Nonrelativistic calculations with FeynOnium

Vladyslav Shtabovenko

Karlsruhe Institute of Technology
Institute for Theoretical Particle Physics

based on 2006.15451 in collaboration with
N. Brambilla, H. S. Chung and A. Vairo

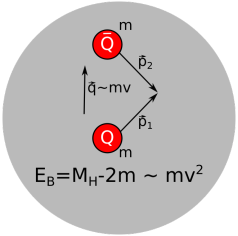
Tangram Nuclear Theory Collaboration Seminar
Virtual - 06.07.2022

Karlsruhe Institute of Technology

Outline

- Effective Field Theories (EFTs) [Wilson & Kogut, 1974; Weinberg, 1979]: a powerful technique to study physical systems with well separated dynamical scales.
- Nonrelativistic Effective Field Theories (NREFTs) describe systems with $v \ll c$, with v being the relevant velocity scale

- Notable examples of such phenomena
  - NR bound states: positronium, muonium [Caswell & Lepage, 1986], heavy quarkonia [Bodwin et al., 1995; Brambilla, Pineda, et al., 2005]
  - systems made of nonrelativistic atoms [Brambilla et al., 2017] and molecules [Brambilla et al., 2018]
  - nonrelativistic dark matter [Hisano et al., 2003, 2004, 2005; Shepherd et al., 2009; An et al., 2016; Biondini & Laine, 2018; Beneke et al., 2019]
  - heavy neutrinos [Biondini et al., 2013]



- Two well-established NREFTs are nonrelativistic QCD (NRQCD) [Caswell & Lepage, 1986; Bodwin et al., 1995] and potential NRQCD (pNRQCD) [Pineda & Soto, 1998a; Brambilla et al., 2000]

- EFT techniques make it *simpler* to describe a physical system and understand its behavior
- Obtaining new higher order predictions within an EFT is still *hard*
- The practical usage of EFTs is thorny (even at *tree*-level) and could hurt the unprepared

- Explicit construction of higher order operators
- Derivation of Feynman rules for the new operators
- Rapid growth of Feynman diagrams beyond LO
- Emergence of unusual propagators
- NREFTs: Possible loss of manifest Lorentz covariance

A typical calculation may involve many different building blocks ...

Diagrams

- Feynman rules
- Diagram generation
- Filtering
- Topologies

Amplitudes

- Expansions
- Dirac algebra
- Color algebra
- Tensor reduction
- Projectors

Loop integrals

- Partial fractioning
- IBP reduction
- Mappings between integrals
- Master integral evaluation

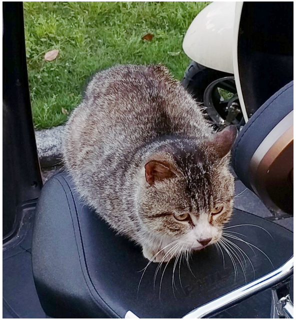
Miscellaneous

- Renormalization
- Phase space
- Matching
- Phenomenology
- ...

... many of which can be automatized completely or to some extent!

- Many tools for streamlining different aspects of EFT calculations are readily available
- Some EFT-specific codes are **Rosetta** [Falkowski et al., 2015], **SMEFTsim** [Brivio et al., 2017], **MatchingTools** [Criado, 2018], **CoDEx** [?, ?], **Wilson** [Aebischer et al., 2018], **DEFT** [Gripaios & Sutherland, 2019] **SmeftFR** [Dedes et al., 2020], **BasisGen** [Criado, 2019], **Sym2Int** [Fonseca, 2017], **ECO** [Marinissen et al., 2020], , **GrIP** [Banerjee et al., 2020], **WCxf** [Aebischer et al., 2018], **DirectDM** [Bishara et al., 2017], **Grinder** [Grozin, 2000], **SoftSERVE** [Bell et al., 2019], **MadOnia** [Artoisenet et al., 2008] , **HELAC-ONIA** [Shao, 2013, 2016], **FDC** [Wang, 2004], **FDCHQHP** [Wan & Wang, 2014] ...

- There are even more general purpose tools that can be used for EFTs: **MadGraph5_aMC@NLO** [Alwall et al., 2014], **GoSam** [Cullen et al., 2012, 2014], **Herwig++** [Bahr et al., 2008], **Sherpa** [Gleisberg et al., 2009], **Whizard** [Moretti et al., 2001; Kilian et al., 2011], **CalcHep** [Belyaev et al., 2013], **CompHep** [?, ?], **FeynArts** [Hahn, 2001], **FeynRules** [Christensen & Duhr, 2009; Alloul et al., 2014], **NLOCT** [Degrande, 2015], **QGRAF** [Nogueira, 1993], **FIRE** [Smirnov, 2015], **Package-X** [Patel, 2015, 2017], ...
- Sadly for NREFT pracitioners, manifest Lorentz covariance is usually taken for granted ...



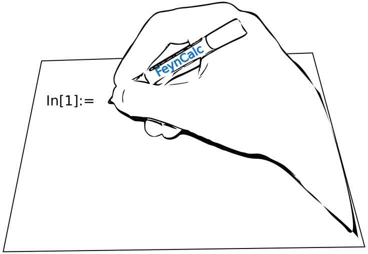
- Indeed, NREFTs are less straightforward to automatize
- Here by *nonrelativistic* we mean manifestly noncovariant
- Without manifest Lorentz covariance we may need to deal with quantities such as

$$
\begin{aligned}
&g^{\mu\nu}, \quad g^{\mu 0}, \quad g^{\mu i}, \quad g^{00}, \quad g^{ij}, \quad \delta^{ij} \\
&\epsilon^{0\mu\nu\rho}, \quad \epsilon^{\mu\nu i}, \quad \epsilon^{\mu ij}, \quad \gamma^0, \quad \gamma^i, \quad \boldsymbol{\gamma}\cdot\boldsymbol{p}, \\
&p^0, \quad \boldsymbol{p}^i, \quad |\boldsymbol{p}|, \quad \boldsymbol{\sigma}^j, \quad \boldsymbol{\sigma}\cdot\boldsymbol{q}\ldots
\end{aligned}
$$

- Even tree-level calculations can quickly become tedious: proliferation of terms when expanding an amplitude in relative 3-momenta to a sufficiently high order
- Explicit evaluation of complicated nonrelativistic expressions (e. g. in a matching calculation)?
- Alternatives to self-written codes?

- Motivation: lack of general purpose amplitude-evaluation codes for NREFTs
- We decided to improve the current situation by
  - creating open-source computational tools that may help to perform NREFT calculations at tree- and 1-loop level
  - building upon an existing framework (**FeynCalc**) that is already well known in the particle physics community
  - making the codes sufficiently generic to cover a wide range of nonrelativistic processes
  - supplying fully worked out examples reproducing selected NREFT results from the literature
- **FeynOnium**: our project to improve the automation of NREFT calculations
- At the moment two main building blocks [Brambilla et al., 2020]
  - Better handling of nonrelativistic quantities in the **Mathematica** package **FeynCalc** [Mertig et al., 1991; Shtabovenko et al., 2016, 2020]
  - A **FeynCalc** add-on called **FeynOnium** with tools specific to particular NREFTs (currently pNRQCD and NRQCD)
- A lot of other NREFTs/NRQFTs (also outside of HEP!) one *could* address, but one has to start somewhere

- **FeynCalc** is a tool that provides many useful functions for symbolic QFT calculations
- Written in **Wolfram Mathematica**
- Open source (GPLv3) and publicly available
- Still in active development
- Time line of relevant publications:



1991	**FeynCalc** 1.0 [Mertig et al., 1991]
1997	**TARCER** [Mertig & Scharf, 1998]
2012	**FeynCalcFormLink** [Feng & Mertig, 2012]
2016	**FeynCalc** 9.0 [Shtabovenko et al., 2016]
2017	**FeynHelpers** [Shtabovenko, 2017]
2020	**FeynCalc** 9.3 [Shtabovenko et al., 2020]
2020	**FeynOnium** [Brambilla et al., 2020]

- How does one calculate Feynman diagrams with **FEYNCALC**?
- **FEYNCALC** only handles symbolic evaluation of the input expressions, no diagram generation, no numerics.
- Entering amplitudes by hand is inconvenient ...
- Realistic calculations require additional tools:
  - Use **FEYNRULES** [Christensen & Duhr, 2009; Alloul et al., 2014] to create new models and export them to **FEYNARTS** (see **examples**)
  - Use **FEYNARTS** [Hahn & Perez-Victoria, 1999] to generate Feynman diagrams
  - Evaluate 1-loop integrals and do the phase-space integration using your favorite tools
- **FEYNCALC** has a built-in interface to **FEYNARTS**
- Interface to **PACKAGE-X** and **FIRE** [Smirnov, 2015] via the **FEYNHELPERS** add-on
- Of course, one can also use only some subset of **FEYNCALC**'s functionality (e. g. tensor reductions or Dirac algebra) and do other steps with **FORM**

- What was possible before **FeynOnium** (i. e. using FC 9.2 + extra tools) w.r.t EFTs?
- The **FR+FA+FC+FH** tool chain to obtain full analytic 1-loop results in some types of *relativistic* EFTs
- Break everything to Passarino-Veltman functions, then use `PaXEvaluate` to invoke **Package-X** $\Rightarrow$ very easy.
- Apparent limitation: only quadratic propagators, no eikonals etc.
- Genuine nonrelativistic calculations with explicit 3-vectors, Pauli matrices and Cartesian loop integrals not feasible.
- **FeynCalc** 9.2 was semi-useful for EFT calculations, but hardly a tool for doing EFTs.
- Idea: extend **FeynCalc** *internally* such, that it can handle NR calculations and nonstandard integrals out-of-the-box!
- An extra add-on **FeynOnium** for some routines needed in EFT calculations (e. g. spin projectors for NRQCD)

- What is actually meant by "extending **FeynCalc** internally"?
- Example: `Pair` is a **FeynCalc** symbol with 2 slots, can represent 4-momenta, scalar products and metric tensors (e. g. in D-dimensions)
  - `Pair[Momentum[`p`,`D`],LorentzIndex[`μ`,`D`]]` $\sim p^\mu$
  - `Pair[Momentum[`p`,`D`],Momentum[`q`,`D`]]` $\sim p \cdot q$
  - `Pair[LorentzIndex[`μ`,`D`],LorentzIndex[`ν`,`D`]]` $\sim g^{\mu\nu}$
- Introduce Cartesian versions of vectors, scalar products and metric tensors via new symbols
  - `CartesianPair[CartesianMomentum[`p`,`$D-1$`],CartesianIndex[`i`,`$D-1$`]]` $\sim \boldsymbol{p}^i$
  - `CartesianPair[CartesianMomentum[`p`,`$D-1$`],CartesianMomentum[`q`,`$D-1$`]]` $\sim \boldsymbol{p} \cdot \boldsymbol{q}$
  - `CartesianPair[CartesianIndex[`i`,`$D-1$`],CartesianIndex[`j`,`$D-1$`]]` $\sim \delta^{ij}$
  - `TemporalPair[TemporalMomentum[`p`],ExplicitLorentzIndex[0]]` $\sim p^0$
- Existing **FeynCalc** functions for manipulating Lorentz tensors should work also with Cartesian tensors (no duplication!)
- Support tensors that mix Lorentz and Cartesian indices as in

$$p^\mu g_\mu^i = \boldsymbol{p}^i$$

- **FeynCalc** symbols can be represented in two ways
- Internal (`FCI`) representation for symbolic manipulations: `Pair[Momentum[`p`,`D`],LorentzIndex[`μ`,`D`]]` $\sim p^\mu$
- External (`FCE`) representation for the user input or export of the results: `FVD[`p`,`μ`]` $\sim p^\mu$
- Most common `FCE`-shortcuts in **FeynCalc** 9.2

Shortcut in **FeynCalc**	Meaning
`MT[`μ`,`ν`]`, `MTD[`μ`,`ν`]` `MTE[`μ`,`ν`]`	$\bar{g}^{\mu\nu}$, $g^{\mu\nu}$, $\hat{g}^{\mu\nu}$
`FV[`p`,`μ`]`, `FVD[`p`,`μ`]`, `FVE[`p`,`μ`]`	$\bar{p}^\mu$, p^μ, $\hat{p}^\mu$
`SP[`p`,`q`]`, `SPD[`p`,`q`]`, `SPE[`p`,`q`]`	$\bar{p} \cdot \bar{q}$, $p \cdot q$, $\hat{p} \cdot \hat{q}$
`GA[`μ`]`, `GAD[`μ`]`, `GAE[`μ`]`	$\bar{\gamma}^\mu$, γ^μ, $\hat{\gamma}^\mu$
`GS[`p`]`, `GSD[`p`]`, `GSE[`p`]`	$\bar{\gamma} \cdot \bar{p}$, $\gamma \cdot p$, $\hat{\gamma} \cdot \hat{p}$
`LC[`μ`,`ν`,`ρ`,`σ`]`, `LC[`μ`,`ν`][`p`,`q`]`	$\bar{\epsilon}^{\mu\nu\rho\sigma}$, $\bar{\epsilon}^{\mu\nu\rho\sigma} p_\rho q_\sigma$
`LCD[`μ`,`ν`,`ρ`,`σ`]`, `LCD[`μ`,`ν`][`p`,`q`]`	$\epsilon^{\mu\nu\rho\sigma}$, $\epsilon^{\mu\nu\rho\sigma} \hat{p}_\rho \hat{q}_\sigma$

- Much more shortcuts in version 9.3 to account for new NR quantities

Shortcut in **FeynCalc**	Meaning
`KD[i,j]`, `KDD[i,j]`, `KDE[i,j]`	$\bar{\delta}^{ij}$, δ^{ij}, $\hat{\delta}^{ij}$
`CV[p,i]`, `CVD[p,i]`, `CVE[p,i]`	$\bar{\boldsymbol{p}}^i$, $\boldsymbol{p}^i$, $\hat{\boldsymbol{p}}^i$
`CSP[p,q]`, `CSPD[p,q]`, `CSPE[p,q]`	$\bar{\boldsymbol{p}} \cdot \bar{\boldsymbol{q}}$, $\boldsymbol{p} \cdot \boldsymbol{q}$, $\hat{\boldsymbol{p}} \cdot \hat{\boldsymbol{q}}$
`TGA[]`	$\bar{\gamma}^0$
`CGA[i]`, `CGAD[i]`, `CGAE[i]`	$\bar{\boldsymbol{\gamma}}^i$, $\boldsymbol{\gamma}^i$, $\hat{\boldsymbol{\gamma}}^i$
`CGS[p]`, `CGSD[p]`, `CGSE[p]`	$\bar{\boldsymbol{\gamma}} \cdot \bar{p}$, $\boldsymbol{\gamma} \cdot \boldsymbol{p}$, $\hat{\boldsymbol{\gamma}} \cdot \hat{\boldsymbol{p}}$
`CLC[i,j,k]`, `CLC[i,j][p]`	$\bar{\epsilon}^{ijk}$, $\bar{\epsilon}^{ijk}\bar{\boldsymbol{p}}^k$
`CLCD[i,j,k]`, `CLCD[i,j][p]`	ϵ^{ijk}, $\epsilon^{ijk}\boldsymbol{p}^k$
`SI[μ]`, `SID[μ]`, `SIE[μ]`	$\bar{\sigma}^\mu$, σ^μ, $\hat{\sigma}^\mu$
`SIS[p]`, `SISD[p]`, `SISE[p]`	$\bar{\sigma} \cdot \bar{p}$, $\sigma \cdot p$, $\hat{\sigma} \cdot \hat{p}$
`CSI[i]`, `CSID[i]`, `CSIE[i]`	$\bar{\boldsymbol{\sigma}}^i$, $\boldsymbol{\sigma}^i$, $\hat{\boldsymbol{\sigma}}^i$
`CSIS[p]`, `CSISD[p]`, `CSISE[p]`	$\bar{\boldsymbol{\sigma}} \cdot \bar{\boldsymbol{p}}$, $\boldsymbol{\sigma} \cdot \boldsymbol{p}$, $\hat{\boldsymbol{\sigma}} \cdot \hat{\boldsymbol{p}}$

- In manifestly Lorentz covariant calculations the distinction between upper and lower Lorentz indices is not necessary
- Einstein's summation convention must be satisfied for each term!
- In the previous **FeynCalc** versions a standalone `FV[`p`,`μ`]` could be interpreted as p^μ or p_μ, depending on the context
- `FV[`p`,`μ`] FV[`q`,`μ`]` may mean $p^\mu q_\mu$ or $p_\mu q^\mu$ (ok for dummy indices)
- In noncovariant calculations ambiguities are to be expected
- Does `CV[p,i]CV[q,i]` mean $\boldsymbol{p}^i \boldsymbol{q}^i$, $\boldsymbol{p}^i \boldsymbol{q}_i$ or $\boldsymbol{p}_i \boldsymbol{q}_i$?
- Minimally invasive solution: Introduce a set of rules that formalize the existing behavior for Lorentz indices and clarify the positions of the Cartesian indices

- New rules in **FeynCalc** 9.3 for interpreting input/output expressions
  1. Every expression must satisfy Einsteins's summation convention, both for Lorentz and Cartesian indices. Single terms containing more than two identical Lorentz or Cartesian indices are illegal and will lead to inconsistent results.
  2. In a contraction of two Lorentz indices it is understood that one of them is upstairs and the other is downstairs.
  3. In a contraction of two Cartesian indices, both indices are understood to be upper indices.
  4. A free Lorentz or Cartesian index is always understood to be an upper index.
- This way input and output expressions become unambiguous

$$\begin{aligned} \texttt{CSP[p, q]} &\equiv \boldsymbol{p}^i \boldsymbol{q}^i = \boldsymbol{p} \cdot \boldsymbol{q}, \\ \texttt{CV[p, i]CV[q, i]} &\equiv \boldsymbol{p}^i \boldsymbol{q}^i, \\ \texttt{CV[l, k]KD[j, k]} &\equiv \boldsymbol{l}^k \delta^{jk}. \end{aligned}$$

- Still, in the case of Cartesian tensors the user should pay attention to his/her input
- For example, if one wants to enter exactly δ^i_k this should be converted to $-\delta^{ik}$ beforehand.
- The default metric signature used in the package is $(1, -1, -1, -1)$
- Can be changed to $(-1, 1, 1, 1)$ via `FCSetMetricSignature` (still experimental)
- Euclidean signature $(1, 1, 1, 1)$ not yet supported

- We allow for Dirac matrices with temporal γ^0 and Cartesian indices $\boldsymbol{\gamma}^i$
- Dirac matrices contracted to a 3-vector $\boldsymbol{\gamma} \cdot \boldsymbol{p}$ also possible
- Useful trick for implementing algebraic manipulations

$$
\begin{aligned}
\boldsymbol{\gamma}^i \left(\gamma^{\cdots} \ldots \gamma^{\cdots}\right) \boldsymbol{\gamma}^i &= \gamma^0 \left(\gamma^{\cdots} \ldots \gamma^{\cdots}\right) \gamma^0 - \gamma^\mu \left(\gamma^{\cdots} \ldots \gamma^{\cdots}\right) \gamma_\mu, \\
\boldsymbol{\gamma} \cdot \boldsymbol{p} \left(\gamma^{\cdots} \ldots \gamma^{\cdots}\right) \boldsymbol{\gamma} \cdot \boldsymbol{p} &= \gamma^0 p^0 \left(\gamma^{\cdots} \ldots \gamma^{\cdots}\right) \gamma^0 p^0 - \not{p} \left(\gamma^{\cdots} \ldots \gamma^{\cdots}\right) \not{p}
\end{aligned}
$$

  - First term: evaluate by anticommuting γ^0 past $(\gamma^{\cdots} \ldots \gamma^{\cdots})$.
  - Second term: use the existing implementation of the Dirac algebra
- Two missing features planned for the future
  - Euclidean Dirac matrices (useful for lattice calculations)
  - Explicit (i. e. nonsymbolic) Cartesian indices e. g. $\boldsymbol{\gamma}^i \boldsymbol{\gamma}^j \boldsymbol{\gamma}^i \leftrightarrow \gamma^1 \gamma^2 \gamma^1$

- Pauli matrices are a completely new set of symbols in **FeynCalc**
- Ubiquitous in NREFTs calculations, can appear contracted with 3-vectors.
- Simplify chains of Pauli matrices using following relations (valid both in 4 and D-dimensions)

$$\boldsymbol{\sigma}^i \boldsymbol{\sigma}^{j_1} \dots \boldsymbol{\sigma}^{j_n} \boldsymbol{\sigma}^i = (-1)^n (D-3) \boldsymbol{\sigma}^{j_1} \dots \boldsymbol{\sigma}^{j_n} + 2 \sum_{i=1}^{n-1} (-1)^{i+1} \boldsymbol{\sigma}^{j_1} \dots \boldsymbol{\sigma}^{j_{i-1}} \boldsymbol{\sigma}^{j_{i+1}} \dots \boldsymbol{\sigma}^{j_n} \boldsymbol{\sigma}^{j_i},$$

$$(\boldsymbol{\sigma} \cdot \boldsymbol{p}) \boldsymbol{\sigma}^{j_1} \dots \boldsymbol{\sigma}^{j_n} (\boldsymbol{\sigma} \cdot \boldsymbol{p}) = (-1)^n \boldsymbol{p}^2 \boldsymbol{\sigma}^{j_1} \dots \boldsymbol{\sigma}^{j_n} + 2 \sum_{i=1}^{n} (-1)^{i+1} \boldsymbol{p}^{j_i} \boldsymbol{\sigma}^{j_1} \dots \boldsymbol{\sigma}^{j_{i-1}} \boldsymbol{\sigma}^{j_{i+1}} \dots \boldsymbol{\sigma}^{j_n} (\boldsymbol{\sigma} \cdot \boldsymbol{p}).$$

- Traces with an even number of matrices can be evaluated (both in 4 and D-dimensions) using

$$\mathrm{Tr}(\boldsymbol{\sigma}^{i_1} \dots \boldsymbol{\sigma}^{i_{2n}}) = \sum_{j=2}^{2n} \delta^{i_1 i_j} (-1)^j \mathrm{Tr}(\boldsymbol{\sigma}^{i_2} \dots \boldsymbol{\sigma}^{i_{j-1}} \boldsymbol{\sigma}^{i_{j+1}} \dots \boldsymbol{\sigma}^{i_{2n}})$$

- Traces with an odd number of matrices are well defined only in 4 dimensions

$$\mathrm{Tr}(\boldsymbol{\sigma}^{i_1} \dots \boldsymbol{\sigma}^{i_{2n}} \boldsymbol{\sigma}^{i_{2n+1}}) = \delta^{i_1 i_2} \mathrm{Tr}(\boldsymbol{\sigma}^{i_3} \dots \boldsymbol{\sigma}^{i_{2n}} \boldsymbol{\sigma}^{i_{2n+1}}) + i \epsilon^{i_1 i_2 k} \mathrm{Tr}(\boldsymbol{\sigma}^k \boldsymbol{\sigma}^{i_3} \dots \boldsymbol{\sigma}^{i_{2n}} \boldsymbol{\sigma}^{i_{2n+1}})$$

- Assuming anticommutativity of Pauli matrices and the cyclicity of the trace, in D-dimensions one finds [Hoang & Ruiz-Femenia, 2006]

$$(D-4)\text{Tr}(\boldsymbol{\sigma}^i \boldsymbol{\sigma}^j \boldsymbol{\sigma}^k) = 0$$

- Remember that we have a similar relation for a D-dimensional γ^5

$$(D-4)\text{Tr}(\gamma^5 \gamma^\mu \gamma^\nu \gamma^\rho \gamma^\sigma) = 0$$

- We handled this issue by implementing a mechanism for different Pauli matrix schemes (`FCSetPauliSigmaScheme`).
- The default behavior is to leave D-dimensional $\boldsymbol{\sigma}$-odd traces untouched.
- There is also a naive scheme that applies

$$\boldsymbol{\sigma}^i \boldsymbol{\sigma}^j = \delta^{ij} + i \epsilon^{ijk} \boldsymbol{\sigma}^k,$$

in D-dimensions (must be enabled explicitly)

- Tree-level NREFT calculations: Cartesian tensors and Pauli matrices + additional routines from **FeynOnium**
- To achieve higher precision we need semi-automation at least at 1-loop
- The challenge is that different (NR)EFTs feature very different loop integrals
  - eikonal integrals
  - Euclidean and Cartesian integrals
  - integrals with explicit temporal components of 4-vectors
  - . . .

- The standard route for integrals with quadratic propagators: tensor reduction $\to$ Passarino-Veltman functions [Passarino & Veltman, 1979] $\to$ analytic or numerical evaluation
- May not be feasible with nonstandard integrals
  - Irreducible numerators
  - No “standard” basis
  - No libraries for numerics
- Recent progress towards a generalization of Passarino-Veltman techniques to noncovariant integrals: [Chang, 2020]
- Focus on operations that are always possible
  - Tensor reduction

$$\int d^D k \frac{k^\mu k^\nu}{k^2\,(k \cdot p - m^2)} = \frac{m^4}{(D-1)p^4} (D\, p^\mu p^\nu - p^2 g^{\mu\nu}) \int \frac{d^D k}{k^2\,(k \cdot p - m^2)}$$

  - Partial fractioning

$$\int d^{D-1} \boldsymbol{k}\, \frac{4(\boldsymbol{k} \cdot \boldsymbol{p})}{\boldsymbol{k}^2 (\boldsymbol{k}+\boldsymbol{p})^2 (\boldsymbol{k}-\boldsymbol{p})^2} = \int \frac{1}{\boldsymbol{k}^2 (\boldsymbol{k}-\boldsymbol{p})^2} - \int \frac{1}{\boldsymbol{k}^2 (\boldsymbol{k}+\boldsymbol{p})^2}$$

- The so-obtained results can be reused in other codes

- New symbols to represent various nonstandard propagators

Shortcut in **FeynCalc**	Meaning
$\texttt{FAD}[\{k - p_1 - \ldots, m, n\}]$	$\left[\frac{1}{(k-p_1-\ldots)^2-m^2+i\eta}\right]^n$
$\texttt{SFAD}[\{\{k - p_1 - \ldots, \pm k.(q_1 + \ldots)\}, \{\pm m^2, \pm 1\}, n\}]$	$\left[\frac{1}{(k-p_1-\ldots)^2 \pm k.(q_1+\ldots) \mp m^2 \pm i\eta}\right]^n$
$\texttt{CFAD}[\{\{k - p_1 - \ldots, \pm k.(q_1 + \ldots)\}, \{\pm m^2, \pm 1\}, n\}]$	$\left[\frac{1}{(\boldsymbol{k}-\boldsymbol{p}_1-\ldots)^2 \pm \boldsymbol{k}.(\boldsymbol{q}_1+\ldots) \pm m^2 \pm i\eta}\right]^n$
$\texttt{GFAD}[\{\{x, \pm 1\}, n\}]$	$\left[\frac{1}{x \pm i\eta}\right]^n$

- `FAD`: original symbol for covariant quadratic propagators
- `SFAD`: new symbol for covariant quadratic or eikonal propagators
- `CFAD`: new symbol for Cartesian quadratic or eikonal propagators
- `GFAD`: new symbol for generic propagators
- Heuristics (e. g. finding useful momenta shifts in `FeynAmpDenominatorSimplify`) still needs to be improved for new integral types.

- The new notation allows for a vast range of loop integrals made of different propagators

$$
\begin{aligned}
\texttt{SFAD}[\{\texttt{p}, \texttt{m}^2\}] &\equiv \frac{1}{p^2 - m^2 + i\eta}, \\
\texttt{SFAD}[\{\texttt{p}, \{-\texttt{m}^2, -1\}\}] &\equiv \frac{1}{p^2 + m^2 - i\eta}, \\
\texttt{SFAD}[\{\{0, 2\,\texttt{p.q}\}\}] &\equiv \frac{1}{2\,p \cdot q + i\eta}, \\
\texttt{SFAD}[\{\{\texttt{p}, -2\,\texttt{p.q}\}, \texttt{m}^2\}] &\equiv \frac{1}{p^2 - 2\,p \cdot q - m^2 + i\eta}, \\
\texttt{CFAD}[\{\texttt{p}, \texttt{m}^2\}] &\equiv \frac{1}{\boldsymbol{p}^2 + m^2 - i\eta}, \\
\texttt{CFAD}[\{\{0, 2\,\texttt{p.q}\}\}] &\equiv \frac{1}{2\,\boldsymbol{p} \cdot \boldsymbol{q} - i\eta}, \\
\texttt{GFAD}[\texttt{TC}[\texttt{p}] - \texttt{En}] &\equiv \frac{1}{p^0 - E_n + i\eta}.
\end{aligned}
$$

- **FeynCalc** classifies loop integrals into 3 possible categories
  1. Loop momenta appear only as 4-vectors, i. e. manifestly Lorentz covariant.
  2. Manifestly noncovariant, each integration measure splits into temporal and spatial components e. g. as in

$$\int dk^0 \, d^{D-1}\boldsymbol{k} \, f(k^0, \boldsymbol{k})$$

  3. Mixtures of covariant and noncovariant quantities e. g. as in

$$\int d^D k \frac{1}{k^0 + x} \frac{1}{k^2} \frac{1}{(\boldsymbol{p} - \boldsymbol{k})^2},$$

- Integrals of type 1 or 2 are straightforward to manipulate within **FeynCalc**

- “Mixed” integrals of type 3 require additional care
- Can try to rewrite them into a covariant fashion by introducing auxiliary vectors

$$k^0 = k \cdot n, \quad \boldsymbol{k} \cdot \boldsymbol{p} = -k \cdot \tilde{p} \quad \text{with } \tilde{p} = (0, \boldsymbol{p})^T, n = (1,0,0,0)^T$$

  or metric tensors with mixed indices

$$\int d^D k \, \boldsymbol{k}^i f(k) = g^i_\mu \int d^D k \, k^\mu f(k).$$

- Not always useful c. f. e. g.

$$\frac{1}{\boldsymbol{k}^2} = \frac{1}{(k \cdot n)^2 - k^2}$$

- Alternatively, eliminate 4-vectors to obtain a noncovariant integral

$$k^2 = (k^0)^2 - \boldsymbol{k}^2, \quad k \cdot p = k^0 p^0 - \boldsymbol{k} \cdot \boldsymbol{p}.$$

- Then there are two ways to proceed
  - Tensor reduce and partial fraction the $\boldsymbol{k}$-integral, while treating k^0 as an external parameter.
  - Integrate over k^0 first, then handle the purely Cartesian $\boldsymbol{k}$-integral (may introduce dependence on $|\boldsymbol{k}|$ and $|\boldsymbol{k} - \boldsymbol{p}|$)
- **FeynCalc** does not automatize the k^0-integration, but can deal with integrals that contain $|\boldsymbol{k}|$.
- Possible way to pick up the residues more automatically: **Lotty** [Bobadilla, 2021]
- In general, the treatment of nonstandard integrals still can be improved in many ways.

- What is inside the **FeynOnium** add-on?
- Expressing Dirac spinor chains in terms of Pauli matrices and Pauli spinors: `FMSpinorChainExplicit2`
- Special kinematic configurations [Braaten & Chen, 1996] for spinors describing a heavy nonrelativistic system via `FMSpinorChainExplicit`
- Covariant projectors for heavy nonrelativistic systems [?, ?]: `FMInsertCovariantProjector`
- Projections with $J = 0, 1$ and 2 for 3-dimensional Cartesian tensors up to rank 5: `FMCartesianTensorDecomposition`
- Repetitive application of the 3D Schouten's identity

$$\epsilon^{ijk}\boldsymbol{p}^l - \epsilon^{jkl}\boldsymbol{p}^i + \epsilon^{kli}\boldsymbol{p}^j - \epsilon^{lij}\boldsymbol{p}^k = 0,$$

  via `FMCartesianSchoutenBruteForce`
- Feynman rules for pNRQCD vertices in the weak-coupling regime at order r (c. f. figure 5 of [Brambilla, Pineda, et al., 2005])

- **Example calculations** bundled with **FeynOnium**
- 1-loop level
  - Euler-Heisenberg Lagrangian [?, ?]
  - 1-loop correction to the heavy nucleon propagator in baryonic ChPT [Ecker & Mojzis, 1996; Scherer, 2003]
  - Dimension six 4-fermion operators in NRQCD (unequal mass case) [Pineda & Soto, 1998b; Brambilla, Vairo, & Rosch, 2005]
  - Virtual corrections to inclusive hadronic decays of P-wave quarkonia in NRQCD [Petrelli et al., 1998]
  - One-loop running of the chromoelectric dipole interaction in pNRQCD [Brambilla et al., 2000; Pineda & Soto, 2000]
- Tree-level
  - $J/\psi \to 3\gamma$ decay in NRQCD [Ore & Powell, 1949; Bodwin et al., 1995]
  - $Q\bar{Q} \to \gamma\gamma$ decays in NRQCD [Brambilla et al., 2006]
  - Relativistic corrections to quarkonium light-cone distribution amplitudes [Brambilla et al., 2019]

- Let us consider one simple example in more details
- Tree-level QCD process: $Q(p_1) + \bar{Q}(p_2) \to \gamma(k_1) + \gamma(k_2)$
- Kinematics

$$p_1 = \frac{1}{2}P + q, \quad p_2 = \frac{1}{2}P - q, \quad P \cdot q = 0,$$

and

$$p_{1,2}^2 = m_Q^2, \quad k_{1,2}^2 = 0,$$

- Expand the amplitude in the relative momentum of the heavy quark pair, $|\boldsymbol{q}|$, up to 4th order.
- Then match to the corresponding NRQCD amplitude
- Nonrelativistic kinematics

$$q^0 = 0, \quad p_{1,2}^0 = k_{1,2}^0 = E_{\boldsymbol{q}}, \quad E_{\boldsymbol{q}} = \sqrt{\boldsymbol{q}^2 + m_Q^2}$$

- Results available in [Brambilla et al., 2006].

Summary

- **FEYNCALC** 9.3 and **FEYNONIUM** are very useful for semi-automatic of NREFT calculations
- We are not aware of a similar public code for loop calculations able to handle nonrelativistic expressions and a wide range of nonstandard integrals
- We did not write everything from a scratch but *extended* an existing software for Lorentz covariant calculations $\Rightarrow$ Blueprint for other tools?
- Not a fully automatic all-in-one solution, but a handy tool for knowledgeable people
- The current focus is on tree-level and 1-loop calculations

Outlook

- In principle, there are endless possibilities to extend this framework (new algorithms, new NREFTs, new examples, ...)
- NREFT diagram generation: An interface between **FEYNCALC** and **QGRAF** [Nogueira, 1993] already in beta-testing
- Additional routines for deriving Feynman rules in NREFTs