
Multipoint Feynman diagrams at the one loop level

Ph.D. Thesis

Krzysztof Kajda

Department of Field Theory and Particle Physics
University of Silesia - Institute of Physics
Katowice, Poland



3rd september 2009

Supervisor:

Dr. hab. Janusz Gluza
Department of Field Theory and Particle Physics
University of Silesia, Katowice, Poland

Acknowledgements

My special thanks go to my supervisor
dr. hab. Janusz Gluza for his helpful
suggestions and important advice during
the course of this work.

Contents

1	Introduction	7
2	Evaluating Feynman integrals	9
2.1	Feynman parametrisation	9
2.2	Mellin-Barnes representation: theoretical background	13
2.3	Numerical calculations in the Euclidean region	19
3	Construction of Mellin-Barnes representations by AMBRE package	23
3.1	Loop by Loop algorithm	23
3.2	Remarks on derivation of Mellin-Barnes representations	27
3.2.1	Order of integration in the loop by loop method	27
3.2.2	Importance of F polynomial in derivation of representations . . .	32
3.2.3	Using Barnes lemmas	34
3.3	One-loop integrals with tensor numerator	39
3.4	Cross-checks in Euclidean region with CSectors	42
3.5	Applications and perspectives of AMBRE	45
4	Tensor reduction of five and six point one loop integrals	47
4.1	Passarino-Veltman reduction	47
4.2	Gram determinants and algebra of signed minors	49
4.3	Tensor integrals and shifted space-time dimensions	51
4.4	Reduction of pentagons up to rank two	51
4.5	Reduction of six point functions	57
4.6	Numerical results and discussion	59
5	Applications: five point integrals in QED	65
5.1	Infrared divergences connected with $e^+e^- \rightarrow \mu^+\mu^-\gamma$	67
5.2	Results and numerical cross-checks for virtual contributions	70
6	Summary and conclusions	81
A	Gamma function	83
B	Software	85
B.1	AMBRE and generation of Mellin-Barnes representations	85
B.2	Reduction of six and five point integrals in hexagon	89
B.3	Sector decomposition with CSectors	92

Bibliography

95

1 Introduction

Research is an organized method
for keeping you reasonably
dissatisfied with what you have.

(Charles F. Kettering)

High energy physics, a branch of science that studies elementary particles of matter and the interactions among them developed over the span of XXth century. Many generations of researchers examined basic properties of subatomic particles [1]. Notably, during the 70's the Standard Model of particle physics [2, 3] evolved as a modern classification of elementary particles and description of the three of four basic interactions. In order to increase our knowledge about elementary particles many international accelerator collaborations exist, like: CERN, DESY(Hamburg), Fermilab, KEK, SLAC. High energy physics experiments may confirm Standard Model, just like the recent discovery of Ω_b^- baryon in DZero experiment at the Fermilab [4] and future search for Higgs particle in LHC, or can hopefully give a clue about "new physics" beyond the Standard Model. All these experiments, due to accuracy they achieve, often require rather painful and difficult theoretical calculations at certain leading order of corrections. The current need of precision goes at large extent beyond the so called tree order, meaning that loop effects must be taken into account.

Nowadays there are numerous methods of involving loop integrals [5]. Some of them became a standard, let us mention here the **FF** [6] or **LoopTools** [7] packages. Lately several new tools appeared, like: **CutTools** [8], which uses Ossola, Papadopoulos and Pittau (OPP) [9] method, or **BlackHat** [10] using unitary cut method and on-shell recursion [11] to construct one-loop amplitudes. Many of them are still under development. All available loop methods are quite advanced and in this thesis we focus only on a few which allow us to examine loop integrals.

The title of this thesis refers to one-loop integrals and although we focus mostly on such cases, multi-loop calculus is also possible within part of the methods and developed tools we discuss. Firstly we focus on Mellin-Barnes method [5] which is described in chapter two. Because it is possible to obtain analytic results using it, M-B can be used to analyse infrared (IR) divergences that may appear in loop integrals. Quite recently we have developed a special tool, **AMBRE** [12] which produces M-B representations of Feynman diagrams in an automatic way. It is the first public tool already used in a research work by several groups. The detailed discussion of **AMBRE** is in the chapter three. Let us note that it is not always easy to solve derived M-B integrals into the final

analytical form. For such cases numerical integration can be made for purely numerical tests. However, due to an oscillatory behaviour of M-B integrals [13] numerical results need non-physical phase space points to be stable numerically. There is no way to calculate physical processes numerically using directly Mellin-Barnes representations of Feynman diagrams.

During practical calculations of virtual corrections one needs a method to compute tensor loop integrals. Often some reduction scheme is applied e.g. the well known Passarino-Veltman [14] for one loop diagrams. Problem with numerical instability related to the so called Gram determinants might appear. In chapter four we present a cure for the problem of Grams, a reduction scheme worked out by us in [15, 16] for six and five point integrals based on algebra of "signed minors". The advance over the previous schemes based on this algebra is that these determinants were cancelled. The scheme itself is a continuous work based on [17], where Grams were still present. Here we also show a special software that uses this scheme, **hexagon** [16]. Although its usage within Monte Carlo event generator would not be a good idea (the program was written in **MATHEMATICA** [18], more suitable for analytic manipulations), it is still a valuable tool for performing many cross-checks e.g. IR tests described in the last chapter. We have used it to perform checks and tests to the well known **LoopTools** program which is based on the early **FF** package. Quite recently the tensor structure for a five point diagrams has been also added to **LoopTools** [19] using work of [20, 21].

The last chapter covers results which are not public yet. Somehow it merges together all the previous chapters. Here we have calculated loop corrections (four and five point diagrams) of the QED $e^+e^- \rightarrow \mu^+\mu^-\gamma$ process. As a primary tool for numerical calculations we used **LoopTools**. The same calculations can be repeated using "signed minors" reduction scheme. Here instead of **hexagon** we would use our reduction scheme encoded into **Fortran**. This program is still under development and so far we used it to test and check single integrals only.

Apart from purely numerical calculations, we have also used tools and techniques to calculate infrared divergences and do proper checks and verifications. Here we used **AMBRE** to obtain IR analytic results and **CSectors** [22] which uses sector decomposition algorithm [23, 24, 25] to perform proper numerical cross-checks.

This thesis ends with two appendices. In the Appendix A a short description of the gamma function is given, while in the Appendix B the software (**AMBRE**, **hexagon** and **CSectors** packages) are presented.

2 Evaluating Feynman integrals

Modern experiments in particle physics need precise theoretical predictions, which are related to the higher order Quantum Field Theory corrections going beyond the so called tree level. Loop corrections can be treated in different ways. First we describe Feynman parametrisation, which is important for the next loop method described and explored in detail i.e. Mellin-Barnes representations. Finally, we focus on numerical evaluation techniques in the Euclidean region, in particular we give a description of the sector decomposition method.

2.1 Feynman parametrisation

As an introductory example we begin with an integration of the following simple two dimensional parametric integral:

$$\begin{aligned} \int_0^1 dx_1 \int_0^1 dx_2 \frac{\delta(1-x_1-x_2)}{P_1x_1 + P_2x_2} &= \int_0^1 dx_1 \frac{1}{P_1x_1 + P_2(1-x_1)} \\ &= \frac{1}{P_1 - P_2} \int_{P_1}^{P_2} dt \frac{1}{t^2} \\ &= \frac{1}{P_1P_2}, \end{aligned} \tag{2.1.1}$$

where Dirac delta properties and integration by substitution were used. In this unsophisticated example we see that product in the denominator can be replaced by its sum together with integration over additional parameters. This is the so called Feynman parametrisation in its simplest two dimensional form¹. Eq.2.1.1 can be generalized in a straightforward way:

$$\frac{1}{P_1 \dots P_n} = \Gamma(n) \int_0^1 \dots \int_0^1 dx_1 \dots dx_n \frac{\delta(1-x_1-\dots-x_n)}{(P_1x_1 + \dots + P_nx_n)^n}. \tag{2.1.2}$$

Here factorial $(n-1)!$ was replaced by the gamma function $\Gamma(n)$ (see: Eq.A.0.3). It is also easy to check above equation for $n > 2$. The only precaution which must be

¹ Feynman parametrisation is closely related to the so called alpha parametrisation and the so called Schwinger trick [5]:

$$\frac{1}{P_1 \dots P_n} = \int_0^\infty \dots \int_0^\infty d\alpha_1 \dots d\alpha_n e^{-P_1\alpha_1} \dots e^{-P_n\alpha_n}.$$

overtaken is setting an appropriate upper limit for the integral after "cancelling" Dirac delta. For example:

$$\begin{aligned} & \int_0^1 dx_1 \int_0^1 dx_2 \int_0^1 dx_3 \delta(1 - x_1 - x_2 - x_3) f(x_1, x_2, x_3) \\ &= \int_0^1 dx_1 \int_0^{1-x_1} dx_2 f(x_1, x_2, 1 - x_1 - x_2), \end{aligned} \quad (2.1.3)$$

where f is some general function. Note that in case of Eq.2.1.1 it was not significant.

It is also worth observing how Feynman parametrisation looks like in case of general powers of terms (propagators) in a denominator:

$$\frac{1}{\prod_{i=1}^n P_i^{\nu_i}} = \frac{\Gamma(N_\nu)}{\Gamma(\nu_1) \dots \Gamma(\nu_n)} \int_0^1 \prod_{j=1}^n dx_j x_j^{\nu_j-1} \frac{\delta(1 - x_1 - \dots - x_n)}{(P_1 x_1 + \dots + P_n x_n)^{N_\nu}}, \quad (2.1.4)$$

with

$$N_\nu = \sum_{i=1}^n \nu_i. \quad (2.1.5)$$

We would like to use Feynman parametrisation in case of loop integrals which in turn can be defined in d dimensions, in the sense of dimensional regularisation, in the following way:

$$G(T(k)) = \frac{1}{(i\pi^{d/2})^L} \int \frac{d^d k_1 \dots d^d k_L T(k)}{(q_1^2 - m_1^2 + i0)^{\nu_1} \dots (q_n^2 - m_n^2 + i0)^{\nu_n}}, \quad (2.1.6)$$

where $T(k)$ is a tensor numerator of some rank. In particular, in case of a scalar integral $T(k) = 1$. The momentum q^2 is a composition of internal k and external p momenta. In this notation single propagator is of the form:

$$P_i = q_i^2 - m_i^2 + i0. \quad (2.1.7)$$

For simplicity we start with a scalar case. It will be expanded to include the case of general m -rank numerator afterwards. Let us insert above definition into Eq.2.1.4. One may see that:

$$\sum_{i=1}^n P_i x_i = \sum_{i=1}^n (q_i^2 - m_i^2 + i0) x_i = \sum_{i,j=1}^L k_i^T M_{ij} k_j - 2 \sum_{j=1}^L k_j^T Q_j + J, \quad (2.1.8)$$

where M is a $L \times L$ matrix containing Feynman parameters, Q is an L -dimensional vector composed of external momenta and Feynman parameters, and J contains kinematic invariants and Feynman parameters. At this point integral $G(1)$ is of the form:

$$\begin{aligned} G(1) &= \frac{\Gamma(N_\nu)}{\Gamma(\nu_1) \dots \Gamma(\nu_n)} \\ &\times \int_0^1 \prod_{j=1}^n dx_j x_j^{\nu_j-1} \delta\left(1 - \sum_{i=1}^n x_i\right) \int \frac{d^d k_1 \dots d^d k_L}{[k^T M k - 2k^T Q + J]^{N_\nu}}. \end{aligned} \quad (2.1.9)$$

After eliminating linear terms in Eq.2.1.8 by performing shift:

$$k_l = \bar{k}_l + \sum_{i=1}^L M_{li}^{-1} Q_i, \quad (2.1.10)$$

it is relatively easy to perform momentum integration resulting in the following parametrisation:

$$G(1) = (-1)^{N_\nu} \frac{\Gamma(N_\nu - \frac{d}{2}L)}{\Gamma(\nu_1) \dots \Gamma(\nu_n)} \int_0^1 \prod_{j=1}^n dx_j x_j^{\nu_j-1} \delta \left(1 - \sum_{i=1}^n x_i \right) \frac{U^{N_\nu-d(L+1)/2}}{F^{N_\nu-dL/2}}, \quad (2.1.11)$$

where

$$\begin{aligned} U &= \det(M), \\ F &= -\det(M)J + Q\tilde{M}Q. \end{aligned} \quad (2.1.12)$$

Matrix $\tilde{M}$ is defined as $\tilde{M} = \det(M)M^{-1}$. According to the graph theory the polynomials U and F can also be constructed from the topology of a given Feynman diagram as follows [26, 27, 28]:

- U - polynomial

$$U = \sum_{T \in T_1} \prod_{l \notin T} x_l \quad (2.1.13)$$

The sum runs over 1-trees of the given graph, i.e. maximal connected sub-graphs without loops,

- F - polynomial

$$F = \sum_{T \in T_2} \prod_{l \notin T} x_l (-s_T)^2 \quad (2.1.14)$$

Here the sum is over 2-trees, i.e. sub-graphs that do not involve loops and consists of two connectivity components. s_T is the sum of the external momenta that flow into the connectivity components of the 2-tree T .

In the case of four point massless on-shell Feynman diagram, the construction of U and F functions using above rules was demonstrated on Fig.2.1 and Fig.2.2, resulting U and F to be as:

$$\begin{aligned} U &= x_1 + x_2 + x_3 + x_4, \\ F &= -sx_1x_3 - tx_2x_4. \end{aligned} \quad (2.1.15)$$

It is interesting to observe that all this mathematical apparatus, together with interpreting these results from the graph theoretical point of view, is the same as that used in the

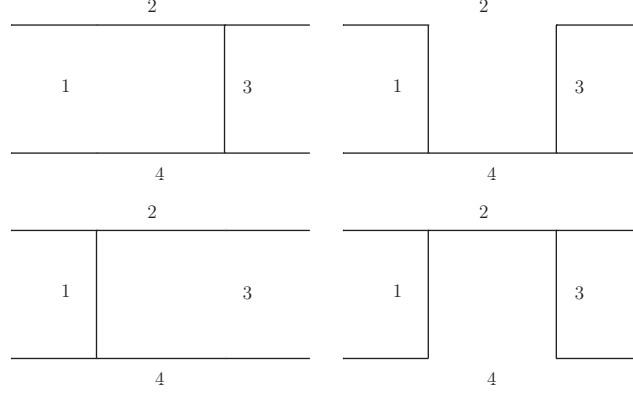


Figure 2.1: 1-trees contributing to the U polynomial

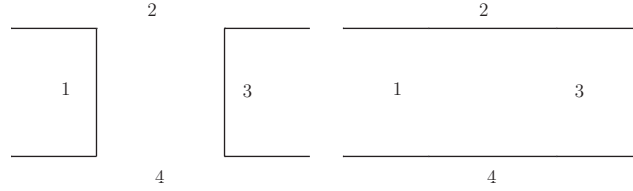


Figure 2.2: 2-trees contributing to the F polynomial

problem of the solution of Kirchhoff's laws for electrical circuits. Feynman parameters x_i play the role of ohmic resistance and the U polynomial is a Kirchhoff result [29].

The Eq.2.1.11 can be generalized to L -loop, m -rank integral resulting in (see e.g. [25]):

$$G(T(k)) = \frac{(-1)^{N_\nu}}{\Gamma(\nu_1) \dots \Gamma(\nu_n)} \int_0^1 \prod_{j=1}^n dx_j x_j^{\nu_j-1} \delta \left(1 - \sum_{i=1}^n x_i \right) \\ \times \sum_{r \leq m} \frac{\Gamma \left(N_\nu - \frac{d}{2}L - \frac{r}{2} \right)}{(-2)^{\frac{r}{2}}} \frac{U^{N_\nu - \frac{d}{2}(L+1) - m}}{F^{N_\nu - \frac{d}{2}L - \frac{r}{2}}} \{ \mathcal{A}_r P^{m-r} \}^{[\mu_1, \dots, \mu_m]}, \quad (2.1.16)$$

where a strange looking $\{ \mathcal{A}_r P^{m-r} \}^{[\mu_1, \dots, \mu_m]}$ object represents the tensor structure of the numerator, illustrated by the following example:

- $m=2$

$$\sum_{r \leq 2} \{ \mathcal{A}_r P^{2-r} \}^{[\mu_1 \mu_2]} = \{ A_0 P^2 + A_1 P^1 + A_2 P^0 \}^{[\mu_1 \mu_2]} \\ = P^{\mu_1} P^{\mu_2} + \tilde{g}^{\mu_1 \mu_2}, \quad (2.1.17)$$

- m=3

$$\begin{aligned} \sum_{r \leq 3} \{ \mathcal{A}_r P^{3-r} \}^{[\mu_1 \mu_2 \mu_3]} &= \{ A_0 P^3 + A_1 P^2 + A_2 P^1 + A_3 P^0 \}^{[\mu_1 \mu_2 \mu_3]} \\ &= P^{\mu_1} P^{\mu_2} P^{\mu_3} + \tilde{g}^{\mu_1 \mu_2} P^{\mu_3} + \tilde{g}^{\mu_2 \mu_3} P^{\mu_1} + \tilde{g}^{\mu_3 \mu_1} P^{\mu_2}, \end{aligned} \quad (2.1.18)$$

where A_0, P^0 is one, A_r is zero for r odd, and $\mathcal{A}_r = \tilde{g}^{[\mu_1 \mu_2} \dots \tilde{g}^{\mu_{r-1} \mu_r]}$ for r even. P^μ and $\tilde{g}^{\mu\nu}$ are defined as:

$$\begin{aligned} P^{\mu_i} &\rightarrow \sum_l [\tilde{M}_{al} Q_l]_{\mu_i}, \\ \tilde{g}^{\mu_i \mu_j} &\rightarrow (\tilde{M}^{-1})_{ab} g^{\mu_i \mu_j}. \end{aligned} \quad (2.1.19)$$

The indices ab correspond to different loop momenta, for example:

$$G(k_1^{\mu_1} k_2^{\mu_2}) \rightarrow P^{\mu_1} P^{\mu_2} + \tilde{g}^{\mu_1 \mu_2} \rightarrow \sum_l [\tilde{M}_{1l} Q_l]_{\mu_1} [\tilde{M}_{2l} Q_l]_{\mu_2} + (\tilde{M}^{-1})_{12} g^{\mu_1 \mu_2}. \quad (2.1.20)$$

Of course in this dissertation we focus on the one loop integrals ($L = 1$) which obviously leads to:

$$\tilde{M} = \det(M) M^{-1} = 1, \quad (2.1.21)$$

if we bear in mind that M is $L \times L$ matrix.

2.2 Mellin-Barnes representation: theoretical background

Mellin integrals [30], are integrals over contours in a complex plane along the imaginary axis of a product and a ratio of gamma functions defined in Eq.A.0.1. The main feature of the method presented in this chapter is the Mellin-Barnes formula used to represent a sum of terms raised to some power by a product of these terms. This operation allows to create the M-B representation for a given Feynman integral.

The backbone of the procedure to build up Mellin-Barnes representations is the following relation:

$$\frac{1}{(A+B)^\lambda} = \frac{1}{\Gamma(\lambda)} \frac{1}{2\pi i} \int_{c-i\infty}^{c+i\infty} dz \Gamma(\lambda+z) \Gamma(-z) A^z B^{-\lambda-z}, \quad (2.2.1)$$

where

- the integration contour separates the poles of $\Gamma(-z)$ from those of $\Gamma(\lambda+z)$,
- A and B are complex numbers such that $|\arg(A) - \arg(B)| < \pi$.

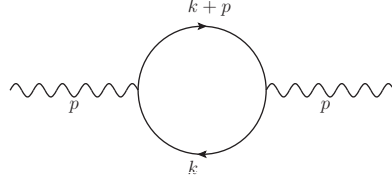


Figure 2.3: Two-point (self-energy) diagram.

The Mellin-Barnes relation Eq.2.2.1 can be iterated and easily extended into a sum of several terms:

$$\begin{aligned} \frac{1}{(A_1 + \dots + A_n)^\lambda} &= \frac{1}{\Gamma(\lambda)} \frac{1}{(2\pi i)^{n-1}} \int_{c-i\infty}^{c+i\infty} \dots \int_{c-i\infty}^{c+i\infty} dz_2 \dots dz_n \prod_{i=2}^n A_i^{z_i} \\ &\times A_1^{-\lambda-z_2-\dots-z_n} \Gamma(\lambda + z_2 + \dots + z_n) \prod_{i=2}^n \Gamma(-z_i). \end{aligned} \quad (2.2.2)$$

As the example of usage of formula Eq.2.2.2, the following self-energy example Fig.2.3 was chosen:

$$G(1)_{\text{SE1l2m}} = \int \frac{d^d k}{(k^2 - m^2 + i0)^{\nu_1} ((k+p)^2 - m^2 + i0)^{\nu_2}}, \quad (2.2.3)$$

where all the typical constant factors were omitted, index SE1l2m stands for self-energy (SE) one-loop (1l) with two massive (2m) internal lines. We will use analogous nomenclature later on. After calculating U and F polynomials (see the previous section) we have:

$$U = x_1 + x_2, \quad F = m^2(x_1 + x_2)^2 - sx_1x_2 - i0. \quad (2.2.4)$$

Feynman parametrisation for this diagram reads:

$$\begin{aligned} G(1)_{\text{SE1l2m}} &= \frac{\Gamma(\nu_1 + \nu_2 - \frac{d}{2})}{\Gamma(\nu_1)\Gamma(\nu_2)} \int_0^1 \prod_{j=1}^2 dx_j x_j^{\nu_j-1} \delta \left(1 - \sum_{i=1}^2 x_i \right) \\ &\times \frac{(x_1 + x_2)^{\nu_1+\nu_2-d}}{(m^2(x_1 + x_2)^2 - sx_1x_2 - i0)^{\nu_1+\nu_2-d/2}}. \end{aligned} \quad (2.2.5)$$

We see that the Dirac δ function causes the U polynomial, which is simply a sum of Feynman parameters, to be one. In general every one-loop n -point diagram has the U polynomial of the form $x_1 + \dots + x_n$, so $U = 1$ for all one-loop cases.

At this point Eq.2.2.1 can be used to start constructing a Mellin-Barnes representation. We use it to replace a sum of Feynman parameters in the F polynomial into its product

with additional integration over the complex space:

$$\begin{aligned}
 \frac{1}{F^\lambda} &= \frac{1}{(m^2(x_1 + x_2)^2 - sx_1x_2 - i0)^\lambda} \\
 &= \frac{1}{\Gamma(\lambda)} \frac{1}{2\pi i} \int_{c-i\infty}^{c+i\infty} dz_1 \Gamma(\lambda + z_1) \Gamma(-z_1) (m^2 - i0)^{z_1} (-s - i0)^{-\lambda - z_1} \\
 &\times (x_1x_2)^{-\lambda - z_1} [x_1 + x_2]^{2z_1},
 \end{aligned} \tag{2.2.6}$$

where $\lambda = \nu_1 + \nu_2 - d/2$. The term $[x_1 + x_2]^{2z_1}$ again can be changed according to Eq.2.2.1 resulting in:

$$\begin{aligned}
 \frac{1}{F^\lambda} &= \frac{1}{\Gamma(\lambda)} \frac{1}{(2\pi i)^2} \int_{c-i\infty}^{c+i\infty} dz_1 \frac{1}{\Gamma(-2z_1)} \int_{c-i\infty}^{c+i\infty} dz_2 \Gamma(\lambda + z_1) \Gamma(-z_1) \Gamma(-2z_1 + z_2) \\
 &\times \Gamma(-z_2) (m^2 - i0)^{z_1} (-s - i0)^{-\lambda - z_1} x_1^{-\lambda - z_1 + z_2} x_2^{-\lambda + z_1 - z_2}.
 \end{aligned} \tag{2.2.7}$$

Next step is to insert Eq.2.2.7 back into Eq.2.2.5 and collect powers of Feynman parameters, which in our case are:

$$\begin{aligned}
 x_1^{a_1-1} &= x_1^{(-\lambda - z_1 + z_2 + \nu_1) - 1}, \\
 x_2^{a_2-1} &= x_2^{(-\lambda + z_1 - z_2 + \nu_2) - 1}.
 \end{aligned} \tag{2.2.8}$$

Finally the integration over Feynman parameters is performed using the following formula:

$$\int_0^1 \prod_{i=1}^n dx_j x_j^{a_j-1} \delta\left(1 - \sum_{i=1}^n x_i\right) = \frac{\Gamma(a_1) \dots \Gamma(a_n)}{\Gamma(a_1 + \dots + a_n)}, \tag{2.2.9}$$

which in practice is restricted to relevant collecting powers of Feynman parameters. The final Mellin-Barnes representation for the self-energy diagram Eq.2.2.3 is²:

$$\begin{aligned}
 G(1)_{\text{SE112m}} &= \frac{(-1)^{\nu_1 + \nu_2}}{\Gamma(\nu_1)\Gamma(\nu_2)} \int_{c-i\infty}^{c+i\infty} dz_1 \int_{c-i\infty}^{c+i\infty} dz_2 (m^2 - i0)^{z_1} (-s - i0)^{d/2 - \nu_1 - \nu_2 - z_1} \\
 &\times \Gamma(-d/2 + \nu_1 + \nu_2 + z_1) \Gamma(-z_1) \Gamma(-2z_1 + z_2) \Gamma(-z_2) \\
 &\times \frac{\Gamma(d/2 - \nu_1 + z_1 - z_2) \Gamma(d/2 - \nu_2 - z_1 + z_2)}{\Gamma(-2z_1) \Gamma(d - \nu_1 - \nu_2)}.
 \end{aligned} \tag{2.2.10}$$

For more complicated cases, F polynomial often contains more than two terms. In such cases it is straightforward to use general Eq.2.2.2 formula.

² Usually in Mellin-Barnes representations infinitesimal complex part $i0$ is omitted e.g. in [5] and later in this thesis. It does not mean that $i0$ is irrelevant. Final analytical results for Feynman integrals in general contain kinematic terms like: $\log(\frac{t}{s}), \log(\frac{-m^2}{s}), \dots$, see e.g. [31]. Even if $i0$ is omitted at the beginning, it must be recreated later to make the analytic continuation to the physical domain possible, e.g. $\log(-x - i0) = \log(|x|) - \Theta(x)i\pi$.

The structure of F polynomial affects form of the final M-B representation. This is obviously due to the fact that Eq.2.2.2 changes sum of n terms raised to some power into $n - 1$ dimensional integral over the complex space. This observation is helpful when one wants to estimate the dimension of the final M-B representation only by looking at F polynomial. It will be discussed in the next chapter in more detail.

During derivation of a Mellin-Barnes representation, one is usually interested in simplifying the final representation (dimensionality of integrals) as much as possible. It is very important if one wants to obtain an analytical result from it. One of the possibilities is to apply one of the following lemmas:

- First Barnes lemma

$$\int_{c-i\infty}^{c+i\infty} dz \quad \Gamma(a+z)\Gamma(b+z)\Gamma(c-z)\Gamma(d-z) = \frac{\Gamma(a+c)\Gamma(a+d)\Gamma(b+c)\Gamma(b+d)}{\Gamma(a+b+c+d)}, \quad (2.2.11)$$

- Second Barnes lemma

$$\int_{c-i\infty}^{c+i\infty} dz \quad \frac{\Gamma(a+z)\Gamma(b+z)\Gamma(c+z)\Gamma(d-z)\Gamma(e-z)}{\Gamma(a+b+c+d+e+z)} = \frac{\Gamma(a+d)\Gamma(a+e)\Gamma(b+d)\Gamma(b+e)\Gamma(c+d)\Gamma(c+e)}{\Gamma(a+b+d+e)\Gamma(a+c+d+e)\Gamma(b+c+d+e)}. \quad (2.2.12)$$

We can now apply first Barnes-Lemma to Eq.2.2.10 and simultaneously substituting powers of propagators equal one $\nu_1 = \nu_2 = 1$ and $d = 4 - 2\epsilon$, we get:

$$G(1)_{\text{SE112m}} = \int_{c-i\infty}^{c+i\infty} dz_1 (m^2)^{z_1} (-s)^{-\epsilon-z_1} \frac{\Gamma(1-\epsilon-z_1)^2 \Gamma(-z_1) \Gamma(\epsilon+z_1)}{\Gamma(2-2\epsilon-2z_1)} \quad (2.2.13)$$

At this point i.e. after defining the Mellin-Barnes representation for the Feynman integral, we are interested in further processing Eq.2.2.13 result so that calculation of the Laurent expanded ϵ terms would be possible. We can use the Cauchy theorem to get representation in terms of a sum of contour integrals, valid at $\epsilon = 0$. Important is that the Mellin-Barnes representation is only well defined, if the integration contour separates the left poles $\Gamma(\dots + z)$ from the right poles $\Gamma(\dots - z)$ and in general for the combination of Γ functions with $\epsilon = 0$ that will not be the case (if the contour is chosen to be a straight line parallel to the imaginary axis). In practice two solutions to this problem have originated:

- "Tausk method" - which fixes the contours parallel to the imaginary axis and accounts for the poles crossing in the analytic continuation [32]. The idea of it is

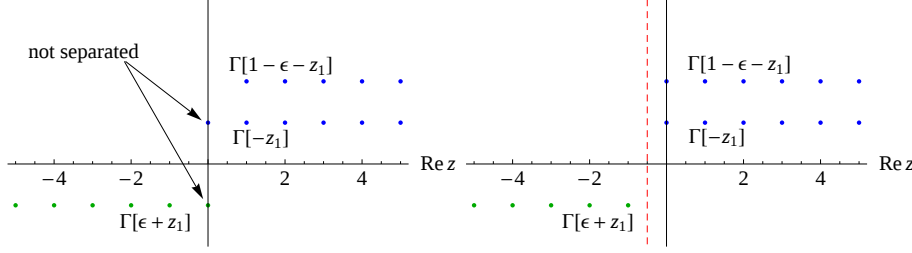


Figure 2.4: Left and right poles of Eq.2.2.13. Left plot shows situation when $\epsilon \rightarrow 0$ which leads to non-separated left and right poles. The right figure presents a proper shift which separates the poles ($\epsilon \rightarrow 1$, $Re(z_1) \rightarrow -1/2$).

presented on Fig.2.4. This method was implemented in **MB** program³ [13] written in the **MATHEMATICA** [18].

- "Smirnov method" - is based on deforming the contour [33] and then shifting them past the poles of the Γ functions, which results in residue integrals. This algorithm has recently been implemented in the **MBresolve** program [34].

Because of applicability of the "Tausk method" in **MB**, we focus on this method and show how it is implemented in the **MB** program by executing two simple commands Eq.2.2.14 and Eq.2.2.15 below. Using this scheme the final result ($\epsilon = 0$) will be obtained from the case where the poles are separated ($\epsilon \neq 0$). Depending whether the poles crossed the contours from left or right (when $\epsilon \rightarrow 0$), one should add or subtract the residue of the integrand on that pole.

Let us focus on the example of Eq.2.2.13. In this case we have the following gamma functions contributing to left and right poles:

- Left poles: $\Gamma(\epsilon + z_1)$
- Right poles: $\Gamma(-z_1)$, $\Gamma(1 - \epsilon - z_1)$,

Graphically we can see what happens if we put $\epsilon \rightarrow 0$, the left plot of Fig.2.4. We note that left and right poles are not separated from each other. To counteract that, the arbitrary choice of $\epsilon \rightarrow 1$, according to the "Tausk method" is chosen, the right plot of Fig.2.4. Separation of left and right poles can be done using the **MB** software, then integration contour is found⁴:

$$\text{rules} = \text{MBOptimizedRules}[\text{final}, \text{eps} \rightarrow 0, \{\}, \{\text{eps}\}]. \quad (2.2.14)$$

³ It allows to analytically continue any MB integral in a given parameter and to resolve the singularity structure in this parameter. The package can also perform numerical integrations at specified kinematic points.

⁴ **MBOptimizedRules** procedure is based on a powerful **MATHEMATICA** function **FindInstance**, which literally finds instance of variables that makes the expression true.

As an output the following choice is calculated: $\epsilon \rightarrow 1$, $Re(z_1) \rightarrow -1/2$. We can see that with such a choice arguments of gamma functions are positive (integral is well defined). The very next step in the calculation of Eq.2.2.13 is the analytic continuation i.e. we have to finally go down to the case where $\epsilon \rightarrow 0$. Again in MB it is done automatically by executing the following command:

$$\text{integrals} = \text{MBcontinue}[\text{final}, \text{eps} \rightarrow 0, \text{rules}], \quad (2.2.15)$$

and as an output we will get the sum of two terms:

$$1: \quad (m^2)^{-\epsilon} \Gamma(\epsilon), \quad (2.2.16)$$

$$2: \quad \int_{-\frac{1}{2}-i\infty}^{-\frac{1}{2}+i\infty} dz_1 (m^2)^{z_1} (-s)^{-\epsilon-z_1} \frac{\Gamma(1-\epsilon-z_1)^2 \Gamma(-z_1) \Gamma(\epsilon+z_1)}{\Gamma(2-2\epsilon-2z_1)}. \quad (2.2.17)$$

Let us note that Eq.2.2.16 is just a residue of Eq.2.2.17. As it has been already written, it appears due to the shift of poles. The `MBcontinue` function implemented in the MB program cares about extracted residues in an automatic way. Now Eq.2.2.17 is well defined in the limit $\epsilon \rightarrow 0$.

If expanded as a Laurent series in ϵ they give contributions to $1/\epsilon$ and the constant part (the last zero argument in the curly brackets below cuts Laurent series at ϵ^0) respectively:

$$\text{MBexpand}[\{\text{integrals}\}, \text{Exp}[\text{eps} * \text{EulerGamma}], \{\text{eps}, 0, 0\}], \quad (2.2.18)$$

where the result was multiplied by an exponent of the Euler gamma γ_e to get rid of it in the final result:

$$1: \quad \frac{1}{\epsilon} - \ln(m^2) \quad (2.2.19)$$

$$2: \quad \int_{-\frac{1}{2}-i\infty}^{-\frac{1}{2}+i\infty} dz_1 (m^2)^{z_1} (-s)^{-z_1} \frac{\Gamma(1-z_1)^2 \Gamma(-z_1) \Gamma(z_1)}{\Gamma(2-2z_1)}. \quad (2.2.20)$$

Obviously the integral which gives contribution to the constant part must be further calculated analytically or integrated numerically. The first is done by summing the residues, the latter can be done within the MB package. Let us now focus on the analytical evaluation of the Eq.2.2.20. This integral can be calculated by closing the contour to the left i.e. it can be written as an infinite sum over the residues by making use of the residue theorem:

$$\oint dz f(z) = 2\pi i \sum \text{Res}[f(z)]. \quad (2.2.21)$$

Taking for simplicity $m = 1$ the sum of Eq.2.2.20 and Eq.2.2.19 is equal:

$$G(1)_{\text{SE112m}} = 2 + \frac{1+x}{1-x} \ln(x) + \frac{1}{\epsilon}, \quad \text{where} \quad x = -\frac{1 - \sqrt{1 - \frac{4}{s}}}{1 + \sqrt{1 - \frac{4}{s}}}. \quad (2.2.22)$$

This analytic result can be cross-checked in Euclidean region by numerical integration Eq.2.2.20, it can be done within **MB** package.

We would like to note that $\ln(x)$ presented in Eq.2.2.22 can be expressed in the language of harmonic polylogarithms [35], here we would have $\ln(x) = H_0(x)$. Analogously we can treat higher orders of expanded terms in ϵ . An alternative way to obtain the same result is differential equation method [36]. In fact, there are problems where M-B and differential equation methods support each other nicely [37].

We went through some simple one dimensional case, where series can be even summed directly by the newest versions of **MATHEMATICA** (starting with version 6.0). But in general multidimensional Mellin-Barnes integrals will be present after using Cauchy's theorem, as nested sums [38]. The state of art research in loop integrals shows that up to four (five) dimensional massive (massless) M-B integrals can be summed up into analytical form [33, 5, 32, 39, 40]. If there is a problem with summation of complicated nested sums, an expansion in a ratio of some kinematic variables can be useful, and the only remedy [31].

2.3 Numerical calculations in the Euclidean region

This section is intended to present the sector decomposition method [41, 42, 23, 25] which isolates divergences from parameter integrals. It is still an active field of research, for the recent idea see [43].

In case of loop integrals, we are interested in the Laurent expansion in ϵ of the Feynman parameter integral Eq.2.1.16 so that the coefficients of the series can be computed numerically later on. The major difficulty is separation of overlapping singularities. Let us look at the following example:

$$\int_0^1 dx_1 \int_0^1 dx_2 \delta \left(1 - \sum_{i=1}^2 x_i \right) \frac{x_1^{-\epsilon} x_2^{-\epsilon}}{x_1(x_1 + x_2)}. \quad (2.3.1)$$

This integral contains the singular region when both x_1 and x_2 vanish simultaneously, they are overlapping for $x_1 \rightarrow 0$ and $x_2 \rightarrow 0$. The first one decomposes the integration range into N sectors, in each l -sector x_l is the largest:

$$\int_0^1 d^N x = \sum_{l=1}^N \int_0^1 d^N x \prod_{\substack{j=1 \\ j \neq l}}^N \theta(x_l \geq x_j), \quad (2.3.2)$$

where θ -function is defined as:

$$\theta(x \geq y) = \begin{cases} 1 & \text{if } x \geq y \text{ is true} \\ 0 & \text{otherwise.} \end{cases}$$

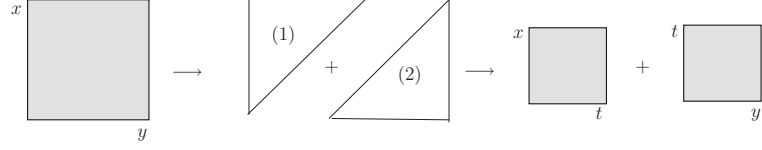


Figure 2.5: The basic idea of sector decomposition method. The figure was taken from [25].

Also the variables are transformed in each l -sector, often called "primary sector" according to the following scheme:

$$x_j = \begin{cases} x_l t_j & \text{for } j < l \\ x_l & \text{for } j = l \\ x_l t_{j-1} & \text{for } j > l \end{cases} \quad (2.3.3)$$

Graphically the situation can be illustrated by Fig.2.5. Because of Eq.2.3, x_l factorizes, also in case of Eq.2.1.16 from F and U i.e. $U(x) \rightarrow U(x)x_l^L$, $F(x) \rightarrow F(x)x_l^{L+1}$. This allows to use:

$$\int_0^1 dx_l/x_l \delta(1 - x_l(1 + \sum_{k=1}^{N-1} t_k)) = 1, \quad (2.3.4)$$

which eliminates x_l , in other words the singular behaviour leading to poles in ϵ comes from the region of small t_i . In general after eliminating x_l we end up with:

$$\int_0^1 \prod_{j=1}^{N-1} t_j t_j^{\nu_j-1} \frac{U_l^{N-(L+1)d/2}(t)}{F_l^{N-Ld/2}(t)} \quad , \quad l = 1, \dots, N. \quad (2.3.5)$$

Of course in general the separation of the singularities is not obtained after first step and all the procedure has to be repeated iteratively (starting from Eq.2.3.5) until a complete separation of overlapping regions is achieved. Finally one obtains a form where all singularities are factorised in terms of parameters t_j and all poles can be extracted leading to integrals which are finite and can be integrated numerically.

Recently public computer implementation of sector decomposition algorithm has appeared [44]. The **sector decomposition** set of libraries allows to calculate numerically given Feynman integral. Because the input of U and F polynomial must be provided, the idea of special interface written in **MATHEMATICA** emerged. Such a program should be capable of preparing all the necessary input for the **sector decomposition**, also execution and output had to be done for the routines in a fully automatic way. Soon **CSectors** [22, 45] was developed allowing easily and quickly obtaining numerical results for a given Feynman integral. Computation is not bounded to scalar L -loop integrals only, but allows to calculate tensor m -rank diagrams as well. Without such an interface

as **CSectors**, user would spend a lot of time preparing U and F polynomials and an appropriate tensor structure. The main purpose of **CSectors** in this thesis is numerical cross-check for the results obtained using Mellin-Barnes method. Let us stress again that numeric works properly for them in Euclidean region.

This section concludes chapter devoted to the evaluation of loop integrals using Mellin-Barnes and sector decomposition methods, precedent by introduction to Feynman parametrisation which is the backbone of both methods. In the following chapter we will focus on derivation of Mellin-Barnes representations using **AMBRE** package and its important issues.

3 Construction of Mellin-Barnes representations by AMBRE package

A brief sketch of Mellin-Barnes techniques presented in the last chapter made it clear that process of derivation of M-B representations should be easily introduced in fully automatic way in a form of some set of program routines. Such an idea came up in 2006, and was practically realized by developing the first public program for constructing Mellin-Barnes representations **AMBRE** [12]. The name itself is an acronym for Automatic Mellin-Barnes Representation. The procedure is designed to calculate:¹

- scalar multi-loop, multi-leg integrals
- tensor m -rank one-loop integrals

AMBRE was written in Computer Algebra System language provided by **MATHEMATICA** and its very favourable point is that the final result of M-B representation can be used as an input in the **MB** [13] program at once.

More detailed technical description of **AMBRE** is provided within this dissertation in the Appendix B.1.

3.1 Loop by Loop algorithm

First section of this chapter will introduce "loop by loop" algorithm which we use to construct Mellin-Barnes representations. Also we will show relevant examples as an output of **AMBRE** and the simplicity of the program usage.

The Feynman parametrisation Eq.2.1.16 contains the so called U function, which is a polynomial depending on Feynman parameters. In the previous chapter it was pointed out that in case of one-loop integrals U can be set as equal one and only F polynomial matters. The scheme we want to use, the so called "loop by loop" or "re-insertion" algorithm means that one treats every sub-loop part of multi-loop diagram as one-loop integral for which the very same Mellin-Barnes methods, as already presented, are applied. The algorithm will be presented using two-loop four point master integral² diagram as an example:

¹ Apart from the public program (ver: 1.2), there is a version of **AMBRE** which is able to derive Mellin-Barnes representations for two-loop m -rank integrals. It has already been used in practical calculations [46].

² It is one of master integrals appearing in two-loop Bhabha $e^+e^- \rightarrow e^+e^-$ process [36].

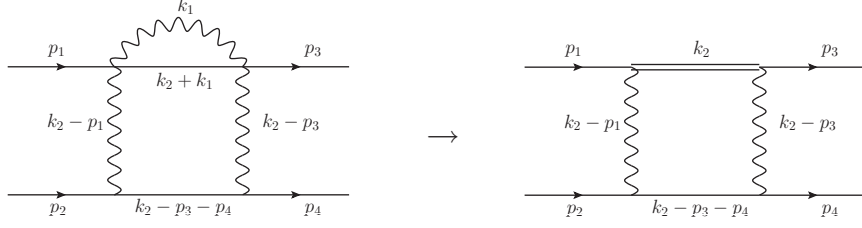


Figure 3.1: The left picture shows two-loop four point master integral (B5l2m2). After applying loop by loop algorithm on the first sub-loop (the one over k_1), the diagram presented on the right side emerges with the modified power of the propagator with momentum k_2 (double internal line).

$$\begin{aligned}
 G_{B5l2m2}(1) &= \int d^d k_1 d^d k_2 \frac{1}{P_1^{n_1} P_2^{n_2} P_3^{n_3} P_4^{n_4} P_5^{n_5}} \\
 &= \int d^d k_2 \frac{1}{P_3^{n_3} P_4^{n_4} P_5^{n_5}} \int d^d k_1 \frac{1}{P_1^{n_1} P_2^{n_2}} \\
 P_1 &= (k_1)^2 \\
 P_2 &= (k_1 + k_2)^2 - m^2 \\
 P_3 &= (k_2 - p_1)^2 \\
 P_4 &= (k_2 - p_3)^2 \\
 P_5 &= (k_2 - p_3 - p_4)^2 - m^2
 \end{aligned} \tag{3.1.1}$$

From now on **AMBRE** notation will be used in forthcoming examples. The detailed description of this convention is presented in Appendix B.1, where appropriate complete example was given. In order to input Eq.3.1.1 into **AMBRE** one has to execute the following:

```
In[.]:= Fullintegral[{1},{PR[k1,0,n1]*PR[k1+k2,m,n2]*PR[k2-p1,0,n3]*
PR[k2-p3,0,n4]*PR[k2-p3-p4,m,n5]},{k1,k2}];
```

Figure 3.2: Eq.3.1.1 as an input in the **AMBRE** package. Symbol `In[.]:=` stands for the **MATHEMATICA** input. From now on we will keep this notation.

where propagators have the following general notation: $\text{PR}[\mathbf{k}, \mathbf{m}, \mathbf{n1}] \equiv (k^2 - m^2)^{-n1}$.

After defining our initial diagram, one has to decide about the order in which one-loop sub-loops will be worked out using Mellin-Barnes Eq.2.2.2 and Feynman parameters integration Eq.2.2.9 formulas. In our example we have two sub-loops in which internal momenta flow over k_1 and k_2 . Let us pick up the one over k_1 as a first sub-loop to compute. In **AMBRE**, demanded order is indicated by the sequence of the list in Fig.3.2, in this case $\{\mathbf{k1}, \mathbf{k2}\}$. The importance of the right choice of ordering of sub-loops will be discussed later.

Using `IntPart` function, **AMBRE** is able to choose right propagators for a given sub-loop without user help Fig.3.3.

```
In[..]:= IntPart[1]
      numerator=1
      integral=PR[k1,0,n1]*PR[k1+k2,m,n2]
      momentum=k1
```

Figure 3.3: `IntPart` separates first sub-loop of Eq.3.1.1. The lines below `In[..]` is an auxiliary output produced by **AMBRE**.

At this point we are ready to calculate F polynomial for the propagators of Fig.3.3. It results in the polynomial of such a form:

$$\begin{aligned} F &= -k_2^2 x_1 x_2 + m^2 x_1 x_2 + m^2 x_2^2 \\ &= -[k_2^2 - m^2] x_1 x_2 + m^2 x_2^2. \end{aligned} \quad (3.1.2)$$

We see that calculation of the sub-loop over k_1 causes k_2 momentum to appear in the F function. At this stage it is treated as a normal momentum which appears because of its presence in one of the propagator in Fig.3.3. After simple rearrangement of F , k_2 can be used to form a new propagator, see right diagram of Fig.3.1 for comparison. This can be always done in this way because of momentum conservation of the nested loops.

Now Mellin-Barnes formula Eq.2.2.2, followed by the integration over Feynman parameters Eq.2.2.9 are both applied. Propagator $k_2^2 - m^2$, which appeared in F will be raised into the power containing integration variables of complex integrals emerging from M-B formula. From the Fig.3.4 we see how Mellin-Barnes representation is automatically obtained for the first sub-loop.

To remove any doubts concerning **AMBRE** notation, the first sub-loop result is repeated in less technical notation below:

$$\begin{aligned} G_{B5l2m2}(1) &= \int \frac{d^d k_2}{[(k_2 - p_1)^2]^{n_3} [(k_2 - p_3)^2]^{n_4} [(k_2 - p_3 - p_4)^2 - m^2]^{n_5} [k_2^2 - m^2]^{-z_1}} \\ &\times \int_{c-i\infty}^{c+i\infty} dz_1 (-1)^{n_1+n_2+z_1} (m^2)^{2-\epsilon-n_1-n_2-z_1} \Gamma(-z_1) \Gamma(n_1+z_1) \\ &\times \frac{\Gamma(4-2\epsilon-2n_1-n_2-z_1) \Gamma(-2+\epsilon+n_1+n_2+z_1)}{\Gamma(n_1) \Gamma(n_2) \Gamma(4-2\epsilon-n_1-n_2)} \end{aligned} \quad (3.1.3)$$

The remaining step is to repeat the same procedure for the second sub-loop i.e. the one over k_2 , which is now just a one-loop box diagram. In **AMBRE** again it is done in a fully automatic way Fig.3.5.

Thus all the planar multi-loop scalar integrals can be treated in this way in order to get Mellin-Barnes representation³. Preceding discussion on the example of master

³ The case of multi-loop tensor integral is more problematic, although technically possible. It is discussed later in this thesis.

```

In[..]:= SubLoop[integral]
Iteration nr1: >>Integrating over k1<<
....
F polynomial...
-PR[k2,m]*X[1]*X[2]+m^2*X[2]^2
Representation after integrating over: k1...
Out[..]:=SubLoop1[((-1)^(n1+n2+z1)*(m^2)^(2-eps-n1-n2-z1)*
Gamma[4-2*eps-2*n1-n2-z1]*Gamma[-z1]*Gamma[n1+z1]*
Gamma[-2+eps+n1+n2+z1])/
(Gamma[n1]*Gamma[4-2*eps-n1-n2]*Gamma[n2]), PR[k2, m, z1]]

```

Figure 3.4: An intermediate result of M-B representation after the first sub-loop was worked out (see Fig.3.1). Note the new propagator appearing in F polynomial. Again `Out[..]:=` should be understood as an output from the program. The rest between `In[..]:=` and `Out[..]:=` has to be treated as an auxiliary **AMBRE** comment output.

```

In[..]:= IntPart[2]
numerator=1
integral=
PR[k2,m,-z1]*PR[k2-p1,0,n3]*PR[k2-p3,0,n4]*PR[k2-p3-p4,m,n5]
momentum=k2
In[..]:= SubLoop[integral]
Iteration nr1: >>Integrating over k2<<
....
F polynomial...
m^2*FX[X[1] + X[4]]^2 - t*X[2]*X[3] - s*X[1]*X[4]
Final representation:
Out[..]:=((-1)^(n1+n2+n3+n4+n5)*(m^2)^(2-eps-n1-n2-z1+z2)*
(-s)^(2-eps-n3-n4-n5+z1-z2-z3)*(-t)^z3*
Gamma[4-2*eps-2*n1-n2-z1]*Gamma[n1+z1]*Gamma[-2+eps+n1+n2+z1]*
Gamma[-z2]*Gamma[-z3]*Gamma[n3+z3]*Gamma[n4+z3]*
Gamma[-2+eps+n3+n4+n5-z1+z2+z3]*Gamma[2-eps-n3-n4+z1+z2-z3-z4]*
Gamma[-z4]*Gamma[-2*z2+z4]*Gamma[2-eps-n3-n4-n5-z2-z3+z4])/
(Gamma[n1]*Gamma[4-2*eps-n1-n2]*Gamma[n2]*Gamma[n3]*Gamma[n4]*
Gamma[n5]*Gamma[4-2*eps-n3-n4-n5+z1]*Gamma[-2*z2])

```

Figure 3.5: A part of **AMBRE** calculation for the second sub-loop of B5l2m2 leading to the full result. The object `FX[...]` appearing in F function is used to indicate appearance of $(x_1 + x_4)^2$ and that it is required to perform another M-B transform on objects appearing in the squared bracket.

integral B5l2m2 can be summarized in the following steps, the so called "loop by loop" or "re-insertion" technique:

1. Define kinematic invariants which depend on the external momenta⁴.
2. Make a decision about the order in which L one-loop sub-loops ($L \geq 1$) will be worked out sequentially.
 - a) Construct a Feynman integral for the chosen sub-loop and perform manipulations on the corresponding F -polynomial to make it optimal for later use of the M-B representations.
 - b) Apply M-B formula Eq.2.2.2 on F -polynomial.
 - c) Integrate over Feynman parameters using equation Eq.2.2.9. For integrals with more than one-loop repeat steps a-c for the remaining sub-loops.
3. Collect the results for all the sub-loops, obtain the final M-B representation result.

In the next section we are going to present some interesting issues related to the construction of Mellin-Barnes representations.

3.2 Remarks on derivation of Mellin-Barnes representations

As it has been already signalized in the previous section, there are interesting issues concerning derivation of Mellin-Barnes representations. Some of them appeared to be a problem during development of **AMBRE** which had to be solved in order to get a working program which could perform loop by loop techniques in an automatic way.

3.2.1 Order of integration in the loop by loop method

During construction of Mellin-Barnes final result, it is possible to choose different order of sub-loops over which representation will be worked out. In this section it will be shown that a free choice of such order, although possible, is not always proper from practical point of view. Wrong sequence leads to significant increase in number of dimensions in the final M-B representation, and makes the final result impossible for use in later calculations.

Before we begin with our discussion on the order of iterations let us consider how the selection of internal momenta flow inside the diagrams impacts derivation of M-B representations. As an example we have massless two-loop vertex diagram presented on Fig.3.6. The first of two diagrams marked by a) has internal momenta chosen in such a way that k_1 and k_2 flow around sub-loops only. None of them is present all around the outer lines, as in case of diagram b). To see the difference between these two diagrams

⁴ On the web page: <http://prac.us.edu.pl/~gluza/ambre/> there is a file called **KinematicsGen**, which generates kinematics for 3-, 4-, 5-, 6- outgoing particles automatically.

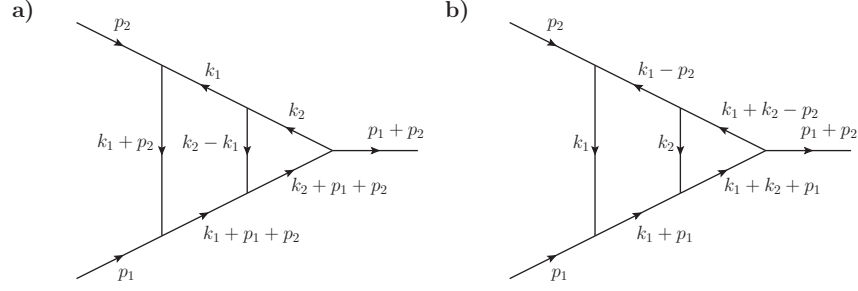


Figure 3.6: Massless vertex diagrams with two different choices of internal momenta flow: a) k_1 and k_2 flow around each of the two sub-loops separately, b) here k_1 flows partly on the same sub-loop as k_2 .

let us take the first of these two graphs and see the process of derivation of Mellin-Barnes representation in **AMBRE** for different order of iteration. As it was mentioned the above vertices are treated as massless diagrams i.e.

$$p_1^2 = p_2^2 = 0, \quad (p_1 + p_2)^2 = s. \quad (3.2.1)$$

In order to prevent reader from being lost when analysing the examples in this section, we introduce the following notation:

- diagram a)
 - example a₁: $k_1 \rightarrow k_2$
 - example a₂: $k_2 \rightarrow k_1$
- diagram b)
 - example b₁: $k_1 \rightarrow k_2$
 - example b₂: $k_2 \rightarrow k_1$.

First we begin with the example a₁, where we choose order $k_1 \rightarrow k_2$. The process of calculating the first sub-loop is presented on Fig.3.7.

We see that the F polynomial will be of the following form, which can be collected in terms of propagators k_2 dependent:

$$\begin{aligned}
 F &= -k_2^2 x_1 x_2 - (k_2^2 + 2k_2 \cdot p_2 + p_2^2) x_2 x_3 - (p_1^2 + 2p_1 \cdot p_2 + p_2^2) x_1 x_4 \\
 &\quad - (k_2^2 + 2k_2 \cdot p_1 + p_1^2 + 2k_2 \cdot p_2 + 2p_1 \cdot p_2 + p_2^2) x_2 x_4 - p_1^2 x_3 x_4 - p_2^2 x_1 x_3 \\
 &= -k_2^2 x_1 x_2 - [k_2 + p_2]^2 x_2 x_3 - s x_1 x_4 - [k_2 + p_1 + p_2]^2 x_2 x_4
 \end{aligned} \quad (3.2.2)$$

Four terms in this polynomial stand for three dimensional complex integral related to k_1 sub-loop. The obtained propagators in the F polynomial will be connected with the

```

In[..]:= IntPart[1]
  numerator=1
  integral=
    PR[k1,0,n1]*PR[-k1+k2,0,n4]*PR[k1+p2,0,n2]*PR[k1+p1+p2,0,n3]
  momentum=k1
In[..]:= SubLoop[integral]
  Iteration nr1: >>Integrating over k1<<
  ...
  F polynomial...
  -PR[k2,0]*X[1]*X[2]-PR[k2+p2,0]*X[2]*X[3]-s*X[1]*X[4]
  -PR[k2+p1+p2,0]*X[2]*X[4]
  Representation after integrating over: k1...
Out[..]:=SubLoop1[((-1)^(2-eps-z3)*(-s)^z3*
  ...,PR[k2,0,z1]*PR[k2+p2,0,z2]*
  PR[k2+p1+p2,0,2-eps-n1-n2-n3-n4-z1-z2-z3]]

```

Figure 3.7: The first sub-loop of example a_1 Fig.3.6 worked out in AMBRE package. A part of the output (dots) was skipped.

```

In[..]:= IntPart[2]
  numerator=1
  integral=PR[k2,0,n6-z1]*PR[k2+p2,0,-z2]*
    PR[k2+p1+p2,0,-2+eps+n1+n2+n3+n4+n5+z1+z2+z3]
  momentum=k2
In[..]:= SubLoop[integral]
  Iteration nr2: >>Integrating over k2<<
  ...
  F polynomial...
  -s*X[1]*X[3]
  Final representation:
Out[..]:=((-1)^(n1+n2+n3+n4+n5+n6)*(-s)^(4-2*eps-n1-n2-n3-n4-n5-n6)*
  Gamma[-z1]*Gamma[2-eps-n1-n2-n4-z1-z2]*Gamma[-z2]*
  Gamma[n2+z2]*Gamma[2-eps-n6+z1+z2]*Gamma[2-eps-n1-n2-n3-z3]*
  Gamma[4-2*eps-n1-n2-n3-n4-n5-z1-z3]*Gamma[-z3]*
  Gamma[-4+2*eps+n1+n2+n3+n4+n5+n6+z3]*Gamma[n1+z1+z3]*
  Gamma[-2+eps+n1+n2+n3+n4+z1+z2+z3])/(Gamma[n1]*Gamma[n2]*
  Gamma[n3]*Gamma[4-2*eps-n1-n2-n3-n4]*Gamma[n4]*Gamma[n6-z1]*
  Gamma[6-3*eps-n1-n2-n3-n4-n5-n6-z3]*
  Gamma[-2+eps+n1+n2+n3+n4+n5+z1+z2+z3])

```

Figure 3.8: The second iteration (i.e. over k_2) of example a_1 Fig.3.6. The final result is a three dimensional Mellin-Barnes representation.

k_2 sub-loop part during second phase of calculations Fig.3.8. Unlike as in the previous Bhabha two-loop example Fig.3.1, now we see more than one propagators in F during first iteration.

If the order of iteration will now be reversed i.e. we move towards example a_2 , where first we start with k_2 and then k_1 , we notice that obtained Mellin-Barnes representation is one dimension smaller than the one presented in the previous example. The complete scheme of **AMBRE** for this calculation is presented in Fig.3.9.

The crucial point which makes example a_1 and example a_2 representations different is the structure of F polynomial during first iteration. In the latter case, F has three terms which lead to two dimensional M-B result. The structure of F polynomial in the last iteration of both examples does not increase dimensionality of the final M-B result.

To convince ourself of the equivalence of these two Mellin-Barnes representations the numerical cross-check can be made using **MB** program [13]. It will be presented later in this chapter. At this moment let us also notice that equivalence between obtained Mellin-Barnes representations (respectively two and three dimensional) imply (after Laurent series expansion in ϵ) non-trivial relations among individual integrals. To our knowledge, this kind of "experimental mathematics" has not yet been explored in the literature.

Another very interesting example is presented on Fig.3.6, diagram b). Let us see what happens when the order $k_1 \rightarrow k_2$ is chosen (example b_1). We limit the following example to show only the F polynomials which are:

$$\begin{aligned} F_1 = & -[k_2 + p_1]^2 x_1 x_3 - k_2^2 x_2 x_3 - s x_2 x_4 - [k_2 + p_1 + p_2]^2 x_3 x_4, \\ & -[k_2 - p_2]^2 x_1 x_5 - [-k_2 + p_1 + p_2]^2 x_2 x_5 - s x_3 x_5 - k_2^2 x_4 x_5 \end{aligned} \quad (3.2.3)$$

for the first sub-loop,

$$F_2 = -s x_2 x_3 - s x_1 x_4 - 2s x_2 x_4 - s x_1 x_5 - 2s x_3 x_5 - 4s x_4 x_5, \quad (3.2.4)$$

for the second sub-loop. One may easily see that the number of terms in both polynomials will cause large (in terms of number of dimensions) Mellin-Barnes representations. This is very unfavourable situation which makes getting analytic ϵ expand result practically impossible, and numerical evaluation must be done on more dimensional complex integrals, which significantly impacts the precision of numerical calculation. In case of example b_2 of Fig.3.6 the reverse situation i.e. $k_2 \rightarrow k_1$ is much more optimal. Such an order generates final 2-dim representation only. As a short summary the following examples are presented. Fig.3.10 shows a two loop box diagram, where the order of iteration is not important, in both cases one derives optimal (with small number of dimensions) Mellin-Barnes representation. In contradiction "court" diagram Fig.3.11 must be worked out according to the following order: $k_2 \rightarrow k_3 \rightarrow k_1$ or $k_3 \rightarrow k_2 \rightarrow k_1$. It is straightforward to notice that starting with k_1 would lead to the same problems as in the example on Fig.3.6. The following shows F polynomials that appear during

```

In[..]:= IntPart[1]
      numerator=1
      integral=PR[k2,0,n6]*PR[-k1+k2,0,n4]*PR[k2+p1+p2,0,n5]
      momentum=k2
In[..]:= SubLoop[integral]
      Iteration nr1: >>Integrating over k1<<
      ...
      F polynomial...
      -PR[k1,0]*X[1]*X[2]-s*X[1]*X[3]-PR[k1+p1+p2,0]*X[2]*X[3]
      Final representation:
      ...
In[..]:= IntPart[2]
      numerator=1
      integral= PR[k1,0,n1-z1]*PR[k1+p2,0,n2]*
      PR[k1+p1+p2,0,-2+eps+n3+n4+n5+n6+z1+z2]
      momentum=k1
In[..]:= SubLoop[integral]
      Iteration nr2: >>Integrating over k2<<
      ...
      F polynomial...
      -s*X[1]*X[3]
      Final representation:
Out[..]:=((-1)^(n1+n2+n3+n4+n5+n6)*(-s)^(4-2*eps-n1-n2-n3-n4-n5-n6)*
      Gamma[2-eps-n4-n6-z1]*Gamma[-z1]*Gamma[2-eps-n1-n2+z1]*
      Gamma[2-eps-n5-n6-z2]*Gamma[4-2*eps-n2-n3-n4-n5-n6-z1-z2]*
      Gamma[-z2]*Gamma[-4+2*eps+n1+n2+n3+n4+n5+n6+z2]*
      Gamma[n6+z1+z2]*Gamma[-2+eps+n4+n5+n6+z1+z2])/(Gamma[n4]*
      Gamma[n5]*Gamma[4-2*eps-n4-n5-n6]*Gamma[n6]*Gamma[n1-z1]*
      Gamma[6-3*eps-n1-n2-n3-n4-n5-n6-z2]*
      Gamma[-2+eps+n3+n4+n5+n6+z1+z2])

```

Figure 3.9: Process of derivation of M-B representation for example a_2 Fig.3.6 in AMBRE. We see that reversing the order of iteration gives two dimensional representation. It is one dimension less comparing to the case of example a_1 .

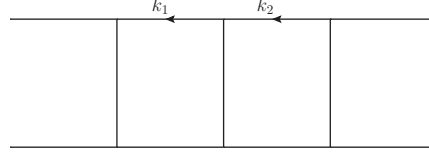


Figure 3.10: Double box diagram, the so called "ladder" diagram. If we choose internal momenta flow as shown, the order of iteration produces optimal representation in both cases.

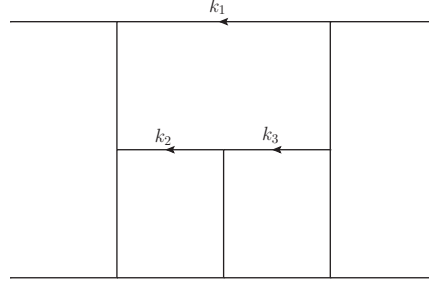


Figure 3.11: Three loop diagram, "court" diagram. In this case $k_2 \rightarrow k_3 \rightarrow k_1$ and $k_3 \rightarrow k_2 \rightarrow k_1$ orders of iterations are optimal.

derivation of the massless "court" diagram:

$$\begin{aligned}
 F_1 &= -[k_3]^2 x_1 x_2 - [k_1 + k_3 + p_1 + p_2]^2 x_2 x_4 - [k_1 + k_3 + p_2]^2 x_2 x_3 \\
 &\quad - [k_1 + p_1 + p_2]^2 x_1 x_4 - [k_1 + p_2]^2 x_1 x_3 \\
 F_2 &= -[k_1 + p_2]^2 x_1 x_2 - [k_1 + p_1 + p_2]^2 x_1 x_3 - [k_1 - p_3]^2 x_1 x_4 - t x_2 x_4 \\
 F_3 &= -s x_1 x_3 - t x_2 x_4,
 \end{aligned} \tag{3.2.5}$$

where index numerates an appropriate iteration (worked out sub-loop). The final Mellin-Barnes representation has eight dimensions.

Note that propagators involving only k_1 internal momenta (i.e. propagators supposed to be used during last iteration) appear already at the first sub-loop. Of course this is normal if one realizes that k_1 momentum flows all the way around outer lines.

3.2.2 Importance of F polynomial in derivation of representations

It is not a big surprise that structure of F polynomial affects the number of dimension of the final Mellin-Barnes representation. The formula Eq.2.2.2 causes to produce $n - 1$ dimension complex integral from n term F polynomial. Let us introduce the following five point example with two different masses Fig.3.12. The Feynman integral introduced

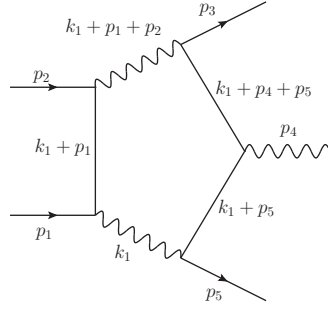


Figure 3.12: QED-like five point diagram.

together with this diagram is as follows:

$$\begin{aligned}
 G(1) &= \int \frac{d^d k}{P_1^{n_1} P_2^{n_2} P_3^{n_3} P_4^{n_4} P_5^{n_5}} \\
 P_1 &= (k_1)^2 \\
 P_2 &= (k_1 + p_1)^2 - m^2 \\
 P_3 &= (k_1 + p_1 + p_2)^2 \\
 P_4 &= (k_1 + p_4 + p_5)^2 - m^2 \\
 P_5 &= (k_1 + p_5)^2 - m^2,
 \end{aligned} \tag{3.2.6}$$

with the independent kinematic variables chosen in the cyclic way i.e. $s_{12}, s_{23}, s_{34}, s_{45}, s_{15}$, where $s_{ij} = (p_i + p_j)^2$. Eq.3.2.6 can be easily connected with QED Bhabha $e^+e^- \rightarrow e^+e^-\gamma$ process if one replaces m by electron mass and massless propagator by photon. The polynomial in this case has the following form:

$$\begin{aligned}
 F &= m^2 x_2^2 + 2m^2 x_2 x_4 + m^2 x_4^2 + 2m^2 x_2 x_5 + 2m^2 x_4 x_5 + m^2 x_5^2 - s_{12} x_1 x_3 \\
 &- s_{15} x_2 x_4 + m^2 x_3 x_4 - s_{34} x_3 x_4 + m^2 x_1 x_5 - s_{45} x_1 x_5 - s_{23} x_2 x_5.
 \end{aligned} \tag{3.2.7}$$

Numerous number of terms leads to Mellin-Barnes representation which is completely useless due to a huge number of integration variables. Simplification of F polynomial can be described in the following steps:

- simplification of all the mass terms into $m^2(x_i + \dots + x_j)^2$ which in **AMBRE** is indicated by `m^2 FX[xi+...+xj]^2`.
- constructing propagators which will be used in the next iteration. This step does not affects above one loop five point integral.
- collecting all the terms in respect to products of Feynman parameters $x_i x_j$.

By default **AMBRE** automatically performs first two steps. In any moment it can be changed by switching to manual mode for calculating F polynomial for a given sub-loop using `Fauto[0]` switch.

Applying first and last step we obtain simplified form of F polynomial:

$$\begin{aligned} F &= m^2(x_2 + x_4 + x_5)^2 \\ &- s_{12}x_1x_3 - s_{15}x_2x_4 + \bar{s}_{34}x_3x_4 + \bar{s}_{45}x_1x_5 - s_{23}x_2x_5, \end{aligned} \quad (3.2.8)$$

where:

$$\begin{aligned} \bar{s}_{34} &= (m^2 - s_{34}) \\ \bar{s}_{45} &= (m^2 - s_{45}). \end{aligned} \quad (3.2.9)$$

From the above polynomial one sees that final M-B result is 7-dim integral. Using first Barnes-Lemma Eq.2.2.11 it can be reduced to 5-dim representation:

$$\begin{aligned} G(1) &= \int_{c-i\infty}^{c+i\infty} \dots \int_{c-i\infty}^{c+i\infty} dz_1 \dots dz_5 (-((m^2)^{z_1} (-s_{12})^{z_2} (-s_{15})^{z_3} (-s_{23})^{-3-\epsilon-z_1-z_2-z_3-z_4-z_5} \\ &\times (\bar{s}_{34})^{z_4} (\bar{s}_{45}q)^{z_5} \Gamma(-z_1) \Gamma(-z_2) \Gamma(-z_3) \Gamma(-2-\epsilon-z_1-z_2-z_3-z_4) \\ &\times \Gamma(-z_4) \Gamma(1+z_2+z_4) \Gamma(1+z_3+z_4) \Gamma(-3-2\epsilon-2z_2-z_4-z_5) \\ &\times \Gamma(-2-\epsilon-z_1-z_2-z_4-z_5) \Gamma(-z_5) \Gamma(1+z_2+z_5) \\ &\times \Gamma(3+\epsilon+z_1+z_2+z_3+z_4+z_5)) / \\ &\times (\Gamma(-1-2\epsilon) \Gamma(-3-2\epsilon-2z_1-2z_2-z_4-z_5)), \end{aligned} \quad (3.2.10)$$

where typical constant factors are omitted and powers of propagators are equal one. We can see that choice of independent kinematic variables also takes effect on number of terms in F polynomial. The substitutions of Eq.3.2.9 could be made on kinematic invariants, although it is relatively easier to do the simplification directly on F polynomial.

Calculations made using **AMBRE** are presented on Fig.3.13 and Fig.3.14. The latter shows application of first Barnes-Lemma. The usage of Barnes-Lemmas as well as some issues related to them will be discussed in the next section.

3.2.3 Using Barnes lemmas

In the introductory chapter about derivation of Mellin-Barnes representations it was mentioned that such complex integrals can be simplified using the so called Barnes lemmas Eq.2.2.11 and Eq.2.2.12, i.e. relations which change integration over complex variable into a product of gamma functions only. Such a simplification means reduction in number of dimensions in a final result. The example of scalar five point function was already discussed in the previous section. There dimension of final Mellin-Barnes representation was reduced by two by applying first Barnes-Lemma twice Fig.3.14.

In **AMBRE** the Barnes-Lemma application can be made using two functions which try to apply Eq.2.2.11, Eq.2.2.12 on M-B result. These functions are:

```

In[..]:= IntPart[1]
      numerator=1
      integral=PR[k,0,n1]*PR[k+p1,m,n2]*PR[k+p1+p2,0,n3]*
      PR[k-p5,m,n5]*PR[k-p4-p5,m,n4]
      momentum=k

In[..]:= Fauto[0]
      ...
      F polynomial...
      fupc=m^2*FX[X[2]+X[4]+X[5]]^2-s12*X[1]*X[3]-s15*X[2]*X[4]+
      m^2*X[3]*X[4]-s34*X[3]*X[4]+m^2*X[1]*X[5]-s45*X[1]*X[5]-
      s23*X[2]*X[5]

In[..]:= fupc = ...
Out[..]:=m^2*FX[X[2]+X[4]+X[5]]^2-s12*X[1]*X[3]-s15*X[2]*X[4]+
      s34p*X[3]*X[4]+s45p*X[1]*X[5]-s23*X[2]*X[5]

In[..]:= repr=SubLoop[integral];
      Iteration nr1: >>Integrating over k<<
      U & F polynomial was computed by user >>Fauto[0]<<
      Final representation:
Out[..]:=((-1)^(n1+n2+n3+n4+n5)*(m^2)^z1*(-s12)^z2*(-s15)^z3*
      (-s23)^(2-eps-n1-n2-n3-n4-n5-z1-z2-z3-z4-z5)*
      (s34p)^z4*(s45p)^z5*
      Gamma[-z1]*Gamma[-z2]*Gamma[-z3]*Gamma[-z4]*
      Gamma[n3+z2+z4]*Gamma[-z5]*Gamma[n1+z2+z5]*
      Gamma[-2+eps+n1+n2+n3+n4+n5+z1+z2+z3+z4+z5]*
      Gamma[-z6]*Gamma[2-eps-n1-n3-n4-n5-z1-z2-z4-z5+z6]*
      Gamma[2-eps-n1-n2-n3-n5+z1-z2-z3-z4-z6-z7]*Gamma[-z7]*
      Gamma[n5+z3+z4+z7]*Gamma[-2*z1+z6+z7])/
      (Gamma[n1]*Gamma[n2]*Gamma[n3]*Gamma[n4]*
      Gamma[4-2*eps-n1-n2-n3-n4-n5]*Gamma[n5]*Gamma[-2*z1])

```

Figure 3.13: Derivation of Mellin-Barnes representation for the scalar five point integral. Here Fauto[0] function was used to allow for the manual modification of F polynomial (fupc).

```

In[...]:= BarnesLemma[repr,1] /. {n1->1,n2->1,n3->1,n4->1,n5->1}
>> Barnes 1st Lemma will be checked for: {z7,z6} <<
Starting with dim=7 representation...

1. Checking z10...Barnes Lemma was applied.
2. Checking z9...Barnes Lemma was applied.
>> Representation after 1st Barnes Lemma: <<
1st Barnes Lemma was applied for: {z6,z7}
Obtained representation has: dim=5

Out[...]:= -(((m^2)^z1*(-s12)^z2*(-s15)^z3*
(-s23)^(-3-eps-z1-z2-z3-z4-z5)*(s34p)^z4*(s45p)^z5*
Gamma[-z1]*Gamma[-z2]*Gamma[-z3]*Gamma[-2-eps-z1-z2-z3-z4]*
Gamma[-z4]*Gamma[1+z2+z4]*Gamma[1+z3+z4]*
Gamma[-3-2*eps-2*z2-z4-z5]*Gamma[-2-eps-z1-z2-z4-z5]*
Gamma[-z5]*Gamma[1+z2+z5]*Gamma[3+eps+z1+z2+z3+z4+z5])/
(Gamma[-1-2*eps]*Gamma[-3-2*eps-2*z1-2*z2-z4-z5]))

```

Figure 3.14: Simplification of Mellin-Barnes QED-like pentagon representation using *AMBRE*. Application of Eq.2.2.11 was possible on integration variables z_7 and z_6 . After eliminating two integration variables final result is 5-dim representation. In the end all the powers of propagators were set to one.

- First Barnes-Lemma: `BarnesLemma[repr_,1]`
- Second Barnes-Lemma: `BarnesLemma[repr_,2]`

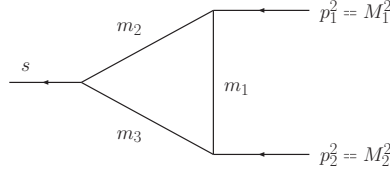


Figure 3.15: One loop vertex diagram with different masses.

In this section we will focus on a situation when normal application of lemmas is not possible due to integration variables z_i appearing in the exponents. Let us follow the example of M-B representation for one-loop vertex diagram Fig.3.15:

$$\begin{aligned}
 G(1) &= \int_{c-i\infty}^{c+i\infty} \dots \int_{c-i\infty}^{c+i\infty} dz_1 \dots dz_7 (-((m_1^2)^{z_1} (m_2^2)^{z_3+z_5} (m_3^2)^{-1-\epsilon-z_1-z_2-z_3-z_4-z_5-z_7} \\
 &\times MM_1^{z_2} MM_2^{z_4} (-s)^{z_7} \\
 &\times \Gamma(-z_1)\Gamma(-z_2)\Gamma(-z_3)\Gamma(-z_4)\Gamma(1+2z_1+z_2+z_4)\Gamma(-z_5)\Gamma(-z_6)\Gamma(-z_7) \\
 &\times \Gamma(1+z_2+2z_3+z_5+z_6+z_7)\Gamma(1+\epsilon+z_1+z_2+z_3+z_4+z_5+z_6+z_7) \\
 &\times \Gamma(-1-2\epsilon-2z_1-2z_2-2z_3-z_4-z_5-z_6-z_7))/\Gamma(1-2\epsilon)). \quad (3.2.11)
 \end{aligned}$$

As it can be seen after carefully analysing Eq.3.2.11 above, Barnes first lemma can be applied to z_6 only, i.e. it appears in gammas twice with negative sign and twice with positive sign, and it is absent in the powers and in the gammas of the denominator. All the remaining integration variables z_i appear in powers which does not allow to introduce first lemma. One can counteract this situation by applying a shift to variables z_3, z_5 i.e. $z_3 \rightarrow z_3 - z_5$. This cancels all the z_5 appearance in the powers. Note that it was possible only because $z_3 + z_5$ was present in one of the powers and $-z_3 - z_5$ in the second one.

Technically the algorithm implemented in the **AMBRE** function responsible for Barnes-Lemmas searches for the pairs of two integration variables $z_i + z_j$ and $z_i - z_j$ which, after application of the appropriate shift cancels them from the powers. This behaviour is implemented in the program as an option which can be turned on or off using boolean parameters. All the procedure of doing first Barnes-Lemma on M-B representation presented, Eq.3.2.11 is presented in Fig.3.16.

Quite recently the special **MATHEMATICA** package **barnesroutines**, specially dedicated to Barnes-Lemma has been released and is publicly available [47].

```

In[...]:= repr=-(((m1^2)^z1*(m2^2)^(z3+z5)*
                (m3^2)^(-1-eps-z1-z2-z3-z4-z5-z7)*
                MM1^z2*MM2^z4*(-s)^z7*
                Gamma[-z1]*Gamma[-z2]*Gamma[-z3]*Gamma[-z4]*
                Gamma[1+2*z1+z2+z4]*Gamma[-z5]*Gamma[-z6]*
                Gamma[-1-2*eps-2*z1-2*z2-2*z3-z4-z5-z6-z7]*
                Gamma[-z7]*Gamma[1+z2+2*z3+z5+z6+z7]*
                Gamma[1+eps+z1+z2+z3+z4+z5+z6+z7])/Gamma[1-2*eps]);

In[...]:= BarnesLemma[repr, 1, Shifts->True]
>> Shifting: {z3->z3-z5}
>> Barnes 1st Lemma will be checked for: {z6,z5} <<

        Starting with dim=7 representation...
        1. Checking z6...Barnes Lemma was applied.
        2. Checking z5...Barnes Lemma was applied.

>> Representation after 1st Barnes Lemma: <<
        1st Barnes Lemma was applied for: {z5,z6}
        Obtained representation has: dim=5

Out[...]:= -(((m1^2)^z1*(m2^2)^z3*(m3^2)^(-1-eps-z1-z2-z3-z4-z7)*
                MM1^z2*MM2^z4*(-s)^z7*Gamma[-z1]*Gamma[-z2]*
                Gamma[-eps-z1-z2-z3]*Gamma[-z3]*Gamma[-2*eps-2*z1-z2-z4]*
                Gamma[-z4]*Gamma[1+2*z1+z2+z4]*Gamma[-z7]*
                Gamma[1+z2+z3+z7]*Gamma[1+eps+z1+z2+z3+z4+z7])/
                (Gamma[1-2*eps]*Gamma[1-eps-z1+z7]))

```

Figure 3.16: **AMBRE** calculations of applying first Barnes-Lemma on Eq.3.2.11. The appropriate shift allows to do lemma substitution not only on z_6 but on z_5 as well.

3.3 One-loop integrals with tensor numerator

Calculation of a given process containing loop integrals involve tensor Feynman integrals. Very nice feature of **AMBRE** is that it allows to derive Mellin-Barnes representations not only for scalar loop diagrams, but also for m -rank one loop integrals. The result can be obtained not only for the tensor numerator contracted with some external momentum and/or metric tensor i.e. $k \cdot p$, $k \cdot k$, ... etc, but also for tensor structure involving internal momentum with Lorentz index only i.e. without contracting it with a projector.

In case of one-loop integrals Eq.2.1.16 can be re-written into the following form:

$$G(T(k)) = \frac{(-1)^{N_\nu}}{\Gamma(\nu_1) \dots \Gamma(\nu_n)} \int_0^1 \prod_{j=1}^n dx_j x_j^{\nu_j-1} \delta \left(1 - \sum_{i=1}^n x_i \right) \times \sum_{r \leq m} \frac{\Gamma(N_\nu - \frac{d}{2} - \frac{r}{2})}{(-2)^{\frac{r}{2}}} \frac{1}{F^{N_\nu - \frac{d}{2} - \frac{r}{2}}} \{ \mathcal{A}_r P^{m-r} \}^{[\mu_1, \dots, \mu_m]}, \quad (3.3.1)$$

where $T(k) = k^{\mu_1} \dots k^{\mu_m}$. One can also note the Eq.2.1.21 property for one-loop cases which simplifies generation of a tensor structure. Let us see the following rank two tensor example (compare with Eq.2.1.20):

$$G(k^{\mu_1} k^{\mu_2}) \rightarrow P^{\mu_1} P^{\mu_2} + \tilde{g}^{\mu_1 \mu_2} \rightarrow Q^{\mu_1} Q^{\mu_2} + g^{\mu_1 \mu_2}. \quad (3.3.2)$$

In **AMBRE** tensor structure generation is implemented in the general way, so it works for any rank. Practically it was tested up to rank nine on some diagrams⁵.

The practical application of **AMBRE** in this thesis is calculation of infrared parts coming from five and four point diagrams for the QED $e^+ e^- \rightarrow \mu^+ \mu^- \gamma$ process (the last chapter). As it is shown later during generation of amplitudes for this process, we deal with Feynman integrals up to rank three. The same maximum rank we have for the Bhabha $e^+ e^- \rightarrow e^+ e^- \gamma$ reaction. Simultaneously in the previous section the one-loop scalar five point QED diagram was presented by Eq.3.2.6. Now this example is going to be extended on the case of rank two tensor integral:

$$G(k^{\mu_1} k^{\mu_2}) = \int d^d k \frac{k^{\mu_1} k^{\mu_2}}{P_1^{n_1} P_2^{n_2} P_3^{n_3} P_4^{n_4} P_5^{n_5}} \begin{aligned} P_1 &= (k_1)^2 \\ P_2 &= (k_1 + p_1)^2 - m^2 \\ P_3 &= (k_1 + p_1 + p_2)^2 \\ P_4 &= (k_1 + p_4 + p_5)^2 - m^2 \\ P_5 &= (k_1 + p_5)^2 - m^2. \end{aligned} \quad (3.3.3)$$

⁵ The mentioned highest rank $m = 9$ was tested on self-energy diagram, although such a rank is not physical it gives confidence that algorithm was implemented correctly. More complicated diagrams were tested as well. Firstly against IBP relations generated by **IdSolver** program [48], and verified by **CSectors**.

From the technical point of view, construction of Mellin-Barnes representation for one-loop tensor integrals is very similar to the scalar ones. In both cases we will have the same F polynomial. The only difference is the numerator represented by the object $\{\mathcal{A}_r P^{m-r}\}^{[\mu_1, \dots, \mu_m]}$, which is present in Eq.3.3.1. It inserts additional Feynman parameters via Q^μ vectors (which is a sum of products of external momentum and Feynman parameter). These new parameters must be included during integration over Feynman parameters Eq.2.2.9. Also because of the sum over r in Eq.3.3.1 the **AMBRE** result is outputted as a list divided according to this sum. This allows to separate terms containing metric tensor and combinations of external momenta (the so called chords). **AMBRE** result for this tensor pentagon integral Eq.3.3.3 is presented on Fig.3.17.

At the end of this section a brief discussion about implementing ability of derivation multi-loop tensor integrals into **AMBRE** is discussed. The main difficulty of such implementation compared to one-loop cases is the "loop by loop" algorithm itself.

We will show this considering the following two-loop self-energy diagram, where all the fields were set to be massless and numerator is of the rank three (note different internal momenta in the numerator):

$$\int \frac{(k_1 \cdot p)(k_1 \cdot p)(k_2 \cdot p)}{[k_1^2]^{n_1} [(k_2 - k_1)^2]^{n_2} [(k_1 + p)^2]^{n_3} [k_2^2]^{n_4} [(k_2 + p)^2]^{n_5}} d^d k_1 d^d k_2. \quad (3.3.4)$$

The calculation starts by working out sub-loop over k_1 , which lead to the following F polynomial:

$$F = -[k_2]^2 x_1 x_2 - s x_1 x_3 - [(k_2 + p)^2] x_2 x_3 \quad (3.3.5)$$

Before we proceed to next sub-loop let us see the structure related to the numerator (tensor structure):

$$\begin{aligned} P^{\mu_1} P^{\mu_2} + \tilde{g}^{\mu_1 \mu_2} &\rightarrow Q^{\mu_1} Q^{\mu_2} + g^{\mu_1 \mu_2} \\ &\rightarrow (k_2^{\mu_1} x_2 - p^{\mu_1} x_3)(k_2^{\mu_2} x_2 - p^{\mu_2} x_3) + g^{\mu_1 \mu_2} \\ &\rightarrow \{k_2^{\mu_1} k_2^{\mu_2} x_2^2, -k_2^{\mu_2} p^{\mu_1} x_2 x_3, k_2^{\mu_1} p^{\mu_2} x_2 x_3, p^{\mu_1} p^{\mu_2} x_3^2, g^{\mu_1 \mu_2}\} \end{aligned} \quad (3.3.6)$$

We clearly see that from that point we will have to perform our second sub-loop calculations separately for all above parts because of different tensor ranks in the next iteration (the one over k_2). Situation after first sub-loop can be presented in the following way:

$$\begin{aligned} &\int \frac{p_{1\mu_1} p_{1\mu_2} (k_2 \cdot p)}{[k_2^2]^{-z_1} [(k_2 + p)^2]^{-3+\epsilon+n_1+n_2+n_3+z_1+z_2}} \\ &\times \{k_2^{\mu_1} k_2^{\mu_2} \text{MB}_1, -k_2^{\mu_2} p^{\mu_1} \text{MB}_2, k_2^{\mu_1} p^{\mu_2} \text{MB}_3, p^{\mu_1} p^{\mu_2} \text{MB}_4, g^{\mu_1 \mu_2} \text{MB}_5\} d^d k_2, \end{aligned} \quad (3.3.7)$$

where MB_i states for a Mellin-Barnes part of the integral for a given part of expression. All these parts are slightly distinct due to different Feynman parameters in Eq.3.3.6. Here as an explicit example we show MB_1 :

$$\begin{aligned} \text{MB}_1 &= ((-1)^{2-\epsilon-z_2} (-s)^{z_2} \Gamma(2-\epsilon-n_1-n_2-z_1) \Gamma(-z_1) \Gamma(4-\epsilon-n_1-n_3-z_2) \\ &\times \Gamma(-z_2) \Gamma(n_1+z_1+z_2) \Gamma(-2+\epsilon+n_1+n_2+n_3+z_1+z_2)) / \\ &\times (\Gamma(n_1) \Gamma(n_2) \Gamma(6-2\epsilon-n_1-n_2-n_3) \Gamma(n_3)). \end{aligned} \quad (3.3.8)$$

```

In[..]:= Fullintegral[{k[mu1],k[mu2]},{PR[k,0,n1]*PR[k+p1,m,n2]*
      PR[k+p1+p2,0,n3]*PR[k+p4+p5,m,n4]*PR[k+p5,m,n5]},{k}];

In[..]:= IntPart[1]
      numerator={k[mu1],k[mu2]}
      integral=PR[k,0,n1]*PR[k+p1,m,n2]*PR[k+p1+p2,0,n3]*
      PR[k+p5,m,n5]*PR[k+p4+p5,m,n4]
      momentum=k

In[..]:= Fauto[0]
      ...

In[..]:= fupc = ...
      m^2*FX[X[2]+X[4]+X[5]]^2-s12*X[1]*X[3]-s15*X[2]*X[4]+
      s34p*X[3]*X[4]+s45p*X[1]*X[5]-s23*X[2]*X[5]

In[..]:= repr = SubLoop[integral];
      Iteration nr1: >>Integrating over k<<
      U & F polynomial was computed by user >>Fauto[0]<<
      Final representation:
      {ARint[1],ARint[2]}

In[..]:= ARint[2,repr]
Out[..]:=-((-1)^(n1+n2+n3+n4+n5)*(m^2)^z1*(-s12)^z2*(-s15)^z3*
      (-s23)^(3-eps-n1-n2-n3-n4-n5-z1-z2-z3-z4-z5)*(s34p)^z4*
      (s45p)^z5*g[mu1*mu2]*Gamma[-z1]*Gamma[-z2]*Gamma[-z3]*
      Gamma[-z4]*Gamma[n3+z2+z4]*Gamma[-z5]*Gamma[n1+z2+z5]*
      Gamma[-3+eps+n1+n2+n3+n4+n5+z1+z2+z3+z4+z5]*Gamma[-z6]*
      Gamma[3-eps-n1-n3-n4-n5-z1-z2-z4-z5+z6]*
      Gamma[3-eps-n1-n2-n3-n5+z1-z2-z3-z4-z6-z7]*Gamma[-z7]*
      Gamma[n5+z3+z4+z7]*Gamma[-2*z1+z6+z7])/
      (2*Gamma[n1]*Gamma[n2]*Gamma[n3]*Gamma[n4]*
      Gamma[6-2*eps-n1-n2-n3-n4-n5]*Gamma[n5]*Gamma[-2*z1])

```

Figure 3.17: Fragment of derivation of M-B representation for QED Bhabha pentagon with rank two numerator in **AMBRE**. Note that calculation was made with not contracted numerator. The final result is by default shortened in the special function **ARint**. Index of it corresponds to the sum over r in Eq.3.3.1 i.e. in this example "1" is for $Q^{\mu_1}Q^{\mu_2}$ term and "2" for $g^{\mu_1\mu_2}$. The latter was extracted using in-build **ARint[2,repr]** function.

It is clear that during more than two loops, additional third iteration can cause to further "fragmentation" of the expression. The above situation was visualised on Fig.3.18, where it is clear that during calculation we get separate Mellin-Barnes terms which must be processed separately.

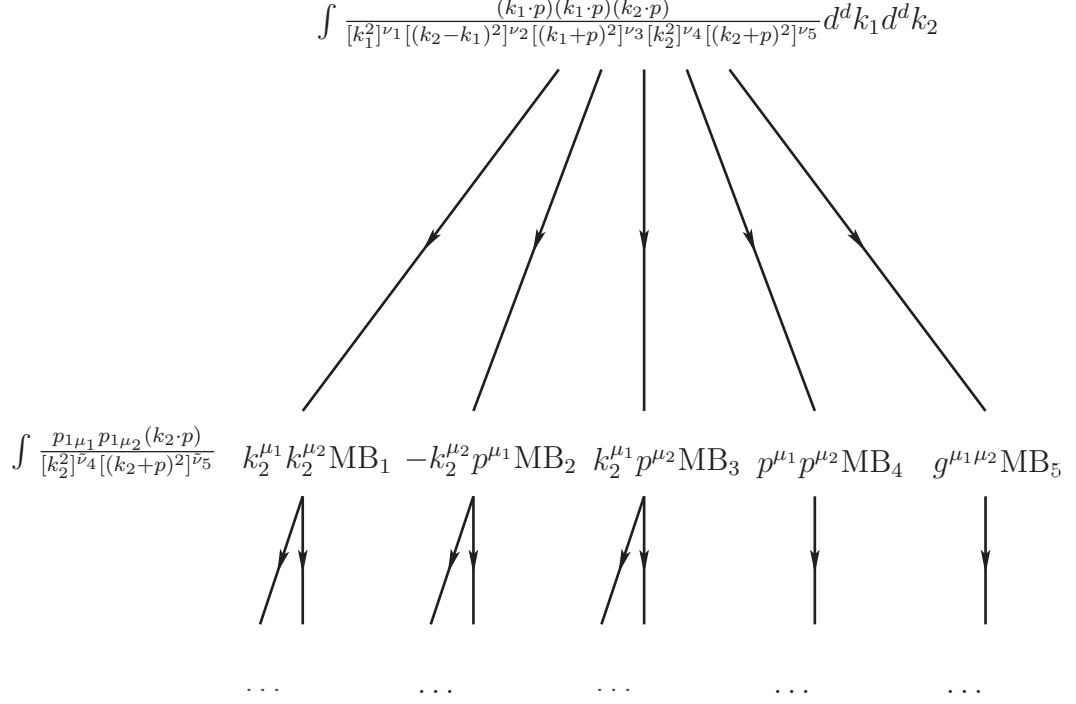


Figure 3.18: Process of calculation of rank three two-loop self-energy diagram Eq.3.3.4. Second step corresponds to Eq.3.3.6. We see that expression fragments to separate M-B terms which must be treated separately.

The last two sections of this chapter summarises work on Mellin-Barnes representation and **AMBRE** package. In the following numerical cross-checks with **CSectors** in Euclidean region and discussion about applications and perspectives is going to be presented.

3.4 Cross-checks in Euclidean region with **CSectors**

Up to now all presented Mellin-Barnes examples were illustrated without independent final result comparison. Current section covers this gap and provides numerical results for M-B representations derived using **AMBRE** and numerically calculated in **MB**, and the very same diagrams computed using sector decomposition method in **CSectors**. As it was already pointed out, **MB** package is able to perform numerical integration of complex integrals which appear after analytic continuation and expansion in ϵ . One dimensional integrals are calculated using **MATHEMATICA** in-built functions and higher dimensional

ones are integrated via **Fortran**⁶. The numerical integration in **MB** is executed using the following function:

$$\text{MBIntegrate}[\text{integrals}, \text{kinematics}, \text{options}], \quad (3.4.1)$$

where **integrals** is a set of integrals to evaluate and **kinematics** is a set of rules for kinematic variables providing numerical input to the computation routines e.g. $\{\mathbf{s} \rightarrow -2, \mathbf{t} \rightarrow -3\}$. The last one is a set of options which in detail are described in [13]. In case of **CSectors** numerical computation was done using **sector decomposition** [44] set of routines⁷.

We begin numerical cross-checks by going back to the two-loop vertex example a) in Fig.3.6 (V6l0m case). Here we will show that order of iteration in "loop by loop" algorithm leads to the same numerical results. Also additional verification was made with the help of **CSectors**. Below we show numerical output up to constant part i.e. ϵ^0 :

$$\begin{aligned} \text{V6l0m}_{\text{example a}_1} &= 8004.28|6140449509 + \frac{2452.280156|9081353}{\epsilon} \\ &+ \frac{596.665332838|975}{\epsilon^2} + \frac{145.07266400430143}{\epsilon^3} + \frac{30.25}{\epsilon^4} \\ \text{V6l0m}_{\text{example a}_2} &= 8004.28|5575775925 + \frac{2452.280156|8656705}{\epsilon} + \\ &+ \frac{596.665332838|1408}{\epsilon^2} + \frac{145.07266400430143}{\epsilon^3} + \frac{30.25}{\epsilon^4}, \\ \text{V6l0m}_{\text{CSectors}} &= (7997.4 \pm 3.9) + \frac{(2451.5 \pm 0.6)}{\epsilon} + \frac{(596.648 \pm 0.05)}{\epsilon^2} + \\ &+ \frac{(145.066 \pm 0.007)}{\epsilon^3} + \frac{(30.248 \pm 0.002)}{\epsilon^4}, \end{aligned} \quad (3.4.2)$$

where point $s = -\frac{1}{11}$ was used as an input parameter. The output of the first two results given by **MB** package may seem incorrectly typed due to a number of digits left and lack of errors coming from numerical integration. Here we indicated common parts of numerical results by vertical lines. In fact terms ϵ^{-4} and ϵ^{-3} are exactly the same because their contribution comes from non-integrals (some analytic expression). Here **MATHEMATICA** can provide huge number of digits of precision for such cases. For ϵ^{-2} , ϵ^{-1} and constant parts the contribution is from one and two dimensional complex integrals, which are calculated in **MATHEMATICA** (within **MB**) and **Fortran** respectively. In contrast to the

⁶ The code is linked with **CUBA** libraries [49], which is a library offering a choice of four independent routines for multidimensional numerical integration: Vegas, Suave, Divonne and Cuhre. **CERNlib** [50] implements complex gamma (Γ) functions and its derivative (ψ). By default deterministic Cuhre algorithm is used for integration when dimension < 5 . Above this dimension threshold Monte Carlo Vegas is used. This behaviour can be changed by an appropriate option.

⁷ It is an implementation in **C++** using **GiNaC** libraries for the symbolic part and the GNU Scientific Library for the numerical part (Monte Carlo integration).

second one the first one does not provide errors which should come within numerical integration. In other words error information is useless. In case of **CSectors** this is not a problem at all. All the uncertainties are calculated via **sector decomposition** libraries (see Appendix B).

The rest of numerical cross-checks is presented in the following tables. During numerical calculations all the masses and powers of propagators were set to one.

	AMBRE and MB	CSectors
ϵ^0	-0.3933253674084889	$-0.393325 \pm 5.6 \cdot 10^{-7}$
ϵ^{-1}	1	$1 \pm 1.4 \cdot 10^{-6}$
T [s]	4.3	7.6
$s = -3$		

Table 3.1: Numerical results for the self-energy diagram Fig.2.3. All the computations have been made on Intel Core Duo 2.66GHz with 1GB memory.

	AMBRE and MB	CSectors
ϵ^0	-20.314 ± 0.001	-20.315 ± 0.001
ϵ^{-1}	-5.3229(62554815002)	-5.3231 ± 0.0002
T [s]	3.8	15.4
$s = -11, t = -1/2$		

Table 3.2: Results for two-loop Master Integral (B5l2m2) Fig.3.1 appearing in two loop $e^+e^- \rightarrow e^+e^-$ Bhabha process.

	AMBRE and MB	CSectors
ϵ^0	0.22858 ± 0.00005	0.228637 ± 0.00004
ϵ^{-1}	0.136722	$0.136726 \pm 9.9 \cdot 10^{-6}$
T [s]	22.3	16.3
$s_{12} = -3, s_{23} = -7, s_{34} = -1/2, s_{45} = -1/11, s_{15} = -6$		

Table 3.3: Numerics for the one loop QED pentagon diagram Fig.3.12.

If we analyse these results and time of calculation we can define the following finding. Numerical calculations using Mellin-Barnes method often takes longer for diagrams involving masses than in case of the very same massless cases. Such situation is due to higher number of terms appearing in F polynomial, which in turn translates to more dimensional M-B representation and more difficult integration in the end. In case of sector decomposition (**CSectors**), situation is reversed. Here massless cases often causes serious troubles. The number of generated sectors easily can cause to fill completely RAM and SWAP memory. For example calculation of the three-loop massless ladder

	AMBRE and MB	CSectors
ϵ^0	$-0.103415 \pm 6 \cdot 10^{-6}$	-0.103354 ± 0.0004
ϵ^{-1}	-0.109074	-0.109271 ± 0.00009
ϵ^{-2}	-0.00965743	-0.00966186 ± 0.00002
ϵ^{-3}	0.0831878	$0.083189 \pm 4 \cdot 10^{-6}$
ϵ^{-4}	-0.0228571	$-0.022858 \pm 1 \cdot 10^{-6}$
T [s]	32	5185
$s = -5, t = -7$		

Table 3.4: Numerical result for the massless double box Fig.3.10.

box (the extension of the diagram in Fig.3.10) is relatively easy from numerical point of view using Mellin-Barnes method. But it is a serious challenge when using sector decomposition method.

3.5 Applications and perspectives of *AMBRE*

Modern high energy physics requires performing sometimes difficult calculations which often are impossible without computer. Many aspects of numerical and analytical computations can be dressed into appropriate algorithms and in the end automatised using some programming language. The automatisation of derivation of Mellin-Barnes representations was the most important reason in developing *AMBRE* package. The existence of such software shortens time spend on M-B calculations and eliminates possible human mistakes during intermediate steps. The *AMBRE* was entirely written in *MATHEMATICA* Computer Algebra System language, which in natural way connects its output with *MB* package also written in the very same CAS.

AMBRE has already been used in a few important calculations in QED and QCD. It helped to solve analytically many new Master Integrals e.g. for three loop massless QCD form factors [39, 40, 51], in massive QED Bhabha [52] and Maximally Supersymmetric Yang-Mills theory [53].

Presently we are using it for numerical checks of new algorithm which solves Master Integrals using relations among two-loop integrals which follows from Gram determinants [46]. It is e.g. used in calculations of the so called "double pentagon" Fig.3.19. Here we meet sixteen dimensional integrals. For that we have introduced an *AMBRE* version capable of computing two-loop tensor integrals. Relations among these integrals include dozens of tensor double pentagons. We use powerful computers to solve them numerically. The obvious perspective of *AMBRE* is generalisation from tensor m -rank one- and two- loop integrals to L -loop cases. Another interesting extension could be tried in case of non-planar diagrams.

This chapter ends discussion about *AMBRE* application of Mellin-Barnes. In the last chapter we will present IR divergences calculated with the help of this program for five

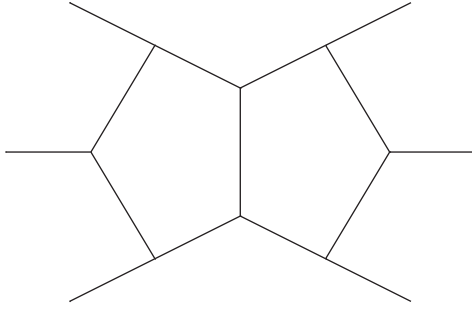


Figure 3.19: The topology of the so called "double pentagon" as an example of one of the various applications of *AMBRE*.

and four point one-loop diagrams presented in the $e^+e^- \rightarrow \mu^+\mu^-\gamma$ QED process. For that the following chapter describes reduction of six and five point one-loop integrals.

4 Tensor reduction of five and six point one loop integrals

Reduction of n point one-loop Feynman diagrams to a sum of its reduced diagrams has a long history going back in time to the early 60's of the last century [54]. It was then indicated that the amplitude for n -point one-loop diagram with $n > 6$ can be written as a sum of the reduced diagrams containing only five external legs [55]. Later this work was generalized and five point one-loop diagrams were introduced as a sum of box diagrams [56]. In fact authors of that time also proved that triangle one-loop diagram could be reduced to the sum of three self-energy diagrams if the space-time would have been two dimensional [57]. Later work was focused on reduction schemes for five and six point integrals so that unwanted Gram determinant could be avoided [17, 20, 21, 16].

In this chapter we focus on evaluation of tensor one-loop integrals especially the ones with five and six external legs. Firstly the well known Passarino-Veltman reduction scheme is shortly introduced in order to brought in Gram determinants which often are the source of numerical instability. Later the scheme of reduction based on algebra of "signed minors" [54] in which unwanted determinants were cancelled is presented [15, 16]. The chapter is summarized by the numerical cross-checks made with the specially developed new software along this scheme: `hexagon`¹ and non-public `Fortran` program.

This chapter can be treated as a continuation of [17]. We not only focus on cancellation of Gram determinants for high ranks of tensor five point integrals but we also finalised our work by developing reduction software. This is an important step towards working out good numerically stable software for this kind of numerical calculations. Because discussion about presented reduction method could fill all the thesis we focus only on some basic ideas on which this reduction scheme is based on. More results can be found in our main paper [16].

4.1 Passarino-Veltman reduction

The Passarino-Veltman tensor reduction [14] was historically first systematic procedure which could be implied for one-loop integrals. This chapter will briefly introduce this method, which is to be the base for the later discussion. As an example, let us first

¹ It is available here: <http://prac.us.edu.pl/~gluza/hexagon/>

write the basic one-loop Feynman integrals up to three legs and up to rank two:

$$\begin{aligned} A(m_1) &= \int \frac{d^d k}{i\pi^{d/2}} \frac{1}{P_1}, \\ B^{0,\mu,\mu\nu}(p_1, m_1, m_2) &= \int \frac{d^d k}{i\pi^{d/2}} \frac{1, k^\mu, k^{\mu\nu}}{P_1 P_2}, \\ C^{0,\mu,\mu\nu}(p_1, p_2, m_1, m_2, m_3) &= \int \frac{d^d k}{i\pi^{d/2}} \frac{1, k^\mu, k^{\mu\nu}}{P_1 P_2 P_3}, \end{aligned} \quad (4.1.1)$$

where

$$\begin{aligned} P_1 &= k^2 - m_1^2, \\ P_2 &= (k - p_1)^2 - m_2^2, \\ P_3 &= (k - p_1 - p_2)^2 - m_3^2. \end{aligned} \quad (4.1.2)$$

Obviously more external legs and higher rank tensor generalisation is possible here. Passarino-Veltman reduction uses the fact that due to the Lorentz symmetry the result can only depend on tensor structures built from the external momenta p_i^μ and the metric tensor $g^{\mu\nu}$. Therefore tensor integrals are written in terms of coefficients multiplied by external momenta and/or metric tensor i.e.:

$$\begin{aligned} B^\mu &= p_1^\mu B_1, \\ B^{\mu\nu} &= p_1^\mu p_1^\nu B_{11} + g^{\mu\nu} B_{00}, \\ C^\mu &= p_1^\mu C_1 + p_2^\mu C_2, \\ C^{\mu\nu} &= p_1^\mu p_1^\nu C_{11} + (p_1^\mu p_2^\nu + p_1^\nu p_2^\mu) C_{12} + q_2^\mu q_2^\nu C_{22} + g^{\mu\nu} C_{00}. \end{aligned} \quad (4.1.3)$$

By contracting both sides of above expressions with external momenta and the metric tensor, it is possible to find a solution for the coefficients: $B_i, B_{ij}, \dots$. After contraction of left hand sides scalar products appear, which can be easily expressed in terms of propagators e.g.:

$$k \cdot p_1 = \frac{1}{2}([k^2 - m_1^2] - [(k - p_1)^2 - m_2^2] + m_1^2 - m_2^2 + p_1^2). \quad (4.1.4)$$

In the case of the two point vector integral we would end up with the following formula allowing to find B_1 coefficient:

$$p_1^2 B_1 = \frac{1}{2}((p_1^2 + m_1^2 - m_2^2)B_1 + A(m_2) - A(m_1)). \quad (4.1.5)$$

Note that A integrals appeared after one of the propagators had been cancelled in the two point B function. For the other case i.e. vector three point integral, where explicit calculation of $p_{i\mu} C^\mu$ was omitted for simplicity, we have:

$$\begin{pmatrix} p_{1\mu} C^\mu \\ p_{2\mu} C^\mu \end{pmatrix} = \begin{pmatrix} p_1^2 & p_1 \cdot p_2 \\ p_1 \cdot p_2 & p_2^2 \end{pmatrix} \begin{pmatrix} C_1 \\ C_2 \end{pmatrix}. \quad (4.1.6)$$

We see that when trying to solve this system of equations the inverse matrix together with determinant appear in the denominator:

$$2 \begin{vmatrix} p_1^2 & p_1 \cdot p_2 \\ p_1 \cdot p_2 & p_2^2 \end{vmatrix} \quad (4.1.7)$$

The most important drawback of this algorithm is related to these determinants. In some phase space regions Gram determinants can tend to zero, resulting in coefficients taking large values, with possible cancellations among them. This is the main difficulty in development of numerically stable program for automated evaluation of tensor loop integrals.

4.2 Gram determinants and algebra of signed minors

In the previous section we have seen how Gram determinants appear in the Passarino-Veltman tensor reduction scheme, the three point diagram example was given Eq.4.1.7. Such determinants are obviously generalized to the case of n -point one-loop diagrams:

$$G_{n-1} = 2 \begin{vmatrix} p_1^2 & p_1 \cdot p_2 & \dots & p_1 \cdot p_{n-1} \\ p_1 \cdot p_2 & p_2^2 & \dots & p_2 \cdot p_{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ p_1 \cdot p_{n-1} & p_2 \cdot p_{n-1} & \dots & p_{n-1}^2 \end{vmatrix}. \quad (4.2.1)$$

At this point we are going to introduce the basic of algebra used in our reduction scheme. For a diagram with internal lines $1 \dots n$ the so called "modified Cayley determinant" can be introduced:

$$()_n = \begin{vmatrix} 0 & 1 & 1 & \dots & 1 \\ 1 & Y_{11} & Y_{12} & \dots & Y_{1n} \\ 1 & Y_{12} & Y_{22} & \dots & Y_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & Y_{1n} & Y_{2n} & \dots & Y_{nn} \end{vmatrix}, \quad (4.2.2)$$

with $Y_{ij} = -(q_i - q_j)^2 + m_i^2 + m_j^2$, where q_i are chords of a given diagram. Both determinants are related by:

$$()_n = -G_{n-1}, \quad (4.2.3)$$

and from now on we will name $()_n$ the Gram determinant of the Feynman integral. By cutting from $()_n$ rows $j_1, j_2, \dots$ and columns $k_1, k_2, \dots$ we get the so-called "signed minors". The sign of this object is determined by the sum of indices of excluded rows and columns and by taking into account the appropriate signatures of the permutations, taken separately from excluded rows and columns i.e.:

$$\begin{pmatrix} j_1 & j_2 & \dots \\ k_1 & k_2 & \dots \end{pmatrix}_n \equiv (-1)^{\sum_i (j_i + k_i)} \text{sgn}_j \text{sgn}_k \begin{vmatrix} \text{rows } j_1 j_2 \dots & \text{deleted} \\ \text{rows } k_1 k_2 \dots & \text{deleted} \end{vmatrix}. \quad (4.2.4)$$

Excluded rows and columns of Cayley determinant are numbered from zero, in this way we have:

$$\binom{0}{0}_n = \begin{vmatrix} Y_{11} & Y_{12} & \dots & Y_{1n} \\ Y_{12} & Y_{22} & \dots & Y_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{1n} & Y_{2n} & \dots & Y_{nn} \end{vmatrix}. \quad (4.2.5)$$

Using "signed minors" one can write tensor reduction formulas in a compact and elegant way but before we proceed further let us see some interesting properties related to them. They are crucial if we want to write reduction formulas without Gram determinants. First of all one evidently sees that due to the symmetric Cayley determinant the "signed minor" objects are symmetric in replacing excluded rows by columns:

$$\binom{i_1 \dots i_r}{j_1 \dots j_r}_n = \binom{j_1 \dots j_r}{i_1 \dots i_r}_n. \quad (4.2.6)$$

The other two very important properties which will be needed are:

$$\sum_{i=1}^n \binom{0}{i}_n = \binom{0}{0}_n, \quad (4.2.7)$$

together with its extension

$$\sum_{i=1}^n \binom{j \ 0}{k \ i}_n = \binom{j}{k}_n, \quad (4.2.8)$$

and similar to Eq.4.2.7 but with $j \neq 0$ inserted

$$\sum_{i=1}^n \binom{j}{i}_n = 0. \quad (4.2.9)$$

The last two that will be needed are:

$$\binom{\alpha}{0}_n \binom{\alpha \ \beta}{k \ l}_n + \binom{\alpha}{k}_n \binom{\alpha \ \beta}{l \ 0}_n + \binom{\alpha}{l}_n \binom{\alpha \ \beta}{0 \ k}_n = 0, \quad (4.2.10)$$

and

$$\binom{i \ l}{j \ k}_n \binom{0}{0}_n = \binom{i}{j}_n \binom{l}{k}_n - \binom{i}{k}_n \binom{l}{j}_n. \quad (4.2.11)$$

All of these properties have been already discussed in [54], where also the detailed and complete introduction to the algebra of "signed minors" was given.

4.3 Tensor integrals and shifted space-time dimensions

At first we give the reduction of tensor integrals to a set of scalar integrals for arbitrary n -point functions. Following [17, 58], and setting all the powers of propagators to be equal to one, one has:

$$I_n^\mu = \int^d k^\mu \prod_{r=1}^n P_r^{-1} = - \sum_{i=1}^{n-1} q_i^\mu I_{n,i}^{[d+]}, \quad (4.3.1)$$

$$I_n^{\mu\nu} = \int^d k^\mu k^\nu \prod_{r=1}^n P_r^{-1} = \sum_{i,j=1}^{n-1} q_i^\mu q_j^\nu \nu_{ij} I_{n,ij}^{[d+]^2} - \frac{1}{2} g^{\mu\nu} I_n^{[d+]}, \quad (4.3.2)$$

where $[d+]$ is an operator shifting the space-time dimension by two units and

$$\begin{aligned} I_{p,ij k \dots}^{[d+]^l, stu \dots} &= \int^{[d+]^l} \prod_{r=1}^n \frac{1}{P_r^{1+\delta_{ri}+\delta_{rj}+\delta_{rk}+\dots-\delta_{rs}-\delta_{rt}-\delta_{ru}-\dots}}, \\ \int^d &\equiv \int \frac{d^d k}{i\pi^{d/2}}, \\ [d+]^l &= 4 + 2l - 2\epsilon, \end{aligned} \quad (4.3.3)$$

with propagator denominators

$$P_j = (k - q_j)^2 - m_j^2. \quad (4.3.4)$$

By combining integration by parts identities with relations connecting integrals in different space-time dimensions [28], one obtains the following basic recurrence relations [17]:

$$()_n \nu_j \mathbf{j}^+ I_n^{[d+]} = \left[-\binom{j}{0}_n + \sum_{k=1}^n \binom{j}{k}_n \mathbf{k}^- \right] I_n, \quad (4.3.5)$$

$$(d - \sum_{i=1}^n \nu_i + 1) ()_n I_n^{[d+]} = \left[\binom{0}{0}_n - \sum_{k=1}^n \binom{0}{k}_n \mathbf{k}^- \right] I_n, \quad (4.3.6)$$

$$\binom{0}{0}_n \nu_j \mathbf{j}^+ I_n = \sum_{k=1}^n \binom{0j}{0k}_n \left[d - \sum_{i=1}^n \nu_i (\mathbf{k}^- \mathbf{i}^+ + 1) \right] I_n. \quad (4.3.7)$$

4.4 Reduction of pentagons up to rank two

We begin the following reduction scheme based on the algebra of "signed minors" with the example of scalar pentagon diagrams. For the scalar five point integral we use recursion relation Eq.4.3.6. The operator $\mathbf{k}^-$ decreases the k -th propagator power

by one. Because in our example we have all powers of propagators equal to one, and because we are interested in the case with $n = 5$, Eq.4.3.6 changes into:

$$(d-4) \binom{0}{0}_5 I_5^{[d+]} = \binom{0}{0}_5 I_5 - \sum_{s=1}^5 \binom{0}{s}_5 I_4^s \quad (4.4.1)$$

With $I_5^{[d+]}$ finite in the limit $d \rightarrow 4$ we get as a final reduction formula for the scalar five point function:

$$I_5 = \frac{1}{\binom{0}{0}_5} \sum_{s=1}^5 \binom{0}{s}_5 I_4^s, \quad (4.4.2)$$

i.e. five point integral is expressed in terms of scalar four point functions I_4^s , which are obtained by scratching s -th line in the pentagon diagram. In the case of vector five point integral (rank $m = 1$) we have:

$$I_5^\mu = \sum_{i=1}^4 q_i^\mu I_{5,i}, \quad (4.4.3)$$

with

$$I_{5,i} \equiv -I_{5,i}^{[d+]} = (d-4) \frac{\binom{0}{i}_5}{\binom{0}{0}_5} I_5^{[d+]} - \frac{1}{\binom{0}{0}_5} \sum_{s=1}^5 \binom{0}{0} \binom{i}{s}_5 I_4^s, \quad (4.4.4)$$

where again in the limit $d \rightarrow 4$ the $I_5^{[d+]}$ disappears and finally:

$$I_{5,i} = -\frac{1}{\binom{0}{0}_5} \sum_{s=1}^5 \binom{0}{0} \binom{i}{s}_5 I_4^s. \quad (4.4.5)$$

I_4^s as in the case of scalar integrals are constructed by cancelling s -th line in five point diagram. Scalar and vector cases are simple and lead to a direct reduction to scalar four point integrals, without the Gram determinant $(\cdot)_5$. In the following we reduce tensor integrals of rank two and show, like in [20, 59], that also in these cases the unwanted Gram determinant can be cancelled.

Let us now progress with more complicated case which is rank two five point integral. This tensor integral can be written Eq.4.3.2:

$$I_5^{\mu\nu} = \sum_{i,j=1}^4 q_i^\mu q_j^\nu \nu_{ij} I_{5,ij}^{[d+]^2} - \frac{1}{2} g^{\mu\nu} I_5^{[d+]}. \quad (4.4.6)$$

By replacing the metric tensor $g_{\mu\nu}$ by²:

$$g^{\mu\nu} = 2 \sum_{i,j=1}^4 \frac{\binom{i}{j}_5}{\binom{}{5}} q_i^\mu q_j^\nu, \quad (4.4.7)$$

and again using recurrence relation Eq.4.3.5 we obtain:

$$\begin{aligned} I_5^{\mu\nu} &= \sum_{i,j=1}^4 q_i^\mu q_j^\nu I_{5,ij} \\ I_{5,ij} = \nu_{ij} I_{5,ij}^{[d+]} &= -\frac{\binom{0}{j}_5}{\binom{}{5}} I_{5,i}^{[d+]} + \sum_{s=1, s \neq i}^5 \frac{\binom{s}{j}_5}{\binom{}{5}} I_{4,i}^{[d+],s}. \end{aligned} \quad (4.4.8)$$

Now $I_{5,i}^{[d+]}$ can be replaced by rank two coefficient i.e. Eq.4.4.4 and $I_{4,i}^{[d+],s}$ is constructed using recurrence relation Eq.4.3.5. Explicit formulas for recurrence relations of specific I_5 and I_4 cases can be found in [16]. Finally we end up with the following result:

$$\begin{aligned} I_{5,ij} &= -\frac{1}{\binom{0}{0}_5 \binom{}{5}} \sum_{s=1, s \neq i}^5 \left\{ \frac{\binom{0}{j}_5 \binom{0}{0}_5 \binom{i}{s}_5}{\binom{}{5}} + \frac{\binom{s}{j}_5 \binom{0}{i}_5 \binom{s}{s}_5}{\binom{s}{s}_5} \right\} I_4^s \\ &+ \frac{1}{\binom{}{5}} \sum_{s,t=1, s,t \neq i}^5 \left\{ \frac{\binom{s}{j}_5 \binom{t}{i}_5 \binom{s}{s}_5}{\binom{s}{s}_5} \right\} I_3^{st}, \end{aligned} \quad (4.4.9)$$

where four point I_4^s and three point I_3^{st} integrals were factorised. Now we would like to use algebra of signed minors to somehow cancel unwanted Gram determinant $\binom{}{5}$. Before we do this let us change Eq.4.4.9 by adding and subtracting into coefficient of I_4^s the following term:

$$\frac{\binom{s}{0}_5 \binom{0}{0}_5 \binom{s}{s}_5 \binom{i}{j}_5}{\binom{s}{s}_5}, \quad (4.4.10)$$

² The Eq.4.4.7 is based on the assumption that the chord $q_5^\mu = 0$ (otherwise the summation variables i and j would run from 1 to 5). With this choice, one can prove Eq.4.4.7 by contracting both sides of the equation with, e.g. $q_{l\mu}$. That is sufficient, because any (four dimensional) vector must be a linear combination of $q_{l\mu}$. To work out the r.h.s, one needs to write $2q_i \cdot q_l = Y_{il} - Y_{i5} - Y_{5l} + Y_{55}$ [54] and then use some identities for sums like $\sum_{i=1}^5 \binom{i}{j}_5 Y_{il}$.

and for the expression in the bracket of I_3^{st} we add the following zero term:

$$\frac{\begin{pmatrix} s \\ 0 \end{pmatrix}_5 \begin{pmatrix} t & s \\ 0 & s \end{pmatrix}_5 \begin{pmatrix} i \\ j \end{pmatrix}_5}{\begin{pmatrix} 0 \\ 0 \end{pmatrix}_5 \begin{pmatrix} s \\ s \end{pmatrix}_5} - \frac{\begin{pmatrix} s \\ 0 \end{pmatrix}_5 \begin{pmatrix} t & s \\ 0 & s \end{pmatrix}_5 \begin{pmatrix} i \\ j \end{pmatrix}_5}{\begin{pmatrix} 0 \\ 0 \end{pmatrix}_5 \begin{pmatrix} s \\ s \end{pmatrix}_5} + \frac{\begin{pmatrix} 0 \\ j \end{pmatrix}_5 \begin{pmatrix} t & s \\ 0 & i \end{pmatrix}_5}{\begin{pmatrix} 0 \\ 0 \end{pmatrix}_5} = 0, \quad (4.4.11)$$

which also uses the fact that:

$$\sum_{s,t} \begin{pmatrix} t & s \\ 0 & i \end{pmatrix}_5 = \sum_s \left(\sum_t \begin{pmatrix} 0 & s \\ t & i \end{pmatrix}_5 \right) = 0. \quad (4.4.12)$$

After simple modifications we get such an intermediate expression:

$$\begin{aligned} I_{5,ij} &= \frac{1}{\begin{pmatrix} 0 \\ 0 \end{pmatrix}_5 \begin{pmatrix} \\ \end{pmatrix}_5} \sum_{s=1, s \neq i}^5 \frac{1}{\begin{pmatrix} s \\ s \end{pmatrix}_5} \left\{ - \begin{pmatrix} 0 \\ j \end{pmatrix}_5 \begin{pmatrix} 0 & s \\ 0 & i \end{pmatrix}_5 \begin{pmatrix} s \\ s \end{pmatrix}_5 - \begin{pmatrix} s \\ j \end{pmatrix}_5 \begin{pmatrix} 0 & s \\ i & s \end{pmatrix}_5 \begin{pmatrix} 0 \\ 0 \end{pmatrix}_5 \right. \\ &+ \left. \begin{pmatrix} s \\ 0 \end{pmatrix}_5 \begin{pmatrix} 0 & s \\ 0 & s \end{pmatrix}_5 \begin{pmatrix} i \\ j \end{pmatrix}_5 \right\} I_4^s \\ &- \frac{1}{\begin{pmatrix} 0 \\ 0 \end{pmatrix}_5 \begin{pmatrix} \\ \end{pmatrix}_5} \sum_{s,t=1, s \neq i, t}^5 \frac{1}{\begin{pmatrix} s \\ s \end{pmatrix}_5} \left\{ - \begin{pmatrix} 0 \\ j \end{pmatrix}_5 \begin{pmatrix} t & s \\ 0 & i \end{pmatrix}_5 \begin{pmatrix} s \\ s \end{pmatrix}_5 - \begin{pmatrix} s \\ j \end{pmatrix}_5 \begin{pmatrix} t & s \\ i & s \end{pmatrix}_5 \begin{pmatrix} 0 \\ 0 \end{pmatrix}_5 \right. \\ &+ \left. \begin{pmatrix} s \\ 0 \end{pmatrix}_5 \begin{pmatrix} t & s \\ 0 & s \end{pmatrix}_5 \begin{pmatrix} i \\ j \end{pmatrix}_5 \right\} I_3^{st} \\ &- \frac{\begin{pmatrix} i \\ j \end{pmatrix}_5}{\begin{pmatrix} 0 \\ 0 \end{pmatrix}_5 \begin{pmatrix} \\ \end{pmatrix}_5} \sum_{s=1, s \neq i}^5 \frac{1}{\begin{pmatrix} s \\ s \end{pmatrix}_5} \begin{pmatrix} s \\ 0 \end{pmatrix}_5 \begin{pmatrix} 0 & s \\ 0 & s \end{pmatrix}_5 I_4^s \\ &+ \frac{\begin{pmatrix} i \\ j \end{pmatrix}_5}{\begin{pmatrix} 0 \\ 0 \end{pmatrix}_5 \begin{pmatrix} \\ \end{pmatrix}_5} \sum_{s,t=1, s \neq i, t}^5 \frac{1}{\begin{pmatrix} s \\ s \end{pmatrix}_5} \begin{pmatrix} s \\ 0 \end{pmatrix}_5 \begin{pmatrix} t & s \\ 0 & s \end{pmatrix}_5 I_3^{st}. \end{aligned} \quad (4.4.13)$$

At this stage we would like to show that Gram determinant $()_5$ really factorises from the bracket terms, so that it cancels with the one in the denominator. So we postulate

the following:

$$\begin{aligned} A_{ij}^{s0} &= -\binom{0}{j}_5 \binom{0 \ s}{0 \ i}_5 \binom{s}{s}_5 - \binom{s}{j}_5 \binom{0 \ s}{i \ s}_5 \binom{0}{0}_5 + \binom{s}{0}_5 \binom{0 \ s}{0 \ s}_5 \binom{i}{j}_5 \\ &= \binom{}{}_5 X_{ij}^{s0} \end{aligned} \quad (4.4.14)$$

$$\begin{aligned} A_{ij}^{st} &= -\binom{0}{j}_5 \binom{t \ s}{0 \ i}_5 \binom{s}{s}_5 - \binom{s}{j}_5 \binom{t \ s}{i \ s}_5 \binom{0}{0}_5 + \binom{s}{0}_5 \binom{0 \ s}{0 \ s}_5 \binom{i}{j}_5 \\ &= \binom{}{}_5 X_{ij}^{st} \end{aligned} \quad (4.4.15)$$

We start from proving Eq.4.4.14. To begin we show that r.h.s of is symmetric in the indices i and j for fixed s . Obviously the third term is symmetric due to the symmetric Cayley determinant Eq.4.2.6. The symmetry of the first two terms means

$$\begin{aligned} &\binom{s}{s}_5 \left[\binom{0}{i}_5 \binom{0 \ j}{0 \ s}_5 - \binom{0}{j}_5 \binom{0 \ i}{0 \ s}_5 \right] \\ &+ \binom{0}{0}_5 \left[\binom{s}{i}_5 \binom{0 \ s}{j \ s}_5 - \binom{s}{j}_5 \binom{0 \ s}{i \ s}_5 \right] = 0. \end{aligned} \quad (4.4.16)$$

By using relation Eq.4.2.10 and basic properties of signed minors we do the following substitution:

$$\begin{aligned} \binom{0}{j}_5 \binom{0 \ i}{0 \ s}_5 &= \binom{0}{j}_5 \binom{0 \ s}{0 \ i}_5 = -\binom{0}{i}_5 \binom{0 \ s}{j \ 0}_5 - \binom{0}{0}_5 \binom{0 \ s}{i \ j}_5 \\ &= \binom{0}{i}_5 \binom{0 \ s}{0 \ j}_5 - \binom{0}{0}_5 \binom{0 \ s}{i \ j}_5, \end{aligned} \quad (4.4.17)$$

where we used Eq.4.2.6, Eq.4.2.10. Inserting above into Eq.4.4.16 results in:

$$\binom{s}{i}_5 \binom{0 \ s}{j \ s}_5 + \binom{s}{j}_5 \binom{0 \ s}{s \ i}_5 + \binom{s}{s}_5 \binom{0 \ s}{i \ j}_5 = 0, \quad (4.4.18)$$

which is easily proved using Eq.4.2.11, all the terms will immediately cancel.

Now let us see the simplest case of Eq.4.4.14 i.e. A_{ss}^{s0} . In this case we see that $A_{ss}^{s0} = 0$ which in turn implicates:

$$X_{ss}^{s0} = 0. \quad (4.4.19)$$

If we move further we see that by applying Eq.4.2.7 and Eq.4.2.9 to Eq.4.4.14, we get:

$$\sum_{j=1}^5 A_{ij}^{s0} = -\binom{}{}_5 \binom{0 \ s}{0 \ i}_5 \binom{s}{s}_5, \quad (4.4.20)$$

It is still presented in the remaining two terms but this is not a problem at all. Again using expression for metric tensor Eq.4.4.7 we finally end with a compact reduction formula for rank two five point integral without $(\cdot)_5$.

$$I_5^{\mu\nu} = \sum_{i,j=1}^4 q_i^\mu q_j^\nu E_{ij} + g^{\mu\nu} E_{00}, \quad (4.4.28)$$

$$E_{ij} = \sum_{s=1}^5 S_{ij}^{4,s} I_4^s + \sum_{s,t=1}^5 S_{ij}^{3,st} I_3^{st}, \quad (4.4.29)$$

where

$$S_{ij}^{4,s} = \frac{1}{\binom{0}{0}_5} \sum_{s=1}^5 \frac{1}{\binom{s}{s}_5} X_{ij}^{s0}, \quad (4.4.30)$$

$$S_{ij}^{3,st} = -\frac{1}{\binom{0}{0}_5} \sum_{s,t=1}^5 \frac{1}{\binom{s}{s}_5} X_{ij}^{st} \quad (4.4.31)$$

$$E_{00} = -\frac{1}{2} \frac{1}{\binom{0}{0}_5} \sum_{s=1}^5 \frac{\binom{s}{0}_5}{\binom{s}{s}_5} \left[\binom{0}{0} \binom{s}{s}_5 I_4^s - \sum_{t=1}^5 \binom{t}{0} \binom{s}{s}_5 I_3^{st} \right]. \quad (4.4.32)$$

The higher rank tensor integrals were derived using the same recurrence relations and "signed minors" algebra. In case of five point integrals the reduction formulas were derived up to rank three in [16].

4.5 Reduction of six point functions

In case of hexagons there is a nice property which states that tensors of rank m can be reduced to a sum of six five point tensor integrals of rank $m - 1$. This property has also been derived in [20], an earlier demonstration of this property has been given already in [17]. This simplification is due to the fact that their Gram determinant vanishes $(\cdot)_6 = 0$ [17]. Apart from that, the above results for the five point tensor integrals can be directly used, thus reducing the six point tensors to scalar four, three and two point integrals. According to recurrence relations we write:

$$I_6 = \sum_{r=1}^6 \frac{\binom{0}{r}_6}{\binom{0}{0}_6} I_5^r = \sum_{r=1}^6 \frac{\binom{r}{k}_6}{\binom{0}{k}_6} I_5^r, \quad k = 1, \dots, 6, \quad (4.5.1)$$

and Eq.4.4.2 now reads:

$$I_5^r = \frac{1}{\binom{0}{0} \binom{r}{r}_6} \sum_{s=1, s \neq r}^6 \binom{0}{s} \binom{r}{r}_6 I_4^{rs}. \quad (4.5.2)$$

Here we see already the general scheme of reducing six point functions to five point functions. In general, in any "signed minor" a further column:

$$\binom{r}{r}_6 \quad (4.5.3)$$

is added. That is because we now use Cayley determinant of dimension six and additional r -part is scratched. As in Eq.4.4.3 and Eq.4.4.4 we obtain:

$$I_6^\mu = \sum_{i=1}^5 q_i^\mu I_{6,i}, \quad (4.5.4)$$

$$\begin{aligned} I_{6,i} &= -I_{6,i}^{[d+]} \\ &= (d-5) \frac{\binom{0}{i}_6}{\binom{0}{0}_6} I_6^{[d+]} - \frac{1}{\binom{0}{0}_6} \sum_{r=1}^6 \binom{0}{0} \binom{i}{r}_6 I_5^r. \end{aligned} \quad (4.5.5)$$

While in Eq.4.4.4 the first part vanishes in the limit $d \rightarrow 4$, here its disappearance is due to Eq.4.2.7 and $()_6 = 0$:

$$\sum_{i=1}^5 q_i^\mu \binom{0}{i}_6 = 0. \quad (4.5.6)$$

Eq.4.5.6 will play a crucial role for the higher tensor reduction. The resulting form in Eq.4.5.5 is already the generic form for the higher tensors too. Therefore it is useful to introduce a vector:

$$\begin{aligned} v_r^\mu &= -\frac{1}{\binom{0}{0}_6} \sum_{i=1}^5 \binom{0}{0} \binom{i}{r}_6 q_i^\mu \\ &= -\frac{1}{\binom{0}{0}_6} \sum_{i=1}^5 \binom{0}{k} \binom{r}{i}_6 q_i^\mu, \quad k = 0, \dots, 6. \end{aligned} \quad (4.5.7)$$

With this definition we can write vector integral in a compact way:

$$I_6^\mu = \sum_{r=1}^6 v_r^\mu I_5^r. \quad (4.5.8)$$

We see that vector six point integral depends on scalar five point function. In fact, tensor rank two and three hexagon integrals will be reduced according to the scheme where dependence on rank one and two respectively will be observed. All the reduction formulas for the hexagons up to rank four were given in [16].

4.6 Numerical results and discussion

In order to check the correctness of the presented reduction scheme the **hexagon** package was written. As it allows to obtain also fully analytic result it appeared to be very helpful tool during process of derivation of reduction formulas. During the stage of cross-checking we have made the following verifications:

- internal checks were used for tensor integrals, and consisted mainly in writing a scalar product of internal and external momenta in terms of lower rank tensor integrals, which had been checked before.
- external checks were made with use of: **LoopTools** [7] (five point integrals), **AMBRE** with **MB** (five point integrals), **Csectors** (five and six point integrals).

We present them in Table 4.1. For some cross-checks the coefficient of tensor integrals was only calculated. Our notation is presented in the following:

$$\begin{aligned}
 F^\mu &= \sum_{i=1}^5 q_i^\mu F_i, \\
 F^{\mu\nu} &= \sum_{i,j=1}^5 q_i^\mu q_j^\nu F_{ij}, \\
 F^{\mu\nu\lambda} &= \sum_{i,j,k=1}^5 q_i^\mu q_j^\nu q_k^\lambda F_{ijk} + \sum_{i=1}^5 g^{\mu\nu} q_i^\lambda F_{00i}, \\
 F^{\mu\nu\lambda\rho} &= \sum_{i,j,k,l=1}^5 q_i^\mu q_j^\nu q_k^\lambda q_l^\rho F_{ijkl} + \sum_{i,j=1}^5 q_i^\mu q_j^{[\nu} g^{\lambda\rho]} F_{00ij}. \\
 E^\mu &= \sum_{i=1}^4 q_i^\mu E_i, \\
 E^{\mu\nu} &= \sum_{i,j=1}^4 q_i^\mu q_j^\nu E_{ij} + g^{\mu\nu} E_{00}, \\
 E^{\mu\nu\lambda} &= \sum_{i,j,k=1}^4 q_i^\mu q_j^\nu q_k^\lambda E_{ijk} + \sum_{i=1}^4 g^{[\mu\nu} q_i^{\lambda]} E_{00i},
 \end{aligned} \tag{4.6.1}$$

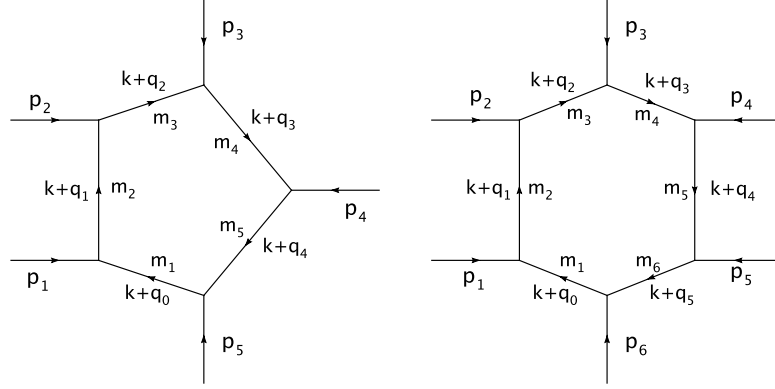


Figure 4.1: Momenta flow used in the numerical examples for five and six point integrals.

where F stands for six point integrals and E for five point functions. The momenta flow convention in six and five point diagrams is presented on Fig.4.1. In all these checks, we used **LoopTools** to calculate the finite parts of the scalar four, three and two point functions which appear after the reduction by **hexagon**. We note that, in general, the functions defined directly in **LoopTools** may not be sufficient to cover the whole kinematic phase space obtained from reduction of six point functions. But as an alternative it is possible to use other available libraries e.g. **QCDLoop** [60].

Numerical results in Table 4.1 are divided in to parts described below:

- 1) Here we used **AMBRE** and **MB** to check the decomposition in the Euclidean kinematic region, including tensor structures. Because **LoopTools** does not provide result containing ϵ terms, so we have limited our result, which is contracted tensor of rank three, to constant part only. At the level typical for Monte-Carlo calculations, it is in agreement with the result from Mellin-Barnes integration which was: 0.218885.
- 2) The second example comes from [61], (Table 2, region I). Here, we use **CSectors** and show for the case of the scalar six point function agreement with the calculation using the sector decomposition method (typically five digits accuracy).
- 3) In this example we give cross-check for scalar five point function between **hexagon**, **LoopTools** and [61] (Table 1, region III). The other tensor calculated tensor coefficients agree directly with **LoopTools**.

Because two independent implementations of reduction scheme were developed: **hexagon** and the one in **Fortran**, the cross-checks were also made among these two packages. In the Table-4.2 we give phase space points corresponding to the reaction $gg \rightarrow t\bar{t}q\bar{q}$, with external momenta generated by **Madgraph** [62, 63]. In the other tables we give numerical results (which is presented as tensor components) for this phase space point. All the results are the same for two programs.

1) Comparison with AMBRE & MB.m : $p_1^\mu p_1^\nu p_1^\lambda E_{\mu\nu\lambda}$
Phase space point: $p_1^2 = p_2^2 = p_3^2 = p_5^2 = 1, p_4^2 = 0, m_1^2 = m_3^2 = 0, m_2^2 = m_4^2 = m_5^2 = 1,$ $s_{12} = -3, s_{23} = -6, s_{34} = -5, s_{45} = -7, s_{15} = -2$
In: RedE3[$p_1^2, \dots, p_5^2, s_{12}, s_{23}, s_{34}, s_{45}, s_{15}, m_1^2, \dots, m_5^2$] Out: 0.218741
2) Comparison with sector decomposition and [61]: F_0
Phase space point: $p_1^2 = p_2^2 = p_3^2 = p_4^2 = p_5^2 = p_6^2 = -1, m_1^2 = m_2^2 = m_3^2 = m_4^2 = m_5^2 = m_6^2 = 1,$ $s_{12} = s_{23} = s_{34} = s_{45} = s_{56} = s_{16} = s_{123} = s_{234} = -1, s_{345} = -5/2$
In: RedF0[$p_1^2, \dots, p_6^2, s_{12}, s_{23}, s_{34}, s_{45}, s_{56}, s_{16}, s_{123}, s_{234}, s_{345}, m_1^2, \dots, m_6^2$] Out: 0.013526
3) Comparison with LoopTools : E_0 [61], $E_1, E_2, E_3, E_4, E_{34}, E_{123}, E_{002}$
Phase space point: $p_1^2 = p_2^2 = 0, p_3^2 = p_5^2 = 49/256, p_4^2 = 9/100,$ $m_1^2 = m_2^2 = m_3^2 = 49/256, m_4^2 = m_5^2 = 81/1600,$ $s_{12} = 4, s_{23} = -1/5, s_{34} = 1/5, s_{45} = 3/10, s_{15} = -1/2$
In: RedE0[$p_1^2, \dots, p_5^2, s_{12}, s_{23}, s_{34}, s_{45}, s_{15}, m_1^2, \dots, m_5^2$] Out: 41.3403 - 45.9721*I
In: RedEget[rank1, $p_1^2, \dots, p_5^2, s_{12}, s_{23}, s_{34}, s_{45}, s_{15}, m_1^2, \dots, m_5^2$] Out: ee1 = -2.38605 + 5.27599*I, ee2 = -5.80875 + 0.597891*I, ee3 = -14.4931 + 20.8149*I, ee4 = -11.3362 + 18.1593*I
In: RedEcoef[ee34, $p_1^2, \dots, p_5^2, s_{12}, s_{23}, s_{34}, s_{45}, s_{15}, m_1^2, \dots, m_5^2$] Out: 7.1964 + 3.10115*I
In: RedEcoef[ee123, $p_1^2, \dots, p_5^2, s_{12}, s_{23}, s_{34}, s_{45}, s_{15}, m_1^2, \dots, m_5^2$] Out: -0.149527 - 0.31059*I
In: RedEcoef[ee002, $p_1^2, \dots, p_5^2, s_{12}, s_{23}, s_{34}, s_{45}, s_{15}, m_1^2, \dots, m_5^2$] Out: 0.154517 - 0.387727*I

Table 4.1: Numerical cross-checks.

p_1	0.21774554 E+03	0.0	0.0	0.21774554 E+03
p_2	0.21774554 E+03	0.0	0.0	- 0.21774554 E+03
p_3	- 0.20369415 E+03	- 0.47579512 E+02	0.42126823 E+02	0.84097181 E+02
p_4	- 0.20907237 E+03	0.55215961 E+02	- 0.46692034 E+02	- 0.90010087 E+02
p_5	- 0.68463308 E+01	0.53063195 E+01	0.29698267 E+01	- 0.31456871 E+01
p_6	- 0.15878244 E+02	- 0.12942769 E+02	0.15953850 E+01	0.90585932 E+01
$m_1 = 110.0, m_2 = 120.0, m_3 = 130.0, m_4 = 140.0, m_5 = 150.0, m_6 = 160.0$				

Table 4.2: The components of external four-momenta for the six point numerics, where all internal particles are massive.

		F_0
		- 0.223393 E-18 - i 0.396728 E-19
μ		F^μ
0		0.192487 E-17 + i 0.972635 E-17
1		- 0.363320 E-17 - i 0.11940 E-17
2		0.365514 E-17 + i 0.106928 E-17
3		0.239793 E-16 + i 0.341928 E-17
μ	ν	$F^{\mu\nu}$
0	0	0.599459 E-14 - i 0.114601 E-14
0	1	0.323869 E-15 + i 0.423754 E-15
0	2	- 0.294252 E-15 - i 0.375481 E-15
0	3	- 0.255450 E-14 - i 0.195640 E-14
1	1	- 0.164562 E-14 - i 0.993796 E-16
1	2	0.920944 E-16 + i 0.706487 E-17
1	3	0.347694 E-15 - i 0.127190 E-16
2	2	- 0.163339 E-14 - i 0.994148 E-16
2	3	- 0.341773 E-15 + i 0.818678 E-17
3	3	- 0.413909 E-14 + i 0.670676 E-15

Table 4.3: Tensor components for scalar, vector, and rank $m = 2$ six-point functions; kinematics defined in Table 4.2 and Fig.4.1.

μ	ν	λ	$F^{\mu\nu\lambda}$
0	0	0	$-0.227754 \text{ E-11} - i 0.267244 \text{ E-12}$
0	0	1	$0.140271 \text{ E-13} - i 0.119448 \text{ E-12}$
0	0	2	$-0.201270 \text{ E-13} + i 0.101968 \text{ E-12}$
0	0	3	$0.102976 \text{ E-12} + i 0.624467 \text{ E-12}$
0	1	1	$0.183904 \text{ E-12} + i 0.142429 \text{ E-12}$
0	1	2	$-0.131028 \text{ E-13} - i 0.610343 \text{ E-14}$
0	1	3	$-0.543316 \text{ E-13} - i 0.158809 \text{ E-13}$
0	2	2	$0.181352 \text{ E-12} + i 0.141686 \text{ E-12}$
0	2	3	$0.506408 \text{ E-13} + i 0.163568 \text{ E-13}$
0	3	3	$0.600542 \text{ E-12} + i 0.130733 \text{ E-12}$
1	1	1	$-0.563539 \text{ E-13} + i 0.178403 \text{ E-13}$
1	1	2	$0.210641 \text{ E-13} - i 0.584990 \text{ E-14}$
1	1	3	$0.120482 \text{ E-12} - i 0.574688 \text{ E-13}$
1	2	2	$-0.201182 \text{ E-13} + i 0.620591 \text{ E-14}$
1	2	3	$-0.686164 \text{ E-14} + i 0.205457 \text{ E-14}$
1	3	3	$-0.447329 \text{ E-13} + i 0.193180 \text{ E-13}$
2	2	2	$0.582201 \text{ E-13} - i 0.163889 \text{ E-13}$
2	2	3	$0.119659 \text{ E-12} - i 0.570084 \text{ E-13}$
2	3	3	$0.457464 \text{ E-13} - i 0.181141 \text{ E-13}$
3	3	3	$0.557081 \text{ E-12} - i 0.374359 \text{ E-12}$

Table 4.4: Tensor components for a massive rank $R = 3$ six-point function; kinematics defined in Table 4.2 and Fig.4.1.

μ	ν	λ	ρ	$F^{\mu\nu\lambda\rho}$
0	0	0	0	0.666615 E-09 + i 0.247562 E-09
0	0	0	1	- 0.200049 E-10 + i 0.294036 E-10
0	0	0	2	0.200975 E-10 - i 0.237333 E-10
0	0	0	3	0.645477 E-10 - i 0.162236 E-09
0	0	1	1	- 0.116956 E-10 - i 0.516760 E-10
0	0	1	2	0.160357 E-11 + i 0.222284 E-11
0	0	1	3	0.792692 E-11 + i 0.729502 E-11
0	0	2	2	- 0.111838 E-10 - i 0.513133 E-10
0	0	2	3	- 0.681086 E-11 - i 0.708933 E-11
0	0	3	3	- 0.804454 E-10 - i 0.801909 E-10
0	1	1	1	0.100498 E-10 - i 0.151735 E-13
0	1	1	2	- 0.348984 E-11 - i 0.195436 E-12
0	1	1	3	- 0.211111 E-10 + i 0.295212 E-11
0	1	2	2	0.357455 E-11 + i 0.662809 E-14
0	1	2	3	0.121595 E-11 - i 0.807388 E-13
0	1	3	3	0.825803 E-11 - i 0.142086 E-11
0	2	2	2	- 0.958961 E-11 - i 0.585948 E-12
0	2	2	3	- 0.209232 E-10 + i 0.289031 E-11
0	2	3	3	- 0.802359 E-11 + i 0.994701 E-12
0	3	3	3	- 0.102576 E-09 + i 0.378476 E-10
1	1	1	1	- 0.246426 E-10 + i 0.276326 E-10
1	1	1	2	0.915670 E-12 - i 0.660629 E-12
1	1	1	3	0.303529 E-11 - i 0.287480 E-11
1	1	2	2	- 0.822697 E-11 + i 0.919635 E-11
1	1	2	3	- 0.116294 E-11 + i 0.100024 E-11
1	1	3	3	- 0.146918 E-10 + i 0.183799 E-10
1	2	2	2	0.908296 E-12 - i 0.654735 E-12
1	2	2	3	0.109510 E-11 - i 0.100875 E-11
1	2	3	3	0.717342 E-12 - i 0.557293 E-12
1	3	3	3	0.450661 E-11 - i 0.485065 E-11
2	2	2	2	- 0.245154 E-10 + i 0.274313 E-10
2	2	2	3	- 0.318500 E-11 + i 0.279750 E-11
2	2	3	3	- 0.146317 E-10 + i 0.182912 E-10
2	3	3	3	- 0.477335 E-11 + i 0.477368 E-11
3	3	3	3	- 0.730168 E-10 + i 0.112865 E-09

Table 4.5: Tensor components for a massive rank $R = 4$ six-point function; kinematics defined in Table 4.2 and Fig.4.1.

5 Applications: five point integrals in QED

We have reached to the final chapter which is devoted to the practical application of the techniques presented in the previous chapters. Here we focus on the NLO QED process $e^+e^- \rightarrow \mu^+\mu^-\gamma$ which involves a set of five point integrals. This process is a more challenging calculation when compared to the $e^+e^- \rightarrow e^+e^-\gamma$ case, due to additional scale (m_μ). However, at least number of diagrams and channels is smaller. What is considered here can be directly applied to the $e^+e^- \rightarrow e^+e^-\gamma$ reaction. We have decided to focus on two important low energy experiments. KLOE [64] which is a detector in accelerator DAΦNE located in Frascati, Italy and BaBar which is located at the SLAC National Accelerator Laboratory (California, USA). The latter ceased operation in 2008, but data analysis is still ongoing.

Low energy physics is presently an important field of activity. Let us mention measurement of $(g - 2)_\mu$ and its theoretical calculation [65], but also low energy hadron physics, e.g. determination of form-factors. In this context, an important quantity is the pion form factor [66]. To describe it properly, experimental data are needed, and for that the process $e^+e^- \rightarrow \mu^+\mu^-\gamma$ serves as normalization reaction [67]:

$$R_{exp} = \frac{\sigma(e^+e^- \rightarrow \pi^+\pi^-\gamma)}{\sigma(e^+e^- \rightarrow \mu^+\mu^-\gamma)}. \quad (5.0.1)$$

To our knowledge so far five point functions have not been included in Monte Carlo generators for the considered process. As an additional motivation, these integrals can be important for measurement of charge asymmetry. As a matter of fact, KLOE measures it. Results are not published so far. There are some discrepancy between data and theoretical calculations [68], that is why, especially at such small energies five point functions are worth considering. For this process interference contributions (i.e. loop

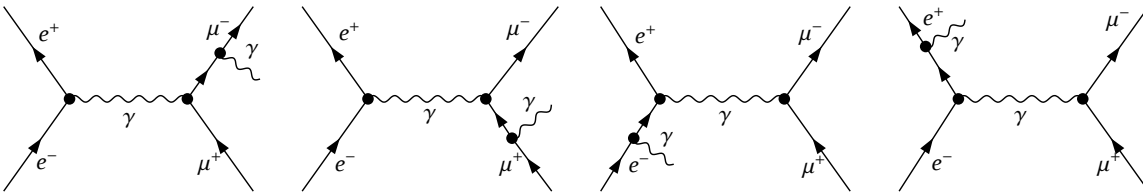


Figure 5.1: FSR and ISR diagrams at the tree level.

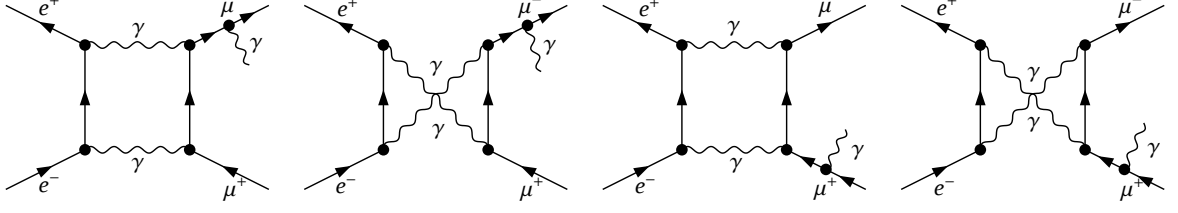


Figure 5.2: FSR four point diagrams.

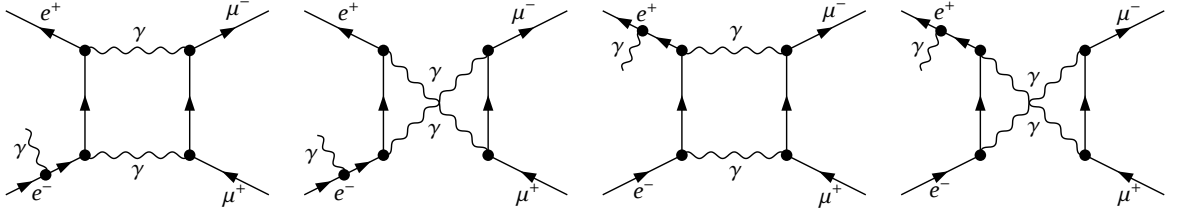


Figure 5.3: ISR four point diagrams.

times tree) were evaluated for both four and five point integrals (final and initial state radiation), see Fig.5.1, Fig.5.2, Fig.5.3 and Fig.5.4. Before any amplitude could be calculated the appropriate diagrams had to be generated. For this step **DIANA** [69] program was used. Then multiplication of loop and tree amplitudes and further processing using trace method was done using symbolic manipulation program **FORM** [70]. The last two steps were made using **MATHEMATICA** and consisted of expressing loop integrals in terms of **LoopTools** tensor coefficients. Here due to number of fermion lines the maximum rank of tensor integrals was three. Final result was transferred then to the **Fortran 77** code. We have decided to use such a code format because all the numerical calculations were performed using **Phokhara-6.0** [71] (a Monte Carlo event generator) which is also written in **Fortran 77**. Our interference amplitudes had been implemented in a separate subroutine, which then was linked with the main program. All these steps were automatised using **Bash** scripting. As an independent check we used **FeynArts** [72] and **FormCalc** [7] to perform the very same calculations independently. The first software generates necessary diagrams with amplitudes and second one is able to do loop times tree computation (here we were also using trace method). Both use **MATHEMATICA** to operate, although in case of **FormCalc** the **FORM** program is used during computations to take advantage of its speed and perform fermion trace calculations. Because of the relation:

$$M_{\text{tree}} M_{\text{loop}}^\dagger + M_{\text{loop}} M_{\text{tree}}^\dagger = 2 \text{Re}(M_{\text{loop}} M_{\text{tree}}^\dagger), \quad (5.0.2)$$

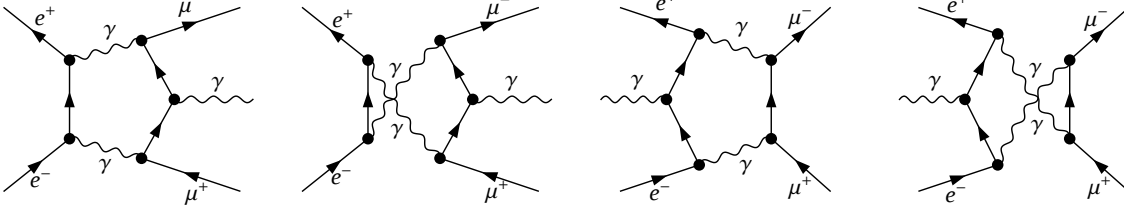


Figure 5.4: FSR and ISR five point diagrams.

only $M_{\text{loop}} M_{\text{tree}}^\dagger$ had to be calculated. From the cross-check reasons, relation Eq.5.0.2 was also checked numerically on computed amplitudes. Further aspects concerning numerical computations will be discussed in the last section. Meanwhile in the next section we are going to focus on infrared divergences which appear in diagrams of Fig.5.2, Fig.5.3 and Fig.5.4.

5.1 Infrared divergences connected with $e^+e^- \rightarrow \mu^+\mu^-\gamma$

In this section we discuss infrared divergences which appear in diagrams that we presented in the introduction of this chapter. All four and five point diagrams we consider here i.e. Fig.5.2, Fig.5.3 and Fig.5.4 are UV finite¹. The only divergences we had to calculate were IR singularities. They appear when there is an appearance of massless line e.g. photon propagator between two on-shell massive lines. If we look at loop diagrams which we want to calculate it is clear how many IR terms we might have. In case of five point integrals two virtual photons generate infrared divergences and in case of four point only one will be involved in IR terms (because the other virtual photon is connected with off shell line).

The main task is to calculate $1/\epsilon$ divergences which has only infrared nature. The procedure of evaluation IR's was the following:

- first Mellin-Barnes representation was derived for a given integral using the **AMBRE** package,
- then **MB** tool was used to regularise M-B representation and perform analytic continuation process,
- finally integrals containing gamma functions (one dimensional integrals) were evaluated using Cauchy theory.

¹ That can be also checked using **FormCalc** which has in-build function called **UVDivergentPart[expr]** which returns UV divergent terms (see e.g. Appendix of [20] or **FormCalc** manual, where UV terms were written).

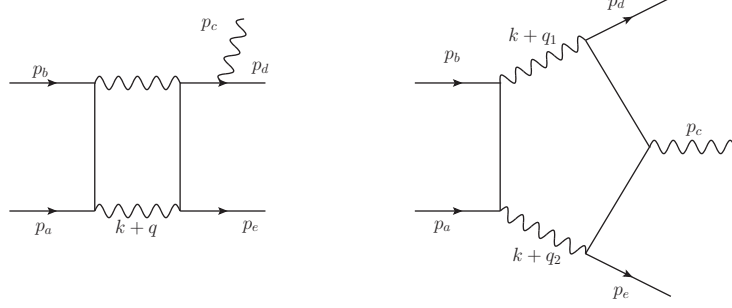


Figure 5.5: Four and five point schematic diagrams, here $p_a + p_b = p_c + p_d + p_e$.

Before we show the final IR result we would like to show explicitly Mellin-Barnes representations as well as form of F polynomials. We limit ourself to scalar four and five point diagram cases here. All the notation refers to the Fig.5.5.

We begin with showing F polynomial for the four point diagram which is of the form:

$$F = m_\mu^2 x_1^2 + \bar{s}_{ae} x_1 x_3 + m_e^2 x_3^2 + \bar{s}_{cd} x_1 x_4 + s_{ab} x_2 x_4, \quad (5.1.1)$$

where

$$s_{ab} = -(p_a + p_b)^2, \bar{s}_{cd} = -(p_c + p_d)^2 + m_\mu^2, \bar{s}_{ae} = -(p_a - p_e)^2 + m_e^2 + m_\mu^2 \quad (5.1.2)$$

are the kinematic variables which should be compared with four point diagram presented in Fig.5.5. Application of Mellin-Barnes apparatus on the above polynomial leads to the following four dimensional M-B representation:

$$\begin{aligned} \text{MB}_{4\text{pt}} &= \int_{c-i\infty}^{c+i\infty} \dots \int_{c-i\infty}^{c+i\infty} dz_1 \dots dz_4 (m_\mu^2)^{z_1} (\bar{s}_{ae})^{z_2} (m_e^2)^{z_3} (\bar{s}_{cd})^{z_4} (s_{ab})^{-2-\epsilon-z_1-z_2-z_3-z_4} \\ &\times \Gamma(-z_1) \Gamma(-z_2) \Gamma(-z_3) \Gamma(-z_4) \Gamma(-1-\epsilon-z_1-z_2-z_3) \Gamma(1+z_2+2z_3) \\ &\times \Gamma(-1-\epsilon-z_1-z_2-z_3-z_4) \Gamma(1+2z_1+z_2+z_4) \\ &\times \Gamma(2+\epsilon+z_1+z_2+z_3+z_4) / \Gamma(-2\epsilon). \end{aligned} \quad (5.1.3)$$

In the very same way we proceed with five point diagrams, firstly one has to calculate F function:

$$F = m_\mu^2 (x_1 + x_5)^2 + \bar{s}_{bd} x_1 x_3 + m_e^2 x_3^2 + s_{ce} x_1 x_4 + s_{ab} x_2 x_4 + s_{cd} x_2 x_5 + \bar{s}_{ae} x_3 x_5, \quad (5.1.4)$$

where again some choice of kinematic variables which optimises F polynomial were introduced:

$$s_{ij} = -(p_i + p_j)^2, \quad \bar{s}_{ij} = -(p_i - p_j)^2 + m_e^2 + m_\mu^2. \quad (5.1.5)$$

The Mellin-Barnes representation for the scalar five point diagram Fig.5.5 is the following:

$$\begin{aligned}
 \text{MB}_{5\text{pt}} &= \int_{c-i\infty}^{c+i\infty} \dots \int_{c-i\infty}^{c+i\infty} dz_1 \dots dz_6 \\
 &\times -((m_\mu^2)^{z_1} (\bar{s}_{bd})^{z_2} (m_e^2)^{z_3} (s_{ce})^{z_4} (s_{ab})^{z_5} (s_{cd})^{z_6} (\bar{s}_{ae})^{-3-\epsilon-z_1-z_2-z_3-z_4-z_5-z_6}) \\
 &\times \Gamma(-z_1)\Gamma(-z_2)\Gamma(-z_3)\Gamma(-z_4)\Gamma(-z_5)\Gamma(-z_6)\Gamma(1+z_2+z_4) \\
 &\times \Gamma(-1-\epsilon+z_1-z_3-z_5)\Gamma(-2-\epsilon-z_1-z_2-z_3-z_4-z_5) \\
 &\times \Gamma(1+z_4+z_5)\Gamma(-2-\epsilon-z_1+z_3-z_4-z_5-z_6)\Gamma(1+z_5+z_6) \\
 &\times \Gamma(3+\epsilon+z_1+z_2+z_3+z_4+z_5+z_6))/ \\
 &\times (\Gamma(-1-2\epsilon)\Gamma(-1-\epsilon-z_1-z_3-z_5))
 \end{aligned} \tag{5.1.6}$$

Because we deal with loop integrals up to rank three (in case of five point functions) and rank two (for four point integrals), $1/\epsilon$ divergences were calculated for integrals with an appropriate tensor numerator as well. Here the ability of **AMBRE** for constructing M-B representations for one-loop integrals with numerator not contracted was used. Finally **MB** package was used to determine divergent parts. This procedure was repeated for all integrals, although the latter is not necessary if one analyses the structure of IR and applies appropriate momenta mapping for Eq.5.1.7 and Eq.5.1.9. In case of tensor integrals with rank two and three there is no IR contribution for terms containing metric tensor.

$$\text{IR}_{4\text{pt}} = \frac{1}{2\epsilon} \frac{\ln \left[\frac{1-x}{1+x} \right]}{s_{ab}\bar{s}_{ae}x} \prod_{i=0}^{m \leq 2} q_{\mu_i}, \quad x = \sqrt{1 - \frac{4m_e^2 m_\mu^2}{\bar{s}_{ae}^2}}, \tag{5.1.7}$$

where

$$s_{ab} = -(p_a + p_b)^2, \quad \bar{s}_{ae} = -(p_a - p_e)^2 + m_e^2 + m_\mu^2. \tag{5.1.8}$$

The index m indicates a rank of an integral, e.g. $m = 0$ stands for a scalar integral, in that case we do not have chords q^μ , which is obvious. In the case of these four point diagrams only one photon line gives rise to IR singularities. On Fig.5.5 it is the line between momenta p_a and p_e . In case of vector, rank two and three integrals the chord q^μ of this line appears in the final IR result.

For the five point integrals infrared divergences come from two photon propagators. This is the reason why IR result is a sum of two terms:

$$\text{IR}_{5\text{pt}} = \frac{1}{2\epsilon} \left(\frac{\ln \left[\frac{1-x}{1+x} \right]}{\bar{s}_{bd}s_{cd}x} \prod_{i=0}^{m \leq 3} q_{1\mu_i} + \frac{\ln \left[\frac{1-y}{1+y} \right]}{\bar{s}_{ae}s_{ce}y} \prod_{i=0}^{m \leq 3} q_{2\mu_i} \right) \frac{1}{s_{ab}}, \tag{5.1.9}$$

where

$$\begin{aligned}
 x &= \sqrt{1 - \frac{4m_e^2 m_\mu^2}{\bar{s}_{bd}^2}}, & y &= \sqrt{1 - \frac{4m_e^2 m_\mu^2}{\bar{s}_{ae}^2}}, \\
 s_{ij} &= -(p_i + p_j)^2, & \bar{s}_{ij} &= -(p_i - p_j)^2 + m_e^2 + m_\mu^2.
 \end{aligned} \tag{5.1.10}$$

Evaluated infrared contributions Eq.5.1.7 and Eq.5.1.9 were checked in the following ways:

- analytic result (Eq.5.1.7 and Eq.5.1.9) was cross-checked among numerical integration using **MB** package (here **MBIntegrate** function discussed in one of the previous chapters was used).
- above results were also verified using **CSectors** package by sector decomposition calculations.
- finally analytic result was transformed to λ dependence² i.e. $A/\epsilon = A \ln \lambda + C$ [73] and in this form IR part was subtracted from four and five point loop integrals leading to invariance on shifting λ parameter during numerical calculation.

Technically the subtraction of IR part was only done for the purpose of cross-checks and was not necessary during KLOE and BaBar numerical simulation, because λ parameter could be set to one in **LoopTools** directly.

In the context of IR divergences we have made also the following test. Five point diagrams were taken as an input to **hexagon** program. Here the big advantage of this software is that it can do the reduction of five point diagrams in a fully analytical way. After the reduction had been done we calculated IR parts both for five (l.h.s) and four point diagrams (r.h.s) and both sides were compared according to:

$$I_5^{\text{IR}} = a_1 I_{4,1}^{\text{IR}} + a_2 I_{4,2}^{\text{IR}} + a_3 I_{4,3}^{\text{IR}} + a_4 I_{4,4}^{\text{IR}}, \quad (5.1.11)$$

where a_i is some coefficient which contains kinematic parameters (see Fig.5.6). It comes from "signed minors" (see e.g. Eq.4.4.2 for a scalar case). This method of checks was also presented in [74] for $e^+e^- \rightarrow e^+e^-\gamma$.

5.2 Results and numerical cross-checks for virtual contributions

In this section we focus on numerical aspects of performed calculations. The infrared parts which we discussed in the previous section are subtracted from the final result. The results presented here should be treated as a basis for the future work in which hard part will be added [75] i.e. our virtual IR will be necessary here. The λ dependent terms will be reintroduced, and only after phase space integration final λ independence will be restored. Here we only touch contributions from virtuality of discussed loop integrals, similar analysis in the context of two-loop Bhabha studies have been made e.g. in [52].

² We are working in the dimensional regularisation but **LoopTools** uses photon mass λ as a regulator. It was checked that in this case $C = 0$. The argumentation is the following. We have IR divergent five point integrals: from **LoopTools** we get E_0 which is λ dependent ($E_0 = \text{LT}(\lambda)$). We also evaluated with **CSectors** and **AMBRE/MB** that $E_0 = A/\epsilon$ which we rewrite as: $E_0 = A \ln(\lambda) = \text{MB}(\lambda)$. Now, if $\text{MB}(\lambda) = \text{LT}(\lambda)$ for any λ , we can say that both constant and divergent part are the same, so $C = 0$.

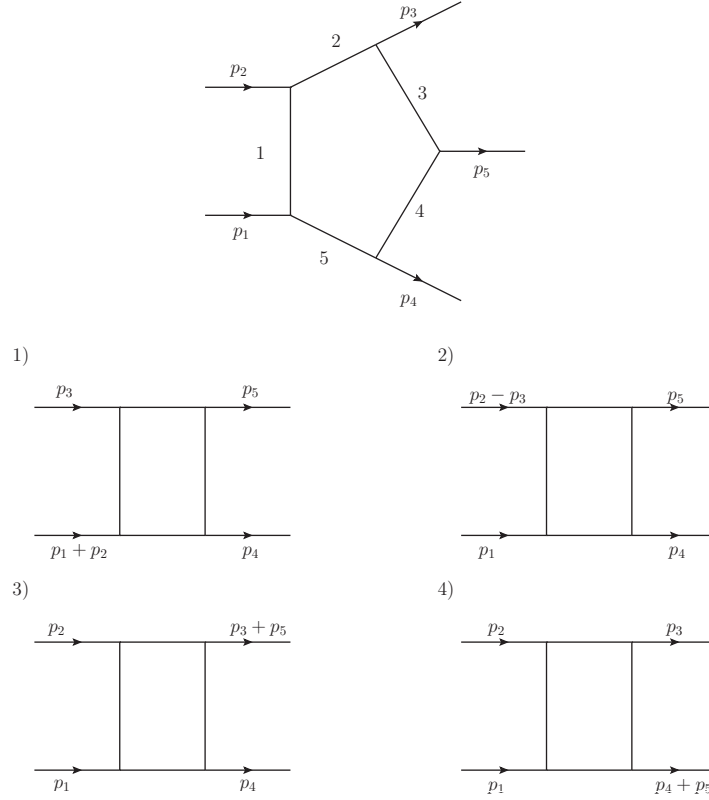


Figure 5.6: Figure refers to the Eq.5.1.11. Here five point diagram was reduced using **hexagon** to the four point functions presented below. Each number i) should be understood as an i -th internal line which has been cut in the pentagon diagram.

The **Phokhara** program comes with a special input file which is used to change behaviour of the program e.g. photon angle range, CMS energy and other necessary cuts and settings. We have chosen cuts so that our numerical calculations simulate behaviour of KLOE and BaBar experiments (see Table 5.1). But before any proper numerical evaluations have been made, several purely technical tests had to be performed. We were interested in examining the numerical stability using different compilers and different real number declarations. In the tests **g77** [76] and **ifort** [77] compilers were used. In the end the second one gave us much better results, so eventually it was chosen later for main calculations. An additional advantage of this compiler is that it has also implementation of floating point numbers in quadrupole precision. At this point we summarise the main difference between two floating point formats and show the number of precision digits one can expect.

- double precision: floating point number is represented by 64 bits (1 sign, 11 expo-

	KLOE	BaBar
E_{CMS}	1.02 GeV	10.56 GeV
Q^2	$0.0447 - 50 \text{ GeV}^2$	$0.0447 - 1.06 \text{ GeV}^2$
$E_{min,\gamma}$	0.02 GeV	3 GeV
θ_μ	$50^\circ - 130^\circ$	$40^\circ - 140^\circ$
θ_γ	$0^\circ - 15^\circ$ and $165^\circ - 180^\circ$	$20^\circ - 138^\circ$

Table 5.1: Parameters used as an input in **Phokhara** event generator. Both sets were chosen to match two experiments KLOE and BaBar. E_{CMS} is a CMS collision energy, Q^2 a squared invariant mass of the muonic system and $E_{min,\gamma}$ is a minimal energy of tagged photon. The angles are measured between direction of a particle and the z axis.

nent and 52 fraction), leading to 15-16 digits of precision.

- quad precision: floating point number is represented by 128 bits (1 sign, 15 exponent and 112 fraction). It means we have approximately 34 digits of precision.

Quad precision was only used within our subroutine and **LoopTools**³ libraries, the rest of **Phokhara** program remained in the standard double precision. We did not want to change the floating numbers declaration everywhere in the well tested software i.e. **Phokhara**, and accidentally generate some unwanted bugs leading to software errors.

Since version 2.2, **LoopTools** allows to choose in which decomposition five point integrals should be calculated. Implementation of both Passarino-Veltman [14] and Denner/Ditmaier [20, 21] schemes exist. Numerical calculations can be switched between these two schemes using so called "version key" [19]. After we had performed numerical tests, we decided to use the latter scheme, as it gave us the best accuracy for five point integrals.

Apart from the numerical tests, several physical checks have been made to ensure that calculated amplitudes are correct. One of it was verification of gauge invariance. It is a well known property which states that some diagrams must cancel among each other when amplitudes are contracted with photon momenta instead of polarisation vectors. In our case, such invariance is presented among tree diagrams and also four and five point loop integrals Fig.5.7. The first case is very simple and was checked analytically and numerically, the second one was checked only numerically. Because we have used trace method (where one uses algebra of gamma traces), the numerical checks of gauge invariance required sum of loop diagrams in Fig.5.7 to be contracted with one of the tree diagrams Fig.5.1.

During the process of preparation of the interference amplitudes (loops times tree) we devoted attention to simplify results as much as possible. Finally we managed to obtain

³ During compilation of these libraries it is possible to compile them with quadrupole precision setting using an appropriate **Makefile** file.

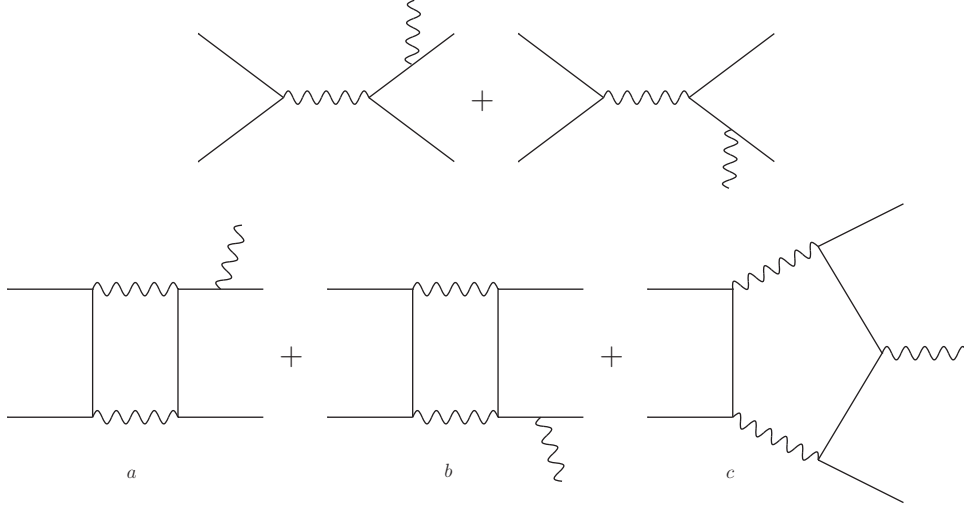


Figure 5.7: FSR gauge invariance between tree diagrams (upper picture), and gauge invariance among four and five point one-loop integrals (below). Here diagrams were limited to FSR cases, the same property is present for ISR amplitudes. It was checked numerically by contracting with appropriate tree diagrams.

	KLOE	BaBar
double precision	10^{-2}	10^{-5}
quadrupole precision	10^{-12}	10^{-10}

Table 5.2: Gauge invariance for loop diagrams Fig.5.7 for KLOE and BaBar setting and different real number declarations. The numbers give relative accuracy defined as $\max\left\{\frac{\sum_{i=a,b,c} \text{Re}(M_{\text{loop}}^i M_{\text{tree}}^\dagger)}{\min(\text{Re}(M_{\text{loop}}^i M_{\text{tree}}^\dagger))}\right\}$. Indices a, b, c refer to a, b, c diagrams in Fig.5.7.

about 200 lines of **Fortran 77** code for a single four point interference amplitude, and about 700 lines for a single pentagon (it is slightly less for gauge invariant amplitudes). Obviously such lengthy expressions might cause numerical instability. That was the reason why we wanted to test calculations in quadrupole precision as well. As it is shown in Table 5.2, numerical results for gauge invariance look much better in quadrupole precision than in double one. This is clear especially in case of KLOE (see Table 5.1), where photon angle is chosen to be around 0° and 180° . We have observed that setting θ_γ out of these values improved results significantly for the numerical gauge invariance results (see BaBar results for comparison). In the figures Fig.5.8, Fig.5.9 and Fig.5.10 we show explicitly which diagrams were used to form interference amplitudes $M_{\text{loop}} M_{\text{tree}}^\dagger$. Note that here we calculate ISR and FSR parts only i.e. we have total number of vertices in tree and loop diagrams odd for every fermion line. The remaining parts would be

INT (an interference among ISR and FSR) amplitudes, where the number of vertices is even. These contributions are important for the charge asymmetry.

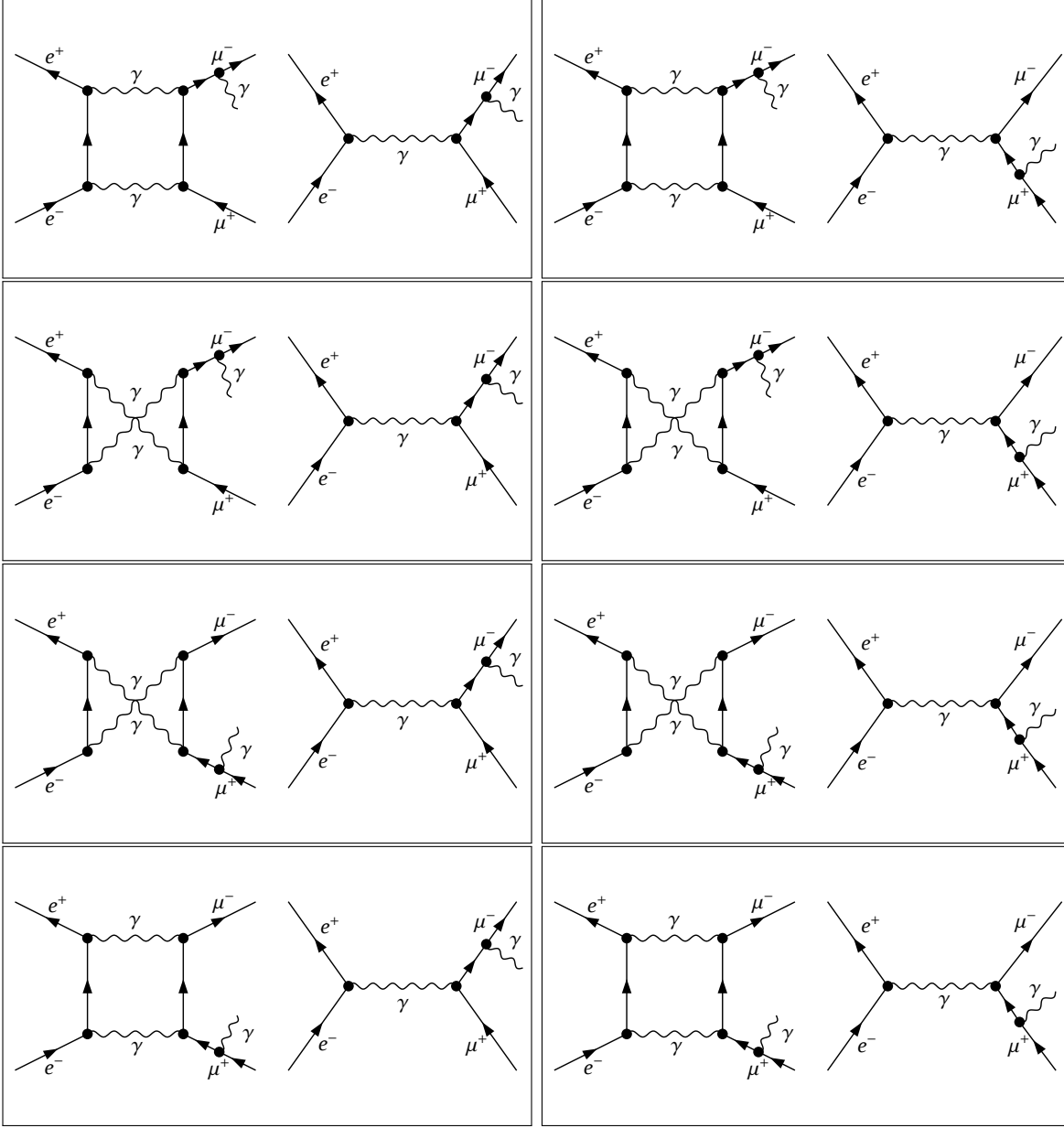


Figure 5.8: Interference amplitudes (loop times tree) involving four point and tree FSR diagrams.

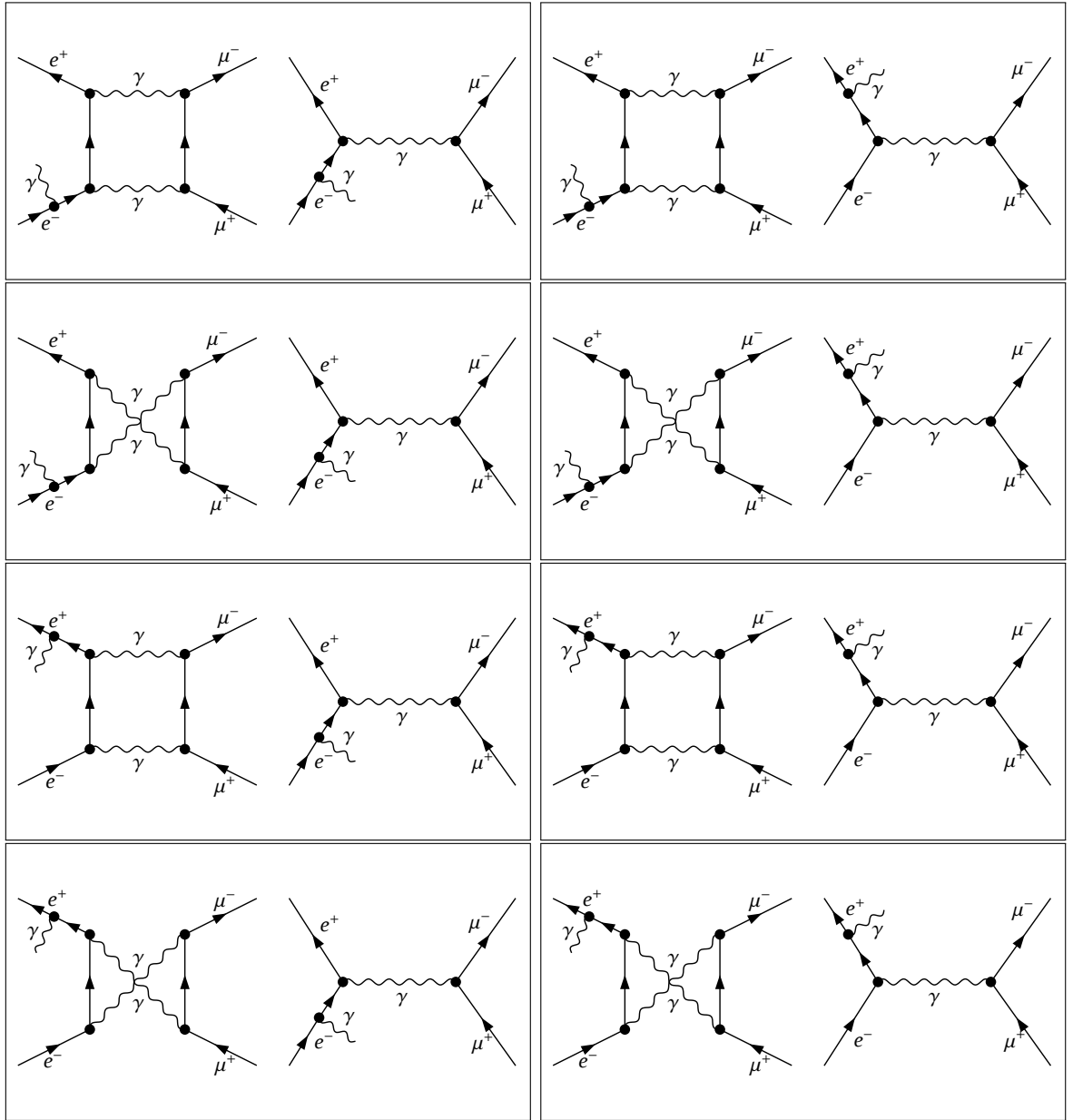


Figure 5.9: Four point one loop and tree diagrams, all ISR. All of them form interference amplitudes.

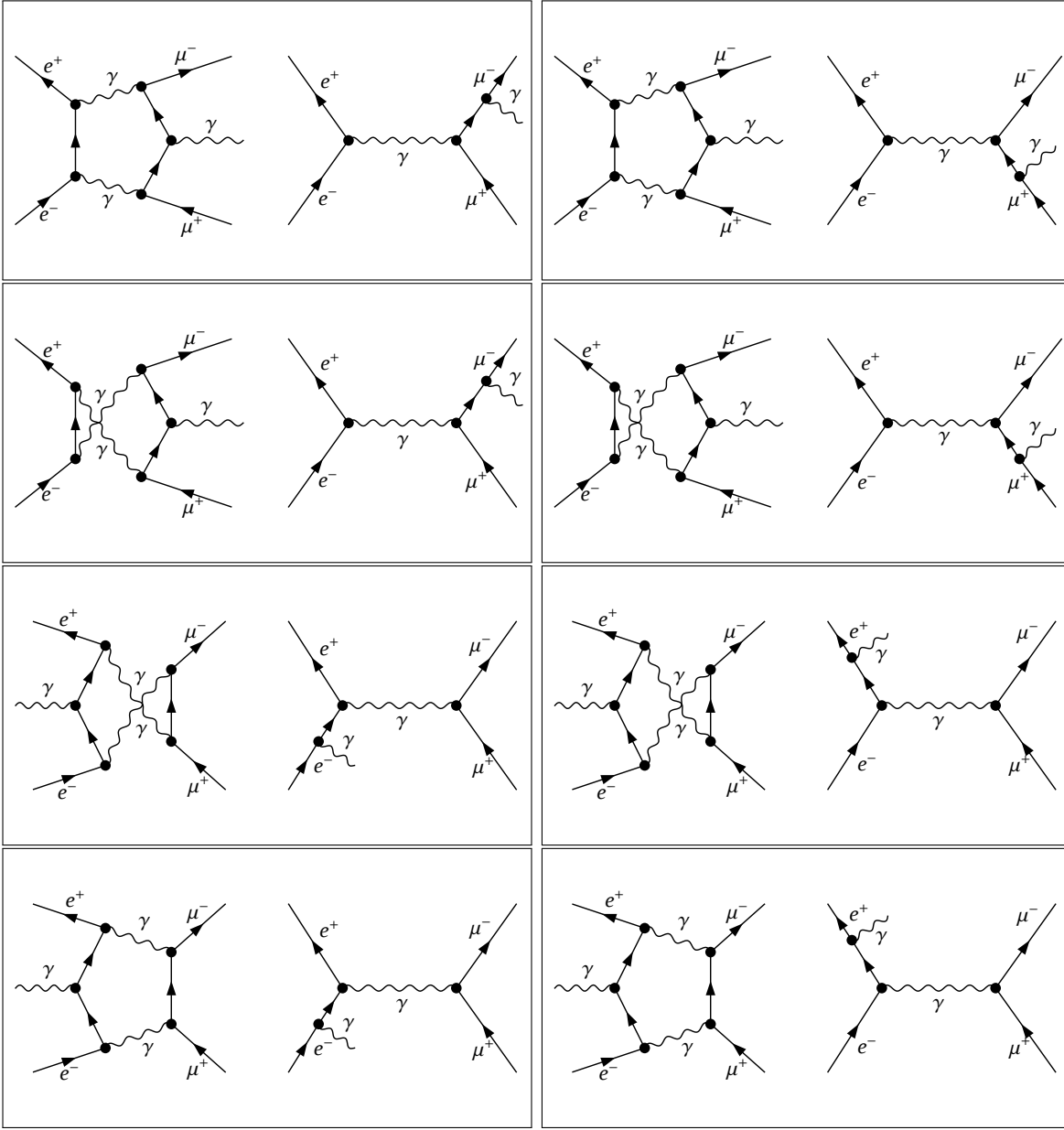


Figure 5.10: Interference amplitudes formed with five point integrals and tree diagrams.

As it was pointed out at the beginning of this chapter, the interference amplitudes were prepared using different softwares. Thereafter both results were cross-checked, that was an additional check for gauge invariance and "normal" numerical calculations. The end of this chapter covers discussion about numerical results for KLOE and BaBar experiments. Some of the numerical results were prepared for tree diagrams only, in this case we used ISR, FSR and INT diagrams which are encoded in *Phokhara*. During

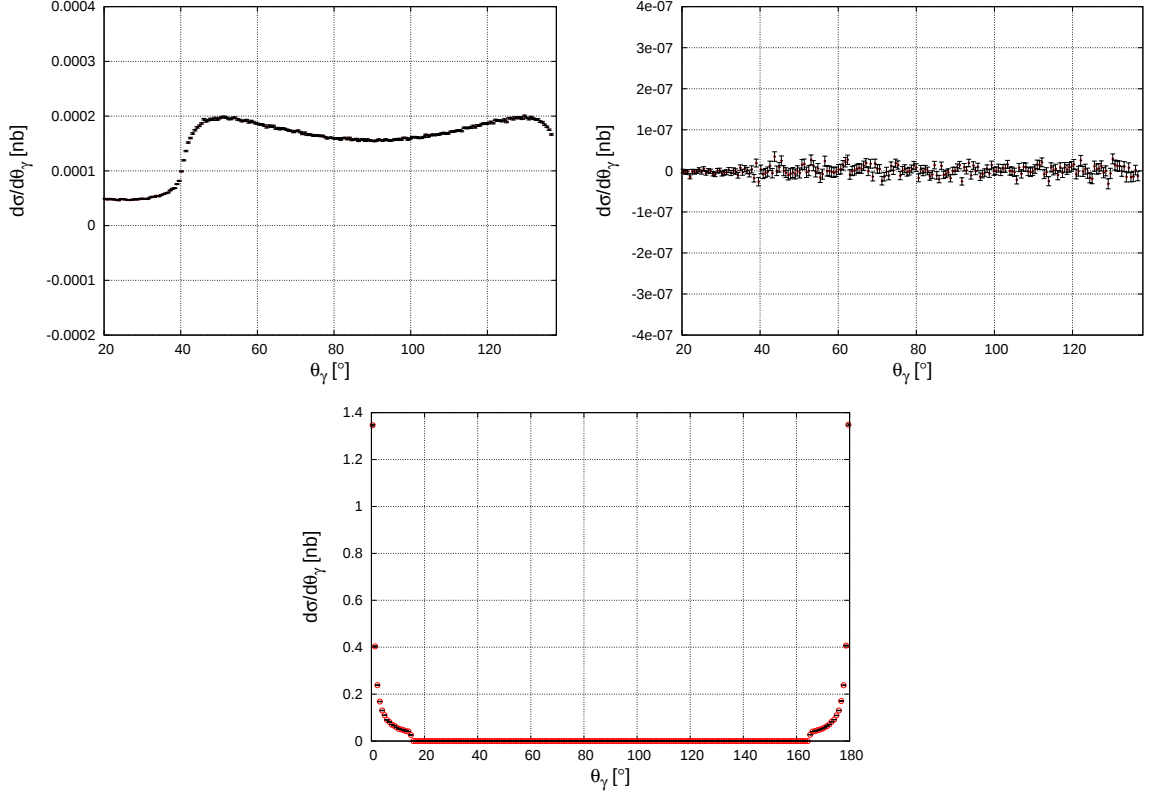


Figure 5.11: Two upper plots show results simulated for BaBar in which the left one presents θ_γ dependence for the complete tree level and the right one is a result for loop contributions based on Fig.5.8, Fig.5.9 and Fig.5.10. The KLOE distribution over photon angle for the complete tree level is presented in the lower plot.

calculations $2.5 \cdot 10^6$ points were generated, from which about $3 \cdot 10^5$ points were accepted within KLOE cuts. For BaBar it was $6 \cdot 10^6$ and $1 \cdot 10^6$ points, respectively.

All the plot points were histogrammed using one of subroutines which is encoded in *Phokhara* program and provides calculation of Monte-Carlo errors. All values i.e. θ_μ , θ_γ and Q^2 were distributed into two hundred bins. We begin our numerical analysis by showing dependence on the photon angle θ_γ Fig.5.11. Plots obtained for BaBar cuts for tree and loop amplitudes are presented there. We see that the θ_γ dependence obtained for the tree is not symmetric due to non symmetric cut which was chosen (see Table 5.1). By comparing the values of tree and virtual corrections we see that the latter are very small compared to the LO ones. The same situation we have observed for the KLOE case but because of symmetric cut we see symmetric dependence. Here we have limited ourselves to the tree contribution plot only (NLO result is again much smaller).

Next we present dependence on θ_μ in Fig.5.12. We see that muon angle dependence

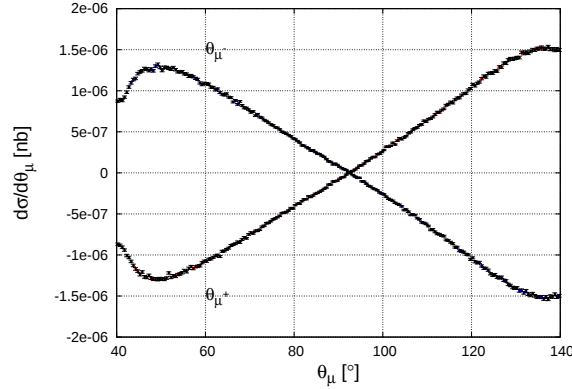


Figure 5.12: Results for BaBar θ_μ angle dependence (NLO contributions based on Fig.5.8-Fig.5.10). Plots were generated both for μ^- and μ^+ .

is antisymmetric between μ^- and μ^+ . This fact is expected but can be interpreted as an additional verification of made calculations.

Finally we present plots showing dependence on the squared invariant mass of the system formed by the muons and the tagged photon. Figures were generated for $\theta_\mu > 90^\circ$ and $\theta_\mu < 90^\circ$, also figure with full range of θ_μ (Table 5.1) were presented for KLOE Fig.5.13 and BaBar Fig.5.14. Careful analysis of these plots reveal that for BaBar plotted dependence goes rapidly to zero for about 47 GeV^2 (that is why the plot is cut at 50 GeV^2), and in case of KLOE the zero value is present up to about 0.5 GeV^2 . That behaviour is dictated by kinematics and implemented cuts. If we had two particles in the final state only, the particles would fly away from the collision point in exactly opposite direction and each of them would carry half of the total energy. Here we have three body problem, so the energetic spectrum for every particle is continuous. If both muons fly away in the same direction the photon would go in opposite direction. Now if we add to that different photon cuts for KLOE and BaBar from Table 5.1, the effect seen on plots Fig.5.13 and Fig.5.14 will appear.

Numerical results for the virtual corrections presented here should be treated as a prelude to the forward backward asymmetry analysis.

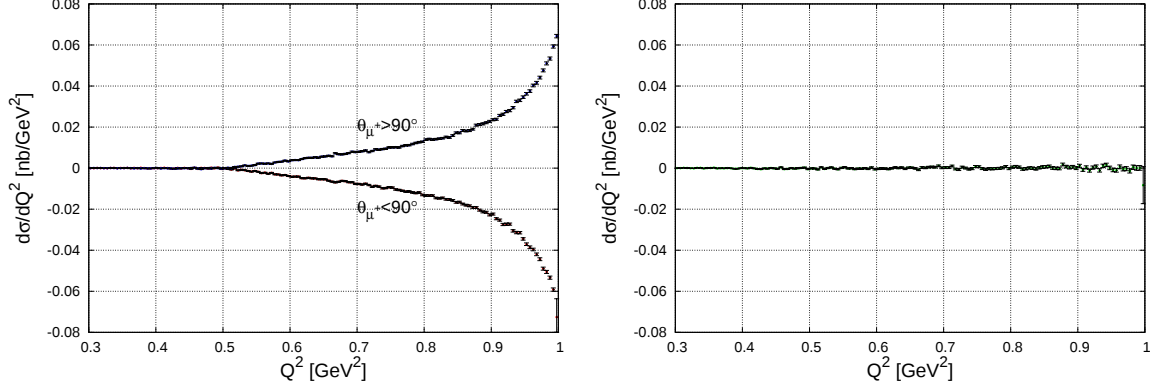


Figure 5.13: KLOE NLO results for $\theta_\mu > 90^\circ$ and $\theta_\mu < 90^\circ$ (left plot). Q^2 dependence for the full θ_μ range allowed by cuts in Table 5.1 (right plot), we see that points subtract to zero. We have chosen θ_μ connected with μ^+ .

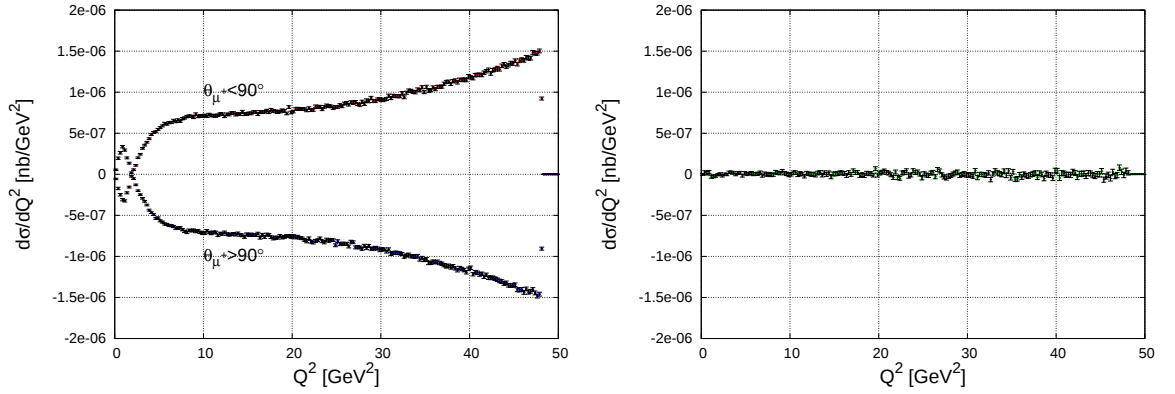


Figure 5.14: NLO results for BaBar for $\theta_\mu > 90^\circ$ and $\theta_\mu < 90^\circ$. If we plot Q^2 dependence for the full θ_μ range we see that points subtract to zero (right plot). We have chosen θ_μ for the μ^+ .

6 Summary and conclusions

In spite of the amount of contents showed within this thesis, we have been able to cover only a part of modern and constantly developing loop techniques. One of the most important projects that we have done was a research within the Mellin-Barnes method, which started during calculations of the massive two-loop Bhabha scattering [78] and ended by developing a very valuable tool **AMBRE**. From the beginning sector decomposition algorithm was used as a method for doing cross-checks with Mellin-Barnes calculations. Here a helpful program was written, **CSectors** which soon is planned to be publicly available. Another tool **hexagon** was developed during research work on reduction scheme of six and five point integrals.

We have used all these techniques to work out NLO virtual corrections involving five and four point one-loop diagrams in the process $e^+e^- \rightarrow \mu^+\mu^-\gamma$. Infrared divergences have been computed using **AMBRE** and **MB**. As an independent check, **CSectors** program was used. Reduction using **hexagon** was very helpful and allowed us to additionally check IRs. As far as numerical calculations are concerned, we have shown λ independent virtual contributions to the $e^+e^- \rightarrow \mu^+\mu^-\gamma$ process. Here we have chosen to use the QED process involving muons in the final state. But there are no circumstances to use exactly the same procedure to the Bhabha $e^+e^- \rightarrow e^+e^-\gamma$ reaction. The same procedure can be also used for calculations within the QCD theory.

It would be interesting to repeat all the numerical calculations with the usage of other than **LoopTools** tools e.g. **QCDloop** or even more modern methods i.e. involving OPP or unitary cuts. Another future perspective point is the usage of helicity method for preparation of amplitudes.

Our results should be treated as the first benchmark for the further development. Virtual corrections considered here are not enough to judge on experimental observables Fig.5.11-Fig.5.14. Adding real corrections may change results substantially.

A Gamma function

The basic function of Mellin-Barnes representation is a special function called Gamma function Fig.A.1. One of possible definitions [79] is the following one ($Re(x) > 0$):

$$\Gamma(x) = \int_0^\infty e^{-t} t^{x-1} dt. \quad (A.0.1)$$

The Gamma function is an extension of the factorial function to real and complex numbers. It also fulfils the function equation:

$$\Gamma(x+1) = x\Gamma(x). \quad (A.0.2)$$

Eq.A.0.2 can be used with $\Gamma(n+1)$, where $n \in \mathbb{N}$ and $n > 0$. Then the following is true:

$$\Gamma(n+1) = n\Gamma(n) = n(n-1)\Gamma(n-1) = \dots = n!\Gamma(1) = n!, \quad (A.0.3)$$

clearly showing connection to the factorial function. It is very important to note that Gamma function $\Gamma(x)$ has poles located on the negative real axis at $x = 0, -1, -2, -3, \dots$ as seen on Fig.A.1. In other words for $\Gamma(z)$, $z \in \mathbb{C}$ it is analytic everywhere except points

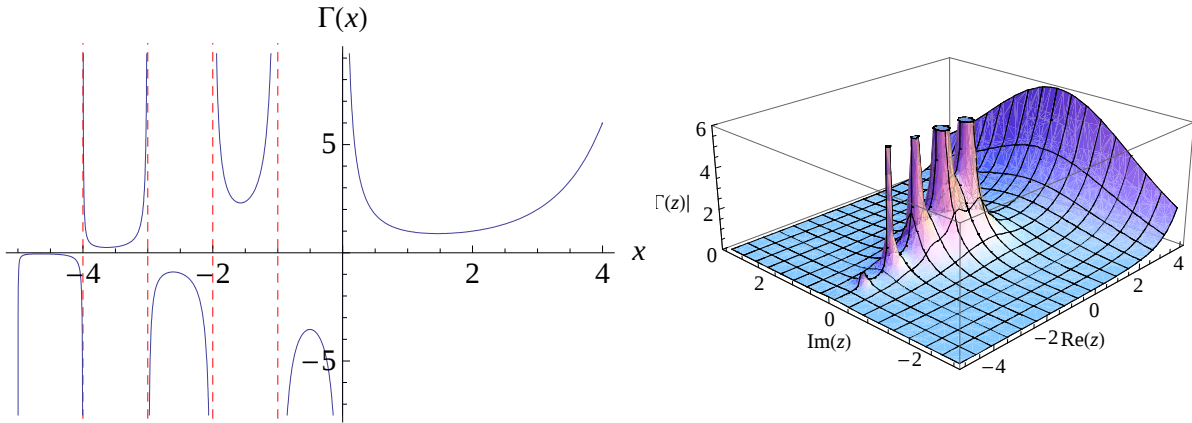


Figure A.1: Gamma function $\Gamma(x)$ plot for x -real (left plot) and $\Gamma(z)$ for z -complex (right plot).

$z = 0, -1, -2, -3, \dots$ and the residue at $z = -k$ is:

$$Res(\Gamma, -k) = \frac{(-1)^k}{k!}. \quad (A.0.4)$$

B Software

B.1 AMBRE and generation of Mellin-Barnes representations

Recently Mellin-Barnes (M-B) representations of Feynman integrals have been used extensively in various phenomenological and theoretical studies of quantum field theory. In order to automatize process of creating such representations, special (semi-) automatic **MATHEMATICA** program was developed [12]. The **AMBRE**¹ stands for Automatic Mellin-Barnes Representation. It can be used to construct planar Mellin-Barnes representations for:

- scalar multi-loop, multi-legs integrals
- tensor m -rank one-loop integrals

```
Fullintegral[numerator, propagators, internal momenta]
```

Figure B.1: The input function of **AMBRE** package.

The arguments are as follows:

- **numerator**: numerator which must be given in the scalar form, see also Fig.B.2.
- **propagators**: product of propagators of the form $\text{PR}[\mathbf{q}, \mathbf{m}, \mathbf{n1}] \equiv (q^2 - m^2)^{-n1}$.
- **internal momenta**: list of internal momenta. The order of internal momenta in the list must be taken with utmost attention. By changing order in this list, one forces **AMBRE** to calculate sub-loops in user defined custom way.

```
{k1*p2, k1*p2, k1*k1}
```

Figure B.2: Proper way of introducing $(k_1 \cdot p_2)^2 k_1 \cdot k_1$ numerator in the **AMBRE**.

Another two functions provided by this package which are necessary to be able to construct Mellin-Barnes representations are: **IntPart** and **SubLoop** presented on Fig.B.3.

First of the above functions prepares a sub-loop of the full integral by collecting all propagators which carry a given loop momentum k_i . It will display a piece of the full integral with:

¹ It can be downloaded from <http://projects.hepforge.org/mbtools/>, or directly from <http://prac.us.edu.pl/~gluza/ambre/>, which also contains example files.

```
IntPart[iteration, options]
...
SubLoop[integral, options]
```

Figure B.3: Proper way of using `IntPart` and `SubLoop` functions.

- the numerator associated with the given sub-loop
- sub-loop for a given internal momentum
- internal momentum for which **AMBRE** will integrate the sub-loop

It must be stressed that execution of `IntPart[iteration]` function proceeds in the order `IntPart[1]`, `IntPart[2]`, then `IntPart[3]` and so on. If there is a need to change the ordering of integrations, one has to change the order in the starting list of internal momenta. Once again compare Fig.B.1 and its description. Inserting e.g. `IntPart[2]` before `IntPart[1]` would not be a proper way to do this. In the output of `IntPart[iteration]`, the so called **MATHEMATICA** tag message will be displayed:

```
Fauto::mode: U and F polynomials will be calculated
in AUTO mode. In order to use MANUAL mode execute Fauto[0].
```

Figure B.4: The tag message giving information, that F -polynomial modification is possible.

By running `Fauto[0]`, **AMBRE** will calculate the F -polynomial (with name **fupc**) for a given sub-loop. At this stage, a user might wish to modify **fupc** manually, e.g. by applying some changes in kinematics.

The second function presented on B.3 is `SubLoop`. It is a basic function used for deriving Mellin-Barnes representation for a given sub-loop. This function takes output generated by `IntPart` and performs the following calculations:

- 1) calculates the F -polynomial for the sub-loop (only if `Fauto[0]` was not set)
- 2) determines the M-B representation for the F -polynomial
- 3) integrates over Feynman parameters

It must be executed always with argument `integral`. As a result, the Mellin-Barnes representation for a given sub-loop integral will be displayed. In multi-loop calculations one will notice additional propagators (marked in red in the output of **AMBRE**) which appear from the intermediate F -polynomial (and are added into propagators of next sub-loop).

The last optional argument in both functions allows to insert additional option, the possibilities are:

- **Text** -> True/False: displays or not information text
- **Result** -> True/False:
- **Xintegration** -> True/False: performs or not integration over Feynman parameters

It is possible to apply Barnes lemmas Eq.2.2.11 and Eq.2.2.12 on obtained Mellin-Barnes representations using **BarnesLemma** function Fig.B.5.

BarnesLemma[M-B representation, number, options]

Figure B.5: Function used to apply Barnes lemmas.

The meaning of arguments is following:

- **M-B representation**: a Mellin-Barnes representation obtained after execution of **SubLoop**.
- **number**: accepts two possible values, 1 for first Barnes lemma and 2 for second one.
- **options**: the following additional options.
 - **Text** -> True/False: displays or not information text.
 - **Shifts** -> True/False: searches for the pairs of two integration variables $z_i + z_j$ and $z_i - z_j$ which, after application of the appropriate shift one of it is cancelled.

The following pedagogical example Fig.B.6 shows usage of **AMBRE** software:

```

In[1]:= <<AMBRE.m
        by K.Kajda    ver: 1.2
        last modified 9 Apr 2008
        last executed on 09.04.2009 at 19:44

In[2]:= invariants = {p^2 -> s};
        Fullintegral[{1}, {PR[k, m, n1] PR[k + p, m, n2]}, {k}];

In[3]:= IntPart[1]
        numerator=1
        integral=PR[k,m,n1] PR[k+p,m,n2]
        momentum=k
        Fauto::mode: U and F polynomials will be calculated in
        AUTO mode. In order to use MANUAL mode execute Fauto[0].

In[4]:= MBrepr = SubLoop[integral];
        Iteration nr1: >>Integrating over k<<
        Computing U & F polynomial in AUTO mode >>Fauto[1]<<
        U polynomial...
        X[1]+X[2]
        F polynomial...
        m^2 FX[X[1]+X[2]]^2-s X[1] X[2]

        Final representation:
        ((-1)^(n1+n2) (m^2)^z1 (-s)^(2-eps-n1-n2-z1) Gamma[-z1]
        Gamma[-2+eps+n1+n2+z1] Gamma[2-eps-n1+z1-z2] Gamma[-z2]
        Gamma[-2 z1+z2] Gamma[2-eps-n2-z1+z2])/(Gamma[n1]
        Gamma[4-2 eps-n1-n2] Gamma[n2] Gamma[-2 z1])

In[5]:= BarnesLemma[MBrepr /. {n1 -> 1, n2 -> 1}, 1]
        >> Barnes 1st Lemma will be checked for: {z2} <<
        Starting with dim=2 representation...

        1. Checking z2...Barnes Lemma was applied.
        >> Representation after 1st Barnes Lemma: <<
        1st Barnes Lemma was applied for: {z2}
        Obtained representation has: dim=1

        ((m^2)^z1 (-s)^(-eps - z1) Gamma[1 - eps - z1]^2
        Gamma[-z1] Gamma[eps + z1])/Gamma[2 - 2 eps - 2 z1]

```

Figure B.6: Scalar one loop self-energy example.

B.2 Reduction of six and five point integrals in *hexagon*

The *hexagon*² is *MATHEMATICA* package for reducing six point Feynman integrals. Additionally, package allows to obtain results for pentagons, as they are connected with hexagons. Results that can be obtained are listed below:

- six point functions up to rank four.
- five point functions up to rank three

These tensor ranks are sufficient to get results within QED model. Output can be numerical or analytical depending on user needs and is presented in a basis formed from metric tensor $g_{\mu\nu}$ and the chords q_i^μ .

$$\begin{aligned}
 F^\mu &= \sum_{i=1}^5 q_i^\mu F_i, \\
 F^{\mu\nu} &= \sum_{i,j=1}^5 q_i^\mu q_j^\nu F_{ij}, \\
 F^{\mu\nu\lambda} &= \sum_{i,j,k=1}^5 q_i^\mu q_j^\nu q_k^\lambda F_{ijk} + \sum_{i=1}^5 g^{\mu\nu} q_i^\lambda F_{00i}, \\
 F^{\mu\nu\lambda\rho} &= \sum_{i,j,k,l=1}^5 q_i^\mu q_j^\nu q_k^\lambda q_l^\rho F_{ijkl} + \sum_{i,j=1}^5 q_i^\mu q_j^{[\nu} g^{\lambda\rho]} F_{00ij}, \\
 E^\mu &= \sum_{i=1}^4 q_i^\mu E_i, \\
 E^{\mu\nu} &= \sum_{i,j=1}^4 q_i^\mu q_j^\nu E_{ij} + g^{\mu\nu} E_{00}, \\
 E^{\mu\nu\lambda} &= \sum_{i,j,k=1}^4 q_i^\mu q_j^\nu q_k^\lambda E_{ijk} + \sum_{i=1}^4 g^{[\mu\nu} q_i^{\lambda]} E_{00i},
 \end{aligned} \tag{B.2.1}$$

Kinematics that was used inside this package is presented in the following diagrams Fig. B.7.

This package is able to output full result of tensor integrals, list of all coefficients or a specific coefficient for a given rank. Result can be numerical or analytical depending on user input.

To obtain full result for scalar six point diagram, the following function must be executed inside *MATHEMATICA* environment:

$$\text{RedF0}[p_1^2, \dots, p_6^2, s_{12}, s_{23}, s_{34}, s_{45}, s_{56}, s_{16}, s_{123}, s_{234}, s_{345}, m_1^2, \dots, m_6^2] \tag{B.2.2}$$

² It is available here: <http://prac.us.edu.pl/~gluza/hexagon/>

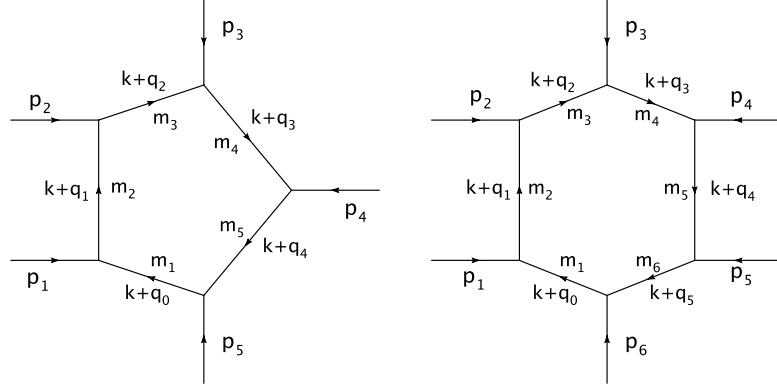


Figure B.7: Momenta flow used in the package in six and five point diagrams.

where $s_{i\dots j} = (p_i + \dots + p_j)^2$. Results for higher ranks (one,two,three and four) can be obtained by `RedF1`, `RedF2`, `RedF3`, `RedF4` respectively, with the same arguments as for `RedF0`. Results for five point function will be generated using:

$$\text{RedE0}[p_1^2, \dots, p_5^2, s_{12}, s_{23}, s_{34}, s_{45}, s_{15}, m_1^2, \dots, m_5^2] \quad (\text{B.2.3})$$

and `RedE1`, `RedE2`, `RedE3`. List of all coefficients for a given rank in case of six point function will be printed after executing:

$$\text{RedFget}[\text{rank}, p_1^2, \dots, p_6^2, s_{12}, s_{23}, s_{34}, s_{45}, s_{56}, s_{16}, s_{123}, s_{234}, s_{345}, m_1^2, \dots, m_6^2] \quad (\text{B.2.4})$$

Here argument `rank` must be replaced by one of these: `rank0`, ..., `rank4`. Additionally `rankALL` displays all coefficients. In similar way, list of coefficients of five point functions can be displayed using `RedEget`. Package can also produce specific tensor coefficient for a given rank:

$$\text{RedFcoef}[\text{coef}, p_1^2, \dots, p_6^2, s_{12}, s_{23}, s_{34}, s_{45}, s_{56}, s_{16}, s_{123}, s_{234}, s_{345}, m_1^2, \dots, m_6^2] \quad (\text{B.2.5})$$

Needed coefficient must be inserted instead of `coef` e.x.: `ff00`. For five point function one has to call function `RedEcoef`.

List of hexagon functions

- `RedF0[arguments]`, `RedF1[arguments]`, `RedF2[arguments]`, `RedF3[arguments]`, `RedF4[arguments]` - displays full result for six point functions: scalar, vector, tensor 2nd order, tensor 3rd order and tensor 4th order respectively
- `RedE0[arguments]`, `RedE1[arguments]`, `RedE2[arguments]`, `RedE3[arguments]` - displays full result for five point functions: scalar, vector, tensor 2nd order, tensor 3rd order

- `RedFget[rank,arguments]` - prints list of rules of all coefficients for a given rank of six point function. Possible options for rank are the following: `rank0`, `rank1`, `rank2`, `rank3`, `rank4`, `rankALL`
- `RedEget[rank,arguments]` - prints list of rules of all coefficients for a given rank of five point function. Possible options for rank are the following: `rank0`, `rank1`, `rank2`, `rank3`, `rankALL`
- `RedFcoef[coef,arguments]` - gives specific tensor coefficient of six point function
- `RedEcoef[coef,arguments]` - gives specific tensor coefficient of five point function

B.3 Sector decomposition with CSectors

The `CSectors`[22, 45] is a `MATHEMATICA` package, which uses `sector decomposition` libraries [44] for numerical calculation of L -loop m -rank tensor integrals in d dimensions using sector decomposition method [41, 42, 23, 24]. It significantly simplifies generation of numerical result for given loop integral, even with tensor numerator. The basic function of this package `DoSectors`. All arguments of this function are presented on Fig.B.8.

```
DoSectors[numerator, propagators, internal momenta, options][min, max]
```

Figure B.8: The basic function of `CSectors` package.

The arguments of `DoSectors` are the following:

- **numerator**: numerator which must be given in the scalar form, see also Fig.B.9.
- **propagators**: product of propagators of the form $\text{PR}[\mathbf{q}, \mathbf{m}, \mathbf{n1}] \equiv (q^2 - m^2)^{n_1}$.
- **internal momenta**: list of internal momenta.
- **options**: this package allows to modify its behaviour by adding additional options. The use of this options is identical to the usage of options in the `MATHEMATICA` environment. List of additional functions is as follows:
 - **SetStrategy**: sets one of strategies available in sector decomposition program [44].
 - **SourceName**: prefix for source, binary and log files It just makes possible to choose any name for the files connected with calculation of a given integral.
 - **TempFileDelete**: by default it is set to `TempFileDelete->True`. When set to `False`, it does not delete `c++` source and binary files as well as log file.
 - **LogFile**: forces package to create or not log file, in which numerical results for given integral and epsilon term are stored.
 - **ShowErrors**: controls whether errors of numerical calculation will be displayed or not in the end of calculation. Errors are calculated using `sector decomposition` function `res.get_error()` [44].
 - **IterationsLow**, **IterationsHigh**, **CallsLow**, **CallsHigh**: are Monte-Carlo parameters, more detailed description can be found in [44].
 - **compiler**: allows to choose other compiler than the default, which is `g++`.
 - **cppflags**, **libs**: contain paths to header and library files required by `sector decomposition` program.

Default values of options used by `CSectors` can also be displayed by executing: `Options[DoSectors]`.

- `min`, `max`: indicate minimum and maximum Laurent series expansion.

`{k1*p2,k1*p2,k1*k2}`

Figure B.9: Proper way of introducing $(k_1 \cdot p_2)^2 k_1 \cdot k_2$ numerator in the **CSectors**.

The **CSectors** computation might be described by the following steps:

1. Main function of this package **DoSectors** is executed in **MATHEMATICA** environment. It causes to prepare all the necessary input for later evaluation: U and F polynomials, as well as numerator tensor structure if present.
2. Then routine generates **c++** source codes to be linked with **sector decomposition** libraries. When this is done, all the generated source code files are automatically compiled.
3. All the binaries are executed in sequence, and the intermediate results for the single ϵ terms (including the errors) are stored in `*.log` files.
4. Finally the results are transferred back to **MATHEMATICA** where are outputted. The overall error for the result is:

$$\Delta I = \sqrt{\sum_{i=1}^N (\Delta I_i)^2}. \quad (\text{B.3.1})$$

A simple scalar one loop self-energy example presented on Fig.B.10 shows usage of **CSectors** package. Obtained output is presented in the list form, where first element is a numerical result followed by sub list of errors calculated for the appropriate epsilon term.

```
In[1]:= <<CSectors.m
        by K.Kajda ver:0.9
        last modified 12 feb 2009

In[2]:= n1 = 1; n2 = 1; n3 = 1;
        m = 1; s = -3;
        invariants = {p^2 -> s};
        DoSectors[{1}, {PR[k, m, n1] PR[k + p, m, n2]}, {k}][-4, 0]
        Using strategy C
        U & F polynomials:
          U = x1+x2
          F = x1^2+5 x1 x2+x2^2
        Generating c++ source...Int 1...done
        Compiling source code...Int 1...done
        Running binary file.....Int 1...done
Out[3]= {-0.393325 + eps^(-1), {5.65622*^-7, 1.34887*^-6/eps}}
```

Figure B.10: Scalar one loop self-energy example.

Bibliography

- [1] **Particle Data Group** Collaboration, C. Amsler *et al.*, “Review of particle physics,” *Phys. Lett.* **B667** (2008) 1.
- [2] S. Weinberg, “A Model of Leptons,” *Phys. Rev. Lett.* **19** (1967) 1264–1266.
- [3] S. Salam *ed.* *N. Svartholm (Almqvist and Wiksells, Stockholm)* (1968) p.367. in Elementary Particle Physics.
- [4] **CDF** Collaboration, T. Aaltonen *et al.*, “Observation of the Ω_b^- and Measurement of the Properties of the ξ_b^- and Ω_b^- ,” [arXiv:0905.3123 \[hep-ex\]](#).
- [5] V. A. Smirnov, “Evaluating Feynman integrals,” *Springer Tracts Mod. Phys.* **211** (2004) 1–244.
- [6] G. J. van Oldenborgh and J. A. M. Vermaseren, “New Algorithms for One Loop Integrals,” *Z. Phys.* **C46** (1990) 425–438.
- [7] T. Hahn and M. Perez-Victoria, “Automatized one-loop calculations in four and D dimensions,” *Comput. Phys. Commun.* **118** (1999) 153–165, [arXiv:hep-ph/9807565](#).
- [8] G. Ossola, C. G. Papadopoulos, and R. Pittau, “CutTools: a program implementing the OPP reduction method to compute one-loop amplitudes,” *JHEP* **03** (2008) 042, [arXiv:0711.3596 \[hep-ph\]](#).
- [9] G. Ossola, C. G. Papadopoulos, and R. Pittau, “Reducing full one-loop amplitudes to scalar integrals at the integrand level,” *Nucl. Phys.* **B763** (2007) 147–169, [arXiv:hep-ph/0609007](#).
- [10] C. F. Berger *et al.*, “An Automated Implementation of On-Shell Methods for One-Loop Amplitudes,” *Phys. Rev.* **D78** (2008) 036003, [arXiv:0803.4180 \[hep-ph\]](#).
- [11] Z. Bern, L. J. Dixon, D. C. Dunbar, and D. A. Kosower, “One-Loop n-Point Gauge Theory Amplitudes, Unitarity and Collinear Limits,” *Nucl. Phys.* **B425** (1994) 217–260, [arXiv:hep-ph/9403226](#).
- [12] J. Gluza, K. Kajda, and T. Riemann, “AMBRE - a Mathematica package for the construction of Mellin-Barnes representations for Feynman integrals,” *Comput. Phys. Commun.* **177** (2007) 879–893, [arXiv:0704.2423 \[hep-ph\]](#).

- [13] M. Czakon, “Automatized analytic continuation of Mellin-Barnes integrals,” *Comput. Phys. Commun.* **175** (2006) 559–571, [arXiv:hep-ph/0511200](#).
- [14] G. Passarino and M. J. G. Veltman, “One Loop Corrections for e^+e^- Annihilation Into $\mu^+\mu^-$ in the Weinberg Model,” *Nucl. Phys.* **B160** (1979) 151.
- [15] T. Diaconidis, J. Fleischer, J. Gluza, K. Kajda, T. Riemann, and J. Tausk, “On the tensor reduction of one-loop pentagons and hexagons,” *Nucl. Phys. Proc. Suppl.* **183** (2008) 109–115, [arXiv:0807.2984 \[hep-ph\]](#).
- [16] T. Diaconidis, J. Fleischer, J. Gluza, K. Kajda, T. Riemann, and J. Tausk, “A complete reduction of one-loop tensor 5- and 6-point integrals,” *Phys. Rev.* **D80** (2009) 036003, [arXiv:0812.2134 \[hep-ph\]](#).
- [17] J. Fleischer, F. Jegerlehner, and O. V. Tarasov, “Algebraic reduction of one-loop Feynman graph amplitudes,” *Nucl. Phys.* **B566** (2000) 423–440, [arXiv:hep-ph/9907327](#).
- [18] “Wolfram MATHEMATICA,” <http://www.wolfram.com/>.
- [19] T. Hahn, “New Features in FormCalc and LoopTools,” *talk at EuroGDR 2005 available online:* , <http://www.feynarts.de/looptools/eurogdr05.pdf>.
- [20] A. Denner and S. Dittmaier, “Reduction of one-loop tensor 5-point integrals,” *Nucl. Phys.* **B658** (2003) 175–202, [arXiv:hep-ph/0212259](#).
- [21] A. Denner and S. Dittmaier, “Reduction schemes for one-loop tensor integrals,” *Nucl. Phys.* **B734** (2006) 62–115, [arXiv:hep-ph/0509141](#).
- [22] J. Gluza, K. Kajda, T. Riemann, and V. Yundin, “New results for loop integrals: AMBRE, CSectors, hexagon,” [arXiv:0902.4830 \[hep-ph\]](#).
- [23] T. Binoth and G. Heinrich, “An automatized algorithm to compute infrared divergent multi-loop integrals,” *Nucl. Phys.* **B585** (2000) 741–759, [arXiv:hep-ph/0004013](#).
- [24] T. Binoth and G. Heinrich, “Numerical evaluation of multi-loop integrals by sector decomposition,” *Nucl. Phys.* **B680** (2004) 375–388, [arXiv:hep-ph/0305234](#).
- [25] G. Heinrich, “Sector Decomposition,” *Int. J. Mod. Phys.* **A23** (2008) 1457–1486, [arXiv:0803.4177 \[hep-ph\]](#).
- [26] N. Nakanishi, “Graph Theory and Feynman Integrals,” *Gordon and Breach, New York* (1971) .
- [27] O. Zavialov, “Renormalized Quantum Field Theory,” *Kluwer* (1990) .

-
- [28] O. V. Tarasov, “Connection between Feynman integrals having different values of the space-time dimension,” *Phys. Rev.* **D54** (1996) 6479–6490, [arXiv:hep-th/9606018](#).
- [29] J. D. Bjorken and S. D. Drell, “Relativistic quantum mechanics,” *New York : McGraw-Hill* **070054932** (1964) .
- [30] R. B. Paris and D. Kaminski, “Asymptotics and Mellin-Barnes Integrals,” *Encyclopedia of Mathematics and its Applications* 85 .
- [31] M. Czakon, J. Gluza, and T. Riemann, “The planar four-point master integrals for massive two- loop Bhabha scattering,” *Nucl. Phys.* **B751** (2006) 1–17, [arXiv:hep-ph/0604101](#).
- [32] J. B. Tausk, “Non-planar massless two-loop Feynman diagrams with four on-shell legs,” *Phys. Lett.* **B469** (1999) 225–234, [arXiv:hep-ph/9909506](#).
- [33] V. A. Smirnov, “Analytical result for dimensionally regularized massless on-shell double box,” *Phys. Lett.* **B460** (1999) 397–404, [arXiv:hep-ph/9905323](#).
- [34] A. V. Smirnov and V. A. Smirnov, “On the Resolution of Singularities of Multiple Mellin- Barnes Integrals,” *Eur. Phys. J.* **C62** (2009) 445–449, [arXiv:0901.0386 \[hep-ph\]](#).
- [35] E. Remiddi and J. A. M. Vermaseren, “Harmonic polylogarithms,” *Int. J. Mod. Phys.* **A15** (2000) 725–754, [arXiv:hep-ph/9905237](#).
- [36] M. Czakon, J. Gluza, and T. Riemann, “Master integrals for massive two-loop Bhabha scattering in QED,” *Phys. Rev.* **D71** (2005) 073009, [arXiv:hep-ph/0412164](#).
- [37] M. Czakon, J. Gluza, and T. Riemann, “On the massive two-loop corrections to Bhabha scattering,” *Acta Phys. Polon.* **B36** (2005) 3319–3326, [arXiv:hep-ph/0511187](#).
- [38] S. Moch, P. Uwer, and S. Weinzierl, “Nested sums, expansion of transcendental functions and multi-scale multi-loop integrals,” *J. Math. Phys.* **43** (2002) 3363–3386, [arXiv:hep-ph/0110083](#).
- [39] G. Heinrich, T. Huber, and D. Maitre, “Master Integrals for Fermionic Contributions to Massless Three-Loop Form Factors,” *Phys. Lett.* **B662** (2008) 344–352, [arXiv:0711.3590 \[hep-ph\]](#).
- [40] G. Heinrich, T. Huber, D. A. Kosower, and V. A. Smirnov, “Nine-Propagator Master Integrals for Massless Three-Loop Form Factors,” *Phys. Lett.* **B678** (2009) 359–366, [arXiv:0902.3512 \[hep-ph\]](#).

- [41] K. Hepp, “Proof of the Bogolyubov-Parasiuk theorem on renormalization,” *Commun. Math. Phys.* **2** (1966) 301–326.
- [42] M. Roth and A. Denner, “High-energy approximation of one-loop Feynman integrals,” *Nucl. Phys.* **B479** (1996) 495–514, [arXiv:hep-ph/9605420](#).
- [43] T. Kaneko and T. Ueda, “A geometric method of sector decomposition,” [arXiv:0908.2897 \[hep-ph\]](#).
- [44] C. Bogner and S. Weinzierl, “Resolution of singularities for multi-loop integrals,” *Comput. Phys. Commun.* **178** (2008) 596–610, [arXiv:0709.4092 \[hep-ph\]](#).
- [45] J. Gluza, K. Kajda, T. Riemann, and V. Yundin, “Numerical Evaluation Of Feynman Tensor Integrals In Euclidean Kinematical Region,” *in preparation*.
- [46] J. Gluza, K. Kajda, and D. A. Kosower, “Towards a Basis of Planar Two-Loop Integrals,” *in preparation*.
- [47] D. A. Kosower, “A tool for automatic application of the first and second Barnes lemmas,” *available online:* , <http://projects.hepforge.org/mbtools/>.
- [48] M. Czakon, “IdSolver program,” *not public*.
- [49] T. Hahn, “CUBA: A library for multidimensional numerical integration,” *Comput. Phys. Commun.* **168** (2005) 78–95, [arXiv:hep-ph/0404043](#).
- [50] “CERN Program Library,” *available online:* , <http://cernlib.web.cern.ch/cernlib/>.
- [51] J. Gluza, A. Mitov, S. Moch, and T. Riemann, “The QCD form factor of heavy quarks at NNLO,” *JHEP* **07** (2009) 001, [arXiv:0905.1137 \[hep-ph\]](#).
- [52] S. Actis, M. Czakon, J. Gluza, and T. Riemann, “Virtual Hadronic and Heavy-Fermion $O(\alpha^2)$ Corrections to Bhabha Scattering,” *Phys. Rev.* **D78** (2008) 085019, [arXiv:0807.4691 \[hep-ph\]](#).
- [53] Z. Bern *et al.*, “The Two-Loop Six-Gluon MHV Amplitude in Maximally Supersymmetric Yang-Mills Theory,” *Phys. Rev.* **D78** (2008) 045007, [arXiv:0803.1465 \[hep-th\]](#).
- [54] D. B. Melrose, “Reduction of Feynman diagrams,” *Nuovo Cim.* **40** (1965) 181–213.
- [55] L. M. Brown *Nuovo Cimento* **22** (1961) 178.
- [56] F. R. Halpern *Phys. Rev. Lett.* **10** (1963) 310.
- [57] G. Kallen and A. Wightman *Dan. Mat. Fys. Skv.* **1** (1958) no. 6.

-
- [58] A. I. Davydychev, “A Simple formula for reducing Feynman diagrams to scalar integrals,” *Phys. Lett.* **B263** (1991) 107–111.
- [59] T. Binoth, J. P. Guillet, G. Heinrich, E. Pilon, and C. Schubert, “An algebraic / numerical formalism for one-loop multi-leg amplitudes,” *JHEP* **10** (2005) 015, [arXiv:hep-ph/0504267](#).
- [60] R. K. Ellis and G. Zanderighi, “Scalar one-loop integrals for QCD,” *JHEP* **02** (2008) 002, [arXiv:0712.1851 \[hep-ph\]](#).
- [61] T. Binoth, G. Heinrich, and N. Kauer, “A numerical evaluation of the scalar hexagon integral in the physical region,” *Nucl. Phys.* **B654** (2003) 277–300, [arXiv:hep-ph/0210023](#).
- [62] T. Stelzer and W. F. Long, “Automatic generation of tree level helicity amplitudes,” *Comput. Phys. Commun.* **81** (1994) 357–371, [arXiv:hep-ph/9401258](#).
- [63] F. Maltoni and T. Stelzer, “MadEvent: Automatic event generation with MadGraph,” *JHEP* **02** (2003) 027, [arXiv:hep-ph/0208156](#).
- [64] **KLOE** Collaboration, F. Ambrosino *et al.*, “Measurement of the DAΦNE luminosity with the KLOE detector using large angle Bhabha scattering,” *Eur. Phys. J.* **C47** (2006) 589–596, [arXiv:hep-ex/0604048](#).
- [65] F. Jegerlehner, “The Anomalous Magnetic Moment of the Muon,” *Springer Tracts in Modern Physics* (2007) .
- [66] A. Hoefer, J. Gluza, and F. Jegerlehner, “Pion pair production with higher order radiative corrections in low energy e^+e^- collisions,” *Eur. Phys. J.* **C24** (2002) 51–69, [arXiv:hep-ph/0107154](#).
- [67] M. Davier, “Precision Measurement of the $e^+e^- \rightarrow \mu^+\mu^-\gamma$ cross section with the ISR Method,” *10th International Workshop On Tau Lepton Physics (Tau08)*. , <http://tau08.inp.nsk.su/>.
- [68] H. Czyż. Private discussion.
- [69] M. Tentyukov and J. Fleischer, “A Feynman diagram analyser DIANA,” *Comput. Phys. Commun.* **132** (2000) 124–141, [arXiv:hep-ph/9904258](#).
- [70] J. A. M. Vermaseren, “FORM Symbolic Manipulation System,” *avaliabile online*: , <http://www.nikhef.nl/~form/>.
- [71] H. Czyz, A. Grzelinska, and J. H. Kuhn, “Spin asymmetries and correlations in Lambda-pair production through the radiative return method,” *Phys. Rev.* **D75** (2007) 074026, [arXiv:hep-ph/0702122](#).

- [72] T. Hahn, “Generating Feynman diagrams and amplitudes with FeynArts 3,” *Comput. Phys. Commun.* **140** (2001) 418–431, [arXiv:hep-ph/0012260](#).
- [73] W. J. Marciano and A. Sirlin, “Dimensional Regularization of Infrared Divergences,” *Nucl. Phys.* **B88** (1975) 86.
- [74] K. Kajda, “The pentagon diagrams of Bhabha scattering,” *Matter To The Deepest (Ustron 2007)*. .
- [75] H. Czyż, M. Gunia, J. Gluza, K. Kajda, T. Riemann, and V. Yundin. Work in progress.
- [76] “g77 - Fortran 77 compiler,” *available online:* , <http://www.gnu.org/software/fortran/fortran.html>.
- [77] “Intel Fortran Compiler,” *available online:* , <http://software.intel.com/en-us/intel-compilers/>.
- [78] M. Czakon, J. Gluza, K. Kajda, and T. Riemann, “Differential equations and massive two-loop Bhabha scattering: The B5l2m3 case,” *Nucl. Phys. Proc. Suppl.* **157** (2006) 16–20, [arXiv:hep-ph/0602102](#).
- [79] E. T. Whittaker and G. N. Watson, “A Course of Modern Analysis,” *Series: Cambridge Mathematical Library* .