Eur. Phys. J. C (2018) 78:1026
https://doi.org/10.1140/epjc/s10052-018-6492-7

THE EUROPEAN PHYSICAL JOURNAL C

Regular Article - Theoretical Physics

wilson: a Python package for the running and matching of Wilson coefficients above and below the electroweak scale

Jason Aebischer[1,a], **Jacky Kumar**[2,b], **David M. Straub**[1,c]

[1] Excellence Cluster Universe, TUM, Boltzmannstr. 2, 85748 Garching, Germany
[2] Department of Physics, Indian Institute of Science Education and Research, Mohali, Punjab 140036, India

Received: 20 August 2018 / Accepted: 2 December 2018

Abstract *wilson* is a Python library for matching and running Wilson coefficients of higher-dimensional operators beyond the Standard Model. Provided with the numerical values of the Wilson coefficients at a high new physics scale, it automatically performs the renormalization group evolution within the Standard Model effective field theory (SMEFT), matching onto the weak effective theory (WET) at the electroweak scale, and QCD/QED renormalization group evolution below the electroweak scale down to hadronic scales relevant for low-energy precision tests. The matching and running encompasses the complete set of dimension-six operators in both SMEFT and WET. The program builds on the Wilson coefficient exchange format (WCxf) and can thus be easily combined with a number of existing public codes.

1 Introduction

The Standard Model (SM) [1–3] is considered to be an effective theory valid only up to a new physics scale Λ, which negative searches for new particles at the LHC likely relegate to well above the electroweak (EW) scale. If no light degrees of freedom beyond the SM are assumed, any new physics effect in processes proceeding at energies well below Λ can be described by local interactions among SM fields invariant under the SM gauge symmetry [4,5]. This effective field theory (EFT) approach [6,7] to new physics not only allows to resum large logarithms that might invalidate calculations in perturbation theory for vastly different scales relevant in a given process, but also serves as a convenient intermediate step between "model building" in the UV and low-energy phenomenology. If new physics predictions for experimental observables are expressed in terms of Wilson coefficients of an EFT beyond the SM, the investigation of the low-energy implications of a concrete new physics model becomes much simpler since only the Wilson coefficients need to be calculated at the appropriate scale.

While the EFT approach to new physics has been ubiquitous in quark flavour physics – dealing with processes at energies of few GeV – for a long time already, the experimental indications that Λ lies well above the electroweak scale have led to the realization that this approach is also valuable for processes of electroweak scale energies like Higgs physics or electroweak precision tests (see [8] and references therein). In contrast to the EFT *below* the electroweak scale, that is conventionally called the weak effective theory (WET) [9–11] and only contains QED and QCD gauge interactions, the EFT *above* the electroweak scale, conventionally called SMEFT[1] [13–15], contains $SU(2)_L$ interactions that do not conserve flavour. Consequently, quantum effects lead to an interesting interplay between processes with and without flavour change and call for a *global* approach.

Starting from the new physics scale Λ, the phenomenological analysis of a UV model typically requires the following technical steps.[2]

1. Compute the SMEFT Wilson coefficients at Λ.
2. Perform the renormalization group (RG) evolution of the SMEFT Wilson coefficients down to the electroweak scale.
3. Match the complete set of SMEFT Wilson coefficients onto the WET.
4. Perform the RG evolution of WET Wilson coefficients.

[1] Throughout, we work with the EFT above the electroweak scale with *linearly* realized electroweak symmetry breaking (see [12] and references therein for a discussion of the non-linear case).

[2] Steps 3.–5. can be omitted for observables at electroweak scale energies.

[a] e-mail: jason.aebischer@tum.de
[b] e-mail: jkumar@iisermohali.ac.in
[c] e-mail: david.straub@tum.de

Published online: 19 December 2018

5. If the process proceeds at energies below the b quark mass, repeat the last two steps for the WET with reduced numbers of quark and lepton flavours as appropriate.
6. Compute the process of interest as a function of the low-energy Wilson coefficients.

While the first five steps are straightforward in principle, the full procedure is technically challenging in practice due to the vast number of Wilson coefficients already at dimension six (cf. [9,16]). The *wilson* package provides an automated solution to steps 2.–5. above. Given the SMEFT Wilson coefficients at the UV scale Λ, it bridges the gap to the low-energy phenomenology in step 6., which is implemented in other public codes such as `flavio` [17]. The package makes use of the following results in the literature.

– The complete basis of SMEFT operators first derived in [4] and for a non-redundant set of operators in [5].
– The complete one-loop RG evolution in SMEFT [16,18,19].
– Analytical solutions to the one-loop RG evolution of all flavour violating operators in WET [10].
– The complete RG evolution of WET operators [11].
– The complete tree-level matching of SMEFT onto the WET [9,20].
– The definition of a Wilson coefficient exchange format (WCxf) that allows to define EFTs, bases of Wilson coefficients, and facilitates exchanging numerical values of Wilson coefficients between different codes [21].

It benefits from the following public physics codes:

– The SMEFT RG evolution was ported from (and is tested against) the DsixTools Mathematica package [22].
– The QCD evolution of quark masses and the strong coupling constant is computed with the `python-rundec` package that wraps the `CRunDec` module [23].
– The SM $\overline{\text{MS}}$ parameters at the electroweak scale have been obtained with the `mr` package [24].

The rest of this note is organized as follows. In Sect. 2, we give some details on the implementation of running and matching in *wilson*. Section 3 describes how to install the package. Section 4 contains details on how to use the code. In Sect. 5, we present a simple example application, reproducing a well-known result from the literature.

2 Description

The *wilson* package consists of several submodules taking care of the RG evolution, basis translation, and matching. A typical internal workflow is shown in Fig. 1, where a set of SMEFT Wilson coefficients in the “Warsaw up” basis [9] at the scale Λ is the input and the WET Wilson coefficients at the scale μ_{low} are returned in the basis used by the `flavio` package. Internally, the Warsaw basis as defined in WCxf [21] is used for the SMEFT running, and the JMS basis [9] for the matching and WET running. From a user's perspective, the entire procedure is performed automatically when using the `match_run` method described in Sect. 4.2, as indicated by the dashed arrow. Below, we discuss some implementation details of the individual submodules.

2.1 Extraction of standard model parameters in SMEFT

Starting from a set of Wilson coefficients at the UV scale, given e.g. in WCxf format, to solve the SMEFT RGEs one additionally requires the values of SM parameters like gauge couplings, Yukawa couplings, and Higgs potential parameters. This is challenging for two reasons. First, these parameters are experimentally determined at the electroweak scale or below, and their evolution to the UV scale depends on the SMEFT Wilson coefficients themselves. Second, the experimental extraction itself is subject to dimension six corrections already at tree level. To solve these two problems, we proceed in three steps.

1. We determine all the SM parameters in the $\overline{\text{MS}}$ scheme [25] at the scale M_Z.
2. We invert the relations between the effective $\overline{\text{MS}}$ SM parameters and their counterparts in SMEFT, that are given e.g. in [26].
3. We iteratively determine the SM parameters at the UV scale by running up and down with the SM boundary conditions imposed at the scale M_Z and the Wilson coefficient boundary conditions at the UV scale Λ.

Concerning the first step, the SM $\overline{\text{MS}}$ parameters used by us are listed in Table 1. The following comments are in order.

– For the running of the quark masses to the scale M_Z, we have used the `python-rundec` package [23].
– For the determination of the running top, W, Z, and Higgs masses, we have used the `mr` package [24].
– For the lepton masses, we have neglected the $O(\alpha_e)$ shift from the conversion to the $\overline{\text{MS}}$ scheme.
– We do not display uncertainties as fixed values are used in the code. We expect the parametric errors to be subdominant to other uncertainties in the calculation, e.g. from the iterative determination of high-scale SM parameters.[3]

[3] To check the accuracy of the iterative determination of SM parameters, the class `wilson.run.smeft.SMEFT`, that is initialized by a `wcxf.WC` instance, provides a method `get_smpar`, that computes

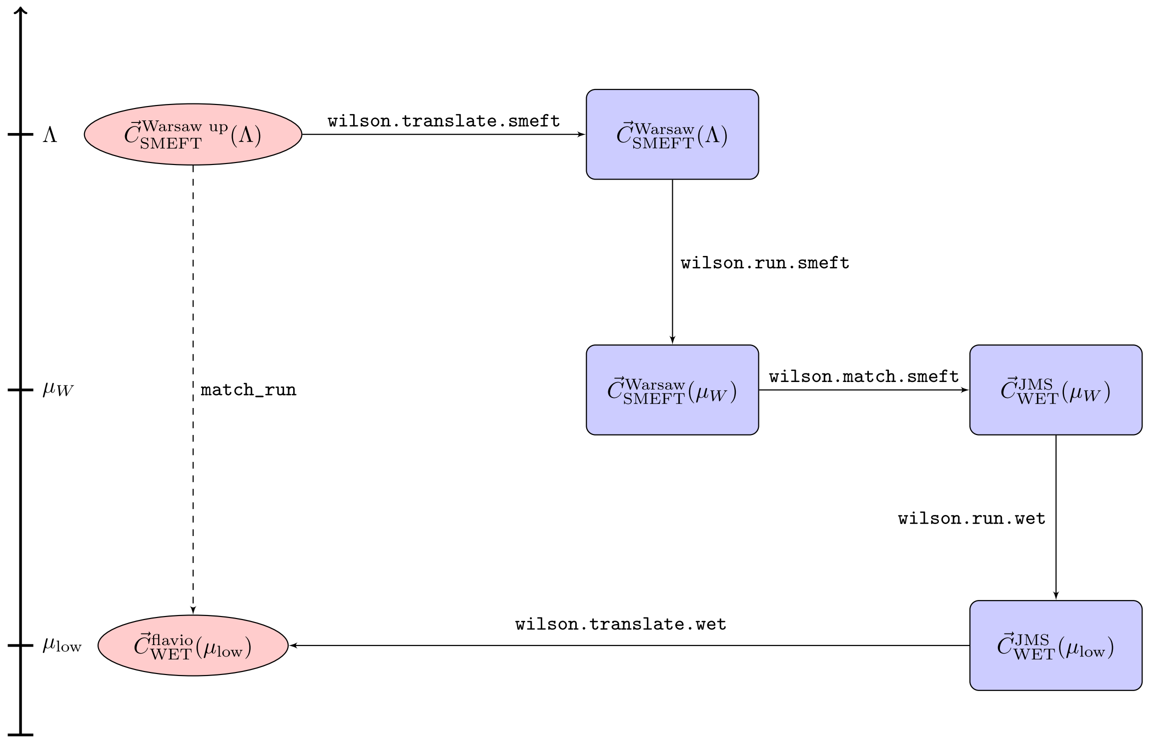
Λ
μ_W
$\mu_{\rm low}$
$\vec{C}^{\rm Warsaw\ up}_{\rm SMEFT}(\Lambda)$
wilson.translate.smeft
$\vec{C}^{\rm Warsaw}_{\rm SMEFT}(\Lambda)$
wilson.run.smeft
match_run
$\vec{C}^{\rm Warsaw}_{\rm SMEFT}(\mu_W)$
wilson.match.smeft
$\vec{C}^{\rm JMS}_{\rm WET}(\mu_W)$
wilson.run.wet
wilson.translate.wet
$\vec{C}^{\rm flavio}_{\rm WET}(\mu_{\rm low})$
$\vec{C}^{\rm JMS}_{\rm WET}(\mu_{\rm low})$

Fig. 1 Typical internal workflow in the *wilson* package: Starting from the SMEFT Wilson coefficients at the scale Λ, various submodules take care of the necessary basis translations, RG running, and matching to finally obtain the WET Wilson coefficients at the low scale. From a user perspective, the `match_run` method (see Sect. 4.2) performs all these steps automatically

Table 1 SM $\overline{\rm MS}$ parameters at the scale M_Z. Masses are given in units of GeV

Par.	Value	Par.	Value
α_e	1/127.9	m_u	0.00127
α_s	0.1185	m_d	0.00270
V_{us}	0.2243	m_s	0.0551
V_{cb}	0.04221	m_c	0.635
V_{ub}	0.00362	m_b	2.85
γ	1.27	m_t	169.0
m_e	0.000511	m_W	80.20
m_μ	0.1057	m_Z	91.46
m_τ	1.777	m_h	130.6

We note that we treat the CKM elements as elements of a unitary 3×3 matrix. Dimension-six contributions to the W coupling to quarks are thus *not* absorbed in effective CKM elements, as done e.g. in [26]. We find this procedure more convenient for our purposes; in particular, it allows to continue to use unitarity relations in low-energy calculations in flavour physics. While this blurs the connection between these CKM elements and the semi-leptonic decays that are used to measure them, we note that this connection is anyway blurred in SMEFT due to direct dimension-six four-fermion contributions to these decays that can lead to a process-dependent shift of the apparent CKM element (see e.g. [27] for a discussion of $s \to u$ transitions and [28] for $b \to c$ transitions).

Footnote 3 continued
the predicted values for the SM $\overline{\rm MS}$ parameters at the electroweak scale, which should correspond to the values in Table 1.

2.2 RG evolution in SMEFT

Once the SM parameters at the input scale have been determined, the SMEFT RGEs, that have the form

$$\frac{dC_i}{d\ln\mu} = \frac{1}{16\pi^2}\sum_j \gamma_{ji} C_i\,, \tag{1}$$

can be solved numerically by integrating the right-hand side. Our implementation closely follows the DsixTools package [22].

As an important caveat, we caution the reader that the numerical inputs and outputs, using the non-redundant basis defined by the WCxf convention, differ from the conventions used in [16,18,19], where a redundant basis of flavour indices is employed, by symmetry factors in some cases. We refer to appendix A of [29], where this issue is discussed in detail.

2.3 Matching from SMEFT to WET

We implement the complete tree-level matching from SMEFT to WET as derived in [9]. It includes the full set of non-redundant gauge-invariant dimension six operators in both theories. The matching is performed at the EW scale.

2.4 RG evolution in WET

In the weak effective theory, the dimension-6 operators are renormalized by QCD and QED. Analytical solutions to the one-loop RGEs of all quark flavour violating operators have been presented in [10].[4] To extend this to the *complete* operator basis[5] of WET, we proceed in three steps.

1. We take the beta functions from [9], discarding terms that are quadratic in dipole operator coefficients (these terms correspond to dimension eight contributions when matching from the SMEFT with linearly realized electroweak symmetry breaking).
2. We rescale dipole operators and three-gluon operators in the following way:

$$\bar{f}_L^i \sigma^{\mu\nu} f_R^j F_{\mu\nu} \to \frac{e}{g_s^2} m_f \bar{f}_L^i \sigma^{\mu\nu} f_R^j F_{\mu\nu}\,, \tag{2}$$

$$\bar{f}_L^i \sigma^{\mu\nu} T^A f_R^j G^A_{\mu\nu} \to \frac{1}{g_s} m_f \bar{f}_L^i \sigma^{\mu\nu} T^A f_R^j G^A_{\mu\nu}\,, \tag{3}$$

$$G_\mu^{A\nu} G_\nu^{B\rho} G_\rho^{C\mu} \to \frac{1}{g_s} G_\mu^{A\nu} G_\nu^{B\rho} G_\rho^{C\mu}\,, \tag{4}$$

   where $m_f = \max(m_{f_i}, m_{f_j})$. This allows us to write the RGEs in the simple form

$$\frac{dC_i}{d\ln\mu} = \frac{g_s^2}{16\pi^2} \sum_j \gamma^s_{ji} C_i + \frac{e^2}{16\pi^2} \sum_j \gamma^e_{ji} C_i\,. \tag{5}$$

   Note in particular that there are no linear or mixed terms in g_s or e. Thanks to the rescalings, the anomalous dimension matrices $\gamma^{s,e}$ only contain numbers and ratios of fermion masses, which are RG invariant to $O(\alpha_s)$ and thus can be treated as constants to good approximation.
3. Having rewritten the RGEs in the simple form (5), we can use the procedure described in [10] to obtain the QCD and QED evolution matrices that solve the RGE as

$$C_i(\mu) = \left[U_s(\mu,\mu_0)_{ij} + \Delta U_e(\mu,\mu_0)_{ij}\right] C_j(\mu_0)\,. \tag{6}$$

[4] See also [30,31] and references therein.

[5] In the WET RG evolution, we restrict ourselves to baryon and lepton number conserving operators for the time being.

3 Installation

Installing *wilson* only requires a system with Python version 3.5 or above. It works on Linux, Mac OS, and Windows. The most recent version can be installed directly from the Python package index by issuing the command[6]

```
python3 -m pip install wilson --user
```

in the terminal, without root privileges. This will automatically install the `wcxf` package and command line interface as well, if not already available on the system. When a new version is available, the package can be upgraded with

```
python3 -m pip install --upgrade wilson --user
```

4 Usage

4.1 Initializiation

Using the *wilson* package in a Python script or interactive session starts by creating a `Wilson` object that represents a point in EFT parameter space. On creating the instance, initial values of the Wilson coefficients have to be specified at some scale, e.g. the new physics scale Λ, in a given EFT and basis. For example, the commands

```
from wilson import Wilson
mywilson = Wilson({'uG_33': 1e-6},
                  scale=1e3, eft='SMEFT', basis='Warsaw')
```

create a new `Wilson` instance where the Wilson coefficient of the chromomagnetic operator with two top quarks in the SMEFT Warsaw basis,

$$O_{uG}^{33} = \left(\bar{q}_3 \sigma^{\mu\nu} T^A u_3\right) \widetilde{\varphi} G^A_{\mu\nu}, \tag{7}$$

is set to the value $1/\text{TeV}^2$ at the scale 1 TeV (note that all dimensionful quantities have to be specified in appropriate powers of GeV, as required by WCxf). At this point, it is important to emphasize the difference between *wilson*'s `Wilson` class and the `WC` class provided by the `wcxf` Python package:

– `wcxf.WC` represents a set of numerical Wilson coefficients at a *fixed* scale in a *fixed* EFT and basis;
– `wilson.Wilson` represents a point in the parameter space of the EFT beyond the SM, that can be evolved to different scales and translated to different bases within the same EFT without loss of generality, or matched to EFTs valid at lower energies.

[6] The name of the Python 3 executable might differ depending on the system.

In fact, after initializing the above object, the Wilson coefficient values at the initial scale can be returned as a `WC` object simply with `mywilson.wc`. Likewise, a `Wilson` object can be easily initialized by loading Wilson coefficient values from a file in WCxf format:

```
from wilson import Wilson
with open('my_wcxf.json') as f:
    mywilson = Wilson.load_wc(f)
```

4.2 Matching and running

Running, i.e. performing the RG evolution in SMEFT and WET, as well as matching from SMEFT to WET (and from WET with five active quark flavours to the variants of WET valid below the bottom and charm mass scales) is the main purpose of the *wilson* package. Having initialized a `Wilson` object as described in Sect. 4.1 – we will continue to call this instance `mywilson` – the user can obtain Wilson coefficient values (in the form of `wcxf.WC` instances) in different EFTs, at different scales, in different bases, through the method `match_run`:[7]

```
# running up to 100 TeV and translating to the 'Warsaw up' basis
wc = mywilson.match_run(scale=1e5, eft='SMEFT', basis='Warsaw up')
# running and matching to WET at 100 GeV in the 'JMS' basis
wc = mywilson.match_run(scale=100, eft='WET', basis='JMS')
# running and matching to WET-3 at 2 GeV in the 'flavio' basis
wc = mywilson.match_run(scale=2, eft='WET-3', basis='flavio')
```

The names of admissible EFTs and bases can be found on the WCxf website [32].

We note that the output scale can also be higher than the input scale, but only if the output EFT is the same as the input EFT. In this case, the RG evolution (in WET or SMEFT) will be performed from the low input scale to the high output scale. Since the matching is not bijective, this cannot be done across EFT thresholds.

The default behaviour of the `Wilson` class can be modified with a few user options that can be modified either on a single instance or globally for all future instances of the class (e.g. when importing the package),

```
mywilson.set_option(OPTION, VALUE)         # set option on instance
Wilson.set_default_option(OPTION, VALUE)   # set option globally
```

The following options are implemented as of version 1.4,

- `'smeft_accuracy'` – set accuracy of the SMEFT RG evolution to numerical integration (value `'integrate'`, default) or leading-logarithmic approximation (`'leadinglog'`), which is less accurate but much faster.
- `'qcd_order'`, `'qed_order'` – set the order of QED and QCD anomalous dimensions to be taken into account in the WET RG running. Currently both values are restricted to 1 (default, leading order) or 0 (off).
- `'smeft_matchingscale'` – set the scale (in GeV) where SMEFT is matched onto WET. Defaults to 91.1876 (the central value of the Z^0 mass).
- `'mb_matchingscale'`, `'mc_matchingscale'` – set the scales (in GeV) where WET is matched onto WET-4 and WET-4 onto WET-3. Default to 4.2 and 1.3, respectively.

[7] Tiny non-zero entries of Wilson coefficients can be traced back to a finite numerical precision used for the solution of the RGEs and can safely be neglected.

4.3 Interfacing with other codes

Since *wilson* builds on the Wilson coefficient exchange format WCxf, it is straightforward to import and export from and to programs supporting this standard. While the import has already been discussed above, the export can simply leverage the methods provided by the `wcxf` Python package, e.g.

```
wc = mywilson.match_run(scale=100, eft='WET', basis='JMS')
with open('my_wcxf_output.json', 'w') as f:
    wc.dump(f)
```

An even simpler data exchange is possible for codes written in Python themselves. In particular, the `flavio` package [17], that can compute predictions for a plethora of observables in quark and lepton flavour physics, directly makes use of the *wilson* package for the RG evolution, matching, and translation, starting from version v0.28. Functions that accept new physics Wilson coefficient values can be directly provided with a `Wilson` instance. This also allows to compute observables in terms of SMEFT Wilson coefficients. For example,

```
from wilson import Wilson
import flavio
mywilson = Wilson({'lq3_3333': 1e-6},
                  scale=1e3, eft='SMEFT', basis='Warsaw')
flavio.np_prediction('Rtaul(B->D*lnu)', mywilson)
```

computes the observable R_{D^*} given a value of $1/\text{TeV}^2$ for the Wilson coefficient of the SMEFT operator

$$\left[O_{lq}^{(3)}\right]_{3333} = \left(\bar\ell_3\gamma_\mu\tau^I\ell_3\right)\left(\bar q_3\gamma^\mu\tau^I q_3\right), \tag{8}$$

at the scale 1 TeV. The SMEFT running, matching, WET running, and conversion to the flavio basis used in the calculation of the observable is done behind the curtains by *wilson*.

5 Example

An interesting example where SMEFT RG effects lead to important constraints on NP scenarios was discussed in Refs. [33–35]. It investigates scenarios attempting to simultaneously explain the deviations from lepton flavour universality observed in $b \to s\ell^+\ell^-$ transitions (with $\ell = e$ vs. μ) and $b \to c\tau\nu$ transitions (with $\ell = \tau$ vs. e or μ) [36–41],

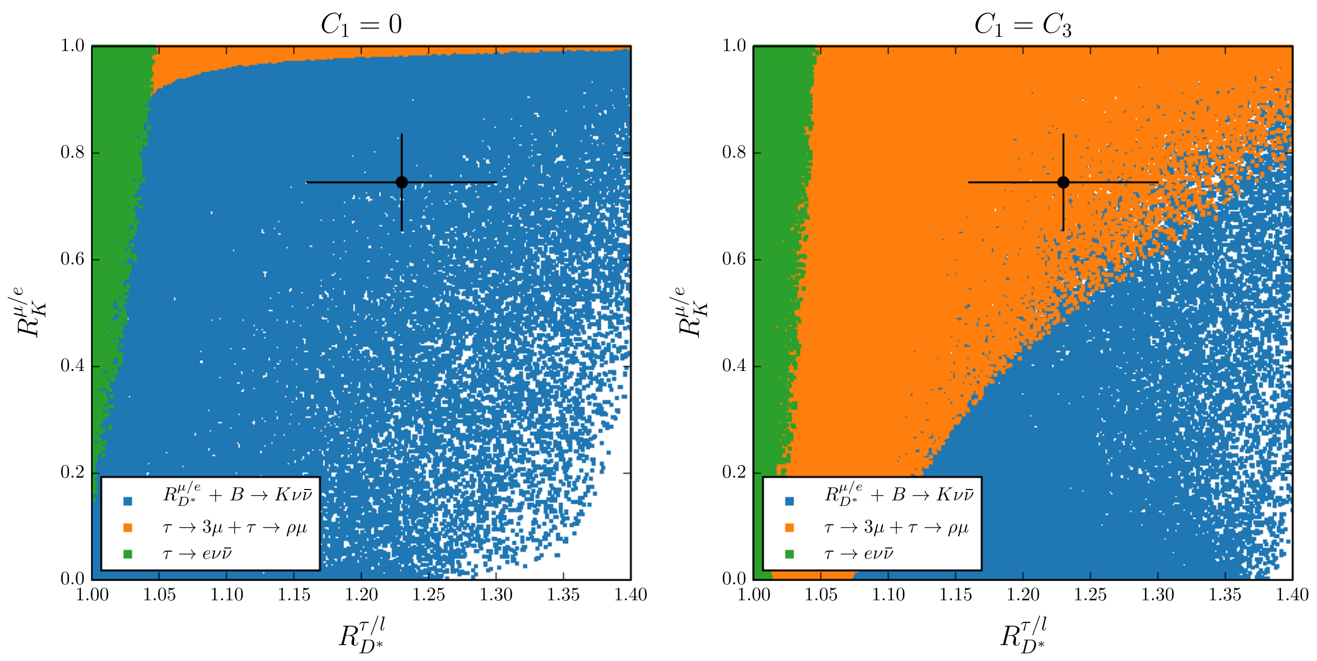

Fig. 2 Constraints on simultaneous solutions to B anomalies through left-handed currents dominantly coupling to the third generation, reproducing fig. 5 of [34]

$$R_{K^{(*)}}^{\mu/e} = \frac{\mathcal{B}(B \to K^{(*)}\mu^+\mu^-)}{\mathcal{B}(B \to K^{(*)}e^+e^-)},$$
$$R_{D^{(*)}}^{\tau/\ell} = \frac{\mathcal{B}(B \to D^{(*)}\tau\bar{\nu})_{exp}/\mathcal{B}(B \to D^{(*)}\tau\bar{\nu})_{SM}}{\mathcal{B}(B \to D^{(*)}\ell\bar{\nu})_{exp}/\mathcal{B}(B \to D^{(*)}\ell\bar{\nu})_{SM}}. \tag{9}$$

Since NP effects in semi-leptonic four-fermion operators with all left-handed fields are well known to fit the low-energy flavour data [42–46], it is interesting to consider NP models coupling dominantly to the third generation of left-handed quarks and leptons, such that the $b \to c\tau\nu$ transition, generated at tree level in the SM, receives sizable NP contributions, while the $b \to s\mu\mu$ transition is suppressed by flavour mixing angles that are assumed to be small [47,48]. It was then shown that strong constraints arise on the simultaneous explanation of charged and neutral current anomalies from lepton flavour non-universality induced in leptonic tau decays, from lepton flavour violating tau decays, and from Z pole observables.

The scenario considered in [34] corresponds to the presence of the operators $[O_{lq}^{(1,3)}]_{3333}$ at a scale Λ in some weak basis that is related to the mass basis by small mixing angles. Choosing a definite weak basis, namely the one conventionally used for the Warsaw basis in WCxf [21], where the down-type quark and charged lepton masses are diagonal, the following Wilson coefficients are present at the scale Λ:

$$\left[C_{lq}^{(1)}\right]_{ijkl} = \lambda_{ij}^{\ell}\lambda_{kl}^{q}\,C_1\,, \qquad \left[C_{lq}^{(3)}\right]_{ijkl} = \lambda_{ij}^{\ell}\lambda_{kl}^{q}\,C_3\,. \tag{10}$$

Assuming without loss of generality $\lambda_{33}^q = \lambda_{33}^\ell = 1$ and adopting the simplified scenario where $\lambda_{22}^{q,\ell} = (\lambda_{23}^{q,\ell})^2$, a scenario can be initialized in *wilson* as a function of the parameters `C1` $= C_1$, `C3` $= C_3$, `lq_23` $= \lambda_{23}^q$, `ll_23` $= \lambda_{23}^\ell$ and `Lambda` $= \Lambda$ as

```python
from wilson import Wilson
ll_33 = ...
...
w = Wilson({'lq3_3333': ll_33 * lq_33 * C3,
            'lq1_3333': ll_33 * lq_33 * C1,
            'lq3_2223': ll_22 * lq_23 * C3, ...},
           scale=Lambda, eft='SMEFT', basis='Warsaw')
```

where the parameter variables have been set to numerical values.[8] This `Wilson` instance can now be used to compute predictions for the relevant constraints using `flavio`, as discussed in Sect. 4.3:

```python
from flavio import np_prediction
np_prediction('<Rmue>(B+->Kll)', w, 1, 6)  # R_K from 1-6 GeV^2
np_prediction('Rtaul(B->D*lnu)', w)        # R_D*(tau/l)
np_prediction('Rmue(B->D*lnu)',w)          # R_D*(mu/e)
np_prediction('BR(B+->Knunu)', w)          # B -> K nu nu
np_prediction('BR(tau->mumumu)', w)        # tau -> 3 mu
np_prediction('BR(tau->rhomu)', w)         # tau -> rho mu
np_prediction('BR(tau->enunu)', w)         # tau -> e nu nu
```

Using this procedure, in Fig. 2 we have reproduced the result of Refs. [33,34], where the four free parameters are scanned as: $\lambda_{23}^q \in [-0.05, 0]$, $\lambda_{23}^\ell \in [-0.5, 0.5]$, $C_{1,3} \in [-4, 0]$, and the scale Λ is set at 1 TeV. This shows that a simultaneous explanation of the charged and neutral current anomalies is disfavoured in this simplified scenario.

[8] Note: *wilson* does not accept symbolic inputs.

6 Summary

We have presented *wilson*, a Python package for the RG evolution, matching, and basis translation of Wilson coefficients beyond the SM. Starting from numerical values of Wilson coefficients at a high scale Λ, it automatically performs the necessary steps to return the Wilson coefficients at low energies relevant for precision measurements probing physics beyond the SM. Built on the Wilson coefficient exchange format (WCxf), *wilson* can be easily linked with a number of public codes, e.g. to directly compute the predictions for low-energy observables, as demonstrated in Sect. 5.

While *wilson* is currently limited to one-loop RG evolution in SMEFT and WET and to tree-level matching, the structure of the code is general enough to be generalized to higher loop orders in the running and to loop-level matching (which is partially known, see e.g. [20]) in the future. It has already been used in several NP analyses in the context of B anomalies [49] and $\varepsilon/'\varepsilon$ [50–52]. Being an open source project with a permissive license,[9] contributions from the community are welcome via the public code repository [53]. Further information to *wilson* can be found on the *wilson* web page https://wilson-eft.github.io/.

Acknowledgements The work of D. S. and J. A. is supported by the DFG cluster of excellence "Origin and Structure of the Universe". We thank Xuanyou Pan and Matthias Schöffel for important technical support in the development phase. J. K. thanks Michael Paraskevas for discussions.

References

1. S. Weinberg, Phys. Rev. Lett. **19**, 1264 (1967). https://doi.org/10.1103/PhysRevLett.19.1264
2. S.L. Glashow, Nucl. Phys. **22**, 579 (1961). https://doi.org/10.1016/0029-5582(61)90469-2
3. A. Salam, Conf. Proc. C **680519**, 367 (1968)
4. W. Buchmuller, D. Wyler, Nucl. Phys. B **268**, 621 (1986). https://doi.org/10.1016/0550-3213(86)90262-2
5. B. Grzadkowski, M. Iskrzynski, M. Misiak, J. Rosiek, JHEP **10**, 085 (2010). https://doi.org/10.1007/JHEP10(2010)085
6. T. Appelquist, J. Carazzone, Phys. Rev. D **11**, 2856 (1975). https://doi.org/10.1103/PhysRevD.11.2856
7. K.G. Wilson, Rev. Mod. Phys. **55**, 583 (1983). https://doi.org/10.1103/RevModPhys.55.583
8. J. Ellis, C.W. Murphy, V. Sanz, T. You, **2018**, 146 (2018). https://doi.org/10.1007/JHEP06(2018)146. arXiv:1803.03252
9. E.E. Jenkins, A.V. Manohar, P. Stoffer, JHEP **03**, 016 (2018). https://doi.org/10.1007/JHEP03(2018)016
10. J. Aebischer, M. Fael, C. Greub, J. Virto, JHEP **09**, 158 (2017). https://doi.org/10.1007/JHEP09(2017)158
11. E.E. Jenkins, A.V. Manohar, P. Stoffer, JHEP **01**, 084 (2018). https://doi.org/10.1007/JHEP01(2018)084
12. D. de Florian, et al., (2016). https://doi.org/10.23731/CYRM-2017-002
13. S. Weinberg, Phys. Lett. B **91**, 51 (1980). https://doi.org/10.1016/0370-2693(80)90660-7
14. S.R. Coleman, J. Wess, B. Zumino, Phys. Rev. **177**, 2239 (1969). https://doi.org/10.1103/PhysRev.177.2239
15. C.G. Callan Jr., S.R. Coleman, J. Wess, B. Zumino, Phys. Rev. **177**, 2247 (1969). https://doi.org/10.1103/PhysRev.177.2247
16. R. Alonso, E.E. Jenkins, A.V. Manohar, M. Trott, JHEP **04**, 159 (2014). https://doi.org/10.1007/JHEP04(2014)159
17. D.M. Straub, flavio: a Python package for flavour and precision phenomenology in the Standard Model and beyond (2018). arXiv:1810.08132
18. E.E. Jenkins, A.V. Manohar, M. Trott, JHEP **10**, 087 (2013). https://doi.org/10.1007/JHEP10(2013)087
19. E.E. Jenkins, A.V. Manohar, M. Trott, JHEP **01**, 035 (2014). https://doi.org/10.1007/JHEP01(2014)035
20. J. Aebischer, A. Crivellin, M. Fael, C. Greub, JHEP **05**, 037 (2016). https://doi.org/10.1007/JHEP05(2016)037
21. J. Aebischer, Comput. Phys. Commun. **232**, 71 (2018). https://doi.org/10.1016/j.cpc.2018.05.022
22. A. Celis, J. Fuentes-Martin, A. Vicente, J. Virto, Eur. Phys. J. C **77**(6), 405 (2017). https://doi.org/10.1140/epjc/s10052-017-4967-6
23. F. Herren, M. Steinhauser, Comput. Phys. Commun. **224**, 333 (2018). https://doi.org/10.1016/j.cpc.2017.11.014
24. B.A. Kniehl, A.F. Pikelner, O.L. Veretin, Comput. Phys. Commun. **206**, 84 (2016). https://doi.org/10.1016/j.cpc.2016.04.017
25. W.A. Bardeen, A.J. Buras, D.W. Duke, T. Muta, Phys. Rev. D **18**, 3998 (1978). https://doi.org/10.1103/PhysRevD.18.3998
26. A. Dedes, W. Materkowska, M. Paraskevas, J. Rosiek, K. Suxho, JHEP **06**, 143 (2017). https://doi.org/10.1007/JHEP06(2017)143
27. M. González-Alonso, J.M. Camalich, JHEP **12**, 052 (2016). https://doi.org/10.1007/JHEP12(2016)052
28. M. Jung, D.M. Straub (2018). arXiv:1801.01112
29. J. Aebischer, J. Kumar, P. Stangl, D.M. Straub (2018). arXiv:1810.07698
30. G. Buchalla, A.J. Buras, M.E. Lautenbacher, Rev. Mod. Phys. **68**, 1125 (1996). https://doi.org/10.1103/RevModPhys.68.1125
31. A.J. Buras (2011). arXiv:1102.5650
32. WCxf web site. https://wcxf.github.io
33. F. Feruglio, P. Paradisi, A. Pattori, Phys. Rev. Lett. **118**(1), 011801 (2017). https://doi.org/10.1103/PhysRevLett.118.011801
34. F. Feruglio, P. Paradisi, A. Pattori, JHEP **09**, 061 (2017). https://doi.org/10.1007/JHEP09(2017)061
35. C. Cornella, F. Feruglio, P. Paradisi, Low-energy effects of Lepton Flavour Universality Violation. JHEP **11**, 012 (2018). https://doi.org/10.1007/JHEP11(2018)012. arXiv:1803.00945
36. R. Aaij, Phys. Rev. Lett. **113**, 151601 (2014). https://doi.org/10.1103/PhysRevLett.113.151601
37. R. Aaij, JHEP **08**, 055 (2017). https://doi.org/10.1007/JHEP08(2017)055
38. J.P. Lees, Phys. Rev. D **88**(7), 072012 (2013). https://doi.org/10.1103/PhysRevD.88.072012
39. M. Huschle, Phys. Rev. D **92**(7), 072014 (2015). https://doi.org/10.1103/PhysRevD.92.072014
40. R. Aaij et al., Phys. Rev. Lett. **115**(11), 111803 (2015). https://doi.org/10.1103/PhysRevLett.115.159901. https://doi.org/10.1103/PhysRevLett.115.111803. [Erratum: Phys. Rev. Lett.115, no.15,159901(2015)]
41. A. Abdesselam, et al. (2016). arXiv:1608.06391

[9] *wilson* is released under the MIT license.

42. W. Altmannshofer, P. Stangl, D.M. Straub, Phys. Rev. D **96**(5), 055008 (2017). https://doi.org/10.1103/PhysRevD.96.055008
43. B. Capdevila, A. Crivellin, S. Descotes-Genon, J. Matias, J. Virto, JHEP **01**, 093 (2018). https://doi.org/10.1007/JHEP01(2018)093
44. A.K. Alok, B. Bhattacharya, A. Datta, D. Kumar, J. Kumar, D. London, Phys. Rev. D **96**(9), 095009 (2017). https://doi.org/10.1103/PhysRevD.96.095009
45. L.S. Geng, B. Grinstein, S. Jäger, J.M. Camalich, X.L. Ren, R.X. Shi, Phys. Rev. D **96**(9), 093006 (2017). https://doi.org/10.1103/PhysRevD.96.093006
46. M. Ciuchini, A.M. Coutinho, M. Fedele, E. Franco, A. Paul, L. Silvestrini, M. Valli, Eur. Phys. J. C **77**(10), 688 (2017). https://doi.org/10.1140/epjc/s10052-017-5270-2
47. L. Calibbi, A. Crivellin, T. Ota, Phys. Rev. Lett. **115**, 181801 (2015). https://doi.org/10.1103/PhysRevLett.115.181801
48. B. Bhattacharya, A. Datta, J.P. Guévin, D. London, R. Watanabe, JHEP **01**, 015 (2017). https://doi.org/10.1007/JHEP01(2017)015
49. J. Kumar, D. London, R. Watanabe (2018). arXiv:1806.07403
50. J. Aebischer, A.J. Buras, J.M. Gérard (2018). arXiv:1807.01709
51. J. Aebischer, C. Bobeth, A.J. Buras, J.M. Gérard, D.M. Straub (2018). arXiv:1807.02520
52. J. Aebischer, C. Bobeth, A.J. Buras, D.M. Straub (2018). arXiv:1808.00466
53. Wilson github repository. https://github.com/wilson-eft/wilson