One-loop corrections at future linear colliders

Otto Rottier
Master's thesis
May 13, 2014

Daily supervisor:
Jos Vermaseren

Supervisor:
Eric Laenen

Universiteit Utrecht

Utrecht University
Physics & Astronomy

NIKHEF

National Institute
for Subatomic Physics

Abstract

In order to benefit from high precision measurements at a future Linear Collider, we need to go beyond the tree level for cross section calculations in the electroweak sector. At the one-loop level, the number of Feynman diagrams has increased rapidly and the internal loops greatly complicate the calculations. Fortunately automated systems have been constructed to deal with these large scale computations. In particular, the GRACE system is capable of calculating general processes of up to four final particles including their masses.

In this report I will review some of the challenges that these automated systems meet and explain how to deal with them. Additionally, I will present my findings on the calculation of $e^+e^- \to ZZH$ and ZHH in order to study the couplings between the Higgs particles and the vector bosons in the standard model. The project will use the GRACE system for this and make a few additions to it. In particular a system to integrate over matrix elements that occupy more than 2 Gbytes in the computer. Because of numerical cancellations between the diagrams this is highly nontrivial.

Contents

Chapter 1

Introduction

Particle physics has been an active field of research for a very long time. Only recently enough evidence has been gathered at the Large Hadron Collider to claim the existence of a particle resembling the Higgs boson, the only Standard Model particle that had not been discovered yet. However just the finding of this boson is actually of limited value. We need precision studies to fully understand the Higgs mechanism (electroweak symmetry breaking). The LHC will be able to supply a few measurements on the couplings of the Higgs to fermions and gauge bosons[1] but the most precise measurements will be achieved in the clean environment of a linear collider that will operate on the TeV scale.

In order to learn from the precision measurements, we also need high precision predicitions from the theory. This means there is a need to go beyond the leading order (tree level) contributions for most processes. Due to the increase in the number of Feynman diagrams and the complications with internal loops this is highly nontrivial. Fortunately over the last decades, automated systems have been built to perform most of the work involved in tree processes and later 1-loop processes. This work will mainly focus on the GRACE-loop system developed by the Minami Tateya group at KEK[2]. This system, and the FeynArts-FormCalc-LoopTools system developed at the Max-Planck-Institute for Physics in Münich[3], are currently the only automated systems that can work with 1-loop processes with up to six (massive or massless) external particles in the electroweak part of the Standard Model.

The main objective of this report is to inform the reader on the need for high precision particle physics, to give an outline of the internals of GRACE-loop and finally to compute a few reactions in depth that contain particles that are involved in the electroweak symmetry breaking.

1.1 The need for high precision particle physics

The Standard Model (SM) is the accepted theoretical model of all known elementary particles and their interactions, including the recently discovered Higgs particle. Despite its success, there are still a few theoretical problems that are not solved by the SM, in particular the hierarchy problem. One of the hierarchy questions is why the electroweak theory is below the TEV scale and consequently why the Higgs boson is so much lighter than the Planck mass. One would expect that the large quantum contributions to the square of the Higgs boson mass would inevitably make it huge, comparable to the Planck energy scale. The fact that this is not observed makes one wonder if there is a subtle fine-tuning cancellation between the quadratic radiative corrections and the bare mass of the Higgs boson. This might suggest that new physics appears at a scale Λ not far beyond the TeV scale.

The origin of the electroweak symmetry breaking might be the key to this new physics. Therefore it is extremely relevant to probe the Higgs-vector boson couplings (HVV) because they are expected to be sensitive to the symmetry breaking physics.

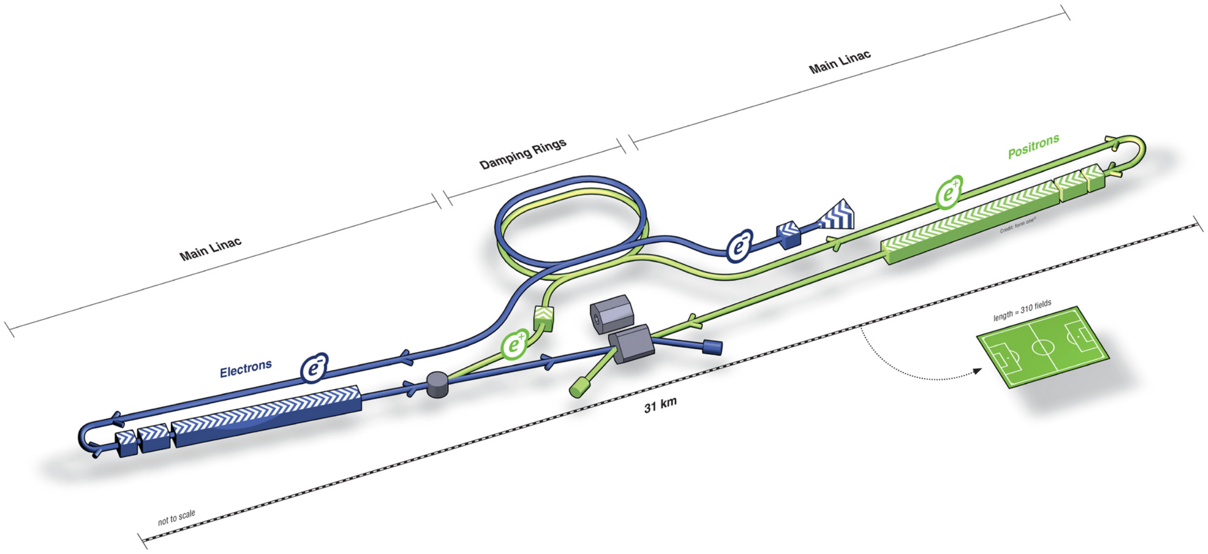
1.2 The next generation collider

High precision particle physics experiments are best done using point like initial particles, such as leptons. This has the advantage that one does not have to cope with Parton Distribution Functions (PDFs) as is the case with composite particles like the proton. With proton collisions, the initial state is not known precisely since the protons are likely to collide only partially and thus have a large number of ‘spectators’ per event. Since the momentum fraction of the colliding partons is generally low, few events yield useful information. At a lepton collider only two particles will interact per collision, yielding a very clean signal with the benefit that every registered event is significant.

1.2.1 Electron collider

An electron collider is the most obvious candidate for a high precision collider, since electrons are easily produced and accelerated. An electron-positron collider has the advantage that the two particles can annihilate, creating just pure energy that is available for any set of particles that obeys the relevant conservation laws. A muon or tau collider would have the advantage of the higher initial mass, because it reduces the relative energy loss of photon radiation from synchrotron radiation (inversely proportional to the fourth power of the mass of the particles). Unfortunately it is harder to generate these heavy leptons, and besides their limited lifetime, the beam focussing is also troublesome for these particles[4]. The energy loss from synchrotron radiation of charged light particles can be avoided by using a linear collider.

The International Linear Collider (ILC) project is aiming to design the next generation particle collider that will provide a high level of accuracy with TeV scale interactions. The machine will collide electrons and positrons 7,000 times per second and will have a length of about 31 kilometers[5]. Each bunch will contain 20 billion electrons or positrons to provide a high luminosity. The collider design is the result of nearly twenty years of research, based on over a decade of work by the TESLA collaboration in the 1990s. The proposed building site is yet to be decided (either Japan, the USA or CERN), and the (unofficial) planned date for operation is after 2026.



ILC Scheme | © www.form-one.de

Figure 1.1: The International Linear Collider (ILC) schematic layout as planned per summer 2013. Taken from the official ILC website[6]

The ILC has a number of significant advantages over the LHC:

- The collisions are cleaner, yielding a low background signal.
- The initial state is well defined (spin, energy of interacting particles).
- The center of mass energy is tunable so one can suppress certain s-channel interactions and investigate treshold behaviour.
- The beam polarization is tunable so one can suppress e.g. left-handed interactions
- It is expected that 1 in 100 events is a relevant Higgs event. This is many orders of magnitude more than at the LHC.

- One-loop ELWK corrections will generally provide enough accuracy for comparsion with the measurements, as opposed to two loop corrections at the LHC.

This makes the ILC a precision collider that will work complementary to the LHC in exploring the TeV scale physics. The ILC will specifically allow precision studies of the top quark and electroweak gauge bosons, it will explore the Higgs sector, and study possible extensions of the Standard Model[5].

Precision measurements of the Higgs decay branching ratios are crucial to check the mass-coupling relation in the SM or to search for physics beyond the SM[7]. For experiments at the Large Hadron Collider it is not easy to measure the branching ratios of the Higgs particle to quarks and gluons due to the large QCD backgrounds in proton-proton collisions. In contrast, precise measurements of branching ratios are anticipated at the ILC, due to the advantages mentioned above.

Important measurements on the Standard Model have already been performed at the e^+e^- colliders SLC and LEP, which started a new era of high precision experiments. Results include the determination of the number of light neutrinos and the precise measurement of the mass of the W and Z bosons.

ILC parameters

The ILC reference design report[5] states the following on the ILC parameters: The maximum center of mass energy is designed to be $\sqrt{s} = 500$ GeV, with a possible upgrade to $\sqrt{s} = 1000$ GeV. This energy should be tunable, with a minimum of $\sqrt{s} = 200$ GeV. Calibrations will take place from $\sqrt{s} = 91$ GeV. The total luminosity is required to be about $500 fb^{-1}$ within the first four years, running up to $1000 fb^{-1}$ during the first phase of operation. The electron beam can be polarized up to 80%, while for the positron beam a polarization up to 30% will be tried for.

The ILC can also run with minor modifications at the GigaZ or GigaW mode. These modes have the proper polarization and energy to run at the Z or W resonance.

With a few modifications the ILC can be changed to an electron-electron collider, providing a tool to measure for example the selectron mass (the super-symmetric partner of the electron), if it exists.

1.2.2 Muon collider

Recently, new plans have been proposed for a muon collider[8]. Muon colliders have the benefit of a larger center of mass energy compared to electron colliders. Unfortunately, it is technically very hard to create a sharp muon beam that can be used inside a collider.

However, if all technical difficulties can be solved, these types of machine can achieve excellent results for investigating the electroweak interactions.

1.3 The role of computational methods

When performing high precision measurements, we also need high precision theory. Since most calculations on the SM interaction are based on perturbation theory, they are not exact. In the electroweak sector, the next to leading order corrections (in α, 1-loop processes) are typically in the order of a few percent in magnitude compared to the leading order (tree diagrams). This correction grows with the center of mass energy of the collider, and the number of external particles of the process. That means that if measurements are performed with accuracies of about a few percent, the classical level of the theory is not sufficient and we need to take into account quantum corrections. In addition, some processes have a zero contribution for tree processes, but do have a loop contribution. That is why we need to include at least the 1-loop contributions to compare the theoretical results to the results from high precision experiments.

1-loop processes add a lot of complications to your field theory computations (see section 2.5.2) such as internal loop integrations and their associated divergences. However, over the last decades, excellent solutions have been found for the problems occurring in 1-loop calculations. The hard part is actually the problem size. Whereas a 2 → 2 process has typically only a few diagrams at tree level, at one-loop order this can be over a hundred. A 2 → 3 process typically has a few dozen tree diagrams, but a few thousand loop diagrams. The number of diagrams increases exponentially if more external particles are present. In principle all diagrams contribute to physical quantities such as the cross section, so for every diagram Feynman rules should be applied, all diagram contributions are collected in the matrix element which is subsequently squared so the number of terms increases dramatically again. One can imagine that this job cannot be done by hand in a reasonable amount of time. Not to mention solving the phase space integral over the final particles that comes next.

Computers however are especially suited for this job, since the rules for these calculations are (relatively) simple and straight forward. Symbolic programming languages such as `FORM`[9] can take care of the various algebraic operations (absolute square, trace, tensor reduction) on large numbers of terms. Subsequently, ‘number crunchers’ written in `Fortran` or `C` can take care of the numerical phase space integration.

Chapter 2

Theory

The standard model (SM) is the accepted theoretical model of all known elementary particles and their interactions, including the recently discovered Higgs particle. It has been successful in explaining the electromagnetic, weak and strong force and has predicted particles that had not been observed before. Experiments are performed continuously at colliders such as the LHC, RHIC and SLC to test the theory and to look for new physics that arises as a deviation from the SM.

The SM is a Yang-Mills theory that is invariant under local $SU(3) \times SU(2) \times U(1)$ gauge transformations. Depending on the gauge fixing one chooses, the SM contains about 61 particles (24 + 24 (anti-)fermions, 12 gauge bosons and the Higgs boson) and 9 classes of interactions (e.g. fermion-fermion-gauge boson, quark-quark-gluon). Again dependent on the gauge and the counter terms used, this constitutes to over 300 vertices.

Currently, the GRACE system can handle the electroweak (ELWK) sector of the standard model, or, if desired, only the QED part. The ELWK sector will be of great importance in the future linear collider, but also in the Large Hadron Collider since the contributions from the next-to-next-to-leading order in the QCD sector is comparable to the next-to-leading order in the ELWK part. This report focuses on the use of the GRACE system for automatic calculations, so in this theory section we will treat cross section computations in the electroweak model.

2.1 Outlook: how to calculate the cross section

In order to get the full $\mathcal{O}(\alpha)$ radiative correction on the cross section of a $2 \to N$ process, one needs to calculate the following phase space integrals,

$$\sigma_{tot} = \int_{\Omega(2\to N)} d\sigma_{\text{Tree}} \tag{2.1}$$

$$+ \int_{\Omega(2\to N)} d\sigma_{\text{Loop}} \tag{2.2}$$

$$+ \delta_{\text{soft}}(m_\gamma, E_\gamma < k_c) \int_{\Omega(2\to N)} d\sigma_{\text{Tree}} \tag{2.3}$$

$$+ \int_{\Omega(2\to N+\gamma)} d\sigma_{\text{hard}}(E_\gamma > k_c). \tag{2.4}$$

Let us explain in detail what this means:

- The first line is the tree contribution (leading order) of the process

- The second line is the one-loop contribution. This contribution will contain UV-divergences with which we deal using counter terms (see section 2.9). Additionally, IR-divergences will appear (see section 2.8), but we know these will be cancelled by ultrasoft photon emission at the external legs. We get a finite result from the one-loop contribution by using a small photon mass (m_γ) as a regulator.

- The dependence on the regulator drops when we include the ultrasoft photon emission at the external legs. We can include this part safely, since we cannot differentiate between a $2 \to N$ reaction and a $2 \to N + \gamma$ reaction where the photon cannot be detected. That is written down on the third line. We use an approximation formula ($\delta_{\text{soft}}(m_\gamma, E_\gamma < k_c)$) for this contribution. The value k_c is the upper limit on the soft photon momentum. This is explained in detail in section 2.8.

- The last line is the contribution from hard photons. The cut k_c must be the same as the one for the soft photon emission formula, so the final result will not depend on this quantity.

We will go into further detail on the cross section calculation later, but for now it is important to note that we will need to calculate the tree and loop contribution of a process, and additionally the tree contribution with an extra photon. In order to do this, one writes down all Feynman diagrams and constructs the amplitude of the process. Then the squared amplitude is integrated over the phase space of the final particles in order to get the cross section.

2.2 The electroweak model

The electroweak part of the standard model consists of the SM fermions (quarks and leptons) and their electromagnetic and weak interactions (thus, the gauge bosons γ, $W^{\pm}$ and Z). The strong interactions have been left out. The Higgs scalar particle is present as a result of the inclusion of the electroweak symmetry breaking principle. The electroweak sector of the Standard Model forms a consistent theory that we can apply on particle colliders such as the ILC to calculate for example cross sections of the colliding electrons and positrons.

In order to perform automatic calculations with this model, the particles and vertices need to be written down with their properties in a computer readable format. This is done in the so called model files that are bundled with GRACE. One can then easily switch between models when studying a process.

The particles and vertices emerge of course from the Langrangian of the model. In this report we will focus only on the electroweak part of the Lagrangian, which is invariant under the gauge group $SU(2) \times U(1)$. A feature of the electroweak part is a single doublet of scalar fields that mixes with the vector bosons. Also the fermions are decomposed in their chiral components in which the left handed components are assigned to a doublet representation of $SU(2)$, and the right handed components are $SU(2)$ singlets.

The ELWK Lagrangian can be written as a sum of pure contributions from the gauge fields, fermionic fields, scalar fields and Yukawa coupling of the fermions and scalars:

$$\mathcal{L}_{EW} = \mathcal{L}_G + \mathcal{L}_F + \mathcal{L}_S + \mathcal{L}_Y \tag{2.5}$$

The gauge fields corresponding to the $SU(2) \times U(1)$ symmetry are denoted by W_a^{μ} with corresponding couplings g^a. The gauge invariant field strength can be written as

$$F_{\mu\nu}^a = \partial_\mu W_\nu^a - \partial_\nu W_\mu^a + g^a f^{abc} W_\mu^b W_\nu^c. \tag{2.6}$$

The structure constant f^{abc} of this gauge group is the Levi-Civita tensor ϵ^{abc} for $a, b, c \neq 0$ (the $SU(2)$ triplet), and zero otherwise (the $U(1)$ singlet). Then it is convenient to write $g^a = g$ for $a = 1, 2, 3$ and $g^0 = g'$. The pure gauge contribution to the Langrangian is

$$\mathcal{L}_G = -\frac{1}{4} F_{\mu\nu}^a F_a^{\mu\nu}. \tag{2.7}$$

The fermionic part of the Langrangian contains fields for the left- and right-handed fermions (L and R). The fields L of each generation belongs to the $SU(2)$ doublet and the fields R belong to the singlet. The interaction with the gauge fields is completely specified by their isospin (T^3) and hypercharge (Y), with the electromagnetic charge $Q = T^3 + Y$. The interaction itself comes from the covariant derivative

$$D_\mu = \partial_\mu - ig \sum_{a=1}^{3} W_\mu^a \frac{\sigma^a}{2} - ig' Y W_\mu^0, \tag{2.8}$$

where σ^a are the Pauli matrices. This covariant derivative is used in the fermionic part of the Lagrangian,

$$\mathcal{L}_F = i \sum \overline{L}\gamma^\mu D_\mu L + i \sum \overline{R}\gamma^\mu D_\mu R, \tag{2.9}$$

as well as in the scalar part, yielding interactions between the scalar field and the gauge fields. The scalar fields is also subject to a nonzero potential V:

$$\mathcal{L}_S = (D_\mu\phi)^\dagger(D^\mu\phi) - V(\phi) \tag{2.10}$$

$$V(\phi) = \lambda(\phi^\dagger\phi)^2 - \mu^2\phi^\dagger\phi \equiv \lambda(\phi^\dagger\phi - \frac{v^2}{2})^2 + \frac{\mu^4}{4\lambda} \tag{2.11}$$

The mass terms for the fermions and gauge bosons are generated in a gauge invariant way through the Higgs mechanism. To that end, the scalar field is chosen to be a doublet with hypercharge $Y = 1/2$ that has a nonzero vacuum expectation value v by choosing $\mu^2 < 0$ in the scalar potential:

$$\phi = \frac{1}{\sqrt{2}}\begin{pmatrix} i\chi_1 + \chi_2 \\ v + H - i\chi_3 \end{pmatrix} = \begin{pmatrix} i\chi^+ \\ (v + H - i\chi_3)/\sqrt{2} \end{pmatrix} \tag{2.12}$$

In this way $\langle 0|\phi|0\rangle = \{0, v\}$, where $v \approx 246.22$ GeV (known from measurements of the properties of electroweak interactions). The Goldstone bosons χ^+, $\chi^- \equiv i\chi_1 - \chi_2$ and χ_3 in $\mathcal{L}_S$ are absorbed by the gauge fields W^a_μ, resulting in the massive particles Z and $W^\pm$, while the photon A remains massless. The relation between these physical fields and the original W quartet is of no relevance for this report, instead we will show the resulting masses of all particles that emerge in the electroweak sector,

$$M_H = \lambda v, \tag{2.13}$$

$$M_W = \frac{1}{2}gv = \frac{ev}{2\sin\theta_W}, \tag{2.14}$$

$$M_Z = \frac{1}{2}\sqrt{g^2 + g'^2}v = \frac{ev}{2\sin\theta_W\cos\theta_W} = \frac{M_W}{\cos\theta_W}, \tag{2.15}$$

$$M_\gamma = 0. \tag{2.16}$$

Here θ_W is the weak mixing angle or Weinberg angle, the angle by which spontaneous symmetry breaking rotates the original W vector boson plane, producing as a result the Z-boson and the photon. This is an unfixed parameter of the SM.

The last part of the Lagrangian is the Yukawa coupling of the fermions with the scalar field. This in turn generates masses for the fermions.

$$\mathcal{L}_Y = -i \sum_{\text{up}} f_U \overline{L}_U \sigma^2 \phi^* R_U - \sum_{\text{down}} f_D \overline{L}_D \phi R_D + (h.c.) \tag{2.17}$$

As a consequence of the chiral anomaly of the electroweak model, only the left handed fermions interact via the electroweak force.

2.3 Gauge fixing

In order to quantize the theory, one is interested in the path integral over the Lagrangian. The propagators and vertices and can be read off the quantity Z, which in the case of a pure gauge theory is defined as a path integral over the fields

$$Z = \int \mathcal{D}A e^{iS[A]} = \int \mathcal{D}A e^{i\int d^4x \mathcal{L}(A)}. \tag{2.18}$$

The gauge freedom manifests itself in the fact that under a transformation $A \to A_g$, the integrand and measure do not change ($S(A) = S(A_g)$ and $\mathcal{D}A = \mathcal{D}A_g$). The gauge freedom in our model is a problem now, since we are overcounting physical field configurations because of the gauge invariance.

Our goal is to rewrite the integral Z in such a way that the redundant integration over g factors out, i.e. $Z = (\int \mathcal{D}g)J$, with J independent of g. This procedure is called 'fixing the gauge'.

In an automated system, the gauge freedom is a blessing in disguise since we can use it to perform checks on the calculation. We can choose to fix the gauge in any way we want, but the final results have to be independent of this choice. The gauge choice is parametrized by the so called gauge parameters, and the Feynman rules will depend on these parameters. A striking result of this is that the gauge invariance of the final result occurs through the summation of all the Feynman diagrams, whereas each of these diagrams is gauge dependent. Varying the gauge parameters and checking the final results will provide a very powerful check (see section 4.1) because if the result is gauge invariant, one can confidently assume that no diagrams are forgotten, the symmetry factors are correct and the Feynman rules have been applied correctly.

In this section we describe the Faddeev-Popov procedure of fixing the gauge along the lines of [10]. We need to introduce a gauge fixing condition at every point x in our space-time. This constraint can be put inside our path integral by using the identity

$$1 = \Delta(A) \int \mathcal{D}g \delta[f(A_g)]. \tag{2.19}$$

$\Delta(A)$ is called the Faddeev-Popov determinant, and f is a function of our choosing.

Using the fact that the gauge transformations form a group, and that $\mathcal{D}g$ is an invariant measure, we see that the Faddeev-Popov determinant is gauge invariant

$$[\Delta(A_{g'})]^{-1} = \int \mathcal{D}g' \delta[f(A_{g'g})] = \int \mathcal{D}g'' \delta[f(A_{g''})] = [\Delta(A)]^{-1}. \tag{2.20}$$

Inserting the identity into the equation for Z, we see:

$$Z = \int \mathcal{D}A e^{iS[A]} \tag{2.21}$$

$$= \int \mathcal{D}A \int \mathcal{D}g \; e^{iS[A]} \Delta(A) \delta[f(A_g)] \tag{2.22}$$

$$= \left(\int \mathcal{D}g \right) \int \mathcal{D}A \; e^{iS[A]} \Delta(A) \delta[f(A)] \quad \text{sending } A_g \to A_{g^{-1}} \tag{2.23}$$

We have factored out the integral over the gauge group. Since the overall factor of Z does not matter, we can now work out physical equations for the field A.

2.3.1 Linear gauge fixing

We will apply the Faddeev-Popov trick on our pure gauge theory involving the field A. The action is invariant under the transformation $A_\mu \to A_\mu - \partial_\mu \Lambda$. This means that g is denoted by Λ and $A_g = A_\mu - \partial_\mu \Lambda$.

A judicious gauge fixing choice can simplify your calculations dramatically, but care has to be taken in order for the theory to remain renormalizable. One type of renormalizable gauge are the ξ_R gauges. Instead of fixing the gauge fields a priori with an auxiliary equation (like the Coulomb or Lorenz gauge), a gauge breaking term is introduced of the form

$$\mathcal{L}_{\text{GF}} = -\frac{(\partial_\mu A^\mu)^2}{2\xi}. \tag{2.24}$$

This term originates from choosing $f(A) = \partial A - \sigma(x)$ and performing Faddeev-Popov's trick. Since Z is independent of σ, we can integrate Z with impunity with an 'arbitrary' function of σ: $e^{-(i/2\xi)\int d^4x \sigma(x)^2}$,

$$Z = \int D\sigma \; e^{-(i/2\xi)\int d^4x \sigma(x)^2} \int \mathcal{D}A \; e^{S[A]} \Delta(A) \delta[f(A)]. \tag{2.25}$$

We can calculate $\Delta(A)$ from $[\Delta(A_{g'})]^{-1}$:

$$[\Delta(A)]^{-1} = \int \mathcal{D}g \delta[f(A_g)] = \int \mathcal{D}\Lambda \delta(\partial A - \partial^2 \Lambda - \sigma) \tag{2.26}$$

Because $\Delta(A)$ is multiplied with $\delta[f(A)] = \delta(\partial A - \sigma)$ we can take the value of $\partial A - \sigma$ to be zero in the above equation:

$$[\Delta(A)]^{-1} = \int \mathcal{D}\Lambda \delta(\partial^2 \Lambda) \tag{2.27}$$

This object does not depend on A, so we can leave it out safely from equation 2.25,

$$\begin{aligned} Z &= \int D\sigma\ e^{-(i/2\xi)\int d^4x\sigma(x)^2} \int \mathcal{D}A\ e^{iS[A]}\delta(\partial A - \sigma) && (2.28) \\ &= \int \mathcal{D}A\ e^{iS[A]-(i/2\xi)\int d^4xA(x)^2}. && (2.29) \end{aligned}$$

With this additional term the propagators of the gauge bosons become:

$$\Delta_{\mu\nu}(k) = \frac{1}{k^2 - M^2}\left(g_{\mu\nu} - (1-\xi)\frac{k_\mu k_\nu}{k^2 - \xi M^2}\right) \tag{2.30}$$

The gauge fixing parameter ξ can be chosen freely. There are three common choices:

$$\begin{cases} \xi \to 0, & \text{Landau gauge} \\ \xi = 1, & \text{Feynman-'t Hooft gauge} \\ \xi \to \infty & \text{Unitary gauge} \end{cases}$$

The Landau gauge is classically equivalent to the Lorentz gauge and is characterized by the fact that the vector boson propagator is transversal, $k^\mu \Delta_{\mu\nu}(k) = 0$. In the unitary gauge all ghost and Goldstone fields may be omitted, reducing the number of vertices and consequently the number of diagrams.

The Feynman-'t Hooft gauge has the advantage that the propagator of the gauge bosons take a very simple form, because only their transverse part $g_{\mu\nu}$ contributes. The longitudinal part, proportional to $k_\mu k_\nu$, introduces higher rank loop integral tensors which are computationally very expensive to deal with (see section 2.5.2). Additionally, the denominator is known to cause numerical instabilities due to singular nature near the treshold. This is one of the large problems when dealing with the kinematics (see section 2.10). On top of that, calculations for the squared amplitude tend to get extremely large, so one needs to attempt to get the smallest espressions possible. These three reasons make the Feynman-'t Hooft gauge very suitable for automatic calculations.

Although the linear Feynman-'t Hooft gauge is very convenient to work with, we will dive into the gauge fixing a bit further. As mentioned earlier, we would like to use the gauge parameter as a check of the squared amplitude since at every point in the phase space the dependence on the gauge parameters should cancel. Therefore, we will calculate the squared amplitude in terms of the (unfixed) gauge parameter and in the end check numerically if the result is independent on the gauge parameter by sticking in different values. The numerical problems with the longitudinal part in the boson propagator makes this approach unfeasible, so we will look for another gauge fix that is more computer friendly in order to build checks for the gauge invariance.

2.3.2 Nonlinear gauge fixing

Linear constraints are the simplest way to fix the gauge, but the Faddeev-Popov trick allows for an arbitrary choice of the function f. This means we can also include nonlinear combinations of the fields in the effective Lagrangian. In this section we treat a particular choice of these so called nonlinear gauges, introduced in [11] where it was used to simplify tree-level calculations. This is a generalization of the ξ_R gauge.

The gauge fixing term takes the following form in the electroweak model,

$$\begin{aligned}\mathcal{L}_{\mathrm{GF}} = &-\frac{1}{\xi_W}\left|(\partial_\mu - ie\tilde{\alpha}A_\mu - igc_W\tilde{\beta}Z_\mu)W^{\mu+} + \xi_W\frac{g}{2}(v+\tilde{\delta}H + i\tilde{\kappa}\chi_3)\chi^+\right|^2\\ &-\frac{1}{2\xi_Z}\left(\partial_\mu Z^\mu + \xi_Z\frac{g}{2c_W}(v+\tilde{\epsilon}H)\chi_3\right)^2 - \frac{1}{2\xi_A}(\partial_\mu A^\mu)^2,\end{aligned} \tag{2.31}$$

where the five parameters $\{\tilde{\alpha},\tilde{\beta},\tilde{\delta},\tilde{\kappa},\tilde{\epsilon}\}$ are now additional gauge parameters.

The full nonlinear gauge Lagrangian and Feynman rules are listed in [12]. In that report also the various choices for the nonlinear modifications are discussed. The gauge boson propagators in this nonlinear gauge are identical to the ones from the linear gauge (equation 2.30),

$$\text{W}^\pm \text{ propagator: } \Delta_{\mu\nu}(k) = \frac{1}{k^2 - M_W^2}\left(g_{\mu\nu} - (1-\xi_W)\frac{k_\mu k_\nu}{k^2 - \xi_W M_W^2}\right), \tag{2.32}$$

$$\text{Z propagator: } \Delta_{\mu\nu}(k) = \frac{1}{k^2 - M_Z^2}\left(g_{\mu\nu} - (1-\xi_Z)\frac{k_\mu k_\nu}{k^2 - \xi_Z M_Z^2}\right), \tag{2.33}$$

$$\text{A propagator: } \Delta_{\mu\nu}(k) = \frac{1}{k^2}\left(g_{\mu\nu} - (1-\xi_A)\frac{k_\mu k_\nu}{k^2}\right). \tag{2.34}$$

However, the additional gauge parameters make their appearance in the vertices. For example, the triple vector vertex of $W^-(p_1)$ (index μ), $W^+(p_2)$ (index ν) and $A(p_3)$ (index ρ) takes the value

$WW\gamma$-vertex (NLG):

$$e\left[g^{\mu\nu}(p_1-p_2)^\rho + (1+\tilde{\alpha}/\xi_W)(p_3^\nu g^{\mu\rho} - p_3^\mu g^{\nu\rho}) + (1-\tilde{\alpha}/\xi_W)(p_2^\mu g^{\nu\rho} - p_1^\nu g^{\mu\rho})\right], \tag{2.35}$$

as opposed to the vertex in the linear gauge,

$WW\gamma$-vertex (LG):

$$e\left[g^{\mu\nu}(p_1-p_2)^\rho + (p_3^\nu - p_1^\nu)g^{\mu\rho} + (p_2^\mu - p_3^\mu)g^{\nu\rho}\right]. \tag{2.36}$$

Although the implementation of such a gauge in an automated system takes obviously more effort than the linear gauge, it has a a great advantage. By setting the gauge parameters $\xi_{W/Z/A} = 1$, we avoid the numerical problems with the propagators, but we are still left with the possibility of extensive gauge invariance checks. One can vary the five additional parameters but the calculated physical quantities should not depend on them. This is a very powerful check of the implementation of the model.

When the gauge check is passed, one can tune the parameters $\{\tilde{\alpha}, \tilde{\beta}, \tilde{\delta}, \tilde{\kappa}, \tilde{\epsilon}\}$ such that the vertices simplify or even vanish. For example, for photonic vertices $\tilde{\alpha} = 1$ is preferred since it removes the $W\chi A$ vertex and also the $WW\gamma$ vertex simplifies (equation 2.35).

2.4 Generating Feynman diagrams

The common way to gain amplitude information on a particular process is to use perturbative field theory on the Standard Model. This means all possible Feynman graphs need to be constructed following the rules of the model. These rules are the result of perturbing the interaction of the fields in the generating functional of the theory[13].

Usually one draws all diagrams manually, but beyond the tree level and in the full electroweak model, the number of diagrams can be extremely large (several thousands). Because of this, and additionally for the use in an automated system, an algorithm is necessary to generate all diagrams.

2 → #	tree diagrams	1-loop diagrams
2	˜10	˜100
3	˜25	˜1,000
4	˜50	˜10,000

One can use various approaches in generating Feynman diagrams. One method is to use a symbolic programming language to perform the required operations on the generating functional, and subsequently translate the resulting expression into diagrams, but this usually takes a lot of computer resources. An alternative method is to generate the diagrams by using graph theory, this tends to work much faster.

The algorithm boils down to a few simple steps. First, all topologies for a given number of external particles and vertices are generated. In figure 2.1, the self energy topologies of a 2 → 2 process are shown. These topologies consist of edges (the propagators) and nodes (the vertices). Then the endnodes are labeled with the external particles in all possible ways. Finally the internal edges are labelled in such a way that the diagram is compatible with the Feynman rules. This means all edges are associated with a (possibly directed) propagator in such a way that every node is allowed in your theory. Then the topologically equivalent diagrams need to be grouped and symmetry factors should be collected. The full algorithm of this method, which is also used in the GRACE system, is described in [14].

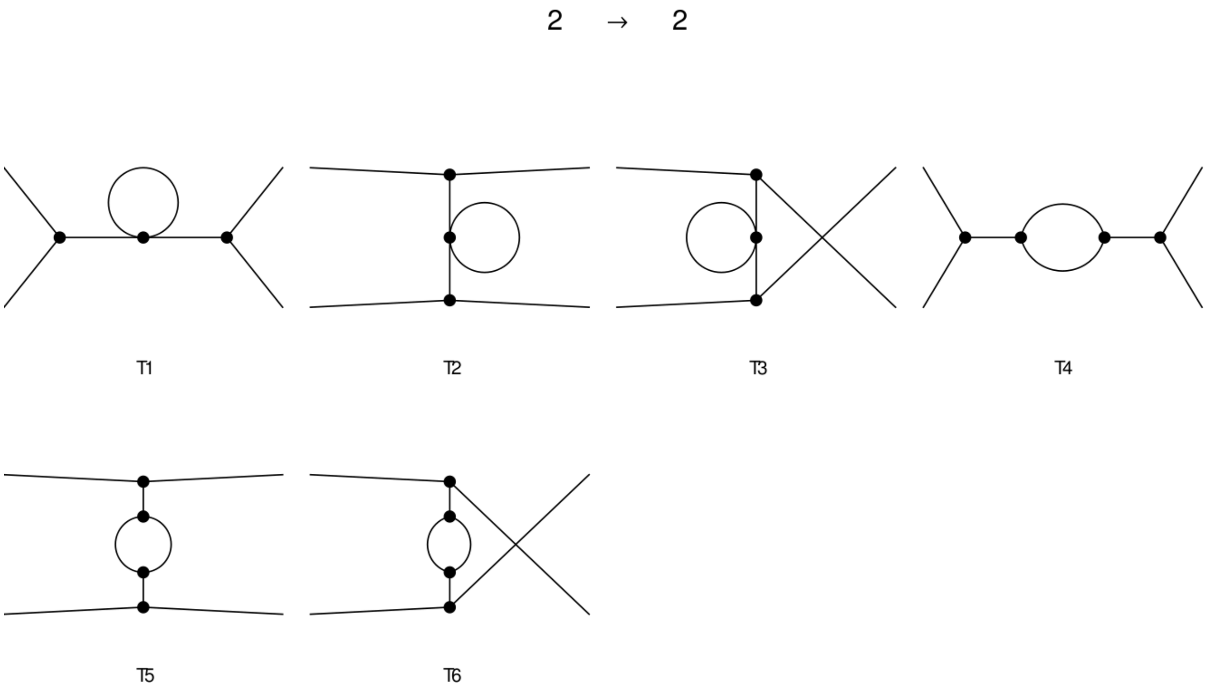


Figure 2.1: The self energy topologies of a 2 → 2 process, generated by FeynArts

2.5 Applying the Feynman rules

When the set of Feynman diagrams belonging to a process has been construced, one needs to apply the Feynman rules to convert the diagrams to symbolic expressions. These expressions are summed up in the matrix element, which is then squared and summed over the spins in order to get information on the process. This quantity $\langle|\mathcal{M}|^2\rangle$ can be extremely complicated if a lot of diagrams are present, and might involve difficult quantities with traces and tensor products. Instead of using the squared amplitude procedure, one can try to write the quantity $\mathcal{M}$ as a complex number directly, square it numerically and then take the spin sums. This is called the spinor method.

Tree diagrams can be handled in both ways in the GRACE system. For loop diagrams, the squared amplitude method is the most viable.

2.5.1 Spinor method

The spinor method can be performed numerically with a library such as CHANEL[15] or HELAS[16]. This method has several advantages:

- The numerical expression for $\mathcal{M}$ is a lot simpler than the symbolic expression for

$|\mathcal{M}|^2$.

- The intermediate symbolic stage can be skipped.
- The spinor method provides information on the polarizations of the particles in a very natural way.
- Divergence cancellations occur before squaring the quantity. This makes them easier to see and understand, to deal with them numerically.

There are also packages, such as `Spinney`[17] (written in `FORM`), that deal with the spinors symbolically in order to perform the full amplitude calculation symbolically.

The spinor approach has been discarded with the release of GRACE-loop. Only the squared amplitude calculations are possible. This is treated in the next section.

2.5.2 Squared amplitude method up to 1-loop level

When the Feynman diagrams up to the 1-loop order have been generated, the GRACE system produces a routine written in a symbolic programming language that contains the expression for every diagram after applying the Feynman rules. In the GRACE system, this symbolic language is `FORM` because it can handle large expressions and can deal with tensor calculations involving gamma matrices, Levi-Civita tensors and traces. The expressions for all diagrams are summed to generate the amplitude,

$$\mathcal{M} = T + L, \tag{2.37}$$

in which T contains all expression for the tree diagrams, and L contains the expression for all 1-loop diagrams. We will need the squared amplitude to calculate physical quantities, such as the cross section

$$|\mathcal{M}|^2 = T^\dagger T + L^\dagger T + T^\dagger L + L^\dagger L. \tag{2.38}$$

However, in the electroweak theory, a full one-loop calculation means we write down all terms up to the order α^2. Therefore we discard the squared loop contribution $L^\dagger L$,

$$|\mathcal{M}|^2 = T^\dagger T + L^\dagger T + T^\dagger L, \text{ up to order } \alpha^2. \tag{2.39}$$

2.6 Spin projection and polarization

For calculations relevant for the ILC, we will have two incoming fermions: the electron and the positron. The outgoing particles might or might not be particles with spin. This means we should take into account the effects of spin.

For unpolarized interactions, one averages over the spin of the incoming particles (no preference for electron spin up or down), and sums over the spins of the final particles (all particles are detected, regardless their spin). This means we can use the completeness relations for the fermions,

$$\sum_s u^s(p)\bar{u}^s(p) = \not{p} + m, \tag{2.40}$$

$$\sum_s v^s(p)\bar{v}^s(p) = \not{p} - m. \tag{2.41}$$

These quantities appear in the expression for $|\mathcal{M}|^2$. The completeness relations allow us to rewrite the spinors u and v in terms of the four momentum and mass of the fermions.

In order to calculate polarized interactions, we need a basis of polarization states. For light particles such as the electron in the relativistic limit, the best way is to quantize the the spin along the direction of the particle's motion. These are the helicity states. The electron can either be left- or right-handed (e_L vs. e_R), depending on whether the spin is pointed against or aligned with the direction of motion.

One of the features of the ILC will be that it can polarize the incoming electron and positron beam. From the ILC design report[5], we read that the electron beam will have a polarization of 80% and the positron beam will have a polarization of about 30%. The total cross section of two partly polarized beams can be written as

$$\begin{aligned}\sigma_{P_{e^-}P_{e^+}} = &\frac{1+P_{e^-}}{2}\frac{1+P_{e^+}}{2}\sigma_{LL} + \frac{1+P_{e^-}}{2}\frac{1-P_{e^+}}{2}\sigma_{LR} + \\ &\frac{1-P_{e^-}}{2}\frac{1+P_{e^+}}{2}\sigma_{RL} + \frac{1-P_{e^-}}{2}\frac{1-P_{e^+}}{2}\sigma_{RR}.\end{aligned} \tag{2.42}$$

The quantities σ_{LR}, σ_{RL} etc. denote the cross section for an interaction with a fully polarized electron and positron beam (respectively with left-handed e^- and right handed e^+ and vice versa). These cross section can be calulated using the spin projection operators,

$$\frac{1+\gamma^5}{2} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}, \tag{2.43}$$

$$\frac{1-\gamma^5}{2} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}. \tag{2.44}$$

The projection operators can be used to take out only the left or right handed part of the fermion. For example, if we make the replacement

$$\bar{v}(p')\gamma^\mu u(p) \to \bar{v}(p')\frac{1+\gamma^5}{2}\gamma^\mu u(p), \tag{2.45}$$

we effectively set the contribution from the left handed part of the spinor to zero, while the right handed part remains unchanged. This we can use to reinsert the spin sums over the

	6-point	5-point	4-point	3-point	2-point
$e^+e^- \to e^+e^-H$					
number	-	20	44	348	98
CPU time	-	33%	11%	47%	9%
$e^+e^- \to \nu_e\bar{\nu}_eHH$					
number	74	218	734	1804	586
CPU time	67%	13%	10%	8%	2%

Table 2.1: Percentage of CPU time spent on the various N-point loop integrals. Data taken from [18].

amplitude squared and use the completeness relations (equations 2.40) when calculating on of the specific cross sections $\sigma_{L/R,L/R}$.

Support for polarized initial particles, and spin information of the outgoing particles, is work in progress in the GRACE system. A complete treatment of polarization is therefore outside the scope of this report.

2.7 Evaluation of the loop integrals

In an automated system for the calculation of processes up to the one-loop order, the evaluation of the loop integrals is one of the most important ingredients. The evaluation of these internal integrals is one of the most time consuming parts of the calculation (see table 2.1).

GRACE handles the one-loop integrals by first rewriting them into simple scalar integrals and subsequently using `Fortran` code to evaluate the scalar integrals numerically.

2.7.1 Tensor reductions

In figure 2.2 a generic loop diagram with N external legs is depicted. The legs have momenta p_i, $1 \leq i \leq N$. Momentum conservation tells us $\sum p_i = 0$.

The Feynman diagram encodes a tensor integral of rank M corresponding to a N-point graph,

$$T^{(N)}_{\underbrace{\mu\nu\ldots\rho}_{M}} = \int \frac{\mathrm{d}^n l}{(2\pi)^n} \frac{l_\mu l_\nu \ldots l_\rho}{D_0 D_1 \ldots D_{N-1}} \tag{2.46}$$

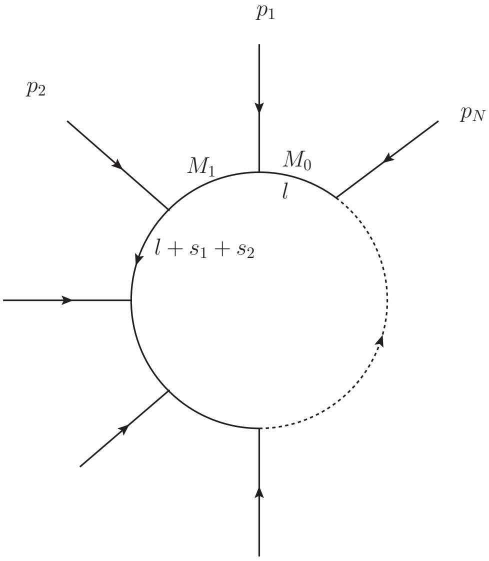


Figure 2.2: General structure of the N-point loop integral. l is the loop momentum, M_i are the masses of the particles circulating in the loop and p_i are the external momenta. The momenta s_i are sums of external momenta and the loop momentum.

with

$$D_i = (l + \sum_{j=1}^{i} s_i)^2 - M_i^2 \tag{2.47}$$

$$s_i = \sum_{j=1}^{i} p_i \tag{2.48}$$

If $M = 0$, this is a scalar integral. However, loop momenta can appear in the numerator when we deal with internal fermion lines or vertices that contain momentum terms. Then we end up with a tensor integral ($M > 1$). It is known that any tensor loop integral can be decomposed in a few basic scalar integrals, by using for example the Passarino-Veltman scheme[19]. This reduction is called tensor reduction. Additionally, all ($N > 4$)-point loop integrals can be expressed in integrals up to Lorentz order 4. So in the end the calculation usually boils down to only a few types of integrals, with different parameters for each diagram.

Reduction of rank The reduction of the tensor integrals is possible due to a few properties of $T^{(N)}_{\mu\nu\ldots\rho}$. That is the invariance under permutation of the propagators D_i, and the full symmetry under the Lorentz indices $\mu\nu\ldots\rho$. Furthermore, Lorentz covariance of the integral allows us to decompose the full expression into tensors constructed from the available 'building blocks': the external momenta p_i and the metric $g_{\mu\nu}$.

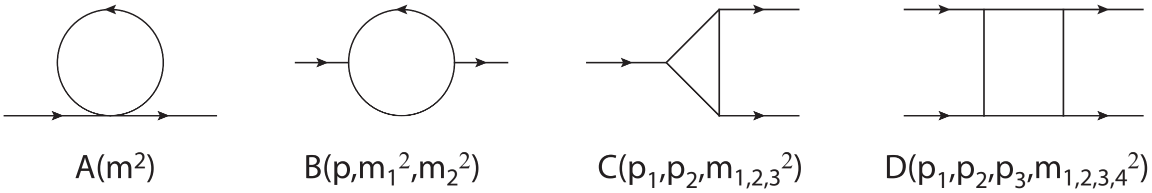


Figure 2.3: The scalar 2-,3- and 4-point functions can be calculated using the `FF` library. Tensor integrals or integrals with a higher number of external particles can be rewritten in terms of these scalar functions.

$$A = A(m^2) \tag{2.49}$$

$$B(p_1, m_{1,2}^2) = B_0(p_1^2, m_{1,2}^2) \tag{2.50}$$

$$B^\mu(p_1, m_{1,2}^2) = p_1^\mu B_1(p_1^2, m_{1,2}^2) \tag{2.51}$$

$$B^{\mu\nu}(p_1, m_{1,2}^2) = g^{\mu\nu} B_{00}(p_1^2, m_{1,2}^2) + p_1^\mu p_1^\nu B_{11}(p_1^2, m_{1,2}^2) \tag{2.52}$$

$$C(p_1, p_2, m_{1,2,3}^2) = C_0(p_1^2, p_2^2, m_{1,2,3}^2) \tag{2.53}$$

$$C^\mu(p_1, p_2, m_{1,2,3}^2) = p_1^\mu C_1(p_1^2, p_2^2, m_{1,2,3}^2) + p_2^\mu C_2(p_1^2, p_2^2, m_{1,2,3}^2) \tag{2.54}$$

$$C^{\mu\nu}(p_1, p_2, m_{1,2,3}^2) = g^{\mu\nu} C_{00} + p_1^\mu p_1^\nu C_{11} + p_2^\mu p_2^\nu C_{22} + (p_1^\mu p_2^\nu + p_2^\mu p_1^\nu) C_{12} \tag{2.55}$$

$$\vdots \tag{2.56}$$

An explicit example follows for the rank 1 two point function: $B^\mu(p, m_1, m_2)$:

$$B^\mu = \int \frac{d^4k}{(2\pi)^4} \frac{k^\mu}{\left[(p^\mu + k^\mu)^2 - m_1^2\right]\left[k^2 - m_2^2\right]} \tag{2.57}$$

$$p_\mu B^\mu = \int \frac{d^4k}{(2\pi)^4} \frac{p \cdot k}{\left[(p^\mu + k^\mu)^2 - m_1^2\right]\left[k^2 - m_2^2\right]} \tag{2.58}$$

$$p \cdot k = \frac{1}{2}\left[(p^\mu + k^\mu)^2 - p^2 - k^2 - m_1^2 + m_2^2 + m_1^2 - m_2^2\right] \tag{2.59}$$

$$\begin{aligned} p_\mu B^\mu = {} & \frac{1}{2} \int \frac{d^4k}{(2\pi)^4} \frac{1}{(k^2 - m_2^2)} - \frac{1}{2} \int \frac{d^4k}{(2\pi)^4} \frac{1}{(p^\mu + k^\mu)^2 - m_1^2} \\ & - \frac{1}{2} \int \frac{d^4k}{(2\pi)^4} \frac{p^2 - m_1^2 + m_2^2}{\left[(p^\mu + k^\mu)^2 - m^2\right]\left[(k^2 - m^2)\right]} \end{aligned} \tag{2.60}$$

$$B_1 = \frac{1}{2p^2}\left[A_0(m_2^2) - A_0(m_1^2) - (p^2 - m_1^2 + m_2^2) B_0(p, m_1, m_2)\right] \tag{2.61}$$

We notice $B^\mu(p, m_1, m_2)$ can be rewritten in terms of the scalar two point function and the one point function. Using this method all tensor integrals can be reduced to scalar integrals.

Reduction of number of external particles Since the four dimensional spacetime is spanned by four Lorentz vectors, the momenta are not all linearly independent in $(N > 4)$-point loop integrals. In particular, we can express the loop momentum in the basis that is spannend by the external legs,

$$l^\mu = \sum_{i,j=1}^{4} (s_i \cdot s_j)^{-1}_{ij} (l \cdot s_i) s_j^\mu. \tag{2.62}$$

Doing so, we can either re-express the denominators using this identity (scalar derived method). Or we can re-express the numerators using this identity (vector derived method). This yields relations between the tensor coefficients and the scalar coefficients. Therefore, in the end we only need the scalar 2-,3- and 4-point functions and relevant coefficients to describe all possible tensor integrals appearing in one-loop calculations.

2.7.2 Evaluation in GRACE

The GRACE system utilizes a slightly modified version of the Passarino-Veltman scheme to reduce the tensor integrals. All 2-,3- and 4-point tensor reductions are performed by taking derivatives of the Feynman parameters x_i that occur when combining all propagators in a loop, known as Feynmans trick,

$$\frac{1}{D_0 D_1 \ldots D_{N-1}} = \Gamma[N] \int [dx] \frac{1}{\left(D_1 x_1 + D_2 x_2 + \ldots + D_0 \left(1 - \sum_{i=1}^{N-1} x_i\right)\right)^N}, \tag{2.63}$$

$$\int [dx] \equiv \int_0^1 dx_1 \int_0^{1-x_1} \ldots \int_0^{1-\sum_{i=1}^{N-2} x_i} dx_{N-1}. \tag{2.64}$$

Then, the generic tensor integral is written as

$$T^{(N)}_{\underbrace{\mu\nu\ldots\rho}_{M}} = \Gamma[N] \int [dx] \mathcal{T}^{(N)}_{\underbrace{\mu\nu\ldots\rho}_{M}}, \tag{2.65}$$

$$\mathcal{T}^{(N)}_{\underbrace{\mu\nu\ldots\rho}_{M}} = \int \frac{d^n l}{(2\pi)^n} \frac{l_\mu l_\nu \ldots l_\rho}{(l^2 - 2l \cdot P(x_i) - M^2(x_i))^N}. \tag{2.66}$$

Since this expression will already be regularized (see section 2.8 and 2.9) the integral over the loop momenta will be dealt with first by reducing it to a scalar integral recursively. This is done analytically in the `FORM` part of the GRACE-system. Then, solutions for integrals over the Feynman parameters of the remaining expression must be calculated. This is done numerically (`Fortran`) by using libraries such as `FF`[20] or `LoopTools`[21]. The implementation is described in detail in the report on GRACE-loop[18].

Both the scalar derived method and the vector derived method are implemented in GRACE.

2.8 IR divergence regularization

Infrared divergences appear when ultrasoft massless particles are exchanged or emitted. In the electroweak sector the photon causes these divergences. It has been shown that these divergences will disappear at every order of perturbation if one includes extra diagrams in the calculation: soft photon radiation. This can be done with impunity since in an experiment one cannot distuingish a particular interaction from the same process with soft photon emission included.

Of course we should avoid the appearance of infinities (although they cancel) during the calculation because we are dealing with numerical values instead of algebraic entities. One can therefore introduce an infrared cutoff in the loop integral, or the infrared divergences can be regulated by giving the photon a small mass m_γ. The dependence on this fictious mass will cancel when one includes diagrams with soft photon emission.

Figure 2.4 shows a general process at tree level in which a photon is emitted and reabsorbed at the external legs, making it a one-loop process. This yields a logarithmic infrared divergence because the photon propagator depends inversely on the momentum of the photon k. In the loop integral over k this is troublesome for $k_\gamma \to 0$.

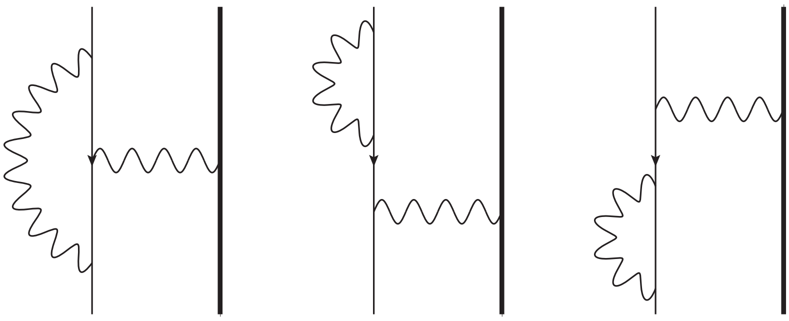


Figure 2.4: A general QED process at tree level with a $\mathcal{O}(\alpha)$ correction in which a photon is emitted and reabsorbed at the external legs, yielding infrared divergences.

However, diagrams in which a soft photon is emitted also contain IR divergences. Figure 2.5 shows two Bremsstrahlung processes at the tree level. This type of diagrams is also divergent for $p_\gamma \to 0$, since the propagator of the particle that has emitted the photon hits its pole.

It can be shown that these divergences cancel precisely at all orders in α[10]. Therefore, in order to deal with the IR divergence, one needs to include soft photon emission in every one-loop diagram in electroweak theory. This is done automatically in GRACE by approximating the contribution from this photon emission.

First we define a value k_c which acts as a cut for the momentum of the soft photon. We

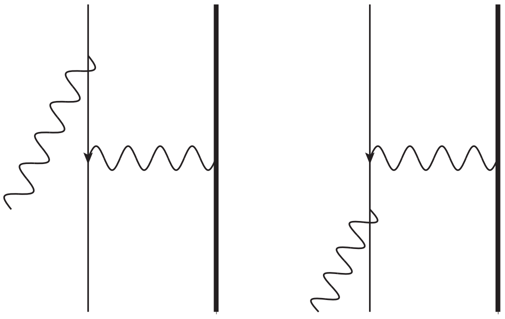


Figure 2.5: Brehmstrahlung diagrams, also yielding a logarithmic infrared divergence.

take $k_c \gg m_\gamma$ but small enough to ensure that the brehmstrahlung differential cross section factorizes as[22]:

$$d\sigma_{\text{soft}}(m_\gamma, E_\gamma < k_c) = d\sigma_{\text{Tree}} \times \delta_{\text{soft}}(m_\gamma, E_\gamma < k_c) \tag{2.67}$$

The value k_c needs to be a lot smaller than the center of mass energy of the process. However if the process contains a narrow resonance (e.g. the s-channel diagram $e^+e^- \to \mu^+\mu^-$) this condition tightens since one needs to take care that the photon with momentum $E_\gamma < k_c$ will not shift the particles off the resonance.

We have now introduced a dependence on the cut k_c. We get rid of this dependence by including all other photon emission, that is by including all photon emission diagrams with $E_\gamma > k_c$. For this we do not use an approximation formula, but we perform the full tree-level $2 \to N+1$ phase space integral.

2.9 UV divergence regularization and renormalization

The integrand of a loop integral may diverge for large values of the loop momentum. This is a UV divergences. We define D for the superficial degree of divergence of a loop integral of the form $\int^\infty \mathrm{d}^4k\, k^{D-4}$,

$$D = \begin{cases} 0 & \text{Logarithmically divergent} \\ 1 & \text{Linearly divergent} \\ 2 & \text{Quadratically divergent} \end{cases} .$$

The actual degree of divergence of a process can be less than D since divergences might cancel (for example by gauge invariance).

Figure 2.6 shows a general diagram at the one-loop level in which an ultraviolet divergence

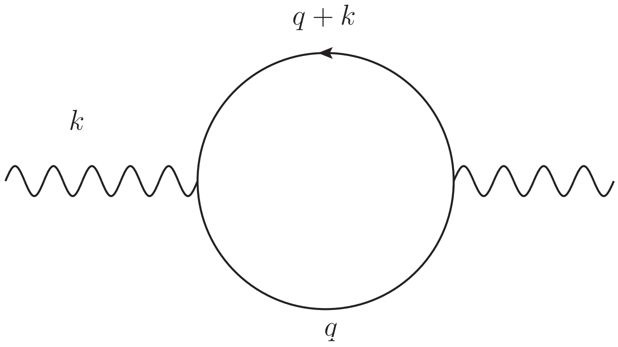


Figure 2.6: A general diagram at the one-loop level in which an ultraviolet divergence occurs (the vacuum polarization graph in QED).

occurs. The corresponding expression is (using the Feynman-'t Hooft gauge)

$$\Pi_{\mu\nu}(k) = \frac{ie^2}{(2\pi)^4} \int d^4q \, \mathrm{Tr} \left[\gamma_\mu \frac{1}{i\not{q}+m} \gamma_\nu \frac{1}{i(\not{q}+\not{k})+m} \right] \tag{2.68}$$

This diverges quadratically for large q: an ultraviolet divergence. By writing the expression as a d-dimensional integral instead of a 4-dimensional one (dimensional regularization), it will appear that the infinities can be written as simple poles in ϵ[10],

$$2\epsilon \equiv 4-d. \tag{2.69}$$

The poles always occur in a specific form, resulting from the Euler gamma function that appears when performing dimensional regularization:

$$C_{\mathrm{UV}} = \frac{1}{\epsilon} - \gamma_E + \log 4\pi \tag{2.70}$$

Evaluating the expression for our divergent diagram, we conclude that

$$\Pi_{\mu\nu}(k) = \frac{-e^2\mu^\epsilon}{6\pi^2}(k^2\eta_{\mu\nu} - k_\mu k_\nu)C_{UV} + \text{finite terms} \tag{2.71}$$

We can cope with the infinite terms by using renormalization[10]. This allows us to get physical results. The free parameters in the SM, such as the masses, are chosen in such a way that they have a physical meaning at the tree level. This direct relation between experimental quantities and parameters in the Lagrangian is destroyed through the higher order corrections in the theory. The UV divergent contributions in loop diagrams result in experimental quantities that differ from the bare parameters.

Fortunately, in the SM the UV divergences that appear in the Feynman diagrams cancel in expressions for physical quantities[23, 24]. One can use the counter term approach to yield meaningful predictions from loop calculations. In the counter term approach one writes the bare parameters as finite renormalized quantities and divergent terms. Also the fields can be replaced by renormalized fields. The procedure works as follows:

1. Choose a set of independent parameters of the model
2. Write the bare parameters as renormalized quantities and renormalization constants (see below)
3. Choose renormalization conditions to fix the counterterms (see below)
4. Express the physical quantities in terms of the renormalized parameters
5. Use experimental results to fix the values of the renormalized parameters
6. Predict values for physical quantities by using the renormalized quantities

Going back to the specific UV divergence in equation 2.71, we note that we can get rid of this divergence by modifying the Langrangian with a counterterm,

$$\mathcal{L}' = \mathcal{L} + \Delta\,\mathcal{L}\,. \tag{2.72}$$

$\Delta\,\mathcal{L}$ must be chosen in such a way that it precisely cancels the divergent term from the original Lagrangian. In this case

$$\Delta\,\mathcal{L} = \frac{-e^2\mu^\epsilon}{24\pi^2}\,(\partial_\mu A_\nu + \partial_\nu A_\mu)^2\,C_{UV} \tag{2.73}$$

This gives rise to a two photon vertex (shown in figure 2.7). This vertex is included in the counterterm library for GRACE so in the end, after regularizing in C_{UV} and summing all diagrams, there are no infinities and there is no dependence on C_{UV}.

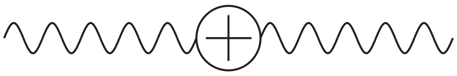


Figure 2.7: The counterterm vertex that cancels the divergent diagram in figure 2.6.

The on shell renormalization scheme In the GRACE system, the on shell renormalization scheme is used[25]. This means that the counterterms are chosen such that the finite renormalized parameters are equal to the physical quantities in all orders of the perturbation theory. Renormalization of the parameters is sufficient to obtain finite S-matrix elements, but in order to remove the divergences in the propagators and vertices, one needs to renormalize the fields too.

The input parameters for the on shell renormalization in the α-scheme are

- $\alpha(q^2 = 0) = 1/137...$, the electroweak coupling constant in the Thomson limit
- M_Z and M_H, the Z and Higgs boson mass
- G_μ, the Fermi constant
- The fermion masses

With this parameters, the W-mass is calculated and the electroweak running coupling is present in the charge renormalization counter terms. Nowadays, the mass of the W-boson is measured with high precision so one can alternatively use the G_μ-scheme. In this scheme, the value of α is expressed in terms of the vector and scalar boson masses and G_μ. As a consequence, the EW corrections are independent of the light quark masses. This scheme is used in for example [26].

2.10 Kinematics

When the matrix element has been constructed and has been squared, one can integrate over the phase space to calculate the cross section of the process. In this section, we only consider the kinematics of a collider having two incoming beams of particles, so there will be an axial symmetry. We take the incoming particles to move on the z-axis, so the natural coordinate system will be cylindrical if the particles have no intrinsic angular momentum.

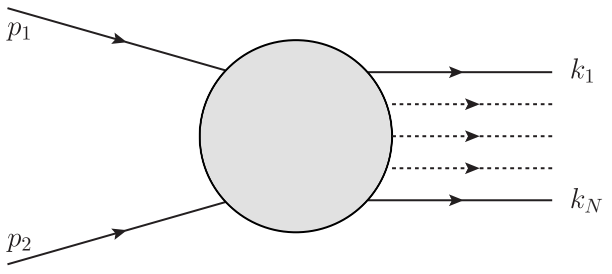


Figure 2.8: Picture of generic 2 → N collision

The general formula for the cross section of two colliding particles ($p_1\ p_2 \to \sum_i k_i$, see figure 2.8) can be written as

$$\begin{aligned}\mathrm{d}\sigma =&\frac{1}{2\lambda^{1/2}(s,m_1^2,m_2^2)}(2\pi)^4\delta^{(4)}(p_1+p_2-\textstyle\sum_i k_i) && (2.74)\\ &\times|\mathcal{M}(p_1,p_2;k_i)|^2\prod_i \mathrm{d}^4k_i\ \delta^{(4)}(k_i^2+m^2)\theta(k_i^0). && (2.75)\end{aligned}$$

We can cast the formula for the cross section is a simpler form by using the identity

$$\int \mathrm{d}^4p\ \delta^{(4)}(p^2+m^2)\theta(p^0)F(p) = \int \frac{\mathrm{d}^3p}{2\omega(\vec{p})}F(\vec{p}). \tag{2.76}$$

This holds due to the fact that the composition $\delta(g(x))$ can be rewritten as a sum of delta functions of x over its roots. Only one root survives because of the positive energy requirement $\theta(p^0)$. $\omega(\vec{p})$ is the energy function $\sqrt{\vec{p}^2+m^2}$. The cross section is then

$$\begin{aligned}\mathrm{d}\sigma =&\frac{1}{2\lambda^{1/2}(s,m_1^2,m_2^2)}(2\pi)^4\delta^{(4)}(p_1+p_2-\textstyle\sum_i k_i) && (2.77)\\ &\times|\mathcal{M}(p_1,p_2;k_i)|^2\prod_i \frac{\mathrm{d}^3k_i}{(2\pi)^3 2\omega_i(\vec{k}_i)}. && (2.78)\end{aligned}$$

In this formula, s is the square of the incoming centre-of-mass energy (using Minkovski metric diag$(1,-1,-1,-1)$) and λ is introduced for convenience,

$$\begin{aligned}s &= (p_1+p_2)^2, && (2.79)\\ \lambda(x,y,z) &= x^2+y^2+z^2-2xy-2xz-2yz. && (2.80)\end{aligned}$$

We already see from equation 2.77 that energy and momentum are conserved in the collision (hence the four dimensional Dirac delta function). We take the outgoing particles on shell, just like the incoming ones. This adds another N equations:

$$p_i^2 - m_i^2 = E_i^2 - \vec{p}_i^{\,2} - m^2 = 0 \tag{2.81}$$

The number of degrees of freedom for N outgoing particles is $4N$, but subject to these constraints only $3N-4$ are left.

2.10.1 2 → 2 process

A two particle final state can be parametrized by two variables since only the angular part of the cylindrical system is of relevance. If there are no polarizations involved, one is left with only one variable since the symmetry is axial. One can rewrite the phase space integral into an integral performed in the center of mass (CM) frame.

$$\int \frac{1}{(2\pi)^6} \frac{\mathrm{d}^3 p_1}{2\omega(\vec{p_1})} \frac{\mathrm{d}^3 p_2}{2\omega(\vec{p_2})} \delta^{(4)}(p_1 + p_2 - P) = \frac{1}{16\pi^2} \frac{p^{\mathrm{CM}}}{E^{\mathrm{CM}}} \int_0^{2\pi} \mathrm{d}\phi \int_{-1}^{1} \mathrm{d}\cos\theta \tag{2.82}$$

E^{CM} is the total energy in the center of mass frame and p^{CM} is the sum of the three momentum of each of the outgoing particles. This is a very convenient form of the phase space integral as we will see when the Monte Carlo integration is introduced.

The integration variable $\cos\theta$ can be changed to another integration variable if desired. For example, we can express t (the transferred momentum $(p_3 - p_1)^2$) using our integration variable $\cos\theta$ and the relativistic invariant s (the total energy $(p_1 + p_2)^2$). In the center of mass frame, the momenta of particles 1 and 2 are equal and opposite. If they have the same mass, they must have the same energy too. After the collision, the particles again move with equal and opposite momentum, so $E_1 = E_2 = E_3 = E_4$ and $|\vec{p_1}| = |\vec{p_2}| = |\vec{p_3}| = |\vec{p_4}|$.

$$\begin{aligned} t &= (p_3 - p_1)^2 = 2m_e^2 - 2p_3 p_1 \\ &= 2m_e^2 - E_3 E_1 + 2p_3 p_1 \cos\theta \\ &= 2m_e^2 - 2(s/4) + 2\cos\theta(s - 4m_e^2)/4 \\ &= -\frac{s - 4m_e^2}{2}(1 - \cos\theta) \end{aligned} \tag{2.83}$$

As we will see, switching the integration variable can be very useful if you expect singular behaviour of the matrix element in another variable.

2.10.2 General process

The kinematics of a process with more than two outgoing particles can be treated as if it is a decay chain. That means for three outgoing particles, the final state first splits into particle 3 and the system of particles 4 and 5. After that the system decays into particle 4 and particle 5.

$$1 + 2 \to 3 + q, \;\; q \to 4 + 5 \tag{2.84}$$

This can be repeated if more final particles are present, so in the end the GRACE system builds up the N-particle phase space iteratively by nested integrations over the invariant mass Q_i^2 of the bound systems and solid angle Ω_i of each outgoing particle i. This is demonstrated in figure 2.9.

2.10.3 Challenges

Dealing with kinematics can be very hard in a automated system since one needs to deal with a large number of numerical challenges during the phase space integration. One of

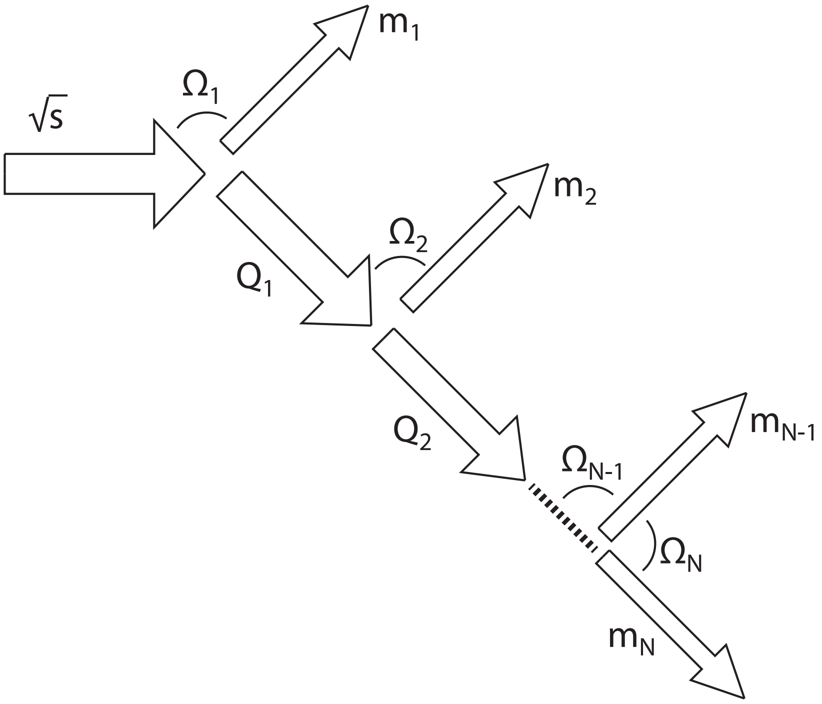


Figure 2.9: Building up the phase space of N outgoing particles

these problems is singular behaviour of the matrix element. This means during the phase space integration, a very large contribution comes from a very small region in the phase space. A naive Monte Carlo integration scheme will have a very hard time trying to get an accurate result for the integral since it will have to compute a large number of data points to get sufficient information on the peak.

Fortunately, already one of the causes of singular behaviour, the occurence of a pole in the propagators, has been dealt with by choosing a nonlinear gauge that takes out the term with the pole but still retains gauge parameters in the vertices that can be used to check the gauge invariance.

However, the matrix element can still exhibit singular behaviour due to other causes. For example t-channel electromagnetic interactions often involve a $1/t$ factor. This means the electromagnetic scattering has an infinite cross section. Fortunately, this is not observed since the case $t = 0$ means no scattering at all, so the incoming particles stay in the beampipe region. This can be read from equation 2.83.

We can compute a finite cross section by implementing cuts on the phase space. Kinematical cuts can be essential for obtaining finite cross sections in t-channel diagrams. For example taking $|\cos\theta^{\text{CM}}| < 0.99$, which means no particles will be detected in the beampipe region up to an angle of about 8° with respect to the beam pipe.

Although the singular region has been excluded, we still get the most contribution from that area. If we know this will occur we can implement a mapping of the integration variable to improve the accuracy of the Monte Carlo integration.

2.10.4 Mapping

Applying a mapping is a very effective procedure to deal with singularities in (or other troublesome behaviour of) the integrand. It basically boils down to taking a suitable integration variable.

Assume we are trying to integrate a function of the form $g(x)f(x)$ from x_- to x_+ with singular behaviour of $f(x)$. We will compensate this with a mapping. The primitive of $f(x)$ is $F(x)$. If $f(x)$ is nonnegative over the integration region, we can define a function $u(x)$ that is increasing with x and is bounded between 0 and 1:

$$u(x) = \frac{F(x) - F(x_-)}{F(x_+) - F(x_-)} \tag{2.85}$$

$u(x)$ is now a measure for the amount of the surface under the function. We then rewrite the original integral

$$\int_{x_-}^{x_+} \mathrm{d}x\, g(x)f(x) = \int_0^1 \mathrm{d}u\, g(x(u))f(x(u))\frac{\mathrm{d}x}{\mathrm{d}u} = (F(x_+) - F(x_-)) \int_0^1 \mathrm{d}u\, g(x(u)). \tag{2.86}$$

We have succesfully mapped away the problematic function $f(x)$ by rewriting the integration variable from x to u. As a result, the primitive of the problematic function has appeared in our integral.

Let us compute a mapping of the problematic $1/t$ integral that occurs in the scattering of electrons. In this case of course $F(t) = \log t$ and $u(t) = \log(t/t_-)/\log(t_+/t_-)$. The result is

$$\int_{t_-}^{t_+} \mathrm{d}t\, t^{-1} = \log(t_+/t_-) \int_0^1 \mathrm{d}u\, g(t(u)), \text{ with} \tag{2.87}$$

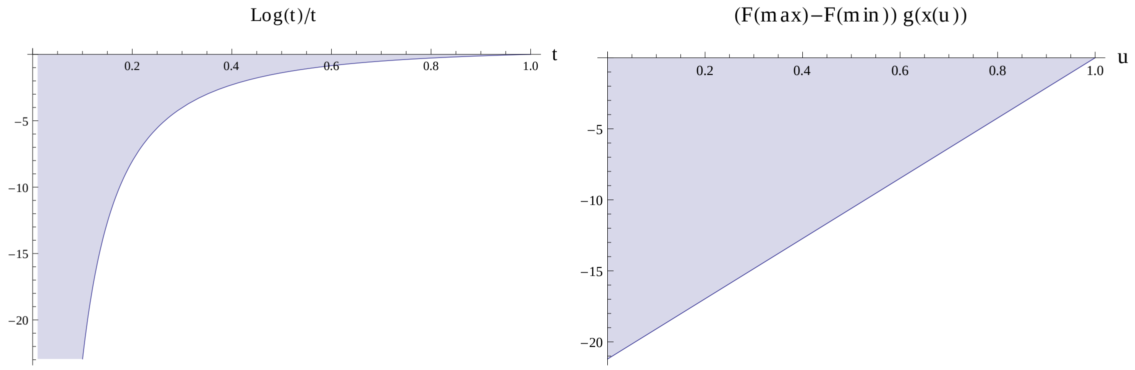
$$t(u) = t_- \exp(u \log(t_+/t_-)) \tag{2.88}$$

Both integrals are shown in figure 2.10, where we have chosen $g(t) = \log t$.

We note that this is a much more suitable form of the integral to be used in a Monte Carlo integration if g only depends weakly on t.

2.11 Phase space integration

In a collider, the incoming particles and their four-momentum are known precisely. The matrix element for a given process is then only dependent on the outgoing momenta of



(a) The function before the mapping, the domain is $t \in [0.01, 1]$

(b) The function after the mapping, the domain is $u \in [0, 1]$

Figure 2.10: The function $(\log t)/t$, before and after the mapping of $t \to u(t) = \log(t/t_-)/\log(t_+/t_-)$. The value of the integral is in both cases equal to -10.604, but the integration of the right-hand side function is much easier to compute using a Monte Carlo integration method.

the generated particles. In order to calculate the cross section of a process (differential cross section) we need to integrate the squared matrix element over the full outgoing phase space.

The integration of the matrix element is highly nontrivial because the number of integration variables can be quite large and the matrix element may behave very singular. One can deal with the singularities by using lots of integration points but then the computation time becomes very large.

Various phase space integration packages have been constructed that try to cope with this problem. Most notable are BASES[27] (used in GRACE) and the routines of the Cuba Library[28]. One can read more on how to cope with problematic integrations in section 5.3.

Numerical integration is usually performed by histogramming the integrand in all required variables (N_{dim} dimensions). One method of histogramming is the Monte Carlo integration method. This method works by randomly sampling the integrand (we call the number of sampling points N) and gives an error that scales as $1/\sqrt{N}$, independent of the number of dimensions N_{dim}. This will not give the exact value of the integral, but one can increase the number of bins to get a better estimate. However, the time required for the integration grows exponentially with the number of variables N_{dim} and linearly with the number of sampling points. This means we need to employ some clever techniques to enhance the precision of the integral without taking too much time. This is explained in section 5.3.

Mathematically, performing an integral by histogramming is calculating

$$I = \int d\vec{x} f(\vec{x}) = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} f(\vec{x}_i), \tag{2.89}$$

where the values $\vec{x}_i$ are random function arguments, distributed evenly over the domain of the function. To allow the Monte Carlo integration method to finish in finite time, a finite number of samples N is picked and the value I_N is calculated instead.

$$I_N = \frac{1}{N}\sum_{i=1}^{N} f(\vec{x}_i), \tag{2.90}$$

We can show that the error scales as $1/\sqrt{N}$ by looking at the variance,

$$\sigma^2 = \int d\vec{x}(I - f(\vec{x}))^2 \tag{2.91}$$

We can relate σ^2 and I_N by performing the integration multiple times and noting that

$$\int d\vec{x}(I - f(\vec{x})) = 0. \tag{2.92}$$

If we take the unit measure, i.e. all $x_i \in [0,1]$, we can show

$$\sigma^2 = \frac{1}{N}\int d\vec{x}_1 \ldots d\vec{x}_N \left(\sum_{i=1}^{N}(I - f(\vec{x}_i))\right)^2 \tag{2.93}$$

$$= N\int d\vec{x}_1 \ldots d\vec{x}_N \left(I_N - I\right)^2. \tag{2.94}$$

Therefore, the error of the Monte Carlo estimate, $I_N - I$, scales with $1/\sqrt{N}$.

2.12 Event generation

The integration of the matrix element over the phase space provides the total cross section. From the phase space integration, one can make plots of the contribution to the cross section versus the integration variables. This shows how every kinematical region contributes to the total cross section. This can help greatly when one wants to distinguish signals from the backgrounds. However, to do a proper comparison with experimental data one should use an event generator.

Each point of the phase space is associated to an event probability. The probability info is stored during the phase space integration. This is subsequently used to randomly generate events closely mimicking experimental data. which means a random point in the phase space is picked. This is called event generation.

2.13 Luminosity

The luminosity L is a quantity that can is used to predict the number of events detected at a collider (N_p) for a given process if the cross section of that process (σ_p) is know. The following formula holds assuming that all particles can be detected:

$$\frac{\mathrm{d}N_p}{\mathrm{d}t} = L(t)\sigma_p \tag{2.95}$$

The unit of luminosity is therefore $cm^{-2}s^{-1}$. Obviously you want your collider to have a luminosity as high as possible. This can be achieved by increasing the number of beam crossings per second or by increasing the number of particles per beam. One should take care that the luminosity is not too high however, since the detectors might not be able to differentiate between various collisions.

An important quantity is the (time) integrated luminosity:

$$L_{\text{int}}(T) = \int_0^T \mathrm{d}t\ L(t) \tag{2.96}$$

The integrated luminosity relates the cross section directly to the number of events. It is usually given in fb^{-1} (femtobarn), a non SI unit that is equivalent to $10^{-39}cm^{-2}$. Fermilab has produced $10fb^{-1}$ in the first decade of the 21st century. Fermilab's Tevatron took about 4 years to reach $1fb^{-1}$ in 2005, while two of CERN's LHC experiments, ATLAS and CMS, reached over $5fb^{-1}$ of proton-proton data in 2011 alone.

The proposed International Linear Collider (ILC) will deliver luminosities of a few $10^{34}cm^{-2}s^{-1}$, or $500fb^{-1}$ over a few years integrated luminosity, see section 1.2.1.

Chapter 3

The GRACE system

During the last decades, the Minami Tateya group at KEK in Japan has been working on a system called GRACE[2]. GRACE is an (almost) fully automated system for calculating cross sections and event rates for arbitrary particle collisions. This is possible since diagrams are constructed based on a set of definite rules (the Feynman rules) and are therefore relatively easy to implement in a computer program. The user only needs to provide the initial and final particles, and the theoretical model (e.g. the Electroweak Model) that is to be used.

3.1 Programming languages: the right tool for the right job

The GRACE system is written in three different languages: `C`, `FORM` and `Fortran`. In short: the C program takes care of the user input, the algebra engine `FORM` will generate an expression for the matrix element and will in the end create `Fortran` source code files. These `Fortran` files will perform the phase space integral over the matrix element, resulting in the cross section and event information.

3.1.1 On `FORM`

> Mathematica is user friendly, FORM is computer friendly
> — J.A.M. Vermaseren

`FORM`[9] is a programming language for speedy and large scale manipulation of formulas, and is therefore heavily used in the GRACE system. The algebraic manipulations required in matrix element calculations are very often simple, local operations that have to be applied to millions of terms. This means we do not require the knowledge of the whole expressions at any given moment. General purpose computer algebra programs, such as Maple or Mathematica, easily hit the memory limits when dealing with expressions of this size, as they hold the whole expression in memory at every point in the program. This makes them less usable for working on very large expressions. The FORM programming language provides only local manipulations of expressions. This is reflected by the memory model, in which an expression is merely a stream of terms. Only a single term needs to reside in memory at any moment while the rest of the expression is stored on a storage device.

FORM has a lot of built in functionality to deal with matrix element calculations effectively, such as identities for traces, levi-cevita tensor and delta functions. In previous versions of the GRACE system, the symbolic program REDUCE was used, but this part was replaced by its counterpart written in FORM when it proved to be too slow for the required manipulations.

3.1.2 On Fortran

In the GRACE system Fortran, specifically FORTRAN77, is used to perform the phase space integration. Fortran is a general-purpose language that is especially suited to numeric computation and scientific computing. Although the language has a feature set similar to that of C, it was praised for its speed back in the days. Up until and including Fortran77, language design considerations had optimization as a main focus. Due to the state of compiler theory and technology, this often meant restricting features and capability in order to give the compiler the best shot at optimizing the code. The lack of pointers and ignoring possible aliasing of memory allows the Fortran compiler to generate more efficient code than a C compiler would have generated. Nowadays, the differences are less distinct but due to historical reasons, a lot of scientific code is still written and maintained in Fortran.

Since many libraries for computations in particle physics are written in Fortran, such as BASES for the phase space integration, SPRING for the event generation and LoopTools for the evaluation of tensor integrals, the GRACE system will produce Fortran code for the matrix element calculation.

3.2 GRACE (tree level)

The earlier versions of GRACE can only calculate cross sections at the tree level. Two approaches can be taken:

Squared Matrix elements This is the straight forward way of dealing with quantum field theory. As the number of external particles increases, the number of Feynman graphs grows very rapidly, so this method can take a substantial amount of time.

Spinor technique Spinors and gamma matrices are dealt with in a numerical way. A library of all possible types of vertices and propagators are defined as subroutines to be called for a numerical calculation. The CHANEL library for GRACE and the HELAS library for MadGraph are examples of such subroutines. The automatic system calls these libraries according to the structure of the Feynman graphs. The result is a complete `Fortran` program for the numerical calculation of differential cross sections without calling any other package. Since this method calculates helicity amplitudes directly, it is natural to calculate polarized cross sections.

3.3 GRACE-loop

In order to achieve higher precision in calculating physical results from an iteraction, such as the cross section, one needs to include radiative corrections. These radiative corrections are higher order perturbations in the theory, which add a lot of complexity mainly due to the loop integrals that appear. The higher order calculations do not only serve as precision tests of the theory, but they are also sensitive to not directly accessible details of the theory. This might give hints for undiscovered, heavy particles.

The loop integrals that appear beyond the tree level give rise to divergences[10], see also section 2.9 and 2.8. The UV divergences can be counteracted by so called counterterms that are included a priori in the theory. Therefore, an automated system such as GRACE-loop needs a library of counterterms. For a given theoretical model, one can have different libraries reflecting the fact that one can choose between different renormalization and regularization schemes. In GRACE, ultraviolet divergences are regularized through dimensional regularization and the electroweak sector is renormalized on-shell. The infrared divergences should also be dealt with. In GRACE this is done by giving the photon a small mass.

3.4 GRACE operation

The standard way to calculate cross sections is as follows:

1. Specify the process.
2. Choose an appropriate model of particles and vertices.
3. Fix the order of perturbation (tree and/or loop).

4. Enumerate all possible diagrams.
5. Write down the amplitudes per diagram.
6. Prepare the kinematics for the final particles.
7. Integrate the amplitude squared over the phase space of final particles, including experimental cuts, if necessary.
8. Generate events such that the simulation of the process in a detector is constructed.

The GRACE system can do the following:

- All diagrams for a given process at the tree or one-loop order.
- Drawings of the diagrams.
- `FORM` source code containing an algebraic expression for the matrix element
- `Fortran` source code, generated by `FORM`, containing numerical code for the matrix element.
- Interface routines to the `Looptools` library (`FF`) which contains subroutines to evaluate the inner loops.
- Default code for dealing with the kinematics.
- Interface routines to the multi-dimensional integration package BASES.
- Interface routines to the event generation package SPRING.
- A test program for the gauge invariance check of the generated amplitude at a specific phase space point.

Figure 3.1 shows how all components of the GRACE system are connected. In the next sections all components are explained.

3.4.1 User input

To define a process, one needs to create a file called `in.prc`. A sample file follows which contains the information on a collision between a electron and a positron, yielding a photon, an electron and a positron:

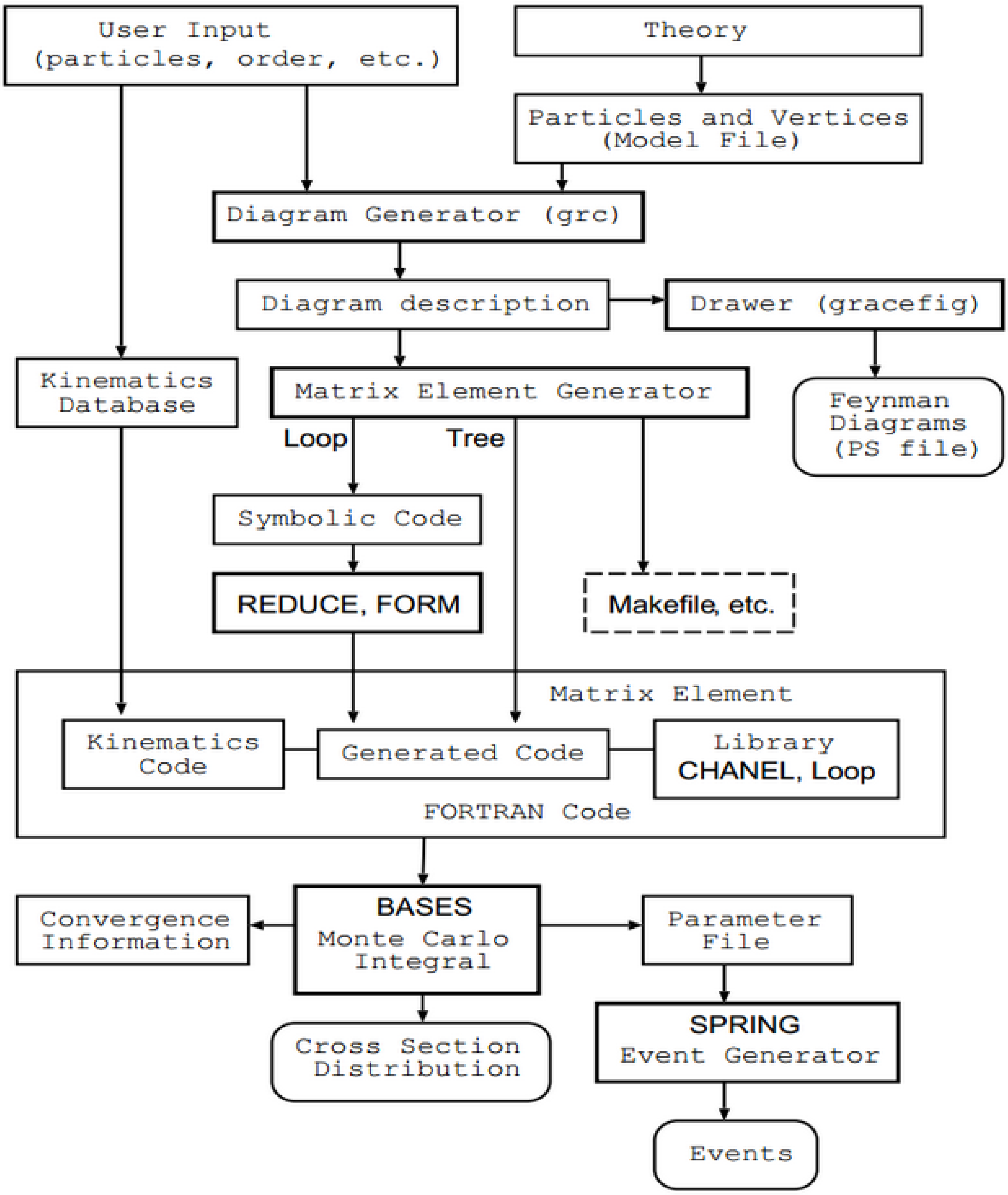


Figure 3.1: GRACE system flow

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Model="nlg.mdl";
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Process;
```

```
  ELWK={5,3};
  Initial={electron positron};
  Final  ={photon electron positron};
  Expand=Yes;
  Block=No;
  AnyCT=Yes;
  Kinem=”2302”;
Pend;
```

The lines beginning with “%” are ignored as comment lines.

The first uncommented line specifies the file which describes the model used for graph generation. `nlg.mdl` is the name of the model file that implements the electroweak model in the nonlinear gauge. This file contains the definition of all particles, propagators, vertices and counter terms.

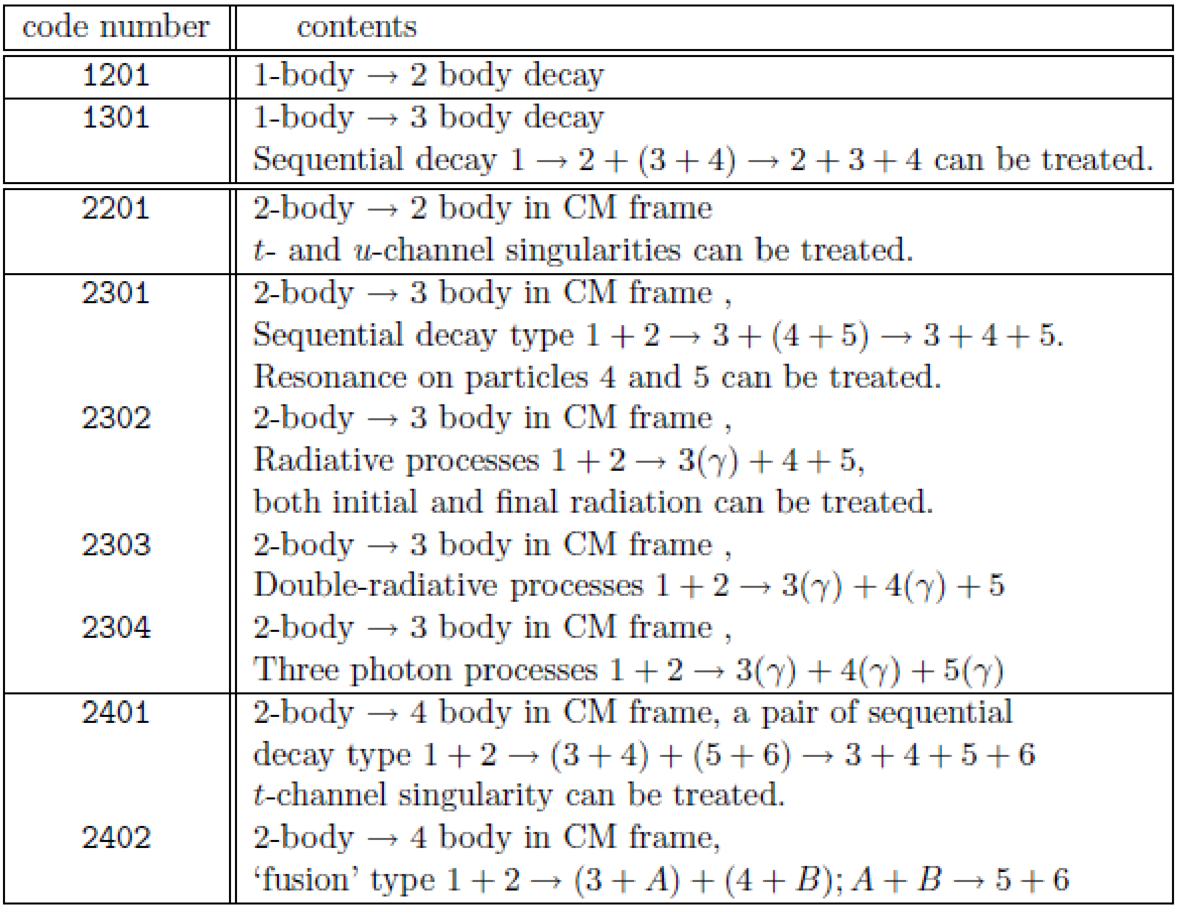
Then follow descriptions of the processes, beginning with the line `Process;` and ending with the line `Pend;`. In this block, descriptions are given for the order of the coupling constants, initial particles, final particles and some options. In this case, both the tree level (3 vertices) and the 1-loop level (5 vertices) are requested. The initial and final external particles are given as lists of particle names, which are defined in the model file.

The name of the coupling constant (“ELWK” in the above example) is defined in the model file. When the value of the coupling constant is given as a list of numbers, the program generates graphs for each value of the coupling constants with the same external particles.

The option `Kinem` defines the kinematics of the process. The first digit tells the number of incoming particles, the second digit the number of outgoing particles. The remaining digits are used to choose a specific type of process. This information is used to optimize the convergence of the phase space integral by using appropriate mappings. In this case, ‘02’ means the emission of one photon. The full list of available kinematical settings can be found in figure 3.2.

3.4.2 Theory (particles and vertices)

The GRACE system comes with various model files. The model files contain all particles and vertices, and additionally all counter terms for your theory. For the one-loop system, the GRACE system provides four different models: `qed`, `nlg`, `nlg_FF` and `nlg_LT`. The first one is meant for QED calculations only, using the nonlinear gauge. The latter three are written for the electroweak model in the nonlinear gauge. They describe the same physics, but the implementation of the loop calculations differ. `nlg` uses a home brewn algorithm for the loop integrals, `nlg_FF` uses the `FF` library, and finally `nlg_LT` uses the `LoopTools` library.



code number	contents
1201	1-body → 2 body decay
1301	1-body → 3 body decay Sequential decay $1 \to 2 + (3+4) \to 2+3+4$ can be treated.
2201	2-body → 2 body in CM frame t- and u-channel singularities can be treated.
2301	2-body → 3 body in CM frame , Sequential decay type $1+2 \to 3 + (4+5) \to 3+4+5$. Resonance on particles 4 and 5 can be treated.
2302	2-body → 3 body in CM frame , Radiative processes $1+2 \to 3(\gamma)+4+5$, both initial and final radiation can be treated.
2303	2-body → 3 body in CM frame , Double-radiative processes $1+2 \to 3(\gamma)+4(\gamma)+5$
2304	2-body → 3 body in CM frame , Three photon processes $1+2 \to 3(\gamma)+4(\gamma)+5(\gamma)$
2401	2-body → 4 body in CM frame, a pair of sequential decay type $1+2 \to (3+4)+(5+6) \to 3+4+5+6$ t-channel singularity can be treated.
2402	2-body → 4 body in CM frame, 'fusion' type $1+2 \to (3+A)+(4+B); A+B \to 5+6$

Figure 3.2: Available settings for the kinematics in GRACE, taken from the documentation file `kinem_ap.pdf`

By default, all vertices are included in the models. However if desired, one can easily leave some particles or vertices out of a model by deleting them from the main model file and making some small changes to the resulting `Fortran` files. This is very useful if the number of generated diagrams is very large. For example, one can leave out the coupling between the scalars (Higgs and χ_3) and the electron, since their contribution will be negligible. This can typically reduce the number of tree and loop diagrams by a factor of two. This means a fourfold reduction in terms in the expression for the squared amplitude. One should not be concerned with the change in counter terms that belongs to this change to the vertices, since the associated counter terms are also negligible. Since the tree diagrams containing counter terms are of the same order in α as the 1-loop diagrams, it is not very profitable to leave these out as it would reduce the number of terms in the squared amplitude only marginally.

3.4.3 Feynman graph generation

The automatic Feynman graph generator `QGRAF`[29] is used in the GRACE system. QGRAF is embedded in the `grc` routine. `grc` reads the process file (`in.prc`) and the model file and feeds that information to `QGRAF`. The output is a text file called `out.grf`. This file contains the structure of all diagrams of the requested order. For example, the s-channel diagram $e^+ \; e^- \to Z \to t \; \bar{t}$ with a photon dressing the $Z \; t \; \bar{t}$-vertex is encoded as follows:

```
Graph=19;
Gtype=0;
Sfactor=-1;
Vertex=4;
   0={      1[positron]};
   1={      2[electron]};
   2={      3[t]};
   3={      4[t-bar]};
   4[order={1,0}]={      1[electron],      2[positron],      5[z]};
   5[order={1,0}]={      5[z],      6[t-bar],      7[t]};
   6[order={1,0}]={      3[t-bar],      6[t],      8[photon]};
   7[order={1,0}]={      4[t],      7[t-bar],      8[photon]};
Vend;
Gend;
% Fline=2;
%     1[type=1;length=3]={ [-1,n:    1,0], [1,n:    4,0], [0,n:    0,-1]};
%     2[type=1;length=5]={ [-1,n:    2,0], [0,n:    6,1], [1,n:    5,2],
%          [1,n:    7,0], [0,n:    3,-1]};
% Fend;
```

The fermion line information is provided so the `FORM` program can take care of the Feynman rules.

A separate routine, called `gracefig`, is constructed to draw the generated Feynman diagrams. It provides a graphical interface to browse all diagrams, or it can produce an EPS file of the diagrams. All diagrams in chapter 7 are generated using `gracefig`.

3.4.4 The matrix element

`grcform` is the part of GRACE-loop that performs the algebraic calculations. It is written in `C` and `FORM` and consists of several parts, explained in the next sections.

Generating files

First, the generated Feynman graphs are read and a primitive expression for the amplitude of each diagram is generated (i.e. of the form ‘incoming positron’-‘eeγ vertex’-‘incoming electron’-‘outgoing photon’). This expression is written down in a `FORM` file, together with additional information such as particle assignments to legs and momentum substitutions from the vertices. Because we are calculating the value of the matrix element squared, each amplitude for tree diagrams is squared and all expression for loop diagrams are multiplied with each expression for the tree diagrams.

Additionally, the grcform routine writes auxiliary `FORM` and `Fortran` files necessary to perform the next steps. These files are hardcoded in the `.fin` and `.vin` files that are bundled with the GRACE system. Also a `Makefile` is generated so the next steps can be automated.

Applying the Feynman rules

After that, the `FORM` files are executed. The job of the generated `FORM` files is to take the algebraic expressions for the amplitudes, apply the Feynman rules and finally write down numerical `Fortran` code for the calculation of the matrix element. This means that traces are performed, gamma matrix identities are inserted and various other algebraic simplifications take place.

In the end, the result is a series of `Fortran` routines that only take kinematical variables and return the contribution for the matrix element.

The `Fortran` files calculate various kinematical quantities such as the inner products of the external momenta. Then, these quantities are used to calculate the amplitude per diagram (squared). The routines for the loop diagrams rely heavily on the external library `LoopTools` for a fast calculation of inner loops.

Output optimization The resulting algebraic expression can be very large, if the number of external particles is large and the full standard model is used to generate all diagrams. This can be very problematic (please refer to section 5.1) so we try to reduce the number of terms in the expression. In appendix A two methods are presented to reduce the number of operations (multiplication, summation, etc.) in large expressions. Both methods are used in the GRACE system.

The first one makes linear shifts in quantities with the same dimension in order to simplify the formulas. These quantities are kinematical quantities, like dot products, levi civita tensors and masses, coupling constants and the nonlinear gauge parameters. If the number of these quantities increase, this can take a considerable amount of time since the algorithm scales quadratically (a shift happens between every two parameters of the same dimension).

However, the results can be very good, usually the number of terms is reduced tenfold.

The second method is a general output optimization scheme that tries to reduce the number of operations by using lots of intermediate variables[30]. This has been implemented recently in FORM. The results are again very good, the reduction in the number of operation is about 80%.

In the following listing we see the optimization of an expression that occurs in the GRACE system. We start with 914 terms. By using linear shifts this is reduced to 100 terms. Calculating these 100 terms would take 577 operations, but after the output optimalization it takes only 94 operations.

```
~~~y12x1(0.02 sec): There are 4 coupling constants.
Starting doshift with 914 terms
  yk1 = yk1+(-2*yk5);         number of terms = 806
  yk2 = yk2+(yk1);            number of terms = 794
  yk4 = yk4+(-yk8);           number of terms = 704
  yk5 = yk5+(-yk4);           number of terms = 698
  yk7 = yk7+(-2*yk5);         number of terms = 608
  yk1 = yk1+(-yk2);           number of terms = 560
  yk2 = yk2+(2*yk5);          number of terms = 482
      .....
  yk7 = yk7+(4*yk3);          number of terms = 296
  yg1 = yg1+(-1);             number of terms = 198
  xcp1 = xcp1+(xcp4);         number of terms = 138
  xcp2 = xcp2+(xcp3);         number of terms = 108
  xcp4 = xcp4+(-1/2*xcp1);    number of terms = 100

*** STATS: original  0P 483M 94A : 577
*** STATS: optimized 0P 58M 36A : 94
```

Building executables

The final step is to build executable files from all generated Fortran code. Three executables are built:

test0, which gives the value of $|\mathcal{M}|^2$ at a specified point in the phase space. This routine can be used as a quick check to see if all went well and a proper numerical answer comes out.

test1 gives the value of $|\mathcal{M}|^2$ at a specified point in the phase space for two different choices of the gauge parameters. This is used to check if the result is truly gauge invariant.

`integ` is the program for the phase space integration. If the gauge check has been passed, the simplest choices for the gauge parameters can be chosen in order to speed up the calculation.

3.4.5 Phase space integration

The integration over phase space is not a job for a symbolic algebra engine (except in the rare cases where it can be done analytically), so the `FORM` program generates a `Fortran` routine that will calculate the value of $|\mathcal{M}|^2$ at a given point in phase space. That is: take a list of random values, convert these values into kinematical quantities and calculate the value of the matrix element at that particular point in phase space. This routine is then fed to an integration program. There are a lot of integration programs available for phase space integration in high energy physics, such as BASES[27] and the four integration routines in the CUBA library[28]. The GRACE system uses BASES for the numerical integration.

The integration of the matrix element is highly nontrivial because the number of integration variables can be quite large and the matrix element may behave very singularly. One can deal with the singularities by using lots of integration points but then the computation time becomes very large. The integration package BASES handles this problem by only dividing the variables that cause the singular behaviour in a great number of hypercubes. This number of variables is usually quite low, so the integration is still fast since most variables can be integrated over in the ordinary way. More on this can be found in section 5.3.

GRACE contains a library for kinematics that is used to convert the random numbers in the Monte Carlo routine to kinematical quantities. In order to improve the convergence of the Monte Carlo integration, one can use options to signal that a particular behaviour of the matrix element is to be expected, such as a $1/t$ pole. The kinematics library then automatically applies a suitable mapping to cope with this singular behaviour (see section 2.10.4). All options are listed in the `dbkinem` documentation[31].

After the phase space integration, GRACE will list the calculated cross section in units of picobarn (pb).

Identical particles The statistical factor for the identical particles is not generated automatically by the GRACE system and should be added by hand. This is because this factor depends on the observables one is interested in. For the total cross section one should divide the result obtained by BASES by the factorial of the number of identical particles.

3.4.6 The IR regularization procedure

By default, the GRACE system generates `Fortran` code for the summation of the tree and 1-loop contribution (equations 2.1 and 2.2). In order to include the soft photon contribution (equation 2.3) one needs to edit the file `ampsum.f`, as described in section B.2.1.

In this way, the values of $|\mathcal{M}|^2$ will not depend on the photon mass λ that plays a role as regulator. However, by including the soft contribution, the result is now dependent on the soft photon energy cut k_c. This value is defined in the `dat22` file. In order to remove this dependence we need to include the hard photon scattering. This is done by preparing a new input file that contains the same process but with one additional external photon. We need this process only at the tree level. It is important to choose the correct kinematics for this process (usually 2302 or 2403, see table 3.2). Although this is only a tree level process, the computing time can still be quite large since the time needed for the phase space integration increases exponentially with the number of external particles.

The sum of the two cross sections should be independent of k_c.

3.4.7 Event generation

The GRACE system uses the SPRING[27] event generator. SPRING utilizes the probability information gathered by BASES in the integration step.

3.5 Controlling parameters in the GRACE system

For each calculation, a file is generated that contains a list of parameters that can be varied. For historical reasons, this file is called `dat22`. By default, it is generated like this:

```
   roots  500.0d0
   ncall  12000
   itmx1  5
   itmx2  5
c
   zmass  91.187d0
   wmass  80.22d0
   tmass  170d0
   hmass  126.d0
   gam-z  0d0
   gam-w  0d0
   lamda  1.d-17
   k-cut  0.001d0
   uvdiv  0
   mu2     1.d10
   delta  1.d10
   ired   1
   cscut  10
   ff-on  1
   redon  2
   nlalp  0
   nlbet  0
   nldel  0
   nleps  0
   nlkap  0
c   memo   334  4
c   fname  zzzzzzyyyyyyxxxx
```

Unfortunately, the first four parameters are ignored. These can be set in the file `kinit.f`. The next lines contain masses, widths, the value for λ (photon mass regulator), k_c (soft photon cut), C_{UV} (UV divergence regulator).

The value C_{UV} is used in the UV divergent part of the 2-, 3- and 4-point loop integrals and of course in the corresponding counter terms. In the end, all the UV divergent parts should cancel. Since 5-point functions in the electroweak sector are UV finite by themselves, we can check those separately. This is done with the `mu2` and `delta` regularization parameters. These are used internally by the `LoopTools` library. The first one is the dimensionful parameter introduced to keep the integral's mass dimension the same in all dimensions D. The second one replaces the actual divergence, like C_{UV} does for the counter terms. We can choose this value different from C_{UV} since the five point functions will be UV finite and thus do not depend on this quantity, although they are reduced to multiple divergent 4 point functions.

The parameter `ired` determines the algorithm used for the tensor reduction of the five point functions. By default it is set to 1, meaning the scalar derived tensor reduction scheme as described in section 2.7.1 is used. However, this fails when one encounters the pentagon diagram with rank 4 that occurs in for example $e^+e^- \to ZHH$. By setting the parameter equal to 2, the vector derived scheme is used.

`cscut`, `ff-on` and `redon` are only used in older loop libraries.

The nonlinear gauge parameters are called `nlalp` through `nlkap`.

The values for the particle masses in this file overrule the values that are set in the file `setmas.f`. This file also contains masses for the particles that are not listed in the `dat22` file, such as the electron mass.

3.6 Alternative systems

There are a lot of systems available for automatic calculation of certain types of interactions. Of course, numerous systems cope with QCD corrections on proton-proton collision and the resulting jet generation as seen in the LHC. But only a few systems cope with the collision of point particles in the electroweak sector while keeping all masses. GRACE is one of these systems, and currently only one other system is capable of performing the same calculations (the Feynarts system).

3.6.1 FeynArts-FormCalc-LoopTools

FeynArts is a Mathematica package for the generation and visualization of Feynman diagrams and amplitudes. FormCalc is a Mathematica package, written in `FORM`, which calculates and simplifies tree-level and one-loop Feynman diagrams. It accepts diagrams generated with FeynArts and returns the results in a way well suited for further numerical (or analytical) evaluation. The `LoopTools` package is used to calculate the loop integrals numerically, just as in the GRACE system.

Chapter 4

Checks of the system

Checking the results is essential for large calculations like the ones performed by the GRACE system. Fortunately there are some checking methods available to see in an early stage of the computation if the result is correct.

The tests should be performed in quadruple precision at the level of the differential cross section before the phase space integration is performed. All interactions should be included. After these tests one could switch off the very small couplings, thus reducing the computation time.

In chapter 7 these checks have been performed and analyzed.

4.1 Gauge invariance

The core principle of a gauge theory is that the physical quantities should not depend on the choice of the gauge. This can be a very powerful check on the matrix element, since in the nonlinear gauge discussed in section 2.3.2 five parameters can be chosen arbitrarily. If the result is independent of these five parameters, one can confidently assume that no diagrams are forgotten, the symmetry factors are correct and the Feynman rules have been applied correctly.

The GRACE system provides a routine for this check, called `test1`. In this routine the a point in the phase space can be picked and two different settings for the five gauge parameters. For both configurations the matrix element is computed at a specific point in phase space and the relative difference is shown.

The gauge independence of the variables $\xi_{W,Z,A}$ is not checked since these have been set to 1 to avoid problems with longitudinal term in the propagators (occurence of a pole and higher rank tensors).

4.2 Numerical precision

The generated `Fortran` code can be run in double (64 bit) or quadruple (128 bit) precision. The former has about 16 decimal digits, while the latter has about 34 [1].

In the ideal case, the results from a double precision calculation should match the first 16 digits of a quadruple result. Due to numerical artefacts, mainly cancellations, typically the last few digits will not match. This is not a problem as long as the identical parts have a much higher precision than the Monte Carlo error. This will nog be the case if extremely severe cancellations take place. Then it is required to run the full program in quadruple precision.

One can use the quadruple precision to interpret results from the other check. We take the gauge invariance check as an example. If switching the nonlinear gauge parameter $\tilde{\alpha}$ from 0 to 1 yields a 6 digit agreement in double precision, we will want to know what limits this agreement. Running in quadruple precision will usually improve the agreement, and one can conclude that the limit of agreement was due to numerical artefacts. If however the agreement does not improve, the issues are not of numerical origin.

If desired, the full phase space integration can be performed in quadruple precision to check if the finite float precision plays a role in the final answer, but this is obviously more resource intensive. It is therefore recommended to check the numerical stability in several points in the phase space (especially at the edge there might occur heavy numerical cancellations) to see if the full integration should be performed in double or quadruple precision. One can use the routine `test0` in the GRACE system to check the value of the matrix element at a given point in spacetime.

More on numerical issues can be found in section 5.2.

4.3 Permutations of the external legs

By relabeling the external legs the matrix element should of course not change. This check is not implemented in the GRACE system, but can be done manually in the `in.prc` file.

4.4 Lorentz invariance

One can check if the square matrix element (summed over helicity) is Lorentz invariant by boosting the momenta with some value. This has not been implemented in the GRACE

[1] The number of decimals can be calculated as follows: $D = {}^{10}\log(2^{\text{bits}})$. Not all bits are available since the sign and the exponent also need to be stored.

system.

4.5 Ultraviolet finiteness

The physical result should not depend on the dimensional regularization parameter C_{UV} (defined in equation 2.70) that occur in UV divergent diagrams, since all these terms should be cancelled by the counter terms. One can test this by using different settings for the value in the `dat22` file. The finiteness of the 5-point functions can be tested separately by varying the equivalent parameters `delta` and `mu2`, as described earlier. Since μ enters logarithmically, in order to decisively check whether an expression is really independent of it, its value must be varied on a large scale, e.g. from 1 to 10^{10}.

Of course, this test will not tell you if the remaining finite part is correct.

4.6 Infrared finiteness

The fictitious 'infinitesimal' photon mass λ (introduced in section 2.8) should play no role in the final answer of a calculation since it plays only a role as a regulator. If it is chosen sufficiently small, usually in the order 10^{-15}, the result should have no dependence on this quantity. One can check this by varying the value of λ in the `dat22` file. This only holds if one includes the soft Bremsstrahlung contribution, introduced in section 2.8.

The value of the photon cut k_c should also leave the cross section invariant, since the contributions of the soft and the hard photon emission precisely cancel this dependence. Again, this can be checked by varying the value of this parameter in the `dat22` file. In this case, also the hard photon contribution should be included.

Chapter 5

Computational challenges

5.1 Problem size

Due to the large number of diagrams in 1-loop processes and the complex expression due to the internal integrals, the generated `Fortran` code for the matrix element can be several GigaBytes (GB) large. This poses a problem for the compilation proces and of course the time required for the phase space integration. There are a few methods to reduce the matrix element expression:

- Reduce the number of diagrams by leaving out particles or vertices that will have a negligible effect on your calculation

- Ignore the electron mass where possible

- Take particles on shell or use the narrow width approximation

- Ignore QED interactions when QCD couplings are present

Some of these simplifications are easy to use in the GRACE system, such as the first one. One can leave out the coupling between the scalars (Higgs and χ_3) and the electron without affecting your physical result, since these couplings are extremely small. Although these methods can reduce the generated program size the problem is still inherently present, especially if more external particles are involved.

Essentially, the program size challenges us with four major problems which will be described in the following sections.

5.1.1 Generating the code

In case of a 2 → 3 process in electroweak theory at 1-loop order, a few hundreds of MegaBytes of `Fortran` code is produced in a few thousands of `FORM` files. The `FORM` files produce the `Fortran` files that contain the expressions for the matrix element for every diagram. While the generation of the `FORM` files does not take a significant amount of time, the execution of the code can take days on a standard PC. Fortunately this can be parallelized very easily, since every file can be executed separately.

2 → #	tree diagrams	1-loop diagrams	`FORM` code size
2	˜10	˜100	several MB
3	˜25	˜1,000	˜300 MB
4	˜50	˜10,000	˜1GB

The `FORM` files generate `Fortran` files when executed. One can choose between three settings for the output optimization: no optimization, simple output optimization, and optimization by linear shifts and general output optimization. This has a striking effect on the code size, as can be read from the following listing for $e^+e^- \to ZHH$ and ZZH:

	#tree	#1-loop	`Fortran` code size	simple optim	shifts and Horner
ZHH	6	1,807	1.2 GB	502 MB	146 MB
ZZH	4	2,017	2.4 GB	1.1 GB	293 MB

The main advantage of the reduction of the code footprint is of course the reduced number of operations. This makes the Monte Carlo integration much faster. Besides this, the optimized code will be much more compiler friendly as all polynomials have been split up in smaller parts.

5.1.2 Compiling and linking the code

The freshly generated `Fortran` files can be compiled immediately in parallel if enough memory is available. Some diagrams yield expressions that are very hard to deal with for the compiler, sometimes resulting in a memory usage of about 10 GB. This usually means it is better to compile the diagrams in sequential order. The linker must operate on all files so this cannot be parallelized. Linking the files belonging to a large process can take several hours and one should take precautions that enough address space is available for the linker. Most compilers use a 32 bit address space and consequently run into troubles when more than 2GB of object files must be linked (yielding the dreaded “Relocation truncated to fit” errors). One can tackle this problem by assigning a larger memory model to the linker but not all compilers support this option. This memory model should also be applied during the compilation stage of the `Fortran` files. Please note that the linking stage can take over 10 GB of internal memory, severly impacting the time needed if you do not have enough RAM available.

An alternative approach can also be taken. One can split the diagrams in several groups (object size smaller than 2GB totalling) and link these into several executables. A master program should then be written to perform the Monte Carlo integration. This master program picks the phase space points and collects the numerical results from the executables. This approach has the additional advantage over the naive method that the total memory usage during the linking stage is limited, so the computer does not spend time on swapping.

We have implemented this in the GRACE system. Figure 5.1 shows the general idea.

5.1.3 Executing the integration

The Monte Carlo phase space integration of the matrix element is the most time consuming step of the full calculation. A naive integration can take thousands of CPU hours, so parallelization is highly desirable. One can take two approaches in this.

There are several parallel Monte Carlo integration procedures available. Most of them rely on the industry standard Message Passing Interface (MPI). The BASES integration routine included in the GRACE system has a parallel extension (vbasesMPI).

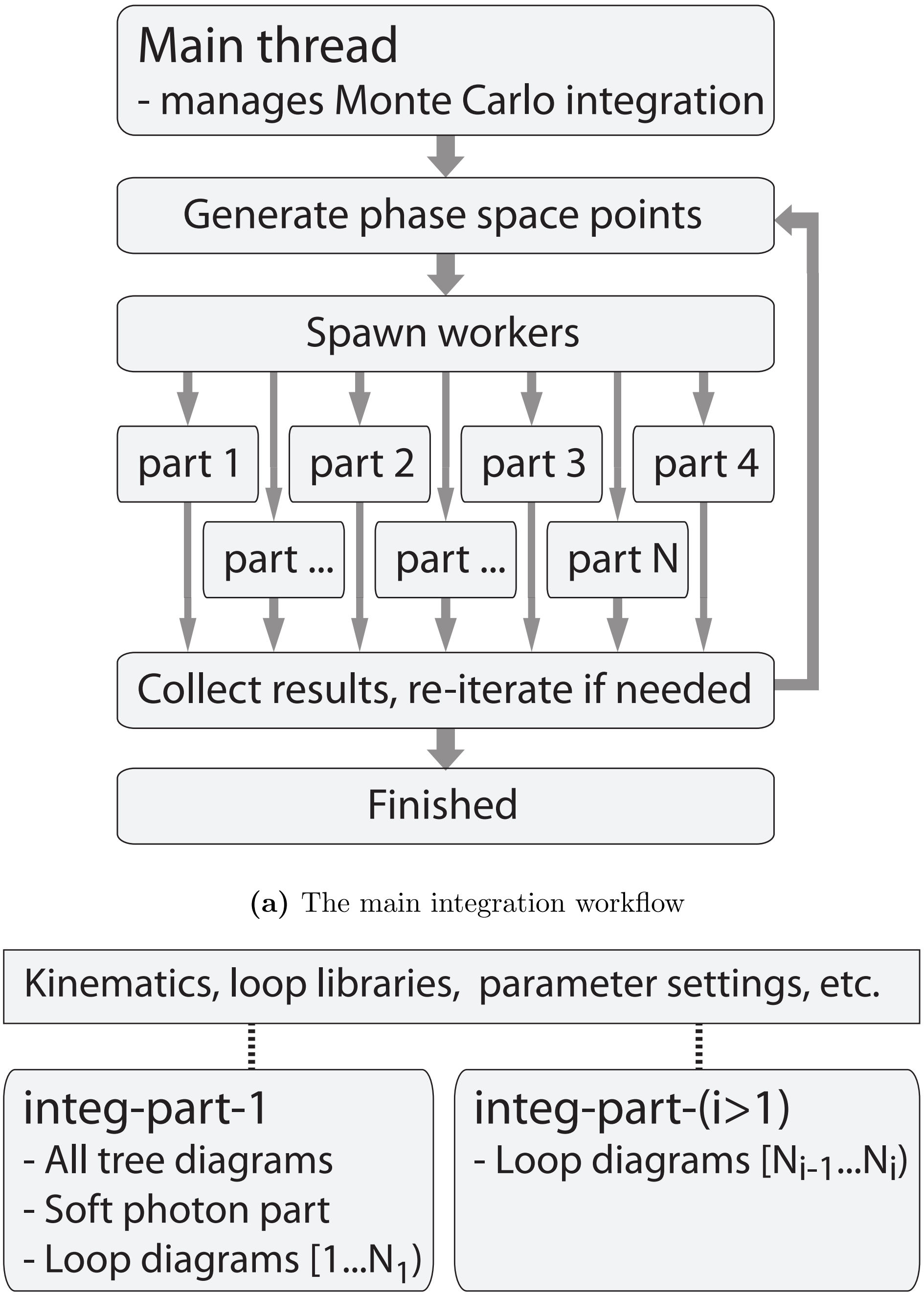
Alternatively, as described in the previous section, several executables can be created that contain parts of the full matrix elements. These executables can be assigned to processors while a master program takes care of the integration procedure. Since the master program only needs to distribute the phase space points, and only collects the numerical results, the overhead from parallelization is very low and one can even use several computers on the network instead of an integrated supercomputer.

5.1.4 Executing the event generation

Executing the event generation takes about the same time as the Monte Carlo integration. No parallel event generator is currently available for GRACE.

5.2 Numerical cancellations

Loss of precision is a problem when dealing with numerical values stored in a computer memory. A computer does not have an infite amount of memory available, so every stored floating point variable has a fixed (though user chosen) precision. The precision of a floating point value, as well as its lower and upper bound, is determined by the number of bits assigned to the variable. Two common choices are 64 (double precision) or 128 bits (quadruple precision). For both cases one bit is used for the sign of the value, leaving 52 +


Main thread
- manages Monte Carlo integration
Generate phase space points
Spawn workers
part 1
part 2
part 3
part 4
part ...
part ...
part N
Collect results, re-iterate if needed
Finished
Kinematics, loop libraries, parameter settings, etc.
integ-part-1
- All tree diagrams
- Soft photon part
- Loop diagrams [1...N$_1$)
integ-part-(i>1)
- Loop diagrams [N$_{i-1}$...N$_i$)

(a) The main integration workflow

(b) The workers

Figure 5.1: Performing the phase space integration by splitting the diagrams in several groups (object size smaller than 2GB totalling) and linking these into several executables. A master program is written to perform the Monte Carlo integration. It picks the phase space points and collects the numerical results from the child executables.

11 (112 + 15) bits for respectively the fraction and the exponent of the double (quadruple) precision value. This constitutes to a precision of 16 (34) decimal digits.

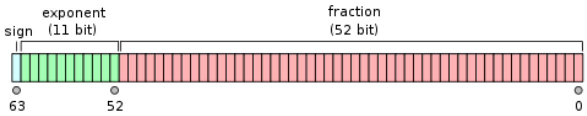


Figure 5.2: Bit layout of a double precision (64 bit) floating point value.

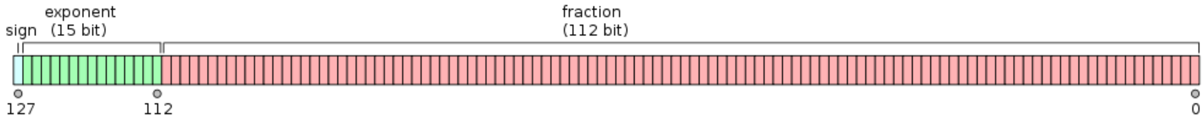


Figure 5.3: Bit layout of a quadruple precision (128 bit) floating point value.

Because of the limited number of decimal places, one can run into problematic numerical cancellations. A simple example shows this for a hypothetical computer with an 6 digit accuracy:

$$\begin{aligned} A &= 1000.00 \\ B &= 1000.00 \\ c &= 1.23456 \end{aligned}$$

$$\begin{aligned} A - B + c &= 1.23456 \\ A + c - B &= 1001.23 - 1000.00 = 1.23000 \end{aligned}$$

We notice the order of the operations can have a disastrous effect on the accuracy of the final answer. This is also obvious if one calculates the mass of a relativistic particle:

$$\begin{aligned} m^2 &= E^2 - p^2 \qquad (5.1) \\ &= (E-p)(E+p) \qquad (5.2) \end{aligned}$$

The former expression is only as precise as E^2 (or p^2) is. This means for the relativistic particle that the uncertainty in E^2 and thus m^2 is quite large compared to the actual value of m^2. The latter expression reduces the complications a bit, since now the value for m^2 will be as precise as the value for $E + p \approx 2E$[1].

This particular problem is circumvented by storing the masses of all particles alongside the value for p and E, but such an easy fix is not always available. This happens for example in gauge cancellations or regulated IR divergence cancellations:

$$\sigma = \infty_{\text{IR}} + \text{finite} - \infty_{\text{brehm}} = \ldots ? \qquad (5.3)$$

[1] For example, a 1 TeV electron has $E^2 = 10^{24}\text{eV}^2$. This value, stored with 16 decimal places, will be precise in terms of 10^8eV^2. The actual value of m^2 is of the order of 10^{11}eV^2 which means the accuracy in the mass calculation is reduced to just 3 decimal places. What a waste if we have 16 decimal places available! For the latter expression the accuracy of m is about 10^{-4} eV which is in practise good enough.

If the phase space integration in double precision does not converge properly, one can check if it does in quadruple precision at the cost of a larger computation time. In the highly unlikely case the integral does not converge in quadruple precision one can only resort to treating the kinematics of the problematic diagrams in an alternative way so the cancellations do not show up in the first place. This technique is used in for example [32] where the problematic convergence of the two photon production of an $\mu^+\mu^-$ pair in an e^+e^- collider is tackeled.

5.2.1 Cancellations in QFT

Numerical cancellations in QFT can destroy the accuracy of any calculation since they tend to be extremely large. A proper regularization scheme will get rid of infinities in a calculation, but will leave a very large quantity instead. This large quantity should cancel another large quantity that appears in an other part of your calculation, and this is where the numerical problems appear. Of course, one can choose the regularization 'better' by cutting of the infinity at a smaller value (for example choosing the value k_c greater when dealing with IR divergences, or increasing the mass of the photon) but this can yield less accurate results at other parts of your calculation. A large photon cut spoils the factorization of the soft brehmstralung, and the photon mass breaks the gauge symmetry.

Gauge cancellations can be circumvented by choosing a proper gauge, for example the axial gauge when calculating processes with a photon in the final state[33, 34]. This yields a dramatic improvement in for example $e^+e^- \to e^+e^-\gamma$:

Amplitude	Non-Axial Gauge	Axial Gauge
$\mathcal{M}_1^2 + \mathcal{M}_2^2$	$0.1116212357 \cdot 10^{13}$	$0.3644158264 \cdot 10^{2}$
$2\mathcal{M}_1 * \mathcal{M}_2$	$-0.1116212356 \cdot 10^{13}$	$0.1546482734 \cdot 10^{3}$
$\lvert\mathcal{M}_1 + \mathcal{M}_2\rvert^2$	$0.1910871582 \cdot 10^{3}$	$0.1910898560 \cdot 10^{3}$

Since most cancellations take place between various diagrams, it is of utmost importance that the phase space points that are picked during the Monte Carlo integration are identical for each diagram, meaning one cannot integrate each diagram separately. This is very unfortunate because one cannot take different mappings and grid adjustments that are suitable per diagram, but a generic optimization must take place.

5.3 Monte Carlo phase space integration

Several methods can be applied to improve on the efficiency of the Monte Carlo integration.

5.3.1 Stratified sampling

The sampling points are chosen arbitrarily (hence the name Monte Carlo integration), but if we choose to divide our phase space into hypercubes and divide the number of sampling points evenly over the cubes, we see that the precision increases. This procedure is called stratified sampling. Let us say we divide a domain X into K subspaces X_k. The Monte Carlo estimate is then

$$I_N = \sum_{k=1}^{K} \frac{\text{Volume}(X_k)}{\text{Volume}(X) N_k} \sum_{i=1}^{N_k} f(\vec{x}_i). \tag{5.4}$$

The variance is

$$(I_N - I)^2 \sum_{k=1} K \frac{(\sigma_k)^2}{N_k}. \tag{5.5}$$

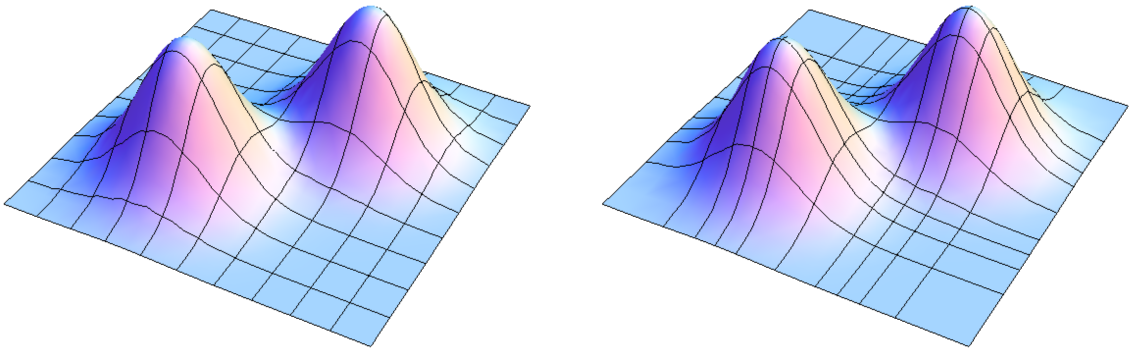
The number of hypercubes grows exponentially with N_{dim}, so if N_{dim} is large we need to refine the stratification so that the number of sampling points remains large for each hypercube. It will appear that in our phase space integral of the matrix element, some variables cause singular behaviour of the integrand, while the integrand only depends weakly on the other variables. We call the variables that cause singular behaviour *wild* variables, and the ones that do not we call *gentle* variables. In order to reduce the number of hypercubes for the stratified sampling, we will only split up the *wild* variables, and handle the *gentle* variables as additional integration variables. This distinction between variables is hardcoded for each kinematical configuration.

In order to get the most out of stratified sampling, it is important to divide the domain X in the best possible way to minimize the error. This is called grid optimization and is explained in the next subsection.

5.3.2 Grid optimization

In section 2.10.4 it is shown how one can deal with severe peaks in the phase space by mapping away the singularities. This will take out the largest numerical problems, but one can only eliminate singularities that are known beforehand. During a real integration of the matrix element, we of course do not have this information on the behaviour of the integrand available. But fortunately there is a numerical method available which is called grid optimization. This also copes with singular parts of the integrand by zooming in (in contrast to map away) at the relevant parts during the phase space integration.

Instead of performing the integral as a whole in one time, various iterations are performed. The full integral volume is covered by a grid of hypercubes, that is initially distributed evenly. After each iteration, the integral value and the variance are calculated by summing up the results from the hypercubes. Then, the grid is adjusted according to the integral



(a) Evenly distributed grid

(b) Optimized grid

Figure 5.4: A two dimensional function overlayed with a regular and an optimized grid. The optimized grid enables us to pick more relevant points (i.e. the ones with high contribution).

value for each hypercube, in such a way that the sizes of the hypercubes are smaller at the parts with larger function value, and larger for the parts with small function value.

In BASES, the full integration is a two step process. First, a few iterations with a small number of sampling points are done to optimize the grid. Then, the data from that integration is dismissed but the grid information is kept and with more sampling points a final integration is performed. This second step has also multiple iterations, after each iteration it is checked if the desired accuracy is reached.

Chapter 6

Main processes in an electron-positron collider

In this section we will focus on the main processes that occur in the linear collider. We can then examine the background of the collider. This will be important for the identification of the Higgs events.

In this project we are interested in the coupling of the Higgs boson to the vector bosons. These can be studied at high precision at a linear electron positron collider. We recall that the Higgs bosons couples to the other elementary particles via the following vertices: fermion-fermion-H, Z-Z-H, W^+-W^--H, H-H-W^+-W^- and H-H-Z-Z. The coupling is proportional to m_f, M_W^2 and M_Z^2. In practise, we ignore the Higgs-electron coupling due to the small electron mass. The Higgs boson can also interact with itself via three or four point vertices. The Higgs boson is a unstable particle, it quickly decays into other particles via the available vertices.

6.1 Two fermion production

Two fermion production processes have clean and simple final states. Precise perturbative predictions of the SM contributions are available. This means ILC experiments will be sensitive to even small deviations from the SM predictions. This allows for the possible discovery of new physics at energy scales far above the center-of-mass energy of the collider.

The single most important process in a electron-positron collider is the exchange of a single photon, i.e. the t-channel process $e^+ \ e^- \rightarrow e^+ \ e^-$ mediated by a photon. All other processes are dominated by s-channel spin 1 exchange due to helicity conservation (exept for the production of $t\bar{t}$ pairs that have a non negligible mass).

6.2 Photon production

The annihilation of the electron and the positron to a single photon is forbidden due to Lorentz invariance. The easiest way to see this is to look at the collision in the center of mass rest frame, which is impossible for the resulting physical photon. Double photon emission on the other hand is perfectly possible, via a t or u channel diagram.

6.3 Higgs production

Since the Higgs boson prefers to couple to heavy particles, at tree level the main production processes are W-fusion and Higgs strahlung via Z-bosons, see figure 6.1,

$$e^+ \, e^- \to ZH \to f\bar{f}H \quad \text{and} \quad e^+ \, e^- \to \bar{\nu}_e \nu_e H \tag{6.1}$$

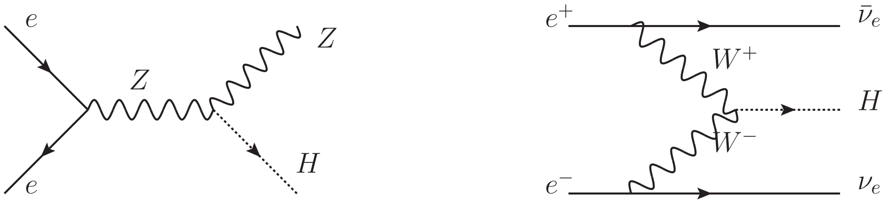


Figure 6.1: Main Higgs production processes: W-fusion and Z-strahlung.

6.4 Higgs decay

The branching ratios for various Higgs masses can be found in figure 6.2. Since the Higgs boson couples to all massive particles in the SM, it can decay via a lot of different channels. The coupling of the Higgs to other particles scales with the mass of these particles, so in general the Higgs will decay to the heaviest particles.

6.4.1 Fermion + anti-fermion

As can be read from the figure, the Higgs most commonly decays to a bottom-antibottom quark pair (56.1% of the time). This is the heaviest fermion-anti-fermion pair that has a combined rest mass that is below 125 GeV, the estimated mass of the Higgs boson. The top-antitop pair is not accessible due to its higher mass.

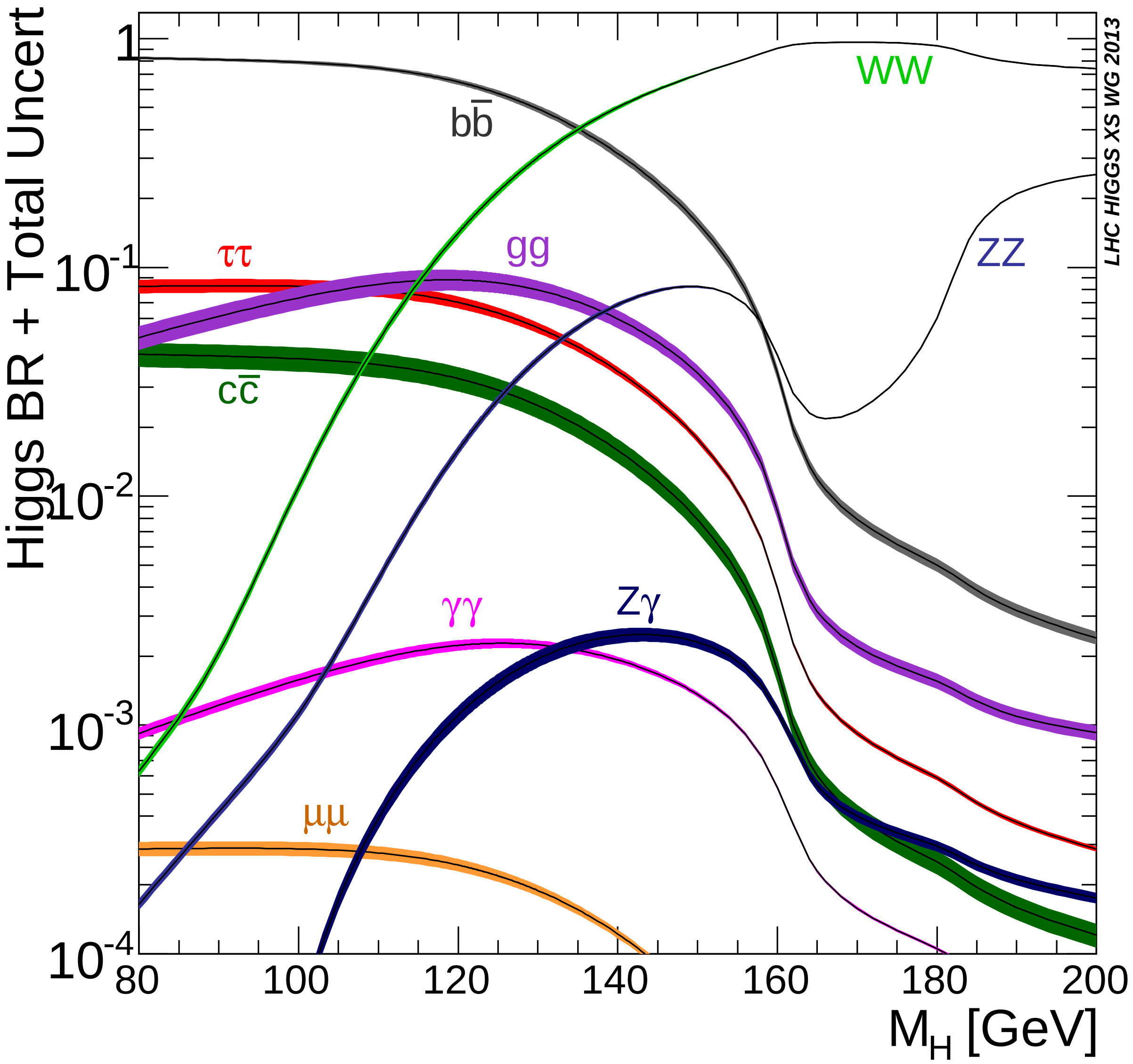


Figure 6.2: Higgs main branching ratios for varying Higgs mass M_H. From the LHC Higgs Cross Section Working Group[35]

The second most common fermion decay is to a tau-antitau pair, which happens about 6% of the time.

6.4.2 Massive gauge bosons

In 23.1% of the cases, the Higgs will decay into a W^+W^- pair. The W bosons will subsequently decay into a quark-antiquark pair or a charged lepton with its corresponding neutrino. Since the neutrinos cannot be detected at the linear collider, this second type of

W decay is not suitable for high precision studies.

The Higgs decay to two Z bosons constitutes 2.9% of the total number of decays. This decay is more convenient to measure because the Z boson decays to a simple fermion-antifermion pair which are in the case of charged leptons detected and reconstructed easily.

6.4.3 Massless gauge bosons

Since the Higgs coupling scales with the mass of the decay product, the Higgs cannot decay immediately to massless particles. The decay can only happen if it is mediated by a loop of virtual (preferably heavy) quarks or massive gauge bosons. In 8.5% of the cases the Higgs boson decays to two gluons, which is the reverse process of gluon fusion. In just 0.2% of the cases the Higgs decays to two photons. Although this is a rare decay it is very important since the photons can be measured easily with high precision.

6.5 Vector boson production

One of the main advantages of the ILC is its staged operation at almost arbitrary CM energies. This offers the opportunity to run the collider at rather low energies on the Z resonance or at the WW threshold to gather large amounts of data and perform precision measurements of the electroweak sector of the SM.

6.6 Vector boson decay

Z bosons decay into a fermion and its antiparticle. W bosons can decay to a lepton and neutrino or to an up-type quark and a down-type quark.

Chapter 7

A case study: $e^+e^- \to ZHH$ and ZZH

We will study the processes $e^+e^- \to ZHH$ and $e^+e^- \to ZZH$ as an example using the GRACE system.

As mentioned in the introduction, one of the primary goals of the future linear collider is a precision study of the properties of the Higgs boson. Although the branching ratios of the Higgs boson have been computed with great precision it is only during the last years that the one-loop electroweak radiative corrections to the main production channel at the ILC, $e^+e^- \to \nu\bar{\nu}H$, have been calculated[36].

An important goal of the electroweak symmetry breaking studies is the reconstruction of the Higgs potential. Although measurements of the Higgs mass and its couplings to the fermions and bosons may give some information on this potential, a more direct probe is the measurement of the Higgs self couplings. In particular, the tri-linear Higgs self-coupling may be accessed in double Higgs production. In electron positron collisions the most promising channel in the ILC is double Higgs-strahlung, $e^+e^- \to ZHH$. In this process, already the tree level diagrams will contain the trilinear Higgs coupling. However, the required luminosity will be quite high for this process since the cross section is small. We will also study $e^+e^- \to ZZH$, because this process has a larger cross section and will access the triple Higgs vertex at the one-loop level.

Some of the important Higgs production processes at tree level in the linear collider are shown in figure 7.1. The cross section is plotted versus the center of mass energy of the colliding beams. At low energy, the production of ZH is dominant, surpassed at higher energies by $\nu\bar{\nu}H$. The production of ZHH is about three orders of magnitude lower, and we will see that the production of ZZH is about the same as ZHH.

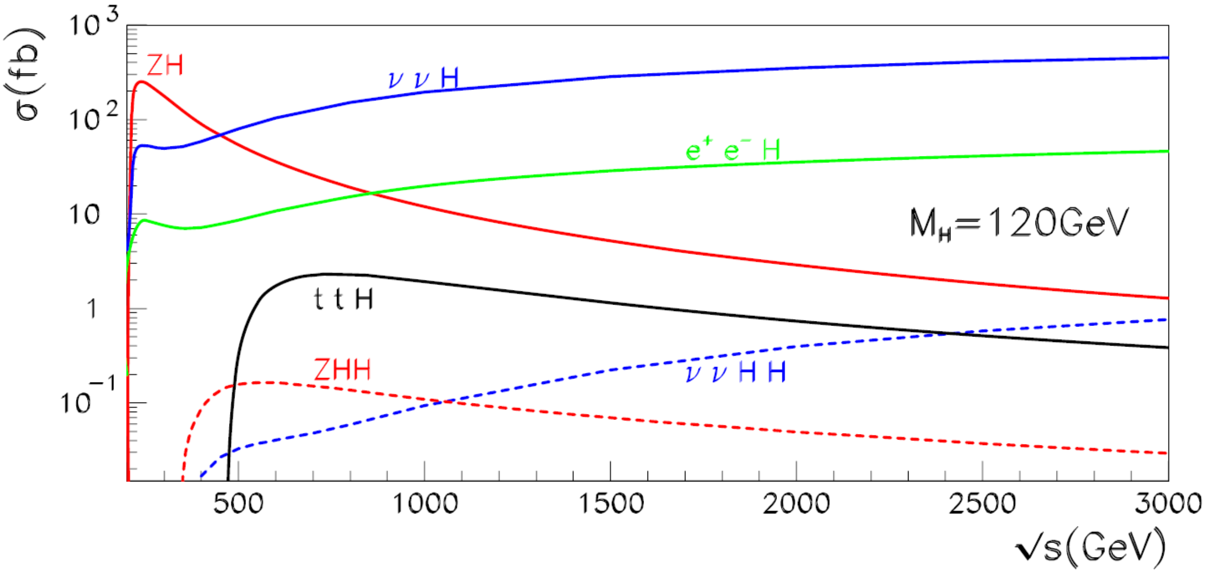


Figure 7.1: Higgs production cross sections for various center of mass energies in the Linear Collider[37]

7.1 Settings

We have used the following settings for the calculations (MadGraph default settings for the heavy particles, but updated Higgs mass):

- $M_Z = 91.188$ GeV
- $M_W = 80.419$ GeV
- $M_t = 172$ GeV
- $M_H = 126$ GeV
- $\Gamma_Z = 0$
- $\Gamma_Z = 0$
- $M_e = 0.51099906$ MeV
- $M_\mu = 105.658389$ MeV
- $M_\tau = 1.7771$ GeV
- $\alpha_0^{-1} = 137.0359895$

7.2 $e^+e^- \to ZHH$

This process has been studied before using an old version of the GRACE system[38], and we will try to reproduce this result. The old version used the `REDUCE` symbolic programming language to perform the matrix element generation. This has been ported to `FORM` after the publication of this article. In addition, the same process has been studied using the Feynarts system[39]. The results are compared in the former paper and have been found to be in agreement.

If we use the nonlinear gauge with its counter terms (26 particles, 409 vertices), the process consists of 27 tree diagrams and 5417 1-loop diagrams.

The GRACE system fails to handle some diagrams involving loops over ghosts and vector bosons if we run at double precision (quadruple works fine, but is very slow). This is being looked at. For the time being, we will leave out the electron-positron-Higgs and the electron-positron-χ_3 vertex. This solves the tensor integral problem and will have no substantial effect on the physical results, since these couplings are extremely small. This is also done in the paper on this process, and is referred to as the ‘production set’ as it is used to quickly calculate the physical results. Without these electron-scalar vertices, the full process consists of 6 tree level diagrams, and 1807 1-loop diagrams.

The production set has the additional benefit that the code size is much smaller:

	#tree	#1-loop	size (optim=0)	(optim=1)	(optim=2)
All vertices	27	5,417		4.5 GB	
No e^+e^- scalar	6	1,807	1.2 GB	502 MB	146 MB

The 6 tree level diagrams can be identified by 4 types of diagrams. These are shown in figure 7.2. The two remaining diagrams radiate the Z boson from the other Higgs leg. The tri-linear Higgs vertex and the $HHZZ$ vertex are the subject of this study.

A center of mass scan of the tree level cross section is shown in figure 7.3. We notice it is peaked at $\sqrt{s} = 600GeV$. The results agree with calculations performed in the paper on ZHH production. There is also agreement with a calculation in MadGraph 5, a comparison is shown in 7.4.

In order to calculate the one-loop corrections to this reaction, we will use the following `in.prc` file:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Model="./nlg2301_LT.mdl";
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Process;
  ELWK={5,3};
   Initial={electron positron};
   Final  ={Z Higgs Higgs};
```

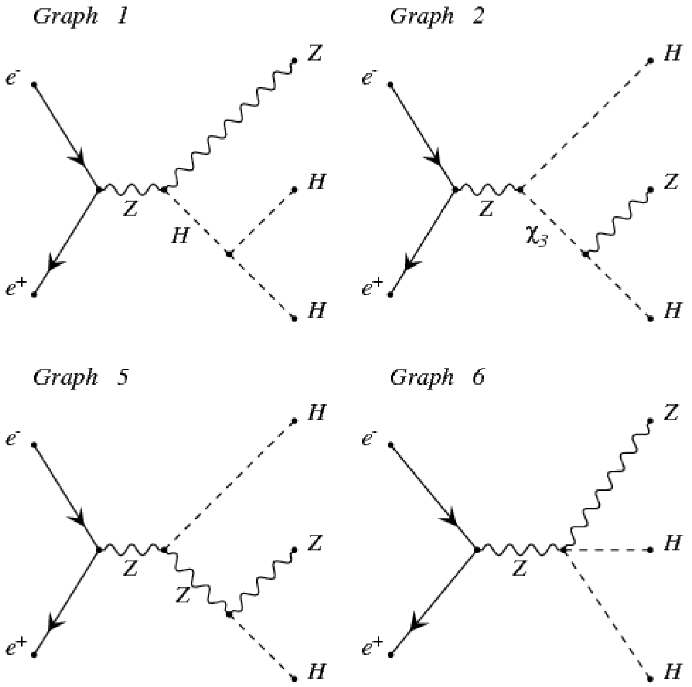


Figure 7.2: Tree level diagrams of $e^+e^- \to ZHH$

```
  Expand=Yes;
  Block=No;
  AnyCT=Yes;
  Kinem="2301";
Pend;
```

This file will generate the code for the cross section calculation of the tree and loop contribution, together with the soft photon scattering. Some of the relevant 1-loop diagrams are depicted in figures 7.5 and 7.6.

For this process we meet the pentagon diagram with rank 4. Diagram 1213 as shown in 7.6 is such a diagram. The fermion propagator contains the loop momentum, as well as the three vertices on the right hand side. The scalar derived tensor reduction method in GRACE cannot handle this loop integral, but if we set the parameter `ired` to 2 in the `dat22` file, the vector derived method is used which is capable of performing the tensor reduction on this type of diagram.

7.2.1 Checks

The checks have been performed on a single phase space point in this case. The following table lists the number of agreeing digits if we vary some of the parameters.

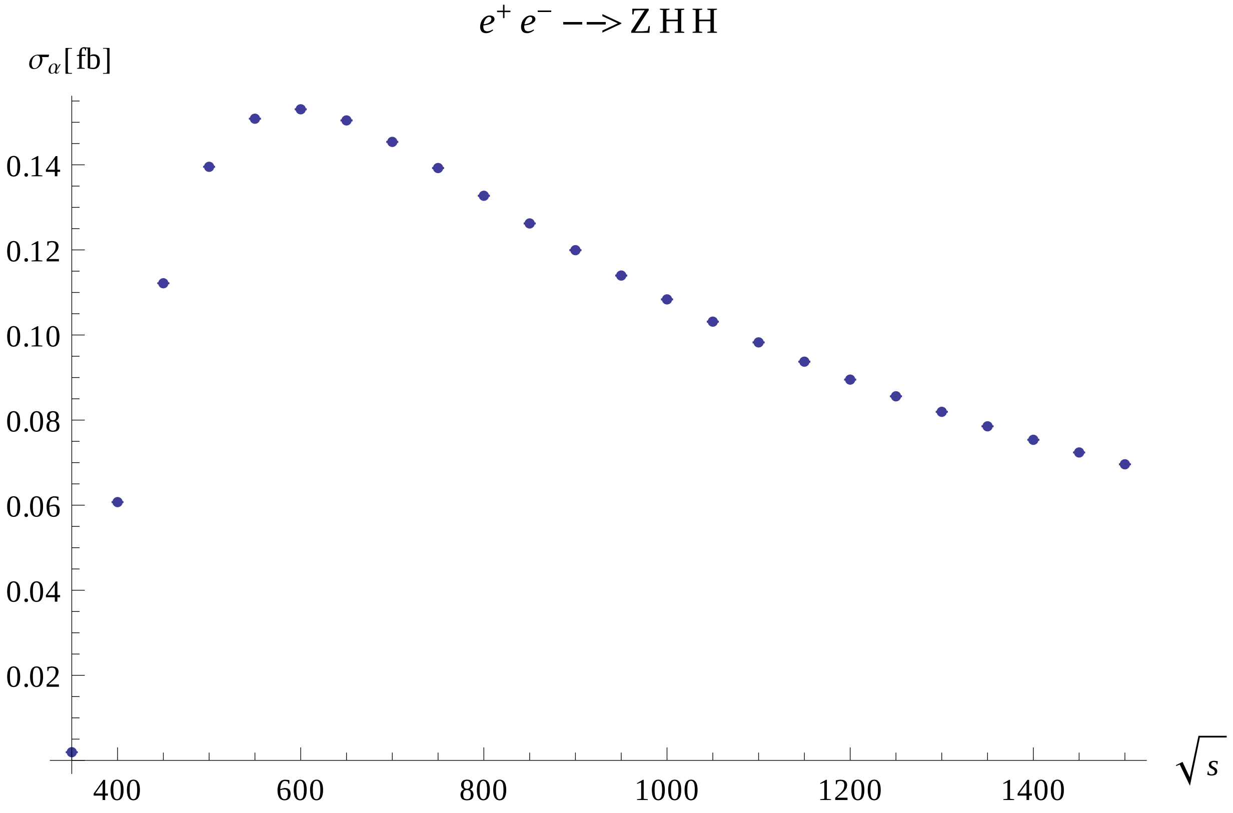


Figure 7.3: Tree level cross section of $e^+e^- \to ZHH$ using the GRACE system

	double	quadruple
Gauge parameters		
α (0 vs. 1)	6	4
β (0 vs. 1)	3	5
δ (0 vs. 1)	5	3
ϵ (0 vs. 1)	2	4
κ (0 vs. 1)	2	12
UV divergence (0 vs. 1000)	9	8
Photon mass (10^{-15} vs. 10^{-19})	9	9
Tensor reduction scheme	1	2

Tensor reduction scheme It is no surprise the different tensor reduction schemes do not agree, since one of the two (scalar derived method) is not suitable for the types of tensor integrals we encounter.

Ultraviolet finiteness The ultraviolet finiteness is checked by looking at the dependence on the parameter C_{UV}. We compared the value 0 and 1000. The infinite part of the calculation behaves well, a relative error of 10^{-8} was found.

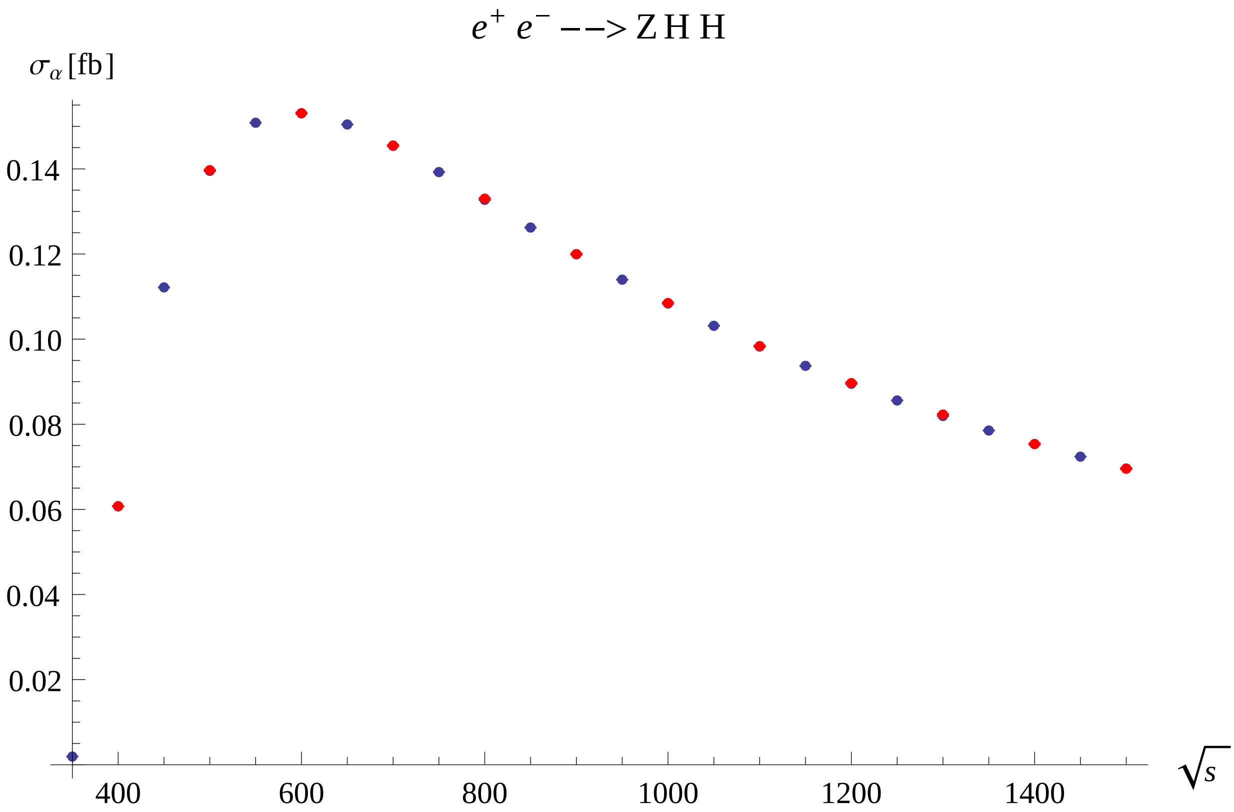


Figure 7.4: Tree level cross section of $e^+e^- \to ZHH$ using the GRACE system (blue) and MadGraph 5 (red)

Infrared finiteness The ultraviolet finiteness is checked by looking at the dependence on the parameter λ. We compared the value 10^{-15} and 10^{-19}. The infinite part of the calculation behaves well, a relative error in the order of 10^{-8} was found.

Gauge invariance The result does not seem to be gauge invariant since there is only a few digits agreement at a single phase space point. There are other phase space points where this agreement is even less. The gauge invariance does not 'improve' (except for κ) if we increase the numerical accuracy, which leads us to believe there is something fundamentally wrong in the program. In the next section a detailed analysis of this failing check is described.

7.2.2 Analysis of the gauge dependence

The matrix element of the process $e^+e^- \to ZHH$ appears to be dependent on the gauge parameters. Other calculations, including $e^+e^- \to ZZH, ZH, ZZ, HH$ and $ZZ \to HH$,$HH \to HH$, appear to be gauge invariant. This means there is a special feature of the ZHH production that does not appear in the other calculations, or that is not significant. Since the

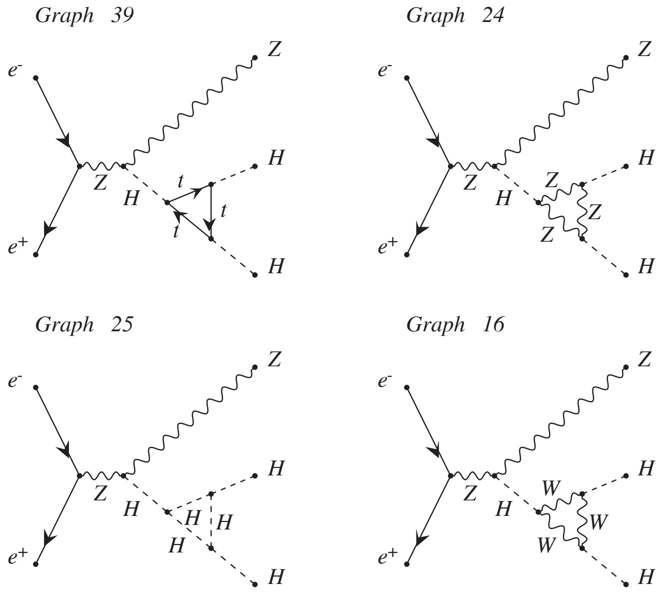


Figure 7.5: Some of the $e^+e^- \to ZHH$ diagrams that add to the correction on the tri-linear Higgs vertex.

GRACE system is quite large and complex, we have first performed superficial checks in order to narrow down the search space for this bug.

The tree level matrix element is gauge invariant and the soft photon scattering contribution scales with the tree level contribution, so the problem must lie in the treatment of the loop diagrams.

One of the first things to try is to include the electron-scalar coupling, which we left out in the previous calculation. Since we can split up the executable with the new integration system, there is no fundamental limit on the program size anymore. After a long calculation including the electron-scalar coupling, the same behaviour was found so we can conclude that this is not the source of the error. This is affirmed by the fact that similar reactions ($e^+e^- \to ZZH, ZH, ZZ$ and $ZZ \to HH$,$HH \to HH$) are gauge invariant without this coupling.

Next, we have used different libraries for the numerical evaluation of the loop integrals. The current GRACE version can use the `FF` library, or the `LoopTools` extension of `FF`. Both these libraries can handle the vector derived method in reducing the tensor integral (`ired` = 2). These versions have been compared and no significant difference has been found. An older version of GRACE uses a homebrewn loop integral library, but this one cannot handle the rank 4 5-point functions.

Although the origin of the gauge dependence does not seem to be of a numerical nature

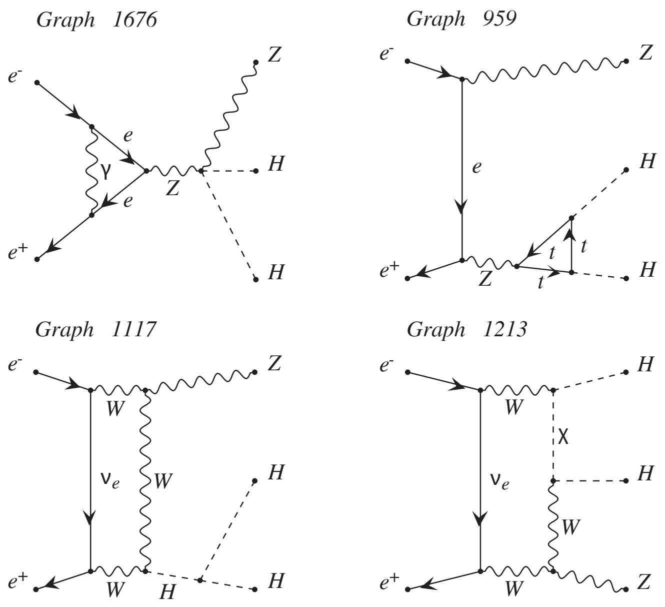


Figure 7.6: Some of the one-loop $e^+e^- \to ZHH$ diagrams that we will encounter. Diagram 1676 is the dressed e^+e^-Z vertex. Diagram 959 contains a three point tensor integral, diagram 1117 a four point tensor integral and diagram 1213 a five point tensor integral.

(since increasing the accuracy is not beneficial), we have performed some numerical checks. Disabling the code simplifications (linear shifts and output optimization) does not improve the situation. Only for extreme values of the gauge parameters the linear shifts can be problematic, since the shifts might produce squared quantities that need to cancel in the end. Numerical instabilities due to the small photon or electron mass have also been analyzed, but the calculation does not perform better for larger masses.

Since the superficial checks did not give a clue on the origin of the gauge dependence, we have dug up the old GRACE `REDUCE` codes. With this setup, the calculations of the aforementioned paper has been performed that claimed gauge invariance.

The couplings that appear in ZHH production but not in ZZH have been listed and the `FORM` and `REDUCE` code for these have been compared and found to be equal. Therefore, we continued by executing the `REDUCE` version of this reaction. The resulting `Fortran` code was an order of magnitude larger than the `FORM` outputs, but using the splitting procedure for the executables we managed to calculate the matrix element. The difference in generated code size is shown in table 7.1.

The results of the `REDUCE` code appeared to be gauge dependent as well, and mostly the same as the `FORM` outputs with the scalar derived tensor reduction scheme (`ired` $= 1$). The authors of the paper managed to achieve gauge invariance by using the primitive 1-loop

	source code lines	source code size
	`REDUCE`	
*.red	6,040,575	116 MB
b*.f	39,938,308	2.4 GB
	`FORM`	
*.frm	1,296,681	26 MB
[yz]*.f	5,567,904	141 MB

Table 7.1: The difference in generated code size for $e^+e^- \to ZHH$ using two different symbolic intermediate stages

library and patching the expressions diagram by diagram when problems were found.

Although most of the diagrams agreed with the two methods, a few were found to differ significantly. The `Fortran` code for these diagrams contained lines with more than 72 characters. Since the `Fortran77` standard enforces a line limit of 72 characters and ignores everything after that, this may result in invalid – or worse, incorrect but valid – code. Let us take a look at the offending line:

```
        10        20        30        40        50        60        70
         |         |         |         |         |         |         |
      q2 =  - 2*amel2 + amh2 - 2*e2e1 + 2*e3e2 + 2*e3e1 + 2*e4e2 + 2*e4e1;
```

This is effectively:

```
        10        20        30        40        50        60        70
         |         |         |         |         |         |         |
      q2 =  - 2*amel2 + amh2 - 2*e2e1 + 2*e3e2 + 2*e3e1 + 2*e4e2 + 2*e4e
```

`FORM` has an option to ouput `Fortran` code in the proper fixed format, respecting the character limit. This option has not been used, so normal output is generated with no line limit and a semicolon at the end of the line (not valid in `Fortran77`).

A series of catastrophic cirumstances prevented this from becoming apparent at an early stage:

- The statement occurs only in diagrams containing a renormalized self energy. Only 31 of the ZHH diagrams contain this term.
- In only four of these cases the line length exceeds the 72 character limit
- Cutting the line at 72 characters did not produce a syntax error (e.g. a + sign with no right hand side)
- Cutting the line at 72 characters gets rid of the semicolon at the end of line. Semicolons are not allowed in `Fortran77`.

- The compiler truncates the line silently, unless an explicit warning setting is provided. This violates the well known software principle of 'least astonishment'[1].

- This warning setting is not used in the GRACE system since the `Fortran` code yields a large amount of warnings, of which almost all can be ignored since the resulting code operates as desired. (This is unacceptable and must be avoided)

- The compiler does not issue an error for the appearance of the *undeclared* variable (`e4e` in this case), since the statement `IMPLICIT REAL (a-h, o-z)` is used. This declares all variables starting with the characters a ... h and o ... z as a REAL variable. (This statement is considered 'evil' and should not be used in modern code)

- The compiler does not issue a warning for the appearance of the *uninitialized* variable (`e4e` in this case), since the warning is not enabled (see above). In combination with the `IMPLICIT` statements this simply sets `e4e` to zero.

- This part of the `FORM` code was probably not checked carefully, since we also found a syntax error nearby. However, due to a bug in the `FORM` parser this incorrect statement was interpreted in the desired way.

With the fixed code we do achieve gauge invariance. All checks have been performed again and the results show a better gauge invariance at the same phase space point as before.

	double	quadruple
Gauge parameters		
α (0 vs. 1)	5	5
β (0 vs. 1)	5	6
δ (0 vs. 1)	5	4
ϵ (0 vs. 1)	7	6
κ (0 vs. 1)	11	12
UV divergence (0 vs. 1000)	8	9
Photon mass (10^{-15} vs. 10^{-19})	9	10
Tensor reduction scheme	0	0

The fact that the agreement does not increase when using quadruple precision might be worrisome. If we do include the electron scalar couplings, we see the following result in quadruple precision:

[1] When two elements of an interface conflict, or are ambiguous, the behavior should be that which will least surprise the user; in particular a programmer should try to think of the behavior that will least surprise someone who uses the program, rather than that behavior that is natural from knowing the inner workings of the program.

Gauge parameters		
α (0 vs. 1)	5	
β (0 vs. 1)	5	
δ (0 vs. 1)	4	
ϵ (0 vs. 1)	6	
κ (0 vs. 1)	28	
UV divergence (0 vs. 1000)	8	9
Photon mass (10^{-15} vs. 10^{-19})	9	10
Tensor reduction scheme	0	0

This test has been performed on 243 ($= 3^5$) points in the five dimensional phase space. We pick the points $x_i \in \{0.1, 0.5, 0.9\}$ in all possible combinations, that are subsequently mapped into kinematical variables. Apparently only the κ invariance benefits from including the coupling to the scalars.

7.2.3 Cross section

Due to time constraints, no 1-loop results (beside the checks) have been achieved. Right now we need to understand why the agreement of the gauge checks does not improve when the numerical accuracy is increased. It might be that the procedure to switch from double to quadruple precision is not complete. In particular approximation formulas need extra work when extended to quadruple precision, and this might not be implemented for every case.

If the gauge invariance is improved, the full integration procedure can be performed. In the next months, this will be studied. As can be read in the next section, some hurdles should still be overcome at full $\mathcal{O}(\alpha)$ calculations but in principle the way is paved for a complete treatment for the calculation of $e^+e^- \to ZHH$ and ZZH.

7.3 $e^+e^- \to ZZH$

If we use the nonlinear gauge with its counter terms (26 particles, 409 vertices), the process consists of 22 tree diagrams and 5209 1-loop diagrams. As done with the previous calculation, for the 'production set', we will leave out the electron-positron-Higgs and the electron-positron-χ_3 vertex. In this case (407 vertices), there are 4 tree level diagrams, and 2017 1-loop diagrams.

	#tree	#1-loop	size (optim=0)	(optim=1)	(optim=2)
All vertices	22	5,209		13 GB	
No e^+e^- scalar	4	2,017		1.1 GB	293 MB

A center of mass scan of the tree level cross section is shown in figure 7.7. We notice it is peaked at $\sqrt{s} = 500 GeV$. The results agree with calculations performed in MadGraph 5, a comparison is shown in 7.8.

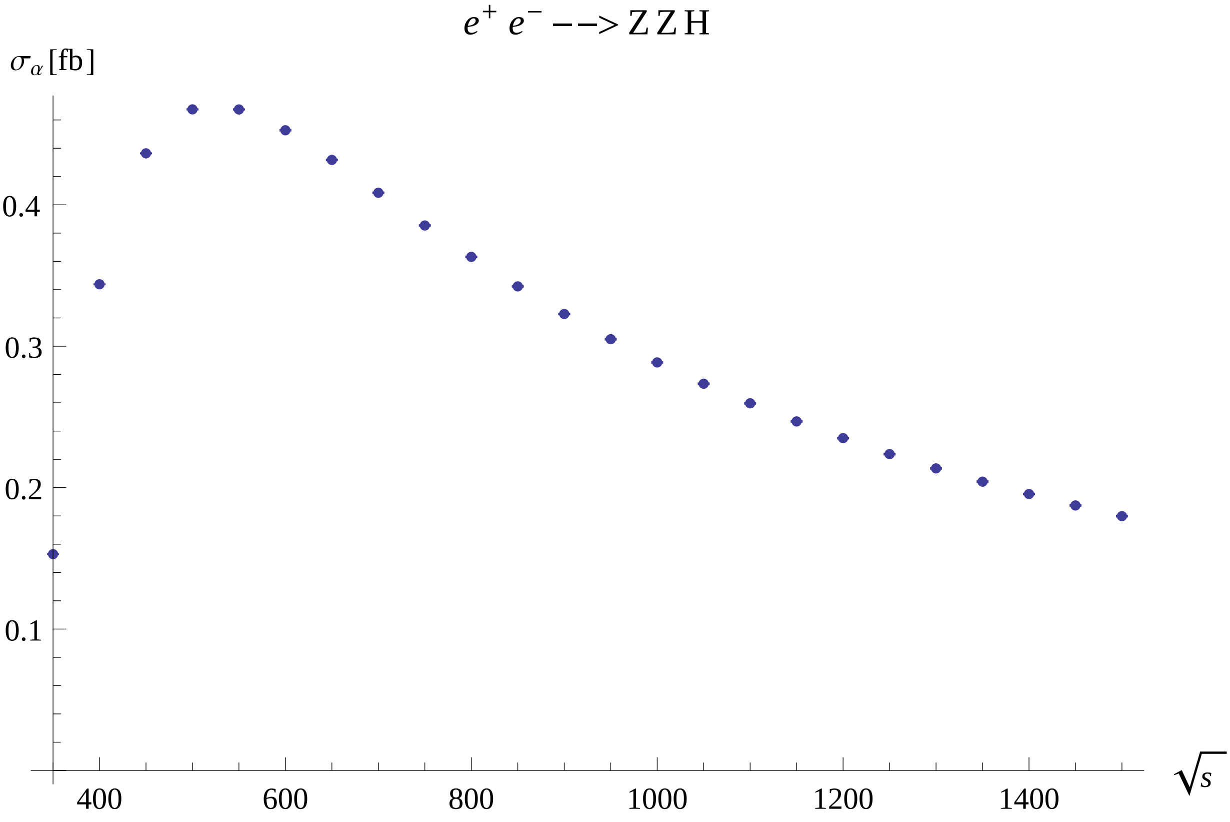


Figure 7.7: Tree level cross section of $e^+e^- \to ZZH$ using the GRACE system

In figure 7.9 and 7.10 the angular dependence of the cross section is shown for the final particles. We notice that the Z particles prefer to stay aligned with the beam pipe, whereas the Higgs particle has no angular preference.

In order to calculate the one-loop corrections for this reaction, we will use the following `in.prc` file:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Model="./nlg2301_LT.mdl";
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Process;
  ELWK={5,3};
  Initial={electron positron};
  Final   ={Z Z Higgs};
  Expand=Yes;
  Block=No;
  AnyCT=Yes;
  Kinem="2301";
Pend;
```

Since the final particles are all massive, we do not expect many numerical problems during

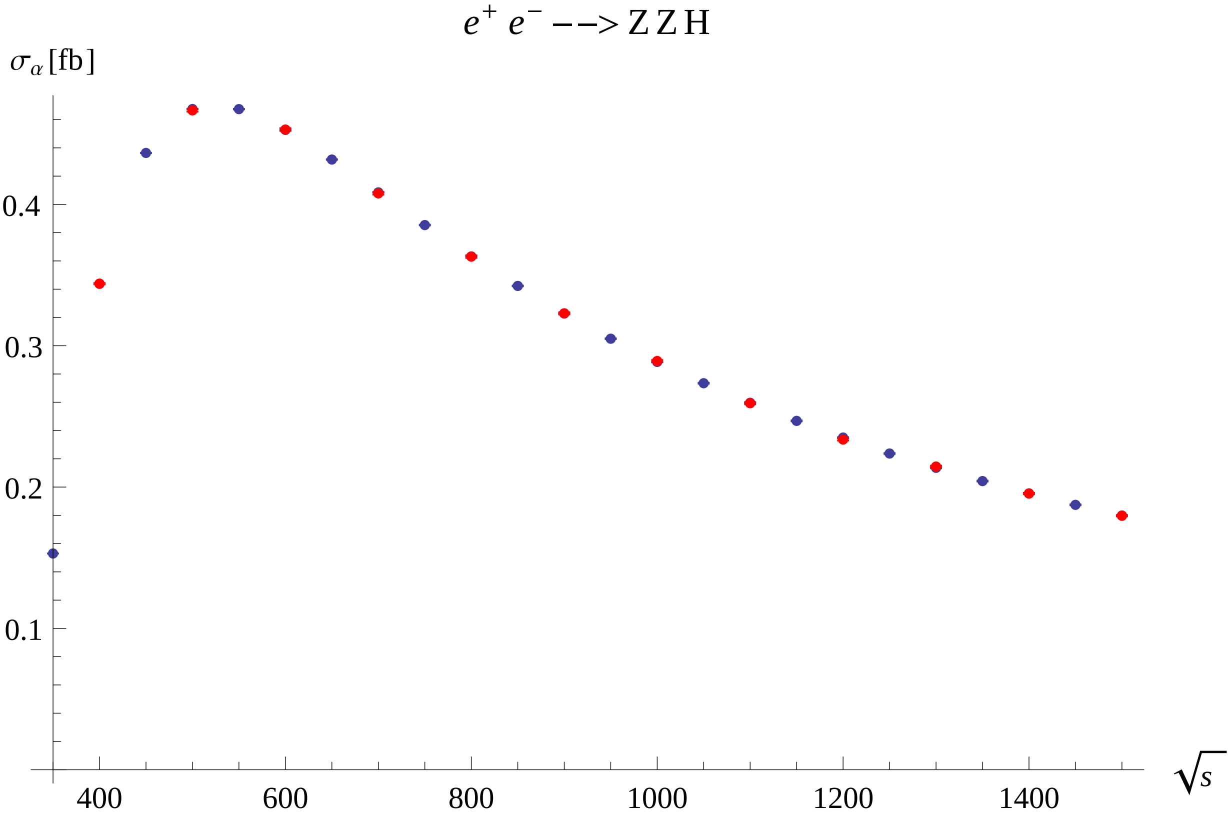


Figure 7.8: Tree level cross section of $e^+e^- \to ZZH$ using the GRACE system (blue) and MadGraph 5 (red)

the phase space integration. As it turns out, the gauge cancellations are not too severe so we can run the calculation in double precision. In total, 26 MB of `FORM` code is generated. This results in 1.1 GB of `Fortran` files when we use the simple optimization scheme, which in turn results in 6.9 GB of object files. This has been put into 48 executables that each contain a part of the total number of diagrams.

7.3.1 Checks

All checks have been performed on 243 ($= 3^5$) points in the five dimensional phase space. We pick the points $x_i \in \{0.1, 0.5, 0.9\}$ in all possible combinations, that are subsequently mapped into kinematical variables.

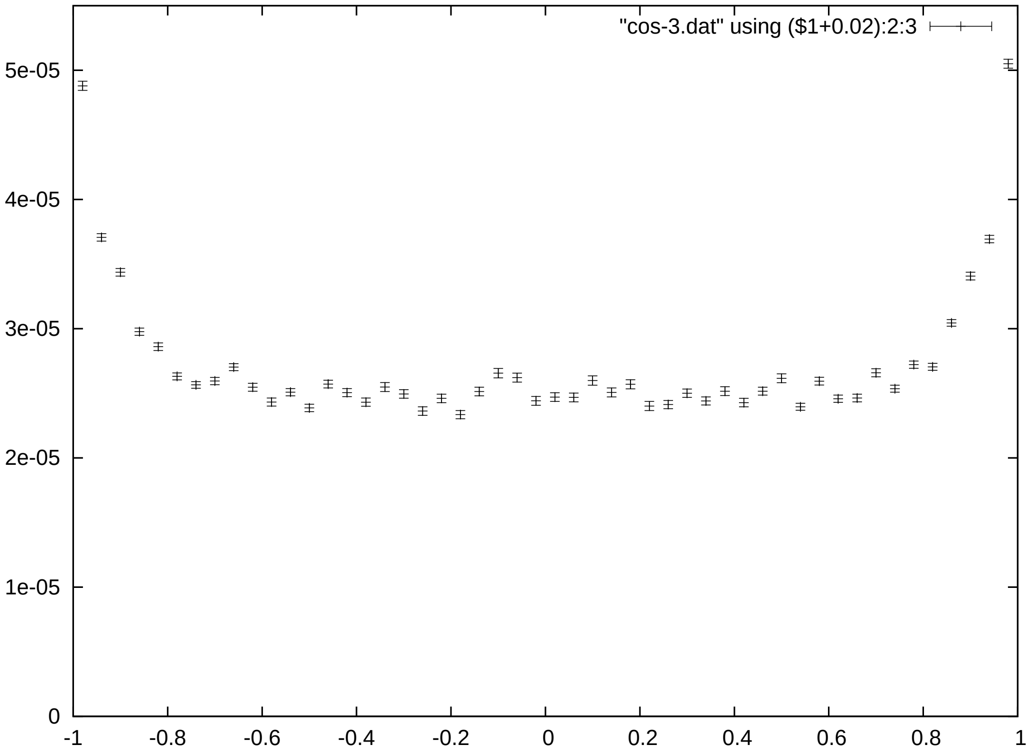


Figure 7.9: Cross section contribution binned over $\cos\theta$ of the Z particle

	double	quadruple
Gauge parameters		
α (0 vs. 1)	13	19
β (0 vs. 1)	3	10
δ (0 vs. 1)	4	11
ϵ (0 vs. 1)	4	10
κ (0 vs. 1)	10	10
UV divergence (0 vs. 1000)	8	8
Photon mass (10^{-15} vs. 10^{-19})	11	11
Tensor reduction scheme	4	10

Tensor reduction scheme Two methods for the tensor reduction have been used (`ired` equal 1 or 2). The two methods agree very well, the worst differences found are of the order of 0.03%, well within the Monte Carlo error.

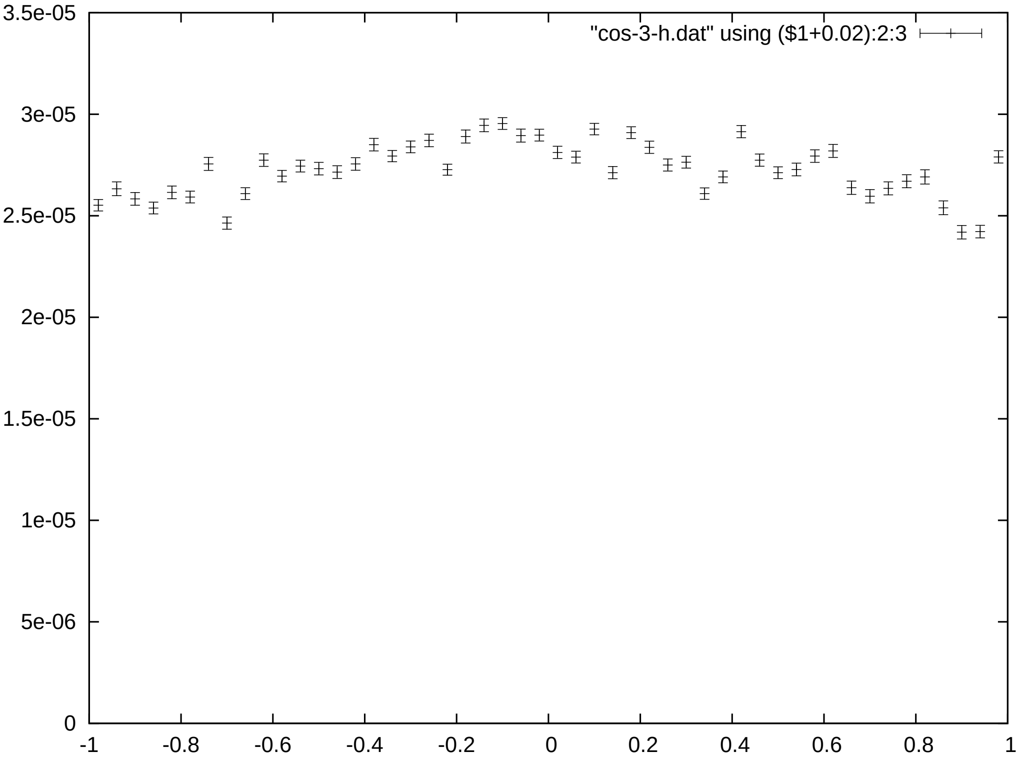


Figure 7.10: Cross section contribution binned over $\cos\theta$ of the H particle

Ultraviolet finiteness The ultraviolet finiteness is checked by looking at the dependence on the parameter C_{UV}. We compared the value 0 and 1000. The infinite part of the calculation behaves well, in the worst case, a relative error of 10^{-8} was found.

Infrared finiteness The ultraviolet finiteness is checked by looking at the dependence on the parameter λ. We compared the value 10^{-15} and 10^{-19}. The infinite part of the calculation behaves well, in the worst case, a relative error in the order of 10^{-11} was found.

Gauge invariance The results appears to gauge invariant, invariance is limited by the numerical accuracy, except for the κ dependence.

7.3.2 Cross section

The phase space integration by BASES is done by taking 5 iterations to perform the grid optimization, and 5 iterations to do the actual integration on the fixed grid. Every iteration consists of picking 8192 ($= 2^{13}$) phase space points. The output of BASES look typically like this:

```
                                            Date: 14/ 2/16  11:47
              Convergency Behavior for the Grid Optimization Step
-----------------------------------------------------------------
<- Result of  each iteration ->  <-     Cumulative Result     ->
 IT Eff  R_Neg   Estimate  Acc %  Estimate(+- Error )order  Acc %
-----------------------------------------------------------------
  1 100 100.00 -3.655E-04  1.706 -3.654756(+-0.062355)E-04  1.706
  2 100 100.00 -3.582E-04  0.911 -3.597298(+-0.028914)E-04  0.804
  3 100 100.00 -3.608E-04  0.821 -3.602358(+-0.020686)E-04  0.574
  4 100 100.00 -3.624E-04  0.845 -3.609199(+-0.017140)E-04  0.475
  5 100 100.00 -3.583E-04  0.728 -3.601348(+-0.014325)E-04  0.398
-----------------------------------------------------------------


                                            Date: 14/ 2/16  11:47
                    Convergency Behavior for the Integration Step
-----------------------------------------------------------------
<- Result of  each iteration ->  <-     Cumulative Result     ->
 IT Eff  R_Neg   Estimate  Acc %  Estimate(+- Error )order  Acc %
-----------------------------------------------------------------
  1 100 100.00 -3.657E-04  0.819 -3.657122(+-0.029952)E-04  0.819
  2 100 100.00 -3.655E-04  0.874 -3.655927(+-0.021844)E-04  0.597
  3 100 100.00 -3.609E-04  0.741 -3.637009(+-0.016919)E-04  0.465
  4 100 100.00 -3.585E-04  0.762 -3.622543(+-0.014385)E-04  0.397
  5 100 100.00 -3.651E-04  0.847 -3.627671(+-0.013042)E-04  0.360
-----------------------------------------------------------------
```

Each iteration has an efficiency of 100%, meaning no phase space points have been rejected due to cuts. The errors stays relatively constant after the first grid optimization, about 0.8% of each individual estimate. This results in a final error of about 0.36%, which is very good.

A center of mass scan of this contribution is shown in figure 7.11. These results are dependent on the 'arbitrary' soft photon energy cut k_c since the hard contribution is not included.

We need to divide the final result by 2, since the final state has two identical particles.

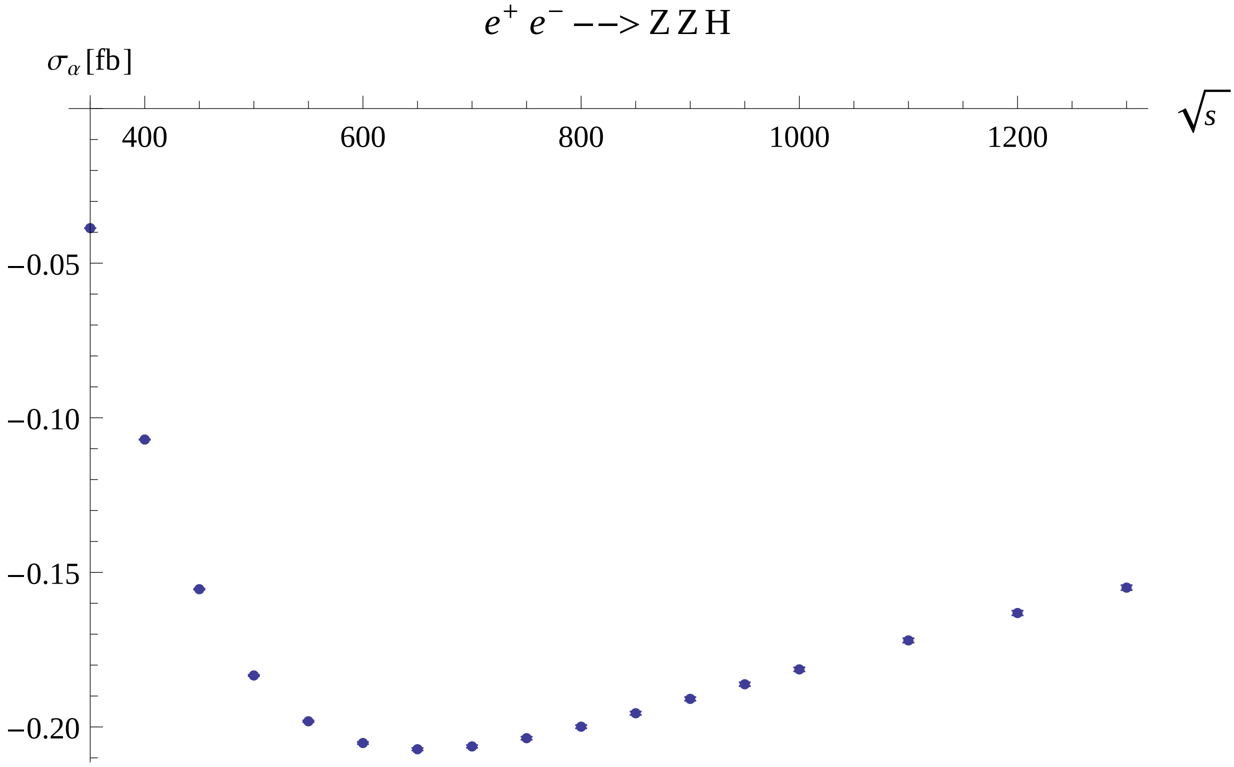


Figure 7.11: 1-loop $\mathcal{O}(\alpha)$ correction on $e^+e^- \to ZZH$ for $k_c = 0.001GeV$

7.3.3 Hard photon part

The hard photon radiation should be added to the loop and soft photon contribution in order to get the full cross section. Unfortunately, in contrast to the kinematics listing shown earlier in figure 3.2, the GRACE system cannot handle configurations with 3 outgoing massive particles and a photon. Fortunately, since only the initial state particles are charged, the kinematics are relatively simple to construct. At the moment this is being looked at in the Minami Tateya group at KEK.

Since the hard photon contribution needs to be calculated only at the tree level, we can use an alternative automated system that support four outgoing particles. We use MadGraph as the matrix element generator and MadEvent to do the phase space integration. Unfortunately already for simpler reactions such as $e^+e^- \to t\bar{t}\gamma$, there is no agreement between MadGraph and GRACE, as can be seen in table 7.2. This is run with no assumptions in the MadGraph system (massive electron, with Higgs coupling), but it appears the MadEvent system has trouble converging for small cuts. For larger photon cuts, the results simply do not agree.

Without agreement of the hard photon results from both systems on simple reactions, we do not feel confident in using the MadGraph system for the hard photon correction on the 1-loop calculations with GRACE. The full cross section is therefore not calculated, only the tree, 1-loop and soft part.

Photon cut (MeV)	GRACE (pb)	MadGraph (pb)
1	290.0 ± 0.7	14150 ± 7400
10	239.8 ± 0.6	2557 ± 2.3
100	189.3 ± 0.5	171.6 ± 2.5
1000	139.2 ± 0.3	160.1 ± 0.69
10000	88.63 ± 0.2	87.10 ± 0.19

Table 7.2: Comparison of the cross section for hard photon emission in $t\bar{t}$ production: $e^+e^- \to t\bar{t}$

Chapter 8

Conclusion

The proposed International Linear Collider will continue the search for new physics by exploring the electroweak sector of the Standard Model with high precision. In order to benefit from high precision measurements we need to go beyond the tree level for cross section calculations. This is where automated systems such as GRACE come in, since the sheer number of diagrams and the complicated one-loop tensor integral cannot be dealt with by hand in a reasonable amount of time.

We have studied the processes $e^+e^- \to ZHH$ and $e^+e^- \to ZZH$ since we are interested in corrections on the vertices related to the electroweak symmetry breaking. The GRACE system is well suited for these calculations since it can perform a full electroweak computation of up to four final particles (massive or massless). The computation consists of generating the matrix element for the tree and one-loop level, and subsequently integrating over the phase space of the final particles to get the cross section. The full cross section is obtained by including the contribution of soft photon emission, e.g. $e^+e^- \to ZHH\gamma$, to counter the IR divergences that occur in the loops.

First, the tree level cross section have been calculated and found to agree with the results from other systems.

For the one-loop results, the size of the generated code for the matrix element proved to be a problem. Therefore, we have constructed a mechanism for dealing with arbitrary code sizes. Additionaly, two types of code simplification have been built in using linear shifts in kinematical quantities and a general optimizing scheme for polynomials. This yields more than a tenfold reduction in code size.

Close inspection of the matrix element for ZHH production at the one-loop level showed that it was not gauge invariant. After a thorough search we found out an incorrect `Fortran` statement spoiled this invariance. With the fixed code it appears the result is gauge invariant again. Unfortunately there was no time for a full cross section calculation.

We were able to use the GRACE system to calculate the tree, loop and soft photon contribution for the production of ZZH. We could not calculate the full cross section since the kinematics for the hard photon emission process were not yet available. Using alternative systems proved to be problematic.

This summer (2014), the hard photon part of $2 \rightarrow 3$ reactions will be built in the GRACE system, as well as support for polarized electron beams. This provides a solid basis for many more studies on relevant ILC processes.

Bibliography

[1] J. Conway, K. Desch, J.F. Gunion, S. Mrenna, and D. Zeppenfeld. The Precision of Higgs boson measurements and their implications. *eConf*, C010630:P1WG2, 2001.

[2] High Energy Accelerator Research Organization(KEK). Grace home page. `http://minami-home.kek.jp/`.

[3] Mnchen Max-Planck-Institut fr Physik. Feynarts home page. `http://www.feynarts.de/`.

[4] Muon Accelerator Program. Website. http://map.fnal.gov.

[5] James Brau, Yasuhiro Okada, and Nicholas Walker. Ilc reference design report volume 1-executive summary. *arXiv preprint arXiv:0712.1950*, 2007.

[6] International Linear Collider. Website. http://www.linearcollider.org.

[7] Marco Battaglia. Measuring Higgs branching ratios and telling the SM from a MSSM Higgs boson at the e+ e- linear collider. pages 163–171, 1999.

[8] Yuri Alexahin, Charles M. Ankenbrandt, David B. Cline, Alexander Conway, Mary Anne Cummings, et al. Muon Collider Higgs Factory for Snowmass 2013. 2013.

[9] J.A.M. Vermaseren. New features of form. *arXiv preprint math-ph/0010025*, 2000.

[10] Michael E. Peskin and Dan V. Schroeder. *An Introduction To Quantum Field Theory (Frontiers in Physics)*. Westview Press, 1995.

[11] F Boudjema and E Chopin. Double higgs production at the linear colliders and the probing of the higgs self-coupling. *Zeitschrift für Physik C Particles and Fields*, 73(1):85–109, 1996.

[12] Jurgis Pasukonis. Implementation of non-linear gauge-fixing in feynarts package. *arXiv preprint arXiv:0710.0159*, 2007.

[13] L.H. Ryder. *Quantum Field Theory*. Cambridge University Press, 1996.

[14] P. Nogueira. Automatic feynman graph generation. *Journal of Computational Physics*, 105(2):279 – 289, 1993.

[15] Hidekazu Tanaka. Numerical calculation of helicity amplitudes for processes involving massive fermions. *Computer Physics Communications*, 58(12):153 – 168, 1990.

[16] H Murayama, I Watanabe, and K Hagiwara. Kek report 91-11, 1992.

[17] Gavin Cullen, Maciej Koch-Janusz, and Thomas Reiter. spinney: A form library for helicity spinors. *Computer Physics Communications*, 182(11):2368–2387, 2011.

[18] Geneviève Bélanger, F Boudjema, J Fujimoto, T Ishikawa, T Kaneko, K Kato, and Y Shimizu. Automatic calculations in high energy physics and grace at one-loop. *Physics reports*, 430(3):117–209, 2006.

[19] G. Passarino and M. Veltman. One-loop corrections for e+e- annihilation into μ+μ- in the weinberg model. *Nuclear Physics B*, 160(1):151 – 207, 1979.

[20] GJ Van Oldenborgh and JAM Vermaseren. New algorithms for one-loop integrals. *Zeitschrift für Physik C Particles and Fields*, 46(3):425–437, 1990.

[21] T. Hahn and M. Perez-Victoria. Automatized one loop calculations in four-dimensions and D-dimensions. *Comput.Phys.Commun.*, 118:153–165, 1999.

[22] Junpei Fujimoto, Masataka Igarashi, Nakazawa Nobuya, Shimizu Yoshimitsu, and Tobimatsu Keijiro. Radiative corrections to e+e reactions in electroweak theory. *Progress of Theoretical Physics Supplement*, 100:1–379, 1990.

[23] G. 't Hooft. Renormalization of massless yang-mills fields. *Nuclear Physics B*, 33(1):173 – 199, 1971.

[24] G. 't Hooft. Renormalizable lagrangians for massive yang-mills fields. *Nuclear Physics B*, 35(1):167 – 188, 1971.

[25] D. A. Ross and J. C. Taylor. *Nucl. Phys. B*, 51(25), 1979.

[26] Anastasiya Bierweiler, Tobias Kasprzik, Johann H. Khn, and Sandro Uccirati. Electroweak corrections to W-boson pair production at the LHC. *JHEP*, 1211:093, 2012.

[27] Setsuya Kawabata. A new version of the multi-dimensional integration and event generation package bases/spring. *Computer Physics Communications*, 88(2):309–326, 1995.

[28] T. Hahn. Cubaa library for multidimensional numerical integration. *Computer Physics Communications*, 168(2):78 – 95, 2005.

[29] Paulo Nogueira. Automatic feynman graph generation. *Journal of Computational Physics*, 105(2):279–289, 1993.

[30] T. Ueda J. Kuipers and J.A.M. Vermaseren. Code optimization in form. *to be published.*

[31] Y. Kurihara. Kinematics library in grace. *GRACE Note*, July 2002.

[32] J.A.M. Vermaseren. Two-photon processes at very high energies. *Nuclear Physics B*, 229(2):347 – 371, 1983.

[33] P.H. Khiem, Y. Kurihara, J. Fujimoto, M. Igarashi, T. Ishikawa, et al. Full $\mathcal{O}(\alpha)$ electroweak radiative corrections to $t\bar{t}\gamma$ and $e^-e^+\gamma$ productions at ILC with GRACE-Loop. 2014.

[34] P.H. Khiem, Y. Kurihara, J. Fujimoto, T. Ishikawa, T. Kaneko, et al. Full $\mathcal{O}(\alpha)$ electroweak radiative corrections to $e^+e^- \to e^+e^-\gamma$ at the ILC with GRACE-Loop. 2014.

[35] LHC Higgs Cross Section Working Group. Lhc higgs cross section wg picture gallery. https://twiki.cern.ch/twiki/bin/view/LHCPhysics/CrossSections.

[36] G. Belanger, F. Boudjema, J. Fujimoto, T. Ishikawa, T. Kaneko, et al. Full O(alpha) corrections to e+ e- → nu anti-nu h by GRACE. *Nucl.Phys.Proc.Suppl.*, 116:353–357, 2003.

[37] F Boudjema, J Fujimoto, T Ishikawa, T Kaneko, K Kato, Y Kurihara, Y Shimizu, and Y Yasui. Electroweak corrections for the study of the higgs potential at the lc. *arXiv preprint hep-ph/0510184*, 2005.

[38] G. Belanger, F. Boudjema, J. Fujimoto, T. Ishikawa, T. Kaneko, et al. Full 0(alpha) electroweak corrections to double Higgs strahlung at the linear collider. *Phys.Lett.*, B576:152–164, 2003.

[39] Zhang Ren-You, Ma Wen-Gan, Chen Hui, Sun Yan-Bin, and Hou Hong-Sheng. Full o (αew) electroweak corrections to e+ e- hhz. *arXiv preprint hep-ph/0308203*.

Acknowledgements

I am very grateful to my daily supervisor, Jos Vermaseren, for his support and guidance throughout the year. He inspired me with enthusiastic explanations and endless stories on all the fascinating stuff that happens when you try to put physics inside your computer. Our discussions more than once derailed completely from the original topic but that did not seem to bother the both of us. Further, I am very grateful for his invitation for me to continue this work for some more time in order to squeeze out a publication of all this.

Next, I would like to thank Eric Laenen for being my formal supervisor and providing great feedback on my presentation and writing of this work. Special thanks go out to Stefan Vandoren for pointing me in Jos' and Eric's direction when I asked him about computational methods in theoretical physics.

The life at the Nikhef was great, I am grateful for the facilities offered but of course it is the people that make the life enjoyable. In particular, I would like to thank my officemates Andrea, Erica, Lodewijk, Sabrina and Davide for their continuous gezelligheid and the possibility to ask stupid questions. I even learned a few Italian phrases, che tempo è!

Finally, a huge thanks goes out to my girlfriend Hannah, my friends and my family, since writing a thesis is not always rose smells and moonshine, but they managed to keep me on track :)

Appendices

Appendix A

Output optimization

A.1 Linear shifts

In the GRACE system, linear shifts are used to choose a alternative basis for quantities of the same mass dimension in order to reduce the number of operations to calculate the matrix element numerically.

The linear shifts take place in

- Kinematical quantities (dot products and masses)
- Gauge parameters
- Couplings
- Levi Civita tensors

If the number of these quantities increase, this can take a considerable amount of time since the algorithm scales quadratically (a shift happens between every two parameters of the same dimension). However, the results can be very good, usually the number of terms is reduced tenfold.

For a general diagram the results look like this:

````
~~~y12x1(0.02 sec):
Starting doshift with 914 terms
  yk1 = yk1+(-2*yk5);      number of terms = 806
  yk2 = yk2+(yk1);         number of terms = 794
  yk4 = yk4+(-yk8);        number of terms = 704
  yk5 = yk5+(-yk4);        number of terms = 698
  yk7 = yk7+(-2*yk5);      number of terms = 608
  yk1 = yk1+(-yk2);        number of terms = 560
````

```
yk2 = yk2+(2*yk5);       number of terms = 482
   .....
yk7 = yk7+(4*yk3);       number of terms = 296
yg1 = yg1+(-1);          number of terms = 198
xcp1 = xcp1+(xcp4);      number of terms = 138
xcp2 = xcp2+(xcp3);      number of terms = 108
xcp4 = xcp4+(-1/2*xcp1); number of terms = 100
```

The values `yk` are dot products, `yg` are nonlinear gauge parameters and `xcp` are the couplings. We notice a reduction from 900 to 100 terms in the output.

A.2 Horner's method

The implementation of Horner's method in `FORM` is described in [30]. The algorithm boils down to rewriting the polynomial in a form that takes less operations:

$$a(x) = \sum_{i=0}^{n} a_i x^i = a_0 + x(a_1 + x(a_2 + x(\ldots + x a_n))) \tag{A.1}$$

For example:

$$\begin{aligned} a(x,y,z) &= y - 3x + 5xz + 2x^2yz - 3x^2y^2z + 5x^2y^2z^2 \\ &= y + x(-3 + 5z + x(y(2z + y(z(-3 + 5z))))) \end{aligned}$$

The number of additions stay the same (5), but the number of multiplications decrease from 18 to 8.

One can then use techniques such as common subexpression elimination to further reduce the number of operations:

$$\begin{aligned} W &= -3 + 5z \\ a(x,y,z) &= y + x(W + x(y(2z + y(zW)))) \end{aligned}$$

Appendix B

Practical information

B.1 Installing GRACE

1. Obtain a KEK account linked to the GRACE git
2. `git clone ssh://yourname@minamiVT.kek.jp/home/git/grace.git`
3. `git pull`
4. Generate the Makefile: `./Config.sh`
5. Edit the Makefile to use the `ifort` compiler (mandatory for use of `LoopTools`), change

   ```
   FC        = .....
   FFLAGS    = ......
   ```

   to

   ```
   FC        = ifort
   FFLAGS    = -O0 -g -r8
   ```

6. `make`
7. `make install`
8. Optional: get the latest `LoopTools` library and compile it yourself. Afterwards, place it in the `lib` directory of GRACE

B.2 Run GRACE (full $\mathcal{O}(\alpha)$ correction)

1. Create a new directory and construct an `in.prc` file

2. run `grace/bin/grc`
3. run `grace/bin/grcvorm` (creates trees x*x.frm, 1-loop y*y.frm and 5-pt z*y.frm)
4. Edit the Makefile, change

   ```
   22 BASESLIB  = vbasesMPI
   23 LTLIB  = ooptoolskh-quad
   ```

   to

   ```
   22 BASESLIB  = bases
   23 LTLIB  = ooptools
   ```

   (for double precision)
5. `make test0 --jobs=...`
6. `test1` needs some repairs, so we will use `test0` for the checks
7. Edit the file `dat22`, use `ired=2` for the proper tensor reduction
8. `./test0` should give a sensible number
9. Edit `dat22`, e.g. one of the gauge parameters
10. `./test0`, must yield the same answer for both tree and loop part
11. Run all other checks. The results will still depend on the photon mass since we did not include the soft photon part.

B.2.1 Including the soft photon contribution

Make the following changes to the file `ampsum.f`:

```
--- ampsum.nosoft.f     2014-02-17 15:30:42.126780006 +0100
+++ ampsum.f    2014-01-31 17:03:10.248065209 +0100
@@ -14,10 +14,11 @@

 *-----------------------------------------------------------
       CALL ampsm0(ans0)
+      ansoft = fsoft(ans0)
 *-----------------------------------------------------------
       CALL ampsm1(ans1)
 *-----------------------------------------------------------
-      ansm = ans0 + ans1
+      ansm = ans0 + ans1 + ansoft
       RETURN
       END
 ***********************************************************
```

B.2.2 Leave out particles or vertices

In order to leave out the eeH and $ee\chi_3$ coupling, one should do the following:

1. Copy the model files (`.mdl`, `.vin`, `.fin`) from the GRACE model directory to the directory containing your `in.prc` file

2. Comment out the vertices in the model file:

```
--- ../grace0/lib/model/nlg2301_LT.mdl  2014-03-24 14:32:08.000000000 +0100
+++ ../ee-ZHH-grcvormFix/nlg2301_LT.mdl 2014-04-23 15:55:23.000000000 +0200
@@ -556,9 +556,9 @@
 %---------------------------------------
 % HFF
 %---------------------------------------
- Vertex={Higgs, positron, electron};
-     ELWK=1; FName=chel(2); CFName=dhel(2);
- Vend;
+% Vertex={Higgs, positron, electron};
+%     ELWK=1; FName=chel(2); CFName=dhel(2);
+% Vend;
  Vertex={Higgs, u-bar, u};
      ELWK=1; FName=chuq(2); CFName=dhuq(2);
  Vend;
@@ -586,9 +586,9 @@
 %---------------------------------------
 % FFX3
 %---------------------------------------
- Vertex={chi-3, positron, electron};
-     ELWK=1; FName=cyel(2); CFName=dyel(2);
- Vend;
+% Vertex={chi-3, positron, electron};
+%     ELWK=1; FName=cyel(2); CFName=dyel(2);
+% Vend;
  Vertex={chi-3, u-bar, u};
      ELWK=1; FName=cyuq(2); CFName=dyuq(2);
  Vend;
```

3. Edit the `in.prc` file, point to the modified model file, e.g.: `Model="./nlg2301_LT.mdl";`

4. Run `grc`, `grcvorm`

5. The file `inclc.h` is now incomplete. You can copy it from a directory containing a full ELWK run (it does not depend on the process)

6. `make`

B.2.3 Running in quadruple precision

You can either do this after running `grcvorm`, of after generating the `Fortran` files. In any case, make sure there are no object files (`.o`) left.

1. Edit `setup.sh`, point it to the correct paths
2. `chmod +x setup.sh`
3. `./setup.sh`, this modifies all `Fortran` and `FORM` files
4. `touch *.f`, so the `Fortran` files will no be generated again
5. Change the `Makefile`, make sure the compiler options look OK (`-DQUAD -r16`), change `libooptools` to `libooptools-quad`
6. `make`

B.2.4 Split up the executables

In order to avoid the 2 GB limit on executable size, we generate executables that contain only a part of the diagrams. The distribution of the diagrams is based on the size of the corresponding object files. This gives a reasonable estimate for the resulting executable size, and also a somewhat educated guess for the running time.

1. `makeparts N`, with N the number of executables
2. The file `MakeIntegSplit` is generated, add it to the `Makefile`: `cat MakeIntegSplit >> Makefile`
3. `make split` (Cannot be run in parallel)
4. Now N executables are generated, and the master files `test0-split` and `integ-split`

One should obviously choose the number of executables in such a way that the generated executables are smaller than 2 GB in size. In order to improve load balance, one can take the number of executables a few times the number of available threads.

The executables are called `integ-part-`i. Every executable reads the `bases.xdata` file. This file is used by the main integration executable to communicate the phase space points. The file is structured as follows:

1. The first lines is an integer telling the number N of phase space points that follow

2. N blocks of $1 + N_{\text{dim}}$ lines with floating point values follow
   (a) The first line contains the weight of the point
   (b) The other lines contain the random values to be converted to kinematical quantities

The executables take one argument that is used to designate the output file. For example running `integ-part-1 1` creates the file `bases.out.1`. The output file will contain N lines with the value for the partial matrix element at the N points in the phase space, multiplicated with the weight.

You can use `test0-split` on a simple `bases.xdata` file to check some points in the phase space. `integ-split` will perform the full phase space integration.

B.3 Gather output of `test0-split`

Using the Wolfram `MathKernel` is a bit overkill but appears to be more reliable than for example `bc`.

```
#xargs trims the output
cat bases.out.* | sed 's/E/\*10\^/' | paste -s -d+ | MathKernel -noprompt | xargs
```

B.4 Perform a phase space scan (`test0`)

For a 5 dimensional phase space:

```
for x1 in `seq 0.1 0.4 0.9`; do
        sed -i "3s/.*/  $x1/" bases.xdata;
for x2 in `seq 0.1 0.4 0.9`; do
        sed -i "4s/.*/  $x2/" bases.xdata;
for x3 in `seq 0.1 0.4 0.9`; do
        sed -i "5s/.*/  $x3/" bases.xdata;
for x4 in `seq 0.1 0.4 0.9`; do
        sed -i "6s/.*/  $x4/" bases.xdata;
for x5 in `seq 0.1 0.4 0.9`; do
        sed -i "7s/.*/  $x5/" bases.xdata;

        ./test0-split > /dev/null;
        (echo "$x1 $x2 $x3 $x4 $x5"; sum-basesout.sh) >> phase-space-grid.dat

done;
done;
done;
done;
```

```
done;

paste - - -d" " < phase-space-grid.dat > output.dat
```

Compare in Mathematica with:

```
nlg00000 = Import["output.dat", "Table"];
nlg10000 = Import["output2.dat", "Table"];

checkDiff[l1_List, l2_List] :=
 Module[
  {diffs, diffonly, maxdiff},
  diffs = {l1[[#, 1 ;; 5]],
      Abs[(l1[[#, 6]] - l2[[#, 6]])/l1[[#, 6]]]} & /@
    Range[1, Length[l1]];
  diffonly = Last[Transpose[diffs]];
  maxdiff = Max[diffonly];
  Select[diffs, #[[2]] >= 0.8 maxdiff &] // TableForm
 ]

checkDiff[nlg00000, nlg10000]
```

B.5 MadGraph comparison

MadGraph operation is very straight forward. One thing to keep in mind is that by default, MadGraph uses $\alpha^{-1} = 132.5070$. This means you will need to multiply the cross section with a factor 137/132.5 for every electroweak vertex that is present.

B.6 Compile `Fortran` code on computer with limited memory

```
find . -max-depth=1 -type f -size +1M -name ’*.f’ > bigfsource;
sed -i ’s/\.\///;s/\.f/\.o/’ bigfsource;
make ‘paste -d" " bigfsource‘;
make -j‘nproc‘;
```

B.7 Make a center of mass energy scan

```
for energy in ‘seq $1 $2 $3‘;
do
        numinteg=‘pgrep -u orottier integ | wc -l‘;
        while [ $numinteg -ge ‘nproc‘ ]; do
                echo "Sleeping..."
                sleep 5s;
                numinteg=‘pgrep -u orottier integ | wc -l‘;
        done
        mkdir s"$energy"integ;
        sed -i "57s/.*/      W = $energy.D0/" kinit.f;
        rm kinit.o;
        make integ;
        mv integ s"$energy"integ;
        cp dat22 s"$energy"integ;
        cd s"$energy"integ;
        ./integ > /dev/null &
        cd ..;
done;

#gather output
for energy in ‘seq $1 $2 $3‘;
do
        echo -n "$energy ";
        tail -n2 s"$energy"integ/bases.result | head -n1 | awk ’{print $6}’ | sed ’s
done;
```

Or with split executables:

```
for energy in ‘seq $1 $2 $3‘;
do
        mkdir s"$energy"integ;
        sed -i "57s/.*/      W = $energy.D0/" kinit.f;
        rm kinit.o
        make split;
        ./integ-split > integ.log;
        mv integ.log bases.split.* s"$energy"integ
done;
```