Semisimple unifications of any gauge theory

Andrew Gomes[*], Maximilian Ruhdorfer[†] and Joseph Tooby-Smith[‡]
Laboratory for Elementary Particle Physics, Cornell University, Ithaca, New York 14853, USA

(Received 14 July 2023; accepted 18 August 2023; published 2 October 2023)

We present a *Mathematica* package that takes any reductive gauge algebra and fully-reducible fermion representation, and outputs all semisimple gauge extensions under the condition that they have no additional fermions, and are free of local anomalies. These include all simple completions, also known as grand unified theories. We additionally provide a list of all semisimple completions for 5835 fermionic extensions of the one-generation Standard Model.

DOI: 10.1103/PhysRevD.108.075001

I. INTRODUCTION

Unification, the idea that the Standard Model (SM) gauge algebra $\mathfrak{su}(3) \oplus \mathfrak{su}(2) \oplus \mathfrak{u}(1)$ and particle representations embed into a semisimple algebra in the UV, is an appealing scenario for physics beyond the Standard Model (BSM). Theories based on semisimple algebras have phenomenological benefits over their reductive counterparts [those algebras containing $\mathfrak{u}(1)$ factors]. These include the simplicity of local anomaly cancellation, a possible origin for flavor symmetries, and the freedom from Landau poles.

Arguably the most famous of such extensions are grand unified theories (GUTs), a particularly elegant subclass where the unifying algebra is simple. These include the well-known $\mathfrak{su}(5)$ and $\mathfrak{so}(10)$ GUTs [1–3], which unify the fermion content of the SM—including right-handed neutrinos (RHNs) in the case of $\mathfrak{so}(10)$—into representations of $\mathfrak{su}(5)$ and $\mathfrak{so}(10)$, respectively. Another theoretically well-motivated subclass are semisimple extensions which mix flavor with gauge symmetries in a nontrivial way. Examples are those based on the Pati-Salam $\mathfrak{su}(4) \oplus \mathfrak{su}(2) \oplus \mathfrak{su}(2)$ model [4] such as "color-flavor" unification [e.g., $\mathfrak{su}(12) \oplus \mathfrak{su}(2) \oplus \mathfrak{su}(2)$ [5]] and "electroweak-flavor" unification [e.g., $\mathfrak{su}(4) \oplus \mathfrak{sp}(6) \oplus \mathfrak{sp}(6)$ [6]].

In the past, searching for semisimple extensions of a model was a tedious process that involved manual calculations and guesswork. The approach developed in [7], which we will refer to as "Flocci" (short for floccinaucinihilipilification), produced a complete list of the 340 possible anomaly-free semisimple extensions of the three generation Standard Model with right-handed neutrinos (SM_ν). However, Flocci was specifically tailored to the SM gauge algebra and fermion content and was not able to deal with different gauge algebras or a modified fermion content.

In this paper, we rectify this shortcoming and introduce "superfloccinaucinihilipilification" (or "SuperFlocci"), a *Mathematica* package that implements a more comprehensive and fundamentally different approach that can handle any reductive Lie algebra and any fully reducible anomaly-free fermion representation as an input. SuperFlocci outputs all semisimple anomaly-free extensions with a fixed number of fermions, including the maximal and minimal extensions and the branching patterns between them.

Like Flocci, the underlying workings of SuperFlocci rely heavily on the theory of Lie algebras, in particular the Refs. [8–13]. However, SuperFlocci uses a much more streamlined approach, building a tree of semisimple extensions starting from the input. Each edge of the tree corresponds to a maximal embedding of semisimple Lie algebras. Flocci, on the other hand, took a much more computationally expensive brute force approach that was difficult to generalize.

SuperFlocci has many applications, including the following:

(1) Determining the existence of a semisimple completion for a given theory.
(2) Finding possible flavor symmetries of a given fermion content and gauge algebra, whether horizontal or mixed with the input gauge algebra.
(3) Helping identify possible breaking patterns from a given semisimple completion down to the input algebra, through other semisimple completions.
(4) Determining the semisimple completions of $\mathfrak{u}(1)$ extensions of a theory.

[*]awg76@cornell.edu
[†]m.ruhdorfer@cornell.edu
[‡]j.tooby-smith@cornell.edu

(5) Scanning over theories of different fermion content, and searching for interesting semisimple extensions of each one.

It is worth noting that SuperFlocci only considers the fermionic sector; i.e., it cannot provide any insight into questions that require information about the scalar degrees of freedom. This includes the running of couplings, how a symmetry-breaking pattern is achieved through scalar vacuum expectation values, and the question of whether and at what scale the gauge couplings unify. However, the output of SuperFlocci is an exhaustive list of all possible unified models and can serve as a starting point for model building of a realistic unified theory.

The paper is organized as follows. We start with an explanation of how to download, run, and interpret the output of SuperFlocci in Sec. II. Section III describes formally the program's inputs and outputs. In Sec. IV we perform a number of checks, showing the agreement of SuperFlocci with previous results, and perform a scan over BSM theories, looking for interesting semisimple extensions. Section V provides an intuitive description of SuperFlocci's underlying algorithm with the help of a worked-out example. The Appendices contain our conventions for Dynkin diagrams and summaries of the results of the example runs of Sec. IV.

II. RUNNING THE CODE

We have designed SuperFlocci so that it is easy to use, with minimal prerequisites. It has been tested on *Mathematica* v12 and is available in the Supplemental Material [14].

The most up-to-date version of our code can be downloaded from GitHub at https://github.com/jstoobysmith/Superfloccinaucinihilipilification.

To install, download the file `SuperFlocci.m`, and import it into a *Mathematica* notebook using

```
Import["SuperFlocci.m"].
```

To find all semisimple extensions (within our criterion) we call the `SuperFlocci` function as

```
SuperFlocci[alg,rep],
```

which takes the required arguments:

`alg`: a reductive gauge algebra in the form of a list of simple or $\mathfrak{u}(1)$ factors labeled according to the Dynkin classification, e.g., `{A1, A2, U1}` for $\mathfrak{su}(2)\oplus\mathfrak{su}(3)\oplus\mathfrak{u}(1)$.

`rep`: a fermion representation in the form of a list of highest weights in the same order as the algebra, e.g., the highest weight of the $(\mathbf{2},\mathbf{3})_1$ representation is written as `{1, 1, 0, 1}`. Charges associated with $\mathfrak{u}(1)$ factors should be scaled to integers.

A list of optional arguments is given in Table I.

SuperFlocci uses Dynkin notation to denote simple Lie algebras. A conversion of the Dynkin notation for the classical Lie algebras to the more commonly used notation in physics, together with their syntax in SuperFlocci, is given in Table II. Our convention (though not essential in the use of SuperFlocci) will be to write algebras with simple factors of smaller rank first, and with $\mathfrak{u}(1)$ factors at the end. For example the SM gauge algebra is $\mathfrak{su}(2)\oplus\mathfrak{su}(3)\oplus\mathfrak{u}(1)$.

The weights follow the ordering convention of simple roots as indicated in the Dynkin diagrams of Appendix A. For an encyclopedic reference on the representations of Lie algebras and their expression in highest weight form, see, e.g., [13]. Here are some of the most common ones:

$$\begin{aligned}
\mathfrak{su}(2)\colon\ \mathbf{1}&=(0) & \mathfrak{su}(3)\colon\ \mathbf{1}&=(0,0)\\
\mathbf{2}&=(1) & \mathbf{3}&=(1,0)\\
\mathbf{3}&=(2) & \bar{\mathbf{3}}&=(0,1)\\
&& \mathbf{6}&=(2,0)\\
&& \bar{\mathbf{6}}&=(0,2)\\
&& \mathbf{8}&=(1,1). \qquad (1)
\end{aligned}$$

As an explicit example, consider the one-generation Standard Model with a right-handed neutrino (which we shall refer to as SM$_\nu$ throughout). This has the fermion representation $Q\oplus U\oplus D\oplus L\oplus E\oplus N$, where

TABLE I. The optional arguments of `SuperFlocci`.

Argument	Type (Default)	Description
`Checkpoint`	String (null)	The filename where checkpointed data should be saved. Can be used for long computations in case of a crash.
`CheckpointUpFreq`	Integer (1000)	How frequently (measured in nodes) checkpoints should be carried out in the upward growth phase.
`StartFromCheckpoint`	Boolean (True)	If a checkpoint filename is given, and the file exists, start from saved checkpoint.
`ClearDataFreq`	Integer (1000)	How often (measured in nodes) Mathematica cache should be cleared.
`SimpleIdealConstraint`	Integer (∞)	An upper limit on the number of simple ideals of nodes generated.
`DetailedProgressData`	Boolean (False)	Display detailed progress data (e.g., memory usage) while the code is running.
`ExtendedKappaCheck`	Boolean (True)	Determines whether or not pruning described in Sec. V C is carried out.

TABLE II. Notation for the classical Lie algebras, and $\mathfrak{u}(1)$, and their syntax in SuperFlocci.

Dynkin	Conventional	`SuperFlocci`
A_n	$\mathfrak{su}(n+1)$	`An`
B_n	$\mathfrak{so}(2n+1)$	`Bn`
C_n	$\mathfrak{sp}(2n)$	`Cn`
D_n	$\mathfrak{so}(2n)$	`Dn`
E_6, E_7, E_8	E_6, E_7, E_8	`E6, E7, E8`
F_4	F_4	`F4`
G_2	G_2	`G2`
	$\mathfrak{u}(1)$	`U1`

$$\begin{aligned} Q &= (\mathbf{2},\mathbf{3})_1, \\ U &= (\mathbf{1},\bar{\mathbf{3}})_{-4}, \\ D &= (\mathbf{1},\bar{\mathbf{3}})_2, \\ L &= (\mathbf{2},\mathbf{1})_{-3}, \\ E &= (\mathbf{1},\mathbf{1})_6, \\ N &= (\mathbf{1},\mathbf{1})_0. \end{aligned} \tag{2}$$

To run SuperFlocci on this example use

```
out = SuperFlocci[{A1, A2, U1}, {{1,1,0,1},
      {0,0,1, - 4},{0,0,1,2},{1,0,0, - 3},
      {0,0,0,6},{0,0,0,0}}].
```

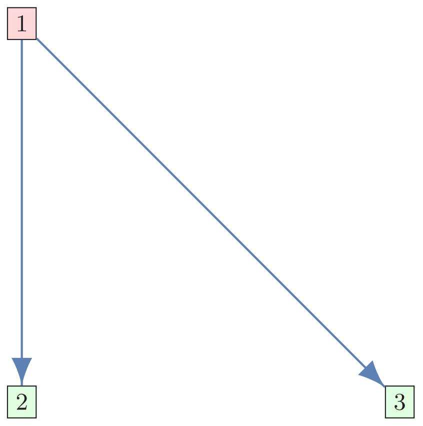
The highest weights here correspond, respectively, to the irreducible representations (irreps) in (2). On a standard laptop, this example takes only a few seconds to run. In comparison, the three generation SM$_\nu$ takes of order an hour, similar to the time taken by Flocci (although SuperFlocci provides more information).

All the information we might need is contained in the `out` variable. This includes the input algebra, the input representation, semisimple extensions (including projections $\Lambda\kappa$ to the input algebra), projection matrices $\Lambda\rho$ between extensions, and timing information. However, for ease of reading we can use the `FlocciOutput` function:

```
FlocciOutput[out].
```

This will display a graph (size permitting) of all semisimple extensions with edges representing embeddings, and lists of the maximal and minimal extensions (see Sec. III B for an explanation of these features). Figure 1 shows the output for this example.

The optional arguments of `FlocciOutput` are given in Table III. Suppose that, in addition to displaying the output, we want to save it to the file `output.m` and write LaTeX code to display all generated algebras and embeddings. We would accomplish this with



```
-------------------------------------
Maximal algebras:
1) so(10) (16)
-------------------------------------
Minimal algebras:
2) su(5) (1)⊕(5̄)⊕(10)
3) su(2)⊕su(2)⊕su(4) (1,2,4)⊕(2,1,4̄)
```

FIG. 1. The output of SuperFlocci, with the one-generation SM$_\nu$ as input.

```
FlocciOutput[out, save -> True,
filename -> "output", latex ->
{"alltables","embeddingsdetail"}].
```

Examples of the tables generated by the LaTeX output can be found in Appendix B. The embedding data comes in the form of the projection matrix $\Lambda\kappa$ to the input algebra and the projection matrices $\Lambda\rho$ to the maximal subalgebras (see Fig. 2). Applying these matrices to the weight system (generated by the highest weights) of a particular semisimple extension, one obtains the weight system of the input representation and the weight systems of all maximal subalgebras, respectively.

While SuperFlocci can in principle handle any input data conforming to the requirements of Sec. III, certain features of the input require more computation time or memory. Large input representations, vectorlike sectors in the input, and singlets are all factors that lead to more anomaly free combinations of irreps, and thus more extensions to be checked. Additionally, as the number of distinct charges assigned to a given weight (coming from different irreps that share a semisimple weight) increases, the program slows with a corresponding combinatoric factor. For example, we were able to run five generations of the $\mathfrak{su}(5)$ GUT fermion content $(\bar{\mathbf{5}} \oplus \mathbf{10})^{\oplus 5}$ on a standard laptop, but not the four generation SM. Finally, any computation involving the high-rank exceptional algebras will take longer due to their many automorphisms, which must be hard coded, in contrast with the automorphisms of high-rank classical Lie algebras.

TABLE III. The optional arguments of `FlocciOutput`.

Argument	Type (Default)	Description
`display`	Boolean (True)	Whether to display the output.
`save`	Boolean (False)	Whether to save the output to a file.
`filename`	String (null)	Where to save the output (without an extension—it will be saved as a .m file).
`latex`	String list (null)	Create a LaTeX file (with a default name) with customizable information given by the following options: "`maxtables`": table with all maximal algebras. "`mintables`": table with all minimal algebras. "`alltables`": tables with all algebras. "`embeddingsdetail`": detailed overview of all embeddings. "`tensorproduct`": (requires "`embeddingsdetail`"): gives all bilinear tensorproducts of representations. "`onlymaxminembeddings`": (requires "`embeddingsdetail`"): restricts "`embeddingsdetail`" to maximal and minimal embeddings. "`projectiontree`": (requires "`embeddingsdetail`"): outputs projection matrices between algebras on same branch of tree (is neglected for "`onlymaxminembeddings`").
`labelrep`	Rule list (null)	Replacement rules for highest weights to increase readability of LaTeX output. For example, use the rule list {`{1, 1, 0, 1}->"Q"`} to replace the left-handed quark representation with the letter Q.

III. DESCRIPTION OF THE INPUT AND OUTPUT

Colloquially, our program finds all semisimple extensions of a given input Lie algebra and fermion representation. In this section, we want to make this statement more precise by carefully describing the allowed inputs to the function `SuperFlocci`, and its output.

Throughout this section and Sec. V, we will assume basic knowledge of the theory of Lie algebras.[1] We note here that all Lie algebras throughout this paper are implicitly assumed to be complexified.

A. The input

The input of our program consists of

(i) A finite-dimensional reductive Lie algebra $\mathfrak{r}$.

(ii) The highest weights of a fully reducible, n-dimensional representation $r: \mathfrak{r} \to \mathfrak{su}(n)$.[2]

The condition of "full reducibility" here is to prevent representations of $\mathfrak{r}$ akin to the nondiagonalizable representation of $\mathfrak{u}(1)$ given by

$$\mathfrak{u}(1) \to \mathfrak{su}(2): \lambda \mapsto \begin{pmatrix} 0 & \lambda \\ 0 & 0 \end{pmatrix}. \tag{3}$$

We can split the reductive Lie algebra $\mathfrak{r}$ into the direct sum $\mathfrak{s} \oplus \mathfrak{a}$ of a semisimple part $\mathfrak{s}$ and an Abelian part $\mathfrak{a}$. Let $\mathfrak{h}_{\mathfrak{s}}$ be a chosen Cartan subalgebra of $\mathfrak{s}$ and $\mathfrak{h}_{\mathfrak{s}}^*$ its dual. The highest weights of r as inputted into `SuperFlocci` must be written with respect to a given basis, which we now describe. For $\mathfrak{h}_{\mathfrak{s}}^*$ we use the basis of fundamental weights with indices as indicated in the Dynkin diagrams of Appendix A.

We also need a basis for the dual of $\mathfrak{a}$, denoted $\mathfrak{a}^*$. To do this, we place the following condition on our input: A basis must exist of $\mathfrak{a}^*$, which when combined with the basis of $\mathfrak{h}_{\mathfrak{s}}^*$, leads to integer highest weights.

Any such basis can then be used as a valid basis of $\mathfrak{a}$. It is for this reason that the SM hypercharges in Eq. (2) have been scaled to be integers. This condition is satisfied if we assume that $\mathfrak{a}$ is the algebra of a compact Lie group [e.g., $U(1)$ rather than $\mathbb{R}$].

B. The output

We now explain the output of SuperFlocci. To do this consider triples $(\mathfrak{g}, \{\beta_i\}, \Lambda\kappa)$ where

(i) $\mathfrak{g}$ is a semisimple Lie algebra.

(ii) $\{\beta_i\}$ are the highest weights of a n-dimensional representation.

(iii) $\Lambda\kappa$: $\mathfrak{h}_{\mathfrak{g}}^* \to \mathfrak{h}_{\mathfrak{r}}^*$ is a linear transformation (represented by a matrix).

These are subject to the following conditions:

(1) $\Lambda\kappa$ must extend to a complete embedding of $\mathfrak{r}$ into $\mathfrak{g}$.

(2) The representation indicated $\{\beta_i\}$ must branch to r under $\Lambda\kappa$ (up to equivalence).

(3) The representation $\{\beta_i\}$ must be free of local anomalies.

We define an equivalence relation on our objects as follows. We say two triples $(\mathfrak{g}, \Lambda\kappa, \{\beta_i\})$ and $(\mathfrak{g}', \Lambda\kappa', \{\beta_i'\})$ are equivalent if there exists a linear map $\Lambda\rho: \mathfrak{h}_{\mathfrak{g}} \to \mathfrak{h}_{\mathfrak{g}'}$ such that there exists a full isomorphism $\rho: \mathfrak{g}' \to \mathfrak{g}$, $\Lambda\kappa = \Lambda\kappa'\Lambda\rho$, and on branching β becomes β'.

[1]Namely the notions of: semisimple and reductive Lie algebras, Cartan subalgebras, automorphisms, (highest) weights, and projections matrices. Those readers who need a refresher are directed to [15] and references therein.

[2]Note that as complexified Lie algebras $\mathfrak{gl}(n)$ and $\mathfrak{u}(n)$ are isomorphic. The condition of anomaly freeness ensures r factors through $\mathfrak{su}(n)$. One can also argue for $\mathfrak{u}(n)$ using the invariance of the kinetic term.

SuperFlocci outputs a directed acyclic graph, often termed a polytree. For brevity, we will refer to it interchangeably as a "tree" or "graph." The nodes of the graph are chosen representatives of each equivalence class mentioned above. An edge between representatives, $(\mathfrak{g}, \Lambda\kappa, \beta_i)$ to $(\mathfrak{g}', \Lambda\kappa', \beta_i')$, is a corresponding $\Lambda\rho$. However, here ρ can be any embedding, not just an isomorphism. We will only provide a minimal subset of edges. All remaining edges can be obtained through compositions, in the natural manner, within our subset or with isomorphisms. An example of a polytree generated in this way can be found in Fig. 4.

The sources of this graph (i.e., those nodes with no edges into them) are the maximal extensions, and the sinks of this graph (i.e., those nodes with no edges out of them) are the minimal extensions.

IV. CHECKS AND EXAMPLES

In this section, we demonstrate that SuperFlocci is able to reproduce and extend well-known results. We present the results of running SuperFlocci on a series of BSM models, we also summarize the results of a scan over possible extensions of the SM. The *Mathematica* output for each of the examples in the text can be found in the GitHub repository [16].

A. Checks

A number of obvious checks can be performed on SuperFlocci by using an input where part of the output is known beforehand. This could be, for example, from a previously known unifying theory. For these checks, we stress that the output will include much more than the sought-after unified theory. It will be a complete list of all possible semisimple extensions of the input gauge algebra and particle spectrum as well as the corresponding projection matrices and all branching patterns between them.

In what follows we go through the various checks of this sort we performed on SuperFlocci.

1. SM

It is well known that one generation of SM fermions fits into the $\bar{\mathbf{5}}$ and $\mathbf{10}$ representations of $SU(5)$ [1]. If instead we consider SM$_\nu$, then there are additional embeddings into $\mathfrak{su}(2) \oplus \mathfrak{su}(2) \oplus \mathfrak{su}(4)$ [4] and $\mathfrak{so}(10)$ [2,3]. In Sec. V we use SuperFlocci to demonstrate not only the existence of these extensions, but also that these are the only semisimple extensions.

For two and three generations of SM$_\nu$, Ref. [7] found that there are 45 and 340 different extensions with 9 and 24 of them being maximal, respectively. Despite using a different approach, SuperFlocci exactly reproduces these results and additionally finds the embeddings between these extensions.

We further ran SuperFlocci on the three-generation SM (without additional RHNs). In this case there are 19 embeddings with five of them being maximal (see Table V). These are all based on $\mathfrak{su}(5)$.

2. Non-SM gauge algebra

SuperFlocci is not limited to the SM gauge algebra $\mathfrak{su}(2) \oplus \mathfrak{su}(3) \oplus \mathfrak{u}(1)$ but can be used with any reductive Lie algebra. As a simple check we took the 19 semisimple extensions of the SM without RHNs as an input and added three singlets. An example would be to take as input the algebra $\mathfrak{su}(5)$ with the fermion representation $\bar{\mathbf{5}}^{\oplus 3} \oplus \mathbf{10}^{\oplus 3} \oplus \mathbf{1}^{\oplus 3}$. We checked that the result was a strict subset of the 340 semisimple extensions of SM$_\nu$.

3. Extending the SM fermion spectrum

Instead of choosing a different gauge algebra we can extend the fermion spectrum. As has been known for a long time, if one extends the one-generation SM by fermions in the vectorlike representations $(\mathbf{2},\mathbf{1})_3 \oplus (\mathbf{2},\mathbf{1})_{-3} \oplus (\mathbf{1},\mathbf{3})_{-2} \oplus (\mathbf{1},\bar{\mathbf{3}})_2 \oplus (\mathbf{1},\mathbf{1})_0^{\oplus 2}$, then the theory fits into the $\mathbf{27}$ representation of E_6 (see, e.g., [17,18]) and into the $\mathbf{15} \oplus \bar{\mathbf{6}} \oplus \bar{\mathbf{6}}$ representation of $\mathfrak{su}(6) \subset E_6$ (see, e.g., [19–21]).

We can use this information to cross-check SuperFlocci. Doing so, we find 68 semisimple extensions with 11 of them being maximal. These are shown in Table VII. The output contains as expected E_6 as a maximal and $\mathfrak{su}(6)$ as a nonmaximal extension.

4. $\mathfrak{u}(1)$ extensions

$\mathfrak{u}(1)$ extensions are important in BSM model building. If the $\mathfrak{u}(1)$ originates from a semisimple gauge algebra in the UV, nontrivial constraints on the model such as anomaly cancellation or the absence of Landau poles can be explained. One might therefore consider models with a semisimple UV completion better motivated from a theoretical point of view. Reference [22] found all of the $\mathfrak{u}(1)$ extensions of SM$_\nu$ that possess a semisimple UV completion. SuperFlocci complements these results by providing the ability to explicitly check the existence of a semisimple completion for $\mathfrak{u}(1)$ extension of any given theory.

Reproducing some of the results of [22] is an additional nontrivial check of SuperFlocci. As an example let us consider three generations of SM$_\nu$ with gauged baryon minus lepton number $B - L$ [23,24]. The resulting 14 maximal embeddings in Table VI agree with the ones in Table 4 of [22].[3] Analogously, the $\mathfrak{u}(1)$ extensions for which no semisimple completion was found in [22] can be checked in SuperFlocci and we find perfect agreement.

[3]Note that Table 4 of [22] gives only a subset of the 14 algebras that we find. In order to complete the list one has to take into account equivalence classes of planes. Once this is done the outputs fully agree.

TABLE IV. Example BSM models run through SuperFlocci. The fourth column shows the total number of semisimple extensions, the fifth column shows the number of maximal algebras, and the last column shows the number of minimal algebras. Here SM_ν indicates the fermionic content of the one-generation SM with a RHN.

	Model	Algebra	Representation	Total	Max	Min
1	SM plus 1 RNH	$\mathfrak{su}(2) \oplus \mathfrak{su}(3) \oplus \mathfrak{u}(1)$	$\mathrm{SM}_\nu^{\oplus 3} \oplus (\mathbf{1}, \mathbf{1}, 0)$	587	34	5
2	1-gen LR-symmetric	$\mathfrak{su}(2) \oplus \mathfrak{su}(2) \oplus \mathfrak{su}(3) \oplus \mathfrak{u}(1)$	$(\mathbf{2}, \mathbf{1}, \mathbf{3}, 1) \oplus (\mathbf{1}, \mathbf{2}, \bar{\mathbf{3}}, 1) \oplus (\mathbf{2}, \mathbf{1}, \mathbf{1}, -3) \oplus (\mathbf{1}, \mathbf{2}, \mathbf{1}, 3)$	2	1	1
3	LR-symmetric	$\mathfrak{su}(2) \oplus \mathfrak{su}(2) \oplus \mathfrak{su}(3) \oplus \mathfrak{u}(1)$	$[(\mathbf{2}, \mathbf{1}, \mathbf{3}, 1) \oplus (\mathbf{1}, \mathbf{2}, \bar{\mathbf{3}}, 1) \oplus (\mathbf{2}, \mathbf{1}, \mathbf{1}, -3) \oplus (\mathbf{1}, \mathbf{2}, \mathbf{1}, 3)]^{\oplus 3}$	173	12	1
4	4-gen $\mathfrak{su}(5)$	$\mathfrak{su}(5)$	$[\bar{\mathbf{5}} \oplus \mathbf{10}]^{\oplus 4}$	156	10	1
5	4-gen $\mathfrak{su}(5)$ with RHNs	$\mathfrak{su}(5)$	$[\bar{\mathbf{5}} \oplus \mathbf{10} \oplus \mathbf{1}]^{\oplus 4}$	3146	38	1
6	5-gen $\mathfrak{su}(5)$	$\mathfrak{su}(5)$	$[\bar{\mathbf{5}} \oplus \mathbf{10}]^{\oplus 5}$	567	20	1
7	5-gen $\mathfrak{su}(5)$ with RHNs	$\mathfrak{su}(5)$	$[\bar{\mathbf{5}} \oplus \mathbf{10} \oplus \mathbf{1}]^{\oplus 5}$	22,820	100	1
8	MSSM	$\mathfrak{su}(2) \oplus \mathfrak{su}(3) \oplus \mathfrak{u}(1)$	$\mathrm{SM}_\nu^{\oplus 3} \oplus (\mathbf{2}, \mathbf{1}, -3) \oplus (\mathbf{2}, \mathbf{1}, 3)$	1337	40	7
9	4-3-2-1	$\mathfrak{su}(2) \oplus \mathfrak{su}(3) \oplus \mathfrak{su}(4) \oplus \mathfrak{u}(1)$	$\mathrm{SM}_\nu^{\oplus 2} \oplus (\mathbf{2}, \mathbf{1}, \mathbf{4}, 0) \oplus (\mathbf{1}, \mathbf{1}, \bar{\mathbf{4}}, -3) \oplus (\mathbf{1}, \mathbf{1}, \mathbf{4}, 3)$	155	11	3
10	Trinification	$\mathfrak{su}(3) \oplus \mathfrak{su}(3) \oplus \mathfrak{su}(3)$	$(\mathbf{3}, \bar{\mathbf{3}}, \mathbf{1}) \oplus (\bar{\mathbf{3}}, \mathbf{3}, \mathbf{1}) \oplus (\mathbf{1}, \mathbf{3}, \bar{\mathbf{3}})$	2	1	1

B. BSM models

In addition to the above checks, we ran our code on a series of example BSM models. In each of these cases, novel semisimple extensions were found.

A summary of the results is presented in Table IV. We now discuss some notable features.

The addition of a singlet to the three-generation SM_ν is model 1 of the list.

Despite having a $\mathfrak{u}(1)$ present, the LR-symmetric models 2 and 3 both only have one minimal extension [25,26]. This corresponds to Pati-Salam. The only other extension in the case of model 2 is the $\mathfrak{so}(10)$ GUT.

Now compare models 4, 5, 6, and 7, which are based on the $\mathfrak{su}(5)$ GUT with a varying number of generations and number of RHNs. Note that adding a single RHN gives a bigger increase in the number of semisimple extensions than adding a new generation (comparing 5 and 6).

Model 8 in the list corresponds to the fermionic representations of the minimal-supersymmetric Standard Model with RHNs (ignoring gauginos), and model 9 to the 4-3-2-1 model [27,28]. Model 10, trinification [29,30], has two semisimple extensions corresponding to E_6 and $\mathfrak{su}(6)$, the former being maximal and the latter minimal.

C. Scans

Arguably the most physically relevant application of SuperFlocci is to find semisimple extensions of the SM gauge algebra with a fermion spectrum that goes beyond that of the SM. For this reason, we performed a scan over 5835 anomaly-free extensions of the SM_ν fermion spectrum and provide the results in [16], such that the user can simply look up the semisimple extensions of their favorite model. In the scan, we assumed one generation of SM fermions including a RHN and added up to five additional BSM particles in representations $(\mathbf{a}, \mathbf{b})_Y$ under $\mathfrak{su}(2) \oplus \mathfrak{su}(3) \oplus \mathfrak{u}(1)$, where $\mathbf{a} = \mathbf{1}, \mathbf{2}, \mathbf{3}$, $\mathbf{b} = \mathbf{1}, \mathbf{3}, \mathbf{6}, \mathbf{8}$ with their conjugate representations and $-6 \leq Y \leq 6$. We additionally restrict the total dimension of the additional representations to be less than 30.[4]

We note that we did not perform the scan for three generations of SM_ν fermions since already for the SM_ν fermion content it takes approximately one hour to run, compared to a few seconds for one generation. However, we note that the one-generation results can also be viewed as the "family universal" three-generation results.

The vast majority of maximal extensions have the same structure as the semisimple extensions for the SM_ν particle content; i.e., the SM gauge algebra is embedded within $\mathfrak{su}(5)$, $\mathfrak{su}(2) \oplus \mathfrak{su}(2) \oplus \mathfrak{su}(4)$, or $\mathfrak{so}(10)$. These often come with additional symmetries which account for the added fermions. However, there are a few exceptions. One of them is the E_6 model that we discussed in the previous section and which is shown in Table VII. Another is the occurrence of Pati-Salam type algebras of the form

$$\mathfrak{su}(2) \oplus \mathfrak{su}(2) \oplus \mathfrak{su}(k), \quad k \geq 4, \tag{4}$$

[4]Note that for 296 models we were not able to perform the pruning as described in Sec. V C due to insufficient computational memory and processing time. We still provide the results after the steps outlined in Sec. V B for all of these models in a separate file and note that the resulting list of extensions is a superset of all true extensions, i.e., no extension is missed. If the user wishes to skip the pruning step due to memory or time issues, this can easily be achieved by setting the option `ExtendedKappaCheck` in the `SuperFlocci` function to False (see Table I).

$$\begin{tikzcd}
 & \mathfrak{su}(n) \\
 & \mathfrak{g}' \arrow[u, "\beta'"'] \\
 & \mathfrak{g} \arrow[u, "\rho"] \\
\mathfrak{s} \arrow[r] \arrow[ur, "\alpha"] \arrow[uur, bend left, "\alpha'"] & \mathfrak{r} \arrow[u, "\kappa"] \arrow[uu, bend right, "\kappa'"']
\end{tikzcd}
\qquad
\begin{tikzcd}
 & (\mathfrak{g}', \beta') \arrow[d, "\Lambda\rho"'] \arrow[dl, bend right, "\Lambda\alpha'"'] \arrow[dd, bend left, "\Lambda\kappa'"] \\
 & (\mathfrak{g}, \beta) \arrow[d, "\Lambda\kappa"'] \arrow[dl, "\Lambda\alpha"'] \\
(\mathfrak{s}, r|_{\mathfrak{s}}) & (\mathfrak{r}, r) \arrow[l]
\end{tikzcd}$$

FIG. 2. A diagram summarizing the projection matrices used within this paper (right), and their corresponding embeddings (left).

with the fermions in the $(\mathbf{2}, \mathbf{1}, \mathbf{k}) \oplus (\mathbf{1}, \mathbf{2}, \bar{\mathbf{k}})$ representation. These are only a few examples and the full result can be found in the GitHub repository.

V. AN EXAMPLE CALCULATION: THE ONE-GENERATION SM

In this section, we will explain the algorithm by working out an explicit example. We begin by specifying as an input (1) a finite-dimensional reductive Lie algebra $\mathfrak{r}$ (the "gauge algebra") and (2) the highest weights of a fully reducible, n-dimensional representation $r: \mathfrak{r} \to \mathfrak{su}(n)$ (the "fermionic representation"). The Lie algebra $\mathfrak{r}$ can be decomposed into its semisimple and Abelian parts as $\mathfrak{r} = \mathfrak{s} \oplus \mathfrak{a}$. As our example, we will use the one-generation SM$_\nu$, that is, $\mathfrak{r} = \mathfrak{su}(2) \oplus \mathfrak{su}(3) \oplus \mathfrak{u}(1)$ and the highest weights of r are $Q \oplus U \oplus D \oplus L \oplus E \oplus N$, where these irreps[5] were defined in Eq. (2). The dimension n of this representation is 16. Note that this was also the example used in Sec. II.

As discussed in Sec. III B, the object that we are trying to construct is a polytree, i.e., a directed acyclic graph consisting of vertices (or nodes) and edges (see Fig. 3 for an example). Every node consists of a semisimple Lie algebra $\mathfrak{g}$, a representation β of that algebra, and a projection matrix $\Lambda\kappa$ that encodes both the embedding of $\mathfrak{r}$ into $\mathfrak{g}$ (up to equivalence) and how β branches down into r. This data is shown in Fig. 2. Every edge (which we direct downwards) connects two nodes, which we may call the parent (upper) and child (lower) node, and consists of a projection matrix $\Lambda\rho$. This projection matrix loosely describes a maximal embedding of the child's algebra into the parent's algebra, and tells one how the parent's representation branches into the child's representation. That the embedding is maximal means that no other semisimple subalgebra of the parent's algebra contains the child's algebra. Finally, note that the graph is acyclic because the property of being a subalgebra is transitive.

At the base of the tree (the sinks) lie the minimal nodes (or "minimal semisimple extensions"). If the input algebra is semisimple then the input node will be in the tree, and the sole minimal node. Since all of the nodes in the tree must have algebras which are subalgebras of $\mathfrak{su}(n)$, there must be maximal nodes. Those nodes that do not branch from any others in the tree are called maximal, and occupy the top nodes of the tree (sources of the tree).

[5]Strictly, these are equivalence classes of irreps, but we will simply refer to them as irreps.

The construction of the tree consists of three phases. First, an auxiliary tree is grown upward from the input algebra. The nodes of the auxiliary tree differ from those of the final tree in that $\Lambda\kappa$ is replaced with a projection matrix $\Lambda\alpha$ for only $\mathfrak{s}$, the semisimple part of $\mathfrak{r}$ (see Fig. 2). If the input's algebra is semisimple then we are done at this point. Second, the final tree is grafted onto the scaffolding of the auxiliary tree, and the projection matrices $\Lambda\kappa$ are constructed. Then, a final upward pruning is required to ensure that the $\Lambda\kappa$ translate into full algebra embeddings.

A. Growing the auxiliary tree

The algorithm generates the tree recursively, by creating all possible parent nodes for each existing node (we can think of $\mathfrak{s}$ and $r|_{\mathfrak{s}}$ as forming the base node of the auxiliary tree). These parent nodes then become child nodes in some future iteration. First, for a given (child) node, a list of minimal superalgebras (specified by an algebra and projection matrix $\Lambda\rho$) of the child's algebra is produced. Note that the same algebra may appear multiple times with different projection matrices.

In our example $\mathfrak{s} = \mathfrak{su}(2) \oplus \mathfrak{su}(3)$. We will do the first step, and thus $\mathfrak{g} = \mathfrak{s}$. There are 10 minimal superalgebras of $\mathfrak{g}$, with 8 distinct algebras $\mathfrak{g}'$:

$$\begin{array}{ll}
\mathfrak{su}(2)^{\oplus 2} \oplus \mathfrak{su}(3) & \mathfrak{su}(2) \oplus \mathfrak{su}(3)^{\oplus 2} \\
\mathfrak{su}(3) \oplus \mathfrak{su}(3) & \mathfrak{su}(3) \oplus \mathfrak{sp}(4) \\
\mathfrak{su}(2) \oplus \mathfrak{su}(4) & \mathfrak{su}(2) \oplus \mathfrak{su}(6) \\
\mathfrak{su}(5) & \mathfrak{su}(6).
\end{array}$$

As an example of an algebra with more than one projection matrix to the base, take the algebra $\mathfrak{su}(2)^{\oplus 2} \oplus \mathfrak{su}(3)$. There is an embedding where the SM $\mathfrak{su}(2)$ embeds via the identity into a single parent $\mathfrak{su}(2)$ (say, the second), and an embedding where the SM $\mathfrak{su}(2)$ embeds diagonally into both parent $\mathfrak{su}(2)$ factors. Their projection matrices are given, respectively, by

$$\Lambda\rho_1 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad \Lambda\rho_2 = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{5}$$

Note how the matrix heights and widths reflect the ranks of the child's (in this case $\mathfrak{g} = \mathfrak{s}$ with rank 3) and parent's [in this case $\mathfrak{su}(2)^{\oplus 2} \oplus \mathfrak{su}(3)$ with rank 4] algebras, respectively. This makes sense since the projection matrices map the parent weight system to the child weight system. The rows and columns of projection matrices are ordered according to our convention. For example, the first two columns of the matrices in Eq. (5) correspond to the parent $\boldsymbol{\mathfrak{su}}(2)$ factors, while the last two correspond to the $\boldsymbol{\mathfrak{su}}(3)$ factor.

For each minimal superalgebra, there are seven steps to find the representations that will form parent nodes. Each candidate may lead to zero, one, or several representations. We will follow the steps of the algorithm for the minimal superalgebra consisting of $\mathfrak{g}' = \boldsymbol{\mathfrak{su}}(2)^{\oplus 2} \oplus \boldsymbol{\mathfrak{su}}(3)$ and $\Lambda\rho_1$. The steps are as follows[6]:

U1: Generate all irreps (labeled by their unique highest weight) of the minimal superalgebra $\mathfrak{g}'$ with dimension less than or equal to n (in our case 16). There are 85 such irreps.

U2: Keep only those highest weights whose projection under $\Lambda\rho$ is present in the weight system of the child representation.[7] There are now 31 such irreps.

U3: Now keep only those irreps whose entire weight system, once projected down, is contained within the child weight system (taking weight multiplicities into account). There are now six such irreps: $(\mathbf{1},\mathbf{1},\mathbf{1})$, $(\mathbf{2},\mathbf{1},\mathbf{1})$, $(\mathbf{1},\mathbf{2},\mathbf{1})$, $(\mathbf{1},\mathbf{1},\bar{\mathbf{3}})$, $(\mathbf{2},\mathbf{1},\bar{\mathbf{3}})$, and $(\mathbf{1},\mathbf{2},\mathbf{3})$. Note that we included the singlet.

U4: From the remaining irreps, construct all representations of dimension exactly n that project down to exactly the child weight system[8] and satisfy anomaly cancellation (see, e.g., [31]). For our case, there are three such representations:

$$
\begin{aligned}
&(\mathbf{1},\mathbf{2},\mathbf{1}) \oplus (\mathbf{1},\mathbf{2},\mathbf{3}) \oplus (\mathbf{2},\mathbf{1},\mathbf{1}) \oplus (\mathbf{2},\mathbf{1},\bar{\mathbf{3}}),\\
&(\mathbf{1},\mathbf{2},\mathbf{1}) \oplus (\mathbf{1},\mathbf{2},\mathbf{3}) \oplus (\mathbf{2},\mathbf{1},\mathbf{1}) \oplus (\mathbf{1},\mathbf{1},\bar{\mathbf{3}})^{\oplus 2},\\
&(\mathbf{1},\mathbf{2},\mathbf{1}) \oplus (\mathbf{1},\mathbf{2},\mathbf{3}) \oplus (\mathbf{1},\mathbf{1},\mathbf{1})^{\oplus 2} \oplus (\mathbf{2},\mathbf{1},\bar{\mathbf{3}}).
\end{aligned}
$$

One immediately sees that each representation is composed of irreps that branch down to the left-handed leptons, left-handed quarks, right-handed leptons, and right-handed quarks, respectively.

U5: For each remaining n-dimensional representation, check whether it is possible to assign charges under $\mathfrak{a}$ to each $\mathfrak{g}'$ weight such that (1) under projection the charges assigned to the $\mathfrak{s}$ weights correspond exactly to the $\mathfrak{r}$ weights of the input representation, and (2) the assignments are $\mathfrak{h}_{\mathfrak{g}'} \oplus \mathfrak{a}$ anomaly free. Of the three representations in the last step, only the first passes this test.

U6: For each remaining n-dimensional representation, check that the weights of each irrep can be grouped into $\mathfrak{s}$-irreps with constant $\mathfrak{a}$ charge.

U7: Finally, the automorphisms of the minimal superalgebra are used to put the remaining representations which we now denote β' and projection matrices $\Lambda\rho$ and $\Lambda\alpha' = \Lambda\alpha \circ \Lambda\rho$ into a unique form. This step is ultimately related to finding a representative of the equivalence classes representing nodes, as discussed in Sec. III B.

$$\begin{tikzcd}
 & \mathfrak{so}(10) \arrow[d] \arrow[dr] & \\
 & \mathfrak{su}(2)^{\oplus 2} \oplus \mathfrak{su}(4) \arrow[dl] \arrow[d] & \mathfrak{su}(5) \arrow[ddl] \\
\mathfrak{su}(2)^{\oplus 2} \oplus \mathfrak{su}(3) \arrow[dr] & \mathfrak{su}(2) \oplus \mathfrak{su}(4) \arrow[d] & \\
 & \mathfrak{su}(2) \oplus \mathfrak{su}(3) &
\end{tikzcd}$$

FIG. 3. The auxiliary tree of the one-generation SM_ν.

If a node, specified by $(\mathfrak{g}', \beta', \Lambda\alpha')$ is not already present in the tree, it is added. However, the node may already be present. It is for this reason that the object we are constructing is not technically a tree, but rather a polytree, since branches can join back up with others.

Once no more nodes can be created, the auxiliary graph is complete. The graph for our example at this stage of the algorithm is given in Fig. 3. If the input algebra is semisimple, this is the full graph and we stop.

B. Grafting the final tree

We must now produce a new graph from those nodes of the auxiliary graph, in which $\Lambda\alpha$ extends to embeddings of the full (reductive) input algebra into $\mathfrak{g}$. Note that for a single semisimple embedding of the auxiliary tree, there may be multiple reductive embeddings. The algorithm begins at the top of the auxiliary tree and works downwards, with four steps being applied at each node.

[6]Strictly speaking, **U5** and **U6** are not required to form the auxiliary graph containing simply the embeddings of $\mathfrak{s}$, since they involve the charges of r under $\mathfrak{a}$. However, if a node in the auxiliary tree fails to satisfy either of these criteria, the corresponding final tree nodes and their parents will not fulfil them either. Consequently, we greatly reduce computation time by adding these steps to the upward tree generation.

[7]Note that this step serves only to speed up the algorithm since the same irrep eliminations would be performed by **U3**.

[8]Rather than being a set, the weight system is a multiset: the number of instances of each weight is important. The projection must not only produce the correct weights but also the correct multiplicity of each weight.

- **D1:** The algorithm attempts to extends $\Lambda\alpha$ to a $\Lambda\kappa$ by adding extra rows for each $\mathfrak{u}(1)$ factor in the input algebra. The entries of these new rows are constrained so that β projects to representations with the correct input $\mathfrak{u}(1)$ charges. There may be zero, one, or several solutions.
- **D2:** It now checks which solutions are related by automorphisms of $\mathfrak{g}$, and creates one node in the final tree for each equivalence class (see Sec. III B).
- **D3:** If there are no solutions, it stops working on the current node of the auxiliary tree, and will skip all nodes below it in the auxiliary tree, which are guaranteed to have no solutions.
- **D4:** For each newly created node (if any) in the final tree, check for all parents p that $\Lambda\kappa_p = \Lambda\kappa \circ \Lambda\rho$, up to automorphism (see Sec. III B). If true, create an edge between them.

As an example, take the maximal algebra from the auxiliary tree, $\mathfrak{so}(10)$. The steps of the last section yield

$$\Lambda\alpha = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{6}$$

One can check that the projection of the weight system given by the highest weight (0, 0, 0, 1, 0) (the **16** irrep) yields the semisimple weights of the input representation. The full weights of the input representation [including $\mathfrak{u}(1)$ charges] can be obtained by extending $\Lambda\alpha$ with one of the two following rows:

$$(3, 6, 8, 2, 4),$$
$$(-3, -6, -4, -4, -2).$$

These two choices turn out to be related by automorphism, so one node in the final tree is created.

Once all nodes of the auxiliary tree have been processed, one has the final tree which "sits on top" of the auxiliary tree. In general, some regions of the auxiliary tree will be absent in the final tree. At the same time, some auxiliary nodes will correspond to many final tree nodes. The graph at this stage is given in Fig. 4.

$$\begin{tikzcd}
 & \mathfrak{so}(10) \arrow[dl] \arrow[dr] & \\
\mathfrak{su}(2)^{\oplus 2} \oplus \mathfrak{su}(4) & & \mathfrak{su}(5)
\end{tikzcd}$$

FIG. 4. The final tree of the one-generation SM_ν. Notice the presence of the usual SM_ν extensions: $\mathfrak{so}(10)$, Georgi-Glashow, and Pati-Salam. Their representations are **16**, $\mathbf{10} \oplus \bar{\mathbf{5}} \oplus \mathbf{1}$, and $(\mathbf{2}, \mathbf{1}, \mathbf{4}) \oplus (\mathbf{1}, \mathbf{2}, \bar{\mathbf{4}})$, respectively.

C. Pruning the final tree

The final step is to ensure the projection matrix implies the existence of an embedding (this is one of the conditions for a node in Sec. III B). This is true for semisimple algebras, but not in general for reductive algebras, though exceptions are not common. An exemplary input representation for the SM gauge algebra which produces projection matrices that do not correspond to an embedding is the one-generation SM_ν with additional vectorlike fermions in the $(\mathbf{1}, \mathbf{8})_4 \oplus (\mathbf{1}, \mathbf{8})_{-4}$ representation.

The problem can arise when one attempts to extend the embedding of the Cartan subalgebra (given by the projection matrix) to an embedding of the full algebra. One parametrizes the embeddings of the non-Cartan generators with unknown parameters. Since commutation relations in the image must be satisfied, this gives a set of polynomial equations that is quadratic in these parameters. We do not actually need a solution, rather we simply need to know if a solution exists. We use a Gröbner basis analysis to carry this out.

All of the projection matrices in our example can be extended to full reductive embeddings. Therefore, the final tree takes the form of Fig. 4.

VI. CONCLUSION

SuperFlocci is a versatile and user-friendly program to find all semisimple extensions to theories of arbitrary gauge algebra and (local) anomaly-free fermion representation. The program returns not only all semisimple extensions, but also detailed information about the embeddings of the input algebra into these extensions, the embeddings of extensions into each other, and the branching structure of their fermion representations. As an example application of the program, we performed a scan of 5835 extensions to the fermion content of the one-generation SM_ν. Beyond extending the fermion content of the SM, we believe SuperFlocci will be useful for those searching for GUTs of BSM theories with an extended gauge algebra.

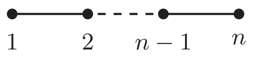
ACKNOWLEDGMENTS

We thank Ben Allanach, Joe Davighi, Mijo Ghosh, Ben Gripaios, Alessandro Podo, and Eric San for helpful discussions. The authors are supported in part by the NSF Grant No. PHY-2014071. A. G. is also supported by the NSERC PGS-D fellowship. M. R. is also supported by a Feodor-Lynen Research fellowship awarded by the Humboldt Foundation.

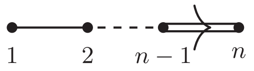
APPENDIX A: DYNKIN DIAGRAM CONVENTION

We follow the same convention in labelling our simple roots as in [13] (which is different from Flocci [7]). As mentioned within the main text, all weights used within our program should be written within this convention.

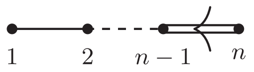
Dynkin diagram for A_n:



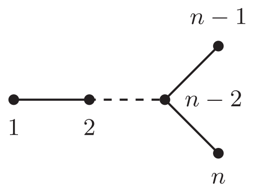
Dynkin diagram for B_n:



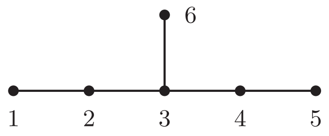
Dynkin diagram for C_n:



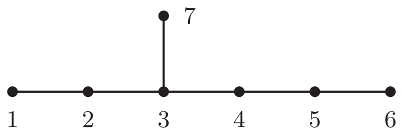
Dynkin diagram for D_n:



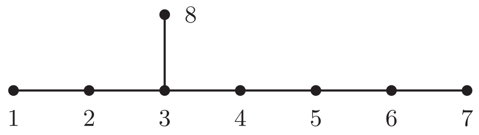
Dynkin diagram for E_6:



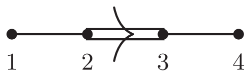
Dynkin diagram for E_7:



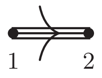
Dynkin diagram for E_8:



Dynkin diagram for F_4:



Dynkin diagram for G_2:



APPENDIX B: TABLES

This appendix contains tables generated by SuperFlocci for inputs specified in Sec. IV.

TABLE V. All maximal and minimal algebras for three generations of SM fermions without RHNs. The gauge algebra is $\mathfrak{su}(2) \oplus \mathfrak{su}(3) \oplus \mathfrak{u}(1)$ and the input representation $[(\mathbf{2},\mathbf{3},1) \oplus (\mathbf{1},\bar{\mathbf{3}},-4) \oplus (\mathbf{1},\bar{\mathbf{3}},2) \oplus (\mathbf{2},\mathbf{1},-3) \oplus (\mathbf{1},\mathbf{1},6)]^{\oplus 3}$.

		Maximal	
	Algebra	Fermion representations corresponding to β
1	$\mathfrak{su}(2)^{\oplus 2} \oplus \mathfrak{su}(5)$	$(\mathbf{1},\mathbf{3},\mathbf{10}) \oplus (\mathbf{3},\mathbf{1},\bar{\mathbf{5}})$
2	$\mathfrak{su}(2)^{\oplus 2} \oplus \mathfrak{su}(5)$	$(\mathbf{1},\mathbf{1},\mathbf{5}) \oplus (\mathbf{1},\mathbf{2},\mathbf{5}) \oplus (\mathbf{3},\mathbf{1},\overline{\mathbf{10}})$
3	$\mathfrak{su}(2)^{\oplus 2} \oplus \mathfrak{su}(5)$	$(\mathbf{1},\mathbf{1},\mathbf{10}) \oplus (\mathbf{1},\mathbf{2},\mathbf{10}) \oplus (\mathbf{3},\mathbf{1},\bar{\mathbf{5}})$
4	$\mathfrak{su}(5)^{\oplus 3}$	$(\mathbf{1},\mathbf{1},\bar{\mathbf{5}}) \oplus (\mathbf{1},\mathbf{1},\mathbf{10}) \oplus (\mathbf{1},\bar{\mathbf{5}},\mathbf{1}) \oplus (\mathbf{1},\mathbf{10},\mathbf{1}) \oplus (\bar{\mathbf{5}},\mathbf{1},\mathbf{1}) \oplus (\mathbf{10},\mathbf{1},\mathbf{1})$
5	$\mathfrak{su}(2)^{\oplus 2} \oplus \mathfrak{su}(5)^{\oplus 2}$	$(\mathbf{1},\mathbf{1},\mathbf{1},\bar{\mathbf{5}}) \oplus (\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{10}) \oplus (\mathbf{1},\mathbf{2},\mathbf{10},\mathbf{1}) \oplus (\mathbf{2},\mathbf{1},\bar{\mathbf{5}},\mathbf{1})$
		Minimal
6	$\mathfrak{su}(5)$	$\bar{\mathbf{5}}^{\oplus 3} \oplus \mathbf{10}^{\oplus 3}$

TABLE VI. All maximal and minimal algebras for three generations of SM$_\nu$ extended by a gauged $\mathfrak{u}(1)_{B-L}$. The gauge algebra is $\mathfrak{su}(2)\oplus\mathfrak{su}(3)\oplus\mathfrak{u}(1)\oplus\mathfrak{u}(1)$ and the input representation $[(\mathbf{2},\mathbf{3},1,1)\oplus(\mathbf{1},\bar{\mathbf{3}},-4,-1)\oplus(\mathbf{1},\bar{\mathbf{3}},2,-1)\oplus(\mathbf{2},\mathbf{1},-3,-3)\oplus(\mathbf{1},\mathbf{1},6,3)\oplus(\mathbf{1},\mathbf{1},0,3)]^{\oplus 3}$.

	Maximal	
	Algebra	Fermion representations corresponding to β
1	$\mathfrak{su}(2)\oplus\mathfrak{so}(10)$	$(\mathbf{3},\mathbf{16})$
2	$\mathfrak{su}(2)\oplus\mathfrak{so}(10)^{\oplus 2}$	$(\mathbf{1},\mathbf{1},\mathbf{16})\oplus(\mathbf{2},\mathbf{16},\mathbf{1})$
3	$\mathfrak{so}(10)^{\oplus 3}$	$(\mathbf{1},\mathbf{1},\mathbf{16})\oplus(\mathbf{1},\mathbf{16},\mathbf{1})\oplus(\mathbf{16},\mathbf{1},\mathbf{1})$
4	$\mathfrak{su}(4)\oplus\mathfrak{sp}(6)^{\oplus 2}$	$(\bar{\mathbf{4}},\mathbf{6},\mathbf{1})\oplus(\mathbf{4},\mathbf{1},\mathbf{6})$
5	$\mathfrak{su}(2)^{\oplus 2}\oplus\mathfrak{su}(12)$	$(\mathbf{1},\mathbf{2},\mathbf{12})\oplus(\mathbf{2},\mathbf{1},\overline{\mathbf{12}})$
6	$\mathfrak{su}(2)^{\oplus 3}\oplus\mathfrak{su}(4)\oplus\mathfrak{sp}(6)$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{4},\mathbf{6})\oplus(\mathbf{1},\mathbf{1},\mathbf{2},\bar{\mathbf{4}},\mathbf{1})\oplus(\mathbf{2},\mathbf{2},\mathbf{1},\bar{\mathbf{4}},\mathbf{1})$
7	$\mathfrak{su}(2)^{\oplus 3}\oplus\mathfrak{su}(4)^{\oplus 2}$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{4},\mathbf{6})\oplus(\mathbf{1},\mathbf{1},\mathbf{2},\bar{\mathbf{4}},\mathbf{1})\oplus(\mathbf{2},\mathbf{2},\mathbf{1},\bar{\mathbf{4}},\mathbf{1})$
8	$\mathfrak{su}(4)^{\oplus 2}\oplus\mathfrak{sp}(6)$	$(\bar{\mathbf{4}},\mathbf{6},\mathbf{1})\oplus(\mathbf{4},\mathbf{1},\mathbf{6})$
9	$\mathfrak{su}(2)^{\oplus 3}\oplus\mathfrak{su}(4)\oplus\mathfrak{sp}(6)$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{4},\mathbf{6})\oplus(\mathbf{1},\mathbf{1},\mathbf{2},\bar{\mathbf{4}},\mathbf{1})\oplus(\mathbf{2},\mathbf{2},\mathbf{1},\bar{\mathbf{4}},\mathbf{1})$
10	$\mathfrak{su}(2)^{\oplus 2}\oplus\mathfrak{su}(4)\oplus\mathfrak{sp}(4)\oplus\mathfrak{so}(10)$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{16})\oplus(\mathbf{1},\mathbf{1},\mathbf{4},\mathbf{4},\mathbf{1})\oplus(\mathbf{2},\mathbf{2},\bar{\mathbf{4}},\mathbf{1},\mathbf{1})$
11	$\mathfrak{su}(2)^{\oplus 4}\oplus\mathfrak{su}(4)\oplus\mathfrak{so}(10)$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{16})\oplus(\mathbf{1},\mathbf{1},\mathbf{2},\mathbf{2},\mathbf{4},\mathbf{1})\oplus(\mathbf{2},\mathbf{2},\mathbf{1},\mathbf{1},\bar{\mathbf{4}},\mathbf{1})$
12	$\mathfrak{su}(2)^{\oplus 2}\oplus\mathfrak{su}(8)\oplus\mathfrak{so}(10)$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{16})\oplus(\mathbf{1},\mathbf{2},\mathbf{8},\mathbf{1})\oplus(\mathbf{2},\mathbf{1},\bar{\mathbf{8}},\mathbf{1})$
13	$\mathfrak{su}(4)\oplus\mathfrak{sp}(4)^{\oplus 2}\oplus\mathfrak{so}(10)$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{16})\oplus(\bar{\mathbf{4}},\mathbf{4},\mathbf{1},\mathbf{1})\oplus(\mathbf{4},\mathbf{1},\mathbf{4},\mathbf{1})$
14	$\mathfrak{su}(2)^{\oplus 2}\oplus\mathfrak{su}(4)\oplus\mathfrak{sp}(4)\oplus\mathfrak{so}(10)$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{16})\oplus(\mathbf{1},\mathbf{1},\mathbf{4},\mathbf{4},\mathbf{1})\oplus(\mathbf{2},\mathbf{2},\bar{\mathbf{4}},\mathbf{1},\mathbf{1})$
	Minimal	
15	$\mathfrak{su}(2)\oplus\mathfrak{su}(4)^{\oplus 2}$	$(\mathbf{1},\mathbf{4},\mathbf{6})\oplus(\mathbf{2},\bar{\mathbf{4}},\mathbf{1})^{\oplus 3}$
16	$\mathfrak{su}(2)^{\oplus 2}\oplus\mathfrak{su}(4)$	$(\mathbf{1},\mathbf{2},\mathbf{4})^{\oplus 3}\oplus(\mathbf{2},\mathbf{1},\bar{\mathbf{4}})^{\oplus 3}$

TABLE VII. All maximal and minimal algebras for one generation of SM fermions plus fermions in the vectorlike $(\mathbf{2},\mathbf{1})_3\oplus(\mathbf{2},\mathbf{1})_{-3}\oplus(\mathbf{1},\mathbf{3})_{-2}\oplus(\mathbf{1},\bar{\mathbf{3}})_2\oplus(\mathbf{1},\mathbf{1})_0^{\oplus 2}$ representation of $\mathfrak{su}(2)\oplus\mathfrak{su}(3)\oplus\mathfrak{u}(1)$.

	Maximal	
	Algebra	Fermion representations corresponding to β
1	$\mathfrak{su}(2)^{\oplus 2}\oplus\mathfrak{su}(5)$	$(\mathbf{1},\mathbf{1},\mathbf{10})\oplus(\mathbf{1},\mathbf{1},\mathbf{5})\oplus(\mathbf{1},\mathbf{2},\mathbf{1})\oplus(\mathbf{2},\mathbf{1},\bar{\mathbf{5}})$
2	E_6	$\mathbf{27}$
3	$\mathfrak{su}(2)\oplus\mathfrak{su}(5)\oplus\mathfrak{so}(10)$	$(\mathbf{1},\mathbf{1},\mathbf{10})\oplus(\mathbf{1},\bar{\mathbf{5}},\mathbf{1})\oplus(\mathbf{1},\mathbf{10},\mathbf{1})\oplus(\mathbf{2},\mathbf{1},\mathbf{1})$
4	$\mathfrak{sp}(10)\oplus\mathfrak{so}(10)$	$(\mathbf{1},\mathbf{1})\oplus(\mathbf{1},\mathbf{16})\oplus(\mathbf{10},\mathbf{1})$
5	$\mathfrak{su}(5)\oplus\mathfrak{sp}(12)$	$(\mathbf{1},\mathbf{12})\oplus(\bar{\mathbf{5}},\mathbf{1})\oplus(\mathbf{10},\mathbf{1})$
6	$\mathfrak{su}(5)\oplus\mathfrak{so}(12)$	$(\mathbf{1},\mathbf{12})\oplus(\bar{\mathbf{5}},\mathbf{1})\oplus(\mathbf{10},\mathbf{1})$
7	$\mathfrak{so}(11)\oplus\mathfrak{so}(10)$	$(\mathbf{1},\mathbf{16})\oplus(\mathbf{11},\mathbf{1})$
8	$\mathfrak{su}(2)^{\oplus 2}\oplus\mathfrak{su}(5)\oplus\mathfrak{sp}(8)$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{8})\oplus(\mathbf{1},\mathbf{1},\bar{\mathbf{5}},\mathbf{1})\oplus(\mathbf{1},\mathbf{1},\mathbf{10},\mathbf{1})\oplus(\mathbf{2},\mathbf{2},\mathbf{1},\mathbf{1})$
9	$\mathfrak{su}(4)\oplus\mathfrak{su}(5)\oplus\mathfrak{sp}(6)$	$(\mathbf{1},\mathbf{1},\mathbf{6})\oplus(\mathbf{1},\bar{\mathbf{5}},\mathbf{1})\oplus(\mathbf{1},\mathbf{10},\mathbf{1})\oplus(\mathbf{6},\mathbf{1},\mathbf{1})$
10	$\mathfrak{su}(4)\oplus\mathfrak{su}(5)\oplus\mathfrak{sp}(6)$	$(\mathbf{1},\mathbf{1},\mathbf{6})\oplus(\mathbf{1},\bar{\mathbf{5}},\mathbf{1})\oplus(\mathbf{1},\mathbf{10},\mathbf{1})\oplus(\mathbf{6},\mathbf{1},\mathbf{1})$
11	$\mathfrak{sp}(4)\oplus\mathfrak{sp}(6)\oplus\mathfrak{so}(10)$	$(\mathbf{1},\mathbf{1},\mathbf{16})\oplus(\mathbf{1},\mathbf{6},\mathbf{1})\oplus(\mathbf{5},\mathbf{1},\mathbf{1})$
	Minimal	
12	$\mathfrak{su}(3)^{\oplus 3}$	$(\mathbf{1},\mathbf{3},\mathbf{3})\oplus(\bar{\mathbf{3}},\bar{\mathbf{3}},\mathbf{1})\oplus(\mathbf{3},\mathbf{1},\bar{\mathbf{3}})$
13	$\mathfrak{su}(5)\oplus\mathfrak{sp}(6)^{\oplus 2}$	$(\mathbf{1},\mathbf{1},\mathbf{6})\oplus(\mathbf{1},\mathbf{6},\mathbf{1})\oplus(\bar{\mathbf{5}},\mathbf{1},\mathbf{1})\oplus(\mathbf{10},\mathbf{1},\mathbf{1})$
14	$\mathfrak{su}(2)^{\oplus 2}\oplus\mathfrak{su}(5)\oplus\mathfrak{sp}(6)$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{1})^{\oplus 2}\oplus(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{6})\oplus(\mathbf{1},\mathbf{1},\bar{\mathbf{5}},\mathbf{1})\oplus(\mathbf{1},\mathbf{1},\mathbf{10},\mathbf{1})\oplus(\mathbf{2},\mathbf{2},\mathbf{1},\mathbf{1})$
15	$\mathfrak{su}(2)^{\oplus 2}\oplus\mathfrak{su}(4)\oplus\mathfrak{su}(5)$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{1})\oplus(\mathbf{1},\mathbf{1},\mathbf{1},\bar{\mathbf{5}})\oplus(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{5})\oplus(\mathbf{1},\mathbf{2},\mathbf{4},\mathbf{1})\oplus(\mathbf{2},\mathbf{1},\bar{\mathbf{4}},\mathbf{1})$
16	$\mathfrak{su}(2)^{\oplus 2}\oplus\mathfrak{su}(4)\oplus\mathfrak{su}(5)$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{1})^{\oplus 2}\oplus(\mathbf{1},\mathbf{1},\mathbf{1},\bar{\mathbf{5}})\oplus(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{10})\oplus(\mathbf{1},\mathbf{1},\mathbf{6},\mathbf{1})\oplus(\mathbf{2},\mathbf{2},\mathbf{1},\mathbf{1})$
17	$\mathfrak{su}(5)$	$\mathbf{1}^{\oplus 2}\oplus\bar{\mathbf{5}}^{\oplus 2}\oplus\mathbf{10}\oplus\mathbf{5}$
18	$\mathfrak{su}(2)^{\oplus 2}\oplus\mathfrak{su}(4)\oplus\mathfrak{sp}(6)$	$(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{1})\oplus(\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{6})\oplus(\mathbf{1},\mathbf{2},\mathbf{4},\mathbf{1})\oplus(\mathbf{2},\mathbf{1},\bar{\mathbf{4}},\mathbf{1})\oplus(\mathbf{2},\mathbf{2},\mathbf{1},\mathbf{1})$
19	$\mathfrak{su}(2)^{\oplus 2}\oplus\mathfrak{su}(4)$	$(\mathbf{1},\mathbf{1},\mathbf{1})\oplus(\mathbf{1},\mathbf{1},\mathbf{6})\oplus(\mathbf{1},\mathbf{2},\mathbf{4})\oplus(\mathbf{2},\mathbf{1},\bar{\mathbf{4}})\oplus(\mathbf{2},\mathbf{2},\mathbf{1})$

[1] H. Georgi and S. L. Glashow, Unity of All Elementary Particle Forces, Phys. Rev. Lett. **32**, 438 (1974).

[2] H. Fritzsch and P. Minkowski, Unified interactions of leptons and hadrons, Ann. Phys. (Leipzig) **93**, 193 (1975).

[3] H. Georgi, The state of the art—gauge theories, AIP Conf. Proc. **23**, 575 (1975).

[4] J. C. Pati and A. Salam, Lepton number as the fourth color, Phys. Rev. D **10**, 275 (1974).

[5] J. Davighi, A. Greljo, and A. E. Thomsen, Leptoquarks with exactly stable protons, Phys. Lett. B **833**, 137310 (2022).

[6] J. Davighi and J. Tooby-Smith, Electroweak flavour unification, J. High Energy Phys. 09 (2022) 193.

[7] B. C. Allanach, B. Gripaios, and J. Tooby-Smith, Semisimple extensions of the Standard Model gauge algebra, Phys. Rev. D **104**, 035035 (2021).

[8] W. de Graaf, *Lie Algebras: Theory and Algorithms* (Elsevier Science, New York, 2000).

[9] M. Lorente and B. Gruber, Classification of semisimple subalgebras of simple lie algebras, J. Math. Phys. (N.Y.) **13**, 1639 (1972).

[10] B. Gruber and M. Samuel, Semisimple subalgebras of semisimple lie algebras: The algebra (su (6)) as a physically significant example, in *Group Theory and Its Applications* (Elsevier, New York, 1975), pp. 95–141.

[11] E. B. Dynkin, Maximal subgroups of the classical groups, Tr. Mosk. Mat. Obs. **1**, 39 (1952).

[12] E. Dynkin, Semisimple subalgebras of semisimple Lie algebras, Mat. Sb. **72**, 349 (1952).

[13] R. Feger, T. W. Kephart, and R. J. Saskowski, LieART 2.0: A *Mathematica* application for lie algebras and representation theory, Comput. Phys. Commun. **257**, 107490 (2020).

[14] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevD.108.075001 for more details.

[15] J. Tooby-Smith, Arithmetical, geometrical, and categorical forays into particle physics, Ph.D. thesis, Cambridge University, 2021, 10.17863/CAM.72061.

[16] A. Gomes, M. Ruhdorfer, and J. Tooby-Smith, Superflocci GitHub repository (2023), https://github.com/jstoobysmith/Superfloccinaucinihilipilification.

[17] F. Gursey, P. Ramond, and P. Sikivie, A universal gauge theory model based on E_6, Phys. Lett. **60B**, 177 (1976).

[18] J. L. Hewett and T. G. Rizzo, Low-energy phenomenology of superstring inspired E_6 models, Phys. Rep. **183**, 193 (1989).

[19] M. Fukugita, T. Yanagida, and M. Yoshimura, NN oscillation without left-right symmetry, Phys. Lett. **109B**, 369 (1982).

[20] B. Dutta, Y. Gao, T. Ghosh, I. Gogoladze, T. Li, and J. W. Walker, SU(6) GUT origin of the TeV-scale vectorlike particles associated with the 750 GeV diphoton resonance, Phys. Rev. D **94**, 036006 (2016).

[21] A. Hartanto and L. T. Handoko, Grand unified theory based on the SU(6) symmetry, Phys. Rev. D **71**, 095013 (2005).

[22] J. Davighi and J. Tooby-Smith, Flatland: Abelian extensions of the standard model with semi-simple completions, J. High Energy Phys. 09 (2022) 159.

[23] A. Davidson, $B-L$ as the fourth color within an $\mathrm{SU}(2)_L \times \mathrm{U}(1)_R \times \mathrm{U}(1)$ model, Phys. Rev. D **20**, 776 (1979).

[24] R. N. Mohapatra and R. E. Marshak, Local B-L Symmetry of Electroweak Interactions, Majorana Neutrinos and Neutron Oscillations, Phys. Rev. Lett. **44**, 1316 (1980).

[25] R. N. Mohapatra and J. C. Pati, "Natural" left-right symmetry, Phys. Rev. D **11**, 2558 (1975).

[26] G. Senjanovic and R. N. Mohapatra, Exact left-right symmetry and spontaneous violation of parity, Phys. Rev. D **12**, 1502 (1975).

[27] H. Georgi and Y. Nakai, Diphoton resonance from a new strong force, Phys. Rev. D **94**, 075005 (2016).

[28] L. Di Luzio, A. Greljo, and M. Nardecchia, Gauge leptoquark as the origin of B-physics anomalies, Phys. Rev. D **96**, 115011 (2017).

[29] K. S. Babu, X.-G. He, and S. Pakvasa, Neutrino masses and proton decay modes in $\mathrm{SU}(3) \times \mathrm{SU}(3) \times \mathrm{SU}(3)$ trinification, Phys. Rev. D **33**, 763 (1986).

[30] A. de Rújula, H. Georgi, and S. L. Glashow, in *Fifth Workshop on Grand Unification* (World Scientific, Singapore, 1984), p. 88.

[31] N. Yamatsu, Finite-dimensional lie algebras and their representations for unified model building, arXiv:1511.08771.