Extended abstract - Physics as Code: From Scans to Theorems with ITP APIs in $SU(5)$ Model Building

Sven Krippendorf[1] [0000−0001−6374−6828] and
Joseph Tooby-Smith[2] [0000−0003−2831−598X]

[1] DAMTP and Cavendish Laboratory, University of Cambridge Wilberforce Road, CB3 0WA, Cambridge
[2] Department of Computer Science, University of Bath, Bath BA2 7AU, UK

Abstract. This is an extended abstract for the paper "Physics as Code: From Scans to Theorems with ITP APIs in $SU(5)$ Model Building". In that paper we present a framework and workflow for answering questions about bounded model spaces, leveraging interactive theorem provers (ITPs) and reusable, formally verified APIs.

Keywords: Physics · Interactive theorem provers · String theory.

Much of theoretical physics is concerned with restricting the space of models that could feasibly describe the universe, either in whole or in part. For example, in particle phenomenology this takes the form of exclusion plots for given couplings or masses, or constraints on couplings from underlying symmetries, while in string theory it manifests in the string landscape [7, 2] and the swampland program [6]. Depending on the ambient space of models under consideration and the method of restriction, these questions can suffer from a combinatorial explosion, making them difficult to answer.

A concrete example used throughout our paper [4], is when one considers $SU(5)$ supersymmetric (SUSY) grand unified theories (GUTs) with an additional Abelian symmetry. Here, we choose the ambient space of models to consist of those with the following properties: (i) the model contains an up-type and a down-type Higgs; (ii) the matter fields transform in the $\overline{\mathbf{5}}$ and $\mathbf{10}$ representations; and (iii) the fields carry charges under the additional Abelian symmetry drawn from given finite sets determined by their $SU(5)$ representation. These finite sets of charges may, for example, be determined by results from F-theory [5]. As the finite sets grow in cardinality, the ambient space of models grows combinatorially, leading to exactly the type of combinatorial explosion described above.

In [4] we study the restriction of this ambient space to models with *viable* charge spectra. Here, our viability condition requires that the charge spectrum allows a top Yukawa coupling while forbidding dangerous operators at leading order and preventing their regeneration through Yukawa singlet insertions. This is similar to the analysis performed in [3].

Traditionally, such questions are addressed using computer algebra systems (CAS), such as Mathematica, via brute-force scans over the space of models, sometimes aided by pattern-matching rules. However, as the model space grows, this approach quickly becomes infeasible.

In fact, often one does not need to rely on brute-force scans to answer such questions. The physicist's intuition about a problem can be channelled into theorems that directly restrict the space of models. In the example above, the physicist's intuition can be framed as follows. Among the charge spectra admitting a top Yukawa coupling, there exist *minimal witnesses* — minimal sets of charges that already guarantee the top Yukawa coupling. Conversely, every viable charge spectrum can be constructed from such a minimal witness by adding further charges, provided that each additional charge does not introduce any new dangerous operators or enable their regeneration through Yukawa singlet insertions. This intuition allows one to construct viable charge spectra from simple building blocks, bypassing a scan through the entire space of charge spectra. The result is a theorem-backed reduction of the model space that tames the combinatorial explosion.

One could attempt to mirror this intuition in a CAS; however, a CAS typically cannot encode the full chain of reasoning—from definitions, through lemmas and their proofs, to the final computations. To overcome this limitation, in [4] we propose a specific use of interactive theorem provers (ITPs) to answer these types of questions.

An ITP allows one to write mathematical definitions that double as computable functions, state lemmas and theorems about those definitions, and write down proofs of those lemmas and theorems, using automated tactics. Crucially, the ITP automatically verifies that the definitions and theorem statements are mathematically consistent and that the proofs are correct. An ITP has all of the features needed, therefore, to transcribe the physicist's intuition into code.

In [4] we propose that this is done via the building of reusable APIs, developed within projects such as Physlib or Mathlib, comprehensive physics and mathematics library for the ITP Lean [1]. In practice, this means building a library of small, reusable definitions, lemmas, and theorems organized around key data structures. These individual results compose incrementally into more complex theorems of special interest.

Building reusable APIs yields much more than the traditional goal of formalizing individual target theorems, which dominates the current use of AI in formal verification. It produces code that encodes the details of the physicist's intuition in a form that is trustworthy, readable, maintainable, and straightforward to reuse or generalize for future work.

Although our focus here is on model-space reduction problems that suffer from combinatorial explosions, the method we propose is much more general. It applies to any of the model-space problems mentioned above; indeed, the same methodology is used throughout physics in Physlib and mathematics in Mathlib. In [4] we detail how these APIs are constructed in practice, using the

$SU(5)$ model above as a running example and the ITP Lean. The corresponding code is available as part of Physlib at

https://github.com/leanprover-community/physlib,

and is organized into a directory structure that makes it easy to navigate and find relevant results.

The key data structure (around which the API is built) is the *charge spectrum*. We equip it with a number of type-class instances, such as the subset relation, which in Lean lets us leverage existing general results and notation from Mathlib. A classical example of this is putting the instance of an inner product space on a type, which unlocks the ability to use e.g. the Cauchy-Schwarz inequality without having to reprove it. We then introduce interface definitions, both data-valued and predicate-valued, that formalize the questions we wish to answer and the objects we wish to study. Examples include whether a given charge spectrum is a minimal witness of the top Yukawa coupling, and whether it permits a given term in the potential. Around these definitions, we state and formally prove many small lemmas. Some of these lemmas belong to natural *streams*; for example, a family of monotonicity results with respect to the subset relation.

Together, these definitions and lemmas build up to deliver two key results. The first is an executable function that outputs a set of charge spectra. The second is a formally verified theorem certifying that this output is exactly the set of viable charge spectra. These are eventually used in practice to solve a specific problem in F-theory, by combining this API with an API for fluxes on matter curves. This demonstrates a key strength of the API-based approach: the reusability of the underlying definitions and results, which can be composed with other APIs to tackle new problems.

The entire API is developed at a high level of generality: it is not specific to, say, a $U(1)$ extension, but works for any Abelian extension whose charges satisfy suitable basic properties.

The code is open-source and was openly developed on GitHub, so that at every stage of the development process anyone could inspect the code and GitHub commits or contribute to the surrounding discussions. It remains open to future improvements, whether more efficient implementations or further extensions and generalizations of our results. We believe that such openness has been essential to the success of formalization in mathematics and will play an equally central role in physics.

Acknowledgments. We thank Tyler Josephson and Andreas Schachner for discussions and the Physlib community for discussions, infrastructure, and the reusable formal environment on which this work builds. SK has been partially supported by STFC consolidated grants ST/T000694/1 and ST/X000664/1.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. De Moura, L., Kong, S., Avigad, J., Van Doorn, F., von Raumer, J.: The lean theorem prover (system description). In: International Conference on Automated Deduction. pp. 378–388. Springer (2015)
2. Douglas, M.R.: The Statistics of string / M theory vacua. JHEP **05**, 046 (2003). https://doi.org/10.1088/1126-6708/2003/05/046
3. Krippendorf, S., Schafer-Nameki, S., Wong, J.M.: Froggatt-Nielsen meets Mordell-Weil: A Phenomenological Survey of Global F-theory GUTs with U(1)s. JHEP **11**, 008 (2015). https://doi.org/10.1007/JHEP11(2015)008
4. Krippendorf, S., Tooby-Smith, J.: Physics as Code: From Scans to Theorems with ITP APIs in $SU(5)$ Model Building (3 2026)
5. Lawrie, C., Schafer-Nameki, S., Wong, J.M.: F-theory and All Things Rational: Surveying U(1) Symmetries with Rational Sections. JHEP **09**, 144 (2015). https://doi.org/10.1007/JHEP09(2015)144
6. Ooguri, H., Vafa, C.: On the Geometry of the String Landscape and the Swampland. Nucl. Phys. B **766**, 21–33 (2007). https://doi.org/10.1016/j.nuclphysb.2006.10.033
7. Susskind, L.: The Anthropic landscape of string theory pp. 247–266 (2 2003)