FeynCalc 9

Vladyslav Shtabovenko

Technische Universität München, Physik-Department T30f, James-Franck-Str. 1, 85747
Garching, Germany

E-mail: `v.shtabovenko@tum.de`

Abstract. We report on a new version of FeynCalc, a well-known Mathematica package for symbolic computations in quantum field theory and provide some explicit examples for using the software in different types of calculations.

1. Introduction

Modern perturbative calculations in quantum field theory (QFT) often heavily rely on different software tools that are used to automatize both the algebraic and the numerical stages of the evaluation.

In the last decades, much effort was invested to develop packages (e.g. FormCalc [1], GoSam [2], GRACE [3]) that are able to perform all the steps needed to arrive to physical observables (e.g. decay rates or cross-sections) in a highly automatized way. The level of the automation in such packages is so advanced, that in many cases the user just has to specify the incoming and outgoing particles, while the underlying code will take care of everything else and produce final results.

A complementary approach is represented by tools (e.g. HEPMath [4], Package-X [5]) that do not attempt to automatize everything bur rather provide a collection of easy to use instruments for accomplishing most common tasks in QFT calculations, like contractions of Lorentz indices, calculation of Dirac traces or tensor decomposition of loop integrals. Such semi-automatic packages give the user much flexibility but also require him or her to carefully handle every step of the calculation. This bears a strong resemblance to calculations done by pen and paper, with the difference that one can proceed in a much faster way .

While both philosophies have their supporters in the high energy physics community, one should of course understand that they are not competing with each other but rather cover different use cases. On the one hand, many tree-level and 1-loop calculations in Standard Model and its popular (e.g. supersymmetric) extensions can be nowadays carried out in a highly automatic fashion, since the essential steps of these calculations are sufficiently well understood. It is therefore natural to use suitable packages like FormCalc or GoSam to study such processes.

On the other hand, multi-loop calculations, determination of matching coefficients in effective field theories or evaluation of Feynman diagrams in non-relativistic QFTs are good examples for tasks that are too complex or non-standard to be automatized in full generality. Although for such projects people often prefer to write their own code from scratch (e.g. implemented in FORM [6] or Mathematica), it can also be useful to use different instruments provided by publicly available semi-automatic packages instead of or in addition to the private codes.

FeynCalc belongs to the oldest semi-automatic packages that are still actively developed and used in research. The first version of the program was published almost 25 years ago [7] and followed the idea to provide a flexible and easy to use Mathematica package for symbolic calculations and for the evaluation of Feynman diagrams in QFT at tree-level and at 1-loop. In this context, "symbolic calculations" means that FeynCalc can deal not only with Feynman diagrams but also standalone QFT expressions like $\int \frac{d^D q}{(2\pi)^D} \frac{l^\mu l^\nu}{l^2 - m^2}$ or $\mathrm{Tr}(\gamma^\mu \gamma^\nu \gamma^\rho \gamma^\sigma)$. The users of FeynCalc are therefore not forced to follow a particular workflow (e.g. first enter the Lagrangian, then generate Feynman diagrams and finally evaluate them) but are free to use the software as some sort of "calculator" for QFT expressions. This design philosophy has been preserved over the years and made FeynCalc a valuable tool for many QFT practitioners.

This note is organized in the following way. In Sec. 2 we briefly describe the most interesting new features of FeynCalc 9 and provide some examples for purposes of illustration. Sec. 3 shows the calculation of the gluon self-energy at 1-loop in QCD, where the final result will be presented in terms of the Passarino-Veltman scalar functions A_0 and B_0. In Sec. 4 we summarize and mention main directions for the future development of FeynCalc.

2. New features in FeynCalc 9

FeynCalc 9 was released in January 2016 [8]. While the development of the program between 2001 and 2014 was mostly limited to providing fixes for the discovered bugs, this version also introduces some new features that are mostly related to the evaluation of 1-loop and multi-loop integrals. Version 9 requires at least Mathematica 8 and can be installed by just evaluating

```
Import["https://raw.githubusercontent.com/FeynCalc/feyncalc/master/install.m"]
InstallFeynCalc[]
```

in a Mathematica notebook. The automatic installer also handles the setup of FeynArts [9], a Mathematica package that can generate Feynman diagrams, which can be then (after some conversion) evaluated with FeynCalc. An example for the conversion from FeynArts to FeynCalc will be described in Sec. 3.

Tensor decomposition of 1-loop integrals using the `TID` routine has received a lot of improvements in FeynCalc 9 and can in principle handle 1-loop tensor integrals of arbitrary rank and multiplicity. Provided that there are no vanishing Gram determinants, one can choose (via the option `UsePaVeBasis`) between the full decomposition into Passarino-Veltman scalar functions

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

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

```
In[2]:= TID[int, q]//ToPaVe[#, q]&
```

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

and the decomposition into Passarino-Veltman coefficient functions

```
In[3]:= TID[int, q, UsePaVeBasis −> True, PaVeAutoReduce −> False] //
        ToPaVe[#, q] &
```

$$\mathbf{Out}[3]:= i\pi^2\mathrm{B}_0\left(p^2,0,m^2\right)(Dm - D\gamma\cdot p + 2\gamma\cdot p) - i\pi^2(D-2)\gamma\cdot p\mathrm{B}_1\left(p^2,0,m^2\right)$$

Zero Gram determinants are automatically recognized and force the algorithm to avoid the full decomposition

In[4]:= **FCClearScalarProducts**[]
SPD[p1, p2] = x;
SPD[p1, p1] = x;
SPD[p2, p2] = x;
FCI[**FVD**[l, μ] **FVD**[l, ν] **FVD**[l, ρ] **FAD**[{l, m0}, {l + p1, m1}, {l + p2, m2}]]

$$\textbf{Out}[4]:= \frac{l^{\mu}l^{\nu}l^{\rho}}{\left(l^2-\mathsf{m0}^2\right).\left((l+\mathsf{p1})^2-\mathsf{m1}^2\right).\left((l+\mathsf{p2})^2-\mathsf{m2}^2\right)}$$

In[5]:= **TID**[(1/(**I Pi**^ 2)) %, l]

$$\begin{aligned}\textbf{Out}[6]:= &\left(\mathsf{p1}^{\rho}g^{\mu\nu}+\mathsf{p1}^{\nu}g^{\mu\rho}+\mathsf{p1}^{\mu}g^{\nu\rho}\right)\mathsf{C}_{001}\left(x,0,x,\mathsf{m0}^2,\mathsf{m1}^2,\mathsf{m2}^2\right)\\ &+\left(\mathsf{p2}^{\rho}g^{\mu\nu}+\mathsf{p2}^{\nu}g^{\mu\rho}+\mathsf{p2}^{\mu}g^{\nu\rho}\right)\mathsf{C}_{002}\left(x,0,x,\mathsf{m0}^2,\mathsf{m1}^2,\mathsf{m2}^2\right)\\ &+\mathsf{p1}^{\mu}\mathsf{p1}^{\nu}\mathsf{p1}^{\rho}\mathsf{C}_{111}\left(x,0,x,\mathsf{m0}^2,\mathsf{m1}^2,\mathsf{m2}^2\right)\\ &+\left(\mathsf{p1}^{\nu}\mathsf{p1}^{\rho}\mathsf{p2}^{\mu}+\mathsf{p1}^{\mu}\mathsf{p1}^{\rho}\mathsf{p2}^{\nu}+\mathsf{p1}^{\mu}\mathsf{p1}^{\nu}\mathsf{p2}^{\rho}\right)\mathsf{C}_{112}\left(x,0,x,\mathsf{m0}^2,\mathsf{m1}^2,\mathsf{m2}^2\right)\\ &+\left(\mathsf{p1}^{\rho}\mathsf{p2}^{\mu}\mathsf{p2}^{\nu}+\mathsf{p1}^{\nu}\mathsf{p2}^{\mu}\mathsf{p2}^{\rho}+\mathsf{p1}^{\mu}\mathsf{p2}^{\nu}\mathsf{p2}^{\rho}\right)\mathsf{C}_{122}\left(x,0,x,\mathsf{m0}^2,\mathsf{m1}^2,\mathsf{m2}^2\right)\\ &+\mathsf{p2}^{\mu}\mathsf{p2}^{\nu}\mathsf{p2}^{\rho}\mathsf{C}_{222}\left(x,0,x,\mathsf{m0}^2,\mathsf{m1}^2,\mathsf{m2}^2\right)\end{aligned}$$

where we multiplied the result by $\frac{1}{i\pi^2}$ to cancel the normalization factor that appears in the conversion of loop integrals to Passarino-Veltman functions.

With the new function `FCMultiLoopTID` it is now also possible to compute tensor decompositions of multi-loop integrals, although in this case a special treatment of integrals with zero Gram determinants is not available yet.

In[7]:= **FCI**[**FVD**[q1, μ] **FVD**[q2, ν] **FAD**[q1, q2, {q1 − p1}, {q2 − p1}, {q1 − q2}]]

$$\textbf{Out}[7]:= \frac{\mathsf{q1}^{\mu}\mathsf{q2}^{\nu}}{\mathsf{q1}^2.\mathsf{q2}^2.(\mathsf{q1}-\mathsf{p1})^2.(\mathsf{q2}-\mathsf{p1})^2.(\mathsf{q1}-\mathsf{q2})^2}$$

In[8]:= **FCMultiLoopTID**[%, {q1, q2}]

$$\begin{aligned}\textbf{Out}[8]:= &\frac{D\mathsf{p1}^{\mu}\mathsf{p1}^{\nu}-\mathsf{p1}^2g^{\mu\nu}}{4(D-1)\mathsf{q2}^2.\mathsf{q1}^2.(\mathsf{q2}-\mathsf{p1})^2.(\mathsf{q1}-\mathsf{q2})^2.(\mathsf{q1}-\mathsf{p1})^2}-\\ &\frac{\mathsf{p1}^2g^{\mu\nu}-\mathsf{p1}^{\mu}\mathsf{p1}^{\nu}}{2(D-1)\mathsf{p1}^2\mathsf{q2}^2.\mathsf{q1}^2.(\mathsf{q2}-\mathsf{p1})^2.(\mathsf{q1}-\mathsf{p1})^2}+\frac{\mathsf{p1}^2g^{\mu\nu}-\mathsf{p1}^{\mu}\mathsf{p1}^{\nu}}{(D-1)\mathsf{p1}^2\mathsf{q2}^2.\mathsf{q1}^2.(\mathsf{q1}-\mathsf{q2})^2.(\mathsf{q1}-\mathsf{p1})^2}-\frac{D\mathsf{p1}^{\mu}\mathsf{p1}^{\nu}-\mathsf{p1}^2g^{\mu\nu}}{2(D-1)\mathsf{p1}^4\mathsf{q1}^2.(\mathsf{q2}-\mathsf{p1})^2.(\mathsf{q1}-\mathsf{q2})^2}\end{aligned}$$

Simplification of scalar loop integrals with linearly dependent propagators is another feature of FeynCalc that was greatly improved in the version 9 by adopting the partial fractioning algorithm of F. Feng [10]. The corresponding function ApartFF is not only faster and more efficient than the old ScalarProductCancel (which is now considered legacy), but is also capable to handle multi-loop integrals

In[9]:= **FCI**[**SPD**[p, q1] ^2 **SPD**[p, q2] **FAD**[{q1, m}, {q2, m}, q1 − p, q2 − p, q1 − q2]]

$$\textbf{Out}[9]:= \frac{(p\cdot\mathsf{q2})(p\cdot\mathsf{q1})^2}{\left(\mathsf{q1}^2-m^2\right).\left(\mathsf{q2}^2-m^2\right).(\mathsf{q1}-p)^2.(\mathsf{q2}-p)^2.(\mathsf{q1}-\mathsf{q2})^2}$$

In[10]:= **ApartFF**[%,{q1,q2}]

$$\begin{aligned}\textbf{Out}[10]:= &\frac{\left(m^2+p^2\right)^3}{8\left(\mathsf{q1}^2-m^2\right).\left(\mathsf{q2}^2-m^2\right).(\mathsf{q2}-p)^2.(\mathsf{q1}-\mathsf{q2})^2.(\mathsf{q1}-p)^2}-\frac{\left(m^2+p^2\right)^2}{4\left(\mathsf{q1}^2-m^2\right).\left(\mathsf{q2}^2-m^2\right).(\mathsf{q1}-\mathsf{q2})^2.(\mathsf{q1}-p)^2}\\ &+\frac{\left(m^2+p^2\right)^2}{4\mathsf{q2}^2.\mathsf{q1}^2.\left((\mathsf{q1}-p)^2-m^2\right).(\mathsf{q1}-\mathsf{q2})^2}+\frac{\left(m^2+p^2\right)\left(p^2-p\cdot\mathsf{q1}\right)}{4\mathsf{q2}^2.\mathsf{q1}^2.(\mathsf{q1}-\mathsf{q2})^2.\left((\mathsf{q2}-p)^2-m^2\right)}\\ &-\frac{\left(m^2+p^2\right)(p\cdot\mathsf{q1})}{4\left(\mathsf{q1}^2-m^2\right).\left(\mathsf{q2}^2-m^2\right).(\mathsf{q2}-p)^2.(\mathsf{q1}-\mathsf{q2})^2}-\frac{p\cdot\mathsf{q1}}{4\left(\mathsf{q2}^2-m^2\right).(\mathsf{q1}-\mathsf{q2})^2.(\mathsf{q1}-p)^2}-\frac{m^2+p\cdot\mathsf{q1}+p^2}{4\left(\mathsf{q1}^2-m^2\right).(\mathsf{q2}-p)^2.(\mathsf{q1}-\mathsf{q2})^2}\\ &+\frac{m^2+2(p\cdot\mathsf{q1})+p^2}{8\left(\mathsf{q1}^2-m^2\right).\left(\mathsf{q2}^2-m^2\right).(\mathsf{q2}-\mathsf{q1})^2}\end{aligned}$$

A more detailed description of these and other new and improved functions can be found in [8].

3. Gluon self-enery at 1-loop in QCD

To make some connection to more realistic examples, let us look at the 1-loop gluon self-energy in QCD. While this is, of course, a standard textbook calculation that can be done by pen and paper with comparably little effort, it shows how FeynCalc can be used to evaluate Feynman diagrams generated by FeynArts.

The first step is always to load FeynCalc with the global option `$LoadFeynArts` set to `True`, which ensures that the patched copy of FeynArts, that usually resides inside the FeynCalc installation, will we loaded in a proper way. It is also convenient to set the option `$FAVerbose` to `0`, which prevents FeynArts from displaying too much extra information.

```
In[1]:= $LoadFeynArts = True;
<< FeynCalc`
$FAVerbose = 0;
```

Then we can use FeynArts' functions `CreateToplogies` and `InsertFields` [1] to generate the diagrams that we want to evaluate.

```
In[2]:= diags = InsertFields[CreateTopologies[1, 1 -> 1 , ExcludeTopologies -> {Tadpoles}],
       {V[5]} -> {V[5]}, InsertionLevel -> {Classes}, Model -> "SMQCD",
        ExcludeParticles  -> {S[_], V[2 | 3], U[1 | 2 | 3 | 4], F [4]}];
```

With the following code

```
In[3]:= amps = FCFAConvert[CreateFeynAmp[diags, Truncated -> True, PreFactor -> -1],
    IncomingMomenta -> {p}, OutgoingMomenta -> {p}, LoopMomenta -> {q},
    DropSumOver -> True, ChangeDimension -> D, UndoChiralSplittings -> True,
  LorentzIndexNames -> {μ, ν}] /. {MQU[Index[Generation, 3]] -> m};
```

the amplitudes generated by FeynArts are converted into a valid FeynCalc input. Here the output of the standard FeynArts function `CreateFeynAmp` is passed to `FCFAConvert`, a new FeynCalc routine that handles the conversion according to the options submitted by the user. The resulting expression `amps` is a list with four entries that correspond to the four diagrams displayed in Fig. 1, where for convenience the FeynArts expression for the up-type quark mass `MQU[Index[Generation, 3]]` was replaced by `m`.

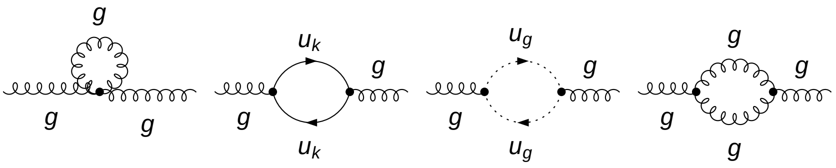

Figure 1. 1-loop contributions to the gluon self-energy in QCD

These four diagrams can be evaluated with

```
In[4]:= amps2 = TID[#, q] & /@ SUNSimplify /@ Contract /@ amps;
```

where we apply the FeynCalc functions `Contract` (contraction of Lorentz indices), `SUNSimplify` (simplification of the $SU(N)$ algebra) and `TID` (tensor reduction of 1-loop integrals) to each amplitude. We known that the first amplitude in Fig. 1 vanishes in dimensional regularization, as it is proportional to a scaleless loop integral. FeynCalc confirms that this is indeed the case

```
In[5]:= amps2[[1]]
Out[5]:= 0
```

[1] The syntax for these and other FeynArts functions can be found in the "FeynArts User's Guide", available from `www.feynarts.de`

Summing up the contributions from all the diagrams (for simplicity we consider here only one quark flavor) we obtain

In[6]:= ampsTotal = **Total**[amps2] // **Simplify**

Out[6]:= $$\frac{g_s^2 \delta^{\mathsf{Glu1Glu2}}\left(p^\mu p^\nu - p^2 g^{\mu\nu}\right)\left(\frac{(3D-2)p^2 C_A}{q^2.(q-p)^2} - \frac{2\left((D-2)p^2+4m^2\right)}{\left(q^2-m^2\right).\left((q-p)^2-m^2\right)} + \frac{4(D-2)}{q^2-m^2}\right)}{2(D-1)p^2}$$

For convenience, we may also rewrite this result in terms of the Passarino-Veltman scalar functions A_0 and B_0 by applying the function `ToPaVe` that is available since FeynCalc 9

In[7]:= res = ampsTotal // **ToPaVe**[#, q] & // **Simplify**

Out[7]:= $$\frac{i\pi^2 g_s^2 \delta^{\mathsf{Glu1Glu2}}\left(p^2 g^{\mu\nu} - p^\mu p^\nu\right)\left((2-3D)p^2 C_A \mathsf{B}_0\left(p^2,0,0\right)+2\left((D-2)p^2+4m^2\right)\mathsf{B}_0\left(p^2,m^2,m^2\right)-4(D-2)\mathsf{A}_0\left(m^2\right)\right)}{2(D-1)p^2}$$

Such an output is the simplest possible form that can be achieved with FeynCalc. However, with the aid of some auxiliary tools one can in principle obtain even more explicit results with the 1-loop master integrals evaluated analytically. An add-on called FeynHelpers that connects FeynCalc to such tools is currently in development, with the early version of the code already available for testing [2]

4. Summary

In summary, we described some of the new developments in FeynCalc 9 and provided an explicit example for using the combination of FeynCalc and FeynArts to compute the 1-loop gluon self-energy in QCD. While better support for multi-loop calculations is one of the important goals in the future development of FeynCalc, we are also looking into possibilities to combine FeynCalc with other useful software tools by developing the corresponding interfaces.

Acknowledgments

This work has been supported by the DFG and the NSFC through funds provided to the Sino-German CRC 110 “Symmetries and the Emergence of Structure in QCD”, and by the DFG cluster of excellence “Origin and structure of the universe” (`www.universe-cluster.de`). This report has the preprint number TUM-EFT 78/16.

References

[1] Hahn T and Perez-Victoria M 1999 *Comput. Phys. Commun.* **118** 153–165 (*Preprint* `hep-ph/9807565`)

[2] 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 and Tramontano F 2014 *Eur. Phys. J. C* **74** 8, 3001 (*Preprint* `1404.7096`)

[3] Belanger G, Boudjema F, Fujimoto J, Ishikawa T, Kaneko T, Kato K and Shimizu Y 2006 *Phys. Rept.* **430** 117–209 (*Preprint* `hep-ph/0308080`)

[4] Wiebusch M 2014 *Comput. Phys. Commun.* **195** 172–190 (*Preprint* `1412.6102`)

[5] Patel H H 2015 *Comput. Phys. Commun.* **197** 276–290 (*Preprint* `1503.01469`)

[6] Vermaseren J A M 2007 (*Preprint* `math-ph/0010025`)

[7] Mertig R, Böhm M and Denner A 1991 *Comput. Phys. Commun.* **64** 345–359

[8] Shtabovenko V, Mertig R and Orellana F 2016 (*Preprint* `1601.01167`)

[9] Hahn T 2001 *Comput. Phys. Commun.* **140** 418–431 (*Preprint* `hep-ph/0012260`)

[10] Feng F 2012 *Comput. Phys. Commun.* **183** 2158–2164 (*Preprint* `1204.2314`)

[2] github.com/FeynCalc/feynhelpers