Rolling down the leptonic BSM landscape using machine learning techniques

Alfredo Aranda,[1,2,3,*] Raymundo Ramos,[4,†] and Alexander J. Stuart[1,2,‡]

[1]*Facultad de Ciencias, Universidad de Colima, C.P. 28045, Colima, México*
[2]*Dual CP Institute of High Energy Physics, C.P. 28045, Colima, México*
[3]*School of Physics, Korea Institute for Advanced Study, Seoul 02455, Korea*
[4]*Quantum Universe Center, Korea Institute for Advanced Study, Seoul 02455, Korea*

(Received 10 June 2026; accepted 5 October 2026; published 28 October 2026)

In this work, we adapt and apply techniques from machine learning to the exploration of physics beyond the Standard Model in the leptonic sector. Namely, we employ initialization and optimization, as they are applied in machine learning, to minimize a loss function that describes textures or conditions which we want in the neutrino mass matrix. The model free parameters are explored during the optimization, and after training for a number of optimization steps, we obtain matrices that approximately follow the desired forms, as well as their corresponding optimized parameters. We also discuss extensions and additional applications of the ideas presented here in conjunction with other methods based on artificial intelligence.

DOI: 10.1103/g5l2-bzs4

I. INTRODUCTION

The Standard Model (SM), inspired by naturalness, simplicity, and experimental results, has endured precise observations as performed up to these days [1]. While this indicates that all the elements of the SM are needed to explain these observations, other phenomena point toward components of nature that remain unexplained. One notable unexplained component is the Yukawa sector, where masses of families of quarks and leptons depend on many free parameters that, rather than being predicted, are set to follow experimental constraints. Moreover, in the leptonic sector, neutrino masses are absent in the SM while observations of neutrino oscillations indicate that they are massive and that their mass eigenstates do not correspond to the same flavor eigenstates of the charged leptons [2,3]. Having massive neutrinos that oscillate complicates any formulation beyond the SM (BSM) that simply adds fields to generate a few additional mass terms in the SM Lagrangian. Instead, we have to deal with diagonalizations of complex matrices, new particles with their own masses and associated energy scales, breaking of new symmetries, and many more characteristics. It is clear that attempting to include masses for neutrinos in (BSM) theories will unavoidably add new parameters, as they must follow constraints from the aforementioned observations.

Interestingly, perceptive observations of experimental results may give insights into the characteristics of an underlying behavior, e.g., a nearly diagonal (quark) mixing matrix may indicate high-energy symmetries that kept particles (quarks) from mixing until it was broken at a lower energy scale. The leptonic sector is not an exception, with "bare" mixing normally identified with a form known as tribimaximal mixing [4–8]. The number of BSM proposals that attempt to repeat this form of the mixing matrix via additional particles and/or symmetries keeps growing even after the reactor mixing angle, θ_{13}, has been measured as nonvanishing [9–11]. These proposals have been made by a mixture of human experience in combination with trial and error. In this work, we want to explore the possibility of using new computational techniques from the artificial intelligence (AI) era to automatize the discovery of hints of properties of an underlying model from known experimental results.[1] In particular, our interest lies in the direction of opening doors to comprehensive and robust analysis of a number of possible configurations that may be discovered, including statistical analysis on their frequency and determination of their predictive power. One could imagine the possibility of starting with a far too general model and using some form of optimization to infer relationships among parameters, potentially pointing toward simpler formulations. Our approach

*Contact author: fefo@ucol.mx
†Contact author: raramos@kias.re.kr
‡Contact author: astuart@ucol.mx

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI. Funded by SCOAP³.

[1]In this work, preferences are determined by simple statistical frequency after running several tests under the assumptions that will be detailed in each section.

will follow the skeleton devised in Refs. [12,13], although we will develop our own loss functions and training system.[2]

To be more precise, we want to focus on a particular and important part of AI, known as machine learning (ML). The process of ML itself is a mathematical one that, put in very simple terms, aims to optimize the parameters of a mathematical function typically identified as a model, via the minimization of a loss function. In this work, our purpose is to use techniques that empower ML with objects found in theoretical setups of the leptonic sector. Toward this purpose, we will formulate elements from the theoretical leptonic model as a function delivering predictions for several observables. The variables of the model will be employed as those optimized by typical algorithms used in ML, and, as is done in ML, we will formulate loss functions where the minimum is achieved when the output of the model is consistent with our desired result. However, an important difference in our case is that, instead of obtaining a final reusable model that encodes inputs to outputs, we obtain a number of neutrino mass matrices with (approximately) the conditions contained in the loss function. In addition, we obtain optimized values for variables of the model that were used during the learning process.

This paper is structured as follows. In Sec. II, we give a brief description of the leptonic sector of the SM and simple extensions of it involving massive neutrinos, as well as the details needed for the numerical setup. In Sec. III, we present all the details of the process used to optimize the variables involved and obtain particular forms of the neutrino mass matrix. This section is where analogies to ML methods are developed. In Sec. IV, we discuss interesting points of the results obtained during the numerical process, make some connections to theoretical details, and comment on future improvements. Finally, in Sec. V, we present the conclusions of this work.

II. NEUTRINOS IN THE STANDARD MODEL AND BEYOND

The neutrinos are part of the leptonic sector of the SM. They are quite unique as they do not have charge and are the only fermionic fields that do not have a right-handed field. Due to this, and the accidental symmetries of the SM, it is impossible to generate neutrino masses at tree or loop level using only what is contained in the SM [1]. To see this explicitly, observe that in the SM, neutrinos are part of $SU(2)$ lepton doublets, i.e.,

$$L_{\ell L} = \begin{pmatrix} \nu_{\ell L} \\ \ell_L \end{pmatrix}, \qquad \ell = e, \mu, \tau, \tag{1}$$

with the left-handed neutrinos in the upper component and the left-handed charged leptons in the lower component. In the SM, masses for fermions are generated from Yukawa couplings of the $L_{\ell L}$ doublets, with the Higgs scalar doublet, ϕ, and the right-handed fermion singlets, i.e.,

$$-\mathcal{L}_Y = \sum_{j,k} Y^{\ell}_{jk} \bar{L}_{jL} \phi e_{kR} + \text{H.c.}, \tag{2}$$

where indices, j and k, run over the three lepton families of Eq. (1). When the doublet, $\phi = (\phi^+, \phi^0)^T$, acquires the vacuum expectation value, $\langle \phi \rangle = (0, v/\sqrt{2})$, charged leptons acquire the masses

$$m^{\ell}_{jk} = Y^{\ell}_{jk} \frac{v}{\sqrt{2}}. \tag{3}$$

Yet, in the SM, there is no right-handed neutrino field analogous to $e_{\ell R}$, and, therefore, this mechanism cannot be used to generate masses for neutrinos. Additionally, the global accidental symmetries of total lepton number and $B - L$, forbid the generation of mass via the product of two $L_{\ell L}$ [1].

A. Dirac neutrinos

In this work, we explore a minimalist extension of the SM by respecting its accidental symmetries and introducing three right-handed neutrinos that generate a 3×3 Dirac neutrino mass matrix, M^{ν}_D. This mass matrix is diagonalized by two unitary matrices, V^R and W^{ν}, i.e.,

$$V^{R\dagger} M^{\nu}_D W^{\nu} = \text{diag}(m_1, m_2, m_3), \tag{4}$$

where m_1, m_2, and m_3 are the neutrino mass eigenstates. The Pontecorvo-Maki-Nakagawa-Sakata mixing matrix (U_{PMNS}) of the leptonic sector is defined as [1]

$$U_{\text{PMNS}} = \mathcal{P}^{\dagger}_{\ell} U^{\ell\dagger} W^{\nu} = \mathcal{P}^{\dagger}_{\ell} U^{\ell\dagger} U^{\nu} \mathcal{P}_{\nu}, \tag{5}$$

where we have added $U^{\ell}\mathcal{P}_{\ell}$, the analogous of W^{ν} for charged leptons, and also expanded $W^{\nu} = U^{\nu}\mathcal{P}_{\nu}$. The $\mathcal{P}_{\ell,\nu}$ matrices are diagonal phase matrices. In the case where charged leptons are assumed to be already in their mass eigenstates $U^{\ell\dagger}$ becomes the identity. Additionally, $\mathcal{P}_{\ell}$ can be removed through unphysical charged-lepton phase redefinition. In the case of $\mathcal{P}_{\nu}$, it is nonremovable only for Majorana neutrinos. Therefore, by assuming that charged leptons are already in their diagonal basis and redefined to absorb $\mathcal{P}_{\ell}$ and that neutrinos receive only Dirac masses ($W^{\nu} \to U^{\nu}$), U_{PMNS} becomes

$$U_{\text{PMNS}} = \mathcal{P}^{\dagger}_{\ell} U^{\ell\dagger} U^{\nu} \mathcal{P}_{\nu} \to U^{\nu}. \tag{6}$$

The elements of this matrix can be expressed using the Particle Data Group (PDG) parametrization given by

[2]For other applications of machine learning methods to leptons, quarks, and flavor in BSM physics see Refs. [14–18].

$$U_{\text{PMNS}} = \begin{pmatrix} c_{12}c_{13} & s_{12}c_{13} & s_{13}e^{-i\delta^{CP}} \\ -s_{12}c_{23} - c_{12}s_{13}s_{23}e^{i\delta^{CP}} & c_{12}c_{23} - s_{12}s_{13}s_{23}e^{i\delta^{CP}} & c_{13}s_{23} \\ s_{12}s_{23} - c_{12}s_{13}c_{23}e^{i\delta^{CP}} & -c_{12}s_{23} - s_{12}s_{13}c_{23}e^{i\delta^{CP}} & c_{13}c_{23} \end{pmatrix}, \tag{7}$$

where $s_{jk} = \sin\theta_{jk}$ and $c_{jk} = \cos\theta_{jk}$. Using Eqs. (4) and (6), we obtain

$$M_D^\nu = V^R \text{diag}(m_1, m_2, m_3) W^{\nu\dagger} = V^R \text{diag}(m_1, m_2, m_3) U_{\text{PMNS}}^\dagger. \tag{8}$$

This allows us to obtain the elements of M_D^ν by setting the values of U_{PMNS} to the measured oscillation parameters, and by writing the elements of V^R using the parametrization[3]

$$V^R = \begin{pmatrix} c_{12}^R c_{13}^R & s_{12}^R c_{13}^R & s_{13}^R e^{-i\delta^R} \\ -s_{12}^R c_{23}^R - c_{12}^R s_{13}^R s_{23}^R e^{i\delta^R} & c_{12}^R c_{23}^R - s_{12}^R s_{13}^R s_{23}^R e^{i\delta^R} & c_{13}^R s_{23}^R \\ s_{12}^R s_{23}^R - c_{12}^R s_{13}^R c_{23}^R e^{i\delta^R} & -c_{12}^R s_{23}^R - s_{12}^R s_{13}^R c_{23}^R e^{i\delta^R} & c_{13}^R c_{23}^R \end{pmatrix} \times \mathcal{P}^R, \tag{9}$$

where $s_{jk}^R = \sin\theta_{jk}^R$ and $c_{jk}^R = \cos\theta_{jk}^R$ and $\mathcal{P}^R = \text{diag}(1, \exp(i\phi_2^R), \exp(i\phi_3^R))$. Additionally, we need to fix the neutrino mass eigenvalues which we can do by choosing the mass of the lightest eigenstate and using squared mass differences to fix the other two masses. For example, if we choose m_1 to be the lightest mass (normal hierarchy), we can fix the other masses using

$$m_2 = \sqrt{\Delta m_{21}^2 + m_1^2}, \tag{10}$$

$$m_3 = \sqrt{\Delta m_{31}^2 + m_1^2}. \tag{11}$$

When using these equations, we will always consider m_1 as real, resulting in m_2 and m_3 being real as well. We will leave the contribution from complex masses as part of the phases in V^R.

At some point, we will need to obtain oscillation parameters starting from numerical M_D^ν. In that case, we will proceed by first obtaining V^R and U^ν from squaring Eq. (4) using

$$V^{R\dagger} M_D^\nu M_D^{\nu\dagger} V^R = \text{diag}(|m_1|^2, |m_2|^2, |m_3|^2), \tag{12}$$

$$U^{\nu\dagger} M_D^{\nu\dagger} M_D^\nu U^\nu = \text{diag}(|m_1|^2, |m_2|^2, |m_3|^2), \tag{13}$$

and obtaining numerical V^R and U^ν from the eigenvectors of $M_D^\nu M_D^{\nu\dagger}$ and $M_D^{\nu\dagger} M_D^\nu$, respectively. The values of $|m_1|^2$, $|m_2|^2$, and $|m_3|^2$ are obtained from the eigenvalues of either $M_D^\nu M_D^{\nu\dagger}$ or $M_D^{\nu\dagger} M_D^\nu$. We make sure that the obtained V^R and U^ν result in real masses by redefining V^R such that $V^{R\dagger} M_D^\nu U^\nu$ is real. From the numerical U^ν [that was equalized to U^{PMNS} in Eq. (6)] we can obtain mixing angles using the parametrization of Eq. (7) and the following expressions

$$\theta_{23} = \tan^{-1}\left(\frac{|U_{\mu 3}|}{|U_{\tau 3}|}\right), \qquad \theta_{12} = \tan^{-1}\left(\frac{|U_{e2}|}{|U_{e1}|}\right),$$
$$\theta_{13} = \sin^{-1}(|U_{e3}|). \tag{14}$$

The amount of CP violation can be quantified using the Jarlskog invariant [20,21] given by

$$J = \text{Im}\,(U_{e1}^\nu U_{\mu 2}^\nu U_{e2}^{\nu\dagger} U_{\mu 1}^{\nu\dagger}), \tag{15}$$

which can be rewritten as

$$J = s_{23}c_{23}s_{12}c_{12}s_{13}c_{13}^2 \sin\delta^{CP}, \tag{16}$$

using the PDG parametrization of Eq. (7). From Eqs. (15) and (16), we can obtain the δ^{CP} phase by using the numerical values of U^ν and the angles obtained from using Eq. (14) as

$$\delta^{CP} = \sin^{-1}\left[\frac{\text{Im}\,(U_{e1}^\nu U_{\mu 2}^\nu U_{e2}^{\nu\dagger} U_{\mu 1}^{\nu\dagger})}{s_{23}c_{23}s_{12}c_{12}s_{13}c_{13}^2}\right]. \tag{17}$$

Note that the range of the arcsine function is $[-\pi/2, \pi/2]$ or $[-90°, 90°]$. Thus, sometimes we will need an extra step to obtain the correct quadrant. This step will be explained when it becomes necessary later in this work. Finally, the value that we need to check against KATRIN's measurement is obtained from the eigenvalues of M_D^ν and the first row of U^ν using [22]

[3]Note that the parametrization of $\mathcal{P}^R$ is compatible with the diagonal phase matrix used by the Particle Data Group review up to 2018 [19] and can be easily related to the parametrization used in more recent versions of the same review.

TABLE I. Numbers used for numerical inputs and constraints. The numbers in the second column correspond to the numbers reported in the most recent global fit made by the NuFIT Collaboration (version 6.1) [24]. The 2σ range has been interpolated from the 1-dimensional projections provided in the same global fit. The value for the upper bound of m_ν was obtained by the KATRIN experiment [22]. The term bfp represents best fit point.

	bfp $\pm 1\sigma$	2σ range
s^2_{12}	$0.3088^{+0.0067}_{-0.0066}$	[0.2957, 0.3224]
θ_{12} [°]	$33.76^{+0.42}_{-0.41}$	[32.94, 34.60]
s^2_{23}	$0.470^{+0.017}_{-0.014}$	[0.444, 0.575]
θ_{23} [°]	$43.27^{+1.0}_{-0.82}$	[41.76, 49.29]
s^2_{13}	0.02249 ± 0.00057	[0.02133, 0.02363]
θ_{13} [°]	8.62 ± 0.11	[8.40, 8.84]
δ^{CP} [°]	207^{+23}_{-20}	[145, 349]
Δm^2_{21} [10^{-5} eV^2]	$7.537^{+0.094}_{-0.10}$	[7.337, 7.728]
Δm^2_{31} [10^{-3} eV^2]	$2.521^{+0.026}_{-0.018}$	[2.479, 2.569]
$m_\nu < 0.45$ eV		

$$m_\nu = \sqrt{\sum_{j=1}^{3} |U^\nu_{ej}|^2 m_j^2}. \tag{18}$$

The numerical values determined in a recent global fit by the NuFIT Collaboration [23] are summarized in Table I assuming normal ordering. The 2σ ranges have been determined by interpolating the 1-dimensional projections provided in the same global fit.

III. LEARNING PARAMETERS OF THE LEPTONIC SECTOR

In this section, we implement ML techniques to find and enforce conditions between elements in the neutrino mass matrix. We do this by starting from randomizing theoretical and experimental inputs. The main idea is to begin by setting some values using experimentally measured neutrino oscillation parameters and continue by randomizing model parameters. Those randomized model parameters are the variables that we will attempt to learn by crafting loss functions which will be minimized. The loss functions will contain conditions that we wish to impose on the neutrino mass matrix, although we will try to stay on simple conditions that enforce some form of coincidences between elements. We will also repeat the process several times and pay special attention to the most common cases or cases where we find interesting and unexpected coincidences.

The first step is the initialization of the involved variables and parameters. This step takes care of setting the initial values of the parameters before any optimization process is used. In the case of the measured oscillation parameters, we initialize them from Gaussian distributions using their central value and the 1σ error. In the cases where the $\pm 1\sigma$ errors are not symmetric, we use the smallest error. In the case of θ_{13} and Δm^2_{31}, considering their rather small error bars, we have fixed them to their best fit central values. The numbers we use are based on the NuFIT 6.1 global fit [23,24]. Additionally, the mass of the lightest neutrino mass eigenstate can be constrained from above using the earth based experiment KATRIN, which sets an upper bound $m_\nu < 0.45$ eV, where $m^2_\nu = \sum_j |U_{ej}|^2 m^2_j$ [22]. This result is independent of mass ordering and issues with the cosmological model. The rest of the variables are the free parameters of V^R, given by the angles θ^R_{jk}, with $jk = 12$, 13, 23, and the phases δ^R, ϕ^R_2 and ϕ^R_3. For these variables we use a random uniform distribution, with the angles θ^R_{jk} being in the $[0, \pi/2]$ range, and the phases in the $[-\pi, \pi]$ range. Considering that the measured oscillation parameters are not completely free to vary, we will apply optimization using only the parameters of V^R and, sometimes, the lightest neutrino mass, m_1. With all these parameters given numerical values, we can obtain a numerical value for M^ν_D using Eq. (8).

The second step after initialization is defining a loss function that will be minimized during the optimization process. This loss function should reflect the conditions that we want to enforce during the optimization process, with its minimum defining the result that we want to obtain. For example, if we want to obtain textures with equal entries, the loss function should be defined such that its minimum occurs when the selected entries are equal. As another example, assume we want to enforce some elements to have a particular value. Then, the minimum of the loss function should be at the point where those elements have acquired exactly that particular value. In all cases, the loss function should increase in value as we move away from the desired conditions. In this work, we will define the loss as a function of the matrix elements $(M^\nu_D)_{jk}$ and some case-dependent additional parameters.

When a loss function has been defined, we can proceed to optimize the variables of our problem. In our case, these variables are the angles θ^R_{jk}, and the phases δ^R, ϕ^R_2, and ϕ^R_3. Additionally, we may include m_1 when relevant. There are several optimization algorithms, but most of them work by applying small variations to the variables that we want to optimize and calculate the change in the loss function from these small variations. After this, the values of the variables are updated in the direction where the small variations were found to have the steepest drop in the loss function. This is repeated for a number of steps until the loss function can be considered minimized. Note that there may be complications when several local minima exist as the optimization process can get stuck in any one of them. Additionally, if the loss function is too flat, the optimization may require many steps to converge to the global minimum. These problems may be somewhat alleviated by adjusting the parameters of the optimization process or by

making adjustments to the loss function. We will comment about these problems in the cases where they are found in our specific examples.

After the optimization process has worked for a number of steps and the loss function does not seem to yield a better minimum, we proceed to analyze the obtained mass matrix to determine if our goal has been reached and to analyze the obtained parameters. Since we are working directly with the parameters of V^R and not with the parameters of a very large model, we can easily present the optimized variables and provide some insights on the relationship between some neutrino textures and V^R. Note that there is always a dependence of the final distribution of the results on choices such as parametrization, weak basis, priors and optimization algorithm, with the latter adding an element of stochasticity. In this work, we will focus on the proposed techniques and leave further discussion of these issues for future works.

A. Loss function

The most relevant point connecting the theory described in Sec. II and the learning process described in Fig. 1 is the loss function. Typically, in ML, the loss function is determined by the objective of the model being trained. For example, when training a model for binary classification, one generally chooses the binary cross entropy loss, while for models that have arbitrary output, one may choose other functions such as the mean of squared or absolute errors. In our case, the objective is to find the parameters for the combination $V^R \mathrm{diag}(m_1, m_2, m_3) U^\dagger_{\mathrm{PMNS}}$ that result in a desirable form of M^ν_D. From this idea, it is clear that the loss function must contain some of the elements of M^ν_D in some form. Additionally, experimental constraints can be included in the loss function to direct the optimization process toward viable values of the parameter space. In our case, we are already enforcing values for the oscillation parameters of U_{PMNS} as part of the initialization. However, when including the lightest neutrino mass, m_1, as an optimizable parameter, we will need to make sure that the values of the neutrino masses remain below the upper limit set by experimental measurements. As previously mentioned, we consider the constraint of the effective electron-antineutrino mass obtained by KATRIN, i.e., $m_\nu < 0.45$ eV [22], where $m^2_\nu = \sum_i |U_{ei}|^2 m^2_i$. The corresponding term in the loss function is given by

$$L_{\mathrm{KATRIN}} = w_K \max(m_\nu - 0.45\ \mathrm{eV}, 0)^2, \tag{19}$$

where w_K is an adjustable weight that will be set to a large number. If one is worried about this term being dimensionful, the weight w_K could be thought as having units of inverse squared mass. Note that, in our case, L_{KATRIN} depends on measured oscillation parameters and m_1 through obtaining neutrino physical masses using Eqs. (10) and (11). Additionally, we will consider different forms of the loss function that directly include elements of M^ν_D. The first loss function we consider enforces equality of two elements and is given by

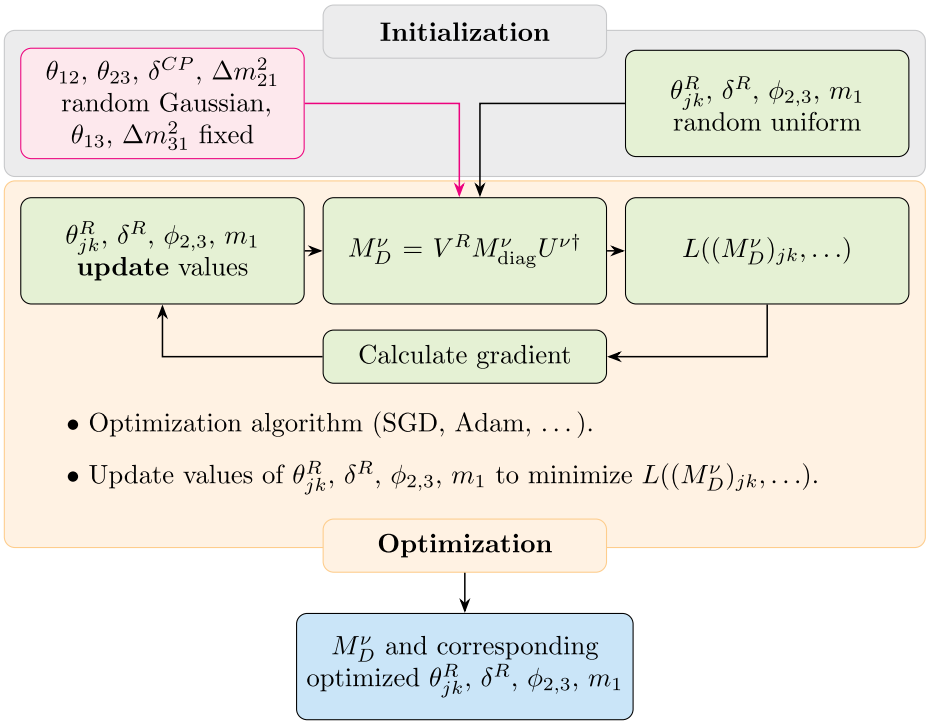


FIG. 1. Flowchart of the optimization process used to obtain desirable forms of M^ν_D and the corresponding parameters.

$$L_{1\text{diff}} = \min\left(\frac{|(M_D^\nu)_{jk} - (M_D^\nu)_{lm}|}{|(M_D^\nu)_{jk}| + |(M_D^\nu)_{lm}|}\right), \tag{20}$$

where the minimum of all possible differences is taken after initialization but before optimization. This means that the values of the j, k, l and m indices depends on the initialization of the parameters. Note that this loss function equalizes the two mass matrix elements down to their real and imaginary components.

Another option we consider is the presence of vanishing elements in the neutrino mass matrix. Thus, we consider a simple loss function that requires us to decide on a series of vanishing elements before optimization, i.e.,

$$L_{n\text{-zero}} = \frac{(9-n)\sum_{jk\in\text{vanish}}|(M_D^\nu)_{jk}|}{n\sum_{lm\in\text{nonzero}}|(M_D^\nu)_{lm}|}, \tag{21}$$

where n is the number of vanishing elements, jk runs over the elements that were chosen to vanish, while lm runs over the elements that are expected to be nonzero. Note that this loss function may not always result in minimization of all the n elements included in the numerator.

We also explore the possibility of letting the optimization process itself decide which elements vanish. Our approach in this case is different to the two options before as we will not be choosing particular elements before optimization. Rather, we design loss functions that are capable of counting the number of elements with absolute value above and below some threshold and use optimization to increase the number of elements below this threshold while also reducing their absolute value. We consider the elements above the threshold as nonvanishing while the elements below threshold are considered as possible good candidates for vanishing elements. At this point, we have two complementary options: count the number of elements below threshold or the number of elements above it. If we have a mass matrix M, the functions that can achieve this counting have the following asymptotic behavior:

$$f_{\text{vc}}(M_{jk};\epsilon_{\text{vc}},\ldots) \approx \begin{cases} 0 & \text{if } |M_{jk}| \gg \epsilon_{\text{vc}}, \\ 1 & \text{if } |M_{jk}| \ll \epsilon_{\text{vc}}, \end{cases} \tag{22}$$

$$f_{\text{nvc}}(M_{jk};\epsilon_{\text{nvc}},\ldots) \approx \begin{cases} 0 & \text{if } |M_{jk}| \ll \epsilon_{\text{nvc}}, \\ 1 & \text{if } |M_{jk}| \gg \epsilon_{\text{nvc}}, \end{cases} \tag{23}$$

where ϵ_{vc} and ϵ_{nvc} are thresholds mentioned earlier. The ellipsis serves as a placeholder for additional parameters that the actual $f_{\text{vc,nvc}}$ may have. These functions need to be applied to all the elements of the mass matrix and then added to a total loss function of the form

$$L_{\text{vc}}(M;\epsilon_{\text{vc}},\ldots) = \left|n_{\max} - \sum_{j,k} f_{\text{vc}}(M_{jk};\epsilon_{\text{vc}})\right|, \tag{24}$$

$$L_{\text{nvc}}(M;\epsilon_{\text{nvc}},\ldots) = \sum_{j,k} f_{\text{nvc}}(M_{jk};\epsilon_{\text{vc}}), \tag{25}$$

where $n_{\max}$ in Eq. (24) can be used to limit the number of elements. In principle, we can also limit the number of elements above threshold in Eq. (25), but in practice it will hardly reach zero as that would mean that all the elements have been brought considerably below threshold. For a randomly initialized 3×3 matrix, the sums in Eqs. (24) and (25) will most likely start as 0 and 9, respectively, unless the respective ϵ threshold is chosen already among the same size as the elements of the matrix. In Eq. (24), the optimization process will bring the sum closer to $n_{\max}$, while for Eq. (25) it will attempt to reduce the value of the sum as much as possible. As usual, there is the possibility that minimizing any loss function with these forms gets stuck in a flat region or a local minimum, which will be consistent with textures with a particular number of vanishing elements. This is actually advantageous, as after many attempts it will allow the discovery of candidate textures with different numbers of vanishing elements depending on the random initialization of all the variables involved. In what follows, we will explain in detail the application of these loss functions and show numerical results for the obtained mass matrices and the corresponding optimized parameters.

There are several possibilities for the counting functions with the behavior shown in Eqs. (22) and (23). One simple form is given by

$$f_{\text{vc-s}}(M_{jk};\epsilon_{\text{vc}},\alpha) = \frac{\epsilon_{\text{vc}}^\alpha}{\epsilon_{\text{vc}}^\alpha + |M_{jk}|^\alpha}, \tag{26}$$

$$f_{\text{nvc-s}}(M_{jk};\epsilon_{\text{nvc}},\alpha) = \frac{|M_{jk}|^\alpha}{\epsilon_{\text{nvc}}^\alpha + |M_{jk}|^\alpha}, \tag{27}$$

where the ϵ and α parameters allow to fine tune the behavior of the functions, affecting the performance of the optimization. They can be decided beforehand, and we will not vary them during optimization to improve its stability. However, one could devise more complicated training schedules that update, for example, the threshold based on the updated values of the mass matrix elements.

B. Random baseline

As a first test, we randomly select the angles and phases in V^R in random uniform distributions. We limit the angles θ_{jk}^R to the range $[0,\pi/2]$ while for the phases we use the full range from $[-\pi,\pi]$. From this first test, it is not expected that we will be able to say anything new with regard to neutrino mass textures. However, we obtain an interesting tendency with regard to the element M_{D11}^ν. After running 5000 tests, randomly selecting all the variables in Eq. (8),

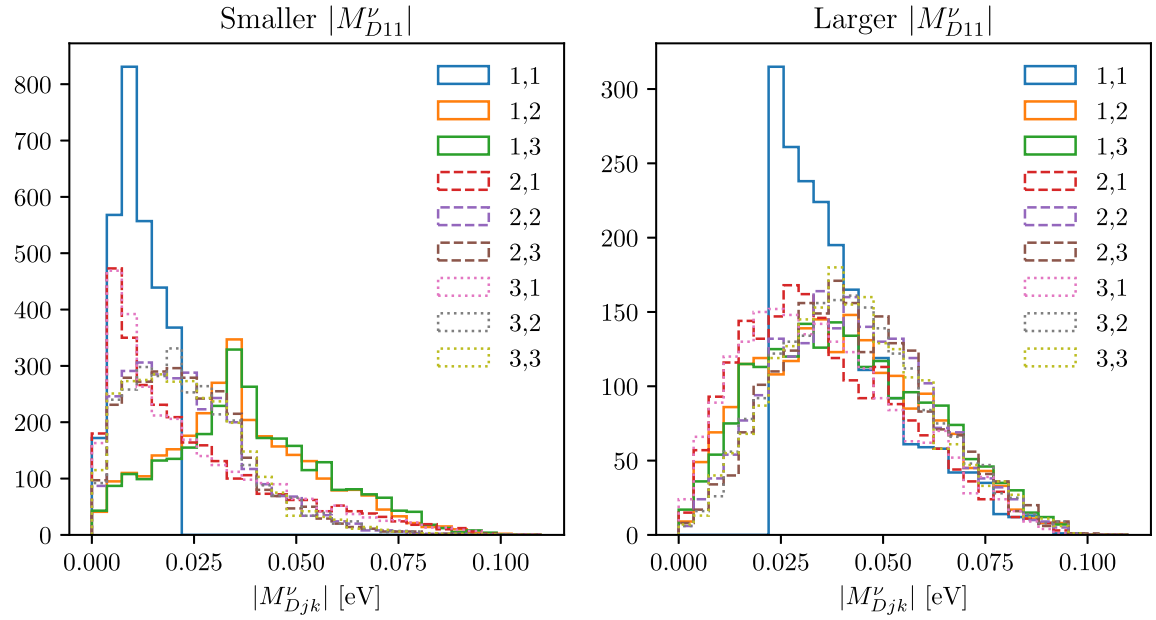


FIG. 2. Distributions for the random test of sizes of M^{ν}_{D} entries. The distributions have been separated according to the size of $|M^{\nu}_{D11}|$, smaller $|M^{\nu}_{D11}|$ (left) is less than 0.022 eV while larger $|M^{\nu}_{D11}|$ (right) is above the same number.

we find that $|M^{\nu}_{D11}|$ has a tendency to peak sharply around ∼0.01 eV.

In Fig. 2 we display distributions of elements according to a hard cut applied on $|M^{\nu}_{D11}|$ at 0.022 eV. In the left panel we show results around the peak, i.e., $|M^{\nu}_{D11}| < 0.022$ eV, where we find that elements in the same row, $|M^{\nu}_{D12}|$ and $|M^{\nu}_{D13}|$, have a tendency to exhibit sharp peaks at around 0.035–0.040 eV. Elements in the same column as $|M^{\nu}_{D11}|$, i.e., $|M^{\nu}_{D21}|$ and $|M^{\nu}_{D31}|$, exhibit peaks at around half the position of the peak in $|M^{\nu}_{D11}|$. Other elements do not feature sharp peaks and follow similar distributions with maximums close to 0.02 eV. This already suggests the presence of relationships among elements of M^{ν}_{D}, in particular for those in the same row and column of $|M^{\nu}_{D11}|$ which present peaks at similar positions. In the right panel of the same figure we show distributions for samples to the right of the peak, i.e., $|M^{\nu}_{D11}| > 0.022$ eV. We find that all the distributions for the other elements tend to have similar shapes with their maximums at approximately the same positions, in the range 0.02–0.05 eV. These distributions are roughly consistent with the sizes of $|M^{\nu}_{D11}|$ shown in the same panel. The fact that the histograms for all the elements in this panel have roughly the same distribution, peaking at similar nonzero values, could be interpreted as a hint at an approximately democratic texture. In fact, the exactly tribimaximal mixing is known to diagonalize the exactly democratic texture. However, tribimaximal mixing is ruled out by the nonzero θ_{13} measurement, which contributes to the little variations between distributions in the right panel of Fig. 2. Yet, it is interesting to see from this random test some coincidences between the distributions of some elements, considering that we have not assumed a particular model other than Dirac neutrino masses and we have used only random values.

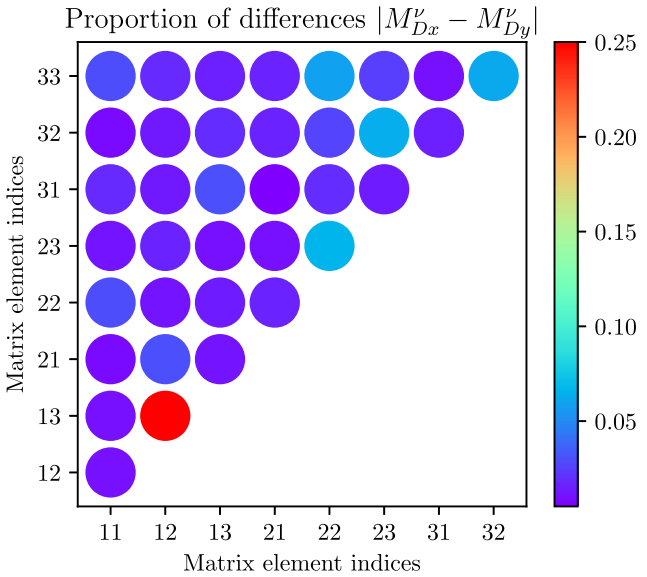


FIG. 3. Proportion of times that each pair of elements was choose for minimization in the loss function of Eq. (20). Each circle represents a difference between the element with index in x axis minus the element with index in y axis. Only half is filled due to the symmetry $|M^{\nu}_{Djk} - M^{\nu}_{Dlm}| = |M^{\nu}_{Dlm} - M^{\nu}_{Djk}|$. The proportion represented by the color is the number of times the difference was chosen over the total number of tests that were used.

C. Learning equality between two entries

In this section, we proceed to actually attempt to *learn* the parameters of V^R by minimizing a loss function that includes only the absolute value of the difference between two elements. The loss function for this is given in Eq. (20). We sample neutrino oscillation parameters randomly using Gaussian distributions as described above. The parameters of V^R will be initialized randomly from uniform distributions as indicated before. In Fig. 3 we show the proportion of times that each pair of matrix elements has the minimum difference in 10,000 random tests. We see that the difference $|M^\nu_{D12} - M^\nu_{D13}|$ happens to be the most common one, occurring almost 25% of the times the test was run. It is followed by $|M^\nu_{D22} - M^\nu_{D23}|$, $|M^\nu_{D32} - M^\nu_{D33}|$, $|M^\nu_{D22} - M^\nu_{D33}|$, and $|M^\nu_{D23} - M^\nu_{D32}|$. We see the appearance of vanishing $|M^\nu_{D22} - M^\nu_{D33}|$ and $|M^\nu_{D23} - M^\nu_{D32}|$ differences with a similar rate. This can be interpreted as a hint at the popular μ-τ symmetry [25–30], as it can be seen that also the $|M^\nu_{D12} - M^\nu_{D13}|$ tends to vanish often. However, note that this analysis mostly attempts to find single vanishing differences. As mentioned before, the values for the measured oscillation parameters are sampled from Gaussian distributions that fall inside the experimental range and are not included in the loss function. From the resulting M^ν_D after initialization, we select the minimum absolute difference between elements and use it in the loss function of Eq. (20). To minimize the loss function, we perform 1000 optimization steps using the Adam [31] algorithm, updating the parameters of V^R at each step. We use an initial learning rate of 0.01 and reduce it by

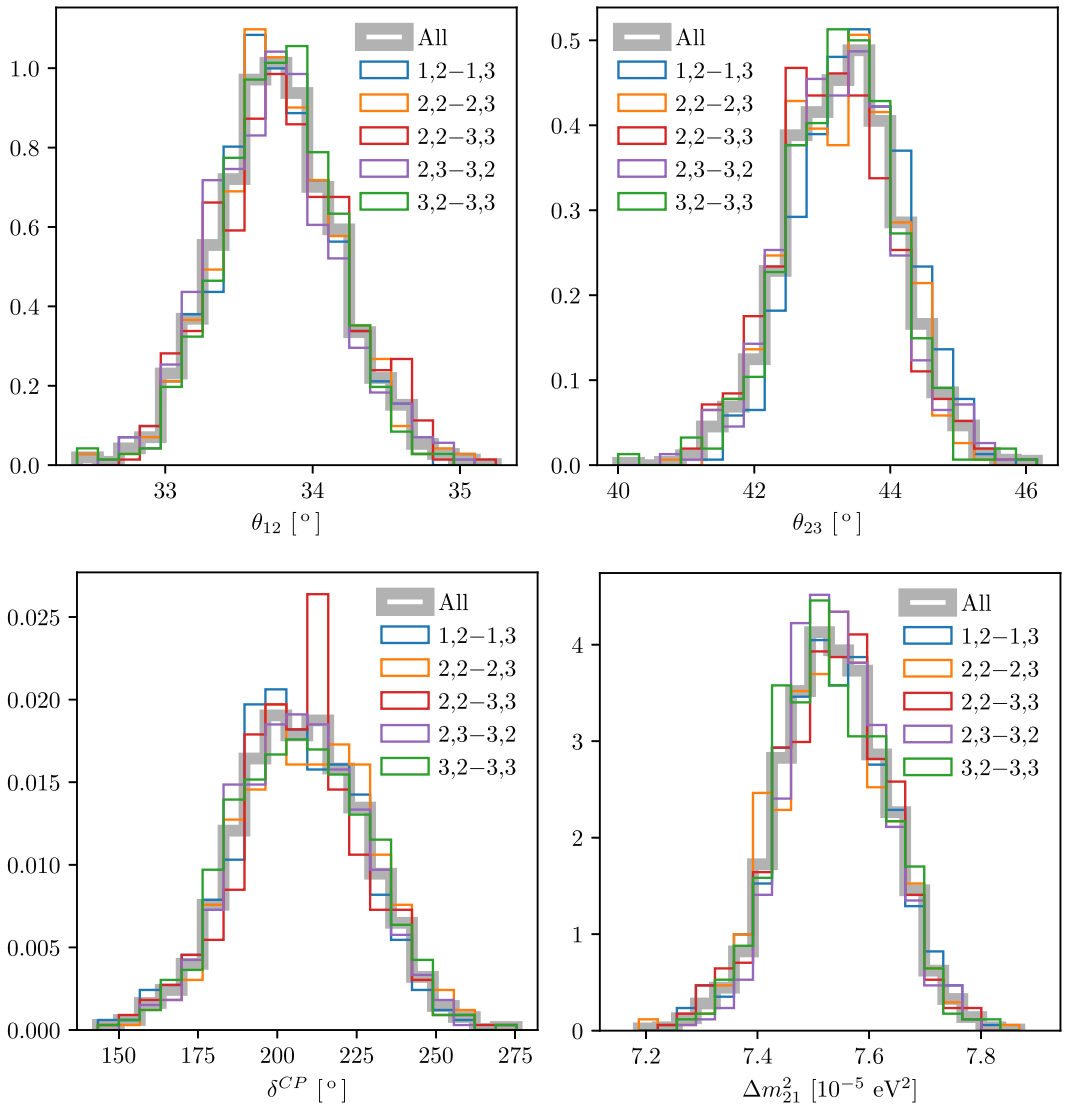


FIG. 4. Distributions for the neutrino oscillation parameters for specific pairs of elements that were chosen for difference minimization. The gray histogram shows the distribution for all the cases. The colored histograms show the distribution only for selected differences.

a factor of 0.9 every 10 steps. If the loss function remains above 10^{-6}, we apply another 1000 optimization steps with a fixed learning rate of 0.001. The final result is an M_D^ν matrix, where at least two elements are equal down to several decimal places, as well as the corresponding parameters for V^R. We repeat this process (random initialization and minimization) for each of the 5 most common cases, 1000 times each, and obtain values for the parameters contained in V^R corresponding to before and after minimization.

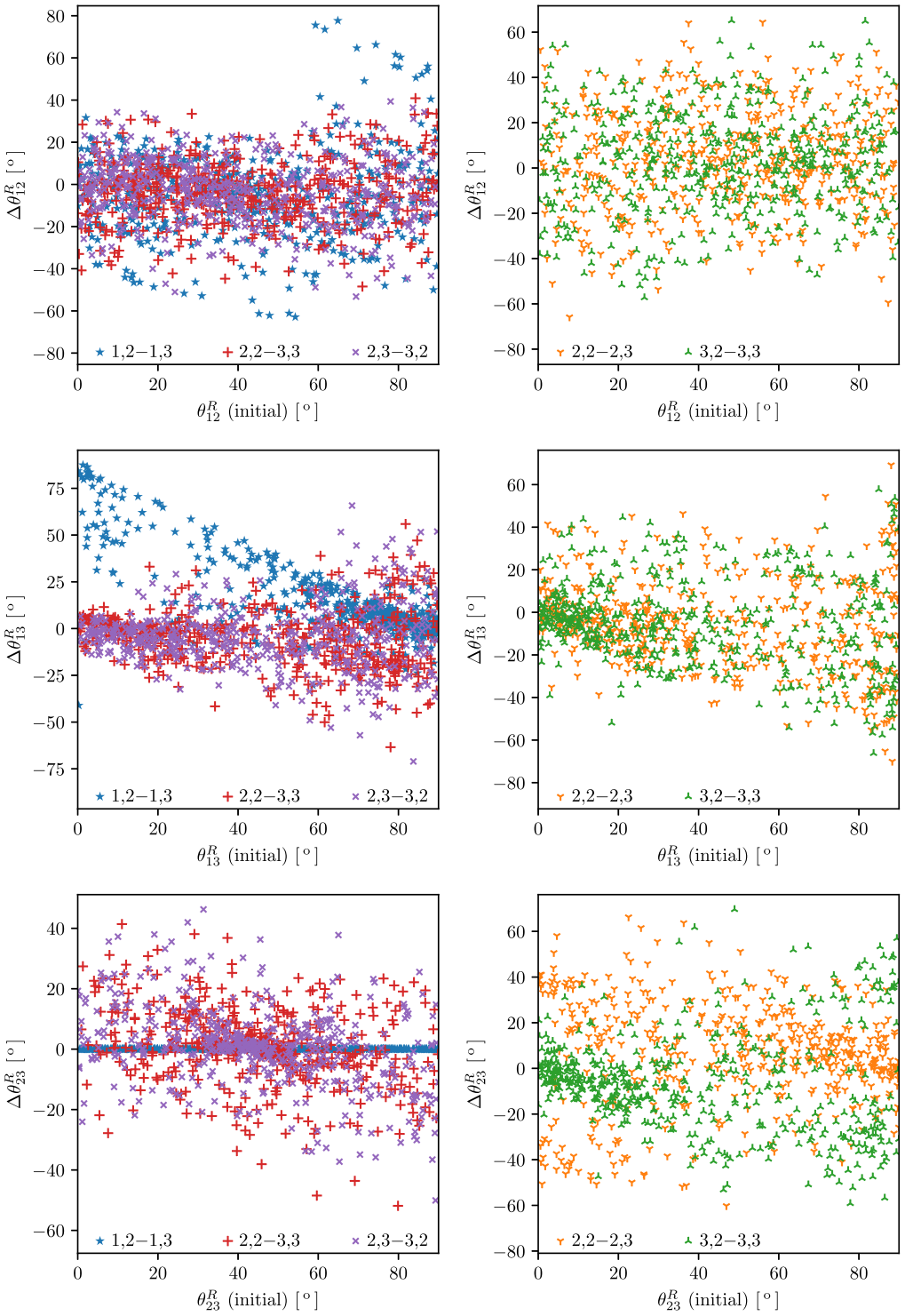


FIG. 5. Change in V^R parameters after the minimization. The horizontal axis shows initial value, while the vertical axis shows the change in parameter value.

Additionally, we check whether selecting specific differences has any effect on the distributions of the neutrino oscillation parameters. In Fig. 4, we show the distributions of θ_{12}, θ_{23}, δ^{CP}, and Δm^2_{21} for all the selected differences in color, while the gray histogram shows the sampled distribution for the corresponding parameters, regardless of the differences. We see that colored and gray distributions follow similar distributions, indicating that selecting those particular differences has almost no effect on the distribution of oscillation parameters. The parameters that are not displayed, θ_{13} and Δm^2_{31}, have been fixed to their central values. Note that colored distributions can go above the gray distribution because they have been normalized to compare their shapes. Therefore, comparisons of height between the different distributions are meaningless.

In the case of the parameters of V^R, we expect to see a heavy dependence on the differences, since these are the parameters that we have optimized in the minimization process. Namely, the parameters that have been included in the minimization process are θ^R_{jk} ($jk = 12, 13, 23$), δ^R, ϕ^R_2, and ϕ^R_3. In Fig. 5, we show the displacement of the θ^R_{jk} parameters from their initial values, with the initial values given in the x-axis. On the y-axis, for each parameter, we use Δ to represent the difference between the parameter value after optimization minus its initialization value, given on the x axis. In the case of the parameter θ^R_{12}, we see in the upper-left panel of Fig. 5 that the differences $|M^\nu_{D12} - M^\nu_{D13}|$, $|M^\nu_{D22} - M^\nu_{D33}|$, and $|M^\nu_{D23} - M^\nu_{D32}|$ tend to accumulate between 20° and 40°, with the displacement from minimization following a rather random pattern. The displacement for $|M^\nu_{D12} - M^\nu_{D13}|$ is much larger than for the other two differences in the same figure and grows above the initial 60°. In the upper-right panel, we see that $|M^\nu_{D22} - M^\nu_{D23}|$, and $|M^\nu_{D32} - M^\nu_{D33}|$ have no preference for θ^R_{12} and that displacement from minimization is similar in both cases. In the middle-left panel, we see that θ^R_{13} accumulates around 90° for $|M^\nu_{D12} - M^\nu_{D13}|$, while for $|M^\nu_{D22} - M^\nu_{D33}|$ and $|M^\nu_{D23} - M^\nu_{D32}|$, it tends to prefer to be closer to 0°. Interestingly, for $|M^\nu_{D12} - M^\nu_{D13}|$, minimization tends to take θ^R_{13} close to 90° while keeping it not exactly at that value, resulting in the diagonal tendency that can be seen from initial 0° to ∼90° in the blue stars. In the middle-right panel, we see that θ^R_{13} is less sensitive to minimization when it is initialized closer to 0° for both differences $|M^\nu_{D22} - M^\nu_{D23}|$ and $|M^\nu_{D32} - M^\nu_{D33}|$. In the lower panels, we display the displacement of θ^R_{23} as a function of its initial value. In the lower-left panel we see that for the cases $|M^\nu_{D22} - M^\nu_{D33}|$ and $|M^\nu_{D23} - M^\nu_{D32}|$ most initializations start around 45°, while the case $|M^\nu_{D12} - M^\nu_{D13}|$ seems to be insensitive to the initial value of θ^R_{23} even showing no displacement after minimization. We also see that optimization tends to displace θ^R_{23} upward when initialized below 45° and downward when initialized above it. In the lower-right panel, we see that the cases $|M^\nu_{D22} - M^\nu_{D23}|$ and $|M^\nu_{D32} - M^\nu_{D33}|$ prefer to start at around 0° and 90°, respectively, with minimization having a rather random distribution.

Initializations and displacements during minimization are shown for phases in Fig. 6. For this case, most initializations happen to be more evenly distributed, and displacements also show little to no dependence on initialization phase. However, a few interesting points can be made about initialization in this case. For example, in the distribution of the initial phase for δ^R, as shown in the upper-right panel, for $|M^\nu_{D32} - M^\nu_{D33}|$ there is a slight preference for initial values in the middle of the range between 0° and 180° or −180°, while for $|M^\nu_{D22} - M^\nu_{D23}|$ this phase appear more uniformly distributed. However, initialization of δ^R remains more evenly distributed when compared to distributions shown in Fig. 5. Something similar can be said about ϕ^R_3, for all the differences (bottom row), and ϕ^R_2 for $|M^\nu_{D22} - M^\nu_{D23}|$ and $|M^\nu_{D32} - M^\nu_{D33}|$ (middle right), which seem to prefer initial values evenly distributed in the whole range and displacements with no clear tendency. One case with noticeable features is the distribution of displacements $\Delta\phi^R_2$ shown on the middle left. Here we can see that for the three differences $|M^\nu_{D12} - M^\nu_{D13}|$, $|M^\nu_{D22} - M^\nu_{D33}|$, and $|M^\nu_{D23} - M^\nu_{D32}|$, optimization gives mostly positive displacements for $\Delta\phi^R_2$ when initialized below 0° and mostly negative displacements when it is initialized above 0°. This indicates that for these three differences optimization brings ϕ^R_2 closer to 0°.

In the following, we give a few numerical examples of M^ν_D after minimization. These matrices contain two elements with the same complex value and can be checked that they can be diagonalized as in Eq. (4), with U^ν consistent with a U_{PMNS} matrix with oscillation parameters within experimental ranges. After each matrix, we show the numerical results obtained for the physical mixing parameters and m_ν obtained from the values set during initialization (see the second paragraph of Sec. III and Fig. 1):

$$\frac{M^\nu_D}{10^{-2}\ \text{eV}} = \begin{pmatrix} 0.42462 + 0.59258i & -3.35011 + 0.19065i & -3.35011 + 0.19065i \\ -0.51059 + 0.07189i & -0.5556 - 0.08104i & 0.4283 - 0.22683i \\ -0.06563 + 0.09068i & -0.0642 - 0.92411i & -0.29787 - 1.08555i \end{pmatrix},$$

$$s^2_{12} = 0.3114, \qquad s^2_{23} = 0.492, \qquad s^2_{13} = 0.02249, \qquad \delta^{CP} = 238.47°, \qquad J = -0.028926,$$

$$\Delta m^2_{21} = 7.574 \times 10^{-5}\ \text{eV}^2, \qquad \Delta m^2_{31} = 2.521 \times 10^{-3}\ \text{eV}^2, \qquad m_\nu = 8.9992 \times 10^{-3}\ \text{eV}, \tag{28}$$

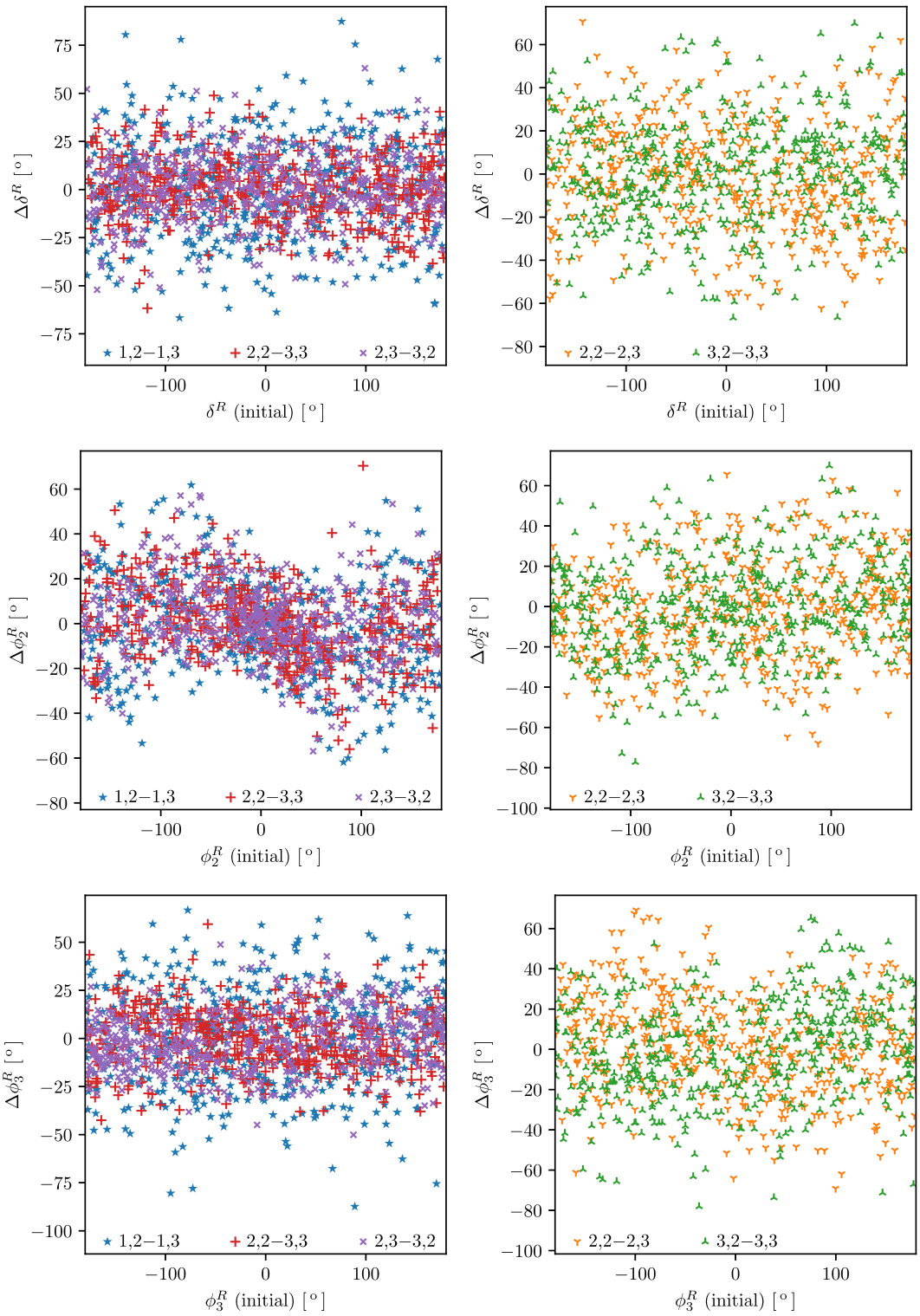


FIG. 6. Change in V^R parameters after the minimization. The horizontal axis shows initial value, while the vertical axis shows the change in parameter value.

$$\frac{M_D^\nu}{10^{-2}\text{ eV}} = \begin{pmatrix} 0.4842 + 0.91175i & -2.10139 - 5.06338i & -2.05571 - 5.10392i \\ -5.96953 + 0.5551i & -0.51133 + 0.01792i & -0.51133 + 0.01792i \\ 0.06789 + 0.0167i & 4.18078 + 0.68614i & -4.26808 - 0.7743i \end{pmatrix},$$
$$s_{12}^2 = 0.3041, \quad s_{23}^2 = 0.4753, \quad s_{13}^2 = 0.02249, \quad \delta^{CP} = 171.39°, \quad J = 0.0050392,$$
$$\Delta m_{21}^2 = 7.541 \times 10^{-5}\text{ eV}^2, \quad \Delta m_{31}^2 = 2.521 \times 10^{-3}\text{ eV}^2, \quad m_\nu = 6.0839 \times 10^{-2}\text{ eV}, \tag{29}$$

$$\frac{M_D^\nu}{10^{-2}\text{ eV}} = \begin{pmatrix} 9.35176 + 0.04943i & 0.42776 - 0.5038i & 0.3971 - 0.52925i \\ -0.0795 - 0.78652i & 8.79824 + 3.1097i & -0.66421 + 3.40956i \\ -0.07241 - 0.83716i & -0.66421 + 3.40956i & 8.68342 + 3.69522i \end{pmatrix},$$
$$s_{12}^2 = 0.3155, \quad s_{23}^2 = 0.4571, \quad s_{13}^2 = 0.02249, \quad \delta^{CP} = 207.02°, \quad J = -0.015416,$$
$$\Delta m_{21}^2 = 7.507 \times 10^{-5}\text{ eV}^2, \quad \Delta m_{31}^2 = 2.521 \times 10^{-3}\text{ eV}^2, \quad m_\nu = 9.4228 \times 10^{-2}\text{ eV}. \tag{30}$$

D. Discovering *n*-zero textures

Taking the previous section as a demonstration of the general process of selection of texture and minimization, now we proceed to apply the same process to a more complicated case, i.e., learning parameters for *n*-zero textures. The study of such textures is popular in the literature [32–54] as they can be obtained from minimal models with a reduced number of parameters. Thus, they represent predictive restricted scenarios. Such textures typically are obtained from hypothesizing a combination of new symmetries and fields that result in vanishing entries in the neutrino mass matrix. Note that we do not expect the optimization process to determine elements with exactly vanishing values, but instead we will consider the matrices which are found to have *n* elements much smaller than the rest to be the *n*-zero candidates. To determine if a particular matrix is a successful candidate, we will set its *n* smaller elements to exactly vanish and recalculate the corresponding predictions for the numbers in Table I using Eqs. (12)–(18). In the two methods presented next, we will initialize oscillation parameters using random Gaussian distributions, as described in Sec. III, but we will keep only initializations within 2σ of those Gaussian distributions. Then, for the final matrices with *n* elements set to zero, we will check if the predictions remain within the 2σ range cited in Table I. The matrices that stay within that range will be marked as surviving *n*-zero textures. Particular details for each method will be described in the following subsections.

1. Discovering n-zero textures by random search plus optimization

In this section, we describe our first approach to find zero textures that are hinted at by the measured oscillation parameters. The optimization process will roughly follow the same process as in the previous section but with a loss function specific to this section. We will make a few assumptions during our search:

(i) The number of vanishing entries will be no more than five.
(ii) No column or row will be full of vanishing entries.

The steps we will follow to consider that oscillation parameters hint at a particular *n*-zero texture are as follows:

(1) Sample randomly from oscillation parameters using Gaussian distributions as in the previous section, but only within 2σ ranges. The parameters of V^R and lightest neutrino mass will be sampled from a random uniform distribution.
(2) Use Eqs. (8), (10), and (11) to obtain numerical values for M_D^ν.
(3) Sort the elements of $|M_D^\nu|$ from smallest to largest and take ratios sequentially, $|M_{Dab}^\nu|/|M_{Dcd}^\nu|$, where $|M_{Dcd}^\nu|$ is the value smaller than $|M_{Dab}^\nu|$ but closest to it. This will result in eight ratios characterizing the change in size between elements that are closer in magnitude.
(4) Select the largest ratio, $\max(|M_{Dab}^\nu|/|M_{Dcd}^\nu|)$, representing the biggest change in size between elements, as the division between elements that will vanish and elements that do not. For example, after determining $\max(|M_{Dab}^\nu|/|M_{Dcd}^\nu|)$ for a particular ab entry, all the entries with magnitude below $|M_{Dab}^\nu|$ will be considered as the elements that vanish.
(5) Construct a loss function that minimizes the magnitude of the entries that were selected in the previous step.
(6) Apply minimization as was done in the previous section.

The loss function employed in this approach is given in Eq. (21) and constructed with the set of elements chosen in step 4 of the process described above as vanishing elements. This loss function works by separating the values of the average of the two sets of elements, making the numerator small and the denominator large relative to each other. For this case, we will also use the lightest neutrino mass as part of the minimization. Therefore, we will also include a term for KATRIN's measurement of the effective

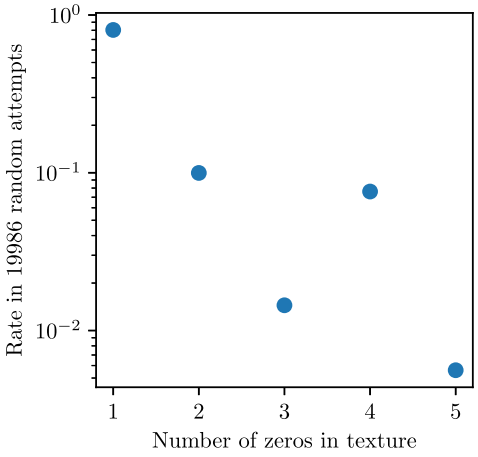


FIG. 7. Rate of found textures by number of zeros by using the random search described in Sec. III D.

electron-antineutrino mass as given in Eq. (19), with the weight $w_K = 1000\ \text{eV}^{-2}$.

Given the process described above, the rate of finding textures with each number of zeros is shown in Fig. 7. As expected, in this figure we can see that it is easier to find 1-zero textures. Additionally, we see that the rate goes down as we search for more zeros, except in the case of 4-zero textures where the rate is comparable to 2-zeros. To try to quantify the minimizing power of the process above, we search randomly for n-zero textures as described above and collect a total of 500 examples per n-zero texture, with $n = 1, 2, 3, 4, 5$. Initially, we minimize the loss function during 1000 optimization steps, attempting 1000 more steps in cases where the loss function of Eq. (21) does not go below 10^{-18}. For the optimization, we use the Adam algorithm with an initial learning rate of 0.001 reduced by a factor of 0.99 every 10 steps.

As previously discussed, we do not expect the optimization process to deliver matrices with exactly n elements equal to zero but instead to deliver n elements much smaller than the others. These optimized matrices will be considered n-zero texture candidates. Then, we do a final test by setting the n smallest elements to zero and checking if the resulting matrix still follows experimental constraints. In particular, we require that the oscillation parameters remain within 2σ of the measurements cited in Table I and that the value of m_ν remains below 0.45 eV, the upper limit set by KATRIN [22]. The patterns of absolute values for the elements of M_D^ν obtained by the optimization process described here are shown in Fig. 8. The elements have been sorted according to their absolute values which are shown in increasing order from left to right. The red bars show results for all optimized matrices, while the blue bars show results only for the matrices that survived the experimental tests after setting the smallest n elements to zero. The dashed gray line separates the elements that were set to zero from those that were kept at their optimized values. The numbers in parentheses in the legend indicate the number of displayed results corresponding to that label. For 1-zero candidates (upper left) and 2-zero candidates (upper right), we see that minimization worked as expected delivering one and two much smaller elements, respectively. For these two cases, we also get a survival of ~490 out of 500 optimized matrices. In the case of 3-zero candidates (middle left), we see that the minimization performance is highly effective, sometimes delivering more than three substantially smaller elements. This is related to cases where the optimization attempts to minimize two elements in the same row resulting in accidental minimization of the remaining element. Survival in this case is lower with 225 cases passing the applied tests. For 4-zero candidates (middle right) and 5-zero candidates (lower), the optimization was much less efficient, delivering inconsistent minimizations and, moreover, resulting in no surviving cases. One particular characteristic of the 4- and 5-zero optimizations is that a single element seems to have been minimized more efficiently than the rest. The three most commonly surviving textures found using the algorithm described here are shown in Fig. 9 for $n = 2$ and 3. Counts for the number of times each texture occurs are shown above each case, counting also equivalent textures that are row permutations of the displayed cases.

Some examples of numerical matrices for 2- and 3-zeros are given below, including their corresponding values for observables. The numerical values for the mixing observables and m_ν shown below each matrix have been obtained using Eqs. (12)–(18):

$$\frac{M_D^\nu}{10^{-2}\ \text{eV}} = \begin{pmatrix} 0.74834 + 0.25015i & -3.46065 + 0.12821i & -3.6963 + 0.18086i \\ -0.80448 - 0.80651i & -0.27798 - 0.31313i & 0 \\ 0 & -0.71088 + 0.50896i & 0.63125 - 0.6378i \end{pmatrix},$$
$$s_{12}^2 = 0.3025, \quad s_{23}^2 = 0.4656, \quad s_{13}^2 = 0.02249, \quad \delta^{CP} = 203.04°, \quad J = -0.013145,$$
$$\Delta m_{21}^2 = 7.373 \times 10^{-5}\ \text{eV}^2, \quad \Delta m_{31}^2 = 2.521 \times 10^{-3}\ \text{eV}^2, \quad m_\nu = 1.3857 \times 10^{-2}\ \text{eV}, \tag{31}$$

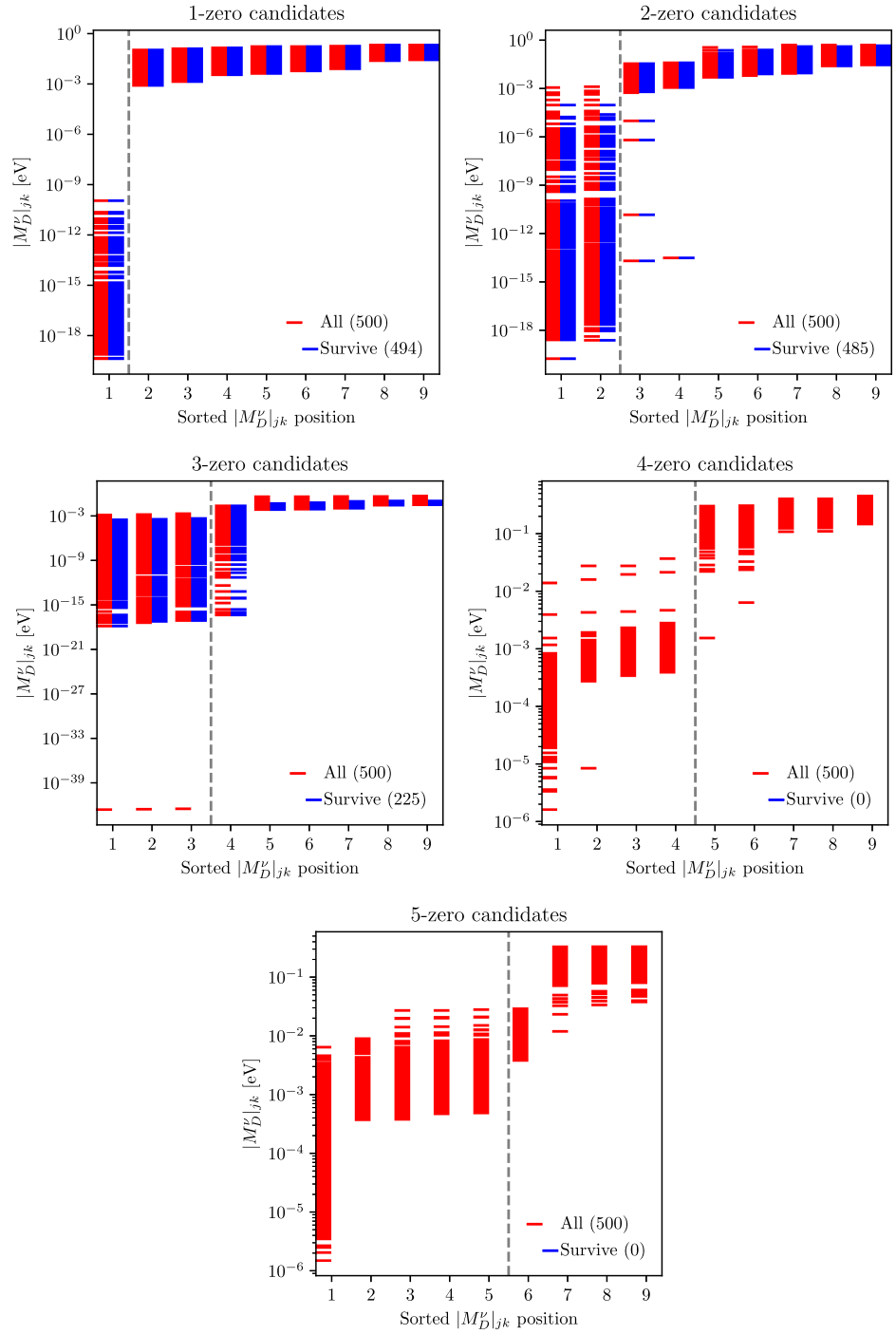


FIG. 8. Sorted absolute values of elements for the optimized matrices using random search as described in Sec. III D 1. The red bars show all the obtained n-zero texture candidates. The blue bars correspond to matrices where the n smallest absolute values have been set to zero while still obtaining all oscillation parameters inside their 2σ ranges (labeled "Survive"). The number in parenthesis shows counts of M_D^ν obtained in the process.

$$\frac{M_D^\nu}{10^{-2}\ \text{eV}} = \begin{pmatrix} 1.90766 & -1.02225 + 0.59142i & -1.40581 + 0.6545i \\ 0 & 2.35306 + 0.01696i & 0 \\ 0 & 2.70405 + 0.04954i & 3.76622 \end{pmatrix},$$

$$s_{12}^2 = 0.3195, \quad s_{23}^2 = 0.4495, \quad s_{13}^2 = 0.02249, \quad \delta^{CP} = 207.04°, \quad J = -0.015457,$$

$$\Delta m_{21}^2 = 7.616 \times 10^{-5}\ \text{eV}^2, \Delta m_{31}^2 = 2.521 \times 10^{-3}\ \text{eV}^2, \quad m_\nu = 1.9077 \times 10^{-2}\ \text{eV}. \tag{32}$$

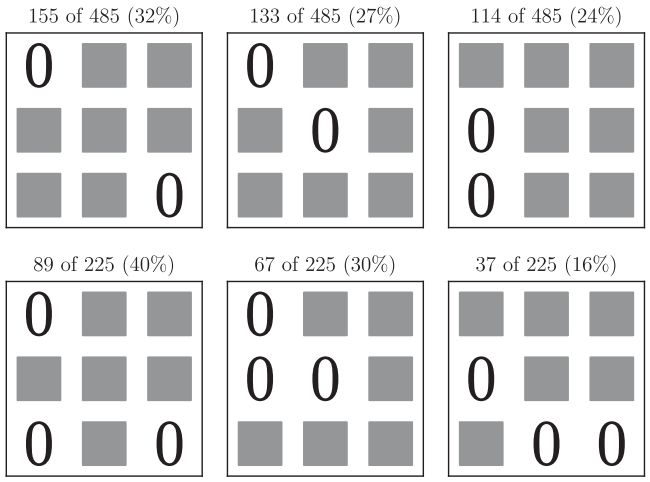


FIG. 9. The three most commonly found n-zero textures using the random search algorithm of Sec. III D 1 for $n = 2$ (top) and 3 (bottom). The counts above each figure represent the number of times the texture occurs in the surviving cases of Fig. 8. These counts also consider equivalent textures due to row permutations. The figures are sorted from most common on the left and descending to the right. The gray squares correspond to values shown in Fig. 8 to the right of the gray dashed vertical line (2-zero and 3-zero cases).

For the δ^{CP} values given above, due to solving from Eq. (16) to obtain Eq. (17), there is an ambiguity when using the arcsine function, where positive $\sin\delta^{CP}$ could result in δ^{CP} in quadrant I or II, while negative $\sin\delta^{CP}$ could give δ^{CP} in quadrant III or IV. To solve this ambiguity, we consider the solution for δ^{CP} that is closer to our initialization range, while checking that this maintains consistency with all numerical results and parametrizations. Now that we have given examples of numerical matrices for 2- and 3-zero textures as well as their predictions, we proceed to explore how to discover n-zero textures via optimization.

2. Discovering n-zero textures via optimization

The idea of this section is to use the optimization process of minimizing the loss function to actually identify elements in the neutrino mass matrix that could be considerably smaller than others. We do not expect that minimization itself will make them completely vanish, but we will be comfortable with them being several orders of magnitude smaller such that setting them to zero makes little to no difference. We will test the two loss functions given in Eqs. (24) and (25), to determine if there is any

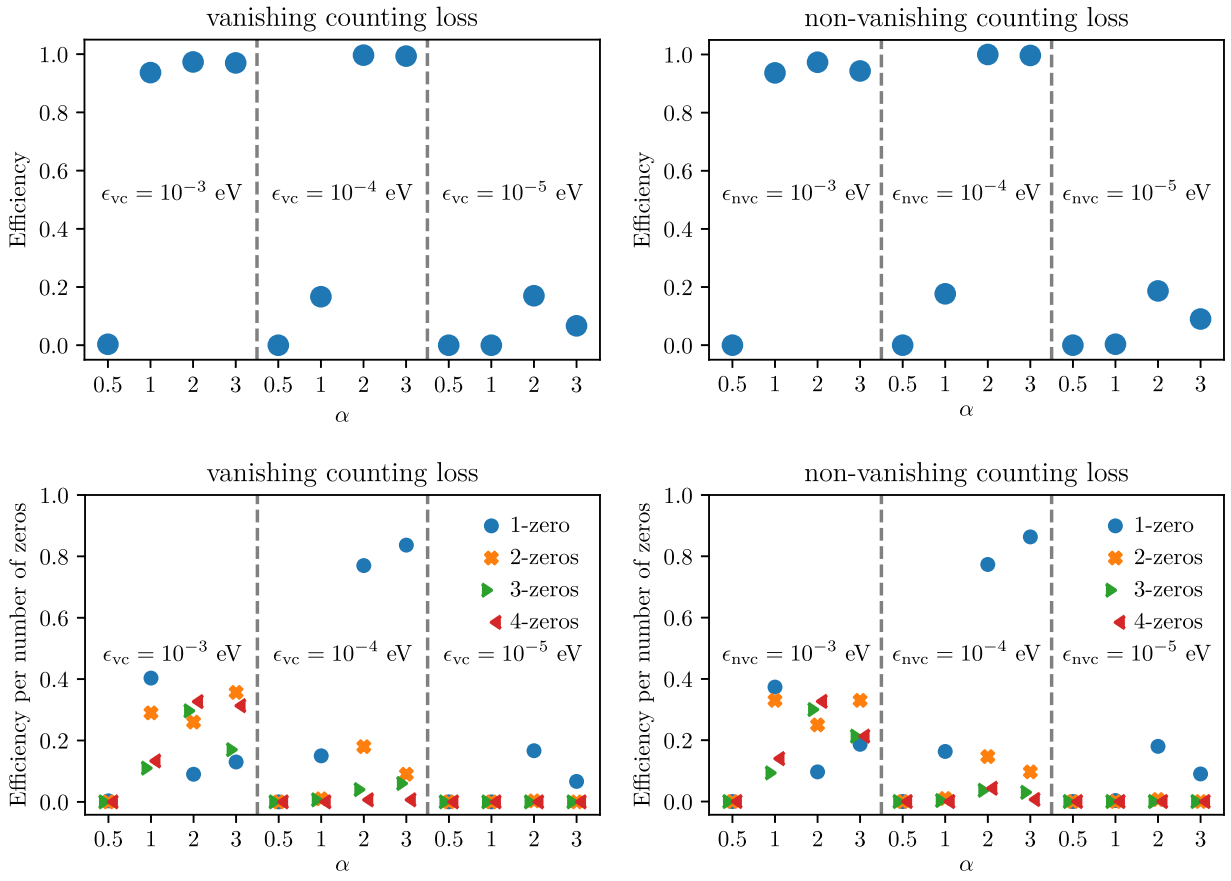


FIG. 10. Efficiencies for the element counting loss functions.

difference in behavior when counting vanishing or nonvanishing elements, respectively.

The strategy to find textures and the parameters that characterize them is the following:

(1) Randomly initialize all the parameters as in the previous cases.
(2) Optimize the parameters using the loss function of either Eq. (24) or (25) to determine which elements can be counted as vanishing. This can be done by selecting elements that were counted as vanishing in the last optimization step.
(3) Attempt to fully minimize the selected elements as much as possible if they have not reached the point of being vanishingly small when compared to other elements.
(4) Count the actual number of vanishing elements depending on the outcome of this second minimization.

The minimization in point number 3 above is done with a simple loss function with the elements that were selected as vanishing from the first minimization.

We start by estimating good values for the loss function parameters $\epsilon_{\mathrm{vc,nvc}}$ and α. To do this, we perform 300 minimizations of the loss functions in Eqs. (24) and (25), with individual matrix element losses defined by Eqs. (26) and (27), respectively. We use all combinations of the values $\epsilon_{\mathrm{vc,nvc}}/\mathrm{eV} \in \{10^{-3}, 10^{-4}, 10^{-5}\}$ and $\alpha \in \{0.5, 1, 2, 3\}$. Considering that minimizing n elements of a complex matrix is equivalent to attempting to bring n real parts and n imaginary parts down to zero, using 7 unconstrained

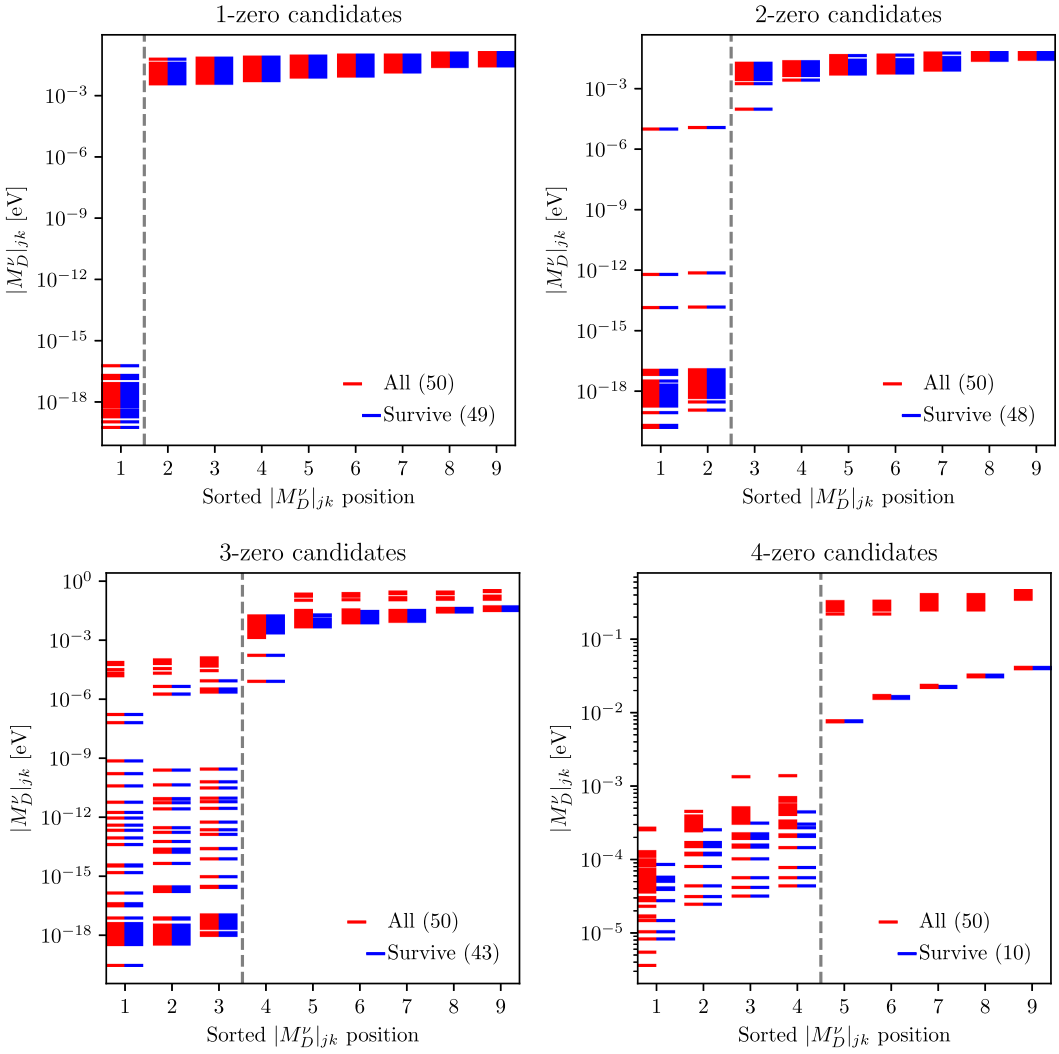


FIG. 11. Sorted absolute values of elements for the optimized matrices using nonvanishing counting loss function search as described in Sec. III D 2. The red bars show all the obtained n-zero texture candidates. The blue bars correspond to matrices where the n smallest absolute values have been set to zero while still obtaining all oscillation parameters inside their 2σ ranges (labeled "Survive"). The number in parenthesis shows counts of M_D^ν obtained in the process.

parameters (three θ^R_{jk}, two $\phi^R_{2,3}$, δ^R, and m_1) may not achieve great results for $n > 3$. Optimistically, we fix $n_{\max} = 5$ in Eq. (24), which means that we will let the search go as far as 5-zero textures. Our objective is to obtain efficiencies that indicate how many times the optimization process results in hints of textures, always starting from randomly initializing all the variables in Eq. (8). Efficiencies are simply calculated as (number of textures found)/(number of optimizations performed). To consider that an element M_{jk} has been "chosen" as vanishing by the loss function, we set a threshold of $f_{\rm vc}(M_{jk}) > 0.99$ $(f_{\rm nvc}(M_{jk}) < 0.01)$. The resulting efficiencies from this test are shown in Fig. 10. Interestingly, when considering global efficiency (finding at least 1-zero texture), shown in the two upper panels of Fig. 10, we find that efficiencies have very similar values across all the values tested for $\epsilon_{\rm vc,nvc}$ and α. We also note that, as $\epsilon_{\rm vc,nvc}$ decreases, it becomes difficult to reach an efficiency above 0.5. The highest efficiencies are achieved with $\epsilon_{\rm vc,nvc} = 10^{-4}$ eV, for $\alpha = 2,\ 3$. However, using $\epsilon_{\rm vc,nvc} = 10^{-4}$ eV results in high global efficiency for $\alpha \in \{1, 2, 3\}$. Using $\epsilon_{\rm vc,nvc} = 10^{-5}$ eV gives the worst performance for all the values of α considered. When looking at efficiencies for individual textures, shown in the two bottom panels of Fig. 10, we see that for $\epsilon_{\rm vc,nvc} \in \{10^{-4}\ \text{eV}, 10^{-5}\ \text{eV}\}$, 1-zero textures dominate over the other textures, while a more even distribution of textures is found for $\epsilon_{\rm vc,nvc} = 10^{-3}$, at least for $\alpha \in \{1, 2, 3\}$. Notably, for $\alpha \in \{2, 3\}$ more optimizations resulted in textures with more than one zero. It is precisely here where the vanishing and nonvanishing counting loss functions show some difference, as, apparently, the nonvanishing counting loss with $\epsilon_{\rm vnc} = 10^{-3}$ eV and $\alpha = 3$ seems to have a more even distribution of efficiencies, with all 1-to-4-zero textures in the range 0.2–0.4. Expectedly, none of the combinations of $\epsilon_{\rm vc,nvc}$ and α, gave a loss function that achieved any textures with more than 4-zeros. Therefore, 5-zero textures and above should have an efficiency below $1/300$. Again, this is closely related to using only 7 free parameters during optimization.

Considering the information gained on efficiency per texture, we perform a dedicated search using the nonvanishing counting loss with parameters set at $\epsilon_{\rm nvc} = 10^{-3}$ eV and $\alpha = 3$. We attempt a search of n-zero textures with $n \leq 4$, by performing multiple optimizations and collecting results after 1000 steps. The optimization algorithm is Adam with a fixed learning rate of 0.001. At the end of the 1000 optimization steps, we determine which elements have absolute values much smaller than the rest and use that information to count the resulting number of vanishing elements. In particular, we require that the value of $f_{\text{nvc-s}}((M^\nu_D)_{jk})$ of Eq. (27) is below a threshold of 0.01 to consider that the $(M^\nu_D)_{jk}$ element could vanish. After that, we attempt to minimize the chosen elements again but this time with a loss function of the type given in Eq. (21). This second optimization is analogous to the one of Sec. III D 1, but with an initial learning rate of 0.001 and a reduction by a factor of 0.99 every 10 steps. As before, we require that all the found textures have at least one nonzero element per column and row. This search takes longer, and therefore we only attempt to collect 50 candidates per texture. The sorted absolute values for the collected candidates are shown in Fig. 11, including the results for the surviving cases. As in the previous approach, we set the n elements of M^ν_D with the smallest absolute values to zero and use Eqs. (12)–(18) to obtain the corresponding observables. We consider the cases where the observables remain inside 2σ ranges as surviving cases, shown as blue bars in Fig. 11. For 1- and 2-zero candidates, shown in the upper-left and upper-right panels, respectively, the result is somewhat compatible with those shown in Fig. 8. The sizes of sorted elements is similar in both figures, and the survival rates have a similar size when considering the "survive" to "all" ratio. However, considerable differences appear for 3- and 4-zero candidates, shown in the lower-left and lower-right panels, respectively. The survival rates (as "survive" to "all" ratio) are larger than in the corresponding cases in Fig. 8, most

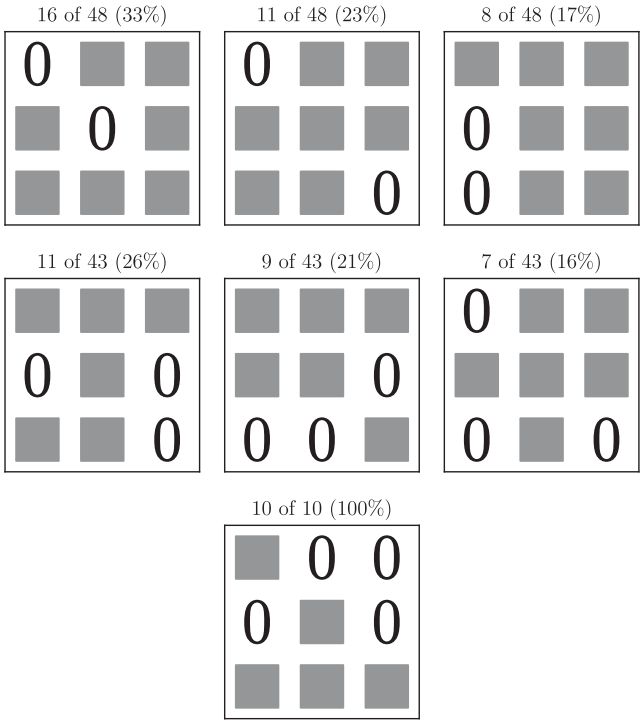


FIG. 12. The three most commonly found n-zero textures using the optimizer search algorithm of Sec. III D 1 for $n = 2$ (top) and 3 (middle). For $n = 4$, all cases found are consistent with the texture shown in the bottom. The counts above each figure represent the number of times the texture occurs in the surviving cases of Fig. 11. These counts also consider equivalent textures due to row permutations. The figures are sorted from most common on the left and descending to the right. The gray squares correspond to values shown in Fig. 11 to the right of the gray dashed vertical line (2-zero, 3-zero, and 4-zero cases).

notably for 4-zero candidates where a few cases were found. In the case of 3-zero candidates, the surviving cases, indeed, have three elements that are much smaller than the other six as one would expect. Similarly, for 4-zero candidates, all the textures collected have four elements always smaller than the other five. The surviving cases have a narrower variation for the elements to the right of the gray dashed line, when compared to the other 1-, 2-, and 3-zero candidates, but this could be related to the low surviving rate. For all four panels shown in Fig. 11, we see that optimization using the nonvanishing-counting loss function of Eq. (25) gives results that most of the time consistently minimize only the indicated number of elements. The reason for this is that this type of loss function identifies the number of elements being minimized and, at the end of optimization, reports the number of elements that were not minimized. The three most commonly surviving textures found using the algorithm described here are shown in Fig. 12 for $n = 2$ and 3. In the case of $n = 4$, all the surviving textures are permutations of the textures displayed in the bottom row of Fig. 12. Counts for the number of times each texture occurs are shown above each case, counting also equivalent textures that are row permutations of the cases displayed.

In the following, we show examples of surviving matrices with 2-, 3-, and 4-zero textures with their corresponding values for the mixing angles and m_ν, obtained in the same way as in Sec. III D 1, with the same solution for the ambiguity of δ^{CP}:

$$\frac{M_D^\nu}{10^{-2}\ \text{eV}} = \begin{pmatrix} 0.5375 + 0.47302i & -2.71315 - 1.77586i & -2.77915 - 1.8634i \\ 0.37786 - 0.38321i & 0 & -0.62809 + 0.71107i \\ 0 & -0.89509 + 0.94085i & -0.57059 + 0.74175i \end{pmatrix},$$
$$s_{12}^2 = 0.3143, \quad s_{23}^2 = 0.4819, \quad s_{13}^2 = 0.02249, \quad \delta^{CP} = 187.35°, \quad J = -0.0043512,$$
$$\Delta m_{21}^2 = 7.588 \times 10^{-5}\ \text{eV}^2, \quad \Delta m_{31}^2 = 2.521 \times 10^{-3}\ \text{eV}^2, \quad m_\nu = 8.9570 \times 10^{-3}\ \text{eV}, \tag{33}$$

$$\frac{M_D^\nu}{10^{-2}\ \text{eV}} = \begin{pmatrix} -0.55823 - 0.53543i & 3.30143 - 0.56209i & 3.54834 - 0.61742i \\ -0.5523 + 0.05066i & -0.75365 - 0.12562i & 0 \\ 0 & -0.74915 - 0.24974i & 0 \end{pmatrix},$$
$$s_{12}^2 = 0.3074, \quad s_{23}^2 = 0.4875, \quad s_{13}^2 = 0.02249, \quad \delta^{CP} = 237.83°, \quad J = -0.02862,$$
$$\Delta m_{21}^2 = 7.49 \times 10^{-5}\ \text{eV}^2, \quad \Delta m_{31}^2 = 2.521 \times 10^{-3}\ \text{eV}^2, \quad m_\nu = 9.5179 \times 10^{-3}\ \text{eV}, \tag{34}$$

$$\frac{M_D^\nu}{10^{-2}\ \text{eV}} = \begin{pmatrix} 1.56882 + 0.0048i & 0 & 0 \\ 0 & 2.17863 + 0.01456i & 0 \\ -0.74382 - 0.05517i & 3.06548 + 0.02102i & 3.93054 + 0.02678i \end{pmatrix},$$
$$s_{12}^2 = 0.3188, \quad s_{23}^2 = 0.4708, \quad s_{13}^2 = 0.02248, \quad \delta^{CP} = 180.0°, \quad J = 0.0,$$
$$\Delta m_{21}^2 = 7.556 \times 10^{-5}\ \text{eV}^2, \quad \Delta m_{31}^2 = 2.521 \times 10^{-3}\ \text{eV}^2, \quad m_\nu = 1.7371 \times 10^{-2}\ \text{eV}. \tag{35}$$

3. Comparison of n-zero texture searches

The two methods discussed in Sec. III D to search for n-zero textures differ mainly in two ways, i.e., the selection of the initial parameters and the choice of loss function. In the case described in Sec. III D 1, we first randomly set parameter values in search of neutrino mass matrices where n elements have absolute values significantly smaller than the other $9 - n$. Then, we use a loss function that directly minimizes only those n elements. For the method of Sec. III D 2, we take the neutrino mass matrices as they come from the initialization and apply a loss function that counts the number of elements below and above some threshold. Then, we apply a second minimization to make the n smallest elements as close to 0 as possible. In both cases, we set conditions for the acceptable textures, such as having no rows or columns full of vanishing elements in the neutrino mass matrix.

The result of these discussions is that we see that for the most common cases, i.e., 1- and 2-zero textures, the sizes of the elements are roughly the same for both methods, with the number of surviving matrices having almost exactly the same proportions. Furthermore, in the case of 3-zero textures, we see that the search of Sec. III D 1 yields mostly matrices with four values smaller than the rest. This is not repeated in the case where we search for matrices using optimization, where the 3-zero texture cases all have

consistently three elements much smaller than the other six. Additionally, in the top rows of Figs. 9 and 12, we see that for 2-zero textures the three most common configurations are actually the same in both searches. When comparing the three most common configurations for 3-zero textures, shown in the second row of the same figures, we see that the most common 3-zero texture found in Sec. III D 1 matches the third most common 3-zero texture found in Sec. III D 2, while the other two configurations are different. Perhaps the most interesting difference is that the search using optimization actually yields usable 4-zero textures at a rate of 10 out of 50, while the random search yields no usable matrix out of 500, indicating that 4-zero textures are not easy to find from random searches. Interestingly, the identified 4-zero textures have the additional result that the Jarlskog invariant is predicted as $J = 0$ unlike the other textures found. One problematic point of the search using optimization is the inefficiency of performing first one optimization to identify the number of vanishing elements and then one more optimization to obtain a matrix where those elements are significantly smaller than the others. Lastly, we want to comment that the textures and their frequencies as determined in these searches have a dependence on the choice of weak basis. Therefore, we can claim that these results are meaningful only under the assumptions described at the beginning of this section. However, we expect this technique to be more informative in cases where additional theoretical details provide assumptions that are more consequential than the ones considered here.

IV. DISCUSSION

Up to this point, we have made a connection between ML techniques and elements in a simple SM extension to obtain Dirac neutrino mass matrices. While the purpose of this work is focused on methods and their applications as a proof of concept and demonstration, the processes detailed here can be adapted for further more specific discussions concerning different origins of neutrino masses. A simple example is already seen from the initialization itself, where differences between elements of the mass matrix elements already suggest a symmetry between μ-τ families. Moreover, there is an interesting tendency to have similarly valued M^{ν}_{D12} and M^{ν}_{D13} which is not repeated for M^{ν}_{D21} and M^{ν}_{D31}, and happens more commonly than any other differences. However, what may be the most interesting use of ML technique is the design of loss functions that could automatically use optimization to look for suitable cases. One example of this is the search of n-zero textures by counting (non)vanishing mass matrix elements. Finding specific forms of mass matrices could be used to inform the search of symmetries. One could start with a much more general setup, allowing many couplings, mass terms, and parameters, and then design a suitable loss function to find coincidences or unnecessary parameters. This could lead to a simpler model that is driven by current data. Additionally in our case, the parameters from the model have been included in the optimization. Therefore, optimization has given us information on suitable choices of parameters, as well as disfavored regions. It has also revealed parameters that make no difference and have no effect on the optimization process. This could be employed in the aforementioned point about deciding which parameters are really relevant for the model.

While this particular work has employed many simplifying assumptions that somewhat reduce the meaningfulness of physical interpretations, we believe that the technique presented here could benefit theoretical searches that are more purposeful. The methodology contained in this work can be a step toward more AI-inspired improvements. Namely, given the type of data generated here, we can envision future extensions where we could apply symbolic regression to infer more involved relationships between parameters or between mass elements, resulting in symbolic expressions that could inspire theoretical formulations, or at least simplify searches for suitable parameters via other numerical methods. Additionally, the parameters and particular forms of neutrino mass matrices that were obtained here can be used to train deep neural networks that would be able to explore the parameter space much faster and predict where certain forms of neutrino masses could be found. This would certainly be useful in models with a large number of parameters as it could learn the relationship between model parameters, predictions, and desirable features, allowing fast predictions of suitable parameters that would later be checked against the full calculation. We have left some of these opportunities as possible future extensions of this work which we plan to explore.

V. CONCLUSION

Neutrinos remain one of the most mysterious particles of the SM of particle physics. They are part of it, but their description remains incomplete. Numerous experiments indicate that neutrinos are massive and oscillate while they propagate. Additionally, future experiments promise to further constrain the involved observables. With no leading candidate to extend the SM to explain neutrino masses, one may wonder if by developing new methods to relate available results and proposed theories it could be possible to shed light on properties that may not be found using traditional methods. While we do not claim to have found any of these new properties, we have presented a method for the applications of ML techniques to the objects involved in BSM extensions for massive (Dirac) neutrinos. We have explained the role played by tools such as initialization, optimization, and the loss function, as well as presented different loss functions with the objective of exploring different results. This permitted us to explore theoretical variables and shapes of the neutrino

mass matrix that are allowed by constraints from neutrino data, namely, from oscillation parameters and neutrino mass scale. We started with a random baseline, which practically defined our proposed initialization. From there, we moved optimization of neutrino mass matrices using specific loss functions crafted to achieve some textures with different degrees of specificity. In some cases, we specified exactly which elements we wanted to be equal or wanted to vanish, but in another case we let the optimization itself decide how many and which parameters should vanish based on experimental constraints. We see this proposal as one step more in opening possible ways of thinking about the relationship between our observations and the theories we propose to explain them. This is thanks to the development of new methods and technologies aimed at letting computers explore different possibilities in automated ways, guided by objectives that have been decided by researchers. Furthermore, this can also be seen as one step toward methods that may one day employ actual artificial intelligence in the discovery of new and interesting relationships from data as we obtain from experiments.

ACKNOWLEDGMENTS

A. A. and A. S. acknowledge support from SNII-SECIHTI (Mexico). R. R. is supported by a KIAS Individual Grant (No. QP094602) via the Quantum Universe Center at Korea Institute for Advanced Study and is deeply grateful to Facultad de Ciencias, Universidad de Colima for its hospitality while performing part of this work. Computational resources used in this work have been supported by the Center for Advanced Computation at Korea Institute for Advanced Study.

DATA AVAILABILITY

There are no publicly available research data or software supporting this manuscript. Requests for further information or data should be sent to the authors.

[1] Particle Data Group, Review of particle physics, Phys. Rev. D **110**, 030001 (2024).
[2] A. K. Mann and H. Primakoff, Neutrino oscillations and the number of neutrino types, Phys. Rev. D **15**, 655 (1977).
[3] L. Wolfenstein, Oscillations among three neutrino types and *CP* violation, Phys. Rev. D **18**, 958 (1978).
[4] P. F. Harrison, D. H. Perkins, and W. G. Scott, Tri-bimaximal mixing and the neutrino oscillation data, Phys. Lett. B **530**, 167 (2002).
[5] P. F. Harrison and W. G. Scott, Symmetries and generalizations of tri—bimaximal neutrino mixing, Phys. Lett. B **535**, 163 (2002).
[6] Z.-z. Xing, Nearly tri bimaximal neutrino mixing and *CP* violation, Phys. Lett. B **533**, 85 (2002).
[7] X. G. He and A. Zee, Some simple mixing and mass matrices for neutrinos, Phys. Lett. B **560**, 87 (2003).
[8] Z.-z. Xing, H. Zhang, and S. Zhou, Nearly tri-bimaximal neutrino mixing and *CP* violation from mu-tau symmetry breaking, Phys. Lett. B **641**, 189 (2006).
[9] Daya Bay Collaboration, Observation of electron-antineutrino disappearance at Daya Bay, Phys. Rev. Lett. **108**, 171803 (2012).
[10] RENO Collaboration, Observation of reactor electron antineutrino disappearance in the RENO experiment, Phys. Rev. Lett. **108**, 191802 (2012).
[11] Double Chooz Collaboration, Improved measurements of the neutrino mixing angle θ_{13} with the Double Chooz detector, J. High Energy Phys. 10 (2014) 086.
[12] K. T. Matchev, K. Matcheva, P. Ramond, and S. Verner, Seeking truth and beauty in flavor physics with machine learning, in *37th Conference on Neural Information Processing Systems* (Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2023).
[13] K. T. Matchev, K. Matcheva, P. Ramond, and S. Verner, Exploring the truth and beauty of theory landscapes with machine learning, Phys. Lett. B **856**, 138941 (2024).
[14] S. Nishimura, C. Miyao, and H. Otsuka, Exploring the flavor structure of quarks and leptons with reinforcement learning, J. High Energy Phys. 12 (2020) 021.
[15] S. Nishimura, C. Miyao, and H. Otsuka, Reinforcement learning-based statistical search strategy for an axion model from flavor, J. High Energy Phys. 10 (2025) 043.
[16] S. Nishimura, H. Otsuka, and H. Uchiyama, Exploring the flavor structure of leptons via diffusion models, Phys. Rev. D **113**, 055030 (2026).
[17] S. Nishimura, H. Otsuka, and H. Uchiyama, Diffusion-model approach to flavor models: A case study for S_4' modular flavor model, Prog. Theor. Exp. Phys. **2026**, 053B08 (2026).
[18] A. Giarnetti and D. Meloni, Reinforcement learning techniques for the flavor problem in particle physics, Symmetry **18**, 131 (2026).
[19] Particle Data Group, Review of particle physics, Phys. Rev. D **98**, 030001 (2018).
[20] C. Jarlskog, Commutator of the quark mass matrices in the standard electroweak model and a measure of maximal *CP* nonconservation, Phys. Rev. Lett. **55**, 1039 (1985).
[21] D.-d. Wu, The pephasing invariants and *CP*, Phys. Rev. D **33**, 860 (1986).
[22] KATRIN Collaboration, Direct neutrino-mass measurement based on 259 days of KATRIN data, Science **388**, adq9592 (2025).

[23] I. Esteban, M. C. Gonzalez-Garcia, M. Maltoni, I. Martinez-Soler, J. P. Pinheiro, and T. Schwetz, NuFIT-6.0: Updated global analysis of three-flavor neutrino oscillations, J. High Energy Phys. 12 (2024) 216.
[24] NuFIT Collaboration, NuFIT v6.1, http://www.nu-fit.org/ (2025).
[25] T. Fukuyama and H. Nishiura, Mass matrix of Majorana neutrinos, arXiv:hep-ph/9702253.
[26] E. Ma and M. Raidal, Neutrino mass, muon anomalous magnetic moment, and lepton flavor nonconservation, Phys. Rev. Lett. **87**, 011802 (2001).
[27] K. R. S. Balaji, W. Grimus, and T. Schwetz, The solar LMA neutrino oscillation solution in the Zee model, Phys. Lett. B **508**, 301 (2001).
[28] C. S. Lam, A 2-3 symmetry in neutrino oscillations, Phys. Lett. B **507**, 214 (2001).
[29] P. F. Harrison and W. G. Scott, μ-τ reflection symmetry in lepton mixing and neutrino oscillations, Phys. Lett. B **547**, 219 (2002).
[30] Z.-z. Xing and Z.-h. Zhao, A review of μ-τ flavor symmetry in neutrino physics, Rep. Prog. Phys. **79**, 076201 (2016).
[31] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980.
[32] H. Fritzsch, Z.-z. Xing, and S. Zhou, Two-zero textures of the Majorana neutrino mass matrix and current experimental tests, J. High Energy Phys. 09 (2011) 083.
[33] D. Meloni, A. Meroni, and E. Peinado, Two-zero Majorana textures in the light of the Planck results, Phys. Rev. D **89**, 053009 (2014).
[34] T. P. Nguyen, Texture zeros of neutrino mass matrix with seesaw mechanism and leptogenesis, Mod. Phys. Lett. A **29**, 1450038 (2014).
[35] R. H. Benavides, D. V. Forero, L. Muñoz, J. M. Muñoz, A. Rico, and A. Tapia, Five texture zeros in the lepton sector and neutrino oscillations at DUNE, Phys. Rev. D **107**, 036008 (2023).
[36] G.-J. Ding, F. R. Joaquim, and J.-N. Lu, Texture-zero patterns of lepton mass matrices from modular symmetry, J. High Energy Phys. 03 (2023) 141.
[37] M. R. Devi, Retrieving texture zeros in $3+1$ active-sterile neutrino framework under the action of A_4 modular-invariants, arXiv:2303.04900.
[38] A. Rico, R. H. Benavides, D. V. Forero, L. Muñoz, and A. Tapia, Five texture zeros in the lepton sector that reproduce the observable neutrino masses and mixings: A mapping between the standard parameterization and our parameterization, Proc. Sci. ICRC2023 (**2023**) 1047.
[39] Z.-h. Zhao, Combinations of the μ-τ reflection symmetry and texture zeros in the Dirac neutrino mass matrix of the seesaw model, Eur. Phys. J. Plus **138**, 1055 (2023).
[40] S. Kumar and R. R. Gautam, Neutrino mass matrices with generalized *CP* symmetries and texture zeros, Nucl. Phys. **B1001**, 116520 (2024).
[41] S. Zhou, Texture zeros for lepton flavor mixing, Int. J. Mod. Phys. A **39**, 2441014 (2024).
[42] T. Nomura, H. Okada, and H. Otsuka, Texture zeros realization in a three-loop radiative neutrino mass model from modular A_4 symmetry, Nucl. Phys. **B1004**, 116579 (2024).
[43] A. Carrillo-Monteverde, S. Gómez-Ávila, and L. López-Lozano, 2-zeros texture and the universal texture constraint in the leptonic sector, Int. J. Mod. Phys. A **39**, 2450052 (2024).
[44] L. Singh, M. Kashav, and S. Verma, Implications of dark-θ_{12} solution on two-zero texture inverse neutrino mass matrix, Springer Proc. Phys. **304**, 675 (2024).
[45] I. A. Mazumder and R. Dutta, Bottom-up approach to texture zeros in the neutrino mass matrix, Int. J. Mod. Phys. A **41**, 2650010 (2026).
[46] T. Kobayashi, H. Otsuka, and M. Tanimoto, Yukawa textures from non-invertible symmetries, J. High Energy Phys. 12 (2024) 117.
[47] T. Kobayashi, Y. Nishioka, H. Otsuka, and M. Tanimoto, More about quark Yukawa textures from selection rules without group actions, J. High Energy Phys. 05 (2025) 177.
[48] T. Kobayashi, H. Otsuka, M. Tanimoto, and H. Uchida, Lepton mass textures from non-invertible multiplication rules, J. High Energy Phys. 08 (2025) 189.
[49] L. Calibbi, X. Gao, and M. Yuan, Hunting for neutrino texture zeros with muon and tau flavor violation, J. High Energy Phys. 05 (2026) 188.
[50] A. Biswas, S. Jangid, and S. C. Park, Investigating two-zero texture in the light of gauged type-II seesaw, Phys. Lett. B **868**, 139786 (2025).
[51] J. Gogoi and M. K. Das, A study of texture zeros using A_4 discrete symmetry group, J. Subatomic Part. Cosmol. **3**, 100071 (2025).
[52] Z. Jiang, B.-Y. Qu, and G.-J. Ding, Texture-zeros in minimal seesaw from noninvertible symmetry fusion rules, Phys. Rev. D **112**, 115029 (2025).
[53] D. Borah, P. Das, and D. Dutta, Neutrino texture-zeros after JUNO's first results: Implications for long-baseline neutrino experiments, arXiv:2512.13587.
[54] Priya, R. Kumar, L. Singh, and S. Verma, Implications of the first JUNO results for Dirac neutrino texture zeros, arXiv:2604.19122.