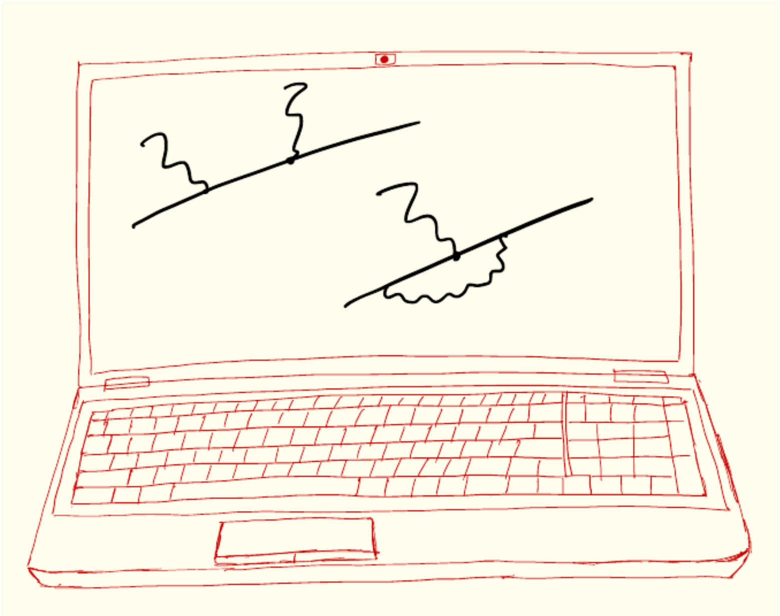
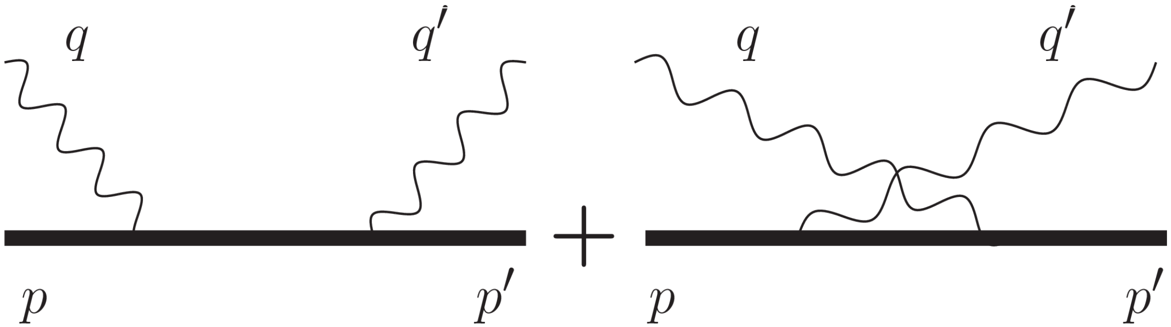
Computer Algebra for Feynman Graphs



Introduction

Why computer algebra for Feynman graphs?

- We want to calculate amplitudes/correlation functions (= Feynman graphs) in quantum field theory
- Feynman graphs are rather cumbersome objects to be straightforwardly implemented numerically. For example, the two graphs that give the leading contribution to photon-electron (Compton) scattering – one of the simplest processes,



already give quite a complicated expression for the amplitude:

$$\mathcal{M} = \epsilon_\mu \epsilon'^*_\nu \bar{u}(p') \left[\frac{\gamma^\nu (\not{p} + \not{q} + M)\gamma^\mu}{(p+q)^2 - M^2} + \frac{\gamma^\mu (\not{p} - \not{q}' + M)\gamma^\nu}{(p-q')^2 - M^2} \right] u(p)$$

Why computer algebra for Feynman graphs?

$$\mathcal{M} = \epsilon_\mu \epsilon'^*_\nu \bar{u}(p') \left[\frac{\gamma^\nu (\not{p} + \not{q} + M) \gamma^\mu}{(p+q)^2 - M^2} + \frac{\gamma^\mu (\not{p} - \not{q}' + M) \gamma^\nu}{(p-q')^2 - M^2} \right] u(p)$$

- Numerical calculation is easily done with scalars; this expression, however, contains many quantities that are not scalars: Dirac gamma matrices, spinors, 4-vectors, and so on – and this is very typical of Feynman graphs
- These quantities can be implemented numerically at an extra cost (16 elements for each of the gamma matrices, 4 elements for each 4-vector or spinor, etc…); this all can get unwieldy very quiclky when the amplitudes become more complex!
- On the other hand, there are many **algebraic** identities that the quantities above satisfy:

$$\gamma^\mu \gamma^\nu + \gamma^\nu \gamma^\mu = 2 g^{\mu\nu}$$ (Dirac algebra)

$$p + q = p' + q'$$ (four-momentum conservation)

$$(\not{p} - M) u(p) = 0$$ (the Dirac's equation)

… and **many** others!!!

Why computer algebra for Feynman graphs?

- With so many algebraic identities at hand (which is also a very typical feature of Feynman graphs, since elements that enter the amplitudes – 4-vectors, spinors, Dirac gamma matrices etc. – as a rule have very specific algebraic properties) it is an obvious, and a very good, idea to crunch the algebra first, trying to simplify the expression algebraically as much as possible, and use the simplified output in a numerical calculation
- Simple cases (for instance, the two graphs of the Compton scattering shown above) can be rather easily done by hand
- They, however, have all long been solved, so using a computer becomes almost a necessity, and this is why we need a computer algebra system!
- One has to notice that the use of computer algebra (in particular, of the tools that we are going to discuss, first of all – FORM) is not limited to the calculation of Feynman graphs. The features of the latter – the appearance of complicated non-scalar elements that cannot be easily implemented numerically but satisfy a set of algebraic identities – can be seen in many other problems in physics an mathematics

What are we going to talk about?

- We start with FORM – a specialised computer algebra system that was designed specially for particle physics; https://www.nikhef.nl/~form/
  - FORM is: fast, specialised, doing what you specifically request, uses low memory
  - FORM processes your scripts. Writing those is a bit difficult from the start (as with virtually any new programming language) but once you get used to it it becomes very natural
  - Why not try doing things with Mathematica? The answer is: with FORM, one can do really big calculations really fast, where Mathematica just cannot do it.
- We will discuss the basic features of the FORM scripting language: declarations, built-in objects, flow control (#if, #do, #switch, #define, …, #include), output (esp. in the context of using it as input for other [numerical or symbolic] software)
- We will (depending on the available time) discuss how FORM works [a nice project: write your own symbolic manipulation program!]

What are we going to talk about?

- We will apply FORM to the calculation of tree Feynman graphs: (quite) a few examples!
- To have some further context, we will discuss specific mathematical tools used to evaluate one-loop Feynman graphs:
  - Dimensional regularisation;
  - Feynman parameterisation;
  - Passarino-Veltman reduction and the calculation of a general one-loop graph.
- After that we will be able to apply FORM to the calculation of one-loop graphs
- The scalar loop functions that we will have separated with the help of FORM need to be calculated numerically (the final stage of the calculation).
- This can be done by many means (e.g., numerical integration over Feynman parameters in Mathematica [or your favourite numerical tool]

What are we going to talk about?

- We will look at LoopTools – the package that numerically implements one-loop integrals (using Fortran, C, or Mathematica), http://www.feynarts.de/looptools/
- We will calculate some one-loop graphs using LoopTools
- Time permitting, we will also discuss certain other computer algebra software that could be useful: FeynCalc, FormCalc, …
- Requests and suggestions welcome!

Excercises and other assignments

- There will be small exercises, mostly involving writing scripts in FORM and later calculating some loop functions in LoopTools (in your favourite programming language; I assume using Mathematica will be the easiest)
- For those of you who want it, there can be larger projects/problems

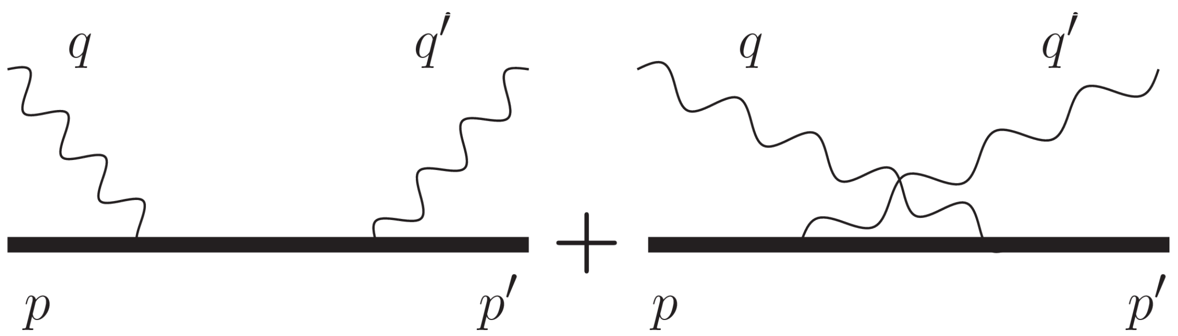
Assessment

- If you want to get credits for this course, you will have to:

1) Check it with whoever is responsible for that at the Dean's Office!
2) Register for the course
3) Learn the stuff
4) Pass the final assessment, which will either be a few problems for you to solve, using the tools that we will be mastering during the semester, or a small project – we can discuss the details

Example: a small (but real) FORM code

- Consider the reaction we had as example in the beginning: electron Compton scattering (it can actually be a nucleon, in which case one has to add the a.m.m. coupling):



$$\mathcal{M} = \epsilon_\mu \epsilon_\nu'^{*} \bar{u}(p') \left[\frac{\gamma^\nu (\not{p} + \not{q} + M)\gamma^\mu}{(p+q)^2 - M^2} + \frac{\gamma^\mu (\not{p} - \not{q}' + M)\gamma^\nu}{(p-q')^2 - M^2} \right] u(p)$$

- I will now show you a FORM script that simplifies this amplitude and decomposes it into basis tensors

Some definitions

- Tensor decomposition of the Compton scattering amplitude [follows from photon crossing symmetry, P and T invariance]

$$T_{fi} = \mathcal{E}'^{*}_{\mu}(q')\,\mathcal{E}_{\nu}(q)\,\sum_{i=1}^{8} \mathcal{A}_i(s,t)\,\bar{u}_{s'}(p')\,O_i^{\mu\nu}\,u_s(p)$$

$$\begin{aligned}
O_1^{\mu\nu} &= -g^{\mu\nu} \\
O_2^{\mu\nu} &= q^{\mu}q'^{\nu} \\
O_3^{\mu\nu} &= -\gamma^{\mu\nu} \\
O_4^{\mu\nu} &= g^{\mu\nu}\,(q'\cdot\gamma\cdot q) \\
O_5^{\mu\nu} &= q^{\mu}q'_{\alpha}\gamma^{\alpha\nu} + \gamma^{\mu\alpha}q_{\alpha}q'^{\nu} \\
O_6^{\mu\nu} &= q^{\mu}q_{\alpha}\gamma^{\alpha\nu} + \gamma^{\mu\alpha}q'_{\alpha}q'^{\nu} \\
O_7^{\mu\nu} &= q^{\mu}q'^{\nu}\,(q'\cdot\gamma\cdot q) \\
O_8^{\mu\nu} &= \gamma^{\mu\nu\alpha\beta}q_{\alpha}q'_{\beta} = -i\gamma_5\epsilon^{\mu\nu\alpha\beta}q'_{\alpha}q_{\beta}
\end{aligned}$$

$$\mathcal{E}_{\mu}(q) = \varepsilon_{\mu} - \frac{P\cdot\varepsilon}{P\cdot q}\,q_{\mu}$$