BasisGen: counting EFT operators

Juan Carlos Criado

SMEFT-Tools 2019

Based on: `1901.03501`

EFT prescription

Include all local operators allowed by the symmetries

Redundancies

- Group-theory identities
- Integration by parts
- Field redefinitions (EOM)

Basis

Complete set of independent operators

How many independent invariant operators with a given field content?

$$N(\phi_1^{e_1}\phi_2^{e_2}\cdots\phi_n^{e_n}D^d)$$

Other approaches

Hilbert series

$$H(\phi_1, \ldots, \phi_n, D) = \sum_{e_1, \ldots e_n, d} N(\phi_1^{e_1} \cdots \phi_n^{e_n} D^d)\, \phi_1^{e_1} \cdots \phi_n^{e_n} D^d$$

Computer tools (other methods): `DEFT`, `abc_eft`

BasisGen's approach

Classical methods (roots, weights...)

- General symmetry: semisimple$\times$abelian
- *Very* fast
- Less requirements

Counting group theory invariants

Plan

1. Compute tensor product: $\phi_1^{e_1} \otimes \ldots \otimes \phi_n^{e_n}$
2. Decompose it into sum of irreps
3. Count # of singlets

Weight/root language

state	weight $(\lambda_1, \ldots, \lambda_n)$
raising/lowering	root $(\alpha_1, \ldots, \alpha_n)$
representation	weight system
irrep	highest weight

```
>>> from basisgen import *
>>> irrep('SU3', '1 0').weights_view()
(1 0)
(-1 1)
(0 -1)
>>> irrep('SU3', '0 1').weights_view()
(0 1)
(1 -1)
(-1 0)
```

```
>>> irrep('SU3', '1 1').weights_view()
    (1 1)
(2 -1) (-1 2)
 (0 0) (0 0)
(1 -2) (-2 1)
   (-1 -1)
>>> irrep('SU3', '2 0').weights_view()
    (2 0)
    (0 1)
(1 -1) (-2 2)
    (-1 0)
    (0 -2)
```

```
>>> irrep('E6', '1 0 0 0 0 0').weights_view()
                 (1 0 0 0 0 0)
                (-1 1 0 0 0 0)
                (0 -1 1 0 0 0)
                (0 0 -1 1 0 1)
       (0 0 0 -1 1 1) (0 0 0 1 0 -1)
      (0 0 0 0 -1 1) (0 0 1 -1 1 -1)
      (0 1 -1 0 1 0) (0 0 1 0 -1 -1)
      (0 1 -1 1 -1 0) (1 -1 0 0 1 0)
(0 1 0 -1 0 0) (-1 0 0 0 1 0) (1 -1 0 1 -1 0)
      (-1 0 0 1 -1 0) (1 -1 1 -1 0 0)
      (1 0 -1 0 0 1) (-1 0 1 -1 0 0)
                     ...
```

Tensor products

Product:

$$\{\lambda_i\}_{i\in I} \otimes \{\omega_j\}_{j\in J} = \{\lambda_i + \omega_j\}_{(i,j)\in I\times J}$$

Symmetric power:

$$\mathrm{Sym}^n\left(\{\lambda_i\}_{i\in I}\right) = \{\lambda_{i_1} + \cdots + \lambda_{i_n}\}_{i_k \le i_{k+1}}$$

Antisymmetric power:

$$\Lambda^n\left(\{\lambda_i\}_{i\in I}\right) = \{\lambda_{i_1} + \cdots + \lambda_{i_n}\}_{i_k < i_{k+1}}$$

Reducible representation decomposition

Repeat:

1. Find highest weight Λ
2. Compute irrep(Λ)
3. Remove all $\lambda \in$ irrep(Λ)

```
>>> triplet = irrep('SU3', '1 0')
>>> triplet * triplet
[2 0] + [0 1]
>>> triplet * triplet * triplet
[3 0] + 2 [1 1] + [0 0]
>>> triplet.power(2, boson)
[2 0]
>>> triplet.power(2, fermion)
[0 1]
```

Slansky

“Group theory for unified model building”

Phys. Rept. 79 (1981) 1

Taking derivatives into account

Same as before, but with fields:

$$\phi, D_\mu\phi, D_\mu D_\nu\phi, D_\mu D_\nu D_\rho\phi, \ldots$$

Equations of motion:

$$\phi, D_\mu\phi, D^2_{\{\mu\nu\}}\phi, D^3_{\{\mu\nu\rho\}}\phi, \ldots$$

Integration by parts: example

Henning, Lu, Melia, Murayama (1512.03433)

$$\mathcal{O} = H^\dagger (D_\nu H)(D_\nu H)^\dagger H\, B_L^{\mu\nu}, \qquad \mathcal{Q} = (D_\mu H)^\dagger (D_\nu H)(H^\dagger H) B_L^{\mu\nu}.$$

Integration by parts: example

Henning, Lu, Melia, Murayama (1512.03433)

$$\mathcal{O} = H^\dagger (D_\nu H)(D_\nu H)^\dagger H\, B_L^{\mu\nu}, \qquad \mathcal{Q} = (D_\mu H)^\dagger (D_\nu H)(H^\dagger H) B_L^{\mu\nu}.$$

$$A_\mu = H^\dagger (D_\nu H)(H^\dagger H) B_L^{\mu\nu}, \qquad B_\mu = (D_\nu H)^\dagger H (H^\dagger H) B_L^{\mu\nu}.$$

Integration by parts: example

Henning, Lu, Melia, Murayama (1512.03433)

$$\mathcal{O} = H^\dagger (D_\nu H)(D_\nu H)^\dagger H\, B_L^{\mu\nu}, \qquad \mathcal{Q} = (D_\mu H)^\dagger (D_\nu H)(H^\dagger H) B_L^{\mu\nu}.$$

$$A_\mu = H^\dagger (D_\nu H)(H^\dagger H) B_L^{\mu\nu}, \qquad B_\mu = (D_\nu H)^\dagger H (H^\dagger H) B_L^{\mu\nu}.$$

$$X_{\mu\nu} = (H^\dagger H)^2 B_L^{\mu\nu},$$

Integration by parts: general procedure

O_k — set of operators with k derivatives, before using IBP

Repeat until all $O_k = \{\}$:

1. Take $\mathcal{O}$ in non-empty O_k with lowest k
2. Decompose $D^n\mathcal{O} \to \sum_i \mathcal{Q}_i$
3. Remove the Q_i from the O_n

Examples

Example: EFT for $\phi \sim 2_{1/2}$

```
>>> phi = Field(
        name='phi',
        lorentz_irrep=scalar,
        internal_irrep=irrep('SU2', '1'),
        charges=[1/2]
    )
>>> myeft = EFT(
        algebra('SU2'),
        [phi, phi.conjugate]
    )
```

Example: EFT for $\phi \sim 2_{1/2}$

```
>>> print(myeft.invariants(max_dimension=8))
phi phi*: 1
(phi)^2 (phi*)^2: 1
(phi)^2 (phi*)^2 D^2: 2
(phi)^2 (phi*)^2 D^4: 3
(phi)^3 (phi*)^3: 1
(phi)^3 (phi*)^3 D^2: 2
(phi)^4 (phi*)^4: 1
```

Example: dim-6 SMEFT

```
> python standard_model.py --dimension 6
(psi)^4: 38
(F)^3: 4
phi F (psi)^2: 16
(phi)^2 (psi)^2 D: 9
(phi)^3 (psi)^2: 6
(phi)^2 (F)^2: 8
(phi)^4 D^2: 2
(phi)^6: 1
```

Time: $\sim$ 3 seconds. Other tools: several minutes

Conclusions

- **General**: for any semisimple Lie algebra, any irreps
- **Fast**: seconds for the dim-6 SM, hundreds of times faster than other tools
- Other features:
  - Optionally, not taking into account EOM
  - Obtaining all covariant operators

Installation (Python 3.5+)

```
> pip install basisgen
```