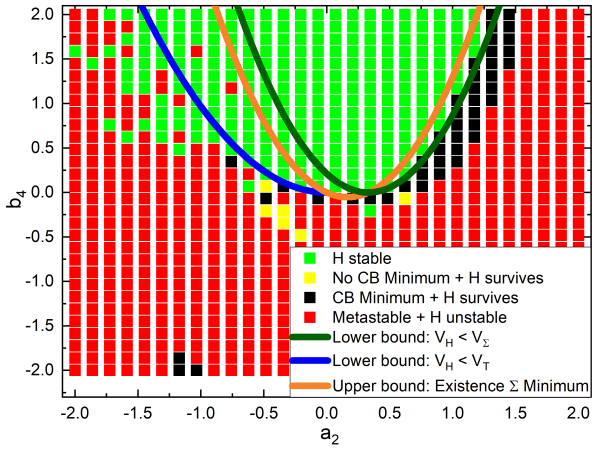
for this charge breaking potential structure. The further solutions PHCpack found are certainly field value configurations corresponding to an extremum. This was confirmed by putting the field solutions into the tadpole equations. Since the blackbox solver of PHCpack always choses the "best" way to solve a given system it may be the case here, that it chooses a different way of solving this system as HOM4PS2 is able to.

For negative b_4, in a very wide range, HOM4PS2 declared the EWSB vacuum as stable, which isn't expected at all. In Fig. 9 these plots are side by side for a tree-level mass of $m_{\Sigma,tree} = 150$ GeV. The fact that HOM4PS2 doesn't find the deeper points for both polyhedral and linear homotopy continuation for negative b_4 is a huge problem, since it isn't reliable anymore for other studies. For non-charge breaking by the way this didn't happen, HOM4PS2 and PHCpack revealed the same vacuum stabilities consistently. Because of that, for every further analysis PHCpack will be used as homotopy continuation program, since we can't be sure if HOM4PS2 works trustfully in every case.

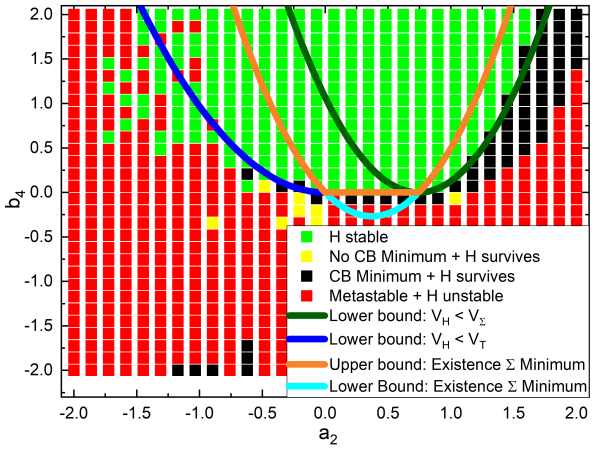
In Fig. 10 the vacuum stability of the TSMZ2 model for charge breaking with different constant $m_{\Sigma,tree}$ is shown. The plots' structures look similar to these in Fig. 6 from before, with the exception that almost all the metastable points where H survives, which probably related to the Σ vacuum before, are now CB minima. This means, that there's a deeper point than the Σ vacuum where the charge is broken, but the tunneling time exceeds the age of the universe.

Comparing the parameter ranges for stability of the EWSB vacuum from Fig. 10 with Fig. 6, we see that the "Metastable + H survives" area in $a_2 > 0$ and $b_4 > 0$ seems to get smaller. This makes sense, because on the edges, there seem to be charge breaking minima which are so deep, that the vacuum tunnels faster than the time of the universe into this new configuration, but when no charge breaking possibilities are included, these points don't exist.

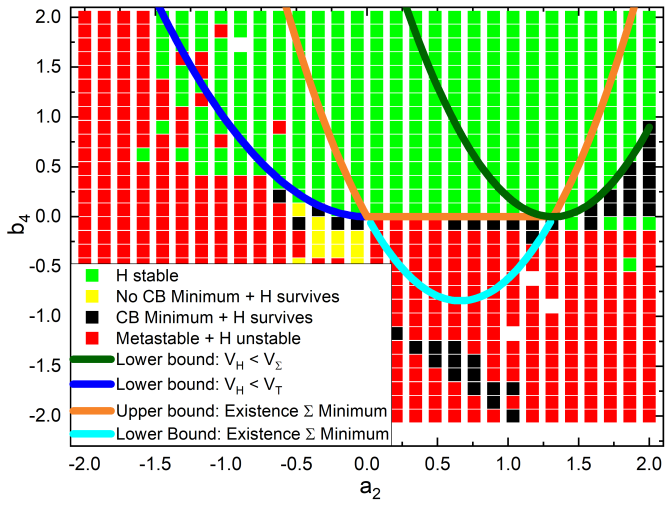
For every mass depicted in the plots the breadths of "Metastable (CB or no CB) + H survives" stripes decreases by about one b_4 stepsize when compared with the only neutral symmetry breaking scenario. Also, for negative b_4 we experience similar problems like before, because there are some stable points and "Metastable + H survives" with either CB or non CB configurations. These are not trustworthy points, since the EWSB must be unstable due to the potential being unbounded from below.



(a) $m_{\Sigma,tree} = 100\,\text{GeV}$



(b) $m_{\Sigma,tree} = 150\,\text{GeV}$



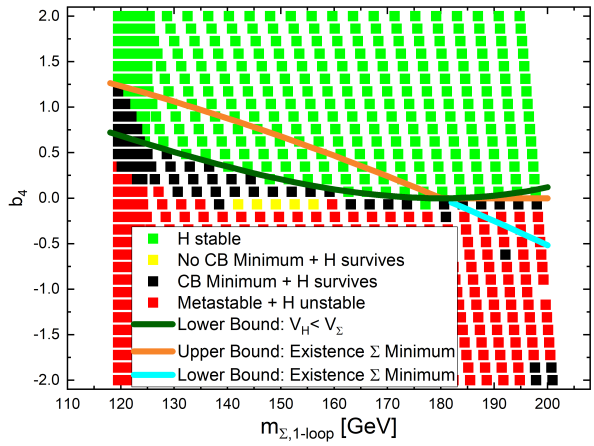
(c) $m_{\Sigma,tree} = 200\,\text{GeV}$

Fig. 10: EWSB Vacuum H stability for tree-level triplet masses of $m_{\Sigma,tree} = 100\,\text{GeV}$ (a), $m_{\Sigma,tree} = 150\,\text{GeV}$ (b) and $m_{\Sigma,tree} = 200\,\text{GeV}$ (c) plotted with respect to the parameters a_2 and b_4 of the Potential Eq. 22 with the possibility of charge breaking minima. The upper and lower bounds of Eqs. 27, 30 and 31 are also depicted in the plots. The derived tree-level bounds for the vacuum stability fit well into the stability of the 1-loop potential calculated by `VevaciousPlusPlus`. Ergo the loop contributions seem to be small for new vacua or VEV shifts. In the bounds of the Σ vacuum, the "Metastable + H survives" points are mostly charge breaking minima, which means that there's a deeper CB minimum than the Σ vacuum. For higher masses, the stability volume in the parameter space gets broader, and as expected, for $b_4 < 0$ there's almost always a deeper vacuum point due to the concave shape, i.e. H, when chosen as a starting point, is metastable. Sometimes the H vacuum survives even for negative b_4 which probably are caused by numerical issues again.

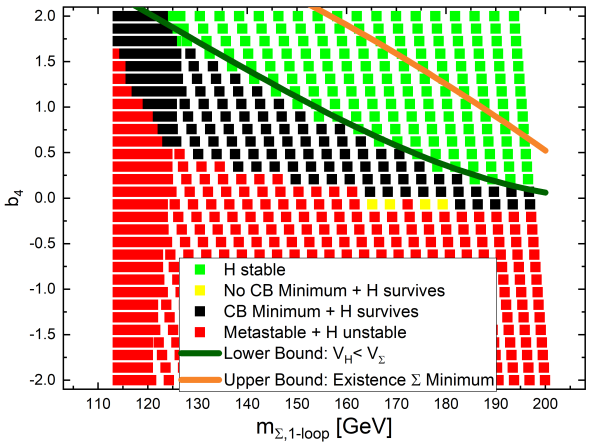
Fig. 11 concludes mostly the same arguments and results in the sense of tree-level bounds as those discussed in section 4.1.2.2 for the case of neutral symmetry breaking, and thus the repeated discussion of all the details which are already given is redundant and therefore omitted. The only difference is the presence of CB minima and the smaller breadths of "Metastable + H survives" areas, which occur due to the possibility of deeper CB minima, which weren't allowed before.

Regarding all the CB plots, it also looks like there exist less stable points at $b_4 < 0$ than in the neutral case. This makes sense, since the potential with CB has more possible directions, which means, that the gradient descent method of `MINUIT` can propagate towards a deeper CB minimum instead of rolling back to the DSB minimum. Although there are less stable points for negative b_4 there seem to exist more metastable configurations, where the tunneling time is high with $b_4 < 0$, when comparing to the neutral symmetry breaking case. It seems, that the gradient descent method seems to stop at some minimum although the potential should be unbounded from below. These points do also differ for different MINUIT settings, so we can again conclude, that the numerical methods had problems with the potential shape.

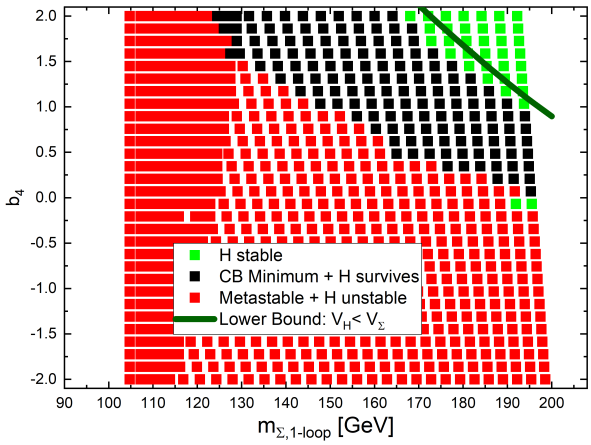
It seems that the Σ vacuum isn't the deepest point in the parameter configurations in Fig. 6, but it should still exist, since without charge breaking (which just means, that the VEVs of charge breaking fields are **set** as exactly zero) it was the deepest state. This needs further work to be done to check the tunneling times from Σ to the other CB vacuums to investigate the potential structure and phase transitions for these "critical" parameter configurations in order to broaden up the parameter space of the derived tree-level relations, which are also given in [28] and [29].



(a) $a_2 = 1.07$



(b) $a_2 = 1.5$



(c) $a_2 = 2$

Fig. 11: The EWSB Vacuum H stabilities for a_2 of Eq. 22 fixed at (a) $a_2 = 1.07$, (b) $a_2 = 1.5$ and (c) $a_2 = 2.0$ are plotted with respect to the parameters $m_{\Sigma,loop}$ and b_4 (see Eq. 22) with the possibility of charge breaking minima. The upper and lower bounds of Eqs. 27, 30 and 31 are also depicted. The derived tree-level bounds for the vacuum stability fit well into the stability of the 1-loop potential calculated by `VevaciousPlusPlus`. One can see, that the vacuum stability gets shifted to higher masses for greater a_2, which is obviously related to the tree-level relations. In (b) the bounds for the conditions Eqs. 30 and 31 are so high, that in the whole parameter space with $b_4 > 0$ a Σ vacuum exists. The Σ vacuum doesn't seem to be the deepest state, since all points with positive b_4 seem to yield a CB minimum instead of a Σ vacuum as the deepest configuration.

4.2 Constrained Minimal Supersymmetric Standard Model

Before stepping into the description of the model details of the **C**onstrained **M**inimal **S**upersymmetric **S**tandard **M**odel (**CMSSM**) let's at first summarize why the extension with **SU**per**SY**mmetry (**SUSY**) is so appealing to physicists. Supersymmetric theories are able to explain many of the unanswered problems which arise in the Standard Model. This includes the hierarchy problem, naturalness and unification of the gauge couplings at the **G**rand **U**nified **T**heory (**GUT**) scale. Furthermore, for the scenario of R-Parity conservation, there's a dark matter candidate as the **L**ightest **S**upersymmetric **P**article (**LSP**) (see [31]).

The Coleman-Mandula theorem, formulated in the 1960s, states that for a 4-dimensional interacting quantum field theory with a mass gap and a discrete particle spectrum, all the additional charges corresponding to the extended symmetry group are Lorentz scalars. This means all the transformations which include a Poincaré transformation and an "internal symmetry group of transformations must be a trivial tensor product of the two groups" [32]. The only extension of the spacetime symmetry, as Haag, Łopuszański and Sohnius showed, can be done by incorporating fermionic charges Q_α to the Poincaré algebra (compare [32]).

These fermionic charges map bosons to fermions and vice versa, which means that bosons and fermions are related by a symmetry transformation! Setting up a supersymmetric Lagrangian with the fields ordered in supermultiplets and imposing the right symmetry breaking mechanisms to match the non-observation of supersymmetry traces is a huge task and all these details are omitted in this thesis but the most important results to understand the CMSSM are summarized in the model specifications, section 4.2.1.

The MSSM itself has numerous free parameters [31] with huge parameter ranges and thus an enormous parameter volume. This would be nearly impossible to scan with a single computer and with many scalar fields acquiring VEVs. The CMSSM, after imposing several supergravity motivated assumptions, only has 5 parameters at the unification scale M_X, see section 4.2.1. Due to the attractiveness of this theory, many global fits to constrain the parameter space even more have been done, like in [31], [33] and [34]. In this thesis, the best CMSSM global fit parameter configurations which can be found in these papers will be checked for CCB minima and vacuum stability.

4.2.1 Model specifications

The paper by G. L. Kane et al. [35] was the first to name the CMSSM, although it is based on many earlier works with supergravity [34], which are very similar to the setup of the CMSSM.

The Minimal Supersymmetric Standard Model is constructed from the same gauge group $SU(3)_C \times SU(2)_L \times U(1)_Y$ as the Standard Model. Due to supersymmetry every SM particle gets a superpartner, as summarized in Table 3.

Field content of the MSSM						
Super-multiplets	Super-field	Bosonic fields	Fermionic partners	SU(3)	SU(2)	U(1)
gluon/gluino	$\widehat{V}_8$	g	$\widetilde{g}$	8	1	0
gauge boson/	$\widehat{V}$	$W^{\pm}$, W^0	$\widetilde{W}^{\pm}$, $\widetilde{W}^0$	1	3	0
gaugino	$\widehat{V}'$	B	$\widetilde{B}$	1	1	0
slepton/	$\widehat{L}$	$(\widetilde{\nu}_L, \widetilde{e}_L^-)$	$(\nu, e^-)_L$	1	2	-1
lepton	$\widehat{E}^c$	$\widetilde{e}_R^+$	e_L^c	1	1	2
squark/	$\widehat{Q}$	$(\widetilde{u}_L, \widetilde{d}_L)$	$(u, d)_L$	3	2	1/3
quark	$\widehat{U}^c$	$\widetilde{u}_R^*$	u_L^c	$\bar{3}$	1	$-4/3$
	$\widehat{D}^c$	$\widetilde{d}_R^*$	d_L^c	$\bar{3}$	1	2/3
Higgs boson/	$\widehat{H}_d$	$(H_d^0$, $H_d^-)$	$(\widetilde{H}_d^0, \widetilde{H}_d^-)$	1	2	-1
higgsino	$\widehat{H}_u$	$(H_u^+$, $H_u^0)$	$(\widetilde{H}_u^+, \widetilde{H}_u^0)$	1	2	1

Table 3: The field content of the MSSM, the supermultiplets with their quantum numbers in the SM gauge group. In this table only one generation of leptons and quarks is shown, but the other generations aren't different regarding the representations. Credits [32].

The superpartners differ from their SM partner by half a spin, thus fermionic SM particles have a bosonic superpartner and the other way round. Due to this, there are several spin-0 particles, the Sfermions, which can acquire a VEV, together with the Higgs bosons. Because of the facts that squarks carry charge and color, the vacuum configurations where they have a non-vanishing VEV correspond to charge and color breaking minima.

As we can see in Table 3, there's a second Higgs doublet in the field content. This is needed to cancel anomalies which "arise from one-loop VVA and AAA triangle diagrams with three external gauge bosons, and fermions running around the loop" [32]. In the SM only one Higgs doublet is required to cancel them, but with the additional Higgsinos, the constraints change, and a second doublet is needed.

To preserve B-L conservation, which is a severe constraint in the Standard Model, an additional Z_2 symmetry is imposed which is called R-parity. The quantum number R is +1 for every SM particle and -1 for every SM superpartner. Considering this, it is clear that the conservation of R results to the existence of the LSP since it can't decay further. R-parity is thus often assumed in minimal supersymmetric extensions of the Standard Model since the most general renormalizable potential has several R-parity violating (RPV) terms. These coupling terms need to either be very small or vanish by the R-parity symmetry to be compliant with the low energy limits (compare [32]).

In the CMSSM, R-parity conservation is imposed and the Superpotential reads [35]

$$W = h^U_{ij}\hat{Q}_i\hat{H}_u\hat{U}^c_j + h^D_{ij}\hat{Q}_i\hat{H}_d\hat{D}^c_j + h^E_{ij}\hat{L}_i\hat{H}_d\hat{E}^c_j + \mu\hat{H}_d\hat{H}_u, \tag{37}$$

where the fields correspond to the ones given in Table 3. There are of course also soft SUSY breaking terms (given in $\mathcal{L}_{soft}$ in [35]), meaning that they don't cause ultraviolet divergences, but the numerous parameters are constrained heavily by coupling "the MSSM to minimal N = 1 supergravity" [35]. By imposing several assumptions at the unification scale, which basically can be summarized as several masses and couplings meeting at the same value, the parameter set can be reduced to only 5 free parameters at the scale M_X: "the universal scalar mass M_0, the gaugino mass parameter $M_{1/2}$, the universal trilinear coupling A_0, the ratio of the two Higgs vacuum expectation values tan β and the sign of the Higgs/Higgsino mass parameter sign(μ)" [33].

The EWSB is implemented by the two Higgs VEVs [32]

$$\left\langle H^0_d \right\rangle = \frac{v_d}{\sqrt{2}} \text{ and } \left\langle H^0_u \right\rangle = \frac{v_u}{\sqrt{2}}, \tag{38}$$

where these need to satisfy $v_d^2 + v_u^2 = 4m_W^2/g^2 \simeq (246\text{GeV})^2$ to yield the correct low energy behaviors. In the context of two Higgs doublet models, the quantity $tan(\beta) = \frac{v_u}{v_d}$ is often defined as a parameter of the theory and thus is also a parameter of the CMSSM.

The tree-level Higgs potential, which can be extracted from the Superpotential and the $\mathcal{L}_{soft}$ reads [35]

$$V_0 = \left(m^2_{H_d} + \mu^2\right)|H^0_d|^2 + \left(m^2_{H_u} + \mu^2\right)|H^0_u|^2 + B\mu\left(H^0_d H^0_u + \text{h.c.}\right) + \frac{g_1^2 + g_2^2}{8}\left(|H^0_d|^2 - |H^0_u|^2\right)^2. \tag{39}$$

Thus, the tadpole equations for the EWSB with the two Higgs doublets then read

$$\begin{aligned} \mu^2 v_d - B\mu v_u + m^2_{H_d} v_d + \frac{1}{8}\left(g_1^2 v_d^3 - g_1^2 v_d v_u^2 + g_2^2 v_d^3 - g_2^2 v_d v_u^2\right) = 0, \\ \mu^2 v_u - B\mu v_d + m^2_{H_u} v_u + \frac{1}{8}\left(-g_1^2 v_d^2 v_u + g_1^2 v_u^3 - g_2^2 v_d^2 v_u + g_2^2 v_u^3\right) = 0. \end{aligned} \tag{40}$$

4.2.2 Global fits

The CMSSM is being checked under heavy pressure and [34], as the name of the paper "Killing the cMSSM softly" hints the results, excluded the cMSSM at the 90% CL range. This means, that scanning the whole parameter volume, as it is already extremely big, would be wasted time. For that reason, the best global fit points from the GAMBIT collaboration [31] and C. Han et al. [33] are investigated for the existence of CCB minima

and the corresponding quantum tunneling survival time of the EWSB vacuum.

The typical scanned parameter ranges of the parameter set are [31] [33]

$$M_0 \in [50, 10000]\,,\ \text{GeV}\ M_{1/2} \in [50, 10000]\,,\ \text{GeV}\ A_0 \in [-10, 10]\,,\ \text{TeV}\ tan(\beta) \in [2, 70]\,, \tag{41}$$

where the sign of μ can be either positive or negative.

In [31] the global fit has been done with the GAMBIT [6] (**G**lobal **A**nd **M**odular **B**eyond the-Standard-Model **I**nference **T**ool) framework. For the scan, they include "likelihoods from a number of direct and indirect dark matter searches, a large collection of electroweak precision and flavour observables, direct searches for supersymmetry at LEP and Runs I and II of the LHC, and constraints from Higgs observables" [31] and present different best fit parameter points, where different dark matter annihilation schemes are picked to fit the dark matter relic density. Those schemes are namely A/H-funnel, $\tilde{\tau}$ co-annihilation, $\tilde{t}$ co-annihilation and $\tilde{\chi}_1^{\pm}$ co-annihilation.

The other team C. Han et al. [33] did a similar global fit to the MSSM, where they "construct a likelihood function including the electroweak precision observables, B-physics measurements, LHC Run-1 and -2 data of SUSY direct searches, Planck observation of the dark matter relic density and the combined LUX Run-3 and -4 detection limits." [33] They incorporate Stau co-annihilation, Stop co-annihilation, Focus-Point, A-funnel dark matter models. They also investigate hybrid scenarios of the named models. The exact meaning and theory behind these models is not important in the scope of this thesis and is therefore omitted.

4.2.3 `SARAH` setup

In the `SARAH` 4.13.0 version, and also in some older versions, there exists the model "CCB-MSSM-SfermionVEVs". The VEVs, which are required to be real because else the field variables count would be too high, are set in the "CCB-MSSM-SfermionVEVs.m" file.

```
1 DEFINITION[EWSB][VEVs]=
2   {{SHd0, {vdR, 1/Sqrt[2]}, {sigmad, \[ImaginaryI]/Sqrt[2]},{phid,1/Sqrt
      [2]}},
3    {SHu0, {vuR, 1/Sqrt[2]}, {sigmau, \[ImaginaryI]/Sqrt[2]},{phiu,1/Sqrt
      [2]}},
4    {SeL, {vLL[3], 1/Sqrt[2]}, {sigmaL, \[ImaginaryI]/Sqrt[2]},{phiL,1/
      Sqrt[2]}},
5    {SeR, {vLR[3], 1/Sqrt[2]}, {sigmaR, \[ImaginaryI]/Sqrt[2]},{phiR,1/
      Sqrt[2]}},
6    {SHdm, {0, 1/Sqrt[2]}, {sigmaM, \[ImaginaryI]/Sqrt[2]},{phiM,1/Sqrt
      [2]}},
```

```
7     {SHup, {0, 1/Sqrt[2]}, {sigmaP, \[ImaginaryI]/Sqrt[2]},{phiP,1/Sqrt
          [2]}},
8     {SvL, {vVL[3], 1/Sqrt[2]}, {sigmaV, \[ImaginaryI]/Sqrt[2]},{phiV,1/
          Sqrt[2]}},
9     {SuLr, {vTL[3], 1/Sqrt[2]}, {sigmauLr, \[ImaginaryI]/Sqrt[2]},{phiuLr
          ,1/Sqrt[2]}},
10    {SuRr, {vTR[3], 1/Sqrt[2]}, {sigmauRr, \[ImaginaryI]/Sqrt[2]},{phiuRr
          ,1/Sqrt[2]}},
11    {SuLg, {0, 1/Sqrt[2]}, {sigmauLg, \[ImaginaryI]/Sqrt[2]},{phiuLg,1/
          Sqrt[2]}},
12    {SuRg, {0, 1/Sqrt[2]}, {sigmauRg, \[ImaginaryI]/Sqrt[2]},{phiuRg,1/
          Sqrt[2]}},
13    {SuLb, {0, 1/Sqrt[2]}, {sigmauLb, \[ImaginaryI]/Sqrt[2]},{phiuLb,1/
          Sqrt[2]}},
14    {SuRb, {0, 1/Sqrt[2]}, {sigmauRb, \[ImaginaryI]/Sqrt[2]},{phiuRb,1/
          Sqrt[2]}},
15    {SdLr, {vBL[3], 1/Sqrt[2]}, {sigmadLr, \[ImaginaryI]/Sqrt[2]},{phidLr
          ,1/Sqrt[2]}},
16    {SdRr, {vBR[3], 1/Sqrt[2]}, {sigmadRr, \[ImaginaryI]/Sqrt[2]},{phidRr
          ,1/Sqrt[2]}},
17    {SdLg, {0, 1/Sqrt[2]}, {sigmadLg, \[ImaginaryI]/Sqrt[2]},{phidLg,1/
          Sqrt[2]}},
18    {SdRg, {0, 1/Sqrt[2]}, {sigmadRg, \[ImaginaryI]/Sqrt[2]},{phidRg,1/
          Sqrt[2]}},
19    {SdLb, {0, 1/Sqrt[2]}, {sigmadLb, \[ImaginaryI]/Sqrt[2]},{phidLb,1/
          Sqrt[2]}},
20    {SdRb, {0, 1/Sqrt[2]}, {sigmadRb, \[ImaginaryI]/Sqrt[2]},{phidRb,1/
          Sqrt[2]}}};
```

Listing 5: VEVs with charge breaking in the CCB-MSSM-SfermionVEVs model

The SHd0 and SHu0 denote the neutral components of the down and up Higgs particles, which are needed for the EWSB mechanism as given in Eq. 38. The VEVs for SeL and SeR, the left handed and right handed Slepton superfields, are given to the third generation, namely to the Stau field, that's why the VEVs are named vLL[**3**] and vLR[**3**] respectively.

The VEV for SvL is also being set to the 3rd generation, meaning to the left handed Stau neutrino.

The SuLr and SuRr VEVs relate to the 3rd generation left and right handed "red" colored Stop quarks. Lastly, the SdLr and SdRr VEVs relate to the 3rd generation left and right handed "red" colored Sbottom quarks. The VEVs for the other colored Squarks can be rotated to zero by the $SU(3)_C$ invariance.

Also, when allowing the CCB configurations, all the gauge bosons of the SM need to mix:

```
1 DEFINITION[EWSB][GaugeSector] =
2 {
3   {{VB,VWB[1],VWB[2],VWB[3],VG[1],VG[2],VG[3],VG[4],VG[5],VG[6],VG[7],VG
      [8]},{VP1,VP2,VP3,VP4,VP5,VP6,VP7,VP8,VP9,VP10,VP11,VP12},ZZ}
4 };
```

Listing 6: Gauge sector mixings in the CCB-MSSM-SfermionVEVs model

Since we are requiring real valued VEVs the real components do not mix with the pseudoscalar (i.e. imaginary) components of the fields

```
1 DEFINITION[EWSB][MatterSector]=
2 {     {{phid, phiu, phiL, phiR, phiV, phiP, phiM, phiuLr, phiuLg, phiuLb, phiuRr,
      phiuRg, phiuRb, phidLr, phidLg, phidLb, phidRr, phidRg, phidRb}, {hh, ZH}},
3       {{sigmad, sigmau, sigmaL, sigmaR, sigmaV, sigmaP, sigmaM, sigmauLr,
          sigmauLg, sigmauLb, sigmauRr, sigmauRg, sigmauRb, sigmadLr, sigmadLg,
          sigmadLb, sigmadRr, sigmadRg, sigmadRb}, {Ah, ZA}},
4       ...
5         };
```

Listing 7: Higgs sector mixings in the CCB-MSSM-SfermionVEVs model

Concluding this, the CCB-MSSM-SfermionVEVs model file assigns **real** VEVs to the Stau $\tilde{\tau}$, Stop $\tilde{t}$, Snu $\tilde{\nu_L}$ and Sbottom $\tilde{b}$ fields. Because of the performance upgrade due to the use of `PHCpack` with multi-tasking, the extrema of the 9 tadpole equations can be found in acceptable computation times.

4.2.4 `VevaciousPlusPlus` results

It has to be noted, that not for every parameter set a SLHA input file using `SPheno` could be generated. For these points, `FlexibleSUSY` [19] was used. The problem with `FlexibleSUSY`s' input files is that they do not create TREE and LOOP SLHA blocks, but instead only print the loop corrected mass parameters. This is indeed a problem, since V++ needs to separate the tree and 1-loop corrections and at first the tree-level minima need to be found. The absence of the tree-level values will shift the tree-level minima, as V++ takes the loop corrected mass parameters in the tree-level potential, which may result to erroneous solutions. Because of this, the tree-level $B\mu$ and μ values were calculated by hand since only three SLHA files (two from the GAMBIT fits and one from Han et al.) needed that treatment. The values were then edited in the way `VevaciousPlusPlus` understands them correctly with the conventions given in section 3.1.2.

Furthermore, SPheno has a flag to skip 2-loop Higgs mass calculations in the LesHouches.in file, which FlexibleSUSY doesn't have, thus the μ and $B\mu$ values include more than just the 1-loop corrections. As experience with the MSSM model files showed, small changes in μ and $B\mu$ change the loop corrections to the VEVs tremendously. Because of this, for example taking corrections up to 2-loop for the masses will alter the results, and thus SPheno was the preferred choice to generate spectrum files.

4.2.4.1 GAMBIT fits

GAMBIT	m_0 [GeV]	$m_{1/2}$ [GeV]	$\tan(\beta)$	sgn(μ)	A_0 [GeV]	stability	lifetime [s]	spectrum gen.
A/H-funnel	9136.379	2532.163	49.048	-	9924.435	stable		FlexibleSUSY
$\tilde{t}$ co-a.	4269.402	1266.043	14.857	-	-9965.036	unstable	$4.2 \cdot 10^8$	SPheno
$\tilde{\tau}$ co-a.	1476.893	2422.34	48.594	+	-1227.154	metastable	10^{100}	SPheno
$\tilde{\chi}_1^{\pm}$ co-a.	9000.628	2256.472	49.879	-	9206.079	stable		FlexibleSUSY

Table 4: The V++ results for the best global fit points given by the GAMBIT collaboration [31]. It seems, that the $\tilde{t}$ co-annihilation is not long lived enough, which excludes these points due to a deeper CCB minimum. The lifetime is way too short to sit in the magnitude of the universe's time $\approx 10^{17}$ s [1]. sgn(x) is a function returning the sign of x for x $\in \mathbb{R}$.

Table 4 shows the input parameters together with the results of V++. It seems that all dark matter models, except the stop co-annihilation, seem to be stable or long-lived. We can conclude from this, that analyses with VevaciousPlusPlus are indeed required to check the vacuum stability even for global best fit points since they can be excluded by CCB minima. This is why GAMBIT is planning to include the library of VevaciousPlusPlus into their future scans to check the survival time of these states. Before the implementation of PHCpack, it was not efficiently possible to allow many fields to acquire a VEV and thus, sometimes parameter configurations were declared long lived although they are actually not when including more VEV degrees of freedom!

It may seem strange at first that all the FlexibleSUSY generated outputs yield "stable" parameter configurations, but this is more just a coincidence. Generating for example the GAMBIT Stop case with FlexibleSUSY yields an unstable EWSB vacuum too. It does produce another tunneling time and differing loop corrections to the field values in V++ though, which is caused by the fact that SPheno and FlexibleSUSY don't produce the exact same SLHA output files but differ by small amounts.

4.2.4.2 C. Han et al. fits

For the fits of Han et al. [33] the Focus Point (FP) model couldn't be investigated, because both `SPheno` and `FlexibleSUSY` couldn't produce a spectrum file, which is why this model is left out in Table 5.

Han et al.	m_0 [GeV]	$m_{1/2}$ [GeV]	$\tan(\beta)$	$\mathrm{sgn}(\mu)$	A_0 [GeV]	stability	lifetime [s]	spectrum gen.
AF	8925	2598	51.2	+	9531	stable		`FlexibleSUSY`
$\tilde{t}$ co-a.	4145	1400	5.6	-	-9891	metastable	10^{100}	`SPheno`
$\tilde{\tau}$ co-a.	1026	1221	31.5	+	-3397	metastable	10^{100}	`SPheno`
$\tilde{\tau}$ co-a. 2	728	1110	26.3	+	-2774	metastable	10^{100}	`SPheno`
Hybrid	1872	3199	37.8	-	-4897	metastable	10^{100}	`SPheno`

Table 5: The V++ results for the best global fit points given by Han et al. [33]. All the configurations seem to yield long-lived metastable or stable states.

As we can see here, all the models seem to be stable or metastable and long-lived. This makes them authentic for further analysis, like maybe allowing more degrees of freedom, since the vacuum state seems to be stable towards the VEV setup from section 4.2.3.

5 Conclusion

5.1 Summary

The developments and tools necessary to effectively scan and check parameter volumes of different models with `VevaciousPlusPlus` were presented and discussed. `PHCpack` proved to be a better alternative to `HOM4PS2` in terms of performance with many degrees of freedom. Also, `HOM4PS2` sometimes doesn't find all solutions.

With that in mind, the triplet extension of the Standard Model was investigated in linear steps inside the parameter ranges. The symmetry breaking was analyzed with only neutral scalar fields getting a VEV and afterwards also with all possible charged degrees of freedom. It showed, that the tree-level expectations with just neutral symmetry breaking were also being fulfilled satisfactory in the `VevaciousPlusPlus` results with the 1-loop potential and allowing charge breaking only changed the results by small amounts. The interesting parameter space, where a two-step electroweak symmetry breaking scenario is possible may be broader than discussed in the literature, but this needs further work to be done.

Finally, the stability of the CMSSM EWSB vacuum is tested against charge and color breaking minima. For this, the best global fit points by GAMBIT and Han et al. were chosen as input for `SPheno` and `FlexibleSUSY`. This concluded, that the Stop co- annihilation parameter set from GAMBIT wasn't long lived and thus the point is excluded by an unstable EWSB vacuum.

5.2 Outlook

From the analyses of the models we see, that an investigation of the vacuum structure is an important aspect for future model building. It is necessary to guarantee a long-lived vacuum state for the chosen parameter set and thus an integration of a program like `VevaciousPlusPlus` in global fits is needed.

The GAMBIT collaboration for example is including `VevaciousPlusPlus` in their future scans, which is absolutely needed regarding the fact, that one of their CMSSM best fit parameter set wasn't long lived.

Also, as this thesis proved, allowing charge and color breaking minima is a good way to further constrain different models and with the implementation of `PHCpack` using `VevaciousPlusPlus` this can be done more efficiently than before. The developments in `VevaciousPlusPlus` aren't completely finished though, since there are still things which need to be done and some problems that need to get fixed.

Acknowledgements

At first, I would like to acknowledge my supervisor W. Porod for always helping me when needed. Your advice was always on point and helped me to understand problems I couldn't solve myself. For every question you always knew a suiting paper to address the problem, which often saved a good amount of time.

Of course I am also grateful to my parents who always covered my back and who motivated me to never give up and to value the important things in live. I don't know if I would have been able to come this far without your aid throughout the whole time in university.

Also I thank F. Pfeuffer for being a great friend and I appreciate the discussions we had, whether if it was about physical problems, society or anything other. It was always fun to visit different courses and seminars with you and having a great time.

Thanks to C. Schaeffer for being the best friend I could imagine, we had great times together, fruitful discussions and the closest friendship I ever had. I hope this will last for a long time, even when there's a spacial distance between us.

I appreciate F. Staub for helping me with problems related to SARAH and C. Eliel to discuss VevaciousPlusPlus and its future and developments with me.

It's hard to say if I would have been able to do this all by myself and thus I am grateful for all the support and help I received!

Appendix

A Extension with PHCpack

A.1 Class PHCRunner

The source code of PHCRunner copies many things from the Hom4ps2Runner, which is why not every piece of code is shown here, since it already existed. PHCRunner, just like Hom4ps2Runner, inherits from the functor PolynomialSystemSolver, which contains the structure of the constraint system with its polynomials and a virtual function for the operator(), which takes the system as input and the solved system as output. Furthermore, there are methods defined which apply sign flips to the fields and check if they are a possible solution, since, as experience showed, HOM4PS2 doesn't recognize some solutions which actually are by inverting signs in some fields.

An instance of PHCRunner needs three arguments: the path to the PHC executable *pathToPHC*, the *resolutionSize* and the *taskcount*.

```
1  void PHCRunner::operator()( std::vector< PolynomialConstraint > const&
       systemToSolve, std::vector< std::vector< double > >& systemSolutions
       ) const {
2  std::string PHCInputFileName = pathToPHC + "
       VevaciousHomotopyContinuation.txt";
3  std::string PHCOutputFilename = pathToPHC + "PHCOutput";
4  std::vector< std::string > variableNames( systemToSolve.size(),"" );
5  std::map< std::string, size_t > nameToIndexMap;
6  WritePHCInput( systemToSolve, variableNames, nameToIndexMap,
       PHCInputFileName );
7  int systemReturn(0);
8  std::string systemCommand = "rm " + PHCOutputFilename;
9  struct stat buffer;
10 if(stat(PHCOutputFilename.c_str(), &buffer)==0){
11 systemReturn = system(systemCommand.c_str());
12 }
13 systemCommand.assign(pathToPHC + "phc -b -t" + std::to_string(taskcount)
       + " "); //calls the blackbox solver
14 systemCommand.append( PHCInputFileName );
15 systemCommand.append(" ");
16 systemCommand.append( PHCOutputFilename );
17 systemReturn = system( systemCommand.c_str() );
18 ParsePHCOutput( PHCInputFileName, systemSolutions, variableNames,
       nameToIndexMap, systemToSolve );
19  }
```

Listing 8: The operator() of PHCRunner - shortened version

Listing 8 is a shortened version of the original one, but it contains the important commands to understand the routine. Console outputs or error handling is neglected in this listing.

The input and output files of PHCpack are in the same folder as the phc executable.

A.2 PHC output parsing

In order to parse the solutions *ParsePHCOutput* is called.

```
1 void PHCRunner::ParsePHCOutput(
2   std::string const& PHCInputFileName,
3   std::vector< std::vector< double >>& purelyRealSolutionSets,
4   std::vector< std::string > const& variableNames,
5   std::map< std::string, size_t > const& nameToIndexMap,
6   std::vector< PolynomialConstraint > const& systemToSolve ) const {
```

Listing 9: The ParsePHCOutput method with its parameters

Its input is the *PHCInputFileName*, a vector of vectors consisting of doubles *purelyRealSolutionSets*, *variableNames*, *nameToIndexMap* and *systemToSolve*. When PHCpack is finished, it appends the final solutions to the input file.

```
1 std::ifstream t(PHCInputFileName);
2 std::string container((std::istreambuf_iterator<char>(t)),
3                 std::istreambuf_iterator<char>());
4 t.close();
```

Listing 10: Writing the input file to a string

The *istreambuf_iterator* writes all the content of the PHC input file into the string *container*.

Due to the fact that only fully real solution sets are needed, the parsing can be done in a fast manner using **Reg**ular **ex**pressions (**Regex**).

```
1 std::map<int,std::vector<double>, std::less<int>> solmap;
2 for(auto it = nameToIndexMap.begin();it!= nameToIndexMap.end();it++)  {
3   std::string doublepattern (it->first);
4   doublepattern += "\\s+:\\s+";
5   doublepattern += "(-?[0-9]+.[0-9]+E[+-][0-9]+)\\s+(-?[0-9]+.[0-9]+E
        [+-][0-9]+)";
6   std::regex pattern(doublepattern);
7   std::sregex_iterator next(container.begin(),container.end(),pattern);
8   std::sregex_iterator end;
9   double Re,Im;
10  int step(0);
```

```
11    if(it == nameToIndexMap.begin()) {
12      while (next != end) {
13        std::smatch match = *next;
14        Re = std::stod(match[1]);
15        Im = std::stod(match[2]);
16        if(fabs(Im) < resolutionSize) solmap[step].push_back(Re);
17        ++step;
18        ++next;
19          }
20      }
21    else {
22      for(auto itm=solmap.begin(); itm != solmap.end();) {
23        for(int k=0;k<(itm->first - step);k++) ++next;
24        step = itm->first;
25        std::smatch match = *next;
26        Re = std::stod(match[1]);
27        Im = std::stod(match[2]);
28        if(fabs(Im) < resolutionSize) {
29          solmap[itm->first].push_back(Re);
30          itm++;
31        }
32        else {
33          auto itm2 = itm;
34          itm++;
35          solmap.erase(itm2);
36        }
37      }
38    }
39  }
```

Listing 11: Algorithm to parse the solutions generated by PHCpack

At first, a map named *solmap*, which is linking a vector consisting of doubles to an integer, is defined. The iteration over the *nameToIndexMap* elements ensures to save the field values of the final solution vector in the right order. *it->first* points to the string naming the field variable of the current iteration.

The following string *doublepattern* defines the pattern with which the regex iterator scans the file. *doublepattern* describes a pattern of the field variable name followed by whitespaces, a colon, further whitespaces and two double precision numbers in the scientific notation. The *sregex_iterator*s are defined to run through the whole *container*. After defining doubles for the real (Re) and imaginary (Im) part, a *step* variable is defined, which basically numbers the solution containers.

In the first iteration one has to check all the solutions for real values, which means that the imaginary part needs to be smaller than the given *resolutionSize*. If a solution meets this requirement, *Re* is saved in *solmap* together with *step*. Incrementing *step* and the iterator *next* means to move to the next container.

In all other iterations which are not the first one, we don't need to check all values again since all solutions in one field value set have to be real. That's the point where *solmap* starts to be useful. The *itm* iterator now runs over all solmap elements. At first, we skip all the regex matches, which yielded imaginary solutions before. If the container, which had real solutions for the field variables before, now has a real occurrence again, the value of *Re* is pushed back in the vector. If *Im* is too big, the current item is erased from solmap.

Afterwards, the found real solutionvectors are appended to *purelyRealSolutionSets*.

```
1  if(!(solmap.empty())){
2    for(auto it = solmap.begin(); it !=solmap.end(); it++)
3    {
4      if((it->second).size() == numberOfVariables) {
5        AppendSolutionAndValidSignFlips(it->second, purelyRealSolutionSets
             , systemToSolve, resolutionSize);
6      }
7      else
8      {
9        std::stringstream errorBuilder;
10       errorBuilder << "There seems to be an error, while parsing the
             real solutions. Check the Output of PHCpack, maybe it's empty
             or faulty because of an error." << std::endl;
11       throw std::runtime_error( errorBuilder.str() );
12     }
13   }
14 }
15 else {
16   std::stringstream errorBuilder;
17   errorBuilder << "No real solutions have been found. Check on your
         ResolutionSize or your input system." << std::endl;
18   throw std::runtime_error( errorBuilder.str() );
19   }
```

Listing 12: Appending the solutions to *purelyRealSolutionSets*

If solmap isn't empty, which means that we found real solution sets, an iterator runs through all found vectors. These vectors obviously need to fulfill the requirement, that the size of the solution vector is the same as the amount of fields. *AppendSolutionAnd-*

ValidSignFlips tries out sign flips as new solutions and writes the solution into *purelyRealSolutionSets*.

A.3 LagrangianParameterManager

Only the relevant changes in the "SlhaCompatibleWithSarahManager" and "LesHouchesAccordBlockEntryManager" are shown here. All the other small changes in other files, which are needed to include the derived parameters are just one liners and are omitted due to lengthiness.

When the LesHouchesAccordBlockEntryManager reads the ScaleAndBlock file, it checks the DerivedParameters tag and runs ParseDerivedParameters with the content.

```
1        else if( xmlParser.CurrentName() == "DerivedParameters" )
2        {
3         ParseDerivedParameters( xmlParser.TrimmedCurrentBody() );
4        }
```

Listing 13: Checking derived parameters in LesHouchesAccordBlockEntryManager

```
1  inline void LesHouchesAccordBlockEntryManager::ParseDerivedParameters(
       std::string xmlbody){
2     std::vector< std::string > dPLines( LHPC::ParsingUtilities::
          SplitBySubstrings( xmlbody, "\n" ) );
3     std::string trimmedLine( "" );
4     for( std::vector< std::string >::const_iterator dPLine( dPLines.
          begin() ); dPLine != dPLines.end(); ++dPLine ) {
5        trimmedLine.assign(*dPLine);
6        trimmedLine.erase(std::remove_if( trimmedLine.begin(),
             trimmedLine.end(),  ::isspace), trimmedLine.end());
7        if( trimmedLine.empty() ) continue;
8        size_t equalsPos =  trimmedLine.find( '=' ) ;
9        if( !( equalsPos < ( trimmedLine.size() - 1 ) ) )
10          {
11           //Error throwing, omitted here
12          }
13       derivedparameters.push_back(std::make_pair(trimmedLine.substr(0,
             equalsPos), trimmedLine.substr(equalsPos+1)));
14    }
15 }
```

Listing 14: Parsing of the derived parameters in the XML tag

ParseDerivedParameters is defined in the header file of LesHouchesAccordBlockEntryManager and parses every line in the xmlbody. What it basically does is sorting the

left side and right side of the "=" into a pair and pushing it back to the vector of pairs "derivedparameters".

The contents of the vector are now parsed in RegisterDerivedParameters.

```
1  void SlhaCompatibleWithSarahManager::RegisterDerivedParameters(std::
       vector<std::pair<std::string,std::string>> derivedparameters) {
2   for( auto it = derivedparameters.begin(); it != derivedparameters.end()
       ; it++){
3    if(((*it).second).find("IFNONZERO")==std::string::npos) {
4     LesHouchesAccordBlockEntryManager::RegisterNewParameter(
          LesHouchesAccordBlockEntryManager::CreateNewBlockEntry((*it).
          second),(*it).first);
5    }
6   }
7   for( auto it = derivedparameters.begin(); it != derivedparameters.end()
       ; it++) //TwoSource
8   {
9    if(((*it).second).find("IFNONZERO")!=std::string::npos)
10   {
11    std::string parnames(it->second);
12    parnames = parnames.substr(parnames.find('[')+1, (parnames.find(']')
         - parnames.find('[') - 1));
13    std::pair< bool, size_t > ifnonzero1 =
          LesHouchesAccordBlockEntryManager::RegisterParameter(parnames.
          substr(0,parnames.find(',')));
14    std::pair< bool, size_t > ifnonzero2 =
          LesHouchesAccordBlockEntryManager::RegisterParameter(parnames.
          substr(parnames.find(',') + 1));
15    if(ifnonzero1.first && ifnonzero2.first){
16     AddNewDerivedParameter((*it).first, new LhaTwoSourceFunctionoid(
          numberOfDistinctActiveParameters,ifnonzero1.second,ifnonzero2.
          second));
17    }
18    else {
19  //Error throwing, omitted here
20    }
21   }
22  }
23 }
```

Listing 15: Registering the derived parameters

This routine goes through every pair in derivedparameters and evaluates their content. Since in the current state we only have IFNONZERO[parameter1,parameter2] or SLH-

ABlock[number] as derived parameters, it suffices to check whether the right side of "=" contains a IFNONZERO or not. If it does, it first checks, whether the parameter1 and parameter2 are registered parameters in V++. If they are registered, Vevacious will register them as a LhaTwoSourceFunctionoid, which basically just returns the second parameter value if the first is zero, else it will return the first parameter value.

B Performance comparison of PHCpack and Hom4ps2, PolyTesting.cpp

In this section the source code, due to redundancy in a slightly shortened version, of the C++ program used to compare performance of PHCpack and Hom4ps2 is given.

```
1 class field{
2   public:
3   string fname;
4   field(string name) { this->fname = name; }
5   string selfcoupling(int pow){ ... }
6   string quadcoupling(field s, int pow){ ... }
7   string quadcoupling(field f1, int i1 ,field f2, int i2){ ... }
8   string cubiccoupling(field f, int fpow){ ... }
```

Listing 16: PolyTesting class field

The class field consists of its name and the memberfunctions which generate the coupling terms as a string.

```
1 string selfcoupling(int pow){
2 random_device rd;
3 default_random_engine rn(rd());
4 uniform_real_distribution<> dist;
5 stringstream strs;
6 if(pow==3){
7   dist = uniform_real_distribution<>(10,300);
8   strs << ""<< dist(rn) <<" * " << this->fname<<"^"<<pow <<" ";
9   return strs.str();
10  }
11 if(pow==2){
12  dist = uniform_real_distribution<>(300,1000);
13  strs << "+ "<< dist(rn) <<" * " << this->fname<<"^"<<pow <<" ";
14  return strs.str();
15  }
16 if(pow==1){
17  dist = uniform_real_distribution<>(100000,2000000);
18  strs << "+ "<< dist(rn) <<" * " << this->fname <<" ";
```

```
19    return strs.str();
20    }
21  }
```

Listing 17: Field memberfunction selfcoupling

selfcoupling(int pow) yields terms which only consists of the field itself.
With the *random_device* a random number, depending of the power of selfcoupling, is generated and returned as a string. As it generates tadpoles, the power doesn't go higher than 3.

```
1  string quadcoupling(field f1, int i1 ,field f2, int i2){
2  random_device rd;
3  RNG rn(rd());
4  uniform_real_distribution<> dist;
5  bernoulli_distribution randsign(0.5);
6  stringstream strs;
7  if(i1 + i2 ==3){
8    dist = uniform_real_distribution<>(0.005,0.20);
9    strs <<((randsign(rn))?"- ":"+ ") << dist(rn) <<" * " << f1.fname <<((
         i1==2)?("^"+to_string(i1)):"") << " * " <<f2.fname <<((i2==2)?"^"+
         to_string(i2):"")<<" ";
10   return strs.str();
11   }
12 if(i1 + i2==2){
13   dist = uniform_real_distribution<>(0.005,0.20);
14   strs <<((randsign(rn))?"- ":"+ ") << dist(rn) <<" * " << f1.fname << "
          * " <<f2.fname << " * " <<this->fname <<" ";
15   return strs.str();
16   }
17 else {
18   cout << "No Coupling" <<endl;
19   return "";
20   }
21 }
```

Listing 18: Field memberfunction quadcoupling

quadcoupling(field f1, int i1 ,field f2, int i2) yields terms which contains of two or three different fields. when the power of the two input fields is 2, the third field is the field object of the class, let's call it this->field, with the name this->fname. If the powers of f1 and f2 are three, there is only these two, since this relates to potential terms linear in this->field. The other member functions of field, $quadcoupling(fields, intpow)$ and $cubiccoupling(fieldf, intfpow)$ are analogous to $quadcoupling(fieldf1, inti1, fieldf2, inti2)$

and are not discussed here.

```
string writeequationgeneral(vector<field> fvec){
stringstream strs;
vector<field> fp;
bool permutation;
for(auto eq = fvec.begin(); eq!=fvec.end();++eq){
  for(int i=1;i<=3;i++) strs << (*eq).selfcoupling(i); //Selfcoupling
  for(auto f = fvec.begin(); f!=fvec.end();++f) {
    if(f!=eq){
    fp.push_back(*f);
    strs << (*eq).cubiccoupling(*f,1) << (*eq).cubiccoupling(*f,2);
    for(int i=0;i<=2;i++) strs << (*eq).quadcoupling(*f,i);
    for(auto g = fvec.begin(); g!=fvec.end();++g){
      permutation=false;
      if(g!=f \&\& g!=eq){
      strs << (*eq).quadcoupling(*f,2,*g,1);
      for(auto fpi = fp.begin(); fpi!=fp.end();++fpi){ //To prevent
          permutations
        if((*g).fname ==(*fpi).fname) permutation=true;
              }
      if(!permutation) strs << (*eq).quadcoupling(*f,1,*g,1);
      }
      }
    }
          }
  strs <<";\n";
  fp.clear();
  }
return strs.str();
}
```

Listing 19: Field memberfunction writeequationgeneral

writeequationgeneral(*vector* < *field* > *fvec*) loops through the field vector fvec and couples it with every other one using the memberfunctions in each field. It uses iterators which add the terms yielded by the coupling functions to a stringstream strs. The quadcoupling with two other distinct fields needs special treatment, since we want to avoid having redundant terms. It basically fills another field vector with the quadcoupling with two other fields and checks if a permutation of this has occurred, if it does, this term is skipped.

```
int main(){
...
for(int k=0; k<=(ecount-count);k++){
```

```
4       cout << (count+k)<<endl;
5       for(int i=1;i<=(count+k);i++){
6         field f(fvn + to_string(i));
7         fvec.push_back(f);
8         }
9   string equations = writeequationgeneral(fvec);
10  string phcinput = ("./PHCTesting/input" + to_string(count+k) );
11  string h4ps2input = ("./Hom4ps2testing/input" + to_string(count+k) );
12  ofstream phcinputf(phcinput);
13  phcinputf << count+k << " " << count+k << endl;
14  phcinputf << equations;
15  phcinputf.close();
16  ofstream hinputf(h4ps2input);
17  hinputf<< "{\n" <<equations<<"}";
18  hinputf.close();
19  string phcoutput=("./PHCTesting/output"+to_string(count+k) );
20  systemcommand.assign( ("./phc -b -t" + to_string(taskcount) + " " +
        phcinput + " " + phcoutput) );
21  chrono::steady_clock::time_point begin = chrono::steady_clock::now();
22  system(systemcommand.c_str());
23  chrono::steady_clock::time_point end= chrono::steady_clock::now();
24  ofstream ptime("./PHCTesting/timePHC", ios::out | ios::app);
25  ptime << "fieldcount: " << to_string(count+k)<< " t: " << chrono::
        duration_cast<chrono::milliseconds>(end - begin).count() << " ms" <<
        endl;
26  ptime.close();
27  if((k+count)<=7){ //more than 7 fields take too long for Hom4Ps2
28  ...
29  chrono::steady_clock::time_point begin = chrono::steady_clock::now();
30  system(systemcommand.c_str());
31  chrono::steady_clock::time_point end= chrono::steady_clock::now();
32  ofstream htime("./Hom4ps2testing/timeHom4ps2", ios::out | ios::app);
33  htime << "fieldcount: " << to_string(count+k)<< " t: " << chrono::
        duration_cast<chrono::milliseconds>(end - begin).count() << " ms" <<
        endl;
34  htime.close();
35  systemcommand.assign( "cp ./data.roots ./Hom4ps2testing/h4ps2output"+
        to_string(count+k) );
36  system(systemcommand.c_str());
37        }
38     }
```

Listing 20: PolyTesting main() function

The main() method consists of simple console inputs and loops, generating input files for

PHCpack and HOM4PS2 which are then executed with a system() call. In listing 20 only the most important part is shown. The field vector is first filled with a given count of field variables in the console. Then, strings with the names of the input files are generated and ofstreams fill the files with equations, with small adjustments due to the different syntax.

After this, the phc executable, which is in the same directory, is executed with -b as blackbox solver and the multitasking flag. the chrono::steady_clock is used to measure the difference of times which are written into timePHC after the calculation. The same goes for HOM4PS2, the systemcommand is spared here, it's basically just the needed flags and commands. After calculation it saves the duration and the fieldcount into timeHom4ps2 and also copies the found solution to the output folder. HOM4PS2 takes too long for a fieldcount >7 which is why this is not done in this program.

C Example VPPparamScanner config file

This is just an example for a config file setup with VPPparamScanner for a scan with the TSMZ2 model where the tree-level triplet mass is held constant throughout the scan. To be precise there are three scans in the following file: for the triplet masses 100, 150 and 200 GeV.

```
1  <VevaciousPlusPlusMainInput>
2      <InitializationFile>./TSMZ2/InitializationFiles/
           VevaciousPlusPlusDefaultObjectInitialization.xml</
           InitializationFile>
3      <SingleParameterPoint>
4          <RunPointInput>./TSMZ2/SPHOutput/</RunPointInput>
5          <OutputFilename>./TSMZ2/OutputsVpp/</OutputFilename>
6      </SingleParameterPoint>
7  </VevaciousPlusPlusMainInput>
8  <SPheno>
9      <executable>pathtoSPheno/SPhenoTSMZ2</executable>
10     <inputfile>./TSMZ2/LesHouches.in.TSMZ2_low</inputfile>
11 </SPheno>
12 <programspecific>
13     <Results>./TSMZ2/</Results>
14     <delimiter>_</delimiter>
15     <Loop>1</Loop>
16     <VevaciousPlusPlusExec>pathtoVpp/VevaciousPlusPlus.exe</
           VevaciousPlusPlusExec>
17     <inputparameter>
18         <name>LHTInput</name>
19         <block>4</block>
20         <beginval>-4</beginval>
```

```
21          <endval>4</endval>
22          <stepsize>29</stepsize>
23      </inputparameter>
24      <inputparameter>
25          <name>LTInput</name>
26          <block>2</block>
27          <beginval>-8</beginval>
28          <endval>8</endval>
29          <stepsize>29</stepsize>
30      </inputparameter>
31      <constant>
32          <name>LH</name>
33          <!-- EW Lambda -->
34          <value>0.25810699</value>
35      </constant>
36      <inputparameter>
37          <name>MassTrip</name>
38          <values>100, 150, 200</values>
39      </inputparameter>
40      <constant>
41          <name>vH</name>
42          <!-- EW VEV -->
43          <value>246.2196507941</value>
44      </constant>
45      <inputparametercalc>
46          <name>MTInput</name>
47          <block>8</block>
48          <term>2*((MassTrip)^2 - 0.25* LHTInput * (vH)^2)</term>
49      </inputparametercalc>
50      <lhablock>
51          <!-- 1loop TripMass -->
52          <name>LoopTripMass</name>
53          <blockname>MASS</blockname>
54          <block>35</block>
55      </lhablock>
56  </programspecific>
```

Listing 21: A VPPparamScanner config file to scan different triplet masses

All the paths given here are specific to the setup and thus irrelevant. The LHTInput, LTInput and LH are related to the potential parameters by factors of 2 and 4. As we can extract from Listing 21 the MTInput, which is proportional to μ_Σ^2 from the potential of TSMZ2, is calculated from the input parameters. Also, after generating a SLHA file, the 1-loop corrected mass is parsed.

References

[1] B. O'Leary, "VevaciousPlusPlus - overhaul of Vevacious, entirely in C++ (no generated Python or external executables!)." `https://github.com/benoleary/VevaciousPlusPlus`.

[2] J. Verschelde, "Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation," vol. 25, pp. 251–276, 06 1999.

[3] T. L. Lee, T. Y. Li, and C. H. Tsai, "Hom4ps-2.0: a software package for solving polynomial systems by the polyhedral homotopy continuation method," *Computing*, vol. 83, p. 109, Oct 2008.

[4] S. Geisler, "A parameter scanner generating slha files, inputting it to v++ and parsing the output." `https://github.com/SimonGeisler/VPPparamScanner`.

[5] W. Porod, "SPheno, a program for calculating supersymmetric spectra, SUSY particle decays and SUSY particle production at e^+e^- colliders," *Computer Physics Communications*, vol. 153, pp. 275–315, June 2003.

[6] P. Athron, C. Balazs, T. Bringmann, A. Buckley, M. Chrząszcz, J. Conrad, J. M. Cornell, L. A. Dal, H. Dickinson, J. Edsjö, B. Farmer, T. E. Gonzalo, P. Jackson, A. Krislock, A. Kvellestad, J. Lundberg, J. McKay, F. Mahmoudi, G. D. Martinez, A. Putze, A. Raklev, J. Ripken, C. Rogan, A. Saavedra, C. Savage, P. Scott, S.-H. Seo, N. Serra, C. Weniger, M. White, and S. Wild, "GAMBIT: the global and modular beyond-the-standard-model inference tool," *European Physical Journal C*, vol. 77, p. 784, Nov. 2017.

[7] P. W. Higgs, "Broken symmetries and the masses of gauge bosons," *Phys. Rev. Lett.*, vol. 13, pp. 508–509, Oct. 1964.

[8] F. Englert and R. Brout, "Broken symmetry and the mass of gauge vector mesons," *Phys. Rev. Lett.*, vol. 13, pp. 321–323, Aug 1964.

[9] P. D. M. M. Mühlleitner, *Beyond the Standard Model Physics*. Karlsruhe: Karlsruhe Institute of Technology (KIT), 2014.

[10] T. Ohl, *Theoretical Elementary Particle Physics*. Würzburg: Institut für Theoretische Physik und Astrophysik, 2017.

[11] C. Patrignani *et al.*, "Review of Particle Physics," *Chin. Phys.*, vol. C40, no. 10, p. 100001, 2016.

[12] R. N. Mohapatra, *Unification and Supersymmetry: The Frontiers of Quark-Lepton Physics, Third Edition*. Springer, 2002.

[13] M. Sher, "Electroweak Higgs potential and vacuum stability," *Physics Reports*, vol. 179, pp. 273–418, Aug. 1989.

[14] W. Porod and F. Staub, "SPheno 3.1: extensions including flavour, CP-phases and models beyond the MSSM," *Computer Physics Communications*, vol. 183, pp. 2458–2469, Nov. 2012.

[15] F. Staub and W. Porod, "Improved predictions for intermediate and heavy Supersymmetry in the MSSM and beyond," *Eur. Phys. J.*, vol. C77, no. 5, p. 338, 2017.

[16] P. Skands, B. C. Allanach, H. Baer, C. Balazs, G. Belanger, F. Boudjema, A. Djouadi, R. Godbole, J. Guasch, S. Heinemeyer, W. Kilian, J.-L. Kneur, S. Kraml, F. Moortgat, S. Moretti, M. Muhlleitner, W. Porod, A. Pukhov, P. Richardson, S. Schumann, P. Slavich, M. Spira, and G. Weiglein, "SUSY Les Houches Accord: Interfacing SUSY Spectrum Calculators, Decay Packages, and Event Generators," *Journal of High Energy Physics*, vol. 7, p. 036, July 2004.

[17] J. E. Camargo-Molina, B. O'Leary, W. Porod, and F. Staub, "Vevacious: a tool for finding the global minima of one-loop effective potentials with many scalars," *European Physical Journal C*, vol. 73, p. 2588, Oct. 2013.

[18] F. Staub, "Sarah," *ArXiv e-prints*, June 2008.

[19] P. Athron, J.-h. Park, D. Stöckinger, and A. Voigt, "FlexibleSUSY-A spectrum generator generator for supersymmetric models," *Computer Physics Communications*, vol. 190, pp. 139–172, May 2015.

[20] F. James and M. Roos, "Minuit – a system for function minimization and analysis of the parameter errors and correlations," *Computer Physics Communications*, vol. 10, pp. 343–367, Dec. 1975.

[21] iminuit team, "iminuit – a python interface to minuit." `https://github.com/iminuit/iminuit`. Accessed: 2018-03-05.

[22] C. L. Wainwright, "CosmoTransitions: Computing Cosmological Phase Transition Temperatures and Bubble Profiles with Multiple Fields," *Comput. Phys. Commun.*, vol. 183, pp. 2006–2013, 2012.

[23] S. Coleman, "Fate of the false vacuum: Semiclassical theory," *Phys. Rev. D*, vol. 15, pp. 2929–2936, May 1977.

[24] C. G. Callan and S. Coleman, "Fate of the false vacuum. ii. first quantum corrections," *Phys. Rev. D*, vol. 16, pp. 1762–1768, Sep 1977.

[25] bertiniteam, "Bertini 2.0: The redevelopment of bertini in c++.." `https://github.com/bertiniteam/b2`.

[26] M. D. Goodsell, K. Nickel, and F. Staub, "Two-loop Higgs mass calculations in supersymmetric models beyond the MSSM with SARAH and SPheno," *European Physical Journal C*, vol. 75, p. 32, Jan. 2015.

[27] A. Partow, "C++ Mathematical Expression Parsing And Evaluation Library `http://www.partow.net/programming/exprtk/index.html`." `https://github.com/ArashPartow/exprtk`.

[28] P. Fileviez Pérez, H. H. Patel, M. J. Ramsey-Musolf, and K. Wang, "Triplet scalars and dark matter at the LHC," *Physical Review D*, vol. 79, p. 055024, Mar. 2009.

[29] H. H. Patel and M. J. Ramsey-Musolf, "Stepping into electroweak symmetry breaking: Phase transitions and Higgs phenomenology," *Physical Review D*, vol. 88, p. 035013, Aug. 2013.

[30] B. C. Hall, *Lie Groups, Lie Algebras, and Representations*. Springer International Publishing, 2015.

[31] P. Athron, C. Balázs, T. Bringmann, A. Buckley, M. Chrząszcz, J. Conrad, J. M. Cornell, L. A. Dal, J. Edsjö, B. Farmer, P. Jackson, A. Krislock, A. Kvellestad, F. Mahmoudi, G. D. Martinez, A. Putze, A. Raklev, C. Rogan, R. R. de Austri, A. Saavedra, C. Savage, P. Scott, N. Serra, C. Weniger, and M. White, "Global fits of GUT-scale SUSY models with GAMBIT," *European Physical Journal C*, vol. 77, p. 824, Dec. 2017.

[32] H. E. Haber and L. Stephenson Haskins, "Supersymmetric Theory and Models," *ArXiv e-prints*, Dec. 2017.

[33] C. Han, K.-i. Hikasa, L. Wu, J. M. Yang, and Y. Zhang, "Status of CMSSM in light of current LHC Run-2 and LUX data," *Physics Letters B*, vol. 769, pp. 470–476, June 2017.

[34] P. Bechtle, J. E. Camargo-Molina, K. Desch, H. K. Dreiner, M. Hamer, M. Krämer, B. O'Leary, W. Porod, B. Sarrazin, T. Stefaniak, M. Uhlenbrock, and P. Wienemann, "Killing the cMSSM softly," *European Physical Journal C*, vol. 76, p. 96, Feb. 2016.

[35] G. L. Kane, C. Kolda, L. Roszkowski, and J. D. Wells, "Study of constrained minimal supersymmetry," *Phys. Rev. D*, vol. 49, pp. 6173–6210, June 1994.

Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe, die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form in keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Datum: .....

(Unterschrift)