Two loop calculations

June 2024

We start in section 3.2 of [1]. In this we have $p^2 = m_b^2$ and $(p-q)^2 = m_s^2 \to 0$.

We have 15 diagrams organised into 5 families as shown in fig 1

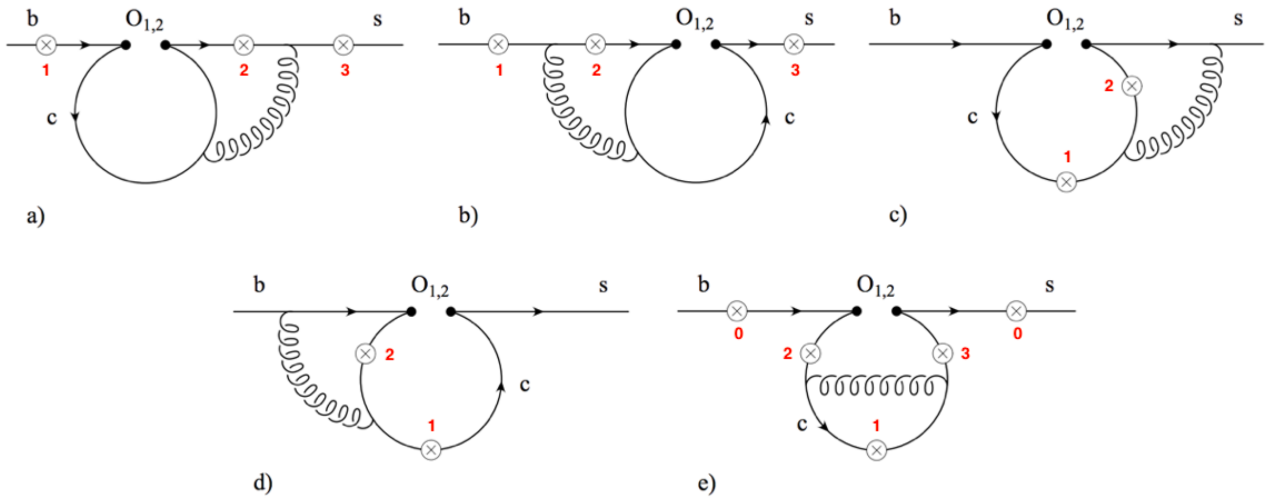
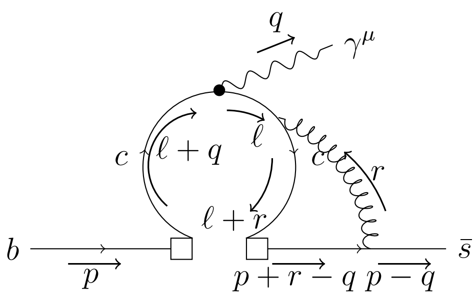


Figure 1: Our 2 loop diagrams

There are 2 loop momenta, l and r, with the gluon loop being labelled as r. I have drawn momentum routings for all diagrams on paper but I will not include them all in this document. To outline the full method I will use diagram c1 as an example.



$$. \tag{1}$$

$$\sim \int d^d\ell d^d r \frac{N^\mu}{(\ell+q)^2 - m_c^2)(\ell^2 - m_c^2)((\ell+r)^2 - m_c^2)r^2(r+p-q)^2} \tag{2}$$

$$= \int d^d\ell d^d r \frac{N^\mu}{P_1 P_2 P_3 P_4 P_5} \tag{3}$$

Where we have,

$$
\begin{aligned}
N^\mu &= P\overline{u}(p-q,0)\gamma_\nu(\not{p}+\not{r}-\not{q})\gamma_\alpha((\not{\ell}+\not{r})+m_c)\gamma^\nu(\not{\ell}+m_c)\gamma^\mu((\not{\ell}+\not{q})+m_c)\gamma_\alpha u(p,mb). \\
P &= eg_s^2 Q_c\frac{-(N_c^2-1)\delta_{ij}}{4N_c^2}
\end{aligned}
\tag{4}
$$

I used the package ColorMath [2] to check the colour algebra. From here, we use Package-X [3] to perform the Dirac algebra. We then implement two functions. The first, converts all scalar products into linear combinations of our 7 propagators. The second, performs Passarino-Veltman reductions and applies equations of motion. At this stage, we can see the tensor structure of our results. The sum of all our integrals from one diagram family (i) will be of the form

$$\overline{u}(p-q,0)P_{\mathbb{R}}V^\mu_{(i)}(q^2)u(p,m_b) \text{ where,} \tag{5}$$

$$V^\mu_{(i)}(q^2) = A_{(i)}q^\mu + B_{(i)}p^\mu + C_{(i)}\gamma^\mu. \tag{6}$$

The scalars $A_{(i)}, B_{(i)}$ and $C_{(i)}$ depend only on m_b, m_c and q^2. We need each diagram family to satisfy the ward identity, i.e.

After this we convert our integrals to "j-integrals". These have the general form

$$j[i;n_{i_1},n_{i_2},n_{i_3},n_{i_4},n_{i_5},n_{i_6},n_{i_7}] = (2\pi)^{-2d}\int\frac{(\tilde{\mu}^2)^{2\epsilon}\ d^d\ell\ d^dr}{P_{i_1}^{n_{i_1}}P_{i_2}^{n_{i_2}}P_{i_3}^{n_{i_3}}P_{i_4}^{n_{i_4}}P_{i_5}^{n_{i_5}}P_{i_6}^{n_{i_6}}P_{i_7}^{n_{i_7}}} \tag{7}$$

where the numbers n_i are integers (positive or negative). Note that this differs from [1] as we have not defined our integrals to be scaleless. In general, each integral will come with a prefactor $\mathcal{O}\big(m^{N_i-4}\big)$, where $N_i = \sum_{j=1}^7 n_{i_j}$.The objects P_i are propagators (see below), and the indices $\{i_1,\ldots,i_7\}$ depend on the class. In addition, $d = 4-2\epsilon$, and $\tilde{\mu}^2 \equiv \mu^2 e^{\gamma_E}/4\pi$, with μ the $\overline{\text{MS}}$ scale.

Our choice of momentum routings fixes the first five propagators in each class, and the other two are chosen to be linear in loop momenta and such that the seven propagators form a linearly-independent set. The complete set of propagators needed is:

$$
\begin{array}{lll}
P_1 = (\ell+q)^2 - m_c^2 & P_5 = (r+p-q)^2 & P_9 = (\ell+q)^2 \\
P_2 = \ell^2 - m_c^2 & P_6 = (r+q)^2 & P_{10} = (r+p-q)^2 - m_b^2 \\
P_3 = (\ell+r)^2 - m_c^2 & P_7 = (\ell+p-q)^2 & P_{11} = (r+p)^2 - m_b^2 \\
P_4 = r^2 & P_8 = (r+p)^2 & P_{12} = (\ell+r+q)^2 - m_c^2 \\
P_{13} = (r+p-q)^2 & &
\end{array}
\tag{8}
$$

Note that propagators P_6, P_7, P_9 and P_{13} differ from [1]. This is because the software we are using to perform our master integral reductions cannot handle scalar products. and the scalar integrals for each class are:

$$
\begin{array}{ll}
j[a;n_2,n_3,n_4,n_5,n_8,n_7,n_9]\ , & j[d;n_1,n_2,n_{12},n_4,n_{11},n_6,n_7]\ , \\
j[b;n_2,n_3,n_4,n_{10},n_{11},n_7,n_9]\ , & j[e;n_1,n_2,n_3,n_4,n_{12},n_7,n_{13}]\ , \\
j[c;n_1,n_2,n_3,n_4,n_5,n_6,n_7]\ . &
\end{array}
\tag{9}
$$

Having rewritten our integrals to this j-integral form we can then perform integration by parts reduction to reduce to the minimum set of master integrals. We use the program Kira [4] to do this.

1 Files in Repository

Within the git repository for this project there are a few different files. We briefly outline below what files are here and what is done in each of them.

The main notebook Automated-2-loop-sbcc-processes.nb is where we perform the calculations that take us from initial amplitude to minimum set of master integrals.

1.1 Packages

In this directory we have a copy of the versions of Package-X and ColorMath used for this project, as well as a file containing the details of the Mathematica version used.

1.2 Functions

There are a few files in this directory.

- TensorIntegrals.m: This is where the replacement rules that take scalar products to linear combinations of propagators are stored
- ScalarIntegrals.m: This is where the functions that reduce our tensor integrals to scalar integrals are stored. They are a combination of Passarino-Veltman rules, Dirac algebra and application of equations of motion
- Determine_Functions.nb: This is where the calculations needed for the functions in ScalarIntegrals and Tensor Integrals are performed. It is also where various tests of these functions were done
- jIntegrals.m: This is where the functions that convert from scalar integrals to the j-integrals are stored.
- MasterIntegralReplacements.m: This is where the integration by parts reduction replacement rules that were generated by kira are stored.

For more details on any of these functions please see the files they are housed in.

1.3 Results

- Amplitudes.m: stores the initial amplitudes created using Package-X
- jIntegrals.m: stores the jIntegral form of the amplitudes
- Final.m: stores the master integral form of the amplitudes (including information on the ward identity)

References

[1] Hrachia M. Asatrian, Christoph Greub, and Javier Virto. Exact NLO matching and analyticity in $b \to s\ell\ell$. *JHEP*, 04(2020)012, 2020. [arXiv: 1912.09099].

[2] M. Sjodahl. ColorMath - A package for color summed calculations in SU(Nc). *Eur. Phys. J. C*, 73(2):2310, 2013. [arXiv:1211.2099].

[3] Hiren H. Patel. Package-X: A mathematica package for the analytic calculation of one-loop integrals. *Comput. Phys. Commun.*, 197:276–290, 2015. [arXiv: 1503.01469].

[4] Jonas Klappert, Fabian Lange, Philipp Maierhöfer, and Johann Usovitsch. Integral reduction with Kira 2.0 and finite field methods. *Comput. Phys. Commun.*, 266:108024, 2021. [arXiv: 2008.06494].