MITP/20-061

DsixTools 2.0

The Effective Field Theory Toolkit

Javier Fuentes-Martín[a],

Pedro Ruiz-Femenía[b],

Avelino Vicente[c],

Javier Virto[d]

[a] *PRISMA+ Cluster of Excellence & Mainz Institute for Theoretical Physics, Johannes Gutenberg University, 55099 Mainz, Germany,*

[b] *Departament de Matemàtiques per a l'Economia i l'Empresa, Universitat de València, E-46022 València, Spain*

[c] *Instituto de Física Corpuscular and Departament de Física Teòrica, Universitat de València - CSIC, E-46071 València, Spain*

[d] *Departament de Física Quàntica i Astrofísica and ICCUB, Universitat de Barcelona, 08028 Barcelona, Catalunya*

Abstract

`DsixTools` is a `Mathematica` package for the handling of the Standard Model Effective Field Theory (SMEFT) and the Low-energy Effective Field Theory (LEFT) with operators up to dimension six, both at the algebraic and numerical level. `DsixTools` contains a visually accessible and operationally convenient repository of all operators and parameters of the SMEFT and the LEFT. This repository also provides information concerning symmetry categories and number of degrees of freedom, and routines that allow to implement this information on global expressions (such as decay amplitudes and cross-sections). `DsixTools` also performs weak basis transformations, and implements the full one-loop Renormalization Group Evolution in both EFTs (with SM beta functions up to five loops in QCD), and the full one-loop SMEFT-LEFT matching at the electroweak scale.

Contents

1 Introduction

The experimental success of the Standard Model (SM) of particle physics and the absence of new physics (NP) signals after LHC run 2, seem to indicate the presence of a mass gap between the Electroweak (EW) scale and the scale of potential new dynamics. If this is the case, non-standard effects in processes at energy scales much smaller than the scale of NP can be described within Effective Field Theory (EFT).

Above the EW scale, the relevant EFT which contains the SM as the low-energy limit is called the Standard Model EFT (SMEFT). The SMEFT accounts for the effect of unknown heavy degrees of freedom by extending the SM Lagrangian with higher-dimensional operators invariant under the SM gauge group. The dominant NP contributions to most of the processes of phenomenological interest are then parametrized by Wilson Coefficients (WCs) of SMEFT operators of canonical dimension five and six [?].

Below the EW scale, heavy SM particles (massive bosons and the top quark) also decouple, and the dynamics is described by the Low-Energy EFT (LEFT). This theory consists of the QCD and QED Lagrangians for the light SM fermions complemented with a set of higher-dimensional operators compatible with the gauge symmetries of QED and QCD. The Wilson coefficients of these higher dimensional operators encode all the physics related to heavy SM states and the NP degrees of freedom, dominated again by operators of canonical dimension five and six [?]. The LEFT is more general than the SMEFT since it is still the correct low-energy EFT when there are new particles at the EW scale. However, under the SMEFT hypothesis, one can define the LEFT (fix its WCs) by doing a matching to the SMEFT at the EW scale.

The basis for automation of calculations within these two EFTs arises from work done within the last decade. First, a complete non-redundant operator basis for the SMEFT up to dimension six was derived in Ref. [?] (aka the *Warsaw basis*). The complete set of one-loop anomalous dimensions of the operators in the Warsaw basis was then calculated in a series of papers [? ? ? ?]. Similarly, a complete and non-redundant basis for the LEFT up to dimension six was constructed in Ref. [?] (aka the *San Diego basis*), and the full one-loop anomalous dimensions were calculated in Ref. [?]. Finally, the tree-level and one-loop matching between the LEFT and the SMEFT was performed in Refs. [?] and [?], respectively.

These advances, together with simultaneous theoretical developments occurring in the field (such as the matching of specific models to the SMEFT at one loop [? ? ? ? ? ? ? ? ? ? ?], or the automation of calculations by means of several computer tools [? ? ? ? ? ? ? ? ? ? ? ? ?]), pave the way to the systematic use of EFT methods in the analysis of new physics models. The power of the this approach is that it allows to relate physics at disparate energy scales, in our case properties of the high-energy dynamics at the new physics scale Λ_{UV}, with measurements that take place at low energies, while performing an expansion in $1/\Lambda_{\text{UV}}$ that allows to keep leading new physics effects in a consistent manner.

The Mathematica[1] package DsixTools [?] was developed as a tool to implement such automated calculations. Since the first release of DsixTools in 2017, further development of the package has occurred in two directions: 1) implementation of new theory results (such as moving from the WET [?] to the LEFT, and the implementation of higher-order effects), and 2) improvements and refinements at the front-end and operational levels (new routines, input, documentation, faster methods for RG evolution, and notation). The result of these developments is the new release DsixTools 2.0, which is available at

https://dsixtools.github.io

This paper presents a description of the program and its new features.

2 DsixTools in a nutshell

2.1 Overview of DsixTools 2.0

DsixTools is a Mathematica package for analytical and numerical computations within the SMEFT and the LEFT. It features routines devoted to RGE running (in the SMEFT and in the LEFT), matching between the two theories, basis transformation, input reading (with consistency checks) and output generation. DsixTools also contains a comprehensive and pedagogical repository with routines that allow the user to display lists of operators with certain properties, and information on WCs in the SMEFT and the LEFT.

The current version of DsixTools (DsixTools 2.0) fully implements the one-loop SMEFT RGEs, the complete one-loop matching between the SMEFT and the LEFT, and the one-loop LEFT RGEs, all up to operators of canonical dimension six. In what concerns the SMEFT RGE running, DsixTools contains:

- Three-loop SM RGEs from Refs. [? ? ? ?], as well as five-loop QCD corrections to the running of the strong gauge coupling and quark Yukawa couplings from Refs. [? ? ?].[2]

- One-loop RGEs for the dimension-six operators in the Warsaw basis from Refs. [? ? ?].[3]

- One-loop RGEs for the dimension-six baryon-number-violating operators from Ref. [?].

- One-loop RGE for the dimension-five lepton-number-violating operator from Ref. [?].

[1] Mathematica is a product from Wolfram Research, Inc. [?].

[2] The one- and two-loop SM RGEs were computed in [? ? ?] and [?], respectively.

[3] We have taken into account the errata published in http://einstein.ucsd.edu/smeft/.

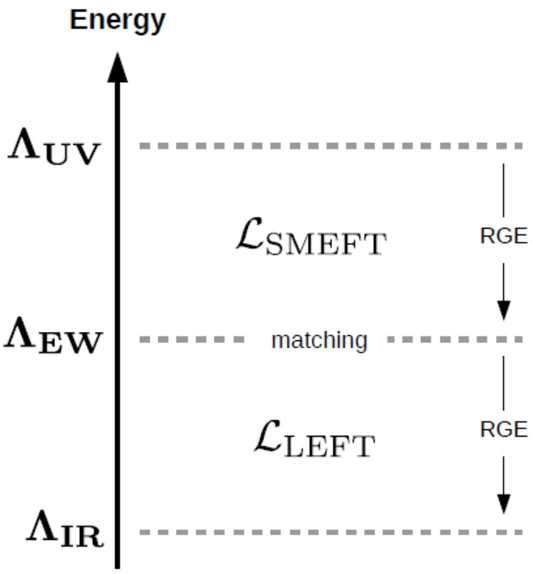


Figure 1: Scketch of the `DsixTools` matching-running routine. The `DsixTools` terminology is: Λ_{UV}=`HIGHSCALE`, Λ_{EW}=`EWSCALE` and Λ_{IR} = `LOWSCALE`. The default is `EWSCALE`=M_Z = 91.1876 GeV.

Regarding the SMEFT-LEFT matching, `DsixTools` implements:

- The tree-level matching of the SMEFT Warsaw basis to the LEFT San Diego basis at the electroweak scale, using the results of Ref. [?]. We have independently derived the matching relations (in two different ways), finding full agreement.

- The complete one-loop matching of the SMEFT Warsaw basis to the LEFT San Diego basis at the electroweak scale, using the results of Ref. [?].

Finally, `DsixTools` also implements several results related to the RGE running in the LEFT:

- Four-loop QCD corrections to the strong coupling beta function and quark mass anomalous dimensions from Ref. [?].

- One-loop RGEs for all LEFT operators up to dimension six in the San Diego basis from Ref. [?].

The structure of `DsixTools` is illustrated in Fig. 1, where one can also see how they relate to the different energy ranges and effective theories. Relevant details of the SMEFT and LEFT implementations are given in Appendices A–C, where our conventions are also presented.

2.2 Differences with `DsixTools` 1.0

The list of improvements and changes that features the new version with respect to the original version published in 2017 is substantial, and programs written with `DsixTools` 1.0 will most likely not work with `DsixTools` 2.0. Thus we collect here a summary of the most relevant changes:

- `DsixTools` 2.0 is now very easy to install, directly within `Mathematica`. See Section 3.
- The notation for WCs has changed such that now they are dimensionful. For example the SMEFT Lagrangian is given by:

  $$\mathcal{L}_{\text{SMEFT}} = \mathcal{L}_{\text{SM}}^{(4)} + \sum_k C_k^{(5)} Q_k^{(5)} + \sum_k C_k^{(6)} Q_k^{(6)} + \mathcal{O}\left(\frac{1}{\Lambda_{\text{UV}}^3}\right) , \tag{2.1}$$

  with $C_k^{(5)} \sim \Lambda_{\text{UV}}^{-1}$ and $C_k^{(6)} \sim \Lambda_{\text{UV}}^{-2}$. Same principle applies also to the LEFT WCs.
- The WET [?] basis has been superseded by the LEFT, in order to implement all the new results derived within the latter.
- Nomenclature for operators and Wilson coefficients has been modified, mainly for global convenience and consistency, and in part to make it closer to more common standards (*e.g.* WCxf [?] or FeynRules [?]).

  First, all operators in the SMEFT start with `Q` (*e.g.* $Q_{\phi\ell}^{(3)}$ = `QHl3`) while the ones in the LEFT start with `O` (*e.g.* $\mathcal{O}_{ud}^{(V8,LL)}$ = `OudV8LL`)

  Second, Wilson coefficients in the SMEFT start with `C` (*e.g.* $[C_{\phi\ell}^{(3)}]_{12}$ = `CHl3[1, 2]`) while the ones in the LEFT start with `L` (*e.g.* $[L_{ud}^{(V8,LL)}]_{1213}$ = `LudV8LL[1, 2, 1, 3]`). In `DsixTools` 1.0, flavor matrices were specified as `WC[`*name*`]`, where *name* was not the same as the name of the Wilson coefficient (*e.g.* `WC[`φ`l3]` vs. φ`L3[1,2]`). Flavor matrices in `DsixTools` 2.0 have the same name as the WCs but with an '`M`' in front, *e.g.*

  $$\begin{aligned} \texttt{MCHl3} &= \{\{[C_{\phi\ell}^{(3)}]_{1,1}, [C_{\phi\ell}^{(3)}]_{1,2}, [C_{\phi\ell}^{(3)}]_{1,3}\}, \cdots\} , \\ \texttt{MLudV8LL} &= \{\{\{\{[L_{ud}^{(V8,LL)}]_{1111}, [L_{ud}^{(V8,LL)}]_{1112}, \cdots\}, \cdots\}\}\} . \end{aligned}$$

  In addition, characters that are not trivially easy to type in `Mathematica` have been avoided (*e.g.* φ`L3[1,2]` → `CHl3[1, 2]` or $\varphi\Box$ → `CHbox`).
- Besides the two options to solve the RGEs avaliable in `DsixTools` 1.0 (exact numerical solution and leading logarithm), `DsixTools` 2.0 includes a third method, as the default setting. This method employs the Evolution Matrix approach, described in Appendix D. This method is numerically very precise and it is computationally faster than solving the RGEs exactly.

- Many of the routines inherited from DsixTools 1.0 have changed names. For example, all routines related to the SMEFT now start with SMEFT... and similarly for the LEFT (*e.g.* SMEFTRunEGEs and LEFTRunRGEs), which makes it easier to use Mathematica's autocompletion feature. In addition, some routines in DsixTools 1.0 have been eliminated (or replaced by improved ones), and new routines have been implemented. See Section 4.2 for the complete list of routines in DsixTools 2.0.

- DsixTools 2.0 incorporates a reference repository of information about the SMEFT and the LEFT accessible through the routines SMEFTObjectList and LEFTObjectList, SMEFTOperators and LEFTOperators, SMEFTParameterList and LEFTParameterList, ObjectInfo, SMEFTOperatorsMenu and LEFTOperatorsMenu, SMEFTOperatorsGrid and LEFTOperatorsGrid, and NIndependent. In addition, DsixTools 2.0 contains a full Mathematica documentation system.

- Setting the input values for the Wilson coefficients in the SMEFT or the LEFT through NewInput[...], ChangeInput[...] or ReadInputFiles[...] now checks the consistency of the given input, printing warnings when necessary. The same is done when setting scales through NewScale[...]. The input in DsixTools 2.0 is *basis-independent*. See Section 4.3 for details. The user can also check the input values for the WCs at any time using the routines InputValues, SMEFTLagrangian[HIGHSCALE] or LEFTLagrangian[EWSCALE].

- DsixTools 2.0 includes higher order corrections to matching coefficients and RG coefficients as compared to DsixTools 1.0. In particular it includes SM beta functions up to five loops, and LEFT matching conditions in the SMEFT at one loop.

3 Downloading, installing and loading DsixTools

DsixTools is free software under the copyright of the GNU General Public License. There are two ways to download the package and install it:

Automatic installation

The simplest way to download and install DsixTools is to run the following command in a Mathematica session:

```
Import["https://raw.githubusercontent.com/DsixTools/DsixTools ↪
    /master/install.m"];
```

This will download and install DsixTools in the *Applications* folder of the Mathematica base directory, activate the documentation and load the package. During the installation process, a pop up window will appear asking if you want to convert the .m files to .mx format. This option is recommended, since it significantly reduces the DsixTools loading time.

Manual installation

Alternatively, the user can also download and install DsixTools manually. The package can be downloaded from the web page [?]:

https://dsixtools.github.io

We recommend placing the DsixTools folder inside the *Applications* folder of Mathematica's base directory, after which loading the package will be automatic. Alternatively, the user can place the DsixTools folder in a different directory. In this case, loading the package will require specifying previously its location via

```
pathtoDsixTools = "<directory>";
AppendTo[$Path, pathtoDsixTools];
```

As a final step, the user can activate the documentation by moving the contents of the zip file Documentation.zip inside the DsixTools folder, and applying

```
If[$VersionNumber>=12.1,PacletDataRebuild[],RebuildPacletData[]];
```

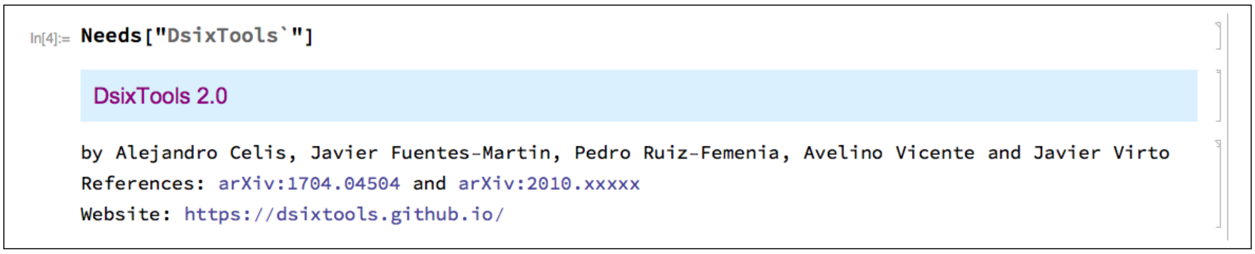
inside a Mathematica notebook.

Loading DsixTools

Once installed, the user can load DsixTools anytime with the command

```
Needs["DsixTools‘"]
```

When DsixTools is loaded, a message is printed out with information about the version, the authors, and links to the relevant references and to the DsixTools website:



```
In[4]:= Needs["DsixTools`"]

DsixTools 2.0

by Alejandro Celis, Javier Fuentes-Martin, Pedro Ruiz-Femenia, Avelino Vicente and Javier Virto
References: arXiv:1704.04504 and arXiv:2010.xxxxx
Website: https://dsixtools.github.io/
```

A typical loading time is about 5-10 s depending on the machine, if the .m to .mx conversion is done. When DsixTools is loaded, several (relatively heavy) Mathematica files containing SMEFT and LEFT beta functions, RGEs and evolution matrices, as well as the SMEFT-LEFT one-loop matching relations are loaded as well. This may be unnecessary for some

DsixTools applications. In this case the user can force DsixTools to load without importing such files, by evaluating the line

```
DsixTools‘ImportFiles = False;
```

before loading DsixTools. This will reduce the loading time to under a second. If running or matching is required after loading DsixTools in this mode, the corresponding files can be loaded by the user a posteriori, there is no need to reload DsixTools.

4 Using DsixTools

In this Section we describe how to use DsixTools in detail. After summarizing the DsixTools routines and functions, the main features of the package will be explained with specific examples of use.

4.1 A DsixTools program

The following is a simple but complete DsixTools program which takes input from the user for the SMEFT Lagrangian at the UV scale $\Lambda_{\text{UV}} = \texttt{HIGHSCALE}$ and calculates the LEFT WCs at the IR scale $\Lambda_{\text{IR}} = \texttt{LOWSCALE}$, printing out one specific WC for illustration:

```
Needs["DsixTools’"]

NewScale[{HIGHSCALE -> 10000}];

NewInput[{Clq1[1,1,1,2] -> 1/HIGHSCALE^2, Clq1[1,1,2,1] -> ↪
    1/HIGHSCALE^2, CH -> -0.5/HIGHSCALE^2}];

RunDsixTools;

D6run[LeuVLL[2,2,1,1]] /. \[Mu] -> LOWSCALE
```

The program begins by loading DsixTools, as explained in Sec. 3. In the next line we provide the numerical value for the global variable HIGHSCALE which corresponds to Λ_{UV}

$$\texttt{HIGHSCALE} = \Lambda_{\text{UV}} = 10\,\text{TeV}\,.$$

In DsixTools **all scales are given in GeV**. The third line defines the input values by means of the NewInput DsixTools routine. In this case the user is *implicitly* specifying that the input WCs correspond to the SMEFT, and these take the values

$$\left[C_{\ell q}^{(1)}\right]_{1112} = \left[C_{\ell q}^{(1)}\right]_{1121} = \frac{1}{\Lambda_{\text{UV}}^2} = 10^{-8}\ \text{GeV}^{-2}\ , \quad C_{\varphi} = -\frac{0.5}{\Lambda_{\text{UV}}^2} = -5 \cdot 10^{-9}\ \text{GeV}^{-2}\,, \quad (4.1)$$

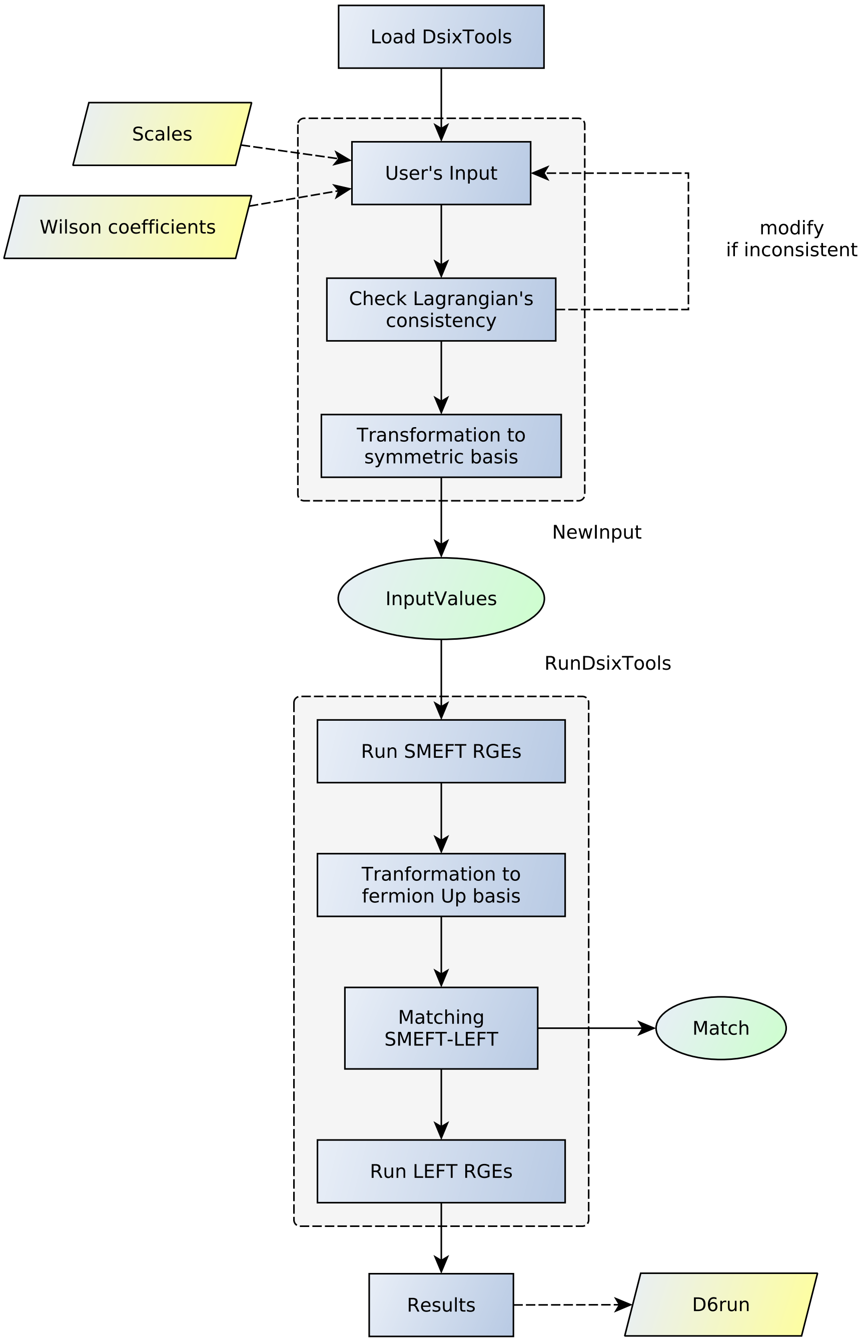


Figure 2: Example of a minimal `DsixTools` program flowchart.

at the new physics scale $\Lambda_{\rm UV} = 10$ TeV, with all the other WCs set to zero. We note that $[C_{\ell q}^{(1)}]_{1112} = [C_{\ell q}^{(1)}]_{1121}$ follows from the hermiticity of the Lagrangian, which implies the general relation $[C_{\ell q}^{(1)}]_{aabc} = [C_{\ell q}^{(1)}]^*_{aacb}$. If this condition were not respected by the arguments of the NewInput routine, a message would be issued by DsixTools and a modification of the input values in order to restore consistency would be applied (see Section 4.3). In the next line, the program makes use of the RunDsixTools routine. This can be regarded as the *master* DsixTools routine, since it performs the three main tasks this package is designed for: it runs the SMEFT parameters from $\Lambda_{\rm UV}$ = HIGHSCALE to $\Lambda_{\rm EW}$ = EWSCALE, matches to the LEFT, and finally runs the LEFT parameters from $\Lambda_{\rm EW}$ = EWSCALE to $\Lambda_{\rm IR}$ = LOWSCALE. The variable LOWSCALE takes the default value LOWSCALE$= 5\,$GeV. After evaluating RunDsixTools, the D6run function becomes available. The last line of the program precisely reads these results by printing the value of the LEFT WC $[L_{eu}^{V,LL}]_{2211}$ at $\mu = \Lambda_{\rm IR} = 5$ GeV, obtaining a numerical result

$$\left[L_{eu}^{V,LL}\right]_{2211} \simeq 6.22 \cdot 10^{-6}\ \text{GeV}^{-2}\,.$$

The general flowchart of this minimal program can be seen in Fig. 2. It clearly involves most of the main routines of DsixTools and serves as an example of use in a practical scenario. However, some of the functionalities used in this program offer alternative possibilities and methods of application. For this reason, in the rest of the paper we explain in greater detail how to take full advantage of DsixTools.

4.2 Summary of DsixTools routines

Once the package has been loaded, the user can already execute all DsixTools functions and routines. Several DsixTools global variables are also introduced at this stage. Here we summarize the DsixTools routines available to the user.

General variables and routines

- DsixToolsVersion: Returns the loaded version of DsixTools.
- DsixToolsDir: Returns the directory holding the loaded version of DsixTools.
- HIGHSCALE: UV scale (in units of GeV) at which the SMEFT input is set and where the running in the SMEFT starts.
- EWSCALE: Electroweak scale (in units of GeV). This is the scale at which the LEFT input is set (either directly or through matching with the SMEFT), and where the running in the LEFT starts. By default EWSCALE = 91.1876 GeV, but it can be modified by means of NewScale or NewInput.

- LOWSCALE: IR scale (in units of GeV) which sets the lower limit beyond which the solution of the LEFT RGEs are only extrapolations. Since the LEFT in DsixTools 2.0 is the five-flavor theory, the default DsixTools value is LOWSCALE $= 5$.

- RunDsixTools: Master DsixTools routine. It runs the SMEFT parameters from Λ_{UV} = HIGHSCALE to Λ_{EW} = EWSCALE, matches to the LEFT, and then runs the LEFT parameters from Λ_{EW} to Λ_{IR} = LOWSCALE.

Reference

- SMEFTObjectList and LEFTObjectList: List of SMEFT and LEFT *objects*, where an *object* is defined as a list of properties of a SMEFT or LEFT operator and its Wilson coefficients.

- SMEFTOperators and LEFTOperators: List of all SMEFT and LEFT operators in DsixTools notation.

- SMEFTParametersTotal and LEFTParametersTotal: List of all SMEFT and LEFT parameters (i.e. couplings, mass parameters and WCs) in DsixTools notation.

- SMEFTParameterList[<*attributes*>] and LEFTParameterList[<*attributes*>]: List of all *independent* SMEFT/LEFT parameters satisfying the condition given by *attributes*. The sequence of *attributes* can be chosen from the following predefined lists in the cases of the SMEFT and the LEFT, respectively:

$$\begin{aligned}\text{SMEFT} \;:\; &\{\text{“}\mathtt{SM}\text{”}, \text{“}\mathtt{D6}\text{”}, \text{“}\mathtt{D5}\text{”}, \text{“}\mathtt{D4}\text{”}, \text{“}\mathtt{D2}\text{”}, \text{“}\mathtt{BNV}\text{”}, \text{“}\mathtt{BNC}\text{”}, \text{“}\mathtt{LNV}\text{”}, \text{“}\mathtt{LNC}\text{”}, \text{“}\mathtt{CPodd}\text{”},\\ &\text{“}\mathtt{CPeven}\text{”}, \text{“}\mathtt{X3}\text{”}, \text{“}\mathtt{H6}\text{”}, \text{“}\mathtt{H4D2}\text{”}, \text{“}\mathtt{X2H2}\text{”}, \text{“}\mathtt{F2H3}\text{”}, \text{“}\mathtt{F2XH}\text{”}, \text{“}\mathtt{F2H2D}\text{”}, \text{“}\mathtt{LLLL}\text{”},\\ &\text{“}\mathtt{RRRR}\text{”}, \text{“}\mathtt{LLRR}\text{”}, \text{“}\mathtt{LRLR}\text{”}, \text{“}\mathtt{LRRL}\text{”}, \text{“}\mathtt{B}-\mathtt{violating}\text{”}, \text{“}\mathtt{L}-\mathtt{violating}\text{”},\\ &\text{“}\mathtt{LFV}\text{”}, \text{“}\mathtt{QFV}\text{”}, \text{“}\mathtt{0F}\text{”}, \text{“}\mathtt{2F}\text{”}, \text{“}\mathtt{4F}\text{”}\}\\ \text{LEFT} \;:\; &\{\text{“}\mathtt{QED\&QCD}\text{”}, \text{“}\mathtt{D6}\text{”}, \text{“}\mathtt{D5}\text{”}, \text{“}\mathtt{D4}\text{”}, \text{“}\mathtt{D3}\text{”}, \text{“}\mathtt{D2}\text{”}, \text{“}\mathtt{BNV}\text{”}, \text{“}\mathtt{BNC}\text{”}, \text{“}\mathtt{LNV}\text{”}, \text{“}\mathtt{LNC}\text{”},\\ &\text{“}\mathtt{CPodd}\text{”}, \text{“}\mathtt{CPeven}\text{”}, \text{“}\mathtt{X3}\text{”}, \text{“}\nu\nu\mathtt{X}\text{”}, \text{“}\mathtt{LRX}\text{”}, \text{“}\mathtt{LLLL}\text{”}, \text{“}\mathtt{RRRR}\text{”}, \text{“}\mathtt{LLRR}\text{”}, \text{“}\mathtt{LRLR}\text{”},\\ &\text{“}\mathtt{LRRL}\text{”}, \text{“}\Delta\mathtt{L}=\mathtt{4}\text{”}, \text{“}\Delta\mathtt{L}=\mathtt{2}\text{”}, \text{“}\Delta\mathtt{B}=\Delta\mathtt{L}=\mathtt{1}\text{”}, \text{“}\Delta\mathtt{B}=-\Delta\mathtt{L}=\mathtt{1}\text{”},\\ &\text{“}\mathtt{LFV}\text{”}, \text{“}\mathtt{QFV}\text{”}, \text{“}\mathtt{0F}\text{”}, \text{“}\mathtt{2F}\text{”}, \text{“}\mathtt{4F}\text{”}\}\end{aligned}$$

  SMEFTParameterList[] and LEFTParameterList[] list all independent SMEFT and LEFT parameters.

- SMEFTFindParameter[<*attributes*>,*parameter*] and LEFTFindParameter[<*attributes*>,*parameter*]: Returns the position of *parameter* in the list SMEFTParameterList[] or LEFTParameterList[]. If the optional entry *attributes* is given, the position refers to the corresponding restricted list.

- `ObjectInfo[`*parameter*`]`: Prints information about *parameter*.

- `SMEFTOperatorsMenu`, `LEFTOperatorsMenu` and `TotalOperatorsMenu`: Displays a clickable menu with information about the operators and parameters of the SMEFT and the LEFT.

- `SMEFTOperatorsGrid` and `LEFTOperatorsGrid`: Creates a grid with all the SMEFT or LEFT operators. Moving the mouse on top of each entry displays the definition of the operator, and clicking on it a window with information about it is displayed.

- `SMEFTLagrangian[`*scale*`]` and `LEFTLagrangian[`*scale*`]`: Returns the SMEFT or LEFT Lagrangians at the renormalization scale given in the argument, corresponding to the current values given by `InputValues`. If necessary, running and/or matching is performed internally.

Input & Output

- `TurnOnMessages` and `TurnOffMessages`: Turn on or off the messages written by the `DsixTools` routines.

- `NewScale[{`*list*`}]`: Sets (or resets) the values of the scales indicated in *list*. For example, if *list* = {`HIGHSCALE` → 5000, `LOWSCALE` → 5} will set (or reset) $\Lambda_{\mathrm{UV}} = 5$ TeV and $\Lambda_{\mathrm{IR}} = 5$ GeV.

- `InputValues`: Dispatch[4] that contains the current values of the Wilson coefficients at the input scale. It might refer to the SMEFT or the LEFT, depending on the last input defined by the user.

- `InputBasis`: Indicates the SMEFT flavor basis of the input in `InputValues`. It can be ‘‘`up`’’ or ‘‘`down`’’ (default).

- `NewInput[{`*list*`},<`*additional*`>]`: Resets the variable `InputValues` putting to zero all $d > 4$ WCs, and then replaces it by a new one in which the changes in *list* are applied. The optional *additional* entries may also contain changes in the scales `HIGHSCALE`, `EWSCALE` and `LOWSCALE`, as well as in `InputBasis`, e.g.,

  `NewInput[{`*list*`},HIGHSCALE->5000,InputBasis->"up"]`.

[4] For those not familiar with `Mathematica` dispatch tables, we clarify that these are lists (or tables) of pointers to replacements rules. In practice they work in exactly the same way as replacement rules, but their execution time is much lower when the list of replacements is long. It is possible to recover a normal replacement rule from the dispatch by applying to it the `Mathematica` command `Normal`.

- ChangeInput[{*list*}]: Replaces (without resetting) the current dispatch InputValues by a new one in which the changes in *list* are applied. s

- SetSMLEFTInput: Resets the variable InputValues with the LEFT coefficients obtained from a matching to the SM.

- ReadInputFiles[options_file,{WCsInput_file},{SMInput_file},{EFT}]: Reads all input files.

- WCXFtoSLHA[WCXF_file,SLHA_file,EFT]: Translates the WCs file in WCxf format WCXF_file into an SLHA format file named SLHA_file.

- SLHAtoWCXF[SLHA_file,WCXF_file,SCALE,EFT]: Translates the WCs file in SLHA format SLHA_file into an WCxf format file named WCXF_file.

- AntisymmetryErrorsTotal: List containing the accumulated set of errors fixed by NewInput, ChangeInput or ReadInputFiles due to input not consistent with flavor-index symmetries.

- NonHermitianErrorsTotal: List containing the accumulated set of errors fixed by NewInput, ChangeInput or ReadInputFiles due to input not consistent with hermiticity of the Lagrangian.

Operations with Wilson coefficients

- D6Simplify[*expression*]: Replaces all redundant Wilson coefficients in *expression* by their expressions in terms of the non-redundant ones. It also eliminates complex conjugates on real parameters.

- SubRedundant: Dispatch that replaces all redundant Wilson coefficients by their expressions in terms of the independent ones present in SMEFTParametersList[] and LEFTParametersList[].

- NIndependent[*parameter*]: Returns the number of independent real parameters in *parameter*: 2 for a general complex parameter, 1 for a real parameter, and 0 for a redundant WC.

- ToSymmetric[X,*cat*]: Returns X in the symmetric basis, where X is an object of category *cat* in array form.

- ToSymmetricSingle[*parameter*]: Returns the form of the SMEFT or LEFT *parameter* in the symmetric basis.

- `ToIndependent[`X, cat`]`: Returns X in the independent basis, where X is an object of category *cat* in array form.

- `ToIndependentSingle[`*parameter*`]`: Returns the form of the SMEFT or LEFT *parameter* in the independent basis.

- `CheckAndSymmetrize[`X, cat`]`: Returns X in the symmetric basis, where X is an object of category *cat* in array form, after checking that all hermiticity and antisymmetry conditions are respected. If any of the conditions are violated, some entries of X are modified.

SMEFT and LEFT running

- `RGEsMethod`: Indicates the method that `DsixTools` is going to use to solve the RGEs. It is either 1 (exact numerical solution), 2 (first leading log) or 3 (via the evolution matrix formalism). This variable is protected.

- `SetRGEsMethod[`n`]`: Sets the value of `RGEsMethod` to $n = 1, 2$ or 3.

- `SMEFTLoopOrder`: Indicates the order that `DsixTools` is going to use for the SM beta functions when running in the SMEFT. The maximum (and default) in `DsixTools` 2.0 is `SMEFTLoopOrder` $= 5$. This variable is protected.

- `LEFTLoopOrder`: Indicates the order in QCD that `DsixTools` is going to use for the strong coupling beta function and quark-mass anomalous dimensions when running in the LEFT. The maximum (and default) in `DsixTools` 2.0 is `LEFTLoopOrder` $= 4$. This variable is protected.

- `SetSMEFTLoopOrder[`n`]` and `SetLEFTLoopOrder[`n`]`: Set the values of `SMEFTLoopOrder` and `LEFTLoopOrder`

- `UseRGEsSM`: If `UseRGEsSM` $= 1$, `DsixTools` will use the pure SM RGEs to run the SM parameters to the initial scale `HIGHSCALE`.

- β`[`*parameter*`]`: Gives the beta function of the SMEFT or LEFT *parameter*.

- β`SM[`*parameter*`]`: Gives the SM beta function of the SM *parameter*.

- `SMEFTRunRGEs` and `LEFTRunRGEs`: Solve the SMEFT and LEFT RGEs in each case.

- `D6run[`*parameter*`,<"log10">]`: Gives the SMEFT or LEFT *parameter* as a function of the renormalization scale μ. Including the optional argument `"log10"` gives the function in terms of $t = \log_{10}(\mu/\text{ GeV})$.

- `SMEFTEvolve[`*parameter*`,`*final*`,`*initial*`,<"log10">]`: Returns the SMEFT *parameter* at $\mu = \mathit{final}$ in terms of the SMEFT parameters at $\mu = \mathit{initial}$ using the evolution matrix method.

- `LEFTEvolve[`*parameter*`,`*final*`,`*initial*`,<"log10">]`: Returns the LEFT *parameter* at $\mu = \mathit{final}$ in terms of the LEFT parameters at $\mu = \mathit{initial}$ using the evolution matrix method.

- `SMEFTrunnerExport[<`*format*`>,<`*name*`>]`: Exports the numerical values of the SMEFT parameters at the scale Λ_{EW} = `EWSCALE` (after running). If no argument is given, `SMEFTrunnerExport` generates a default output file named `Output_SMEFTrunner.dat`. This routine can also export the output to file with a *name* (without extension) and *format* chosen by the user (both arguments are required). The available formats are `"SLHA"` (default `DsixTools` format), `"JSON"` and `"YAML"`.

- `LEFTrunnerExport[<`*format*`>,<`*name*`>]`: Exports the numerical values of the LEFT parameters at the scale Λ_{IR} = `LOWSCALE` (after running). If no argument is given, `LEFTrunnerExport` generates a default output file named `Output_LEFTrunner.dat`. This routine can also export the output to file with a *name* (without extension) and *format* chosen by the user (both arguments are required). The available formats are `"SLHA"` (default `DsixTools` format), `"JSON"` and `"YAML"`.

Matching at the EW scale

- `MatchingLoopOrder`: Indicates if the SMEFT-LEFT matching will be done at tree-level (`MatchingLoopOrder` = 0) or at one-loop (`MatchingLoopOrder` = 1). This variable is protected.

- `SetMatchingLoopOrder[`n`]`: Sets the value of `MatchingLoopOrder` to n.

- `Match`: Dispatch that replaces all LEFT parameters by their numerical values at the matching scale, obtained after matching to the SMEFT.

- `MatchEW[`*parameter*`]`: Returns the matching condition of the LEFT *parameter* in terms of SMEFT parameters at the EW scale, in analytical form.

- `MatchAnalytical`: Dispatch that replaces all LEFT parameters by their analytical matching conditions, in terms of SMEFT parameters.

- `SMEFTLEFTMatch`: Perfoms the matching between the SMEFT and the LEFT, at the order especified by `MatchingLoopOrder`.

- `SMEFTRotateParameters[`*scale*`]`: Returns a list containing two dispatches that transform the SMEFT parameters to the ‘‘`up`’’ and ‘‘`down`’’ bases at $\mu = scale$.

- `SMEFTToNewBasis[`*basis*`,`*scale*`]`: Dispatch that transforms the SMEFT parameters to a specific flavor basis (‘‘`up`’’ or ‘‘`down`’’) at $\mu = scale$.

- `LEFTToNewBasis[`*scale*`]`: Dispatch that transforms the LEFT parameters to the mass basis at $\mu = scale$.

- `EWmatcherExport[<`*format*`>,<`*name*`>]`: Exports the numerical values of the LEFT parameters at the scale $\Lambda_{\rm EW}$ obtained after matching to the SMEFT. If no argument is given, `EWmatcherExport` generates a default output file named `Output_EWmatcher.dat`. This routine can also export the output to file with a *name* (without extension) and *format* chosen by the user (both arguments are required). The available formats are `"SLHA"` (default `DsixTools` format), `"JSON"` and `"YAML"`.

Other variables and routines

- `Biunitary[`*matrix*`]`: Applies a biunitary transformation diagonalizing the square *matrix*, and provides the rotation matrices and the eigenvalues.

- `LoopParameter`: Appears in analytical expressions such as beta functions and matching conditions, separating different loop orders. An n-loop term is proportional to (`LoopParameter`)^n (except for $n = 1$ in the beta functions). This variable is protected.

4.3 Input values in `DsixTools`

One of the first steps in every `DsixTools` program is to define the input. This includes the numerical values of the SMEFT or LEFT parameters at the input scale, the relevant scales for matching and RGE running ($\Lambda_{\rm UV}$ = `HIGHSCALE`, $\Lambda_{\rm EW}$ = `EWSCALE` and $\Lambda_{\rm IR}$ = `LOWSCALE`), and some `DsixTools` options. The input values for the SM parameters, which are used by default and in the evolution matrix method, are given in Table 1.

There are two ways of defining an input. The first way, which we call **notebook input**, is to introduce the input values directly in the `Mathematica` notebook. This is the method used in the example program shown in Section 4.1. Alternatively, the user can also set the input by reading external files containing the input values. We will refer to this approach as **external files input**. We now explain these two approaches and how to use them. For definiteness, we will concentrate on the SMEFT. For setting input in the LEFT, the steps and routines are completely analogous.

Table 1: Default DsixTools inputs for the SM parameters at the scale $M_Z = 91.1876$ GeV.

Parameter	Value
g	0.6515
g'	0.3576
g_s	1.220
λ	0.2813
m^2	$8528\ \mathrm{GeV}^2$
Γ_u	$\begin{pmatrix} 7.109\times10^{-6} & -8.175\times10^{-4} & (8.176+3.265\,i)\times10^{-3} \\ 1.636\times10^{-6} & 3.551\times10^{-3} & -4.017\times10^{-2} \\ (0.782+2.522\,i)\times10^{-8} & 1.540\times10^{-4} & 0.970 \end{pmatrix}$
Γ_d	$\mathrm{diag}(1.551\times10^{-5}, 3.165\times10^{-4}, 1.637\times10^{-2})$
Γ_e	$\mathrm{diag}(2.944\times10^{-6}, 6.071\times10^{-4}, 1.021\times10^{-2})$
$\theta_s, \theta, \theta'$	0

Notebook input

The simplest way of setting the input in DsixTools is to introduce the values directly in the Mathematica notebook. The DsixTools options and the relevant scales for the RGE running can be introduced easily. For instance,

```
UseRGEsSM = 0;
NewScale[{HIGHSCALE->10000}];
```

would set the UseRGEsSM option to 0 and the high-energy scale $\Lambda_{\mathrm{UV}} = 10$ TeV. The SMEFT or LEFT parameters (including the SM or QCD & QED inputs) can be introduced by means of the NewInput routine. This routine resets the input so that the WCs take their default values and then applies the changes indicated by the user.[5] For instance, the program of Sec. 4.1 includes the line

```
NewInput[{Clq1[1,1,1,2] -> 1/HIGHSCALE^2, Clq1[1,1,2,1] -> ↪
    1/HIGHSCALE^2, CH -> -0.5/HIGHSCALE^2}];
```

[5] The default SMEFT and LEFT values correspond to the SM and QED&QCD benchmarks, respectively, in both cases with all Wilson coefficients of Dimension-five and -six operators set to zero and default values for the coefficients of Dimension ≤ 4 operators.

which, as discussed already, sets $[C_{\ell q}^{(1)}]_{1112} = [C_{\ell q}^{(1)}]_{1121} = 1/\Lambda_{\text{UV}}^2 = 10^{-8}$ GeV^{-2} and $C_\varphi = -0.5/\Lambda_{\text{UV}}^2 = -5 \cdot 10^{-9}$ GeV^{-2}, if the new physics scale Λ_{UV} is previously set to 10 TeV. We note that only the non-vanishing WCs must be given and the rest are assumed to be zero.

As explained in Appendix C, some of the 2- and 4-fermion operators in the SMEFT and the LEFT possess specific symmetries under the exchange of flavor indices. In particular, these symmetries imply conditions to be enforced in the input WCs in order to avoid two types of inconsistencies:

1. **Hermiticity:** The hermiticity of the Lagrangian imposes certain conditions on some WCs, and these must be respected by the input provided by the user. For instance, an input with $[C_{\ell q}^{(1)}]_{1112} \neq [C_{\ell q}^{(1)}]_{1121}^*$ would be inconsistent.

2. **Antisymmetry:** Some LEFT operators are antisymmetric under the exchange of two flavor indices and thus vanish. For practical reasons, we have not excluded these operators from the WC input list, but rather require that the corresponding WCs vanishes. For instance, an input with $[L_{\nu\gamma}]_{11} \neq 0$ would be inconsistent.

In order to avoid potential issues associated to inconsistent inputs, `DsixTools` includes user-friendly input routines that simplify the user's task. `DsixTools` accepts input values for the WCs of any set of operators (belonging to the Warsaw or San Diego bases) and then checks for possible consistency problems. When the user's input is not consistent, a warning is issued and `DsixTools` corrects the input by replacing it by a new one that ensures a complete consistency of the Lagrangian. For instance, this would be case if the user runs

```
NewInput[{Clq1[1,1,1,2] -> 1/HIGHSCALE^2}];
```

since this command sets $[C_{\ell q}^{(1)}]_{1112} = 1/\Lambda_{\text{UV}}^2$ and $[C_{\ell q}^{(1)}]_{1121} = 0 \neq [C_{\ell q}^{(1)}]_{1112}^*$. The list of invalid input values can be seen by clicking on a button named `Input errors` that appears after running `NewInput`. We note however that in some cases other Wilson coefficients, related to these by the two reasons given above, are also modified. The user should therefore pay attention to these messages to make sure that the input has been correctly introduced.

Furthermore, `DsixTools` transforms all WCs to the *symmetric basis*, defined as the basis in which the WCs follow the same symmetry conditions as the associated operators. We refer to Appendix C.3 for more information about this basis. In the symmetric basis $(C_{\ell\ell})_{1122} = (C_{\ell\ell})_{2211}$ since $(Q_{\ell\ell})_{1122} = (Q_{\ell\ell})_{2211}$. This is the basis used internally by `DsixTools`. Nevertheless, the user needs not to worry about this, **since the input/output is always unambiguous**.

After defining the input values with the `NewInput` routine the dispatch `InputValues` gets initialized. This dispatch can be used to print the input value of any SMEFT or LEFT parameter. For instance, after running

```
NewInput[{Cll[2,2,3,3] -> 10^(-8)}];
```

one can evaluate

```
Cll[2,2,3,3] /. InputValues
```

and obtain the result $5\cdot 10^{-9}$ GeV^{-2}. This is the input value given with the NewInput routine to the SMEFT WC $[C_{\ell\ell}]_{2233}$, after transforming to the symmetric basis. We note that in this basis $[C_{\ell\ell}]_{2233} = [C_{\ell\ell}]_{3322}$, and due to $[Q_{\ell\ell}]_{2233} = [Q_{\ell\ell}]_{3322}$ this is equivalent to the input given by the user:

$$\begin{aligned} \text{User's input:} \quad & [C_{\ell\ell}]_{2233} = 10^{-8}\ \text{GeV}^{-2} \quad \text{and} \quad [C_{\ell\ell}]_{3322} = 0\,, \\ \text{In symmetric basis:} \quad & [C_{\ell\ell}]_{2233} = 5\cdot 10^{-9}\ \text{GeV}^{-2} \quad \text{and} \quad [C_{\ell\ell}]_{3322} = 5\cdot 10^{-9}\ \text{GeV}^{-2}\,. \end{aligned}$$

This can be clearly seen by evaluating the command

```
MCll /. InputValues
```

which prints the complete $C_{\ell\ell}$ WC in array form. The input values in the *independent* basis (see Appendix C.3) can be obtained by applying the routine ToIndependent:

```
ToIndependent[MCll,6] /. InputValues
```

which in this case results in the same input introduced before since $[C_{\ell\ell}]_{2233}$ is one of the independent WCs.

Finally, once the input values have been set, the user can change them individually at any moment in the notebook. This is done with the ChangeInput routine. In contrast to NewInput, this routine does not reset the input to default values, but just applies the changes demanded by the user. For instance,

```
ChangeInput[{CHG -> 10^(-6)}]
```

would change the value of $C_{\varphi G}$ to 10^{-6} GeV^{-2} in the current InputValues dispatch, without altering the values of the other SMEFT parameters. Exactly as NewInput, the ChangeInput routine also checks the consistency of the input Lagrangian provided by the user and then translates the 2- and 4-fermion WCs to the symmetric basis.

External files input

Alternatively, the user can set the program options and provide input values from external files. This is done with the ReadInputFiles routine. For instance,

```
ReadInputFiles["Options.dat","WCsInput.dat","SMInput.dat", ↪
    "SMEFT"]}
```

applies the content of three SMEFT input files.[6] The file Options.dat contains the option values to be used in the program, the file WCsInput.dat contains the input values for the

[6] The use of input files for the LEFT is completely analogous, the only difference being that instead of input values for the SM parameters one must provide input values for the QCD & QED parameters, and that the last option should be "LEFT" instead of "SMEFT".

SMEFT WCs at $\mu = \Lambda_{\mathrm{UV}}$, and the file SMInput.dat contains the input values for the SM parameters. Examples for all of these files (and the corresponding ones for the LEFT) can be found in the IO folder of DsixTools. Each of the entries in these files are accompanied by comments that make them self-explanatory. Similarly to the case of notebook input, the InputValues dispatch gets initialized and can be used after using ReadInputFiles.

The default DsixTools input and output format is inspired by the Supersymmetry Les Houches Accord (SLHA) [? ?]. Input files are distributed in blocks, each devoted to a set of parameters. Any complex parameter is given in two blocks, so that real and imaginary parts should be provided separately. Furthermore, WCs carrying flavor indices should be provided individually for each flavor combination. Analogously to the notebook input case, all WCs are assumed to vanish by default. Therefore, it suffices to include the non-zero WCs (and only these) in the input card. Furthermore, the routine ReadInputFiles will also check that the set of input values provided by the user is consistent. If any of the hermiticity or antisymmetry conditions on the WCs are not satisfied, a message will be issued and the corresponding input values modified in order to restore consistency.

Additionally, DsixTools can also read WCs input files in WCxf format [?], a standard data exchange format for numerical values of Wilson coefficients. In this case, the WCs input card can be a JSON or YAML file. Note however that reading YAML input files requires previous installation of a YAML importer for Mathematica [?]. For more details about the WCxf format, such as the specific fermion basis that is implicitly assumed, we refer to [?].

4.4 RGE running

Once the initial conditions at some energy scale Λ_{start} are defined, the user can apply the RGEs to obtain the resulting Lagrangian parameters at the different energy scale Λ_{end}. The SMEFT running between $\Lambda_{\mathrm{UV}} =$ HIGHSCALE and $\Lambda_{\mathrm{EW}} =$ EWSCALE is performed with the SMEFTRunRGEs routine, while the LEFT running between $\Lambda_{\mathrm{EW}} =$ EWSCALE and $\Lambda_{\mathrm{IR}} =$ LOWSCALE is performed with the LEFTRunRGEs routine. Alternatively, the user can also perform the full RGE evolution from $\Lambda_{\mathrm{UV}} > \Lambda_{\mathrm{EW}}$ down to $\Lambda_{\mathrm{IR}} < \Lambda_{\mathrm{EW}}$ by means of the RunDsixTools master routine, which internally makes use of SMEFTRunRGEs and LEFTRunRGEs and also applies the SMEFT-LEFT matching at Λ_{EW} with SMEFTLEFTMatch.

DsixTools has three different methods for the resolution of the RGEs, which the user can choose by setting the flag RGEsMethod:

- *"Exact"* (RGEsMethod $= 1$): This method applies the Mathematica internal command NDSolve for the numerical resolution of the differential equations. Given the large number of differential equations involved in this case (several thousands), this might be time consuming, with each evaluation requiring a few (< 10) seconds, the exact number depending on the particular case and computer.

- *"First leading log"* (RGEsMethod $= 2$): This approximate method might be sufficient for many phenomenological studies, in particular when the initial and final scales are not too far from each other. The solution of the RGEs is obtained as

$$C_i(\mu) = C_i(\Lambda_{\text{start}}) + \frac{\beta_i}{16\pi^2} \log\left(\frac{\mu}{\Lambda_{\text{start}}}\right) , \tag{4.2}$$

where C_i is any of the running parameters, μ is the renormalization scale and β_i is the beta function for the C_i parameter evaluated at $\mu = \Lambda_{\text{start}}$. This method is much faster but neglects leading log resummation.

- *"Evolution matrix"* (RGEsMethod $= 3$): This method uses an evolution matrix formalism, explained in detail in Appendix D.

By default, the SM parameters are assumed to be given at the electroweak scale $\Lambda_{\text{EW}} = M_Z = 91.1876$ GeV. Therefore, before running down from Λ_{UV} to Λ_{EW} they must be computed at Λ_{UV}. In case the user chooses to solve the RGEs with RGEsMethod=1 (NDSolve) or RGEsMethod=2 (leading log), this can be done by running up from the electroweak scale using pure SM RGEs, hence neglecting possible deviations caused by non-zero SMEFT WCs.[7] However, in case the user prefers to give the SM parameters directly at the high-energy scale Λ_{UV}, this can be done by setting the UseRGEsSM option to 0. This choice is recommended when the user wants to use the *First leading log* method to solve the RGEs. In the case the user chooses RGEsMethod=3 (DsixTools default) for the resolution of the RGEs (the evolution matrix method), this is implicitly taken into account. Our derivation of the evolution matrix already enforces the SM parameters to be fixed to their measured values at the EW scale.

The user chooses between these three methods by setting the global option RGEsMethod to 1 (for the *"Exact"* method), to 2 (for the *"First leading log"* method) or to 3 (for the *"Evolution matrix"* method), via the routine SetRGEsMethod. After running, the results are saved in the function D6run, such that D6run[*parameter*] returns the parameter *parameter* after RGE running as a function of the renormalization scale μ. Therefore, the user can easily read the results by running commands such as

```
D6run[Clq1[2,2,3,3]] /. \[Mu] -> EWSCALE
```

which would give the result for $[C_{\ell q}^{(1)}(\Lambda_{\text{EW}})]_{2233}$.

The results obtained after running can be also exported to a text file. This is done with the routines SMEFTrunnerExport[] and LEFTrunnerExport[], which generates the files Output_SMEFTrunner.dat or Output_LEFTrunner.dat in each case, with SLHA format

[7] The user can check the validity of this approximation by using the DsixTools routines, for instance by checking whether the resulting values for the SM parameters at the electroweak scale (after running down) do not match their initial values. This can be fixed by readjusting the SM parameters at Λ_{UV}. We note, however, that one should also take into account NP corrections to the standard electroweak parameters induced by non-zero SMEFT WCs.

(completely analogous to the WCs input card in this format). Alternatively, the user can export the results into text files following the WCxf convention [?] by adding an argument to the previous routines: `SMEFTrunnerExport[`*format*`]` and `LEFTrunnerExportWCXF[`*format*`]`, with *format* being `JSON` or `YAML`.

The evolution matrix method is also used internally by default when evaluating the routines `SMEFTEvolve` and `LEFTEvolve`. These routines provide a semi-analytical solution of the RGEs. For example,

```
EvolveSMEFT[CdG[2,2], EWSCALE, HIGHSCALE]
```

returns an analytical expression for the SMEFT WC $[C_{dG}]_{22}$ at $\mu = \Lambda_{\mathrm{EW}}$ as a function of the SMEFT parameters at $\mu = \Lambda_{\mathrm{UV}}$, with numerical coefficients. This easily allows the user to identify the most relevant contributions to the running, as well as running fast numerical scans of the EFT parameter space.

Finally, we point out that `DsixTools` can also be used for analytical calculations involving the SMEFT or LEFT beta functions, since these are available to the user right after loading the package. They can be printed simply by evaluating `β[parameter]`, where `parameter` must be a valid SMEFT or LEFT parameter (a member of `SMEFTParameterListTotal` or `LEFTParameterListTotal`). For instance, `β[LdddSRR[2, 3, 3, 3]]` returns the beta function of the LEFT WC $[L_{ddd}^{S,RR}]_{2333}$.

4.5 SMEFT-LEFT matching at the electroweak scale

In the first step of the matching process, `DsixTools` transforms all the SMEFT parameters at the EW scale to the *up basis*, applying the required biunitary transformations to the fermion mass matrices (which include contributions from dimension-six operators). The up basis, defined in Appendix A, allows one to properly identify the top quark, one of the fields that decouples in the matching. After this transformation, the LEFT parameters at the electroweak scale are computed, using either the full tree-level matching [?][8] (if `MatchingLoopOrder` $= 0$) or the full one-loop matching [?] (if `MatchingLoopOrder` $= 1$). In order to set the value of `MatchingLoopOrder` prior to the matching procedure, the user can use the routine `SetMatchingLoopOrder`. The result of the matching of the LEFT coefficients at the EW scale is given in the tree-level mass basis.

The SMEFT-LEFT matching is performed by evaluating

```
SMEFTLEFTMatch;
```

This routine (re)initializes the `Match` dispatch, which can be used to obtain the numerical values of the LEFT parameters after the matching at the electroweak scale. Therefore

[8] We have independently derived the tree-level results in two different ways, finding full agreement.

```
LeeVLL[1,1,1,1] /. Match
```

would return the numerical value of $[L_{ee}^{V,LL}(\Lambda_{\rm EW})]_{1111}$ in units of GeV^{-2}. The corresponding analytical expressions can be obtained by using MatchEW, e.g.

```
LeeVLL[1,1,1,1] // MatchEW
```

Note that MatchEW does not require running SMEFTLEFTMatch.

Since the LEFT is more general than the SMEFT low-energy limit, not all the LEFT operators are generated from a matching to the SMEFT. For instance, applying the command ML$\nu\gamma$ /.MatchAnalytical would return a 3×3 matrix full of zeros, since the LEFT operator $\mathcal{O}_{\nu\gamma}$ is not present in the SMEFT.

Furthermore, as explained, the first step of the routine MatchSMEFTLEFT is to rotate all SMEFT parameters to the fermion up basis. These rotations can be readily obtained by means of the SMEFTRotateParameters routine by evaluating e.g.,

```
{ToUpBasis, ToDownBasis} = SMEFTRotateParameters[EWSCALE];
```

This will create the dispatches ToUpBasis and ToDownBasis, which can be used to obtain any SMEFT parameter in the up and down bases at the electroweak scale. For instance,

```
CuH[1,2] /. ToUpBasis
CuH[1,2] /. ToDownBasis
```

would return $[C_{u\varphi}]_{12}$ in GeV in the up and down bases at $\Lambda_{\rm EW}$. We also note that the SMEFTRotateParameters routine can be used to obtain the SMEFT parameters in the up and down bases at any scale $\mu \geq \Lambda_{\rm EW}$. For instance, running

```
{ToUpBasis, ToDownBasis} = SMEFTRotateParameters[500];
```

creates the dispatches ToUpBasis and ToDownBasis, now applicable to obtain any SMEFT parameter in the up and down bases at 500 GeV. Finally, if the user is interested in only one of the two fermion bases, up or down, the routine to be used is SMEFTToNewBasis. For instance,

```
ToUpBasis = SMEFTToNewBasis["up",EWSCALE];
```

would only create the ToUpBasis dispatch.

All these results can be exported to external text files with the routine EWmatcherExport. This generates the file Output_EWmatcher.dat, in SLHA format. The results can also be exported in WCxf convention by adding two arguments (*format* and *name*) to the previous routine: EWmatcherExport[*format*, *name*], with *format* being "JSON" or "YAML". The resulting file will always be in the up basis, denoted as Warsaw Up basis in the WCxf exchange format documentation [?].

4.6 Reference guide and tools in DsixTools

DsixTools aims at a simple and visual experience. This is accomplished via a variety of routines, some of which grant the user simple access to the most basic, useful and comprehensive information about the LEFT and the SMEFT, while others implement practical operations on the Wilson coefficients.

The first repository of information available is contained in the variables SMEFTObjectList and LEFTObjectList, which are lists of certain *objects*, one for each operator of the EFT (75 for the SMEFT, 103 for the LEFT, up to dimension six). Each object is itself a list containing: the flavor matrix of WCs, the name of the (head of) the WCs, the name of the operator, the symmetry category, the flavor dimension, the canonical dimension, the EFT, the operator class, the broken symmetry (if any), and the LaTeX form for both the operator and its definition. A flattened list of all the parameters appearing in the first position of the objects in SMEFTObjectList and LEFTObjectList is given in SMEFTParametersTotal and LEFTParametersTotal:

$$\texttt{SMEFTParametersTotal} = \texttt{Flatten}[\texttt{SMEFTObjectList}[[\texttt{All}, 1]]]$$

$$\texttt{LEFTParametersTotal} = \texttt{Flatten}[\texttt{LEFTObjectList}[[\texttt{All}, 1]]]$$

which are all the parameters that might receive input values or output results. However, not all these parameters are independent, and not all are complex-valued. The function NIndependnet[*parameter*] returns the number of independent real parameters in a given parameter: 2 for a general complex parameter, 1 for a real parameter and 0 for a redundant one (as chosen by DsixTools convention). The list of independent parameters are contained in the lists SMEFTParameterList[] and LEFTParameterList[], which match the operators in SMEFTOperators and LEFTOperators. For example the LEFT Lagrangian in the “independent basis” (containing only non-redundant operators) is given by

```
LEFTParameterList[].LEFTOperators
```

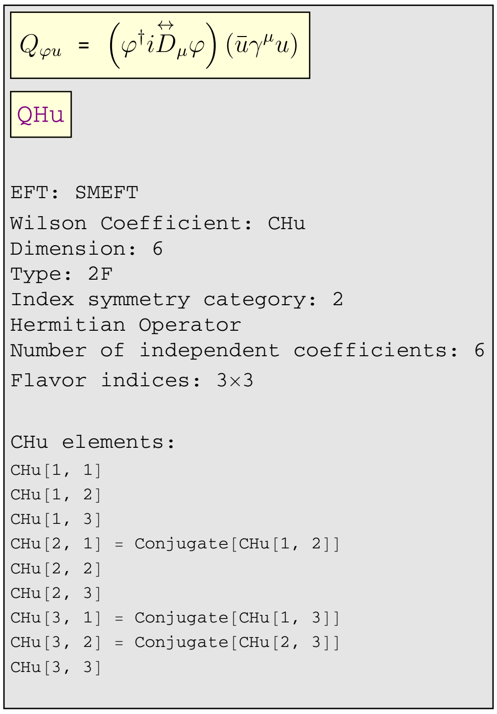
In addition, in order to find the position that a *parameter* occupies in SMEFTParameterList[] or LEFTParameterList[] one can use the routines SMEFTFindParameter[*parameter*] and LEFTFindParameter[*parameter*].

The routines SMEFTParameterList and LEFTParameterList also admit arguments in order to choose subsets of parameters with certain properties. For example

```
LEFTParameterList["D5"]
```

lists the non-redundant LEFT parameters of canonical dimension five. For a list of attributes that can be chosen in SMEFTParameterList and LEFTParameterList see Section 4.2.

More visual information on the properties of operators and parameters can be obtained via a series of new routines. The routine ObjectInfo displays a large amount of useful information on any WC, or operator of the SMEFT or the LEFT specified by the user. For instance,



$Q_{\varphi u} = \left(\varphi^\dagger i\overset{\leftrightarrow}{D}_\mu\varphi\right)(\bar{u}\gamma^\mu u)$

QHu

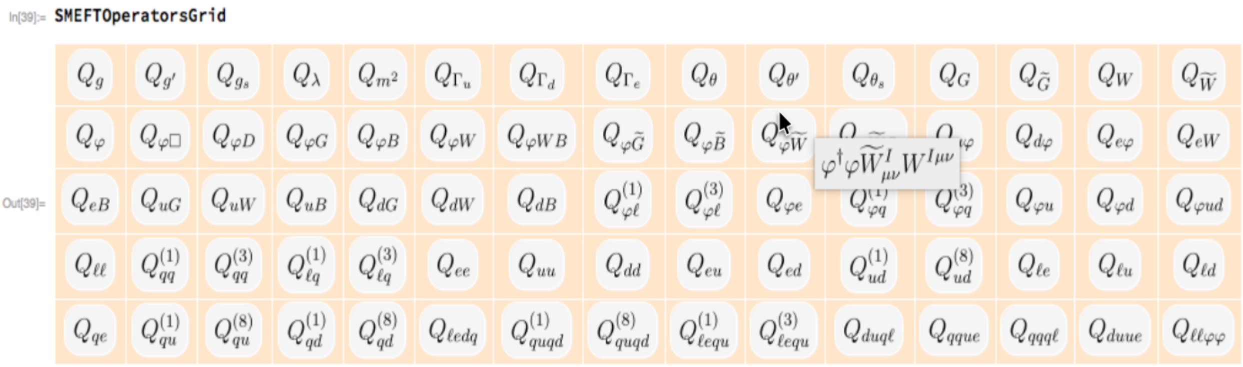
```
EFT: SMEFT
Wilson Coefficient: CHu
Dimension: 6
Type: 2F
Index symmetry category: 2
Hermitian Operator
Number of independent coefficients: 6
Flavor indices: 3×3

CHu elements:
CHu[1, 1]
CHu[1, 2]
CHu[1, 3]
CHu[2, 1] = Conjugate[CHu[1, 2]]
CHu[2, 2]
CHu[2, 3]
CHu[3, 1] = Conjugate[CHu[1, 3]]
CHu[3, 2] = Conjugate[CHu[2, 3]]
CHu[3, 3]
```

Figure 3: Information about the SMEFT WC $C_{\varphi u}$ obtained after evaluating `ObjectInfo[CHu]`, or by using the interfaces `SMEFTOperatorsMenu` or `SMEFTOperatorsGrid`.

```
ObjectInfo[CHu]
```

displays a menu with information about the SMEFT WC $C_{\varphi u}$, including the EFT to which it belongs, the name of the associated WC, the dimension (2, 3, 4, 5 or 6) and type (0F, 2F or 4F), whether it corresponds to an Hermitian operator or not, the number of independent real parameters, the number of flavor indices and the list of elements, as shown in Fig. 3. A clickable menu with information about the SMEFT and LEFT parameters can be loaded with `SMEFTOperatorsMenu`, `LEFTOperatorsMenu` and `TotalOperatorsMenu`, while grid menus with all the SMEFT or LEFT parameters can be generated with `SMEFTOperatorsGrid` and `LEFTOperatorsGrid`. These grids are interactive, and the definition of any operator appears o screen when dragging the mouse pointer on top (see Fig. 4). In addition, clicking on the corresponding operator creates a pop-up window with the same chart created by `ObjectInfo`. The `Mathematica` notebook `OperatorsGrid.nb` can be found in the main `DsixTools` folder. This notebook already contains the result of using `OperatorsGridSMEFT` and `OperatorsGridLEFT`, and the two grid menus can be used right after opening the notebook, without any need to load `DsixTools`. This can be useful as an *out of the box* visual reference on the SMEFT and the LEFT.

Concerning handy routines for handling WCs and expressions with WCs (such as amplitudes or cross-sections), we highlight `D6Simplify`. This routine is used to simplify expres-



```
In[39]:= SMEFTOperatorsGrid
```

Out[39]=

Q_g	$Q_{g'}$	Q_{g_s}	Q_λ	Q_{m^2}	Q_{Γ_u}	Q_{Γ_d}	Q_{Γ_e}	Q_θ	$Q_{\theta'}$	Q_{θ_s}	Q_G	$Q_{\tilde{G}}$	Q_W	$Q_{\widetilde{W}}$
Q_φ	$Q_{\varphi\Box}$	$Q_{\varphi D}$	$Q_{\varphi G}$	$Q_{\varphi B}$	$Q_{\varphi W}$	$Q_{\varphi WB}$	$Q_{\varphi\tilde{G}}$	$Q_{\varphi\tilde{B}}$	$Q_{\varphi\widetilde{W}}$	[illegible]	$Q_{u\varphi}$	$Q_{d\varphi}$	$Q_{e\varphi}$	Q_{eW}
Q_{eB}	Q_{uG}	Q_{uW}	Q_{uB}	Q_{dG}	Q_{dW}	Q_{dB}	$Q_{\varphi\ell}^{(1)}$	$Q_{\varphi\ell}^{(3)}$	$Q_{\varphi e}$	$Q_{\varphi q}^{(1)}$	$Q_{\varphi q}^{(3)}$	$Q_{\varphi u}$	$Q_{\varphi d}$	$Q_{\varphi ud}$
$Q_{\ell\ell}$	$Q_{qq}^{(1)}$	$Q_{qq}^{(3)}$	$Q_{\ell q}^{(1)}$	$Q_{\ell q}^{(3)}$	Q_{ee}	Q_{uu}	Q_{dd}	Q_{eu}	Q_{ed}	$Q_{ud}^{(1)}$	$Q_{ud}^{(8)}$	$Q_{\ell e}$	$Q_{\ell u}$	$Q_{\ell d}$
Q_{qe}	$Q_{qu}^{(1)}$	$Q_{qu}^{(8)}$	$Q_{qd}^{(1)}$	$Q_{qd}^{(8)}$	$Q_{\ell edq}$	$Q_{quqd}^{(1)}$	$Q_{quqd}^{(8)}$	$Q_{\ell equ}^{(1)}$	$Q_{\ell equ}^{(3)}$	$Q_{duq\ell}$	Q_{qque}	$Q_{qqq\ell}$	Q_{duue}	$Q_{\ell\ell\varphi\varphi}$

$\varphi^\dagger\varphi\widetilde{W}^I_{\mu\nu}W^{I\mu\nu}$

Figure 4: Result of evaluating SMEFTOperatorsGrid. Positioning the mouse on top of any operator displays its definition, and clicking on it opens a pop-up containing the corresponding chart of Fig. 3. This grid can be saved in a notebook and used later in a fresh Kernel without loading DsixTools.

sions involving SMEFT or LEFT parameters, by replacing all redundant WCs in terms of non-redundant ones and eliminating complex conjugates on real parameters. For instance,

```
D6Simplify[2 m2 CHq1[3,2] CC[Gd[3,1]]]
```

returns `2 m2 CHq1[2,3]*` `Gd[3,1]*`, where the hermiticity relation $[C_{\varphi q}^{(1)}]_{32} = [C_{\varphi q}^{(1)}]_{23}^*$ for the SMEFT object $C_{\varphi q}^{(1)}$ has been used in order to express the result in terms of the independent parameter $[C_{\varphi q}^{(1)}]_{23}$. As already mentione, the function NIndependnet returns the number of independent real parameters in a given parameter. Finally, the routines ToSymmetric, ToSymmetricSingle, ToIndependent and ToIndependentSingle can be used to transform WCs to the symmetric and independent bases (see Appendix C.3 for the definition of these bases). The routine CheckAndSymmetrize also checks whether all hermiticity and antisymmetry conditions are satisfied in a given argument.

5 Summary

DsixTools is a Mathematica package for simbolic and numerical operations within the SMEFT and the LEFT, facilitating the treatment of these two effective theories in a systematic and complete manner.

Here we have presented DsixTools 2.0, a new and improved version of DsixTools. This version features the complete one-loop evolution from a high-energy scale $\Lambda_{\rm UV} > \Lambda_{\rm EW}$ (where the physics is described by the SMEFT) down to a low-energy scale $\Lambda_{\rm IR} < \Lambda_{\rm EW}$ (where the physics is described by the LEFT). This includes complete one-loop RGE evolution and complete one-loop matching at the EW scale. In addition, the new version contains a large number of improvements regarding notation and utilities, operational efficiency and simplicity, user interface, input and output, a set of reference tools for the SMEFT and the LEFT, and a complete Mathematica documentation system.

DsixTools is a project that can be extended with future improvements, including additional tools and functionalities. The final outcome of this endeavour will be a complete and powerful framework for the systematic exploration of new physics models using the language of Effective Field Theories.

Acknowledgements

We thank Alejandro Celis for his decisive contributions to the development of DsixTools as a whole, and his early involvement in the development of this second version. We are also grateful to Jorge Terol-Calvo, for testing an early version of DsixTools 2.0. This work has been partially funded by the DFG Scientific Network "DixTools", Project Number 406385113, and by Generalitat Valenciana, Project SEJI/2018/033. The work of J.F. was supported by the Cluster of Excellence 'Precision Physics, Fundamental Interactions, and Structure of Matter' (PRISMA+ EXC 2118/1) funded by the German Research Foundation (DFG) within the German Excellence Strategy (Project ID 39083149). A.V. acknowledges financial support from the Spanish grants FPA2017-85216-P (MINECO/AEI/FEDER, UE) and FPA2017-90566-REDC (Red Consolider MultiDark), and from MINECO through the "Ramón y Cajal" contract RYC2018-025795-I. J.V. acknowledges funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 700525 'NIOBE', and from the Spanish MINECO through the "Ramón y Cajal" program RYC-2017-21870.

A Standard Model Effective Field Theory

The SMEFT is the EFT obtained after extending the SM Lagrangian with all operators invariant under the $SU(3)_c \times SU(2)_L \times U(1)_Y$ gauge group up to an arbitrary dimension. The Lagrangian for the SMEFT can be written as

$$\mathcal{L}_{\text{SMEFT}} = \mathcal{L}_{\text{SM}} + \sum_k C_k^{(5)} Q_k^{(5)} + \sum_k C_k^{(6)} Q_k^{(6)} + \mathcal{O}\left(\frac{1}{\Lambda_{\text{UV}}^3}\right) . \tag{A.1}$$

The dimensionful Wilson coefficients $C_k^{(5)}$ and $C_k^{(6)}$ are implicitly suppressed by $1/\Lambda_{\text{UV}}$ and $1/\Lambda_{\text{UV}}^2$, respectively, where Λ_{UV} is the EFT cutoff scale, assumed to be much larger than the electroweak scale. The implementation of the SMEFT in DsixTools mainly follows the conventions used in Ref. [?].[9] The SM Lagrangian is given by

$$\mathcal{L}_{\text{SM}} = -\frac{1}{4} G_{\mu\nu}^A G^{A\mu\nu} - \frac{1}{4} W_{\mu\nu}^I W^{I\mu\nu} - \frac{1}{4} B_{\mu\nu} B^{\mu\nu} + (D_\mu \varphi)^\dagger (D^\mu \varphi) + m^2 \varphi^\dagger \varphi - \frac{\lambda}{2} \left(\varphi^\dagger \varphi\right)^2$$

[9] The reader should keep in mind that these conventions differ from those used in [? ? ?]. The differences appear in the normalization of λ and m, the definition of the Yukawa matrices and the name of the gauge couplings. However, DsixTools 2.0 adopts the convention of [? ? ?] of introducing the EFT cutoff scale into the definition of the WCs.

$$+ i \left(\bar{\ell} \not{D} \ell + \bar{e} \not{D} e + \bar{q} \not{D} q + \bar{u} \not{D} u + \bar{d} \not{D} d\right) - \left(\bar{\ell} \Gamma_e e \varphi + \bar{q} \Gamma_u u \widetilde{\varphi} + \bar{q} \Gamma_d d \varphi + \text{h.c.}\right) + \mathcal{L}_\theta \,. \quad \text{(A.2)}$$

Here $G^A_{\mu\nu}$ ($A = 1 \ldots 8$), $W^I_{\mu\nu}$ ($I = 1 \ldots 3$) and $B_{\mu\nu}$ denote, respectively, the $SU(3)_c$, $SU(2)_L$ and $U(1)_Y$ field-strength tensors. The fields ℓ and q correspond to the lepton and quark $SU(2)_L$ doublets of the SM, while e, u, d are the SM right-handed fields. The Higgs $SU(2)_L$ doublet is denoted by φ. The Yukawa couplings $\Gamma_{e,u,d}$ are 3×3 matrices in flavor space. Using appropriate field redefinitions, and without loss of generality, one can choose a particular flavor basis where Γ_e and Γ_d are diagonal and $\Gamma_u = V_{\text{CKM}} \hat{\Gamma}_u$, with $\hat{\Gamma}_u$ diagonal and V_{CKM} denoting the CKM matrix. This is the so-called *down basis*, and it is the default basis choice for `DsixTools`. Another basis choice that is also useful is the *up basis*, where Γ_e and Γ_u are diagonal and $\Gamma_d = V^\dagger_{\text{CKM}} \hat{\Gamma}_d$ with $\hat{\Gamma}_d$ diagonal. Note, however, that these bases are not stable under RGE evolution. The covariant derivative is defined as

$$D_\mu = \partial_\mu + i g_s T^A G^A_\mu + i g T^I W^I_\mu + i g' Y B_\mu \,, \quad \text{(A.3)}$$

where $\{g_s, g, g'\}$ and $\{G, W, B\}$ are, respectively, the $SU(3)_c$, $SU(2)_L$ and $U(1)_Y$ gauge couplings and gauge fields, and T^A and T^I are the corresponding gauge group generators in the appropriate representations. The hypercharge assignments for the matter fields are given in Table 2. The θ terms are given by

$$\mathcal{L}_\theta = \frac{\theta' g'^2}{32\pi^2} \widetilde{B}_{\mu\nu} B^{\mu\nu} + \frac{\theta g^2}{32\pi^2} \widetilde{W^I_{\mu\nu}} W^{\mu\nu}_I + \frac{\theta_s g_s^2}{32\pi^2} \widetilde{G}^A_{\mu\nu} G^{\mu\nu}_A \,, \quad \text{(A.4)}$$

with the dual tensors defined as $\widetilde{X} = \frac{1}{2} \epsilon_{\mu\nu\rho\sigma} X^{\rho\sigma}$ (with $\epsilon_{0123} = +1$). There is only one operator of dimension five, the so-called Weinberg operator,

$$Q_{\ell\ell\varphi\varphi} = \left(\widetilde{\varphi}^\dagger \ell\right)^T C \left(\widetilde{\varphi}^\dagger \ell\right) \,, \quad \text{(A.5)}$$

with C denoting the Dirac charge conjugation matrix. This operator gives a Majorana mass term for the neutrinos after spontaneous symmetry breaking [?]. A non-redundant basis of dimension-six operators was defined in [?], the so called *Warsaw basis*. Table 3 classifies the SMEFT operators in the Warsaw basis indicating the number of independent operators in each category. We list the Baryon-number-conserving operators in Tables 4, 5 and 6. Barring flavor structure, these constitute a total of 59 operators, some of which are non-Hermitian, yielding in total 76 real coefficients. Taking into account flavor indices, the Baryon-number-conserving dimension-six Lagrangian contains 1350 CP-even and 1149 CP-odd operators, for a total of 2499 Hermitian operators [?]. The complete set of independent dimension-six Baryon number violating operators were identified in [?]. Barring flavor structure, there are only 4 Baryon-number-violating operators. These are listed in Table 7.

The beta functions for the SMEFT WCs C_i are defined as

$$\frac{dC_i}{d \log \mu} \equiv \frac{1}{16\pi^2} \, \beta_i \,. \quad \text{(A.6)}$$

Table 2: Hypercharge assignments in the SMEFT.

Field	ℓ_L	e_R	q_L	u_R	d_R	φ
Y	$-\frac{1}{2}$	-1	$\frac{1}{6}$	$\frac{2}{3}$	$-\frac{1}{3}$	$\frac{1}{2}$

Here μ is the renormalization scale, and β_i are the individual beta functions of each WC. The complete set of one-loop beta functions for the SM and dimension-six WCs were computed in [? ? ? ?]. The beta functions in these references neglect the contributions to the running of the dimension-six WCs from two insertions of the dimension-five Weinberg operator. Given the smallness of neutrino masses, it is natural to expect that the scale suppressing this operator is much larger than the one of the dimension-six operators, which justifies having neglected these contributions. The beta function for the Weinberg operator can be found in [?]. The complete set of one-loop SMEFT beta functions can be read off directly from `DsixTools` with the command β`[`*parameter*`]`.

B Low-Energy Effective Field Theory

The LEFT is the EFT below the electroweak scale after integrating out the Higgs boson, the massive $W^\pm$ and Z gauge bosons and the top quark from the SM particle content, as well as any BSM degrees of freedom at or above the EW scale. The resulting theory is invariant under the $SU(3)_c \times U(1)_Q$ gauge group and contains $n_u = 2$ up-type quarks, $n_d = 3$ down-type quarks, $n_e = 3$ charged leptons and $n_\nu = 3$ left-handed neutrinos. The LEFT Lagrangian is given by

$$\mathcal{L}_{\text{LEFT}} = \mathcal{L}_{\text{QCD+QED}} + \sum_k L_k^{(3)} \mathcal{O}_k^{(3)} + \sum_k L_k^{(5)} \mathcal{O}_k^{(5)} + \sum_k L_k^{(6)} \mathcal{O}_k^{(6)} + \mathcal{O}\left(\frac{1}{\Lambda_{\text{EW}}^3}\right) . \tag{B.1}$$

The dimensionful Wilson coefficients $L_k^{(5)}$ and $L_k^{(6)}$ are implicitly suppressed by $1/\Lambda_{\text{EW}}$ and $1/\Lambda_{\text{EW}}^2$, respectively, where Λ_{EW} is the electroweak scale. A non-redundant basis of dimension-three, -five and -six operators was introduced in [?], and this will be known in the following as the *San Diego basis*. Table 8 classifies the LEFT operators in the San Diego basis indicating the number of independent operators in each category. Barring flavor structure and Hermitian conjugation there are 96 independent operators. It can be shown that no linear combination of these operators vanish after the application of the equations of motion, which makes them completely independent operators. We list these operators in Tables 9 - 13, omitting flavor (and $SU(3)_c$ indices in the last tables) to simplify the notation. The only operator present at dimension 3 is a Majorana mass term for the left-handed neutrinos, shown in Table 9. There are two categories of dimension-five operators, $(\nu\nu)\,X$ and

Table 3: SMEFT operators in the Warsaw basis. The third column lists the number of operators in the category whereas the last column indicates whether they violate baryon (B) or lepton (L) numbers.

dim	class	# operators	quantum numbers
5	Dimension-five	1	$\Delta L = 2$
6	X^3	4	
6	φ^6	1	
6	$\varphi^4 D^2$	2	
6	$X^2\varphi^2$	8	
6	$\psi^2\varphi^3$	3	
6	$\psi^2 X\varphi$	8	
6	$\psi^2\varphi^2 D$	8	
6	$(\bar{L}L)\,(\bar{L}L)$	5	
6	$(\bar{R}R)\,(\bar{R}R)$	7	
6	$(\bar{L}L)\,(\bar{R}R)$	8	
6	$(\bar{L}R)\,(\bar{L}R)$	4	
6	$(\bar{L}R)\,(\bar{R}L)$	1	
6	Baryon-number-violating	4	$\Delta B = \Delta L = 1$

Table 4: SMEFT purely bosonic operators.

X^3		$X^2\varphi^2$	
Q_G	$f^{ABC}G_\mu^{A\nu}G_\nu^{B\rho}G_\rho^{C\mu}$	$Q_{\varphi G}$	$\varphi^\dagger\varphi G^A_{\mu\nu}G^{A\mu\nu}$
$Q_{\widetilde{G}}$	$f^{ABC}\widetilde{G}_\mu^{A\nu}G_\nu^{B\rho}G_\rho^{C\mu}$	$Q_{\varphi B}$	$\varphi^\dagger\varphi B_{\mu\nu}B^{\mu\nu}$
Q_W	$\epsilon^{IJK}W_\mu^{I\nu}W_\nu^{J\rho}W_\rho^{K\mu}$	$Q_{\varphi W}$	$\varphi^\dagger\varphi W^I_{\mu\nu}W^{I\mu\nu}$
$Q_{\widetilde{W}}$	$\epsilon^{IJK}\widetilde{W}_\mu^{I\nu}W_\nu^{J\rho}W_\rho^{K\mu}$	$Q_{\varphi WB}$	$\varphi^\dagger\tau^I\varphi W^I_{\mu\nu}B^{\mu\nu}$
φ^6		$Q_{\varphi\widetilde{G}}$	$\varphi^\dagger\varphi\widetilde{G}^A_{\mu\nu}G^{A\mu\nu}$
Q_φ	$\left(\varphi^\dagger\varphi\right)^3$	$Q_{\varphi\widetilde{B}}$	$\varphi^\dagger\varphi\widetilde{B}_{\mu\nu}B^{\mu\nu}$
φ^4D^2		$Q_{\varphi\widetilde{W}}$	$\varphi^\dagger\varphi\widetilde{W}^I_{\mu\nu}W^{I\mu\nu}$
$Q_{\varphi\Box}$	$\left(\varphi^\dagger\varphi\right)\Box\left(\varphi^\dagger\varphi\right)$	$Q_{\varphi\widetilde{W}B}$	$\varphi^\dagger\tau^I\varphi\widetilde{W}^I_{\mu\nu}B^{\mu\nu}$
$Q_{\varphi D}$	$\left(\varphi^\dagger D^\mu\varphi\right)^*\left(\varphi^\dagger D_\mu\varphi\right)$		

Table 5: SMEFT mixed operators involving bosons and fermions.

$\psi^2\varphi^3$		$\psi^2\varphi^2D$	
$Q_{u\varphi}$	$\left(\varphi^\dagger\varphi\right)\left(\bar{q}u\widetilde{\varphi}\right)$	$Q^{(1)}_{\varphi\ell}$	$\left(\varphi^\dagger i\overleftrightarrow{D}_\mu\varphi\right)\left(\bar{\ell}\gamma^\mu\ell\right)$
$Q_{d\varphi}$	$\left(\varphi^\dagger\varphi\right)\left(\bar{q}d\varphi\right)$	$Q^{(3)}_{\varphi\ell}$	$\left(\varphi^\dagger i\overleftrightarrow{D}^I_\mu\varphi\right)\left(\bar{\ell}\tau^I\gamma^\mu\ell\right)$
$Q_{e\varphi}$	$\left(\varphi^\dagger\varphi\right)\left(\bar{\ell}e\varphi\right)$	$Q_{\varphi e}$	$\left(\varphi^\dagger i\overleftrightarrow{D}_\mu\varphi\right)\left(\bar{e}\gamma^\mu e\right)$
$\psi^2X\varphi$		$Q^{(1)}_{\varphi q}$	$\left(\varphi^\dagger i\overleftrightarrow{D}_\mu\varphi\right)\left(\bar{q}\gamma^\mu q\right)$
Q_{eW}	$\left(\bar{\ell}\sigma^{\mu\nu}e\right)\tau^I\varphi W^I_{\mu\nu}$	$Q^{(3)}_{\varphi q}$	$\left(\varphi^\dagger i\overleftrightarrow{D}^I_\mu\varphi\right)\left(\bar{q}\tau^I\gamma^\mu q\right)$
Q_{eB}	$\left(\bar{\ell}\sigma^{\mu\nu}e\right)\varphi B_{\mu\nu}$	$Q_{\varphi u}$	$\left(\varphi^\dagger i\overleftrightarrow{D}_\mu\varphi\right)\left(\bar{u}\gamma^\mu u\right)$
Q_{uG}	$\left(\bar{q}\sigma^{\mu\nu}T^Au\right)\widetilde{\varphi}G^A_{\mu\nu}$	$Q_{\varphi d}$	$\left(\varphi^\dagger i\overleftrightarrow{D}_\mu\varphi\right)\left(\bar{d}\gamma^\mu d\right)$
Q_{uW}	$\left(\bar{q}\sigma^{\mu\nu}u\right)\tau^I\widetilde{\varphi}W^I_{\mu\nu}$	$Q_{\varphi ud}$	$\left(\widetilde{\varphi}^\dagger iD_\mu\varphi\right)\left(\bar{u}\gamma^\mu d\right)$
Q_{uB}	$\left(\bar{q}\sigma^{\mu\nu}u\right)\widetilde{\varphi}B_{\mu\nu}$		
Q_{dG}	$\left(\bar{q}\sigma^{\mu\nu}T^Ad\right)\varphi G^A_{\mu\nu}$		
Q_{dW}	$\left(\bar{q}\sigma^{\mu\nu}d\right)\tau^I\varphi W^I_{\mu\nu}$		
Q_{dB}	$\left(\bar{q}\sigma^{\mu\nu}d\right)\varphi B_{\mu\nu}$		

Table 6: SMEFT purely fermionic operators which preserve Baryon number.

$(\bar{L}L)\,(\bar{L}L)$		$(\bar{L}L)\,(\bar{R}R)$	
$Q_{\ell\ell}$	$(\bar{\ell}\gamma_\mu \ell)\,(\bar{\ell}\gamma^\mu \ell)$	$Q_{\ell e}$	$(\bar{\ell}\gamma_\mu \ell)\,(\bar{e}\gamma^\mu e)$
$Q_{qq}^{(1)}$	$(\bar{q}\gamma_\mu q)\,(\bar{q}\gamma^\mu q)$	$Q_{\ell u}$	$(\bar{\ell}\gamma_\mu \ell)\,(\bar{u}\gamma^\mu u)$
$Q_{qq}^{(3)}$	$(\bar{q}\gamma_\mu \tau^I q)\,(\bar{q}\gamma^\mu \tau^I q)$	$Q_{\ell d}$	$(\bar{\ell}\gamma_\mu \ell)\,(\bar{d}\gamma^\mu d)$
$Q_{\ell q}^{(1)}$	$(\bar{\ell}\gamma_\mu \ell)\,(\bar{q}\gamma^\mu q)$	Q_{qe}	$(\bar{q}\gamma_\mu q)\,(\bar{e}\gamma^\mu e)$
$Q_{\ell q}^{(3)}$	$(\bar{\ell}\gamma_\mu \tau^I \ell)\,(\bar{q}\gamma^\mu \tau^I q)$	$Q_{qu}^{(1)}$	$(\bar{q}\gamma_\mu q)\,(\bar{u}\gamma^\mu u)$
$(\bar{R}R)\,(\bar{R}R)$		$Q_{qu}^{(8)}$	$(\bar{q}\gamma_\mu T^A q)\,(\bar{u}\gamma^\mu T^A u)$
Q_{ee}	$(\bar{e}\gamma_\mu e)\,(\bar{e}\gamma^\mu e)$	$Q_{qd}^{(1)}$	$(\bar{q}\gamma_\mu q)\,(\bar{d}\gamma^\mu d)$
Q_{uu}	$(\bar{u}\gamma_\mu u)\,(\bar{u}\gamma^\mu u)$	$Q_{qd}^{(8)}$	$(\bar{q}\gamma_\mu T^A q)\,(\bar{d}\gamma^\mu T^A d)$
Q_{dd}	$(\bar{d}\gamma_\mu d)\,(\bar{d}\gamma^\mu d)$	$(\bar{L}R)\,(\bar{R}L)$	
Q_{eu}	$(\bar{e}\gamma_\mu e)\,(\bar{u}\gamma^\mu u)$	$Q_{\ell edq}$	$(\bar{\ell}^j e)\,(\bar{d}q^j)$
Q_{ed}	$(\bar{e}\gamma_\mu e)\,(\bar{d}\gamma^\mu d)$	$(\bar{L}R)\,(\bar{L}R)$	
$Q_{ud}^{(1)}$	$(\bar{u}\gamma_\mu u)\,(\bar{d}\gamma^\mu d)$	$Q_{quqd}^{(1)}$	$(\bar{q}^j u)\,\epsilon_{jk}\,(\bar{q}^k d)$
$Q_{ud}^{(8)}$	$(\bar{u}\gamma_\mu T^A u)\,(\bar{d}\gamma^\mu T^A d)$	$Q_{quqd}^{(8)}$	$(\bar{q}^j T^A u)\,\epsilon_{jk}\,(\bar{q}^k T^A d)$
		$Q_{\ell equ}^{(1)}$	$(\bar{\ell}^j e)\,\epsilon_{jk}\,(\bar{q}^k u)$
		$Q_{\ell equ}^{(3)}$	$(\bar{\ell}^j \sigma_{\mu\nu} e)\,\epsilon_{jk}\,(\bar{q}^k \sigma^{\mu\nu} u)$

Table 7: SMEFT Baryon-number-violating operators.

Baryon-number-violating	
$Q_{duq\ell}$	$(d^T C u)\,(q^T C \ell)$
Q_{qque}	$(q^T C q)\,(u^T C e)$
$Q_{qqq\ell}$	$\epsilon_{il}\epsilon_{jk}\,(q_i^T C q_j)\,(q_k^T C \ell_l)$
Q_{duue}	$(d^T C u)\,(u^T C e)$

$(\bar{L}R)\, X$, both listed in Table 10. While the former violates Lepton number in two units, the latter preserves both lepton and Baryon numbers. All the dimension-five LEFT operators are dipole operators. The remaining 89 independent operators arise at dimension 6. There are 2 purely gluonic operators, shown in Table 11, 56 4-fermion operators that conserve both Baryon and Lepton numbers, shown in Table 12, and 31 4-fermion operators that violate Baryon and/or Lepton numbers, shown in Table 13. Assuming that the SMEFT is the correct theory above the EW scale, all parameters of the LEFT can be fixed though a matching calculation at the EW scale. The complete set of matching conditions in the Warsaw and San Diego bases are known at tree-level [?] and one-loop [?] orders. They can be can be read off directly from `DsixTools` with the command `MatchEW[`*parameter*`]`.

The implementation of the LEFT in `DsixTools` follows the same (or analogous) conventions as for the SMEFT. [10] The QCD and QED Lagrangian is given by

$$\mathcal{L}_{\text{QCD+QED}} = -\frac{1}{4} G^A_{\mu\nu} G^{A\mu\nu} - \frac{1}{4} F_{\mu\nu} F^{\mu\nu} + \theta_{\text{QCD}} \, \frac{g_s^2}{32\,\pi^2} \widetilde{G}^A_{\mu\nu} G^{\mu\nu}_A + +\theta_{\text{QED}} \, \frac{e^2}{32\,\pi^2} \widetilde{F}_{\mu\nu} F^{\mu\nu} + \sum_{\psi=u,d,e,\nu_L} \overline{\psi}\, i \not{D} \psi - \left[\sum_{\psi=u,d,e} \overline{\psi}_L M_\psi \psi_R + \text{h.c.} \right] . \tag{B.2}$$

The Dirac mass matrices M_u and $M_{e,d}$ are, respectively, 2×2 and 3×3 matrices in flavor space and we will omit flavor indices whenever possible. The absence of a Dirac mass matrix for the neutrinos is due to the fact that right-handed neutrinos are not included in the LEFT. The covariant derivative is defined as

$$D_\mu = \partial_\mu + i g_s T^A G^A_\mu + i e Q A_\mu \,, \tag{B.3}$$

where g_s and e are the $SU(3)_c$ and $U(1)_Q$ gauge couplings, respectively, and T^A $(A = 1 \ldots 8)$ are the Gell-Mann matrices. The gauge field tensors are defined as usual,

$$G^A_{\mu\nu} = \partial_\mu G^A_\nu - \partial_\nu G^A_\mu - g_s f^{ABC} G^B_\mu G^C_\nu \,, \tag{B.4}$$

$$F_{\mu\nu} = \partial_\mu F_\nu - \partial_\nu F_\mu \,, \tag{B.5}$$

with covariant derivatives

$$(D_\rho G_{\mu\nu})^A = \partial_\rho G^A_{\mu\nu} - g_s f^{ABC} G^B_\rho G^C_{\mu\nu} \,, \tag{B.6}$$

$$(D_\rho F_{\mu\nu}) = \partial_\rho F_{\mu\nu} \,. \tag{B.7}$$

Finally, dual tensors are defined as $\widetilde{X} = \frac{1}{2} \epsilon_{\mu\nu\rho\sigma} X^{\rho\sigma}$ (with $\epsilon_{0123} = +1$).

The RGEs governing the renormalization scale evolution of the LEFT Wilson coefficients L_i are given by

$$\frac{dL_i}{d\log\mu} = \frac{1}{16\pi^2}\,\beta_i \,. \tag{B.8}$$

[10] The reader should keep in mind that these conventions differ from those used in [? ?]. The differences appear in the definition of the fermion mass matrices and the name of the strong gauge coupling.

Table 8: LEFT operators in the San Diego basis. The third column lists the number of operators in the category whereas the last column indicates whether they violate baryon (B) or lepton (L) numbers.

dim	class	# operators	quantum numbers
3	$\nu\nu$	1	$\Delta L = 2$
5	$(\nu\nu)\,X$	1	$\Delta L = 2$
5	$(\bar{L}R)\,X$	5	
6	X^3	2	
6	$(\bar{L}L)\,(\bar{L}L)$	12	
6	$(\bar{R}R)\,(\bar{R}R)$	7	
6	$(\bar{L}L)\,(\bar{R}R)$	19	
6	$(\bar{L}R)\,(\bar{L}R)$	15	
6	$(\bar{L}R)\,(\bar{R}L)$	3	
6	$\Delta L = 4$	1	$\Delta L = 4$
6	$\Delta L = 2$	14	$\Delta L = 2$
6	$\Delta B = \Delta L = 1$	9	$\Delta B = \Delta L = 1$
6	$\Delta B = -\Delta L = 1$	7	$\Delta B = -\Delta L = 1$

Table 9: LEFT dimension-three operator.

$\nu\nu$	
$\mathcal{O}_\nu$	$\nu_L^T C \nu_L$

Table 10: LEFT dimension-five operators.

$(\nu\nu)\, X$		$\left(\bar{L}R\right) X$	
$\mathcal{O}_{\nu\gamma}$	$\left(\nu_L^T C \sigma^{\mu\nu} \nu_L\right) F_{\mu\nu}$	$\mathcal{O}_{e\gamma}$	$\left(\bar{e}_L \sigma^{\mu\nu} e_R\right) F_{\mu\nu}$
		$\mathcal{O}_{u\gamma}$	$\left(\bar{u}_L \sigma^{\mu\nu} u_R\right) F_{\mu\nu}$
		$\mathcal{O}_{d\gamma}$	$\left(\bar{d}_L \sigma^{\mu\nu} d_R\right) F_{\mu\nu}$
		$\mathcal{O}_{uG}$	$\left(\bar{u}_L \sigma^{\mu\nu} T^A u_R\right) G_{\mu\nu}^A$
		$\mathcal{O}_{dG}$	$\left(\bar{d}_L \sigma^{\mu\nu} T^A d_R\right) G_{\mu\nu}^A$

Table 11: LEFT purely gluonic operators.

X^3	
$\mathcal{O}_G$	$f^{ABC} G_\mu^{A\nu} G_\nu^{B\rho} G_\rho^{C\mu}$
$\mathcal{O}_{\widetilde{G}}$	$f^{ABC} \widetilde{G}_\mu^{A\nu} G_\nu^{B\rho} G_\rho^{C\mu}$

Table 12: LEFT Baryon and Lepton number conserving dimension-six operators.

$(\bar{L}L)(\bar{L}L)$	
$\mathcal{O}_{\nu\nu}^{V,LL}$	$(\bar{\nu}_L\gamma_\mu\nu_L)(\bar{\nu}_L\gamma^\mu\nu_L)$
$\mathcal{O}_{ee}^{V,LL}$	$(\bar{e}_L\gamma_\mu e_L)(\bar{e}_L\gamma^\mu e_L)$
$\mathcal{O}_{\nu e}^{V,LL}$	$(\bar{\nu}_L\gamma_\mu\nu_L)(\bar{e}_L\gamma^\mu e_L)$
$\mathcal{O}_{\nu u}^{V,LL}$	$(\bar{\nu}_L\gamma_\mu\nu_L)(\bar{u}_L\gamma^\mu u_L)$
$\mathcal{O}_{\nu d}^{V,LL}$	$(\bar{\nu}_L\gamma_\mu\nu_L)\big(\bar{d}_L\gamma^\mu d_L\big)$
$\mathcal{O}_{eu}^{V,LL}$	$(\bar{e}_L\gamma_\mu e_L)(\bar{u}_L\gamma^\mu u_L)$
$\mathcal{O}_{ed}^{V,LL}$	$(\bar{e}_L\gamma_\mu e_L)\big(\bar{d}_L\gamma^\mu d_L\big)$
$\mathcal{O}_{\nu edu}^{V,LL}$	$(\bar{\nu}_L\gamma_\mu e_L)\big(\bar{d}_L\gamma^\mu u_L\big)$
$\mathcal{O}_{uu}^{V,LL}$	$(\bar{u}_L\gamma_\mu u_L)(\bar{u}_L\gamma^\mu u_L)$
$\mathcal{O}_{dd}^{V,LL}$	$\big(\bar{d}_L\gamma_\mu d_L\big)\big(\bar{d}_L\gamma^\mu d_L\big)$
$\mathcal{O}_{ud}^{V1,LL}$	$(\bar{u}_L\gamma_\mu u_L)\big(\bar{d}_L\gamma^\mu d_L\big)$
$\mathcal{O}_{ud}^{V8,LL}$	$\big(\bar{u}_L\gamma_\mu T^A u_L\big)\big(\bar{d}_L\gamma^\mu T^A d_L\big)$

$(\bar{R}R)(\bar{R}R)$	
$\mathcal{O}_{ee}^{V,RR}$	$(\bar{e}_R\gamma_\mu e_R)(\bar{e}_R\gamma^\mu e_R)$
$\mathcal{O}_{eu}^{V,RR}$	$(\bar{e}_R\gamma_\mu e_R)(\bar{u}_R\gamma^\mu u_R)$
$\mathcal{O}_{ed}^{V,RR}$	$(\bar{e}_R\gamma_\mu e_R)\big(\bar{d}_R\gamma^\mu d_R\big)$
$\mathcal{O}_{uu}^{V,RR}$	$(\bar{u}_R\gamma_\mu u_R)(\bar{u}_R\gamma^\mu u_R)$
$\mathcal{O}_{dd}^{V,RR}$	$\big(\bar{d}_R\gamma_\mu d_R\big)\big(\bar{d}_R\gamma^\mu d_R\big)$
$\mathcal{O}_{ud}^{V1,RR}$	$(\bar{u}_R\gamma_\mu u_R)\big(\bar{d}_R\gamma^\mu d_R\big)$
$\mathcal{O}_{ud}^{V8,RR}$	$\big(\bar{u}_R\gamma_\mu T^A u_R\big)\big(\bar{d}_R\gamma^\mu T^A d_R\big)$

$(\bar{L}L)(\bar{R}R)$	
$\mathcal{O}_{\nu e}^{V,LR}$	$(\bar{\nu}_L\gamma_\mu\nu_L)(\bar{e}_R\gamma^\mu e_R)$
$\mathcal{O}_{ee}^{V,LR}$	$(\bar{e}_L\gamma_\mu e_L)(\bar{e}_R\gamma^\mu e_R)$
$\mathcal{O}_{\nu u}^{V,LR}$	$(\bar{\nu}_L\gamma_\mu\nu_L)(\bar{u}_R\gamma^\mu u_R)$
$\mathcal{O}_{\nu d}^{V,LR}$	$(\bar{\nu}_L\gamma_\mu\nu_L)\big(\bar{d}_R\gamma^\mu d_R\big)$
$\mathcal{O}_{eu}^{V,LR}$	$(\bar{e}_L\gamma_\mu e_L)(\bar{u}_R\gamma^\mu u_R)$
$\mathcal{O}_{ed}^{V,LR}$	$(\bar{e}_L\gamma_\mu e_L)\big(\bar{d}_R\gamma^\mu d_R\big)$
$\mathcal{O}_{ue}^{V,LR}$	$(\bar{u}_L\gamma_\mu u_L)(\bar{e}_R\gamma^\mu e_R)$
$\mathcal{O}_{de}^{V,LR}$	$\big(\bar{d}_L\gamma_\mu d_L\big)(\bar{e}_R\gamma^\mu e_R)$
$\mathcal{O}_{\nu edu}^{V,LR}$	$(\bar{\nu}_L\gamma_\mu e_L)\big(\bar{d}_R\gamma^\mu u_R\big)$
$\mathcal{O}_{uu}^{V1,LR}$	$(\bar{u}_L\gamma_\mu u_L)(\bar{u}_R\gamma^\mu u_R)$
$\mathcal{O}_{uu}^{V8,LR}$	$\big(\bar{u}_L\gamma_\mu T^A u_L\big)\big(\bar{u}_R\gamma^\mu T^A u_R\big)$
$\mathcal{O}_{ud}^{V1,LR}$	$(\bar{u}_L\gamma_\mu u_L)\big(\bar{d}_R\gamma^\mu d_R\big)$
$\mathcal{O}_{ud}^{V8,LR}$	$\big(\bar{u}_L\gamma_\mu T^A u_L\big)\big(\bar{d}_R\gamma^\mu T^A d_R\big)$
$\mathcal{O}_{du}^{V1,LR}$	$\big(\bar{d}_L\gamma_\mu d_L\big)(\bar{u}_R\gamma^\mu u_R)$
$\mathcal{O}_{du}^{V8,LR}$	$\big(\bar{d}_L\gamma_\mu T^A d_L\big)\big(\bar{u}_R\gamma^\mu T^A u_R\big)$
$\mathcal{O}_{dd}^{V1,LR}$	$\big(\bar{d}_L\gamma_\mu d_L\big)\big(\bar{d}_R\gamma^\mu d_R\big)$
$\mathcal{O}_{dd}^{V8,LR}$	$\big(\bar{d}_L\gamma_\mu T^A d_L\big)\big(\bar{d}_R\gamma^\mu T^A d_R\big)$
$\mathcal{O}_{uddu}^{V1,LR}$	$(\bar{u}_L\gamma_\mu d_L)\big(\bar{d}_R\gamma^\mu u_R\big)$
$\mathcal{O}_{uddu}^{V8,LR}$	$\big(\bar{u}_L\gamma_\mu T^A d_L\big)\big(\bar{d}_R\gamma^\mu T^A u_R\big)$

$(\bar{L}R)(\bar{L}R)$	
$\mathcal{O}_{ee}^{S,RR}$	$(\bar{e}_L e_R)(\bar{e}_L e_R)$
$\mathcal{O}_{eu}^{S,RR}$	$(\bar{e}_L e_R)(\bar{u}_L u_R)$
$\mathcal{O}_{eu}^{T,RR}$	$(\bar{e}_L\sigma_{\mu\nu} e_R)(\bar{u}_L\sigma^{\mu\nu} u_R)$
$\mathcal{O}_{ed}^{S,RR}$	$(\bar{e}_L e_R)\big(\bar{d}_L d_R\big)$
$\mathcal{O}_{ed}^{T,RR}$	$(\bar{e}_L\sigma_{\mu\nu} e_R)\big(\bar{d}_L\sigma^{\mu\nu} d_R\big)$
$\mathcal{O}_{\nu edu}^{S,RR}$	$(\bar{\nu}_L e_R)\big(\bar{d}_L u_R\big)$
$\mathcal{O}_{\nu edu}^{T,RR}$	$(\bar{\nu}_L\sigma_{\mu\nu} e_R)\big(\bar{d}_L\sigma^{\mu\nu} u_R\big)$
$\mathcal{O}_{uu}^{S1,RR}$	$(\bar{u}_L u_R)(\bar{u}_L u_R)$
$\mathcal{O}_{uu}^{S8,RR}$	$\big(\bar{u}_L T^A u_R\big)\big(\bar{u}_L T^A u_R\big)$
$\mathcal{O}_{ud}^{S1,RR}$	$(\bar{u}_L u_R)\big(\bar{d}_L d_R\big)$
$\mathcal{O}_{ud}^{S8,RR}$	$\big(\bar{u}_L T^A u_R\big)\big(\bar{d}_L T^A d_R\big)$
$\mathcal{O}_{dd}^{S1,RR}$	$\big(\bar{d}_L d_R\big)\big(\bar{d}_L d_R\big)$
$\mathcal{O}_{dd}^{S8,RR}$	$\big(\bar{d}_L T^A d_R\big)\big(\bar{d}_L T^A d_R\big)$
$\mathcal{O}_{uddu}^{S1,RR}$	$(\bar{u}_L d_R)\big(\bar{d}_L u_R\big)$
$\mathcal{O}_{uddu}^{S8,RR}$	$\big(\bar{u}_L T^A d_R\big)\big(\bar{d}_L T^A u_R\big)$

$(\bar{L}R)(\bar{R}L)$	
$\mathcal{O}_{eu}^{S,RL}$	$(\bar{e}_L e_R)(\bar{u}_R u_L)$
$\mathcal{O}_{ed}^{S,RL}$	$(\bar{e}_L e_R)\big(\bar{d}_R d_L\big)$
$\mathcal{O}_{\nu edu}^{S,RL}$	$(\bar{\nu}_L e_R)\big(\bar{d}_R u_L\big)$

Table 13: LEFT Baryon and/or Lepton number violating dimension-six operators. We use C to denote the Dirac charge conjugation matrix.

$\Delta L = 2$		$\Delta B = \Delta L = 1$		$\Delta B = -\Delta L = 1$	
$\mathcal{O}_{\nu e}^{S,LL}$	$\big(\nu_L^T C \nu_L\big)\,(\bar{e}_R e_L)$	$\mathcal{O}_{udd}^{S,LL}$	$\big(u_L^T C d_L\big)\,\big(d_L^T C \nu_L\big)$	$\mathcal{O}_{ddd}^{S,LL}$	$\big(d_L^T C d_L\big)\,(\bar{e}_R d_L)$
$\mathcal{O}_{\nu e}^{T,LL}$	$\big(\nu_L^T C \sigma_{\mu\nu} \nu_L\big)\,(\bar{e}_R \sigma^{\mu\nu} e_L)$	$\mathcal{O}_{duu}^{S,LL}$	$\big(d_L^T C u_L\big)\,\big(u_L^T C e_L\big)$	$\mathcal{O}_{udd}^{S,LR}$	$\big(u_L^T C d_L\big)\,(\bar{\nu}_L d_R)$
$\mathcal{O}_{\nu e}^{S,LR}$	$\big(\nu_L^T C \nu_L\big)\,(\bar{e}_L e_R)$	$\mathcal{O}_{uud}^{S,LR}$	$\big(u_L^T C u_L\big)\,\big(d_R^T C e_R\big)$	$\mathcal{O}_{ddu}^{S,LR}$	$\big(d_L^T C d_L\big)\,(\bar{\nu}_L u_R)$
$\mathcal{O}_{\nu u}^{S,LL}$	$\big(\nu_L^T C \nu_L\big)\,(\bar{u}_R u_L)$	$\mathcal{O}_{duu}^{S,LR}$	$\big(d_L^T C u_L\big)\,\big(u_R^T C e_R\big)$	$\mathcal{O}_{ddd}^{S,LR}$	$\big(d_L^T C d_L\big)\,(\bar{e}_L d_R)$
$\mathcal{O}_{\nu u}^{T,LL}$	$\big(\nu_L^T C \sigma_{\mu\nu} \nu_L\big)\,(\bar{u}_R \sigma^{\mu\nu} u_L)$	$\mathcal{O}_{uud}^{S,RL}$	$\big(u_R^T C u_R\big)\,\big(d_L^T C e_L\big)$	$\mathcal{O}_{ddd}^{S,RL}$	$\big(d_R^T C d_R\big)\,(\bar{e}_R d_L)$
$\mathcal{O}_{\nu u}^{S,LR}$	$\big(\nu_L^T C \nu_L\big)\,(\bar{u}_L u_R)$	$\mathcal{O}_{duu}^{S,RL}$	$\big(d_R^T C u_R\big)\,\big(u_L^T C e_L\big)$	$\mathcal{O}_{udd}^{S,RR}$	$\big(u_R^T C d_R\big)\,(\bar{\nu}_L d_R)$
$\mathcal{O}_{\nu d}^{S,LL}$	$\big(\nu_L^T C \nu_L\big)\,\big(\bar{d}_R d_L\big)$	$\mathcal{O}_{dud}^{S,RL}$	$\big(d_R^T C u_R\big)\,\big(d_L^T C \nu_L\big)$	$\mathcal{O}_{ddd}^{S,RR}$	$\big(d_R^T C d_R\big)\,(\bar{e}_L d_R)$
$\mathcal{O}_{\nu d}^{T,LL}$	$\big(\nu_L^T C \sigma_{\mu\nu} \nu_L\big)\,\big(\bar{d}_R \sigma^{\mu\nu} d_L\big)$	$\mathcal{O}_{ddu}^{S,RL}$	$\big(d_R^T C d_R\big)\,\big(u_L^T C \nu_L\big)$	$\Delta L = 4$	
$\mathcal{O}_{\nu d}^{S,LR}$	$\big(\nu_L^T C \nu_L\big)\,\big(\bar{d}_L d_R\big)$	$\mathcal{O}_{duu}^{S,RR}$	$\big(d_R^T C u_R\big)\,\big(u_R^T C e_R\big)$	$\mathcal{O}_{\nu\nu}^{S,LL}$	$\big(\nu_L^T C \nu_L\big)\,\big(\nu_L^T C \nu_L\big)$
$\mathcal{O}_{\nu edu}^{S,LL}$	$\big(\nu_L^T C e_L\big)\,\big(\bar{d}_R u_L\big)$				
$\mathcal{O}_{\nu edu}^{T,LL}$	$\big(\nu_L^T C \sigma_{\mu\nu} e_L\big)\,\big(\bar{d}_R \sigma^{\mu\nu} u_L\big)$				
$\mathcal{O}_{\nu edu}^{S,LR}$	$\big(\nu_L^T C e_L\big)\,\big(\bar{d}_L u_R\big)$				
$\mathcal{O}_{\nu edu}^{V,RL}$	$\big(\nu_L^T C \gamma_\mu e_R\big)\,\big(\bar{d}_L \gamma^\mu u_L\big)$				
$\mathcal{O}_{\nu edu}^{V,RR}$	$\big(\nu_L^T C \gamma_\mu e_R\big)\,\big(\bar{d}_R \gamma^\mu u_R\big)$				

Table 14: Index symmetry categories used in `DsixTools`.

Category	Meaning
0	0F scalar object
1	2F general 3 × 3 matrix
2	2F Hermitian matrix
3	2F symmetric matrix
4	2F antisymmetric matrix
5	4F general 3 × 3 × 3 × 3 object
6	4F two identical $\overline{\psi}\psi$ currents
7	4F two independent $\overline{\psi}\psi$ currents
8	4F two identical $\overline{\psi}\psi$ currents (ψ singlet)
9	4F symmetric current × general current
10	4F antisymmetric current × general current
11	4F SMEFT special case C_{qqql}
12	4F LEFT special case $L_{\nu\nu}^{S,LL}$
13	4F LEFT special case $L_{ddd}^{S,LL/RR}$

which define the LEFT beta functions β_i. We use a notation completely analogous to that in the SMEFT. The complete set of one-loop beta functions for the LEFT has been computed in Ref. [**?**]. They can be read off directly from DsixTools with the command β[*parameter*].

C SMEFT and LEFT parameters

In this Appendix we provide additional details about the variables used in `DsixTools`. These can be useful to properly read and write some variables or apply some global dispatches and substitution rules in a `Mathematica` session using `DsixTools`. We also introduce the notation used in `DsixTools` for the SMEFT and LEFT parameters.

It is well known that some of the 2- and 4-fermion operators in the SMEFT and the LEFT posess specific symmetries under the exchange of flavor indices. For instance, the flavour components of the SMEFT operator $Q_{\varphi e}$ form a Hermitian matrix, hence following the symmetry relation $[Q_{\varphi e}]_{ij} = [Q_{\varphi e}]^*_{ji}$, while the LEFT operator components of $\mathcal{O}_{\nu\gamma}$ form an antisymmetric matrix, hence following the symmetry relation $[\mathcal{O}_{\nu\gamma}]_{ij} = -[\mathcal{O}_{\nu\gamma}]_{ji}$. More complicated index symmetries exist for some of the 4-fermion operators. In all these cases, the number of independent operator components gets reduced, and thus the number of independent WCs. For example, the C_{ee} 4-fermion WC does not contain 81 ($= 3^4$) independent complex WCs, but just 21 real and 15 imaginary independent components. It is convenient to restrict the number of parameters considered in SMEFT or LEFT calculations to just the independent ones. In `DsixTools` we have followed this approach, dropping redundant WCs in all internal calculations by transforming the user input into two minimal bases of operators: the *independent basis* and the *symmetric basis*. These bases, which are described in Appendix C.3, have the same set of independent WCs, although with different numerical values. Since the number of independent WCs depends on the symmetry of the operators involved, it is sufficient to know the independent WCs for each index symmetry category of the operators in the SMEFT and LEFT. The different categories are given in Table 14. We see that, apart from the operators belonging to categories 0, 1 and 5, all other operators have index symmetries. Furthermore, there are two dimension-six operators with special symmetries, not shared by any other operator, Q_{qqql} and $\mathcal{O}_{\nu\nu}^{S,LL}$, and two operators as only representatives of the last index symmetry category, $\mathcal{O}_{ddd}^{S,LL}$ and $\mathcal{O}_{ddd}^{S,RR}$. Similarly, the dimension-five SMEFT operator $Q_{\ell\ell\varphi\varphi}$ and the dimension-three LEFT neutrino mass matrix M_ν are the only symmetric matrices, while the dimension-five LEFT operator $\mathcal{O}_{\nu\gamma}$ is the only antisymmetric matrix.

In Tables 15 and 16 we list the independent WCs contained in each category. This, combined with Tables 17 and 18, completely allows the user to determine the position of a given parameter in the `ParametersSMEFT` and `ParametersLEFT` arrays. In any case, we remind the reader that the functions `FindParametersSMEFT` and `FindParametersLEFT` can also be used for this purpose.

Table 15: Independent WCs in each 2F category. Numbers between curly brackets refer to the WC flavour indices. Elements in red denote real WCs.

	1	2	3	4
1	{1,1}	{1,1}	{1,1}	{1,2}
2	{1,2}	{1,2}	{1,2}	{1,3}
3	{1,3}	{1,3}	{1,3}	{2,3}
4	{2,1}	{2,2}	{2,2}	
5	{2,2}	{2,3}	{2,3}	
6	{2,3}	{3,3}	{3,3}	
7	{3,1}			
8	{3,2}			
9	{3,3}			

Table 16: Independent WCs in each 4F category. Elements in red denote real WCs.

	5	6	7	8	9	10	11	12	13
1	{1,1,1,1}	{1,1,1,1}	{1,1,1,1}	{1,1,1,1}	{1,1,1,1}	{1,2,1,1}	{1,1,1,1}	{1,1,2,2}	{1,2,1,1}
2	{1,1,1,2}	{1,1,1,2}	{1,1,1,2}	{1,1,1,2}	{1,1,1,2}	{1,2,1,2}	{1,1,1,2}	{1,1,3,3}	{1,2,1,2}
3	{1,1,1,3}	{1,1,1,3}	{1,1,1,3}	{1,1,1,3}	{1,1,1,3}	{1,2,1,3}	{1,1,1,3}	{2,2,3,3}	{1,2,1,3}
4	{1,1,2,1}	{1,1,2,2}	{1,1,2,2}	{1,1,2,2}	{1,1,2,1}	{1,2,2,1}	{1,1,2,1}	{1,1,2,3}	{1,2,2,1}
5	{1,1,2,2}	{1,1,2,3}	{1,1,2,3}	{1,1,2,3}	{1,1,2,2}	{1,2,2,2}	{1,1,2,2}	{1,2,2,3}	{1,2,2,2}
6	{1,1,2,3}	{1,1,3,3}	{1,1,3,3}	{1,1,3,3}	{1,1,2,3}	{1,2,2,3}	{1,1,2,3}	{1,2,3,3}	{1,2,2,3}
7	{1,1,3,1}	{1,2,1,2}	{1,2,1,1}	{1,2,1,2}	{1,1,3,1}	{1,2,3,1}	{1,1,3,1}		{1,2,3,1}
8	{1,1,3,2}	{1,2,1,3}	{1,2,1,2}	{1,2,1,3}	{1,1,3,2}	{1,2,3,2}	{1,1,3,2}		{1,2,3,2}
9	{1,1,3,3}	{1,2,2,1}	{1,2,1,3}	{1,2,2,2}	{1,1,3,3}	{1,2,3,3}	{1,1,3,3}		{1,2,3,3}
10	{1,2,1,1}	{1,2,2,2}	{1,2,2,1}	{1,2,2,3}	{1,2,1,1}	{1,3,1,1}	{1,2,1,1}		{1,3,1,1}
11	{1,2,1,2}	{1,2,2,3}	{1,2,2,2}	{1,2,3,2}	{1,2,1,2}	{1,3,1,2}	{1,2,1,2}		{1,3,1,2}
12	{1,2,1,3}	{1,2,3,1}	{1,2,2,3}	{1,2,3,3}	{1,2,1,3}	{1,3,1,3}	{1,2,1,3}		{1,3,1,3}
13	{1,2,2,1}	{1,2,3,2}	{1,2,3,1}	{1,3,1,3}	{1,2,2,1}	{1,3,2,1}	{1,2,2,1}		{1,3,2,1}
14	{1,2,2,2}	{1,2,3,3}	{1,2,3,2}	{1,3,2,3}	{1,2,2,2}	{1,3,2,2}	{1,2,2,2}		{1,3,2,2}
15	{1,2,2,3}	{1,3,1,3}	{1,2,3,3}	{1,3,3,3}	{1,2,2,3}	{1,3,2,3}	{1,2,2,3}		{1,3,2,3}
16	{1,2,3,1}	{1,3,2,2}	{1,3,1,1}	{2,2,2,2}	{1,2,3,1}	{1,3,3,1}	{1,2,3,1}		{1,3,3,1}
17	{1,2,3,2}	{1,3,2,3}	{1,3,1,2}	{2,2,2,3}	{1,2,3,2}	{1,3,3,2}	{1,2,3,2}		{1,3,3,2}
18	{1,2,3,3}	{1,3,3,1}	{1,3,1,3}	{2,2,3,3}	{1,2,3,3}	{1,3,3,3}	{1,2,3,3}		{1,3,3,3}
19	{1,3,1,1}	{1,3,3,2}	{1,3,2,1}	{2,3,2,3}	{1,3,1,1}	{2,3,1,1}	{1,3,1,1}		{2,3,1,2}
20	{1,3,1,2}	{1,3,3,3}	{1,3,2,2}	{2,3,3,3}	{1,3,1,2}	{2,3,1,2}	{1,3,1,2}		{2,3,1,3}
21	{1,3,1,3}	{2,2,2,2}	{1,3,2,3}	{3,3,3,3}	{1,3,1,3}	{2,3,1,3}	{1,3,1,3}		{2,3,2,2}
22	{1,3,2,1}	{2,2,2,3}	{1,3,3,1}		{1,3,2,1}	{2,3,2,1}	{1,3,2,1}		{2,3,2,3}
23	{1,3,2,2}	{2,2,3,3}	{1,3,3,2}		{1,3,2,2}	{2,3,2,2}	{1,3,2,2}		{2,3,3,2}
24	{1,3,2,3}	{2,3,2,3}	{1,3,3,3}		{1,3,2,3}	{2,3,2,3}	{1,3,2,3}		{2,3,3,3}
25	{1,3,3,1}	{2,3,3,2}	{2,2,1,1}		{1,3,3,1}	{2,3,3,1}	{1,3,3,1}		
26	{1,3,3,2}	{2,3,3,3}	{2,2,1,2}		{1,3,3,2}	{2,3,3,2}	{1,3,3,2}		
27	{1,3,3,3}	{3,3,3,3}	{2,2,1,3}		{1,3,3,3}	{2,3,3,3}	{1,3,3,3}		
28	{2,1,1,1}		{2,2,2,2}		{2,2,1,1}		{2,1,2,1}		
29	{2,1,1,2}		{2,2,2,3}		{2,2,1,2}		{2,1,2,2}		
30	{2,1,1,3}		{2,2,3,3}		{2,2,1,3}		{2,1,2,3}		
31	{2,1,2,1}		{2,3,1,1}		{2,2,2,1}		{2,1,3,1}		
32	{2,1,2,2}		{2,3,1,2}		{2,2,2,2}		{2,1,3,2}		
33	{2,1,2,3}		{2,3,1,3}		{2,2,2,3}		{2,1,3,3}		
34	{2,1,3,1}		{2,3,2,1}		{2,2,3,1}		{2,2,2,1}		
35	{2,1,3,2}		{2,3,2,2}		{2,2,3,2}		{2,2,2,2}		
36	{2,1,3,3}		{2,3,2,3}		{2,2,3,3}		{2,2,2,3}		
37	{2,2,1,1}		{2,3,3,1}		{2,3,1,1}		{2,2,3,1}		
38	{2,2,1,2}		{2,3,3,2}		{2,3,1,2}		{2,2,3,2}		
39	{2,2,1,3}		{2,3,3,3}		{2,3,1,3}		{2,2,3,3}		
40	{2,2,2,1}		{3,3,1,1}		{2,3,2,1}		{2,3,1,1}		

41	{2,2,2,2}	{3,3,1,2}	{2,3,2,2}	{2,3,1,2}
42	{2,2,2,3}	{3,3,1,3}	{2,3,2,3}	{2,3,1,3}
43	{2,2,3,1}	{3,3,2,2}	{2,3,3,1}	{2,3,2,1}
44	{2,2,3,2}	{3,3,2,3}	{2,3,3,2}	{2,3,2,2}
45	{2,2,3,3}	{3,3,3,3}	{2,3,3,3}	{2,3,2,3}
46	{2,3,1,1}		{3,3,1,1}	{2,3,3,1}
47	{2,3,1,2}		{3,3,1,2}	{2,3,3,2}
48	{2,3,1,3}		{3,3,1,3}	{2,3,3,3}
49	{2,3,2,1}		{3,3,2,1}	{3,1,3,1}
50	{2,3,2,2}		{3,3,2,2}	{3,1,3,2}
51	{2,3,2,3}		{3,3,2,3}	{3,1,3,3}
52	{2,3,3,1}		{3,3,3,1}	{3,2,3,1}
53	{2,3,3,2}		{3,3,3,2}	{3,2,3,2}
54	{2,3,3,3}		{3,3,3,3}	{3,2,3,3}
55	{3,1,1,1}			{3,3,3,1}
56	{3,1,1,2}			{3,3,3,2}
57	{3,1,1,3}			{3,3,3,3}
58	{3,1,2,1}			
59	{3,1,2,2}			
60	{3,1,2,3}			
61	{3,1,3,1}			
62	{3,1,3,2}			
63	{3,1,3,3}			
64	{3,2,1,1}			
65	{3,2,1,2}			
66	{3,2,1,3}			
67	{3,2,2,1}			
68	{3,2,2,2}			
69	{3,2,2,3}			
70	{3,2,3,1}			
71	{3,2,3,2}			
72	{3,2,3,3}			
73	{3,3,1,1}			
74	{3,3,1,2}			
75	{3,3,1,3}			
76	{3,3,2,1}			
77	{3,3,2,2}			
78	{3,3,2,3}			
79	{3,3,3,1}			
80	{3,3,3,2}			
81	{3,3,3,3}			

C.1 SMEFT parameters

Table 17 provides a complete list of the SMEFT parameters used in DsixTools. In addition to the SMEFT WCs, this includes the SM parameters (gauge couplings, Yukawa matrices and scalar and θ parameters). This table is particularly useful to identify the names given to the elements of 2- and 4-fermion WCs, as well as the corresponding beta functions, which can be readily obtained by evaluating β[parameter]. For instance, the beta function for the g_s gauge coupling is obtained by evaluating β[gs] and the beta function for the $[C_{\ell q}^{(1)}]_{2233}$ WC is obtained with β[Clq1[2,2,3,3]].

Table 17: SMEFT parameters. *Position* denotes the position of the parameter (or parameters for 2- and 4-fermion objects) in the ParametersSMEFT global array. *Type* indicates the type of parameter (with nF standing for n-fermion) and *Category* denotes the index symmetry category of the coefficient, being relevant for 2- and 4-fermion WCs.

Position	Parameter(s)	DsixTools name	Elements	Type	Category
1	g	g	-	0F	0
2	g'	gp	-	0F	0
3	g_s	gs	-	0F	0
4	λ	λ	-	0F	0
5	m^2	m2	-	0F	0
6-14	Γ_u	MGu	Gu[i,j]	2F	1
15-23	Γ_d	MGd	Gd[i,j]	2F	1
24-32	Γ_e	MGe	Ge[i,j]	2F	1
33	θ	θ	-	0F	0
34	θ'	θp	-	0F	0
35	θ_s	θs	-	0F	0
36	C_G	CG	-	0F	0
37	$C_{\widetilde{G}}$	CGtilde	-	0F	0
38	C_W	CW	-	0F	0
39	$C_{\widetilde{W}}$	CWtilde	-	0F	0
40	C_φ	CH	-	0F	0
41	$C_{\varphi\Box}$	CHbox	-	0F	0
42	$C_{\varphi D}$	CHD	-	0F	0
43	$C_{\varphi G}$	CHG	-	0F	0
44	$C_{\varphi B}$	CHB	-	0F	0
45	$C_{\varphi W}$	CHW	-	0F	0
46	$C_{\varphi WB}$	CHWB	-	0F	0
47	$C_{\varphi\widetilde{G}}$	CHGtilde	-	0F	0
48	$C_{\varphi\widetilde{B}}$	CHBtilde	-	0F	0

49	$C_{\varphi\widetilde{W}}$	CHWtilde	-	0F	0
50	$C_{\varphi\widetilde{W}B}$	CHWtildeB	-	0F	0
51-59	$C_{u\varphi}$	MCuH	CuH[i,j]	2F	1
60-68	$C_{d\varphi}$	MCdH	CdH[i,j]	2F	1
69-77	$C_{e\varphi}$	MCeH	CeH[i,j]	2F	1
78-86	C_{eW}	MCeW	CeW[i,j]	2F	1
87-95	C_{eB}	MCeB	CeB[i,j]	2F	1
96-104	C_{uG}	MCuG	CuG[i,j]	2F	1
105-113	C_{uW}	MCuW	CuW[i,j]	2F	1
114-122	C_{uB}	MCuB	CuB[i,j]	2F	1
123-131	C_{dG}	MCdG	CdG[i,j]	2F	1
132-140	C_{dW}	MCdW	CdW[i,j]	2F	1
141-149	C_{dB}	MCdB	CdB[i,j]	2F	1
150-155	$C^{(1)}_{\varphi\ell}$	MCHl1	CHl1[i,j]	2F	2
156-161	$C^{(3)}_{\varphi\ell}$	MCHl3	CHl3[i,j]	2F	2
162-167	$C_{\varphi e}$	MCHe	CHe[i,j]	2F	2
168-173	$C^{(1)}_{\varphi q}$	MCHq1	CHq1[i,j]	2F	2
174-179	$C^{(3)}_{\varphi q}$	MCHq3	CHq3[i,j]	2F	2
180-185	$C_{\varphi u}$	MCHu	CHu[i,j]	2F	2
186-191	$C_{\varphi d}$	MCHd]	CHd[i,j]	2F	2
192-200	$C_{\varphi ud}$	MCHud	CHud[i,j]	2F	1
201-227	$C_{\ell\ell}$	MCll	Cll[i,j,k,l]	4F	6
228-254	$C^{(1)}_{qq}$	MCqq1	Cqq1[i,j,k,l]	4F	6
255-281	$C^{(3)}_{qq}$	MCqq3	Cqq3[i,j,k,l]	4F	6
282-326	$C^{(1)}_{\ell q}$	MClq1	Clq1[i,j,k,l]	4F	7
327-371	$C^{(3)}_{\ell q}$	MClq3	Clq3[i,j,k,l]	4F	7
372-392	C_{ee}	MCee	Cee[i,j,k,l]	4F	8
393-419	C_{uu}	MCuu	Cuu[i,j,k,l]	4F	6
420-446	C_{dd}	MCdd	Cdd[i,j,k,l]	4F	6
447-491	C_{eu}	MCeu	Ceu[i,j,k,l]	4F	7
492-536	C_{ed}	MCed	Ced[i,j,k,l]	4F	7
537-581	$C^{(1)}_{ud}$	MCud1	Cud1[i,j,k,l]	4F	7
582-626	$C^{(8)}_{ud}$	MCud8	Cud8[i,j,k,l]	4F	7
627-671	$C_{\ell e}$	MCle	Cle[i,j,k,l]	4F	7
672-716	$C_{\ell u}$	MClu	Clu[i,j,k,l]	4F	7
717-761	$C_{\ell d}$	MCld	Cld[i,j,k,l]	4F	7
762-806	C_{qe}	MCqe	Cqe[i,j,k,l]	4F	7
807-851	$C^{(1)}_{qu}$	MCqu1	Cqu1[i,j,k,l]	4F	7
852-896	$C^{(8)}_{qu}$	MCqu8	Cqu8[i,j,k,l]	4F	7

897-941	$C_{qd}^{(1)}$	MCqd1	Cqd1[i,j,k,l]	4F	7
942-986	$C_{qd}^{(8)}$	MCqd8	Cqd8[i,j,k,l]	4F	7
987-1067	$C_{\ell edq}$	MCledq	Cledq[i,j,k,l]	4F	5
1068-1148	$C_{quqd}^{(1)}$	MCquqd1	Cquqd1[i,j,k,l]	4F	5
1149-1229	$C_{quqd}^{(8)}$	MCquqd8	Cquqd8[i,j,k,l]	4F	5
1230-1310	$C_{\ell equ}^{(1)}$	MClequ1	Clequ1[i,j,k,l]	4F	5
1311-1391	$C_{\ell equ}^{(3)}$	MClequ3	Clequ3[i,j,k,l]	4F	5
1392-1472	$C_{duq\ell}$	MCduql	Cduql[i,j,k,l]	4F	5
1473-1526	C_{qque}	MCqque	Cqque[i,j,k,l]	4F	9
1527-1583	$C_{qqq\ell}$	MCqqql	Cqqql[i,j,k,l]	4F	11
1584-1664	C_{duue}	MCduue	Cduue[i,j,k,l]	4F	5
1665-1670	$C_{\ell\ell\varphi\varphi}$	MCllHH	CllHH[i,j]	2F	3

C.2 LEFT parameters

Table 18 provides a complete list of the LEFT parameters used in DsixTools. In addition to the LEFT WCs, this includes the QCD and QED parameters (gauge couplings, fermion mass matrices and θ parameters). This table is particularly useful to identify the names given to the elements of 2- and 4-fermion WCs, as well as the corresponding beta functions, which can be readily obtained by evaluating β[parameter]. For instance, the beta function for the e gauge coupling is obtained by evaluating β[eQED] and the beta function for the $[L_{dG}]_{22}$ WC is obtained with β[LdG[2,2]].

Table 18: LEFT parameters. *Position* denotes the position of the parameter (or parameters for 2- and 4-fermion objects) in the ParametersLEFT global array. *Type* indicates the type of parameter (with nF standing for n-fermion) and *Category* denotes the index symmetry category of the coefficient, being relevant for 2- and 4-fermion WCs.

Position	Parameter(s)	DsixTools name	Elements	Type	Category
1	g_s	gQCD	-	0F	0
2	e	eQED	-	0F	0
3	θ_{QCD}	θQCD	-	0F	0
4	θ_{QED}	θQED	-	0F	0
5-10	M_ν	MMν	Mν[i,j]	2F	3
11-19	M_e	MMe	Me[i,j]	2F	1
20-28	M_u	MMu	Mu[i,j]	2F	1
29-37	M_d	MMd	Md[i,j]	2F	1
38	L_G	LG	-	0F	0
39	$L_{\widetilde{G}}$	LGtilde	-	0F	0

40-42	$L_{\nu\gamma}$	MLνγ	Lνγ[i,j]	2F	4
43-51	$L_{e\gamma}$	MLeγ	Leγ[i,j]	2F	1
52-60	$L_{u\gamma}$	MLuγ	Luγ[i,j]	2F	1
61-69	$L_{d\gamma}$	MLdγ	Ldγ[i,j]	2F	1
70-78	L_{uG}	MLuG	LuG[i,j]	2F	1
79-87	L_{dG}	MLdG	LdG[i,j]	2F	1
88-108	$L_{\nu\nu}^{V,LL}$	MLννVLL	LννVLL[i,j,k,l]	4F	8
109-129	$L_{ee}^{V,LL}$	MLeeVLL	LeeVLL[i,j,k,l]	4F	8
130-210	$L_{\nu e}^{V,LL}$	MLνeVLL	LνeVLL[i,j,k,l]	4F	5
211-291	$L_{\nu u}^{V,LL}$	MLνuVLL	LνuVLL[i,j,k,l]	4F	5
292-372	$L_{\nu d}^{V,LL}$	MLνdVLL	LνdVLL[i,j,k,l]	4F	5
373-453	$L_{eu}^{V,LL}$	MLeuVLL	LeuVLL[i,j,k,l]	4F	5
454-534	$L_{ed}^{V,LL}$	MLedVLL	LedVLL[i,j,k,l]	4F	5
535-615	$L_{\nu edu}^{V,LL}$	MLνeduVLL	LνeduVLL[i,j,k,l]	4F	5
616-642	$L_{uu}^{V,LL}$	MLuuVLL	LuuVLL[i,j,k,l]	4F	6
643-669	$L_{dd}^{V,LL}$	MLddVLL	LddVLL[i,j,k,l]	4F	6
670-750	$L_{ud}^{V1,LL}$	MLudV1LL	LudV1LL[i,j,k,l]	4F	5
751-831	$L_{ud}^{V8,LL}$	MLudV8LL	LudV8LL[i,j,k,l]	4F	5
832-852	$L_{ee}^{V,RR}$	MLeeVRR	LeeVRR[i,j,k,l]	4F	8
853-933	$L_{eu}^{V,RR}$	MLeuVRR	LeuVRR[i,j,k,l]	4F	5
934-1014	$L_{ed}^{V,RR}$	MLedVRR	LedVRR[i,j,k,l]	4F	5
1015-1041	$L_{uu}^{V,RR}$	MLuuVRR	LuuVRR[i,j,k,l]	4F	6
1042-1068	$L_{dd}^{V,RR}$	MLddVRR	LddVRR[i,j,k,l]	4F	6
1069-1149	$L_{ud}^{V1,RR}$	MLudV1RR	LudV1RR[i,j,k,l]	4F	5
1150-1230	$L_{ud}^{V8,RR}$	MLudV8RR	LudV8RR[i,j,k,l]	4F	5
1231-1311	$L_{\nu e}^{V,LR}$	MLνeVLR	LνeVLR[i,j,k,l]	4F	5
1312-1392	$L_{ee}^{V,LR}$	MLeeVLR	LeeVLR[i,j,k,l]	4F	5
1393-1473	$L_{\nu u}^{V,LR}$	MLνuVLR	LνuVLR[i,j,k,l]	4F	5
1474-1554	$L_{\nu d}^{V,LR}$	MLνdVLR	LνdVLR[i,j,k,l]	4F	5
1555-1635	$L_{eu}^{V,LR}$	MLeuVLR	LeuVLR[i,j,k,l]	4F	5
1636-1716	$L_{ed}^{V,LR}$	MLedVLR	LedVLR[i,j,k,l]	4F	5
1717-1797	$L_{ue}^{V,LR}$	MLueVLR	LueVLR[i,j,k,l]	4F	5
1798-1878	$L_{de}^{V,LR}$	MLdeVLR	LdeVLR[i,j,k,l]	4F	5
1879-1959	$L_{\nu edu}^{V,LR}$	MLνeduVLR	LνeduVLR[i,j,k,l]	4F	5
1960-2040	$L_{uu}^{V1,LR}$	MLuuV1LR	LuuV1LR[i,j,k,l]	4F	5
2041-2121	$L_{uu}^{V8,LR}$	MLuuV8LR	LuuV8LR[i,j,k,l]	4F	5
2122-2202	$L_{ud}^{V1,LR}$	MLudV1LR	LudV1LR[i,j,k,l]	4F	5
2203-2283	$L_{ud}^{V8,LR}$	MLudV8LR	LudV8LR[i,j,k,l]	4F	5
2284-2364	$L_{du}^{V1,LR}$	MLduV1LR	LduV1LR[i,j,k,l]	4F	5
2365-2445	$L_{du}^{V8,LR}$	MLduV8LR	LduV8LR[i,j,k,l]	4F	5

2446-2526	$L_{dd}^{V1,LR}$	MLddV1LR	LddV1LR[i,j,k,l]	4F	5
2527-2607	$L_{dd}^{V8,LR}$	MLddV8LR	LddV8LR[i,j,k,l]	4F	5
2608-2688	$L_{uddu}^{V1,LR}$	MLudduV1LR	LudduV1LR[i,j,k,l]	4F	5
2689-2769	$L_{uddu}^{V8,LR}$	MLudduV8LR	LudduV8LR[i,j,k,l]	4F	5
2770-2796	$L_{ee}^{S,RR}$	MLeeSRR	LeeSRR[i,j,k,l]	4F	6
2797-2877	$L_{eu}^{S,RR}$	MLeuSRR	LeuSRR[i,j,k,l]	4F	5
2878-2958	$L_{eu}^{T,RR}$	MLeuTRR	LeuTRR[i,j,k,l]	4F	5
2959-3039	$L_{ed}^{S,RR}$	MLedSRR	LedSRR[i,j,k,l]	4F	5
3040-3120	$L_{ed}^{T,RR}$	MLedTRR	LedTRR[i,j,k,l]	4F	5
3121-3201	$L_{\nu edu}^{S,RR}$	MLνeduSRR	LνeduSRR[i,j,k,l]	4F	5
3202-3282	$L_{\nu edu}^{T,RR}$	MLνeduTRR	LνeduTRR[i,j,k,l]	4F	5
3283-3309	$L_{uu}^{S1,RR}$	MLuuS1RR	LuuS1RR[i,j,k,l]	4F	6
3310-3336	$L_{uu}^{S8,RR}$	MLuuS8RR	LuuS8RR[i,j,k,l]	4F	6
3337-3417	$L_{ud}^{S1,RR}$	MLudS1RR	LudS1RR[i,j,k,l]	4F	5
3418-3498	$L_{ud}^{S8,RR}$	MLudS8RR	LudS8RR[i,j,k,l]	4F	5
3499-3525	$L_{dd}^{S1,RR}$	MLddS1RR	LddS1RR[i,j,k,l]	4F	6
3526-3552	$L_{dd}^{S8,RR}$	MLddS8RR	LddS8RR[i,j,k,l]	4F	6
3553-3633	$L_{uddu}^{S1,RR}$	MLudduS1RR	LudduS1RR[i,j,k,l]	4F	5
3634-3714	$L_{uddu}^{S8,RR}$	MLudduS8RR	LudduS8RR[i,j,k,l]	4F	5
3715-3795	$L_{eu}^{S,RL}$	MLeuSRL	LeuSRL[i,j,k,l]	4F	5
3796-3876	$L_{ed}^{S,RL}$	MLedSRL	LedSRL[i,j,k,l]	4F	5
3877-3957	$L_{\nu edu}^{S,RL}$	MLνeduSRL	LνeduSRL[i,j,k,l]	4F	5
3958-3963	$L_{\nu\nu}^{S,LL}$	MLννSLL	LννSLL[i,j,k,l]	4F	12
3964-4017	$L_{\nu e}^{S,LL}$	MLνeSLL	LνeSLL[i,j,k,l]	4F	9
4018-4044	$L_{\nu e}^{T,LL}$	MLνeTLL	LνeTLL[i,j,k,l]	4F	10
4045-4098	$L_{\nu e}^{S,LR}$	MLνeSLR	LνeSLR[i,j,k,l]	4F	9
4099-4152	$L_{\nu u}^{S,LL}$	MLνuSLL	LνuSLL[i,j,k,l]	4F	9
4153-4179	$L_{\nu u}^{T,LL}$	MLνuTLL	LνuTLL[i,j,k,l]	4F	10
4180-4233	$L_{\nu u}^{S,LR}$	MLνuSLR	LνuSLR[i,j,k,l]	4F	9
4234-4287	$L_{\nu d}^{S,LL}$	MLνdSLL	LνdSLL[i,j,k,l]	4F	9
4288-4314	$L_{\nu d}^{T,LL}$	MLνdTLL	LνdTLL[i,j,k,l]	4F	10
4315-4368	$L_{\nu d}^{S,LR}$	MLνdSLR	LνdSLR[i,j,k,l]	4F	9
4369-4449	$L_{\nu edu}^{S,LL}$	MLνeduSLL	LνeduSLL[i,j,k,l]	4F	5
4450-4530	$L_{\nu edu}^{T,LL}$	MLνeduTLL	LνeduTLL[i,j,k,l]	4F	5
4531-4611	$L_{\nu edu}^{S,LR}$	MLνeduSLR	LνeduSLR[i,j,k,l]	4F	5
4612-4692	$L_{\nu edu}^{V,RL}$	MLνeduVRL	LνeduVRL[i,j,k,l]	4F	5
4693-4773	$L_{\nu edu}^{V,RR}$	MLνeduVRR	LνeduVRR[i,j,k,l]	4F	5
4774-4854	$L_{udd}^{S,LL}$	MLuddSLL	LuddSLL[i,j,k,l]	4F	5
4855-4935	$L_{duu}^{S,LL}$	MLduuSLL	LduuSLL[i,j,k,l]	4F	5
4936-4962	$L_{uud}^{S,LR}$	MLuudSLR	LuudSLR[i,j,k,l]	4F	10

4963-5043	$L_{duu}^{S,LR}$	`MLduuSLR`	`LduuSLR[i,j,k,l]`	4F	5
5044-5070	$L_{uud}^{S,RL}$	`MLuudSRL`	`LuudSRL[i,j,k,l]`	4F	10
5071-5151	$L_{duu}^{S,RL}$	`MLduuSRL`	`LduuSRL[i,j,k,l]`	4F	5
5152-5232	$L_{dud}^{S,RL}$	`MLdudSRL`	`LdudSRL[i,j,k,l]`	4F	5
5233-5259	$L_{ddu}^{S,RL}$	`MLdduSRL`	`LdduSRL[i,j,k,l]`	4F	10
5260-5340	$L_{duu}^{S,RR}$	`MLduuSRR`	`LduuSRR[i,j,k,l]`	4F	5
5341-5364	$L_{ddd}^{S,LL}$	`MLdddSLL`	`LdddSLL[i,j,k,l]`	4F	13
5365-5445	$L_{udd}^{S,LR}$	`MLuddSLR`	`LuddSLR[i,j,k,l]`	4F	5
5446-5472	$L_{ddu}^{S,LR}$	`MLdduSLR`	`LdduSLR[i,j,k,l]`	4F	10
5473-5499	$L_{ddd}^{S,LR}$	`MLdddSLR`	`LdddSLR[i,j,k,l]`	4F	10
5500-5526	$L_{ddd}^{S,RL}$	`MLdddSRL`	`LdddSRL[i,j,k,l]`	4F	10
5527-5607	$L_{udd}^{S,RR}$	`MLuddSRR`	`LuddSRR[i,j,k,l]`	4F	5
5608-5631	$L_{ddd}^{S,RR}$	`MLdddSRR`	`LdddSRR[i,j,k,l]`	4F	13

C.3 The symmetric and independent bases

`DsixTools` allows the user to introduce an arbitrary input for the WCs of the SMEFT in the Warsaw basis, and of the LEFT in the San Diego basis. In order to work only with independent parameters two different operator bases are used in `DsixTools` that drop all redundant WCs, and the user input and the results obtained from it can be expressed in terms of any of them. The first non-redundant basis, the *independent basis*, contains only the WCs with the flavour indices as listed in Tables 15 and 16 for each symmetry category, all other WCs being set to zero. In the second minimal basis, the *symmetric basis*, the redundancies in the WCs are removed by imposing that the latter follow the same symmetry relations as the corresponding operators. This is a convenient choice since the index symmetry of the operators is translated to the corresponding WCs. In order to simplify intermediate calculations done in this basis (*e.g.* in the RGEs) only the independent WCs listed for each category in Tables 15 and 16 are used. But unlike in the independent basis, the rest of WCs do not vanish but relate to the former following the same symmetry relations as the operators of the corresponding category. For instance, if we consider a 4-fermion operator with two identical $\bar{\psi}\psi$ currents (ψ singlet) (*i.e.* either Q_{ee} in the SMEFT, or $\mathcal{O}_{ee}^{V,RR}$, $\mathcal{O}_{ee}^{V,LL}$ and $\mathcal{O}_{\nu\nu}^{V,LL}$ in the LEFT), which belongs to category 8 in Table 14,

$$\sum_{prst} C_{prst}\, Q_{prst}\,, \tag{C.1}$$

its WCs in the symmetric basis fulfill the relations $C_{stpr} = C_{prst}$ (because the two flavour currents are identical), $C_{rpts} = C_{prst}$ (due to hermiticity), and $C_{ptsr} = C_{prst}$ (as a consequence of the Fierz identity satisfied by the flavour components of the operator). Note that the sum in (C.1) runs over all posible values of the fermion flavour indices (p, r, s, t). The same operator in the independent basis reads, however,

$$\sum_{\{prst\}\in\,\mathrm{cat.}\,8} \widetilde{C}_{prst}\, Q_{prst}\,, \tag{C.2}$$

where now the sum extends only to the 21 independent components listed under category 8 of Table 16, and all other $\widetilde{C}_{prst}$ vanish.

It is straightforward to relate the WCs in the symmetric basis to those of the independent basis by using the symmetry relations satisfied by the operators. Let us provide an explicit example for illustration. Consider the contribution to the Lagrangian of the operator $\mathcal{O}^{S,LL}_{\nu\nu,prst} = \left(\nu^T_{L,p} C \nu_{L,r}\right)\left(\nu^T_{L,s} C \nu_{L,t}\right)$ of the LEFT, which belongs to the symmetry category 12. Its flavour components are symmetric under the exchange of indices $p \leftrightarrow r$, $s \leftrightarrow t$ and $(p, r) \leftrightarrow (s, t)$, and further satisfy the Fierz identity $\mathcal{O}_{prst} = -\mathcal{O}_{ptsr} - \mathcal{O}_{trsp}$. These relations reduce the number of independent components to just six. In the symmetric basis the contribution of this operator to the Lagrangian reads

$$\sum_{prst} C_{prst}\, \mathcal{O}^{S,LL}_{\nu\nu,\,prst}\,, \tag{C.3}$$

where the C_{prst} inherit the same index symmetries as those of the operator. Using those we can relate the 81 flavour components to the six independent ones chosen for category 12 (see Table 16). In this way, (C.3) reduces to

$$\begin{aligned} &3\, C_{1122}\, \mathcal{O}^{S,LL}_{\nu\nu,1122} + 6\, C_{1123}\, \mathcal{O}^{S,LL}_{\nu\nu,1123} + 3\, C_{1133}\, \mathcal{O}^{S,LL}_{\nu\nu,1133} \\ &+ 24\, C_{1223}\, \mathcal{O}^{S,LL}_{\nu\nu,1223} + 6\, C_{1233}\, \mathcal{O}^{S,LL}_{\nu\nu,1233} + 3\, C_{2233}\, \mathcal{O}^{S,LL}_{\nu\nu,2233}\,. \end{aligned} \tag{C.4}$$

(C.4) matches the form of this operator in the independent basis, and thus allow us to read off the WCs in that basis in terms of the symmetric basis WCs:

$$\widetilde{C}_{1122} = 3\, C_{1122}\ , \widetilde{C}_{1123} = 6\, C_{1123}\ , \widetilde{C}_{1133} = 3\, C_{1133}\ , \tag{C.5}$$

$$\widetilde{C}_{1223} = 24\, C_{1223}\ , \widetilde{C}_{1233} = 6\, C_{1233}\ , \widetilde{C}_{2233} = 3\, C_{2233}\,. \tag{C.6}$$

D Evolution matrix formalism

`DsixTools` 2.0 provides a new and much faster method of solving the RGE equations that relies on an semi-analytical solution of the RGE equations. To explain this method, we focus on the case where only dimension four and dimension six operators are present, and discuss the addition of dimension five operators at the end. The SMEFT and LEFT RGE equations can then be generically written as

$$\frac{dC_i^{(4)}(t)}{dt} = \frac{1}{16\pi^2}\, \gamma_{ij}^{(4)}(C_k^{(4)}, C_k^{(6)})\, C_j^{(4)}(t)\,, \tag{D.1}$$

$$\frac{dC_i^{(6)}(t)}{dt} = \frac{1}{16\pi^2}\,\gamma_{ij}^{(6)}(C_k^{(4)})\,C_j^{(6)}(t)\,, \tag{D.2}$$

where i, j, k span the number of EFT operators, $t \equiv \ln\mu$, and γ is the anomalous dimension matrix (ADM). The superindices (4) and (6) denote, respectively, quantities associated to dimension four and six operators, and we have neglected contributions from $C_k^{(6)}$ in $\gamma^{(6)}$, since these correspond to higher orders in the EFT expansion. An analytical to solution to this system of coupled differential equations is not known, and one is generally forced to solve it numerically. Given the large number of equations involved, such numerical solution can be relatively slow. However, it is important to note that (D.2) still contains contributions that are higher order in the EFT expansion. Indeed, by noting that $C_k^{(6)} \sim \mathcal{O}(1/\Lambda^2)$, we can rewrite (D.1) as

$$\frac{dC_i^{(4)}(t)}{dt} = \frac{1}{16\pi^2}\,\gamma_{ij}^{(4)}(C_k^{(4)})\,C_j^{(4)}(t) + \mathcal{O}(1/\Lambda^2)\,, \tag{D.3}$$

with $\gamma_{ij}^{(4)}(C_k^{(4)}) \equiv \gamma_{ij}^{(4)}(C_k^{(4)}, 0)$. These equations correspond to the SM (or QCD and QED) RGE equations, and $\gamma_{ij}^{(4)}(C_k^{(4)})$ is known up to three loops [? ? ? ?] and even up to five loops in QCD for the quark masses and QCD coupling [? ? ?]. The numerical solution of this system of equations is much faster, given the reduced number of $C_k^{(4)}$ coefficients, and only needs to be performed once for a given set of experimental inputs. As a result, we get

$$C_k^{(4)}(t) = \hat{C}_k^{(4)}(t) + \mathcal{O}(1/\Lambda^2)\,, \tag{D.4}$$

with $\hat{C}_k^{(4)}(t)$ being interpolating functions obtained from the numerical solution of (D.3). Using this solution, we can rewrite (D.2) as

$$\frac{dC_i^{(6)}(t)}{dt} = \frac{1}{16\pi^2}\,\gamma_{ij}^{(6)}(\hat{C}_k^{(4)})\,C_j^{(6)}(t) + \mathcal{O}(1/\Lambda^4) \equiv \hat{\gamma}_{ij}^{(6)}(t)\,C_j^{(6)}(t) + \mathcal{O}(1/\Lambda^4)\,, \tag{D.5}$$

such that, up to corrections that are higher order in the EFT expansion, the ADM is just a function of t, completely fixed in terms of the interpolating functions $\hat{C}_k^{(4)}(t)$. Neglecting terms of $\mathcal{O}(1/\Lambda^2)$, the system of differential equations in (D.5) is solved by

$$C_i^{(6)}(t) = U_{ij}^{(6)}(t, t_0)\,C_j^{(6)}(t_0)\,, \tag{D.6}$$

where $t_0 \equiv \ln\mu_0$, with μ_0 being the input scale of the dimension-six WCs, and $U^{(6)}$ is an evolution matrix that is given in terms of $\hat{\gamma}^{(6)}(t)$ by [11]

$$\begin{aligned} U^{(6)}(t, t_0) &= \mathcal{T}\left\{\exp\left(\int_{t_0}^{t} \hat{\gamma}^{(6)}(\omega)\,d\omega\right)\right\} \\ &= \sum_{n=0}^{\infty} \int_{t_0}^{t}\int_{t_0}^{\omega_n}\int_{t_0}^{\omega_{n-1}}\cdots\int_{t_0}^{\omega_2} \hat{\gamma}^{(6)}(\omega_1)\ldots\hat{\gamma}^{(6)}(\omega_n)\,d\omega_1\ldots d\omega_n\,. \end{aligned} \tag{D.7}$$

[11] In practice, it proves more convenient to determine the evolution matrix in (D.6) by numerically solving (D.5) for a set of linearly independent $C_j^{(6)}(t_0)$ test inputs, rather than by using (D.7).

Obtaining the evolution matrix is computationally expensive. However, since it is independent of the dimension-six input, it only needs to be determined once (for a given set of SM inputs). DsixTools 2.0 already contains a pre-computed evolution matrix for the inputs given in Table 1. Once the evolution matrix is known, the evaluation of (D.6) is very fast. The solution for $C_i^{(6)}(t)$ in (D.6) can then be plugged into the equations for the dimension-four WCs in (D.1). These equations need to be solved numerically, but given the small number of equations, obtaining this numerical solution is considerably faster than solving the whole system.

Finally, we comment on the inclusion of dimension five operators, since these can potentially modify the method discussed here. The only dimension five operator in the SMEFT is the Weinberg operator. Since its WC is expected to be very small, given the smallness of the neutrino masses, we neglect its mixing to dimension-six SMEFT operators, which requires a double insertion of this operator. Once this contribution is neglected, the evolution matrix formalism can be trivially extended to include also the Weinberg operator. In the case of the LEFT, the presence of dimension-five dipole could be addressed by extending the above procedure order-by-order. The end result would be a numerical evolution matrix which takes into account the effect of double dipole insertions in the running of $C^{(6)}$. What we do is to produce the numerical evolution matrix neglecting double dipole insertions in the beta functions, and test the results of running with this evolution matrix against the exact results. We find that the agreement is numerically very accurate for all practical cases. However, the user should keep this in mind when considering applications with large contributions to dipole operators. In such situations it might be wise to compare the results of RGEsMethod=3 with those obtained with RGEsMethod=1 in a few cases. If significant effects from double dipole insertions are found, then running with RGEsMethod=1 would be advised.

References

[1] W. Buchmuller and D. Wyler, *Effective Lagrangian Analysis of New Interactions and Flavor Conservation*, *Nucl. Phys. B* **268** (1986) 621–653.

[2] E. E. Jenkins, A. V. Manohar and P. Stoffer, *Low-Energy Effective Field Theory below the Electroweak Scale: Operators and Matching*, *JHEP* **03** (2018) 016, [1709.04486].

[3] B. Grzadkowski, M. Iskrzynski, M. Misiak and J. Rosiek, *Dimension-Six Terms in the Standard Model Lagrangian*, *JHEP* **10** (2010) 085, [1008.4884].

[4] 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, [1308.2627].

[5] 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, [1310.4838].

[6] 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* **04** (2014) 159, [1312.2014].

[7] R. Alonso, H.-M. Chang, E. E. Jenkins, A. V. Manohar and B. Shotwell, *Renormalization group evolution of dimension-six baryon number violating operators*, *Phys. Lett. B* **734** (2014) 302–307, [1405.0486].

[8] E. E. Jenkins, A. V. Manohar and P. Stoffer, *Low-Energy Effective Field Theory below the Electroweak Scale: Anomalous Dimensions*, *JHEP* **01** (2018) 084, [1711.05270].

[9] W. Dekens and P. Stoffer, *Low-energy effective field theory below the electroweak scale: matching at one loop*, *JHEP* **10** (2019) 197, [1908.05295].

[10] B. Henning, X. Lu and H. Murayama, *How to use the Standard Model effective field theory*, *JHEP* **01** (2016) 023, [1412.1837].

[11] A. Drozd, J. Ellis, J. Quevillon and T. You, *The Universal One-Loop Effective Action*, *JHEP* **03** (2016) 180, [1512.03003].

[12] F. del Aguila, Z. Kunszt and J. Santiago, *One-loop effective lagrangians after matching*, *Eur. Phys. J. C* **76** (2016) 244, [1602.00126].

[13] M. Boggia, R. Gomez-Ambrosio and G. Passarino, *Low energy behaviour of standard model extensions*, *JHEP* **05** (2016) 162, [1603.03660].

[14] B. Henning, X. Lu and H. Murayama, *One-loop Matching and Running with Covariant Derivative Expansion*, *JHEP* **01** (2018) 123, [1604.01019].

[15] S. A. R. Ellis, J. Quevillon, T. You and Z. Zhang, *Mixed heavy–light matching in the Universal One-Loop Effective Action*, *Phys. Lett. B* **762** (2016) 166–176, [1604.02445].

[16] J. Fuentes-Martin, J. Portoles and P. Ruiz-Femenia, *Integrating out heavy particles with functional methods: a simplified framework*, *JHEP* **09** (2016) 156, [1607.02142].

[17] Z. Zhang, *Covariant diagrams for one-loop matching*, *JHEP* **05** (2017) 152, [1610.00710].

[18] S. A. R. Ellis, J. Quevillon, T. You and Z. Zhang, *Extending the Universal One-Loop Effective Action: Heavy-Light Coefficients*, *JHEP* **08** (2017) 054, [1706.07765].

[19] M. Krämer, B. Summ and A. Voigt, *Completing the scalar and fermionic Universal One-Loop Effective Action*, *JHEP* **01** (2020) 079, [1908.04798].

[20] S. A. Ellis, J. Quevillon, P. N. H. Vuong, T. You and Z. Zhang, *The Fermionic Universal One-Loop Effective Action*, 2006.16260.

[21] J. Aebischer, M. Fael, A. Lenz, M. Spannowsky and J. Virto, *Computing Tools for the SMEFT*, 1910.11003.

[22] A. Celis, J. Fuentes-Martin, A. Vicente and J. Virto, *DsixTools: The Standard Model Effective Field Theory Toolkit*, *Eur. Phys. J. C* **77** (2017) 405, [1704.04504].

[23] B. Gripaios and D. Sutherland, *DEFT: A program for operators in EFT*, *JHEP* **01** (2019) 128, [1807.07546].

[24] J. C. Criado, *BasisGen: automatic generation of operator bases*, *Eur. Phys. J. C* **79** (2019) 256, [1901.03501].

[25] A. Dedes, M. Paraskevas, J. Rosiek, K. Suxho and L. Trifyllis, *SmeftFR – Feynman rules generator for the Standard Model Effective Field Theory*, *Comput. Phys. Commun.* **247** (2020) 106931, [1904.03204].

[26] J. C. Criado, *MatchingTools: a Python library for symbolic effective field theory calculations*, *Comput. Phys. Commun.* **227** (2018) 42–50, [1710.06445].

[27] S. Das Bakshi, J. Chakrabortty and S. K. Patra, *CoDEx: Wilson coefficient calculator connecting SMEFT to UV theory*, *Eur. Phys. J. C* **79** (2019) 21, [1808.04403].

[28] J. Aebischer, J. Kumar and D. M. Straub, *Wilson: a Python package for the running and matching of Wilson coefficients above and below the electroweak scale*, *Eur. Phys. J. C* **78** (2018) 1026, [1804.05033].

[29] N. P. Hartland, F. Maltoni, E. R. Nocera, J. Rojo, E. Slade, E. Vryonidou et al., *A Monte Carlo global analysis of the Standard Model Effective Field Theory: the top quark sector*, *JHEP* **04** (2019) 100, [1901.05965].

[30] J. Aebischer, J. Kumar, P. Stangl and D. M. Straub, *A Global Likelihood for Precision Constraints and Flavour Anomalies*, *Eur. Phys. J. C* **79** (2019) 509, [1810.07698].

[31] D. van Dyk et al., "EOS — A HEP program for Flavor Observables." https://eos.github.io.

[32] D. M. Straub, *flavio: a Python package for flavour and precision phenomenology in the Standard Model and beyond*, 1810.08132.

[33] I. Brivio, Y. Jiang and M. Trott, *The SMEFTsim package, theory and tools*, *JHEP* **12** (2017) 070, [1709.06492].

[34] Wolfram Research, Inc., Mathematica, Version 11.0, Champaign, IL (2016).

[35] J. Aebischer, M. Fael, C. Greub and J. Virto, *B physics Beyond the Standard Model at One Loop: Complete Renormalization Group Evolution below the Electroweak Scale*, *JHEP* **09** (2017) 158, [1704.06639].

[36] A. Bednyakov, A. Pikelner and V. Velizhanin, *Anomalous dimensions of gauge fields and gauge coupling beta-functions in the Standard Model at three loops*, *JHEP* **01** (2013) 017, [1210.6873].

[37] A. Bednyakov, A. Pikelner and V. Velizhanin, *Yukawa coupling beta-functions in the Standard Model at three loops*, *Phys. Lett. B* **722** (2013) 336–340, [1212.6829].

[38] A. Bednyakov, A. Pikelner and V. Velizhanin, *Higgs self-coupling beta-function in the Standard Model at three loops*, *Nucl. Phys. B* **875** (2013) 552–565, [1303.4364].

[39] A. Bednyakov, A. Pikelner and V. Velizhanin, *Three-loop SM beta-functions for matrix Yukawa couplings*, *Phys. Lett. B* **737** (2014) 129–134, [1406.7171].

[40] T. van Ritbergen, J. Vermaseren and S. Larin, *The Four loop beta function in quantum chromodynamics*, *Phys. Lett. B* **400** (1997) 379–384, [hep-ph/9701390].

[41] J. Vermaseren, S. Larin and T. van Ritbergen, *The four loop quark mass anomalous dimension and the invariant quark mass*, *Phys. Lett. B* **405** (1997) 327–333, [hep-ph/9703284].

[42] P. Baikov, K. Chetyrkin and J. Kühn, *Five-loop fermion anomalous dimension for a general gauge group from four-loop massless propagators*, *JHEP* **04** (2017) 119, [1702.01458].

[43] M. E. Machacek and M. T. Vaughn, *Two Loop Renormalization Group Equations in a General Quantum Field Theory. 1. Wave Function Renormalization*, *Nucl. Phys. B* **222** (1983) 83–103.

[44] M. E. Machacek and M. T. Vaughn, *Two Loop Renormalization Group Equations in a General Quantum Field Theory. 2. Yukawa Couplings*, *Nucl. Phys. B* **236** (1984) 221–232.

[45] M. E. Machacek and M. T. Vaughn, *Two Loop Renormalization Group Equations in a General Quantum Field Theory. 3. Scalar Quartic Couplings*, *Nucl. Phys. B* **249** (1985) 70–92.

[46] M.-x. Luo and Y. Xiao, *Two loop renormalization group equations in the standard model*, *Phys. Rev. Lett.* **90** (2003) 011601, [hep-ph/0207271].

[47] S. Antusch, M. Drees, J. Kersten, M. Lindner and M. Ratz, *Neutrino mass operator renormalization revisited*, *Phys. Lett. B* **519** (2001) 238–242, [hep-ph/0108005].

[48] K. Chetyrkin, J. H. Kuhn and M. Steinhauser, *RunDec: A Mathematica package for running and decoupling of the strong coupling and quark masses*, *Comput. Phys. Commun.* **133** (2000) 43–65, [hep-ph/0004189].

[49] J. Aebischer et al., *WCxf: an exchange format for Wilson coefficients beyond the Standard Model*, *Comput. Phys. Commun.* **232** (2018) 71–83, [1712.05298].

[50] DsixTools website: https://dsixtools.github.io.

[51] P. Z. Skands et al., *SUSY Les Houches accord: Interfacing SUSY spectrum calculators, decay packages, and event generators*, *JHEP* **07** (2004) 036, [hep-ph/0311123].

[52] B. Allanach et al., *SUSY Les Houches Accord 2*, *Comput. Phys. Commun.* **180** (2009) 8–25, [0801.0045].

[53] Z. Bjornson, “MYaml.” https://github.com/zbjornson/MYaml.

[54] S. Weinberg, *Baryon and Lepton Nonconserving Processes*, *Phys. Rev. Lett.* **43** (1979) 1566–1570.

[55] L. Abbott and M. B. Wise, *The Effective Hamiltonian for Nucleon Decay*, *Phys. Rev. D* **22** (1980) 2208.