Hadronic Light by Light Calculation

Gregory Turnberg

October 2021

1 Introduction

The purpose of this document is to explain some results relating to the pion pole contribution and workings of the code used in the hadronic light by light calculations. The requirements to run the code are:

- C++
- CERN Root
- GSL
- Make

Disclaimer - I am by no means an expert on C++, so the methods used in the following code are likely not the most ideal implementation.

2 Background

Our ultimate goal is to calculate the pseudoscalar pion pole contribution $a_\mu^{\mathrm{HLbL}:\pi^0}$, which can be found through the following equation:

$$a_\mu^{\mathrm{HLbL}:\pi^0} = \left(\frac{\alpha}{\pi}\right)\left[a_\mu^{\mathrm{HLbL}:\pi^0(1)} + a_\mu^{\mathrm{HLbL}:\pi^0(2)}\right]$$

where α is the fine structure constant. The two terms on the right both have triple integral representations:

$$a_\mu^{\mathrm{HLbL}:\pi^0(1)} = \int_0^\infty dQ_1 \int_0^\infty dQ_2 \int_{-1}^1 d\tau w_1(Q_1,Q_2,\tau)\mathcal{F}_{\pi^0\gamma^*\gamma^*}(-Q_1^2,-(Q_1+Q_2)^2)\mathcal{F}_{\pi^0\gamma^*\gamma^*}(-Q_2^2,0) \quad (1)$$

$$a_\mu^{\mathrm{HLbL}:\pi^0(2)} = \int_0^\infty dQ_1 \int_0^\infty dQ_2 \int_{-1}^1 d\tau w_2(Q_1,Q_2,\tau)\mathcal{F}_{\pi^0\gamma^*\gamma^*}(-Q_1^2,-Q_2^2)\mathcal{F}_{\pi^0\gamma^*\gamma^*}(-(Q_1+Q_2)^2,0) \quad (2)$$

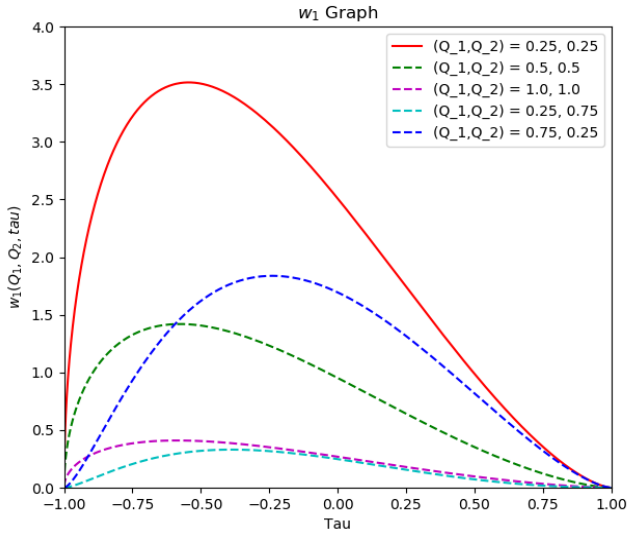
where w_1 and w_2 are weighting functions:

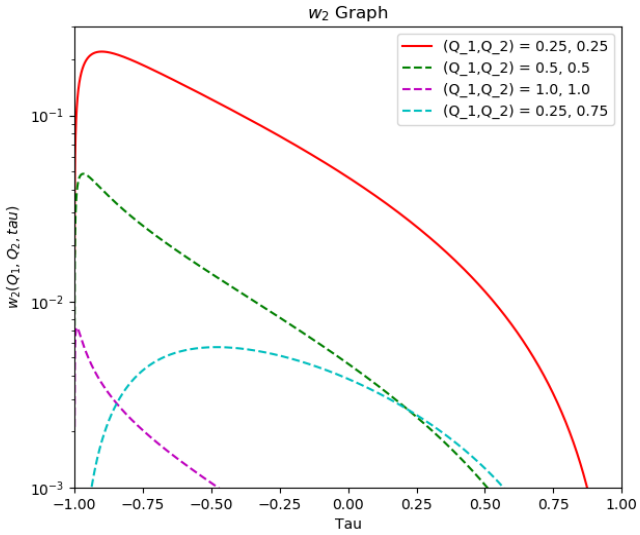
$$w_1(Q_1,Q_2,\tau) = \left(\frac{-2\pi}{3}\right)\sqrt{1-\tau^2}\frac{Q_1^3Q_2^3}{Q_2^2+m_\pi^2}I_1(Q_1,Q_2,\tau)$$

$$w_2(Q_1,Q_2,\tau) = \left(\frac{-2\pi}{3}\right)\sqrt{1-\tau^2}\frac{Q_1^3Q_2^3}{(Q_1+Q_2)^2+m_\pi^2}I_2(Q_1,Q_2,\tau)$$

The definitions of I_1 and I_2 are quite complex, so they are omitted for now. Their exact definitions can be found in the equation appendix at the end of this document, as well as a table of all relevant constants.

The following figure contains plots of the weighting functions with various fixed values of Q_1 and Q_2 while varying τ (note that the y-axis on the right plot is logarithmic):





The integrals also involve the on-shell transition form factor for the pion. In particular, we need the lowest meson dominance plus vector parameterization, or LMD+V form factor:

$$\mathcal{F}^{\text{LMD+V}}_{\pi^0\gamma^*\gamma^*}(q_1^2, q_2^2) = \frac{F_\pi}{3} \frac{q_1^2 q_2^2 (q_1^2 + q_2^2) + h_2 q_1^2 q_2^2 + h_5 q_1^2 q_2^2 + h_5 (q_1^2 + q_2^2) + h_7}{(q_1^2 - M_{V_1}^2)(q_1^2 - M_{V_2}^2)(q_2^2 - M_{V_1}^2)(q_2^2 - M_{V_2}^2)} \tag{3}$$

Descriptions of all relevant constants can be found at the end of the document.

3 Pion Pole Contribution Calculations

In order to calculate the pion pole contribution, we must first compute two triple integrals. Of course, it would be impossible to do this by hand given the complexity of the integrands, so we resort to numerical methods. The standard Riemann sum or trapezoid rule algorithms are not be the best course of action however, since as the number of dimensions d in an integral increases they run in $\mathcal{O}(n^d)$. A better algorithm would be Monte Carlo integration, which runs in $\mathcal{O}(n)$ regardless of the number of dimensions, making it well-suited for high-dimensional integrals. Monte Carlo integration works by evaluating the integrand at random points in the domain of integration in order to compute the average value of the function over the domain. This number is then multiplied by the "volume" of the domain of integration to produce the final result. A naive algorithm uses uniform sampling over the whole domain, while more sophisticated algorithms such as MISER and VEGAS use stratified and importance sampling to place samples in areas which decrease the overall variance of the result.

Although the upper bounds of integration on Q_1 and Q_2 are both ∞, we do not need to integrate out this far in practice to get an accurate result. Both of the weighting functions approach 0 as $Q_1, Q_2 \to \infty$, so a much smaller upper bound of 20 can be used.

For implementation, GSL provides many optimized Monte Carlo integration algorithms in C++. Using the VEGAS algorithm with 40 million samples and a momentum cutoff of 20, we obtain the result

$$a_{\mu:\text{LMD+V}}^{\text{HLbL:}\pi^0} = 62.9201422692142 \times 10^{-11}$$

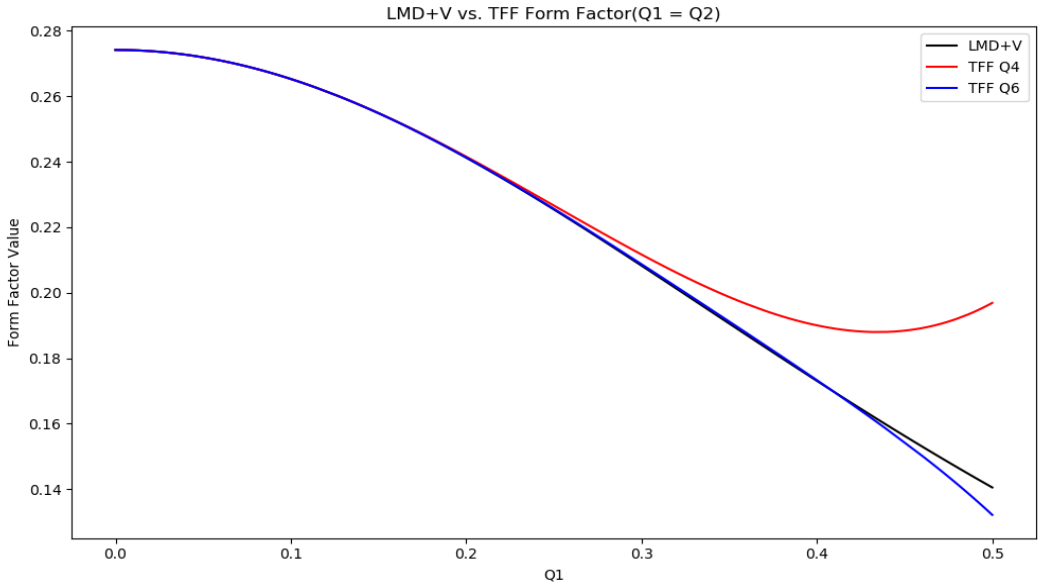
which agrees with the value calculated by Nyffeler of 62.9×10^{-11}.

4 Low Momentum Expansion

Another topic of interest is the low momentum form factor expansion, which approximates the LMD+V form factor for sufficiently small Q_1 and Q_2. The Q^6 expansion is:

$$\begin{aligned}\mathcal{F}_{Q^6}(-Q_1^2,-Q_2^2) = \frac{1}{4\pi^2 F_\pi}\big[1 - a(Q_1^2+Q_2^2) + b(Q_1^4+Q_2^4) + cQ_1^2Q_2^2 \\ +d(Q_1^6+Q_2^6) + e(Q_1^4Q_2^2+Q_1^2Q_2^4) + \cdots\big]\end{aligned} \tag{4}$$

This expansion is valid in the region $Q_1^2 < 0.1,\ Q_2^2 < 0.1$. Below is a graph of the LMD+V form factor along with the Q^4 and Q^6 expansions.



The two expansions are quite accurate in low momentum regions. We can obtain a measure of how accurate they are by performing the integrals in equations (1) and (2) and calculating $a_\mu^{\mathrm{HLbL:}\pi^0}$ using a small momentum cutoff of $Q_{1,2} < 0.1$. The following result used 40 million samples per integral:

```
Integration to Q < 0.1 with 40,000,000 samples
Integral 1 (LMD+V):     0.001005460894  Sigma: 1.864345197e-08
Integral 2 (LMD+V):     0.0001717746828 Sigma: 4.577028494e-09

Integral 1 (Q4)    :     0.001005481982  Sigma: 1.832570684e-08
Integral 2 (Q4)    :     0.000171768857  Sigma: 4.722432529e-09

Integral 1 (Q6)    :     0.001005465541  Sigma: 1.846705857e-08
Integral 2 (Q6)    :     0.0001717705784 Sigma: 4.358205314e-09

Final LMD+V        :     1.475399897e-11
Final Q4           :     1.475419025e-11
Final Q6           :     1.475400577e-11


% Error Q4 = 0.001296464429
% Error Q6 = 4.605865611e-05
```

The percent error is extremely small, so we can be confident that these expansions accurately model the LMD+V form factor. Even when we integrate out to the $Q < 0.55$ region, the Q^6 expasion is still reasonably accurate:

```
Integration to Q < 0.55 with 40,000,000 samples
Integral 1 (LMD+V):     0.03059591231   Sigma: 1.143014661e-06
Integral 2 (LMD+V):     0.001228622807  Sigma: 7.092076856e-08

Integral 1 (Q4)   :     0.03445752339   Sigma: 1.29445538e-06
Integral 2 (Q4)   :     0.001255706317  Sigma: 7.560896752e-08

Integral 1 (Q6)   :     0.0292823835    Sigma: 1.31088181e-06
Integral 2 (Q6)   :     0.001222610122  Sigma: 7.34236183e-08

Final LMD+V       :     3.98848937e-10
Final Q4          :     4.475849735e-10
Final Q6          :     3.823114535e-10


% Error Q4 = 12.21917171
% Error Q6 = 4.146302504
```

Additionally, we are interested in the parameters a, b, c, d, e, and the constant $\Gamma_{\pi^0 \to \gamma\gamma'}$. We can find the uncertainty in these values by calculating the partial derivatives of the pseudoscalar pion pole contribution $a_\mu^{\text{HLbL}:\pi^0}$ with respect to each parameter. We can do this by by using the standard two-sided finite difference algorithm for derivatives:

Let $f : \mathbb{R}^6 \to \mathbb{R}$ be a function that takes the parameters $a, b, c, d, e, \Gamma_{\pi^0 \to \gamma\gamma'}$ as input and outputs the value of the pseudoscalar pion pole contribution using the Q^6 form factor expansion. If we wanted to find the uncertainty in a, for example, we would need to calculate $\dfrac{\partial f}{\partial a}$, which using the two-sided finite differnce is:

$$\frac{\partial f}{\partial a} \approx \frac{f\big(a(1+p), b, c, d, e, \Gamma\big) - f\big(a(1-p), b, c, d, e, \Gamma\big)}{2ap}$$

where $0 < p << 1$ is some small percent offset. In this case, we would choose all parameter values to be their mean value, as in the table of constants. Ideally we want p to be as small as possible, but due to the limitations of floating point arithmetic if p is too small we introduce floating point errors into the calculation. On the other hand, if p is too large the approximation of the partial derivative becomes less valid. In an attempt to mitigate these errors, we will calculate the partials for each parameter for a range of percent offsets and compare them to see if they agree. We find:

```
Integration up to Q < 0.32 with 10,000,000 samples

Partials - parameters varied by 0.25%
a     :       -3.3649414139212751e-11
b     :       6.5597577850400077e-13
c     :       7.5336493834179759e-13
d     :       -2.2389899845111552e-14
e     :       -2.5921871988282577e-13
gamma :       0.028048597506709651

Partials - parameters varied by 0.5%
a     :       -3.4449383778532745e-11
b     :       3.4142804377747982e-12
c     :       8.8450944603176724e-13
d     :       2.8415787365649684e-15
e     :       -9.0179242866092002e-14
gamma :       0.028048597506709985
```

```
Partials - parameters varied by 0.75%
a     :       -3.3191362216371211e-11
b     :       2.5755926746937237e-12
c     :       8.9698494346664814e-13
d     :       5.5273532206700047e-13
e     :       1.1963893088477835e-13
gamma :       0.02804859750671121 3

Partials - parameters varied by 1%
a     :       -3.2643828344676348e-11
b     :       2.7937088821432139e-12
c     :       7.6711187905363668e-13
d     :       2.4753257230670531e-13
e     :       3.3237715190804097e-14
gamma :       0.02804859750671098 7

Partials - parameters varied by 2%
a     :       -3.3252659401737983e-11
b     :       2.5268070693977975e-12
c     :       8.0415729114617521e-13
d     :       6.3470922586978126e-14
e     :       3.0352326917181194e-14
gamma :       0.02804859750671098 7
```

We see that the parameters a, b, and Γ have good agreement, while c, d, and e have less agreement. Increasing the integration bound to $Q < 0.55$ results in better agreement among all parameters:

```
Integration up to Q < 0.55 with 10,000,000 samples

Partials - parameters varied by 0.5%
a       :       -1.64032569337445e-10
b       :       3.38827807205793e-11
c       :       1.14428571606979e-11
d       :       9.08712120200189e-12
e       :       4.35623158169835e-12
gamma   :       0.0494522072073776

Partials - parameters varied by 1%
a       :       -1.64967438673216e-10
b       :       3.24854023731518e-11
c       :       1.1522291345106e-11
d       :       9.46112561685376e-12
e       :       3.93324334449662e-12
gamma   :       0.0494522072073796

Partials - parameters varied by 2%
a       :       -1.63683649020039e-10
b       :       3.33407981304216e-11
c       :       1.26548343516877e-11
d       :       9.09222852041125e-12
e       :       4.04912057232516e-12
gamma   :       0.0494522072073795
```

5 Code Documentation

The subsections here detail what each file does as well as how to compile and run them. The files themselves are also documented with comments in the code. There are 5 files in total:

- functions.h
- main.cpp
- error.cpp
- propagate.cpp
- Makefile

5.1 functions.h

This file defines the functions and physical constants needed in the HLbL calculation. Such functions include the form factors and weighting functions.

5.2 main.cpp

This program calculates the value $a_\mu^{\mathrm{HLbL:}\pi^0}$. It does this using the VEGAS Monte Carlo integration algorithm (implemented by GSL) to calculate the relevant integrals.

5.3 error.cpp

This program calculates the percentage error of the Q^4 and Q^6 form factor expansions. The upper integration bounds on Q_1 and Q_2 can be changed by altering the value of the `limit` variable, and the number of samples used in the integration algorithm can be changed with the `calls` variable. Since a total of 6 integrals need to be calculated, the MISER algorithm (implemented in GSL) is used because it is faster than the VEGAS algorithm.

5.4 propagate.cpp

This program computes the partial derivatives of $a_\mu^{\mathrm{HLbL:}\pi^0}$ with respect to the parameters a, b, c, d, e, and Γ using the Q^6 expansion. The MISER algorithm is used here since many integrals need to be calculated. The number of samples can be changed by changing the `samples` variable, and the integration bound can be changed with the `cutoff` variable.

6 Compiling and Running

To compile all of the code, simply run the command `make` in the same directory as the Makefile using the command line. This should generate several files. The important ones are `main`, `error`, and `propagate` (note that these files don't have extensions since they are executables). To run the relevant program, type `./<fileName>` into the command line. For example, to run `main.cpp`, type `./main` into the command line. To remove all of the generated files, run `make clean`. This will not affect any of the source files.

7 Equation Appendix

The functions involved in the calculations are quite complicated, so the details are provided here.

7.1 Weighting Functions

$$w_1(Q_1,Q_2,\tau)=\left(\frac{-2\pi}{3}\right)\sqrt{1-\tau^2}\frac{Q_1^3Q_2^3}{Q_2^2+m_\pi^2}I_1(Q_1,Q_2,\tau)$$

$$w_2(Q_1,Q_2,\tau)=\left(\frac{-2\pi}{3}\right)\sqrt{1-\tau^2}\frac{Q_1^3Q_2^3}{(Q_1+Q_2)^2+m_\pi^2}I_2(Q_1,Q_2,\tau)$$

$$\begin{aligned}I_1(Q_1,Q_2,\tau)=X(Q_1,Q_2,\tau)\big[&8P_1P_2(Q_1\cdot Q_2)-2P_1P_3(Q_2^4/m_\mu^2-2Q_2^2)+4P_2P_3Q_1^2-4P_2\\&-2P_1(2-Q_2^2/m_\mu^2+2(Q_1\cdot Q_2)/m_\mu^2)-2P_3(4+Q_1^2/m_\mu^2-2Q_2^2/m_\mu^2)+2/m_\mu^2\big]\\&-2P_1P_2(1+(1-R_{m1})(Q_1\cdot Q_2)/m_\mu^2)+P_1P_3(2-(1-R_{m1})Q_2^2/m_\mu^2)\\&+P_2P_3(2+(1-R_{m1})^2(Q_1\cdot Q_2)/m_\mu^2)+P_1(1-R_{m1})/m_\mu^2+3P_3(1-R_{m1})/m_\mu^2\end{aligned}$$

$$\begin{aligned}I_2(Q_1,Q_2,\tau)=X(Q_1,Q_2,\tau)\big[&4P_1P_2(Q_1\cdot Q_2)+2P_1P_3Q_2^2-2P_1+2P_2P_3Q_1^2-2P_2-4P_3-4/m_\mu^2\big]\\&-2P_1P_2-3P_1(1-R_{m2})/(2m_\mu^2)-3P_2(1-R_{m1})/(2m_\mu^2)-P_3(2-R_{m1}-R_{m2})/(2m_\mu^2)\\&+P_1P_3(2+3(1-R_{m2})Q_2^2/(2m_\mu^2)+(1-R_{m2})^2(Q_1\cdot Q_2)/(2m_\mu^2))\\&+P_2P_3(2+3(1-R_{m1})Q_1^2/(2m_\mu^2)+(1-R_{m1})^2(Q_1\cdot Q_2)/(2m_\mu^2))\end{aligned}$$

$$Q_3^2=Q_1^2+2Q_1\cdot Q_2+Q_2^2$$
$$Q_1\cdot Q_2=Q_1Q_2\tau$$

$$P_i=\frac{1}{Q_i^2},\ i=1,2,3$$

$$X(Q_1,Q_2,\tau)=\frac{1}{Q_1Q_2x}\arctan\left(\frac{zx}{1-z\tau}\right)$$

$$x=\sqrt{1-\tau^2}$$

$$z=\frac{Q_1Q_2}{4m_\mu^2}(1-R_{m1})(1-R_{m2})$$

$$R_{mi}=\sqrt{1+\frac{4m_\mu^2}{Q_i^2}},\ i=1,2$$

7.2 Form Factors

$$\mathcal{F}^{\text{LMD+V}}_{\pi^0\gamma^*\gamma^*}(q_1^2, q_2^2) = \frac{F_\pi}{3} \frac{q_1^2 q_2^2 (q_1^2 + q_2^2) + h_2 q_1^2 q_2^2 + h_5 (q_1^2 + q_2^2) + h_7}{(q_1^2 - M_{V_1}^2)(q_1^2 - M_{V_2}^2)(q_2^2 - M_{V_1}^2)(q_2^2 - M_{V_2}^2)}$$

$$\mathcal{F}_{Q^4}(-Q_1^2, -Q_2^2) = \sqrt{\frac{4\Gamma_{\pi^0 \to \gamma\gamma'}}{\pi \alpha^2 m_\pi^3}} \left[1 - a(Q_1^2 + Q_2^2) + b(Q_1^4 + Q_2^4) + c Q_1^2 Q_2^2 + \cdots\right]$$

$$\mathcal{F}_{Q^6}(-Q_1^2, -Q_2^2) = \sqrt{\frac{4\Gamma_{\pi^0 \to \gamma\gamma'}}{\pi \alpha^2 m_\pi^3}} \left[1 - a(Q_1^2 + Q_2^2) + b(Q_1^4 + Q_2^4) + c Q_1^2 Q_2^2 + d(Q_1^6 + Q_2^6) + e(Q_1^4 Q_2^2 + Q_1^2 Q_2^4) + \cdots\right]$$

8 Table of Constants

Name	Symbol	Value	Units
Fine Structure Constant	α	0.0072973525693	-
Pion Mass	m_π	0.1349768	GeV/c^2
Muon Mass	m_μ	0.1056583745	GeV/c^2
Pion Decay Constant	F_π	0.0924	GeV
Vector Meson Mass 1	M_{V_1}	0.77549	GeV
Vector Meson Mass 2	M_{V_2}	1.465	GeV
LMD+V Parameter 1	h_2	-10.634883404844444	GeV^2
LMD+V Parameter 2	h_5	6.93	GeV^4
LMD+V Parameter 3	h_7	-14.827668978756119	GeV^6
TFF Expansion Param 1	a	1.6613939123981294*	GeV^{-2}
TFF Expansion Param 2	b	2.7619453491551749*	GeV^{-4}
TFF Expansion Param 3	c	3.259027816403921*	GeV^{-6}
TFF Expansion Param 4	d	-4.59258	GeV^{-6}
TFF Expansion Param 5	e	-5.58268	GeV^{-6}
?	Γ and $\Gamma_{\pi^0 \to \gamma\gamma'}$	7.7291993×10^{-9}	GeV

$*$ - The values for the parameters a, b, and c in the table are approximate. Their exact forms are:

$$a = \frac{1}{M_{V_1}^2} + \frac{1}{M_{V_2}^2} + \frac{h_5}{h_7}$$

$$b = \frac{1}{M_{V_1}^4} + \frac{1}{M_{V_2}^4} + \frac{1}{M_{V_1}^2 M_{V_2}^2} + \frac{h_5}{h_7}\left(\frac{1}{M_{V_1}^2} + \frac{1}{M_{V_2}^2}\right)$$

$$c = \left(\frac{1}{M_{V_1}^2} + \frac{1}{M_{V_2}^2}\right)^2 + \frac{h_2}{h_7} + 2\frac{h_5}{h_7}\left(\frac{1}{M_{V_1}^2} + \frac{1}{M_{V_2}^2}\right)$$