**Package-X 2.0: A Mathematica package for the analytic calculation of one-loop integrals**

Hiren H. Patel

1Particle and Astro-Particle Physics Division
Max-Planck Institut fuer Kernphysik (MPIK)
Saupfercheckweg 1, 69117 Heidelberg, Germany

This article summarizes new features and enhancements of the first major update of Package-X. Package-X 2.0 can now generate analytic expressions for arbitrarily high rank dimensionally regulated tensor integrals with up to four distinct propagators, each with arbitrary integer weight, near an arbitrary even number of spacetime dimensions, giving UV divergent, IR divergent, and finite parts at (almost) any real-valued kinematic point. Additionally, it can generate multivariable Taylor series expansions of these integrals around any non-singular kinematic point to arbitrary order. All special functions and abbreviations output by Package-X 2.0 supports Mathematica’s arbitrary precision evaluation capabilities to deal with issues of numerical stability. Finally, tensor algebraic routines of Package-X have been polished and extended to support open fermion chains both on and off shell. The documentation (equivalent to over 100 printed pages) is accessed through Mathematica’s Wolfram Documentation Center and contains information on all Package-X symbols, with over 300 basic usage examples, 3 project-scale tutorials, and instructions on linking to FeynCalc and LoopTools.

**Program summary**

*Program title:* Package-X  
*Program obtainable from:* CPC Program Library, Queen’s University, Belfast, N. Ireland, or [http://packagex.hepforge.org](http://packagex.hepforge.org)  
*Licensing provisions:* Standard CPC license, [http://cpc.cs.qub.ac.uk/licence/licence.html](http://cpc.cs.qub.ac.uk/licence/licence.html)  
*Programming language:* Mathematica (Wolfram Language)  
*Operating systems:* Windows, Mac OS X, Linux (or any system supporting Mathematica 8.0 or higher)  
*Journal reference of previous version:* H. H. Patel, Comput. Phys. Commun 197, 276 (2015)  
*Does the new version supersede the previous version?:* Yes  
*Summary of revisions:* Extension to four point one-loop integrals with higher powers of denominator factors, separate extraction of UV and IR divergent parts, testing for power IR divergences, construction of Taylor series expansions of one-loop integrals, numerical evaluation with arbitrary precision arithmetic, manipulation of fermion chains, improved tensor algebraic routines, and much expanded documentation.  
*RAM required for execution:* 10 MB, depending on size of computation  
*Vectorised/parallelized?:* No  
*Nature of problem:* Analytic calculation of one-loop integrals in relativistic quantum field theory.  
*Solution method:* Passarino-Veltman reduction formula, Denner-Dittmaier reduction formulae, and additional algorithms described in the manuscript.  
*Restrictions:* One-loop integrals are limited to those involving no more than four denominator factors.  
*Running Time:* 5ms to 10s for integrals typically occurring in practical computations; longer for higher rank tensor integrals.
I. INTRODUCTION

Package-X is a Mathematica package with the principal purpose of generating analytic results for dimensionally regulated one-loop rank-$P$ tensor integrals of the form

\[ T^{\mu_1 \cdots \mu_P}_N = \left( \frac{i e^{-\gamma_E}}{4\pi^d/2} \right)^{-1} \mu^2 \int \frac{d^d k}{(2\pi)^d} \left\{ k^{\mu_1} \cdots k^{\mu_P} \right. \]
\[ \times \left[ (k+p_0)^2 - m_0^2 + i\varepsilon \right]^{-\nu_0} \left[ (k+p_1)^2 - m_1^2 + i\varepsilon \right]^{-\nu_1} \]
\[ \cdots \left[ (k+p_{N-1})^2 - m_{N-1}^2 + i\varepsilon \right]^{-\nu_{N-1}} \], \tag{1} \]

with up to $N = 4$ denominator factors, each with integer weights $\nu_i$, for arbitrary configurations of external momenta $p_i$ and real-valued internal masses $m_i$.

While many packages are publicly available to numerically evaluate one loop integrals \cite{1,2} with the aim of automatizing the calculation of cross sections with full kinematic dependence, none exist to the author’s knowledge that provide complete analytic expressions. Package-X serves to fill this gap with the aim of obtaining results for quantum field theory calculations where compact analytic expressions exist (e.g., pole masses, electroweak oblique parameters, particle moments, decay rates, cross sections at threshold, counterterms, Wilson coefficients, etc). The application files along with an introductory tutorial are hosted at the Hepforge \cite{8} project page http://packagex.hepforge.org.

The original release of Package-X in 2015 suffered from several limitations, which are listed in Section IX of the accompanying publication \cite{9}. In addition to addressing all these limitations, Package-X 2.0 features many new features and enhancements described below. Familiarity with the use of the package is assumed. Furthermore, since the underlying algorithms and program structure is already detailed at length in the original publication, only a brief description of updates to Package-X is provided in the sections to follow. The interested reader is encouraged to consult the original Package-X publication and the references in the text below for details.

The most prominent features of version 2.0 that are described below are the ability to compute one-loop integrals with four distinct denominator factors (Sections \cite{II,III} and \cite{IV}), construct Taylor series expansions of one-loop integrals (Section \cite{V}), and compute integrals with open fermion lines (Section \cite{VI}). Finally, a listing of changes and new functions/symbols are collected in the Appendices.

II. UPDATES TO LoopIntegrate

The evaluation of a one-loop integral is initiated with LoopIntegrate and performs its covariant tensor decomposition in terms of Passarino-Veltman coefficient functions. As of version 2.0, LoopIntegrate supports integrals with up to four distinct propagator factors, each with arbitrary integer powers $\nu_1, \ldots, \nu_{N-1}$. The tensor covariant decomposition follows the basic property of Lorentz covariance of dimensionally regulated loop integrals \cite{10}, and generates an expression in terms of coefficient functions FVA, FVB, PVC, PVD as described in section III of \cite{9}.

In the original version of Package-X, the syntax of LoopIntegrate was such that the user had to put the integral in a form such that $p_0 = 0$ in \cite{II}. This meant that the user either had to strategically route the momenta through the one-loop diagram so that at least one denominator factor had no external momenta flowing thorough it, or had to shift the integration variable before using LoopIntegrate to evaluate the integral. As of version 2.0, LoopIntegrate accepts a new syntax accommodating nonzero $p_0$ which means that any momentum routing is possible.

For integrals involving denominator factors with linearly dependent momenta, it is standard practice to perform a partial fraction expansion of the denominator factors before making a covariant decomposition as it leads to a sum of integrals with fewer number of denominator factors. As of version 2.0, LoopIntegrate has an option Apart that toggles whether to expand linearly dependent denominator factors into partial fractions. The algorithms are based on \cite{II}, but are specifically tailored to efficiently process one-loop integrals.

Additionally, if the numerator of the loop integral contains dot products involving the loop momentum ($k^2$ or $k.p_i$), it is profitable to write them as differences of propagator factors so that they may be cancelled against the denominator. This procedure simultaneously lowers the rank of tensor integrals and reduces the number of denominator factors. As of version 2.0, LoopIntegrate has an optionCancel that controls whether to expand and cancel these factors before making a covariant decomposition. This, along with expanding in partial fractions, generally leads to substantially increased performance and quality of the output of LoopRefine.

Finally, in version 2.0, LoopIntegrate can now perform the covariant decomposition for integrands involving DiracMatrix, FermionLine, and FermionLineProduct objects, enabling it to process integrands with open fermion lines. This capability is described in Section VI below.

III. UPDATES TO LoopRefine

After the covariant decomposition of a loop integral is carried out with LoopIntegrate and kinematic conditions are supplied, the final step is to apply LoopRefine, which replaces the coefficient functions with explicit analytic expressions.

In version 2.0, LoopRefine can convert four-point coef-
efficient functions PVD to elementary functions. In the most
general case, the standard Passarino-Veltman reduction
is used. To handle cases with vanishing Grammian
determinant, the algorithms described in are used,
and are direct generalizations of Cases 1, 3, 5 and 6
appearing in section IV C of to four propagator factors.
These algorithms are not applicable if the determinant
of the modified Cayley matrix vanishes. For further (but
still incomplete) coverage, two new reduction formulae,
valid when certain elements of the adjugate Grammian
matrix vanish, were derived by explicitly integrating over
Feynman parameters. When appropriate, Package-X
applies these formulae. However, further research is needed
to provide complete kinematic coverage.

In the original release of Package-X, only integrals with
unit weight were supported ($\nu_0 = \ldots = \nu_{N-1} = 1$). As
of version 2.0, LoopRefine can convert integrals of ar-
bitrary integer weight to analytic expressions. Weighted
loop integrals can arise when Feynman diagrams with
massless gauge bosons in general covariant gauge are con-
sidered, or when the small momentum approximation is
applied at the level of the integrand. They are also in-
dispensable for the construction of series expansions as
described in Section below. Conventionally, integra-
tion by parts methods together with recurrence re-
lations in $d$ dimensions are used to reduce these
functions. However, Package-X uses a slightly different
method derived as follows: by comparing Feynman pa-
rameter integral representations, the weighted coefficient
functions can be related to linear combinations of un-
weighted coefficient functions with fewer number of 00
index pairs. Then one of the existing reduction formulae
can be used to convert them to analytic expressions.
However, weighted scalar functions in this scheme are re-
lated to coefficient functions with formally negative num-
er of 00 pairs. But since their Feynman parameter in-
tegral representations imply that they are analytic func-
tions of “number of 00 pairs”, a reduction based on Cay-
ley determinants presented in Section 5.3 of is used
to raise the number of 00 index pairs, thereby making it
possible to convert them to known basis functions.

Similar to weighted coefficient functions, coefficient
functions arising from the decomposition of integrals near
an even number of spacetime dimensions other than $4$ can
also be related to those defined near 4 with a different
number of 00 index pairs by equating Feynman param-
eter integral representations. Therefore, it is possible to
convert these functions to analytic expressions by using
the same strategy.

In addition to the aforementioned major changes,
LoopRefine has also received a number of minor fea-
tures. Among them is the ability for LoopRefine to test
for the presence of power infrared divergences in loop in-
tegrals in some cases. These integrals have the feature
that their Feynman parameter integrals do not converge
for genuinely small dimensional regulator parameter $\epsilon$.
Their presence is detected in Package-X by retaining, in
all basis functions that feature the power divergence, the
$+i\epsilon$ term as a finite quantity which acts like a mass regu-
lator. Then, upon converting its entire input to analytic
expressions, LoopRefine checks whether the $+i\epsilon \to 0$
limit in the total is well-behaved. This accommodates
the possibility that the power infrared divergence cancels
among various terms in the input expression. Furthermore,
LoopRefine takes an additional option Analytic, which
when set to True, generates a result with the pa-
rameter $\epsilon$ analytically continued to large and negative
values, rendering power infrared divergences formally con-
vergent.

Finally LoopRefine can restrict its computation to
just the infrared or ultraviolet divergent parts by ap-
propriately setting the option Part to UVDivergent or
IRDivergent, respectively. In addition to providing a
quick way to obtain the $1/\epsilon$ pole parts of loop integrals,
this new feature also enables one to identify their origin
as either ultraviolet or infrared. The calculation of the
ultraviolet divergent part is fast owing to the existence
of an iterative formula applicable to an arbitrary coefficient
function . The calculation of the IR divergent part is
sped up by setting to zero all coefficient functions with
at least one pair of 00 indices since they are known to be
infrared finite.

IV. THE SCALAR $D_0$ FUNCTION: ANALYTIC
EXPRESSIONS AND NUMERICAL
IMPLEMENTATION

The reduction algorithm for converting coefficient $D$-
functions, PVD, in the case of non-vanishing Grammian
determinant ends with the UV-finite scalar function $D_0$.
To complete the computation of the one-loop integral,
LoopRefine replaces this scalar function.

In order for LoopRefine to faithfully display the $1/\epsilon$
poles in the final output, all IR divergent cases are substi-
tuted. Expressions with massive internal lines are drawn
from and those with massless internal lines are adapted from . Care has been taken to ensure
that all analytic expressions are consistent with the $+i\epsilon$
prescription so that their numerical evaluation yields the
correct sign for their imaginary parts.

The situation for the IR finite four-point functions is
unlike the case of the three- and lower-point scalar func-
tions in that no reasonably compact analytic expressions
are known unless the Grammian determinant vanishes.
For these cases, LoopRefine simply outputs ScalarD0,
with the function itself implemented numerically. The
principle behind the numerical implementation of the
scalar four-point function follows that of the three-point
function outlined in Section V of . In the region of
positive modified Cayley determinant (which covers the
physical region), the imaginary part is obtained by ap-
plying Cutkosky’s rule, and requires the evaluation of a
single logarithm for each channel above normal thresh-
hold. The real part is based on a representation in terms of sixteen numerically evaluated dilogarithm func-
functions. Outside this region, the separation of the real and imaginary parts is not used since the application of Cutkosky’s rule is more complicated. For rapid numerical evaluation, the code for ScalarDO is compiled to the Wolfram Virtual Machine.

Additionally, as of version 2.0, every special numerical function available in Package-X (including ScalarDO) is written in terms of native Wolfram Kernel functions so that the user may take advantage of Mathematica’s arbitrary precision evaluation capabilities. This new feature of Package-X allows one to obtain numerically stable results for kinematic configurations that would otherwise lead to severe loss of precision.

If an analytic expression is desired, ExpandDO can be applied to ScalarDO, which replaces it with the corresponding analytic formula from Package-X’s library. The simpler expressions are adapted from [23] and [24], and the most general and complicated cases are derived from [18]. In order to minimize the size of the output, the analytic expressions are given compactly as RootSum over four Denner-Beenakker continued dilogarithms \( \text{Li}_2(x, y) \) defined originally in [25] and implemented in Package-X as ContinuedDiLog.

V. LoopRefineSeries: Taylor Series Expansions of Loop Integrals

Calculations within the standard model and its extensions usually involves integrals with widely separated scales. In these cases, an exact analytic representation of these integrals are too verbose to be of any value, and their numeric evaluation usually suffers from loss of precision due to large numerical cancellations. As a result, one usually desires an approximate expression obtained from the first few terms of its series expansion.

Package-X 2.0 provides a way to obtain Taylor series expansions of loop integrals with a new routine LoopRefineSeries, which is to be used in place of LoopRefine. Internally, the algorithm for constructing a series is as follows:

1. Differentiate the Passarino-Veltman coefficient functions as needed to generate the necessary terms in the Taylor expansion.

2. Express the differentiated coefficient functions in terms of undifferentiated ones with fewer number of 00 index pairs.

3. Apply the basic reduction formula on the functions to convert the expression to analytic form.

In step 2, the conventional practice is to relate the differentiated coefficient functions to those defined in a lower number of spacetime dimensions [15]. But as described in Section [15], it is more convenient to use the standard coefficient functions with fewer pairs of 00 indices, and apply the reduction formula based on Cayley determinants in step 3 for negative pairs. This method of constructing series expansions easily generalizes to multiple series expansions, and to arbitrarily high order, constrained only by memory and computation time.

A major limitation of this method is that it is unable to construct expansions around Landau singularities because the necessary derivatives at those points usually do not exist. Therefore, it is not possible to construct small mass, threshold, or other asymptotic expansions with LoopRefineSeries alone. While one can still circumvent this problem in limited circumstances by applying Mathematica’s Series function on the output of LoopRefine, a fully automated routine to construct such expansions is absent.

VI. Support for Open Fermion Lines

Although the primary task of Package-X is to assist in the analytic calculation of one-loop integrals of the form given in (1), the kinds of integrals one encounters when calculating in physical theories are those involving fermions. The original release of Package-X included the function Spur to calculate traces of Dirac matrices, and satisfactorily handled Feynman integrals involving a closed fermion loop. However, no direct support for evaluating integrals with open fermion lines was provided. In order to evaluate these integrals, the integrals had to be projected onto form factors one at a time. For calculations involving several Feynman integrals, projecting onto separate form factors for each one is laborious. In order to alleviate this problem, version 2.0 now provides direct support for open fermion lines.

The object DiracMatrix represents a product of Dirac matrices, arising from an off shell fermion line in a Feynman diagram. Additionally, two new algebraic objects are introduced in version 2.0: FermionLine represents a product of Dirac matrices sandwiched on either side by on shell spinors, and FermionLineProduct represents the direct product of FermionLine objects. These new objects can be used for integrals involving one or more open fermion lines in a given diagram. They are algebraically manipulated by LoopIntegrate and FermionLineExpand to bring them to canonical form by following a series of steps:

1. Expand the product of Dirac matrices by distributing multiplication over addition.

2. Relate products of gamma matrices with repeated Lorentz indices (such as \( \gamma_\mu \gamma^\nu \gamma^\rho \gamma^\sigma \)) to products with fewer gamma matrices.

3. Apply the Dirac algebra bringing the \( \slashed{\gamma} \) to either end of the fermion line in order to apply the Dirac equation.

4. Apply the Sirlin identities [26] to doubly-contracted direct products of Dirac matrices (FermionLineProduct only).
5. Resolve remaining products of Dirac matrices into SVTAP basis \[27\].

6. Apply the Gordon identities \[28\] to express the vector and axial-vector convective transition currents in terms of Dirac/Pauli and Anapole/EDM transition currents.

The on-shell relations in steps 3 and 5 are not used for manipulating DiracMatrix objects since they are not applicable. Additionally, when manipulating FermionLineProduct objects, the entire algorithm is repeated until the identities can be no longer applied. Note that since \(\gamma_5\) is implemented naively in dimensional regularization, any algebraic manipulations involving \(\gamma_5\) is valid in exactly 4 dimensions only. Likewise, the identities used in steps 4 and 5 are valid in exactly 4 dimensions. Therefore, care should be taken to ensure that the integrals are either manifestly finite, or have been properly regulated to ensure correctness of the finite part of the result.

VII. EXPANSION OF THE DOCUMENTATION

Included with the package software is a set of documentation files accessible from within Mathematica. The documentation contains information for all front end functions and available options in Package-X. In version 2.0, the documentation is expanded to include more details and usage examples for greater clarity. Additionally, three project-scale tutorials are included illustrating how Package-X may be used in research, briefly summarized below:

- **Ward Identities and \(\gamma_5\) in Dimensional Regularization** – In Package-X, \(\gamma_5\) is naively defined to anti-commute with all other gamma matrices, and may lead to incorrect results for logarithmically divergent integrals. This tutorial illustrates, with the \(Z^*\gamma\gamma\) Green function as an example, how chiral Ward identities can be enforced using Alder’s method \[29, 30\].

- **Extracting Form Factors from the \(\mu \to e\gamma\) amplitude** – This tutorial explains the use of Projector in detail, while also explaining how Package-X can be used to verify the decoupling theorem, gauge invariance, and Ward identities.

- **Scattering of Light by Light** – This tutorial explains how to compute the light-by-light scattering matrix element at leading order, and illustrates the capabilities and limitations of covariant methods used in Package-X.

The documentation also contains instructions on linking Package-X to two other publicly available packages: FeynCalc \[31, 32\] (through FeynHelpers \[33\]) and to LoopTools \[2\].

VIII. SUMMARY AND AVENUES FOR FURTHER DEVELOPMENT

In this paper, new features of Package-X in the most recent major release is described. Version 2.0 significantly expands the scope of the software by supporting four point integrals, Taylor expansions, and open fermion lines. However, there are still many limitations of Package-X which continue to guide its development summarized below:

1. Currently, Package-X is unable to automatically construct asymptotic expansions of loop integrals around Landau singularities. Since such expansions are very relevant in quantum field theory calculations (e.g. for mass regularization, threshold expansions, eikonal expansions), an update including this feature is desirable.

2. Additionally, the reduction algorithms for the four-point functions do not completely cover the case when the Cayley determinant vanishes. These cases become physically relevant when the Grammian determinant also vanishes, since they can correspond to IR divergent scattering amplitudes at physical threshold. It would be ideal to provide complete kinematic coverage for these singular cases as well.

3. As Package-X is applied to larger problems, manual input of all integrals required for a specific calculation becomes increasingly tedious and prone to input errors. It would be desirable to have a way, e.g. by linking to the FeynArts package \[34\], to automatically generate all the needed integrals in Package-X automatically.

4. For large problems, it may not be possible to simplify the integrals to obtain compact analytic expressions. Although not its primary purpose, it would be convenient to be able to directly evaluate the coefficient functions numerically, without needing their analytic expressions. While linking to LoopTools \[2\] is described in the documentation, linking to more modern packages like Collier \[6\] is desirable.

5. Package-X is limited to the calculation of Lorentz covariant Feynman integrals. However, non-covariant integrals are also frequently encountered, for example, when one is working in Coulomb or Axial gauges, or within effective field theories like NRQED, HQET and SCET. The ability to automatically obtain analytic results for these cases would be convenient.

ACKNOWLEDGMENTS

I would like to thank the numerous testers who participated during the alpha and beta phases of devel-
Appendix A: Convention and package structure changes between versions 1.0 and 2.0

- `pvA`, `pvB`, `pvC`, `pvC0` and `pvC0IR6` are now named `PVA`, `PVB`, `PVC`, `ScalarC0` and `ScalarC0IR6`, respectively, consistent with `Mathematica`’s naming convention of capitalizing the first letter of pre-defined symbols. Dimensional regularization ’t Hooft scale $\mu R$ is renamed to $\mu$ ([$\mu$Micro]).

- `LoopIntegrate` now normalizes its integration measure so that $\epsilon^{n-v} \geq 1$ is factored out instead of $\tau_n = \Gamma(1-\epsilon)\Gamma(1+n) \epsilon^{n-v}$. This was done to prepare `Package-X` for computing two loop integrals in the future. With respect to version 1.0, this change in version 2.0 only modifies the output of `LoopRefine` exhibiting 1/$\epsilon^2$ poles, arising from overlapping soft and collinear IR divergences, by an amount proportional to $-\pi^2/12$ to the finite part.

- The order of arguments of Passarino-Veltman $C$ functions have been changed to match that of other popular packages such as `FEYNCALC`, `LOOPTOOLS`, `COLLIER`, etc., including other authors in the literature. The relations are

\[
PVC[r, n1, n2, s1, s12, s2, m0, m1, m2] = \quad \text{pvC}[r, n1, n2, s1, s12, s2, m0, m1, m2]
\]

and

\[
ScalarC0[s1, s12, s2, m0, m1, m2] = \quad \text{pvC0}[s1, s12, s2, m0, m1, m2]
\]

- Contexts `X'IndexAlg`, `X'Spur`, `X'OneLoop` are deprecated; all package symbols now belong to a common context `X`.

- Auxiliary Passarino-Veltman function `pvB` is now obsolete, and is covered by by higher weight `PVB` functions.

- Off shell fermion self energy form factor $C(p^2)$ has been redefined in `Projector` without a factor of $i$ in front.

Appendix B: New functions/symbols introduced in version 2.0

- `LoopRefineSeries` generates a (multiple) Taylor series expansion of one loop tensor integrals.

- `PVD` represents the Passarino-Veltman tensor coefficient function $D_{0,..,0,1,1,2,2,3,3,3}$, and substituted with an analytic expression by `LoopRefine`.

- `ScalarD0`, `ScalarDOIR12`, `ScalarDOIR13`, `ScalarDOIR16` gives the finite parts of the scalar four point function $D_0$ for real external invariants and real positive masses, as classified in [17].

- `C0Expand` and `DOExpand` expands out scalar functions `ScalarC0`, `ScalarDO` and related functions in terms of analytic expressions.

- `Kibble\phi` gives the Kibble kinematic polynomial $\phi(s_1, \ldots, s_6)$ (the Grammian determinant associated with four particle processes).

- `ContinuedDiLog` gives the Beenakker-Denner continued dilogarithm function $\text{Li}_2(x, y)$ [25].

- `MandelstamRelations` gives a list of replacement rules expressing Lorentz scalar products in terms of Mandelstam invariants and masses.

- `LScalarQ` allows declaring symbols as being Lorentz scalars.

- `FermionLine` and `FermionLineProduct` represents products of Dirac matrices sandwiched between on shell $u$ and $v$ spinors, and their direct products.
FermionLineExpand expands out and performs the Dirac algebra on FermionLine, and FermionLineProduct objects to put them in canonical SVTAP form.

[1] G. J. van Oldenborgh, Comput. Phys. Commun. 66, 1 (1991).
[2] T. Hahn and M. Perez-Victoria, Comput. Phys. Commun. 118, 153 (1999), arXiv:hep-ph/9807565 [hep-ph].
[3] S. Carrazza, R. K. Ellis, and G. Zanderighi, Comput. Phys. Commun. 209, 134 (2016), arXiv:1605.03181 [hep-ph].
[4] A. van Hameren, Comput. Phys. Commun. 182, 2427 (2011), arXiv:1007.4716 [hep-ph].
[5] G. Cullen, J. P. Guillet, G. Heinrich, T. Kleinenschmidt, E. Pilon, T. Reiter, and M. Rodgers, Comput. Phys. Commun. 182, 2276 (2011), arXiv:1101.5595 [hep-ph].
[6] A. Denner, S. Dittmaier, and L. Hofer, (2016), arXiv:1604.06792 [hep-ph].
[7] A. Arbuzov, D. Bardin, S. Bondarenko, P. Christova, L. Kalinovskaya, R. Sadykov, A. Sapronov, and T. Riemann, Proceedings, 17th International Workshop on Advanced Computing and Analysis Techniques in Physics Research (ACAT 2016): Valparaiso, Chile, January 18-22, 2016, J. Phys. Conf. Ser. 762, 012062 (2016); A. Andonov, A. Arbuzov, D. Bardin, S. Bondarenko, P. Christova, L. Kalinovskaya, G. Nanava, and W. von Schlippe, Comput. Phys. Commun. 174, 481 (2006), [Erratum: Comput. Phys. Commun. 177, 623 (2007)], arXiv:hep-ph/0411186 [hep-ph].
[8] A. Buckley, M. R. Whalley, W. J. Stirling, J. M. Butterworth, E. Nurse, and B. Waugh, in 15th International Conference on Computing in High Energy and Nuclear Physics (CHEP 2006) Mumbai, Maharashtra, India, February 13-17, 2006 (2006) arXiv:hep-ph/0605046 [hep-ph].
[9] H. H. Patel, Comput. Phys. Commun. 197, 276 (2015), arXiv:1503.01469 [hep-ph].
[10] G. Passarino and M. Veltman, Nucl. Phys. B160, 151 (1979).
[11] F. Feng, Comput. Phys. Commun. 183, 2158 (2012), arXiv:1204.2314 [hep-ph].
[12] A. Denner and S. Dittmaier, Nucl. Phys. B734, 62 (2006), arXiv:hep-ph/0509141 [hep-ph].
[13] K. G. Chetyrkin and F. V. Tkachov, Nucl. Phys. B192, 159 (1981).
[14] A. I. Davydychev, Phys. Lett. B263, 107 (1991).
[15] O. V. Tarasov, Phys. Rev. D54, 6479 (1996).
[16] G. Sulyok, (2006), arXiv:hep-ph/0609282 [hep-ph].
[17] R. K. Ellis and G. Zanderighi, JHEP 02, 002 (2008), arXiv:0712.1851 [hep-ph].
[18] A. Denner and S. Dittmaier, Nucl. Phys. B844, 199 (2011), arXiv:1005.2076 [hep-ph].
[19] G. Duplancic and B. Nizic, Eur. Phys. J. C20, 357 (2001), arXiv:hep-ph/0006249 [hep-ph].
[20] U. Nierste, Skalare Einschleifen-Integrale der Relativistischen Quantenfeldtheorie, Master’s thesis, Würzburg (1991).
[21] A. Denner, U. Nierste, and R. Scharf, Nucl. Phys. B367, 637 (1991).
[22] C. Osácar, J. Palacián, and M. Palacios, Celest. Mech. Dyn. Astro. 62, 93 (1995).
[23] A. I. Davydychev, in International Seminar on Quarks (Quarks 92) Zvenigorod, Russia, May 11-17, 1992 (1993) pp. 260–270, arXiv:hep-ph/9307323 [hep-ph].
[24] G. Duplancic and B. Nizic, Eur. Phys. J. C24, 385 (2002), arXiv:hep-ph/0201306 [hep-ph].
[25] W. Beenakker and A. Denner, Nucl. Phys. B338, 349 (1990).
[26] A. Sirlin, Nucl. Phys. B192, 93 (1981).
[27] J. S. R. Chisholm, J. Comput. Phys. 8, 1 (1971).
[28] W. Gordon, Z. Phys. 50, 630 (1928).
[29] F. Jegerlehner, Eur. Phys. J. C18, 673 (2001), arXiv:hep-th/0005255 [hep-th].
[30] S. Adler, in Lectures on Elementary Particles and Quantum Field Theory, Vol. 1, edited by S. Deser, M. Grisaru, H. Pendleton (M.I.T. Press, Cambridge, 1970) p. 1.
[31] R. Mertig, M. Bohm, and A. Denner, Comput. Phys. Commun. 64, 345 (1991).
[32] V. Shtabovenko, R. Mertig, and F. Orellana, Comput. Phys. Commun. 207, 432 (2016), arXiv:1601.01167 [hep-ph].
[33] V. Shtabovenko, (2016), arXiv:1611.06793 [physics.comp-ph].
[34] T. Hahn, Comput. Phys. Commun. 140, 418 (2001), arXiv:hep-ph/0012260 [hep-ph].
[35] T. W. B. Kibble, Phys. Rev. 117, 1159 (1960).