`FormFlavor` Manual – Version 1.1.0

Jared A. Evans[1] and David Shih[2]

[1]Department of Physics, University of Illinois at Urbana-Champaign, Urbana, IL 61801
[2]NHETC, Department of Physics and Astronomy, Rutgers University, Piscataway, NJ 08854

September 25, 2017

Abstract

This manual describes the usage and structure of `FormFlavor`, a `Mathematica`-based tool for computing a broad list of flavor and CP observables in general new physics models. Based on the powerful machinery of `FeynArts` and `FormCalc`, `FormFlavor` calculates the one-loop Wilson coefficients of the dimension 5 and 6 Standard Model effective Lagrangian entirely from scratch. These Wilson coefficients are then evolved down to the low scale using one-loop QCD RGEs, where they are transformed into flavor and CP observables. The last step is accomplished using a model-independent, largely stand-alone package called `FFObservables` that is included with `FormFlavor`. The SM predictions in `FFObservables` include up-to-date references and accurate current predictions. Using the functions and modular structure provided by `FormFlavor`, it is straightforward to add new observables. Currently, `FormFlavor` is set up to perform these calculations for the general, non-MFV MSSM, but in principle it can be generalized to arbitrary `FeynArts` models. `FormFlavor` and an up-to-date manual can be downloaded from: http://formflavor.hepforge.org.

Contents

1   Introduction

Precision flavor and CP observables, such as Δm_K, ϵ_K and $\mathrm{BR}(B \to X_s \gamma)$, have long been invaluable probes of physics beyond the Standard Model (BSM) (see e.g. the review [1]). While the Standard Model enjoys approximate flavor and CP symmetries that keep these observables small, new physics generally does not. Indeed, many models of new physics predict deviations at some level from Standard Model expectations. A notable example is supersymmetry (SUSY), which is highly motivated on many grounds, yet generically predicts large contributions to flavor- and CP-violating observables (see e.g. [2] for a review and original references).

With new, high-precision experimental results in flavor physics on the horizon, such as those promised by Belle II [3], LHCb [4], and NA62 [5, 6]; and lattice calculations evolving into an era of higher and higher precision [7], there is increasing need for equally precise theoretical tools to facilitate the exploration of constraints on new physics models. In general, theoretical predictions for flavor observables are derived from the Wilson coefficients of dimension 5 and dimension 6 effective operators built out of Standard Model fields. There are many such operators, and, in many models (e.g., SUSY), these Wilson coefficients do not arise at tree-level, necessitating the calculation of one- or even higher-loop diagrams. With the many loop functions and Wilson operators involved, performing an accurate assessment of the flavor constraints on even a single parameter point in the MSSM across a dozen distinct observables is an onerous task to perform by hand. Although several publicly available programs exist to calculate flavor and CP observables, these often assume minimal flavor violation (MFV), lack a sufficiently broad list of flavor observables, are numerically unstable, or contain bugs, likely introduced in transcribing loop formulas by hand from the literature.

It was with these issues in mind that `FormFlavor` was designed. `FormFlavor` is a `Mathematica`-based, general-purpose tool for computing a broad list of flavor and CP observables in new physics models. `FormFlavor` is built on the powerful machinery of `FeynArts` [8] and `FormCalc` [9], which facilitate the automatic generation and evaluation of Feynman diagrams for general Lagrangians. Using `FeynArts` and `FormCalc`, `FormFlavor` calculates the one-loop Wilson coefficients from scratch, greatly improving the reliability of the code. Currently, `FormFlavor` is set up to perform these calculations for the general, non-MFV MSSM, but in principle it can be generalized to arbitrary models. `FormFlavor` contains two distinct running modes, Fast and Accurate, allowing for a safer evaluation of Passarino-Veltmann integrals without suffering from numerical instabilities.
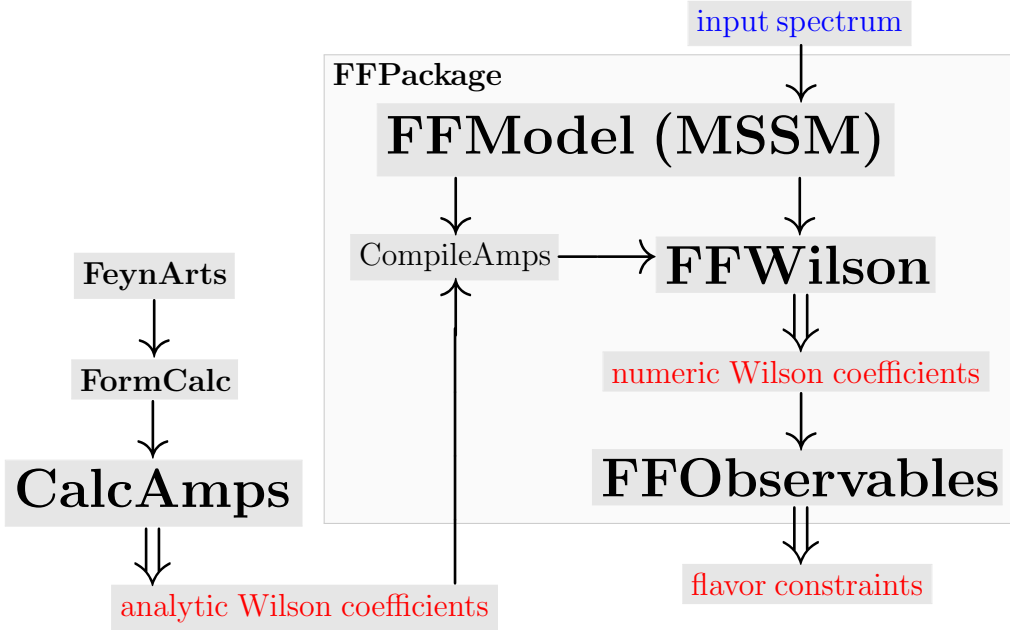
Figure 1: Schematic illustration of the `FormFlavor` code. The left branch is `CalcAmps`, which, in principle, only needs to be run once per model; the middle is the compiling portion of `FormFlavor`, which must be run once per session; to the right is the core code of `FormFlavor`, which is run once per parameter point evaluated. In blue is the primary input, a spectrum. In red are the primary outputs of `FormFlavor`: the analytic Wilson coefficients, numerical Wilson coefficients, and flavor constraints. The four main pieces of the code: `FFObservables`, `FFWilson`, `FFModel` (MSSM), and `CalcAmps`, are discussed in sections 4, 5, 6, and 7, respectively. `CalcAmps` requires `FeynArts` and `FormCalc`, but the rest of the code does not.

The structure of the `FormFlavor` code is illustrated by the flowchart in fig. 1. `FormFlavor` can be viewed as two distinct programs, `CalcAmps` and `FFPackage`, and `FFPackage` contains a number of separate modules with different functionality.

- `CalcAmps` automatically generates one-loop amplitudes from a `FeynArts`/`FormCalc` model file and extracts analytic expressions for the Wilson coefficients. `CalcAmps` only needs to be run once per model in principle, or re-run whenever the user wishes to add an observable. (For the default observables and the default model – the general flavor- and CP-violating MSSM – `CalcAmps` need not be run at all; rather the user can use the amplitude files that come with `FormFlavor`.)

- `FFPackage` contains the core code of `FormFlavor`, which is run repeatedly per `Mathematica` session in order to turn input spectra into flavor and CP observables.

The modules of `FFPackage` include:

– `FFWilson` takes input spectra and numericizes the analytic Wilson coefficients generated by `CalcAmps`. It also compiles the analytic Wilson coefficients for faster numerical evaluation. The compilation step is taken care of by `CompileAmps` and needs to be run only once per `Mathematica` session.

– `FFObservables` converts numerical Wilson coefficients into flavor and CP observables. `FFObservables` is entirely model independent, and could in principle be run as a standalone package, given a list of Wilson coefficients and the scale where they are defined. Observables are treated in a modular way in `FFObservables`, making it straightforward to add new ones to `FormFlavor`. This is described in section 7.2.

– `FFModel` contains all of the model-specific code used by `FormFlavor`. This includes code to read in input spectra in a user-specified format, and code to link the `CalcAmps` output with `FFWilson`. Although the only `FFModel` file that currently exists is the fully general, non-MFV MSSM, in principle, other `FeynArts` models can be used to generate one-loop amplitudes. A user only needs to write new `FFModel` modules, at which point the existing `FFWilson` and `FFObservables` machinery can evaluate them.

`FormFlavor` does not treat any higher-loop contributions, such as the double Higgs penguins [10] or Barr-Zee diagrams [11] that can be important for Δm_{B_s} and the neutron EDM, respectively. Unlike the publicly available program, `SUSY_Flavor` [12], `FormFlavor` does not account for chirally-enhanced contributions, important mostly at large $\tan \beta$, that enter at 2-loops or higher.

In this manual, we will first present a simple QuickStart guide in section 2. For the user who wants to acquire flavor constraints from an SLHA2 file, this QuickStart guide contains all of the information needed. Section 3 introduces the main `FormFlavor` package. Section 4 presents the details of `FFObservables`, and how all of the observables contained in `FormFlavor` are evaluated. All details of `FFWilson` are discussed in section 5. The details of the single `FFModel` included with the public release, the non-MFV MSSM model, are presented in section 6. The `CalcAmps` package is described in section 7, along with a tutorial on adding new observables to `FormFlavor`. A comparison of `FormFlavor` with the public codes `SUSY_Flavor` [12] and `FlavorKit` [13] is presented in the appendix.

`FormFlavor` has been tested with `Mathematica` 8, 9, and 10 in Mac OS X 10.9.5 (Mavericks) and Mac OS X 10.10.5 (Yosemite). All comments on speed throughout the text concern `Mathematica` 9 with Mac OS X 10.10.5 using 8 GB of RAM and a 2.9 GHz Intel Core i5.

2 Basic User's Guide

The basic use of `FormFlavor` takes as input an SLHA2 file in the flavor- and CP-violating MSSM and computes the one-loop contribution to various flavor observables. Contributions from diagrams containing only standard model particles utilize hardcoded, detailed treatments extracted from the literature, while the one-loop MSSM contributions are evaluated from scratch. In order to combine the two, the new physics Wilson coefficients must be RG evolved from the SUSY scale down to the relevant low scale for the flavor observable of interest.

In this section, we present a brief QuickStart guide that will illustrate how to take the package out of the box and evaluate flavor observables from an SLHA2 file. For many users, it is likely that the information presented in this section is all of the functionality of `FormFlavor` that will be needed.

2.1 Starting the Program

`FormFlavor` can be downloaded from: http://formflavor.hepforge.org. Once downloaded, the tar ball should be unpacked. No installation of the package is necessary.

The `FormFlavor` program is loaded using a front-end `Mathematica` notebook, such as the `FormFlavor.nb` notebook provided with the package. Before loading, the path of `FormFlavor` must be specified with:

```
FormFlavor`$FFPath={PATH}
```

The package can be loaded using the `Mathematica` command (`Get`):

```
<<FormFlavor`$FFPath<>"FormFlavor`"
```

Within a few seconds, the package should load, listing `FormFlavor`'s version number, the `FFModel`'s version number, and several different portions of the code that were loaded. No other `Mathematica` packages (such as `FeynArts` or `FormCalc`) need to be loaded in order for the code to work.

2.2 Compiling the Amplitudes

Once loaded, many of the code's routines are already functional. Notably, all of the `FFObservables` capabilities can be run (see section 4 for details). However, the main operation of `FFWilson`, calculating numerical Wilson coefficients from MSSM parameters, needs to be compiled before it will function. This compiling takes time and has two separate running modes ("Fast" and accurate, "Acc"). Although the differences between these two modes are discussed in detail in section 5.1, the basic difference is that "Fast" mode evaluates results more quickly, but is subject to occasional numerical instabilities in Passarino-Veltman integral calculations due to the use of double precision, while "Acc" mode (accurate mode) is substantially slower, but is more resistant to these instabilities. "Fast" mode is good for performing parameter scans; "Acc" mode is good for evaluating individual points.

When `FormFlavor` is loaded, all existing observable amplitude files are stored in `$FFAmpFileList`. Running:

> `CompileFF[$FFAmpFileList] or CompileFF[$FFAmpFileList,"Fast"]`

will compile the code in "Fast" mode. Trading "Fast" for "Acc" will compile in "Acc" mode.

`CompileFF[amplist]`   compiles all amplitudes in `amplist` in "Fast" mode

`CompileFF[amplist,mode]`   as above, but for running mode `mode`

Compiling the amplitudes takes several minutes. Although only one mode needs to be compiled for the program to function, both running modes can be simultaneously loaded. After compiling a particular mode, the default running mode is set to that mode. This can be changed at any time by setting `$FFActiveRunningMode="Fast" or "Acc"`. However, we note that compiling these large amplitudes stores them into memory, which can consume several GB of RAM. Compiling both modes simultaneously effectively doubles this RAM consumption. By decreasing the number of processes loaded, especially the larger memory hogs, such as $B_q \to \mu\mu$, the memory expenditure can be reduced.

Optionally, one can pre-evaluate many of the time-consuming operations performed in the compiling step by building the amplitudes (section 5.2). This exchanges disk space to cut down compiling time by more than factor of two.

status	exclusion value	explanation
0	$\dfrac{X_{FF}-X_{exp}}{\sqrt{\sigma_{exp}^2+\sigma_{th}^2}}$	existing theoretical prediction and measurement
1	$\dfrac{X_{FF}}{X_{exp,UB}}$	an experimental upper bound only
2	$\dfrac{X_{FF}}{X_{exp}+2\sigma_{exp}}$	experimental measurement, but no reliable theory prediction

Table 1: The definitions of the exclusion values outputted by `FFConstraintsfromSLHA2` & `FFConstraints` (which is defined in section 3), according to the observable status. Here, X_{FF} is the standard model + new physics value as determined by FormFlavor for the observable X, X_{exp} ($X_{exp,UB}$) is the current experimental measurement's central value (upper bound), σ_{exp} is the experimental uncertainty on the measurement of X, and σ_{th} is the theoretical uncertainty on the standard model prediction for X. The treatment of these observables is defined in the function `FFObsMeasure` within `Core/FFPackage.m`.

2.3 Calculating Observables

Once the code has been compiled, the flavor observables can be calculated from the SUSY-scale parameters of an SLHA2 file by using the command:

`FFfromSLHA2[file]`	read in `file` at the SUSY scale and output flavor observables using current active running mode
`FFfromSLHA2[file,mode]`	as above, but for running mode `mode`

This function returns a nested list of flavor observables in `Mathematica` list format as,

{{observable name, observable value},...}

In order to get the flavor constraints, one can use the command:

`FFConstraintsfromSLHA2[file]`	read in `file` at the SUSY scale and give flavor constraints with current active running mode
`FFConstraintsfromSLHA2[file,mode]`	as above, but for running mode `mode`

The constraints from this are presented in a list of the form

{{observable name, exclusion value, status},...}

where the definition of exclusion value depends on the current experimental/theoretical status of the observable. These definitions are summarized in table 1. A status of "0" corresponds to the common situation where there is both a reliable theoretical prediction in the standard model and an experimental measurement – example: $\mathrm{BR}(B \to X_s\gamma)$. For observables with this status, an exclusion value with a magnitude greater than two represents a roughly 95% CL exclusion. A status of "1" is a situation where there is an experimental upper bound only – example: the neutron EDM. For many of these observables, no standard model prediction is included in the `FormFlavor` calculation as these values are much lower than the current experiment measurement. For observables of this class, an exclusion value with a magnitude larger than one represents a spectrum excluded at roughly 90% CL. A status of "2" corresponds to a situation where there is an experimental measurement, but no existing, reliable theoretical prediction – example: Δm_D. For observables with this status, an exclusion value much larger than one represents a substantial tuning between the unknown (or poorly predicted) standard model contributions and the new physics contributions.

3 The Main Package: FFPackage

In this section we will describe the contents of `FFPackage`, which is the main wrapper package of `FormFlavor`, responsible for loading all other packages (except `CalcAmps`, see section 7) and defining the main `FormFlavor` routine that turns input spectra into flavor and CP observables.

`FormFlavor` has many parameters common to multiple observables, for example, the CKM matrix. In order to simplify the process of updating these parameters with improved measurements, most parameters are stored in a single file, `Core/SMParameters.m`, that is automatically loaded by `FFPackage`. Within this file there are conversion factors from SI units to natural units, standard model particle masses at different scales and often in different schemes, electroweak parameters such as the Higgs vacuum expectation value, Fermi's constant G_F, and $\sin\theta_W$, meson masses, decay constants, lifetimes, and branching ratios, and lepton lifetimes. All parameters are given in natural units of GeV^n. Note that `FormFlavor` will ignore SM parameters provided in an SLHA2 file, choosing to use the hardcoded values provided in `Core/SMParameters.m`.

`GetSMParameter["name"]`	returns SM parameter `"name"`
	type `??GetSMParameter` for a complete list

10

`FFPackage` also establishes the basis of Wilson operators used by `FormFlavor`:

$$\mathcal{O}_A^M(f_1, f_2) = e\overline{f}_1\sigma^{\mu\nu}P_M f_2 F_{\mu\nu} \tag{3.1}$$

$$\mathcal{O}_G^M(f_1, f_2) = g\overline{f}_1\sigma^{\mu\nu}P_M f_2 G_{\mu\nu} \tag{3.2}$$

$$\mathcal{O}_S^{MN}(f_1, f_2, f_3, f_4) = (\overline{f}_1 P_M f_2)(\overline{f}_3 P_N f_4) \tag{3.3}$$

$$\mathcal{O}_V^{MN}(f_1, f_2, f_3, f_4) = (\overline{f}_1 \gamma^\mu P_M f_2)(\overline{f}_3 \gamma_\mu P_N f_4) \tag{3.4}$$

$$\mathcal{O}_T^{MN}(f_1, f_2, f_3, f_4) = (\overline{f}_1 \sigma^{\mu\nu} P_M f_2)(\overline{f}_3 \sigma_{\mu\nu} P_N f_4) \tag{3.5}$$

where $M, N = L, R$ with $P_R = \frac{1}{2}(1 + \gamma_5)$ and $P_L = \frac{1}{2}(1 - \gamma_5)$. All colors are contracted within the bilinear when such a Fierz rearrangement is possible. Although currently no operators are used in `FormFlavor` that cannot be rearranged, this information is encoded by the `FormCalc` notation of:

`SUNT[Col1,Col2] SUNT[Col3,Col4]`	color contractions within the fermion bilinears, i.e., $(\overline{f}_{1,\alpha}X_M f_{2,\alpha})(\overline{f}_{3,\beta}X_N f_{4,\beta})$
`SUNT[Col1,Col4] SUNT[Col3,Col2]`	color contractions outside of the fermion bilinears, i.e., $(\overline{f}_{1,\alpha}X_M f_{2,\beta})(\overline{f}_{3,\beta}X_N f_{4,\alpha})$

All Wilson operators in `FormFlavor` are expressed as

$$\texttt{OpX}[P_1,P_2][\{f_1, f_2\}, \{f_3, f_4\}] \text{ or } \texttt{OpY}[P_1][\{f_1, f_2\}, \{v\}]$$

for $X = S, V, T$ or $Y = A, G$, $P_i = $ "L", "R", $v = $ "γ", "g" $f_i = $ "b", "d", etc. As examples `OpV`["L","L"][{"s","d"},{"s","d"}] is the $\mathcal{O}_V^{LL}$ for kaon mixing and `OpA`["R"][{"s","b"},{"γ"}] is the $\mathcal{O}_A^R$ relevant for $b \to s\gamma$. These effective operators are combined with their respective Wilson coefficients into the effective Hamiltonian:

$$H_{eff} = \sum_i C_i \mathcal{O}_i \tag{3.6}$$

More details about the Wilson coefficients will be discussed in section 5.3.

Next, `FFPackage` loads `FFObservables` and `FFWilson`, the two central packages of `FormFlavor`. The first contains all the observable functions, taking general Wilson coefficients as inputs. So in principle it could be used independently of the rest of `FormFlavor`, for any model, provided one has a way to turn that model into Wilson coefficients. The second contains the code that compiles the analytic amplitudes and extracts the Wilson coefficients from them. We will describe them in more detail in sections 4 and 5.

Finally, `FFPackage` defines the main function, also called `FormFlavor`, that takes input spectra and turns them into flavor and CP observables. This function is called internally by the observable and constraint functions described in section 2.3.

`FormFlavor[VariableList]`	evaluate flavor observables for all compiled processes from the numerical input `VariableList` in the current active running mode
`FormFlavor[VariableList,mode]`	as above, but for running mode `mode`
`FFConstraints[FFout]`	compute the flavor constraints from the `FormFlavor` output in the current active running mode
`FFConstraints[FFout,mode]`	as above, but for running mode `mode`

Here `VariableList` is the output from `CalcSpec` or `FFRunFile` (see 6.1) in the form presented in `$FFCompileVarList`. `FFConstraints` takes as input the output of the `FormFlavor` command, and outputs constraints in the form discussed in table 1 and section 2.3.

4   The Observable Functions: FFObservables

In this section, we will walk through the explicit treatment of the many flavor and CP observables provided in the standard release of `FormFlavor`. (Detailed instructions on how to add additional observables are discussed in section 7.2.) These observables are listed in table 2 and they are all loaded by the `FFObservables` portion of the `FormFlavor` package. We stress that `FFObservables` is in principle a general-purpose, model-independent package in its own right, that takes Wilson coefficients at the new physics scale and returns flavor and CP observables. Even if the user does not take advantage of the rest of `FormFlavor`'s functionality, he or she may find `FFObservables` useful on its own.

Loading the `FFObservables` package loads the individual observable module files. (There is one such file for each grouped set of rows in table 2.) These are all located in the `Core/Observables` directory. Each observable module is a model-independent file containing an observable function, as well as the current experimental values, and the experimental and theoretical uncertainties. We will describe the structure of the observable modules in more detail in the next subsection.

`FFObservables` also defines the main function, also called `FFObservables`. This function takes as input the RG scale of the Wilson coefficients, the effective Hamiltonian `Heff` as defined in (3.6), and a list of desired observables. Then `FFObservables` loops over these observables, calls the required observable function, and returns a list of the observables and their numerical values for the given Wilson coefficients.

Observable	Experiment	SM prediction
Δm_K	$(3.484 \pm 0.006) \times 10^{-15}$ GeV	$-$
ϵ_K	$(2.28 \pm 0.011) \times 10^{-3}$	$(2.24 \pm 0.19) \times 10^{-3}$ [14]
Δm_{B_d}	$(3.36 \pm 0.02) \times 10^{-13}$ GeV	$(4.21 \pm 0.34) \times 10^{-13}$ GeV [15]
Δm_{B_s}	$(1.169 \pm 0.0014) \times 10^{-11}$ GeV	$(1.303 \pm 0.078) \times 10^{-11}$ GeV [15]
Δm_D	$(6.2^{+2.7}_{-2.8}) \times 10^{-15}$ GeV	$-$
$\mathrm{BR}(K_L \to \pi^0 \nu\overline{\nu})$	$< 2.60 \times 10^{-8}$ (90% CL)	$(2.49 \pm 0.39) \times 10^{-11}$ [16]
$\mathrm{BR}(K^+ \to \pi^+ \nu\overline{\nu})$	$(17 \pm 11) \times 10^{-11}$	$(7.8 \pm 0.8) \times 10^{-11}$ [17]
$\mathrm{BR}(B \to X_s \gamma)$	$(3.49 \pm 0.19) \times 10^{-4}$	$(3.36 \pm 0.23) \times 10^{-4}$ [18]
$A_{CP}(B \to X_s \gamma)$	$(1.5 \pm 2.0) \times 10^{-2}$	$(1.1 \pm 1.7) \times 10^{-2}$ [19]
$\Delta A_{CP}(B \to X_s \gamma)$	$(5.0 \pm 4.2) \times 10^{-2}$ [20]	$(0.0 \pm 0.0) \times 10^{-2}$ [19]
$\mathrm{BR}(B \to X_d \gamma)$	$(1.41 \pm 0.57) \times 10^{-5}$ [21, 22]	$(1.54^{+0.26}_{-0.31}) \times 10^{-5}$ [22]
$\mathrm{BR}(B_s \to \mu^+ \mu^-)$	$(2.9 \pm 0.7) \times 10^{-9}$ [23]	$(3.65 \pm 0.23) \times 10^{-9}$ [24]
$\mathrm{BR}(B_d \to \mu^+ \mu^-)$	$(3.6^{+1.6}_{-1.4}) \times 10^{-10}$ [23]	$(1.06 \pm 0.09) \times 10^{-10}$ [24]
$|d_n|$	$< 2.9 \times 10^{-26}$ e cm (90% CL)	$\sim 10^{-34}$ e cm [25]
$\mathrm{BR}(\mu \to e\gamma)$	$< 5.7 \times 10^{-13}$ (90% CL) [26]	~ 0
$\mathrm{BR}(\tau \to \mu\gamma)$	$< 4.4 \times 10^{-8}$ (90% CL)	~ 0
$\mathrm{BR}(\tau \to e\gamma)$	$< 3.3 \times 10^{-8}$ (90% CL)	~ 0

Table 2: Current experimental and theoretical values used in `FormFlavor`. Except where noted, the experimental values are taken from recent PDG or HFAG fits [27, 28]. No reliable theoretical prediction for Δm_D currently exists. Although literature on the subject exists, we do not use a theoretical prediction for Δm_K.

`FFObservables[`μ`,proclist,Heff]`	compute the flavor constraints for the processes contained in `proclist` (a subset of `$FFProcessList`) for the effective Hamiltonian `Heff` input at the scale μ

4.1 Structure of the observable modules

Within the observable module files, the first block of code provides some necessary linking to the model-dependent portion of the code.

- Each observable module has an associated amplitude file containing analytic expressions for the Wilson coefficients used by the module.[1] The amplitude files are generated by the `CalcAmps` package (see section 7.1 for details on this package), and they are stored within the {`model`}/`ObservableAmps` folder. (Within the {`model`}/`ObservableAmps`/`SubAmps` directory, the pre-compiled, built amplitudes are stored.) The observable module ensures that the necessary amplitude file is loaded by appending it to the global `FormFlavor` variable `$FFAmpFileList`.

- Each observable module inherits a name that is defined in the `CalcAmps` amplitude file. The observable module automatically finds the name from the amplitude file and appends it to `$FFProcessList`.

- Lastly, it links the process name with the observable function name (defined in the final block of the file) via the line

  > `ObservableFunction[TempProcessName]=`{*observable function name*}`;`

  where `TempProcessName` is a dummy variable storing the current process name. This information is used by the main `FFObservables` routine.

---

[1]Let us the stress the distinction between observables and amplitudes. An amplitude or process is an S-matrix calculation with a well-defined initial and final state, for instance $\bar{d}s \to \bar{s}d$, from which Wilson coefficients are extracted. An observable is something tied to a physical measurement, such as Δm_K or $\mathrm{BR}(B \to X_s \gamma)$. Importantly, the map between observables and amplitudes is neither one-to-one nor onto. A single amplitude can give rise to multiple observables, for instance, the KK mixing process contains the necessary information to evaluate both Δm_K and ϵ_K. Meanwhile, a single observable can depend on multiple processes due to mixing under RG evolution, for instance $\mathrm{BR}(B \to X_s \gamma)$ depends on both $b \to s\gamma$ and $b \to sg$.

Since each observable module can contain multiple observables (e.g. the K-K mixing observable module has both Δm_K and ϵ_K), the next block of code in the observable module steps through each observable, defines its `FormFlavor` internal name `name`, and sets the parameters defining its current experimental status:

> `FFObsClass[name]` observable's status ($= 0$, 1, or 2) (see Table 1)
>
> `FFExpValue[name]` observable's current experimental value
>
> `FFExpUnc[name]` observable's current experimental uncertainty
>
> `FFSMValue[name]` observable's current theoretical value (not used)
>
> `FFSMUnc[name]` observable's current theoretical uncertainty

The observable names are appended to the variable `$FFObsNameList` for later access, notably by the `FFConstraints` function.

Next within the code is an optional block that may contain global definitions, either functions or constant variables specific to the process, that are evaluated when `FormFlavor` is loaded, rather than each time the process is called.

The observable function is defined in the final block and contains information hardcoded from the literature on how to evaluate the observables from the high scale Wilson coefficients. The input to each observable function is the effective Hamiltonian (3.6), where C_i are the numerical values of the Wilson coefficients (in general complex). The observable functions also need the scale where the Wilson coefficients were evaluated as an input. The output of each observable function is a list of related observables (e.g. Δm_K and ϵ_K) and their numerical values.

For each observable function, there are two options that default to `True`: `IncludeSM`, which, when set to false, does not include the standard model contribution in the evaluation of the observable; and `QCDRG`, which when disabled stops the code from RG evolving the new physics contribution down to the lower scale, instead evaluating with the high scale Wilson coefficient value. In some cases, such as the neutron EDM, and Δm_D, no standard model portion is included at all, so `IncludeSM` does nothing. Similarly, many observables, such as $B_q \to \mu^+ \mu^-$ and $\mu \to e\gamma$ have no QCD RG evolution, so `QCDRG` does nothing. As these are primarily valuable for validation, it is recommended that users leave these settings set to `True`.

We will now turn to a detailed description of each observable function.

4.2 Meson Mixing

The matrix element relevant for meson mixing can be written as,

$$
\frac{\langle X|H_{eff}|\overline{X}\rangle}{2m_X} \equiv M_{X,12} - \frac{i}{2}\Gamma_{X,12}
$$
$$
\frac{\langle \overline{X}|H_{eff}|X\rangle}{2m_X} = M_{X,12}^* - \frac{i}{2}\Gamma_{X,12}^*
$$

(4.1)

where m_X is the averaged meson mass, H_{eff} is the effective Hamiltonian, and $M_{X,12}$ and $\Gamma_{X,12}$ are, respectively, the dispersive and absorptive parts of $\langle X|H_{eff}|\overline{X}\rangle$. Short-distance effects contribute only to the dispersive portion in terms of $|\Delta F| = 2$ operators, whereas long-distance effects consisting of intermediate on-shell and off-shell particle exchanges, i.e., $(|\Delta F| = 1)^2$, can contribute to both parts. The relevant $\Delta F = 2$ Wilson operators are shown in (3.3)–(3.5) with $f_1 = f_3 = q_1$, and $f_2 = f_4 = q_2$, for $(q_1, q_2) = (s,d), (c,u), (b,d), (b,s)$ for K, D, B_d and B_s respectively. The short-distance portion of $M_{X,12}$ (which is all that is relevant for the B_q systems and for new physics) is given by [29]

$$
M_{X,12}^{SD} = \frac{m_X f_X^2}{24}\left(8B_V^{LL}(C_V^{LL} + C_V^{RR}) - R_X\left[4B_V^{LR}C_V^{LR} - 6B_S^{LR}C_S^{LR}\right] \right.
$$
$$
\left. - R_X\left[5B_S^{LL}\left(C_S^{LL} + C_S^{RR}\right) + 12B_T^{LL}\left(C_T^{LL} + C_T^{RR}\right)\right]\right)
$$

(4.2)

where,

$$
R_X \equiv \left(\frac{m_X}{m_{q_1} + m_{q_2}}\right)^2,
$$

(4.3)

and the non-perturbative B-parameters have been computed on the lattice. For all mesons, only C_V^{LL} is nonzero in the Standard Model. The parameters relevant for meson mixing are summarized in Table 3.

In terms of these quantities, the mass splitting mixing parameter can be expressed as [27],

$$
\Delta m_X = 2\,\mathrm{Re}\left[\left(M_{X,12} - \frac{i}{2}\Gamma_{X,12}\right)\sqrt{\frac{M_{X,12}^* - \frac{i}{2}\Gamma_{X,12}^*}{M_{X,12} - \frac{i}{2}\Gamma_{X,12}}}\right].
$$

(4.4)

For both $X = B_d$ and B_s, where in the standard model the short-distance contributions are dominant, and it is known experimentally that $\Gamma_{X,12} \ll M_{X,12}$, (4.4) can be well approximated by,

$$
\Delta m_X \approx 2\,|M_{X,12}| = 2\left|M_{X,12}^{SM} + M_{X,12}^{NP}\right|.
$$

(4.5)

Meson	m_X(GeV)	f_X(GeV)	R_X	B_V^{LL}	B_V^{LR}	B_S^{LR}	B_S^{LL}	B_T^{LL}	$C_{V,SM}^{LL}\vert_{\mu=m_b}$(GeV^{-2})
K	0.4976	0.160	24.3	0.56	0.85	1.08	0.62	0.43	$-$
D	1.8645	0.209	3.20	0.76	0.97	0.95	0.64	0.39	$-$
B_d	5.2796	0.188	1.65	0.913	1.838	1.145	0.761	0.555	$(2.34 - 2.20i) \times 10^{-12}$
B_s	5.3668	0.226	1.65	0.952	1.799	1.125	0.806	0.610	$(6.96 - 0.26i) \times 10^{-11}$

Table 3: Parameters of mesons relevant for the `FormFlavor` meson mixing observables. For both the K and D system, f_X and $B_{V,X}^{LL}$ are taken from the FLAG review [7], while f_{B_d} and f_{B_s} come from [30] and [31], respectively. The other non-perturbative B-parameters were collected from several sources: for Δm_K from [32] (at $\mu = 2$ GeV), Δm_D from [33] (at $\mu = 3$ GeV, rescaled for a common R_D, and converted to our basis using $B_T^{LL} = \frac{5}{3}B_2 - \frac{2}{3}B_3$), and for B_s and B_d from [15] (at $\mu = \overline{m_b} = 4.18$ GeV, rescaled to match the form in (4.2), and converted to our basis). $C_{V,SM}^{LL}\vert_{\mu=m_b}$ are the `FormFlavor` values in the CKM basis of the PDG [27].

For $X = K$ and D, long-distance contributions to $M_{X,12}$ are appreciable and even assumed to dominate in the case of Δm_D. However, experimentally one has that $M_{X,12}/\Gamma_{X,12}$ is approximately real, and both $M_{X,12}$ and $\Gamma_{X,12}$ are predicted to be approximately real in the CKM convention used in `FormFlavor` (where the CPV phase is primarily in V_{td} and V_{ub}). Using that information, one can approximate (4.4) as,

$$\Delta m_X \approx 2\,\mathrm{Re}\,M_{X,12} = \Delta m_{X,SM} + 2\,\mathrm{Re}\,M_{X,12}^{NP} \tag{4.6}$$

In the case of Δm_D, this approximation is not great, however, relative to the enormous theoretical uncertainties in Δm_D, this treatment is sufficient to determine regions where new physics contributions overwhelm the experimental measurement.

The QCD RG evolution of the Wilson coefficients in (4.2) is controlled by the function `DeltaF2RG`. This RG evolution [29] is for Wilson coefficients in the NDR-$\overline{\mathrm{MS}}$ with the BMU evanescent operator scheme [34].

`DeltaF2RG[`μ_L`,`μ_H`,list]`   RG evolve $\vert\Delta F\vert = 2$ Wilson coefficients from high scale, μ_H, to low scale, μ_L, with input and output in the form `list=` $\{C_V^{LL}, C_V^{RR}, C_V^{LR}, C_S^{LR}, C_S^{LL}, C_T^{LL}, C_S^{RR}, C_T^{RR}\}$

The indirect CP-violation in $K^0 - \overline{K}^0$ mixing, ϵ_K, can be expressed as [35]:

$$\epsilon_K \equiv \frac{\tilde{\kappa}_\epsilon \,\mathrm{Im}\,M_{X,12}}{\sqrt{2}\Delta m_{K,exp}} = \epsilon_{K,SM} + \frac{\tilde{\kappa}_\epsilon \,\mathrm{Im}\,M_{X,12}^{NP}}{\sqrt{2}\Delta m_{K,exp}} \tag{4.7}$$

where $\tilde{\kappa}_\epsilon = 0.94 \pm 0.02$ accounts for small long-distance corrections in $\mathrm{Im}\,\Gamma_{12}$ and $\mathrm{Im}\,M_{12}$ from $\left(H_{|\Delta F|=1}\right)^2$ contributions [36]. Although we refer to this here and in the manual and code as ϵ_K, many references in the literature include a factor of the "superweak phase" which has been experimentally determined to be very close to $\pi/4$ [27]. New short distance physics will not change the phase of ϵ_K appreciably, so for the treatment of this observable, it makes sense to remove this phase, as we have done in (4.7).

The SM contribution for $\Delta m_{B_d,B_s}$ is hardcoded and combined as in (4.6); whereas no SM contribution is used for Δm_K and Δm_D due to the large uncertainties [37]. The following `FormFlavor` functions evaluate these meson mixing observables:

`KKmixing[`μ`,Heff]`	evaluate both Δm_K and ϵ_K with Wilson coefficients input at the scale μ
`DDmixing[`μ`,Heff]`	evaluate Δm_D with Wilson coefficients at the scale μ
`BdBdmixing[`μ`,Heff]`	evaluate Δm_{B_d} with Wilson coefficients at the scale μ
`BsBsmixing[`μ`,Heff]`	evaluate Δm_{B_s} with Wilson coefficients at the scale μ

4.3 $b \to q\gamma$

The $b \to s\gamma$ and $b \to d\gamma$ observables come primarily from dimension 5 effective Hamiltonian (3.1)–(3.2) with $f_2 = b$ and $f_1 = s$ or d. The leading order branching ratio can be written as,

$$[\mathrm{BR}(b \to q\gamma)]_{LO} = c_\gamma v^2 \left(\left|C_A^R\right|^2 + \left|C_A^L\right|^2 \right) \tag{4.8}$$

where [38, 39]

$$
\begin{aligned}
c_\gamma &= \left(8\pi^2\right)^2 \frac{6}{\pi} \frac{\mathrm{BR}(b \to X_c e\nu)_{\mathrm{EXP}}}{\Phi \left|V_{cb}\right|^2} \frac{v^2}{m_b^2} \alpha_{\mathrm{EM}} = 3.3 \times 10^7 \\
\Phi &= \left|\frac{V_{ub}}{V_{cb}}\right|^2 \frac{\mathrm{BR}(b \to X_c e\nu)}{\mathrm{BR}(b \to X_u e\nu)} = 0.569.
\end{aligned}
\tag{4.9}
$$

The branching ratio into charm final states is used to remove the hadronic matrix element, and the Φ factor accounts for the nontrivial phase space differences between the two decays (due mostly to the charm quark mass). However, this leading order treatment is insufficient for an accurate SM prediction of $b \to s\gamma$. Currently, the uncertainty from the combined experimental world average and from the most precise theory determination are very close in size. As $\mathrm{BR}(B \to X_s\gamma)$ is often one of the most constraining observables for BSM physics, an accurate SM prediction is essential. In this section, we explain how `FormFlavor` handles the various $b \to q\gamma$ observables. Due to the expansive

$C_{eff}(\mu_b)$	1	2	3	4	5	6	7	8
$C_{eff}^{(0)}$	-0.8999	1.073	-0.0151	-0.1393	0.0014	0.0032	-0.3848	-0.1776
$C_{eff}^{(1)}$	14.94	-2.210	0.0842	-0.5902	-0.0207	-0.0069	2.087	-0.6311
$C_{eff}^{(2)}$							18.86	

Table 4: The standard model effective Wilson coefficients relevant for $\mathrm{BR}(B \to X_s\gamma)$ evaluated at $\mu_b = 2$ GeV. At second order in α_s, only $C_{eff,7}^{(2)}$ is necessary.

literature on the subject, we will not replicate many expressions and refer interested users to the primary references.

Both $\mathrm{BR}(B \to X_s\gamma)$ and $\mathrm{BR}(B \to X_d\gamma)$ can be expressed as,

$$\mathrm{BR}(B \to X_q\gamma)_{E>E_0} = c_\gamma \frac{\left|V_{tq}^* V_{tb}\right|^2 m_b^2}{(8\pi^2)^2 v^2}\bigg(P(E_0) + N(E_0)\bigg), \qquad (4.10)$$

where $P(E_0)$ and $N(E_0)$ are the perturbative and small, non-perturbative contributions, respectively. The leading order expression for $P(E_0)$ can be easily extracted from (4.8)–(4.10). As $\mathrm{BR}(b \to s\gamma)$ and $\mathrm{BR}(b \to d\gamma)$ are known experimentally to very different levels of precision, in `FormFlavor` we give a slightly different treatment for each of these observables. We will first discuss the $b \to s\gamma$ process and the three associated observables, before discussing the treatment of $b \to d\gamma$.

To NNLO order, the perturbative portion of $\mathrm{BR}(B \to X_s\gamma)$ can be written [40],

$$P(E_0) = \sum_{i,j=1}^{8} C_{eff,i}(\mu_b) K_{ij}(\mu_b, E_0) C_{eff,j}^*(\mu_b) \qquad (4.11)$$

where $K_{ij}(\mu_b, E_0)$ is a Hermitian matrix, and the effective Wilson coefficients $C_{eff,i}$ are evaluated at the scale μ_b. The values of $C_{eff,i}^{(1,2)}$ and $K_{ij}^{(1,2)}$ have been determined through decades of work [39–54]. For details of the definitions of $C_{eff,i}$, see [39] and references within. For our purposes, it suffices to know that $C_{eff,7} \sim C_A^R$ and $C_{eff,8} \sim C_G^{R,}$[2] whereas the others are coefficients of four Fermi or $(\bar{b}s)\{current\}$ operators. Both $C_{eff,i}$ and $K_{ij}(\mu_b, E_0)$ can be expanded perturbatively in powers of $\tilde{\alpha}_s = \frac{\alpha_s(\mu_b)}{4\pi}$. The values for $C_{eff,i}$ in the SM up to second order in $\tilde{\alpha}_s$ are given in table 4; while $K^{(0)} = \delta_{i7}\delta_{j7}$ and

---

[2]In actuality, $C_{eff,7} = y_0 C_A^R + \sum_i^6 y_i C_i$ and $C_{eff,8} = z_0 C_G^R + \sum_i^6 z_i C_i$ [40]. These details are not important for the new physics, as the leading order contributions to C_{1-6} are tree level, while $C_{7,8}$ leading order contributions enter at one-loop level, so new physics added at loop level in `FormFlavor` only contributes to C_7 and C_8. That $C_{eff,7}$ etc are *linear* in the other C_i Wilson coefficients means that the SM and new physics contributions can be completely decoupled.

the values for $K_{ij}^{(1,2)}$ (at the scale $\mu_b = 2$ GeV and $E_0 = 1.6$ GeV) are [40]:

$$K^{(1)} = \left(\begin{array}{cccccc|cc}
0.0029 & -0.017 & 0.0005 & 0.0042 & 0.0054 & -0.062 & 0.091 & -0.0017 \\
-0.017 & 0.104 & -0.0029 & -0.026 & -0.032 & 0.37 & -0.55 & 0.010 \\
0.0005 & -0.0029 & 0.029 & -0.0048 & 0.32 & -0.028 & 8.37 & -0.055 \\
0.0042 & -0.026 & -0.0048 & 0.0008 & -0.053 & 0.0047 & -1.73 & -0.204 \\
0.0054 & -0.032 & 0.32 & -0.053 & 3.55 & -0.084 & 123.5 & -0.838 \\
-0.062 & 0.37 & -0.028 & 0.0047 & -0.084 & 1.81 & 18.36 & -1.93 \\
\hline
0.091 & -0.55 & 8.37 & -1.73 & 123.5 & 18.36 & 5.62 & -0.507 \\
-0.0017 & 0.010 & -0.055 & -0.204 & -0.838 & -1.93 & -0.507 & 0.452
\end{array}\right) \quad (4.12)$$

and [39]

$$K^{(2)} = \left(\begin{array}{cccccc|cc}
0.11 & -0.69 & 0 & 0 & 0 & 0 & 9.11 & 0.22 \\
-0.69 & 4.12 & 0 & 0 & 0 & 0 & -8.86 & -1.29 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
9.11 & -8.86 & 0 & 0 & 0 & 0 & -37.32 & -13.41 \\
0.22 & -1.29 & 0 & 0 & 0 & 0 & -13.41 & 22.32
\end{array}\right). \quad (4.13)$$

The terms in $K^{(2)}$ aligned with $C_{eff,3-6}$ are believed to be very small, and thus approximated to be 0 at this level. The small non-perturbative correction of $N(E_0) = 3.82 \times 10^{-7}$ [55–57] is included in `FormFlavor`, but has almost no effect on the net BR.[3]

In order to include new physics in the $b \to s\gamma$ branching ratio, the NP contribution is RG evolved from the high scale, down to the low scale using the `FormFlavor` function `btosgammaRG`.

`btosgammaRG[`μ_L`,`μ_H`,list]`	RG evolve dimension 5 Wilson coefficients from high scale, μ_H, to low scale, μ_L, with input and output in the form `list=` $\{C_A^R, C_G^R, C_A^L, C_G^L\}$

This evolution only includes mixing between C_7 (C_A^R) and C_8 (C_G^R) and between $\tilde{C}_7$ (C_A^L) and $\tilde{C}_8$ (C_G^L). Importantly, the RG evolution of the dimension-6 Wilson coefficients, C_{1-6}, is not affected by the dimension-5 coefficients, $C_{7,8}$; and C_{1-6} contribute to the RG evolution of $C_{7,8}$ *linearly*. Since we do not include new physics in the dimension-6

---

[3]We are extremely grateful to M. Misiak for providing code to compute the standard model $C_{eff}^{(i)}$, $K_{ij}^{(1,2)}$ and $N(E_0)$ values used in `FormFlavor`.

operators (which would contribute to $b \to q\gamma$ at a higher order), the effects of C_{1-6}, including their mixing through the RG, have already been included in the SM portion.

It was realized long ago that direct CP asymmetries observed in $b \to s\gamma$ could be a sign of new physics [58], and some early estimates predicted very small uncertainty ($\pm 0.5\%$) in the SM [59]. However, initially neglected long-distance contributions to the asymmetry greatly increase the uncertainty [19]. Still, a model predicting a substantially larger than measured value of $A_{CP}(B \to X_s\gamma)$ could be conclusively ruled out. The direct CP symmetry in B decays is defined as:

$$A_{CP}(B \to X_s\gamma) \equiv \frac{\Gamma\left(\overline{B} \to X_s\gamma\right) - \Gamma\left(B \to X_{\overline{s}}\gamma\right)}{\Gamma\left(\overline{B} \to X_s\gamma\right) + \Gamma\left(B \to X_{\overline{s}}\gamma\right)}, \tag{4.14}$$

this can be expressed as [19],

$$A_{CP}\big|_{E_\gamma > E_0} = A_{CP}^{dir}\big|_{E_\gamma > E_0} + A_{CP}^{res}\big|_{E_\gamma > E_0} \tag{4.15}$$

where A_{CP}^{dir} and A_{CP}^{res} are the direct CPV contribution and the resolved contribution that accounts for the hadronic substructure of the photon [60, 56]. The direct contribution can be approximated with [61, 62, 19],

$$A_{CP}^{dir}\big|_{E_\gamma > E_0} \approx \alpha_s \left(\frac{40}{81} \operatorname{Im}\left[\frac{C_2}{C_7}\right] - \frac{4}{9} \operatorname{Im}\left[\frac{C_8}{C_7}\right] - \frac{40\Lambda_c}{9m_b} \operatorname{Im}\left[(1+\epsilon_s)\frac{C_2}{C_7}\right]\right) \tag{4.16}$$

where $\epsilon_s = \frac{V_{ub}V_{us}^*}{V_{tb}V_{ts}^*}$, corrections of $\mathcal{O}\left(\frac{\Lambda_{QCD}^2}{m_b^2}\right)$ have been dropped, and

$$\Lambda_c \equiv \frac{m_c^2}{m_b}\left(1 - \frac{2}{5}\ln\frac{m_b}{m_c} + \frac{4}{5}\ln^2\frac{m_b}{m_c} - \frac{\pi^2}{15}\right). \tag{4.17}$$

The resolved photon contribution can be expressed as [19],

$$A_{CP}^{res}\big|_{E_\gamma > E_0} \approx \frac{\pi}{m_b}\left(\operatorname{Im}\left[(1+\epsilon_s)\frac{C_2}{C_7}\right]\tilde{\Lambda}_{27}^c - \operatorname{Im}\left[\epsilon_s\frac{C_2}{C_7}\right]\tilde{\Lambda}_{27}^u + 4\pi\alpha_s\operatorname{Im}\left[\frac{C_8}{C_7}\right]\tilde{\Lambda}_{78}^{\overline{B}}\right). \tag{4.18}$$

The hadronic $\tilde{\Lambda}$ parameters are extremely uncertain, but will be estimated to their central values of $\tilde{\Lambda}_{27}^c = 0.001$ GeV, $\tilde{\Lambda}_{27}^u = 0.1$ GeV, and $\tilde{\Lambda}_{78}^{\overline{B}}$ additionally depends linearly on the charge of the spectator quark. We will use that BR$[\Upsilon(4S) \to B^+B^- (B^0\overline{B}^0)] = 0.513\,(0.487)$ to write $\tilde{\Lambda}_{78}^{\overline{B}} = -(2\times 0.513 - 0.487)/3\tilde{\Lambda}_{78} = -0.18\tilde{\Lambda}_{78}$, with $\tilde{\Lambda}_{78} \approx 0.1$ GeV. In `FormFlavor`, we apply an SM theory uncertainty of $\pm 1.7\%$ [19] to this observable.

It was also pointed out in reference [19] that a resolved CP asymmetry difference between charged and neutral meson decays can be a sensitive probe of new physics. This

quantity,

$$\Delta A_{CP}(B \to X_s\gamma)\big|_{E_\gamma > E_0} \equiv \left(A_{CP}(B \to X_s^-\gamma) - A_{CP}(B \to X_s^0\gamma)\right)\big|_{E_\gamma > E_0}$$

$$\approx 4\pi^2 \alpha_s \frac{\tilde{\Lambda}_{78}}{m_b} \operatorname{Im}\left[\frac{C_G^R}{C_A^R}\right] \tag{4.19}$$

where the hadronic parameter $\tilde{\Lambda}_{78}$ is the dominant source of uncertainty, finishes in the standard model and serves as an additional constraint on new physics. In `FormFlavor`, we follow [19] and use central value $\tilde{\Lambda}_{78} = 0.1$ GeV, although more recent QCD sum rule calculations [63] suggest a larger value such as 0.3 GeV might be more accurate. For excluding parameter points, a smaller value is more conservative.

The branching ratio and both CP asymmetries in $b \to s\gamma$ are evaluated with the `FormFlavor` function `btosgamma`.

`btosgamma[`μ`,Heff]`	evaluate $\mathrm{BR}(B \to X_s\gamma)$, $A_{CP}(B \to X_s\gamma)$, and $\Delta A_{CP}(B \to X_s\gamma)$ with Wilson coefficients input at the scale μ

Finally, for $\mathrm{BR}(B \to X_d\gamma)$, which is currently known to a much lower experimental precision, we follow the treatment in [59] to derive the SM branching ratio. Unlike in the case of $\mathrm{BR}(B \to X_s\gamma)$ this result uses Wilson coefficients evaluated at the scale m_t as input. As the treatment uses Wilson coefficients evaluated at m_t, disabling `QCDRG` only removes the running of new physics Wilson coefficients from m_{SUSY} to m_t. We can express $P(E_0)$ in (4.10) as [59],

$$
\begin{aligned}
P(E_0) = {} & a_0 + a_{AA}\left(|R_A|^2 + \left|\tilde{R}_A\right|^2\right) + a_{GG}\left(|R_G|^2 + \left|\tilde{R}_G\right|^2\right) + a_A^r \operatorname{Re} R_A + a_A^i \operatorname{Im} R_A \\
& + a_G^r \operatorname{Re} R_G + a_G^i \operatorname{Im} R_G + a_{\epsilon\epsilon}|\epsilon_d|^2 + a_\epsilon^r \operatorname{Re} \epsilon_d + a_\epsilon^i \operatorname{Im} \epsilon_d \\
& + a_{A\epsilon}^r \operatorname{Re}\left(R_A \epsilon_d^*\right) + a_{A\epsilon}^i \operatorname{Im}\left(R_A \epsilon_d^*\right) + a_{G\epsilon}^r \operatorname{Re}\left(R_G \epsilon_d^*\right) + a_{G\epsilon}^i \operatorname{Im}\left(R_G \epsilon_d^*\right) \\
& + a_{GA}^r \operatorname{Re}\left(R_G R_A^* + \tilde{R}_G \tilde{R}_A^*\right) + a_{GA}^i \operatorname{Im}\left(R_G R_A^* + \tilde{R}_G \tilde{R}_A^*\right)
\end{aligned} \tag{4.20}
$$

where,

$$\epsilon_d = \frac{V_{ub}V_{ud}^*}{V_{tb}V_{td}^*}, \qquad R_X = \frac{C_X^R(\mu = m_t)}{C_{X,SM}^R(\mu = m_t)}, \qquad \tilde{R}_X = \frac{C_X^L(\mu = m_t)}{C_{X,SM}^R(\mu = m_t)}, \tag{4.21}$$

and the a coefficients are given in table 5.

$\mathrm{BR}(B \to X_d\gamma)$ is evaluated with the `FormFlavor` function `btodgamma`.

`btodgamma[`μ`,Heff]`	evaluate $\mathrm{BR}(B \to X_d\gamma)$ with Wilson coefficients input at the scale μ

a_0	7.8221	a_{AA}	0.8161	a_A^r	4.8802	a_A^i	0.3546
a_{GG}	0.0197	a_G^r	0.5680	a_G^i	-0.0987	$a_{\epsilon\epsilon}$	0.4384
a_ϵ^r	-1.6981	a_ϵ^i	2.4997	a_{GA}^r	0.1923	a_{GA}^i	-0.0487
$a_{\epsilon A}^r$	-1.0940	$a_{\epsilon A}^i$	-1.0447	$a_{\epsilon G}^r$	-0.0819	$a_{\epsilon G}^i$	-0.0779

Table 5: Numerical values for the coefficients in $P(E_0)$ (4.20) relevant for $\mathrm{BR}(B \to X_d\gamma)$. Values shown are for $E_0 = 1.6$ GeV and $m_c/m_b = 0.23$ [59].

4.4 $K \to \pi\nu\nu$

Rare $K \to \pi\nu\nu$ decays can be expressed in terms of the Wilson operators $C_{V,\ell_1\ell_2}^{ML}$ where we have $f_1 = d$, $f_2 = s$, $f_3 = \nu_{\ell_1}$, $f_4 = \nu_{\ell_2}$ in (3.4). The branching ratios for charged and neutral $K \to \pi\nu\bar{\nu}$ can be written [64],

$$
\begin{aligned}
\mathrm{BR}(K^\pm \to \pi^\pm \nu\bar{\nu}) &= \frac{c_+}{3} v^4 (1 + \Delta_{EM}) \sum_{\ell_1,\ell_2=e,\mu,\tau} \left| C_{V,\ell_1\ell_2}^{LL} + C_{V,\ell_1\ell_2}^{RL} \right|^2 \\
\mathrm{BR}(K_L \to \pi^0 \nu\bar{\nu}) &= \frac{c_0}{3} v^4 (1 + \delta_\epsilon) \sum_{\ell_1,\ell_2=e,\mu,\tau} \left(\mathrm{Im}\left[C_{V,\ell_1\ell_2}^{LL} + C_{V,\ell_1\ell_2}^{RL} \right] \right)^2
\end{aligned}
\tag{4.22}
$$

where $v = 246\,\mathrm{GeV}$, $\delta_\epsilon \approx -0.011$ [65] accounts for the indirect CP violation in mixing, $\Delta_{EM} = -0.003$ [64] accounts for electromagnetic effects, and

$$
\begin{aligned}
c_+ &= \frac{3\,r_{K^+}}{2\,|V_{us}|^2} \mathrm{BR}(K^\pm \to \pi^0 e^\pm \nu) = 1.35 \\
c_0 &= \frac{3\,r_{K^0}}{2\,|V_{us}|^2} \frac{\Gamma(K^\pm)}{\Gamma(K_L)} \mathrm{BR}(K^\pm \to \pi^0 e^\pm \nu) = 5.84.
\end{aligned}
\tag{4.23}
$$

Here, the branching ratio (0.0507) has been included to remove dependence on the hadronic matrix element, and factors $r_{K^+} = 0.901$ and $r_{K^0} = 0.944$ contains electroweak corrections and isospin violating quark mass effects that were computed in [66]. The RG evolution of the relevant $C_{V,\ell_1\ell_2}^{ML}$ operators is negligible [1].

In the standard model, lepton number is a good symmetry, so all mixed flavor operators ($\ell_1 \neq \ell_2$) vanish. Meanwhile, the top loop contributions have no sensitivity to different lepton generations, whereas the charm loop contributions are the same for e and μ, but different for τ. Thus, the SM contribution can be expressed as,

$$
(C_{V,\ell\ell}^{LL})_{SM} = \frac{\alpha_2}{\pi v^2} \left(\lambda_c X_c^\ell + \lambda_t X_t \right)
\tag{4.24}
$$

where $\lambda_i = V_{is}^* V_{id}$, $X_t = 1.481$ [64], $X_c^e = X_c^\mu = 1.055 \times 10^{-3}$, and $X_c^\tau = 7.01 \times 10^{-4}$ [1].

As for the new physics contribution, although lepton flavor violation is treated in `FormFlavor`, the effect of lepton flavor violation on this observable in most models should be small, so the current implementation of $K \to \pi\nu\nu$ does not treat this possibility. We will simplify expressions by further assuming lepton flavor universality, i.e. $C_{V,NP,\ell_1\ell_2}^{XY} \equiv C_{V,NP}^{XY}\delta_{\ell_1\ell_2}$. We can then make a simplifying approximation to equation (4.22) yielding,

$$
\begin{aligned}
\mathrm{BR}(K^{\pm} \to \pi^{\pm}\nu\bar{\nu}) &\approx c_+(1 + \Delta_{EM})v^4 \left|C_V^{LL} + C_V^{RL}\right|^2 \\
\mathrm{BR}(K_L \to \pi^0\nu\bar{\nu}) &\approx c_0 \left(1 + \delta_\epsilon\right) v^4 \operatorname{Im}\!\left[C_V^{LL} + C_V^{RL}\right]^2
\end{aligned}
\tag{4.25}
$$

where $C_V^{XY} = C_{V,NP}^{XY} + C_{V,SM}^{XY}$, with

$$
\begin{aligned}
C_{V,SM}^{RL} &= 0 \\
C_{V,SM}^{LL} &= \frac{\alpha_2}{\pi v^2}\left(\lambda_c\left(P_c + \delta P_{c,u}\right) + \lambda_t X_t\right) = (-12.5 + 3.7i) \times 10^{-11}\,\mathrm{GeV}^{-2},
\end{aligned}
\tag{4.26}
$$

and P_c is now the charm loop contribution averaged over the three neutrino flavors,

$$
P_c = \left(\frac{2}{3}X_{NL}^e + \frac{1}{3}X_{NL}^\tau\right) \sim 9.37 \times 10^{-4}.
\tag{4.27}
$$

Using P_c simplifies the expression at the cost of reducing the standard model charm contribution by roughly 3%. This corresponds to a 0.3% decrease of the overall SM contribution, which, compared to the current theoretical uncertainty, is completely negligible. Importantly, interference between standard model and new physics contributions are properly captured with this simplification. Contained in $\delta P_{c,u}$ are long-distance and dimension-8 contributions which increase the effective P_c by about 10% [67],

$$
\delta P_{c,u} = \frac{\pi^2 f_\pi^2}{m_W^2}\left(\frac{4|G_8|}{\sqrt{2}G_F} - \frac{4}{3}\right) \sim (1.08 \pm 0.54) \times 10^{-4}.
\tag{4.28}
$$

For $K_L \to \pi^0\nu\bar{\nu}$, the $\lambda_t X_t$ portion of the standard model dominates as $\lambda_c P_c$ is approximately real.

We note that the `FormFlavor` values are above the "official" SM predictions shown in table 2. This is largely due to the change in central value of $|V_{cb}| = A\lambda^2$ from $0.0406 \to 0.0418$ (between [17] and [68, 69]). As $\lambda_t^2 \propto A^4$ and λ has not changed appreciably, this is an $\mathcal{O}(10\%)$ enhancement to both $K \to \pi\nu\nu$ branching ratios. This shift is of course entirely consistent with [17], where it was clearly presented that the dominant uncertainty was in $|V_{cb}|$. This is further supported by the larger values claimed in [64].

Both $K \to \pi\nu\nu$ decays are evaluated with the `FormFlavor` function `Ktopinunu`.

`Ktopinunu[`μ`,Heff]`	evaluate both $\mathrm{BR}(K_L \to \pi^0\nu\bar{\nu})$ and $\mathrm{BR}(K^{\pm} \to \pi^{\pm}\nu\bar{\nu})$ with Wilson coefficients input at the scale μ

4.5 $B_q \to \mu^+ \mu^-$

The $B_q \to \mu^+ \mu^-$ observables can be expressed in terms of operators within (3.3)–(3.4) for $f_1 = b$, $f_2 = s, d$ and $f_3 = f_4 = \mu$. The branching ratio for $B_{s,d} \to \mu^+ \mu^-$ can be written as [70, 71]:

$$\mathrm{BR}(B_i \to \mu^+ \mu^-) = X_i \left[\left(1 - \frac{4m_\mu^2}{m_{B_i}^2}\right) |F_S^{(i)}|^2 + |F_P^{(i)} + F_A^{(i)}|^2 \right] \qquad (4.29)$$

where

$$X_i = \frac{f_{B_i}^2}{128\pi\, m_{B_i} \Gamma_{B_i,H}} \sqrt{1 - \frac{4m_\mu^2}{m_{B_i}^2}} \implies X_s = 5.76 \times 10^7 \text{ and } X_d = 3.84 \times 10^7, \qquad (4.30)$$

$$\begin{aligned}
F_S^{(i)} &= \frac{m_{B_i}^3}{m_b + m_i}(C_S^{LL} + C_S^{LR} - C_S^{RR} - C_S^{RL}), \\
F_P^{(i)} &= \frac{m_{B_i}^3}{m_b + m_i}(-C_S^{LL} + C_S^{LR} - C_S^{RR} + C_S^{RL}), \\
F_A^{(i)} &= 2m_{B_i} m_\mu (C_V^{LL} - C_V^{LR} + C_V^{RR} - C_V^{RL}).
\end{aligned} \qquad (4.31)$$

The width of the heavier B meson is used in the expression [72] $\Gamma_{B_d,H} \approx \Gamma_{B_d} \equiv \tau_{B_d}^{-1} = 4.33 \times 10^{-13}$ GeV and $\Gamma_{B_s,H} \equiv \tau_{B_s,H}^{-1} = 4.104 \times 10^{-13}$ GeV. m_b in these expressions is $m_b(m_b)$ the $\overline{\mathrm{DR}}$ renormalization scheme [71]. The values of other parameters can be taken from table 3. The standard model contribution to $B_i \to \mu^+ \mu^-$ has been evaluated to QCD NNLO order [73] and EW NLO order [74] to be [24]:

$$F_{A,SM}^{(i)} = 0.4690 \times \frac{4\alpha_2 V_{tb}^* V_{ti}}{\pi} \frac{m_\mu m_{B_i}}{v^2}; \; F_{S,SM}^{(i)} = F_{P,SM}^{(i)} = 0 \qquad (4.32)$$

which translates to $F_{A,SM}^{(d)} = (1.5 - 0.6i) \times 10^{-9}$ and $F_{A,SM}^{(s)} = (-7.8 - 0.1i) \times 10^{-9}$. The form factors in (4.31) undergo no QCD running [71].

The observables $\mathrm{BR}(B_s \to \mu^+ \mu^-)$ and $\mathrm{BR}(B_d \to \mu^+ \mu^-)$ are evaluated with the `FormFlavor` functions `Bstomumu` and `Bdtomumu`. We note that `FormFlavor`'s evaluation and compilation of the Wilson coefficients for $B_s \to \mu^+ \mu^-$ and $B_d \to \mu^+ \mu^-$ is one of the largest bottlenecks in the program. For more detailed discussions of program speed, see sections 5.1 – 5.2.

`Bstomumu[`μ`,Heff]`	evaluate $\mathrm{BR}(B_s \to \mu^+ \mu^-)$ for Wilson coefficients at μ
`Bdtomumu[`μ`,Heff]`	evaluate $\mathrm{BR}(B_d \to \mu^+ \mu^-)$ for Wilson coefficients at μ

4.6 Neutron EDM

The electric dipole moment of the neutron can be expressed in terms of the dimension 5 operators (3.1)–(3.2) with $f_1 = f_2 = d$ or $f_1 = f_2 = u$. The expression for the neutron EDM (at 1-loop from quarks) can be written as [75],

$$d_n = 0.12^{+0.09}_{-0.06} \, e \left[(4d^e_d - d^e_u) + 1.5(2d^c_d + d^c_u) \right], \tag{4.33}$$

where $e \sim 0.30$ is the electromagnetic coupling, and the electromagnetic d^e_q and chromomagnetic d^c_q portions can be written in terms of Wilson coefficients as,

$$\begin{aligned} d^e_q &= 2 \operatorname{Im}\left[C^R_{A,q} - C^L_{A,q} \right] \\ d^c_q &= 2 \operatorname{Im}\left[C^R_{G,q} - C^L_{G,q} \right], \end{aligned} \tag{4.34}$$

where the factor of two is correcting for a relative normalization between `FormFlavor` operators and the operators in reference [75].

The RG evolution of the operators relevant for the neutron EDM [1, 76] is performed in `FormFlavor` with the function `EDMRG`. As the Wilson operators (3.1)–(3.2) of interest in `FormFlavor` and [75] do not explicitly contain the quark mass as those in [1, 76] do, the RG evolved operators must further be scaled by the factor $\left(\frac{\alpha_s(\mu_H)}{\alpha_s(\mu_L)} \right)^{4/\beta_0}$ to account for the running of m_q, where the QCD beta function is $\beta_0 = 11 - 2n_F/3$ for n_F light quark flavors.

`EDMRG[`μ_L`,`μ_H`,dlist,ulist]`	RG evolve the EDM Wilson coefficients from the high scale, μ_H, to the low scale, μ_L, with the output in the form {`dlist`,`ulist`} where for both input and output `qlist`= $\{C^R_{A,q}, C^R_{G,q}, C^L_{A,q}, C^L_{G,q}\}$

We stress that the uncertainty on the coefficient in (4.33) is $\mathcal{O}(1)$. Moreover, while this coefficient's value was determined from QCD sum rules, results in χPT [77, 78], on the lattice [79–84], and even other QCD sum rules evaluations [85], can vary drastically, so that the uncertainty on that uncertainty is also $\mathcal{O}(1)$ or more. So any theoretical prediction of the neutron EDM, from `FormFlavor` or otherwise, should be viewed as an order of magnitude estimate at best.

Further limitations of the `FormFlavor` prediction include: we neglect potentially important contributions from the Weinberg operator [86], $\mathcal{O}_W = g_3 f^{abc} \epsilon^{\mu\nu\rho\sigma} G^a_{\mu\lambda} G^{b,\lambda}_\nu G^c_{\rho\sigma}$; and `FormFlavor` is limited to one-loop contributions to the neutron EDM, while it is well-known that sometimes two-loop contributions can dominate [87].

`neutronEDM[`μ`,Heff]`	estimate neutron EDM (in units of e·cm) for Wilson coefficients at μ

4.7 $\ell_i \to \ell_j \gamma$

The leading contribution to radiative lepton decays come from the dimension five photon operators, $\mathcal{O}_A^M$ (3.1), with $f_1 = \ell_i$ and $f_2 = \ell_j$. Although the existence of neutrino oscillations implies that these are nonzero, all standard model lepton flavor violation is proportional to the very small neutrino masses. In particular, for the radiative decays, a suppression of at least $m_\nu^2/m_W^2 < 10^{-23}$ is expected to accompany the decay rate, resulting in total rates that are immeasurably low compared to the reach of foreseeable experiments [88].

Although, more sophisticated treatments are certainly possible, see for instance [89], the effects beyond leading order are typically very small, especially given the current absence of an observed signal. These radiative decay branching ratios can be simply expressed as [90],

$$\mathrm{BR}(\ell_i \to \ell_j \gamma) = \frac{\alpha_{EM} m_{\ell_i}^3}{\Gamma_{\ell_i}} \left(1 - \frac{m_{\ell_j}^2}{m_{\ell_i}^2}\right)^3 \left(\left|C_A^L\right|^2 + \left|C_A^R\right|^2\right) \tag{4.35}$$

In `FormFlavor`, no standard model contribution is provided, and the very small phase space factor is neglected. The Wilson coefficients are assumed not to run from the high scale, which is a good approximation.

> `mutoegamma[`μ`,Heff]`   evaluate $\mathrm{BR}(\mu \to e\gamma)$ for Wilson coefficients at μ
>
> `tautoegamma[`μ`,Heff]`   evaluate $\mathrm{BR}(\tau \to e\gamma)$ for Wilson coefficients at μ
>
> `tautomugamma[`μ`,Heff]`   evaluate $\mathrm{BR}(\tau \to \mu\gamma)$ for Wilson coefficients at μ

5 Numerical Wilson Coefficients: FFWilson

In this section we will describe the contents of the `FFWilson` package. One of the main components of `FFWilson` is the code to compile the analytic amplitudes. This was discussed already in section 2.2. Here we will focus on the other components of the `FFWilson` package. These include libraries for loop integral evaluation, basic functions to build the amplitudes for faster compilation, and functions to extract the numerical Wilson coefficients at the SUSY scale.

5.1 Loop Integral Libraries

Analytic expressions for Passarino-Veltman integrals can have "artificial instabilities" due to machine precision when mass parameters are nearly degenerate. In short, terms of the form

$$\frac{M_1^2 \ln M_2^2 - M_2^2 \ln M_1^2}{M_1^2 - M_2^2} \tag{5.1}$$

often appear. While (5.1) is perfectly well behaved when $M_1 - M_2 \to 0$, the machine precision rounding will cause this expression to artificially inflate. For this reason, FormFlavor has two separate running modes: "Fast" and "Acc". The difference between these two modes is their treatment of the Passarino-Veltman loop integrals. All code discussed in this subsection is contained in the Core/LoopIntegrals subdirectory.

5.1.1 "Fast" Loop Integrals

When running in "Fast" mode, FormFlavor explicitly substitutes exact algebraic degeneracies with their analytic limit through use of overloaded integral functions. But "Fast" mode does not solve the issue of accidental or near degeneracies. The file LoopIntegrals.nb contains code to compute the limits and generate LoopIntegrals.m.

Compiling in "Fast" mode typically takes $\mathcal{O}(5 \text{ min})$, and less than half that using the precompiled built amplitudes (see section 5.2). Once compiled, evaluating all processes included with FormFlavor at a single point in "Fast" mode generally takes less than 0.1 seconds.

5.1.2 "Acc" Loop Integrals

In "Acc" mode (accurate mode), mass parameters in the loop are checked case-by-case to assess whether there are any near degeneracies or not. In the case of a near degeneracy, a previously derived analytic expression that uses a Taylor expansion to fourth order in the mass splitting(s) is evaluated. The file LoopIntegralsAcc.nb contains code to compute the Taylor expansions and generate LoopIntegralsAcc.m. Importantly, "Acc" mode has a somewhat simplified table of possibilities presented in LoopIntegralsAcc.nb based on the different loop functions that actually appear in the current array of observables. This simplification is introduced in order to reduce the number of cases that need to be checked before evaluating, which slightly improves the overall speed. The integral expressions within LoopIntegralsAcc.m are actually automatically compiled by the FFWilson package.

Compiling observables in "Acc" mode typically takes $\mathcal{O}(3\,\mathrm{min})$, and less than half that using the precompiled built amplitudes (see section 5.2). Due to the very large number of loop integrals that need to be evaluated on-the-fly for each observable at every point, "Acc" mode is roughly a factor of 20 slower than "Fast" mode, and typically takes about 2 seconds to evaluate each point.

5.2 Building for Faster Compiling

Building amplitudes is a way to decrease the compiling time. When building amplitudes, many of the time-consuming substitutions performed prior to compiling are enacted in advance. These precompiled amplitudes are stored in a separate directory. These files are much, much larger than the amplitudes generated by `CalcAmps` as all summations are expanded, many standard model parameters are evaluated, and, in the case of "Fast" mode, long form integral expressions are presented explicitly.

Building amplitudes takes time, while "Acc" mode is a bit faster, building all amplitudes for both running modes takes on the order of 10 minutes. However, once this step has been performed once it does not need to be performed again unless:

a) standard model parameters (see Sec. 3), such as the CKM matrix, are changed

b) a new observable is added (although one may build only that observable)

c) a new model is introduced (amplitudes for other models do not need to be rebuilt)

d) built amplitudes are removed to save disk space

For both running modes, building the amplitudes reduces compiling time by more than a factor of two, and has no affect on evaluation time. These built amplitudes, which are stored in the {model}/ObservableAmps/SubAmps directory, occupy a fair bit of disk space, which scales with additional observables. The current array of observables occupies about 0.5 GB of disk space.

BuildFF[proclist]   Build all amplitudes in `proclist` for both running modes

BuildFF[proclist,mode]   as above, but for running mode `mode`

CleanBuiltFFFiles[]   Removes all built amplitudes

Built amplitudes can be removed with `CleanBuiltFFFiles[]`. The command opens a message box requiring the user to type "Y" in order to remove the files. Alternatively, the files can be deleted from the `{model}/ObservableAmps/SubAmps` directory by hand. All files in that directory must be removed as compiling will default to attempting to compile the built files if any files exist.

5.3   Extracting Wilson Coefficients

From the amplitudes, `FormFlavor` can extract the numerical Wilson coefficients for a model. The Wilson coefficient extraction is handled by the function `FFWilson`, which loops over all compiled processes and evaluates the amplitudes numerically.

`FFWilson[VariableList]`	extract nested table of processes and numerical Wilson coefficients from the analytic amplitudes

where `VariableList` is the output from `CalcSpec` or `FFRunFile` (see 6.1) defined by the model specific global variable `$FFCompileVarList` within `{model}/CompileAmps.m`. The `VariableList` must be given as real or complex numbers as appropriate for the variable in question, if this is not done, then the compiled amplitudes will not be used and the code will run extremely slowly (potentially with errors).

The output of `FFWilson` is a nested list of the form:

$$\{\{\{\text{process name, external state, topology type}\}, \{\text{scale}, \textstyle\sum_i C_i \mathcal{O}_i\}\},...\}$$

where process name (an element of `$FFProcessList`), the external state are the fields, e.g., $\{\{``s", ``d"\}, \{``s", ``d"\}\}$, the topology type is either "boxes", "penguins", or "wavefncorr" (wave function corrections), and scale is the scale at which the Wilson coefficients are evaluated (this is typically the $\mathcal{O}(\text{TeV})$ SUSY scale). The sum over operators includes all those relevant for the process, in the basis (3.1)–(3.5), with C_i being the numerical Wilson coefficients.

6   The Flavor-Violating MSSM: FFModel

`FFModel` is the name given to the model-dependent part of the code that reads in spectra, compiles Wilson coefficients, and performs other miscellaneous model-dependent tasks. Currently, the only `FFModel` provided with `FormFlavor` is the flavor-violating MSSM, i.e. `FFModel=MSSM`. It resides in the subdirectory `MSSM/`. In principle, `FormFlavor` allows for additional `FeynArts` models to be used with the `CalcAmps` package to produce

the amplitudes. The onus is then on the user to link the new model with the core code of `FormFlavor`, by constructing new `FFModel` code. In principle, only the spectrum calculator, some form of I/O for obtaining the parameters, dictionary of standard model field names for `CalcAmps`, and model specific compiling definitions are needed. Although `FormFlavor` defaults to the MSSM, a new model can be used by setting `FormFlavor'$FFModel={Model Directory}` prior to loading `FormFlavor`. In this section, we describe the spectrum calculator and SLHA2 I/O, the RG created with the aid of SARAH, and custom functions that come with the MSSM `FFModel`.

6.1 Spectrum Calculator & SLHA2 I/O

The MSSM-specific part of `FormFlavor` allows for three different input modes:

1. `FormFlavor` can read raw *input* SLHA2 files (basically, MSSM soft masses and Higgs parameters) and perform a basic tree-level calculation of the spectrum and mixing angles. In more detail: an SLHA2 input file contain the soft parameters, notably 3 x 3 mass squared matrices for $\tilde{m}_Q^2$, $\tilde{m}_U^2$, $\tilde{m}_D^2$, $\tilde{m}_L^2$, and $\tilde{m}_E^2$ in the super CKM basis. For input, these require the following SLHA2 blocks: `MSQ2`, `MSU2`, `MSD2`, `MSL2`, `MSE2`, `TU`, `TD`, `TE`, `MSOFT` (parameters $\{1, 2, 3, 21, 22, 25\} = \{M_1, M_2, M_3, M_{H_d}^2, M_{H_u}^2, \tan\beta\}$), `HMIX` ($1 = \mu$), `ALPHA` ($1 = \alpha$ the CP even Higgs mixing angle), `MASS` ($\{25, 36\} = \{m_h, m_A\}$ only) are required. The blocks `IMMSQ2`, `IMTU`, etc., are also used for the sfermion mass matrices and A terms, however, if not provided these parameters are set to zero with no error. See [91] for detailed descriptions for the conventions of these parameters. We will note that in SLHA2 conventions the matrix $\tilde{m}_Q^2$ is aligned with the down-type quark basis, i.e., the 1, 2, 3 rows and columns point in the $\{d, s, b\}$ direction (as opposed to the $\{u, c, t\}$ directions). If `ALPHA` is not provided, the code defaults to the alignment limit, that is $\alpha = \beta - \frac{\pi}{2}$.

2. Alternatively, `FormFlavor` can read *output* spectrum files containing masses and mixings in the SLHA2 format, which could include threshold corrections coming from one of the many public MSSM spectrum calculating codes. An SLHA2 output file contains the physical masses, notably 6 x 6 rotation matrices to connect the mass eigenbasis to the flavor eigenbasis. The six physical up-type squarks are ordered by ascending mass, i.e., the lightest squark is called $\tilde{u}_1$ and is identified with PDG ID 1000002, and the first row of the 6 x 6 rotation matrix indicates the flavor composition of this eigenstate (basis $\{\tilde{u}_L, \tilde{c}_L, \tilde{t}_L, \tilde{u}_R, \tilde{c}_R, \tilde{t}_R\}$). Analogous definitions

31

apply for down-type squarks, sleptons, and the three flavors of sneutrino. The following SLHA2 blocks are required, `MSOFT` (only needs $25 = \tan\beta$), `MASS` (all super particle masses as well as Higgs masses), `HMIX` ($1 = \mu$ only), `ALPHA` ($1 = \alpha$), the 4 x 4 neutralino mixing matrix `NMIX`, the two 2 x 2 charging mixing matrices `UMIX` and `VMIX`, the 6 x 6 sfermion mixing matrices `USQMIX`, `DSQMIX`, `SELMIX`, as well as the 3 x 3 sneutrino mixing matrix `SNUMIX`. `FormFlavor` additionally requires the A-terms, as these are couplings, so `TU`, `TD`, and `TE` are required. Again, `IMUSQMIX`, `IMTU`, etc., are used, but are simply set to zero if not provided.

3. Finally, `FormFlavor` has an internal soft mass and Higgs parameter format that it uses via the routine `CalcSpec` (see below), and this can also be input directly to the program:

$$\text{gaugino}[\{M_1, M_2, M_3\}], \text{higgs}[\{\mu, M_h, M_A, \tan\beta, \alpha\}], \text{squarkQLL}[m_Q],$$
$$\text{squarkURR}[m_U], \text{squarkDRR}[m_D], \text{sleptonLL}[m_L], \text{sleptonRR}[m_E], \qquad (6.1)$$
$$\text{Au}[A_u], \text{Ad}[A_d], \text{Ae}[A_e]$$

This mode is useful for quickly generating grids without needing to create SLHA2 files.

The primary function for reading SLHA2 files is `FFReadFile`.

`FFReadFile[file]`	read in parameters from SLHA2 `file` and output spectrum in format to be fed to compiled amplitudes

This function automatically decides whether the file is SLHA2 input or output. In the case of SLHA2 input, `FFReadFile` automatically calls a simple spectrum calculator, called `CalcSpec`,

`CalcSpec[CS]`	convert soft parameters in super CKM basis (input form shown in (6.1)) to output spectrum to be fed into compiled amplitudes

that will convert the SLHA2 input to a spectrum that can be directly fed into the compiled amplitudes. The function joins the soft parameters with the D-term contributions, manipulates the A-terms and μ-parameter expressions to compute the 6 x 6 mixing matrices and eigenvalues. Importantly, the spectrum calculator sets all neutralino eigenvalues to be positive and uses a complex mixing matrix. No matter the read in format, the output of `FFReadFile` and `CalcSpec` is put into the format of `$FFCompileVarList` defined within {`model`}/`CompileAmps.m`.[4]

---

[4]In truth, the output needs to be flattened to match `$FFCompileVarList`, but the output is more readable prior to flattening. This flattening is automatically performed by `FFWilson`.

Once these parameters are evaluated, the resulting spectrum may be output to an SLHA2 file using the `FFWriteFile` command.

`FFWriteFile[CalcSpecOutput,file]`	takes the `CalcSpec` output and writes to `file` in SLHA2 format

Both the spectrum calculator and the SLHA2 input and output routines are contained in MSSM/`CalcSpec`.m. `FormFlavor` comes with equivalent SLHA2 input and output files, called `ExampleSLHA2in.dat` and `ExampleSLHA2out.dat`, that are parameter points in a Q-class model of extended gauge mediation [92, 93].

6.2 RG Evolution

`FormFlavor` additionally contains a simple function for performing renormalization group evolution. This evolution contains full 3 x 3 running at one loop level including CP violating phases and applies BMPZ QCD threshold corrections [94]. The basic RG equations were derived using the SARAH package [95]. As the RG was designed with flavor physics in mind, many features for a more precise spectra determination, such as two-loop Higgs corrections, back-and-forth running for better specification of scale, and threshold corrections (except for the QCD BMPZ threshold corrections), are not included in this module.

`RGFile[infile,outfile,`μ`]`	RG evolve high scale `infile` down to scale μ and output `outfile`

The RGE is contained in the file MSSM/`RGE`.m, although some of the SLHA2 I/O is defined in MSSM/`CalcSpec`.m. Importantly, the RGE is not used by any other portion of the code. A user creating a new `FFModel` would not need to create an RGE for that model.

6.3 Custom User Functions

There are two simple places where one can define user functions to be accessed by the code. The first is the file Core/`UserCore`.m, which contains no functions in the default release of `FormFlavor`. The second place that one can define new user functions is {`model`}/UserModel.m. Within the MSSM/`UserModel`.m file, several functions have been defined that are useful chains of code. In addition to the `FFfromSLHA2` and `FFConstraintsfromSLHA2` defined in section 2.3, there is `FFWilsonfromSLHA2`, which uses the `FFWilson` command of section 5.3.

`FFWilsonfromSLHA2[file]`	load SLHA2 `file` and run `FFWilson` using current active running mode
`FFWilsonfromSLHA2[file,mode]`	as above, but for running mode `mode`

7 The CalcAmps Package

One of the most important and distinguishing features of `FormFlavor` is the ability to calculate one-loop Wilson coefficients for new flavor and CP observables and/or new models, entirely from scratch. These tasks are performed by the `CalcAmps` package, based on the machinery of `FeynArts` [8] and `FormCalc` [9]. The various routines used by `CalcAmps` are described in section 7.1, while section 7.2 contains a detailed explanation of how a user can add new observables to the program. The wrapper package {model}/CalcAmps/CalcAmps.m loads the main code in Core/CalcAmpsPackage.m and links to the model-specific definitions in {model}/CalcAmps/CalcAmpsModel.m.

7.1 CalcAmps

The main routine to generate new processes is also called `CalcAmps` and is found in the file Core/CalcAmpsPackage.m.

`CalcAmps[proc,`$\mathcal{O}1$`,`$\mathcal{O}2$`]`	Calculate amplitudes relevant for process `proc` involving external states contained in $\mathcal{O}1$ and $\mathcal{O}2$. This is the main routine of the `CalcAmps` package.

It takes as input a process name (e.g., "K-K mixing"), and two lists of external SM fields, which define the operator basis for the observable. The syntax is $\{f_1, f_2\}, \{f_3, f_4\}$ for the set of 4-fermi operators with Dirac indices contracted between the braces $(\overline{f}_1(\dots)f_2)(\overline{f}_3(\dots)f_4)$ and $\{f_1, f_2\}, \{b_1\}$ for a dipole-type operator $(\overline{f}_1(\dots)f_2)(b_1)$. Note that the order of the particles is important. Additionally, the f_i should only be SM particles, anti-particles will not be accepted. Currently only 3- and 4-body processes are supported, so these lists must be length 2 and 2, or 2 and 1, or 1 and 2. Note that $1 \to 3$ processes, such as $K \to \pi\nu\nu$, are expressed as 2 and 2, i.e., $\{d, s\}\{\nu, \nu\}$.

One of the most important subroutines called by the `CalcAmps` function is the function `GenerateDiagrams`.

`GenerateDiagrams[`$\mathcal{O}1$`,`$\mathcal{O}2$`]`	Generate all diagrams connecting initial and final states $\mathcal{O}1$ and $\mathcal{O}2$.

`GenerateDiagrams` is a general purpose routine that takes as input the initial and final state particles and then draws all one-loop diagrams connecting them. These diagrams are further classified into topologies (boxes, penguins and wave-function corrections) that are relevant for 3- and 4-particle processes. (Currently, 5+ particle processes are not supported.) Default options for `GenerateDiagrams` include: `Model→"FVMSSM"`, other models are in principle possible; and `GenerateSM→False`, this eliminates all diagrams involving only SM fields, since these are accounted for (often with much higher precision than 1-loop) by the observable functions discussed in section 4. In order for `GenerateSM→False` to work, the user has to specify all the SM fields with the list, `SMlist` defined in `{model}/CalcAmps/CalcAmpsModel.m`. `GenerateDiagrams` returns a list of diagrams in FeynArts format indexed by their topology type, e.g.,

{{ "boxes",box diagrams}, { "penguins",penguin diags}, { "wavefncorr", wfc diags}}

`CalcAmps` uses subroutines defined within the package to turn the "DiracChains" of `FormCalc` (basically Gamma matrices sandwiched between external state spinor wave-functions, e.g., $\overline{v}(p_1)\gamma^\mu u(p_2)\overline{u}(p_3)\gamma_\mu v(p_4)$) into the Wilson operators used throughout `FormFlavor`. It is careful to include a factor of 2 when the initial and final states are CP conjugate as is the case for $\Delta F = 2$ observables, i.e., the operator $\langle \overline{d}s|(\overline{s}d)(\overline{s}d)|d\overline{s}\rangle = 2(\overline{u}v)(\overline{v}u)$. `CalcAmps` will also apply the Gordon identity to extract the magnetic dipole operator from amplitudes of the form $(\overline{u}v)\epsilon \cdot k$.[5]

Another important feature of `CalcAmps` is the third generation dominant approximation. Currently, the option `ThirdGenDominance→True` is set by default for `CalcAmps`. In principle, one can set `ThirdGenDominance→False`, but this has not been tested and

---

[5]In order to have appropriate signs in front of the relevant Wilson operators, `CalcAmps` must address two technical subtleties concerning fermion ordering within `FeynArts` and `FormCalc`. First, the diagrams generated with `FeynArts` can result in different signs in `FormCalc` when the order of the initial and/or final state fermions are changed. For the operators $(\overline{f}_1(\dots)f_2)(\overline{f}_3(\dots)f_4)$, the correct order is $|f_2, \overline{f}_1\rangle$ and $|f_3, \overline{f}_4\rangle$ for the initial and final states. Then $\langle f_3, \overline{f}_4|C(\overline{f}_1(\dots)f_2)(\overline{f}_3(\dots)f_4)|f_2, \overline{f}_1\rangle = C\overline{v}_1(\dots)u_2\overline{u}_3(\dots)v_4$ with no additional signs. The second technical subtlety is that `FormCalc` contains an option `FermionOrder` (which defaults in `FormCalc` to `Automatic`). A different choice of `FermionOrder` results in a different DiracChain related to one another by Fierz rearrangement identities, which can result in different signs or even different operator bases, potentially leading to an incorrect identification of the Wilson coefficient. `CalcAmps` sets `FermionOrder=`$\{2, 1, 3, 4\}$ for a four-fermi interaction, and `FermionOrder=`$\{2, 1\}$ for $2 \to 1$ process. This hardwired choice is a function of the order of the initial and final states in the call to `GenerateDiagrams`, which is called automatically from `CalcAmps` to avoid user error. The user only has to specify the operator basis and in principle these details will be taken care of automatically.

may result in extremely slow evaluation times. In the 3rd gen dominant approximation, all 1st/2nd generation SM fermion masses are set to zero, with the exception of m_μ which is preserved to keep important terms in $B_q \to \mu\mu$. This zeroing is enacted to significantly speed up the computation of the loop diagrams, both in `CalcAmps` and `FormFlavor`.

Formally, terms that are suppressed by powers of m_f/M_W or m_f/M_{SUSY} are actually higher dimension (e.g., dimension 8) operators. To remove these, all the Mandelstam invariants S, T, etc., are set to zero in the computation. Essentially, `FormFlavor` evaluates the Wilson operators at zero external momentum. In order to further simplify expressions and reduce evaluation time, `CalcAmps` also drops all powers of m_b and m_μ unless they are coming from Yukawa couplings that do not appear suppressed by $\tan\beta$. Finally, there are a number of MSSM-specific simplifications (everything preceding this is in principle model-independent or at least 2HDM model independent, provided one uses the same notation as default FeynArts MSSM for the fermions and their masses) to further speed up the later compilation and numerical evaluation of the Wilson coefficients. All of these model specific simplifications are contained in the file {model}/CalcAmps/CalcAmpsModel.m

The output of `CalcAmps` should be stored in a file with the use of `WriteAmp` which should automatically put the file into the {model}/ObservableAmps directory that contains all amplitudes. It is from this directory that `FormFlavor` accesses the amplitude files.

`WriteAmp[amp,file]`	Write an `amp` generated by `CalcAmps` to `file` within the {model}/ObservableAmps directory.

For ease of use, there is a simple default `CalcAmps` front-end notebook in the `MSSM/` directory, {model}/CalcAmps/CalcAllAmps.nb, which can be used as a template for adding new processes.

7.2 Adding New Observables

New observables can be added to `FormFlavor` rather easily. First, observables that depend on the same Wilson coefficients, for instance Δm_K & ϵ_K, can be added within the same observable function, and do not need to be separately produced in `CalcAmps`. However, this subsection is meant to act as a tutorial for adding observables that depend on new Wilson coefficients. By following the instructions provided in this subsection, the user should be able to add their own observables to `FormFlavor`.

1. **Generate amplitude with** `CalcAmps` – The `CalcAmps` package should be used to generate an amplitude file. The file `{model}/CalcAmps/CalcAllAmps.nb` contains several existing processes that can be mirrored in order to determine the initial and final state definitions. Generating this amplitude will typically take a little bit of time, but once generated the amplitude should be stored as a ∗.m file in the `{model}/ObservableAmps` directory with the use of `WriteAmp`. (We will refer to this amplitude file later on as `{X}.m`.) It is recommended that the user gives an intuitive name to both the process and the amplitude file. Note: `CalcAmps` and `FormFlavor` are not designed to be run simultaneously. If one package is loaded, the `Mathematica` kernel should be quit in order to load the other.

2. **Create and link an** `Obs{X}.m` **file** – With the amplitude stored within the `Core/Observables` directory, the user should create an `Obs{X}.m` file for the process of interest. It is recommended that the user copy an existing file that is most similar to the process of interest to use as a template. Within `Core/FFObservables.m`, the user should add a line,

   ```
   Get[FormFlavor`$FFPath<>"/Core/Observables/Obs{X}.m"]
   ```

   within an appropriate subsection (or create a new subsection if no appropriate one exists). This will ensure that when the user loads `FormFlavor`, the new observable will also be loaded.

3. **Modify** `Obs{X}.m` **for the process of interest** – All observable files follow a simple convention to facilitate the addition of new observables. We will present the individual portions and what needs to be modified in the order they appear within the files.

   (a) **Linking** – The first block contains several lines of code, but only the first and last need to be modified. The first line is

   ```
   TempAmpFileName={X}.m;
   ```

   The user should set this to the filename used in step 1. This line and the subsequent code informs `FormFlavor` of the name of the amplitude file within the `{model}/ObservableAmps` directory, and appends the filename to `$FFAmpFileList`. The next line automatically accesses the amp file to extract the process that was defined in step 1, and appends the process name to `$FFProcessList`. The last line

```
ObservableFunction[TempProcessName]={observable function};
```

needs to be modified by the user so that the name of the observable function defined in the final block will be linked to the process.

(b) **Observable details** – The next block contains details on the current experimental and theoretical status of the observable, as well as its name within `FormFlavor`. The first line is `TempObsName="NAME";`, and should contain the name as you want it to appear in `FormFlavor`. To reduce user error, `TempObsName` is used in every block, but the last line `FF{X}Name=TempObsName;` defines a unique identifier for the observable's name that will be called again later. The definitions of `FFObsClass`, `FFExpValue`, `FFExpUnc`, and `FFSMUnc` should be introduced and assigned the appropriate values. `FFSMValue` also appears here, however, its definition is typically not used in the code. See section 4 for more details about these definitions. The second to last line for each observable,

```
AppendTo[$FFObsNameList,TempObsName];
```

informs `FormFlavor` of the name and presence of the observable. Multiple observables originating from the same process file can appear in this block. The user is encouraged to document these with original references.

(c) **Observable specific pre-evaluations** – An optional third block contains any code that is specific to the observable, but is pre-evaluated in the interest of expediting the runtime calculation of `FormFlavor`. Often, it may be in the user's best interest to separate a detailed calculation that depends on standard model parameters from the bulk of the evaluation so that it will not be reevaluated each time the observable's subroutine is passed new numerical Wilson coefficients.

(d) **Observable function** – The final block of code within each `Obs{X}.m` file is the observable function. This is the all important block of code that translates Wilson coefficients into flavor observables. The function should be given an intuitive, unique name and referred to in the previous block. Wilson coefficients can be extracted by `Coefficient[wilson,$\mathcal{O}$];` where $\mathcal{O}$ is the operator name within `FormFlavor`, e.g., `OpV["$L$","$L$"][{"$b$","$s$"},{"$b$","$s$"}]` (see eqs. 3.1–3.5). The final line of this function is a list of observable names and values, e.g.,

```
{{FFObs1Name,Obs1val},{FFObs2Name,Obs2val},...}}
```

where `FFObs1Name` is the name of the observable defined in the second block, and `Obs1val` is the numerical value of the observable as determined by the observable functions code. Any new SM parameters should be added to `Core/SMParameters.m`. Again, the user is strongly encouraged to provide detailed references for the origin of all expressions within the observables code.

Acknowledgements

A   Comparing FormFlavor to Other Public Codes

There are several existing public codes that calculate flavor and CP observables in the MSSM. In this brief appendix, we will present a validation of FormFlavor against some of these codes. Our focus will be on the few other codes that can handle a general MSSM spectra without mandating minimal flavor violation, and that compute a broad range of flavor and CP violating observables. These are: `SPheno` [96], `SARAH − SPheno − FlavorKit` (`FlavorKit` for short) [13], and `SUSY_Flavor` [12]. Other codes include: `SuperIso` [97], `NMHDECAY` [98], `MicrOMEGAs` [99], `SusyBSG` [100], `SuperLFV` [101], `SuSeFLAV` [102], `ISAJET` with `ISATOOLS` [103], and `flavio` [104].

Table 6 compares various features of these four codes. For instance, whether the code can automatically compute the one-loop Wilson coefficients from scratch, or whether these loop functions are hard-coded. Some codes include effects beyond one-loop order such as chiral ($\tan\beta$) resummations [105] or double Higgs penguins [10]. Other features include the QCD RG evolution of SUSY scale Wilson coefficients down to the low scale of the flavor or CP violating observable. In some cases, this evolution can affect the ultimate result by as much as a factor of three or more. As far as we can tell, `SPheno` and `FlavorKit` do not run the Wilson coefficients with the QCD beta functions from the SUSY scale to the low scale (passing through the top threshold). For some observables, but not all (e.g., not for kaon mixing), `FlavorKit` includes running from m_t (e.g., input is

39

Evaluation	FormFlavor	SUSY_Flavor	FlavorKit	SPheno
Automated One-Loop	✓	X	✓	X
Chiral Resummations	X	✓/X	X	X
Double Higgs Penguins	X	X*	✓	✓
Wilson Coefficient Running	✓/X	✓	✓	X

Inputs	FormFlavor	SUSY_Flavor	FlavorKit	SPheno
Non-MFV General MSSM	✓	✓	✓	✓
Soft Parameter Input (SLHA2 in)	✓	✓	✓	✓
Full Spectra Input (SLHA2 out)	✓	X†	X	X
Messenger/GUT Scale Input	✓	X	✓	✓
Threshold Corrections	✓/X	X	✓/X	✓

Table 6: A ✓ means the feature is included, ✓ means the feature is partially included, while an X means that it is not. ✓/X means the feature is included and can be enabled/disabled.

*Although SUSY_Flavor does not explicitly include double Higgs penguins, these are presumably accounted for at some level when chiral resummation is enabled.
†The SUSY_Flavor 2.5 manual states that it also accepts "output files produced by other public libraries calculating various aspects of the MSSM phenomenology," but it is not clear what this means from reading the rest of the SUSY_Flavor manual and reading the SUSY_Flavor code. It appears to not use the threshold corrected masses and mixings, but still performs the tree-level diagonalization itself.

assumed to be at the scale m_t) to the low scale. `SPheno` says in its manual that all Wilson coefficients are evaluated with couplings at the scale m_t. No mention is made of running down to the low scale. `SUSY_Flavor` and `FormFlavor` both have running from the SUSY scale (generally defined to be some average of the gluino and squark masses) to the low scale. Finally, there are various types of inputs that the codes could possibly accept: messenger-scale soft parameters, weak-scale soft parameters, masses and mixings with or without threshold corrections. Features of the four codes are summarized in table 6.

A.1   Comparison of the codes

We will now describe a more detailed, quantitative comparison of `FormFlavor` against other public codes, specializing further to `FlavorKit` v2.53 and `SUSY_Flavor` v4.8.6, because all three of these have the option of taking soft parameters as inputs and computing tree-level masses and mixings. This allows our comparison of the Wilson coefficients, QCD RG and flavor observable functions to be isolated from the complicated issue of threshold correcting the SUSY spectrum. `SPheno` does not have the capability of turning off threshold corrections, so, although examined, we decided not to include it in the comparison. We will note that a lot of disagreement was found between `SPheno` and the other codes; however, it is difficult to disentangle these from the threshold corrections, and especially regions of parameter space where the threshold corrections would not converge.

In `SUSY_Flavor`, chiral resummations ($\tan\beta$ resummation) can be toggled on and off to different orders of resummation. Since `FormFlavor` and `FlavorKit` do not have this feature currently, we will compare against `SUSY_Flavor` with chiral resummations turned off. The different programs have (slightly) different input parameter choices for all of the flavor observables.

The parameter space of the general flavor and CP violating MSSM is enormous. To compare the codes in a simple and presentable fashion, we choose the following two lines through the parameter space:

- Along the "LLRR line", the A-terms are zero and we turn on equal deformations to all the sfermion soft mass-squareds

$$
m^2_{QLL} = m^2_{URR} = m^2_{DRR} = m^2_{LLL} = m^2_{ERR} = m^2 \mathbf{1}_3 + \begin{pmatrix} 0 & \delta m^2 & \delta m^2 \\ (\delta m^2)^* & 0 & \delta m^2 \\ (\delta m^2)^* & (\delta m^2)^* & 0 \end{pmatrix}
$$
(A.1)

41

- Along the "LR line", the soft mass-squareds are diagonal and we are turning on

$$A_u = A_d = A_\ell = \begin{pmatrix} 0 & A & A \\ A & 0 & A \\ A & A & 0 \end{pmatrix} \tag{A.2}$$

We have set all the gaugino soft masses and μ to 1 TeV, $m = 500$ GeV, $m_A = 2$ TeV, $\tan \beta = 10$ and $\arg A = \arg \delta m^2 = 0.1$.

Shown in figs. 2 and 3 are the comparison of the three codes along the LLRR line and the LR line respectively. The solid lines indicate the respective code run "out of the box," i.e., with all default input parameters (except for disabling chiral resummations in the case of SUSY_Flavor) and no other modifications to the codes. We see that all three codes generally agree well on the $\Delta F = 0, 1$ observables; while for $\Delta F = 2$ observables, SUSY_Flavor and FormFlavor agree well but FlavorKit disagrees strongly with both.

In some instances (SUSY_Flavor's neutron EDM and FlavorKit's Δm_K and ϵ_K), the QCD RG from the SUSY scale to the low scale is not included. In these cases, we display a blue dashed line where the QCD RG in FormFlavor has been turned off in order to better compare FormFlavor's evaluation with those of the other code. The ability to disable the QCD RG for comparison and debugging purposes is a useful option only found in FormFlavor.

Inspecting all three codes, we find that the flavor observable functions are typically using similar references. One major exception to this is $b \to s\gamma$. Here all three codes are in decent qualitative agreement (within a factor of ~ 2), but a more detailed comparison is difficult because the treatments of $b \to s\gamma$ is quite different in all three cases. In FormFlavor, the latest, full NNLO results are used [18]. SUSY_Flavor follows NLO results from '96 [106], further enfolding hardcoded, unpublished SUSY loop calculations performed by the authors. FlavorKit follows the partial NNLO calculation of [40, 107].

In the remainder of this appendix, we will focus on the observables that show the largest discrepancies between the three codes. These are: Δm_K and ϵ_K, Δm_{B_q}, and $B_q \to \mu^+ \mu^-$.

A.2   In depth comparison: Δm_K and ϵ_K

The sign of Δm_K is physical relative to the sign of $\Delta \Gamma$ between K_S and K_L. FormFlavor and SUSY_Flavor find a minus sign on the LLRR line, whereas FlavorKit does not. By examining the code, we find that FlavorKit is taking the absolute value of Δm_K.
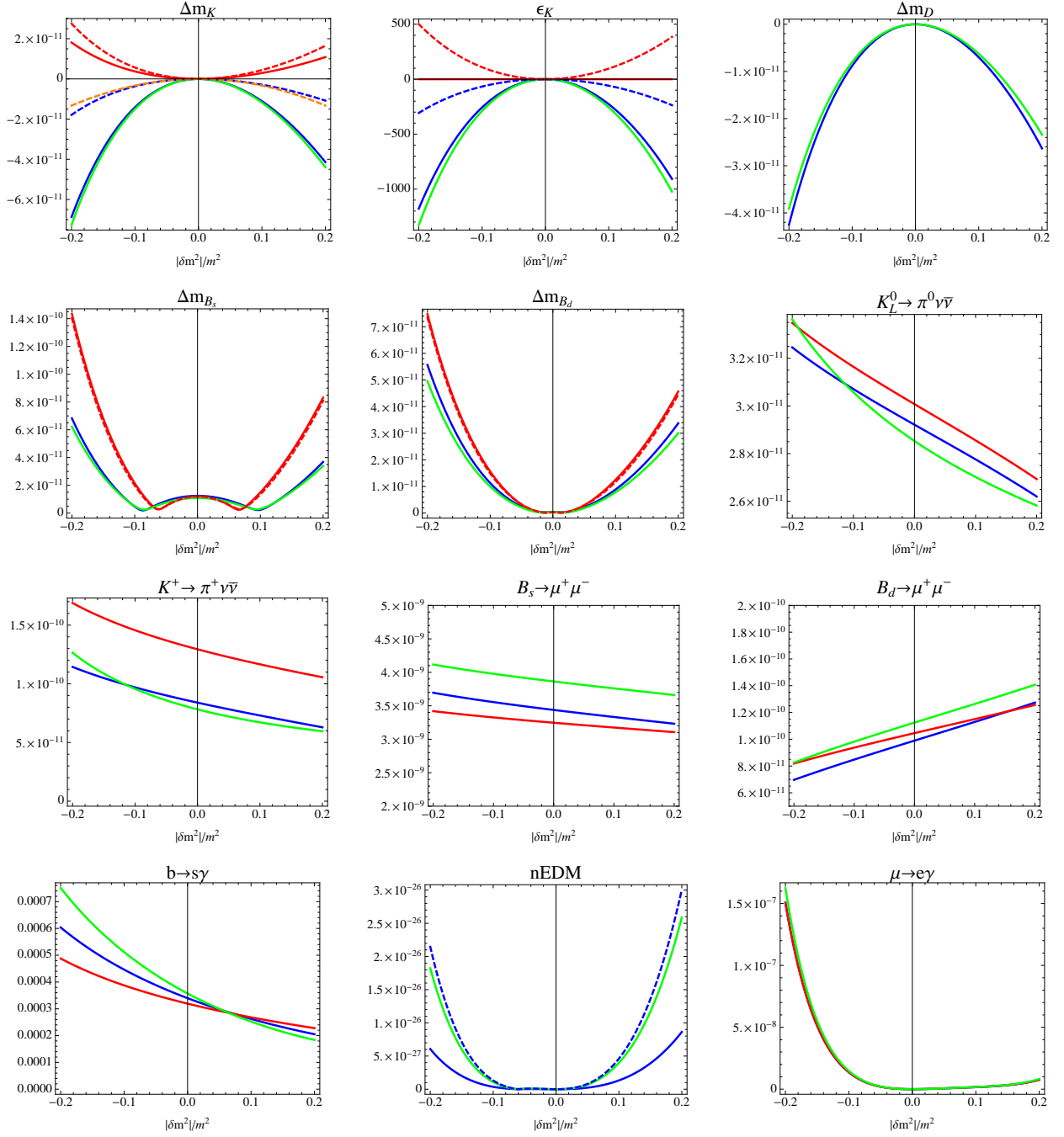
Figure 2: LLRR line as described in the text. Blue is `FormFlavor`, green is SUSY_Flavor, and red is FlavorKit. Solid lines indicate the flavor observable computed with the respective code "out of the box" i.e. with no modifications to the default settings. Dashed lines, where present indicate non-default options or modified source code as described in the text.
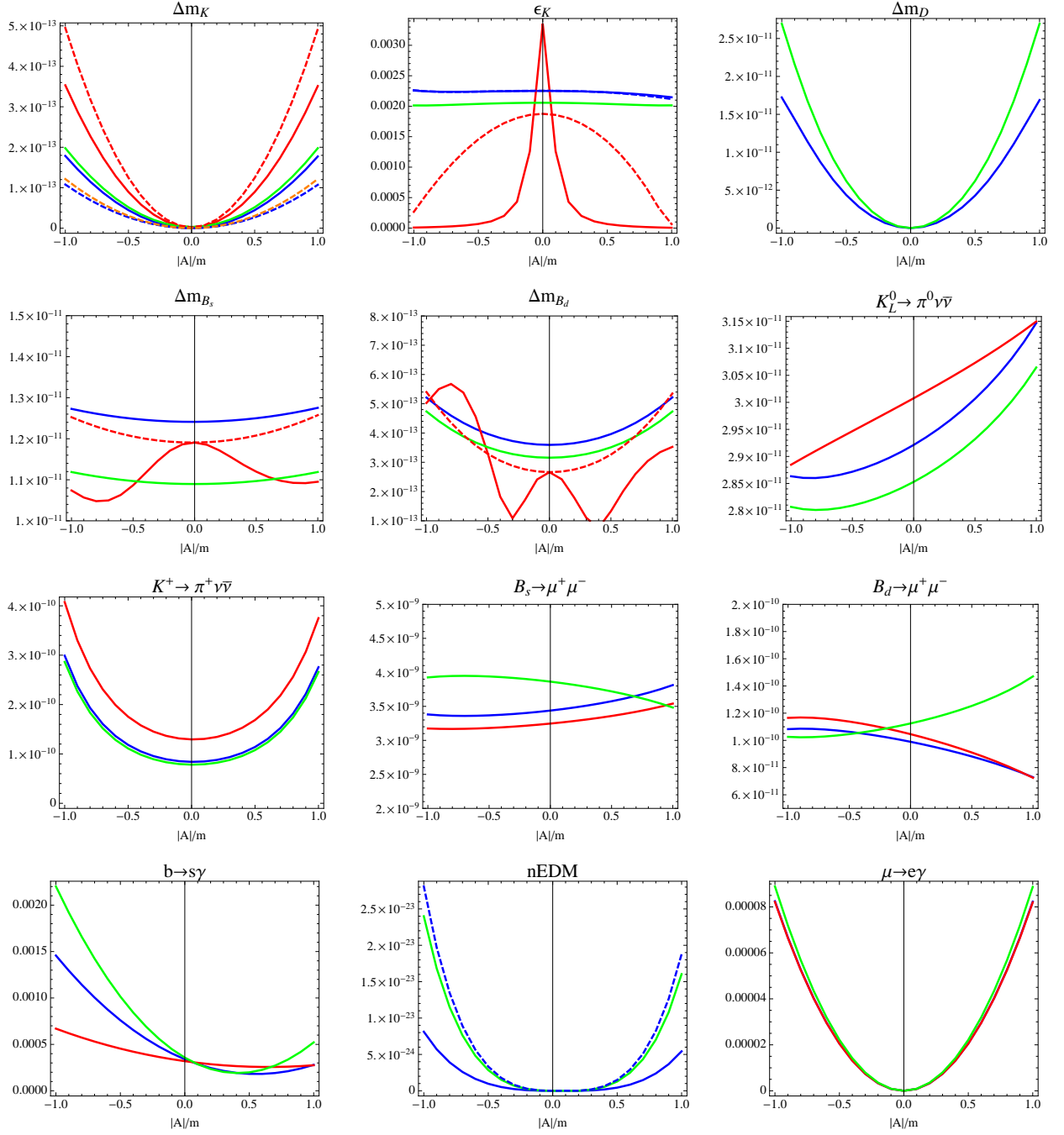
43

Figure 3: LR line as described in the text. Blue is `FormFlavor`, green is SUSY_Flavor, and red is FlavorKit. Solid lines indicate the flavor observable computed with the respective code "out of the box" i.e. with no modifications to the default settings. Dashed lines, where present indicate non-default options or modified source code as described in the text.

For ϵ_K, `FormFlavor` and `SUSY_Flavor` are in excellent agreement, while `FlavorKit` disagrees substantially. Part of the reason for this is that `FlavorKit` is dividing by the theoretical prediction for Δm_K. In most definitions of ϵ_K, e.g. [36], it is divided by the very well known experimental value, a convention which both `FormFlavor` and `SUSY_Flavor` follow. This choice results in the possibility that ϵ_K values can, in principle, be larger than one, but obviously, this is a very sick region of parameter space with either definition.

To facilitate an apples-to-apples comparison of `FormFlavor` and `FlavorKit`, we made the following changes to the latter: we modified the source code to divide ϵ_K by $(\Delta m_K)_{exp}$ instead of $(\Delta m_K)_{th}$, and we changed `FlavorKit`'s hadronic input parameters (the "bag parameters" and R_K) to match `FormFlavor`. The result of this modification to `FlavorKit`'s result is shown in red dashed in figs. 2 and 3. As explained above, `FlavorKit` is not performing any QCD RG between the SUSY scale and the low scale, so we should compare red dashed against blue dashed, which is `FormFlavor` with QCD RG turned off.

While doing this brings the LLRR line into better agreement, the agreement with the LR line actually worsens. This implies there is a mismatch between `FormFlavor` and `FlavorKit` at the level of the Wilson coefficients themselves. We have checked our analytic Wilson coefficients (gluino boxes) for against the MIA expressions of Gabbiani et al [108] and found excellent agreement. This is shown in figs. 2 and 3 in orange dashed. We also agree well with `SUSY_Flavor`, so we conclude there is likely a mistake in the treatment in `FlavorKit`.

A.3   In depth comparison: Δm_{B_q}

For Δm_{B_q}, `FlavorKit` disagrees with `SUSY_Flavor` and `FormFlavor` along the LLRR line, but this disagreement is at most a factor of two. However, along the LR line `FlavorKit` disagrees immensely.

Most of this discrepancy is due to the two-loop double-Higgs penguins included in `FlavorKit`. To make a improved comparison, we have turned off these double-Higgs penguins (DHPs) in `FlavorKit`'s treatment of Δm_{B_q} by modifying the source code (shown with the red dashed lines in figs. 2 and 3). We see that impact on the the LLRR line is small, but the agreement along the LR line becomes much better. Thus, the DHPs in `FlavorKit` are quite significant. As $\tan\beta$ is not that large, and the heavy Higgs states are in the decoupling limit, this result is a bit surprising. Checking against analytic expressions for the DHPs [10], we find that the DHPs should be several orders

of magnitude smaller than the values determined by `FlavorKit`.

Even with the DHPs removed by hand from `FlavorKit`, the result has a noticeably different shape than that found in `FormFlavor` and `SUSY_Flavor`, which agree well in their shapes. There is more to the discrepancy than just what can be accounted for by the DHPs.

A.4 In depth comparison: $B_q \to \mu^+\mu^-$

For $B_q \to \mu\mu$, `FlavorKit` and `FormFlavor` are in good agreement. `SUSY_Flavor` also agrees along the LLRR line, but differs qualitatively along the LR line. Along the LR line, both the SM and new physics contribution enter the branching ratio dominantly in F_A, while for the LLRR line the SM contribution is in F_A, but the new physics contribution is mostly in F_P. From (4.29), we can see that,

$$\mathrm{BR}(B_i \to \mu^+\mu^-) \propto |F_{A,SM} + F_{A,NP} + F_{P,NP}|^2 \,. \tag{A.3}$$

Thus there is a relative sign discrepancy between the two codes in $F_{A,SM}$ and $F_{A,NP}$ in the LR case. Using `FormFlavor`, we can compute both the new physics and SM piece in situ to conclusively determine the relative sign between the two, since all other factors are treated identically between the two pieces. For this reason, we are confident in the sign found with `FormFlavor`. (And `FlavorKit` confirms it.) In `SUSY_Flavor` both the SM and new physics contributions are hard-coded formulas, so a mistake in the overall sign could very plausibly have been introduced.

References

[1] A. J. Buras, *Weak Hamiltonian, CP violation and rare decays*, `hep-ph/9806471`.

[2] S. P. Martin, *A Supersymmetry primer*, *Adv.Ser.Direct.High Energy Phys.* **21** (2010) 1–153, [`hep-ph/9709356`].

[3] BELLE-II collaboration, T. Abe et al., *Belle II Technical Design Report*, `1011.0352`.

[4] LHCB collaboration, A. A. Alves, Jr. et al., *The LHCb Detector at the LHC*, *JINST* **3** (2008) S08005.

[5] NA62 collaboration, R. Fantechi, *The NA62 experiment at CERN: status and perspectives*, `1407.8213`.

[6] NA62 collaboration, *NA62 Technical Design Report*, 2010.

[7] S. Aoki, Y. Aoki, C. Bernard, T. Blum, G. Colangelo et al., *Review of lattice results concerning low-energy particle physics*, *Eur.Phys.J.* **C74** (2014) 2890, [1310.8555].

[8] T. Hahn, *Generating Feynman diagrams and amplitudes with FeynArts 3*, *Comput.Phys.Commun.* **140** (2001) 418–431, [hep-ph/0012260].

[9] T. Hahn and M. Perez-Victoria, *Automatized one loop calculations in four-dimensions and D-dimensions*, *Comput.Phys.Commun.* **118** (1999) 153–165, [hep-ph/9807565].

[10] A. J. Buras, P. H. Chankowski, J. Rosiek and L. Slawianowska, *Δ M(s) / Δ M(d), sin 2 Beta and the angle γ in the presence of new $\Delta F = 2$ operators*, *Nucl. Phys.* **B619** (2001) 434–466, [hep-ph/0107048].

[11] S. M. Barr and A. Zee, *Electric Dipole Moment of the Electron and of the Neutron*, *Phys. Rev. Lett.* **65** (1990) 21–24.

[12] A. Crivellin, J. Rosiek, P. H. Chankowski, A. Dedes, S. Jaeger and P. Tanedo, *SUSY_FLAVOR v2: A Computational tool for FCNC and CP-violating processes in the MSSM*, *Comput. Phys. Commun.* **184** (2013) 1004–1032, [1203.5023].

[13] W. Porod, F. Staub and A. Vicente, *A Flavor Kit for BSM models*, *Eur. Phys. J.* **C74** (2014) 2992, [1405.1434].

[14] Z. Ligeti and F. Sala, *A new look at the theory uncertainty of ϵ_K*, 1602.08494.

[15] FERMILAB LATTICE, MILC collaboration, A. Bazavov et al., *$B^0_{(s)}$-mixing matrix elements from lattice QCD for the Standard Model and beyond*, 1602.03560.

[16] F. Mescia and C. Smith, *Improved estimates of rare K decay matrix-elements from $K_{\ell 3}$ decays*, *Phys.Rev.* **D76** (2007) 034017, [0705.2025].

[17] J. Brod, M. Gorbahn and E. Stamou, *Two-Loop Electroweak Corrections for the $K \to \pi \nu \bar{\nu}$ Decays*, *Phys.Rev.* **D83** (2011) 034030, [1009.0947].

[18] M. Misiak, H. Asatrian, R. Boughezal, M. Czakon, T. Ewerth et al., *Updated NNLO QCD predictions for the weak radiative B-meson decays*, 1503.01789.

[19] M. Benzke, S. J. Lee, M. Neubert and G. Paz, *Long-Distance Dominance of the CP Asymmetry in $B \to X_{s,d}\gamma$ Decays*, *Phys. Rev. Lett.* **106** (2011) 141801, [1012.3167].

[20] BaBar collaboration, J. P. Lees et al., *Measurements of direct CP asymmetries in $B \to X_s\gamma$ decays using sum of exclusive decays*, *Phys. Rev.* **D90** (2014) 092001, [1406.0534].

[21] BaBar collaboration, P. del Amo Sanchez et al., *Study of $B \to X\gamma$ decays and determination of $|V_{td}/V_{ts}|$*, *Phys.Rev.* **D82** (2010) 051101, [1005.4087].

[22] A. Crivellin and L. Mercolli, *$B \to X_d\gamma$ and constraints on new physics*, *Phys.Rev.* **D84** (2011) 114005, [1106.5499].

[23] CMS AND LHCb collaboration, CMS and LHCb, *Combination of results on the rare decays $B^0_{(s)} \to \mu^+\mu^-$ from the CMS and LHCb experiments*, .

[24] C. Bobeth, M. Gorbahn, T. Hermann, M. Misiak, E. Stamou et al., *$B_{s,d} \to \ell^+\ell^-$ in the Standard Model with Reduced Theoretical Uncertainty*, *Phys.Rev.Lett.* **112** (2014) 101801, [1311.0903].

[25] A. Czarnecki and B. Krause, *Neutron electric dipole moment in the standard model: Valence quark contributions*, *Phys.Rev.Lett.* **78** (1997) 4339–4342, [hep-ph/9704355].

[26] MEG collaboration, J. Adam et al., *New constraint on the existence of the $\mu^+ \to e^+\gamma$ decay*, *Phys. Rev. Lett.* **110** (2013) 201801, [1303.0754].

[27] Particle Data Group collaboration, J. Beringer et al., *Review of Particle Physics (RPP)*, *Phys.Rev.* **D86** (2012) 010001.

[28] Heavy Flavor Averaging Group collaboration, Y. Amhis et al., *Averages of B-Hadron, C-Hadron, and tau-lepton properties as of early 2012*, 1207.1158.

[29] A. J. Buras, S. Jager and J. Urban, *Master formulae for Delta F=2 NLO QCD factors in the standard model and beyond*, *Nucl.Phys.* **B605** (2001) 600–624, [hep-ph/0102316].

[30] HPQCD collaboration, R. J. Dowdall, C. T. H. Davies, R. R. Horgan, C. J. Monahan and J. Shigemitsu, *B-Meson Decay Constants from Improved Lattice*

Nonrelativistic QCD with Physical u, d, s, and c Quarks, *Phys. Rev. Lett.* **110** (2013) 222003, [`1302.2644`].

[31] J. L. Rosner, S. Stone and R. S. Van de Water, *Leptonic Decays of Charged Pseudoscalar Mesons - 2015*, `1509.02220`.

[32] T. Bae, Y.-C. Jang, H. Jeong, J. Kim, J. Kim et al., *Beyond the Standard Model B-parameters with improved staggered fermions in $N_f = 2 + 1$ QCD*, `1310.7372`.

[33] ETM collaboration, N. Carrasco et al., *K and D oscillations in the Standard Model and its extensions from $N_f = 2 + 1 + 1$ Twisted Mass LQCD*, `1310.5461`.

[34] A. J. Buras, M. Misiak and J. Urban, *Two loop QCD anomalous dimensions of flavor changing four quark operators within and beyond the standard model*, *Nucl. Phys.* **B586** (2000) 397–426, [`hep-ph/0005183`].

[35] A. J. Buras, *New Physics Patterns in ε'/ε and ε_K with Implications for Rare Kaon Decays and ΔM_K*, `1601.00005`.

[36] A. J. Buras, D. Guadagnoli and G. Isidori, *On ϵ_K Beyond Lowest Order in the Operator Product Expansion*, *Phys. Lett.* **B688** (2010) 309–313, [`1002.3612`].

[37] J. Brod and M. Gorbahn, *Next-to-Next-to-Leading-Order Charm-Quark Contribution to the CP Violation Parameter ϵ_K and ΔM_K*, *Phys. Rev. Lett.* **108** (2012) 121801, [`1108.2036`].

[38] C. W. Bauer, Z. Ligeti, M. Luke, A. V. Manohar and M. Trott, *Global analysis of inclusive B decays*, *Phys.Rev.* **D70** (2004) 094017, [`hep-ph/0408002`].

[39] M. Czakon, P. Fiedler, T. Huber, M. Misiak, T. Schutzmeier and M. Steinhauser, *The (Q_7 , $Q_{1,2}$) contribution to $\overline{B} \to X_s \gamma$ at $\mathcal{O}(\alpha_s^2)$*, *JHEP* **04** (2015) 168, [`1503.01791`].

[40] M. Misiak and M. Steinhauser, *NNLO QCD corrections to the $\overline{B} \to X_s \gamma$ matrix elements using interpolation in m_c*, *Nucl.Phys.* **B764** (2007) 62–82, [`hep-ph/0609241`].

[41] G. Buchalla and A. J. Buras, *Two loop large m_t electroweak corrections to $K \to \pi \nu \overline{\nu}$ for arbitrary Higgs boson mass*, *Phys. Rev.* **D57** (1998) 216–223, [`hep-ph/9707243`].

[42] C. Bobeth, M. Misiak and J. Urban, *Photonic penguins at two loops and m_t dependence of $BR[B \to X_s\ell^+\ell^-]$*, *Nucl. Phys.* **B574** (2000) 291–330, [hep-ph/9910220].

[43] P. Gambino and M. Misiak, *Quark mass effects in $\overline{B} \to X_s\gamma$*, *Nucl. Phys.* **B611** (2001) 338–366, [hep-ph/0104034].

[44] K. Bieri, C. Greub and M. Steinhauser, *Fermionic NNLL corrections to $b \to s\gamma$*, *Phys. Rev.* **D67** (2003) 114019, [hep-ph/0302051].

[45] T. Huber, E. Lunghi, M. Misiak and D. Wyler, *Electromagnetic logarithms in $\overline{B} \to X_s\ell^+\ell^-$*, *Nucl. Phys.* **B740** (2006) 105–137, [hep-ph/0512066].

[46] H. M. Asatrian, T. Ewerth, H. Gabrielyan and C. Greub, *Charm quark mass dependence of the electromagnetic dipole operator contribution to $\overline{B} \to X_s\gamma$ at $O(\alpha_s^2)$*, *Phys. Lett.* **B647** (2007) 173–178, [hep-ph/0611123].

[47] M. Czakon, U. Haisch and M. Misiak, *Four-Loop Anomalous Dimensions for Radiative Flavour-Changing Decays*, *JHEP* **03** (2007) 008, [hep-ph/0612329].

[48] R. Boughezal, M. Czakon and T. Schutzmeier, *NNLO fermionic corrections to the charm quark mass dependent matrix elements in $\overline{B} \to X_s\gamma$*, *JHEP* **09** (2007) 072, [0707.3090].

[49] T. Ewerth, *Fermionic corrections to the interference of the electro- and chromomagnetic dipole operators in $\overline{B} \to X_s\gamma$ at $O(\alpha_s^2)$*, *Phys. Lett.* **B669** (2008) 167–172, [0805.3911].

[50] H. M. Asatrian, T. Ewerth, A. Ferroglia, C. Greub and G. Ossola, *Complete (O_7, O_8) contribution to $B \to X_s\gamma$ at order α_s^2*, *Phys. Rev.* **D82** (2010) 074006, [1005.5587].

[51] A. Ferroglia and U. Haisch, *Chromomagnetic Dipole-Operator Corrections in $\overline{B} \to X_s\gamma$ at $O(\beta_0\alpha_s^2)$*, *Phys. Rev.* **D82** (2010) 094012, [1009.2144].

[52] M. Misiak and M. Poradzinski, *Completing the Calculation of BLM corrections to $\overline{B} \to X_s\gamma$*, *Phys. Rev.* **D83** (2011) 014024, [1009.5685].

[53] M. Kaminski, M. Misiak and M. Poradzinski, *Tree-level contributions to $B \to X_s\gamma$*, *Phys. Rev.* **D86** (2012) 094004, [1209.0965].

[54] T. Huber, M. Poradzinski and J. Virto, *Four-body contributions to $\overline{B} \to X_s\gamma$ at NLO*, *JHEP* **01** (2015) 115, [`1411.7677`].

[55] T. Ewerth, P. Gambino and S. Nandi, *Power suppressed effects in $\overline{B} \to X_s\gamma$ at $O(\alpha_s)$*, *Nucl. Phys.* **B830** (2010) 278–290, [`0911.2175`].

[56] M. Benzke, S. J. Lee, M. Neubert and G. Paz, *Factorization at Subleading Power and Irreducible Uncertainties in $\overline{B} \to X_s\gamma$ Decay*, *JHEP* **08** (2010) 099, [`1003.5012`].

[57] A. Alberti, P. Gambino and S. Nandi, *Perturbative corrections to power suppressed effects in semileptonic B decays*, *JHEP* **01** (2014) 147, [`1311.7381`].

[58] J. M. Soares, *CP violation in radiative b decays*, *Nucl. Phys.* **B367** (1991) 575–590.

[59] T. Hurth, E. Lunghi and W. Porod, *Untagged $\overline{B} \to X_{s+d}\gamma$ CP asymmetry as a probe for new physics*, *Nucl.Phys.* **B704** (2005) 56–74, [`hep-ph/0312260`].

[60] S. J. Lee, M. Neubert and G. Paz, *Enhanced Non-local Power Corrections to the $\overline{B} \to X_s\gamma$ Decay Rate*, *Phys. Rev.* **D75** (2007) 114005, [`hep-ph/0609224`].

[61] A. L. Kagan and M. Neubert, *Direct CP violation in $B \to X_s\gamma$ decays as a signature of new physics*, *Phys. Rev.* **D58** (1998) 094012, [`hep-ph/9803368`].

[62] A. Ali, H. Asatrian and C. Greub, *Inclusive decay rate for $B \to X_d + \gamma$ in next-to-leading logarithmic order and CP asymmetry in the standard model*, *Phys. Lett.* **B429** (1998) 87–98, [`hep-ph/9803314`].

[63] T. Nishikawa and K. Tanaka, *QCD Sum Rules for Quark-Gluon Three-Body Components in the B Meson*, *Nucl. Phys.* **B879** (2014) 110–142, [`1109.6786`].

[64] A. J. Buras, D. Buttazzo, J. Girrbach-Noe and R. Knegjens, *$K^+ \to \pi^+\nu\overline{\nu}$ and $K_L \to \pi^0\nu\overline{\nu}$ in the Standard Model: status and perspectives*, *JHEP* **11** (2015) 033, [`1503.02693`].

[65] G. Buchalla and A. J. Buras, *$K \to \pi\nu\overline{\nu}$ and high precision determinations of the CKM matrix*, *Phys. Rev.* **D54** (1996) 6782–6789, [`hep-ph/9607447`].

[66] W. Marciano and Z. Parsa, *Rare kaon decays with 'missing energy'*, *Phys.Rev.* **D53** (1996) 1–5.

[67] G. Isidori, F. Mescia and C. Smith, *Light-quark loops in $K \to \pi\nu\overline{\nu}$*, *Nucl. Phys.* **B718** (2005) 319–338, [`hep-ph/0503107`].

[68] CKMFITTER GROUP collaboration, J. Charles, A. Hocker, H. Lacker, S. Laplace, F. R. Le Diberder, J. Malcles et al., *CP violation and the CKM matrix: Assessing the impact of the asymmetric B factories*, *Eur. Phys. J.* **C41** (2005) 1–131, [`hep-ph/0406184`].

[69] CKMFITTER GROUP collaboration, *CKMfitter – Summer 2015*, 2015.

[70] C. Bobeth, T. Ewerth, F. Kruger and J. Urban, *Enhancement of $\mathcal{B}(\overline{B}_d \to \mu^+\mu^-)/\mathcal{B}(\overline{B}_s \to \mu^+\mu^-)$ in the MSSM with minimal flavor violation and large tan β*, *Phys.Rev.* **D66** (2002) 074021, [`hep-ph/0204225`].

[71] A. Dedes, J. Rosiek and P. Tanedo, *Complete One-Loop MSSM Predictions for $B^0 \to \ell^+\ell'^-$ at the Tevatron and LHC*, *Phys.Rev.* **D79** (2009) 055006, [`0812.4320`].

[72] K. De Bruyn, R. Fleischer, R. Knegjens, P. Koppenburg, M. Merk, A. Pellegrino et al., *Probing New Physics via the $B_s^0 \to \mu^+\mu^-$ Effective Lifetime*, *Phys. Rev. Lett.* **109** (2012) 041801, [`1204.1737`].

[73] T. Hermann, M. Misiak and M. Steinhauser, *Three-loop QCD corrections to $B_s \to \mu^+\mu^-$*, *JHEP* **1312** (2013) 097, [`1311.1347`].

[74] C. Bobeth, M. Gorbahn and E. Stamou, *Electroweak Corrections to $B_{s,d} \to \ell^+\ell^-$*, *Phys. Rev.* **D89** (2014) 034023, [`1311.1348`].

[75] J. Hisano, J. Y. Lee, N. Nagata and Y. Shimizu, *Reevaluation of Neutron Electric Dipole Moment with QCD Sum Rules*, *Phys. Rev.* **D85** (2012) 114044, [`1204.2653`].

[76] G. Degrassi, E. Franco, S. Marchetti and L. Silvestrini, *QCD corrections to the electric dipole moment of the neutron in the MSSM*, *JHEP* **11** (2005) 044, [`hep-ph/0510137`].

[77] E. Mereghetti, J. de Vries, W. H. Hockings, C. M. Maekawa and U. van Kolck, *The Electric Dipole Form Factor of the Nucleon in Chiral Perturbation Theory to Sub-leading Order*, *Phys. Lett.* **B696** (2011) 97–102, [`1010.4078`].

[78] F.-K. Guo and U.-G. Meissner, *Baryon electric dipole moments from strong CP violation*, *JHEP* **12** (2012) 097, [`1210.5887`].

[79] E. Shintani, S. Aoki, N. Ishizuka, K. Kanaya, Y. Kikukawa, Y. Kuramashi et al., *Neutron electric dipole moment from lattice QCD*, *Phys. Rev.* **D72** (2005) 014504, [`hep-lat/0505022`].

[80] F. Berruto, T. Blum, K. Orginos and A. Soni, *Calculation of the neutron electric dipole moment with two dynamical flavors of domain wall fermions*, *Phys. Rev.* **D73** (2006) 054509, [`hep-lat/0512004`].

[81] E. Shintani, S. Aoki, N. Ishizuka, K. Kanaya, Y. Kikukawa, Y. Kuramashi et al., *Neutron electric dipole moment with external electric field method in lattice QCD*, *Phys. Rev.* **D75** (2007) 034507, [`hep-lat/0611032`].

[82] E. Shintani, S. Aoki and Y. Kuramashi, *Full QCD calculation of neutron electric dipole moment with the external electric field method*, *Phys. Rev.* **D78** (2008) 014503, [`0803.0797`].

[83] C. Alexandrou, A. Athenodorou, M. Constantinou, K. Hadjiyiannakou, K. Jansen, G. Koutsou et al., *Neutron electric dipole moment using N_f=2+1+1 twisted mass fermions*, `1510.05823`.

[84] E. Shintani, T. Blum, T. Izubuchi and A. Soni, *Neutron and proton electric dipole moments from $N_f = 2 + 1$ domain-wall fermion lattice QCD*, `1512.00566`.

[85] M. Pospelov and A. Ritz, *Neutron EDM from electric and chromoelectric dipole moments of quarks*, *Phys. Rev.* **D63** (2001) 073015, [`hep-ph/0010037`].

[86] S. Weinberg, *Larger Higgs Exchange Terms in the Neutron Electric Dipole Moment*, *Phys. Rev. Lett.* **63** (1989) 2333.

[87] J. Hisano, M. Nagai and P. Paradisi, *New Two-loop Contributions to Hadronic EDMs in the MSSM*, *Phys. Lett.* **B642** (2006) 510–517, [`hep-ph/0606322`].

[88] A. Crivellin, S. Najjari and J. Rosiek, *Lepton Flavor Violation in the Standard Model with general Dimension-Six Operators*, *JHEP* **04** (2014) 167, [`1312.0634`].

[89] S. Davidson, $\mu \to e\gamma$ and matching at m_W, `1601.07166`.

[90] Y. Kuno and Y. Okada, $\mu \to e\gamma$ *search with polarized muons*, *Phys. Rev. Lett.* **77** (1996) 434–438, [`hep-ph/9604296`].

[91] B. Allanach, C. Balazs, G. Belanger, M. Bernhardt, F. Boudjema et al., *SUSY Les Houches Accord 2*, *Comput.Phys.Commun.* **180** (2009) 8–25, [`0801.0045`].

[92] J. A. Evans and D. Shih, *Surveying Extended GMSB Models with m_h=125 GeV*, *JHEP* **08** (2013) 093, [`1303.0228`].

[93] J. A. Evans, D. Shih and A. Thalapillil, *Chiral Flavor Violation from Extended Gauge Mediation*, *JHEP* **07** (2015) 040, [`1504.00930`].

[94] D. M. Pierce, J. A. Bagger, K. T. Matchev and R.-j. Zhang, *Precision corrections in the minimal supersymmetric standard model*, *Nucl.Phys.* **B491** (1997) 3–67, [`hep-ph/9606211`].

[95] F. Staub, *Automatic Calculation of supersymmetric Renormalization Group Equations and Self Energies*, *Comput. Phys. Commun.* **182** (2011) 808–833, [`1002.0840`].

[96] W. Porod and F. Staub, *SPheno 3.1: Extensions including flavour, CP-phases and models beyond the MSSM*, *Comput. Phys. Commun.* **183** (2012) 2458–2469, [`1104.1573`].

[97] F. Mahmoudi, *SuperIso v2.3: A Program for calculating flavor physics observables in Supersymmetry*, *Comput. Phys. Commun.* **180** (2009) 1579–1613, [`0808.3144`].

[98] U. Ellwanger, J. F. Gunion and C. Hugonie, *NMHDECAY: A Fortran code for the Higgs masses, couplings and decay widths in the NMSSM*, *JHEP* **02** (2005) 066, [`hep-ph/0406215`].

[99] G. Belanger, F. Boudjema, A. Pukhov and A. Semenov, *microOMEGAs_3: A program for calculating dark matter observables*, *Comput. Phys. Commun.* **185** (2014) 960–985, [`1305.0237`].

[100] G. Degrassi, P. Gambino and P. Slavich, *SusyBSG: A Fortran code for $BR[B \to X_s\gamma]$ in the MSSM with Minimal Flavor Violation*, *Comput. Phys. Commun.* **179** (2008) 759–771, [`0712.3265`].

[101] B. Murakami, *SuperLFV: An SLHA tool for lepton flavor violating observables in supersymmetric models*, *Comput. Phys. Commun.* **185** (2014) 622–637, [`1302.4469`].

[102] D. Chowdhury, R. Garani and S. K. Vempati, *SuSeFLAV: Program for supersymmetric mass spectra with seesaw mechanism and rare lepton flavor violating decays*, *Comput. Phys. Commun.* **184** (2013) 899–918, [`1109.3551`].

[103] F. E. Paige, S. D. Protopopescu, H. Baer and X. Tata, *ISAJET 7.69: A Monte Carlo event generator for pp, $\bar{p}p$, and e^+e^- reactions*, `hep-ph/0312045`.

[104] D. Straub.

[105] A. Crivellin, L. Hofer and J. Rosiek, *Complete resummation of chirally-enhanced loop-effects in the MSSM with non-minimal sources of flavor-violation*, *JHEP* **07** (2011) 017, [`1103.4272`].

[106] K. G. Chetyrkin, M. Misiak and M. Munz, *Weak radiative B meson decay beyond leading logarithms*, *Phys. Lett.* **B400** (1997) 206–219, [`hep-ph/9612313`].

[107] E. Lunghi and J. Matias, *Huge right-handed current effects in $B \to K^*(K\pi)\ell^+\ell^-$ in supersymmetry*, *JHEP* **04** (2007) 058, [`hep-ph/0612166`].

[108] F. Gabbiani, E. Gabrielli, A. Masiero and L. Silvestrini, *A Complete analysis of FCNC and CP constraints in general SUSY extensions of the standard model*, *Nucl.Phys.* **B477** (1996) 321–352, [`hep-ph/9604387`].

Index