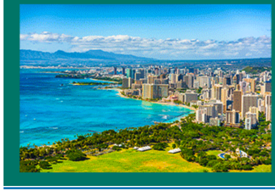
PAPER • **OPEN ACCESS**

FeynCalc goes multiloop

To cite this article: Vladyslav Shtabovenko 2023 *J. Phys.: Conf. Ser.* **2438** 012140



TTP21-060, P3H-21-101

FeynCalc goes multiloop

Vladyslav Shtabovenko

Institut für Theoretische Teilchenphysik (TTP), Karlsruhe Institute of Technology (KIT), Wolfgang-Gaede-Straße 1, 76131 Karlsruhe, Germany

E-mail: `v.shtabovenko@kit.edu`

Abstract. We report on the new functionality of the open-source Mathematica package FeynCalc relevant for multiloop calculations. In particular, we focus on such tasks as topology identification by means of the Pak algorithm, search for equivalent master integrals and their graph representations as well as automatic derivations of Feynman parametric representations for a wide class of multiloop integrals. The functions described in this report are expected to be finalized with the official release of FeynCalc 10. The current development snapshot of the package including the documentation is publicly available on the project homepage. User feedback is highly encouraged.

1. Introduction

There is a widespread consensus among particle theorists that in order to match the expected experimental precision at the High Luminosity Large Hadron Collider (HL-LHC) we need new ideas and algorithms for our analytical and numerical calculations. This is especially true for the field of multiloop computations (cf. e.g. [1] for an overview of recent developments), where analytic evaluations of hundreds, thousands and sometimes even millions of Feynman diagrams often require the practitioners to expand the borders of what is feasible using computer algebraic methods. Despite such remarkable advances that corroborate the healthy state of the field, one should not forget that breakthrough calculations with many loops or legs are far from being commonplace. In fact, the number of groups worldwide that possess the expertise, computational resources and *software tools* to work at the forefront of the precision frontier is rather low. Especially the role of computer programs in this context should not be underestimated. The truth is that such codes are of utmost importance to the task of calculating multiloop amplitudes in a straightforward and efficient fashion. Unfortunately, for various reasons many of such powerful codes are not publicly available, which severely impedes the adoption of new and efficient computational techniques among particle phenomenologists. After all, the number of theorists that have time, motivation, diligence and programming skills to implement relevant algorithms single-handedly for the project at hand, is not that high. Most people, therefore, tend to rely on tools that are (i) freely available, (ii) well documented, (iii) easy to use and (iv) are actively supported by their developer teams.

One of such programs is the open-source Mathematica package FeynCalc [2, 3, 4] that has been available to the community since more than three decades. FeynCalc is known as a tool that can be employed in highly nonstandard scenarios that require full control over each calculational step. For example, a rather unique feature of the package that has been added very recently, comprises the ability to automatize manifestly noncovariant calculations [5].

Although one might argue that nonrelativistic quantum field theories is a very small niche, this functionality has been warmly embraced by the effective field theory (EFT) community and helped to obtain interesting new research results [6, 7, 8, 9, 10, 11, 12, 13, 14]

However, the common perception of FeynCalc is that it is useful only for tree-level and one-loop calculations, while everything at two loops and beyond should be handled using other tools. In this conference note we intend to challenge this viewpoint by reporting on selected capabilities of the upcoming FeynCalc 10 [15]. More explicitly, we will discuss the usage of FeynCalc for the tasks of topology identification and manipulations of master integrals obtained after a successful Integration-By-Parts (IBP) [16, 17] reduction. These two steps necessarily arise in almost any multiloop calculation and can be usually handled using Mathematica codes without hitting performance bottlenecks. Furthermore, they have been inspired by the author's multiloop-related research work [18, 19, 20, 21] and thus at least partially bench-tested on real-life problems.

2. Topology identification

Topology identification is commonly understood as the procedure of assigning all loop integrals occurring in the given calculation to a set of integral families. An integral family or topology consists of a list of linearly independent propagators that form a basis. While the problem at hand may easily contain tens of thousands of integrals, the number of topologies they belong to is usually much smaller ranging from dozens to several hundreds.[1] Topology identification is also a necessary step for running IBP reduction using tools[2] such as FIRE [24], KIRA [25, 26] or LiteRed [27, 28].

The process of mapping integrals to topologies can be done on the level of graphs or analytic amplitudes. In the former case each of the given Feynman diagrams or amplitudes is converted to a graph, which transforms the initial problem to the task of finding subgraph isomorphisms. In the latter case one works directly with symbolic propagators and tries to identify sets that are equivalent upon applying suitable loop momentum shifts. Pure graph-based approach is realized in the C++ programs q2e/exp [29, 30], while Mathematica package TopoID [31] implements the analytic mapping procedure. Other codes such as Reduze (C++) [22, 23], feynson (C++) [32], tapir (Python) [33] or pySecDec [34, 35, 36] combine multiple methods in creative ways.

In this report we would like to focus on the task of finding a set of unique integral topologies for the given amplitude. This amplitude can equally represent a single Feynman diagram or a sum thereof. The first step is always to look at all the occurring sets of loop integral denominators and try to detect which of them can be mapped into each other. At this stage we do not need to look at the numerators. This is because for a set of propagators forming a basis, each scalar product involving loop momenta can be always expressed as a linear combination of inverse propagators.

Naively, one might try to search for equivalences between different topologies by applying all possible loop momentum shifts in the hope that a right combination of shifts will produce a match. While this procedure works in principle, the combinatorial growth of complexity effectively prohibits its application to most realistic multiloop problems. A much more efficient algorithm for this special problem was devised by Alexey Pak in [37]. A very detailed and pedagogical summary of Pak's ideas can be found in the doctoral thesis of Jens Hoff [38], which was enormously useful for the presented implementation.

The main idea behind the Pak algorithm is to compare topologies represented in form of the so-called graph or Symanzik polynomials $\mathcal{U}$ and $\mathcal{F}$ (cf. [39] for an extensive review). These two

[1] Of course, the exact number highly depends on the considered process and the number of relevant scales.
[2] Reduze [22, 23] is somewhat special in this respect, as it has built-in topology identification routines.

quantities naturally arise when converting a loop integral into a Feynman parametric integral using the well-known formula (cf. e.g. [40])

$$\left(\frac{e^{\varepsilon\gamma_E}}{i\pi^{d/2}}\right)^L \int \frac{\left(\prod_{i=1}^L d^d k_i\right)}{P_1^{m_1}\ldots P_N^{m_N}} \\ = \frac{(-1)^{N_m}\Gamma(N_m - \frac{Ld}{2})}{\prod_{j=1}^N \Gamma(m_j)} \int_0^\infty \prod_{j=1}^N dx_j x_j^{m_j-1}\,\delta\left(1-\sum_{i=1}^N x_i\right) \frac{\mathcal{U}^{N_m-\frac{(L+1)d}{2}}}{\mathcal{F}^{N_m-\frac{Ld}{2}}} \tag{1}$$

where P_i denote quadratic or eikonal propagators and $N_m = \sum_{i=1}^N m_i$. Notice that this formula is valid only for positive propagator powers i.e. $m_i \geq 0$. Nevertheless, it can be easily extended to negative powers and even tensor structures in the numerators (cf. e.g. doctoral thesis of Stefan Jahn [41] for a nice summary). From here one can start comparing topologies by looking at their *characteristic polynomials* defined as $\mathcal{P} \equiv \mathcal{U} \times \mathcal{F}$. The usage of these polynomials removes ambiguities arising from momentum shifts that often make equivalent topologies seem very different. However, the given characteristic polynomial still cannot be regarded as a unique representation of the corresponding loop integral. This is because $\mathcal{P}$ is not invariant under arbitrary renamings of Feynman parameters (e.g. $x_1 \to x_3$, $x_2 \to x_1$ etc.). Each such renaming yields a new polynomial that differs from the old one, even though both of them describe the same integral. Pak algorithm removes this ambiguity by introducing the notion of *canonical ordering* for the given $\mathcal{P}$. This means that by comparing two canonical polynomials $\mathcal{P}_1$ and $\mathcal{P}_2$ one can unambiguously answer the question whether the corresponding topologies T_1 and T_2 are identical or not.

Let us now come to the implementation of this technology in FeynCalc. In order to avoid cluttering the text we will abstain from including code examples in the body of this report. Those examples can be found in the ancillary file[3] accompanying this work.

First of all, FeynCalc 10 introduces two new symbols representing single loop integrals and loop integral topologies. These are `GLI` and `FCTopology`. For example, `GLI[topo, {1,0,1,1,2}]` is a placeholder for some *generic loop integral* that belongs to the integral family `topo` containing 5 propagators. Our integral contains only 4 lines, with one of the propagators being raised to a quadratic power. `FCTopology`, on the contrary, features a richer structure that consists of 6 arguments as in `FCTopology[topo, {propagators}, {loop momenta}, {external momenta}, {kinematics}, {}]`. The first 5 arguments should be self-explanatory, while the last argument (an empty list) is reserved for additional information such as cuts or symmetries but is currently ignored by the existing functions. The majority of new multiloop-related routines expect their first two arguments to be a list of `GLI`s and a list of the corresponding `FCTopology` objects. In many cases it is also possible to input loop integrals in the traditional `(S)FAD`-notation that uses explicit D-dimensional scalar products (`SPD`s) and propagators (`FAD`s or `SFAD`s[4]). Furthermore, it is always possible to convert an integral in the `GLI`-notation to the `SFAD`-notation using `FCLoopFromGLI`.

We now want to proceed to the characteristic polynomials and the Pak algorithm. Traditionally, many FeynCalc routines that perform complicated manipulations on symbolic expressions (e.g. `TID` or `DiracSimplify`) are actually built on top of numerous auxiliary functions that are equally accessible to the user. This modular approach is one of the reasons why FeynCalc easily integrates into different workflows. Instead of being a black box that forces

[3] `https://arxiv.org/src/2112.14132/anc`
[4] The main difference between `FAD`s and `SFAD`s is that the former can encode only quadratic propagators, while the latter allow for both quadratic and eikonal propagators cf. Sec. 4.3 in [5].

the user to follow specific paths and patterns, FeynCalc is designed to adapt itself to the user's needs. The multiloop extension of the package strictly follows this philosophy.

One of the most fundamental routines related to the handling of multiloop integrals is `FCFeynmanPrepare`. The main task of this function is to compute the $\mathcal{U}$ and $\mathcal{F}$ polynomials of the given integral using the algorithm from the famous `UF.m` program [40] that is currently part of FIESTA [42]. In addition to that, `FCFeynmanPrepare` also returns other important building blocks such as the matrix M with $\mathcal{U} = \det M$ as well as J and Q^μ from $\mathcal{F} = \det M (Q M^{-1} Q - J)$. These quantities are not really needed for the topology identification but can be very useful for other purposes e.g. when constructing Feynman parametric representations.

The characteristic polynomial $\mathcal{P}$ can be obtained from `FCLoopToPakForm`, where we would like to stress that the polynomial returned by this function is already canonically ordered. Of course, it is also possible to determine the canonical ordering of a user-defined polynomial that does not necessarily has to be related to a particular loop integral. The corresponding function is called `FCLoopPakOrder`. It is worth noting that by analyzing $\mathcal{P}$ it is possible to detect scaleless loop integrals that vanish in dimensional regularization. This also concerns cases where one cannot readily recognize the scalelessness by merely looking at the integral. Here we refer to Sec. 2.3 of ref. [38] for a description of the underlying algorithm. This functionality is implemented in FeynCalc in form of the functions `FCLoopPakScalelessQ` (for characteristic polynomials) and `FCLoopScalelessQ` (for loop integrals).

The actual task of performing the topology identification is handled by `FCLoopFindTopology-Mappings`. The function takes a list of `FCTopology` objects as an input and searches for equivalent integral topologies among the elements of this list. In the case of success, it returns a set of mapping rules that contain loop momentum shifts and replacement rules for the `GLI`s. Using the option `PreferredTopologies` one can specify a set of target topologies that should be primarily considered during the construction of the mapping rules. Notice that in its current form `FCLoopFindTopologyMappings` can only find equivalent topologies with the same number of propagators. However, in real life calculations one often encounters topologies that do not have enough propagators to form a basis. Such incomplete topologies often fit into larger complete topologies, sometimes also denoted as *supertopologies*. This issue can be addressed in the current development snapshot of FeynCalc as follows. Provided that one has some candidate supertopologies, one would first employ `FCLoopFindSubtopologies` to identify all nonvanishing subtopologies of the given supertopology. Then, one would add the list of subtopologies to the list of preferred topologies when running `FCLoopFindTopologyMappings`. Finally, using the function `FCLoopCreateRuleGLIToGLI` one can generate replacement rules that would eliminate the subtopologies in favor of the corresponding supertopology.

To apply this machinery to amplitudes FeynCalc also features two dedicated functions called `FCLoopFindTopologies` and `FCLoopApplyTopologyMappings`. The former identifies all distinct topologies present in the given amplitude, while the latter applies the mappings uncovered by `FCLoopFindTopologyMappings` to this amplitude. However, for practical reasons we would recommend against the idea of doing multiloop calculations entirely in Mathematica. The number of intermediate expressions in such computations can easily go into hundreds of thousands or even millions of terms even for a single diagram. Unlike FORM [43], Mathematica is simply not optimized to handle that level of complexity and any attempts to ignore this fact would most likely result in frustration as well as wasted time and efforts. The most efficient way to approach such calculations is to use QGRAF [44] and FORM. Having extracted the list of distinct topologies from FORM one can readily employ FeynCalc to carry out the topology identification and then export the obtained mappings back into FORM. This is essentially how we envisage the usage of the new multiloop capabilities of FeynCalc in loop calculations. Of course, one might also think of situations where it should be admissible to do the full calculation within a single Mathematica execution. One of such scenarios would be asymptotic [45] expansions of single loop integrals in conjunction with ASY [46] and FEYNHELPERS [47]. Here we would like to refer to the upcoming version of the FEYNHELPERS [48] extension that would feature dedicated routines for automatizing such expansions.

3. Master integrals

After having completed all relevant IBP reductions, one usually starts to analyze the resulting master integrals. One of the first things to do is to create a list of master integrals from all integral families and check whether all of them are distinct. Just as in the case of equivalent topologies, equivalent integrals often cannot be recognized by eye. Here FEYNCALC offers a function called `FCLoopFindIntegralMappings` that works similarly to `FindRules` in FIRE. The `PreferredIntegrals` option allows to map (if possible) the given list of integrals to a set of some predefined master integrals.

When presenting results from new multiloop calculations to the community, it is customary to visualize the relevant master integrals in form of graphs with styled edges. For example, edges representing massless propagators are often drawn as dashed lines, while a solid line stands for a massive denominator and a dot or a cross means that the corresponding propagator appears raised to an integer power (usually squared). Owing to the fact that the reconstruction of the graph representation from the propagator representation is not entirely trivial, it may take some time and effort to generate such figures for a new set of integrals. This task can be partially automatized using the functionality available in AZURITE [49], PLANARITYTEST [50], LITERED [27, 28] and now also FEYNCALC. Our implementation consists of two functions called `FCLoopIntegralToGraph` (for obtaining the graph) and `FCLoopGraphPlot` (for plotting the graph). Notice that the output of `FCLoopIntegralToGraph` can be, in principle, visualized using other tools e.g. GRAPHVIZ [51].

When it comes to the analytic evaluation of master integrals, currently the methods of differential equations [52, 53, 54, 55, 56, 57] and Mellin-Barnes [58, 59, 60, 61] seem to be the most popular techniques to attack this problem. Yet in this report we would like to advocate another method that often turns out to be very successful when applied to integrals with only few scales. What is meant is the direct analytic integration starting from the Feynman parameter representation of the integral. In this context we would like to mention the MAPLE package HYPERINT [62] that currently constitutes the most advanced publicly available tool for automatizing such integrations. Multiscale integrals often might require a suitable variable transformation to eliminate square roots that hinder the integration sequence. To this end we found the package RATIONALIZEROOTS [63] (available both for MAPLE and MATHEMATICA) to be very handy. Let us also remark that the results obtained by HYPERINT in terms of complicated Goncharov Polylogarithms (GPLs) [64] often can be further simplified using the packages HYPERLOGPROCEDURES [65] and POLYLOGTOOLS [66].

FEYNCALC function `FCFeynmanParametrize` can obtain Feynman parametric representation for a wide range of loop integrals with quadratic or eikonal propagators. Possible types of supported integrals range from expressions with unit numerators over integrals with scalar products to tensor integrals with open indices. Furthermore, `FCFeynmanParametrize` can handle not only Minkowskian but also Cartesian and Euclidean integrals, which covers almost every type of Feynman integral one might encounter in various Standard Model and EFT calculations. In addition to that, the function `FCFeynmanParameterJoin` allows the user to apply Feynman's formula for joining denominators sequentially. This means that instead of joining all propagators at once one may also split them into smaller subsets and join the propagators in those subsets first. The joining of the resulting propagators stemming from each subset is then done at the very last step. This procedure generates several sets of Feynman parameter variables that can be often integrated in a simpler way owing to the additional freedom when exploiting the Cheng-

Wu [67] theorem. In our experience, choosing a smart way to join the propagators sequentially often can enable an analytic integration that would otherwise seem hopeless without introducing a proper sequence of Feynman variable transformations.

4. Summary

We presented a set of new FeynCalc functions that improve the usefulness of the package in multiloop calculations. The ideas and algorithms behind the routines are readily available in the literature [37, 38, 41] and we gratefully acknowledge that the new way of dealing with multiloop integrals in the package draws many inspirations from such well-established tools as FIRE, FIESTA, LiteRed, TopoID and pySecDec. Nevertheless, we believe that our implementation of the relevant methods should be very useful not only to the existing FeynCalc users but also to the whole particle physics community. Although FeynCalc 10 has not yet been officially released, all the functions described in this report are publicly available through the so-called *development version* of the package[5]. We would also like to draw the attention of the reader to the new documentation system[6] that already includes descriptions of all new functions and features helpful examples.

In general, it is very gratifying to see the progress in the functionality of the package as compared to the situation some years ago when FeynCalc 9 was announced during the ACAT 2017 [68]. It would be far from correct, however, to claim that the package is now feature complete. Two main directions we would like to see FeynCalc go are helicity amplitude methods and a FORM-based library for computationally heavy tasks. Only time will tell to which extent those can be realized in future versions of the program.

Acknowledgements

The research of V. S. was supported by the DFG under grant 396021762 – TRR 257 "Particle Physics Phenomenology after the Higgs Discovery.". The author would like to thank Matthias Steinhauser and Marvin Gerlach for many useful discussions on different aspects of multiloop calculations and to Erik Panzer for his friendly and patient support on the usage of HyperInt.

References

[1] Tancredi L 2021 *European Physical Society Conference on High Energy Physics 2021* (*Preprint* `2111.00205`)
[2] Mertig R, Bohm M and Denner A 1991 *Comput. Phys. Commun.* **64** 345–359
[3] Shtabovenko V, Mertig R and Orellana F 2016 *Comput. Phys. Commun.* **207** 432–444 (*Preprint* `1601.01167`)
[4] Shtabovenko V, Mertig R and Orellana F 2020 *Comput. Phys. Commun.* **256** 107478 (*Preprint* `2001.04407`)
[5] Brambilla N, Chung H S, Shtabovenko V and Vairo A 2020 *JHEP* **11** 130 (*Preprint* `2006.15451`)
[6] Brambilla N, Chen W, Jia Y, Shtabovenko V and Vairo A 2018 *Phys. Rev. D* **97** 096001 [Erratum: Phys.Rev.D 101, 039903 (2020)] (*Preprint* `1712.06165`)
[7] Brambilla N, Chung H S, Lai W K, Shtabovenko V and Vairo A 2019 *Phys. Rev. D* **100** 054038 (*Preprint* `1907.06473`)
[8] Assi B and Kniehl B A 2020 (*Preprint* `2011.06437`)
[9] Assi B and Kniehl B A 2020 (*Preprint* `2011.06447`)
[10] Urban K 2021 *Large Electroweak Corrections to Heavy WIMP Dark Matter Annihilation and Resummation* Ph.D. thesis Munich, Tech. U.
[11] Biondini S and Shtabovenko V 2021 *JHEP* **08** 114 (*Preprint* `2106.06472`)
[12] Chung H S 2021 *JHEP* **09** 195 (*Preprint* `2106.15514`)
[13] Frugiuele C and Peset C 2021 (*Preprint* `2107.13512`)
[14] Biondini S and Shtabovenko V 2021 (*Preprint* `2112.10145`)
[15] Shtabovenko V, Mertig R and Orellana F *in preparation*
[16] Chetyrkin K G and Tkachov F V 1981 *Nucl. Phys. B* **192** 159–204

[5] `https://github.com/FeynCalc/feyncalc/wiki/Installation#dev_automatic_installation`
[6] `https://feyncalc.github.io/referenceDev`

[17] Tkachov F V 1981 *Phys. Lett. B* **100** 65–68
[18] Dixon L J, Luo M X, Shtabovenko V, Yang T Z and Zhu H X 2018 *Phys. Rev. Lett.* **120** 102001 (*Preprint* 1801.03219)
[19] Luo M X, Shtabovenko V, Yang T Z and Zhu H X 2019 *JHEP* **06** 037 (*Preprint* 1903.07277)
[20] Gao J, Shtabovenko V and Yang T Z 2021 *JHEP* **02** 210 (*Preprint* 2012.14188)
[21] Gerlach M, Nierste U, Shtabovenko V and Steinhauser M 2021 *JHEP* **07** 043 (*Preprint* 2106.05979)
[22] Studerus C 2010 *Comput. Phys. Commun.* **181** 1293–1300 (*Preprint* 0912.2546)
[23] von Manteuffel A and Studerus C 2012 (*Preprint* 1201.4330)
[24] Smirnov A V and Chuharev F S 2020 *Comput. Phys. Commun.* **247Â** 106877 (*Preprint* 1901.07808)
[25] Maierhöfer P, Usovitsch J and Uwer P 2018 *Comput. Phys. Commun.* **230** 99–112 (*Preprint* 1705.05610)
[26] Klappert J, Lange F, Maierhöfer P and Usovitsch J 2021 *Comput. Phys. Commun.* **266** 108024 (*Preprint* 2008.06494)
[27] Lee R N 2012 (*Preprint* 1212.2685)
[28] Lee R N 2014 *J. Phys. Conf. Ser.* **523** 012059 (*Preprint* 1310.1145)
[29] Harlander R, Seidensticker T and Steinhauser M 1998 *Phys. Lett. B* **426** 125–132 (*Preprint* hep-ph/9712228)
[30] Seidensticker T 1999 *6th International Workshop on New Computing Techniques in Physics Research: Software Engineering, Artificial Intelligence Neural Nets, Genetic Algorithms, Symbolic Algebra, Automatic Calculation* (*Preprint* hep-ph/9905298)
[31] Hoff J 2016 *J. Phys. Conf. Ser.* **762** 012061 (*Preprint* 1607.04465)
[32] *https://github.com/magv/feynson/*
[33] Gerlach M, Herren F and Lang M 2022 (*Preprint* 2201.05618)
[34] Borowka S, Heinrich G, Jahn S, Jones S P, Kerner M, Schlenk J and Zirke T 2018 *Comput. Phys. Commun.* **222** 313–326 (*Preprint* 1703.09692)
[35] Borowka S, Heinrich G, Jahn S, Jones S P, Kerner M and Schlenk J 2019 *Comput. Phys. Commun.* **240** 120–137 (*Preprint* 1811.11720)
[36] Heinrich G, Jahn S, Jones S P, Kerner M, Langer F, Magerya V, Põldaru A, Schlenk J and Villa E 2021 (*Preprint* 2108.10807)
[37] Pak A 2012 *J. Phys. Conf. Ser.* **368** 012049 (*Preprint* 1111.0868)
[38] Hoff J S 2015 *Methods for multiloop calculations and Higgs boson production at the LHC* Ph.D. thesis KIT, Karlsruhe
[39] Bogner C and Weinzierl S 2010 *Int. J. Mod. Phys. A* **25** 2585–2618 (*Preprint* 1002.3458)
[40] Smirnov V A 2012 *Analytic tools for Feynman integrals* vol 250
[41] Jahn S 2020 *Automation of Multi-Loop Amplitude Calculations* Ph.D. thesis Munich, Tech. U.
[42] Smirnov A V, Shapurov N D and Vysotsky L I 2021 (*Preprint* 2110.11660)
[43] Kuipers J, Ueda T, Vermaseren J A M and Vollinga J 2013 *Comput. Phys. Commun.* **184** 1453–1467 (*Preprint* 1203.6543)
[44] Nogueira P 1993 *J. Comput. Phys.* **105** 279–289
[45] Beneke M and Smirnov V A 1998 *Nucl. Phys. B* **522** 321–344 (*Preprint* hep-ph/9711391)
[46] Jantzen B, Smirnov A V and Smirnov V A 2012 *Eur. Phys. J. C* **72** 2139 (*Preprint* 1206.0546)
[47] Shtabovenko V 2017 *Comput. Phys. Commun.* **218** 48–65 (*Preprint* 1611.06793)
[48] Shtabovenko V *in preparation*
[49] Georgoudis A, Larsen K J and Zhang Y 2017 *Comput. Phys. Commun.* **221** 203–215 (*Preprint* 1612.04252)
[50] Bielas K, Dubovyk I, Gluza J and Riemann T 2013 *Acta Phys. Polon. B* **44** 2249–2255 (*Preprint* 1312.5603)
[51] *https://www.graphviz.org/*
[52] Kotikov A V 1991 *Phys. Lett. B* **267** 123–127 [Erratum: Phys.Lett.B 295, 409–409 (1992)]
[53] Kotikov A V 1991 *Phys. Lett. B* **254** 158–164
[54] Kotikov A V 1991 *Phys. Lett. B* **259** 314–322
[55] Bern Z, Dixon L J and Kosower D A 1994 *Nucl. Phys. B* **412** 751–816 (*Preprint* hep-ph/9306240)
[56] Remiddi E 1997 *Nuovo Cim. A* **110** 1435–1452 (*Preprint* hep-th/9711188)
[57] Gehrmann T and Remiddi E 2000 *Nucl. Phys. B* **580** 485–518 (*Preprint* hep-ph/9912329)
[58] Smirnov V A 1999 *Phys. Lett. B* **460** 397–404 (*Preprint* hep-ph/9905323)
[59] Tausk J B 1999 *Phys. Lett. B* **469** 225–234 (*Preprint* hep-ph/9909506)
[60] Anastasiou C and Daleo A 2006 *JHEP* **10** 031 (*Preprint* hep-ph/0511176)
[61] Czakon M 2006 *Comput. Phys. Commun.* **175** 559–571 (*Preprint* hep-ph/0511200)
[62] Panzer E 2015 *Comput. Phys. Commun.* **188** 148–166 (*Preprint* 1403.3385)
[63] Besier M, Wasser P and Weinzierl S 2020 *Comput. Phys. Commun.* **253** 107197 (*Preprint* 1910.13251)
[64] Goncharov A B 1998 *Math. Res. Lett.* **5** 497–516 (*Preprint* 1105.2076)
[65] Schnetz O *https://www.math.fau.de/person/oliver-schnetz*
[66] Duhr C and Dulat F 2019 *JHEP* **08** 135 (*Preprint* 1904.07279)

[67] Cheng H and Wu T T 1987 *EXPANDING PROTONS: SCATTERING AT HIGH-ENERGIES*
[68] Shtabovenko V 2016 *J. Phys. Conf. Ser.* **762** 012064 (*Preprint* `1604.06709`)