

Regularization of Feynman 4-Loop Integrals with Numerical Integration and Extrapolation

E. de Doncker[1(✉)] and F. Yuasa[2]

[1] Western Michigan University, Kalamazoo, MI 49008, USA
elise.dedoncker@wmich.edu
[2] High Energy Accelerator Research Organization (KEK),
1-1 OHO Tsukuba, Ibaraki 305-0801, Japan
fukuko.yuasa@kek.jp
http://www.cs.wmich.edu/elise

Abstract. In this paper we continue our recent work on evaluating numerical approximations for a set of 4-loop self-energy integrals required in the computation of higher orders in perturbation theory. The results are given by a Laurent expansion in the dimensional regularization parameter, ε, where ε is related to the space-time dimension as $\nu = 4 - 2\varepsilon$. Although the leading-order coefficients for the diagrams with massless internal lines are given in analytic form in the literature, we obtain them using a numerical approach and with modern computational techniques. In a similar manner, we derive results for diagrams with massive lines. The SIMD (Single Instruction, Multiple Data) nature of the computation of loop integrals based on composite lattice rules lends itself to an efficient GPU implementation. We further apply double exponential numerical integration layered over the message passing interface (MPI) parallel platform. Limits of integral sequences (as the regularization parameter tends to zero) are implemented numerically using linear and nonlinear extrapolation procedures. Numerical results are given to illustrate the versatility of the methods and show some robustness with regard to the selection of sequence parameters.

Keywords: Feynman loop diagrams · Dimensional regularization · Lattice rules · High performance computing · GPUs

1 Introduction

As elementary particle physics experiments are becoming more and more precise, it is an important task to provide theoretical predictions with commensurate accuracy. To achieve this, the study of radiative corrections based on the perturbative method in Quantum Field Theory is inevitable. In higher order corrections, Feynman loop integrals appear and their accurate evaluation is a fundamental ingredient to obtain physically meaningful results. For the calculation of loop integrals, the most important issue is how to deal with divergences due to thresholds and/or ultra-violet or infrared singularities, and many excellent methods have been developed following analytical (e.g., [2,17]), semi-numerical or fully numerical schemes (e.g., [3,16]).

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
O. Gervasi et al. (Eds.): ICCSA 2022 Workshops, LNCS 13378, pp. 388–405, 2022.
https://doi.org/10.1007/978-3-031-10562-3_28

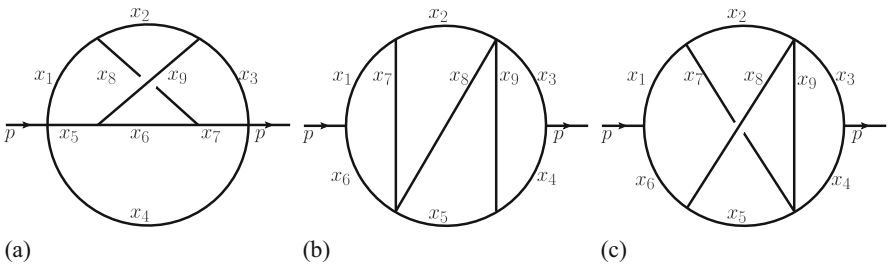


Fig. 1. 4-loop $N = 9$ diagrams [2] (a) M_{43}; (b) M_{44}; (c) M_{45}

Our work on the development of a fully numerical method for the evaluation of Feynman loop integrals emphasizes two distinct components, an extrapolation with respect to regularization parameters incorporated in the integrands to treat singularities and an efficient multivariate integration method. This versatile approach can be applied to multi-loop integrals with arbitrary masses occurring in quantum electrodynamics (QED), quantum chromodynamics (QCD) and the electroweak theory. We have presented our achievements so far for 1-, 2-, 3- and some 4-loop integrals in our past work (e.g., [5,6,8,9,26]). In this paper, we apply our methods to three other 4-loop diagrams, which are among the first thirteen complicated diagrams shown in Fig. 2 of [2]. Of these three, we presented some numerical results for two in the massless cases [8].

The 4-loop self-energy diagrams with $N = 9$ internal lines represented in Fig. 1 are referenced as M_{43}, M_{44} (*lala*) and M_{45} (*nono*) in [2,17]. The associated Feynman integral with L loops and N internal lines satisfies $\mathcal{F} = \mathcal{I}/(4\pi)^{\nu L/2}$ with

$$\begin{aligned}\mathcal{I} &= \Gamma(N - \frac{\nu L}{2})\,(-1)^N \int_{\mathcal{C}_N} \prod_{r=1}^{N} dx_r\ \delta(1 - \sum x_r)\ \frac{C^{N-\nu(L+1)/2}}{(D - i\varrho C)^{N-\nu L/2}} \\ &= \Gamma(N - \frac{\nu L}{2})\,(-1)^N \int_{\mathcal{S}_{N-1}} \prod_{r=1}^{N-1} dx_r\ \frac{C^{N-\nu(L+1)/2}}{(D - i\varrho C)^{N-\nu L/2}}. \end{aligned} \tag{1}$$

C and D are polynomials in Feynman parameters x_r of degree L and $L+1$, respectively, determined by the topology of the diagram and physical parameters; $\nu = 4 - 2\varepsilon$ is the space-time dimension (i.e., where $\nu = 4$ is modified in terms of the regularization parameter ε), and ϱ in the denominator is zero unless D vanishes within the domain. Eliminating the δ-function in the integral over the N-dimensional unit hypercube $\mathcal{C}_N$ yields the integral (1) over the d-dimensional unit simplex with $d = (N-1)$, $\mathcal{S}_d = \{\mathbf{x} \in \mathcal{C}_d \mid \sum_{j=1}^{d} x_j \le 1\}$.

Substituting $L = 4$ and $N = 9$ in (1), the integrand function is

$$f(\mathbf{x}) = \frac{C^{-1+5\varepsilon}}{D^{1+4\varepsilon}}. \tag{2}$$

Furthermore, the loop integral (1) is multiplied with a normalization factor,

$$\eta(\varepsilon)^L = \left(\frac{\Gamma(2-2\varepsilon)}{\Gamma(1+\varepsilon)\Gamma(1-\varepsilon)^2}\right)^L, \tag{3}$$

in [2], which we apply in the computation for the massless diagrams (for comparison purposes).

The integral $\mathcal{I} = \mathcal{I}(\varepsilon)$ is expanded asymptotically as $\varepsilon \to 0$ in integer powers of ε,

$$\mathcal{I}(\varepsilon) \sim \sum_{k \geq \kappa} C_k \, \varepsilon^k \qquad \text{as } \varepsilon \to 0. \tag{4}$$

The expansion is satisfied with $\kappa = 0$ for the diagrams considered, and it is our goal to approximate C_0, C_1 and C_2.

The numerical methods are outlined in Sect. 2 of this paper. Results are presented in Sect. 3, with Conclusions in Sect. 4. We include the C and D functions for the integrand definitions in Appendix A.

2 Methods

As the diagrams of Fig. 1 give rise to an expansion of the form (4) with $\kappa = 0$, an approximation to C_0 can be obtained as an integral approximation to (1) for $\varepsilon = 0$. The integrand function (2) is then $f(\mathbf{x}) = 1/(CD)$.

For calculating leading term coefficients, a sequence of integral approximations $A(\varepsilon) \approx \mathcal{I}(\varepsilon)$ for a decreasing sequence of ε can be used in an extrapolation as shown below.

2.1 Integration and Transformations

Transformations. The integral (1) is first transformed to the $(N-1)$-dimensional unit cube by the transformation $x_j = (1 - \sum_{k=1}^{j-1} x_k)\, t_j$, $1 \leq j \leq d$, with Jacobian $\prod_{j=1}^{d} (1 - \sum_{k=1}^{j-1} x_k)$, for $d = N - 1$.

We apply Sidi's Ψ_6 transformation [20] to the resulting integral,

$$\Psi_6(t) = t - (45\sin(2\pi t) - 9\sin(4\pi t) + \sin(6\pi t))/(60\pi), \quad \Psi_6'(t) = \frac{16}{5}\sin^6(\pi t),$$

which helps alleviate boundary singularities.

Lattice Rules. We perform Quasi-Monte Carlo (QMC) integration based on lattice rules [21]. For the rules with $100M \approx 100,000,007$ points, $200M \approx 200,000,033$ points, and $400M \approx 400,000,009$ points, we previously computed the generators (see, e.g., [1]) using the component by component (CBC) algorithm [13,14]. We also use an 8-d projection of a rule with 5^{13} points, originally derived for dimension $d = 15$ (see [7]) via Lattice Builder [10].

For increased accuracy, we apply m^d-copy lattice rules $Q^{(m)}$ of a basic rank-1 lattice rule Q [21] with $m = 2$ or 3, which corresponds to subdividing the $[0, 1]$-range into m equal parts in each coordinate direction and scaling the basic rule to each subcube. An error estimate is also given.

In cases where an analytic result is not available, we compare our results with the pySecDec 1.4.5 [3] package, which uses randomized (shifted) lattice rules [21] obtained by adding a random vector to all integration points, with an error estimate given in terms of the standard deviation.

Double Exponential Integration. The double-exponential method (DE) [22,23] transforms the one-dimensional integral $\int_0^1 f(x)\,dx = \int_{-\infty}^{\infty} f\left(\phi(t)\right)\phi'(t)\,dt$ using

$$x = \phi\left(t\right) = \frac{1}{2}(\tanh\left(\frac{\pi}{2}\sinh\left(\mathrm{t}\right)\right) + 1),$$

with

$$\phi'(t) = \frac{\pi\cosh\left(\mathrm{t}\right)}{4\cosh^2(\frac{\pi}{2}\sinh\left(\mathrm{t}\right))}.$$

The DE technique involves a truncation of the infinite range and application of the trapezoidal rule,

$$I_h^{N_{eval}} = \sum_{k=-N_-}^{N_+} f(\phi(kh))\,\phi'(kh)$$

with mesh size h and $N_{eval} = N_- + N_+ + 1$ function evaluations in each direction.

2.2 Numerical Extrapolation

Nonlinear Extrapolation. We apply the ϵ-algorithm [18,25] for nonlinear extrapolation, with an implementation based on the code of *qext()* from the QuadPack package [15]. The computation produces a triangular table, where the entry (input) sequence of integral approximations $A(\varepsilon)$ for a decreasing geometric sequence of ε forms the first column, and a new lower diagonal of the table is generated when a new input element is available. A table element *"new"* relative to preceding $e_0,\ e_1,\ e_2,\ e_3$ is obtained as

$$\begin{matrix} & e_0 & \\ e_3 & e_1 & \textit{new} \\ & e_2 & \end{matrix} = e_1 + 1/(1/(e_1 - e_3) + 1/(e_2 - e_1) - 1/(e_1 - e_0)),$$

and its *distance* measure from neighboring elements is set to

$$\Delta = |e_2 - e_1| + |e_1 - e_0| + |e_2 - \textit{new}|. \tag{5}$$

The new lower diagonal element with the smallest value of the distance estimate is returned as the extrapolated result.

Linear Extrapolation. Whereas the ϵ-algorithm does not utilize information regarding the underlying asymptotic expansion, linear extrapolation generates a solution to a linear system determined by the orders (e.g., powers in ε) of the leading terms. The expansion (4) is truncated after $2, 3, \ldots, K$ terms in the summation, to form a linear system in the first K variables C_k (with $k = \kappa,\ \kappa+1,\ \ldots,\ \kappa+K-1$, and κ is the index of the first term in (4)). This procedure is a generalized form of Richardson extrapolation [4,12,19].

3 Results

The computations were performed in double precision on an x86_64 host machine with Intel(R) Xeon(R) Gold (dual) 6230 CPU@2.10 GHz, 20 cores (40 HT−hyper-threaded), under GNU/ Linux and using CUDA C, accelerated by an NVIDIA Quadro GV100 (Volta) GPU with compute capability 7.0, 5120 cores and 640 tensor cores, and 32 GB HBM2 global memory, (boost) clock rate of 1627 MHz, and memory clock rate 848 MHz [24]. The GPU performs at (a theoretical peak of) 8.33 TFLOPS for FP64 (double).

The main program ran sequentially on the host, and called the CUDA kernel for the lattice rule computation. The number of threads per block was set to the 1024 maximum for one dimension, except for the M_{44} computations where the number of threads per block was 512. Whereas the maximum number of blocks in a grid is $2^{31} - 1$, it is suggested in [11] to have the number of CUDA blocks close to the number of threads per block. For the results obtained subsequently, the number of blocks was set to 1024 unless otherwise noted. Timing was done using *cudaEvent* timing functions in the main program for each iteration of the while loop over successive values of the ε-parameter.

In Sects. 3.1–3.3 below, we report results for the diagrams of Fig. 1 with massive internal lines where all masses $= 1$, and $s = p^2 = 1$, and for massless lines with all masses $= 0$.

3.1 M_{43}

Table 1 lists the integral approximations $Q^{(m)} = Q^{(m)}(\varepsilon)$ for the massive M_{43} diagram with $\varepsilon = 0$ and $m = 1, 2$ and 3, for lattice rules with $100M$, $200M$, $400M$ and 5^{13} points. Also given are the estimated absolute error $|Est_a|$, and the actual absolute error $|Err_a|$ with respect to the result from pySecDec 1.4.5 [3] considered as the "exact" value. The digits underlined in the results correspond to the accuracy estimated by pySecDec.

The pySecDec code was run on the same system with the GV100 GPU, and further used 64 threads on the CPU host, yielding estimated accuracies for C_0, C_1 and C_2 as listed in Table 4 by performing 3,328,627,552 evaluations in a total execution time of 34,238 s. A pySecDec result obtained with 337,930,896 function evaluations in 3,471 s estimated its error for the integral at $\pm 3.07\,\mathrm{e}{-09}$.

For the lattice rules, it emerges in Table 1 that the computation time increases proportionally with the number of points used, as expected since the work per function evaluation is more or less fixed. The accuracy improves with higher m, and (with some exceptions) for higher numbers of evaluations and the same m, but appears to reach a limiting accuracy level at $m = 3$. Note that an estimated error is not available for $m = 1$.

Table 1. M_{43}, massive lines, $Q^{(m)}(\varepsilon)$ with $\varepsilon = 0$

m	Rule # points	$Q^{(m)}(\varepsilon = 0)$	$\lvert Est_a \rvert$	$\lvert Err_a \rvert$	Time [s]
1	$100M$	−1.0315282429238		1.58 e−04	0.035
	$200M$	−1.0314118459476		4.16 e−05	0.069
	$400M$	−1.0313596638223		1.06 e−05	0.136
	5^{13}	−1.0313649504581		5.33 e−06	0.384
2	$100M$	−1.0313702568810	2.53 e−07	2.76 e−08	7.03
	$200M$	−1.0313702864127	5.92 e−08	1.97 e−09	14.0
	$400M$	−1.0313702892440	5.92 e−09	4.80 e−09	28.1
	5^{13}	−1.0313702831956	2.19 e−09	1.25 e−09	89.4
3	$100M$	−1.0313702845002	2.97 e−08	5.57 e−11	218.2
	$200M$	−1.0313702844244	2.61 e−09	2.01 e−11	452.2
	$400M$	−1.0313702844655	5.86 e−10	2.10 e−11	909.7
pySecDec:		−1.0313702844445			
Accuracy:		±8.10 e−12			

Table 2. Nonlinear extrapolation for M_{43}, massive lines, $Q^{(m)}(\varepsilon)$ with 5^{13}-point rule Q and $m = 2, \varepsilon = 1.2^{-\ell}, \ell = 25, 26, \ldots$

ℓ	$\mathcal{Q}^{(2)}(\varepsilon)$	$\lvert Est_a \rvert$	Time [s]	Extrapolated	Distance Δ
25	−0.949721884198076	1.77 e−09	108.5		
26	−0.962670060436187	1.83 e−09	117.1		
27	−0.973654888967547	1.89 e−09	118.5	−1.0351144345	
28	−0.982946959438472	1.93 e−09	118.8	−1.0339539968	1.16 e−03
29	−0.990787906493011	1.97 e−09	119.4	−1.0312791735	2.67 e−03
30	−0.997390772578446	2.01 e−09	119.4	−1.0313177886	3.86 e−05
31	−1.002941486113812	2.04 e−09	119.4	−1.0313716995	5.39 e−05
32	−1.007600983997484	2.06 e−09	119.9	−1.0313709638	7.36 e−07
33	−1.011507644305459	2.08 e−09	119.9	−1.0313702688	6.95 e−07
34	−1.014779802141804	2.10 e−09	119.9	−1.0313702772	8.39 e−09
35	−1.017518202051021	2.12 e−09	119.8	−1.0313702847	7.57 e−09
36	−1.019808298726949	2.13 e−09	119.8	−1.0313702823	2.45 e−09
	pySecDec Korobov-3:			−1.0313702844	
	pySecDec accuracy:			±8.10 e−12	

Table 3. Nonlinear extrapolation for M_{43}, massive lines, $Q^{(m)}(\varepsilon)$ with 5^{13}-point rule Q and $m = 2, \varepsilon = 1.2^{-\ell}, \ell = 30, 31, \ldots$

ℓ	$\mathcal{Q}^{(2)}(\varepsilon)$	$\lvert Est_a \rvert$	Time [s]	Extrapolated	Distance Δ
30	−0.997390772578446	2.01 e−09	119.4		
31	−1.002941486113812	2.04 e−09	119.4		
32	−1.007600983997484	2.06 e−09	119.9	−1.0319620020	
33	−1.011507644305459	2.08 e−09	119.9	−1.0317802705	1.82 e−04
34	−1.014779802141804	2.10 e−09	119.9	−1.0313644627	4.16 e−04
35	−1.017518202051021	2.12 e−09	119.8	−1.0313669208	2.46 e−06
36	−1.019808298726949	2.13 e−09	119.8	−1.0313703196	3.40 e−06
37	−1.021722358649551	2.14 e−09	119.5	−1.0313703008	1.89 e−08
38	−1.023321342972643	2.15 e−09	119.5	−1.0313702829	1.79 e−08
39	−1.024656569986113	2.15 e−09	119.6	−1.0313702833	4.41 e−10
	pySecDec Korobov-3:			−1.0313702844	
	pySecDec accuracy:			±8.10 e−12	

Table 4. Linear extrapolation for M_{43}, massive lines, $Q^{(m)}(\varepsilon)$ with 5^{13}-point rule Q and $m = 2$, $\varepsilon = 1.2^{-\ell}, \ell = 25, 26, \ldots$

ℓ	$Q^{(2)}(\varepsilon)$	$\lvert Est_a \rvert$	Time [s]	C_0	C_1	C_2
25	−0.9497218841980715	1.77 e−09	126.7			
26	−0.9626700604361829	1.83 e−09	132.3	−1.0274109416	7.411242153	
27	−0.9736548889675424	1.89 e−09	133.6	−1.0312337816	8.213547992	−41.74742269
28	−0.9829469594384685	1.93 e−09	134.1	−1.0313666693	8.259692219	−47.02980432
29	−0.9907879064930063	1.97 e−09	134.8	−1.0313702070	8.261503803	−47.37292710
30	−0.9973907725784428	2.01 e−09	135.1	−1.0313702819	8.261556977	−47.38777961
31	−1.0029414861138075	2.04 e−09	135.5	−1.0313702832	8.261558197	−47.38825191
32	−1.0076009839974789	2.06 e−09	135.5	−1.0313702832	8.261558221	−47.38826436
33	−1.0115076443054538	2.08 e−09	135.6	−1.0313702832	8.261558220	−47.38826400
	pySecDec Korobov-3:		34,238	−1.0313702844	8.261558253	−47.38826443
	pySecDec accuracy:			±8.10 e−12	±5.29 e−11	±2.13 e−10

As another comparison with $\varepsilon = 0$, DE/MPI returned a result of −1.031370286859 with an absolute error of 2.41 e−09 in 4473.0 s using 121 threads.

Tables 2 and 3 show nonlinear extrapolations with the ϵ-algorithm [18,25], for a sequence of integral approximations $A(\varepsilon) = Q^{(2)}(\varepsilon)$, with the 5^{13}-point lattice rule Q, and $\varepsilon = 1.2^{-\ell},\ \ell \geq \ell_0$. Different starting points, $\ell_0 = 25$ and 30, are used in Tables 2 and 3, respectively, resulting in similar accuracy. $|Est_a|$ is the estimated absolute accuracy for the integration result $Q^{(2)}(\varepsilon)$, whereas Δ refers to the distance estimate (5) for the extrapolation, used to assess convergence in the extrapolation table. It should be noted that the error in the input sequence $A(\varepsilon)$ affects the accuracy of the extrapolated results.

A linear extrapolation for M_{43} with massive lines achieves excellent agreement with pySecDec in Table 4, for $Q^{(2)}(\varepsilon)$, the 5^{13}-point lattice rule Q and $\varepsilon = 1.2^{-\ell},\ \ell \geq \ell_0 = 25$. This computation was performed using 64 CUDA blocks, resulting in slightly higher execution times than those of Tables 2 and 3 with 1024 blocks.

Computation for the massless case is in progress and will be presented at the next opportunity.

3.2 M_{44}

For the M_{44} diagram with massive lines, Table 5 displays results $Q^{(m)}(\varepsilon)$, $m = 2, 3$, and $\varepsilon = 0$ for lattice rules Q with $100M$, $400M$ and 5^{13} points. Also given are $|Est_a|$, $|Err_a|$ and the time in seconds Time[s]. The number of blocks was set to 1024, which delivered results in slightly shorter times than those incurred using 512 blocks.

According to the estimated error, the accuracy appears to plateau at $m = 3$. The estimated accuracy of pySecDec is reported at $\pm 4.75\,\mathrm{e}{-11}$.

Table 5. M_{44}, massive lines, $Q^{(m)}(\varepsilon)$ with $\varepsilon = 0$

m	Rule # points	$Q^{(m)}(\varepsilon = 0)$	$\lvert Est_a\rvert$	$\lvert Err_a\rvert$	Time [s]
2	$100M$	−5.19282488199	9.18 e−07	1.33 e−06	7.17
	$400M$	−5.19282330726	8.05 e−07	2.47 e−07	28.7
	5^{13}	−5.19282357097	1.03 e−06	1.72 e−08	90.1
3	$100M$	−5.19282354549	2.12 e−07	8.29 e−09	223.3
	$400M$	−5.19282354800	2.50 e−08	5.77 e−09	935
	5^{13}	−5.19282355028	1.69 e−08	3.50 e−09	2880
pySecDec:		−5.19282355049			
Accuracy:		±4.75 e−11			

Table 6. Linear extrapolation for M_{44}, massive lines, $Q^{(m)}(\varepsilon)$ with 5^{13}-point rule Q and $m = 2$, $\varepsilon = 1.2^{-\ell}, \ell = 20, 21, \ldots$

ℓ	$Q^{(2)}(\varepsilon)$	$\lvert Est_a \rvert$	Time [s]	C_0	C_1	C_2
20	−4.0601894422804277	5.32 e−07	109.7			
21	−4.2238374065452442	5.93 e−07	118.3	−5.0420772279	37.643221095	
22	−4.3671653932011543	6.49 e−07	119.9	−5.1786419142	49.161458176	−240.8626718
23	−4.4917175874154758	7.00 e−07	120.7	−5.1918120637	50.999337564	−325.4145335
24	−4.5992353524890035	7.46 e−07	120.8	−5.1927668799	51.195835153	−340.3714509
25	−4.6915260313418741	7.86 e−07	121.0	−5.1928210212	51.211281283	−342.1053175
26	−4.7703690293820973	8.22 e−07	120.6	−5.1928234777	51.212216470	−342.2508460
27	−4.8374532649071398	8.52 e−07	121.0	−5.1928235682	51.212261249	−342.2601444
28	−4.8943393898615151	8.79 e−07	121.2	−5.1928235709	51.212262985	−342.2606132
	pySecDec Korobov-3:		5573	−5.1928235505	51.212262947	−342.2606447
	pySecDec accuracy:			±4.75 e−11	±5.89 e−10	±7.84 e−09

Table 7. Linear extrapolation for M_{44}, massive lines, $Q^{(m)}(\varepsilon)$ with 5^{13}-point rule Q and $m = 2$, $\varepsilon = 1.2^{-\ell}, \ell = 25, 26, \ldots$

ℓ	$Q^{(2)}(\varepsilon)$	$\lvert Est_a \rvert$	Time[s]	C_0	C_1	C_2
25	−4.6915260313418741	7.86 e−07	121.0			
26	−4.7703690293820973	8.22 e−07	120.6	−5.1645840196	45.127942331	
27	−4.8374532649071398	8.52 e−07	121.0	−5.1917163129	50.822242214	−296.2989083
28	−4.8943393898615151	8.79 e−07	121.2	−5.1927909230	51.195392228	−339.0154278
29	−4.9424405485681984	9.02 e−07	121.4	−5.1928228185	51.211725494	−342.1090275
30	−4.9830163208244791	9.21 e−07	121.5	−5.1928235571	51.212249839	−342.2554878
31	−5.0171755408111256	9.38 e−07	121.4	−5.1928235708	51.212262778	−342.2604972
	pySecDec Korobov-3:		5573	−5.1928235505	51.212262947	−342.2606447
	pySecDec accuracy:			±4.75 e−11	±5.89 e−10	±7.84 e−09

Tables 6 and 7 show linear extrapolation results based on $Q^{(2)}(\varepsilon)$ with the 5^{13}-point lattice rule Q, $\varepsilon = 1.2^{-\ell}, \ell \geq \ell_0 = 20$ or 25, respectively. Linear extrapolation results are further given in Table 8, for $Q^{(3)}(\varepsilon)$, the $100M$-point lattice rule Q, and a sequence of $\varepsilon = 1.15^{-\ell}, \ell \geq \ell_0 = 30$. Note that $\lvert Est_a \rvert$ applies to $Q^{(m)}(\varepsilon)$. Here pySecDec supplies $C_0 \pm 4.75\,\text{e}{-11}$, $C_1 \pm 5.89\,\text{e}{-10}$, and $C_2 \pm 7.84\,\text{e}{-09}$, using 3,328,627,552 integrand evaluations in 5573 s total.

Table 8. Linear extrapolation for M_{44}, massive lines, $Q^{(m)}(\varepsilon)$ with $100M$-point rule Q and $m = 3$, $\varepsilon = 1.15^{-\ell}$, $\ell = 30, 31, \ldots$

ℓ	$Q^{(3)}(\varepsilon)$	$\lvert Est_a \rvert$	Time [s]	C_0	C_1	C_2
30	−4.4913762732977878	1.21 e−07	264.7			
31	−4.5752619125386005	1.30 e−07	277.2	−5.1344995075	42.582328922	
32	−4.6499486706965047	1.38 e−07	278.5	−5.1892894903	50.381973890	−276.2290982
33	−4.7162480921373637	1.46 e−07	279.0	−5.1926552224	51.155824291	−335.1528054
34	−4.7749487570631164	1.53 e−07	278.9	−5.1928169988	51.209310838	−341.7306166
35	−4.8268027974846168	1.60 e−07	279.0	−5.1928233329	51.212138553	−342.2306794
36	−4.8725170366719457	1.66 e−07	279.1	−5.1928235396	51.212258325	−342.2592792
37	−4.9127477956401648	1.71 e−07	279.1	−5.1928235456	51.212262716	−342.2606404
	pySecDec Korobov-3:		5573	−5.1928235505	51.212262947	−342.2606447
	pySecDec accuracy:			±4.75 e−11	±5.89 e−10	±7.84 e−09

Table 9. M_{44}, massless lines, $Q^{(m)}(\varepsilon)$ with $\varepsilon = 0$

m	Rule # points	$Q^{(m)}(\varepsilon = 0)$	$\lvert Est_a \rvert$	$\lvert Err_a \rvert$	Time [s]
2	$100M$	55.586159640	7.92 e−04	9.06 e−04	7.19
	$400M$	55.585470422	1.21 e−04	2.17 e−04	28.7
	5^{13}	55.585200215	2.23 e−04	5.37 e−05	90.6
3	$100M$	55.585320218	8.05 e−05	6.63 e−05	223.3
	$400M$	55.585267930	2.80 e−05	1.40 e−05	930
	5^{13}	55.585249864	2.55 e−05	4.05 e−06	2861
Baikov & Chetyrkin: $= 441\,\zeta_7/8$		55.585253916			

M_{44} with massless lines is covered in [2], and in [17] as *lala*, yielding the coefficients C_0, C_1 and C_2 of the expansion (4) as

$$C_0 = \frac{441}{8}\zeta_7 \approx 55.585253915678496$$

$$C_1 = -\frac{81}{5}\zeta_{5,3} + \frac{18567}{80}\zeta_8 - \frac{1323}{4}\zeta_7 - 135\,\zeta_3\,\zeta_5 \approx -268.8399321507655$$

$$C_2 = \frac{486}{5}\zeta_{5,3} + \frac{4583}{2}\zeta_9 - \frac{55701}{40}\zeta_8 + \frac{1323}{2}\zeta_7 - 81\,\zeta_4\,\zeta_5 - \frac{675}{2}\zeta_3\,\zeta_6 + 810\,\zeta_3\,\zeta_5 - 267\,\zeta_3^3 \approx 1607.707392047691$$

Table 9 lists values of $Q^{(m)}(\varepsilon)$ for $\varepsilon = 0$, $m = 2, 3$, and lattice rules Q with $100M$, $400M$ and 5^{13} points. Table 10 shows an approximation of C_0 by a nonlinear extrapolation with the ϵ-algorithm [18,25] on a sequence of $Q^{(3)}(\varepsilon)$ for $\varepsilon = 1.2^{-\ell}$, $\ell = 25, 26, \ldots$, and $400M$-point lattice rule Q. The results improve significantly on our integral approximation reported in [8], obtained with ParInt using adaptive multivariate integration (which was 55.585150 with absolute error estimate 1.80 e−04).

Table 10. Nonlinear extrapolation for M_{44}, massless lines, $Q^{(m)}(\varepsilon)$ with $400M$-point rule Q and $m = 3$, $\varepsilon = 1.2^{-\ell}$, $\ell = 25, 26, \ldots$

ℓ	$\mathcal{Q}^{(3)}(\varepsilon)$	$\lvert Est_a \rvert$	Time [s]	Extrapolated	Distance Δ
25	56.2134036983660	3.01 e−05	1094		
26	56.0899314726530	2.97 e−05	1095		
27	55.9927679820992	2.94 e−05	1112	55.63392352	
28	55.9157800203784	2.92 e−05	1112	55.62200104	1.19 e−02
29	55.8543905428634	2.90 e−05	1093	55.58542148	3.66 e−02
30	55.8051558873379	2.88 e−05	1109	55.58531681	1.05 e−04
31	55.7654636982214	2.86 e−05	1112	55.58525139	6.54 e−05
32	55.7333158567362	2.85 e−05	1111	55.58525674	5.35 e−06
	Baikov & Chetyrkin: $= 441\,\zeta_7/8$			55.58525392	

Table 11. Linear extrapolation for M_{44}, massless lines, $A(\varepsilon) = \eta(\varepsilon)\, Q^{(m)}(\varepsilon)$ with $400M$-point rule Q and $m = 3$, $\varepsilon = 1.15^{-\ell}$, $\ell = 20, 21, \ldots$.

ℓ	$A(\varepsilon)$	$\lvert Est_a \rvert$	Time[s]	C_0	C_1	C_2
20	43.826387228315269	3.23 e−05	1094			
21	44.930346690376020	3.14 e−05	1114	52.2900764	−138.5213	
22	45.977317510031128	3.08 e−05	1097	55.0254843	−234.7750	842.623
23	46.958581053438841	3.02 e−05	1114	55.4904447	−261.1200	1339.98
24	47.869144536903718	2.98 e−05	1114	55.5728826	−267.9371	1544.79
25	48.706947067994378	2.95 e−05	1100	55.5837878	−269.1405	1597.39
26	49.472182119424062	2.92 e−05	1116	55.5851203	−269.3314	1608.66
27	50.166723337484783	2.90 e−05	1117	55.5852549	−269.3558	1610.52
	Ruijl, Ueda & Vermaseren:			55.5852539	−268.8399	1607.71

Linear extrapolation results for M_{44} with massless lines are given in Table 11 for $A(\varepsilon) = \eta(\varepsilon)\, Q^{(3)}(\varepsilon)$, $\varepsilon = 1.15^{-\ell}$ with $\ell = 25, 26, \ldots$, and $400M$-point lattice rule Q. The normalization factor $\eta(\varepsilon)$ of (3) is used so the coefficients C_k of the expansion (4) would correspond to the published expressions in [2,17]. Without the normalization factor, C_0 remains the same but the coefficients differ from C_1 on, as can be seen by multiplying the series expansions of $\eta(\varepsilon)$ and $Q^{(m)}(\varepsilon)$. Good agreement was achieved for C_0, while C_1 and C_2 are in the range of the analytic results in [17]. In view of the integrand structure, the numerical evaluation for these massless diagrams is more difficult than for the corresponding diagrams with massive lines, which also emerges from the larger estimated errors for the massless cases.

We further obtained a corresponding expansion with DE as

$$\eta(\varepsilon)\, Q^{(3)}(\varepsilon) \sim 55.5852571 - 269.3577\,\varepsilon + 1610.69\,\varepsilon^2 + \ldots$$

Table 12. M_{45}, massive lines, $Q^{(m)}(\varepsilon)$ with $\varepsilon = 0$

m	Rule # points	$Q^{(m)}(\varepsilon = 0)$	$\lvert Est_a \rvert$	$\lvert Err_a \rvert$	Time [s]
2	$100M$	−4.69481370796	2.00e−06	2.73 e−06	6.97
	$400M$	−4.69481095482	1.78 e−06	2.42 e−08	28.07
	5^{13}	−4.69481104268	2.52 e−07	6.37 e−08	90.0
3	$100M$	−4.69481102162	6.70 e−07	4.26 e−08	217.6
	$400M$	−4.69481098486	1.56 e−07	5.87 e−09	909
	5^{13}	−4.69481098180	2.97 e−08	2.80 e−09	2791
pySecDec:		−4.69481098172			
Accuracy:		±4.14 e−11			

3.3 M_{45}

Approximations with $\varepsilon = 0$ for the constant coefficient C_0 in the expansion (4) of the massive M_{45} diagram are displayed in Table 12. Table 13 gives a nonlinear extrapolation that derives C_0. In Table 12, $m = 2$ and 3 are used for the lattice rules with $100M$,

Table 13. Nonlinear extrapolation for M_{45}, massive lines, $Q^{(m)}(\varepsilon)$ with $100M$-point rule Q and $m = 3$, $\varepsilon = 1.15^{-\ell}$, $\ell = 35, 36, \ldots$

ℓ	$\mathcal{Q}^{(3)}(\varepsilon)$	$\lvert Est_a \rvert$	Time [s]	Extrapolated	Distance Δ
35	−4.3616359607327739	5.10 e−07	258.0		
36	−4.4032370844791213	5.28 e−07	270.5		
37	−4.4398507092408011	5.45 e−07	272.2	−4.7086342263	
38	−4.4720250633753444	5.60 e−07	257.4	−4.7052140602	3.42 e−03
39	−4.5002602654438046	5.73 e−07	270.6	−4.6944830070	1.07 e−02
40	−4.5250094047667391	5.85 e−07	273.3	−4.6945961688	1.13 e−04
41	−4.5466805811799018	5.95 e−07	258.0	−4.6948163050	2.20 e−04
42	−4.5656395641499143	6.05 e−07	275.5	−4.6948140308	2.27 e−06
43	−4.5822128112051645	6.13 e−07	272.1	−4.6948109712	3.06 e−06
44	−4.5966906515695101	6.20 e−07	273.0	−4.6948109828	1.15 e−08
		pySecDec:		−4.6948109817	
		Accuracy:		±4.14 e−11	

Table 14. Linear extrapolation for M_{45}, massive lines, $Q^{(m)}(\varepsilon)$ with $400M$-point rule Q and $m = 3, \varepsilon = 1.15^{-\ell}, \ell = 35, 36, \ldots$

ℓ	$Q^{(3)}(\varepsilon)$	$\lvert Est_a \rvert$	Time [s]	C_0	C_1	C_2
35	−4.3616359324804321	1.22 e−07	1217			
36	−4.4032370552356630	1.26 e−07	1238	−4.6805778736	42.475259950	
37	−4.4398506791081358	1.29 e−07	1293	−4.6943713695	46.424715423	−281.3332185
38	−4.4720250324481521	1.32 e−07	1221	−4.6948003601	46.623102971	−311.7166389
39	−4.5002602338092768	1.35 e−07	1240	−4.6948107758	46.630029426	−313.4299532
40	−4.5250093725044307	1.38 e−07	1242	−4.6948109814	46.630213989	−313.4956014
41	−4.5466805483618833	1.40 e−07	1220	−4.6948109849	46.630218037	−313.4975457
	pySecDec Korobov-3:		7718	−4.6948109817	46.630217902	−313.4975487
	pySecDec accuracy:			±4.14 e−11	±6.27 e−10	±5.19 e−09

Table 15. M_{45}, massless lines, $Q^{(m)}(\varepsilon)$ with $\varepsilon = 0$

m	Rule # points	$Q^{(m)}(\varepsilon = 0)$	$\lvert Est_a \rvert$	$\lvert Err_a \rvert$	Time [s]
2	$100M$	52.0184973113	7.91 e−04	6.29 e−04	6.96
	$400M$	52.0178434626	4.99 e−05	2.53 e−05	28.1
	5^{13}	52.0178700693	6.73 e−05	1.33 e−06	88.3
3	$100M$	52.0179169807	5.87 e−05	4.82 e−05	217.6
	$400M$	52.0178734009	8.27 e−06	4.66 e−06	907
	5^{13}	52.0178680744	6.66 e−06	6.69 e−07	2791
Exact: $= 36\,\zeta_3^2$		52.0178687436			

$400M$ and 5^{13} points. Table 13 applies $m = 3$, a sequence of the regularization parameter $\varepsilon = 1.15^{-\ell}, \ell = 35, 36, \ldots$, and lattice rule Q with $100M$ points. Both tables agree with the pySecDec value, which is supplied within $\pm 4.14\,\mathrm{e}{-11}$ accuracy.

Excellent agreement with the pySecDec results is also observed in Table 14, which presents a linear extrapolation to approximate C_0, C_1 and C_2 in the series expansion for the massive M_{45} diagram.

Subsequently, we address the M_{45} diagram with massless lines. The expansion is given in [2], and as the *nono* diagram in [17], with coefficients

$$
\begin{aligned}
C_0 &= 36\,\zeta_3^2 \approx 52.01786874361083 \\
C_1 &= 108\,\zeta_3\,\zeta_4 - 378\,\zeta_7 \approx -240.64650248085806 \\
C_2 &= \frac{3024}{5}\zeta_{5,3} - \frac{26901}{10}\zeta_8 + 2844\,\zeta_3\,\zeta_5 + 432\,\zeta_3^2 \\
&\approx 1471.472731850490
\end{aligned}
$$

Table 16. Nonlinear extrapolation for M_{45}, massless lines, $A(\varepsilon) = \eta(\varepsilon)\, Q^{(m)}(\varepsilon)$ with 5^{13}-point rule Q and $m = 2$, $\varepsilon = 1.2^{-\ell}$, $\ell = 25, 26, \ldots$

ℓ	$A(\varepsilon)$	$\lvert Est_a \rvert$	Time [s]	Extrapolated	Distance Δ
25	49.651577093927614	7.28 e−05	125.7		
26	50.025083811898384	7.18 e−05	131.5		
27	50.342512562887130	7.10 e−05	133.1	52.13931472	
28	50.611407572808432	7.04 e−05	134.2	52.10118612	3.81 e−02
29	50.838572116548526	6.98 e−05	134.9	52.01527075	8.59 e−02
30	51.030048013982416	6.94 e−05	134.5	52.01636014	1.09 e−03
31	51.191137039502031	6.90 e−05	134.7	52.01791539	1.56 e−03
32	51.326447659351238	6.87 e−05	135.2	52.01789174	2.36 e−05
33	51.439955375016694	6.84 e−05	135.4	52.01786959	2.21 e−05
34	51.535068675662060	6.82 e−05	135.7	52.01786990	3.09 e−07
		Exact: $= 36\,\zeta_3^2$		52.01786874	

Table 17. Linear extrapolation for M_{45}, massless lines, $A(\varepsilon) = \eta(\varepsilon)\, Q^{(m)}(\varepsilon)$ with 5^{13}-point rule Q and $m = 2$, $\varepsilon = 1.2^{-\ell}$, $\ell = 20, 21, \ldots$.

ℓ	$A(\varepsilon)$	$\lvert Est_a \rvert$	Time [s]	C_0	C_1	C_2
20	46.647094292297766	8.22 e−05	125.9			
21	47.427685634869917	7.94 e−05	132.4	51.3306423	−179.555992	
22	48.110521177339116	7.72 e−05	134.1	51.9657365	−233.121558	1120.1320
23	48.703053492158347	7.54 e−05	134.4	52.0143244	−239.901941	1432.0643
24	49.213792353197562	7.40 e−05	134.8	52.0176864	−240.593844	1484.7303
25	49.651577093927614	7.28 e−05	125.7	52.0178620	−240.643944	1490.3534
26	50.025083811898384	7.18 e−05	131.5	52.0178698	−240.646890	1490.8126
	Ruijl, Ueda & Vermaseren:			52.0178687	−240.646503	1471.4727

Table 15 provides results for C_0 computed with $\varepsilon = 0$. $Q^{(m)}(\varepsilon = 0)$ is listed for $m = 2, 3$ and lattice rules Q with $100M, 400M$ and 5^{13} points, together with their estimated absolute error $|Est_a|$, actual absolute error $|Err_a|$, and execution time in seconds. $|Est_a|$ is a slight upper bound of $|Err_a|$ in these cases.

Tables 14, 16, and 17 were computed using 64 CUDA blocks. Table 16 gives a nonlinear extrapolation for the massless M_{45} diagram, using the 5^{13}-point lattice rule Q, $m = 2$, and the regularization sequence $\varepsilon = 1.2^{-\ell}$ for $\ell = 25, 26, \ldots$. The extrapolated results exhibit good convergence to within a relative accuracy of 2.2×10^{-8}.

Linear extrapolation results are shown in Table 17 for $A(\varepsilon) = \eta(\varepsilon)\, Q^{(2)}(\varepsilon)$ where $\eta(\varepsilon)$ is the normalization factor (3) from Baikov and Chetyrkin [2], with $\varepsilon = 1.2^{-\ell}$, $\ell = 20, 21, \ldots$ and lattice rule Q with 5^{13} points.

4 Conclusions

We handled three self-energy diagrams with four loops and 9 internal lines, for the computation of the associated loop integrals and leading-order coefficients in their asymptotic expansions with respect to the regularization parameter.

We obtained results by various solution methods, using composite lattice rule approximations and double exponential integration, as well as linear and nonlinear extrapolation on the sequences of integral approximations for different sequences of the regularization parameter or with different starting points.

For the massless diagrams, comparisons could be made with expansions given in the literature. We improved on our previously published results, and hope to make further improvements in future work. New numerical results are provided for the massive versions of the diagrams, where all masses $= 1$, and $s = p^2 = 1$. Good agreement is found with computations by the pySecDec package.

For these diagrams, it has thus been shown that a numerical treatment is possible, only relying on a definition of the integrand function in the loop integral.

Acknowledgments. We acknowledge the support of the Grant-in-Aid for Scientific Research (JP20K03941) from JSPS KAKENHI, and the National Science Foundation Award Number 1126438 that funded work on multivariate integration. The authors of this paper sincerely appreciate the reviewers' comments, which have helped us improve the paper considerably.

Appendix

A C and D Functions

In the code of the C and D functions below for the diagrams of Fig. 1, we denote $x_{i_1 \ldots i_k} = \sum_{j=i_1}^{i_k} x_j$.

A.1 M_{43} Massive Diagram

```
C = (x129*x4 + x1234*x5)*(x6*x7 + x67*x8) +
    x39*(x2*x47*x56 + x1*x6*x7 + x4*x5*x8 +
    x4567*(x1*x2 + x12*x8)) + x3*(x18*x47 + x1478*x56)*x9
    + (x1*x45 + x4*x5)*(x3*x6 + x2*x67 + x7*x9);

D = x12356789*(x29*x3*x5*x7 + (x1259*x3 + x2*x5)*x6*x7 +
    x3*x67*(x18*x29 + x15*x8) +
    (x15*x4 + x1*x5)*(x37*x6 + x3679*x8) + x12*x56*x7*x9 +
    x67*(x1*x28 + x2*x8)*x9 + x1568*x4*(x29*x37 + x2*x9) +
    x18*x5*(x2*x3679 + x3*x9) + x4*x6*(x158*x2 + x8*x9)) +
    x4*x4*(x1259*x37*x6 + x29*x367*x8 + x1568*x2*x9 +
    x15*(x28*x367 + x37*x9 + x8*x9));
```

A.2 M_{44}

For M_{44} (and M_{45}) we use the expression for W from [8], and further

$$U = C$$
$$V = M^2 - \frac{W}{U}$$
$$D = UV = UM^2 - W$$
$$M^2 = \sum_{r=1}^{N} m_r^2 x_r$$

Thus $M^2 = 1$ for massive diagrams with all masses $m_r = 1$, and $D = U - W = C - W$. For massless diagrams, $M^2 = 0$, so $D = -W$.

```
C =  x7*x8*x9*x123456 + x7*x8*x1256*x34 + x7*x9*x126*x345
     + x8*x9*x16*x2345 + x7*x5*x126*x34 +x8*x16*x25*x34 +
     x9*x2*x16*x345 + x2*x5*x16*x34;

W = x7*x8*x9*x123*x456 + x7*x8*(x12*x3*x456 + x123*x4*x56)
    + x7*x9*(x12*x3*x456 + x123*x45*x6)
    + x8*x9*(x1*x23*x456 + x123*x45*x6) + x7*x5*(x12*x3*x46
    + x123*x4*x6) + x8*(x12*x34*x5*x6 + x1*x2*x34*x56 +
    x16*x3*x4*x25) + x9*x2*(x13*x45*x6 + x1*x3*x456) +
    x2*x5*(x1*x3*x46 + x13*x4*x6);

D = C - W; // massive diagram
// D = -W; // massless diagram
```

A.3 M_{45}

```
C =  x7*x8*x9*x123456 + x7*x8*(x12*x34 + x34*x56) +
     x7*x9*x126*x345 + x8*x9*x156*x234 + x7*x5*x126*x34
     + x8*x2*x156*x34 + x9*(x16*x2*x345 + x126*x34*x5)
     + x2*x5*x16*x34;

W = x7*x8*x9*x123*x456 + x7*x8*(x12*x4*x356 + x124*x3*x56)
    + x7*x9*(x12*x3*x456 + x123*x45*x6) + x8*x9*(x1*x23*x456
    + x123*x4*x56) + x7*x5*(x12*x3*x46 + x123*x4*x6) +
    x8*x2*(x13*x4*x56 + x1*x3*x456) + x9*(x1*x2*x36*x45 +
    x12*x36*x4*x5 + x14*x25*x3*x6) + x2*x5*(x1*x3*x46
    + x13*x4*x6);

D = C - W; // massive diagram
// D = -W; // massless diagram
```

References

1. Almulihi, A., de Doncker, E.: Accelerating high-dimensional integration using lattice rules on GPUs. In: Proceedings of the 2017 International Conference on Computational Science and Computational Intelligence, CSCI 2017. CPS IEEE (2017)
2. Baikov, B.A., Chetyrkin, K.G.: Four loop massless propagators: an algebraic evaluation of all master integrals. Nucl. Phys. B **837**, 186–220 (2010)
3. Borowka, S., Heinrich, G., Jahn, S., Jones, S.P., Kerner, M., Schlenk, J.: A GPU compatible quasi-Monte Carlo integrator interfaced to pySecDec. Comput. Phys. Commun. **240**, 120–137 (2019). Preprint: arXiv:1811.11720v1 [hep-ph]. https://arxiv.org/abs/1811.11720. https://doi.org/10.1016/j.cpc.2019.02.015
4. Brezinski, C.: A general extrapolation algorithm. Numer. Math. **35**, 175–187 (1980)
5. de Doncker, E., Shimizu, Y., Fujimoto, J., Yuasa, F.: Computation of loop integrals using extrapolation. Comput. Phys. Commun. **159**, 145–156 (2004)
6. de Doncker, E., Yuasa, F.: Self-energy Feynman diagrams with four loops and 11 internal lines. In: Gervasi, O., et al. (eds.) ICCSA 2021. LNCS, vol. 12953, pp. 160–175. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86976-2_11
7. de Doncker, E., Yuasa, F., Almulihi, A., Nakasato, N., Daisaka, H., Ishikawa, T.: Numerical multi-loop integration on heterogeneous many-core processors. J. Phys. Conf. Ser. (JPCS) **1525**(012002) (2019). https://doi.org/10.1088/1742-6596/1525/1/012002
8. de Doncker, E., Yuasa, F., Kato, K., Ishikawa, T., Kapenga, J., Olagbemi, O.: Regularization with numerical extrapolation for finite and UV-divergent multi-loop integrals. Comput. Phys. Commun. **224**, 164–185 (2018). https://doi.org/10.1016/j.cpc.2017.11.001
9. de Doncker, E., Yuasa, F., Olagbemi, O., Ishikawa, T.: Large scale automatic computations for Feynman diagrams with up to five loops. In: Gervasi, O., et al. (eds.) ICCSA 2020. LNCS, vol. 12253, pp. 145–162. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58814-4_11
10. L' Equyer, P., Munger, D.: Algorithm 958: lattice builder: a general software tool for constructing rank-1 lattice rules. ACM Trans. Math. Softw. **42**(2), 15:1–15:30 (2016)
11. Lyman, J.: Optimizing CUDA for GPU architecture – choosing the right dimensions. http://selkie.macalester.edu/csinparallel/modules/CUDAArchitecture/build/html/index.html
12. Lyness, J.N.: Applications of extrapolation techniques to multidimensional quadrature of some integrand functions with a singularity. J. Comput. Phys. **20**, 346–364 (1976)
13. Nuyens, D., Cools, R.: Fast algorithms for component-by-component construction of rank-1 lattice rules in shift-invariant reproducing kernel Hilbert spaces. Math. Comp. **75**, 903–920 (2006)
14. Nuyens, D., Cools, R.: Fast component-by-component construction of rank-1 lattice rules with a non-prime number of points. J. Complex. **22**, 4–28 (2006)
15. Piessens, R., de Doncker, E., Überhuber, C.W., Kahaner, D.K.: QUADPACK, A Subroutine Package for Automatic Integration, Springer Series in Computational Mathematics, vol. 1. Springer-Verlag, Heidelberg (1983). https://doi.org/10.1007/978-3-642-61786-7
16. Pittau, R., Webber, B.: Direct numerical evaluation of multi-loop integrals without contour deformation. Eur. Phys. J. C **82**(1), 1–22 (2022). https://doi.org/10.1140/epjc/s10052-022-10008-6
17. Ruijl, B., Ueda, T., Vermaseren, J.A.M.: Forcer, a Form program for the parametric reduction of four-loop massless propagator diagrams. Comput. Phys. Commun. **253**, 107198 (2020)
18. Shanks, D.: Non-linear transformations of divergent and slowly convergent sequences. J. Math. Phys. **34**, 1–42 (1955)
19. Sidi, A.: Practical Extrapolation Methods - Theory and Applications. Cambridge University Press (2003). ISBN 0-521-66159-5

20. Sidi, A.: Extension of a class of periodizing transformations for numerical integration. Math. Comp. **75**(253), 327–343 (2005)
21. Sloan, I., Joe, S.: Lattice Methods for Multiple Integration. Oxford University Press (1994)
22. Sugihara, M.: Optimality of the double exponential formula - functional analysis approach. Numer. Math. **75**(3), 379–395 (1997)
23. Takahasi, H., Mori, M.: Double exponential formulas for numerical integration. Publ. Res. Inst. Math. Sci. **9**(3), 721–741 (1974)
24. Techpowerup. https://www.techpowerup.com/gpu-specs/quadro-gv100.c3066
25. Wynn, P.: On a device for computing the $e_m(s_n)$ transformation. Math. Tables Aids Comput. **10**, 91–96 (1956)
26. Yuasa, F., et al.: Numerical computation of two-loop box diagrams with masses. J. Comput. Phys. Commun. **183**, 2136–2144 (2012)