The MARTY user interface for the calculation of general Wilson coefficients

Grégoire Uhlrich[a,*]

[a] *University of Geneva, 24 rue du Général-Dufour, 1211 Geneva 4, Switzerland*

E-mail: gregoire.uhlrich@unige.ch

The calculation of one-loop Wilson coefficients for general Beyond the Standard Model (BSM) scenarios is a technical challenge often addressed by doing long and error prone analytical calculations by hand. Several software programs already provide squared amplitude calculations at the loop-level, but few of them are also able to derive general loop-level Wilson coefficients necessary e.g. for the study of quark decays in flavor physics. MARTY, a computer program that automates tree-level and one-loop perturbative calculations for general BSM scenarios can in particular be used to obtain such Wilson coefficients. We present in details the simple user interface allowing to derive common Wilson coefficients in MARTY, and the most general use case of MARTY to extract the coefficient of any effective operator.

Computational Tools for High Energy Physics and Cosmology (CompTools2021)
22-26 November 2021
Institut de Physique des 2 Infinis (IP2I), Lyon, France

[*] Speaker

1. Introduction

Automated calculations beyond the Standard Model have always been a challenge, in particular at the loop level. A lot of software development work has been dedicated to this particular issue in the past decades, with the use of symbolic computation frameworks required for this type of theoretical calculations. Open-source codes implementing their own symbolic computation modules exist such as LanHEP [1] for the vertex derivation from the Lagrangian, or CompHEP [2] and CalcHEP [3] that automate the calculation of tree-level squared amplitudes in a variety of BSM scenarios. Finally, MadGraph_aMC@NLO [4, 5] is also open-source and provides tree-level and one-loop calculation facilities for squared amplitudes. If Mathematica [6], a closed and commercial computer algebra system, can be used, several other packages exist such as FeynRules [7, 8], FeynArts/FormCalc [9, 10] that make use of FORM [11], SARAH [12] that was initially specialized for supersymmetric (SUSY) models, or packages also dedicated to the Wilson coefficients calculations e.g. FormFlavor [13] or FlavorKit [14].

MARTY [15] is a public C++ program, using its own symbolic computation machinery, automating the calculation of amplitudes, squared amplitudes and Wilson coefficients up to the one-loop level for a very large variety of BSM scenarios. A comprehensive documentation is available for MARTY, including manuals [16, 17] and an interactive HTML documentation [18]. All the publicly available material related to MARTY (code, publications, talks, documentation, etc) can be found on the website [19].

The automated analytical calculation of one-loop Wilson coefficients for general BSM scenarios, in particular up to dimension-6 operators (with four fermions), is currently not provided by free-to-use packages other than MARTY. In this conference paper, we present the main steps to calculate such quantities in any model that can be built with MARTY (for more details about model building in MARTY see [16, 20]). The procedure is fully general and allows users to derive for example the loop-level Wilson coefficients relevant for flavor physics, including for (chromo-)magnetic operators and $\Delta F = 1, 2$ dimension-6 operators for e.g. $b \to s$ transitions. Section 2 introduces definitions important to understand what Wilson coefficients are in MARTY, and section 3 presents the main features necessary to extract these quantities for any BSM model. While [21] was mainly focusing on Wilson coefficients for flavor physics giving one complete example, here we highlight the general the general procedure and discuss the extension to general Wilson coefficients.

2. Definitions in MARTY

2.1 Generalities

Wilson coefficients are symbolic scalar expressions in front of operator structures in MARTY's amplitudes. In Effective Field Theories (EFT), amplitudes are the matrix elements of an effective Hamiltonian

$$\mathcal{H}_{eff} \equiv \sum_i C_i \hat{O}_i, \tag{1}$$

with $\hat{O}_i$ effective operators and C_i their respective Wilson coefficients. The transition amplitude between an initial state i and a final state f is defined as the matrix element of this Hamiltonian:

$$i\mathcal{M}(i \to f) = \langle f|(-i\mathcal{H}_{eff})|i\rangle = -i\sum_i C_i \langle f|\hat{O}|i\rangle. \tag{2}$$

The operator matrix elements $\langle f|\hat{O}|i\rangle$ may not in general be calculated perturbatively and can contain long distance effects. However, the BSM dependence lies in the Wilson coefficients and a perturbative calculation is enough to determine their respective values as explained in e.g. [22]. In `MARTY`, a matrix element is simply a particular contraction of external fields. The general case for an amplitude with N external fields $\{\Phi_I^{\{A_I\}}\}_I$ with indices $\{A_I\}$ can be written as

$$i\mathcal{M} = -i\alpha\sum_i C_i \cdot T_i^{\{A_1\}\cdots\{A_N\}} \cdot \Phi_1^{\{A_1\}} \cdots \Phi_N^{\{A_N\}}, \tag{3}$$

with $T_i^{\{A_1\}\cdots\{A_N\}}$ all different tensors contracting the external fields to each other in the resulting amplitude and α a convention dependent constant. Therefore, by multiplying the amplitude by i/α the Wilson coefficients can be directly identified in front of the different matrix elements.

`MARTY` can decompose amplitudes in independent external field contractions and give the coefficients in front, taking into account a global user-defined factor α. The matrix element (a.k.a operator in `MARTY`) is therefore the contraction of fields in the amplitude (including possible tensor couplings), and the Wilson coefficient is the scalar multiplicative factor in front. The particular cases of dimension-5 and dimension-6 operators are discussed in the following.

LO vs. NLO As implicitly stated above, the matching used by `MARTY` is trivial and in particular no explicit calculation is performed in the effective theory. This is because `MARTY` provides automated procedures only for the Leading Order (LO), at tree-level or at the one-loop level. In order to obtain Next-to-Leading Order (NLO) Wilson coefficients (e.g. a one-loop calculation for a process that is non-zero at tree-level) one has to perform the same calculation in the effective theory and match the result on the full theory. Although this can be done with `MARTY`, for now no automated procedure allows us to obtain NLO coefficients. Such a procedure could be developed in the future. For further numerical computations from the Wilson coefficients e.g. applying the Renormalization Group Equations (RGE), dedicated open-source codes already exist such as `SuperIso` [23–26] for flavor physics.

2.2 (Chromo-)Magnetic operators

Dimension-5 operators are defined for two fermions ψ_1, ψ_2 and a vector boson B as

$$O_{\text{mag}} \equiv \left(\bar{\psi}_1 (T^A)\sigma^{\mu\nu}\Gamma\psi_2\right) F_{\mu\nu}^{(A)}, \tag{4}$$

with (T^A) the algebra generator when relevant, $F_{\mu\nu}^{(A)}$ the field strength of B and

$$\Gamma \in \left\{\mathbb{1}, \gamma^5, P_L, P_R\right\}. \tag{5}$$

To fully define a magnetic operator, a user therefore only has to choose one element picked in a set of 4 elements. The algebra generator (T^A) does not have to be user defined as its presence is determined by the particle types.

2.3 4-fermions operators

Dimension-6 operators with fermions ψ_1, ψ_2, ψ_3 and ψ_4 are defined by operators of the type

$$O_{d=6} \equiv T_{ijkl} \left(\bar{\psi_1}^i \Gamma^A \psi_2^j \right) \left(\bar{\psi_3}^k \Gamma^B \psi_4^l \right), \tag{6}$$

with Dirac couplings

$$\begin{aligned} \Gamma^A, \Gamma^B \in \{ & \\ & \mathbb{1}, \gamma^5, P_L, P_R, \\ & \gamma^\mu, \gamma^\mu\gamma^5, \gamma^\mu P_L, \gamma^\mu P_R, \\ & \sigma^{\mu\nu}, \sigma^{\mu\nu}\gamma^5, \sigma^{\mu\nu} P_L, \sigma^{\mu\nu} P_R \\ \}, & \end{aligned} \tag{7}$$

with Γ^A and Γ^B contracting to leave no free Minkowski index. The indices i, j, k, and l in equation 6 are gauge indices, contracted by T_{ijkl} that can be of four main kinds:

$$\begin{aligned} T_{ijkl} &= \delta_{ij}\delta_{kl}, \\ T_{ijkl} &= \delta_{il}\delta_{kj}, \\ T_{ijkl} &= \delta_{ik}\delta_{jl}, \\ T_{ijkl} &= T^A_{ij}T^A_{kl}. \end{aligned} \tag{8}$$

To fully define a dimension-6 operator, Γ^A, Γ^B and T_{ijkl} must therefore be provided.

3. The user interface

In this section the user interface to obtain the Wilson coefficients of the operators defined above is presented. Four main steps have to be followed in MARTY:

- Options setup for the amplitude calculation.
- Amplitude calculation, including the decomposition on an operator basis.
- Definition of the operator of which the coefficient must be extracted.
- Extraction of the coefficient.

Considering e.g. a process $\psi_1 \to \psi_2 B$ and a MARTY model in the `model` variable, the two first steps can be performed using:

```
FeynOptions options;
Expr factor = ...; // Convention-dependent factor to be defined if needed
options.setWilsonOperatorCoefficient(factor);
vector<Wilson> wilsons = model.computeWilsonCoefficients(
        OneLoop,
        {Incoming("psi1"), Outgoing("psi2"), Outgoing("B")},
        options);
```

For the details on how to define the convention-dependent factor we refer to the user manual [16]. After the calculation, the `wilsons` variable contains the decomposed amplitude but work still needs to be done to extract particular coefficients from the result as explained in the the next sections.

The fermion ordering option For 4-fermion operators, the order of external fermions in the operator basis must be user-defined. From the initial order given when defining the external particles of the calculation, the final order is defined as a permutation of the initial order. Considering a four fermion process $\psi_1 \to \bar{\psi}_2\psi_3\psi_4$, a fermion order $(2, 0, 3, 1)$ corresponds to operators of the type

$$(\bar{\psi}_3\Gamma^A\psi_1)(\bar{\psi}_4\Gamma^B\psi_2), \tag{9}$$

where $\Gamma^{A,B}$ are generalized couplings. The indices are defined starting from 0, a valid permutation is therefore a permutation of $(0, 1, 2, 3)$. The fact that particles are incoming or outgoing is not relevant for this ordering. Such orderings have to be defined in the options before the amplitude calculation. In the example above the following option must be defined

```
options.setFermionOrder({2, 0, 3, 1});
```

3.1 Operator definition

For common operators, built-in functions exist to create them without having to explicitly construct their explicit analytical expression.[1] This is the case for magnetic dimension-5 operators and dimension-6 operators with 4 fermions.

In order to easily define all possible operators for $d = 5$ and $d = 6$ discussed in section 2.2 and 2.3 respectively, the different Dirac and color couplings are stored in enumerations. These enumerations are presented in tables 1 and 2 respectively.

Using the enumeration presented in table 1, the `chromoMagneticOperator()` method can be used to build the relevant dimension-5 operators defined in equation 4 e.g. for C_7 ($b \to s\gamma$ decay) or $(g-2)_\mu$:

```
vector<Wilson> O_7 = chromoMagneticOperator(
    model, wilsons, DiracCoupling::R);
   // Fermion current (sigma P_R)
vector<Wilson> O_gm2 = chromoMagneticOperator(
    model, wilsons, DiracCoupling::S);
   // Fermion current (sigma)
```

A similar principle exists for the dimension-6 operators defined in equation 6, this time two Dirac couplings must be given to define the two fermion currents:

[1]General operators can also be defined explicitly, see the user manual [16].

Enumeration element	Name	Expression
`DiracCoupling::S`	Scalar	$\mathbb{1}$
`DiracCoupling::P`	Pseudo-scalar	γ^5
`DiracCoupling::L`	Left	P_L
`DiracCoupling::R`	Right	P_R
`DiracCoupling::V`	Vector	γ^μ
`DiracCoupling::A`	Axial	$\gamma^\mu\gamma^5$
`DiracCoupling::VL`	Vector left	$\gamma^\mu P_L$
`DiracCoupling::VR`	Vector right	$\gamma^\mu P_R$
`DiracCoupling::T`	Tensor	$\sigma^{\mu\nu}$
`DiracCoupling::TA`	Tensor axial	$\sigma^{\mu\nu}\gamma^5$
`DiracCoupling::TL`	Tensor left	$\sigma^{\mu\nu}P_L$
`DiracCoupling::TR`	Tensor right	$\sigma^{\mu\nu}P_R$

Table 1: Dirac couplings available to define operator structures in `MARTY`.

Enumeration element	Name	Expression
`ColorCoupling::Id`	Identity	$\delta_{ij}\delta_{kl}$
`ColorCoupling::Crossed`	Crossed	$\delta_{il}\delta_{kj}$
`ColorCoupling::InvCrossed`	Crossed inversed	$\delta_{ik}\delta_{jl}$
`ColorCoupling::Generator`	Generator	$T^A_{ij}T^A_{kl}$

Table 2: Color couplings possible to define dimension-6 operators in `MARTY`. See equation 6 for the definition of the indices $ijkl$.

```
vector<Wilson> O_1 = dimension6Operator(model, wilsons,
    DiracCoupling::L, DiracCoupling::R); // (P_L)x(P_R)
vector<Wilson> O_2 = dimension6Operator(model, wilsons,
    DiracCoupling::VL, DiracCoupling::V); // (G^mu P_L)x(G_mu)
```

When a gauge tensor coupling of a dimension-6 operator is not trivial, it is possible to specify another one giving the gauge group name (`"C"` for color group in the example) and an element of the enumeration presented in table 2:

```
vector<Wilson> O1_crossed = dimension6Operator(model, wilsons,
    DiracCoupling::L, DiracCoupling::R,
    {"C", ColorCoupling::Crossed}
    ); // (P_L)_ij x (P_R)_ji
```

3.2 Wilson coefficient extraction

Finally, after calculating the amplitude and building the relevant operators as previously discussed, the extraction of the final Wilson coefficients is very simple using the `getWilsonCoefficient()` method:

```
Expr C7 = getWilsonCoefficient(wilsons, O_7);
Expr gm2 = getWilsonCoefficient(wilsons, O_gm2);
Expr C1 = getWilsonCoefficient(wilsons, O_1);
Expr C2 = getWilsonCoefficient(wilsons, O_2);
Expr C1p = getWilsonCoefficient(wilsons, O1_crossed);
```

The resulting variables are simple `MARTY` symbolic expressions and can therefore directly be used for library generation and numerical evaluation as usual.[2]

3.3 Generalization of the operator definition

In `MARTY` it is also possible to extract Wilson coefficients of generic operators. The principle is to create the analytical expression, in `MARTY`, of the operator of which the coefficient must be extracted. Then, `MARTY` automatically searches in the amplitude for the user-defined operator and extracts its coefficient. The creation of custom effective operators is very similar to the creation of general Lagrangian terms and should feel familiar for a user already accustomed to model building procedures in `MARTY`.

The momenta of a process need in general to be obtained from `MARTY` to define operators. This can be done using:

```
vector<Tensor> p = wilsons.kinematics.getOrderedMomenta();
// p[0], p[1], p[2] are momenta p1, p2, p3 for a three particles process
```

To obtain the particles, tensors and indices, the procedure is the same as for model building. In order to create gauge indices in the relevant vector spaces, it is necessary to specify the group (or its name) and the irreducible representation (or a particle). For the example of a $b \to s\gamma$ process this gives:

```
// The fields
Particle b = model.getParticle("b");
Particle s = model.getParticle("s");
Particle A = model.getParticle("A");
```

[2]For more details on this procedure see the simple example on the website `https://marty.in2p3.fr/gettingStarted.html` or the user manual [16].

```
// Additional tensors
Tensor gamma = dirac4.gamma;
// Index for the triplet (e.g. quark "b")
// in the SU(3) color group "C":
Index i = model.generateIndex("C", "b");
// Dirac and Minkowski indices
vector<Index> al = DiracIndices(2);
Index mu = MinkowskiIndex();
```

Once all objects have been retrieved from MARTY, the operator expression can be built explicitly in a symbolic MARTY expression. Considering the example of the $\bar{s}(p_2)\not{A}(p_3)b(p_1)$ operator, the corresponding MARTY expression is:[3]

```
Expr Op = GetComplexConjugate(s({i, al[0]}, p[1]))*A(mu, p[2])
       *gamma({+mu, al[0], al[1]})*b({i, al[1]}, p[0]);
```

Finally, the corresponding coefficient can be extracted by MARTY giving the operator previously constructed:

```
Expr C = getWilsonCoefficient(wilsons, Op);
```

This procedure is completely general, is valid for all groups and representations, and allows users to extract the Wilson coefficients of all operators that have not been explicitly implemented in the simple user interface for $d = 5$ and $d = 6$ operators.

4. Conclusion

We presented the user interface in MARTY allowing one to extract in a simple way the Wilson coefficients of $d = 5$ and $d = 6$ operators at the one-loop level. These coefficients are necessary for the calculation of phenomenogically-motivated quantities such as e.g. $(g-2)_\mu$ or the coefficients of quark decays in flavor physics. Furthermore, we showed how to generalize the Wilson coefficient extraction to any effective operator using a procedure similar to the Lagrangian construction in MARTY.

We defined analytically the operators in section 2 by highlighting their specificity and the minimum quantity of information required from a user to uniquely define them. Section 3 then presented the interface to build these operators in a MARTY program and extract their coefficients for a given process. In a few lines of code, it is possible to extract Wilson coefficients for $d = 5$ and

[3]For signed indices such as Minkowski indices, it is necessary to specify if the index given to MARTY is up or down. By default mu is down (e.g. A_μ) and +mu is up (e.g. A^μ).

$d = 6$ operators that can directly be used by the library generation facility of MARTY for numerical evaluation. With a reasonable amount of work, the Wilson coefficients of general operators can also be obtained with MARTY using the generic operator definition features.

The procedures presented in this proceeding have two very important features:

- The code is completely model-independent. Once the procedure is set for the extraction of one or several Wilson coefficients, changing the model in MARTY (considering that it has been built) takes only one line and the program will execute in the exact same way.
- Downloading and installing MARTY, a free and open-source code, is a sufficient condition to use all the features discussed here.

As it has already been showed for beauty quark decays in non-minimal flavor violating MSSM[4] scenarios [27], the user interface for the automated extraction of Wilson coefficients with MARTY will greatly facilitate the BSM analyses relying on such theoretical calculations in a large variety of models.

References

[1] A. Semenov, *LanHEP: A Package for the automatic generation of Feynman rules in field theory. Version 3.0*, Comput. Phys. Commun. **180** (2009) 431 [0805.0555].

[2] CompHEP collaboration, *CompHEP 4.4: Automatic computations from Lagrangians to events*, Nucl. Instrum. Meth. A **534** (2004) 250 [hep-ph/0403113].

[3] A. Pukhov et al., *CompHEP: A Package for evaluation of Feynman diagrams and integration over multiparticle phase space*, hep-ph/9908288.

[4] J. Alwall et al., *The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations*, Journal of High Energy Physics **2014** (2014) .

[5] R. Frederix et al., *The automation of next-to-leading order electroweak calculations*, Journal of High Energy Physics **2018** (2018) .

[6] Wolfram Research, Inc., "Mathematica, Version 12.1." https://www.wolfram.com/mathematica.

[7] A. Alloul et al., *FeynRules 2.0 - A complete toolbox for tree-level phenomenology*, Comput. Phys. Commun. **185** (2014) 2250 [1310.1921].

[8] C. Degrande, *Automatic evaluation of UV and R2 terms for beyond the Standard Model Lagrangians: a proof-of-principle*, Comput. Phys. Commun. **197** (2015) 239 [1406.3030].

[9] T. Hahn, *Generating Feynman diagrams and amplitudes with FeynArts 3*, Computer Physics Communications **140** (2001) 418–431.

[4]Minimal Supersymmetric Standard Model.

[10] T. Hahn and M. Perez-Victoria, *Automatized one loop calculations in four-dimensions and D-dimensions*, *Comput. Phys. Commun.* **118** (1999) 153 [hep-ph/9807565].

[11] B. Ruijl, T. Ueda and J. Vermaseren, *FORM version 4.2*, 1707.06453.

[12] F. Staub, *Exploring new models in all detail with SARAH*, *Adv. High Energy Phys.* **2015** (2015) 840780 [1503.04200].

[13] J.A. Evans and D. Shih, *FormFlavor Manual*, 1606.00003.

[14] W. Porod, F. Staub and A. Vicente, *A Flavor Kit for BSM models*, *Eur. Phys. J. C* **74** (2014) 2992 [1405.1434].

[15] G. Uhlrich, F. Mahmoudi and A. Arbey, *MARTY – Modern ARtificial Theoretical phYsicist: A C++ framework automating theoretical calculations Beyond the Standard Model*, *Computer Physics Communications* **264** (2021) 107928.

[16] G. Uhlrich, "MARTY – User manual." https://marty.in2p3.fr/doc/marty-manual.pdf, 2021.

[17] G. Uhlrich, "CSL – User manual." https://marty.in2p3.fr/doc/csl-manual.pdf, 2020.

[18] G. Uhlrich, "Documentation of MARTY." https://marty.in2p3.fr/doc/marty/html/index.html, 2021.

[19] G. Uhlrich, "MARTY – Website." https://marty.in2p3.fr/, 2021.

[20] G. Uhlrich, F. Mahmoudi and A. Arbey, *Semi-automated BSM model building procedures in MARTY-1.1 through a 2HDM example*, *PoS* **TOOLS2020** (2021) 042.

[21] G. Uhlrich and F. Mahmoudi and A. Arbey, *Automatic extraction of one-loop Wilson coefficients in general BSM scenarios using MARTY-1.4*, 2110.14515.

[22] A.J. Buras, *Weak Hamiltonian, CP violation and rare decays*, in *Les Houches Summer School in Theoretical Physics, Session 68: Probing the Standard Model of Particle Interactions*, Jun, 1998 [hep-ph/9806471].

[23] F. Mahmoudi, *SuperIso: A Program for calculating the isospin asymmetry of $B \to K^*\gamma$ gamma in the MSSM*, *Comput. Phys. Commun.* **178** (2008) 745 [0710.2067].

[24] F. Mahmoudi, *SuperIso v2.3: A Program for calculating flavor physics observables in Supersymmetry*, *Comput. Phys. Commun.* **180** (2009) 1579 [0808.3144].

[25] F. Mahmoudi, *SuperIso v3.0, flavor physics observables calculations: Extension to NMSSM*, *Comput. Phys. Commun.* **180** (2009) 1718.

[26] S. Neshatpour and F. Mahmoudi, *Flavour Physics with SuperIso*, *PoS* **TOOLS2020** (2021) 036 [2105.03428].

[27] M. A. Boussejra and F. Mahmoudi and G. Uhlrich, *Flavour anomalies in supersymmetric scenarios with non-minimal flavour violation*, 2201.04659.