universität
wien

Diplomarbeit

Titel der Diplomarbeit

Lepton mixing modelling
and
numerical methods

angestrebter akademischer Grad

Magister der Naturwissenschaften (Mag. rer. nat.)

Verfasser: Andreas Singraber
Matrikelnummer: 0403346
Studienrichtung: A 411 Diplomstudium Physik
Betreuer: Ao. Univ.-Prof. Dr. Walter Grimus

Wien, am 25. Juni 2010

Danksagung

Zunächst möchte ich mich hier bei allen Personen bedanken, die diese Arbeit erst möglich gemacht haben.

An erster Stelle bedanke ich mich besonders herzlich bei meinem Betreuer Prof. Walter Grimus, der mir stets mit Rat und Tat zur Seite gestanden hat. Mit viel Engagement und Geduld hat er sich all meinen Fragen gestellt, mit mir viele anregende Diskussionen geführt und mich immer ausgezeichnet betreut.

Bedanken möchte ich mich außerdem bei meinen Freunden und Studienkollegen, insbesondere bei Patrick Ludl für die vielen interessanten Gespräche über Physik und allerlei Anderes. Ich wünsche ihm weiterhin viel Erfolg bei seinem Studium.

Meinen Dank möchte ich auch an alle Professoren und Lehrer richten, die mich auf meinem Ausbildungsweg begleitet haben. Besonders hervorzuheben ist hierbei Prof. Martin Neumann, der mich während meines Studiums mit dem nötigen Rüstzeug auf dem Gebiet der Numerik und Programmierung ausgestattet hat.

Von ganzem Herzen möchte ich meinen Eltern dafür danken, dass sie mir dieses Studium ermöglicht haben. Stets habe ich von meiner Familie den nötigen Rückhalt und auch in schwierigen Zeiten viel Unterstützung bekommen.

It doesn't matter how beautiful your theory is, it doesn't matter how smart you are. If it doesn't agree with experiment, it's wrong.

Richard Feynman

Contents

Chapter 1

Introduction

The confrontation between theory and experiment is one of the basic ingredients of the scientific method. It is a thrilling moment for every physicist when theoretical predictions face experimental observations and either consistency or diverging results are brought to light. Todays particle physics offers a wide field of activity with new developments regarding theoretical, experimental and also computational aspects. Especially the lepton sector with the discovery of neutrino oscillations, implications on dark matter and hints of new physics has become a playground for model building over the last decades. Often based on the Standard Model numerous techniques and extensions were investigated, e.g. discrete symmetries, the seesaw mechanism or even grand unification, to find an accurate description of neutrino phenomena. Rising interest in neutrino physics also provides us with frequently renewed measurement data since more and more collaborations are researching solar, atmospheric or reactor neutrinos. This allows to test if model predictions are compatible with combined experiment data.

Referring to the introductory statement the present work describes methods to compare theory with experiment even in the presence of involved dependencies on model parameters and calculated observables therefrom. The analysis relies on numerical techniques which will be tested on a model for tri-bimaximal mixing and its modifications. This thesis is organized as follows. Chapter 2 gives an overview of the Standard Model and its basic extensions in the lepton sector, e.g. Majorana neutrinos and the seesaw mechanism. Chapter 3 then focuses on neutrino phenomena like neutrino oscillations and provides the current data for mixing angles and mass squared differences. Chapter 4 introduces a figure of merit function which allows to quantify the agreement between model predictions

and measurement data. The best possible match is obtained by minimization of the function. Depending on the model this task may only be accomplished by application of numerical tools. Thus, the Nelder-Mead method, a procedure for non-linear optimization problems applicable also in higher dimensions, is presented and its *Fortran* implementation detailed. In Chapter 5 two models extending the Standard Model in the lepton sector with additional right-handed neutrinos are reviewed. First, the model for tri-bimaximal mixing found by Grimus and Lavoura [1] is discussed, followed by a modification thereof employing CP invariance and additional spontaneous symmetry breaking. These two models are put to test with the described numerical techniques and the results are presented in Chapter 6.

A major aspect of this work is to develop the program which includes the minimization via the Nelder-Mead procedure. The source code is provided in the appendix and allows future application to more elaborate models.

Chapter 2

The Standard Model and its extensions in the lepton sector

During the last decades there has been an enormous progress in both experimental and theoretical particle physics. One of the greatest achievements was the discovery of what we call today the Standard Model of elementary particle physics, which is still a very accurate description of particles and their interactions. But it would not be physics if there were not some details missing and indeed experiments gave results that lack an explanation by the Standard Model (SM). Neutrinos in the SM are described as massless particles, a fact that can not be brought into accordance with experimental data of neutrino oscillations. Today there is a great variety of models, many of them based on the SM, which imply new features like massive neutrinos. In this chapter we will first recapitulate the most important characteristics of the SM (see also [2, 3]) and in the second part discuss the basic extensions in the lepton sector.

2.1 The Standard Model

The SM is a quantum field theory based on the non-abelian gauge group $SU(3)_C \times SU(2)_I \times U(1)_Y$. The three Lie groups in the direct product represent color charge, weak isospin and weak hypercharge respectively. Because this work deals with models concerning only the lepton sector, the $SU(3)_C$ gauge invariance is not discussed in detail. Unlike quarks, leptons do not carry any color charge and

therefore are not affected by strong interaction. We shall now concentrate on the remaining symmetries and particles.

2.1.1 The GWS model of electroweak interaction

Ignoring the $SU(3)_C$ color symmetry one gets a model for electroweak interactions based on the remaining gauge group $SU(2)_I \times U(1)_Y$ first discovered by Glashow, Weinberg and Salam [4, 5, 6]. It contains spin 0, $\frac{1}{2}$ and 1 fields organized in different multiplets and generations as follows:

- 3 left-handed quark[1] doublets (spin $\frac{1}{2}$)

$$Q_{0L}^{(d)} = \begin{pmatrix} u_{0L} \\ d_{0L} \end{pmatrix} \quad , \quad Q_{0L}^{(s)} = \begin{pmatrix} c_{0L} \\ s_{0L} \end{pmatrix} \quad , \quad Q_{0L}^{(b)} = \begin{pmatrix} t_{0L} \\ b_{0L} \end{pmatrix}, \tag{2.1}$$

- 3 left-handed lepton doublets (spin $\frac{1}{2}$)

$$L_L^{(e)} = \begin{pmatrix} \nu_{eL} \\ e_L \end{pmatrix} \quad , \quad L_L^{(\mu)} = \begin{pmatrix} \nu_{\mu L} \\ \mu_L \end{pmatrix} \quad , \quad L_L^{(\tau)} = \begin{pmatrix} \nu_{\tau L} \\ \tau_L \end{pmatrix}, \tag{2.2}$$

- 6 right-handed quark singlets (spin $\frac{1}{2}$)

$$u_{0R} \ , \ c_{0R} \ , \ t_{0R} \ , \ d_{0R} \ , \ s_{0R} \ , \ b_{0R}, \tag{2.3}$$

- 3 right-handed lepton singlets (spin $\frac{1}{2}$, note the missing right-handed neutrino fields)

$$e_R \ , \ \mu_R \ , \ \tau_R, \tag{2.4}$$

- 4 gauge boson fields corresponding to the generators of $SU(2)_I$ and $U(1)_Y$ respectively (spin 1)

$$W_\mu^a \ , \ a = 1, 2, 3 \quad \text{and} \quad B_\mu, \tag{2.5}$$

- 1 scalar Higgs doublet (spin 0)

$$\Phi = \begin{pmatrix} \phi^+ \\ \phi^0 \end{pmatrix}. \tag{2.6}$$

[1]The subscript "0" indicates that the quark fields listed here are preliminary fields, which will be transformed into physical ones (mass eigenfields) in chapter 2.1.2.

These fields contribute to the total Lagrangian of the GWS model, which is invariant under the following $SU(2)_I$ gauge transformation.

$$SU(2)_I : \quad \begin{cases} \Psi & \rightarrow & U\Psi \quad \text{for all doublets} \\ \Psi & \rightarrow & \Psi \quad \text{for all singlets} \\ W_\mu^a \frac{\tau^a}{2} & \rightarrow & U W_\mu^a \frac{\tau^a}{2} U^{-1} + \frac{i}{g} \left(\partial_\mu U\right) U^{-1} \\ B_\mu & \rightarrow & B_\mu \end{cases} \tag{2.7}$$

This transformation, where $U \in SU(2)_I$, acts on all doublets and the three gauge bosons corresponding to the weak isospin. The second gauge group $U(1)_Y$ gives multiplication with different phase factors.

$$U(1)_Y : \quad \begin{cases} \Psi & \rightarrow & e^{-i\alpha\frac{1}{2}Y}\Psi \\ W_\mu^a & \rightarrow & W_\mu^a \\ B_\mu & \rightarrow & B_\mu + \frac{1}{g'}\partial_\mu\alpha \end{cases} \tag{2.8}$$

The real number Y denotes the weak hypercharge, which can be chosen independently to a large degree for every field. In the GWS model the weak hypercharge is correlated with the weak isospin and the electric charge by

$$Q = I_3 + \frac{1}{2}Y. \tag{2.9}$$

A list of all these quantities is provided in Table (2.1).

	$Q_{0L}^{(d,s,b)}$	$L_L^{(e,\mu,\tau)}$	u_{0R}, c_{0R}, t_{0R}	d_{0R}, s_{0R}, b_{0R}	e_R, μ_R, τ_R	Φ
I_3	$\begin{pmatrix} +\frac{1}{2} \\ -\frac{1}{2} \end{pmatrix}$	$\begin{pmatrix} +\frac{1}{2} \\ -\frac{1}{2} \end{pmatrix}$	0	0	0	$\begin{pmatrix} +\frac{1}{2} \\ -\frac{1}{2} \end{pmatrix}$
Y	$\begin{pmatrix} +\frac{1}{3} \\ +\frac{1}{3} \end{pmatrix}$	$\begin{pmatrix} -1 \\ -1 \end{pmatrix}$	$\frac{4}{3}$	$-\frac{2}{3}$	-2	$\begin{pmatrix} +1 \\ +1 \end{pmatrix}$
Q	$\begin{pmatrix} +\frac{2}{3} \\ -\frac{1}{3} \end{pmatrix}$	$\begin{pmatrix} 0 \\ -1 \end{pmatrix}$	$\frac{2}{3}$	$-\frac{1}{3}$	-1	$\begin{pmatrix} +1 \\ 0 \end{pmatrix}$

Table 2.1: List of weak isospins, weak hypercharges and electric charges of fields contained in the GWS model.

We may now write down the Lagrangian for the GWS model as a sum of the fermion, gauge boson, Higgs and Yukawa terms:

$$\mathcal{L}_{\text{GWS}} = \mathcal{L}_{\text{fermion}} + \mathcal{L}_{\text{gauge}} + \mathcal{L}_{\text{Higgs}} + \mathcal{L}_{\text{Yukawa}} \tag{2.10}$$

Each part is individually $SU(2)_I \times U(1)_Y$ gauge invariant. The first term contains the kinetics of all fermions and their couplings to the gauge bosons:

$$\begin{aligned}\mathcal{L}_{\text{fermion}} = &\sum_{q=d,s,b} \overline{Q_{0L}^{(q)}} i\gamma^\mu D_\mu Q_{0L}^{(q)} + \sum_{l=e,\mu,\tau} \overline{L_L^{(l)}} i\gamma^\mu D_\mu L_L^{(l)} + \\ &+ \sum_{q=d,s,b,u,c,t} \overline{q_{0R}} i\gamma^\mu D_\mu q_{0R} + \sum_{l=e,\mu,\tau} \overline{l_R} i\gamma^\mu D_\mu l_R.\end{aligned} \tag{2.11}$$

Here we used the covariant derivative D_μ, which is different for doublets and singlets and varies also with different weak hypercharges.

$$D_\mu = \partial_\mu + igW_\mu^a I^a + i\frac{1}{2} Y g' B_\mu \tag{2.12}$$

The term with the $SU(2)_I$ gauge bosons contributes only for the doublets.

$$I^a = \begin{cases} \frac{\tau^a}{2} & \text{for } SU(2) \text{ doublets} \\ 0 & \text{for } SU(2) \text{ singlets} \end{cases} \tag{2.13}$$

The second term in the total Lagrangian is the kinetic term for all four gauge bosons.

$$\mathcal{L}_{\text{gauge}} = -\frac{1}{4} W_{\mu\nu}^a W^{a\mu\nu} - \frac{1}{4} B_{\mu\nu} B^{\mu\nu} \tag{2.14}$$

The tensor fields $W_{\mu\nu}^a$ and $B_{\mu\nu}$ are defined by the relations:

$$W_{\mu\nu}^a = \partial_\mu W_\nu^a - \partial_\nu W_\mu^a - g\varepsilon_{abc} W_\mu^b W_\nu^c \tag{2.15}$$

$$B_{\mu\nu} = \partial_\mu B_\nu - \partial_\nu B_\mu. \tag{2.16}$$

Comparing these two definitions one finds an additional term proportional to W^2 in the first line. The reason lies in $SU(2)_I$ being a non-abelian group resulting in self-interaction of related gauge bosons. The structure constants of $SU(2)_I$ also enter at this point.

The Higgs Lagrangian comprises the kinetic term and the potential of the scalar doublet Φ.

$$\mathcal{L}_{\text{Higgs}} = \left(D_\mu \Phi\right)^\dagger \left(D^\mu \Phi\right) - \mu^2 \Phi^\dagger \Phi - \lambda \left(\Phi^\dagger \Phi\right)^2 \tag{2.17}$$

Together with the Yukawa Lagrangian

$$\begin{aligned}\mathcal{L}_{\text{Yukawa}} = &- \sum_{\substack{q=d,s,b \\ q'=d,s,b}} \Gamma_{qq'} \overline{Q_{0L}^{(q)}} \Phi q'_{0R} - \sum_{\substack{q=d,s,b \\ q'=u,c,t}} \Delta_{qq'} \overline{Q_{0L}^{(q)}} \tilde{\Phi} q'_{0R} \\ &- \sum_{l=e,\mu,\tau} \gamma_l \overline{L_L^{(l)}} \Phi l_R + \text{H.c.}\end{aligned} \tag{2.18}$$

these terms are essential for the mass creation in the GWS model since all the masses of fermions and bosons are generated via spontaneous symmetry breaking (SSB) of $SU(2)_I \times U(1)_Y \to U(1)_{em}$. In order to get all allowed Yukawa interaction terms one defines the conjugate scalar doublet $\tilde{\Phi}$.

$$\tilde{\Phi} := i\tau_2 \Phi^* = i \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \begin{pmatrix} \phi^{+*} \\ \phi^{0*} \end{pmatrix} = \begin{pmatrix} \phi^{0*} \\ -\phi^- \end{pmatrix} \quad \to \quad Y(\tilde{\Phi}) = \begin{pmatrix} -1 \\ -1 \end{pmatrix} \tag{2.19}$$

We have now listed all terms constructing the GWS model before the symmetry breaking Higgs mechanism. In the next chapter the focus will lie on the mass generation and the physical fields in the Standard Model.

2.1.2 Higgs mechanism

So far there have been no mass terms in the GWS-Lagrangian, neither for the fermions nor for the gauge bosons. Simply adding mass terms would destroy the gauge invariance in both cases. One possible procedure to get massive fermions and bosons despite these difficulties was introduced by Peter Higgs [7, 8, 9, 10, 11, 12]. The so called Higgs-mechanism breaks the $SU(2)_I \times U(1)_Y$ symmetry spontaneously to one remaining $U(1)_{em}$ symmetry and creates the desired masses. The essence of this process lies in the form of the Higgs potential V

$$V(\Phi) = \mu^2 \Phi^\dagger \Phi + \lambda \left(\Phi^\dagger \Phi \right)^2 \tag{2.20}$$

where the constant μ^2 is assumed to fulfill

$$\mu^2 < 0. \tag{2.21}$$

Searching for the minimum of the potential one finds the relation

$$\frac{\mathrm{d}V(\Phi)}{\mathrm{d}\left(\Phi^\dagger \Phi\right)} \stackrel{!}{=} 0 \Rightarrow \sqrt{\Phi^\dagger \Phi} = \left(-\frac{\mu^2}{2\lambda} \right)^{\frac{1}{2}} =: \frac{1}{\sqrt{2}} v \tag{2.22}$$

which constrains the four parameters of the complex vector Φ. To obtain a minimum we can choose three parameters freely, the last one is then fixed by (2.22). Evidently this leads not to a single, but to a continuous set of minima of the Higgs potential. The vacuum state (which is the state minimizing the potential) is therefore degenerate and the choice of one particular Φ fixes the gauge and hides the original $SU(2)_I \times U(1)_Y$ symmetry. In addition the Higgs field is shifted to a physical field whose vacuum expectation value vanishes. One can now eliminate

three of four real components of the doublet of complex scalar fields by an appropriate choice of the gauge. A useful convention is the unitary gauge, which leaves a real scalar field h in the ϕ^0 component.

$$\Phi = \begin{pmatrix} \sigma^+ + i\eta^+ \\ \sigma^0 + i\eta^0 \end{pmatrix} \xrightarrow[\text{unitary gauge}]{\text{SSB}} \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ v + h \end{pmatrix} = \Phi_u \tag{2.23}$$

The spontaneous symmetry breaking has now effects on various parts of the GWS-Lagrangian.

Higgs mass

Upon SSB the Higgs doublet is transformed into a real scalar field with mass and self-interaction terms.

$$(\partial_\mu \Phi)^\dagger (\partial^\mu \Phi) - V(\Phi) \xrightarrow{\text{SSB}} \frac{1}{2} (\partial_\mu h) (\partial^\mu h) - \lambda v^2 h^2 \left(1 + \frac{1}{2v} h\right)^2 + \frac{\lambda v^4}{4} \tag{2.24}$$

The Higgs mass is therefore

$$m_h^2 = 2\lambda v^2. \tag{2.25}$$

Gauge boson masses

Since the Higgs mechanism breaks a symmetry with 4 generators down to one $U(1)_{em}$ symmetry three gauge bosons gain masses. The mass eigenstates are not the initial gauge fields W_μ^a and B_μ but fields rotated by the following prescription:

$$W_\mu^\pm = \frac{1}{\sqrt{2}} \left(W_\mu^1 \mp i W_\mu^2\right) \tag{2.26}$$

$$\begin{pmatrix} Z_\mu \\ A_\mu \end{pmatrix} = \begin{pmatrix} \cos\vartheta_w & -\sin\vartheta_w \\ \sin\vartheta_w & \cos\vartheta_w \end{pmatrix} \begin{pmatrix} W_\mu^3 \\ B_\mu \end{pmatrix}. \tag{2.27}$$

While $W^\pm$ and Z bosons are now massive fields, the photon A stays massless and represents the electromagnetic $U(1)_{em}$ symmetry. The weak angle ϑ_w may be expressed in terms of the couplings g and g'.

$$\cos\vartheta_w = \frac{g}{\sqrt{g^2 + g'^2}} \qquad \sin\vartheta_w = \frac{g'}{\sqrt{g^2 + g'^2}} \tag{2.28}$$

The electromagnetic charge associated with the unbroken generator Q is now

$$e = g \sin\vartheta_w. \tag{2.29}$$

The kinetic term of the Higgs-Lagrangian comprises the kovariant derivative and therefore the gauge fields. Applying the Higgs mechanism leads to mass terms and interactions with the scalar Higgs field h:

$$\begin{aligned}&\Phi^\dagger \left(-igW_\mu^a \frac{\tau^a}{2} - i\frac{1}{2}Y_\Phi g' B_\mu\right)\left(igW_\mu^a \frac{\tau^a}{2} + i\frac{1}{2}Y_\Phi g' B_\mu\right)\Phi \\ &\xrightarrow[\text{physical fields}]{\text{SSB}} \frac{1}{4}g^2 v^2 W_\mu^+ W^{-\mu}\left(1+\frac{h}{v}\right)^2 + \frac{1}{8}\left(g^2+g'^2\right)v^2 Z_\mu Z^\mu \left(1+\frac{h}{v}\right)^2. \end{aligned} \tag{2.30}$$

The masses of the physical gauge fields are

$$m_{W^\pm}^2 = \frac{1}{4}g^2 v^2 \quad , \quad m_Z^2 = \frac{1}{4}\left(g^2+g'^2\right)v^2 \quad , \quad m_A^2 = 0. \tag{2.31}$$

The masses of the W and Z bosons are related by the weak angle:

$$\frac{m_{W^\pm}^2}{m_Z^2} = \frac{g^2}{g^2+g'^2} = \cos^2\vartheta_w. \tag{2.32}$$

Fermion masses

As previously mentioned, fermion masses cannot be created by simple addition of mass terms because of gauge invariance. Anyway, the Yukawa-Lagrangian together with the Higgs mechanism is capable of generating masses. After SSB the Lagrangian contains Yukawa couplings with the scalar h as well as terms quadratic in the fermion fields proportional to the vacuum expectation value v. The later may be used to form mass terms for all fermions except for the neutrinos.[2]

$$\begin{aligned}\mathcal{L}_{\text{Yukawa}} =& -\frac{1}{\sqrt{2}}(v+h)\begin{pmatrix}\overline{d_{0L}} & \overline{s_{0L}} & \overline{b_{0L}}\end{pmatrix}\begin{pmatrix}\Gamma_{dd} & \Gamma_{ds} & \Gamma_{db}\\ \Gamma_{sd} & \Gamma_{ss} & \Gamma_{sb}\\ \Gamma_{bd} & \Gamma_{bs} & \Gamma_{bb}\end{pmatrix}\begin{pmatrix}d_{0R}\\ s_{0R}\\ b_{0R}\end{pmatrix} \\ &-\frac{1}{\sqrt{2}}(v+h)\begin{pmatrix}\overline{u_{0L}} & \overline{c_{0L}} & \overline{t_{0L}}\end{pmatrix}\begin{pmatrix}\Delta_{uu} & \Delta_{uc} & \Delta_{ut}\\ \Delta_{cu} & \Delta_{cc} & \Delta_{ct}\\ \Delta_{tu} & \Delta_{tc} & \Delta_{tt}\end{pmatrix}\begin{pmatrix}u_{0R}\\ c_{0R}\\ t_{0R}\end{pmatrix} \\ &-\frac{1}{\sqrt{2}}(v+h)\sum_{l=e,\mu,\tau}\gamma_l \overline{l_L} l_R + \text{H.c.}\end{aligned} \tag{2.33}$$

[2] One could argue at this point that no Yukawa coupling matrices comparable to Γ and Δ are introduced in the lepton sector. It is not necessary to write down a non-diagonal matrix $\gamma \neq \text{diag}(\gamma_e, \gamma_\mu, \gamma_\tau)$ since no physical consequences arise from this generalization as long as no right-handed neutrino fields are present. A possible mixing matrix resulting from the diagonalization of a charged-lepton mass matrix (comparable to (2.36)) can be absorbed into the left handed neutrino fields in the charged current term (2.87) of the Lagrangian.

One can now easily read off the masses of the charged leptons:

$$m_l = \frac{1}{\sqrt{2}} v\gamma_l \quad , \quad l = e, \mu, \tau. \tag{2.34}$$

The quark mass matrices Γ and Δ are in general not diagonal, so the quark fields until now where not mass eigenstates. Fortunately there is a procedure which allows us to bring the mass matrices in a positive, diagonal form.

Theorem. Let M be an arbitrary non-singular complex matrix, then M can always be decomposed as

$$M = \mathcal{U}\mathcal{M}\mathcal{V}^\dagger \tag{2.35}$$

where $\mathcal{U}, \mathcal{V}$ are unitary and $\mathcal{M}$ is a diagonal and positive matrix.

The diagonal elements of $\mathcal{M}$ are the singular values and the whole procedure therefore called singular value decomposition or bi-unitary diagonalization. Applying this to our problem gives:

$$\frac{v}{\sqrt{2}}\Gamma = \mathcal{U}_d \mathcal{M}_d \mathcal{V}_d^\dagger \quad \text{and} \quad \frac{v}{\sqrt{2}}\Delta = \mathcal{U}_u \mathcal{M}_u \mathcal{V}_u^\dagger. \tag{2.36}$$

The quark masses are now contained in the diagonal mass matrices

$$\mathcal{M}_d = \begin{pmatrix} m_d & 0 & 0 \\ 0 & m_s & 0 \\ 0 & 0 & m_b \end{pmatrix} \quad \text{and} \quad \mathcal{M}_u = \begin{pmatrix} m_u & 0 & 0 \\ 0 & m_c & 0 \\ 0 & 0 & m_t \end{pmatrix} \tag{2.37}$$

while the unitary matrices are absorbed into the quark fields by definition of the physical mass eigenfields.

$$\begin{pmatrix} d_L \\ s_L \\ b_L \end{pmatrix} := \mathcal{U}_d^\dagger \begin{pmatrix} d_{0L} \\ s_{0L} \\ b_{0L} \end{pmatrix} \qquad \begin{pmatrix} d_R \\ s_R \\ b_R \end{pmatrix} := \mathcal{V}_d^\dagger \begin{pmatrix} d_{0R} \\ s_{0R} \\ b_{0R} \end{pmatrix} \tag{2.38}$$

$$\begin{pmatrix} u_L \\ c_L \\ t_L \end{pmatrix} := \mathcal{U}_u^\dagger \begin{pmatrix} u_{0L} \\ c_{0L} \\ t_{0L} \end{pmatrix} \qquad \begin{pmatrix} u_R \\ c_R \\ t_R \end{pmatrix} := \mathcal{V}_u^\dagger \begin{pmatrix} u_{0R} \\ c_{0R} \\ t_{0R} \end{pmatrix} \tag{2.39}$$

This results in the quark mass terms:

$$\mathcal{L}_{\text{mass}}^{\text{quark}} = -\begin{pmatrix} \overline{u_L} & \overline{c_L} & \overline{t_L} \end{pmatrix} \mathcal{M}_u \begin{pmatrix} u_R \\ c_R \\ t_R \end{pmatrix} - \begin{pmatrix} \overline{d_L} & \overline{s_L} & \overline{b_L} \end{pmatrix} \mathcal{M}_d \begin{pmatrix} d_R \\ s_R \\ b_R \end{pmatrix} + \text{H.c.} \tag{2.40}$$

By now we have depicted the generation of all particle masses in the GWS model. We will continue describing the physical effects of the quark mixture in the next section.

2.1.3 Charged currents and CKM matrix

The unitary matrices $\mathcal{U}_u$ and $\mathcal{U}_d$ introduced before play an important role in the charged current terms of the GWS Lagrangian.

$$\mathcal{L}_{\mathrm{CC}}^{\mathrm{quark}} = -\frac{g}{\sqrt{2}} \begin{pmatrix} \overline{u_{0L}} & \overline{c_{0L}} & \overline{t_{0L}} \end{pmatrix} \gamma^\mu \begin{pmatrix} d_{0L} \\ s_{0L} \\ b_{0L} \end{pmatrix} W_\mu^+ + \mathrm{H.c.} \tag{2.41}$$

Expressing this in terms of mass eigenfields yields

$$\mathcal{L}_{\mathrm{CC}}^{\mathrm{quark}} = -\frac{g}{\sqrt{2}} \begin{pmatrix} \overline{u_L} & \overline{c_L} & \overline{t_L} \end{pmatrix} \gamma^\mu \underbrace{\mathcal{U}_u^\dagger \mathcal{U}_d}_{:=U_{\mathrm{CKM}}} \begin{pmatrix} d_L \\ s_L \\ b_L \end{pmatrix} W_\mu^+ + \mathrm{H.c.}\ . \tag{2.42}$$

The combination of $\mathcal{U}_u$ and $\mathcal{U}_d$ that appears at this point is defined as the famous Cabbibo-Kobayashi-Maskawa matrix U_{CKM} [13, 14]. Naturally the CKM matrix is unitary itself and thus allows a simple parametrization [15]. Consider an arbitray unitary matrix A fulfilling $A^\dagger A = \mathbb{1}$. The unitarity constraint in index notation gives

$$\sum_{j=1}^{3} (A^\dagger)_{ij} A_{jk} = \sum_{j=1}^{3} A_{ji}^* A_{jk} = \delta_{ik}. \tag{2.43}$$

These are 3 real ($i = k$) and 3 complex ($i \neq k$) equations for the 9 complex entries giving a total of $18 - 3 - 3 \cdot 2 = 9$ independent real elements. Conventionally one chooses a parametrization with 3 angles and 6 phases. 5 of 6 phases can be absorbed into the quark fields by redefinition. Finally we end up with a unitary matrix U_{CKM} with 4 parameters commonly parametrized as follows ($s_{ij} := \sin\vartheta_{ij}$,$c_{ij} := \cos\vartheta_{ij}$):

$$\begin{aligned} U_{\mathrm{CKM}} &= U_{23} U_{13} U_{12} \\ &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_{23} & s_{23} \\ 0 & -s_{23} & c_{23} \end{pmatrix} \begin{pmatrix} c_{13} & 0 & s_{13} e^{-i\delta^{CP}} \\ 0 & 1 & 0 \\ -s_{13} e^{i\delta^{CP}} & 0 & c_{13} \end{pmatrix} \begin{pmatrix} c_{12} & s_{12} & 0 \\ -s_{12} & c_{12} & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} c_{12}c_{13} & s_{12}c_{13} & s_{13} e^{-i\delta^{CP}} \\ -s_{12}c_{23} - c_{12}s_{23}s_{13} e^{i\delta^{CP}} & c_{12}c_{23} - s_{12}s_{23}s_{13} e^{i\delta^{CP}} & s_{23}c_{13} \\ s_{12}c_{23} - c_{12}s_{23}s_{13} e^{i\delta^{CP}} & -c_{12}c_{23} - s_{12}s_{23}s_{13} e^{i\delta^{CP}} & c_{23}c_{13} \end{pmatrix}. \end{aligned} \tag{2.44}$$

Note that it is not possible to eliminate the last phase δ^{CP} in the present case of 3 quark generations [14]. Also a non-zero phase is the only possibility for a complex U_{CKM} and thus is responsible for CP violating processes in weak interactions. The whole procedure here, starting from Yukawa couplings, then applying

the bi-unitary diagonalization and ending up with the quark mixing matrix, was motivated by S. Glashow, J. Iliopoulos and L. Maiani [16] who suggested an additional fourth quark c to u, d and s known at the time. In the GIM model they introduced two doublets containing the four quarks and the well-known Cabbibo angle

$$Q_L^{(d)} = \begin{pmatrix} u_L \\ d_L \cos\vartheta_C + s_L \sin\vartheta_C \end{pmatrix} \qquad Q_L^{(s)} = \begin{pmatrix} c_L \\ -d_L \sin\vartheta_C + s_L \cos\vartheta_C \end{pmatrix} \tag{2.45}$$

where the mixture of d and s was set by hand. This way they solved the problem of flavour changing neutral currents appearing in the three quark predecessor model, which led to wrong predictions, e.g. in K^+ decays. A different approach to the quark mixture is via the diagonalization of the mass matrix depicted in the previous chapter. In the GIM model this would correspond to two doublets

$$Q_{0L}^{(d)} = \begin{pmatrix} u_{0L} \\ d_{0L} \end{pmatrix} \qquad Q_{0L}^{(s)} = \begin{pmatrix} c_{0L} \\ s_{0L} \end{pmatrix} \tag{2.46}$$

and a 2×2 lepton mixing matrix

$$U_{GIM} = \mathcal{U}_u^\dagger \mathcal{U}_d = \begin{pmatrix} \cos\vartheta_C & \sin\vartheta_C \\ -\sin\vartheta_C & \cos\vartheta_C \end{pmatrix} \tag{2.47}$$

Besides the natural outcome of the Cabbibo angle ϑ_C this procedure gives also an explanation for the orthogonal mixture in (2.45).[3]

2.2 Extensions in the lepton sector

The aim of this chapter is to discuss the most basic extensions to the SM leading to massive neutrinos (see also [17, 18, 19, 20]). Before one can introduce additional mass terms to the Lagrangian one has to decide whether neutrinos are Dirac- or Majorana particles.

2.2.1 Dirac- and Majorana particles

The fermionic particles of the SM until now were all assumed to be Dirac particles represented by fields fulfilling the Dirac equation:

$$(i\gamma^\mu \partial_\mu - m)\,\psi(x) = 0. \tag{2.48}$$

[3] See [2], p.216ff. for a detailed analysis.

Fourier expansion of the four-component Dirac field ψ yields

$$\psi(x) = \sum_{s=\pm 1/2} \int \frac{\mathrm{d}^3 p}{(2\pi)^{3/2}(2p_0)^{1/2}} \left\{ b(\vec{p},s) u(\vec{p},s) e^{-ipx} + d^\dagger(\vec{p},s) v(\vec{p},s) e^{ipx} \right\} \quad (2.49)$$

with different annihilation and creation operators b and $d^\dagger$ for particles and antiparticles. The charge conjugated field ψ^c is defined by the action of the charge conjugation operation $\mathcal{C}$ on the original field as follows:

$$\mathcal{C}\,\psi\,\mathcal{C}^{-1} = \eta_c \psi^c = \eta_c C \overline{\psi}^T \quad (2.50)$$

where η_c is an arbitrary phase. In the case of Dirac particles one can clearly distinguish between particles and antiparticles, so

$$\psi \neq \psi^c. \quad (2.51)$$

This is particularly obvious for particles that carry any additional conserved quantum number, e.g. electric charge. An electron is always distinguishable from its antiparticle, because of the different charges. The situation gets more involved with neutrinos, since they do not carry an electric charge. Thus it is not clear whether neutrinos and antineutrinos are distinct particles. One could stick to the Dirac particle concept and to statement (2.51). Another possibility is to set particle and antiparticle indistinguishable imposing the so called Majorana condition

$$\psi = \psi^c. \quad (2.52)$$

Applying this to the explicit form of ψ (2.49) gives together with

$$C\overline{u(\vec{p},s)}^T = v(\vec{p},s) \qquad C\overline{v(\vec{p},s)}^T = u(\vec{p},s) \quad (2.53)$$

the desired constraint on the creation and annihilation operators:

$$b(\vec{p},s) = d(\vec{p},s) \qquad b^\dagger(\vec{p},s) = d^\dagger(\vec{p},s). \quad (2.54)$$

Hence there is no mathematical difference between the original and the charge conjugated fields, in other words the Majorana particle is its own antiparticle. In the Standard model there is an additional global $U(1)$ symmetry in the lepton sector that ensures the lepton number conservation (L). This symmetry is accidental and may be broken by a Majorana-type lepton by 2 units.

Switching to the chiral characterization of fields with the well known projection operators

$$P_L = \frac{1}{2}(1-\gamma_5) \qquad P_R = \frac{1}{2}(1+\gamma_5) \quad (2.55)$$

one has to modify the usual decomposition for Dirac particles

$$\psi = \psi_L + \psi_R. \tag{2.56}$$

In the Majorana case only one projection eigenstate is necessary to create a physical field. If one chooses the left-handed field ψ_L, the right-handed part can be played by the charge conjugate state $(\psi_L)^c$.

$$\psi = \psi_L + (\psi_L)^c \tag{2.57}$$

Obviously ψ defined in (2.57) fulfills the Majorana condition (2.52) because $(\psi^c)^c = \psi$. A right-handed field ψ_R may describe a different particle or may be completely absent in the theory.

2.2.2 Mass terms

Depending on the model and the desired particle type it is possible to write down different bilinear terms that are interpreted as mass terms.

Dirac mass term

A Dirac mass term is constructed using the chiral fields ψ_L and ψ_R. Both fields must be present and are combined to form a bilinear and Lorentz-invariant expression

$$-m\left(\overline{\psi_R}\psi_L + \overline{\psi_L}\psi_R\right) = -m\overline{\psi}\psi \qquad \text{with (2.56).} \tag{2.58}$$

Note that the second term in (2.58) is the Hermitean conjugate of the first. Adding a Dirac mass term for the neutrinos in the Standard model is quite simple. It is necessary to add the (so far missing) right-handed neutrino fields $\nu_{eR}, \nu_{\mu R}, \nu_{\tau R}$ as singlets to the particle content. The most general Dirac mass term with these 3 lepton flavours is

$$\mathcal{L}^D = -\sum_{l,l'} \overline{\nu_{lR}} M^D_{ll'} \nu_{l'L} + \text{H.c.} \qquad \text{with} \qquad l, l' = e, \mu, \tau. \tag{2.59}$$

The complex 3×3 matrix M^D is composed of the Yukawa couplings and the VEV of the Higgs field. The mass term can be diagonalized in the same manner as the quark fields described in (2.35) (See also 2.2.4).

Majorana mass term

A Majorana mass term makes only use of one chiral projection of ψ (ψ_L or ψ_R) to construct a bilinear term

$$-\frac{1}{2}m\overline{(\psi_L)^c}\psi_L + \text{H.c.} = -\frac{1}{2}m\overline{\psi}\psi \qquad \text{with (2.57).} \tag{2.60}$$

The above expression may be recast with help of the identity

$$\overline{(\psi_L)^c} = \left(C\overline{\psi_L}^T\right)^\dagger \gamma_0 = \psi_L^T \gamma_0^* C^{-1} \gamma_0 = -\psi_L^T \left(\gamma_0 \gamma_0^\dagger\right)^* C^{-1} = -\psi_L^T C^{-1} \tag{2.61}$$

to a form where the charge conjugation matrix enters

$$\frac{1}{2} m \psi_L^T C^{-1} \psi_L + \text{H.c.} \tag{2.62}$$

In the case of 3 neutrino flavours this would give the following Lagrangian:

$$\mathcal{L}^{M^L} = -\frac{1}{2} \sum_{l,l'} \overline{(\nu_{lL})^c} M^L_{ll'} \nu_{l'L} + \text{H.c.} \qquad \text{with} \qquad l, l' = e, \mu, \tau. \tag{2.63}$$

Note that although the left-handed neutrino fields ν_{lL} are present a Majorana mass term (2.63) is forbidden within the Standard model due to gauge invariance. The same Lagrangian for the right-handed neutrinos would be allowed if the fields $\nu_{eR}, \nu_{\mu R}$ and $\nu_{\tau R}$ are added to the model.

$$\mathcal{L}^{M^R} = -\frac{1}{2} \sum_{l,l'} \overline{(\nu_{lR})^c} M^R_{ll'} \nu_{l'R} + \text{H.c.} \qquad \text{with} \qquad l, l' = e, \mu, \tau \tag{2.64}$$

To get mass eigenstates from a Majorana mass terms like (2.64) one has to diagonalize M^R first. It turns out that the matrix is symmetric because of the antisymmetry of C and the anticommutation property of the fermionic fields.

$$\begin{aligned} \nu_{lR}^T C^{-1} \nu_{l'R} &= \sum_{i,j} (\nu_{lR})_i (C^{-1})_{ij} (\nu_{l'R})_j = \sum_{i,j} \underbrace{(\nu_{lR})_i (\nu_{l'R})_j}_{anticommute} \underbrace{(C^{-1T})_{ji}}_{antisymmetric} \\ &= \sum_{i,j} (\nu_{l'R})_j (C^{-1})_{ji} (\nu_{lR})_i = \nu_{l'R}^T C^{-1} \nu_{lR} \end{aligned} \tag{2.65}$$

From this follows immediately

$$-\frac{1}{2} \sum_{l,l'} \overline{(\nu_{lR})^c} M^R_{ll'} \nu_{l'R} = -\frac{1}{2} \sum_{l,l'} \overline{(\nu_{lR})^c} M^R_{l'l} \nu_{l'R} \quad \Rightarrow \quad M^R = (M^R)^T. \tag{2.66}$$

A theorem of I. Schur [21, 22, 23] ensures that the symmetric matrix M^R can always be diagnonalized:

Theorem. Let M be a symmetric, complex $n \times n$ matrix. Then there exists a diagonal, positive matrix $\hat{m}$ and a unitary matrix $\mathcal{U}$ so that

$$M = (\mathcal{U}^\dagger)^T \hat{m}\, \mathcal{U}^\dagger. \tag{2.67}$$

The resulting mass eigenstates fulfill again the Majorana condition.

Dirac-Majorana mass term

One may find models where Majorana *and* Dirac mass terms occur. In general this would look like

$$\mathcal{L}^{D+M} = -m_D\overline{\psi_R}\psi_L - \frac{1}{2}m_L\overline{(\psi_L)^c}\psi_L - \frac{1}{2}m_R\overline{(\psi_R)^c}\psi_R + \text{H.c.} \tag{2.68}$$

Splitting up the Dirac part using

$$-m_D\overline{\psi_R}\psi_L = -\frac{1}{2}m_D\overline{\psi_R}\psi_L - \frac{1}{2}m_D\overline{(\psi_L)^c}\,(\psi_R)^c \tag{2.69}$$

gives the possibility to write all mass terms in a compact notation

$$\begin{aligned}\mathcal{L}^{D+M} =& -\frac{1}{2}\left(\overline{(\psi_L)^c} \quad \overline{\psi_R}\right)\underbrace{\begin{pmatrix} m_L & m_D \\ m_D & m_R\end{pmatrix}}_{=:M^{D+M}}\underbrace{\begin{pmatrix}\psi_L \\ (\psi_R)^c\end{pmatrix}}_{=:n_L} + \text{H.c.} \\ =& -\frac{1}{2}\overline{(n_L)^c}M^{D+M}n_L + \text{H.c.}\end{aligned} \tag{2.70}$$

The symmetric matrix M^{D+M} is diagonalized by the unitary matrix $\mathcal{U}$

$$M^{D+M} = (\mathcal{U}^\dagger)^T\hat{m}(\mathcal{U}^\dagger) \qquad \text{with} \qquad \hat{m} = (m_i\delta_{ij}) \quad m_i > 0. \tag{2.71}$$

Defining new fields ψ_1, ψ_2

$$N = \begin{pmatrix}\psi_1 \\ \psi_2\end{pmatrix} := \mathcal{U}^\dagger n_L + (\mathcal{U}^\dagger n_L)^c \tag{2.72}$$

the mass Lagrangian takes the simple form

$$\mathcal{L}^{D+M} = -\frac{1}{2}\overline{N}\hat{m}N = -\frac{1}{2}m_1\overline{\psi_1}\psi_1 - \frac{1}{2}m_2\overline{\psi_2}\psi_2. \tag{2.73}$$

Although we started with both Dirac- and Majorana mass terms we finally ended up with two Majorana fields since

$$N = N^c. \tag{2.74}$$

In the case of 3 or more lepton flavours we have to replace ψ_L and ψ_R with vectors containing the various neutrino fields

$$\nu_L := \begin{pmatrix}\nu_{eL} \\ \nu_{\mu L} \\ \nu_{\tau L}\end{pmatrix} \qquad \nu_R := \begin{pmatrix}\nu_{s_1R} \\ \nu_{s_2R} \\ \vdots \\ \nu_{s_{n_R}R}\end{pmatrix} \tag{2.75}$$

and n_L with

$$n_L := \begin{pmatrix} \nu_L \\ (\nu_R)^c \end{pmatrix}. \tag{2.76}$$

Note that we assumed 3 (e, μ, τ) and n_R lepton flavours for the left- and right-handed neutrino fields, respectively. The number of right-handed flavours is free, because there has been no experimental or theoretical limitation so far. The combined mass matrix M^{D+M} has now $(3 + n_R) \times (3 + n_R)$ elements arranged in 3 matrices M^L, M^R and M^D

$$M^{D+M} = \begin{pmatrix} M^L & (M^D)^T \\ M^D & M^R \end{pmatrix}. \tag{2.77}$$

The Majorana particles resulting from the usual diagonalization are again denoted by a vector

$$N := \begin{pmatrix} \nu_1 \\ \nu_2 \\ \vdots \\ \nu_{3+n_R} \end{pmatrix} = \mathcal{U}^\dagger n_L + (\mathcal{U}^\dagger n_L)^c. \tag{2.78}$$

2.2.3 The seesaw mechanism

Starting from a general Dirac-Majorana mass term the seesaw mechanism [24, 25, 26] gives a possible explanation for the smallness of the neutrino masses. Of course it would be possible to simply set the corresponding Yukawa couplings accordingly, but this would be an unmotivated step. Instead one tries to find a different mass production procedure that naturally explains the mass discrepancies between charged and neutral leptons. It is based on a scenario with Dirac-Majorana mass term described in (2.75) to (2.78) with 3 left-handed and n_s right-handed neutrinos. There are different types of the seesaw mechanism, we will concentrate on type I seesaw:

Type I seesaw

Based on the mass matrix (2.77) with $M^L = 0$, the type I seesaw mechanism corresponds to the SM with n_s additional right-handed neutrinos and all possible mass terms.

$$M^{D+M} = \begin{pmatrix} 0 & (M^D)^T \\ M^D & M^R \end{pmatrix} \tag{2.79}$$

There is no Majorana mass term for ν_L because the left-handed fields are arranged in doublets, thus a gauge invariant bilinear term can not be produced. The seesaw

mechanism makes now assumptions on the scales of the matrices M^D and M^R. The eigenvalues of M^D are at the same level as the masses of the charged leptons or the quarks, whereas those of M^R are at much larger scale.

$$M^D \simeq m_{\text{quarks,charged leptons}} \ll M^R \tag{2.80}$$

Under these conditions one may try to bring (2.79) into a block-diagonal form, applying a small rotation

$$W^T M^{D+M} W \simeq \begin{pmatrix} \mathcal{M}_\nu & 0 \\ 0 & \mathcal{M}_{\text{heavy}} \end{pmatrix} = \mathcal{M}. \tag{2.81}$$

The rotation matrix W that performs the partial diagonalization is

$$W \simeq \begin{pmatrix} 1 - \frac{1}{2}(M^D)^\dagger (M^R (M^R)^\dagger)^{-1} M^D & (M^D)^\dagger (M^R)^{\dagger -1} \\ -(M^R)^{-1} M^D & 1 - \frac{1}{2}(M^R)^{-1} M^D (M^D)^\dagger (M^R)^{\dagger -1} \end{pmatrix} \tag{2.82}$$

up to orders $(M^R)^{-1} M^D$. The resulting Majorana mass matrix $\mathcal{M}_\nu$ is now

$$\begin{aligned} \mathcal{M}_\nu = & -(M^D)^T (M^R)^{-1} M^D \\ & + \frac{1}{2}(M^D)^T (M^R)^{-1} M^D (M^D)^T ((M^R)^* (M^R)^T)^{-1} (M^D)^* \\ & + (M^D)^T (M^R)^{-1T} M^D (M^D)^\dagger (M^R (M^R)^\dagger)^{-1} M^D \\ = & -(M^D)^T (M^R)^{-1} M^D \left\{ 1 + O\left[((M^R)^{-1} M^D)^2 \right] \right\} \end{aligned} \tag{2.83}$$

which explains the term "seesaw". If M^R gets bigger in (2.83) then consequently $\mathcal{M}_\nu$ gets smaller due to the inverse operator. Thus the small masses for the left-handed neutrinos observed until now is explained by the existence of n_s heavy right-handed neutrinos and the possible mass terms in the Lagrangian

$$\begin{aligned} \mathcal{L}^{D+M} = & -\frac{1}{2} \overline{(n_L)^c} M^{D+M} n_L + \text{H.c.} \\ = & -\frac{1}{2} \overline{(n_L)^c} W^* \mathcal{M} W^\dagger n_L + \text{H.c.} + O\left[((M^R)^{-1} M^D)^2 \right]. \end{aligned} \tag{2.84}$$

The rotation W changes the states n_L slightly to n'_L, but the mixture of ν_L and $(\nu_R)^c$ is suppressed by $(M^R)^{-1} M^D$

$$\begin{aligned} n'_L = W^\dagger n_L = & \begin{pmatrix} \nu_L - (M^D)^\dagger (M^R)^{\dagger -1} \left[\frac{1}{2} (M^R)^{-1} M^D \nu_L + (\nu_R)^c \right] \\ (\nu_R)^c - (M^R)^{-1} M^D \left[\frac{1}{2} (M^D)^\dagger (M^R)^{\dagger -1} (\nu_R)^c - \nu_L \right] \end{pmatrix} \\ = & \begin{pmatrix} \nu_L \\ (\nu_R)^c \end{pmatrix} + O\left[(M^R)^{-1} M^D \right]. \end{aligned} \tag{2.85}$$

Finally we end up with a Lagrangian containing Majorana mass terms for the light left-handed neutrinos and their heavy right-handed counterparts which have not been observed yet.

$$\begin{aligned}\mathcal{L}^{D+M} = &-\frac{1}{2}\overline{(\nu_L)^c}\mathcal{M}_\nu\nu_L + \text{H.c.} + (\text{terms with } \mathcal{M}_{\text{heavy}}) \\ &+O\left[((M^R)^{-1}M^D)^2\right]\end{aligned} \tag{2.86}$$

Note that the matrix $\mathcal{M}_\nu$ is still not diagonal since we only performed a block-diagonalization.

2.2.4 The lepton mixing matrix

Most models with massive neutrinos and neutrino oscillations operate with non-diagonal mass matrices in the lepton sector. A diagonalization has effects on the charged current terms in the Lagrangian just like in the quark sector.

$$\mathcal{L}^{\text{lepton}}_{CC} = -\frac{g}{\sqrt{2}}\left(\overline{e_L} \quad \overline{\mu_L} \quad \overline{\tau_L}\right)\gamma^\mu \begin{pmatrix}\nu_{eL}\\ \nu_{\mu L}\\ \nu_{\tau L}\end{pmatrix} W^-_\mu + \text{H.c.} \tag{2.87}$$

Consider a Dirac mass term like (2.59) in short notation

$$\mathcal{L}^D = -\overline{\nu_R}M^D\nu_L + \text{H.c.} \quad \text{with} \quad \nu = \begin{pmatrix}\nu_e\\ \nu_\mu\\ \nu_\tau\end{pmatrix}. \tag{2.88}$$

Diagonalization via

$$M^D = \mathcal{U}\hat{m}\mathcal{V}^\dagger \tag{2.89}$$

gives in analogy to the quark case the mass eigenfields

$$\begin{pmatrix}\nu_{1L}\\ \nu_{2L}\\ \nu_{3L}\end{pmatrix} = \mathcal{U}^\dagger \begin{pmatrix}\nu_{eL}\\ \nu_{\mu L}\\ \nu_{\tau L}\end{pmatrix} \qquad \begin{pmatrix}\nu_{1R}\\ \nu_{2R}\\ \nu_{3R}\end{pmatrix} = \mathcal{V}^\dagger \begin{pmatrix}\nu_{eR}\\ \nu_{\mu R}\\ \nu_{\tau R}\end{pmatrix}. \tag{2.90}$$

Inserting into (2.87) yields

$$\mathcal{L}^{\text{lepton}}_{\text{CC}} = -\frac{g}{\sqrt{2}}\left(\overline{e_L} \quad \overline{\mu_L} \quad \overline{\tau_L}\right)\underbrace{\mathcal{U}}_{=:U_{\text{PMNS}}}\gamma^\mu \begin{pmatrix}\nu_{1L}\\ \nu_{2L}\\ \nu_{3L}\end{pmatrix} W^-_\mu + \text{H.c.} \tag{2.91}$$

where $\mathcal{U}$ is the analogy to the CKM matrix which in the lepton sector is often referred to as the Pontecorvo-Maki-Nakagawa-Sakata (PMNS) matrix. Note that

we assumed here that the charged lepton mass matrix is already in a diagonal form. Without this assumption the PMNS matrix would consist of two unitary matrices like in (2.42).

$$U_{PMNS} = \mathcal{U}_e^\dagger \mathcal{U}_\nu \tag{2.92}$$

The parametrization of U_{PMNS} [15] is also analogous to the quark case, where

$$\vartheta_{23} \to \vartheta_{\text{atm}} \qquad \vartheta_{12} \to \vartheta_\odot \qquad \delta_{CP} \to \delta_{13} \tag{2.93}$$

so that

$$\begin{aligned} U_{\text{PMNS}}^{\text{Dirac}} &= U_{\text{atm}} U_{13} U_\odot \\ &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_{\text{atm}} & s_{\text{atm}} \\ 0 & -s_{\text{atm}} & c_{\text{atm}} \end{pmatrix} \begin{pmatrix} c_{13} & 0 & s_{13} e^{-i\delta_{13}} \\ 0 & 1 & 0 \\ -s_{13} e^{i\delta_{13}} & 0 & c_{13} \end{pmatrix} \begin{pmatrix} c_\odot & s_\odot & 0 \\ -s_\odot & c_\odot & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} c_\odot c_{13} & s_\odot c_{13} & s_{13} e^{-i\delta_{13}} \\ -s_\odot c_{\text{atm}} - c_\odot s_{\text{atm}} s_{13} e^{i\delta_{13}} & c_\odot c_{\text{atm}} - s_\odot s_{\text{atm}} s_{13} e^{i\delta_{13}} & s_{\text{atm}} c_{13} \\ s_\odot c_{\text{atm}} - c_\odot s_{\text{atm}} s_{13} e^{i\delta_{13}} & -c_\odot c_{\text{atm}} - s_\odot s_{\text{atm}} s_{13} e^{i\delta_{13}} & c_{\text{atm}} c_{13} \end{pmatrix}. \end{aligned} \tag{2.94}$$

In the Majorana mass term scenario with

$$\mathcal{L}^M = -\frac{1}{2} \overline{(\nu_L)^c} M^L \nu_L + \text{H.c.} \tag{2.95}$$

the proper diagonalization is described in (2.67). Only one matrix $\mathcal{U}$ is needed and the parametrization is the same with one exception: In the Dirac case 5 of 6 phases were absorbed into the lepton fields. With a Majorana mass term only the 3 charged leptons are allowed to be redefined, since (2.95) itself is not invariant under rephasing. This results in a lepton mixing matrix that contains two additional phases η_1 and η_2 called Majorana phases.

$$U_{\text{PMNS}}^{\text{Majorana}} = U_{\text{PMNS}}^{\text{Dirac}} \cdot \begin{pmatrix} e^{i\eta_1} & 0 & 0 \\ 0 & e^{i\eta_2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.96}$$

Right-handed Majorana neutrino fields added to the SM do not carry electric or color charge nor do they take part in weak interactions (2.87). Thus their number and their mass is unknown and they are often called *sterile* neutrinos.

Chapter 3

Neutrino phenomena and experiments

After describing the basic extensions in the lepton sector we will proceed describing the phenomena resulting from massive neutrinos. The most important consequence is the possibility for neutrino oscillations, by now verified by several collaborations (see section 3.4). Also, one still has to decide experimentally whether neutrinos are Dirac or Majorana particles. Currently the most promising candidate for a crucial experiment is the search for neutrinoless double beta ($(\beta\beta)_{0\nu}$) decay. If $(\beta\beta)_{0\nu}$ decay is discovered neutrinos must be Majorana particles.

3.1 Neutrino oscillations

We start with the question why massive neutrinos give rise to neutrino oscillations, for detailed discussions see [17, 18, 19, 20]. Consider a model with a unitary lepton mixing matrix U_{PMNS}, so that

$$\nu_{\alpha L} = \sum_{k=1}^{3} U_{\alpha k} \nu_{kL}. \tag{3.1}$$

where $\nu_{\alpha L}$ is a flavour eigenfield and ν_{kL} is a mass eigenfield. This means that, depending on the explicit form of U, each flavour field may be a superposition of up to 3 mass eigenfields. Now, neutrinos are produced in CC interactions in (2.87), with the flavour of the lepton occuring in the process. This is the only possible way to define the neutrino flavour. The same is true for the neutrino detection, again the flavour is not measured directly but defined by the associated

lepton. Producing a neutrino of flavour α is described by the action of the creation operator $b_\alpha^\dagger$, contained in $\nu_\alpha^\dagger$, on the vacuum state $|0\rangle$.

$$b_\alpha^\dagger|0\rangle = |\nu_\alpha\rangle \tag{3.2}$$

Because of (3.1) this state is a superposition of the mass eigenstates $|\nu_k\rangle$.

$$|\nu_\alpha\rangle = \sum_{k=1}^{3} U_{\alpha k}^* |\nu_k\rangle = \sum_{k=1}^{3} U_{\alpha k}^* b_k^\dagger(\vec{p_k}, s_k)|0\rangle \tag{3.3}$$

To get the neutrino oscillation formula, we now consider a neutrino with flavour α produced at time $t = 0$ at $\vec{x} = \vec{0}$. The propability for the neutrino to be detected with another flavour β at $t = \Delta t$ and $\vec{x} = (\Delta x, 0, 0)$ is then given by

$$P_{\nu_\alpha \to \nu_\beta}(\Delta t, \Delta x) = |\mathcal{A}_{\nu_\alpha \to \nu_\beta}(\Delta t, \Delta x)|^2 = |{}_{(0,0)}\langle \nu_\beta | \nu_\alpha \rangle_{(\Delta t, \Delta x)}|^2. \tag{3.4}$$

The propagation of ν_α from the production to the detection point is carried out via the operators H and P:

$$|\nu_\alpha\rangle_{(\Delta t, \Delta x)} = e^{-i(H\Delta t - P\Delta x)} |\nu_\alpha\rangle_{(0,0)}. \tag{3.5}$$

According to (3.3) $|\nu_\alpha\rangle_{(0,0)}$ may be rewritten as a coherent superposition of the mass eigenstates. We assume that they all have the same energy E, but differ in their momenta p_k. Additionally we take the relativistic limit $E \gg m_k$, which is a good approximation for all neutrino experiments:

$$p_k = \sqrt{E^2 - m_k^2} \simeq E - \frac{m_k^2}{2E}. \tag{3.6}$$

Then (3.5) becomes

$$|\nu_\alpha\rangle_{(\Delta t, \Delta x)} = e^{-iE\Delta t} \sum_{k=1}^{3} U_{\alpha k}^* e^{ip_k \Delta x} |\nu_k\rangle_{(0,0)}. \tag{3.7}$$

With the inverse of (3.3),

$$|\nu_k\rangle = \sum_\gamma U_{\gamma k} |\nu_\gamma\rangle, \tag{3.8}$$

we get

$$|\nu_\alpha\rangle_{(\Delta t, \Delta x)} = e^{-iE\Delta t} \sum_\gamma \sum_{k=1}^{3} U_{\alpha k}^* e^{ip_k \Delta x} U_{\gamma k} |\nu_\gamma\rangle_{(0,0)}. \tag{3.9}$$

Projecting out the flavour of interest β and taking the relativistic limit delivers the amplitude for the process:

$$\mathcal{A}_{\nu_\alpha \to \nu_\beta}(\Delta t, \Delta x) = {}_{(0,0)}\langle \nu_\beta | \nu_\alpha \rangle_{(\Delta t, \Delta x)} = e^{-iE(\Delta t - \Delta x)} \sum_{k=1}^{3} U_{\alpha k}^* e^{-i\frac{m_k^2}{2E}\Delta x} U_{\beta k}. \tag{3.10}$$

The time dependence in the phase $e^{-iE(\Delta t-\Delta x)}$ becomes irrelevant in the resulting propability

$$P_{\nu_\alpha\to\nu_\beta}(\Delta x = L) = \left|\sum_{k=1}^{3} U^*_{\alpha k} e^{-i\frac{m_k^2 L}{2E}} U_{\beta k}\right|^2 . \tag{3.11}$$

This is the well-known formula for neutrino oscillations with dependence on the energy E, the distance source - detector L and the differences of the squared masses Δm^2_{ij}. The latter will become clear if we write down all terms of the sum in (3.11)

$$\begin{aligned} P_{\nu_\alpha\to\nu_\beta}(L) &= \left|U^*_{\alpha 1} e^{-i\frac{m_1^2 L}{2E}} U_{\beta 1} + U^*_{\alpha 2} e^{-i\frac{m_2^2 L}{2E}} U_{\beta 2} + U^*_{\alpha 3} e^{-i\frac{m_3^2 L}{2E}} U_{\beta 3}\right|^2 \\ &= \left|U^*_{\alpha 1} U_{\beta 1} + U^*_{\alpha 2} e^{-i\frac{(m_2-m_1)^2 L}{2E}} U_{\beta 2} + U^*_{\alpha 3} e^{-i\frac{(m_3-m_1)^2 L}{2E}} U_{\beta 3}\right|^2 \\ &= \left|\delta_{\alpha\beta} + \sum_{k=2}^{3} U^*_{\alpha k}\left[e^{-i\frac{\Delta m^2_{k1} L}{2E}} - 1\right] U_{\beta k}\right|^2 , \end{aligned} \tag{3.12}$$

where $\Delta m^2_{ij} = m_i^2 - m_j^2$ and we used the unitarity relation

$$\sum_{k=1}^{3} U^*_{\alpha k} U_{\beta k} = 1 \tag{3.13}$$

for the mixing matrix. Next we investigate the case of transitions between two neutrino flavours (and two mass eigenstates). Then the mixing matrix U is simply parametrized as

$$U = \begin{pmatrix} U_{\alpha 1} & U_{\alpha 2} \\ U_{\beta 1} & U_{\beta 2} \end{pmatrix} = \begin{pmatrix} \cos\vartheta & \sin\vartheta \\ -\sin\vartheta & \cos\vartheta \end{pmatrix} . \tag{3.14}$$

Inserting the explicit form of U in (3.12) we end up with the transition propability in the two-neutrino case

$$P_{\nu_\alpha\to\nu_\beta}(L) = \frac{1}{2}\sin^2(2\vartheta)(1 - \underbrace{\cos(\frac{\Delta m^2 L}{2E})}_{=\cos(2\pi L/L_{\rm osc})}) \tag{3.15}$$

The oscillation length L is

$$L_{\rm osc} = \frac{4\pi E}{\Delta m^2} \simeq 2.48\frac{E(\text{MeV})}{\Delta m^2(\text{eV}^2)}\text{m}. \tag{3.16}$$

Here lies the justification for the term neutrino oscillation. An experiment varying either the length L or the energy E may be able to capture the oscillatory behaviour of the transition propability $P_{\nu_\alpha\to\nu_\beta}$. The region of (L, E) where to find significant oscillations is determined by the scale of the mass square difference

Δm^2. Furthermore the mixing angle ϑ enters as a parameter controlling the "amplitude" of the oscillation curve. Some additional remarks concerning neutrino oscillations are in order at this point:

- The transition propability (3.11) is invariant under rephasing of the mixing matrix U:
$$U \to \mathrm{diag}(e^{i\phi_1}, e^{i\phi_2}, e^{i\phi_3})\, U\, \mathrm{diag}(e^{i\psi_1}, e^{i\psi_2}, e^{i\psi_3}). \tag{3.17}$$
Consequently, neutrino oscillation experiments cannot determine the Dirac or Majorana nature hidden in two phases of the mixing matrix (see (2.96)).

- The transformation $U \to U^*$ gives the transition propability $P_{\bar{\nu}_\alpha \to \bar{\nu}_\beta}$ for the case of antineutrinos. Additional exchange of the flavours $\alpha \leftrightarrow \beta$ gives the equality
$$P_{\nu_\alpha \to \nu_\beta} = P_{\bar{\nu}_\beta \to \bar{\nu}_\alpha}, \tag{3.18}$$
being a consequence of the CPT-invariance of the theory.

- Evidently neutrino oscillations violate family lepton numbers L_α.

- The derivation of the neutrino oscillation formula was performed without taking into account various aspects of quantum mechanics that may play an important role during production and detection. Also the relativistic limit (3.6) may be questioned in favour of a more general description. Many authors have approached the problem in different ways by now, but the validity of (3.11) has been confirmend within the experimental limitations [27, 28, 29, 30].

- Until now the neutrinos were assumed to propagate in vacuum. A more detailed analysis shows that effects of neutrinos passing through matter are not negligible for oscillation phenomena, especially in the case of the solar neutrino deficit [31, 32, 33, 34, 35].

3.2 Mass spectra

In the previous section we found that neutrino oscillations depend on the mass squared differences (3.12). Thus, experiments observing oscillations cannot give absolute values for the masses and, moreover, allow for two possible mass spectra [36]. The two mass squared differences are denoted as $\Delta m^2_{\mathrm{atm}}$ and $\Delta m^2_{\odot}$ for atmospheric and solar neutrino oscillations, respectively. From experiments we know

that they are not at the same scale (for a detailed listing see section 3.4):

$$\Delta m^2_{\text{atm}} \sim 30 \cdot \Delta m^2_{\odot}. \tag{3.19}$$

We make use of this information and choose a convention for the so far arbitrary numbering of the neutrino mass eigenfields. The smaller mass squared difference $\Delta m^2_{\odot}$ is associated with the masses m_1 and m_2 and chosen to fulfill:

$$\Delta m^2_{\odot} = \Delta m^2_{21} > 0 \;\Rightarrow\; m_1 < m_2. \tag{3.20}$$

The remaining mass m_3 may now be smaller or greater than $m_1 \approx m_2$, i.e. the sign of Δm^2_{atm} is not fixed by experiment. Thus, one has to distinguish between two different mass spectra.

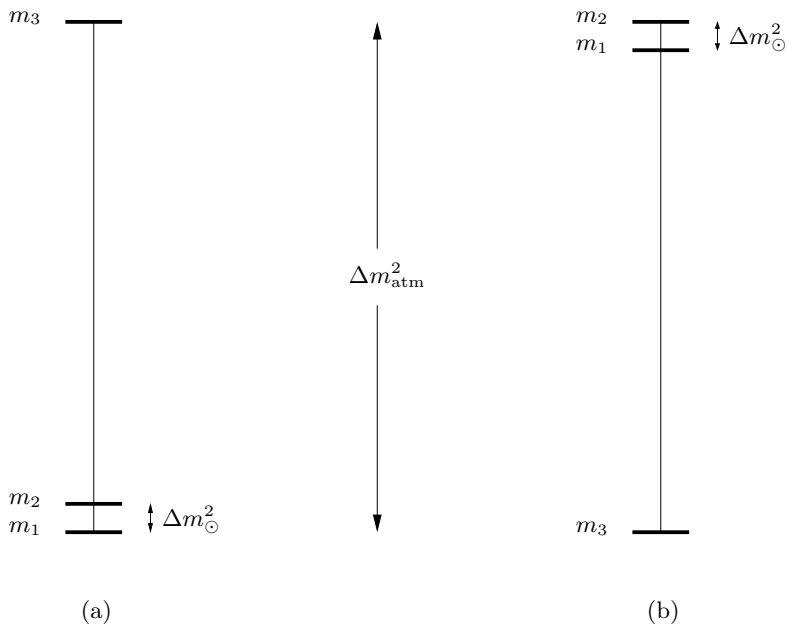


Figure 3.1: Normal (a) and inverted (b) mass spectrum with mass squared differences

- Normal spectrum

$$m_1 < m_2 < m_3 \quad , \quad \Delta m^2_{\text{atm}} = \Delta m^2_{31} \tag{3.21}$$

- Inverted spectrum

$$m_3 < m_1 < m_2 \quad , \quad \Delta m^2_{\text{atm}} = \Delta m^2_{23}. \tag{3.22}$$

See Figure 3.1 for an illustration of the possibilities.

3.3 Neutrinoless double beta decay

Beside neutrino oscillations there is another possibility to extract neutrino properties from experiment. The neutrinoless double beta decay ($(\beta\beta)_{0\nu}$ decay) not only allows us to get information about masses and mixing angles but maybe also a decision in the question whether neutrions are Dirac- or Majorana particles (see e.g. [37, 38]). The idea of a $(\beta\beta)_{0\nu}$ decay is based on the already observed double beta decay

$$(Z, A) \to (Z + 2, A) + 2e^- + 2\overline{\nu}_e. \tag{3.23}$$

If the neutrinos in the process are Majorana fermions, then one could imagine another double beta decay without the outgoing $\overline{\nu}_e$:

$$(Z, A) \to (Z + 2, A) + 2e^-. \tag{3.24}$$

This $(\beta\beta)_{0\nu}$ decay is possible only if $\nu^c = \nu$ and thus both neutrinos can be eliminated via the Wick contraction. Figure 3.2 shows the process on the quark level. The propagator $\langle 0|T\nu_{eL}(x)\nu_{eL}^T(y)|0\rangle$ is nonzero because one can employ

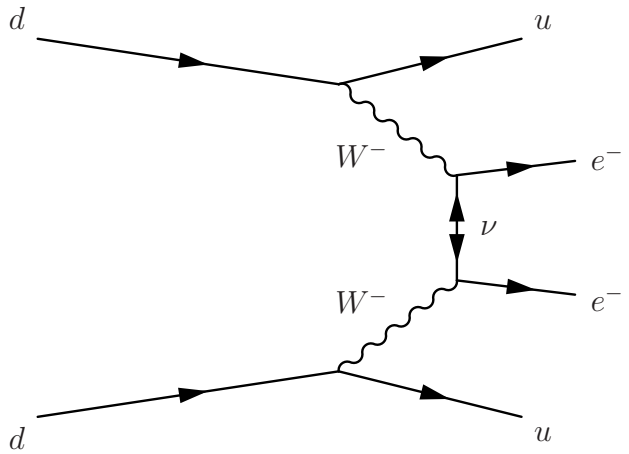


Figure 3.2: $(\beta\beta)_{0\nu}$ decay - Feynman diagram at quark level

$\nu_k^T = -\overline{\nu_k}\mathcal{C}$ and ends up with

$$\begin{aligned}\langle 0|T\nu_{eL}(x)\nu_{eL}^T(y)|0\rangle &= \langle 0|T\sum_k U_{ek}\frac{1-\gamma_5}{2}\nu_k(x)\sum_{k'}U_{ek'}\nu_{k'}^T(y)\frac{1-\gamma_5^T}{2}|0\rangle \\ &= \frac{1-\gamma_5}{2}\sum_k U_{ek}^2\langle 0|T\nu_k(x)\nu_k^T(y)|0\rangle\frac{1-\gamma_5^T}{2} \\ &= -\frac{1-\gamma_5}{2}\sum_k U_{ek}^2\langle 0|T\nu_k(x)\overline{\nu}_k(y)|0\rangle\frac{1-\gamma_5}{2}\mathcal{C} \\ &= -\sum_k U_{ek}^2 m_k i\int\frac{\mathrm{d}^4p}{(2\pi)^4}\frac{e^{-ip(x-y)}}{p^2-m_k^2}\frac{1-\gamma_5}{2}\mathcal{C}.\end{aligned} \tag{3.25}$$

Consequently the lepton number conservation is violated by two units. Models beyond the standard model may be equipped with other Majorana particles or different mechanisms for lepton number violation. Therefore, these additional effects must be taken into account when drawing any conclusions about neutrino masses. Anyway, if $(\beta\beta)_{0\nu}$ decay exists then a Majorana neutrino mass term cannot be forbidden by a symmetry and thus contributes to lepton number violating processes [39, 40]. If the sole source of $(\beta\beta)_{0\nu}$ decay is the contraction of Majorana neutrinos the decay rate for this process is proportional to the effective Majorana neutrino mass

$$|\langle m_{\beta\beta}\rangle| = \left|\sum_k U_{ek}^2 m_k\right|, \tag{3.26}$$

as already indicated in (3.25). Writing (3.26) in terms of mass squared differences and inserting the standard parametrization of the lepton mixing matrix (2.94) and (2.96) gives

$$\begin{aligned}|\langle m_{\beta\beta}\rangle| = \Big|&\left(m_1 c_\odot^2\, e^{i\eta_1} + \sqrt{m_1^2+\Delta m_\odot^2}\, s_\odot^2\, e^{i\eta_2}\right)c_{13}^2 \\ &+\sqrt{m_1^2+\Delta m_{\mathrm{atm}}^2}\, s_{13}^2\, e^{-i\delta_{13}}\Big|\end{aligned} \tag{3.27}$$

and

$$\begin{aligned}|\langle m_{\beta\beta}\rangle| = \Big|&\left(\sqrt{m_3^2+\Delta m_{\mathrm{atm}}^2-\Delta m_\odot^2}\, c_\odot^2\, e^{i\eta_1}\right. \\ &\left.+\sqrt{m_3^2+\Delta m_{\mathrm{atm}}^2}\, s_\odot^2\, e^{i\eta_2}\right)c_{13}^2 + m_3 s_{13}^2\, e^{-i\delta_{13}}\Big|\end{aligned} \tag{3.28}$$

for normal and inverted spectrum, respectively. Note that a possible nonzero phase δ_{13} may be absent if a different parametrization of the lepton mixing matrix (2.94) is chosen. In contrast Majorana phases cannot be removed from the effective Majorana mass.

Observable O_i	$\bar{O}_i \pm 1\sigma_i$	$3\sigma_i$ interval
$\Delta m^2_{\odot}[10^{-5}\ \mathrm{eV}^2]$	$7.65^{+0.23}_{-0.20}$	$7.05 - 8.34$
$\Delta m^2_{\mathrm{atm}}[10^{-3}\ \mathrm{eV}^2]$	$2.40^{+0.12}_{-0.11}$	$2.07 - 2.75$
$\sin^2\vartheta_{13}$	$0.010^{+0.016}_{-0.011}$	≤ 0.056
$\sin^2\vartheta_{\odot}$	$0.304^{+0.022}_{-0.016}$	$0.25 - 0.37$
$\sin^2\vartheta_{\mathrm{atm}}$	$0.50^{+0.07}_{-0.06}$	$0.36 - 0.67$

Table 3.1: Experimental global fit values.

3.4 Experiments and data

This section presents the experimental data employed in the later chapters of this work.

Oscillation parameters

While the search for neutrinoless double beta decay is still in progress, neutrino oscillations have been confirmed by many collaborations up to now. The data from several experiments is combined to obtain a global fit in recent reviews [41, 42]. The resulting mass squared differences and mixing angles therein are presented in Table 3.1. The review includes the latest data from the following collaborations:

- **MINOS** (Main Injector Neutrino Oscillation Search)

  MINOS [43, 44] is a long-baseline accelerator experiment studying neutrino oscillations on the basis of a ν_μ-beam and two separate detectors. The neutrino beam is produced at Fermilab and first analysed by the MINOS Near detector on-site. The second MINOS Far detector is located 735km away at the Soudan Underground Laboratory in Northern Minnesota. The survival propability of the ν_μ-beam with a peak energy of 3 GeV allows measurement of $\Delta m^2_{\mathrm{atm}}$ and $\sin^2 2\vartheta_{\mathrm{atm}}$. In addition MINOS searches for sterile neutrino flavours and further limits $\sin^2 2\vartheta_{13}$.

- **SNO-NCD** (The Sudbury Neutrino Observatory)

  The Sudbury Neutrino Observatory [45, 46] uses a Cherenkov detector with 1 kton of heavy water to investigate the neutrino flux coming from the sun. The laboratory is placed 2km under the surface in the Creighton mine near

Sudbury, Ontario. While charged current interactions with the deuterium allow only the detection of ν_e - neutrinos the neutral current detection (NCD) is sensitive to all neutrino flavours. The experiment gives results for $\Delta m^2_\odot$ and $\vartheta_\odot$.

- **KamLAND** (Kamioka Liquid-scintillator Anti-Neutrino Detector)

  The KamLAND experiment [47], located at the site of the former Kamiokande experiment, utilizes a 1 kton liquid scintillator detector. It observes $\bar{\nu}_e$-neutrinos emitted from the 55 Japanese nuclear reactors via inverse β - decay. The experiment confirmes neutrino oscillations and provides valuable data for the determination of $\Delta m^2_\odot$ and $\sin^2\vartheta_\odot$.

- **Borexino**

  Borexino [48] is a scintillator detector experiment at the Laboratori Nazionali del Gran Sasso. The focus lies on low energy neutrinos from the electron capture decay of ^7Be in the sun.

This enumeration adds up to a long list of experiments already finished or still in progress: CHOOZ [49] , GALLEX [50, 51], SAGE [52], Superkamiokande [53], K2K [54], T2K [55], OPERA [56], MiniBooNE [57], KATRIN [58].

The numerical values for the mixing angles encouraged theorists to find possible analytic expressions matching the global fit data. The so-called tri-bimaximal mixing is a common choice of a mixing pattern introduced by Harrison, Perkins and Scott [59]. With the mixing angles fulfilling

$$\sin^2\vartheta_{13} = 0, \quad \sin^2\vartheta_\odot = \frac{1}{3}, \quad \sin^2\vartheta_{\text{atm}} = \frac{1}{2} \tag{3.29}$$

we get the Harrison-Perkins-Scott mixing matrix

$$U_{\text{HPS}} = \begin{pmatrix} \sqrt{\frac{2}{3}} & \frac{1}{\sqrt{3}} & 0 \\ -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{2}} \end{pmatrix}. \tag{3.30}$$

Note that tri-bimaximal mixing does not fully comply with current data in Table 3.1. While $\sin^2\vartheta_{13}$ and $\sin^2\vartheta_{\text{atm}}$ are within the 1σ - bound, $\sin^2\vartheta_\odot$ is only within the 2σ - bound. In order to allow deviation from fixed angles Grimus and Lavoura [60] suggested a trimaximal mixing pattern. While the second column in (3.30) remains unchanged, i.e.

$$|U_{e2}|^2 = |U_{\mu 2}|^2 = |U_{\tau 2}|^2 = \frac{1}{3}, \tag{3.31}$$

no further constraints are stated. Thus, trimaximal mixing allows e.g. for a nonzero ϑ_{13}.

Neutrinoless double beta decay

For years the existence of neutrinoless double beta decay was subject of ongoing discussion. Among the few so far finished experiments are IGEX [61], CUORICINO [62] and Heidelberg-Moscow [63]. While the first two could not confirm neutrinoless double beta decay and gave only upper bounds on the effective Majorana mass $|\langle m_{\beta\beta}\rangle|$, a subgroup of Heidelberg-Moscow [64] claimed the observation and reported

$$|\langle m_{\beta\beta}\rangle| = 0.32^{+0.03}_{-0.03}\ \text{eV}. \tag{3.32}$$

Though heavily criticized [65, 66, 67, 68], they repeatedly defended their claim in [69, 70, 71]. Hopefully this issue will be resolved in the upcoming neutrinoless double beta decay experiments, namely GERDA [72, 73], CUORE [74] and others.

Chapter 4

Numerical Methods

In the following chapter numerical methods are presented which are helpful for the comparison of model predictions with experimental data. In order to achieve the best possible agreement one often has to deal with involved dependencies between observable quantities and the theory underlying model parameters. In such cases, especially when there is a large number of parameters, one is restricted to numerical tools to find the parameter values that provide a good match.

4.1 A figure of merit function

The numerical analysis is based on the concept of a figure of merit function χ^2, as previously described and applied in [75, 76, 77, 78, 79, 80, 81]. This function depending on the model parameters quantifies the agreement or discrepancy between theoretical predictions and experimental data values with a single number ≥ 0. Consider the following typical case: The measurement results for several obervables O_i are given in the form of mean value $\bar{O}_i$ and the corresponding 1σ standard deviation.

$$O_i = \bar{O}_i \pm \sigma_i \tag{4.1}$$

From theory we know how to calculate the predictions $P_i(\mathbf{x})$ for these observables from the model parameters $\mathbf{x}$, where $\mathbf{x} = (x_1, x_2, \ldots)$ contains all free parameters x_k of the model. Obviously for N observables

$$\chi^2(\mathbf{x}) := \sum_{i=1}^{N} \left(\frac{P_i(\mathbf{x}) - \bar{O}_i}{\sigma_i} \right)^2 \tag{4.2}$$

is a useful definition of the χ^2 function, since it provides the best fit values $P_i(\mathbf{x}_{\min})$ at the global minimum

$$\chi^2_{\min} := \chi^2(\mathbf{x}_{\min}) \quad \text{with} \quad 0 \leq \chi^2_{\min} \leq \chi^2(\mathbf{x}) \quad \forall \mathbf{x}. \tag{4.3}$$

Once the global minimum is found, one often denotes the difference between prediction and data for every observable with the "pull"

$$\text{pull}(O_i) = \frac{P_i(\mathbf{x}_{\min}) - \bar{O}_i}{\sigma_i}. \tag{4.4}$$

With the figure of merit function the task of comparing model predictions to experimental data is transformed into a search for the global minimum of χ^2. A lot of numerical tools are known to cope with this standard minimization problem. A good choice for our case is the Nelder-Mead method, see [75], p.19 for details on pros and cons.

4.2 The Nelder-Mead method

The Nelder-Mead method (NMM) [82, 83, 84, 85] or downhill simplex method for minimization problems can be categorized as a direct search method [86]. It does not require the computation of derivatives and therefore is even applicable if the function is not differentiable or continuous. The procedure is based on the movement of a geometric figure, the simplex, in the n-dimensional parameter space $\mathbb{R}^n$ towards a (local) minimum of the scalar function of interest $f(\mathbf{x})$. The simplex is best described as the convex hull of $n+1$ vertices $\mathbf{x}_i$, e.g. for $n = 2$ the simplex is a triangle. The movement of the simplex is performed stepwise, at each iteration step the function values at the vertices determine the position and shape of the next simplex in the series from the starting point to the minimum.

We will now discuss in detail the constituent branchings and calculations of a single Nelder-Mead iteration step. The procedure requires the definition of four parameters ρ, χ, γ and σ corresponding to different actions on the simplex, namely reflection, expansion, contraction and shrinkage, respectively. We employ the standard choice[1]

$$\begin{aligned} \rho &= 1, \\ \chi &= 2, \\ \gamma &= \sigma = 1/2. \end{aligned} \tag{4.5}$$

The following enumeration shows all ingredients of one Nelder-Mead iteration step.

[1] Allowed values are: $\rho > 0$, $\chi > 1$, $\chi > \rho$, $0 < \gamma < 1$ and $0 < \sigma < 1$.

1. **Start of the iteration step**

   The simplex containing the vertices $\mathbf{x}_i$ and the function values $f_i := f(\mathbf{x}_i)$ are taken from the last iteration (or the initial values).

2. **Sorting**

   The vertices are sorted according to increasing function values, i.e. after sorting the vertices fulfill

   $$f_1 \leq \ldots \leq f_n \leq f_{n+1}. \tag{4.6}$$

3. **Calculate Centroid**

   The "mean value" of the n best vertices, the centroid, is calculated:

   $$\overline{\mathbf{x}} := \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i. \tag{4.7}$$

4. **Reflection point**

   The reflection point $\mathbf{x}_\mathrm{r}$ is calculated:

   $$\mathbf{x}_\mathrm{r} := \overline{\mathbf{x}} + \rho(\overline{\mathbf{x}} - \mathbf{x}_{n+1}). \tag{4.8}$$

   If $f_1 \leq f_\mathrm{r} < f_n$ then $\mathbf{x}_\mathrm{r}$ is accepted, i.e. $\mathbf{x}_{n+1}$ is replaced by $\mathbf{x}_r$ (Iteration step terminated).

5. **Expansion Point**

   If $f_\mathrm{r} < f_1$ then calculate the expansion point

   $$\mathbf{x}_\mathrm{e} := \overline{\mathbf{x}} + \chi(\mathbf{x}_\mathrm{r} - \overline{\mathbf{x}}). \tag{4.9}$$

   If $f_\mathrm{e} < f_\mathrm{r}$ then $\mathbf{x}_\mathrm{e}$ is accepted (Iteration step terminated).

   If $f_\mathrm{e} \geq f_\mathrm{r}$ then $\mathbf{x}_\mathrm{r}$ is accepted (Iteration step terminated).

6. **Outside Contraction**

   If $f_n \leq f_\mathrm{r} < f_{n+1}$, an outside contraction is performed:

   $$\mathbf{x}_\mathrm{oc} := \overline{\mathbf{x}} + \gamma(\mathbf{x}_\mathrm{r} - \overline{\mathbf{x}}). \tag{4.10}$$

   If $f_\mathrm{oc} \leq f_\mathrm{r}$ then $\mathbf{x}_\mathrm{oc}$ is accepted (Iteration step terminated).

   Otherwise a shrinkage is performed.

7. **Inside Contraction**

   If $f_\mathrm{r} \geq f_{n+1}$, an outside contraction is performed:

   $$\mathbf{x}_\mathrm{ic} := \overline{\mathbf{x}} - \gamma(\overline{\mathbf{x}} - \mathbf{x}_{n+1}). \tag{4.11}$$

   If $f_\mathrm{ic} < f_{n+1}$ then $\mathbf{x}_\mathrm{ic}$ is accepted (Iteration step terminated).

   Otherwise a shrinkage is performed.

8. **Shrinkage**

   If either outside or inside contraction fails, then all vertices are contracted towards the best point $\mathbf{x}_1$:

   $$\mathbf{x}_i \to \mathbf{x}_1 + \sigma(\mathbf{x}_i - \mathbf{x}_1), \quad i = 2, \ldots, n+1. \tag{4.12}$$

   Then the iteration step is terminated.

Sorting (2.), calculating the centroid (3.) and the reflection point (4.) are always performed, while the steps (5.) to (8.) are optional and depend on the function value of the reflection point f_r. The sequence of decisions and calculations becomes clearer with the flow chart of the NMM, provided in Figure 4.1. An illustration of the different actions on the simplex in two dimensions is given in Figure 4.2. Obviously the reflection step is the least time-consuming possibility, since it requires only one function evaluation per iteration.[2] Expansion and both contraction options demand a second function call and shrinkage enforces the recalculation of $n + 2$ function values. Fortunately in a typical scenario the shrinkage step is executed repeatedly only at the end of the search, when the simplex is contracted towards the minimum. The basic step during the procedure is the reflection, while expansion and contractions allow the simplex to adapt its geometrical shape to the surrounding function shape. In this way the NMM can cope with rather complex function landscapes, e.g. where the same relative change of parameters causes a deviation of the function at different orders of magnitude. In conclusion the major advantages of the NMM in our search for the mimimum of χ^2 are the low number of function evaluations and the ability to adopt to a complex topology.

The NMM described so far lacks a procedure to stop the simplex movement when no further optimization is made. In this case the function values differ only slightly at all vertices. Thus, a good criterion to halt the iteration is the variance of the functions values

$$\frac{1}{n+1}\sum_{i=1}^{n+1}(f_i - \bar{f})^2 \leq \varepsilon \quad \text{with} \quad \bar{f} := \frac{1}{n+1}\sum_{i=1}^{n+1} f_i. \tag{4.13}$$

[2]Note that every function evaluation is probably computationally intensive, because it may include singular value decomposition, diagonalization, etc.

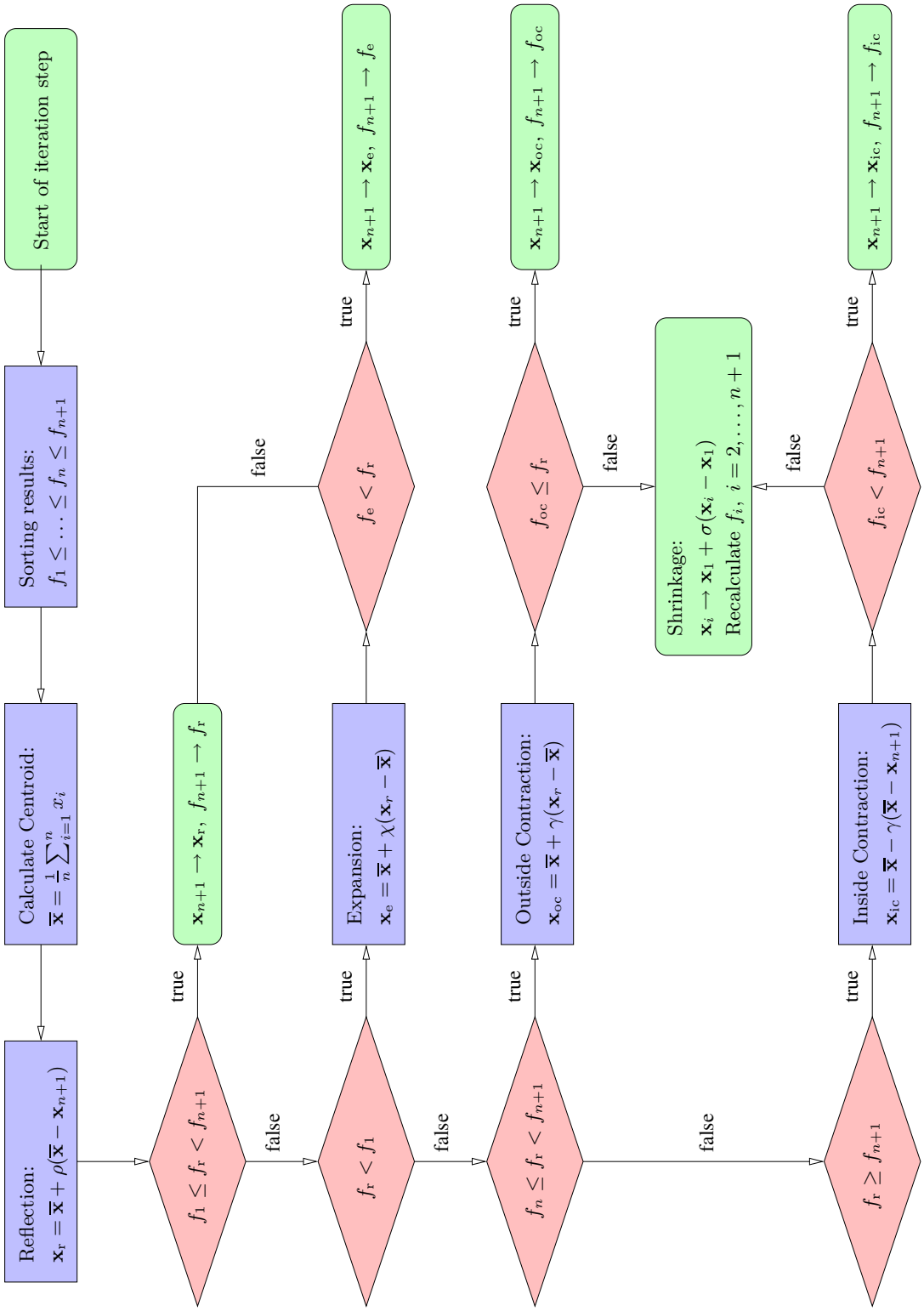


Figure 4.1: Flowchart for one Nelder-Mead iteration step.

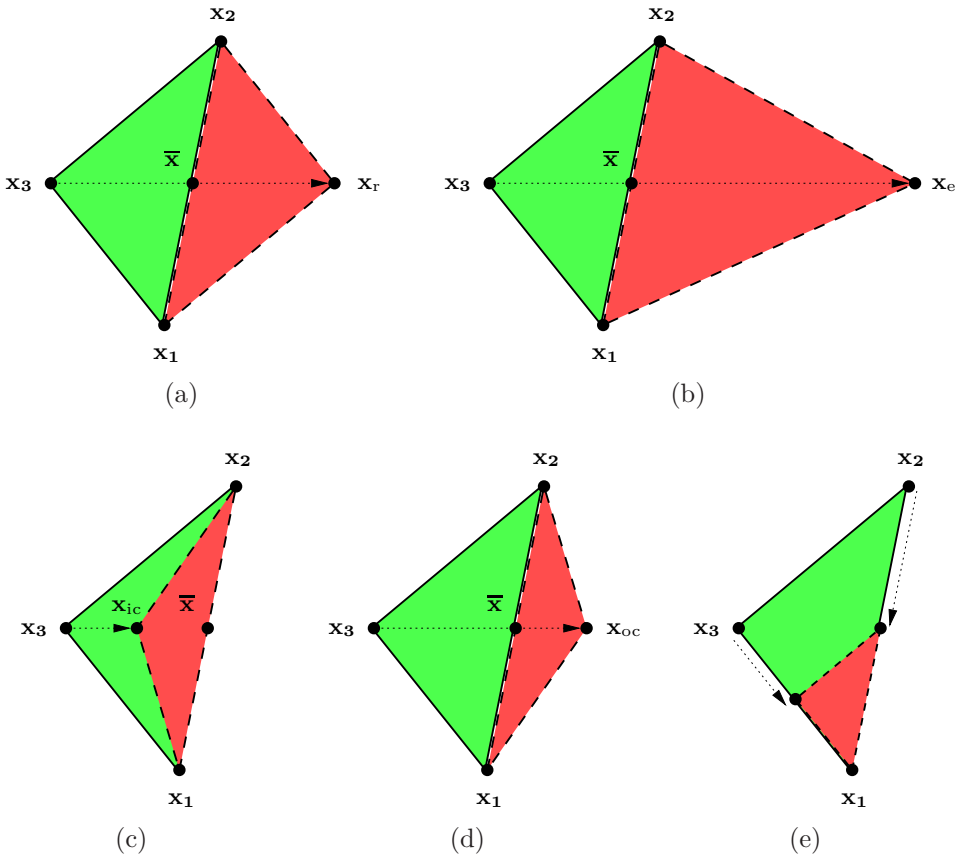


Figure 4.2: Different actions on the Nelder-Mead simplex in two dimensions after the sorting step: (a) reflection, (b) expansion, (c) inside contraction, (d) outside contraction and (e) shrinkage. The green filling denotes the simplex before, the red one after the iteration step.

If the variance falls below the predefined accuracy parameter ε the NMM is stopped and the minimum reached.

One major disadvantage of the NMM should not be overlooked: as the alternative name downhill simplex method suggests, there are no measures that prevent the simplex from getting stuck in a local minimum. The simplex moves strictly downhill and may never reach the global minimum. Therefore several extensions of the NMM with other computational methods were employed by [75, 85]. Among these hybrid algorithms are the NMM plus Simulated Annealing or NMM plus Perturbations. Both procedures were also coded for testing purposes but turned out not to be advantageous in view of the models of chapter 5. Another simple method to reach the global minimum is to repeat the Nelder-Mead procedure with different

initial simplices, e.g. set by a random number generator as implemented in the presented program.

4.3 Pinning term and fine-tuning properties

Besides the so far presented normal operation of the NMM, two supplemental techniques are employed to investigate models: the addition of a so called pinning term to the χ^2-function and an analysis of the fine-tuning properties.

Pinning term

The pinning term method is a helpful tool to elaborate the dependency between χ^2 and any observable O of interest. Consider the following extension of (4.2)

$$\chi^2(\mathbf{x}) := \sum_{i=1}^{N} \left(\frac{P_i(\mathbf{x}) - \bar{O}_i}{\sigma_i} \right)^2 + \underbrace{\left(\frac{P(\mathbf{x}) - \bar{O}}{0.01\bar{O}} \right)^2}_{\chi_p^2}, \tag{4.14}$$

where $\bar{O}$ is the numerical value and P is the theoretical prediction for the observable O. The additional term χ_p^2 pins O down to the value $\bar{O}$ in a minimization procedure because of the assigned small 1%-error. This allows to check how low χ^2 can get while incorporating additional constraints on observables. Repeated application of the pinning term method with a whole set of values gives the possibility to plot χ^2 against O. The procedure requires that the pinning term itself gives a negligible contribution to χ. Also, if O is identical to any of the other observables O_i, the original term with σ_i is removed.

Fine-tuning properties

If the search for a global minimum of χ^2 suceeds, one can analyse the fine-tuning properties of the best-fit solution in a very simple way. While all other parameters are fixed at the values corresponding to the minimum, one variable x_i is slightly changed by a factor ξ. The effect on the χ^2-function

$$\chi^2(\xi) = \chi^2(x_1^{\min}, \ldots, \xi x_i^{\min}, \ldots, x_n^{\min}) \tag{4.15}$$

can be plotted easily. Comparing the graphs of all parameters one can extract robust and fine-tuned variables, although a generalization of the properties from one sample solution is not allowed.

4.4 Implementation

The source code for the NMM is written in the *Fortran* programming language and compiled with the *g95 Fortran* compiler.[3] No additional libraries or code segments from other authors are used, the program is "made from scratch". It is divided into two separate files, `nmm.f90` for the iteration itself and `dnmm.f90` for the data analysis. In order to achieve the best possible accuracy all floating point numbers are defined as 10-byte real which leads to the following smallest and largest possible numbers.

```
tiny(fn) = 3.3621031431120935063E-4932
huge(fn) = 1.1897314953572317650E+4932
```

Thus, function values and simplex vertices are calculated and saved with at least 18-digit precision. Next, the content of the source files is roughly described.

`nmm.f90`

This is the main program file, which contains the placement of initial simplices, the Nelder-Mead iteration and an outer loop for repetition. The minima found are saved in an array and finally written to a binary file. The source code contains two *Fortran* modules, one for the global variables and one for the following subroutines:

- `init()`

  Asks the user for the number of repetitions and allocates memory.

- `initf()`

  Calculates model constants required for the evaluation of the χ^2 function.

- `prepf()`

  Sets a random simplex and calls `calcf` to compute χ^2 at each vertex.

- `calcf(a,b,sres)`

  This subroutine contains most of the model depending information. It takes coordinates in parameter space as input, calculates the predictions for the observables and returns χ^2. If `sres` is true, intermediate data is saved into an additional array.

[3] The source code is presented in Appendix A.

- `sort()`

  Sorts the vertices according to their function value, i.e. step (2) of the Nelder-Mead procedure.

- `centroid()`

  Calculates the centroid of a given simplex, step (3) of the NMM.

- `shrinkage()`

  Performs a shrinkage of the simplex as described in step (8).

- `calcd()`

  Computes the variance of the function values for the stopping criterion.

- `wfile()`

  Writes a binary file with all found minima, the corresponding model parameters and important observables. Figure 4.3 shows the arrangement of the data inside the array saved to disk.

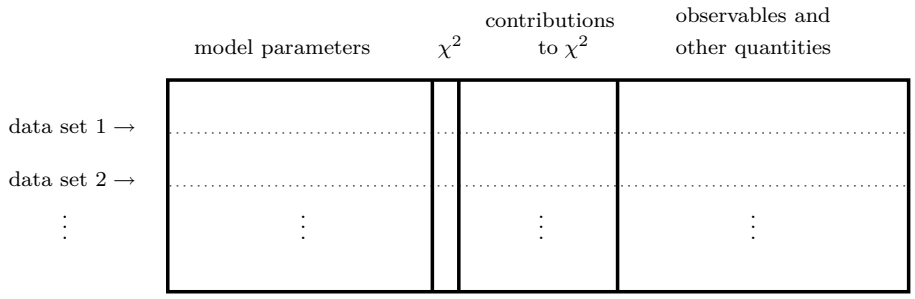


Figure 4.3: pattern for the data arrangement. Each data set stands for a minimum found by the NMM.

Besides the model depending constants and parameters, the user has to set two important values: `eps` is the accuracy parameter ε for the stopping routine and `crit` defines the maximum χ^2 allowed for a data set (model parameters, χ^2 contributions and observables) to be saved and further processed.

As previously indicated the present implementation tries to find the global minimum via repetition of the NMM with randomly set starting simplices. The user enters the desired number of turns at the beginning, every 1000 a short status information is displayed. Each turn contains a search for a minimum via Nelder-Mead, two additional stopping criteria are implemeted. First, if no minimum is

found after 1000 iterations (i.e. simplex movements), the iteration is terminated as unsuccessful. Also, every 100 iterations the vertices are checked for infinite, undetermined or unreasonable values (`NaN, Inf`). After the last repetition all gathered data is written into the binary file `nmm.dat`, which is opened by the data analysis program.

`dnmm.f90`

This file contains routines to perform various tasks on the data supplied by `nmm.f90`. In doing so, no additional model-specific calculations (e.g. computing observables) are made. Therefore, the previously mentioned `calcf` is not included in the list of subroutines.

- `readin()`

  The content of `nmm.dat` is read into allocated memory.

- `remdup()`

  The data sets are searched for duplicate entries which are removed.

- `sort()`

  The data array is sorted in rising order of χ^2. Afterwards the best-fit data set is on the first position.

- `cmvsd()`

  Calculates mean value and standard deviation for model parameters and observables for possible later use.

- `wdata()`

  Writes the previously sorted data, all model constants and statistics into a formatted file named `nmm.txt`.

- `clist(lin,lout,lstart,l)`

  This subroutine accepts one column in Figure 4.3 as input. The interval from the smallest to the largest value thereof is divided into a user-specified number of bins. The number of entries per bin is computed and thus histogram data for plotting created.

- `wlist()`

  Data from all calls of the `clist` subroutine is written to a file named `list.txt`, which can be easily used for plotting results.

Pinning term method implementation

The pinning term method presented in this chapter requires some changes in the source code. Based on the original files the major modifications comprise:

- An additional loop encloses the program to run through a list of values belonging to the desired pinned-down observable.
- The pinning term χ_p^2 in (4.14) is added in the `calcf` routine.
- Since there are now multiple output files, they are numbered and saved into a subfolder. The sorting in rising order of χ^2 is now carried out in the main program.
- The data analysis does not create any histogram data. Instead it picks out the best minima for each pinning value respecting that the pinning term itself should be negligible. The desired output, χ^2 as a function of the pinned observable, is saved into a file named `pin.txt`.

Analysing Fine-tuning properties

In order to carry out the procedure described in 4.3 the output file of `dnmm.f90` is opened with *Mathematica*. The model parameters for the best-fit solution are read in and entered into the χ^2 function. A combined plot shows then the variation of χ^2 as a function of ξ for all parameters.

The *Mathematica* script presented in A.3 serves another important purpose. The χ^2 function is not copied from the *Fortran* program code, but derived from simple expressions taken from the investigated model. Thus, if the results match the correctness of the main program is supported, typing errors are almost excluded.

Chapter 5

Models

In the following chapter a model for tri-bimaximal mixing and its modifications are described. These will undergo the numerical analysis of chapter 4 and the results are presented in the next chapter. The basic model was found by Grimus and Lavoura [1] and makes use of five right-handed neutrinos and the seesaw mechanism. Together with symmetries in the lepton sector, tri-bimaximal mixing is achieved. A slight modification concerning the symmetries allows to unfix the mixing angles and gives the possibility to find better fitting values for current data.

5.1 Model for tri-bimaximal mixing

5.1.1 Field content

The model for tri-bimaximal mixing found by Grimus and Lavoura [1] is an extension of the Standard model, both in the lepton and the scalar sector. Additionally the gauge group $SU(2)_I \times U(1)_Y$ is complemented by symmetries ensuring the desired form of the neutrino mass matrix obtained via the seesaw mechanism. Neglecting the quark sector, the field content is as follows:

- 3 left-handed lepton doublets

$$D_{eL} = \begin{pmatrix} \nu_{eL} \\ e_L \end{pmatrix} \quad , \quad D_{\mu L} = \begin{pmatrix} \nu_{\mu L} \\ \mu_L \end{pmatrix} \quad , \quad D_{\tau L} = \begin{pmatrix} \nu_{\tau L} \\ \tau_L \end{pmatrix}, \tag{5.1}$$

- 3 charged right-handed lepton singlets

$$e_R \ , \ \mu_R \ , \ \tau_R \ , \tag{5.2}$$

- 5 right-handed neutrino singlets

$$\nu_{eR} \ , \ \nu_{\mu R} \ , \ \nu_{\tau R} \ , \ \nu_{1R} \ , \ \nu_{2R}, \tag{5.3}$$

- 1 complex scalar singlet

$$\chi, \tag{5.4}$$

- 4 scalar Higgs doublets

$$\Phi_e = \begin{pmatrix} \phi_e^+ \\ \phi_e^0 \end{pmatrix} \ , \ \Phi_\mu = \begin{pmatrix} \phi_\mu^+ \\ \phi_\mu^0 \end{pmatrix} \ , \ \Phi_\tau = \begin{pmatrix} \phi_\tau^+ \\ \phi_\tau^0 \end{pmatrix} \ , \ \Phi_0 = \begin{pmatrix} \phi_0^+ \\ \phi_0^0 \end{pmatrix}. \tag{5.5}$$

The doublets $D_{\alpha L}$ and the singlets α_R ($\alpha = e, \mu, \tau$) are analogous to those in the Standard model. Five right-handed neutrino singlets $\nu_{\alpha R}$ and ν_{lR} ($l = 1, 2$) are added and give rise to various neutrino mass terms described in 5.1.3. The Yukawa Lagrangian is also enlarged by the four Higgs doublets Φ_α, Φ_0 and one scalar singlet χ in the model.

5.1.2 Symmetries

Besides the $SU(2)_I \times U(1)_Y$ gauge group there are three additional symmetries employed in the model.

- S_3 **permutation symmetry**
  S_3 is the group of all permutations of three objects, in our case the three flavour indices e, μ and τ. The $3! = 6$ elements of S_3 may be expressed in the two row notation

$$S_3 = \left\{ \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix} \right\}, \tag{5.6}$$

  or in the cycle notation

$$S_3 = \{(), (12), (23), (13), (123), (132)\}. \tag{5.7}$$

To construct all elements of the permutation group we have to choose two non-commuting elements as generators. A convenient choice is the cyclic permutation (123) and the interchange (23). The fields contained in the model are now arranged in multiplets with respect to S_3. There are four triplets

$$\begin{pmatrix} D_{eL} \\ D_{\mu L} \\ D_{\tau L} \end{pmatrix} \ , \ \begin{pmatrix} e_R \\ \mu_R \\ \tau_R \end{pmatrix} \ , \ \begin{pmatrix} \nu_{eR} \\ \nu_{\mu R} \\ \nu_{\tau R} \end{pmatrix} \ , \ \begin{pmatrix} \Phi_e \\ \Phi_\mu \\ \Phi_\tau \end{pmatrix} \tag{5.8}$$

together with the three-dimensional representations of the cyclic transformation C and the $\mu\tau$ interchange I

$$C_{e\mu\tau} \mapsto \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \ , \ I_{\mu\tau} \mapsto \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} . \tag{5.9}$$

The three-dimensional representation of S_3 is reducible, while the two-dimensional representation for the doublets

$$\begin{pmatrix} \nu_{1R} \\ \nu_{2R} \end{pmatrix} \ , \ \begin{pmatrix} \chi \\ \chi^* \end{pmatrix} \tag{5.10}$$

is irreducible:

$$C_{e\mu\tau} \mapsto \begin{pmatrix} \omega & 0 \\ 0 & \omega^2 \end{pmatrix} \ , \ I_{\mu\tau} \mapsto \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} , \tag{5.11}$$

where $\omega := e^{\frac{2\pi i}{3}}$. The $SU(2)_I$ Higgs doublet Φ_0 is a singlet with respect to S_3. The horizontal permutation symmetry is essential for the form of the mass matrix and therefore for tri-bimaximal mixing.

- **Three $U(1)_{L_\alpha}$ lepton family symmetries**
  Eeach of the lepton families is assigned a $U(1)$ symmetry:

$$U(1)_{L_\alpha} : \quad \left\{ \begin{array}{rcl} D_{\alpha L} & \to & e^{i\psi_\alpha} D_{\alpha L} \\ \alpha_R & \to & e^{i\psi_\alpha} \alpha_R \\ \nu_{\alpha R} & \to & e^{i\psi_\alpha} \nu_{\alpha R} \end{array} \right. \tag{5.12}$$

  with $\psi \in [0, 2\pi[$. Thus the family lepton number $L_\alpha = 1$ for $D_{\alpha L}, \alpha_R$ and $\nu_{\alpha R}$ and $L_\alpha = 0$ for all other fields.

- **Three $\mathbb{Z}_2^{(\alpha)}$ symmetries**
  The cyclic groups $\mathbb{Z}_2^{(\alpha)}$ concerns only the right-handed charged leptons α_R and the Higgs doublets Φ_α ($\alpha = e, \mu$ and τ):

$$\mathbb{Z}_2^{(\alpha)} : \quad \left\{ \begin{array}{rcl} \alpha_R & \to & -\alpha_R \\ \Phi_\alpha & \to & -\Phi_\alpha \end{array} \right. \tag{5.13}$$

This symmetry is important to restrict certain couplings in the Yukawa Lagrangian.

The whole group structure aside from $SU(2)_I \times U(1)_Y$ may be written in a compact form as a semidirect product

$$G = (N \times H) \rtimes S_3 \tag{5.14}$$

where

$$N = \mathbb{Z}_2^{(e)} \times \mathbb{Z}_2^{(\mu)} \times \mathbb{Z}_2^{(\tau)} \ , \ H = U(1)_{L_e} \times U(1)_{L_\mu} \times U(1)_{L_\tau}. \tag{5.15}$$

5.1.3 Lagrangian

With the given fields and symmetries one has to find all allowed terms of the Lagrangian. The Yukawa Lagrangian

$$\begin{aligned}
\mathcal{L}_{\text{Yukawa}} = &- y_1 \sum_{\alpha=e,\mu,\tau} \overline{D}_{\alpha L} \alpha_R \phi_\alpha && \text{(5.16a)} \\
&- y_2 \sum_{\alpha=e,\mu,\tau} \overline{D}_{\alpha L} \nu_{\alpha R} \left(i\tau_2 \phi_0^*\right) && \text{(5.16b)} \\
&+ \frac{y_3}{2} \left(\chi \nu_{1R}^T C^{-1} \nu_{1R} + \chi^* \nu_{2R}^T C^{-1} \nu_{2R}\right) + \text{H.c.} && \text{(5.16c)}
\end{aligned}$$

introduces three coupling constants y_1, y_2, y_3 and respects all symmetries described. Spontaneous symmetry breaking leads to VEVs of the Higgs fields

$$\begin{aligned}
v_\alpha &= \langle \phi_\alpha^0 \rangle_0 && \text{(5.17a)} \\
v_0 &= \langle \phi_0^0 \rangle_0 && \text{(5.17b)} \\
v_\chi &= \langle \phi_\chi^0 \rangle_0, && \text{(5.17c)}
\end{aligned}$$

which are employed to form various mass terms, e.g. for the masses of the charged leptons

$$m_\alpha = |y_1 v_\alpha|. \tag{5.18}$$

In addition to the Dirac mass terms of (5.16) the symmetries also allow dimension-three Majorana mass terms for the right-handed neutrino fields which respect the S_3 symmetry. The three $U(1)_{L_\alpha}$ symmetries on the other hand are softly broken

at the seesaw scale [87, 88, 89].

$$\begin{aligned}
\mathcal{L}_{\text{Majorana}} =& \frac{M_0^*}{2} \sum_{\alpha=e,\mu,\tau} \nu_{\alpha R}^T C^{-1} \nu_{\alpha R} && \text{(5.19a)}\\
&+ M_1^* \left(\nu_{eR}^T C^{-1} \nu_{\mu R} + \nu_{\mu R}^T C^{-1} \nu_{\tau R} + \nu_{\tau R}^T C^{-1} \nu_{eR} \right) && \text{(5.19b)}\\
&+ M_2^* \Big[\nu_{1R}^T C^{-1} \left(\nu_{eR} + \omega \nu_{\mu R} + \omega^2 \nu_{\tau R} \right) \\
&+ \nu_{2R}^T C^{-1} \left(\nu_{eR} + \omega^2 \nu_{\mu R} + \omega \nu_{\tau R} \right) \Big] && \text{(5.19c)}\\
&+ M_4^* \nu_{1R}^T C^{-1} \nu_{2R} + \text{H.c.} && \text{(5.19d)}
\end{aligned}$$

For a detailed view on the scalar potential and the symmetry breaking therein, see [1]. Putting all mass terms together to prepare the application of the seesaw mechanism we find

$$\begin{aligned}
&- \begin{pmatrix} \overline{\nu}_{eR} & \overline{\nu}_{\mu R} & \overline{\nu}_{\tau R} & \overline{\nu}_{1R} & \overline{\nu}_{2R} \end{pmatrix} M_D \begin{pmatrix} \nu_{eL} \\ \nu_{\mu L} \\ \nu_{\tau L} \end{pmatrix} \\
&\qquad - \frac{1}{2} \begin{pmatrix} \overline{\nu}_{eR} & \overline{\nu}_{\mu R} & \overline{\nu}_{\tau R} & \overline{\nu}_{1R} & \overline{\nu}_{2R} \end{pmatrix} M_R C \begin{pmatrix} \overline{\nu}_{eR}^T \\ \overline{\nu}_{\mu R}^T \\ \overline{\nu}_{\tau R}^T \\ \overline{\nu}_{1R}^T \\ \overline{\nu}_{2R}^T \end{pmatrix} + \text{H.c.} \qquad \text{(5.20)}
\end{aligned}$$

with the Dirac- and Majorana mass matrices

$$M_D = \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \;,\; M_R = \begin{pmatrix} M_0 & M_1 & M_1 & M_2 & M_2 \\ M_1 & M_0 & M_1 & \omega^2 M_2 & \omega M_2 \\ M_1 & M_1 & M_0 & \omega M_2 & \omega^2 M_2 \\ M_2 & \omega^2 M_2 & \omega M_2 & M_N & M_4 \\ M_2 & \omega M_2 & \omega^2 M_2 & M_4 & M_N' \end{pmatrix}, \qquad \text{(5.21)}$$

where $a := y_2^* v_0$, $M_N := y_3^* v_\chi^*$ and $M_N' := y_3^* v_\chi$. According to the seesaw mechanism described in 2.2.3, we compute the light-neutrino mass matrix as

$$\mathcal{M}_\nu = -M_D^T M_R^{-1} M_D = \begin{pmatrix} x+y+t & z+\omega^2 y+\omega t & z+\omega y+\omega^2 t \\ z+\omega^2 y+\omega t & x+\omega y+\omega^2 t & z+y+t \\ z+\omega y+\omega^2 t & z+y+t & x+\omega^2 y+\omega t \end{pmatrix}. \qquad \text{(5.22)}$$

In order to keep focus on the basic form of $\mathcal{M}_\nu$ we introduced abbreviations

$$x := \frac{-a^2}{\det M_R}\left[\left(M_0^2 - M_1^2\right)\left(M_N M_N' - M_4^2\right) + \left(4M_0 + 2M_1\right) M_2^2 M_4 - 3M_2^4\right], \tag{5.23a}$$

$$z := \frac{-a^2}{\det M_R}\left[\left(M_1^2 - M_0 M_1\right)\left(M_N M_N' - M_4^2\right) + \left(M_0 - 4M_1\right) M_2^2 M_4 - 3M_2^4\right], \tag{5.23b}$$

$$y := \frac{-a^2}{\det M_R}\left(M_0 + 2M_1\right) M_2^2 M_N', \tag{5.23c}$$

$$t := \frac{-a^2}{\det M_R}\left(M_0 + 2M_1\right) M_2^2 M_N \tag{5.23d}$$

and

$$\det M_R = \left(M_0 + 2M_1\right)\left\{\left(M_0 - M_1\right)^2 M_N M_N' - \left[\left(M_0 - M_1\right) M_4 - 3M_2^2\right]^2\right\}. \tag{5.23e}$$

Obviously y and t fulfill

$$\frac{y}{t} = \frac{M_N'}{M_N} = \frac{v_\chi}{v_\chi^*}. \tag{5.24}$$

As shown in [1] there is a range of parameters of the scalar potential upon SSB for which the symmetry $I_{\mu\nu}$ is preserved. Thus, v_χ is real and $y = t$ holds.

$$\begin{aligned} & \frac{y_3}{2}\left(\chi \nu_{1R}^T C^{-1} \nu_{1R} + \chi^* \nu_{2R}^T C^{-1} \nu_{2R}\right) + \text{H.c.} \\ \xrightarrow{SSB} \quad & \frac{y_3}{2}\left(v_\chi \nu_{1R}^T C^{-1} \nu_{1R} + v_\chi^* \nu_{2R}^T C^{-1} \nu_{2R}\right) + \text{H.c.} \\ \xrightarrow{I_{\mu\nu}} \quad & \frac{y_3}{2}\left(v_\chi \nu_{2R}^T C^{-1} \nu_{2R} + v_\chi^* \nu_{1R}^T C^{-1} \nu_{1R}\right) + \text{H.c.} \end{aligned} \tag{5.25}$$

Together with a property of the cubic root $\omega^2 = \omega^*$ we can simplify (5.22) and achieve

$$\mathcal{M}_\nu = \begin{pmatrix} x + 2y & z - y & z - y \\ z - y & x - y & z + 2y \\ z - y & z + 2y & x - y \end{pmatrix}. \tag{5.26}$$

The tri-bimaximal mixing matrix U_{HPS} (3.30) diagonalizes $\mathcal{M}_\nu$ as desired:

$$\hat{m} = U_{\text{HPS}}^T \mathcal{M}_\nu U_{\text{HPS}} = \begin{pmatrix} \mu_1 & 0 & 0 \\ 0 & \mu_2 & 0 \\ 0 & 0 & \mu_3 \end{pmatrix}, \tag{5.27}$$

where

$$\mu_1 = x + 3y - z \tag{5.28a}$$
$$\mu_2 = x + 2z \tag{5.28b}$$
$$\mu_3 = x - 3y - z. \tag{5.28c}$$

Since in general x, y, z and thus the μ_i are complex, we have to take the absolute values to get the light neutrino masses

$$m_i = |\mu_i|, \quad (i = 1, 2, 3) \tag{5.29}$$

This step is justified because any phases occuring in $\hat{m}$ can be absorbed into Majorana phases or by rephasing of the charged lepton fields. Consider the case

$$\mu_i \in \mathbb{C}, \quad \mu_i = m_i \cdot e^{i\alpha_i}. \tag{5.30}$$

Then from (5.27) we deduce

$$\tilde{m} := U^T \mathcal{M}_\nu U = \begin{pmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{pmatrix} \tag{5.31}$$

with the help of the phase matrix P

$$U := U_{\text{HPS}} P = U_{\text{HPS}} \begin{pmatrix} e^{-i\alpha_1/2} & 0 & 0 \\ 0 & e^{-i\alpha_2/2} & 0 \\ 0 & 0 & e^{-i\alpha_3/2} \end{pmatrix} = e^{i\beta} U_{\text{HPS}} \begin{pmatrix} e^{i\eta_1} & 0 & 0 \\ 0 & e^{i\eta_2} & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{5.32}$$

where

$$\eta_1 := \frac{\alpha_3 - \alpha_1}{2}, \quad \eta_2 := \frac{\alpha_3 - \alpha_2}{2}, \quad \beta := -\frac{\alpha_3}{2}. \tag{5.33}$$

The phase factor $e^{i\beta}$ turns out to be unphysical because it may be absorbed into the lepton fields in (2.91), while η_1 and η_2 act as Majorana phases in the mixing matrix.

5.2 Modifications

Alongside with the basic model for tri-bimaximal mixing, two modifications are described in this section. One motivation for the changes is the reduction of parameters that have to be fitted to experimental data. The described model contains five complex parameters relevant for the numerical procedure but only two known data values, i.e. the mass squared differences. To include also the mixing angles in the fitting process an additional symmetry breaking is included, giving rise to a possible non-zero ϑ_{13}. In short, the modifications applied are:

- CP **invariance**
  Stating an additional CP symmetry renders the Yukawa couplings y_i, $i = 1, 2, 3$ and the constants M_j, $j = 0, 1, 2, 4$ real and thus reduces the number of free parameters of the model.

- $I_{\mu\tau}$**-violation**
  The $I_{\mu\nu}$ symmetry is now spontaneously broken by a complex VEV v_χ leading to a deviation from tri-bimaximal mixing. For a recent discussion see also [90].

5.2.1 CP invariance

The CP symmetry is realized via the non-standard CP tranformation introduced in [91] and previously applied in [92]. The doublets and singlets with different flavours are collected in the following form:

$$D_L = \begin{pmatrix} D_{eL} \\ D_{\mu L} \\ D_{\tau L} \end{pmatrix}, \quad l_R = \begin{pmatrix} e_R \\ \mu_R \\ \tau_R \end{pmatrix}, \quad \nu_R = \begin{pmatrix} \nu_{eR} \\ \nu_{\mu R} \\ \nu_{\tau R} \end{pmatrix}, \quad \Phi = \begin{pmatrix} \Phi_e \\ \Phi_\mu \\ \Phi_\tau \end{pmatrix}. \tag{5.34}$$

Introducing the matrix

$$S = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \tag{5.35}$$

that acts on flavour space as an interchange of the μ and τ components, we can write down the CP transformation for all fields.

$$CP: \quad \begin{cases} D_L \to iSCD_L^* \\ l_R \to iSCl_R^* \\ \nu_R \to iSC\nu_R^* \\ \phi \to S\phi^* \\ \nu_{sR} \to iC\nu_{sR}^* \quad s = 1, 2 \\ \chi \to \chi^* \end{cases} \tag{5.36}$$

C is the standard charge-conjugation operator satisfying $C^{-1} = C^\dagger$ and $C^T = -C$. We may now apply this CP transformation to the Lagrangian for the lepton sector and obtain constraints on the Yukawa-coupling constants and elements of the Majorana mass matrix M_R.

First we write down two general results, which will be useful for our calculations. The CP transformation acts on a dirac spinor as follows:

$$\psi \xrightarrow{CP} iC\psi^* \tag{5.37}$$

Using the identity $C^{-1}\gamma^\mu C = -\gamma^{\mu T}$ one obtains the CP transformation for the Dirac-conjugate field:

$$\overline{\psi} = \psi^\dagger\gamma^0 \xrightarrow{CP} (iC\psi^*)^\dagger\,\gamma^0 = -i\psi^T C^\dagger\gamma^0 = i\overline{\psi}^* C^\dagger\gamma^{0\dagger}\gamma^0 = i\overline{\psi}^* C^\dagger. \tag{5.38}$$

This of course also holds for doublets like $\overline{D}_{\alpha L}$. Next we investigate the effect of the CP transformation on an arbitrary Majorana mass term:

$$\begin{aligned}\psi_1^T C^{-1}\psi_2 \xrightarrow{CP} (iC\psi_1^*)^T\, C^{-1} iC\psi_2^* &= -\,\psi_1^\dagger C^T C^{-1} C\psi_2^* = \psi_1^\dagger C\psi_2^* \\ &= \psi_2^\dagger C\psi_1^* = \left(\psi_1^T C^{-1}\psi_2\right)^\dagger.\end{aligned} \tag{5.39}$$

Yukawa Lagrangian

We now apply CP transformation and demand that the Lagrangian stays invariant. For (5.16a) we simply have to use (5.37), (5.38) and then the anticommutation relations for fermions.

$$\begin{aligned}
& -\,y_1 \sum_{\alpha=e,\mu,\tau} \overline{D}_{\alpha L}\alpha_R\phi_\alpha + H.c. \\
\xrightarrow{CP} \quad & -\,y_1 \sum_{\alpha=e,\mu,\tau} i\overline{D}^*_{\alpha L} C^\dagger iC\alpha_R^*\phi_\alpha^* + H.c \\
= \quad & +\,y_1 \sum_{\alpha=e,\mu,\tau} \left(\overline{D}_{\alpha L}\alpha_R\phi_\alpha\right)^* + H.c \\
= \quad & +\,y_1 \sum_{\alpha=e,\mu,\tau} \left(\phi_\alpha^T\alpha_R^T\overline{D}_{\alpha L}^T\right)^\dagger + H.c. \\
= \quad & -\,y_1 \sum_{\alpha=e,\mu,\tau} \left(\overline{D}_{\alpha L}\alpha_R\phi_\alpha\right)^\dagger + H.c \\
\overset{!}{=} \quad & -\,y_1^* \sum_{\alpha=e,\mu,\tau} \left(\overline{D}_{\alpha L}\alpha_R\phi_\alpha\right)^\dagger + H.c
\end{aligned} \tag{5.40}$$

$$\Downarrow$$

$$y_1 = y_1^* \quad \Leftrightarrow \quad y_1 \in \mathbb{R} \tag{5.41}$$

The second Yukawa Lagrangian term (5.16b) requires the same treatment.

$$\begin{aligned}
& - y_2 \sum_{\alpha=e,\mu,\tau} \overline{D}_{\alpha L} \nu_{\alpha R} \left(i\tau_2 \phi_0^*\right) + H.c. \\
\xrightarrow{CP} \quad & - y_2 \sum_{\alpha=e,\mu,\tau} i\overline{D}^*_{\alpha L} C^\dagger i C \nu^*_{\alpha R} \left(i\tau_2 \phi_0\right) + H.c \\
= \quad & + y_2 \sum_{\alpha=e,\mu,\tau} \left[\overline{D}_{\alpha L} \nu_{\alpha R} \left(i\tau_2 \phi_0^*\right)\right]^* + H.c \\
= \quad & - y_2 \sum_{\alpha=e,\mu,\tau} \left[\overline{D}_{\alpha L} \nu_{\alpha R} \left(i\tau_2 \phi_0^*\right)\right]^\dagger + H.c \\
\overset{!}{=} \quad & - y_2^* \sum_{\alpha=e,\mu,\tau} \left[\overline{D}_{\alpha L} \nu_{\alpha R} \left(i\tau_2 \phi_0^*\right)\right]^\dagger + H.c
\end{aligned} \tag{5.42}$$

$$\Downarrow$$

$$y_2 = y_2^* \quad \Leftrightarrow \quad y_2 \in \mathbb{R} \tag{5.43}$$

Using the CP transformation relation for Majorana mass terms (5.39) one gets from the sterile neutrino Yukawa terms (5.16c).

$$\begin{aligned}
& \frac{y_3}{2} \left(\chi \nu_{1R}^T C^{-1} \nu_{1R} + \chi^* \nu_{2R}^T C^{-1} \nu_{2R}\right) + H.c. \\
\xrightarrow{CP} \quad & \frac{y_3}{2} \left[\chi^* \left(\nu_{1R}^T C^{-1} \nu_{1R}\right)^\dagger + \chi \left(\nu_{2R}^T C^{-1} \nu_{2R}\right)^\dagger\right] + H.c. \\
= \quad & \frac{y_3}{2} \left(\chi \nu_{1R}^T C^{-1} \nu_{1R} + \chi^* \nu_{2R}^T C^{-1} \nu_{2R}\right)^\dagger + H.c. \\
\overset{!}{=} \quad & \frac{{y_3}^*}{2} \left(\chi \nu_{1R}^T C^{-1} \nu_{1R} + \chi^* \nu_{2R}^T C^{-1} \nu_{2R}\right)^\dagger + H.c.
\end{aligned} \tag{5.44}$$

$$\Downarrow$$

$$y_3 = y_3^* \quad \Leftrightarrow \quad y_3 \in \mathbb{R} \tag{5.45}$$

As a result of CP invariance all three Yukawa coupling constants y_1, y_2 and y_3 must be real. Especially the constraint on y_3 is important for later calculations concerning the mass matrix.

Majorana-Lagrangian

The Majorana Lagrangian of the considered model consists of 4 terms which respect all allowed combinations of right handed neutrinos. The 4 constants M_i ($i = 0, 1, 2, 4$) corresponding to these terms are in general complex and will become real through CP invariance like the Yukawa coupling constants The first Majorana mass term (5.19a) is easily CP transformed via (5.39). The matrix S mixes μ and τ but since there is a summation over all flavour indices this effect

cancels out.

$$\begin{aligned} & \frac{M_0^*}{2} \sum_{\alpha=e,\mu,\tau} \nu_{\alpha R}^T C^{-1} \nu_{\alpha R} + H.c. \\ \xrightarrow{CP} \quad & \frac{M_0^*}{2} \sum_{\alpha=e,\mu,\tau} \left(\nu_{\alpha R}^T C^{-1} \nu_{\alpha R}\right)^\dagger + H.c. \\ \stackrel{!}{=} \quad & \frac{M_0}{2} \sum_{\alpha=e,\mu,\tau} \left(\nu_{\alpha R}^T C^{-1} \nu_{\alpha R}\right)^\dagger + H.c. \end{aligned} \tag{5.46}$$

$$\Downarrow$$

$$M_0 = M_0^* \quad \Leftrightarrow \quad M_0 \in \mathbb{R} \tag{5.47}$$

In order to show that M_1 in (5.19b) is real, we use the antisymmetry of the charge conjugation matrix C.

$$\begin{aligned} & M_1^* \left(\nu_{eR}^T C^{-1} \nu_{\mu R} + \nu_{\mu R}^T C^{-1} \nu_{\tau R} + \nu_{\tau R}^T C^{-1} \nu_{eR}\right) + H.c. \\ \xrightarrow{CP} \quad & M_1^* \left(\nu_{eR}^T C^{-1} \nu_{\tau R} + \nu_{\tau R}^T C^{-1} \nu_{\mu R} + \nu_{\mu R}^T C^{-1} \nu_{eR}\right)^\dagger + H.c. \\ = \quad & M_1^* \left(\nu_{\tau R}^T C^{-1} \nu_{eR} + \nu_{\mu R}^T C^{-1} \nu_{\tau R} + \nu_{eR}^T C^{-1} \nu_{\mu R}\right)^\dagger + H.c. \\ \stackrel{!}{=} \quad & M_1 \left(\nu_{eR}^T C^{-1} \nu_{\mu R} + \nu_{\mu R}^T C^{-1} \nu_{\tau R} + \nu_{\tau R}^T C^{-1} \nu_{eR}\right)^\dagger + H.c. \end{aligned} \tag{5.48}$$

$$\Downarrow$$

$$M_1 = M_1^* \quad \Leftrightarrow \quad M_1 \in \mathbb{R} \tag{5.49}$$

For the third term (5.19c) one has to utilise the special form of $\omega = e^{\frac{2\pi i}{3}}$. Since $\omega^* = \omega^2$ the $\mu\tau$-exchange is again canceled out and leads to a real M_2.

$$\begin{aligned} & M_2^* \Big[\nu_{1R}^T C^{-1} \left(\nu_{eR} + \omega \nu_{\mu R} + \omega^2 \nu_{\tau R}\right) \\ & \qquad + \nu_{2R}^T C^{-1} \left(\nu_{eR} + \omega^2 \nu_{\mu R} + \omega \nu_{\tau R}\right)\Big] + H.c. \\ \xrightarrow{CP} \quad & M_2^* \Big[\nu_{1R}^T C^{-1} \left(\nu_{eR} + \omega^* \nu_{\tau R} + (\omega^2)^* \nu_{\mu R}\right) \\ & \qquad + \nu_{2R}^T C^{-1} \left(\nu_{eR} + (\omega^2)^* \nu_{\tau R} + \omega^* \nu_{\mu R}\right)\Big]^\dagger + H.c. \\ \stackrel{!}{=} \quad & M_2 \Big[\nu_{1R}^T C^{-1} \left(\nu_{eR} + \omega \nu_{\mu R} + \omega^2 \nu_{\tau R}\right) \\ & \qquad + \nu_{2R}^T C^{-1} \left(\nu_{eR} + \omega^2 \nu_{\mu R} + \omega \nu_{\tau R}\right)\Big]^\dagger + H.c. \end{aligned} \tag{5.50}$$

$$\Downarrow$$

$$M_2 = M_2^* \quad \Leftrightarrow \quad M_2 \in \mathbb{R} \tag{5.51}$$

CP invariance is straightforward in the fourth term of the Majorana Lagrangian

(5.19d).

$$M_4^* \nu_{1R}^T C^{-1} \nu_{2R} + H.c. \xrightarrow{CP} M_4^* \left(\nu_{1R}^T C^{-1} \nu_{2R}\right)^\dagger + H.c. \stackrel{!}{=} M_4 \left(\nu_{1R}^T C^{-1} \nu_{2R}\right)^\dagger + H.c. \tag{5.52}$$

$$\Downarrow$$

$$M_4 = M_4^* \quad \Leftrightarrow \quad M_4 \in \mathbb{R} \tag{5.53}$$

Finally we have now obtained that the following model constants must be real in order to ensure CP invariance: y_i for $i = 1, 2, 3$ and M_j for $j = 0, 1, 2, 4$.

5.2.2 Spontaneous $I_{\mu\nu}$-violation in the neutrino sector

So far a real VEV v_χ ensured the permutation symmetry S_3 leading to $y = t$, see (5.25). We will now generalize [1] for a complex VEV and thus allow spontaneous symmetry breaking of $I_{\mu\nu}$ in the neutrino sector. This modification alters the lepton mixing matrix from tri-bimaximal mixing to trimaximal mixing.

To work out the differences to the model with tri-bimaximal mixing the complex v_χ is parametrized in the following form:

$$v_\chi = \widetilde{v_\chi} \cdot e^{i\varepsilon}, \quad \widetilde{v_\chi} \in \mathbb{R}, \quad \varepsilon \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]. \tag{5.54}$$

Note that this is *not* the regular polar decomposition since $\widetilde{v_\chi}$ may be positive or *negative*. y and t may be written as follows:

$$\begin{aligned} y &= \widetilde{y}\, e^{i\varepsilon} \\ t &= \widetilde{y}\, e^{-i\varepsilon} \\ \widetilde{y} &= \frac{-a^2}{\det M_R} \left(M_0 + 2M_1\right) M_2^2 y_3^* \widetilde{v_\chi}. \end{aligned} \tag{5.55}$$

Inserting this into the mass matrix gives:

$$\mathcal{M}_\nu = \begin{pmatrix} x + \widetilde{y}\left(e^{i\varepsilon} + e^{-i\varepsilon}\right) & z + \widetilde{y}\left(\omega^2 e^{i\varepsilon} + \omega e^{-i\varepsilon}\right) & z + \widetilde{y}\left(\omega e^{i\varepsilon} + \omega^2 e^{-i\varepsilon}\right) \\ z + \widetilde{y}\left(\omega^2 e^{i\varepsilon} + \omega e^{-i\varepsilon}\right) & x + \widetilde{y}\left(\omega e^{i\varepsilon} + \omega^2 e^{-i\varepsilon}\right) & z + \widetilde{y}\left(e^{i\varepsilon} + e^{-i\varepsilon}\right) \\ z + \widetilde{y}\left(\omega e^{i\varepsilon} + \omega^2 e^{-i\varepsilon}\right) & z + \widetilde{y}\left(e^{i\varepsilon} + e^{-i\varepsilon}\right) & x + \widetilde{y}\left(\omega^2 e^{i\varepsilon} + \omega e^{-i\varepsilon}\right) \end{pmatrix}. \tag{5.56}$$

The expressions in brackets can be written in terms of cosine, where $\phi = \frac{2\pi}{3}$ denotes the angle corresponding to ω:

$$\omega e^{i\varepsilon} + \omega^2 e^{-i\varepsilon} = 2\cos(\varepsilon + \phi), \tag{5.57}$$

$$\omega^2 e^{i\varepsilon} + \omega e^{-i\varepsilon} = 2\cos(\varepsilon - \phi). \tag{5.58}$$

This leads to the following form of the mass matrix which is now real except for $-a^2$ hidden in x, y, z and t (see (5.23)). $-a^2$ can be pulled out and written as an overall factor of the matrix. Thus, $\mathcal{M}_\nu$ is treated as real for diagonalization purposes.

$$\mathcal{M}_\nu = \begin{pmatrix} x + 2\widetilde{y}\cos(\varepsilon) & z + 2\widetilde{y}\cos(\varepsilon - \phi) & z + 2\widetilde{y}\cos(\varepsilon + \phi) \\ z + 2\widetilde{y}\cos(\varepsilon - \phi) & x + 2\widetilde{y}\cos(\varepsilon + \phi) & z + 2\widetilde{y}\cos(\varepsilon) \\ z + 2\widetilde{y}\cos(\varepsilon + \phi) & z + 2\widetilde{y}\cos(\varepsilon) & x + 2\widetilde{y}\cos(\varepsilon - \phi) \end{pmatrix} \tag{5.59}$$

We can try to find a real U by solving the eigenvalue diagonalization $U^T \mathcal{M}_\nu U = \mathrm{diag}\,(\mu_1, \mu_2, \mu_3)$. The eigenvalues are solutions of the following equation:

$$-\mu^3 + 3x\mu^2 + \left(3z^2 - 3x^2 + 9\widetilde{y}^2\right)\mu + 2z^3 + x^3 - 3xz^2 - 18z\widetilde{y}^2 - 9x\widetilde{y}^2 = 0. \tag{5.60}$$

It turns out that they are very similar to the expressions obtained in (5.28).

$$\mu_1 = x + 3\widetilde{y} - z = x + 3ye^{-i\varepsilon} - z \tag{5.61a}$$

$$\mu_2 = x + 2z \tag{5.61b}$$

$$\mu_3 = x - 3\widetilde{y} - z = x - 3ye^{-i\varepsilon} - z \tag{5.61c}$$

The corresponding mixing matrix is composed of the normalized eigenvectors.

$$U = \begin{pmatrix} \sqrt{\frac{2}{3}}\,\cos(\frac{\varepsilon}{2}) & \frac{1}{\sqrt{3}} & \sqrt{\frac{2}{3}}\,\sin(\frac{\varepsilon}{2}) \\ -\frac{1}{\sqrt{6}}\left(\cos(\frac{\varepsilon}{2}) - \sqrt{3}\sin(\frac{\varepsilon}{2})\right) & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{2}}\left(\cos(\frac{\varepsilon}{2}) + \frac{1}{\sqrt{3}}\sin(\frac{\varepsilon}{2})\right) \\ -\frac{1}{\sqrt{6}}\left(\cos(\frac{\varepsilon}{2}) + \sqrt{3}\sin(\frac{\varepsilon}{2})\right) & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}}\left(\cos(\frac{\varepsilon}{2}) - \frac{1}{\sqrt{3}}\sin(\frac{\varepsilon}{2})\right) \end{pmatrix} \tag{5.62}$$

Obviously a small but nonvanishing phase ε results in a deviation from tri-bimaximal mixing with nonzero $\sin^2\vartheta_{13}$ (note U_{e3}). However, trimaximal mixing is always ensured. In addition there is again no Dirac phase, this results from the group structure (special form of ω) of the model. Remarkably this diagonalization holds also if one waives the CP invariance, a recent publication [90] discusses this model.

Clearly, mixing angles are independent of the model parameters M_i $(i = 0, 1, 2, 4)$ and y_3. On the other hand the phase ε, solely affecting the mixing angles, does not alter the neutrino masses in (5.61). As a consequence the parameter space splits up into two separate parts affecting different observables.

5.2.3 Mixing angles and figure of merit function

Comparing (5.62) with the standard parametrization (2.94) allows to express the mixing angles in terms of the phase ε. From U_{e3} we deduce

$$\sin^2\vartheta_{13} = \frac{2}{3}\cdot\sin^2\frac{\varepsilon}{2}. \tag{5.63}$$

Next, we exploit trimaximal mixing using $\sin\vartheta_\odot \cos\vartheta_{13} = 1/3$ to find

$$\sin^2\vartheta_\odot = \frac{1}{3 - 2\sin^2\frac{\varepsilon}{2}} = \frac{1}{3}\cdot\frac{1}{1-\sin^2\vartheta_{13}}. \tag{5.64}$$

Finally from the matrix element $U_{\nu 3}$ we derive

$$\sin^2\vartheta_{\rm atm} = \frac{1}{2} + \frac{1}{2\sqrt{3}}\cdot\frac{\sin\varepsilon}{1-\sin^2\vartheta_{13}}. \tag{5.65}$$

Since all three mixing angles depend on only one model parameter, it is an easy task to find analytically the minimum of the χ^2_a (a for angles) function. The individual contributions corresponding to the mixing angles are shown in Figure 5.1, where

$$\chi^2_a = \chi^2_{13} + \chi^2_\odot + \chi^2_{\rm atm}. \tag{5.66}$$

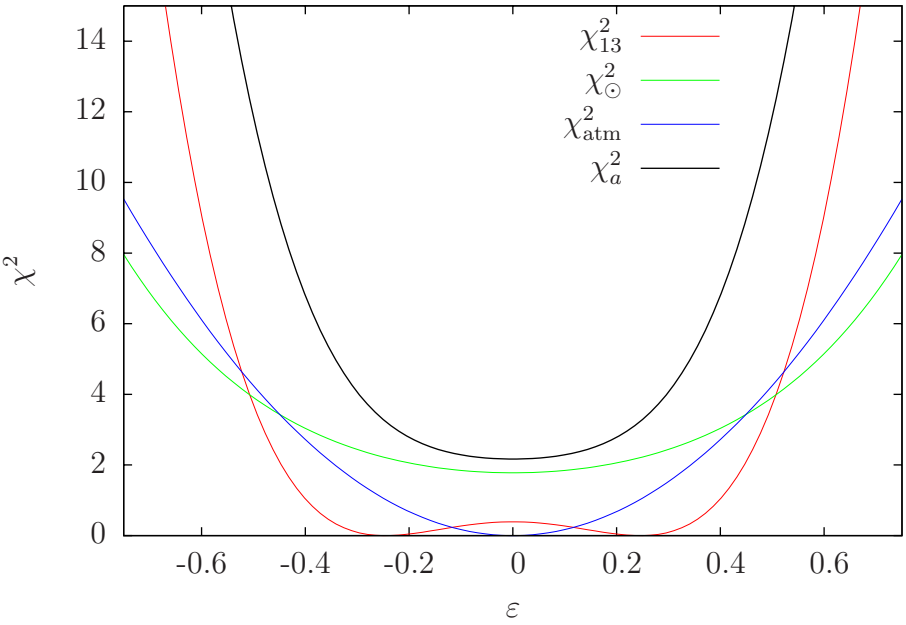


Figure 5.1: Different contributions to χ^2 as functions of ε.

While the contribution from χ^2_{13} would prefer $\varepsilon \approx \pm 0.25$, the total χ^a function shows only a minimum at $\varepsilon = 0$. Evidently, any deviation of ε from zero leads both solar and atmospheric mixing angle further away from the experimental mean values. The overall curve shows an approximate interval, $\varepsilon \in [-0.1, 0.1]$, where only minor additional stress is added between theory and experiment.

Chapter 6

Results

In the following chapter the results of the numerical analysis described in chapter 4 are presented. Two different models of chapter 5, each with normal and inverted mass spectrum, are investigated.

6.1 Model for tri-bimaximal mixing

Prior to the numerical analysis one has to decide which parameters in the model are independent and count as free variables in the fitting procedure. The model for tri-bimaximal mixing fixes the PMNS-matrix (except for the Majorana phases) and thus introduces a constant contribution to χ^2, denoted with χ^2_a.

$$\begin{aligned} \chi^2_a &= 2.1684 \\ \sin^2\vartheta_{13} &= 0 \\ \sin^2\vartheta_{\odot} &= 1/3 \\ \sin^2\vartheta_{\text{atm}} &= 1/2 \end{aligned} \tag{6.1}$$

The pull coming from the solar mixing angle is largest with $\text{pull}(\sin^2\vartheta_{\odot}) = 1.333$, while $\text{pull}(\sin^2\vartheta_{13}) = -0.625$ and $\text{pull}(\sin^2\vartheta_{\text{atm}}) = 0$. Since the angles are fixed the only remaining observables for a fit are the mass squared differences. These depend on eight complex model parameters

$$y_2, y_3, v_0, v_\chi, M_0, M_1, M_2, M_4, \tag{6.2}$$

see section 5.1.3. Some of them can be absorbed into each other or alternatively their values may be fixed. The following list shows the reduction of free parameters.

- y_2 and v_0 appear only in the product $a = y_2^* v_0 \Rightarrow$ One complex parameter removed.
- The model restricts v_χ to be real, $v_\chi = v_\chi^* \Rightarrow$ One phase removed.
- v_χ and y_3 appear only in the product $M_N' = y_3^* v_\chi \Rightarrow$ One real parameter removed.
- Consider the expressions of x, y, z and $\det M_R$ in (5.23). With

$$x, y, z \propto \frac{M_i^4}{\det M_R} \quad \text{and} \quad \det M_R \propto M_i^5, \quad i = 0, 1, 2, 4, N' \tag{6.3}$$

  we get

$$x, y, z \propto \frac{M_i^4}{M_i^5} = \frac{1}{M_i}. \tag{6.4}$$

  Thus, $a = y_2^* v_0$ can be absorbed into the $M_i \Rightarrow$ One complex parameter removed.

Altogether three of eight complex parameters can be removed, the parameter space of the Nelder-Mead procedure is reduced from 16 to 10 dimensions. For the three fixed parameters we choose v_0, v_χ and y_2 because their values are deductable from the seesaw mechanism. The VEV v_0 is identified with the electroweak scale, while v_χ is at the GUT scale.

$$\begin{aligned} v_0 &= 170 \text{ GeV} \\ v_\chi &= 2 \cdot 10^{16} \text{ GeV} \end{aligned} \tag{6.5}$$

The value of y_2 can be estimated because $a = y_2^* v_0$ should be comparable to the scale of $m_{\mu,\tau} \approx 10^2 - 10^3$ MeV. So y_2 lies in the region of $10^{-3} - 10^{-2}$, we choose

$$y_2 = 10^{-3}. \tag{6.6}$$

The remaining parameters

$$y_3, M_0, M_1, M_2, M_4 \in \mathbb{C} \tag{6.7}$$

span the 10-dimensional space wherein the Nelder-Mead simplex is placed. The 11 vertices that build up the initial simplex before the iteration procedure are set by a random number generator. In order to get reasonable neutrino masses while

employing the seesaw mechanism, the initial values are taken from the following intervals:

$$\begin{aligned} \mathrm{Re}\{M_0\}, \mathrm{Re}\{M_1\}, \mathrm{Re}\{M_2\}, \mathrm{Re}\{M_4\} &\in [-10^{10}, 10^{10}] \text{ GeV} \\ \mathrm{Im}\{M_0\}, \mathrm{Im}\{M_1\}, \mathrm{Im}\{M_2\}, \mathrm{Im}\{M_4\} &\in [-10^{10}, 10^{10}] \text{ GeV} \\ \mathrm{Re}\{y_3^*\} = \mathrm{Re}\{M_N'/v_\chi\} &\in [-10^{-6}, 10^{-6}] \\ \mathrm{Im}\{y_3^*\} = \mathrm{Im}\{M_N'/v_\chi\} &\in [-10^{-6}, 10^{-6}] \ . \end{aligned} \tag{6.8}$$

After preparing the first simplex the Nelder-Mead method for minimizing χ_m^2 (m for masses) is carried out until one of the stopping criteria (4.13) is fulfilled. The data sets with $\chi_m^2 < 1$ are saved for further processing. The whole procedure of setting a random simplex and searching for a (local) minimum is repeated 10^6 and $5 \cdot 10^6$ times for normal and inverted spectrum, respectively.

6.1.1 Normal spectrum

The minimization procedure in the case of the normal mass spectrum succeds all 10^6 times, with 48 data sets identified as duplicates and removed. The criterion $\chi_m^2 < 1$ is fulfilled 215462 times, the data is sorted according to the χ_m^2-value. The best fit for the input data is

$$\begin{aligned} \chi_m^2 &= 2.07022 \cdot 10^{-8} \\ m_1 &= 9.97243 \cdot 10^{-4} \text{ eV} \\ m_2 &= 8.80311 \cdot 10^{-3} \text{ eV} \\ m_3 &= 4.89998 \cdot 10^{-2} \text{ eV}, \end{aligned} \tag{6.9}$$

with the following model parameters:

$$\begin{aligned} M_0 &= (-3.79927 + 1.45186i) \cdot 10^9 \text{ GeV} \\ M_1 &= (2.25056 - 2.32944i) \cdot 10^9 \text{ GeV} \\ M_2 &= (3.51239 + 5.03910i) \cdot 10^9 \text{ GeV} \\ M_4 &= (4.00159 - 4.03739i) \cdot 10^9 \text{ GeV} \\ y_3 &= (-4.48194 - 2.23229i) \cdot 10^{-7}. \end{aligned} \tag{6.10}$$

To get an idea of the mass distribution of all solutions, the interval between the smallest m_1 and the largest m_3 found is cut into 10^3 bins. The number of solutions lying in each bin is counted, resulting in Figure 6.1. One can clearly see the cut at $m_2 = \sqrt{\Delta m_\odot^2} \approx 8.75 \cdot 10^{-3}$ eV, values below this bound can only be achieved with increasing χ_m^2.

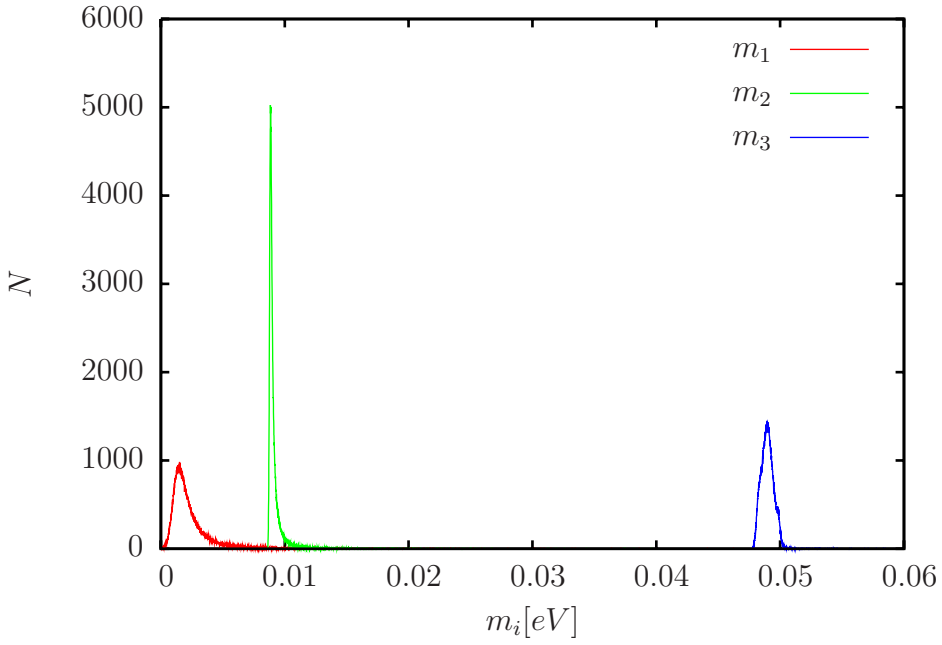


Figure 6.1: Mass distribution for normal mass ordering.

6.1.2 Inverted spectrum

The minimization procedure for the inverted mass spectrum is carried out $5 \cdot 10^6$ times, of which 4999973 stable minima are reached. 58898 data sets give $\chi^2_m < 1$, 60 duplicate solutions are removed. The best fit values are

$$\begin{aligned} \chi^2 &= 6.82855 \cdot 10^{-6} \\ m_1 &= 4.82388 \cdot 10^{-2} \text{ eV} \\ m_2 &= 4.90252 \cdot 10^{-2} \text{ eV} \\ m_3 &= 1.87701 \cdot 10^{-3} \text{ eV}, \end{aligned} \tag{6.11}$$

the corresponding model parameters are

$$\begin{aligned} M_0 &= (1.72909 - 3.25560i) \cdot 10^9 \text{ GeV} \\ M_1 &= (-1.15458 + 1.57532i) \cdot 10^9 \text{ GeV} \\ M_2 &= (5.63025 - 0.79025i) \cdot 10^9 \text{ GeV} \\ M_4 &= (3.21648 + 1.72047i) \cdot 10^9 \text{ GeV} \\ y_3 &= (3.82251 - 5.18326i) \cdot 10^{-7}. \end{aligned} \tag{6.12}$$

Figure 6.2 gives an overview of the mass distribution in the case of the inverted mass spectrum (Again 10^3 bins are used to produce this figure).

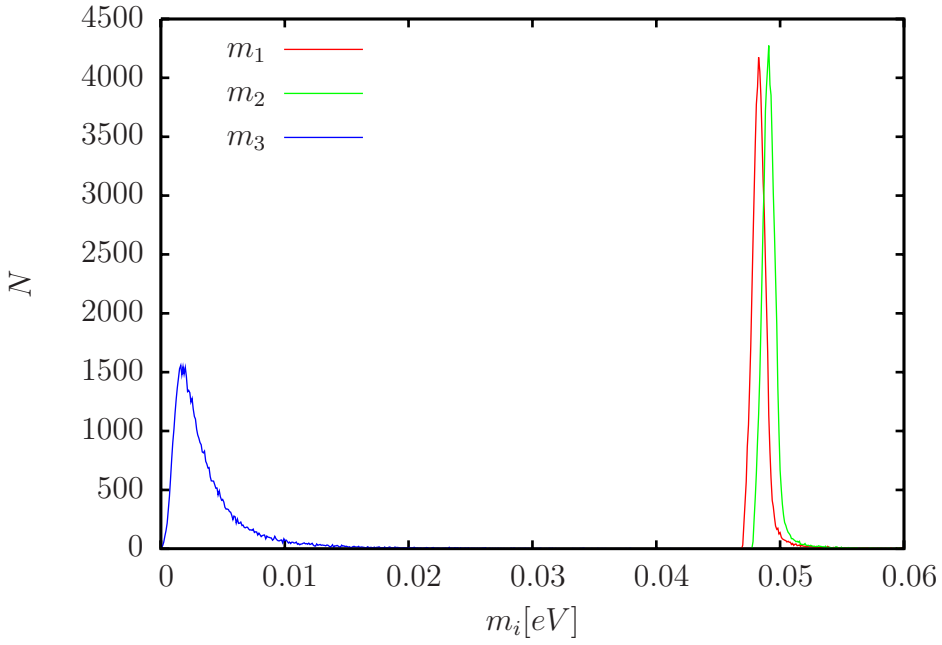


Figure 6.2: Mass distribution for inverted mass ordering.

6.2 Modified model

The modifications of the model for tri-bimaximal mixing described in section 5.2 are now employed. Starting from the list of model parameters (6.2) we can reduce the number of the degrees of freedom by taking into account the following properties.

- CP-invariance forces $y_2, y_3, M_0, M_1, M_2, M_4 \in \mathbb{R}$.
- y_2 and v_0 appear only in the product $a = y_2^* v_0 \Rightarrow$ One parameter removed (not the phase of v_0).
- v_χ and y_3 appear only in the product $M'_N = y_3^* v_\chi$ and $M_N = y_3^* v_\chi^* \Rightarrow$ One parameter removed (not the phase of v_χ)
- The absolute value of $a = y_2^* v_0$ can be absorbed into the $M_i \Rightarrow$ One parameter removed.
- The phase of $a = y_2^* v_0$ appears only inside a absolute value $\Rightarrow$ One phase removed.

Altogether there are now six free parameters including one phase $\varepsilon_\chi \in [-\pi/2, \pi/2]$, see (5.54). Since the Nelder-Mead method is based on a simplex in $\mathbb{R}^n$ we reparametrize the phase:

$$e^{i\varepsilon_\chi} = e^{i \arctan r_\chi} = \frac{1}{\sqrt{1+r_\chi^2}}(1 + i r_\chi). \tag{6.13}$$

We choose the values for the fixed values in the same manner as with the model for tri-bimaximal mixing:

$$\begin{aligned} v_0 &= 170 \text{ GeV} \\ \tilde{v}_\chi &= 2 \cdot 10^{16} \text{ GeV} \\ y_2 &= 10^{-3}. \end{aligned} \tag{6.14}$$

The six degrees of freedom for the Nelder-Mead simplex are:

$$r_\chi, y_3, M_0, M_1, M_2, M_4 \in \mathbb{R}. \tag{6.15}$$

The initial values are chosen randomly from the previously mentioned intervals.

$$\begin{aligned} M_0, M_1, M_2, M_4 &\in [-10^{10}, 10^{10}] \text{ GeV} = [-10^{19}, 10^{19}] \text{ eV} \\ y_3^* = M_N'/v_\chi &\in [-10^{-6}, 10^{-6}] \\ r_\chi &\in [-10^{-1}, 10^{-1}]. \end{aligned} \tag{6.16}$$

The interval limits of r_χ correspond to angles of $\pm 5.7°$. The fitting procedure is started 10^6 and $5 \cdot 10^6$ times for normal and inverted spectrum, respectively. All solutions fulfilling $\chi^2 < 10$ are accepted for further processing this time.

6.2.1 Normal spectrum

The normal spectrum run of the numerics gives 393132 data sets, 96 duplicates are removed. The best fit values for the masses are

$$\begin{aligned} \chi_m^2 &= 2.41483 \cdot 10^{-5} \\ m_1 &= 2.13116 \cdot 10^{-3} \text{ eV} \\ m_2 &= 9.00222 \cdot 10^{-3} \text{ eV} \\ m_3 &= 4.90421 \cdot 10^{-2} \text{ eV}, \end{aligned} \tag{6.17}$$

while the mixing angles give

$$\begin{aligned} \chi_a^2 &= 2.16841 \\ \sin^2 \vartheta_{13} &= 1.50553 \cdot 10^{-7} \\ \sin^2 \vartheta_\odot &= 3.33333 \cdot 10^{-1} \\ \sin^2 \vartheta_{\text{atm}} &= 5.00274 \cdot 10^{-1}. \end{aligned} \tag{6.18}$$

The corresponding model parameter values are:

$$\begin{aligned} r_\chi &= 9.50432 \cdot 10^{-4} \\ M_0 &= -1.10014 \cdot 10^{10}\ \text{GeV} \\ M_1 &= 7.10587 \cdot 10^{9}\ \text{GeV} \\ M_2 &= 3.74447 \cdot 10^{9}\ \text{GeV} \\ M_4 &= -5.75068 \cdot 10^{9}\ \text{GeV} \\ y_3 &= -1.75045 \cdot 10^{-7}. \end{aligned} \tag{6.19}$$

The fine-tuning properties (see section 4.3) of the best fit solution are shown in Figure 6.3. The fit is most sensitive to slight change of M_2 and M_4 where x varies from 0.995 to 1.005. Other interesting quantities calculated from these results are the effective Majorana mass for the neutrinoless double beta decay $|\langle m_{\beta\beta}\rangle|$, the mass ratio $R = m_1/\sqrt{\Delta m^2_\odot}$ and the angle ε_χ.

$$\begin{aligned} |\langle m_{\beta\beta}\rangle| &= 4.42152 \cdot 10^{-3}\ \text{eV} \\ R &= 2.43664 \cdot 10^{-1} \\ \varepsilon_\chi &= 9.50432 \cdot 10^{-4} = 0.054^\circ \end{aligned} \tag{6.20}$$

The distributions of the neutrino masses, the effective Majorana mass, the mass ratio, the phase ε_χ and the mixing angles are shown in Figure 6.4 to 6.8. The mass distributions look very similar to those of the model for tri-bimaximal mixing, except for the shape of the m_3-peak (Figure 6.4). The effective mass for neutrinoless double beta decay is calculated under the assumption of vanishing Majorana phases, i.e. $\beta_1 = \beta_2 = 0$ (Figure 6.5). The ratio R shows a maximum between 0.1 and 0.3 with a clear restriction to values < 1 (Figure 6.6). As expected from the analytical analysis in section 5.2.3 values of $\varepsilon_\chi \neq 0$ are disfavoured (Figure 6.7). The distribution of the mixing angles shows no noticeable movement away from $\sin^2\vartheta_\odot = 1/3$ (Figure 6.8). The variation of $\sin^2\vartheta_{\text{atm}}$ is quite broad with values from 0.47 to 0.53, $\sin^2\vartheta_{13}$ is hardly visible on the combined plot and thus plotted separately (Figure 6.9). In order to show the rise of χ^2_m for small values of the effective Majorana mass, the numerical method of a pinning term (section 4.3) was applied to generate Figure 6.10.

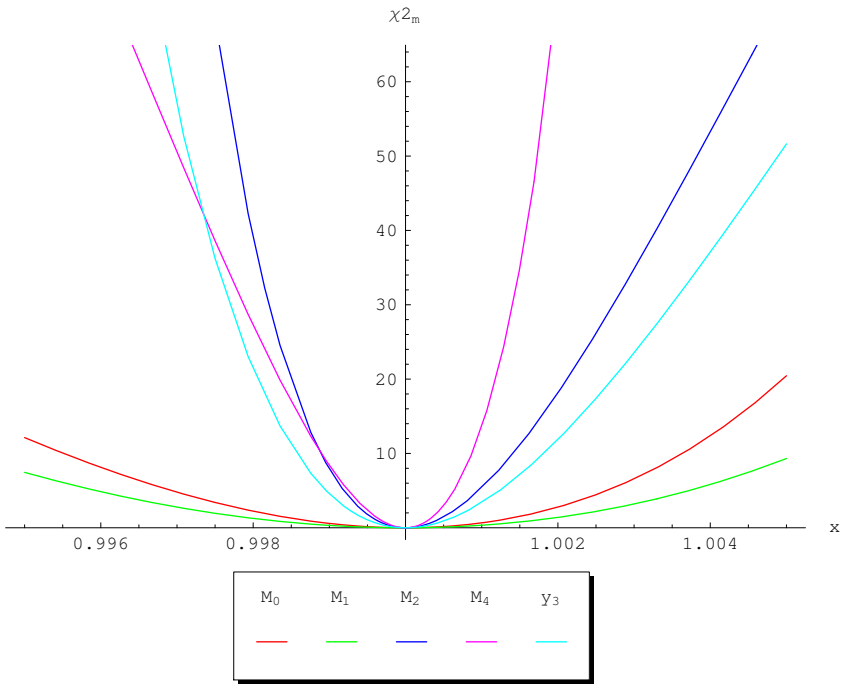


Figure 6.3: Fine-tuning properties of the best fit solution.

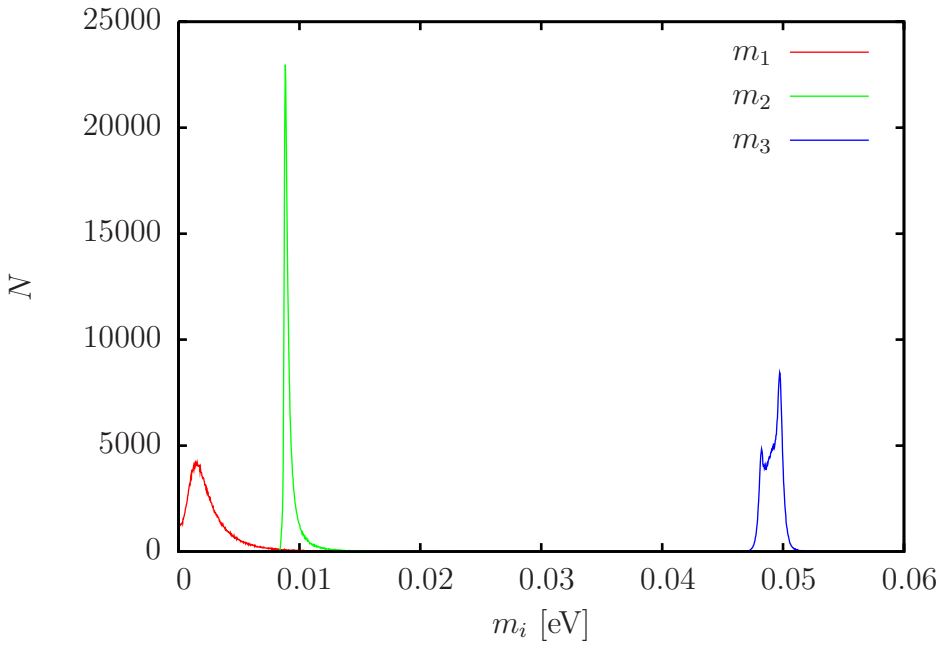


Figure 6.4: Mass distribution of the normal mass spectrum.

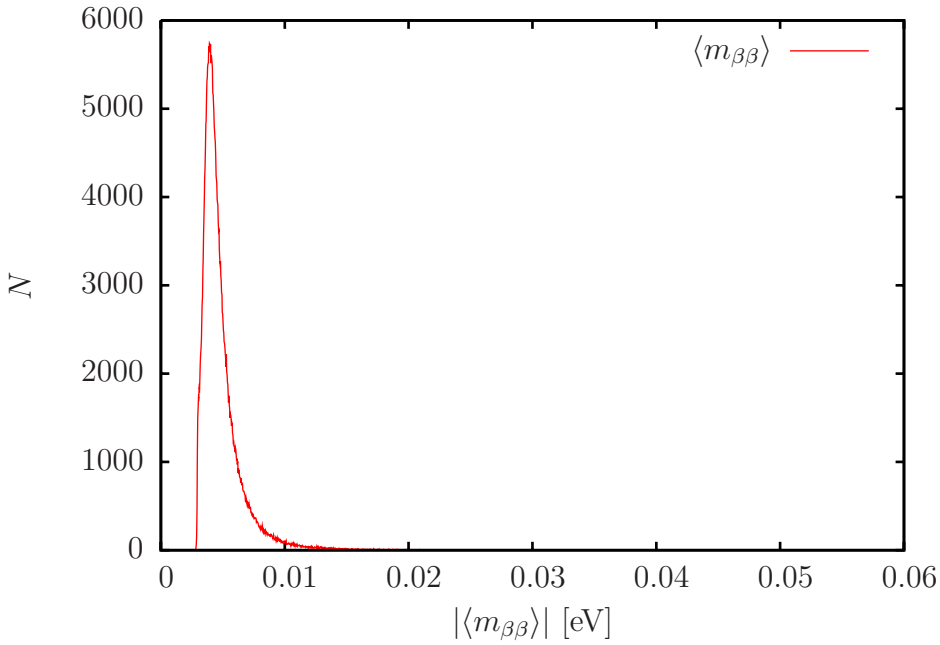


Figure 6.5: Distribution of the effective Majorana mass.

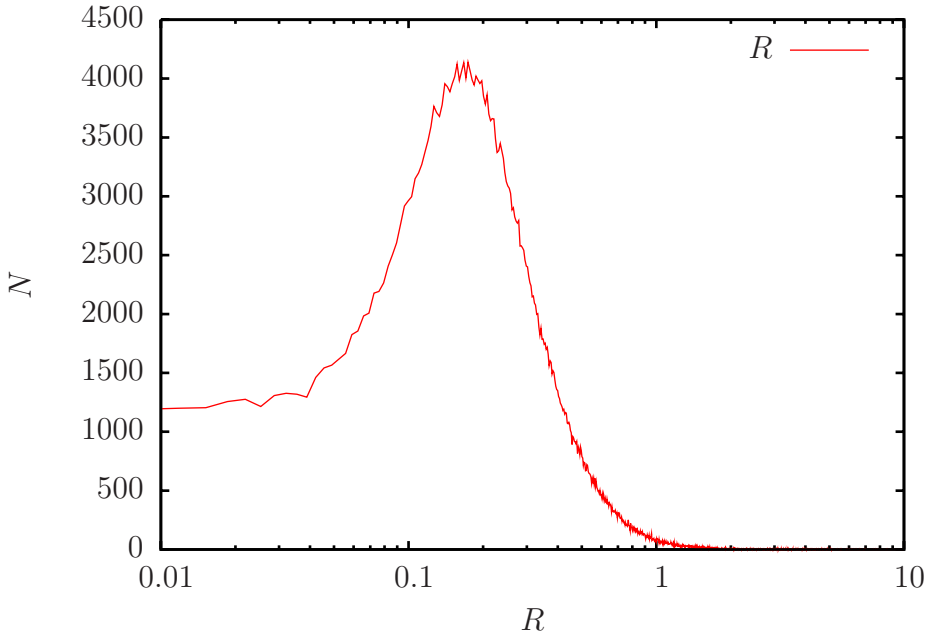


Figure 6.6: Distribution of the mass ratio R.

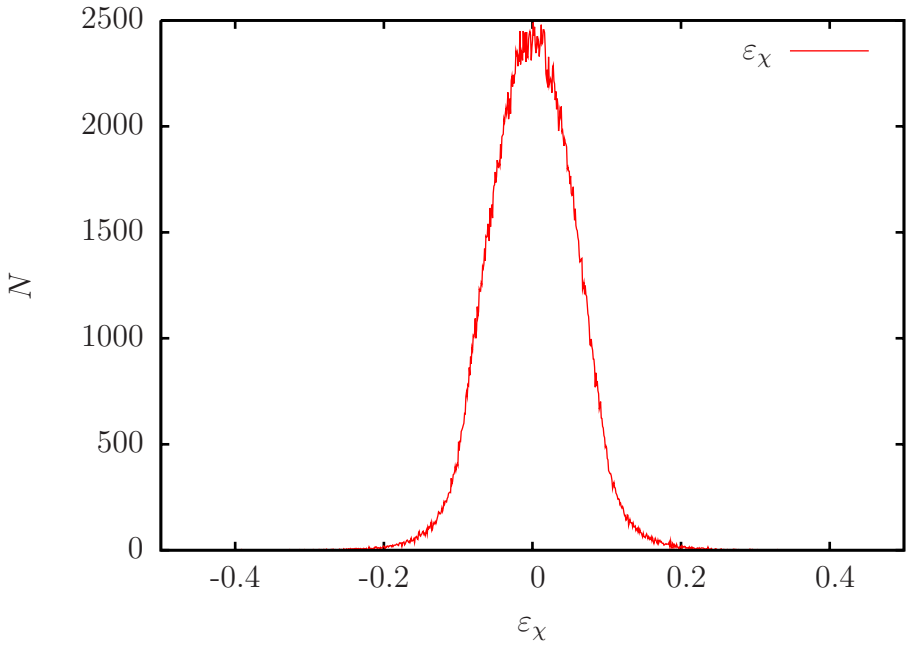


Figure 6.7: Distribution of the phase ε_χ.

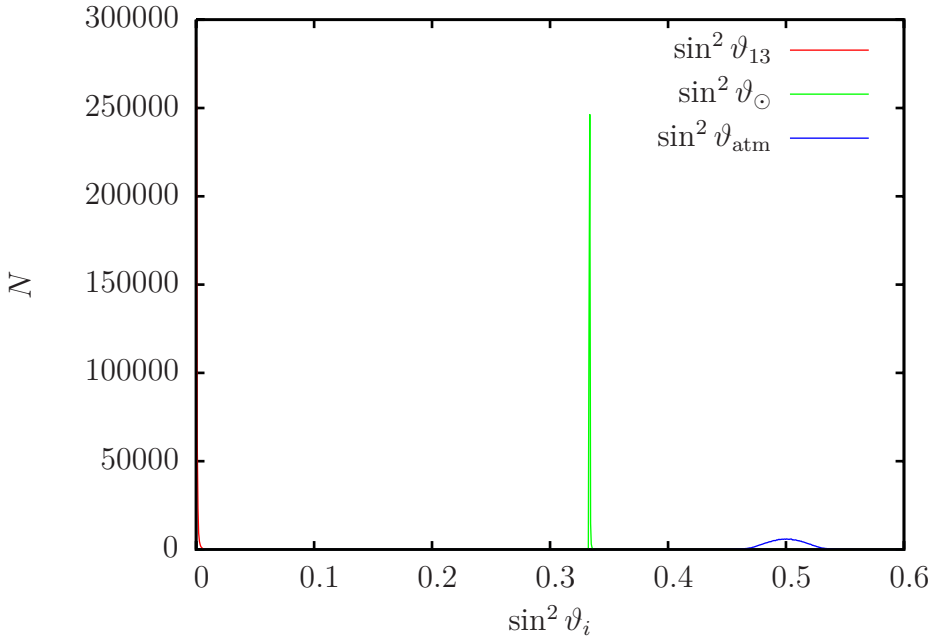


Figure 6.8: Distribution of the lepton mixing angles.

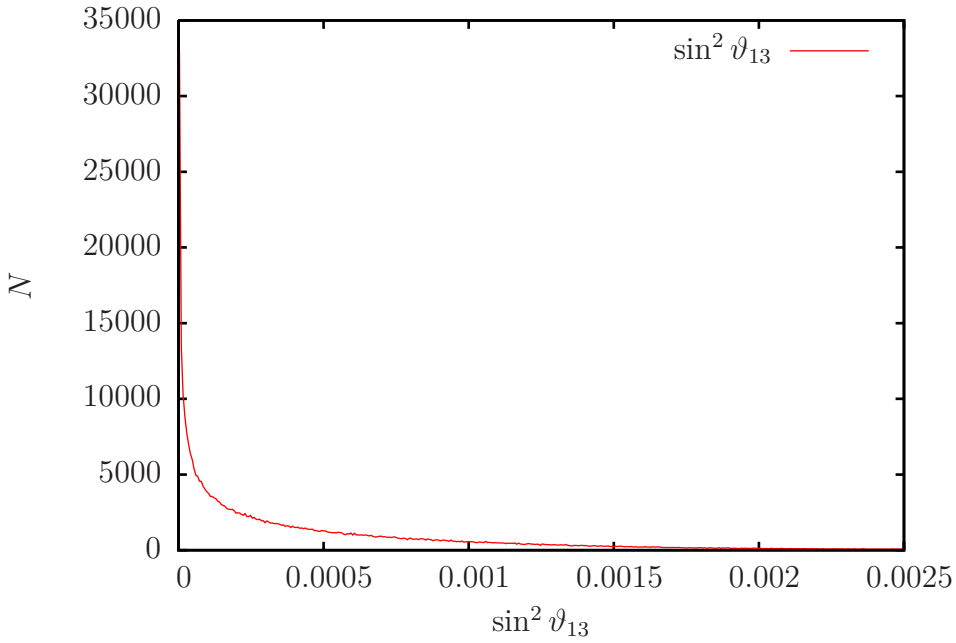


Figure 6.9: Distribution of $\sin^2 \vartheta_{13}$.

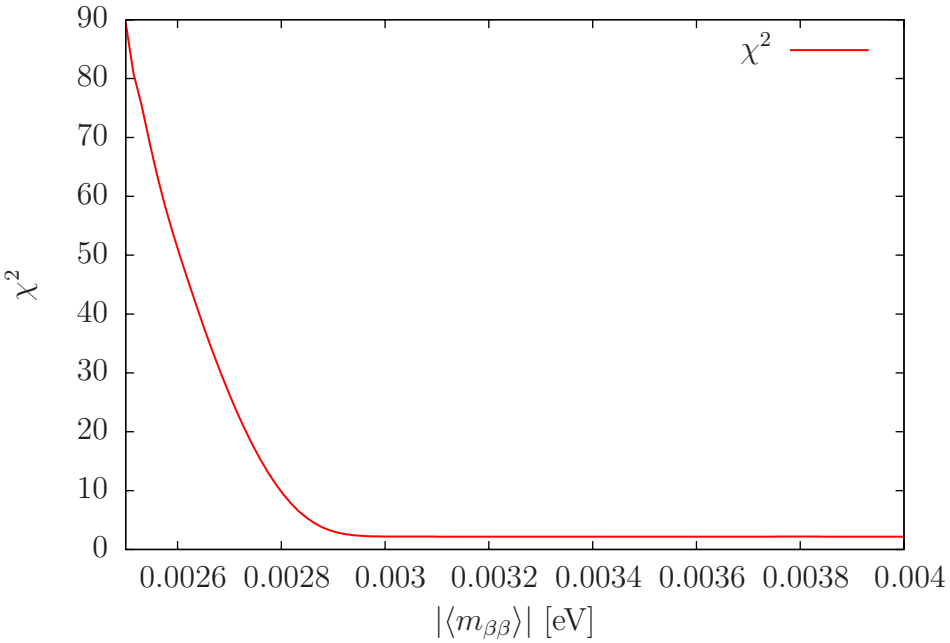


Figure 6.10: Lower bound of the effective Majorana mass.

6.2.2 Inverted spectrum

The Nelder-Mead method yields 282132 data sets fulfilling $\chi^2 < 10$, 383 entries are duplicates. The best fit values for the masses are:

$$\begin{aligned} \chi^2_m &= 3.17560 \cdot 10^{-5} \\ m_1 &= 4.83400 \cdot 10^{-2}\ \text{eV} \\ m_2 &= 4.91249 \cdot 10^{-2}\ \text{eV} \\ m_3 &= 3.72542 \cdot 10^{-3}\ \text{eV}. \end{aligned} \tag{6.21}$$

As expected the mixing angles do not veer away from tri-bimaximal mixing. The results are comparable to those of normal mass ordering because the parameter space for masses and angles are completely separated as shown in section 5.2.3.

$$\begin{aligned} \chi^2_a &= 2.16841 \\ \sin^2\vartheta_{13} &= 4.59687 \cdot 10^{-8} \\ \sin^2\vartheta_{\odot} &= 3.33333 \cdot 10^{-1} \\ \sin^2\vartheta_{\text{atm}} &= 4.99848 \cdot 10^{-1} \end{aligned} \tag{6.22}$$

The best fit solution presented corresponds to the following model parameters:

$$\begin{aligned} r_\chi &= -5.25178 \cdot 10^{-4} \\ M_0 &= 3.02744 \cdot 10^{9}\ \text{GeV} \\ M_1 &= -1.21957 \cdot 10^{9}\ \text{GeV} \\ M_2 &= 7.38539 \cdot 10^{9}\ \text{GeV} \\ M_4 &= -6.41894 \cdot 10^{9}\ \text{GeV} \\ y_3 &= 2.00966 \cdot 10^{-6}. \end{aligned} \tag{6.23}$$

In contrast to the normal mass spectrum the fine-tuning properties are slightly changed as shown in Figure 6.11. Instead of M_4 now M_1 seems to be rather fine-tuned. The effective Majorana mass $|\langle m_{\beta\beta}\rangle|$ is comparable to $m_{1,2}$, also note that the definition of $R = m_3/\sqrt{\Delta m^2_\odot}$ is different for the inverted mass spectrum.

$$\begin{aligned} |\langle m_{\beta\beta}\rangle| &= 4.86016 \cdot 10^{-2}\ \text{eV} \\ R &= 4.25950 \cdot 10^{-1} \\ \varepsilon &= -5.25178 \cdot 10^{-4} \end{aligned} \tag{6.24}$$

The distributions for the various quantities are again depicted in Figure 6.12 to 6.17. Besides the mass spectrum there are no changes compared to the normal mass ordering. Again the lower bound of $|\langle m_{\beta\beta}\rangle|$ is shown in Figure 6.18.

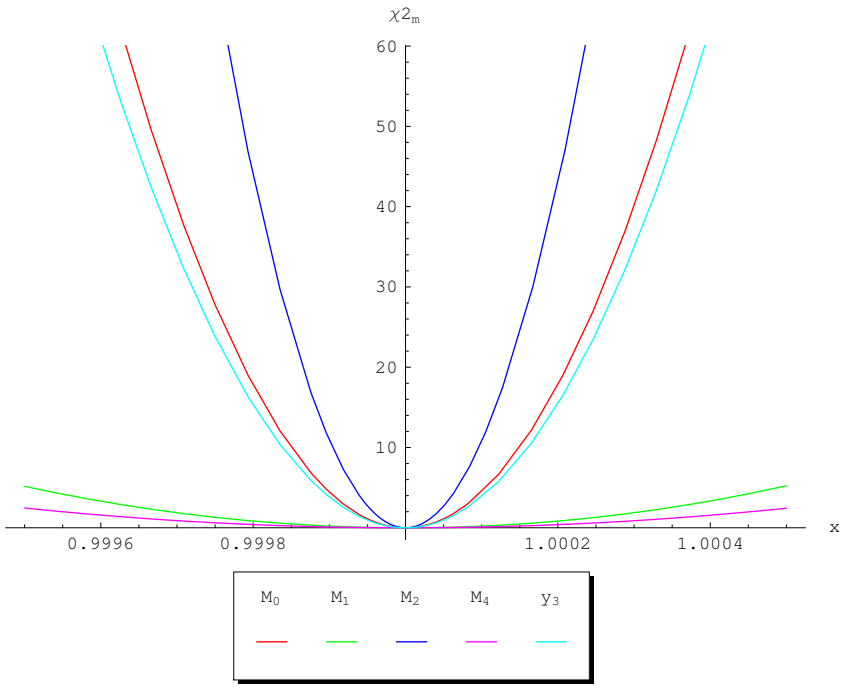


Figure 6.11: Fine-tuning properties of the best fit solution.

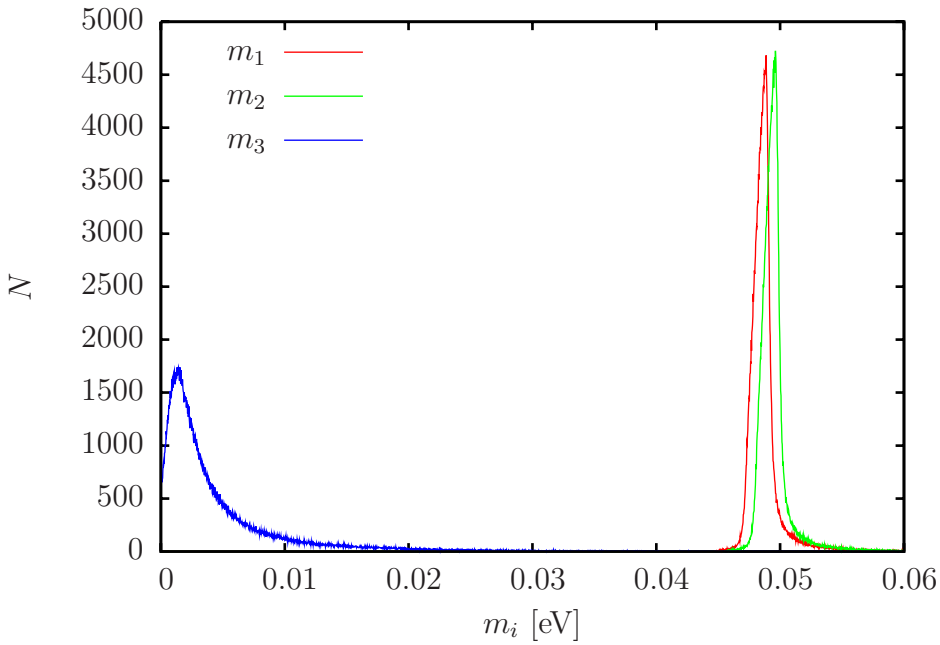


Figure 6.12: Mass distribution of the normal mass spectrum.

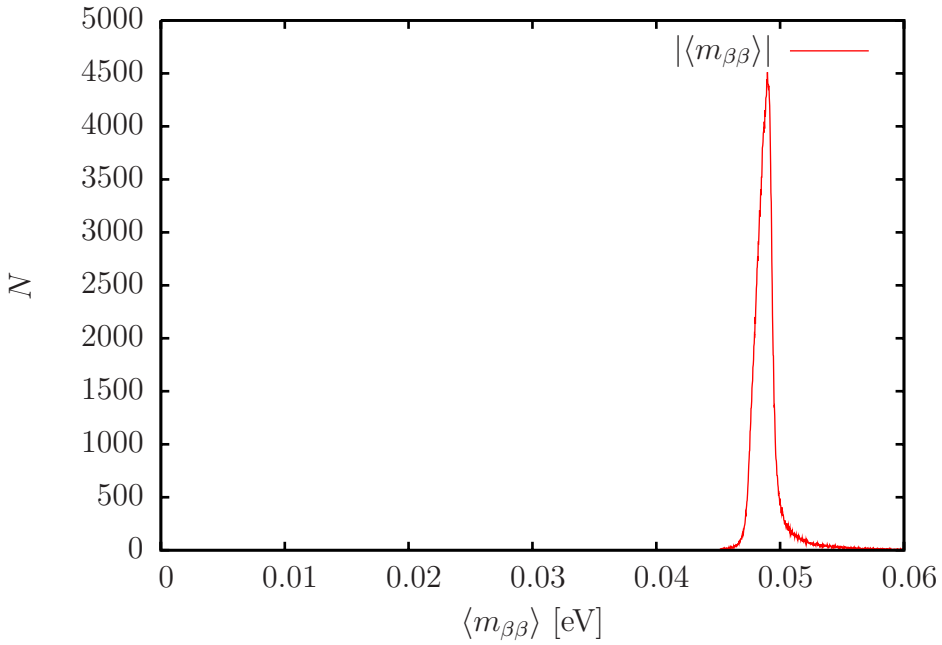


Figure 6.13: Distribution of the effective Majorana mass.

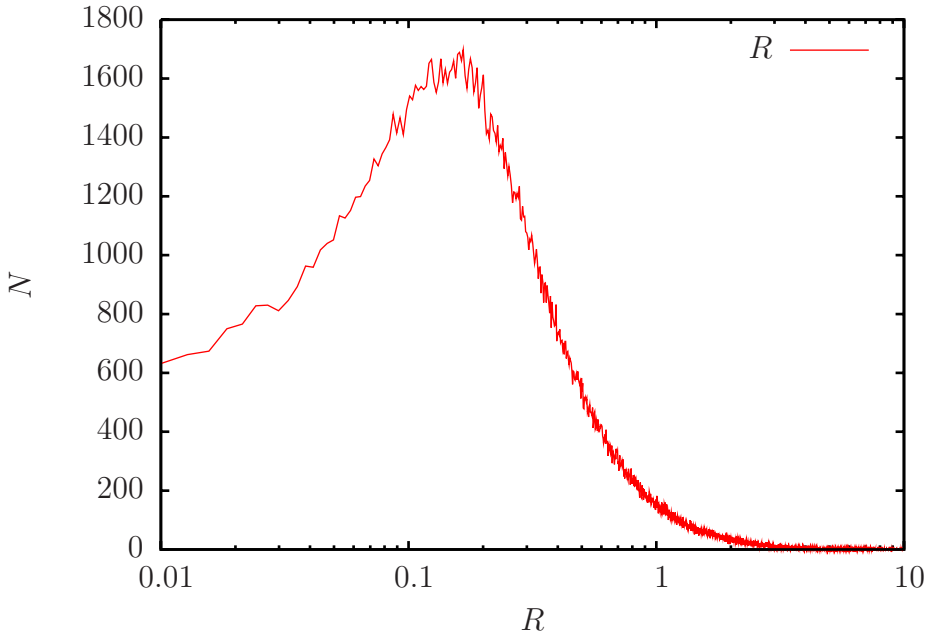


Figure 6.14: Distribution of the mass ratio R.

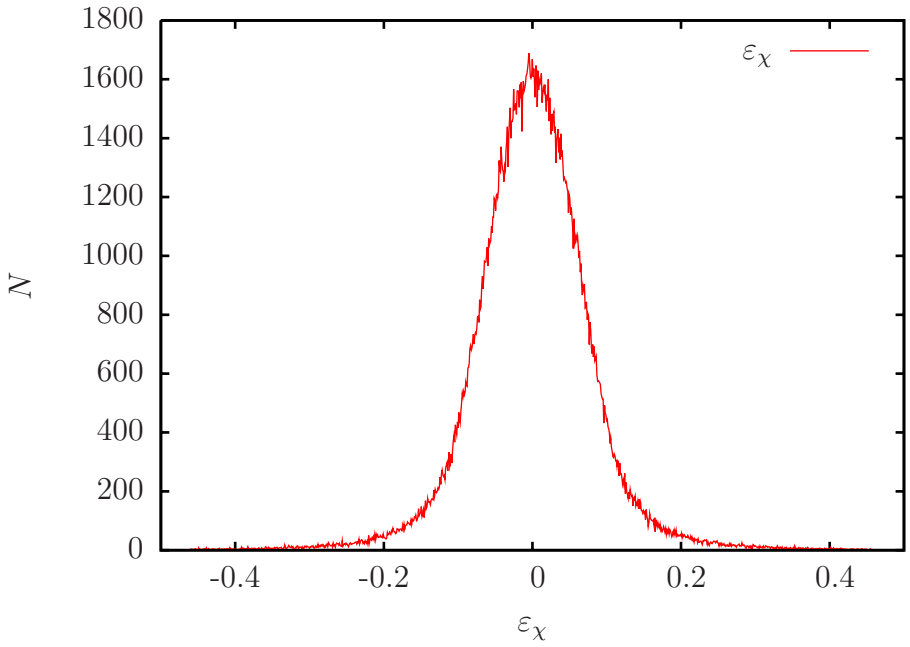


Figure 6.15: Distribution of the phase ε_χ.

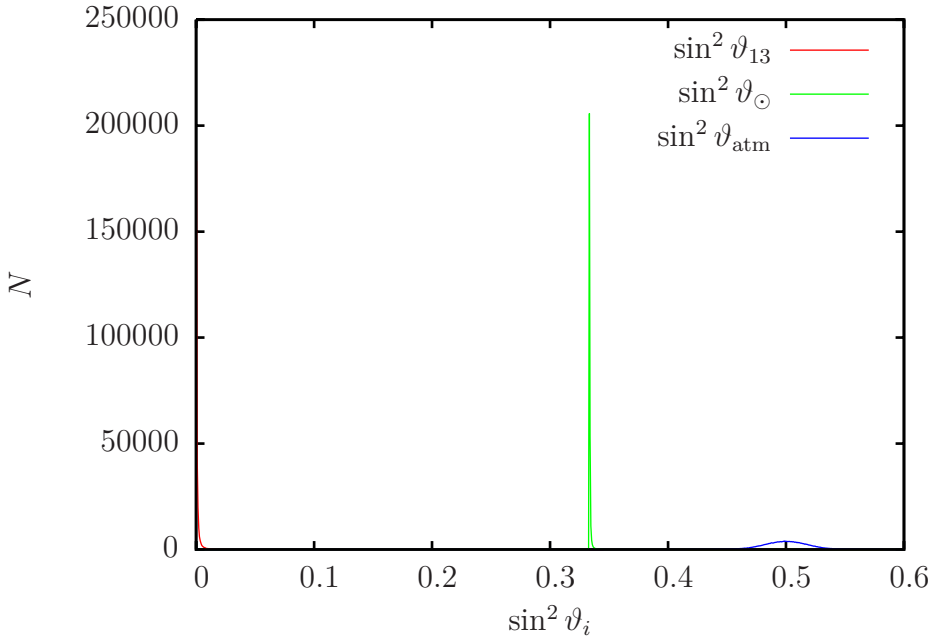


Figure 6.16: Distribution of the lepton mixing angles.

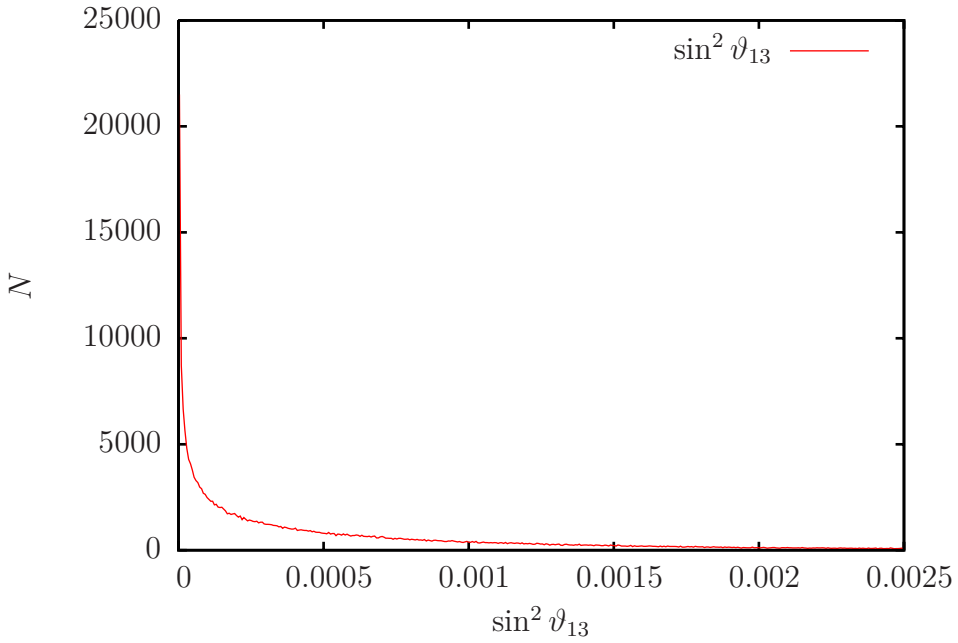


Figure 6.17: Distribution of $\sin^2 \vartheta_{13}$.

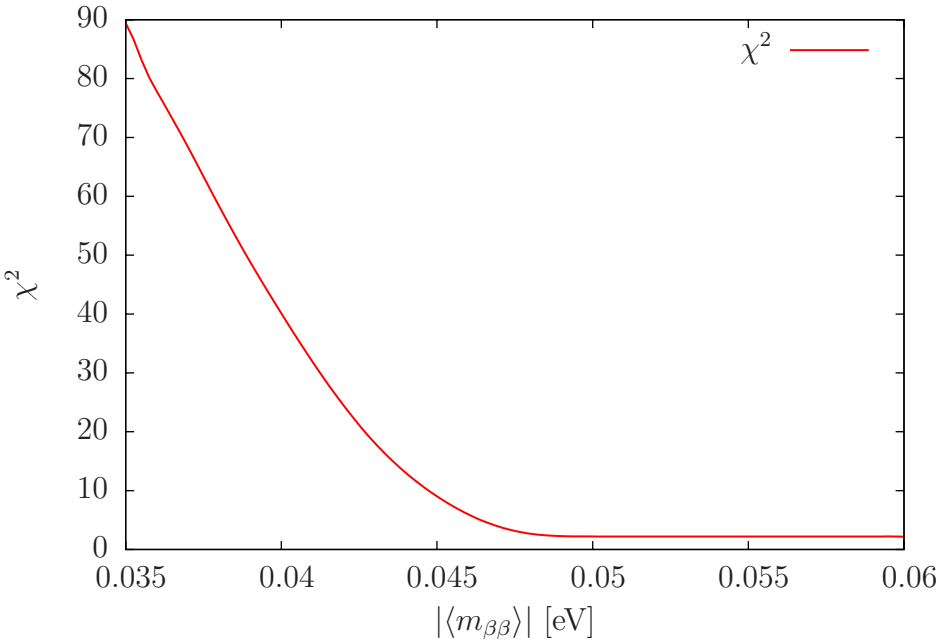


Figure 6.18: Lower bound of the effective Majorana mass.

6.2.3 Effective Majorana mass versus lightest mass

Reconsider equations (3.27) and (3.28) for the effective Majorana neutrino mass for normal and inverted spectrum, respectively. While the experimental data for mass squared differences and mixing angles is given in Table 3.1, the lightest mass m_0 ($:= m_1$ for normal, $:= m_3$ for inverted spectrum) and the phases β_1 and β_2 are not yet determined. This allows to plot those regions in the $(m_0, |\langle m_{\beta\beta}\rangle|)$-plane that are accessible via variation of both phases. Figure 6.19 shows $|\langle m_{\beta\beta}\rangle|$ versus m_0, the area between the red lines denotes possible values for the normal spectrum, the green lines the same for the inverted spectrum. Note that Figure

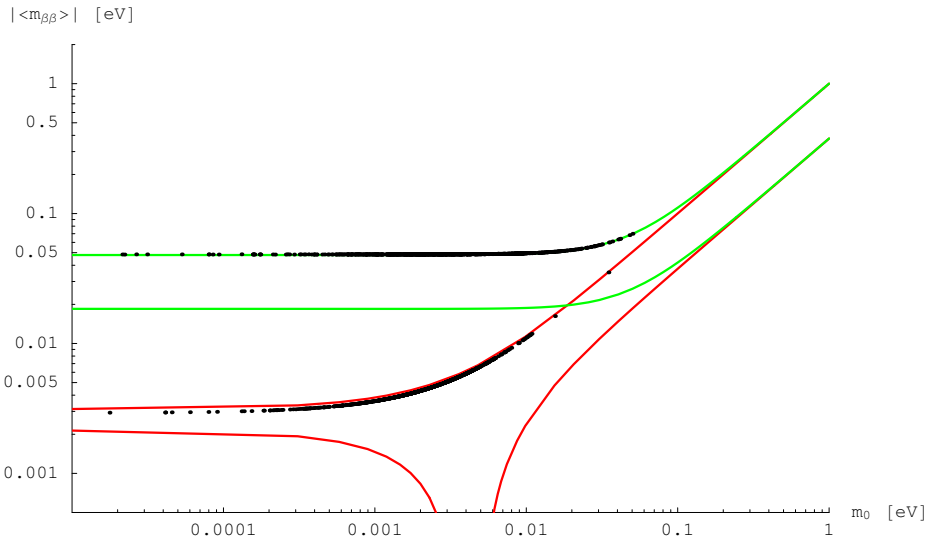


Figure 6.19: Effective Majorana mass $|\langle m_{\beta\beta}\rangle|$ versus the lightest neutrino mass m_0.

6.19 disregards the uncertainties of the experimental data, see [93] for a detailed analysis. The black dots show values taken from the best 1000 data sets (modified model) for both mass spectra, where $\beta_1 = \beta_2 = 0$. Obviously the combination of random set simplices and the Nelder-Mead minimization procedure leads to a spread in the smallest mass of several orders of magnitude.

Chapter 7

Conclusions

In this thesis we discussed the basic extensions in the lepton sector and presented two models which allow for tri-bimaximal and trimaximal lepton mixing. These models were tested with the help of numerical tools to compare predictions with experimental data.

The main conclusions concerning the investigated models are as follows. Both models allow a perfect fit to current mass squared differences for normal and inverted mass spectrum. The contributions to the total χ^2 are negligible, i.e. $10^{-8} - 10^{-5}$. Unlike the masses the mixing angles introduce some stress between theory and experiment, since $\chi_a^2 = 2.1684$ for tri-bimaximal mixing. Also with the described modifications which unfix the angles no further improvement is made, as expected from the analytical discussion in 5.2.3.

Regarding the numerics the Nelder-Mead method proved to be an effective tool for the minimization of the figure of merit function. With the tested models there were no problems finding excellent minima. Further application of the provided sample code should be considered with more elaborate models including also the quark sector. This would be a good chance to test the algorithm with more parameters and observables. Adaption of the source code would be an easy task since most of the model-related information is part of one subroutine.

Considering the great variety of models today there is still a lot of potential for the application of the Nelder-Mead method in combination with the figure of merit function to serve as a tool to provide information about consistency of theory and experiment.

Appendix A

Appendix

The Appendix contains sample program code as described in section 4.4. Both the normal and the pinning term versions are provided, each with `nmm.f90` and `dnmm.f90`. The Mathematica script to read data and analyse the fine-tuning properties is also included.

A.1 Main program

nmm.f90 1

```
!*****************************************************************************
!
! file: nmm.f90
! Nelder Mead Method
!
! Modified model, CP-invariance, nonzero theta13
!
! constants:  v0t,vct,r0,y2
! parameters: rc,M0,M1,M2,M4,y3
! res values:
chi2,chi2m,chi2a,mass1,mass2,mass3,dm2atm,dm2sol,mbb,R,epsilo,sin213,sin2sol,sin2atm
!
!*****************************************************************************

module nmm_var

        implicit none
        integer,parameter :: N=6,Nres=13,Nc=4,fout=11,p18=selected_real_kind(p=18)
        real(kind=p18),parameter ::
rho=1.0_p18,chi=2.0_p18,gam=0.5_p18,sig=0.5_p18,eps=1.0E-6_p18,crit=1.0E1_p18
        complex(kind=p18),parameter :: c1=(1.0_p18,0.0_p18), ci=(0.0_p18,1.0_p18)
        real(kind=p18) :: c_v0t,c_vct,c_r0,c_y2
        complex(kind=p18) :: c_a
        integer :: i,j,k,loop,nrand,nloop=10000
        integer,dimension(6) :: stats
        integer,dimension(:),allocatable :: seed
        real :: tstart,tstop
        real(kind=p18) :: fr,fn,d,tmp
        real(kind=p18),dimension(Nres) :: res
        real(kind=p18),dimension(N) :: xc,xr,xn
        real(kind=p18),dimension(N+1) :: f
        real(kind=p18),dimension(N+1,N) :: x
        real(kind=p18),dimension(:,:),allocatable :: sav

end module nmm_var

!*****************************************************************************
!*****************************************************************************

module nmm_sub

        use nmm_var

        implicit none

        contains

!*****************************************************************************
        subroutine init()

                write(unit=*,fmt='("nloop = ")',advance='no')
                read(*,*) nloop
                call random_seed(size=nrand)
                allocate(seed(nrand),sav(nloop,N+Nres+1))
                seed=0
                xc=0
                x=0
                f=0
                sav=0
                res=0
                stats=0

        end subroutine init
!*****************************************************************************
        subroutine initf()

                c_v0t=1.7E11_p18
                c_vct=2.0E25_p18
```

nmm.f90 2

```
67                 c_r0=0.0E0_p18
68                 c_y2=1.0E-3_p18
69
70                 c_a= -1.0_p18*(c_y2*c_v0t*(c1+c_r0*ci)/(1.0_p18+c_r0**2.0_p18))**2.0_p18
71
72         end subroutine initf
73 !*****************************************************************************
74         subroutine prepf()
75
76                 call random_seed
77                 call random_seed(get=seed)
78                 !write(unit=*,fmt='(4(Z8.8,TR1))') seed
79                 do i=1,N+1
80                         do j=1,1
81                                 call random_number(x(i,j))
82                                 x(i,j)=1.0E-1_p18  *  2.0_p18*(x(i,j)-0.5_p18)
83                         end do
84                         do j=2,N-1
85                                 call random_number(x(i,j))
86                                 x(i,j)=1.0E19_p18  *  2.0_p18*(x(i,j)-0.5_p18)
87                         end do
88                         do j=N,N
89                                 call random_number(x(i,j))
90                                 x(i,j)=2.0_p18  *   1.0E-06_p18  *  2.0_p18*(x(i,j)-0.5_p18)
91                         end do
92                         call calcf(f(i),x(i,:),.FALSE.)
93                 end do
94
95         end subroutine prepf
96 !*****************************************************************************
97
98         subroutine calcf(a,b,sres)
99
100                implicit none
101                logical,intent(in) :: sres
102                real(kind=p18),intent(inout) :: a
103                real(kind=p18),dimension(N),intent(in) :: b
104                real(kind=p18) :: rc,M0,M1,M2,M4,y3
105                real(kind=p18) :: MN2,x,yt,z,detM
106                real(kind=p18) :: mass1,mass2,mass3,dm2atm,dm2sol,chi2m,mbb,R
107                real(kind=p18),parameter ::
    MV_dm2atm=2.4E-3_p18,SD_dm2atm=0.12E-3_p18,MV_dm2sol=7.65E-5_p18,SD_dm2sol=0.23E-5_p18
108                real(kind=p18) :: epsilo,sin213,sin2sol,sin2atm,chi2a
109                real(kind=p18),parameter ::
    MV_sin213=1.0E-2_p18,SD_sin213=1.6E-2_p18,MV_sin2sol=3.04E-1_p18,SD_sin2sol=0.22E-1_p18, &
110 MV_sin2atm=5.0E-1_p18,SD_sin2atm=0.7E-1_p18
111
112                rc=b(1)
113                M0=b(2)
114                M1=b(3)
115                M2=b(4)
116                M4=b(5)
117                y3=b(6)
118
119                !MASSES
120                MN2=(y3*c_vct)**2.0_p18
121
122                detM = (M0+2.0_p18*M1)*( (M0-M1)**2.0_p18*MN2 - ( (M0-M1)*M4 -
    3.0_p18*M2**2.0_p18)**2.0_p18 )
123                x = 1.0_p18/detM * ( (M0**2.0_p18-M1**2.0_p18)*(MN2-M4**2.0_p18) + &
124 (4.0_p18*M0+2.0_p18*M1)*M2**2.0_p18*M4 - 3.0_p18*M2**4.0_p18 )
125                z = 1.0_p18/detM * ( (M1**2.0_p18-M0*M1)*(MN2-M4**2.0_p18) +
    (M0-4.0_p18*M1)*M2**2.0_p18*M4 - 3.0_p18*M2**4.0_p18 )
126                yt = 1.0_p18/detM * ( (M0+2.0_p18*M1)*M2**2.0_p18*y3*c_vct )
127
128                mass1 = abs( c_a * (x+3.0_p18*yt-z) )
129                mass2 = abs( c_a * (x+2.0_p18*z) )
130                mass3 = abs( c_a * (x-3.0_p18*yt-z) )
```

nmm.f90 3

```
131
132                 dm2atm = mass3**2.0_p18 - mass1**2.0_p18
133                 dm2sol = mass2**2.0_p18 - mass1**2.0_p18
134
135                 chi2m = ((dm2atm-MV_dm2atm)/SD_dm2atm)**2.0_p18 +
    ((dm2sol-MV_dm2sol)/SD_dm2sol)**2.0_p18
136
137                 !ANGLES
138                 epsilo = atan(rc)
139
140                 sin213 = 2.0_p18/3.0_p18 * sin(epsilo/2.0_p18)**2.0_p18
141                 sin2sol = 1.0_p18 / (3.0_p18 * (1.0_p18 - sin213))
142                 sin2atm = 0.5_p18 + sin(epsilo) / (sqrt(12.0_p18)*(1-sin213))
143
144                 chi2a = ((sin213-MV_sin213)/SD_sin213)**2.0_p18 +
    ((sin2sol-MV_sin2sol)/SD_sin2sol)**2.0_p18 + &
145 ((sin2atm-MV_sin2atm)/SD_sin2atm)**2.0_p18
146
147                 a = chi2m + chi2a
148
149                 if(sres) then
150
151                         mbb = abs( ( mass1*(1.0_p18-sin2sol) +
    sqrt(mass1**2.0_p18+dm2sol)*sin2sol*exp(ci*0.0_p18) )*(1.0_p18-sin213) + &
152 sqrt(mass1**2.0_p18 + dm2atm)*sin213*exp(ci*0.0_p18) )
153
154                         R = mass1/sqrt(dm2sol)
155
156                         res(1)=chi2m
157                         res(2)=chi2a
158                         res(3)=mass1
159                         res(4)=mass2
160                         res(5)=mass3
161                         res(6)=dm2atm
162                         res(7)=dm2sol
163                         res(8)=mbb
164                         res(9)=R
165                         res(10)=epsilo
166                         res(11)=sin213
167                         res(12)=sin2sol
168                         res(13)=sin2atm
169                 end if
170
171         end subroutine calcf
172 !*****************************************************************************
173         subroutine sort()
174
175                 integer,dimension(1) :: imin
176                 real(kind=p18),dimension(N) :: vtemp
177                 real(kind=p18) :: stemp
178
179                 do i=1,N
180                         imin=i-1+minloc(f(i:N+1))
181                         vtemp=x(i,:)
182                         stemp=f(i)
183                         x(i,:)=x(imin(1),:)
184                         f(i)=f(imin(1))
185                         x(imin(1),:)=vtemp
186                         f(imin(1))=stemp
187                 end do
188
189         end subroutine sort
190 !*****************************************************************************
191         subroutine centroid()
192
193                 xc=0
194                 do j=1,N
195                         do i=1,N
```

nmm.f90 4

```
196                                 xc(j) = xc(j) + x(i,j)
197                         end do
198                         xc(j)=xc(j)/N
199                 end do
200                 !write(*,*) "centroid =", xc
201
202         end subroutine centroid
203 !*****************************************************************************
204         subroutine shrinkage()
205
206                 do j=2,N+1
207                         x(j,:) = x(1,:) + sig*(x(j,:)-x(1,:))
208                         call calcf(f(j),x(j,:),.FALSE.)
209                 end do
210
211         end subroutine shrinkage
212 !*****************************************************************************
213         subroutine show()
214
215                 do i=1,N+1
216                         write(unit=*,fmt='("x("I2.2") =")',advance='no') i
217                         do j=1,N
218                                 write(unit=*,fmt='(F13.8)',advance='no') x(i,j)
219                         end do
220                         write(*,*)
221                 end do
222                 write(*,*) "f =",f
223
224         end subroutine show
225 !*****************************************************************************
226         subroutine calcd()
227
228                 real(kind=p18) :: fm
229
230                 fm=0.0_p18
231                 d=0.0_p18
232                 do j=1,N+1
233                         fm = fm + f(j)
234                 end do
235                 fm=fm/(N+1)
236                 do j=1,N+1
237                         d = d + (f(j)-fm)**2.0**p18
238                 end do
239                 d=d/(N+1)
240
241         end subroutine calcd
242 !*****************************************************************************
243         subroutine wfile()
244
245                 open(unit=fout,file="nmm.dat",status="replace",action="write",form="unformatte
    d",position="rewind")
246                 write(unit=fout) N,Nres,nloop,Nc
247                 write(unit=fout) stats
248                 do loop=1,nloop
249                         if(sav(loop,N+Nres+1)/=0) write(unit=fout) sav(loop,:)
250                 end do
251                 write(unit=fout) c_v0t,c_vct,c_r0,c_y2
252                 close(fout)
253
254         end subroutine wfile
255 !*****************************************************************************
256         subroutine finish()
257
258                 !write(*,*) "Steps: ",k
259                 deallocate(seed,sav)
260
261         end subroutine finish
262 !*****************************************************************************
```

nmm.f90 5

```
263 end module nmm_sub
264
265 !*****************************************************************************
266 !*****************************************************************************
267
268 program nmm
269
270         use nmm_var
271         use nmm_sub
272
273         implicit none
274
275         write(*,*) huge(fn)
276         write(*,*) tiny(fn)
277
278         call cpu_time(tstart)
279
280         call init()
281         call initf()
282         do loop=1,nloop
283                 call prepf()
284                 k=1
285                 do
286                         k=k+1
287                         call sort()
288                         call centroid()
289                         xr = xc + rho*(xc-x(N+1,:))
290                         call calcf(fr,xr,.FALSE.)
291                         if(f(1)<=fr .AND. fr<f(N)) then
292                                 x(N+1,:)=xr
293                                 f(N+1)=fr
294                                 !write(*,*) "Reflection accepted"
295                                 go to 10
296                         else if(fr<f(1)) then
297                                 xn = xc + chi*(xr-xc)
298                                 call calcf(fn,xn,.FALSE.)
299                                 if(fn<fr) then
300                                         x(N+1,:)=xn
301                                         f(N+1)=fn
302                                         !write(*,*) "Expansion accepted"
303                                         go to 10
304                                 else
305                                         x(N+1,:)=xr
306                                         f(N+1)=fr
307                                         !write(*,*) "Expansion rejected -> Reflection"
308                                         go to 10
309                                 end if
310                         else if(f(N)<=fr .AND. fr<f(N+1)) then
311                                 xn = xc + gam*(xr-xc)
312                                 call calcf(fn,xn,.FALSE.)
313                                 if(fn<=fr) then
314                                         x(N+1,:)=xn
315                                         f(N+1)=fn
316                                         !write(*,*) "Outside contraction accepted"
317                                         go to 10
318                                 else
319                                         !write(*,*) "Outside contraction rejected ->
    Shrinkage"
320                                         call shrinkage()
321                                         go to 10
322                                 end if
323                         else if(fr>=f(N+1)) then
324                                 xn = xc - gam*(xc-x(N+1,:))
325                                 call calcf(fn,xn,.FALSE.)
326                                 if(fn<f(N+1)) then
327                                         x(N+1,:)=xn
328                                         f(N+1)=fn
329                                         !write(*,*) "Inside contraction accepted"
```

nmm.f90 6

```
330                                     go to 10
331                             else
332                                     !write(*,*) "Inside contraction rejected -> Shrinkage"
333                                     call shrinkage()
334                                     go to 10
335                             end if
336                     end if
337 10                  call calcd()
338                     if(d<=eps) then
339                             !write(*,*) "Minimum reached: stats(1)",x(1,1),f(1)
340                             stats(1)=stats(1)+1
341                             exit
342                     end if
343                     if(mod(k,100)==0) then
344                             tmp=0.0_p18
345                             do i=1,N+1
346                                     do j=1,N
347                                             tmp=tmp+x(i,j)
348                                     end do
349                                     tmp=tmp+f(i)
350                             end do
351                             if(tmp*0.0_p18 /= 0.0_p18 .OR. tmp >= 1E99_p18) then
352                                     !write(*,*) "Detect infinity: stats(4)"
353                                     stats(4)=stats(4)+1
354                                     go to 20
355                             end if
356                     end if
357                     if(k>=1000) then
358                             !write(*,*) "No minimum reached: stats(3)"
359                             stats(3)=stats(3)+1
360                             go to 20
361                     end if
362             end do
363             call sort()
364             if(f(1)<=crit) then
365                     call calcf(f(1),x(1,:),.TRUE.)
366                     if(res(3)<=res(4) .AND. res(4)<=res(5)) then
367                             !write(*,*) "Criterion fulfilled: stats(2)"
368                             sav(loop,1:N)=x(1,:)
369                             sav(loop,N+1)=f(1)
370                             sav(loop,N+2:N+Nres+1) = res
371                             stats(2)=stats(2)+1
372                     else
373                             !write(*,*) "Wrong mass order: stats(5)"
374                             stats(5)=stats(5)+1
375                     end if
376             end if
377 20          if(mod(loop,1000)==0) write(unit=*,fmt='(7(I10,TR1))') loop,stats
378         end do
379         call wfile()
380         call finish()
381
382         call cpu_time(tstop)
383
384         write(*,*) "Time: ",tstop-tstart,"sec"
385
386 end program nmm
387
388
```

dnmm.f90 1

```
!*****************************************************************************
!
! file: dnmm.f90
! Data Processing for Nelder Mead Method
!
! Modified model, CP-invariance, nonzero theta13
!
! constants:  v0t,vct,r0,y2
! parameters: rc,M0,M1,M2,M4,y3
! res values:
chi2,chi2m,chi2a,mass1,mass2,mass3,dm2atm,dm2sol,mbb,R,epsilo,sin213,sin2sol,sin2atm

!*****************************************************************************

module dnmm_var

        implicit none
        integer,parameter :: fin=11,fout=12,p18=selected_real_kind(p=18)
        integer :: i,j,k,N,Nres,nloop,Nc,nch=100
        integer,dimension(6) :: stats
        real(kind=p18),dimension(:,:),allocatable :: sav,mvsd
        real(kind=p18),dimension(:),allocatable :: const
        type list1
                sequence
                real(kind=p18) :: c
                integer :: n
        end type list1
        type(list1),dimension(:),allocatable ::
lM1,lM2,lM3,lmbb,lR,lepsilo,lsin213,lsin2sol,lsin2atm
        real(kind=p18) :: lstarttemp,ltemp

end module dnmm_var

!*****************************************************************************
!*****************************************************************************

module dnmm_sub

        use dnmm_var

        contains

!*****************************************************************************
        subroutine readin()

                write(unit=*,fmt='("nch = ")',advance='no')
                read(*,*) nch
                allocate(lM1(nch),lM2(nch),lM3(nch),lmbb(nch),lR(nch),lepsilo(nch),lsin213(nch
),lsin2sol(nch),lsin2atm(nch))

                open(unit=fin,file="nmm.dat",status="old",action="read",form="unformatted",pos
ition="rewind")
                read(unit=fin) N,Nres,nloop,Nc
                read(unit=fin) stats
                allocate(sav(1:stats(2),1:N+Nres+1),mvsd(1:2,1:N+Nres+1),const(1:Nc))
                mvsd=0
                do i=1,stats(2)
                        read(unit=fin) sav(i,:)
                end do
                read(unit=fin) const(1:Nc)
                close(fin)

        end subroutine readin
!*****************************************************************************
        subroutine remdup()

                write(unit=*,fmt='("Removing duplicates: ")',advance='no')
                do i=1,stats(2)-stats(6)-1
```

dnmm.f90 2

```
65                          if(mod(i,floor(stats(2)/10.0))==0)
   write(unit=*,fmt='("*")',advance='no')
66                          j=i+1
67                          do while(j<=stats(2)-stats(6))
68                                  do k=1,N
69                                          !write(*,*)
   i,j,k,sav(i,k),sav(j,k),abs(sav(i,k)-sav(j,k))
70 !                                        if(abs((sav(i,k)-sav(j,k))/sav(i,k)) > 1E-5_p18) exit
71                                          if(sav(i,k) .NE. sav(j,k)) exit

72                                  end do
73                                  if(k==N+1) then
74                                          !write(*,*) i,sav(i,1)
75                                          !write(*,*) j,sav(j,1)
76                                          sav(j,:)=sav(stats(2)-stats(6),:)
77                                          sav(stats(2)-stats(6),:)=0
78                                          stats(6)=stats(6)+1
79                                          if(i/=stats(2)-stats(6)) j=j-1
80                                  end if
81                                  j=j+1
82                          end do
83                  end do
84                  stats(2)=stats(2)-stats(6)
85                  write(*,*)
86
87          end subroutine remdup
88 !*****************************************************************************
89          subroutine sort()
90
91                  real(kind=p18),dimension(N+Nres+1) :: tempsav
92                  integer,dimension(1) :: imin
93
94                  write(unit=*,fmt='("Sorting results:     ")',advance='no')
95                  do i=1,stats(2)-1
96                          if(mod(i,floor(stats(2)/10.0))==0)
   write(unit=*,fmt='("*")',advance='no')
97                          imin = i-1+minloc(sav(i:stats(2),N+1))
98                          tempsav = sav(imin(1),:)
99                          sav(imin(1),:) = sav(i,:)
100                         sav(i,:) = tempsav
101                 end do
102                 write(*,*)
103
104         end subroutine sort
105 !*****************************************************************************
106         subroutine cmvsd()
107
108                 do i=1,N+Nres+1
109                         do j=1,stats(2)
110                                 mvsd(1,i)=mvsd(1,i)+sav(j,i)
111                         end do
112                 end do
113                 mvsd(1,:)=mvsd(1,:)/stats(2)
114                 do i=1,N+Nres+1
115                         do j=1,stats(2)
116                                 mvsd(2,i)=mvsd(2,i)+(sav(j,i)-mvsd(1,i))**2.0_p18
117                         end do
118                 end do
119                 mvsd(2,:)=sqrt(mvsd(2,:)/(real(stats(2),p18)*real(stats(2)-1,p18)))
120
121         end subroutine cmvsd
122 !*****************************************************************************
123         subroutine wdata()
124
125                 open(unit=fout,file="nmm.txt",status="replace",action="write",form="formatted"
   ,position="rewind")
126                 write(unit=fout,fmt='("             rc            ")',advance='no')
127                 write(unit=fout,fmt='("             M0            ")',advance='no')
```

dnmm.f90 3

```
128                 write(unit=fout,fmt='("             M1            ")',advance='no')
129                 write(unit=fout,fmt='("             M2            ")',advance='no')
130                 write(unit=fout,fmt='("             M4            ")',advance='no')
131                 write(unit=fout,fmt='("             y3            ")',advance='no')
132                 write(unit=fout,fmt='("           Chi^2           ")',advance='no')
133                 write(unit=fout,fmt='("           Chi^2m          ")',advance='no')
134                 write(unit=fout,fmt='("           Chi^2a          ")',advance='no')
135                 write(unit=fout,fmt='("           mass1           ")',advance='no')
136                 write(unit=fout,fmt='("           mass2           ")',advance='no')
137                 write(unit=fout,fmt='("           mass3           ")',advance='no')
138                 write(unit=fout,fmt='("           dm2atm          ")',advance='no')
139                 write(unit=fout,fmt='("           dm2sol          ")',advance='no')
140                 write(unit=fout,fmt='("            mbb            ")',advance='no')
141                 write(unit=fout,fmt='("              R            ")',advance='no')
142                 write(unit=fout,fmt='("           epsilo          ")',advance='no')
143                 write(unit=fout,fmt='("           sin213          ")',advance='no')
144                 write(unit=fout,fmt='("           sin2sol         ")',advance='no')
145                 write(unit=fout,fmt='("           sin2atm         ")',advance='yes')
146                 do i=1,stats(2)
147                         do j=1,N+Nres+1
148                                 write(unit=fout,fmt='(ES26.18,TR1)',advance='no') sav(i,j)
149                         end do
150                         write(unit=fout,fmt=*)
151                 end do
152                 write(unit=fout,fmt=*)
153                 do i=1,N+Nres+1
154                         write(unit=fout,fmt='(ES26.18,TR1)',advance='no') mvsd(1,i)
155                 end do
156                 write(unit=fout,fmt=*)
157                 do i=1,N+Nres+1
158                         write(unit=fout,fmt='(ES26.18,TR1)',advance='no') mvsd(2,i)
159                 end do
160                 write(unit=fout,fmt=*)
161                 write(unit=fout,fmt=*)
162                 write(unit=fout,fmt='("Constants:")')
163                 write(unit=fout,fmt='("              v0t           ")',advance='no')
164                 write(unit=fout,fmt='("              vct           ")',advance='no')
165                 write(unit=fout,fmt='("              r0            ")',advance='no')
166                 write(unit=fout,fmt='("              y2            ")',advance='yes')
167                 do i=1,Nc
168                         write(unit=fout,fmt='(ES26.18,TR1)',advance='no') const(i)
169                 end do
170                 write(unit=fout,fmt=*)
171                 write(unit=fout,fmt=*)
172                 write(unit=fout,fmt='("Stable minimum reached: "I10)') stats(1)
173                 write(unit=fout,fmt='("Criterion fulfilled:    "I10)') stats(2)
174                 write(unit=fout,fmt='("No minimum reached:     "I10)') stats(3)
175                 write(unit=fout,fmt='("Infinity detected:      "I10)') stats(4)
176                 write(unit=fout,fmt='("Wrong mass order:       "I10)') stats(5)
177                 write(unit=fout,fmt='("Duplicates removed:     "I10)') stats(6)
178                 close(fout)
179
180         end subroutine wdata
181 !*****************************************************************************
182         subroutine clist(lin,lout,lstart,l)
183
184                 implicit none
185                 real(kind=p18),dimension(stats(2)),intent(in) :: lin
186                 real(kind=p18),intent(in) :: lstart,l
187                 type(list1),dimension(nch),intent(out) :: lout
188                 real(kind=p18) :: step
189                 integer :: tmp
190
191                 lout%c=0
192                 lout%n=0
193                 !lstart=minval(lin)
194                 !l=maxval(lin)-lstart
195                 !lstart=lstart-0.05_p18*l
```

dnmm.f90 4

```
196                 !l=1.1_p18*l
197                 step=l/(nch-1)
198                 do i=1,nch
199                         lout(i)%c=lstart + i*step - 0.5_p18*step
200                 end do
201                 do i=1,stats(2)
202                         tmp=floor((lin(i)-lstart)/step)+1
203                         !write(*,*) i,(lin(i)-lstart)/step,tmp
204                         lout(tmp)%n=lout(tmp)%n+1
205                 end do
206
207         end subroutine clist
208 !*****************************************************************************
209         subroutine wlist()
210
211                 open(unit=fout,file="list.txt",status="replace",action="write",form="formatted
    ",position="rewind")
212                 do i=1,nch
213                         write(unit=fout,fmt='(9(ES27.19,TR1,I6))')
    lM1(i)%c,lM1(i)%n,lM2(i)%c,lM2(i)%n,lM3(i)%c,lM3(i)%n,lmbb(i)%c,lmbb(i)%n, &
214 lR(i)%c,lR(i)%n,lepsilo(i)%c,lepsilo(i)%n,lsin213(i)%c,lsin213(i)%n,lsin2sol(i)%c,lsin2sol(i)%
    n,lsin2atm(i)%c,lsin2atm(i)%n
215                 end do
216                 close(fout)
217
218         end subroutine wlist
219 !*****************************************************************************
220         subroutine wscreen()
221
222                 write(unit=*,fmt='("Stable minimum reached: "I10)') stats(1)
223                 write(unit=*,fmt='("Criterion fulfilled:    "I10)') stats(2)
224                 write(unit=*,fmt='("No minimum reached:     "I10)') stats(3)
225                 write(unit=*,fmt='("Infinity detected:      "I10)') stats(4)
226                 write(unit=*,fmt='("Wrong mass order:       "I10)') stats(5)
227                 write(unit=*,fmt='("Duplicates removed:     "I10)') stats(6)
228
229         end subroutine wscreen
230 !*****************************************************************************
231         subroutine finish()
232
233                 deallocate(sav,lM1,lM2,lM3,lmbb,lR,lepsilo,lsin213,lsin2sol,lsin2atm,mvsd,cons
    t)
234
235         end subroutine finish
236 !*****************************************************************************
237
238 end module dnmm_sub
239
240 !*****************************************************************************
241 !*****************************************************************************
242
243 program dnmm
244
245         use dnmm_var
246         use dnmm_sub
247
248         call readin()
249         call remdup()
250         call sort()
251         call cmvsd()
252         call wdata()
253
254         lstarttemp=min(minval(sav(:,N+4)),minval(sav(:,N+5)),minval(sav(:,N+6)))
255         ltemp=max(maxval(sav(:,N+4)),maxval(sav(:,N+5)),maxval(sav(:,N+6)))-lstarttemp
256         call clist(sav(:,N+4),lM1,lstarttemp,ltemp)
257         call clist(sav(:,N+5),lM2,lstarttemp,ltemp)
258         call clist(sav(:,N+6),lM3,lstarttemp,ltemp)
259
```

dnmm.f90 5

```
260         lstarttemp=minval(sav(:,N+9))
261         ltemp=maxval(sav(:,N+9))-lstarttemp
262         call clist(sav(:,N+9),lmbb,lstarttemp,ltemp)
263
264         lstarttemp=minval(sav(:,N+10))
265         ltemp=maxval(sav(:,N+10))-lstarttemp
266         call clist(sav(:,N+10),lR,lstarttemp,ltemp)
267
268         lstarttemp=minval(sav(:,N+11))
269         ltemp=maxval(sav(:,N+11))-lstarttemp
270         call clist(sav(:,N+11),lepsilo,lstarttemp,ltemp)
271
272         lstarttemp=min(minval(sav(:,N+12)),minval(sav(:,N+13)),minval(sav(:,N+14)))
273         ltemp=max(maxval(sav(:,N+12)),maxval(sav(:,N+13)),maxval(sav(:,N+14)))-lstarttemp
274         call clist(sav(:,N+12),lsin213,lstarttemp,ltemp)
275         call clist(sav(:,N+13),lsin2sol,lstarttemp,ltemp)
276         call clist(sav(:,N+14),lsin2atm,lstarttemp,ltemp)
277
278         call wlist()
279         call wscreen()
280         call finish()
281
282 end program dnmm
283
284
```

A.2 Program for pinning term method

nmm.f90 1

```
!*******************************************************************************
!
! file: nmm.f90
! Nelder Mead Method
!
! Modified Model, CP-invariance, nonzero theta13, inverted spectrum
!
! constants:  v0t,vct,r0,y2
! parameters: rc,M0,M1,M2,M4,y3
! res values:
chi2,chi2m,chi2a,mass1,mass2,mass3,dm2atm,dm2sol,mbb,R,epsilo,sin213,sin2sol,sin2atm
!
!*******************************************************************************

module nmm_var

        implicit none
        integer,parameter :: N=6,Nres=14,Nc=4,fout=11,p18=selected_real_kind(p=18)
        real(kind=p18),parameter :: rho=1.0_p18,chi=2.0_p18,gam=0.5_p18,sig=0.5_p18
        real(kind=p18),parameter :: eps=1.0E-6_p18,crit=1.0E2_p18
        complex(kind=p18),parameter :: c1=(1.0_p18,0.0_p18),ci=(0.0_p18,1.0_p18)
        real(kind=p18) :: c_v0t,c_vct,c_r0,c_y2,startpin,stoppin
        complex(kind=p18) :: c_a
        integer :: i,j,k,loop,nrand,nloop=10000,ipin,npin=10
        integer,dimension(6) :: stats
        integer,dimension(:),allocatable :: seed
        real :: tstart,tstop
        real(kind=p18) :: fr,fn,d,tmp,vpin
        real(kind=p18),dimension(Nres) :: res
        real(kind=p18),dimension(N) :: xc,xr,xn
        real(kind=p18),dimension(N+1) :: f
        real(kind=p18),dimension(N+1,N) :: x
        real(kind=p18),dimension(:,:),allocatable :: sav
        real(kind=p18),dimension(:,:),allocatable :: respin
        character(len=4) :: nstr

end module nmm_var

!*******************************************************************************
!*******************************************************************************

module nmm_sub

        use nmm_var

        implicit none

        contains

!*******************************************************************************
        subroutine init()

                write(unit=*,fmt='("nloop = ")',advance='no')
                read(*,*) nloop
                write(unit=*,fmt='(" npin = ")',advance='no')
                read(*,*) npin
                call random_seed(size=nrand)
                allocate(seed(nrand),sav(nloop,N+Nres+1),respin(1:npin,1:N+Nres+2))
                startpin=3.5E-2_p18
                stoppin=6.0E-2_p18
                seed=0
                xc=0
                x=0
                f=0
                sav=0
                res=0
                stats=0

```

nmm.f90 2

```
68          end subroutine init
69  !*****************************************************************************
70          subroutine initf()
71
72                  c_v0t=1.7E11_p18
73                  c_vct=2.0E25_p18
74                  c_r0=0.0E0_p18
75                  c_y2=1.0E-3_p18
76
77                  c_a= -1.0_p18*(c_y2*c_v0t*(c1+c_r0*ci)/(1.0_p18+c_r0**2.0_p18))**2.0_p18
78
79                  stats=0
80                  sav=0
81
82                  !vpin = 10.0_p18**(log10(startpin) +
    (ipin-1)*(log10(stoppin)-log10(startpin))/(npin-1))
83                  vpin = startpin + (ipin-1) * (stoppin-startpin)/(npin-1)
84                  write(unit=*,fmt='(" ipin = "I10)') ipin
85                  write(unit=*,fmt='(" vpin = "ES10.3)') vpin
86
87
88          end subroutine initf
89  !*****************************************************************************
90          subroutine prepf()
91
92                  call random_seed
93                  call random_seed(get=seed)
94                  !write(unit=*,fmt='(4(Z8.8,TR1))') seed
95                  do i=1,N+1
96                          do j=1,1
97                                  call random_number(x(i,j))
98                                  x(i,j)=1.0E-1_p18  *  2.0_p18*(x(i,j)-0.5_p18)
99                          end do
100                         do j=2,N-1
101                                 call random_number(x(i,j))
102                                 x(i,j)=1.0E19_p18  *  2.0_p18*(x(i,j)-0.5_p18)
103                         end do
104                         do j=N,N
105                                 call random_number(x(i,j))
106                                 x(i,j)=2.0_p18  *   1.0E-06_p18  *  2.0_p18*(x(i,j)-0.5_p18)
107                         end do
108                         call calcf(f(i),x(i,:),.FALSE.)
109                 end do
110
111         end subroutine prepf
112 !*****************************************************************************
113
114         subroutine calcf(a,b,sres)
115
116                 implicit none
117                 logical,intent(in) :: sres
118                 real(kind=p18),intent(inout) :: a
119                 real(kind=p18),dimension(N),intent(in) :: b
120                 real(kind=p18) :: rc,M0,M1,M2,M4,y3
121                 real(kind=p18) :: MN2,x,yt,z,detM
122                 real(kind=p18) :: mass1,mass2,mass3,dm2atm,dm2sol,chi2m,chi2pin,mbb,R
123                 real(kind=p18),parameter ::
    MV_dm2atm=2.4E-3_p18,SD_dm2atm=0.12E-3_p18,MV_dm2sol=7.65E-5_p18,SD_dm2sol=0.23E-5_p18
124                 real(kind=p18) :: epsilo,sin213,sin2sol,sin2atm,chi2a
125                 real(kind=p18),parameter ::
    MV_sin213=1.0E-2_p18,SD_sin213=1.6E-2_p18,MV_sin2sol=3.04E-1_p18,SD_sin2sol=0.22E-1_p18, &
126 MV_sin2atm=5.0E-1_p18,SD_sin2atm=0.7E-1_p18
127
128                 rc=b(1)
129                 M0=b(2)
130                 M1=b(3)
131                 M2=b(4)
132                 M4=b(5)
```

nmm.f90 3

```
133                 y3=b(6)
134
135                 !MASSES
136                 MN2=(y3*c_vct)**2.0_p18
137
138                 detM = (M0+2.0_p18*M1)*( (M0-M1)**2.0_p18*MN2 - ( (M0-M1)*M4 -
    3.0_p18*M2**2.0_p18)**2.0_p18 )
139                 x = 1.0_p18/detM * ( (M0**2.0_p18-M1**2.0_p18)*(MN2-M4**2.0_p18) + &
140 (4.0_p18*M0+2.0_p18*M1)*M2**2.0_p18*M4 - 3.0_p18*M2**4.0_p18 )
141                 z = 1.0_p18/detM * ( (M1**2.0_p18-M0*M1)*(MN2-M4**2.0_p18) +
    (M0-4.0_p18*M1)*M2**2.0_p18*M4 - 3.0_p18*M2**4.0_p18 )
142                 yt = 1.0_p18/detM * ( (M0+2.0_p18*M1)*M2**2.0_p18*y3*c_vct )
143
144                 mass1 = abs( c_a * (x+3.0_p18*yt-z) )
145                 mass2 = abs( c_a * (x+2.0_p18*z) )
146                 mass3 = abs( c_a * (x-3.0_p18*yt-z) )
147
148                 dm2atm = mass2**2.0_p18 - mass3**2.0_p18
149                 dm2sol = mass2**2.0_p18 - mass1**2.0_p18
150
151                 chi2m = ((dm2atm-MV_dm2atm)/SD_dm2atm)**2.0_p18 +
    ((dm2sol-MV_dm2sol)/SD_dm2sol)**2.0_p18
152
153                 !ANGLES
154                 epsilo = atan(rc)
155
156                 sin213 = 2.0_p18/3.0_p18 * sin(epsilo/2.0_p18)**2.0_p18
157                 sin2sol = 1.0_p18 / (3.0_p18 * (1.0_p18 - sin213))
158                 sin2atm = 0.5_p18 + sin(epsilo) / (sqrt(12.0_p18)*(1-sin213))
159
160                 chi2a = ((sin213-MV_sin213)/SD_sin213)**2.0_p18 +
    ((sin2sol-MV_sin2sol)/SD_sin2sol)**2.0_p18 + &
161 ((sin2atm-MV_sin2atm)/SD_sin2atm)**2.0_p18
162
163                 mbb = abs( ( sqrt(mass3**2.0_p18 + dm2atm -
    dm2sol)*(1.0_p18-sin2sol)*exp(ci*0.0_p18) + &
164 sqrt(mass3**2.0_p18 + dm2atm)*sin2sol*exp(ci*0.0_p18) )*(1.0_p18-sin213) + mass3*sin213)
165
166                 chi2pin = (mbb-vpin)**2.0_p18/(0.01_p18*vpin)**2.0_p18
167
168                 a = chi2m + chi2a + chi2pin
169
170                 if(sres) then
171
172                         R = mass3/sqrt(dm2sol)
173
174                         res(1)=chi2m
175                         res(2)=chi2a
176                         res(3)=chi2pin
177                         res(4)=mass1
178                         res(5)=mass2
179                         res(6)=mass3
180                         res(7)=dm2atm
181                         res(8)=dm2sol
182                         res(9)=mbb
183                         res(10)=R
184                         res(11)=epsilo
185                         res(12)=sin213
186                         res(13)=sin2sol
187                         res(14)=sin2atm
188                 end if
189
190         end subroutine calcf
191 !*****************************************************************************
192         subroutine sort()
193
194                 integer,dimension(1) :: imin
195                 real(kind=p18),dimension(N) :: vtemp
```

nmm.f90 4

```
196                 real(kind=p18) :: stemp
197
198                 do i=1,N
199                         imin=i-1+minloc(f(i:N+1))
200                         vtemp=x(i,:)
201                         stemp=f(i)
202                         x(i,:)=x(imin(1),:)
203                         f(i)=f(imin(1))
204                         x(imin(1),:)=vtemp
205                         f(imin(1))=stemp
206                 end do
207
208         end subroutine sort
209 !*****************************************************************************
210         subroutine centroid()
211
212                 xc=0
213                 do j=1,N
214                         do i=1,N
215                                 xc(j) = xc(j) + x(i,j)
216                         end do
217                         xc(j)=xc(j)/N
218                 end do
219                 !write(*,*) "centroid =", xc
220
221         end subroutine centroid
222 !*****************************************************************************
223         subroutine shrinkage()
224
225                 do j=2,N+1
226                         x(j,:) = x(1,:) + sig*(x(j,:)-x(1,:))
227                         call calcf(f(j),x(j,:),.FALSE.)
228                 end do
229
230         end subroutine shrinkage
231 !*****************************************************************************
232         subroutine show()
233
234                 do i=1,N+1
235                         write(unit=*,fmt='("x("I2.2") =")',advance='no') i
236                         do j=1,N
237                                 write(unit=*,fmt='(ES10.3,TR1)',advance='no') x(i,j)
238                         end do
239                         write(*,*)
240                 end do
241                 write(*,*) "f =",f
242
243         end subroutine show
244 !*****************************************************************************
245         subroutine calcd()
246
247                 real(kind=p18) :: fm
248
249                 fm=0.0_p18
250                 d=0.0_p18
251                 do j=1,N+1
252                         fm = fm + f(j)
253                 end do
254                 fm=fm/(N+1)
255                 do j=1,N+1
256                         d = d + (f(j)-fm)**2.0**p18
257                 end do
258                 d=d/(N+1)
259
260         end subroutine calcd
261 !*****************************************************************************
262         subroutine sortpin()
263
```

nmm.f90 5

```
264                 real(kind=p18),dimension(N+Nres+1) :: tempsav
265                 integer,dimension(1) :: imin
266
267                 do i=1,stats(2)
268                         if(sav(i,N+1)==0.0_p18) then
269                                 do j=i+1,nloop
270                                         if(sav(j,N+1)/=0.0_p18) then
271                                                 sav(i,:)=sav(j,:)
272                                                 sav(j,:)=0
273                                                 exit
274                                         end if
275                                 end do
276                         end if
277                 end do
278                 do i=1,stats(2)-1
279                         imin = i-1+minloc(sav(i:stats(2),N+1))
280                         tempsav = sav(imin(1),:)
281                         sav(imin(1),:) = sav(i,:)
282                         sav(i,:) = tempsav
283                 end do
284
285         end subroutine sortpin
286 !*****************************************************************************
287         subroutine wfile()
288
289                 write(unit=nstr,fmt='(I4.4)') ipin
290                 open(unit=fout,file="data/nmm-" // nstr //
    ".dat",status="replace",action="write",form="unformatted",position="rewind")
291                 write(unit=fout) N,Nres,stats(2),vpin
292                 do loop=1,stats(2)
293                         write(unit=fout) sav(loop,:)
294                 end do
295                 close(fout)
296
297         end subroutine wfile
298 !*****************************************************************************
299         subroutine finish()
300
301                 !write(*,*) "Steps: ",k
302                 deallocate(seed,sav,respin)
303
304         end subroutine finish
305 !*****************************************************************************
306 end module nmm_sub
307
308 !*****************************************************************************
309 !*****************************************************************************
310
311 program nmm
312
313         use nmm_var
314         use nmm_sub
315
316         implicit none
317
318         write(*,*) huge(fn)
319         write(*,*) tiny(fn)
320
321         call cpu_time(tstart)
322
323         call init()
324
325         do ipin=1,npin
326
327 30              call initf()
328                 do loop=1,nloop
329                         call prepf()
330                         k=1
```

nmm.f90 6

```
331                         do
332                                 k=k+1
333                                 call sort()
334                                 call centroid()
335                                 xr = xc + rho*(xc-x(N+1,:))
336                                 call calcf(fr,xr,.FALSE.)
337                                 if(f(1)<=fr .AND. fr<f(N)) then
338                                         x(N+1,:)=xr
339                                         f(N+1)=fr
340                                         !write(*,*) "Reflection accepted"
341                                         go to 10
342                                 else if(fr<f(1)) then
343                                         xn = xc + chi*(xr-xc)
344                                         call calcf(fn,xn,.FALSE.)
345                                         if(fn<fr) then
346                                                 x(N+1,:)=xn
347                                                 f(N+1)=fn
348                                                 !write(*,*) "Expansion accepted"
349                                                 go to 10
350                                         else
351                                                 x(N+1,:)=xr
352                                                 f(N+1)=fr
353                                                 !write(*,*) "Expansion rejected -> Reflection"
354                                                 go to 10
355                                         end if
356                                 else if(f(N)<=fr .AND. fr<f(N+1)) then
357                                         xn = xc + gam*(xr-xc)
358                                         call calcf(fn,xn,.FALSE.)
359                                         if(fn<=fr) then
360                                                 x(N+1,:)=xn
361                                                 f(N+1)=fn
362                                                 !write(*,*) "Outside contraction accepted"
363                                                 go to 10
364                                         else
365                                                 !write(*,*) "Outside contraction rejected ->
    Shrinkage"
366                                                 call shrinkage()
367                                                 go to 10
368                                         end if
369                                 else if(fr>=f(N+1)) then
370                                         xn = xc - gam*(xc-x(N+1,:))
371                                         call calcf(fn,xn,.FALSE.)
372                                         if(fn<f(N+1)) then
373                                                 x(N+1,:)=xn
374                                                 f(N+1)=fn
375                                                 !write(*,*) "Inside contraction accepted"
376                                                 go to 10
377                                         else
378                                                 !write(*,*) "Inside contraction rejected ->
    Shrinkage"
379                                                 call shrinkage()
380                                                 go to 10
381                                         end if
382                                 end if
383 10                              call calcd()
384                                 !call show()
385                                 if(d<=eps) then
386                                         !write(*,*) "Minimum reached: stats(1)",x(1,1),f(1)
387                                         stats(1)=stats(1)+1
388                                         exit
389                                 end if
390                                 if(mod(k,10)==0) then
391                                         tmp=0.0_p18
392                                         do i=1,N+1
393                                                 do j=1,N
394                                                         tmp=tmp+x(i,j)
395                                                 end do
396                                                 tmp=tmp+f(i)
```

nmm.f90 7

```
397                                     end do
398                                     if(tmp*0.0_p18 /= 0.0_p18 .OR. tmp >= 1E99_p18) then
399                                             !write(*,*) "Detect infinity: stats(4)"
400                                             stats(4)=stats(4)+1
401                                             go to 20
402                                     end if
403                             end if
404                             if(k>=1000) then
405                                     !write(*,*) "No minimum reached: stats(3)"
406                                     stats(3)=stats(3)+1
407                                     go to 20
408                             end if
409                     end do
410                     call sort()
411                     !call show()
412                     if(f(1)<=crit) then
413                             call calcf(f(1),x(1,:),.TRUE.)
414                             if((res(6)<=res(4)) .AND. (res(4)<=res(5))) then
415                                     !write(*,*) "Criterion fulfilled: stats(2)"
416                                     sav(loop,1:N)=x(1,:)
417                                     sav(loop,N+1)=f(1)
418                                     sav(loop,N+2:N+Nres+1) = res
419                                     stats(2)=stats(2)+1
420                             else
421                                     !write(*,*) "Wrong mass order: stats(5)"
422                                     stats(5)=stats(5)+1
423                             end if
424                     end if
425 20                  if(mod(loop,1000)==0) write(unit=*,fmt='(7(I10,TR1))') loop,stats
426             end do
427
428             call sortpin()
429             call wfile()
430
431         end do
432
433         call finish()
434
435         call cpu_time(tstop)
436
437         write(*,*) "Time: ",tstop-tstart,"sec"
438
439 end program nmm
440
441
```

dnmm.f90 1

```
!*******************************************************************************
!
! file: dnmm.f90
! Data Processing for Nelder Mead Method
!
! Modified Model, CP-invariance, nonzero theta13
!
! constants:  v0t,vct,r0,y2
! parameters: rc,M0,M1,M2,M4,y3
! res values:
chi2,chi2m,chi2a,mass1,mass2,mass3,dm2atm,dm2sol,mbb,R,epsilo,sin213,sin2sol,sin2atm

!*******************************************************************************

module dnmm_var

        implicit none
        integer,parameter :: fin=11,fout=12,p18=selected_real_kind(p=18)
        integer :: i,j,ipin,N,Nres,stats2,npin=10
        real(kind=p18),parameter :: critpin=1.0E-2_p18
        real(kind=p18) :: vpin
        real(kind=p18),dimension(:,:),allocatable :: sav,respin
        character(len=4) :: nstr

end module dnmm_var

!*******************************************************************************
!*******************************************************************************

module dnmm_sub

        use dnmm_var

        contains
!*******************************************************************************
        subroutine readin()

                write(unit=nstr,fmt='(I4.4)') ipin
                open(unit=fin,file="data/nmm-" // nstr //
".dat",status="old",action="read",form="unformatted",position="rewind")
                read(unit=fin) N,Nres,stats2,vpin
                allocate(sav(1:stats2,1:N+Nres+1))
                do i=1,stats2
                        read(unit=fin) sav(i,:)
                end do
                close(fin)

        end subroutine readin
!*******************************************************************************
        subroutine init()

                allocate(respin(1:npin,1:N+Nres+2))

        end subroutine init
!*******************************************************************************
        subroutine find()

                if(stats2==0) then
                        write(*,*) " ipin = ",ipin,": No data"
                        respin(ipin,1:N+Nres+1)=0
                        respin(ipin,N+Nres+2)=vpin
                end if
                do i=1,stats2
                        if((sav(i,N+1+3)/sav(i,N+1+1) <= critpin) .AND.
(sav(i,N+1+3)/sav(i,N+1+2) <= critpin)) then
                                respin(ipin,1:N+Nres+1)=sav(i,:)
                                respin(ipin,N+Nres+2)=vpin
                                write(unit=*,fmt='(TR1,I10,I10,I10,ES14.3)')
```

dnmm.f90 2

```
   ipin,i,stats2,vpin
 66                              exit
 67                      end if
 68                      if(i==stats2) then
 69                              write(*,*) " ipin = ",ipin,": Nothing found"
 70                              respin(ipin,1:N+Nres+1)=0
 71                              respin(ipin,N+Nres+2)=vpin
 72                      end if
 73              end do
 74
 75      end subroutine find
 76 !*****************************************************************************
 77      subroutine wdata()
 78
 79              open(unit=fout,file="data/nmm-" // nstr //
   ".txt",status="replace",action="write",form="formatted",position="rewind")
 80              write(unit=fout,fmt='("             rc            ")',advance='no')
 81              write(unit=fout,fmt='("             M0            ")',advance='no')
 82              write(unit=fout,fmt='("             M1            ")',advance='no')
 83              write(unit=fout,fmt='("             M2            ")',advance='no')
 84              write(unit=fout,fmt='("             M4            ")',advance='no')
 85              write(unit=fout,fmt='("             y3            ")',advance='no')
 86              write(unit=fout,fmt='("           Chi^2           ")',advance='no')
 87              write(unit=fout,fmt='("           Chi^2m          ")',advance='no')
 88              write(unit=fout,fmt='("           Chi^2a          ")',advance='no')
 89              write(unit=fout,fmt='("           Chi^2pin        ")',advance='no')
 90              write(unit=fout,fmt='("            mass1          ")',advance='no')
 91              write(unit=fout,fmt='("            mass2          ")',advance='no')
 92              write(unit=fout,fmt='("            mass3          ")',advance='no')
 93              write(unit=fout,fmt='("           dm2atm          ")',advance='no')
 94              write(unit=fout,fmt='("           dm2sol          ")',advance='no')
 95              write(unit=fout,fmt='("            mbb            ")',advance='no')
 96              write(unit=fout,fmt='("             R             ")',advance='no')
 97              write(unit=fout,fmt='("           epsilo          ")',advance='no')
 98              write(unit=fout,fmt='("           sin213          ")',advance='no')
 99              write(unit=fout,fmt='("           sin2sol         ")',advance='no')
100              write(unit=fout,fmt='("           sin2atm         ")',advance='no')
101              write(unit=fout,fmt='("           vpin            ")',advance='yes')
102              do i=1,stats2
103                      do j=1,N+Nres+1
104                              if(sav(i,11)/=0.0_p18)
   write(unit=fout,fmt='(ES26.18,TR1)',advance='no') sav(i,j)
105                      end do
106                      if(sav(i,11)/=0.0_p18)
   write(unit=fout,fmt='(ES26.18,TR1)',advance='no') vpin
107                      write(unit=fout,fmt=*)
108              end do
109              close(fout)
110
111      end subroutine wdata
112 !*****************************************************************************
113      subroutine wpindata()
114
115              open(unit=fout,file="pin.txt",status="replace",action="write",form="formatted"
   ,position="rewind")
116              write(unit=fout,fmt='("             rc            ")',advance='no')
117              write(unit=fout,fmt='("             M0            ")',advance='no')
118              write(unit=fout,fmt='("             M1            ")',advance='no')
119              write(unit=fout,fmt='("             M2            ")',advance='no')
120              write(unit=fout,fmt='("             M4            ")',advance='no')
121              write(unit=fout,fmt='("             y3            ")',advance='no')
122              write(unit=fout,fmt='("           Chi^2           ")',advance='no')
123              write(unit=fout,fmt='("           Chi^2m          ")',advance='no')
124              write(unit=fout,fmt='("           Chi^2a          ")',advance='no')
125              write(unit=fout,fmt='("           Chi^2pin        ")',advance='no')
126              write(unit=fout,fmt='("            mass1          ")',advance='no')
127              write(unit=fout,fmt='("            mass2          ")',advance='no')
128              write(unit=fout,fmt='("            mass3          ")',advance='no')
```

dnmm.f90 3

```
129                 write(unit=fout,fmt='("            dm2atm         ")',advance='no')
130                 write(unit=fout,fmt='("            dm2sol         ")',advance='no')
131                 write(unit=fout,fmt='("             mbb           ")',advance='no')
132                 write(unit=fout,fmt='("              R            ")',advance='no')
133                 write(unit=fout,fmt='("            epsilo         ")',advance='no')
134                 write(unit=fout,fmt='("            sin213         ")',advance='no')
135                 write(unit=fout,fmt='("            sin2sol        ")',advance='no')
136                 write(unit=fout,fmt='("            sin2atm        ")',advance='no')
137                 write(unit=fout,fmt='("            vpin           ")',advance='yes')
138                 do ipin=1,npin
139                         do j=1,N+Nres+2
140                                 if(respin(ipin,11)/=0.0_p18)
    write(unit=fout,fmt='(ES26.18,TR1)',advance='no') respin(ipin,j)
141                         end do
142                         write(unit=fout,fmt=*)
143                 end do
144                 close(fout)
145
146         end subroutine wpindata
147 !*****************************************************************************
148         subroutine finish()
149
150                 deallocate(respin)
151
152         end subroutine finish
153 !*****************************************************************************
154
155 end module dnmm_sub
156
157 !*****************************************************************************
158 !*****************************************************************************
159
160 program dnmm
161
162         use dnmm_var
163         use dnmm_sub
164
165
166         write(unit=*,fmt='(" npin = ")',advance='no')
167         read(*,*) npin
168
169         do ipin=1,npin
170                 call readin()
171                 if(ipin==1) call init()
172                 !call wdata()
173                 call find()
174                 deallocate(sav)
175         end do
176         call wpindata()
177         call finish()
178
179 end program dnmm
180
181
```

A.3 Program for fine-tuning properties

nmm3finetune.nb 1

```
In[80]:= Clear["Global`*"]

In[81]:= MVdm2atm = 2.4*^-3
         SDdm2atm = 0.12*^-3
         MVdm2sol = 7.65*^-5
         SDdm2sol = 0.23*^-5

Out[81]= 0.0024

Out[82]= 0.00012

Out[83]= 0.0000765

Out[84]= 2.3×10^-6

In[85]:= str = OpenRead["nmm.txt"]

Out[85]= InputStream[nmm.txt, 16]

In[86]:= SetStreamPosition[str, 0];
         Read[str, String]
         start = StreamPosition[str];

Out[87]=      rc          M0          M1          M2          M4
              y3        Chi^2       Chi^2m      Chi^2a      mass1
              mass2     mass3       dm2atm      dm2sol      mbb
              R         epsilo      sin213      sin2sol     sin2atm

In[89]:= SetStreamPosition[str, start];
         data1 = SetPrecision[ReadList[str, Real, 18], 19]

Out[90]= {0.0009504318008238979248, -1.100141136440566817×10^19, 7.105865183385642507×10^18,
           3.744470897331724208×10^18, -5.750681316081949160×10^18, -1.750449985696605240×10^-7,
           2.168436609718819464, 0.00002414827643799047893, 2.168412461442381474, 0.00213116121354992 0478,
           0.009002220644959019749, 0.04904205743944022251, 0.002400581549775214504, 0.000076498128422386 81975,
           0.004421521419944718966, 0.2436637358659582784, 0.000950431514642508 9665, 1.505533326711212685×10^-7}

In[91]:= Vrc = data1[[1]]
         VM0 = data1[[2]]
         VM1 = data1[[3]]
         VM2 = data1[[4]]
         VM4 = data1[[5]]
         Vy3 = data1[[6]]

Out[91]= 0.0009504318008238979248

Out[92]= -1.100141136440566817×10^19

Out[93]= 7.105865183385642507×10^18

Out[94]= 3.744470897331724208×10^18

Out[95]= -5.750681316081949160×10^18

Out[96]= -1.750449985696605240×10^-7
```

```
In[97]:= SetStreamPosition[str, start]
         Find[str, "Constants"];
         StreamPosition[str];
         Skip[str, String];
         Read[str, String]
         data2 = SetPrecision[ReadList[str, Real, 4], 19]
```

```
Out[97]= 541
```

```
Out[101]=
          v0t              vct              r0               y2
```

```
Out[102]=
{1.700000000000000000 × 10^11, 2.000000000000000000 × 10^25, 0, 0.001000000000000000000}
```

```
In[103]:=
v0t = data2[[1]]
vct = data2[[2]]
r0 = data2[[3]]
y2 = data2[[4]]
```

```
Out[103]=
1.700000000000000000 × 10^11
```

```
Out[104]=
2.000000000000000000 × 10^25
```

```
Out[105]=
0
```

```
Out[106]=
0.001000000000000000000
```

```
In[107]:=
detM = SetPrecision[
    (M0 + 2 M1) ((M0 - M1) ^2 * y3^2 * vct^2 - ((M0 - M1) * M4 - 3 M2 * M3)^2) /. M3 → M2, 19];
```

```
In[108]:=
MN2 = y3^2 * vct^2
```

```
Out[108]=
4.000000000000000000 × 10^50 y3^2
```

```
In[109]:=
replxzyt = SetPrecision[
    ({x → ((M0^2 - M1^2) * (MN2 - M4^2) + (4 * M0 + 2 * M1) * M2 * M3 * M4 - 3 * M2^2 * M3^2) / detM,
       z → ((M1^2 - M0 * M1) * (MN2 - M4^2) + (M0 - 4 * M1) * M2 * M3 * M4 - 3 M2^2 * M3^2) / detM,
       y → ((M0 + 2 * M1) * M2^2 * y3 * vct) / detM,
       t → ((M0 + 2 M1) * M3^2 * y3 * vct) / detM}) /. M3 → M2, 19];
```

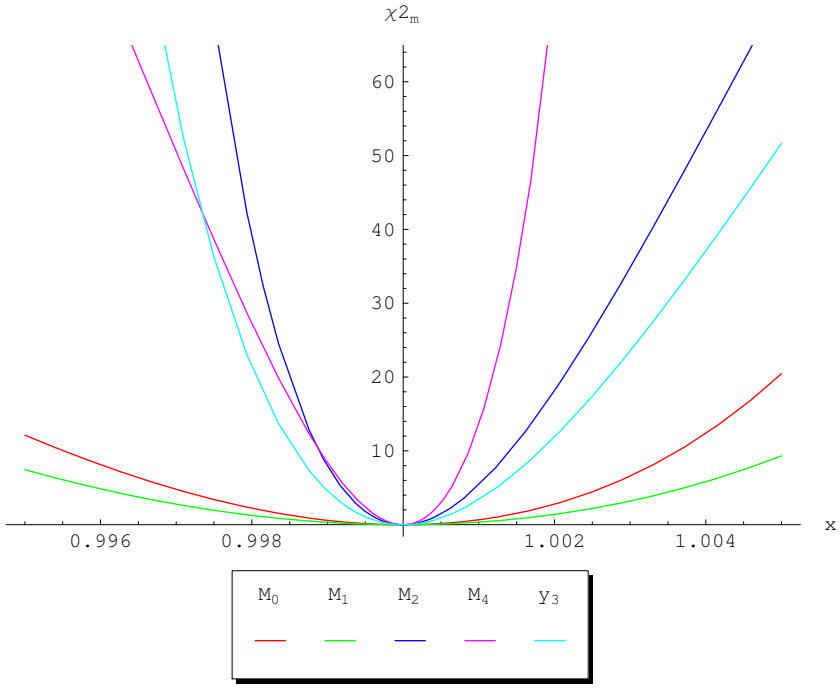
```
In[110]:=
mass1 =
  SetPrecision[Abs[-y2^2 * v0t^2 * (1 + I * r0) / (1 + r0^2) * (x + 3 y - z)] /. replxzyt, 20];
mass2 = SetPrecision[Abs[-y2^2 * v0t^2 * (1 + I * r0) / (1 + r0^2) * (x + 2 z)] /. replxzyt,
   20];
mass3 = SetPrecision[Abs[-y2^2 * v0t^2 * (1 + I * r0) / (1 + r0^2) * (x - 3 y - z)] /. replxzyt,
   20];
dm2atm = mass3^2 - mass1^2;
dm2sol = mass2^2 - mass1^2;
```

```
In[115]:=
chi2m[M0_, M1_, M2_, M4_, y3_] =
  (dm2atm - MVdm2atm)^2 / SDdm2atm^2 + (dm2sol - MVdm2sol)^2 / SDdm2sol^2;
```

```
In[116]:=
<< Graphics`Graphics`
<< Graphics`Legend`
```

In[118]:=

```
pl = Plot[{chi2m[x * VM0, VM1, VM2, VM4, Vy3], chi2m[VM0, x * VM1, VM2, VM4, Vy3],
    chi2m[VM0, VM1, x * VM2, VM4, Vy3], chi2m[VM0, VM1, VM2, x * VM4, Vy3],
    chi2m[VM0, VM1, VM2, VM4, x * Vy3]}, {x, 0.995, 1.005},
  PlotStyle → {{RGBColor[1, 0, 0]}, {RGBColor[0, 1, 0]}, {RGBColor[0, 0, 1]},
    {RGBColor[1, 0, 1]}, {RGBColor[0, 1, 1]}}, AxesLabel → {x, χ2m},
  PlotLegend → {"M0", "M1", "M2", "M4", "y3"}, LegendOrientation → Horizontal,
  LegendPosition → {-0.43, -0.9}, LegendShadow → {.01, -.01}, LegendSpacing → .6]
```



Out[118]=

- Graphics -

In[119]:=

```
Export["nmm3finetune.eps", pl, "EPS", ImageSize → {500, Automatic}]
```

Out[119]=

nmm3finetune.eps

Software

The following software was used for programming, data analysis and creating figures:

- **g95** - version 0.91 (March 2008)

  Fortran Compiler implementing the Fortran 95 standard,
  http://www.g95.org/

- **Mathematica** - version 5.2

  Computational software program,
  Wolfram Research, Inc.,
  http://www.wolfram.com/

- **gnuplot** - version 4.2 patchlevel 2

  Command-line driven graphing utility,
  http://www.gnuplot.info/

- **Xfig** - version 3.2 patchlevel 5

  Vector graphics editor,
  http://www.xfig.org/

- **feynMP**

  LaTeX tool to draw Feynman graphs,
  http://osksn2.hep.sci.osaka-u.ac.jp/~taku/osx/feynmp.html

List of Tables

List of Figures

Bibliography

[1] W. Grimus and L. Lavoura, *Tri-bimaximal lepton mixing from symmetry only*, JHEP **0904**, 013 (2009) [arXiv:0811.4766 [hep-ph]].

[2] J. Horejsi, *Fundamentals of electroweak theory*, *Prague, Czech Republic: Karolinum Pr. (2002) 351 p*

[3] A. Pich, *The Standard model of electroweak interactions*, arXiv:0705.4264 [hep-ph].

[4] S. L. Glashow, *Partial Symmetries Of Weak Interactions*, Nucl. Phys. **22**, 579 (1961).

[5] S. Weinberg, *A Model Of Leptons*, Phys. Rev. Lett. **19**, 1264 (1967).

[6] A. Salam, *Weak And Electromagnetic Interactions*, In the Proceedings of 8th Nobel Symposium, Lerum, Sweden, 19-25 May 1968, pp 367-377.

[7] P. W. Higgs, *Broken symmetries, massless particles and gauge fields*, Phys. Lett. **12**, 132 (1964).

[8] P. W. Higgs, *BROKEN SYMMETRIES AND THE MASSES OF GAUGE BOSONS*, Phys. Rev. Lett. **13**, 508 (1964).

[9] P. W. Higgs,None *Spontaneous Symmetry Breakdown without Massless Bosons*, Phys. Rev. **145**, 1156 (1966).

[10] P. W. Anderson, *Plasmons, Gauge Invariance, and Mass*, Phys. Rev. **130**, 439 (1963).

[11] F. Englert and R. Brout, *BROKEN SYMMETRY AND THE MASS OF GAUGE VECTOR MESONS*, Phys. Rev. Lett. **13**, 321 (1964).

[12] G. S. Guralnik, C. R. Hagen and T. W. B. Kibble, *GLOBAL CONSERVATION LAWS AND MASSLESS PARTICLES*, Phys. Rev. Lett. **13**, 585 (1964).

[13] N. Cabibbo, Phys. Rev. Lett. **10** (1963) 531.

[14] M. Kobayashi and T. Maskawa, *CP Violation In The Renormalizable Theory Of Weak Interaction*, Prog. Theor. Phys. **49**, 652 (1973).

[15] L. L. Chau and W. Y. Keung, *Comments On The Parametrization Of The Kobayashi-Maskawa Matrix*, Phys. Rev. Lett. **53**, 1802 (1984).

[16] S. L. Glashow, J. Iliopoulos and L. Maiani, Phys. Rev. D **2**, 1285 (1970).

[17] F. Boehm and P. Vogel, *Physics of massive neutrinos*, Cambridge, UK: Univ. Pr. (1992) 249 p

[18] S. M. Bilenky, C. Giunti and W. Grimus, *Phenomenology of neutrino oscillations*, Prog. Part. Nucl. Phys. **43** (1999) 1 [arXiv:hep-ph/9812360].

[19] W. Grimus, *Neutrino physics: Theory*, Lect. Notes Phys. **629**, 169 (2004) [arXiv:hep-ph/0307149].

[20] M. C. Gonzalez-Garcia and M. Maltoni, *Phenomenology with Massive Neutrinos*, Phys. Rept. **460**, 1 (2008) [arXiv:0704.1800 [hep-ph]].

[21] I. Schur, *EIN SATZ UEBER QUADRATISCHE FORMEN MIT KOMPLEXEN KOEFFIZIENTEN*, Am. J. Math **67** 472 (1945)

[22] B. Zumino, *Normal Forms of Complex Matrices*, J. Math. Phys. **3** 1055 (1962)

[23] W. Grimus and G. Ecker, *Basis transformations in generation space and a criterion for the existence of standard forms for unitarily congruent matrices*, J. Phys. A **21**, 2825 (1988).

[24] M. Gell-Mann, P. Ramond and R. Slansky, *Supergravity*, p.315, edited by F. van Nieuwenhuizen and D. Freedman, North Holland, Amsterdam, 1979.

[25] T. Yanagida, Proc. of the *Workshop on Unified Theory and the Baryon Number of the Universe*, KEK, Japan, 1979.

[26] R. N. Mohapatra and G. Senjanovic, *Neutrino mass and spontaneous parity nonconservation*, Phys. Rev. Lett. **44**, 912 (1980).

[27] B. Kayser, *On The Quantum Mechanics Of Neutrino Oscillation*, Phys. Rev. D **24**, 110 (1981).

[28] C. Giunti, C. W. Kim and U. W. Lee, *When do neutrinos really oscillate?: Quantum mechanics of neutrino oscillations*, Phys. Rev. D **44**, 3635 (1991).

[29] J. Rich, *The Quantum mechanics of neutrino oscillations*, Phys. Rev. D **48**, 4318 (1993).

[30] E. K. Akhmedov and A. Y. Smirnov, *Paradoxes of neutrino oscillations*, Phys. Atom. Nucl. **72** (2009) 1363 [arXiv:0905.1903 [hep-ph]].

[31] L. Wolfenstein, *Neutrino oscillations in matter*, Phys. Rev. D **17**, 2369 (1978).

[32] L. Wolfenstein, *Neutrino Oscillations And Stellar Collapse*, Phys. Rev. D **20**, 2634 (1979).

[33] S. P. Mikheyev and A. Yu. Smirnov, *Resonance Amplification of Oscillations in Matter and Spectroscopy of Solar Neutrinos*, Yad.Fiz. **42** (1985) 1441 [Sov. J. Nucl. Phys. **42**] (1985) 913.

[34] S. P. Mikheyev and A. Yu. Smirnov, *Resonant amplification of neutrino oscillations in matter and solar neutrino spectroscopy*, Nuovo Cim. C **9** (1986) 17.

[35] S. P. Mikheyev and A. Y. Smirnov, *Resonant neutrino oscillations in matter*, Prog. Part. Nucl. Phys. **23**, 41 (1989).

[36] S. M. Bilenky, C. Giunti, J. A. Grifols and E. Masso, *Absolute Values of Neutrino Masses: Status and Prospects*, Phys. Rept. **379** (2003) 69 [arXiv:hep-ph/0211462].

[37] R. N. Mohapatra, *Particle physics implications of neutrinoless double beta decay*, Nucl. Phys. Proc. Suppl. **77** (1999) 376 [arXiv:hep-ph/9808284].

[38] S. R. Elliott and P. Vogel, *Double beta decay*, Ann. Rev. Nucl. Part. Sci. **52**, 115 (2002) [arXiv:hep-ph/0202264].

[39] J. Schechter and J. W. F. Valle, *Neutrinoless double-beta decay in SU(2) x U(1) theories*, Phys. Rev. D **25**, 2951 (1982).

[40] E. Takasugi, *Can The Neutrinoless Double Beta Decay Take Place In The Case Of Dirac Neutrinos?*, Phys. Lett. B **149**, 372 (1984).

[41] T. Schwetz, M. A. Tortola and J. W. F. Valle, *Three-flavour neutrino oscillation update*, New J. Phys. **10**, 113011 (2008) [arXiv:0808.2016 [hep-ph]].

[42] M. Maltoni and T. Schwetz, *Three-flavour neutrino oscillation update and comments on possible hints for a non-zero theta_{13}*, PoS **IDM2008**, 072 (2008) [arXiv:0812.3161 [hep-ph]].

[43] P. L. Vahle [MINOS Collaboration], *Neutrino Oscillation Studies with MINOS*, arXiv:0810.3627 [hep-ex].

[44] M. V. Diwan, *Recent Results from the MINOS experiment*, arXiv:0904.3706 [hep-ex].

[45] J. Boger *et al.* [SNO Collaboration], *The Sudbury Neutrino Observatory*, Nucl. Instrum. Meth. A **449**, 172 (2000) [arXiv:nucl-ex/9910016].

[46] A. B. McDonald [The SNO Collaboration], *Sudbury Neutrino Observatory results*, Phys. Scripta **T121**, 29 (2005) [arXiv:hep-ex/0412060].

[47] S. Abe *et al.* [KamLAND Collaboration], *Precision Measurement of Neutrino Oscillation Parameters with KamLAND*, Phys. Rev. Lett. **100**, 221803 (2008) [arXiv:0801.4589 [hep-ex]].

[48] G. Alimonti *et al.* [Borexino Collaboration], *The Borexino detector at the Laboratori Nazionali del Gran Sasso*, Nucl. Instrum. Meth. A **600**, 568 (2009) [arXiv:0806.2400 [physics.ins-det]].

[49] M. Apollonio *et al.* [CHOOZ Collaboration], *Initial Results from the CHOOZ Long Baseline Reactor Neutrino Oscillation Experiment*, Phys. Lett. B **420**, 397 (1998) [arXiv:hep-ex/9711002].

[50] P. Anselmann *et al.* [GALLEX Collaboration], *Implications Of The Gallex Determination Of The Solar Neutrino Flux*, Phys. Lett. B **285**, 390 (1992).

[51] W. Hampel *et al.* [GALLEX Collaboration], *GALLEX solar neutrino observations: Results for GALLEX III.*, Phys. Lett. B **388**, 384 (1996).

[52] J. N. Abdurashitov *et al.* [SAGE Collaboration], *Measurement of the solar neutrino capture rate with gallium metal*, Phys. Rev. C **60**, 055801 (1999) [arXiv:astro-ph/9907113].

[53] M. Ishitsuka [Super-Kamiokande Collaboration], *Super Kamiokande results: Atmospheric and solar neutrinos*, arXiv:hep-ex/0406076.

[54] M. H. Ahn *et al.* [K2K Collaboration], *Measurement of Neutrino Oscillation by the K2K Experiment*, Phys. Rev. D **74**, 072003 (2006) [arXiv:hep-ex/0606032].

[55] Y. Kudenko [T2K Collaboration], *The near neutrino detector for the T2K experiment* Nucl. Instrum. Meth. A **598**, 289 (2009) [arXiv:0805.0411 [physics.ins-det]].

[56] D. Duchesneau [OPERA Collaboration], *The OPERA Long Baseline Experiment: Status and First Results*, arXiv:0810.2476 [hep-ex].

[57] A. A. Aguilar-Arevalo *et al.* [MiniBooNE Collaboration], *The MiniBooNE Detector*, Nucl. Instrum. Meth. A **599**, 28 (2009) [arXiv:0806.4201 [hep-ex]].

[58] J. Wolf [KATRIN Collaboration], *The KATRIN Neutrino Mass Experiment*, arXiv:0810.3281 [physics.ins-det].

[59] P. F. Harrison, D. H. Perkins and W. G. Scott, *Tri-bimaximal mixing and the neutrino oscillation data*, Phys. Lett. B **530**, 167 (2002) [arXiv:hep-ph/0202074].

[60] W. Grimus and L. Lavoura, *A model for trimaximal lepton mixing*, JHEP **0809**, 106 (2008) [arXiv:0809.0226 [hep-ph]].

[61] C. E. Aalseth *et al.* [IGEX Collaboration], *The Igex 76ge Neutrinoless Double-Beta Decay Experiment: Prospects for Next Generation Experiments*, Phys. Rev. D **65**, 092007 (2002) [arXiv:hep-ex/0202026].

[62] C. Arnaboldi *et al.*, *First results on neutrinoless double beta decay of Te-130 with the calorimetric CUORICINO experiment*, Phys. Lett. B **584**, 260 (2004).

[63] H. V. Klapdor-Kleingrothaus *et al.*, *Latest Results from the Heidelberg-Moscow Double Beta Decay Experiment*, Eur. Phys. J. A **12**, 147 (2001) [arXiv:hep-ph/0103062].

[64] H. V. Klapdor-Kleingrothaus, A. Dietz, H. L. Harney and I. V. Krivosheina, *Evidence for Neutrinoless Double Beta Decay*, Mod. Phys. Lett. A **16**, 2409 (2001) [arXiv:hep-ph/0201231].

[65] F. Feruglio, A. Strumia and F. Vissani, *Neutrino oscillations and signals in beta and 0nu 2beta experiments*, Nucl. Phys. B **637**, 345 (2002) [Addendum-ibid. B **659**, 359 (2003)] [arXiv:hep-ph/0201291].

[66] C. E. Aalseth *et al.*, *Comment on 'Evidence for Neutrinoless Double Beta Decay'*, Mod. Phys. Lett. A **17**, 1475 (2002) [arXiv:hep-ex/0202018].

[67] H. L. Harney, *Reply to the comment on 'Evidence for neutrinoless double beta decay'. (Mod. Phys. Lett. A16(2001) 2409)*, arXiv:hep-ph/0205293.

[68] Yu. G. Zdesenko, F. A. Danevich and V. I. Tretyak, *Has neutrinoless double beta decay of Ge-76 been really observed?*, Phys. Lett. B **546**, 206 (2002).

[69] H. V. Klapdor-Kleingrothaus, A. Dietz and I. V. Krivosheina, *Status of Evidence for Neutrinoless Double Beta Decay*, Found. Phys. **32**, 1181 (2002) [Erratum-ibid. **33**, 679 (2003)] [arXiv:hep-ph/0302248].

[70] H. V. Klapdor-Kleingrothaus, I. V. Krivosheina, A. Dietz and O. Chkvorets, *Search for neutrinoless double beta decay with enriched 76Ge in Gran Sasso 1990-2003*, Phys. Lett. B **586**, 198 (2004) [arXiv:hep-ph/0404088].

[71] H. V. Klapdor-Kleingrothaus and I. V. Krivosheina, *The Evidence For The Observation Of 0nu Beta Beta Decay: The Identification Of 0nu Beta Beta Events From The Full Spectra*, Mod. Phys. Lett. A **21**, 1547 (2006).

[72] K. T. Knopfle [GERDA Collaboration], *Search for neutrinoless double-beta decay of Ge-76 with GERDA*, arXiv:0809.5207 [hep-ex].

[73] A. A. Smolnikov [GERDA Collaboration], *Status of the GERDA experiment aimed to search for neutrinoless double beta decay of 76Ge*, arXiv:0812.4194 [nucl-ex].

[74] R. Ardito *et al.*, *CUORE: A cryogenic underground observatory for rare events*, arXiv:hep-ex/0501010.

[75] H. Kühbock, *Neutrino masses in SO(10) models*, Dissertation, University of Vienna (2007).

[76] L. Lavoura, H. Kuhbock and W. Grimus, *Charged-fermion masses in SO(10): Analysis with scalars in 10+120*, Nucl. Phys. B **754**, 1 (2006) [arXiv:hep-ph/0603259].

[77] W. Grimus and H. Kuhbock, *Fermion masses and mixings in a renormalizable SO(10) x Z_2 GUT*, Phys. Lett. B **643**, 182 (2006) [arXiv:hep-ph/0607197].

[78] W. Grimus and H. Kuhbock, *A renormalizable SO(10) GUT scenario with spontaneous CP violation*, Eur. Phys. J. C **51**, 721 (2007) [arXiv:hep-ph/0612132].

[79] W. Grimus and H. Kuhbock, *Embedding the Zee-Wolfenstein neutrino mass matrix in an SO(10) x A4 GUT scenario*, Phys. Rev. D **77**, 055008 (2008) [arXiv:0710.1585 [hep-ph]].

[80] W. Grimus and H. Kuhbock, *Renormalisable SO(10) models and neutrino masses and mixing*, Acta Phys. Polon. B **38**, 3373 (2007) [arXiv:0710.2857 [hep-ph]].

[81] L. Lavoura and H. Kuhbock, *A_4 model for the quark mass matrices*, Eur. Phys. J. C **55**, 303 (2008) [arXiv:0711.0670 [hep-ph]].

[82] J. A. Nelder and R. Mead, *A Simplex Method for Function Minimization*, Comput. J. **7** 308 (1965).

[83] J. C. Lagarias, J. A. Reeds, M. H. Wright and P. E. Wright, *Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions*, SIAM J. Optim. **9** 112 (1998).

[84] K. I. M. McKinnon, *Convergence of the Nelder-Mead Simplex Method to a Nonstationary Point*, SIAM J. Optim. **9** 148 (1998).

[85] W. H. Press, S. A. Teutolsky, W. T. Vetterling and B. P. Flannery, *Numerical Recipes in C++*, Cambridge University Press, 2002.

[86] R. M. Lewis, V. Torczon and M. W. Trosset, *Direct Search Methods: Then and Now*, J. Comput. Appl. Math. **124** 191 (2000).

[87] L. Lavoura and W. Grimus, *Seesaw model with softly broken L(e) - L(mu) - L(tau)*, JHEP **0009**, 007 (2000) [arXiv:hep-ph/0008020].

[88] W. Grimus and L. Lavoura, *Softly broken lepton numbers and maximal neutrino mixing*, JHEP **0107**, 045 (2001) [arXiv:hep-ph/0105212].

[89] W. Grimus and L. Lavoura, *Softly broken lepton numbers: An approach to maximal neutrino mixing*, Acta Phys. Polon. B **32**, 3719 (2001) [arXiv:hep-ph/0110041].

[90] W. Grimus, L. Lavoura and A. Singraber, *Trimaximal lepton mixing with a trivial Dirac phase*, Phys. Lett. B **686**, 141 (2010) [arXiv:0911.5120 [hep-ph]].

[91] W. Grimus and L. Lavoura, *A non-standard CP transformation leading to maximal atmospheric neutrino mixing*, Phys. Lett. B **579**, 113 (2004) [arXiv:hep-ph/0305309].

[92] W. Grimus and L. Lavoura, *A three-parameter neutrino mass matrix with maximal CP violation*, Phys. Lett. B **671**, 456 (2009) [arXiv:0810.4516 [hep-ph]].

[93] S. Pascoli and S. T. Petcov, *The SNO solar neutrino data, neutrinoless double-beta decay and neutrino mass spectrum*, Phys. Lett. B **544**, 239 (2002) [arXiv:hep-ph/0205022].

Zusammenfassung

Der direkte Vergleich zwischen Theorie und Experiment ist ein Grundpfeiler der wissenschaftlichen Methode. Für jeden Physiker ist es ein spannender Augenblick wenn theoretische Vorhersagen auf Resultate von experimentellen Beobachtungen treffen, um entweder Übereinstimmung oder widersprüchliche Ergebnisse zu Tage zu fördern. Heutzutage eröffnet die Teilchenphysik ein weites Betätigungsfeld mit bahnbrechenden Entwicklungen auf theoretischer und experimenteller Ebene. Auch das relativ junge Teilgebiet der computergestützten Physik leistet einen wichtigen Beitrag, der Einsatz von leistungsfähigen Rechnern schreitet immer weiter voran. In den letzten Jahren wurde speziell der Lepton-Sektor zu einer Spielwiese für die Entwicklung neuer theoretischer Modelle, angespornt durch die Entdeckung der Neutrinooszillationen, Beiträge zur Frage der Dunklen Materie und Hinweise auf Neue Physik. Basierend auf dem Standardmodell der Teilchenphysik wurden bereits eine Vielzahl von Techniken und Erweiterungen untersucht, um Eigenheiten der Neutrinos zu erklären, darunter etwa diskrete Symmetriegruppen oder der seesaw-Mechanismus, bis hin zu Modellen zur Großen Vereinheitlichung (GUT). Das steigende Interesse an der Neutrinophysik führt auch zu einer wachsenden Zahl an Experimenten über solare, atmosphärische und Reaktor-Neutrinos, wodurch die Genauigkeit der Messwerte weiter zunimmt und theoretische Modelle mit diesen konfrontiert werden können.

Im Sinne der einleitenden Worte behandelt die vorliegende Arbeit Methoden, um Theorie und Experiment zu vergleichen, auch wenn ein komplizierter Zusammenhang zwischen Modellparametern und daraus zu berechnenden Observablen besteht. Die Analyse beruht dabei auf numerischen Verfahren, die anhand eines Modells zur tri-bimaximalen Mischung und dessen Modifikationen getestet werden. Zu Beginn werden die wichtigsten Eigenschaften des Standardmodells wiederholt und grundlegende Erweiterungen, wie Majorana-Neutrinos oder der seesaw-Mechanismus diskutiert. Darauf aufbauend werden verschiedene Phänomene der Neutrinophysik beschrieben, beispielsweise Neutrinooszillationen und der neutrinolose doppelte Beta-Zerfall. Die neueren experimentellen Ergebnisse dazu werden ebenfalls bereitgestellt. Danach folgt ein thematischer Wechsel zu den verwendeten numerischen Methoden. Es wird die χ^2-Funktion als Beurteilungskriterium eingeführt, das die Übereinstimmung zwischen Modellvorhersagen und Messwerten quantifiziert. Es entspricht dann das Minimum dieser Funktion der bestmöglichen Anpassung zwischen Theorie und Experiment. Abhängig der Komplexität des untersuchten Modells ist eine analytische Minimierung mitunter nicht möglich, sodass

man auf den Einsatz von numerischen Methoden angewiesen ist. Daher wird das sogenannte Nelder-Mead Verfahren und die *Fortran*-Implementation ausführlich beschrieben, die bei der Analyse der getesteten Modelle zur Anwendung kommen. Als Ausgangspunkt dient dabei ein Modell für tri-bimaximale Mischung, das um CP-Erhaltung und zusätzliche spontane Symmetriebrechung erweitert wird. Schließlich werden die Resultate der numerischen Verfahren präsentiert.

Ein wesentlicher Bestandteil der vorliegenden Arbeit war die Erstellung und Anwendung des Nelder-Mead Programms. Daher wird der Quellcode im Anhang bereitgestellt.

Curriculum Vitae

Persönliche Daten:

Name: Andreas Singraber

Geburtsdatum: 29. Juli 1986

Geburtsort: Wien

Staatsbürgerschaft: Österreich

Ausbildung:

1992 - 1996 Besuch der Volksschule Lacknergasse in Wien 18.

1996 - 2000 Besuch des BRG Reinprechtsdorfer Straße in Wien 5.

2000 - 2004 Besuch des BRG Neusiedl am See.

Schuljahr 2003/2004 Verfassen der Fachbereichsarbeit *"Theorien über das Licht: Von Sehstrahlen bis zur Quantentheorie"* bei Mag. Andreas Wenth.

Juni 2004 Reifeprüfung mit ausgezeichnetem Erfolg bestanden.

Oktober 2004 Beginn des Studiums der Physik an der Universität Wien.

7. Oktober 2005 1. Diplomprüfung mit Auszeichnung bestanden.

7. Februar 2008 2. Diplomprüfung mit Auszeichnung bestanden.

April 2009 - Juni 2010 Verfassen der Diplomarbeit bei Ao. Univ.-Prof. Dr. Walter Grimus.

Publikationen:

- W. Grimus, L. Lavoura and A. Singraber, *Trimaximal lepton mixing with a trivial Dirac phase*, Phys. Lett. B **686** (2010) 141 [arXiv:0911.5120 [hep-ph]].