Rosetta: Mapping general HEFT via the Higgs Basis to Monte Carlo

B. Fuks, F. Maltoni, K. Mawatari, K. Mimasu, F. Riva, V. Sanz…

Higgs cross section working group meeting
26th February 2015

University of Sussex

Foreword

- Announcement/advertisement of a MC tool for HEFT
  - Model implementation for event generation
  - + Basis translation tool
  - For use by experimental and theoretical community
- HXSWG: forum for discussion
- We would like to present our idea
  - Comments/suggestions/criticisms/advice/…
  - Wishlist?
  - People interested in contributing
- Work in progress (only a few weeks old!)
- Feel free to get in touch

k.mimasu@sussex.ac.uk

Status

- In anticipation of LHC13 data…
- Higgs EFT in a generally realised spontaneous breaking of EW symmetry looks to be an important framework to have in place
- Along with a proposal on how to parametrise deviations from SM expectations, tools are needed to generate signal
  - For use by both experimentalists & theorists
- General enough in scope to meet needs of both communities by linking to:
  - Dimension 6 bases/UV completions
  - Pseudo-observables (relating to interactions of mass eigenstates)

Higgs Basis

- Higgs Basis
  - Provides a bridge between needs of theorists and experimentalists
- Divide operator basis into classes based on observables that are constrained by measurements of different precision
  - LEP1, precision EW observables measured at Z pole (~0.1%)
  - LEP2, off peak line shapes, TGCs (~1%)
  - LHC, Higgs signal strengths (~10%)

[Massó, Sanz: 1211.1320]
[Gupta, Pomarol, Riva: 1405.0181]
…

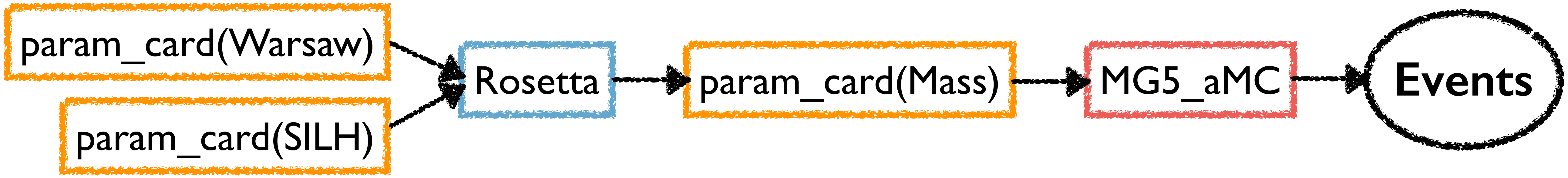
- Isolates current measurements of Higgs properties to a subset of relevant operators
  - Encode SU(2)xU(1) invariance by requiring certain relations between coefficients
  - Translate to other dim 6 bases such as Warsaw, SILH

Mass Basis

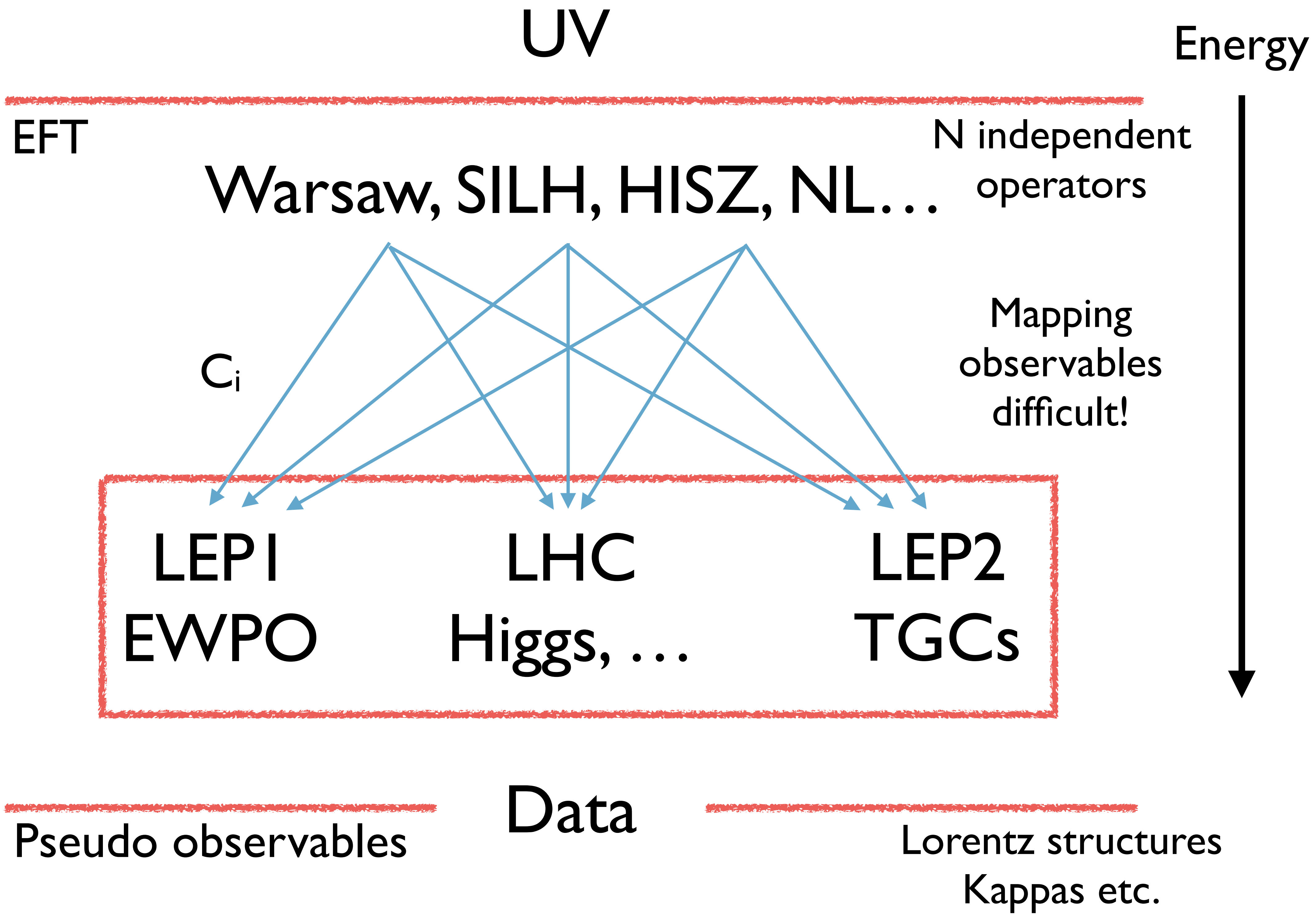
- HXSWG is working on a proposal to experimental collaborations using this basis
  - We have started working a tool to implement this in a MC-friendly way
- Our current idea:
  - Use the set of operators characterised by the Higgs Basis
  - Have a FeynRules/UFO implementation of these operators
  - Provide the possibility of multiple input formats in terms of coefficients of your favourite basis
- Aims to put all dim 6 bases on the same footing
  - Develop an additional 'translation' layer between the user defined coefficients in a given basis to a general implementation in terms of the 'redundant' Higgs Basis: Mass Basis

Mass Basis

- We will not assume any a-priori relationships between any of the coefficients (redundancy)
  - Allow theory users to encode the relations implicitly by calculating the values of the Mass Basis coefficients in terms of their chosen basis i.e. Warsaw (see mapping to dim 6 in proposal draft)
  - Experimentalists can simply pick the input format and generate signal
- Complete in terms of mass eigenstate interactions
- Rosetta: One tool serves all
  - Input/Output: LHA-style parameter card in an implemented model



HEFT Cartoon



UV

Energy

EFT

Warsaw, SILH, HISZ, NL…

N independent operators

C_i

Mapping observables difficult!

LEP1
EWPO

LHC
Higgs, …

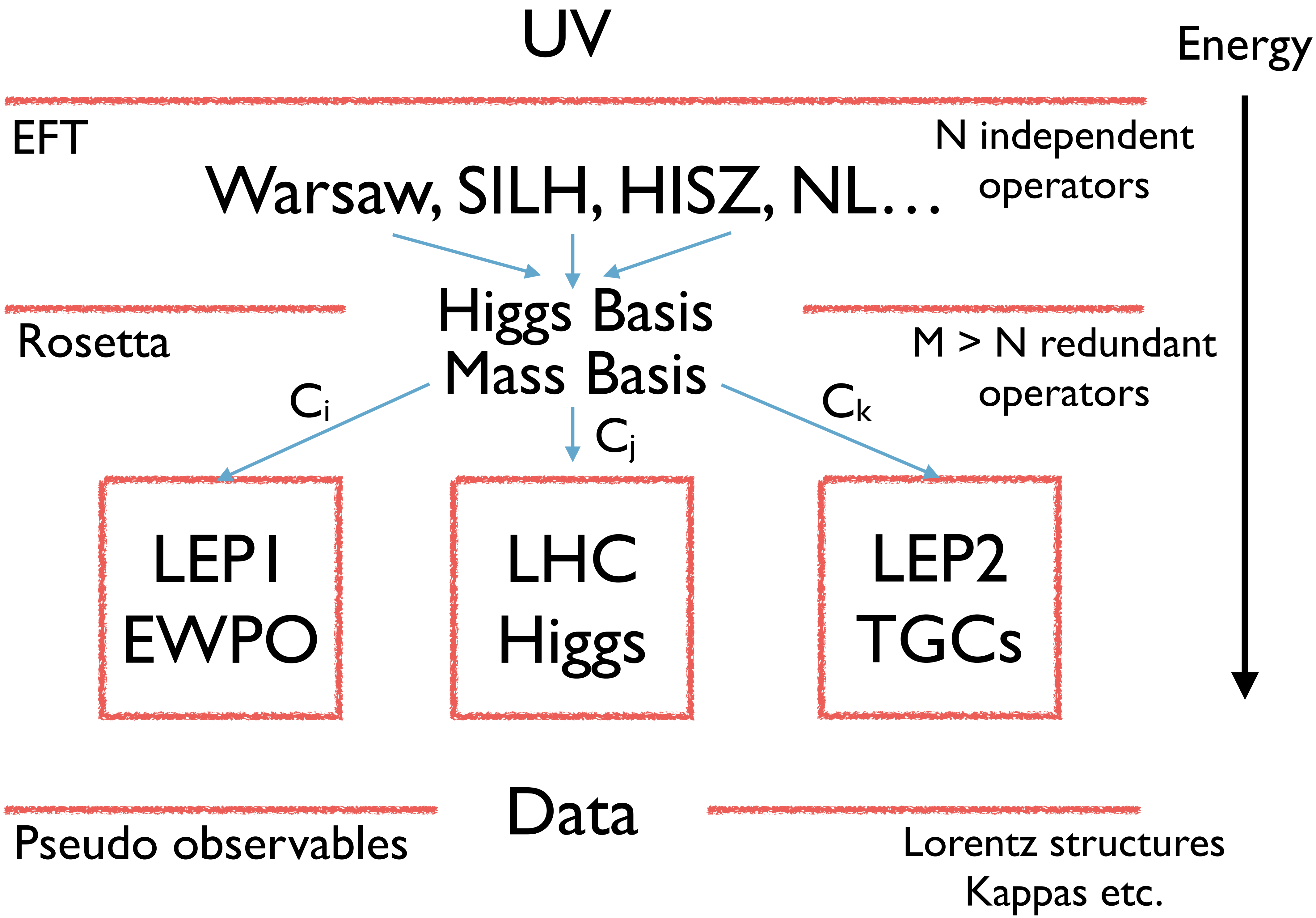
LEP2
TGCs

Data

Pseudo observables

Lorentz structures
Kappas etc.

HEFT Cartoon



UV

Energy

EFT

N independent operators

Warsaw, SILH, HISZ, NL…

Higgs Basis

Mass Basis

Rosetta

M > N redundant operators

C_i

C_j

C_k

LEP1 EWPO

LHC Higgs

LEP2 TGCs

Data

Pseudo observables

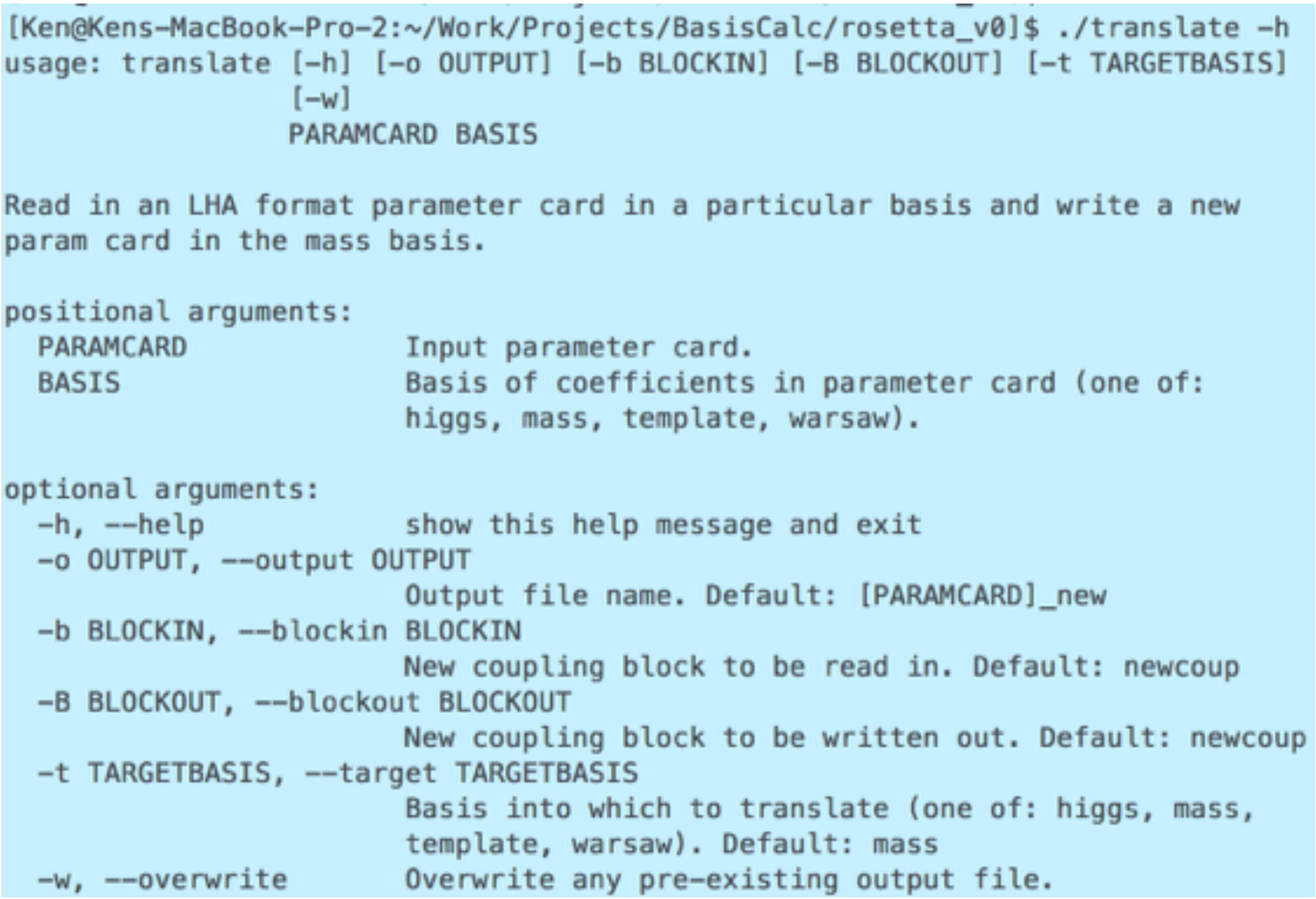
Lorentz structures
Kappas etc.

Rosetta

- Tries to avoid duplicating efforts
  - Implementing a translation should be less effort than implementing a full model
  - Some independent MC implementations needed for validation
- Simplify usage for non experts
  - ‘Community reviewed’ implemenations of EFT frameworks
  - Facilitate doing pheno and connecting to experiment
- Modular, user-implemented translations
  - Calculate dependent parameters, field/coupling redefinitions
  - Impose any further restrictions i.e. flavour structure/universality, custodial symmetry
  - Makes extensive theory user input/testing possible

Usage

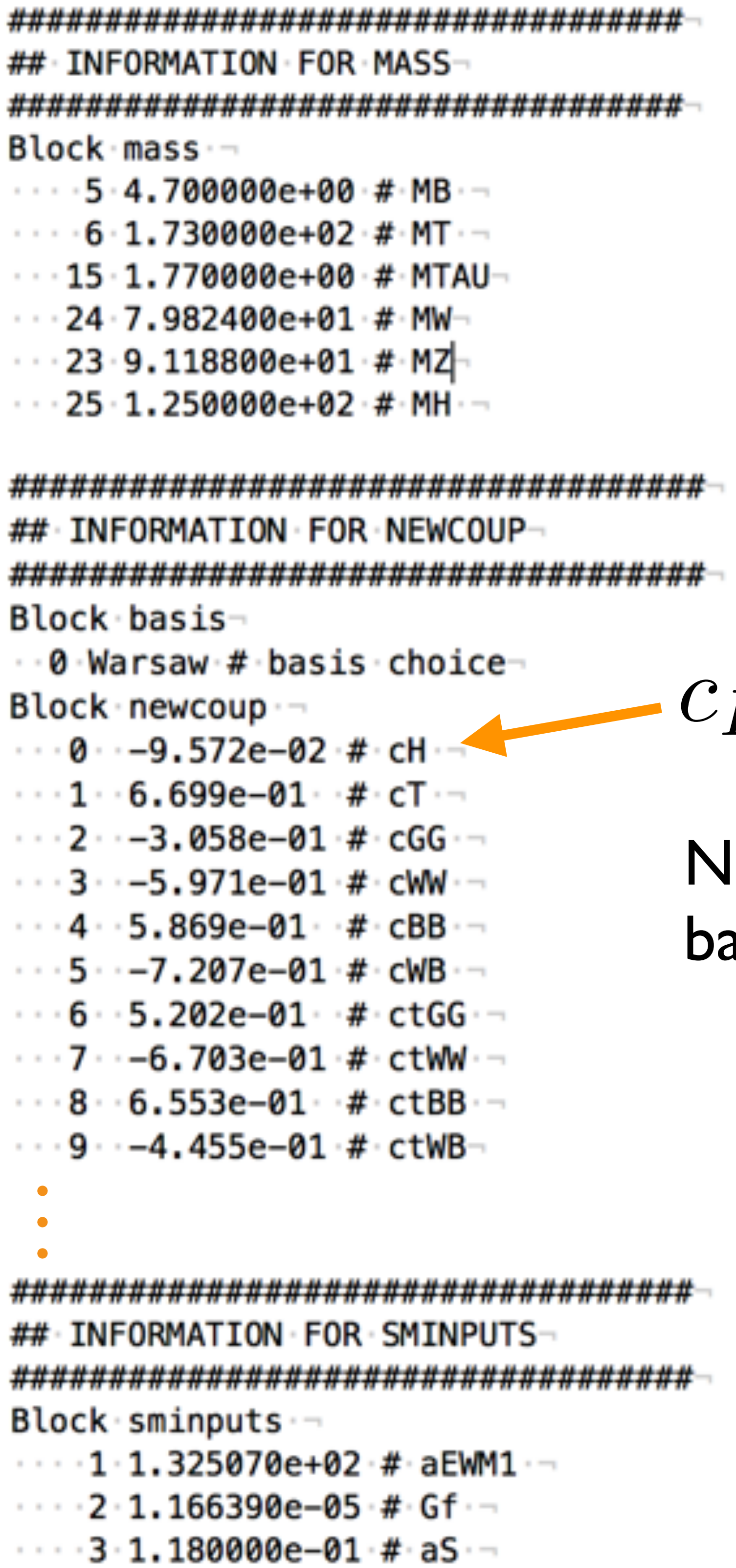
- Rosetta will ship with the Mass Basis FeynRules/UFO
  - Can also be used as a stand-alone program
- `./translate` : LHA card in → LHA card out



```
[Ken@Kens-MacBook-Pro-2:~/Work/Projects/BasisCalc/rosetta_v0]$ ./translate -h
usage: translate [-h] [-o OUTPUT] [-b BLOCKIN] [-B BLOCKOUT] [-t TARGETBASIS]
                 [-w]
                 PARAMCARD BASIS

Read in an LHA format parameter card in a particular basis and write a new
param card in the mass basis.

positional arguments:
  PARAMCARD             Input parameter card.
  BASIS                 Basis of coefficients in parameter card (one of:
                        higgs, mass, template, warsaw).

optional arguments:
  -h, --help            show this help message and exit
  -o OUTPUT, --output OUTPUT
                        Output file name. Default: [PARAMCARD]_new
  -b BLOCKIN, --blockin BLOCKIN
                        New coupling block to be read in. Default: newcoup
  -B BLOCKOUT, --blockout BLOCKOUT
                        New coupling block to be written out. Default: newcoup
  -t TARGETBASIS, --target TARGETBASIS
                        Basis into which to translate (one of: higgs, mass,
                        template, warsaw). Default: mass
  -w, --overwrite       Overwrite any pre-existing output file.
```

Example: input

param_card_WarsawBasis.dat



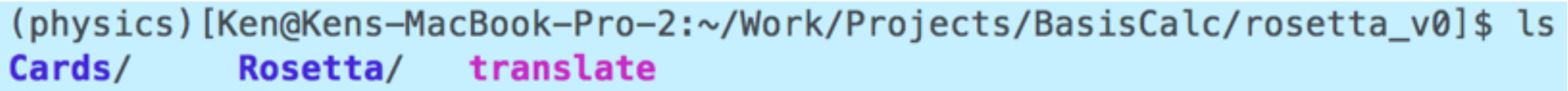
```
#####################################
## INFORMATION FOR MASS
#####################################
Block mass
    5 4.700000e+00 # MB
    6 1.730000e+02 # MT
   15 1.770000e+00 # MTAU
   24 7.982400e+01 # MW
   23 9.118800e+01 # MZ
   25 1.250000e+02 # MH

#####################################
## INFORMATION FOR NEWCOUP
#####################################
Block basis
  0 Warsaw # basis choice
Block newcoup
    0  -9.572e-02 # cH
    1  6.699e-01  # cT
    2  -3.058e-01 # cGG
    3  -5.971e-01 # cWW
    4  5.869e-01  # cBB
    5  -7.207e-01 # cWB
    6  5.202e-01  # ctGG
    7  -6.703e-01 # ctWW
    8  6.553e-01  # ctBB
    9  -4.455e-01 # ctWB
 ⋮

#####################################
## INFORMATION FOR SMINPUTS
#####################################
Block sminputs
    1 1.325070e+02 # aEWM1
    2 1.166390e-05 # Gf
    3 1.180000e-01 # aS
```

EFT input

$c_H\,,\ c_T\,,\ c_{GG}\,,\ \tilde{c}_{GG}\,,\ \ldots$

Names declared in basis implementation

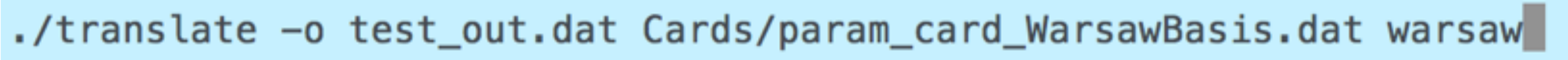
Example: translate



```
(physics)[Ken@Kens-MacBook-Pro-2:~/Work/Projects/BasisCalc/rosetta_v0]$ ls
Cards/      Rosetta/    translate
```

Python package

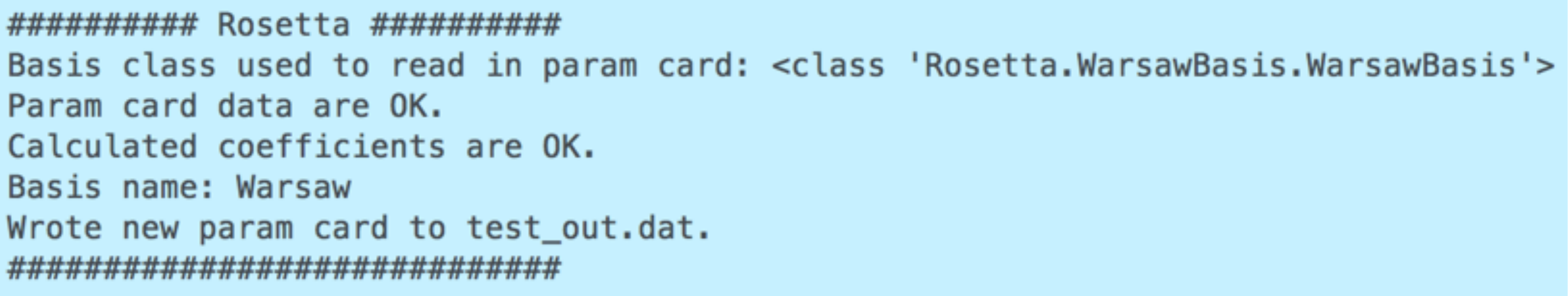
Command line script



```
./translate -o test_out.dat Cards/param_card_WarsawBasis.dat warsaw
```

Output

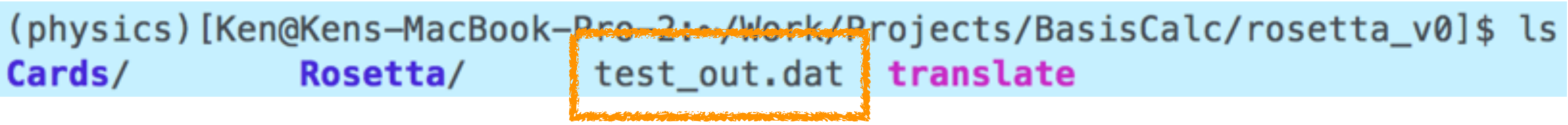
Basis name



```
########## Rosetta ##########
Basis class used to read in param card: <class 'Rosetta.WarsawBasis.WarsawBasis'>
Param card data are OK.
Calculated coefficients are OK.
Basis name: Warsaw
Wrote new param card to test_out.dat.
##############################
```

Some info/sanity checks:

Are all required inputs declared?
Are all dependent coefficients calculated?



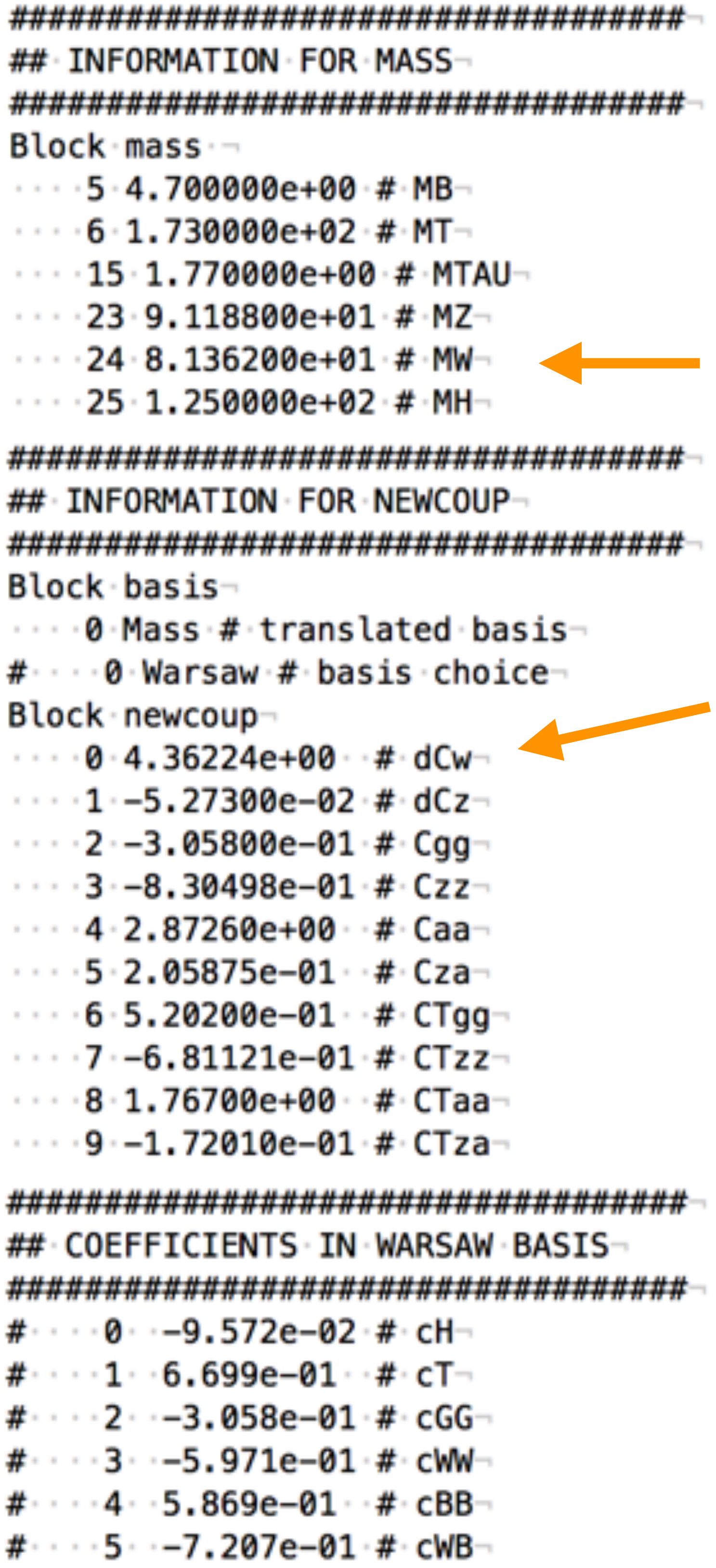
```
(physics)[Ken@Kens-MacBook-Pro-2:~/Work/Projects/BasisCalc/rosetta_v0]$ ls
Cards/         Rosetta/       test_out.dat  translate
```

Example: output

test_output.dat

Fit for use by the FeynRules/UFO implementation

Applies eqns. (4.5) - (4.15) in proposal draft



```
####################################
## INFORMATION FOR MASS
####################################
Block mass
    5 4.700000e+00 # MB
    6 1.730000e+02 # MT
    15 1.770000e+00 # MTAU
    23 9.118800e+01 # MZ
    24 8.136200e+01 # MW
    25 1.250000e+02 # MH
####################################
## INFORMATION FOR NEWCOUP
####################################
Block basis
    0 Mass # translated basis
#    0 Warsaw # basis choice
Block newcoup
    0 4.36224e+00  # dCw
    1 -5.27300e-02 # dCz
    2 -3.05800e-01 # Cgg
    3 -8.30498e-01 # Czz
    4 2.87260e+00  # Caa
    5 2.05875e-01  # Cza
    6 5.20200e-01  # CTgg
    7 -6.81121e-01 # CTzz
    8 1.76700e+00  # CTaa
    9 -1.72010e-01 # CTza
####################################
## COEFFICIENTS IN WARSAW BASIS
####################################
#    0  -9.572e-02 # cH
#    1  6.699e-01  # cT
#    2  -3.058e-01 # cGG
#    3  -5.971e-01 # cWW
#    4  5.869e-01  # cBB
#    5  -7.207e-01 # cWB
```

New W mass: $M_W + \delta m$

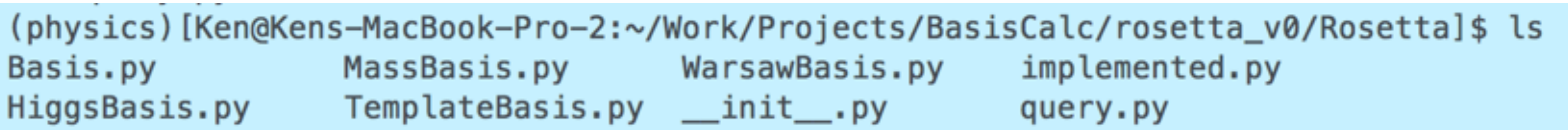
EFT output

$\delta c_z, \, \delta c_z, \, c_{gg}, \, \tilde{c}_{gg}, \, \ldots$

Names declared in Mass Basis implementation

Retains old input

Usage: implementing



```
(physics)[Ken@Kens-MacBook-Pro-2:~/Work/Projects/BasisCalc/rosetta_v0/Rosetta]$ ls
Basis.py           MassBasis.py       WarsawBasis.py     implemented.py
HiggsBasis.py      TemplateBasis.py   __init__.py        query.py
```

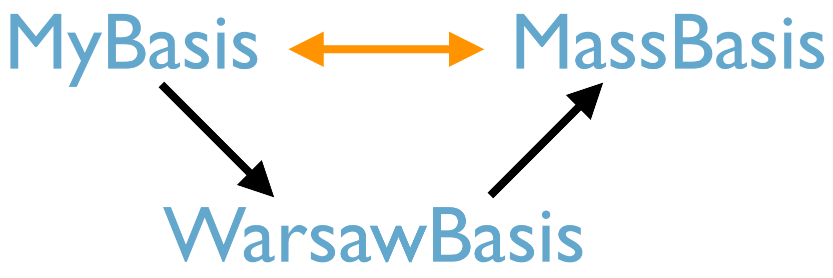
```
+ MyBasis.py
```

- Users can add their own basis to the module
- Requires a little bit of python
    - Declare independent [and dependent] coefficients
    - Declare required inputs (masses, EW parameters…)
    - Implement required functions:

        `translate()` [and `calculate_dependent()`]
    - Map coefficients to Mass Basis
- See Backup slides for more details on structure

Usage: implementing

- Existing implementation of Higgs Basis can be used to perform a few cross checks of output
  - Assuming MyBasis comes from a dim 6 linear SU(2)xU(1) realisation
  - Output of translation to Mass Basis → Input for Higgs Basis
  - Compare values dependent parameters



- Multi-step translation also possible
  - e.g. I already have a translation from my basis to Warsaw Basis
  - Implement those instead & use preexisting Warsaw Basis implementation to take me to the Mass Basis
  - One can even 'close the triangle' for additional consistency checks

More translation

- There is in principle no restriction to the direction of translation
  - Going ‘sideways’ between bases
- Going ‘up’ from mass basis
  - The LHC may give us constraints in terms of coefficients relating to pseudo-observables
  - Closer to Mass Basis rather than generic dim 6
  - The possibility of translating back to dim 6 bases may make the interpretation of constraints more efficient
- Optional choice of independent/dependent parameters
  - Users may want to have some freedom over this i.e. taking Cww as input rather than derived for studying WH associated production

In progress

$$\mathcal{L}_{\text{Higgs Basis}} = \boxed{\mathcal{L}^{\text{SM}} + \mathcal{L}^{(1)}_{\text{ewpt}} + \mathcal{L}^{(2)}_{\text{ewpt}} + \mathcal{L}_{\text{hff}} + \mathcal{L}^{(1)}_{\text{hvv}} + \mathcal{L}^{(2)}_{\text{hvv}} + \mathcal{L}_{hvff}} + \mathcal{L}_{\text{other}}$$

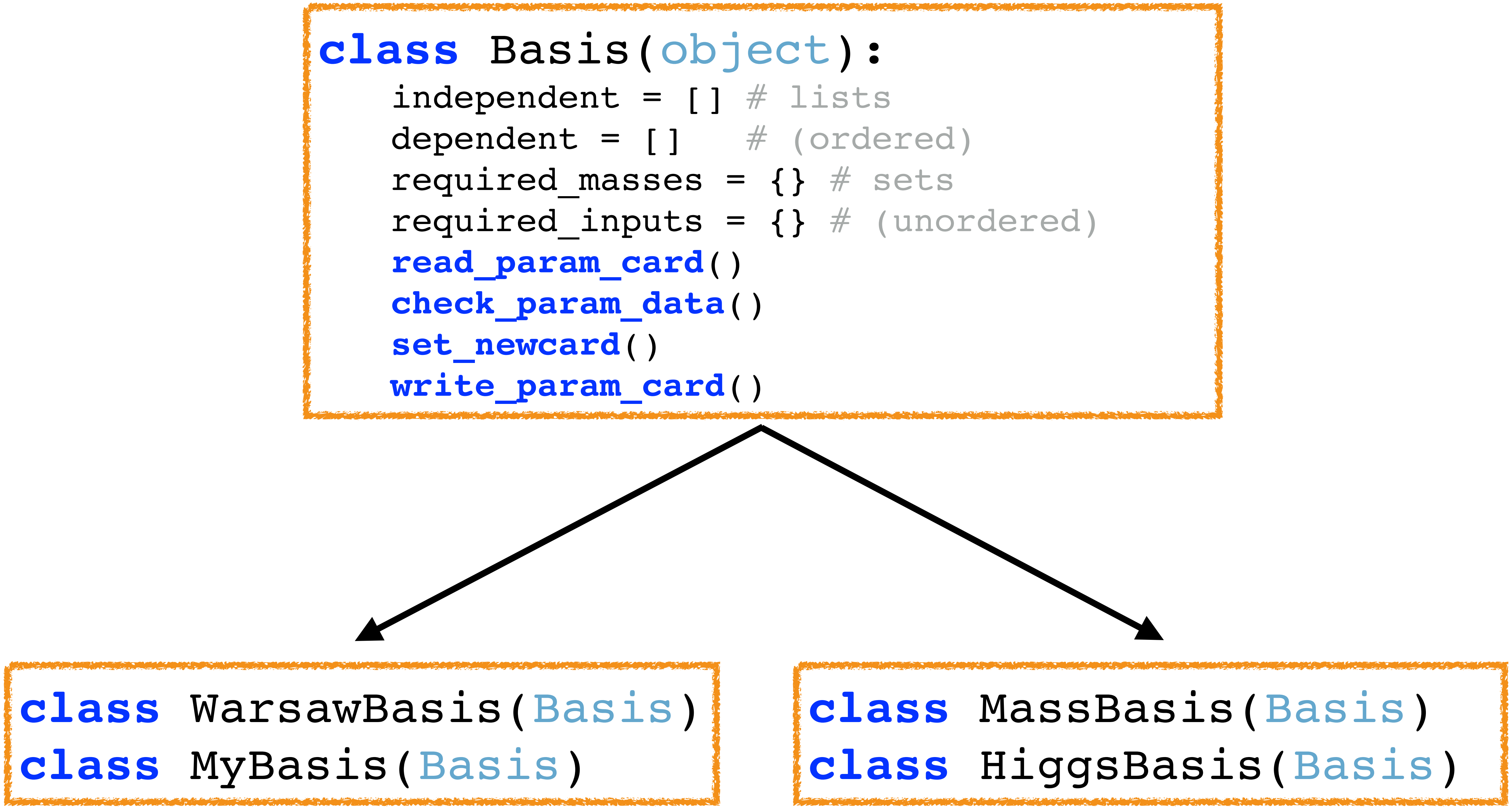
- Working on the model implementation derived from HC model (Maltoni et al.)
  - The full set of operators in the Higgs Basis proposal including ‘L_{other}’ is large
  - So far have restricted ourselves to the content of the draft version circulated at the last meeting
  - At validation stage for this subset of operators
- Already have at least one dim 6 implementation for validation, HEL model (Alloul et al.)
- Basic version of the Rosetta using info in proposal
  - Higgs Basis: calculation of dependent parameters
  - Warsaw Basis: translation to the Mass Basis

NLO?

- Once we have a complete LO framework
  - Possible to upgrade to NLO in QCD/EW
  - Add counterterms with help from a dim 6 SU(2)xU(1) invariant implementation (NLOCT…)
  - Upgrade Rosetta to modify counterterms in UFO
- Future work

Thank you

Base class



```
class Basis(object):
    independent = [] # lists
    dependent = []   # (ordered)
    required_masses = {} # sets
    required_inputs = {} # (unordered)
    read_param_card()
    check_param_data()
    set_newcard()
    write_param_card()
```

```
class WarsawBasis(Basis)
class MyBasis(Basis)
```

```
class MassBasis(Basis)
class HiggsBasis(Basis)
```

Base methods

```
class Basis(object)

self.par_dict = OrderedDict()
self.coeffs = namedtuple()
…
read_param_card() # fills these
self.par_dict, self.coeffs, self.input, self.masses, self.name, self.card
check_param_data() # check consistency
self.par_dict ↔ (self.independent, self.dependent, self.required_masses,
self.required_inputs)
set_newcard()
self.newcard = self.card ← self.newpar, self.newmasses
write_param_card()
```

Basic methods & members are under the hood, inherited by child basis classes

HiggsBasis

```
class HiggsBasis(Basis)

independent = [‘dCw’,‘dCz’,‘Czz’,‘Cgg’,…]
dependent   = [‘Cww’,‘CTww’,‘CLzu11’,‘CRWq23’,…]
required_masses = {1,2,3,4,5,6,11,12,13,14,15,16} # PIDs
required_inputs = {‘aEWM1’,‘MZ’,‘Gf’}
calculate_dependent():
self.dependent ← self.independent # eqns. (3.7), (3.9), (3.11) in draft
translate():
    NotImplemented
```

Basis & relations described in proposal draft:
Can be used as partial cross check for user implemented linear dimension 6 bases

MassBasis

```
class MassBasis(Basis)

independent = HiggsBasis.independent+HiggsBasis.dependent
calculate_dependent():
   NotImplemented
translate():
   NotImplemented
```

‘Container’ class for target coefficients:
User implemented `translate()` method in
MyBasis can create one of these instances to fill

MyBasis

```
class MyBasis(Basis)

independent = [‘A’,’B’,’C’,…] # My coefficients
dependent   = [‘D’,’E’,‘F’,…]
required_masses = {1,2,3,4,5,…} # PIDs
required_inputs = {‘aEWM1’,’MZ’,‘Gf’,…}
calculate_dependent():
self.dependent ← self.independent
translate():
M = MassBasis().coeffs._asdict() # Empty MassBasis instance
M[‘Czz’] = self.coeffs.A         # Fill
M[‘dCz’] = self.coeffs.as_dict()[’B’]
M[‘dM’] = self.par_dict[’C’]
…
self.newpar = M
useful_function():
   # Anything you like (providing it doesn’t override exiting functions)
```