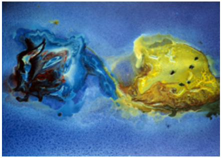
FeynHelpers: Connecting FeynCalc to FIRE and Package-X

Vladyslav Shtabovenko

Technische Universität München

Instituto de Física Corpuscular, Valencia

Technische Universität München



Physik-Department T30f

Outline

A generic perturbative QFT calculation may involve many different steps

- Feynman diagrams
  - Feynman rules from $\mathcal{L}$
  - Diagram generation
  - Amplitudes
  - ...
- Dirac algebra
  - Simplification of γ-matrix chains
  - Dirac traces
  - SPVAT form
  - Fierz identities
  - ...
- Loop integrals
  - Tensor reduction
  - Partial fractioning
  - Mapping of topologies
  - IBP-Reduction
  - Numerics
  - ...

...

We can automatize each step separately using standalone packages (e. g. FEYNARTS, LOOPTOOLS [Hahn & Perez-Victoria, 1999], FEYNRULES [Christensen & Duhr, 2008], QGRAF [Nogueira, 1993], TRACER [Jamin & Lautenbacher, 1993], FORMTRACER [Cyrol et al., 2016], FORCER [Ruijl et al., 2017], PYSECDEC [Borowka et al.,], ...) and self-written codes.

Or we can employ all-in-one packages that handle most of these steps in one framework.

Two big categories of all-in-one packages

- Fully-automatic (FORMCALC [Hahn & Perez-Victoria, 1999], GOSAM [Cullen et al., 2014], GRACE [Belanger et al., 2006], DIANA [Tentyukov & Fleischer, 2000], FDC [Wang, 2004], . . .)
- Semi-automatic (FEYNCALC [Mertig et al., 1991, Shtabovenko et al., 2016], HEPMATH [Wiebusch, 2014], PACKAGE-X [Patel, 2015], . . .)

Fully-automatic tools

Blackbox: Require only minimal user input and provide a small set of options. The code takes care of the rest.

- ⊕ Easy to use
- ⊕ Foolproof
- ⊕ Constantly good performance
- ⊕ Saves your time
- ⊖ Limited number of templated calculations
- ⊖ Difficult to extend/modify for your needs
- ⊖ Not easy to obtain intermediate results

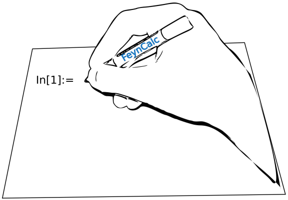
Semi-automatic tools

Toolbox: Combine different tools with many options to get the computation done in your way.

- ⊖ You must know what you are doing
- ⊖ Easy to make mistakes
- ⊖ The performance depends on your skills
- ⊖ Writing codes may take quite some time
- ⊕ Very broad range of applications
- ⊕ Extendable with user-defined objectsb
- ⊕ Intermediate results at each step

FEYNCALC

Open source (GPLv3) MATHEMATICA package for symbolic semi-automatic evaluation of Feynman diagrams and algebraic expressions in QFT.



Features

- Suitable for evaluating both single expressions and full Feynman diagrams.
- The calculation can be organized in many different ways (flexibility)
- Extensive typesetting for better readability
- Lorentz index contractions, $\mathrm{SU}(N)$ algebra, Dirac algebra, etc.
- Passarino-Veltman reduction of one-loop amplitudes to standard scalar integrals
- Basic support for manipulating multi-loop integrals
- General tools for non-commutative algebra
- BUT: Essentially only algebraic manipulations, everything else requires extra tools.

FEYNCALC developer team

- Rolf Mertig (GluonVision GmbH): original author of the package, first release 1991
- Frederik Orellana (Technical University of Denmark): joined 1997
- VS (TUM, soon Zhejiang University): joined 2014

Recent developments (since 2014)

- Large parts of the code improved or rewritten from scratch.
- Public source code repository on GITHUB: `https://github.com/FeynCalc`
- Online documentation `https://feyncalc.github.io/reference`
- Ships with many sample calculations
- Extensive unit testing framework
- New and improved functions for loop calculations.
- Big emphasis on using FEYNCALC for Effective Field Theory (EFT) calculations.
    - Original motivation for FEYNHELPERS: Matching calculations in relativistic EFTs
    - Upcoming FEYNCALC 9.3 and FEYNONIUM: Matching calculations in nonrelativistic EFTs (in particular NRQED/NRQCD [Caswell & Lepage, 1986, Bodwin et al., 1995], pNRQED/pNRQCD [Pineda & Soto, 1998b, Pineda & Soto, 1998a, Brambilla et al., 2000])

When is FEYNCALC useful?

- Small or medium-sized calculations, too specific for fully automatic packages
- FEYNCALC as a “calculator” for QFT expressions
- Cross-check results from other people
- Extensive manipulations on the level of the amplitudes

Limitations of FEYNCALC?

- Written entirely in WOLFRAM language, cannot be used without MATHEMATICA
- Inherits MATHEMATICA’s performance problems with large number of terms
- Not really suited for large and complex calculations
- Much slower than FORM

Why not combine FEYNCALC/MATHEMATICA with FORM?

- Thomas Hahn already had a similar idea many years ago.
- FORMCALC is much faster than FEYNCALC, but also less flexible
- Performance-wise it is not so clever to constantly pass very large expressions between MATHEMATICA and FORM
- However, that would be necessary(?) to preserve the flexibility of FEYNCALC
- FEYNCALCFORMLINK employs FORM for index contractions and Dirac traces.
- FORMTRACER is a recent package that provides access to FORM from MATHEMATICA

Most used functions for loop calculations

- `ApartFF`: Partial fractioning for 1-loop and multi-loop integrals
- `FDS`: Shifts in loop momenta for 1-loop and multi-loop integrals
- `TID`: Tensor reduction for 1-loop integrals
- `ToPaVe`: Converts scalar 1-loop integrals to Passarino–Veltman scalar functions
- `PaVeReduce`: Reduction of Passarino–Veltman coefficient functions to scalar functions
- `FCMultiLoopTID`: Tensor reduction for multi-loop integrals

Less known functions

- `FCLoopBasisIncompleteQ`
- `FCLoopBasisOverDeterminedQ`
- `FCLoopBasisFindCompletion`
- `FCLoopIBPReducableQ`

Partial fractioning

- Scalar loop integrals can be often simplified even further by using partial fractioning.
- Well known identities (implemented in `SPC` and `Apart2`) are

$$q \cdot p = \frac{1}{2}[(q+p)^2 + m_2^2 - (q^2 + m_1^2) - p^2 - m_2^2 + m_1^2],$$
$$\frac{1}{(q^2 - m_1^2)(q^2 - m_2^2)} = \frac{1}{m_1^2 - m_2^2}\left(\frac{1}{q^2 - m_1^2} - \frac{1}{q^2 - m_2^2}\right).$$

- But: Many decompositions, e.g.

$$\int d^D q \, \frac{1}{q^2 (q-p)^2 (q+p)^2} = \frac{1}{p^2} \int d^D q \left(\frac{1}{q^2 (q-p)^2} - \frac{1}{(q-p)^2 (q+p)^2} \right),$$

  require more sophisticated algorithms.
- New in FeynCalc 9: `ApartFF` introduces partial fractioning algorithm from [Feng, 2012]
- Compared to the reference Mathematica implementation (`https://github.com/F-Feng/APart`), it is fully integrated into FeynCalc

---

In[1]:= ApartFF[FAD[{q}, {q − p}, {q + p}], {q}]

Out[2]:= $\frac{1}{p^2 q^2 . (q-p)^2} - \frac{1}{p^2 q^2 . (q-2p)^2}$

Tensor reduction

- 1-loop tensor reduction is done via Passarino-Veltman technique: `TID`
- `TID` has received many improvements in FEYNCALC 9 and above
- Default mode: Reduce each tensor integral to PaVe scalar functions (A_0, B_0, C_0, D_0)

In[1]:= FCI[GAD[μ].(m + GSD[q]).GAD[μ] FAD[{q, m}]]

Out[1]:= $\frac{\gamma^{\mu}.(m+\gamma\cdot q).\gamma^{\mu}}{(q^2-m^2).(q-p)^2}$

In[2]:= TID[%, −p + q], q]// ToPaVe[#, q]&

Out[2]= $\frac{i\pi^2(D-2)\mathrm{A}_0\left(m^2\right)\gamma\cdot p}{2p^2} - \frac{i\pi^2\mathrm{B}_0\left(p^2,0,m^2\right)\left(Dm^2\gamma\cdot p-2Dmp^2+Dp^2\gamma\cdot p-2m^2\gamma\cdot p-2p^2\gamma\cdot p\right)}{2p^2}$

Tensor reduction

- Zero Gram determinants? Detected automatically, reductions switches to Passarino–Veltman coefficient functions (e.g. B_1, B_{00}, C_{222} etc.)

  Consider $\int \frac{d^D q}{(2\pi)^D} \frac{\gamma^\mu (m + \not{q}) \gamma^\mu}{(q^2 - m^2)(q-p)^2}$ with $p^2 = 0$

In[1]:= SPD[p, p] = 0;
TID[GAD[μ].(m + GSD[q]).GAD[μ] FAD[{q, m}, −p + q], q];

Out[2]:= $i\pi^2 \mathrm{B}_0\left(0,0,m^2\right)(Dm - D\gamma\cdot p + 2\gamma\cdot p) - i\pi^2(D-2)\gamma\cdot p\mathrm{B}_1\left(0,0,m^2\right)$

- Useful options:
  - `UsePaVeBasis`: Enforces reduction into coefficient functions for any kinematics.
  - `GenPaVe`: Allows define PaVe functions in a different way (standard is the LOOPTOOLS convention)
  - `Isolate`: Kinematic coefficients in front of the loop inetgrals will be abbreviated. Use `FRH` to recover the original form.

How about multi-loop tensor reduction?

- In general, not very useful above 1-loop, many scalar products in the denominators can't be cancelled against propagators in the numerators.
- Still practical for loop momenta contracted with Dirac matrices and Levi-Civita tensors. FEYNCALC 9 features `FCMultiLoopTID`:
  - uses the same PaVe algorithm as for 1-loop.
  - currently no proper way to handle zero Gram determinants.

In [1]:= FCI[FVD[q1, μ] FVD[q2, ν] FAD[q1, q2, {q1 − p1}, {q2 − p1}, {q1 − q2}]]

Out[1]:= $$\frac{q1^{\mu} q2^{\nu}}{q1^2.q2^2.(q1-p1)^2.(q2-p1)^2.(q1-q2)^2}$$

In [2]:= FCMultiLoopTID[%, {q1, q2}]

Out[2]:= $$\frac{D\,p1^{\mu} p1^{\nu} - p1^2 g^{\mu\nu}}{4(D-1)q2^2.q1^2.(q2-p1)^2.(q1-q2)^2.(q1-p1)^2} - \frac{p1^2 g^{\mu\nu} - p1^{\mu} p1^{\nu}}{2(D-1)p1^2 q2^2.q1^2.(q2-p1)^2.(q1-p1)^2} + \frac{p1^2 g^{\mu\nu} - p1^{\mu} p1^{\nu}}{(D-1)p1^2 q2^2.q1^2.(q1-q2)^2.(q1-p1)^2} - \frac{D\,p1^{\mu} p1^{\nu} - p1^2 g^{\mu\nu}}{2(D-1)p1^4 q1^2.(q2-p1)^2.(q1-q2)^2}$$

EXTRACTION OF LOOP INTEGRALS

- To evaluate the loop integrals outside of FEYNCALC, we need to extract all the unique integrals from the given expression
- New in FeynCalc 9: `FCLoopIsolate`

In[1]:= gse = FCI[FAD[q, −p + q] MTD[Lor3, Lor4] (FVD[−p − q, Lor5] MTD[Lor1, Lor3] + FVD[2 p − q, Lor3] MTD[Lor1, Lor5] + FVD[−p + 2 q, Lor1] MTD[Lor3, Lor5]) (FVD[p + q, Lor6] MTD[Lor2, Lor4] + FVD[−2 p + q, Lor4] MTD[Lor2, Lor6] + FVD[p − 2 q, Lor2] MTD[Lor4, Lor6]) MTD[Lor5, Lor6]]

Out[1]:= $$\frac{g^{\text{Lor3Lor4}} g^{\text{Lor5Lor6}} \left(g^{\text{Lor3Lor5}}(2q-p)^{\text{Lor1}} + g^{\text{Lor1Lor5}}(2p-q)^{\text{Lor3}} + g^{\text{Lor1Lor3}}(-p-q)^{\text{Lor5}}\right)}{q^2.(q-p)^2} \left(g^{\text{Lor4Lor6}}(p-2q)^{\text{Lor2}} + g^{\text{Lor2Lor6}}(q-2p)^{\text{Lor4}} + g^{\text{Lor2Lor4}}(p+q)^{\text{Lor6}}\right)$$

In[2]:= FCLoopIsolate[Contract[gse], {q}, **Head** −> loop] // Cases2[#, loop] &

Out[2]:= $$\left\{\text{loop}\left(\frac{1}{q^2.(q-p)^2}\right), \text{loop}\left(\frac{q^{\text{Lor1}}}{q^2.(q-p)^2}\right), \text{loop}\left(\frac{q^{\text{Lor2}}}{q^2.(q-p)^2}\right), \text{loop}\left(\frac{q^{\text{Lor1}} q^{\text{Lor2}}}{q^2.(q-p)^2}\right), \text{loop}\left(\frac{pq\cdot}{q^2.(q-p)^2}\right), \text{loop}\left(\frac{q^2}{q^2.(q-p)^2}\right)\right\}$$

- Furthermore:
  - `FCLoopSplit` to separate different types of loop integrals (free of loops, scalar integrals with and without scalar products in the numerators, tensor integrals)
  - `FCLoopExtract` for combined application of `FCLoopIsolate` and `FCLoopSplit`

TOOLS FOR IBP-REDUCTION

- Reduction of scalar loop integrals using integration-by-parts (IBP) identities [Chetyrkin & Tkachov, 1981] is a standard technique in modern loop calculations.
- Many publicly available IBP-packages on the market: FIRE [Smirnov & Smirnov, 2013], LITERED [Lee, 2012], REDUZE [Studerus, 2009], AIR [Anastasiou & Lazopoulos, 2004], ...
- Expected input: loop integrals with propagators that form a basis.
- What about integrals with an incomplete or overdetermined basis?
  - `FCLoopBasisIncompleteQ` detects integrals that require a basis completion
  - `FCLoopBasisFindCompletion` gives a list of propagators (with zero exponents) required to complete the basis
  - `FCLoopBasisOverdeterminedQ` checks if the propagators are linearly dependent. Such integrals can be decomposed further using `ApartFF`.

In[1]:= FCI[FAD[{q1, m, 2}, {q1 + q3, m}, {q2 − q3}, q2]]

Out[1]:= $\frac{1}{(\text{q1}^2 - m^2) . (\text{q1}^2 - m^2) . ((\text{q1} + \text{q3})^2 - m^2) . (\text{q2} - \text{q3})^2 . \text{q2}^2}$

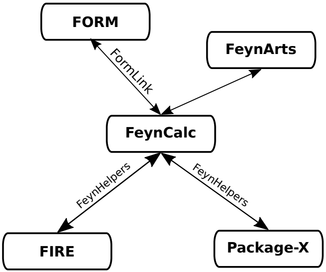
In[2]:= FCLoopBasisIncompleteQ[%, {q1, q2, q3}]
Out[2]:= **True**

In[3]:= FCLoopBasisFindCompletion[%%, {q1, q2, q3}][[2]]
Out[3]:= $\{(\text{q1} \cdot \text{q2}), (\text{q1} \cdot \text{q3})\}$

Motivation

- The field of automatic calculations appears to be a very competitive environment.
- Some groups do not share their codes at all
- Others make them available to collaborators only.
- People behind similar software regarded as competitors.
- It is more efficient to combine useful tools together than to compete.

- Useful tools to be used with FEYNCALC for the evaluation of 1-loop integrals:
  - FIRE [Smirnov, 2015]
  - PACKAGE X [Patel, 2015]
- Challenges: Need to convert between the conventions used in each package and avoid variable shadowing.
- Solution: FEYNHELPERS [Shtabovenko, 2016] seamlessly integrates both tools into FeynCalc.



Tensor reduction a la Passarino–Veltman

- Very old technique for dealing with tensor 1-loop integrals [Passarino & Veltman, 1979]
- Still widely used in many loop calculations.
- Main idea: convert all the tensor integrals into scalar ones (Passarino–Veltman coefficient functions)
- Evaluation of any 1-loop integral can be reduced to the evaluation of the resulting coefficient functions
- A lot of tools for numerical evaluation: FF [van Oldenborgh, 1991], LOOPTOOLS [Hahn & Perez-Victoria, 1999], QCDLOOP [Carrazza et al., 2016], ONELOOP [van Hameren, 2011], GOLEM95C [Cullen et al., 2011], PJFRY [Fleischer & Riemann, 2011], COLLIER [Denner et al., 2017], . . .
- Where to get analytic results for singular kinematics or zero Gram determinants? Often needed for renormalization, EFTs, . . .
- Most of the results can be found somewhere in the literature.

PACKAGE-X

- Recent [Patel, 2015] MATHEMATICA package for semi-automatic 1-loop calculations (closed-source freeware)
- Unique feature: Library of analytic expressions for Passarino–Veltman functions with up to 4 legs and almost arbitrary kinematics.
- Can also extract UV- and IR-parts and expand coefficient functions in their arguments.
- Someone indeed has collected all those results from the literature!

Interface to PACKAGE-X

- Main function: `PaXEvaluate`
- Works: on scalar 1-loop integrals (unit numerators) and Passarino–Veltman coefficient functions A, B, C and D
- Takes two arguments (plus options): input expression, loop momentum.
- Use `PaXEvaluateUV`(`PaXEvaluateIR`) to get the UV(IR)-divergent part of the result
- `PaXEvaluateUVIRSplit` returns the full result with the explicit distinction between ϵ_{UV} and ϵ_{IR}.
- All four functions share the same set of options

Let us compute $\int \frac{d^D q}{(2\pi)^D} \frac{1}{q^2 - m^2}$

In[1]:= **int=PaXEvaluate[FAD[{q,m}],q,PaXImplicitPrefactor→1/(2Pi)^D]**

Out[1]= $\frac{i m^2}{16\pi^2 \varepsilon} - \frac{i m^2 \left(-\log\left(\frac{\mu^2}{m^2}\right) + \gamma - 1 - \log(4\pi)\right)}{16\,\pi^2}$

Make the result look more compact ($\Delta \equiv 1/\epsilon - \gamma_E + \log(4\pi)$) using `FCHideEpsilon`

In[2]:= **int//FCHideEpsilon**

Out[2]= $\frac{i \Delta m^2}{16\pi^2} + \frac{i m^2 \left(\log\left(\frac{\mu^2}{m^2}\right) + 1\right)}{16\pi^2}$

Evaluation of Passarino–Veltman functions:

In[3]:= **PaXEvaluate[B0[SPD[p,p],0,m^2]]**

Out[3]= $\frac{1}{\varepsilon} + \log\left(\frac{\mu^2}{\pi m^2}\right) - \frac{m^2 \log\left(\frac{m^2}{m^2 - p^2}\right)}{p^2} + \log\left(\frac{m^2}{m^2 - p^2}\right) - \gamma + 2$

We can also expand coefficient functions in their parameters (masses or external momenta). To expand $B_0(p^2, 0, m^2)$ around $p^2 = m^2$ up to first order with `PaXEvaluate` we first need to assign an arbitrary symbolic value to the scalar product p^2, e.g. `pp`

In[4]:= **SPD[p,p]=pp;**

Then use the option `PaXSeries` to specify the expansion parameters and activate the option `PaXAnalytic`

In[5]:= **PaXEvaluate[B0[SPD[p,p],0,m^2],PaXSeries→{{pp,m^2,1}},PaXAnalytic→True]**

Out[5]= $\frac{3\,\mathrm{m}^2-\mathrm{pp}}{2\varepsilon \mathrm{m}^2}-\frac{(3\,\mathrm{m}^2-\mathrm{pp})\left(-\log\left(\frac{\mu^2}{\mathrm{m}^2}\right)+\gamma-2+\log(\pi)\right)}{2\mathrm{m}^2}$

Get only in the UV-part of this series: `PaXEvaluate` with `PaXEvaluateUV`

In[6]:= **PaXEvaluateUV[B0[SPD[p,p],0,m^2],PaXSeries→{{pp,m^2,1}},PaXAnalytic→True]**

Out[6]= $\frac{1}{\varepsilon_{\mathrm{UV}}}$

The IR-part is equally easy

In[7]:= **PaXEvaluateIR[B0[SPD[p,p],0,m^2],PaXSeries→{{pp,m^2,1}},PaXAnalytic→True]**

Out[7]= $\frac{\mathrm{m}^2-\mathrm{pp}}{2\mathrm{m}^2\varepsilon_{\mathrm{IR}}}$

Full result with the explicit distinction between UV and IR singularities

In[8]:= **PaXEvaluateUVIRSplit[B0[SPD[p,p],0,m^2],PaXSeries→{{pp,m^2,1}},PaXAnalytic→True]**

Out[8]= $\frac{\mathrm{m}^2-\mathrm{pp}}{2\mathrm{m}^2\varepsilon_{\mathrm{IR}}}-\frac{(3\,\mathrm{m}^2-\mathrm{pp})\left(-\log\left(\frac{\mu^2}{\mathrm{m}^2}\right)+\gamma-2+\log(\pi)\right)}{2\,\mathrm{m}^2}+\frac{1}{\varepsilon_{\mathrm{UV}}}$

Interface to FIRE

- Main function: `FIREBurn`
- Reduces scalar multi-loop integrals to simpler ones using IBP-techniques.
- Takes three arguments (plus options): input expression, list of loop momenta and the list of external momenta.
- Automatically adds propagators to integrals with incomplete bases of propagators
- Automatically detects integrals with linearly dependent propagators

Current limitations

- No recognition of integral families
- Each loop integral is evaluated separately
- Hence, rather inefficient ...

IBP-reduce the 1-loop integral $\int \frac{d^D l}{[l^2]^2[(l-p)^2-m^2]^2}$

In[9]:= **FIREBurn[FAD[{l,0,2},{l−p,m,2}],{l},{p}]**

Out[9]= $\frac{(D-2)(2\,D\,m^2-9\,m^2-pp)}{2m^2(m^2-pp)^3((l-p)^2-m^2)} - \frac{(D-3)(D\,m^2+D\,pp-4m^2-6\,pp)}{(m^2-pp)^3 l^2.((l-p)^2-m^2)}$

No dependence on external momenta → supply an empty list for the third argument. For example, for $\int \frac{d^D q_1 d^D q_2 d^D q_3}{[q_1^2-m^2]^2[(q_1+q_3)^2-m^2][(q_2-q_3)^2][q_2^2]^2}$

In[10]:= **FIREBurn[FAD[{q1,m,2},{q1+q3,m},{q2−q3},{q2,0,2}],{q1,q2,q3},{}]**

Out[10]= $-\frac{(D-3)(3\,D-10)(3\,D-8)}{16(2\,D-7)m^4(q1^2-m^2).q2^2.(q2-q3)^2.((q1+q3)^2-m^2)}$

My favourite example: Calculation of the QCD on-shell vertex for QCD/NRQCD matching [Manohar, 1997]

The HQET/NRQCD Lagrangian to order α/m^3

Aneesh V. Manohar

Department of Physics, University of California at San Diego,

9500 Gilman Drive, La Jolla, CA 92093-0319

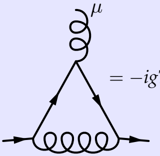
(January 1997)

Abstract

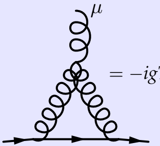
The HQET/NRQCD Lagrangian is computed to order α/m^3. The computation is performed using dimensional regularization to regulate the ultraviolet and infrared divergences. The results are consistent with reparametrization invariance to order $1/m^3$. Some subtleties in the matching conditions for NRQCD are discussed.

Reproducing results of Manohar

- QCD side of the matching: The on-shell vertex function is evaluated using background field formalism [Abbott, 1981, Abbott, 1982] and expanded up to the first order in the relative momentum squared.
- The abelian and non-abelian diagrams can be parametrized as



$$= -igT^a \bar{u}(p_2) \left(F_1^{(V)}(q^2)\gamma^\mu + iF_2^{(V)}(q^2)\frac{\sigma^{\mu\nu}q_\nu}{2m} \right) u(p_1), \qquad (1)$$



$$= -igT^a \bar{u}(p_2) \left(F_1^{(g)}(q^2)\gamma^\mu + iF_2^{(g)}(q^2)\frac{\sigma^{\mu\nu}q_\nu}{2m} \right) u(p_1), \qquad (2)$$

where $q \equiv p_2 = p_1$. Our goal is to compute the form-factors $F_{1/2}^{(V)}(q^2)$ and $F_{1/2}^{(g)}(q^2)$ expanded up to $\mathcal{O}(q^2/m^2)$.

Not so simple to do with software

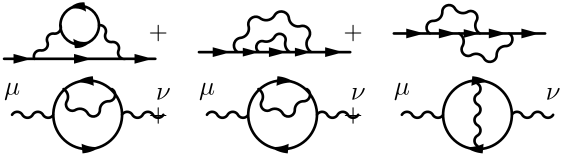
- It is not a total cross-section/decay rate, so fully automatic tools are not useful.
- Need to expand Passarino–Veltman integrals in the relative momentum.
- Distinguish between UV and IR singularities in DR using different regulators ϵ_{UV} and ϵ_{IR}.
- Since this is a matching, we want analytic results.
- With FEYNHELPERS this computation is straight-forward. We use the abbreviation $\Delta \equiv 1/\epsilon - \gamma_E + \log(4\pi)$ and use $D = 4 - 2\epsilon$
- To compare to the literature we need to switch to $D = 4 - \epsilon$ via $1/\epsilon \to 2/\epsilon$ and eliminate γ_E and $\log(4\pi)$ by substituting μ^2 with $\mu^2 \frac{e^{\gamma_E}}{4\pi}$ (following the conventions of Manohar).

$$F_1^{(g)} = \frac{\alpha_s}{8\pi} C_2(ad) \int_0^1 dx \int_0^{1-x} dy \Big\{ -\Gamma(1+\epsilon/2) \big[2q^2 (x+y) + 2m^2 (1-x-y)(2(x+y) + (4-\epsilon)(1-x-y)) \big] \left(m^2 (x+y-1)^2 - q^2 xy \right)^{-1-\epsilon/2} + (2-\epsilon)\Gamma(\epsilon/2) \left(m^2 (x+y-1)^2 - q^2 xy \right)^{-\epsilon/2} \Big\}$$
$$= \frac{\alpha_s}{8\pi} C_2(ad) \left[\frac{2}{\epsilon_{UV}} + \frac{4}{\epsilon_{IR}} + 4 - 6 \log \frac{m}{\mu} + \frac{q^2}{m^2} \left(-\frac{3}{\epsilon_{IR}} - 1 + 3 \log \frac{m}{\mu} \right) + \ldots \right], \quad (31)$$

$$F_2^{(g)} = -\frac{\alpha_s}{4\pi} C_2(ad)\, m^2 \Gamma(1+\epsilon/2) \int_0^1 dx \int_0^{1-x} dy\, (1-x-y) \times (\epsilon + (2-\epsilon)(x+y)) \left(m^2 (x+y-1)^2 - q^2 xy \right)^{-1-\epsilon/2}$$
$$= \frac{\alpha_s}{8\pi} C_2(ad) \left[\frac{4}{\epsilon_{IR}} + 6 - 4 \log \frac{m}{\mu} + \frac{q^2}{m^2} \left(\frac{4}{\epsilon_{IR}} + 1 - 4 \log \frac{m}{\mu} \right) + \ldots \right]. \quad (32)$$

$$F_1^{(V)} = \frac{\alpha}{\pi} \left(C_2(Q) - \frac{1}{2} C_2(ad) \right) \left[\frac{1}{2\epsilon_{UV}} + \frac{1}{\epsilon_{IR}} + 1 - \frac{3}{2} \log \frac{m}{\mu} + \frac{q^2}{m^2} \left(-\frac{1}{3} \frac{1}{\epsilon_{IR}} - \frac{1}{8} + \frac{1}{3} \log \frac{m}{\mu} \right) \right], \quad (29)$$
$$F_2^{(V)} = \frac{\alpha}{\pi} \left(C_2(Q) - \frac{1}{2} C_2(ad) \right) \left[\frac{1}{2} + \frac{q^2}{12 m^2} \right]. \quad (30)$$

Another example: photon and electron self-energies (with full gauge dependence) in massless QED at 2-loops. Requires evaluation of six 2-loop diagrams



$$\equiv i\not{p}\Sigma_{2V}(p^2),$$

$$\equiv -i(p^2 g^{\mu\nu} - p^\mu p^\nu)\Pi_2(p^2),$$

- Final results contain only two master integrals
- Need to use `FCMultiLoopTID` instead of `TID`

As expected, the vacuum polarization amplitude is gauge invariant, while the electron self-energy depends on the gauge parameter ξ. These results precisely agree with the literature, e.g. Eq. 5.18 and Eq. 5.51 from [Grozin, 2005].

- With FeynHelpers many types of calculations that were difficult or hardly feasible with FeynCalc previously become very simple.
- Goals for future development: improve the integration with Package-X and FIRE but also to add new interfaces to interesting and useful HEP tools.
- FeynHelpers comes with many examples. Highlight: 1-loop QED renormalization in MS, $\overline{\mathrm{MS}}$ and on-shell schemes with full gauge dependence (also useful for teaching).

Abbott, L. (1981).
The background field method beyond one loop.
Nucl. Phys. B, 185, 189–203.

Abbott, L. (1982).
Introduction to the Background Field Method.
Acta Phys. Polon., B13, 33.

Anastasiou, C. & Lazopoulos, A. (2004).
Automatic Integral Reduction for Higher Order Perturbative Calculations.
JHEP, 0407, 046.

Belanger, G., Boudjema, F., Fujimoto, J., Ishikawa, T., Kaneko, T., Kato, K., & Shimizu, Y. (2006).
GRACE at ONE-LOOP: Automatic calculation of 1-loop diagrams in the electroweak theory with gauge parameter independence checks.
Phys. Rept., 430, 117–209.

Bodwin, G. T., Braaten, E., & Lepage, G. P. (1995).
Rigorous QCD Analysis of Inclusive Annihilation and Production of Heavy Quarkonium.
Phys. Rev. D, 51, 1125–1171.

Borowka, S., Heinrich, G., Jahn, S., Jones, S. P., Kerner, M., Schlenk, J., & Zirke, T.
pySecDec: a toolbox for the numerical evaluation of multi-scale integrals.

Brambilla, N., Pineda, A., Soto, J., & Vairo, A. (2000).
Potential NRQCD: an effective theory for heavy quarkonium.
Nucl. Phys. B, 566, 275.

Carrazza, S., Ellis, R. K., & Zanderighi, G. (2016).
QCDLoop: a comprehensive framework for one-loop scalar integrals.
Comput. Phys. Commun., 209, 134–143.

Caswell, W. & Lepage, G. (1986).
Effective lagrangians for bound state problems in QED, QCD, and other field theories.
Phys. Lett. B, 167(4), 437–442.

Chetyrkin, K. & Tkachov, F. (1981).
Integration by parts: The algorithm to calculate β-functions in 4 loops.
Nucl. Phys. B, 192(1), 159–204.

Christensen, N. D. & Duhr, C. (2008).
FeynRules - Feynman rules made easy.
Comput. Phys. Commun., 180, 1614–1641.

Cullen, G., Guillet, J. P., Heinrich, G., Kleinschmidt, T., Pilon, E., Reiter, T., & Rodgers, M. (2011).
Golem95C: A library for one-loop integrals with complex masses.
Comput. Phys. Commun., 182, 2276–2284.

Cullen, G., van Deurzen, H., Greiner, N., Heinrich, G., Luisoni, G., Mastrolia, P., Mirabella, E., Ossola, G., Peraro, T., Schlenk, J., von Soden-Fraunhofen, J. F., & Tramontano, F. (2014).
GoSam-2.0: a tool for automated one-loop calculations within the Standard Model and beyond.
Eur. Phys. J. C, 74, 8, 3001.

Cyrol, A. K., Mitter, M., & Strodthoff, N. (2016).
FormTracer - A Mathematica Tracing Package Using FORM.

Denner, A., Dittmaier, S., & Hofer, L. (2017).
Collier: a fortran-based Complex One-Loop LIbrary in Extended Regularizations.
Comput. Phys. Commun., 212, 220–238.

Feng, F. (2012).
$Apart: A Generalized Mathematica Apart Function.
Comput. Phys. Commun., 183, 2158–2164.

Fleischer, J. & Riemann, T. (2011).
A complete algebraic reduction of one-loop tensor Feynman integrals.
Phys. Rev. D, 83, 073004.

Grozin, A. (2005).
Lectures on QED and QCD.
In *3rd Dubna International Advanced School of Theoretical Physics Dubna, Russia, January 29-February 6, 2005* (pp. 1–156).

Hahn, T. & Perez-Victoria, M. (1999).
Automatized One-Loop Calculations in 4 and D dimensions.
Comput. Phys. Commun., 118, 153–165.

Jamin, M. & Lautenbacher, M. E. (1993).
TRACER version 1.1.
Comput. Phys. Commun., 74(2), 265–288.

Lee, R. N. (2012).
Presenting LiteRed: a tool for the Loop InTEgrals REDuction.

Manohar, A. (1997).
The HQET/NRQCD Lagrangian to order α/m^3.

Phys. Rev. D, 56, 230–237.

Mertig, R., Böhm, M., & Denner, A. (1991).
Feyn Calc - Computer-algebraic calculation of Feynman amplitudes.
Comput. Phys. Commun., 64(3), 345–359.

Nogueira, P. (1993).
Automatic Feynman graph generation.
J. Comput. Phys., 105, 279–289.

Passarino, G. & Veltman, M. (1979).
One Loop Corrections for e^+e^- Annihilation Into $\mu^+\mu^-$ in the Weinberg Model.
Nucl. Phys., B160, 151.

Patel, H. H. (2015).
Package-X: A Mathematica package for the analytic calculation of one-loop integrals.
Comput. Phys. Commun., 197, 276–290.

Pineda, A. & Soto, J. (1998a).
Effective Field Theory for Ultrasoft Momenta in NRQCD and NRQED.
Nucl.Phys.Proc.Suppl., 64, 428–432.

Pineda, A. & Soto, J. (1998b).

The Lamb Shift in Dimensional Regularization.
Phys. Lett. B, 420, 391–396.

Ruijl, B., Ueda, T., & Vermaseren, J. A. M. (2017).
Forcer, a FORM program for the parametric reduction of four-loop massless propagator diagrams.

Shtabovenko, V. (2016).
FeynHelpers: Connecting FeynCalc to FIRE and Package-X.

Shtabovenko, V., Mertig, R., & Orellana, F. (2016).
New Developments in FeynCalc 9.0.
Comput. Phys. Commun., 207, 432–444.

Smirnov, A. V. (2015).
FIRE5: a C++ implementation of Feynman Integral REduction.
Comput. Phys. Commun., 189, 182–191.

Smirnov, A. V. & Smirnov, V. A. (2013).
FIRE4, LiteRed and accompanying tools to solve integration by parts relations.
Comput. Phys. Commun., 184(12), 2820–2827.

Studerus, C. (2009).

Reduze - Feynman Integral Reduction in C++.
Comput. Phys. Commun., 181, 1293–1300.

Tentyukov, M. & Fleischer, J. (2000).
A Feynman Diagram Analyser DIANA.
Comput. Phys. Commun., 132, 124–141.

van Hameren, A. (2011).
OneLOop: for the evaluation of one-loop scalar functions.
Comput. Phys. Commun., 182, 2427–2438.

van Oldenborgh, G. (1991).
FF - a package to evaluate one-loop Feynman diagrams.
Comput. Phys. Commun., 66(1), 1–15.

Wang, J.-X. (2004).
Progress in FDC project.
Nucl. Instrum. Meth. A, 534(1-2), 241–245.

Wiebusch, M. (2014).
HEPMath: A Mathematica Package for Semi-Automatic Computations in High Energy Physics.
Comput. Phys. Commun., 195, 172–190.