Computing tools for the SMEFT

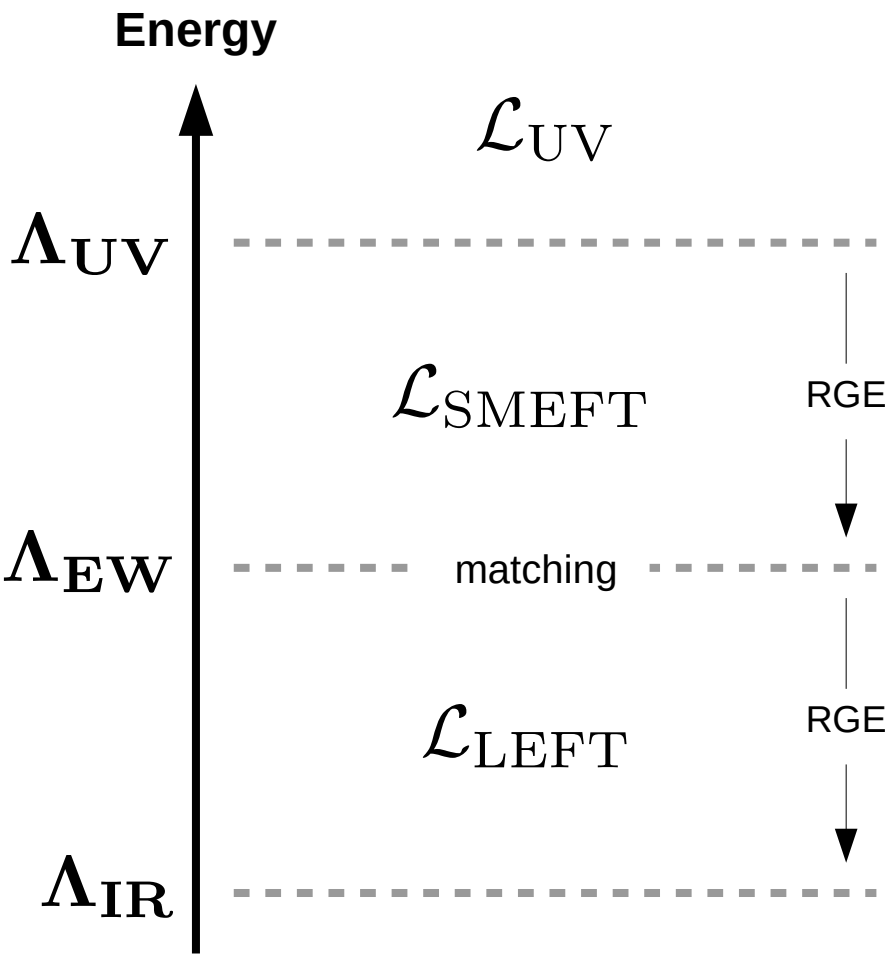
- RGE running and SMEFT-LEFT matching -

Avelino Vicente
IFIC – CSIC / U. Valencia

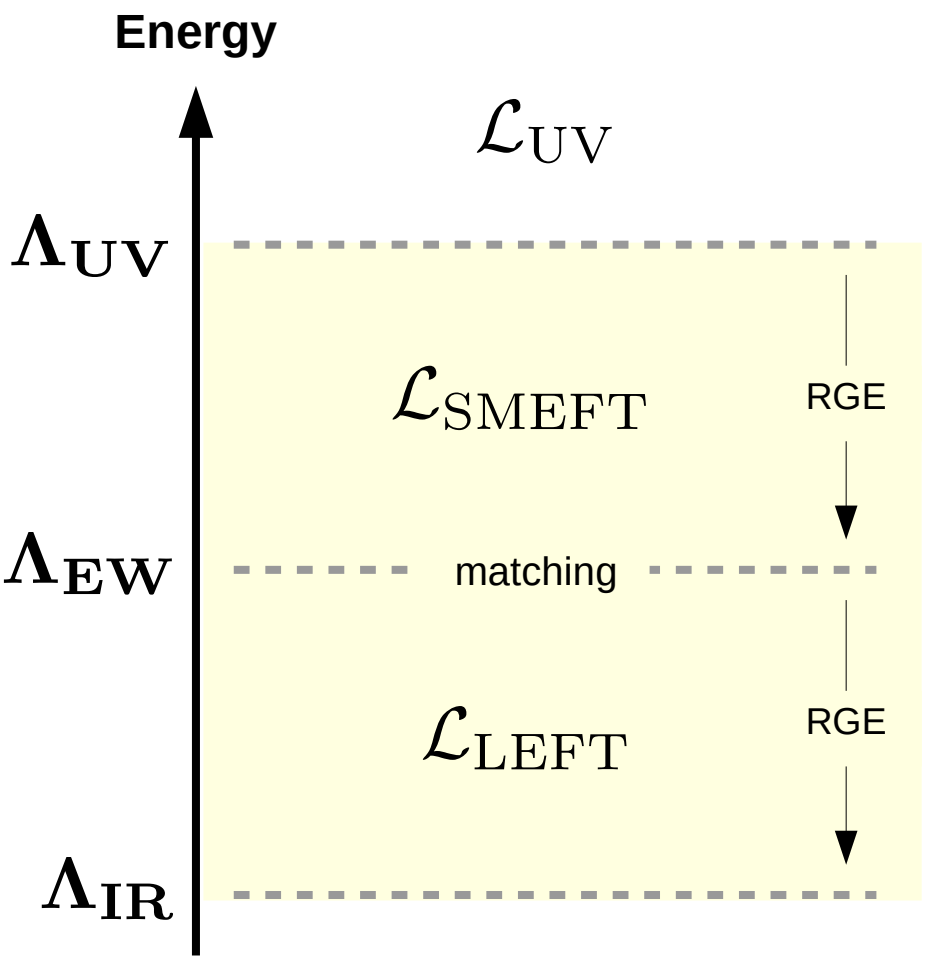
SMEFT'2022 Physics School

Siegen
July 11-15 2022

Problem we want to solve



Problem we want to solve



SMEFT

Warsaw basis

[Grzadkowski, Iskrzynski, Misiak, Rosiek, 2010]

Full 1-loop RGEs

[Alonso, Chang, Jenkins, Manohar, Shotwell, Trott, 2013-2014]

Full 1-loop matching

[Jenkins, Manohar, Stoffer, 2017]

[Dekens, Stoffer, 2019]

LEFT

San Diego basis

Full 1-loop RGEs

[Jenkins, Manohar, Stoffer, 2017]

DsixTools

DsixTools is a Mathematica package for the matching and RGE evolution from the new physics scale to the scale of low energy observables.

Alejandro Celis, Javier Fuentes-Martín,
Pedro Ruiz-Femenía, Avelino Vicente, Javier Virto

- https://dsixtools.github.io/
- arXiv:1704.04504 (version **1.0**) and arXiv:2010.16341 (version **2.0**)
- Mathematica package
- Installation: manual or automatic (recommended)
- It requires **Mathematica** 9+
- Alternative: **wilson**

What DsixTools can do for you

- Full 1-loop RGE running in the SMEFT and the LEFT

  Higher order for the SM parameters (5-loops for g_s, for instance)

- Full 1-loop SMEFT-LEFT matching
- Analytical and numerical routines
- Consistency check for SMEFT or LEFT parameter points
- Many useful resources for the study of the SMEFT and the LEFT

Current version: 2.0.1

$S_{iege}N$ model parameter point

$$M_N = 1\,\text{TeV} \qquad M_S = 1.2\,\text{TeV} \qquad \lambda_2 = \lambda_3 = 0.1$$

$$Y_N = \begin{pmatrix} 0.001 & 0.4 & -0.02 \end{pmatrix}$$

$$Y_S = \begin{pmatrix} -0.7 & 0.9 & -0.8 \end{pmatrix}$$

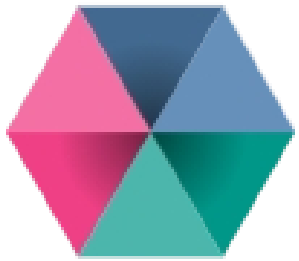
Note: phenomenologically excluded, as you will see, but illustrative

Chuck Norris fact of the day

Chuck Norris can kill two stones with one bird

Tomorrow

Lecture 4 : Observable calculation



flavio is a python package for flavour physics and other precision tests of the Standard Model.

David M. Straub + large community

(~21 contributors, including Peter Stangl, the main developer)

- https://flav-io.github.io/
- arXiv:1810.08132
- Python package

Backup slides

The SiegeN model

right-handed neutrino

	N	S
$\mathrm{SU(3)_c}$	**1**	**3**
$\mathrm{SU(2)_L}$	**1**	**2**
$\mathrm{U(1)_Y}$	0	$\frac{1}{6}$
GENERATIONS	1	1

scalar leptoquark

$$\mathcal{L} = \mathcal{L}_{\rm SM} + \mathcal{L}_{\rm NP}$$

$$\mathcal{L}_{\rm NP} = \mathcal{L}_N + \mathcal{L}_S + \mathcal{L}_{SH} + \mathcal{L}_{\rm Y}$$

$$\mathcal{L}_N = i\overline{N}\,\gamma_\mu D^\mu N - \frac{1}{2} M_N\,\overline{N^c}\,N$$

$$\mathcal{L}_S = D_\mu S^\dagger\, D^\mu S - M_S^2\, S^\dagger\, S$$

$$\mathcal{L}_{SH} = -\lambda_2\, H^\dagger H\, S^\dagger S - \lambda_3\, H^\dagger S\, S^\dagger H$$

$$\mathcal{L}_{\rm Y} = -Y_N^\alpha\,\overline{N}\,\ell_L^\alpha H - Y_S^\alpha\,\overline{q}_L^\alpha\, N S +$$

Input: making the user's life easier

Even after fixing the operator basis (Warsaw & San Diego) the user must choose a set of independent operators and make sure that the input values lead to a consistent Lagrangian. This could be tricky...

Examples of inconsistencies:

Hermiticity: $\left(C_{\ell q}^{(1)}\right)_{2223} \neq \left(C_{\ell q}^{(1)}\right)^*_{2232}$

Antisimmetry: $\left(L_{\nu\gamma}\right)_{11} \neq 0$

Nooo!

DsixTools 2.0...

- Accepts input values for the Wilson coefficients of **any** set of operators
- Fixes all consistency problems
- Transforms all Wilson coefficients to the 'symmetric basis'

Fast evolution matrix running

RGEs in **DsixTools 1.0**:

Method 1: NDSolve **Method 2:** leading log

RGEs in **DsixTools 2.0**:

Method 3: Evolution matrix

SMEFT RGEs

$d = 4$

$$\frac{d\hat{C}_i(t)}{dt} = \frac{1}{16\pi^2}\,\hat{\gamma}_{ij}(\hat{C}_k, C_k)\,\hat{C}_j(t) = \frac{1}{16\pi^2}\,\hat{\gamma}_{ij}(\hat{C}_k)\,\hat{C}_j(t) + \mathcal{O}(1/\Lambda^2)$$

$$\rightarrow \hat{C}_k^{\rm SM}(t)$$

$d = 6$

(analogous for d = 5)

$$\frac{dC_i(t)}{dt} = \frac{1}{16\pi^2}\,\gamma_{ij}(\hat{C}_k)\,C_j(t) = \frac{1}{16\pi^2}\,\gamma_{ij}(\hat{C}_k^{\rm SM})\,C_j(t) + \mathcal{O}(1/\Lambda^2)$$

$$\equiv \bar{\gamma}_{ij}(t)\,C_j(t) + \mathcal{O}(1/\Lambda^2)$$

$$C_i(t) = U_{ij}(t, t_0)\,C_j(t_0)$$