Open-source Software to Search for Polynomial-like Formulas for Fermion Masses

Kevin Loch[*]

(Dated: 2020/02/19)

An open-source program is described to semi-automatically search for polynomial-like non-linear equations that solve for the three charged lepton masses or any three user-supplied masses. The equations are designed to have three positive real roots corresponding to the three masses. This might explain the three generations of ordinary matter and give insight into the underlying physics of fermion Higgs field Yukawa couplings.

I. INTRODUCTION

Explanations for the structure and absolute values of the fermion mass spectrum and why there are exactly three generations of ordinary matter remain elusive. This paper describes a software program `nle-lepton`[1] that searches for polynomial-like non-linear equations (NLE's). The NLE's are designed to have three real positive roots representing three user-supplied "solution masses" (by default charged leptons). These empirical formulas could then be reviewed by physicists for possible relevance to the fermion mass generation process. If one of these formulas did represent the underlying physics then three generations of matter and the mass spectrum itself would be a consequence of its non-linear structure with three roots.

II. RUNNING MASSES

One problem with this or any empirical mass formula search is to select which masses to test with. Observed charged lepton masses are known to vary (or run) with energy scale due to vacuum polarization effects. Specific running masses depend on the chosen renormalization scheme and energy scale. By default `nle-lepton` uses the charged lepton rest masses but can be reconfigured to use other masses with the configuration file `nle-lepton.cfg`. A variety of approaches could be taken to select masses representing higher energy scales.

However, even the use of rest masses is not necessarily problematic. While simple relationships between masses are commonly expected to occur only at higher energy scales this is not an established fact. Additionally, some of the factors included in this search such as α_{em} and $\sin^2\theta_W$ also run with energy scale which might lead to an exact or approximate formula using rest masses. In any case `nle-lepton` is designed to be as flexible as possible and to find formulas that generate the user-supplied masses.

III. SEARCH SPACE

Proper polynomials have only positive integer exponents. Replicating the charged lepton mass spectrum with a polynomial would require large coefficients on some terms and a large variance between coefficients which this author considers unnatural. Using fractional exponents less than one on each term leads to much more "reasonable looking" coefficients. One possible physical explanation for fractional exponents would be if each term represented a length scale derived from a higher dimensional manifold, such as the radius of an n-ball or n-sphere.

As of version 4.0 `nle-lepton` searches for formulas with three mass terms and a constant term of -1,

$$C_1\left(\frac{M}{m_1}\right)^{\frac{1}{n_1}} - C_2\left(\frac{M}{m_2}\right)^{\frac{1}{n_2}} + C_3\left(\frac{M}{m_3}\right)^{\frac{1}{n_3}} - 1 = 0, \tag{1}$$

where solving for M gives the three solution masses, C_i are real coefficients, $\frac{1}{n_i}$ are exponents selected from the range $n = \{-26... - 1, 1...26\}$ (some string theories have up to 25 spatial dimensions), where $n_1 < n_2 < n_3$, and m_i are reference masses selected from the set $\{v, m_{\text{P}}, m_Z, m_W, m_{H^0}\}$. The reference masses can be the same or different for each term. Mass ratios are used inside each term so they are dimensionless as terms with different fractional exponent masses makes no sense physically. The range of exponents and set of allowed reference masses can be configured in `nle-lepton.cfg`. Support for different numbers of mass terms and/or integer exponents could be added in the future.

[*] kevinmloch@gmail.com

IV. PHASE 1

The formula search process is broken down into two main phases. In Phase 1 exponents are randomly selected and assigned to the terms of equation 1 in the correct order to generate three real positive roots. The function `solveNLEforCoefficients()` then solves equation 1 for coefficients C_i using computational (MC) methods. During this process three copies of equation 1 are used, `sm1_test`, `sm2_test`, `sm3_test`, one for each solution mass. For simplicity and speed the Higgs vacuum expectation value v is temporarily used as the reference mass m_i in all three terms of all three copies of the NLE so that

$$\texttt{sm1\_test} = C_1\left(\frac{M_{\text{sm1}}}{v}\right)^{\frac{1}{n_1}} - C_2\left(\frac{M_{\text{sm1}}}{v}\right)^{\frac{1}{n_2}} + C_3\left(\frac{M_{\text{sm1}}}{v}\right)^{\frac{1}{n_3}} - 1, \tag{2}$$

$$\texttt{sm2\_test} = C_1\left(\frac{M_{\text{sm2}}}{v}\right)^{\frac{1}{n_1}} - C_2\left(\frac{M_{\text{sm2}}}{v}\right)^{\frac{1}{n_2}} + C_3\left(\frac{M_{\text{sm2}}}{v}\right)^{\frac{1}{n_3}} - 1, \tag{3}$$

$$\texttt{sm3\_test} = C_1\left(\frac{M_{\text{sm3}}}{v}\right)^{\frac{1}{n_1}} - C_2\left(\frac{M_{\text{sm3}}}{v}\right)^{\frac{1}{n_2}} + C_3\left(\frac{M_{\text{sm3}}}{v}\right)^{\frac{1}{n_3}} - 1. \tag{4}$$

C_i are set to an initial value of 1.0 and processing successively refines C_i until each copy of the formula converges towards zero and

$$\texttt{precision} = |\texttt{sm1\_test}| + |\texttt{sm2\_test}| + |\texttt{sm3\_test}| < 1.0 \times 10^{-11}, \tag{5}$$

which should be sufficiently precise to give useful results.

V. FACTORING COEFFICIENTS AND ASSEMBLING FORMULAS FOR PHASE 2

Once the real coefficients C_i are found they are sent to function `cscanner()` which substitutes v for other reference masses (so that all intended combinations are tested) and searches for interesting factors for the coefficients. Possible factor combinations are sourced from "static" ingredients with exact or low uncertainty values, and "dynamic" ingredients with significant experimental uncertainty like $\sin^2\theta_W$. Two pre-computed tables of static ingredient combinations are used to accelerate this process, one for inside the radical (`infactors`) and one for outside the radical (`outfactors`). Values for dynamic ingredients and the solution masses are selected at random from within their experimental uncertainties each time phase 1 is run. Filtering of which coefficient factor combinations are considered a match is controlled by `phase1_filter` and `phase1_int_match_max` in `nle-lepton.cfg`. A match occurs when potential_factor_combination$*C_i$ is within $1.0\times10^{-\text{phase1\_filter}}$ of an integer between 1 and `phase1_int_match_max` inclusive. This factoring process allows for extremely rapid processing of billions of possible factor and reference mass combinations without having to solve the NLE for each combination.

Up to this point factoring of the coefficients has been done on each term independently of the others. When these terms are combined into complete formulas in phase 2 each unique combination of potential factors for each term needs to be tried and these combinations are assembled in function `verifyMatches()`. This often results in thousands of formulas to be processed from a single run of phase 1. To avoid processing uninteresting formulas, symmetry and complexity scores are assigned to each combination of terms with higher symmetry and lower complexity generally meaning a simpler and more interesting formula. The configuration options `phase2_symmetry_min` and `phase2_complexity_max` are provided to filter the formulas allowed to be passed to phase 2.

VI. PHASE 2

In phase 2 processing is reversed - formulas are constructed with the factors found in phase 1 and then solved for the three solution masses and/or other outputs in function `solveNLEforMasses()` using computational methods similar to phase 1. Before solving each proposed formula, the variables with experimental uncertainty are ranked by relative standard uncertainty and the three with the highest uncertainty are used as outputs (solved for) with the rest used as inputs. This allows for the lowest possible relative uncertainty in the outputs. Only results with all outputs within `phase2_results_window` (default=1.1) of experimental uncertainty are shown unless `phase2_results_always` is set to `yes`.

VII. SAMPLE RESULT

Ultimately the results from nle-lepton need to be reviewed by a physicist to determine any significance or usefulness (if any) of any result. `nle-lepton` can easily find tens of thousands of formulas which match the charged lepton rest masses and other ingredients to within their experimental uncertainties. Most of them can be quickly dismissed as "uninteresting" based on their complexity, or seemingly random mix of ingredients. An example of a formula the author found to be "somewhat interesting" is presented as a sample output:

$$\frac{\pi}{2}\left(\mathrm{R}_{18}\frac{1}{2\pi\alpha^2}\frac{m_W}{M}\right)^{\frac{1}{18}} - \frac{\pi}{\sqrt{2}}\left(\mathrm{R}_{10}\frac{3\pi}{2}\frac{m_{H^0}}{M}\right)^{\frac{1}{10}} + \pi\sqrt{\alpha}\left(\mathrm{R}_7\frac{1}{2\pi\alpha^2}\frac{m_W}{M}\right)^{\frac{1}{7}} - 1 = 0, \tag{6}$$

where M is solved for the three charged lepton rest masses, R_n is the the a geometric constant for the radius of an n-sphere with surface area S embedded in $n+1$ dimensional euclidean space such that the radius $r = (R_n S)^{\frac{1}{n}}$,

$$\mathrm{R}_{18} = \frac{34459425}{1024\pi^9}, \qquad \mathrm{R}_{10} = \frac{945}{64\pi^5}, \qquad \mathrm{R}_7\frac{3}{\pi^4}. \tag{7}$$

While it is not clear what significance if any equation 6 has to real-world physics it is a good example of the type of formulas that can be found with `nle-lepton` version 4.0.

VIII. INSTALLATION CONFIGURATION AND OPERATION

To install: unpack the source archive, change to the `src` directory and type `make`. Manually copy the binary executable `nle-lepton` to the desired executable directory and make a copy of `nle-lepton.cfg`. You can specify the location of the configuration file at run-time with the `-c` command line option. Help is available with the `-h` command line option, and an external random seed can optionally be set with the `-s` option. The file `README.txt` provides a summary of operation and output formatting.

An external user-editable configuration file `nle-lepton.cfg` is used to set operating modes, reference values, and coefficient factor parameters, some of which have already been described above. Each option has a comment describing its function and valid values so they will not be exhaustively repeated here. Once example of coefficient factor parameters is `outfactor_pi_exp_up_max` and `outfactor_pi_exp_down_max`. These control what factors of π are tested outside the radical of each term. Setting them to `outfactor_pi_exp_up_max` = 3, and `outfactor_pi_exp_down_max` = 2, would enable testing of π^{-3}, $\pi^{-\frac{3}{2}}$, π^{-2}, π^{-1}, $\pi^{-\frac{1}{2}}$, 1, π, $\pi^{\frac{1}{2}}$, π^2, π^3, $\pi^{\frac{3}{2}}$. Obviously increasing the ranges of any parameter increases the time it takes to scan coefficient factors. To disable a factor from being tested set `_up_max` = 0, and `_down_max` = 1 for that ingredient.

IX. CONCLUSION

A semi-automated approach to finding empirical mass formulas has advantages and disadvantages. Trillions of possible formulas can be tried and evaluated automatically to some extent but human analysis is ultimately required and a large volume of filtered results can make that process inefficient. More challenging is selecting the correct search parameters and formula structure to scan with. It is possible that NLE's are the wrong approach to this problem or that the right ingredients are not included yet. The author fully expects that further development will be required before a useful formula offering clues to the underlying physics of fermion mass generation is found, if ever. `nle-lepton` was designed to be as flexible as possible to adapt to different investigative approaches through configuration parameters, including user-defined input options, and changes to the source code to support additional formula structures and features. `nle-lepton` is released under a standard BSD 3-clause open source license for maximum flexibility by other developers.

[1] https://github.com/kevinloch/nle-lepton/releases