Rosetta: an operator basis translator for Standard Model effective field theory

Adam Falkowski¹, Benjamin Fuks², Kentarou Mawatari³, Ken Mimasu⁴, Francesco Riva⁵, Verónica Sanz⁴

¹ Laboratoire de Physique Théorique, Bat. 210, Université Paris-Sud, 91405 Orsay, France
² Institut Pluridisciplinaire Hubert Curien/Département Recherches Subatomiques, Université de Strasbourg/CNRS-IN2P3, 23 rue du Loess, F-67037 Strasbourg, France
³ Theoretische Natuurkunde and IHE/ELEM, Vrije Universiteit Brussel, and International Solvay Institutes, Pleinlaan 2, B-1050 Brussels, Belgium
⁴ Department of Physics and Astronomy, University of Sussex, Brighton BN1 9QH, UK
⁵ CERN, Theory Division, 1211 Geneva, Switzerland.

Abstract. We introduce Rosetta, a program allowing for the translation between different bases of effective field theory operators. We present the main functions of the program and provide an example of usage. One of the Lagrangians which Rosetta can translate into has been implemented into FeynRules, which allows Rosetta to be interfaced into various high-energy physics programs such as Monte Carlo event generators. In addition to popular bases choices, such as the Warsaw and Strongly Interacting Light Higgs bases already implemented in the program, we also detail how to add new operator bases into the Rosetta package. In this way, phenomenological studies using an effective field theory framework can be straightforwardly performed.

1 Introduction

The start of a second LHC experimental era raises new hopes to detect physics beyond the Standard Model (BSM). The high energy of the experiment increases the chances of a direct discovery of new physics resonances, while a combination of high energy and high luminosity favors the possible observation of new phenomena via Standard Model (SM) precision tests. Interestingly the latter offers a complementary and model-independent tool for BSM searches if the results are interpreted in the context of an effective field theory (EFT). The EFT indeed captures in a general way the low-energy effects of heavy new physics from a bottom-up perspective. More precisely, it systematically organizes possible departures from the SM as an expansion in the energy at which the processes of interest occur over the (high) new physics scale, and simultaneously provides a dictionary to interpret these departures in the context of explicit BSM models.

Given the SM field content (including a single Higgs doublet), assuming baryon and lepton number conservation, flavor universality and a linear realization of the electroweak symmetry, the leading effects implied by an EFT description consist of dimension-six operators that are supplemented to the SM Lagrangian. At this order, 59 (76 real) new independent coefficients [1, 2] of dimension-six operators grows to 2499 [3].

¹ Relaxing flavor universality, the number of independent dimension-six operators grows to 2499 [3].

for parameterizing BSM searches but is also testable per se by looking at correlations among the expected signatures.

Another important aspect of the EFT approach is the choice of the operator basis, so that a given physical effect could be modeled by different combinations of operators at a fixed order in the EFT expansion. This well-known fact is related to the possibility of redefining the SM fields in such a way that the zeroth order Lagrangian in the EFT expansion (i.e., the SM Lagrangian) is unaltered, while combinations of the first-order operators (i.e., dimension-six operators) proportional to the SM equations of motion can be eliminated up to subleading higher-dimensional effects. For this reason, different complete and non-redundant operator bases have been proposed in the literature, sharing the same physical predictions but having different advantages. The most popular choices include the so-called Warsaw basis [2], SILH (strongly interacting light Higgs) basis [10, 11] and BSM primaries basis [6, 12, 13]. The Warsaw basis represents the first set of non-redundant operators that has been proposed and is particularly appropriate for comparisons with BSM theories that modify the interactions of the SM fermions. In contrast, the SILH basis has been designed to capture the effects of universal theories where new physics mostly couples to the SM bosons. Finally, the BSM primaries basis is more suitable for a bottom-up approach since it is formulated in terms of mass-eigenstates and has a more transparent connection to measurable quantities, its operators being aligned with physical observables.

Given these multiple viewpoints, it is cumbersome to express the experimental results in a basis-independent manner that can be readily interpreted in any of the above-mentioned frameworks. On the other hand, different bases may be conve-
nient for particular applications, either because they facilitate the comparison with a given class of BSM theories or simply because different experimental analyses look more transparent in a specific basis. For instance, the Warsaw basis contains an apparent blind direction with respect to the electroweak precision tests \cite{4,14}, which introduces large theoretical correlations among all LEP constraints. As a result, the bounds on the strength of the dimension-six interactions appear less transparent \cite{15}. The SILH basis has a similar drawback yielding a correlation between LEP2 and LHC constraints, while the downside of the BSM primaries basis lies in the comparison with explicit BSM models that is complicated. The Rosetta package that we present in this paper has been designed to explicitly solve such problems by allowing for a straightforward translation between different EFT languages.

In addition to translating, another important goal of the Rosetta program is to provide a platform for communication with Monte Carlo event generators, no matter which EFT basis is chosen. To achieve this, we have implemented in Rosetta the Higgs basis for EFT operators that has been recently designed by the LHC Higgs Cross Section working group (LHC-HXSWG) \cite{16}. This proposal, built on the BSM primaries basis (see Ref. \cite{13}), combines two ingredients. First, all possible operators of dimension up to six are written in terms of the SM mass-eigenstates

$$\Delta L^{(mass)} = \sum_i \frac{c_i}{\Lambda_i^2} O_i (G^a_{\mu\nu}, W^\pm_\mu, Z_\mu, A_\mu, h, t, b, \nu_\tau, \tau, \ldots ), \quad (1)$$

where the operators $O_i$ have a mass dimension ranging from two to six. The dimensionless coefficients $c_i$ are then suppressed by an appropriate power $d_i$ of the high-energy scale $\Lambda_i$, with $d_i = -2, \ldots, 2$. We refer the ensemble of operators included in the resulting Lagrangian, which is in spirit very similar to the Higgs characterisation Lagrangian of Ref. \cite{17}, as the BSM characterisation (BSM) Lagrangian. Due to the lack of manifest $SU(2)_L \times U(1)_Y$ invariance, the BSM Lagrangian is associated with a larger number of independent coefficients compared to the Warsaw, SILH or BSM primaries bases. For this reason, the second index defining the Higgs basis consists of relations among the $c_i$ coefficients that restore the full $SU(3)_C \times SU(2)_L \times U(1)_Y$ symmetry. As summarized in Table \ref{tab:results}, the BSM primaries and Higgs Lagrangians are both of the form of $\Delta L^{(mass)}$, but they additionally include constraints among the different Wilson coefficients that render the Lagrangian invariant under the electroweak symmetry. In contrast, the Warsaw and SILH basis Lagrangians are written in terms of the SM gauge-eigenstates,

$$\Delta L^{(gauge)} = \sum_i \frac{c^i}{\Lambda_i^2} O_i (G^a_{\mu\nu}, W^i_\mu, B_\mu, \Phi, Q_L, u_R, d_R, L_L, e_R), \quad (2)$$

and are manifestly symmetric under the electroweak symmetry group.

We have implemented the mass basis Lagrangian $\Delta L^{(mass)}$ into \textsc{FeynRules} \cite{18,19} and tuned the output format of Rosetta so that the translation maps an EFT Lagrangian given in a specific basis to $\Delta L^{(mass)}$ and generates an output file that is compatible with the \textsc{FeynRules} implementation. As a consequence, any high-energy physics tool (and in particular any Monte Carlo event generator) that is interfaced to \textsc{FeynRules} can be employed within the context of any EFT basis of operators that is included in Rosetta.

With the advent of automated next-to-leading order (NLO) accurate Monte Carlo event generation software, it is important that Rosetta remains flexible enough to eventually provide compatibility with this new generation of tools. Recent progress has been made on the theory side both in implementing the linear dimension-six description discussed above in the \textsc{FeynRules} framework \cite{20} and in calculating the renormalisation group (RG) evolution of the full set of operators and their mutual mixing \cite{3,21,22}. In the former case, Rosetta can simply be extended to provide an output compatible with the eventual NLO model implementation, analogously to the BSMC Lagrangian. The latter case of evaluating the RG running effects, while being a slightly separate issue, highlights a key feature of our tool, given that the calculation of these effects has only been performed in the original Warsaw basis of Ref. \cite{2}. The framework provided by Rosetta allows for the application of these results in any desired basis. Although the initial version of the software does not explicitly deal with these effects, their translation functionality can already be used in their context and we plan for future versions to incorporate RG evolution of the SM EFT Wilson coefficients.

The remainder of this paper is organized as follows. In Section \ref{sec:introduction} we describe the basic functionalities of Rosetta and how to make use of the program. Section \ref{sec:applications} is dedicated to an example of usage of Rosetta in which we focus on new physics Higgs couplings to the SM bosons. We express them in different bases and detail the output that is provided by Rosetta. Our work is summarized in Section \ref{sec:conclusions}.

\section{Rosetta}

The aim of Rosetta is to provide a modular and flexible package for EFT basis translation and communication with event generation tools. The primary framework which Rosetta has been designed to translate into is the phenomenological effective Lagrangian, $\Delta L^{(mass)}$, which will be explicitly defined in Section \ref{sec:results}. The motivation for this choice lies in the availability of an implementation within the \textsc{FeynRules} frame-

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
Basis & Underlying gauge symmetry & Fields used in the Lagrangian \\
\hline
Warsaw, SILH & $SU(3)_C \times SU(2)_L \times U(1)_Y$ & Gauge-eigenstates \\
BSM primaries, Higgs & $SU(3)_C \times SU(2)_L \times U(1)_Y$ & Mass-eigenstates \\
Higgs/BSM characterisation & $SU(3)_C \times U(1)_{EM}$ & Mass-eigenstates \\
\hline
\end{tabular}
\caption{Main features of the different EFT basis choices discussed in this document.}
\end{table}
work [13], to be downloaded from the FeynRules model repository [24], which ensures the link with event generators and high-energy physics programs [25, 26].

The most basic functionality of Rosetta is to map a chosen set of input parameters (the Wilson coefficients in a specific basis choice) onto the BSMC coefficients such that the output can be employed within tools relying on a BSMC basis description. As long as the input format respects the conventions sketched in Section 2.2 and that are inspired by the Supersymmetry Les Houches Accord (SLHA) [27, 28], the user may define his/her own map to the BSMC coefficients (or to any other basis implementation) and proceed with event generation using the accompanying FeynRules implementation. This highlights one of the key features of Rosetta, the possibility to easily define one’s own input basis and directly use it in the context of many programs via the translation functionality of Rosetta. The strength of this approach is that it is much simpler than developing from scratch new modules for existing tools in the context of a new basis. To this end, Rosetta not only enables the translation of an EFT basis into the BSMC Lagrangian, but also allows for translations into any of the other bases included in the package, i.e., currently the Higgs, Warsaw and SLH bases. Translations between these three bases in any direction are possible, so that the addition of a new basis by the user only requires the specification of translation rules to any one of the three core bases. One is subsequently able to indirectly translate the new basis into any of the other two bases, as well as into the BSMC Lagrangian. The details of how one can implement a new basis in Rosetta are discussed in Section 2.3.

2.1 Getting started with Rosetta

The latest release of Rosetta can be obtained from http://rosetta.hepforge.org

The package contains a Python executable named translate, an information file named README and two directories, a first folder (named Cards) collecting example input files and a second folder (named Rosetta) including the source code of Rosetta. The executable takes as input an SLHA-style parameter file with the coefficients of the dimension-six operators associated with a particular basis. Information on the format of such an input file can be found in Section 2.2. The execution of the translate script from a shell yields the generation of an output parameter file where all parameters are this time the coefficients of the dimension-six operators associated with a specified new basis, the default choice being the BSMC Lagrangian. The tool can be used by typing in

```
./translate PARAMCARD.dat OPTIONS
```

where PARAMCARD.dat is the name of the SLHA-style input file and OPTIONS stands for optional arguments. The latter could consist of one or more of the following choices that will modify the behavior of the program.

- **h or --help** This displays a help message and exits the program.
- **-o or --output** This allows for the specification of the name of the output file, that is by default PARAMCARD_new.dat.
- **-s or --silent** The program suppresses warnings and takes the default answer to any question that may have to be asked to the user.
- **-t or --target** This allows for providing the name of the basis into which the translation occurs, the default being bsmc and the other acceptable choices being higgs, silh or warshaw. This allows the program to overwrite any pre-existing output file.
- **-w or --overwrite** This allows to use the interface with the eHDecay program [11] for the calculation of the Higgs boson width and branching fractions. See Section 2.5.2.
- **-f or --flavor** This allows to specify the treatment of the flavor structure relevant for the fermionic operators, the default being general and the other acceptable choices being universal and diagonal. See Section 2.5.1.
- **-d or --dependent** This allows the program to also write out any dependent parameters calculated by the translation function to the output file.

On run time, Rosetta starts by performing several checks on the input parameters and verifies the consistency of the input file with respect to the specifications of the internal basis implementation. In this way, any missing SM inputs (with respect to the requirements included in the required inputs and required masses attributes of the basis class, see Section 2.3) can be included using the value provided in the Particle Data Group (PDG) review [29], while any missing coefficient associated with an operator that is present in the basis (and thus declared in the independent attribute of the basis class, see Section 2.3) can be included with a zero value.

Once the translation is achieved, Rosetta outputs a new parameter file that is by default named PARAMCARD_new.dat. This file contains the values of all parameters relevant for the target basis and also includes the necessary modifications to the input parameters, such as the W-boson mass that may depend on some dimension-six operator coefficients.

2.2 Input files and their handling in Rosetta

Rosetta requires input parameters to be given under the form of a file encoded in a format similar to the SLHA one detailed in Refs. [27, 25]. Parameters are grouped into blocks and each parameter is identified inside its own block by one or more integer numbers called counters. For instance, the SM inputs necessary for the definition of the SILH basis would read

```
BLOCK SMINPUTS #
1 +1.27916e+02 # aEMW1
2 +1.16638e-05 # Gf
3 +1.18400e-01 # aS
4 +9.11876e+01 # MZ
25 +1.25000e+02 # MH
```

where the different entries respectively correspond to the inverse of the electromagnetic coupling constant (aEMW1), the
Fermi constant \((G_F)\), the strong coupling constant \((\alpha_S)\), the 
\(Z\)-boson mass \((M_Z)\) and the Higgs boson mass \((m_H)\). Inspired
by the usual SLHA conventions, all masses are also collected
into a block called MASS where the counters correspond to
the PDG identifiers of the particles \([29]\). Furthermore, matrix
quantities receive a block of their own with counters specifying
the position inside the matrix. In this way, a single block
would be needed to encode, for instance, the \(c_{Hud}\) coefficients
associated with the \(O_{Hud}\) operator of the Warsaw basis that is
defined by
\[
O_{Hud} = -i [\bar{u} \gamma^\mu d] [\Phi^\dagger D^\mu \Phi].
\]
In this expression, \(u\) and \(d\) denote the \(SU(2)_L\) singlets of right-
handed up-type and down-type quark fields, respectively, and \(\Phi\) and \(D^\mu \Phi\) stand for the weak doublet of Higgs fields and its
gauge-covariant derivative. In flavor space, the \(c_{Hud}\) coefficients
take the form of a matrix, implemented in the input file as

```plaintext
BLOCK WBxHud
  1 1 0.1e+00 # cHud11
  1 2 0.0e+00 # cHud12
  1 3 0.0e+00 # cHud13
  2 1 0.0e+00 # cHud21
  2 2 0.1e+00 # cHud22
  2 3 0.0e+00 # cHud23
  3 1 0.0e+00 # cHud31
  3 2 0.0e+00 # cHud32
  3 3 1.0e+00 # cHud33
```

The block name contains information on the basis (WB) and
on the considered operator (Hud). Sample parameter files for
all core bases can be found in the Cards directory shipped
with the program. Within those files, we have adopted the
above block naming scheme. The name of each block starts
with two letters identifying the basis (BC, HB, SB and WB for
the BSMC, Higgs, SILH and Warsaw bases respectively) that
are followed by a separator (\(x\)), and ends with the name of
the considered coefficient as it is defined in the LHCHXSWG
proposal for an EFT basis choice \([10]\). In the case of EFT
operators independent of fermions, the related (non-matrix)
coefficients are collected in different blocks as a function of
the Lorentz structure of the operators. For instance, the \(SBxV2H2\)
block will include all operators of the SILH basis containing two
occurrences of the Higgs field and two occurrences of the gauge
fields. Their ordering follows their order of appearance in the
LHCHXSWG proposal. The imaginary parts of all parameters
are stored in corresponding blocks whose names are prefixed
with the IM tag.

The Rosetta package contains built-in methods for dealing
with an SLHA-like structure, and these methods have all
been implemented in the Rosetta/SLHA.py file. When an
input file is read, the parser included in the SLHA.py file rec-
ognizes the existing BLOCK and DECAY structures of the input
file and stores them as instances of the SLHA.NamedBlock,
SLHA.NamedMatrix and of the SLHA.Decay classes. These
are dictionary-like objects that can be assigned, indexed and it-
erated over as regular PYTHON dictionaries. An SLHA.Named-
Block object reflects the information embedded in an SLHA
block so that it possesses a name attribute and stores values
associated with integer keys as well as a mapping from the
integer keys to the parameter names. In this way, parameters
can be accessed by indexing either their integer key or their
name. Similarly, SLHA.NamedMatrix objects function anal-
ogously but operate with a pair of indices for indexing. An

| Counter | Parameter | Rosetta name |
|---------|-----------|--------------|
| 1       | The inverse of the electromagnetic coupling con-
|         | tant \(\alpha^{-1}\) | aEM1         |
| 2       | The Fermi constant \(G_F\) | GF           |
| 3       | The strong coupling con-
|         | tant \(\alpha_s\) | aS           |
| 4       | The \(Z\)-boson mass \(M_Z\) | MZ           |
| 5       | The bottom quark mass | MB           |
| 6       | The top quark mass \(m_t\) | MT           |
| 7       | The tau lepton mass \(m_\tau\) | MTAU         |
| 25      | The Higgs boson mass \(m_H\) | MH           |

| Table 2. Identifying counters of the SMINPUTS block with the SM parameters allowed to be used within Rosetta. This
generalizes the SLHA standards where the Higgs mass is ignored \([28]\). The internal names used by Rosetta are also given.

SLHA.Decay object contains an integer attribute PID that is
the PDG identifier of the particle whose decays are described
by the considered block, as well as a total attribute allowing
for the storage of the total width. Individual decay channels
are then indexed by tuples of PDG codes associated with the
decay products, and the stored values are the branching frac-
tions. Finally, the SLHA.Card.py file also includes the definition of
the SLHA.Card class that serves as a container for a collection
of instances of the above objects. The implementation of any
basis in Rosetta therefore requires the user to provide defini-
tions for the blocks and parameters to be specified in the input
file that will be read into an SLHA.Card instance belonging to
that basis class. More practical information and examples are
given in Sections \([23]\) and \([27]\).

Three special blocks named BASIS, SMINPUTS and MASS
must always be present. The first and only element of the
BASIS block refers the name of the basis into which Roset-
ta must read the input file and informs the program on the
other blocks it should look for, based on the structure speci-
fied in the implementation of that particular basis. This name
should be a single unique string with no spaces. The next two
mandatory blocks consist of conventional input blocks speci-
fying the values of the SM inputs and of the particle masses.
The set of required inputs will depend on the specifications
in the corresponding basis implementation. Moreover, the user
can optionally specify the value of the elements of the CKM
matrix by setting their real and imaginary parameters. The
PDG review \([29]\) is used by Rosetta. All extra blocks and
decay structures are stored, left unchanged and passed to the
output file unless the user demands to use the eHDECAY pro-
gram, which will overwrite any existing decay information on
the Higgs.

2.3 Structure of Rosetta

Rosetta is a PYTHON package containing the implementation of
a Basis class equipped with several utility functions for
reading, processing and writing SLHA-style parameter files.
Working implementations of bases are derived from this class
and only require a small amount of information specifying the block structure of the EFT parameters, the required SM inputs and a series of translation functions to other existing basis implementations. In order to be able to define a new basis class, we describe in this section the properties of the Basis objects.

The Basis class has a number of intrinsic data members that should be defined in order to get a working implementation of an EFT basis. These consist of the independent, required_inputs and required_masses attributes already mentioned in Section 2.1 together with the name, blocks and flavored members of the class.

name
Unique string identifier for the basis implementation, e.g., higgs, bsmc, silh or warsaw for the core bases shipped with the package.

independent
List of strings containing the names of the independent EFT operator coefficients of the basis. These are expected to be present in the input parameter file.

required_inputs
Set of integers containing the SLHA counters of the required SM inputs. See Table 2 for the complete list of those allowed in Rosetta.

required_masses
Set of integers containing the PDG identifiers of the particles whose masses are required as input and that are not included in required_inputs.

blocks
Dictionary with the non-matrix SLHA block names as keys and lists of coefficients stored in that block as values.

flavored
Dictionary with matrix SLHA block names as keys. The values are other dictionaries with the keywords kind, domain and cname as keys. This describes the properties of the matrices to be created from its name, which facilitates the implementation of the translation functions that in general extensively refer to individual parameter values. This however assumes that there are no duplicate parameter names in the SLHA-like input file, which nevertheless leads to a program exception. There are hence multiple ways to access a given parameter. For example, a parameter A stored as the third element of a block MyBlock that is part of the definition of a basis MyBasis could be equally accessed as

\[
\text{MyBasis}'A'\]
\[
\text{MyBasis.card}'A'\]
\[
\text{MyBasis.card.blocks}'MyBlock'['A']\]
\[
\text{MyBasis.card.blocks}'MyBlock'['3']\]

In the lines above, the parameter A is respectively accessed from the MyBasis object, from the SLHA.Card instance associated with the current basis and from the SLHA.NamedBlock object associated with the MyBlock block (using either the parameter name or the counter as an index).

2.4 Implementing a new basis

One of the important features of Rosetta is the intended ease with which a user can define a new basis class to suit his/her specific physics needs. In the context of an ultraviolet complete model, he/she may be interested in the phenomenological consequences of a particular high-scale scenario in the EFT framework. Imagining that he/she has derived all dimension-six Wilson coefficients in a particular basis, Rosetta could be used to map these coefficients to the FeynRules effective Lagrangian implementation in the mass-eigenstate basis so that the collider phenomenology of such a scenario could be investigated. This task is realized by implementing a new basis in Rosetta and by connecting the new basis input parameters to those of one of the existing core basis implementations.

Alternatively, the user may have developed a particular resource performing a useful analysis or calculation in a non-standard basis choice. The corresponding basis implementation in Rosetta with a translation to one of the core bases could then allow one to use this tool in the context of all other existing basis implementations in Rosetta and therefore greatly widen its scope. The HDecay feature of Rosetta is an example of this, as it works with a set of operators corresponding to the SILH basis. Via Rosetta, eHDecay is now available for calculations in the SILH, Warsaw and Higgs bases, as well as in any additional basis that may be implemented in the future.

In this section, we provide an example that outlines the basic requirements for implementing a new basis in Rosetta. We also refer the reader to the file Rosetta/TemplateBasis.py which serves as a concrete toy example that can be used as a template for creating a new basis class as well as the core basis implementations for more complete realizations.

All Rosetta basis classes inherit from the mother class Basis implemented in the Rosetta/external/Basis.py file. This class contains all the machinery necessary for reading, writing and translating so that a new basis implementation solely demands the user to create a new file that must be
saved in the Rosetta directory and that includes the declaration of a basis subclass. The user has then to define the class attributes described in Section 2.4. First, it is essential that the name of the basis class matches the filename in which it is saved minus the extension in order to ensure a proper running of the program. Second, the independent, blocks and flavored attributes of the class define the input parameters of the basis and their desired SLHA-like structure, while the required inputs and required masses lists are specified according to the needs of the translation functions that are planned to be implemented. One can also specify a dependent attribute to explicitly define a particular parameter as dependent. For instance, the following code refers to the implementation of a new basis class called MyBasis and has been included in the file Rosetta/MyBasis.py.

```python
from internal import Basis
class MyBasis(Basis.Basis):
    name = 'mybasis'
    independent = ['A', 'B', 'one', 'two', 'MYxMAT']
    dependent = ['Cmat3x3']
    blocks = {'letters': ['A', 'B', 'C'],
              'numbers': ['one', 'two', 'three']}
    flavored = {'MYxMAT': {'kind': 'hermitian',
                           'domain': 'complex',
                           'cname': 'mat'}}
    required_inputs = {1, 2, 4}
    required_masses = {24, 25, 6}

This snippet of code specifies the declaration of the basis class MyBasis whose unique string identifier is given by mybasis. The independent parameters to be read from an input SLHA-like file are defined to be A, B, one and two and are assumed to be organized into the two SLHA blocks LETTERS and NUMBERS. A flavored matrix, MYxMAT, is also present and deemed to be an independent input parameter except for its (3,3) component that is explicitly included within the dependent attribute of the basis class. The translation methods to be implemented require the knowledge of six SM masses and parameters that must be specified via the required inputs and required masses attributes of the basis class. In our case, the electroweak inputs $\alpha^{-1}$, $G_F$ and $m_Z$ are connected to the SMINPUTS block of the SLHA-like input structure, while the $W$-boson, Higgs boson and top quark masses are connected to its MASS block. The extra parameters C and three are dependent parameters that the user has to define in terms of the independent and SM parameters (see below). The non-SM part of an illustrative input file could be

```
BLOCK BASIS
  1 mybasis # input basis
```

```python
1 2 9.9e-3 # Imat1x2
1 3 1.9e-4 # Imat1x3
2 3 4.6e-3 # Imat2x3
```

while its SM part would include the SMINPUTS and MASS blocks with values for the six above-mentioned SM inputs, as well as the two blocks related to the CKM matrix in the case where one would be interested in using non-default values for its elements. Only the relevant elements of MYxMAT need be provided given that it is declared to be Hermitian, and the (3,3) element is left unspecified as it is a dependent parameter.

The dependent parameters are evaluated via a method named calculate_dependent() that must be provided by the user. Continuing with the example above, we include in the new basis class implementation the code

```python
    def calculate_dependent(self):
        self['C'] = (self['A'] + self['B']) / 2.
        self['three'] = (self['one'] - self['two']) / 2.
        self['MYxMAT'][3, 3] = 10. * self['MYxMAT'][2, 2]
```

This imposes that the C parameter is defined as the mean of the A and B parameters, that the three parameter equals half of the difference of the one and two parameters and that the (3,3) entry of the MYxMAT matrix is equal to 10 times the value of its (2,2) entry.

When executed, ROSETTA begins with the reading of the input file and next calls the calculate_dependent() method for evaluating the remaining basis parameters. ROSETTA finally performs the translation to another basis by using the translation methods defined by the user. Their implementation requires the use of a translation decorator with an argument that refers to the name of the target basis and that must match a basis implementation contained in the Rosetta directory. For example, we could link the mybasis class above to the Warsaw basis by implementing

```python
@Basis.translation('warsaw')
def mytranslation(self, wbasis):
    a_EW = 1 ./ self.inputs['aEWm1']
    wbasis['cWW'] = a_EW * self['C']
    wbasis['WBxHpl'][1, 1] = self['two']
    return wbasis
```

Translation functions such as the mytranslation(...) one above take an instance of the target basis class as their only argument and return it after setting its parameter values. Relations involving (matrix) parameters with a flavor structure should be performed in a flavor general way, as discussed in Section 2.5.1.

If modifications to the SM input parameters need to be made (i.e., the mass and inputs attributes of the basis class), the function modify_inputs() must be implemented similarly to the calculate_dependent() method. The following example defines a shift of the $W$-boson mass by the A parameter,

```python
    def modify_inputs(self):
        self.mass[24] = self.mass[24] + self['A']
```

In general, the user-defined functions may require the evaluation of parameters such as the weak and hypercharge gauge couplings or the electroweak mixing angle. The choice of relations (e.g., tree- or loop-level) to be used to consistently derive these parameters from the inputs is left to the user. In the core bases provided with ROSETTA, the calculate_inputs()
method relies on tree-level relations to deduce all the SM parameters. Having defined a basis class according to these specifications, Rosetta is able to detect the presence of the basis implementation and to automatically construct possible translation paths to other existing bases from the user-defined translation functions. The recognition of the implemented basis by Rosetta is also reflected in the help message accompanying the translate script, the name of the new basis appearing as one of the possibilities for the target basis option.

2.5 Additional features

2.5.1 Flavor structure of the fermionic operators

In the general case, each matrix block of the input file includes one entry for each possible flavor assignment of the corresponding operator. The flavor option of the translate executable introduced in Section 2.1 allows the user to make assumptions over, being a charged-current operator, the MFV construction requires the insertion of the CKM matrix such that the operator is composed of right-handed quarks only. More specifically, the flavor structure of the operator requires two Yukawa insertions as it is composed of right-handed quarks only. Thus, the flavor-symmetry-breaking Yukawa-like operators are normalized by the masses of the fermions such that their flavor-diagonal elements are retained. In the universal case one is allowed to define matrix blocks containing only the (1,1) element while in the diagonal case, all three diagonal elements must be provided. Sample input files can be found in the Cards directory of the program. In the definitions of the three core bases, the flavor-symmetry-breaking Yukawa-like operators are normalized by the masses of the fermions such that the universal flavor option will lead to a minimally flavor-violating (MFV) structure where the physical effects of the coefficients are scaled by the corresponding fermion masses [30].

For example, in the Higgs basis, these Yukawa-like terms are written as:

$$\Delta \mathcal{L}_{\text{Yuk}} = \sqrt{m_i^2 m_j^2} \delta \gamma^i \bar{f}^i \left( \cos \phi^i - i \gamma^5 \sin \phi^i \right) f^j.$$  \hspace{1cm} (4)

The corresponding normalizations are also used for the Warsaw and SLH basis implementations in Rosetta to simplify the translations and also the possibility of encoding MFV into any EFT description. The same argument applies to the dipole operators, $O_{1V}$, as well as the $O_{1\text{ud}}$ operator mentioned in Section 2.2. The former set of operators breaks the flavor symmetry in an identical way to the Yukawa-like operators and will hence receive the same normalization. In the latter case, the flavor structure of the operator requires two Yukawa insertions as it is composed of right-handed quarks only. Moreover, being a charged-current operator, the MFV construction requires the insertion of the CKM matrix such that the operator is normalized as

$$O_{1\text{ud}} = -i m_u m_d \sqrt{V_{ij}^\text{CKM}} \left[ \gamma^\mu d^\dagger \right] \left[ d^\dagger D_\mu d \right].$$  \hspace{1cm} (5)

Since this particular operator is unique and maps to a single operator in all of the other core bases, the corresponding translations remain unaffected. This normalization is however not the same as the one chosen in Ref. [16]. Users should therefore bear in mind these normalizations which have been chosen to single out operators that explicitly break the flavor symmetry of the Lagrangian. That being said, they are merely a convenient way for the user to implement MFV and can be worked around if the user so desires.

Rosetta recognizes coefficients by their names so that the naming of the elements of the matrix coefficients must respect the conventions described in the previous section for their real (an r prefix) and imaginary (an i prefix) parts, and for their position $(i,j)$ inside the matrix (an ixj suffix). Implementing translations from flavored parameters should ideally always be done in the most general case such that the various flavor options work correctly. To this aim, basic matrix algebra operations have been implemented in the internal/matrices.py module of Rosetta. The available functions are matrix_mult, matrix_add, matrix_sub and matrix_eq and correspond to matrix multiplication, addition, subtraction and assignment respectively. They can be used to assign values to a matrix SLHA block according to the result of a specific operation between two other matrix SLHA blocks. These functions require two mandatory arguments for the objects between which the operation should be performed and a third optional argument specifying the matrix block to which the result of the operation should be assigned. For instance, matrix_mult(M1, M2, M3) assigns to the matrix M3 the result of the multiplication of the matrices M1 and M2. If the M3 argument is omitted, a generic matrix object is returned such that matrix utility functions can be combined. The matrix_eq(M1, M2) method is the only exception. It takes two mandatory arguments M1 and M2 and allows for the assignment of the elements of the M1 matrix to the M2 matrix.

A concrete example can be found in the calculate_depend() function included in the Higgs basis implementation. The deviations of the W-boson couplings to the weak doublet of left-handed quark fields $\delta g_{W_L}$ are related to those of the Z-boson couplings to the individual left-handed up-type and down-type quarks $\delta g_{Z_u}$ and $\delta g_{Z_d}$ via the CKM matrix $V_{\text{CKM}}$, $\delta g_{W_L} = \delta g_{Z_u} \cdot V_{\text{CKM}} - V_{\text{CKM}} \cdot \delta g_{Z_d}$.  \hspace{1cm} (6)

The Rosetta implementation of this relation makes use of a combination of the matrix_mult and matrix_sub method,

```
matrix_sub(
matrix_mult(HB['HBxdGLzu'], HB.ckm),
matrix_mult(HB.ckm, HB['HBxdGLzd']),
HB['HBxdGLwq'])
```

where HB is an instance of the Higgs basis class. The third argument of the matrix_sub method allows one to assign the result of the subtraction to the elements of the HBxdGLwq matrix block. Matrix blocks also come with the T() and dag() methods for transposing and Hermitian conjugation.

2.5.2 Interface to eHDECAY

In order to calculate dimension-six operator contributions to the Higgs boson width and branching ratios, Rosetta includes an interface to the eHDECAY program [11]. It can be switched on by executing the translate script with the eHDECAY option (see Section 2.3). In order to use this feature, the path to a local installation of eHDECAY on the user system should

\[ \begin{align*}
\Delta \mathcal{L}_{\text{Yuk}} &= \sqrt{m_i^2 m_j^2} \delta \gamma^i \bar{f}^i \left( \cos \phi^i - i \gamma^5 \sin \phi^i \right) f^j. \\
O_{1\text{ud}} &= -i m_u m_d \sqrt{V_{ij}^\text{CKM}} \left[ \gamma^\mu d^\dagger \right] \left[ d^\dagger D_\mu d \right].
\end{align*} \]
be specified in the Rosetta/config.txt file, next to the eHDECAY_dir keyword, and a (possibly indirect) translation linking the basis of interest to the SILH basis should exist. If so, the translation will be performed, eHDECAY will be run internally and an SLHA decay block for the Higgs boson will be appended to the output file.

Since the SILH basis description adopted in eHDECAY assumes the MFV paradigm, an additional layer of translation is internally performed by Rosetta to render its internal SILH basis implementation MFV-compliant. Details can be found in Section 5.3.

3 Mapping different EFT basis choices

In this section, we discuss the BSMC Lagrangian containing redundant parameters that is the default basis which Rosetta has been designed to translate into. We explain the relations with the non-redundant Higgs, Warsaw and SILH bases and focus on a subset of operators connected to single Higgs production at the LHC to provide examples of usage of Rosetta.

3.1 The BSMC Lagrangian and the Higgs basis

To study the Higgs boson properties in detail at the next-to-leading order runs, the LHCHXSWG has proposed a parameterization of anomalous interactions of the SM fermions, gauge bosons and the Higgs boson allowing both for a transparent linking to physical observables and for an easy implementation in Monte-Carlo event generators. The framework is that of a general effective Lagrangian defined in the mass-eigenstate basis, where all kinetic terms are canonically normalized and diagonal, and where all mass terms are diagonal. Moreover, the tree-level relations between the gauge couplings and the usual electroweak input parameters \( G_F, \alpha(0), m_Z \) are the same as in the SM. In such a frame, i.e., in the BSMC Lagrangian, the coefficients of the interaction terms in the Lagrangian are related in an intuitive way to quantities observable in experiment, and any parameter in the effective Lagrangian can be measured.

The BSMC Lagrangian captures all physics effects that may arise in the presence of lepton-number and baryon-number violation, and any parameter in the effective Lagrangian can be related in an intuitive way to quantities observable in experiments. In such a frame, tree-level relations between the gauge couplings and the usual electroweak mixing angle, \( \alpha \), in the BSMC Lagrangian, \( \xi_{\mu\nu} \) and all kinetic terms are canonically normalized and diagonal, and where all mass terms are diagonal. Moreover, the tree-level relations between the gauge couplings and the usual electroweak input parameters \( G_F, \alpha(0), m_Z \) are the same as in the SM. In such a frame, i.e., in the BSMC Lagrangian, the coefficients of the interaction terms in the Lagrangian are related in an intuitive way to quantities observable in experiment, and any parameter in the effective Lagrangian can be measured.

The BSMC Lagrangian contains all physics effects that may arise in the presence of lepton-number and baryon-number violation, and any parameter in the effective Lagrangian can be related in an intuitive way to quantities observable in experiments. In such a frame, tree-level relations between the gauge couplings and the usual electroweak mixing angle, \( \alpha \), in the BSMC Lagrangian, \( \xi_{\mu\nu} \) and all kinetic terms are canonically normalized and diagonal, and where all mass terms are diagonal. Moreover, the tree-level relations between the gauge couplings and the usual electroweak input parameters \( G_F, \alpha(0), m_Z \) are the same as in the SM. In such a frame, i.e., in the BSMC Lagrangian, the coefficients of the interaction terms in the Lagrangian are related in an intuitive way to quantities observable in experiment, and any parameter in the effective Lagrangian can be measured.
In the first of these relations, \( \delta m \) denotes the shift of the W-boson mass that is possibly induced by the presence of higher-dimensional operators and that we normalize as

\[
\Delta L_{\text{mass}} = 25m^2 \frac{2g^2\sigma^2}{4} W_\mu^a W_-^\mu .
\]

An input file (that we name HiggsBasis.dat in our example) describing this part of the Higgs basis Lagrangian would be of the form

**BLOCK BASIS**

1 higgs \# basis

HiggsBasis.dat

It includes, in addition to the blocks above, the SM parameters as well as vanishing values for all other EFT coefficients. In order to export this setup to the BSMC Lagrangian, we use ROSETTA by typing in a shell.

```
./translate HiggsBasis.dat
```

ROSETTA first calculates all dependent coefficients and next generates an output file named HiggsBasis.new.dat, given in the framework of the BSMC Lagrangian. This file contains in particular values for the four \( \delta c_{\text{cw}}, c_{\text{ww}}, c_{\text{BB}} \) and \( c_{\text{zz}} \) dependent parameters, the corresponding output block being, according to Eq. (9).

**BLOCK BCxh**

1 +1.00000e-01 \# dCw

2 +1.00000e-01 \# dCz

3 +1.00000e-01 \# Cgg

4 +1.52190e-01 \# Cwa

5 +1.00000e-01 \# Cza

6 +1.00000e-01 \# Czbx

7 +3.41838e-01 \# Czbx

In addition, the HiggsBasis.new.dat file also includes extra non-vanishing parameters that are linked to the six independent parameters \( \delta c_{\text{cw}}, c_{\text{gg}}, c_{\text{bb}}, c_{\text{zt}}, c_{\text{zz}} \) and \( c_{\text{zz}} \) by gauge invariance. For instance, a di-Higgs coupling to two gluonic field strength tensors is present.

**BLOCK BChhh**

4 1.00000e-01 \# cgg2

### 3.2 The Warsaw basis

The ten interaction terms of the \( \Delta L_h \) Lagrangian introduced in Section 3.1 can be seen as generated by six independent operators of the Warsaw basis,

\[
\Delta L_h = \frac{1}{2} \left[ c_{\text{gg}} \frac{g^2}{4} \Phi^i D_\mu \Phi^i (D_\mu \Phi) + c_{\text{BB}} \frac{g^2}{4} \phi^i D_\mu \phi^i (D_\mu \phi) + c_{\text{ww}} \frac{g^2}{4} \phi^i D_\mu \phi^i (D_\mu \phi) + c_{\text{zz}} \frac{g^2}{4} \phi^i D_\mu \phi^i (D_\mu \phi) \right] + \frac{1}{2} c_{\text{gg}} \frac{g^2}{4} \phi^i D_\mu \phi^i (D_\mu \phi) + \frac{1}{2} c_{\text{gg}} \frac{g^2}{4} \phi^i D_\mu \phi^i (D_\mu \phi) .
\]

In this expression, we have introduced the Pauli matrices \( \sigma_i \), the Hermitian derivative operator,

\[
(\Phi^i D_\mu \Phi)^i - (D_\mu \Phi)^i ,
\]

the gauge-covariant derivative and the hypercharge and weak field strength tensors

\[
D_\mu \Phi = \left( \partial_\mu - \frac{i}{2} g \sigma_i W^i_\mu - \frac{i}{2} g' B_\mu \right) \Phi ,
\]

\[
W_\mu^i = \partial_\mu W^i_\nu - \partial_\nu W^i_\mu + g e_\mu W^i_\ell W^\ell_\nu ,
\]

\[
B_\mu = \partial_\mu B_\nu - \partial_\nu B_\mu .
\]

The six Wilson coefficients \( c_{\text{gg}}, c_{\text{ww}}, c_{\text{BB}}, c_{\text{zz}} \) and \( c_{\text{zt}} \) appearing in \( \Delta L_h \) are related to the ten couplings in the effective Lagrangian \( \Delta L_h \) as

\[
\delta c_{\text{cw}} = - c_{\text{H}} - 4g^2 g^2 \frac{g^2}{g^2 - g^2} c_{\text{BB}} + 4g^2 g^2 \frac{g^2}{g^2 - g^2} c_{\text{BB}} - 3g^2 + g^2 \delta v ,
\]

\[
\delta c_{\text{gg}} = - c_{\text{H}} - 3 \delta v ,
\]

\[
c_{\text{ww}} = c_{\text{GG}} ,
\]

\[
c_{\text{zz}} = c_{\text{GG}} ,
\]

\[
c_{\text{BB}} = c_{\text{GG}} ,
\]

\[
c_{\text{BB}} = c_{\text{GG}} ,
\]

\[
c_{\text{BB}} = c_{\text{GG}} ,
\]

\[
\delta v = \frac{1}{2} \left[ c_{\text{H}} \left( c_{\text{H}} \right) + c_{\text{H}} \left( c_{\text{H}} \right) \right] - \frac{1}{2} c_{\text{H}} \left( c_{\text{H}} \right) .
\]

and summarizes the dependence on the additional Warsaw basis operators

\[
\frac{i}{2} \left[ c_{\text{BB}} \left( c_{\text{BB}} \right) \right] \left[ \phi^i D_\mu \phi^i \right] + \frac{1}{2} c_{\text{BB}} \left( c_{\text{BB}} \right) \left( c_{\text{BB}} \right) .
\]

Starting from the Higgs basis example of Section 3.1 where all independent parameters are fixed to 0.1, we employ ROSETTA to invert the relations of this section and calculate the numerical values of the Warsaw basis coefficients included in \( \Delta L_h \) that would yield the same \( \Delta L_h \) Lagrangian. Typing in a shell

```
./translate HiggsBasis.dat -t warsaw
```

we obtain an output file where several non-zero EFT coefficients can be found. The numerical value of those on which we focus here can be extracted from the generated file.

\[\text{These operators contribute to the muon decay at the tree level. Taking this into account leads to a shift between the measured Fermi constant and the vacuum expectation value of the Higgs field, which motivates the notation } \delta v.\]
3.3 SILH basis

We now consider the case where all operators included in the $\Delta L_h$ Lagrangian of Section 3.1 are induced by a set of operators of the SILH basis,

\[
\Delta L_h^S = \frac{1}{2} \left( s_{GG} g_w^2 \phi^i \phi^j G_{\mu \nu}^i G_{\mu \nu}^j + s_{BB} g_w^2 \frac{1}{4} \phi^i \phi^j B_{\mu \nu} B_{\mu \nu}^j + s_{WW} g_w^2 + s_{TB} g_{\mu} g_{\nu} D_{\mu} W_i^j + s_{TB} g_{\mu} g_{\nu} D_{\mu} W_i^j B_{\mu \nu}^j + i s_{W} D_{\mu} W_i^j D_{\mu} W_i^j + s_{BB} g_{\mu} g_{\nu} D_{\mu} B_i^j + s_{TB} g_{\mu} g_{\nu} D_{\mu} B_i^j B_{\mu \nu}^j + i s_{H} D_{\mu} W_i^j + s_{TB} g_{\mu} g_{\nu} D_{\mu} B_i^j + s_{TB} g_{\mu} g_{\nu} D_{\mu} B_i^j B_{\mu \nu}^j + i s_{H} D_{\mu} W_i^j D_{\mu} W_i^j + s_{TB} g_{\mu} g_{\nu} D_{\mu} B_i^j + s_{TB} g_{\mu} g_{\nu} D_{\mu} B_i^j B_{\mu \nu}^j + i s_{H} D_{\mu} W_i^j D_{\mu} W_i^j + s_{TB} g_{\mu} g_{\nu} D_{\mu} B_i^j + s_{TB} g_{\mu} g_{\nu} D_{\mu} B_i^j B_{\mu \nu}^j \right) .
\]

The ten Wilson coefficients included in $\Delta L_h^S$ can be rewritten in terms of the eleven parameters appearing in $\Delta L_h$ as

\[
\delta c_y = - s_y - g_y^2 (s_{WW} + s_{TB} + s_{TB} + s_{TB}) - \frac{4 g_y^2}{g^2 - g_y^2} s_T,
\]

\[
\delta c_T = - s_T - 3 s_y s_T + \frac{4 g_y^2}{g^2 - g_y^2} s_T,
\]

\[
\delta c_W = s_{WW},
\]

\[
\delta c_B = - s_{WW},
\]

\[
\delta c_t = - g_y^2 (s_{WW} + s_{BB} + s_{TB} + s_{TB}) - \frac{4 g_y^2}{g^2 - g_y^2} s_{BB},
\]

\[
\delta c_T = - s_{HH} - s_{HH} - \frac{4 g_y^2}{g^2 - g_y^2} s_{TB},
\]

\[
\delta c_y = s_{BB},
\]

\[
\delta c_2 = \frac{1}{2} s_{WW} + g_y^2 (s_{WW} + s_{WB} + s_{TB} + s_{TB}) - \frac{4 g_y^2}{g^2 - g_y^2} s_{TB},
\]

\[
\delta c_3 = g_y^2 (s_{WW} + s_{WB} + s_{TB} + s_{TB}) + \frac{4 g_y^2}{g^2 - g_y^2} s_{TB},
\]

\[
\delta c_7 = s_{WW} - s_{WW} - \frac{4 g_y^2}{g^2 - g_y^2} s_{TB},
\]

\[
\delta c_8 = s_{BB},
\]

\[
\delta c_9 = \frac{1}{2} s_{WW} + g_y^2 (s_{WW} + s_{WB} + s_{TB} + s_{TB}) - \frac{4 g_y^2}{g^2 - g_y^2} s_{TB},
\]

\[
\delta c_9 = g_y^2 (s_{WW} + s_{WB} + s_{TB} + s_{TB}) + \frac{4 g_y^2}{g^2 - g_y^2} s_{TB},
\]

\[
\delta c_7 = s_{WW} - s_{WW} - \frac{4 g_y^2}{g^2 - g_y^2} s_{TB},
\]

\[
\delta c_8 = s_{BB},
\]

\[
\delta c_9 = \frac{1}{2} s_{WW} + g_y^2 (s_{WW} + s_{WB} + s_{TB} + s_{TB}) - \frac{4 g_y^2}{g^2 - g_y^2} s_{TB},
\]

\[
\delta c_9 = g_y^2 (s_{WW} + s_{WB} + s_{TB} + s_{TB}) + \frac{4 g_y^2}{g^2 - g_y^2} s_{TB},
\]

where $s_T = \frac{1}{2} (s_{TB})^2$. ROSETTA can be used to extract the numerical values of the independent SILH parameters by inverting the above relations. Adopting the benchmark scenario of Section 3.1 where all the relevant Higgs basis independent parameters have been fixed to 0.1, we type in a shell
These relations are used internally for the eHDECAY interface of ROSSETTA, which takes SILH basis input parameters assuming the MFV convention of Eq. (22). In order to consistently use eHDECAY, ROSSETTA translates these coefficients from the alternative version of the SILH basis detailed in Ref. [11] as $c_b$, etc. Although it is in principle possible to input different values for the $c_{c_MV}$, $c_{c_b}$, $c_{c_MP}$, $c_{t_MV}$ and $c_{t_MP}$ parameters (referred to in Ref. [11] as $c_b$, etc.) when running eHDECAY on its own, large deviations from non-universality in these coefficients consist of a significant departure from the MFV paradigm and should not be used for complete consistency within ROSSETTA.

4 Summary

In this paper, we have introduced the ROSSETTA package, a PYTHON program dedicated to the translation of a given EFT basis of independent operators to other viable basis choices. We have also included, in this document, technical details so that users can easily extend the program and implement their own choices of EFT operator basis.

Currently, the program allows the user to translate benchmarks designed in the Higgs, SILH and Warsaw bases into any of these three bases. In addition, the code also expresses any scenario in terms of the BSMC Lagrangian of EFT operators, a basis that has been defined from the Higgs basis after ignoring all relations among the operators that are induced by a linear realization of the electroweak symmetry. A FeynRULES implementation allows ROSSETTA to be linked to other high-energy physics tools. The relations among the different Wilson coefficients that hold in the context of the Higgs, SILH and Warsaw bases of independent operators have been implemented into ROSSETTA so that they are preserved when a setup is exported to the BSMC Lagrangian by the program. This scheme has the strength to be easily generalizable to study different setups including higher-dimensional operators beyond dimension six.

In the future, we believe that translations from one basis to another will allow for broadening the scope and the use of past calculations very relevant for precision Higgs physics. Along these lines, higher-order calculations in QCD performed in the BSMC Lagrangian [31] could be used within any given EFT language, and the renormalization group running of the Wilson coefficients, that has been calculated in the SILH basis [34] and in the Warsaw bases [3,21,22], could be exported to different bases too.

Acknowledgments

The authors are grateful to Fabio Maltoni for lively interactions during all phases of this project and would also like to thank Christophe Grojean, Alex Pomarol and Michael Trott for useful comments. This work has been partially supported by the FP7 Marie Curie Initial Training Network MCnetITN (PITN-GA-2012-315877), by the Belgian Federal Science Policy Office through the Interuniversity Attraction Pole P7/37, by the Strategic Research Program 'High Energy Physics' and the Research Council of the Vrije Universiteit Brussel, by the Swiss National Science Foundation under the Ambizione grant PZ00P2 136932, and by the Theory-LHC-France initiative of the CNRS (INP and IN2P3).

References

1. W. Buchmuller and D. Wyler, Effective Lagrangian Analysis of New Interactions and Flavor Conservation, Nucl.Phys. B268 (1986) 621.

2. B. Grzadkowski, M. Iskrzynski, M. Misiak, and J. Rosiek, Dimension-Six Terms in the Standard Model Lagrangian, JHEP 1010 (2010) 085, arXiv:1008.4884.

3. R. Alonso, E. E. Jenkins, A. V. Manohar, and M. Trott, Renormalization Group Evolution of the Standard Model Dimension Six Operators III: Gauge Coupling Dependence and Phenomenology, JHEP 1404 (2014) 159, arXiv:1312.2014.

4. A. Pomarol and F. Riva, Towards the Ultimate SM Fit to Close in on Higgs Physics, JHEP 1401 (2014) 151, arXiv:1308.2803.

5. J. Ellis, V. Sanz, and T. You, Complete Higgs Sector Constraints on Dimension-6 Operators, JHEP 1407 (2014) 036, arXiv:1404.3367.

6. R. S. Gupta, A. Pomarol, and F. Riva, BSM Primary Effects, Phys.Rev. D91 (2015), no. 3 035001, arXiv:1405.0181.

7. J. Ellis, V. Sanz, and T. You, The Effective Standard Model after LHC Run I, JHEP 1503 (2015) 157, arXiv:1410.7703.

8. A. Falkowski and F. Riva, Model-independent precision constraints on dimension-6 operators, JHEP 1502 (2015) 039, arXiv:1411.0669.

9. A. Efrati, A. Falkowski, and Y. Soreq, Electroweak constraints on flavorful effective theories, JHEP 07 (2015) 018, arXiv:1503.07872.

10. G. Guidice, C. Grojean, A. Pomarol, and R. Rattazzi, The Strongly-Interacting Light Higgs, JHEP 0706 (2007) 045, hep-ph/0703164.

11. R. Contino, M. Gherardi, C. Grojean, M. Muhlleitner, and M. Spira, Effective Lagrangian for a light Higgs-like scalar, JHEP 1307 (2013) 035, arXiv:1303.3876.

12. E. Masso, An Effective Guide to Beyond the Standard Model Physics, JHEP 1410 (2014) 128, arXiv:1406.6376.

13. A. Pomarol, Higgs Physics, in 2014 European School of High-Energy Physics (ESHEP 2014) Garderen, The Netherlands, June 18-July 1, 2014, 2014, arXiv:1412.4410.

14. C. Grojean, W. Skiba, and J. Terning, Disguising the oblique parameters, Phys.Rev. D73 (2006) 075008, hep-ph/0602154.

15. Z. Han and W. Skiba, Effective theory analysis of precision electroweak data, Phys.Rev. D71 (2005) 075009, hep-ph/0412166.

16. M. Duehrssen-Debling, A. T. Mendes, A. Falkowski, and M. Riva, Effective Lagrangian for a light Higgs-like scalar, JHEP 1307 (2013) 035, arXiv:1303.3876.

17. P. Artoisenet, P. de Aquino, F. Demartin, R. Frederix, S. Frixione, et al., A framework for Higgs characterisation, JHEP 1311 (2013) 043, arXiv:1306.6464.
18. A. Alloul, N. D. Christensen, C. Degrande, C. Duhr, and B. Fuks, *FeynRules 2.0 - A complete toolbox for tree-level phenomenology*, Comput. Phys. Commun. **185** (2014) 2250–2300, [arXiv:1310.1921](http://arxiv.org/abs/1310.1921).

19. A. Alloul, B. Fuks, and V. Sanz, *Phenomenology of the Higgs Effective Lagrangian via FEYNRULES*, JHEP **1404** (2014) 110, [arXiv:1310.5150](http://arxiv.org/abs/1310.5150).

20. C. Degrande, B. Fuks, K. Mawatari, K. Mimasu, and V. Sanz, in preparation.

21. E. E. Jenkins, A. V. Manohar, and M. Trott, *Renormalization Group Evolution of the Standard Model Dimension Six Operators I: Formalism and lambda Dependence*, JHEP **10** (2013) 087, [arXiv:1308.2627](http://arxiv.org/abs/1308.2627).

22. E. E. Jenkins, A. V. Manohar, and M. Trott, *Renormalization Group Evolution of the Standard Model Dimension Six Operators II: Yukawa Dependence*, JHEP **01** (2014) 035, [arXiv:1310.4838](http://arxiv.org/abs/1310.4838).

23. B. Henning, X. Lu, and H. Murayama, *How to use the Standard Model effective field theory*, [arXiv:1412.1837](http://arxiv.org/abs/1412.1837).

24. [http://feynrules.irmp.ucl.ac.be/wiki/BSMCharacterisation](http://feynrules.irmp.ucl.ac.be/wiki/BSMCharacterisation).

25. N. D. Christensen, P. de Aquino, C. Degrande, C. Duhr, B. Fuks, M. Herquet, F. Maltoni, and S. Schumann, *A Comprehensive approach to new physics simulations*, Eur. Phys. J. **C71** (2011) 1541, [arXiv:0906.2474](http://arxiv.org/abs/0906.2474).

26. C. Degrande, C. Duhr, B. Fuks, D. Grellscheid, O. Mattelaer, et al., *UFO - The Universal FeynRules Output*, Comput. Phys. Commun. **183** (2012) 1201–1214, [arXiv:1108.2040](http://arxiv.org/abs/1108.2040).

27. P. Z. Slands, B. Allanach, H. Baer, C. Balazs, G. Belanger, et al., *SUSY Les Houches accord: Interfacing SUSY spectrum calculators, decay packages, and event generators*, JHEP **0407** (2004) 036, [hep-ph/0311123](http://arxiv.org/abs/hep-ph/0311123).

28. B. C. Allanach et al., *SUSY Les Houches Accord 2*, Comput. Phys. Commun. **180** (2009) 8–25, [arXiv:0801.0045](http://arxiv.org/abs/0801.0045).

29. **Particle Data Group** Collaboration, K. Olive et al., *Review of Particle Physics*, Chin. Phys. **C38** (2014) 090001.

30. G. D’Ambrosio, G. F. Giudice, G. Isidori, and A. Strumia, *Minimal flavor violation: An Effective field theory approach*, Nucl. Phys. **B645** (2002) 155–187, [hep-ph/0207036](http://arxiv.org/abs/hep-ph/0207036).

31. F. Maltoni, K. Mawatari, and M. Zaro, *Higgs characterisation via vector-boson fusion and associated production: NLO and parton-shower effects*, Eur. Phys. J. **C74** (2014), no. 1 2710, [arXiv:1311.1829](http://arxiv.org/abs/1311.1829).

32. F. Demartin, F. Maltoni, K. Mawatari, B. Page, and M. Zaro, *Higgs characterisation at NLO in QCD: CP properties of the top-quark Yukawa interaction*, Eur. Phys. J. **C74** (2014), no. 9 3065, [arXiv:1407.5089](http://arxiv.org/abs/1407.5089).

33. F. Demartin, F. Maltoni, K. Mawatari, and M. Zaro, *Higgs production in association with a single top quark at the LHC*, Eur. Phys. J. **C75** (2015), no. 6 267, [arXiv:1504.00611](http://arxiv.org/abs/1504.00611).

34. J. Elias-Mir, J. R. Espinosa, E. Masso, and A. Pomarol, *Renormalization of dimension-six operators relevant for the Higgs decays h → γγ,γZ*, JHEP **08** (2013) 033, [arXiv:1302.5661](http://arxiv.org/abs/1302.5661).

35. J. Elias-Miro, J. R. Espinosa, E. Masso, and A. Pomarol, *Higgs windows to new physics through d=6 operators.*