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

Hiren H. Patel[1, *]

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

Package-X, a *Mathematica* package for the analytic computation of one-loop integrals dimensionally regulated near 4 spacetime dimensions is described. *Package*-X computes arbitrarily high rank tensor integrals with up to three propagators, and gives compact expressions of UV divergent, IR divergent, and finite parts for any kinematic configuration involving real-valued external invariants and internal masses. Output expressions can be readily evaluated numerically and manipulated symbolically with built-in *Mathematica* functions. Emphasis is on evaluation speed, on readability of results, and especially on user-friendliness. Also included is a routine to compute traces of products of Dirac matrices, and a collection of projectors to facilitate the computation of fermion form factors at one-loop. The package is intended to be used both as a research tool and as an educational tool.

I. INTRODUCTION

Many packages are available to assist with the evaluation of one-loop integrals that appear in higher order calculations of perturbative quantum field theory. The most widely used ones are the *Mathematica* packages FeynCalc[1], FormCalc[2] and the Fortran program Golem95[3]. These packages compute one-loop integrals using the Passarino-Veltman reduction algorithm[4]

(FeynCalc and FormCalc also feature a collection of routines designed to streamline the numerical computation of a differential cross section; as such, they do substantially more than to simply compute one-loop integrals).

Nevertheless FeynCalc falls short in that it gives results of one-loop computations in terms of basis scalar functions which cannot be evaluated on their own. Instead, it is up to the user to supply their analytical forms from an external source, or to link them to yet another package (such as FF[5], LoopTools[2], or OneLoop[6]).

Moreover, one-loop integrals have many more applica-

* hiren.patel@mpi-hd.mpg.de

tions than to calculate cross sections and decay rates. Examples are the computation of ultraviolet counterterms, pole positions, residues, Peskin-Takeuchi oblique parameters, electromagnetic moments, *etc.* Many of these applications require the calculation of Feynman integrals at singular kinematic points such as at physical thresholds or zero external momenta. Since the Passarino-Veltman reduction algorithm typically breaks down at these points, it is nearly impossible to obtain results with

FEYNCALC or FORMCALC (GOLEM95 can give numerical results). But, it is also at these points where compact analytic expressions exist.

Although smaller-scale packages are available that are designed around a particular application (such as LOOL[7] and ANT[8]), there is no general-purpose software that gives analytic results to one-loop integrals for all kinematic configurations. In this regard, *Package*-X serves to fill this gap.

Package-X calculates dimensionally regulated ($d = 4 - 2\epsilon$) rank-P one-loop tensor integrals of the form

$$T_N^{\mu_1\cdots\mu_P}(p_1,\ldots,p_N;m_0,m_1\ldots,m_N) = \mu^{2\epsilon}\int\frac{d^dk}{(2\pi)^d}\frac{k^{\mu_1}\cdots k^{\mu_P}}{[k^2-m_0^2+i\varepsilon][(k+p_1)^2-m_1^2+i\varepsilon]\cdots[(k+p_N)^2-m_N^2+i\varepsilon]}, \qquad (1)$$

with up to $N = 3$ denominator factors, and finds compact analytic expressions for arbitrary configurations of external momenta p_i and real-valued internal masses m_i. The functional paradigm of the *Wolfram Language* together with the supplementary trace-taking routines included in *Package*-X allows one to compute an entire one-loop diagram at once. All output is ready for numerical evaluation and symbolic manipulation with *Mathematica*'s internal functions.

This article details the technical aspects of *Package*-X, and assumes familiarity in the use of the package. The application files are found at the Hepforge webpage `http://packagex.hepforge.org`, where the software will be maintained and periodically updated. Included among the package files is a tutorial that provides an introduction, and a complete set of documentation files that becomes embedded with the *Wolfram Documentation Center* upon installation which provides details and examples of all functions defined in *Package*-X.

II. STRUCTURE AND DESIGN OF PACKAGE

The subroutines in this package belong to one of three *Mathematica contexts* organized as in Fig. 1. The module `IndexAlg`‘ contains the rudimentary tensor-algebraic routines and serves as the backbone of *Package*-X. `OneLoop`‘ contains the algorithms and look-up tables for the computation of one-loop integrals, and `Spur`‘ includes the algorithms to perform traces over products of Dirac matrices and contains a catalog of fermion form factor projectors.

The basic *Package*-X workflow for the computation of a one-loop integral consists of three steps:

1. Call `LoopIntegrate` to carry out the covariant tensor decomposition (section III).

2. Apply on-shell conditions and other kinematic relations with *Mathematica*'s built-in functions `ReplaceAll` (`/.`) and `Rules` ($\rightarrow$).
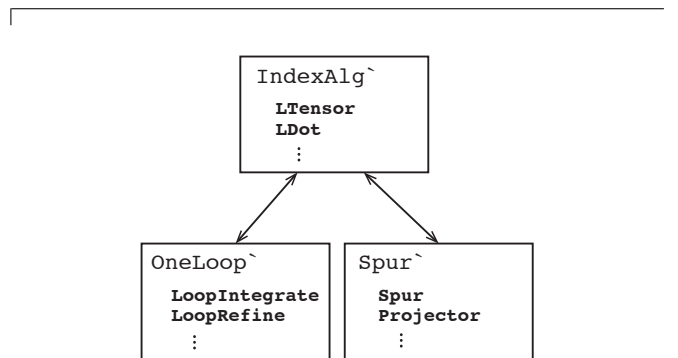

FIG. 1. Organization of functions into *contexts* as defined in *Package*-X.

3. Call `LoopRefine` to convert coefficient functions into explicit expressions (section IV).

The reasoning behind the three-step design is as follows: kinematic configurations of external invariants $p_i.p_j$ and internal masses m_i relevant to many physical applications occur at singular points of one-loop integrals, such that if they were applied *after* obtaining the general expressions, errors like $0/0$ or $0 \times \ln(0)$ would inevitably occur. To avoid such errors and to facilitate the generation of compact results, `LoopRefine` uses algorithms depending critically on the kinematic configuration supplied by the user *beforehand*.

Two other supplementary functions are provided to streamline computations involving fermions:

- `Spur` (*German for 'trace'*) computes traces of Dirac matrices that may appear in the numerators of one-loop integrals (section VII).

- `Projector` is used to project fermion self-energy and vertex form factors out of the loop integrals (section VIII).

The algorithms used by these functions are detailed in the aforementioned sections below.

III.   LOOPINTEGRATE: **COVARIANT TENSOR DECOMPOSITION**

The evaluation of an integral is initiated with `LoopIntegrate`, which carries out its covariant tensor decomposition in terms of scalar coefficient functions. For example (omitting the $+i\varepsilon$),

$$\texttt{LoopIntegrate}[\texttt{k}_\mu \texttt{k}_\nu \texttt{k}_\rho, \texttt{k}, \texttt{p1}, \texttt{m0}, \texttt{m1}] :$$

$$\left(\tfrac{i}{16\pi^2}\right)^{-1} \mu^{2\epsilon} \int \frac{d^d k}{(2\pi)^d} \frac{k^\mu k^\nu k^\rho}{[k^2 - m_0^2][(k+p_1)^2 - m_1^2]} \longrightarrow \{[p_1][g]\}^{\mu\nu\rho} B_{001} + \{[p_1]^3\}^{\mu\nu\rho} B_{111} , \qquad (2)$$

$$\texttt{LoopIntegrate}[\texttt{k}_\mu \texttt{k}_\nu, \texttt{k}, \texttt{p1}, \texttt{p2}, \texttt{m0}, \texttt{m1}, \texttt{m2}] :$$

$$\left(\tfrac{i}{16\pi^2}\right)^{-1} \mu^{2\epsilon} \int \frac{d^d k}{(2\pi)^d} \frac{k^\mu k^\nu}{[k^2 - m_0^2][(k+p_1)^2 - m_1^2][(k+p_2)^2 - m_2^2]} \longrightarrow$$
$$\{[g]\}^{\mu\nu} C_{00} + \{[p_1]^2\}^{\mu\nu} C_{11} + \{[p_1][p_2]\}^{\mu\nu} C_{12} + \{[p_2]^2\}^{\mu\nu} C_{22} \qquad (3)$$

Here, B_{001}, B_{111}, C_{00} *etc.* are coefficient functions that depend only on Lorentz invariants, $p_i.p_j$ and m_i. Note that as indicated in the left hand sides, an overall constant $(\tfrac{i}{16\pi^2})$ is factored out of the natural integration measure $\mu^{2\epsilon} \frac{d^d k}{(2\pi)^d}$ to simplify the output. Each coefficient function multiplies a totally symmetric tensor, denoted $\{\dots\}^{\mu\cdots}$ in the notation of [9], containing products of external momentum four-vectors p_i^μ and the metric tensor $g^{\mu\nu}$. These tensors are generated by a *Package*-X internal function (inside `IndexAlg`), which utilizes *Mathematica*'s built-in function `Permutations`. The time to generate the corresponding symmetric tensors grows factorially with the rank of tensor integrals.

For integrals with high powers of contracted loop momenta, such as

$$\mu^{2\epsilon} \int \frac{d^d k}{(2\pi)^d} \frac{k^\alpha k^\beta (k.k)^5}{[k^2 - m_0^2][(k+p_1)^2 - m_1^2]} , \qquad (4)$$

it is necessary to obtain explicit expressions of self-contracted symmetrized high-rank tensors like $\{[p_1]^6[g]^3\}^{\alpha\beta\mu\mu\nu\nu\rho\rho\sigma\sigma\lambda\lambda}$. It would be wasteful to first generate the totally symmetric high-rank tensors, only to subsequently contract indices down to lower-rank symmetric tensors. Instead, the contraction formulae

$$(p_k)_{\mu_1} \{[p_1]^{n_1} \cdots [p_N]^{n_N} [g]^r\}^{\mu_1 \cdots \mu_P}$$
$$= \sum_{\ell=1}^N p_k \cdot p_\ell \{[\hat{p}_\ell][p_1]^{n_1} \cdots [p_N]^{n_N} [g]^r\}^{\mu_2 \cdots \mu_P}$$
$$+ (n_k + 1)\{[p_k][p_1]^{n_1} \cdots [p_N]^{n_N} [g]^{r-1}\}^{\mu_2 \cdots \mu_P} \quad (5)$$

$$g_{\mu_1\mu_2} \{[p_1]^{n_1} \cdots [p_N]^{n_N} [g]^r\}^{\mu_1 \cdots \mu_P}$$
$$= \sum_{i,j}^N p_i \cdot p_j \{[\hat{p}_i][\hat{p}_j][p_1]^{n_1} \cdots [p_N]^{n_N} [g]^r\}^{\mu_3 \cdots \mu_P}$$
$$+ \bar{\delta}_{r,0}(d+P-2+\sum_k^N n_k)\{[p_1]^{n_1} \cdots [p_N]^{n_N} [g]^{r-1}\}^{\mu_3 \cdots \mu_P} ,$$
$$(6)$$

are employed to carry out the self-contractions symbolically before converting any remaining symmetric tensors with free indices into explicit forms in terms of p_i^μ and $g^{\mu\nu}$. The time to construct self-contracted tensors in this way is reduced to follow a power law.

IV.   LOOPREFINE: **REDUCTION TO ELEMENTARY FUNCTIONS**

Once the covariant decomposition is made, and any on-shell or kinematic conditions are applied, the final step is to feed the results into `LoopRefine`, which replaces the coefficient functions with explicit expressions. The basic algorithm followed by `LoopRefine` is as follows:

STEP 1: For each coefficient function (`pvA`, `pvB`, `pvb`, or `pvC`) encountered by `LoopRefine`, symbols for internal masses are recorded (for STEP 4), and the appropriate reduction routine (see corresponding subsections below) is called.

STEP 2: The reduction of C functions for more general kinematic configurations end with the scalar function C_0. If the C_0 function is IR-divergent or has an explicit form that is sufficiently compact (as controlled by the option `ExplicitC0`), the explicit form is substituted.

STEP 3: All instances of the spacetime dimension d is replaced by $4 - 2\epsilon$, and *Mathematica*'s built-in function `Series` is called to keep the leading terms in the ϵ expansion. UV-divergences appear as $1/\epsilon$ poles, and IR-divergences appear as $1/\epsilon$ and/or $1/\epsilon^2$ poles.

STEP 4: Combine and simplify logarithms, organize the expression by the logarithms, and group the 't Hooft parameter μ^2-dependent logarithm with the $1/\epsilon$ pole in the expression (see section VI).

In the following subsections, the algorithms and accompanying formulae used by `LoopRefine` to reduce the

coefficient functions are summarized. It should be noted that nearly all algorithms are drawn from the 2005 paper by Denner and Dittmaier [9], and will be referenced henceforth as [DD]. The only formulae not taken directly from their paper are those for the auxiliary b^ξ functions in Section IV B (which is only a slight modification of the reduction formulae for B functions), and those of two additional algorithms for the reduction of C functions in special kinematic configurations (*Cases 2* and *4* in section IV C).

A. Reduction of A and B functions

The Passarino-Veltman coefficient A functions are simple enough to be obtained by direct integration (eqn 3.4 of [DD]):

$$A_{\underbrace{0\ldots0}_{2r}}(m_0) = \frac{(m_0^2)^{r+1}}{2^r(r+1)!}\left(\frac{1}{\bar{\epsilon}} + \ln(\frac{\mu^2}{m_0^2}) + H_{r+1}\right), \quad (7)$$

where $1/\bar{\epsilon} = 1/\epsilon - \gamma_E + \ln(4\pi)$, and H_n is the nth harmonic number.

The $B_{0\ldots0\,1\ldots1}$ functions, with at least one pair of 00 indices are obtained iteratively in terms of those with fewer number of 00 indices using (eqn 4.5 of [DD]):

$$B_{\underbrace{0\ldots0}_{2r}\underbrace{1\ldots1}_{n}}(p^2; m_0, m_1) = \frac{-1}{2(n+1)}\Big[(-1)^{n+1}A_{\underbrace{0\ldots0}_{2(r-1)}}(m_1)$$
$$+ (p^2 - m_1^2 + m_0^2)B_{\underbrace{0\ldots0}_{2(r-1)}\underbrace{1\ldots1}_{n+1}}(p^2; m_0, m_1)$$
$$+ 2p^2 B_{\underbrace{0\ldots0}_{2(r-1)}\underbrace{1\ldots1}_{n+2}}(p^2; m_0, m_1)\Big], \qquad r \geq 1 \quad (8)$$

Then the $B_{1\ldots1}$ integrals (with no 00 index pairs) are obtained by explicit integration over the single Feynman parameter in (B2). Results are given in (eqn 4.8 of [DD]), but the form that tends to generate most compact expressions is

$$B_{\underbrace{1\ldots1}_{n}}(p^2; m_0, m_1) = \frac{(-1)^n}{n+1}\Big[\frac{1}{\bar{\epsilon}} + \ln\left(\frac{\mu^2}{m_1^2}\right) + \sum_{k=0}^{n}\frac{2}{n+1}\sum_{j=0}^{\lfloor\frac{n-k}{2}\rfloor}\binom{n-k}{j}\left(\frac{p^2+m_0^2-m_1^2}{2p^2}\right)^{n-k-2j}\left(\frac{\lambda(p^2,m_0^2,m_1^2)}{4(p^2)^2}\right)^j$$
$$-\sum_{k=0}^{\lfloor\frac{n-1}{2}\rfloor}\binom{n+1}{2k}\left(\frac{p^2+m_0^2-m_1^2}{2p^2}\right)^{n+1-2k}\left(\frac{\lambda(p^2,m_0^2,m_1^2)}{4(p^2)^2}\right)^k\ln\left(\frac{m_0^2}{m_1^2}\right)$$
$$+\sum_{k=0}^{\lfloor\frac{n}{2}\rfloor}\binom{n+1}{2k+1}\left(\frac{p^2+m_0^2-m_1^2}{2p^2}\right)^{n-2k}\left(\frac{\lambda(p^2,m_0^2,m_1^2)}{4(p^2)^2}\right)^k\Lambda(p^2;m_0,m_1)\Big]. \quad (9)$$

Here $\lambda(a,b,c) = a^2 + b^2 + c^2$ is the Källén function, implemented as `Kallenλ[a,b,c]`, and $\Lambda(p^2; m_0, m_1)$ is the abbreviation

$$\Lambda(p^2; m_0, m_1) = \frac{\sqrt{\lambda(p^2, m_0^2, m_1^2)}}{p^2}\ln\left(\frac{2m_0 m_1}{-p^2 + m_0^2 + m_1^2 - \sqrt{\lambda(p^2, m_0^2, m_1^2)}} + i\varepsilon\right), \quad (10)$$

implemented as `DiscB[s,m0,m1]`. In order to access $B_{1\ldots1}(p^2; m_0, m_1)$ at its singular points, a limiting procedure would need to be made at runtime in order to avoid errors such as `0/0` or `0 × ln(0)`. While *Mathematica*'s function `Limit` can eventually generate an expression, computation time is long, and output expressions are always unwieldy. Instead, a catalog of explicit expressions (also obtained by direct integration) of $B_{1\ldots1}$ at all its singular points (see Table I) is included in the source code. They may be accessed directly within *Package*-X using `LoopRefine[pvB[0, n, s, m_0, m_1]]`.

B. Reduction of auxiliary b^ξ functions

In covariant gauges, the propagator for massless vector fields

$$i\tilde{D}^{\mu\nu}(k) = \frac{-i}{k^2}\left[g^{\mu\nu} - (1-\xi)\frac{k^\mu k^\nu}{k^2}\right], \quad (11)$$

contains a gauge term that leads to an additional factor in the denominator of one-loop integrals. *Package*-X can handle such propagators inside bubble integrals, with the coefficient functions given by the auxiliary Passarino-

Veltman b^ξ functions [10]. For example,

$$\left(\frac{i}{16\pi^2}\right)^{-1}\mu^{2\epsilon}\int\frac{d^d k}{(2\pi)^d}\frac{k^\mu k^\nu k^\rho}{[k^2]^2[(k+p)^2-m^2]}$$
$$= \{[p][g]\}^{\mu\nu\rho}b_{001}^\xi + \{[p]^3\}^{\mu\nu\rho}b_{111}^\xi.$$

The reduction formulae for these functions essentially mirror those for the standard B functions. Auxiliary $b_{0\ldots0\,1\ldots1}^\xi$ functions with at least one pair of 00 indices are iteratively determined in terms of functions with fewer

00 index pairs using

$$b^\xi_{\underbrace{0\ldots0}_{2r}\underbrace{1\ldots1}_{n}}(p^2;m) = \frac{-1}{2(n+1)}\Big[B_{\underbrace{0\ldots0}_{2(r-1)}}(p^2;0,m)$$

$$+ (p^2 - m^2)b^\xi_{\underbrace{0\ldots0}_{2(r-1)}\underbrace{1\ldots1}_{n+1}}(p^2;m)$$

$$+ 2p^2 b^\xi_{\underbrace{0\ldots0}_{2(r-1)}\underbrace{1\ldots1}_{n+2}}(p^2;m)\Big], \qquad r \ge 1, \quad (12)$$

and the $b^\xi_{1\ldots1}$ functions with no 00 index pairs are obtained by direct integration over the single Feynman parameter in (B3). The integral is finite if $n \ge 1$, with the result

$$b^\xi_{\underbrace{1\ldots1}_{n}}(p^2;m) =$$

$$\frac{(-1)^{n-1}}{p^2}\Big[-\frac{1}{n} + \sum_{k=1}^{n-1}\frac{1}{n-k}\frac{m^2}{p^2-m^2}\left(\frac{p^2-m^2}{p^2}\right)^k$$

$$+ \frac{m^2}{p^2-m^2}\left(\frac{p^2-m^2}{p^2}\right)^n \ln\left(\frac{m^2}{m^2-p^2}+i\varepsilon\right)\Big].$$

If $n = 0$ (a case that is not met in practice since the gauge part of the spin-1 propagator guarantees two powers of momenta in the numerator), the auxiliary b^ξ function is IR-divergent.

Explicit expressions at the various singular points of $b^\xi_{1\ldots1}$ (see Table I) are included in the *Package-X* source code.

C. Reduction of C functions

The reduction of coefficient C functions is significantly complicated by its numerous singular points. Although the standard Passarino-Veltman reduction algorithm is applicable at almost all points (*Case 1* below), different formulae are needed to handle the various singular cases (*Cases 2 – 6*). `LoopRefine` identifies the nature of the kinematic configuration and applies the appropriate reduction method.

Cases 1, 3, 5 and *6* are taken from [DD]. Note that since the emphasis of [DD] is on numerical stability and not on generating analytic expressions, the algorithms presented there do not automatically give compact expressions. The algorithm under *Case 2* is new, and while technically it is covered by *Case 1*, it leads to more compact expressions. Furthermore, an algorithm to handle the reduction at physical thresholds (applied in *Case 3* below) is not completely covered by [DD]. This gap is filled by the formulae under *Case 4*.

The arguments of the coefficient C functions are ordered differently in *Package-X* as compared to those used by other authors. See Appendix A for details.

In the reduction formulae below, the following kinematic abbreviations are used (which differ slightly from [DD] by numeric factors):

$$f_j = p_j^2 - m_j^2 + m_0^2, \qquad j = \{1, 2\}$$

$$Z = \begin{pmatrix} p_1^2 & p_1.p_2 \\ p_2.p_1 & p_2^2 \end{pmatrix} \qquad \text{(Gramian matrix)}$$

$$q^2 = p_1^2 + p_2^2 - 2p_1.p_2$$

$$\det Z = \tfrac{1}{4}\lambda(q^2, p_1^2, p_2^2) \qquad (13)$$

$$\tilde{Z} = \begin{pmatrix} p_2^2 & -p_1.p_2 \\ -p_1.p_2 & p_1^2 \end{pmatrix} \qquad \text{(cofactor matrix)}$$

$$\tilde{X}_{0j} = \begin{pmatrix} p_2^2 f_1 - p_1.p_2 f_2 \\ -p_1.p_2 f_1 + p_1^2 f_2 \end{pmatrix} \quad j = \{1, 2\}$$

Furthermore, hatted indices on coefficient functions (*e.g.* $B_{\hat{k}0\ldots01\ldots1}$) indicate the removal of those indices. Coefficient B functions derived by canceling denominators from three-point integrals are abbreviated by

$$B_{\ldots}(\hat{D}_1) = B_{\ldots}(p_2^2; m_0, m_2) \qquad (14)$$

$$B_{\ldots}(\hat{D}_2) = B_{\ldots}(p_1^2; m_0, m_1). \qquad (15)$$

If the denominator $(k^2 - m_0^2)^{-1}$ independent of an external momentum vector is cancelled, a shifted form of the B function is used:

$$B_{\underbrace{0\ldots0}_{2r}\underbrace{1\ldots1}_{n_1}\underbrace{2\ldots2}_{n_2}}(\hat{D}_0) =$$

$$(-1)^{n_1}\sum_{j=0}^{n_1}\binom{n_1}{j}B_{\underbrace{0\ldots0}_{2r}\underbrace{1\ldots1}_{n_2+j}}(q^2;m_1,m_2). \quad (16)$$

Whenever $n_1 > n_2$ the invariance property

$$B_{\underbrace{0\ldots0}_{2r}\underbrace{1\ldots1}_{n_1}\underbrace{2\ldots2}_{n_2}}(\hat{D}_0) = B_{\underbrace{0\ldots0}_{2r}\underbrace{1\ldots1}_{n_2}\underbrace{2\ldots2}_{n_1}}(\hat{D}_0)\Big|_{m_1\leftrightarrow m_2} \quad (17)$$

is used to keep the number of terms in the sum to a minimum. *Cases 2* and *4* require expressions for the B functions with the number of 00 index pairs continued to $r = -1$. Details of this function are found in Appendix E.

Finally, formulae for *Cases 1, 3* and *5* below contain explicit dependence on spacetime dimension $d = 4 - 2\epsilon$ appearing in denominators of certain prefactors. In the course of reduction, the $\mathcal{O}(\epsilon)$ part multiplying any lower coefficient functions combines with their UV $1/\epsilon$ poles[1], and gives rise to finite polynomials in kinematic variables. Although this can be automatically handled by `Series` at STEP 3, the reduction algorithm performs much faster if these polynomials are explicitly supplied. They are obtained by integration over the Feynman parameters as described at the end of Appendix B.

Case 1: $\det Z \neq 0$

At non-singular kinematic configurations with $\det Z \neq 0$, the original [4] Passarino-Veltman reduction formula is used (eqns 5.10, 5.11 of [DD]):

$$\begin{cases} C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}} = \dfrac{1}{2\det Z}\sum_{k=1}^{2}\tilde{Z}_{jk}\Big[\delta_{n_k,\delta_{jk}}B_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_{\bar{k}}-\delta_{\bar{k}1}}}(\hat{D}_k) - B_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1-1}\underbrace{2...2}_{n_2}}(\hat{D}_0) \\ \qquad\qquad\qquad\qquad - f_k C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1-1}\underbrace{2...2}_{n_2}}(\hat{D}_0) - 2(n_k - \delta_{jk})C_{\hat{k}\underbrace{0...0}_{2r+2}\underbrace{1...1}_{n_1-1}\underbrace{2...2}_{n_2}}\Big], \quad n_1 \geq 1 \\[4ex] C_{\underbrace{0...0}_{2r}} = \dfrac{1}{2(d-4+2r)}\Big[B_{\underbrace{0...0}_{2r-2}}(\hat{D}_0) + 2m_0^2 C_{\underbrace{0...0}_{2r-2}1} + f_2 C_{\underbrace{0...0}_{2r-1}2}\Big], \qquad\qquad\qquad r \geq 1 \end{cases} \quad (18)$$

where $\bar{k} = \begin{cases} 1, & k=2 \\ 2, & k=1 \end{cases}$. In the first equation, $j=1$ is taken, although the choice $j=2$ would give equivalent results. If $n_1 = 0$ with $n_2 > 0$, then the relation (B5) is used and the first equation is applied.

Case 2: Ellis-Zanderighi triangle 6

Coefficient C functions for which arguments are $(m_0^2, s, m_2^2; m_2, 0, m_0)$—or an equivalent permutation thereof—are already covered by *Case 1*. However, final expressions obtained from it tend not to give the most compact formulae for this kinematic configuration. More compact formulae are obtained by directly integrating over the Feynman parameters in (B4); see Appendix C for derivation. It is of note that the corresponding scalar function C_0 is the IR-divergent three-point function, 'triangle 6', as classified by Ellis and Zanderighi [11]. In Eqs. (19) and (20), it is assumed that at least one of r, n_1 or n_2 is nonzero.

$$\begin{cases} C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}(m_0^2, s, m_2^2; m_2, 0, m_0) = \\ \qquad\qquad \dfrac{(-1)^{n_1}}{2}\dfrac{n_1!(n_2+2r-1)!}{(n_1+n_2+2r)!}\big(1+2\epsilon(H_{n_1+n_2+2r}-H_{n_2+2r-1})\big)B_{\underbrace{0...0}_{2r-2}\underbrace{1...1}_{n_2}}(s;m_0,m_2), \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad n_2 \neq 0 \text{ or } r \neq 0 \\ C_{\underbrace{1...1}_{n_1}}(m_0^2, s, m_2^2; m_2, 0, m_0) = \\ \qquad\qquad (-1)^{n_1}\Big[C_0(m_0^2, s, m_2^2; m_2, 0, m_0) - \dfrac{1}{2}\Big(H_{n_1} + \epsilon(H_{n_1}^2 + H_{n_1}^{(2)})\Big)B_{\underbrace{0...0}_{-2}}(s;m_0,m_2)\Big] \end{cases} \quad (19)$$

where $H_n^{(r)}$ is the nth harmonic number of order r. If the arguments take the form $(s, m_0^2, m_2^2; 0, m_2^2, m_0)$, then the identity (B5) is applied, and the equations above are valid.

A different formula is needed if the off-shell momentum s is in the third position:

$$C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}(m_2^2, m_0^2, s; m_0, m_2, 0) = \dfrac{(-1)^{n_2}}{2}\dfrac{1}{n_1+n_2+2r}\sum_{k=0}^{n_2}\binom{n_2}{k}\Big(1+\dfrac{2\epsilon}{n_1+n_2+2r}\Big)B_{\underbrace{0...0}_{2r-2}\underbrace{1...1}_{n_1+k}}(s;m_0,m_2) \quad (20)$$

To apply Eqs. (19) and (20) above, explicit forms of the scalar B function with the number of 00 index pairs taken to $r = -1$ is occasionally needed. These functions are discussed in Appendix E.

————

[1] For the argument that they are only of UV origin (and not IR), see the argument in Sec. 5.8 of [DD]

Case 3: $\det Z = 0$ *but* $\tilde{X}_{0j} \neq 0$

With $\det Z = 0$, the primary reduction formulae are rearranged to give: (eqns 5.38 and 5.40 of [DD])

$$
\begin{cases}
C_{\underbrace{0...0}_{2r}} = \dfrac{1}{d+2r-3}\left(B_{\underbrace{0...0}_{2r-2}}(\hat{D}_0) - m_0^2 C_{\underbrace{0...0}_{2r-2}}\right) + \dfrac{1}{2(d+2r-3)\tilde{Z}_{kl}}\sum_{n,m=1}^{2}\left(\delta_{km}\delta_{nl} - \delta_{kl}\delta_{nm}\right) \\[2em]
\quad \times \left\{\displaystyle\sum_{j=1}^{2} Z_{nj}\left[(1-\delta_{mj})B_{\underbrace{0...0}_{2r-2}1}(\hat{D}_m) - B_{j\underbrace{0...0}_{2r-2}}(\hat{D}_0)\right] + \dfrac{1}{2}f_m\left[-B_{\underbrace{0...0}_{2r-2}}(\hat{D}_n) + B_{\underbrace{0...0}_{2r-2}}(\hat{D}_0) + f_n C_{\underbrace{0...0}_{2r-2}}\right]\right\} \quad r>0 \\[2em]
C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}} = \dfrac{1}{\tilde{X}_{0j}}\displaystyle\sum_{k=1}^{2}\tilde{Z}_{jk}\left(\delta_{n_k 0}B_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_{\bar{k}}}}(\hat{D}_k) - B_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}(\hat{D}_0) - 2n_k C_{\hat{k}00\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}\right)
\end{cases}
$$
(21)

The value of j chosen (1 or 2) is the one for which the corresponding $\tilde{X}_{0j}$ is non-vanishing. If both elements are vanishing, then *Case 4* is applied. Note that the second relation is valid even when either $n_1 = 0$ or $n_2 = 0$. In particular, when $r = n_1 = n_2 = 0$ the final term in that relation vanishes, and leads to the reduction of the scalar C_0 function in terms of scalar B_0 functions.

Case 4: vanishing $\det Z$ *and* $\tilde{X}_{0j}$

When the physical threshold (corresponding to $\tilde{X}_{0j} = 0$ for both $j = \{1,2\}$) coincides with the boundary of the physical region ($\det Z = 0$), then *Cases 1—3* are inapplicable. For this kinematic configuration, the reduction formulae in [DD] eqns (5.49) and (5.53) can be used provided at least one element of

$$
\tilde{X}_{ij} = \begin{pmatrix} 4m_0^2 p_2^2 - f_2^2 & -2m_0^2(p_1^2 + p_2^2 - q^2) + f_1 f_2 \\ -2m_0^2(p_1^2 + p_2^2 - q) + f_1 f_2 & 4m_0^2 p_1^2 - f_1^2 \end{pmatrix}
$$

is non-vanishing. However, no reduction methods are presented in [DD] that are valid when all four elements of $\tilde{X}_{ij}$ are vanishing, because an expansion around that point is not known[2]. This exceptional configuration is needed for the computation of elastic form factors at zero momentum such as electron $g - 2$. To fill this gap, a new set of reduction formulae are used that is valid regardless of the form of $\tilde{X}_{ij}$, provided at least one of p_1^2, p_2^2 or q^2 is non-vanishing. These formulae are derived in Appendix D.

If $p_2^2 \neq 0$,

$$
\begin{aligned}
C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}} &= \frac{(-1)^{n_1+n_2}}{2}\sum_{j=0}^{n_2}\binom{n_2}{j}\alpha^{n_2-j}\left\{\frac{n_1!(n_2-j)!}{(n_1+n_2-j+1)!}\sum_{k=0}^{j}\left[\binom{j}{k}(-\alpha)^{j-k}(-1)^k B_{\underbrace{0...0}_{2r-2}\underbrace{1...1}_{k}}(\hat{D}_1)\right]\right. \\
&\left.+ \sum_{k=0}^{n_1}\frac{(-1)^{n_2}}{n_2-j+k+1}\binom{n_1}{k}\left[(1-\alpha)^{j+1}(-1)^{n_2}B_{\underbrace{0...0}_{2r-2}\underbrace{1...1}_{n_2+k+1}}(\hat{D}_0) - (-\alpha)^{j+1}B_{\underbrace{0...0}_{2r-2}\underbrace{1...1}_{n_2+k+1}}(\hat{D}_2)\right]\right\},
\end{aligned}
$$
(22)

where $\alpha = -q^2 + p_1^2 + p_2^2/(2p_2^2)$.

If $p_2^2 = 0$, then $\det Z = 0$ implies $q^2 = p_1^2$, and the formula

$$
C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}} = \frac{(-1)^{n_1+1}}{2(n_2+1)}\sum_{k=0}^{n_1}\binom{n_1}{k}B_{\underbrace{0...0}_{2r-2}\underbrace{1...1}_{n_2+k+1}}(\hat{D}_2)
$$
(23)

is used. If $p_1^2 = p_2^2 = q^2 = 0$, then these formulae are inapplicable and *Case 5* is needed. Note that when $r = 0$ in either (22) or (23), the B functions continued to $r = -1$ are needed (see Appendix E).

---

[2] A. Denner, *private correspondence*

Case 5: All elements of Z vanishing

If all external invariants are vanishing $p_1^2 = p_2^2 = q^2 = 0$, then the following are applied (eqns 5.62 and 5.63 of [DD]):

$$\begin{cases} C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}} = \dfrac{1}{d + 2(n_1 + n_2 + r - 1)}\Big[B_{\underbrace{0...0}_{2r-2}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}(\hat{D}_0) + m_0^2 C_{\underbrace{0...0}_{2r-2}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}\Big], & r \geq 1 \\[2em] C_{\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}} = \dfrac{1}{f_k}\Big[\delta_{n_k 0}B_{\underbrace{1...1}_{n_{\bar{k}}}}(\hat{D}_k) - B_{\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}(\hat{D}_0) - 2n_k C_{\hat{k}00\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}\Big] \end{cases} \tag{24}$$

In the second equation, the index k is chosen such that f_k is non-vanishing. As in *Case 3*, the second relation is valid for vanishing n_1 or n_2, and is used to reduce the scalar C_0 function to B_0 functions for $n_1 = n_2 = 0$.

Case 6: All elements of Z and f_k vanishing

Finally, if also the f_k's are vanishing, the following formulae are used (eqns 5.71 and 5.72 of [DD]):

$$\begin{cases} C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}} = \dfrac{-1}{2(n_k + 1)}B_{k\underbrace{0...0}_{2r-2}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}(\hat{D}_0), & r \geq 1 \\[2em] C_{\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}} = \dfrac{1}{m_0^2}\Big[(d + 2n_1 + 2n_2)C_{00\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}} - B_{\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}(\hat{D}_0)\Big] \end{cases} \tag{25}$$

Equivalent results are obtained for $k = 1$ or 2 in the first equation. The choice $k = 1$ is used in *Package*-X. Note that when $r = n_1 = n_2 = 0$, these relations together permit the reduction of the scalar C_0 function in terms of scalar B_0 functions.

V. THE SCALAR C_0 FUNCTION: ANALYTIC EXPRESSIONS AND NUMERICAL IMPLEMENTATION

The algorithms for the reduction of coefficient C-functions for which $\det Z \neq 0$ (*Cases 1* and *2* in the previous section) end with the UV-finite scalar function $C_0(p_1^2, p_2^2, q^2, m_2, m_1, m_0)$. To complete the computation of the one loop integral and to make the final output useable, the scalar function must be replaced. For this purpose, a complete catalog of analytic expressions for C_0 where $\det Z \neq 0$—each one obtained by direct integration—is included in the source file (see Table I). Many such expressions are scattered throughout the literature. The general formula with non-zero kinematic variables appears in [12]. All IR-divergent three-point formulae are given in [11], and some special cases appear in unpublished notes [13].

Although a complete catalog of analytic expressions of C_0 is available, not all cases are automatically substituted by `LoopRefine` at STEP 2. The functions that are substituted are only those that are IR-divergent (to faithfully display the $1/\epsilon$ poles in the final output), and those for which a sufficiently simple/compact expression is known. For more complicated finite cases, `LoopRefine`

simply outputs[3] `pvC0[...]`, with the function itself implemented numerically, (summarized below). The reason for this design choice is as follows:

Firstly, in cases for which no simple form is known, the general formula [12] in terms of 12 dilogarithms would have to be given. This expression for C_0 alone would occupy a very large part of the output overwhelming the remainder of the expression, thus defeating the original purpose of producing *compact* expressions. Secondly, the dilogarithm function is computationally very expensive. When numerics are required, a brute-force evaluation of all the dilogarithms is grossly inefficient, leading to excessively slow numerical evaluations.

The main features of the code for the rapid numerical evaluation of the three-point scalar function for real masses and external momenta are as follows:

- The imaginary part of C_0 in the physical region (defined by $\lambda(q^2, p_1^2, p_2^2) > 0$) is obtained by applying Cutkosky's rule, and with a straightforward continuation into the unphysical region (defined by $\lambda(q^2, p_1^2, p_2^2) < 0$) [14, 15]. Its computation requires the evaluation of a single logarithm.

———

[3] If the explicit analytic form *is* desired, the option `ExplicitC0 → All` can be supplied to `LoopRefine`.

B-functions — Section IV A

$B_{1\ldots1}(0;0,0)$	$B_{1\ldots1}(p^2;0,0)$	$B_{1\ldots1}(m_0^2;m_0,0)$	$B_{1\ldots1}(p^2;m_0,0)$
$B_{1\ldots1}(m_0^2;m_0,m_0)$	$B_{1\ldots1}(0;m_0,m_0)$	$B_{1\ldots1}(0;m_0,m_1)$	$B_{1\ldots1}((m_0{+}m_1)^2;m_0,m_1)$
$B_{1\ldots1}(0;0,m_1)$	$B_{1\ldots1}(m_1^2;0,m_1)$	$B_{1\ldots1}(p^2;0,m_1)$	$B_{1\ldots1}((m_0{-}m_1)^2;m_0,m_1)$
$B_{1\ldots1}(0;m_0,0)$			

B-functions with $r = -1$ — Appendix E

$B_{1\ldots1}(0;0,0)$	$B_{1\ldots1}(p^2;0,0)$	$B_{1\ldots1}(m_0^2;m_0,0)$	$B_{1\ldots1}(p^2;m_0,0)$
$B_{1\ldots1}(0;0,m_1)$	$B_{1\ldots1}(0;m_0,m_0)$	$B_{1\ldots1}(0;m_0,m_1)$	$B_{1\ldots1}((m_0{+}m_1)^2;m_0,m_1)$
$B_{1\ldots1}(0;m_0,0)$	$B_{1\ldots1}(m_1^2;0,m_1)$	$B_{1\ldots1}(p^2;0,m_1)$	$B_{1\ldots1}((m_0{-}m_1)^2;m_0,m_1)$

Auxiliary b^ξ-functions — Section IV B

$b^\xi_{1\ldots1}(0,0)$	$b^\xi_{1\ldots1}(p^2,0)$
$b^\xi_{1\ldots1}(0,m)$	$b^\xi_{1\ldots1}(m^2,m)$

Scalar C-functions — Section V

$C_0(0,0,0;0,0,0)$	$C_0(0,0,q^2;0,m_0,m_0)$	$C_0(m_0^2,0,q^2;0,0,m_0)$	$C_0(0,p_2^2,q^2;m_2,0,0)$
$C_0(0,0,q^2;0,0,0)$	$C_0(0,0,q^2;0,m_1,m_0)$	$C_0(0,m_2^2,q^2;m_2,0,0)$	$C_0(p_1^2,0,q^2;m_2,m_1,m_0)$
$C_0(0,0,m_2^2;m_2,0,0)$	$C_0(0,0,q^2;m_0,m_0,m_0)$	$C_0(m_0^2,0,m_2^2;m_2,0,m_0)$	$C_0(m_0^2,m_0^2,q^2;0,0,m_0)$
$C_0(0,0,q^2;m_2,0,0)$	$C_0(0,0,q^2;m_2,m_0,m_0)$	$C_0(p_1^2,p_2^2,q^2;m_2,m_1,m_0)$	$C_0(m_0^2,p_2^2,m_0^2;m_0,0,m_0)$
$C_0(0,0,q^2;0,0,m_0)$	$C_0(0,0,q^2;m_2,m_1,m_0)$	$C_0(0,m_0^2,q^2;0,m_0,m_0)$	$C_0(m_0^2,p_2^2,m_2^2;m_2,0,m_0)$
$C_0(0,0,m_0^2;m_0,m_0,m_0)$	$C_0(p_1^2,0,q^2;0,0,0)$	$C_0(0,p_2^2,q^2;m_0,m_0,m_0)$	$C_0(p_1^2,p_2^2,q^2;0,0,0)$

TABLE I. Special kinematic cases of the Passarino-Veltman coefficient functions B, b^ξ and C for which explicit expressions are included in the source file `OneLoop.m`. Further information for these functions is found in the indicated sections

- The real part of C_0 requires evaluations of only the real part (in the physical region) or only the imaginary part (in the unphysical region) of the dilogarithm, but not both. Calling `PolyLog` would lead to needless computation of both parts by the *Mathematica* Kernel. Following [16], the real and imaginary parts of the dilogarithm function are implemented separately.

- For the real part of C_0 in the physical region, the $+i\varepsilon$ prescription is irrelevant (since it influences only the imaginary part which is anyway evaluated using Cutkosky's rule). Then, either the arguments of the 12 dilogarithms come in complex-conjugate pairs (for which the real part of the dilogarithms are identical and are added reducing the number of dilog evaluations), or the arguments are purely real (for which the real parts of the dilogarithms are rapidly evaluated using real arithmetic).

- The code is compiled to the *Wolfram Virtual Machine* (using `Compile`), leading to a substantial boost in computation speed.

In the physical region, up to a 200-fold increase in speed is achieved compared to brute-force *Mathematica* evaluation by the Kernel. In the unphysical region, up to a 20-fold increase in speed is obtained. If the option `CompilationTarget → "C"` to `Compile` is set, its performance rivals that of the Fortran implementation in LoopTools, with *Package-X* generating results approximately twice as fast.

VI. HANDLING THE $+i\varepsilon$ PRESCRIPTION AND SIMPLIFYING LOGARITHMS

The $+i\varepsilon$ prescription appearing in the denominators of propagator functions enforce causality in the time-ordered Green functions of a relativistic quantum field theory. In one-loop computations, it determines the branch on which the logarithms are to be evaluated. All output expressions of `LoopRefine` observe the $+i\varepsilon$ prescription and are consistent with the analytic conventions of the built-in *Mathematica* functions `Log` and `PolyLog`, which are

$$\texttt{Log[x]} \longrightarrow \lim_{\varepsilon\to0^+} \ln(x + i\varepsilon), \text{ and}$$
$$\texttt{PolyLog[2, x]} \longrightarrow \lim_{\varepsilon\to0^+} \mathrm{Li}_2(x - i\varepsilon).$$

Because *Package-X* assumes real external invariants and internal masses, almost all analytic formulae can be expressed compactly in terms of the built-in functions.

Whenever `LoopRefine` generates a logarithm containing the ratio of two internal masses, the ratio may be flipped to bring the logarithm to 'canonical form', *e.g.*

$$\texttt{Log}\left[\frac{\texttt{m1}^2}{\texttt{m0}^2}\right] \longrightarrow -\texttt{Log}\left[\frac{\texttt{m0}^2}{\texttt{m1}^2}\right]. \tag{26}$$

Since internal masses are assumed to be positive real, this is allowed, and helps to keep the logarithmic parts compact.

In the course of reduction, regardless of whether the final expression is divergent or finite, multiple logarithms of ratios of several scales with the 't Hooft parameter μ^2 are typically generated, *e.g.*

$$\mathtt{a}\,\mathtt{Log}\Big[\frac{\mu\mathtt{R}^2}{-\mathtt{s}}\Big] + \mathtt{b}\,\mathtt{Log}\Big[\frac{\mu\mathtt{R}^2}{\mathtt{m}^2}\Big] + \mathtt{c}\,\mathtt{Log}\Big[\frac{\mu\mathtt{R}^2}{\mathtt{m}^2-\mathtt{s}}\Big]\,. \qquad (27)$$

It is found that by consistently keeping μ^2 in the numerator, the $+i\varepsilon$ *prescription* is always observed – even when the other scales are external invariants that may become time-like. The μ^2 from each logarithm are brought into a single logarithm by forming the ratio with a variable that is known to be positive (which were recorded at STEP 1):

$$(27) = \mathtt{a}\,\mathtt{Log}\Big[\frac{\mathtt{m}^2}{-\mathtt{s}}\Big] + (\mathtt{a}+\mathtt{b}+\mathtt{c})\,\mathtt{Log}\Big[\frac{\mu\mathtt{R}^2}{\mathtt{m}^2}\Big] + \mathtt{c}\,\mathtt{Log}\Big[\frac{\mathtt{m}^2}{\mathtt{m}^2-\mathtt{s}}\Big]\,. \qquad (28)$$

That way, the coefficient—(a+b+c) in this example—of the μ^2-dependent logarithm always matches that of the $1/\epsilon$ pole elsewhere in the expression, and are grouped before presenting the results. If the final expression were in fact finite without a $1/\epsilon$ pole, the coefficient would cancel exactly.

In more complicated cases, expressions cannot be given compactly assuming a universal sign for the infinitesimal imaginary part. In this case, *Mathematica*'s built-in functions `Log` and `PolyLog` are unsuitable. For this purpose, two new analytic functions are defined in *Package*-X (within `OneLoop`‘):

$$\mathtt{Ln}[\mathtt{x},a] \longrightarrow \lim_{\varepsilon\to 0^+} \ln(x+ia\varepsilon)\,, \text{ and}$$
$$\mathtt{DiLog}[\mathtt{x},a] \longrightarrow \lim_{\varepsilon\to 0^+} \mathrm{Li}_2(x+ia\varepsilon)\,.$$

The (real part of the) second argument a controls the side of the branch on which these functions evaluate. A simple example that uses `DiLog` in its output can be found by running

$$\mathtt{LoopRefine}[\mathtt{pvC0}[0,\mathtt{m}^2,\mathtt{s},0,\mathtt{m},\mathtt{m}]]\,.$$

VII. SPUR: COMPUTATION OF TRACES OF DIRAC MATRICES

To assist in the evaluation of one-loop integrals with internal (closed) fermion lines, *Package*-X includes the rudimentary function `Spur` (inside the module `Spur`‘) to evaluate traces over products of Dirac gamma matrices appearing in numerators. The function `Projector` helps to handle loop integrals with open fermion lines and is described in the next section. Because the primary function of *Package*-X is to compute loop integrals, with the computation of traces being a secondary feature, only a cursory description of the algorithms are given in the following two sections.

As with the rest of the algorithms in *Package*-X, the calculation of traces is rule-based at its core, and bears some resemblance to that of a much earlier *Mathematica* package TRACER[17]. However there are a number of differences listed below that lead to greater computation speed.

- Throughout the evaluation of the trace, expressions can grow very large containing many terms. Groups of terms are temporarily enclosed within a `List` to prevent the *Mathematica* kernel from automatically simplifying the large expression at each step of the computation process.
- While more complicated trace formulae such as those for products of numerous gamma matrices are recursive, non-iterative rules are used for simpler tasks such as for collecting γ_5 and $\hat{P}_L/\hat{P}_R$ within each term.
- Products of gamma matrices with repeated Lorentz indices (such as $\gamma^\mu\gamma^\nu\gamma^\rho\gamma_\mu$) are related to products with fewer gamma matrices. With more gamma matrices interposed between contracted matrices, the number of terms in the identity grows. On account of the cyclic property of the trace, these contraction identities may be applied in one of two directions. Additional rules are included so as to apply the identity in the direction with fewer number of interposed gamma matrices.
- Traces that multiply γ_5 are tagged differently to set it apart from those without it. This way, rules for computing traces with γ_5 and those without γ_5 are separated, and saves some time when the kernel searches for the appropriate rules.

When compared to the other *Mathematica* packages FEYNCALC and TRACER, *Package*-X generally gives results around 10 times faster. As an example, the trace

$$\mathrm{Tr}\left[(\not{k}-\not{p}_1-\not{p}_2+m)\gamma^\nu(g_L\hat{P}_L+g_R\hat{P}_R)(\not{k}-\not{p}_2+m)\right.$$
$$\left.\gamma^\rho(g_L\hat{P}_L+g_R\hat{P}_R)(\not{k}+m)\gamma^\mu(g_L\hat{P}_L+g_R\hat{P}_R)\right] \qquad (29)$$

was calculated with each package and computation times were recorded (with `Timing`). The results on a 2.93 GHz Intel i7 processor are:

Package-X	0.096 s
FEYNCALC 8.2.0	1.03 s
TRACER 1.1	0.81 s

Part of the motivation for refining the trace-taking algorithms is due to the inclusion of fermion projectors described in the next section. When a `Projector` is included inside `Spur`, the number of terms within the trace is increased to an extent that a noticeable slowdown is observed. However, with the refinements described above,

projections onto form factors are nearly instantaneous on a modern computer.

VIII. PROJECTOR: PROJECTION ONTO FERMION FORM FACTORS

Package-X does not directly handle expressions involving open fermion chains that are relevant for fermion self energy and form factor calculations. In order to provide some support for such computations, *Package*-X comes equipped with a set of projectors. The projectors permit the projection of a loop integral with an open fermion line onto specific form factors functions.

For example, the one-loop expression for the off-shell fermion self-energy function takes the form

$$I(\not{p}) = \mu^{2\epsilon} \int \frac{d^d k}{(2\pi)^d} \frac{\mathbb{M}(k,p)}{[k^2 - m^2][(k+p)^2]}, \qquad (30)$$

where $\mathbb{M}(k,p)$ is a Dirac matrix structure that depends on the integration variable k and external momentum p. Parity conservation and Lorentz covariance allow I to be written in the form

$$I(\not{p}) = A(p^2)\not{p} + B(p^2)m, \qquad (31)$$

where the form factors A and B depend on Lorentz invariants p^2 and m^2 only. By multiplying the appropriate projectors

$$\mathcal{F}^{[A]}(p,m) = \frac{1}{4p^2}\not{p} \text{ and } \mathcal{F}^{[B]}(p,m) = \frac{1}{4m^2}$$

with the numerator of (30), and taking the trace, the form factors are obtained:

$$A(p^2) = \mu^{2\epsilon} \int \frac{d^d k}{(2\pi)^d} \frac{\text{Tr}[\mathbb{M}(k,p)\,\mathcal{F}^{[A]}(p,m)]}{[k^2 - m^2][(k+p)^2]}$$

$$B(p^2) = \mu^{2\epsilon} \int \frac{d^d k}{(2\pi)^d} \frac{\text{Tr}[\mathbb{M}(k,p)\,\mathcal{F}^{[B]}(p,m)]}{[k^2 - m^2][(k+p)^2]}.$$

The trace over the projectors convert the expressions into ordinary tensors integrals that are readily computed with *Package*-X.

A large set of pre-programmed projectors for off-shell self energy functions and on-shell scalar- and vector-vertex functions in various bases (L/R-chiral or Vector/Axial-vector) are available (as `Projector`) to streamline the computation of such integrals. These projectors are generalizations of those used in [18] for the calculation of lepton anomalous magnetic moments, and in [19] for dipole moments. A comprehensive list of available projectors is given in the built-in documentation files.

IX. CROSSCHECKS AND FURTHER DEVELOPMENT

The verification of loop integrals obtained by *Package*-X is divided into two parts: checking the reduction algorithms in Section IV, and checking the basis functions in Table I. The reduction routines for A and B coefficient functions and the basis $B_{1...1}$ functions were compared against another (unpublished) computer program developed by Huaike Guo. The reduction of C functions for *Cases 1, 3, 5* and *6* were checked against explicit formulae for the low rank functions listed in [DD]. Each C_0 scalar function was derived by hand and compared against explicit formulae in the literature where they exist [8, 11, 13, 20]. In cases where they did not exist, the analytic expressions were checked by comparing with the results of numerically integrating the corresponding Feynman parameter representations given in Appendix B.

Finally, as a combined check of the various algorithms in *Package*-X, the following well known physical quantities were computed and verified: $H \rightarrow gg$ and $\gamma\gamma$ standard model decay rates [21], electron $g-2$, and the neutrino electric and magnetic moments [22]. Each was found to be in agreement with literature.

There are a number of important limitations of *Package*-X, listed below, that guides its current line of development.

1. An analytic series expansion of the loop integral in kinematic variables is not generally possible. Currently, the only available method is to use *Mathematica*'s `Series` on the output of `LoopRefine`. However, if the result of loop integral contains specially defined function like `pvC0`, then `Series` will not work. Given that much information about a loop-integral can be gleaned from its expansion, the omission of this feature is most conspicuous.

2. *Package*-X currently supports loop integrals with up to only three denominator factors. But, as the number of denominator factors increases, so does the complexity of their analytic forms. Thus, it would not be so practical to work with such expressions for higher-point functions even if *Package*-X were to provide them. However, at special kinematic points such as at zero external momenta or at thresholds compact expressions could be obtained.

3. Gamma-5 is implemented naively in dimensional regularization. This means that the VVA or AAA three-point functions may not automatically satisfy Ward identities appropriate to the physical problem. However, the versatility of *Package*-X makes it easy to apply Adler's method [23] (see also [24]) to enforce the Ward identities.

4. Loop integrals with open fermion chains are not directly supported. As explained in Section VIII, there is no way to input an open string of Dirac matrices. Instead, the computation of fermion form factors can be done by projecting out the needed form factors.

ACKNOWLEDGMENTS

Appendix A: Conventions

For reference, the conventions for spacetime quantities are summarized in Table II. Conventions for the Passarino-Veltman functions are displayed in Table III. Note that a slightly-unconventional ordering and form for the arguments of the Passarino-Veltman functions is taken. However, this choice makes the invariance property under their pairwise interchange clear:

$$
\begin{aligned}
C_0(p_1^2, p_2^2, q^2, m_2, m_1, m_0) \\
= C_0(p_2^2, p_1^2, q^2, m_1, m_2, m_0) \\
= C_0(q^2, p_2^2, p_1^2, m_0, m_1, m_2) .
\end{aligned} \tag{A1}
$$

Appendix B: Feynman parameter integral representations of Passarino-Veltman coefficient functions

In this section, the Feynman parameter integral representation of the Passarino-Veltman coefficient one, two, and three point functions are given. They are obtained

Quantity	Convention
Metric signature	$g_{\mu\nu} = \mathrm{diag}(+,-,-,-)$
Spacetime dimension	$d = 4 - 2\epsilon$
Dirac matrix commutator	$\sigma_{\mu\nu} = \frac{i}{2}[\gamma_\mu, \gamma_\nu]$
Fifth gamma matrix	$\gamma_5 = i\gamma^0\gamma^1\gamma^2\gamma^3$
Chiral projectors	$\hat{P}_L = \frac{1}{2}(1 - \gamma_5), \hat{P}_R = \frac{1}{2}(1 + \gamma_5)$
Levi-civita symbol	$\epsilon^{0123} = +1$

TABLE II. Conventions for spacetime quantities

Function	Diagram
$A_0(m_0)$	
$B_0(p^2, m_0, m_1)$, and $b_0^\xi(p^2, m_0, 0)$	
$C_0(p_1^2, p_2^2, q^2, m_2, m_1, m_0)$, $q^2 = (p_2 - p_1)^2$	

TABLE III. Conventions for the arguments of the Passarino-Veltman functions

[25] by writing tensor integrals as derivatives of the integral representation of the corresponding scalar integral with respect to external momenta, and then matching the result to the respective covariant tensor decomposition.

$$
A_{\underbrace{0\ldots0}_{2r}}(m_0) = (4\pi\mu^2)^\epsilon \frac{(-1)^{1+r}}{2^r}\Gamma(-1+\epsilon-r)m_0^{1-\epsilon+r} \tag{B1}
$$

$$
B_{\underbrace{0\ldots0}_{2r}\underbrace{1\ldots1}_{n}}(p^2; m_0, m_1) = (4\pi\mu^2)^\epsilon \frac{(-1)^{2+r+n}}{2^r}\Gamma(\epsilon - r)
$$
$$
\times \int_0^1 \frac{dx\, x^n}{\left(p^2 x^2 + (-p^2 + m_1^2 - m_0^2)x + m_0^2 - i\varepsilon\right)^{\epsilon-r}} \tag{B2}
$$

$$
b^\xi_{\underbrace{0\ldots0}_{2r}\underbrace{1\ldots1}_{n}}(p^2; m) = (4\pi\mu^2)^\epsilon \frac{(-1)^{3+r+n}}{2^r}\Gamma(1+\epsilon - r)
$$
$$
\times \int_0^1 \frac{dx\, x^n(1-x)}{\left(p^2 x^2 + (-p^2 + m^2)x - i\varepsilon\right)^{1+\epsilon-r}} \tag{B3}
$$

$$C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}(p_1^2, p_2^2, q^2; m_2, m_1, m_0) = (4\pi\mu^2)^\epsilon \frac{(-1)^{3+r+n_1+n_2}}{2^r}\Gamma(1+\epsilon-r)$$

$$\times \int_0^1 dy \int_0^{1-y} dz\, y^{n_1} z^{n_2} \left[p_1^2 y^2 + p_2^2 z^2 + (-q^2+p_1^2+p_2^2)yz + (-p_1^2+m_1^2-m_0^2)y + (-p_2^2+m_2^2-m_0^2)z + m_0^2 - i\varepsilon\right]^{-1-\epsilon+r} \tag{B4}$$

The coefficient C function exhibits an invariance under the simultaneous interchange of indices $n_1 \leftrightarrow n_2$, external momenta $p_1^2 \leftrightarrow p_2^2$ and internal masses $m_1 \leftrightarrow m_2$,

$$C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}(p_1^2, p_2^2, q^2; m_2, m_1, m_0) = C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_2}\underbrace{2...2}_{n_1}}(p_2^2, p_1^2, q^2; m_1, m_2, m_0) \tag{B5}$$

and is frequently employed during its reduction in the most general kinematic case ($\det Z \neq 0$).

In certain reduction formulae of C-functions, some terms are multiplied by ϵ, which in the $\epsilon \to 0$ limit, pick up the UV-divergent parts of the coefficient functions in those terms. The UV divergent parts are readily obtained from the integral representation. They are controlled by the leading gamma function which for large enough r develops a $1/\epsilon$ pole as $\epsilon \to 0$. When r is large, the integrand becomes polynomial in the Feynman parameters and are readily integrated with the help of the multinomial theorem. The needed UV-divergent parts are those of the B and C functions, shown below.

$$B_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n}}(p^2; m_0, m_1)\Big|_{\substack{\text{UV-}\\\text{Div.}}} = \frac{(-1)^n}{2^r r!} \sum_{k_1+k_2+k_3=r} \binom{r}{k_1,\, k_2,\, k_3} \frac{a^{k_1} b^{k_2} c^{k_3}}{2k_1+k_2+n+1}\frac{1}{\epsilon}, \tag{B6}$$

where $a = p^2$, $b = -p^2+m_1^2-m_0^2$, and $c = m_0^2$, are polynomial coefficients of the integrand in (B2).

$$C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}(p_1^2, p_2^2, q^2; m_2, m_1, m_0)\Big|_{\substack{\text{UV-}\\\text{Div.}}}$$
$$= \frac{(-1)^{n_1+n_2}}{2^r(r-1)!} \sum_{k_1+...+k_6=r-1} \binom{r-1}{k_1,\ldots,k_6} a^{k_1} b^{k_2} c^{k_3} d^{k_4} e^{k_5} f^{k_6} \frac{(2k_1+k_3+k_4+n_1)!(2k_2+k_3+k_5+n_2)!}{(2k_1+2k_2+2k_3+k_4+k_5+n_1+n_2+2)!}\frac{1}{\epsilon}, \tag{B7}$$

where a, b, c, d, e, and f are polynomial coefficients of the integrand in (B4) in the order displayed.

Appendix C: Derivation of reduction formulae for C functions *Case 2*

The derivation of the first equation in (19) begins with the Feynman parameter representation of the coefficient C function,

$$C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}(m_0^2, s, m_2^2; m_2, 0, m_0) = (4\pi\mu^2)^\epsilon \frac{(-1)^{3+r+n_1+n_2}}{2^r}\Gamma(1+\epsilon-r)$$

$$\times \int_0^1 dy \int_0^{1-y} dz\, y^{n_1} z^{n_2} \left[m_0^2 y^2 + s z^2 + (-m_2^2+m_0^2+s)yz + (-m_0^2+m_2^2-s)z + m_0^2 - i\varepsilon\right]^{-1-\epsilon+r}. \tag{C1}$$

Upon making a change of integration variables $y = 1 - y'$ and $z = y'z'$, the nested integrals are factored:

$$\text{integrals} = \int_0^1 dy'\, y'^{-1+n_2-2\epsilon+2r}(1-y')^{n_1} \int_0^1 dz'\, z'^{n_2}\left[sz'^2 + (-s+m_2^2-m_0^2)z' + m_0^2 - i\varepsilon\right]^{-1-\epsilon+r}. \tag{C2}$$

The y' integral gives the Euler Beta function, while the z' integral is identified as the integral representation of coefficient B-function (B2).

$$C_{\underbrace{0...0}_{2r}\underbrace{1...1}_{n_1}\underbrace{2...2}_{n_2}}(m_0^2, s, m_2^2; m_2, 0, m_0) = \frac{(-1)^{n_1}}{2} B(n_2 - 2\epsilon + 2r, n_1 + 1) B_{\underbrace{0...0}_{2r-2}\underbrace{1...1}_{n_2}}(s; m_0, m_2) \tag{C3}$$

As long as one of n_2 or r is non-zero, the Beta function is finite, and its expansion to $\mathcal{O}(\epsilon)$ may be inserted yielding the first equation in (19).

On the other hand, if $n_2 = r = 0$, the Beta function develops a $1/\epsilon$ pole, so that to $\mathcal{O}(\epsilon)$, $B(-2\epsilon, n_1 + 1) = \frac{-1}{2\epsilon} - H_{n_1} - \epsilon(H_{n_1}^2 - H_{n_1}^{(2)})$. In this case, (C1) is written as

$$C_{\underbrace{1\ldots1}_{n_1}}(m_0^2, s, m_2^2; m_2, 0, m_0) = (4\pi\mu^2)^\epsilon(-1)^{n_1}\Gamma(1+\epsilon)\frac{1}{2\epsilon}\int_0^1 dz'\big(sz'^2 + z'(-s + m_2^2 - m_0^2) + m_0^2 - i\varepsilon\big)^{-1-\epsilon}$$

$$+ (4\pi\mu^2)^\epsilon(-1)^{n_1}\Gamma(1+\epsilon)\big(H_{n_1} + \epsilon(H_{n_1}^2 - H_{n_1}^{(2)})\big)\int_0^1 dz'\big(sz'^2 + z'(-s + m_2^2 - m_0^2) + m_0^2 - i\varepsilon\big)^{-1-\epsilon}. \quad \text{(C4)}$$

While the z' integral in the second line can be identified with the integral representation of the coefficient B function, the first line is identified[4] as the integral representation of the scalar function $C_0(m_0^2, s, m_2^2; m_2, 0, m_0)$ classified by Ellis and Zanderighi [11] as IR-divergent triangle 6. These identifications lead to the second equation of (19).

If the off-shell momentum s is in the third argument, the derivation starts with the change of variables $z = 1 - y - x$ in (B4) followed by an interchange of the x and y integrals to give

$$C_{\underbrace{0\ldots0}_{2r}\underbrace{1\ldots1}_{n_1}\underbrace{2\ldots2}_{n_2}}(m_2^2, m_0^2, s; m_0, m_2, 0) = (4\pi\mu^2)^\epsilon\frac{(-1)^{3+r+n_1+n_2}}{2^r}\Gamma(1+\epsilon-r)$$

$$\times \int_0^1 dx \int_0^{1-x} dy\, y^{n_1}(1-x-y)^{n_2}\big[m_0^2 x^2 + sy^2 + (s + m_0^2 - m_2^2)xy - 2m_0^2 x + (-s - m_0^2 + m_2^2)y + m_0^2 - i\varepsilon\big]^{-1-\epsilon+r}.$$

$$\text{(C5)}$$

The nested integrals are factored by making a further change of variables $x = 1 - x'$ and $y = y'x'$ to give

$$\text{integrals} = \int_0^1 dx'\, x'^{n_1+n_2+2r-1-2\epsilon}\int_0^1 dy'\, y'^{n_1}(1-y')^{n_2}\big[sy'^2 + (-s + m_2^2 - m_0^2)y' + m_0^2 - i\varepsilon\big]^{-1-\epsilon+r}. \quad \text{(C6)}$$

The x' integral is straightforward. The y' integral can be brought to a recognizable form after expanding the factor $(1 - y')^{n_2}$ as a binomial series

$$= \frac{1}{n_1 + n_2 + 2r - 2\epsilon}\sum_{k=0}^{n_2}\binom{n_2}{k}(-1)^k\int_0^1 dy'\, y'^{n_1+k}\big[sy'^2 + (-s + m_2^2 - m_0^2)y' + m_0^2 - i\varepsilon\big]^{-1-\epsilon+r}. \quad \text{(C7)}$$

The y' integral is now identified as the integral representation of the coefficient B function, yielding (20).

Appendix D: Derivation of reduction formulae for C functions *Case 4*

Two cases are distinguished for *Case 4* ($\det Z = 0$, $\tilde{X}_{0j} = 0$) depending on whether p_2^2 is vanishing. Although the steps below leading to (22) and (23) appear complicated, they essentially follow that of [12] for the evaluation of the scalar function C_0. Beginning with the integral representation (B4), a change of integration variables $y = 1 - y'$ brings the Feynman integrals to the form

$$\text{integrals} = \int_0^1 dy' \int_0^{y'} dz\,(1 - y')^{n_1} z^{n_2}\big[a\, y'^2 + b\, z^2 + c\, y'z + d\, y' + e\, z + f\big]^{-1-\epsilon+r} \quad \text{(D1)}$$

where $a = p_1^2$, $b = p_2^2$, $c = q^2 - p_1^2 - p_2^2$, $d = -p_1^2 + m_0^2 - m_1^2$, $e = p_1^2 - q^2 - m_0^2 + m_2^2$, and $f = m_1^2 - i\varepsilon$.

Under the assumption that $p_2^2 \neq 0$, a second change of variables is made $z = z' + \alpha y'$, with $\alpha = \frac{-c}{2b}$ chosen to make the coefficient of y'^2 in square brackets vanish.

$$\text{integrals} = \int_0^1 dy' \int_{-\alpha y}^{(1-\alpha)y} dz'\,(1 - y')^{n_1}(z' + \alpha y')^{n_2}\big[bz'^2 + (c + 2b\alpha)y'z' + (d + e\alpha)y' + e\, z' + f\big]^{-1-\epsilon+r} \quad \text{(D2)}$$

_____

[4] see http://qcdloop.fnal.gov/tridiv6.pdf

The choice for α implies that $c + 2b\alpha$ vanishes, and the kinematic relations $\det Z = \tilde{X}_{0j} = 0$ imply that $d + e\alpha$ vanishes, yielding

$$\text{integrals} = \int_0^1 dy \int_{-\alpha y}^{(1-\alpha)y} dz \, (1-y)^{n_1} (z + \alpha y)^{n_2} \left[bz^2 + e\,z + f \right]^{-1-\epsilon+r}, \tag{D3}$$

where the primes have been omitted. The binomial theorem is applied to the factor $(z+\alpha y)^{n_2} = \sum_j \binom{n_2}{j} \alpha^{n_2-j} y^{n_2-j} z^j$, and the order of integrations is interchanged so that

$$\text{integrals} = \sum_{j=0}^{n_2} \binom{n_2}{j} \alpha^{n_2-j} \left[\int_0^{1-\alpha} dz \int_{z/(1-\alpha)}^1 dy - \int_0^{-\alpha} dz \int_{-z/\alpha}^1 dy \right] (1-y)^{n_1} y^{n_2-j} z^j \left[bz^2 + e\,z + f \right]^{-1-\epsilon+r}. \tag{D4}$$

The y integrals in both terms yield terminating hypergeometric series most compactly written in terms of the incomplete Beta function:

$$\int_X^1 dy (1-y)^{n_1} y^{n_2-j} = \frac{n_1!(n_2-j)!}{(n_1+n_2-j+1)!} - \text{B}_X(n_2-j+1, n_1+1)$$

$$= \frac{n_1!(n_2-j)!}{(n_1+n_2-j+1)!} - \sum_{k=0}^{n_1} \frac{(-1)^k X^{n_2-j+k+1}}{(n_2-j+k+1)} \binom{n_1}{k}. \tag{D5}$$

Since the first term of (D5) is common to both integrations in (D4), they are combined to yield a total of three terms:

$$\text{integrals} = \sum_{j=0}^{n_2} \binom{n_2}{j} \alpha^{n_2-j} \left[\frac{n_1!(n_2-j)!}{(n_1+n_2-j+1)!} \int_{-\alpha}^{1-\alpha} dz \frac{z^{n_2+j}}{(bz^2+ez+f)^{1+\epsilon-r}} \right.$$

$$\left. - \int_0^{1-\alpha} dz \frac{\text{B}_{z/(1-\alpha)}(n_2-j+1, n_1+1) z^j}{(bz^2+ez+f)^{1+\epsilon-r}} + \int_0^{-\alpha} dz \frac{\text{B}_{-z/\alpha}(n_2-j+1, n_1+1) z^j}{(bz^2+ez+f)^{1+\epsilon-r}} \right] \tag{D6}$$

A change of integration variables is carried out in each term to stretch their ranges to $0 \to 1$: In the first integral, $z = z' - \alpha$, in the second integral $z = (1-\alpha)z'$, and in the third integral $z = -\alpha z'$. Consequently, the polynomials $bz^2 + ez + f$ take the shape of integrands for the B functions[5]:

$$\begin{array}{lll} \text{First term:} & p_2^2 z'^2 + (-p_2^2 + m_2^2 - m_0^2)z' + m_0^2 - i\varepsilon & := P_2(z') \\ \text{Second term:} & q^2 z'^2 + (-q^2 + m_2^2 - m_1^2)z' + m_1^2 - i\varepsilon & := P_{12}(z') \\ \text{Third term:} & p_1^2 z'^2 + (-p_1^2 + m_0^2 - m_1^2)z' + m_1^2 - i\varepsilon & := P_1(z') \end{array}$$

Upon inserting the series representation of the incomplete Beta function (D5) the result is (after dropping the primes on z)

$$\text{integrals} = \sum_{j=0}^{n_2} \binom{n_2}{j} \alpha^{n_2-j} \left\{ \frac{n_1!(n_2-j)!}{(n_1+n_2-j+1)!} \int_0^1 dz (z-\alpha)^j P_2(z)^{-1-\epsilon+r} \right.$$

$$\left. + \sum_{k=0}^{n_1} \frac{(-1)^k}{n_2-j+k+1} \binom{n_1}{k} \left[-(1-\alpha)^{j+1} \int_0^1 dz \, z^{n_2+k+1} P_{12}(z)^{-1-\epsilon+r} + (-\alpha)^{j+1} \int_0^1 dz \, z^{n_2+k+1} P_1(z)^{-1-\epsilon+r} \right] \right\} \tag{D7}$$

In the first term, the binomial theorem is applied to $(z-\alpha)^j = \sum_k \binom{j}{k}(-\alpha)^{j-k} z^k$, and the three z integrals are finally identified as integral representations of the coefficient B functions upon which (22) is obtained.

If $p_2^2 = 0$, equation (22) breaks down and another formulae is needed. In this case, $\det Z = 0$ implies $p_1^2 = q^2$ and $\tilde{X}_{0j} = 0$ implies $m_0 = m_2$ provided $p_1^2 \neq 0$. With these relations, the integrals in (D1) are already factored:

$$\text{integrals} = \int_0^1 dy' \int_0^{y'} dz \, (1-y')^{n_1} z^{n_2} \left[p_1^2 y'^2 + (-p_1^2 + m_0^2 - m_1^2)y' + m_1^2 - i\varepsilon \right]^{-1-\epsilon+r}. \tag{D8}$$

_____

[5] These quadratic polynomials may be recognized as the 'pinch functions' originating from the three cut channels of the triangle graph.

The z integration gives a factor $1/(n_2+1)$, and the factor $(1-y')^{n_1} = \sum_k \binom{n_1}{k}(-y)^k$ is expressed as a binomial series.

$$\text{integrals} = \frac{1}{n_2+1} \sum_{k=0}^{n_1} \binom{n_1}{k}(-1)^k \int_0^1 dy'\, y'^{n_2+k+1}\left[p_1^2 y'^2 + (-p_1^2 + m_0^2 - m_1^2)y' + m_1^2 - i\varepsilon\right]^{-1-\epsilon+r}. \qquad (D9)$$

Eqn (23) is obtained after identifying the y' integration as the integral representation of the coefficient B function. If all external invariants are vanishing $p_1^2 = p_2^2 = q^2 = 0$ then neither (22) nor (23) are valid, and *Case 5* is needed.

Appendix E: Coefficient B functions with $r = -1$

The two new reduction algorithms for C functions (*Cases 2* and *4*) require extending the set of basis functions to include B functions in which the index r in (B2) is continued to -1. In a certain sense, these new reduction formulae may be closely related to those in [26]. There, the authors present different reduction formulae for coefficient functions which likewise require extending the set of basis functions to scalar functions with repeated propagators.

A set of explicit expressions for the general case and at singular points is constructed and included in the *Package*-X source file (see Table I). The integration is straightforward in most cases. The functions are UV-finite for all $n \geq 0$, but with many kinematic configurations developing IR-divergent $1/\epsilon$ poles.

However, there are three kinematic cases, all corresponding to physical threshold for which the Feynman parameter integral nominally diverges even for finite but infinitesimal ϵ. To handle these cases, ϵ is taken sufficiently large and negative so that the integral converges, and then analytically continued to $\epsilon \to 0$. The results of these integrations are given below:

$$B_{\underbrace{0\ldots0}_{r=-1}\underbrace{1\ldots1}_{0}}(m_1^2; 0, m_1)$$

$$= \frac{-2}{m_1^2}\left(\frac{4\pi\mu^2}{m_1^2}\right)^\epsilon \Gamma(1+\epsilon) \int_0^1 dx\, x^{-2-2\epsilon}$$

$$= \frac{2}{m_1^2} \qquad (E1)$$

$$B_{\underbrace{0\ldots0}_{r=-1}\underbrace{1\ldots1}_{n}}(m_0^2; m_0, 0)$$

$$= \frac{2}{m_0^2}\left(\frac{4\pi\mu^2}{m_0^2}\right)^\epsilon (-1)^{n+1}\Gamma(1+\epsilon)\int_0^1 dx\, x^n(x-1)^{-2-2\epsilon}$$

$$= \begin{cases} \frac{(-1)^{n+1}}{m_0^2}n\left(\frac{1}{\epsilon} + \ln\left(\frac{\mu^2}{m_0^2}\right) + 2H_{n-1} - 2\right), & n \geq 1 \\ \frac{2}{m_0^2}, & n = 0 \end{cases}$$

$$\qquad (E2)$$

$$B_{\underbrace{0\ldots0}_{r=-1}\underbrace{1\ldots1}_{n}}\left((m_0+m_1)^2; m_0, m_1\right) = \frac{2(-1)^{n+1}}{(m_0+m_1)^2}$$

$$\times \left(\frac{4\pi\mu^2}{(m_0+m_1)^2}\right)^\epsilon \Gamma(1+\epsilon)\int_0^1 dx \frac{x^n}{\left[(x-x_+)^2\right]^{1+\epsilon}}$$

$$= \frac{2(-1)^{n+1}}{(m_0+m_1^2)}\left[\sum_{k=0}^{n-2}\frac{x_+^{n-2-k}}{k+1} + n\,x_+^{n-1}\ln\left(1 - \frac{1}{x_+}\right)\right.$$

$$\left. - \frac{1}{1-x_+} - \frac{\delta_{n,0}}{x_+}\right], \qquad x_+ = \frac{m_0}{m_0 - m_1} \quad (E3)$$

Among these integrals, only (E3) gives numerical results that are related to limiting values as threshold is reached: the real part of (E3) corresponds to the limiting value of $\operatorname{Re} B(s; m_0, m_1)$ when approached from above threshold, and the imaginary part corresponds to the limiting value of $\operatorname{Im} B(s; m_0, m_1)$ when approached below threshold.

That the integrals (E1-E3) give numerical results that do not match their limiting values as threshold is reached are not of any concern. The results above should be viewed as ill-defined divergent integrals arising at intermediate stages in the reduction of coefficient C functions. They serve to facilitate the analytic cancellation of these integrals at the end of a physically meaningful computation, such as for the electromagnetic contribution to the electron anomalous magnetic moment.

[1] R. Mertig, M. Bohm, and A. Denner, Comput.Phys.Commun. **64**, 345 (1991).
[2] T. Hahn and M. Perez-Victoria, Comput.Phys.Commun. **118**, 153 (1999), arXiv:hep-ph/9807565 [hep-ph].
[3] T. Binoth, J.-P. Guillet, G. Heinrich, E. Pilon, and T. Reiter, Comput.Phys.Commun. **180**, 2317 (2009).
[4] G. Passarino and M. Veltman, Nucl.Phys. **B160**, 151 (1979).
[5] G. van Oldenborgh, Comput.Phys.Commun. **66**, 1 (1991).
[6] A. van Hameren, Comput.Phys.Commun. **182**, 2427 (2011).
[7] A. Ilakovac and L. Popov, (2014), arXiv:1407.2727 [hep-ph].
[8] P. W. Angel, Y. Cai, N. L. Rodd, M. A. Schmidt, and R. R. Volkas, JHEP **1310**, 118 (2013), arXiv:1308.0463

[hep-ph].

[9] A. Denner and S. Dittmaier, Nucl.Phys. **B734**, 62 (2006), arXiv:hep-ph/0509141 [hep-ph].

[10] D. Y. Bardin and G. Passarino, *The standard model in the making: Precision study of the electroweak interactions* (Oxford Science Publications, 1999).

[11] R. K. Ellis and G. Zanderighi, JHEP **0802**, 002 (2008), arXiv:0712.1851 [hep-ph].

[12] G. 't Hooft and M. Veltman, Nucl.Phys. **B153**, 365 (1979).

[13] J. C. Romão, "Modern Techniques for One-Loop Calculations," (2006), http://porthos.ist.utl.pt/OneLoop/one-loop.pdf.

[14] C. Fronsdal and R. E. Norton, J.Math.Phys **5**, 100 (1964).

[15] W. Lucha, D. Melikhov, and S. Simula, Phys.Rev. **D75**, 016001 (2007), arXiv:hep-ph/0610330 [hep-ph].

[16] C. Osácar, J. Palacián, and M. Palacios, Celes. Mech. Dyn. Astro. **62**, 93 (1995).

[17] M. Jamin and M. E. Lautenbacher, Comput.Phys.Commun. **74**, 265 (1993).

[18] E. R. R. Roskies, M. Levine, *Quantum Electrodynamics*, edited by T. Kinoshita, pp. 162–217 (World Scientific, Singapore, 1990).

[19] A. Czarnecki and B. Krause, Acta Phys.Polon. **B28**, 829 (1997), arXiv:hep-ph/9611299 [hep-ph].

[20] L. G. Cabral-Rosetti and M. A. Sanchis-Lozano, J.Phys.Conf.Ser. **37**, 82 (2006), arXiv:hep-ph/0206081 [hep-ph].

[21] A. Djouadi, Phys.Rept. **457**, 1 (2008).

[22] C. Giunti and A. Studenikin, (2014), arXiv:1403.6344 [hep-ph].

[23] S. L. Adler, *Lectures on Elementary Particles and Quantum Field Theory*, edited by H. P. S. Deser, M. Grisaru, Vol. 1 (M.I.T. Press, Cambridge, 1970).

[24] F. Jegerlehner, Eur.Phys.J. **C18**, 673 (2001).

[25] A. I. Davydychev, Phys.Lett. **B263**, 107 (1991).

[26] G. Duplancic and B. Nizic, Eur.Phys.J. **C35**, 105 (2004).