# Agentic AI for QCD perturbation theory or: how I learned to stop worrying and love the bot

Erik Lundstrum

Columbia University

July 31, 2026

# Outline

# Preface

**This talk is two things**

1. Details a perturbative QCD calculation.
2. Describes agentic AI workflow which quickly complete these calculations.

**Points I would like to emphasize**

- Choosing appropriately sized tasks.
- Verification strategy.
- Rapid completion of calculations.
- Focus on more accessible one-loop strategy.

Results

- One- and two-loop RI/SMOM-$\overline{\text{MS}}$ matching factors.
- Improved perturbative input for $\epsilon_K$ calculation [N. Christ, Y.Huo in progress].

# Background knowledge

- Graduate student level knowledge of perturbative QCD.
- Confident with dimensional regularization, $\overline{\text{MS}}$ scheme.
- Familiar with the renormalization of weak operators described in Buchalla, Buras and Lautenbacher review article [Rev.Mod.Phys. 68 (1996) 1125-1144].
- Not particularly familiar with RI/SMOM - $\overline{\text{MS}}$ matching.
- Not familiar with FeynCalc.

This is to say I had to figure out quite a bit while working through this. An expert would have a much easier time interacting with Claude.

# Role of perturbation theory in lattice QCD

# Role of QCD perturbation theory in lattice QCD

Perturbation theory plays several roles:

1. Wilson coefficients $C_i(x)$ [Rev.Mod.Phys. 68 (1996) 1125-1144]
2. Converting between regularization schemes

Dimension regulator + $\overline{\text{MS}}$ $\downarrow$

$$H_W = \sum_{i=1}^{10} C_i(\mu) Q_i^{\Delta S=1}(\mu) \tag{1}$$

$\uparrow$ Lattice regulator

Lattice $\rightarrow$ Regularization-independent renorm. $\rightarrow$ DimReg

# Regularization-independent - $\overline{\text{MS}}$ matching factors

Regularization-independent renormalization condition [Nucl. Phys. B445, 81 (1995)]: suitable Green's functions, computed between external off-shell quark and gluon states, in a fixed gauge, coincide with their tree-level value:

1. Evaluate renormalized $\overline{\text{MS}}$ Green's functions with free covariant gauge-fixing parameter $\xi$.
2. Choose an appropriate set of projectors onto spin, color and flavor indices.
3. Apply projectors to renormalized $\overline{\text{MS}}$ Green's functions to determine matching factors.

# Matching factor calculations

$\Delta S = 1$ four-quark operators (four flavors): One-loop

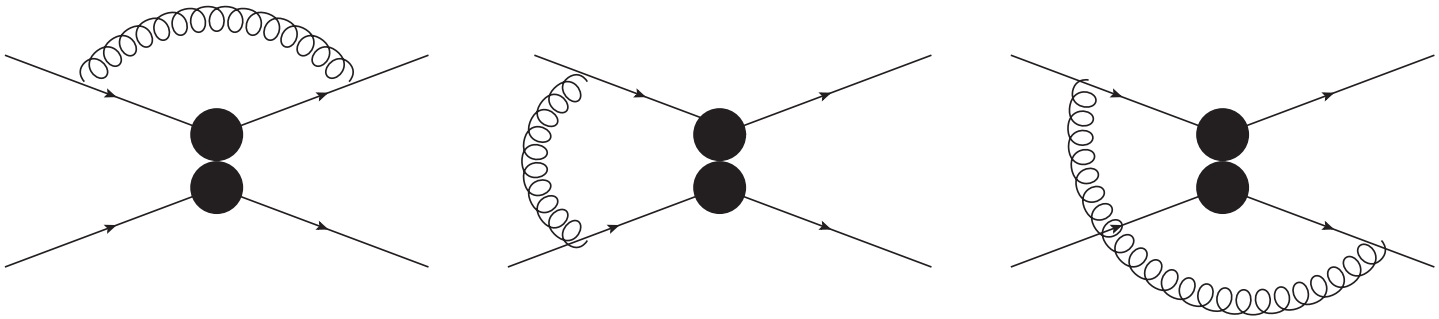

Figure: One-loop QCD corrections to the $\Delta S = 1$ four-quark operators.

$\Delta S = 2$ bi-local operators: Two-loop

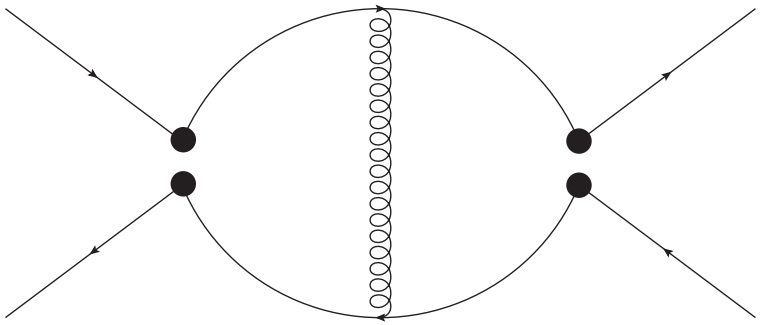

Figure: Two-loop QCD corrections to the $\Delta S = 2$ process.

# Two-loop renormalization

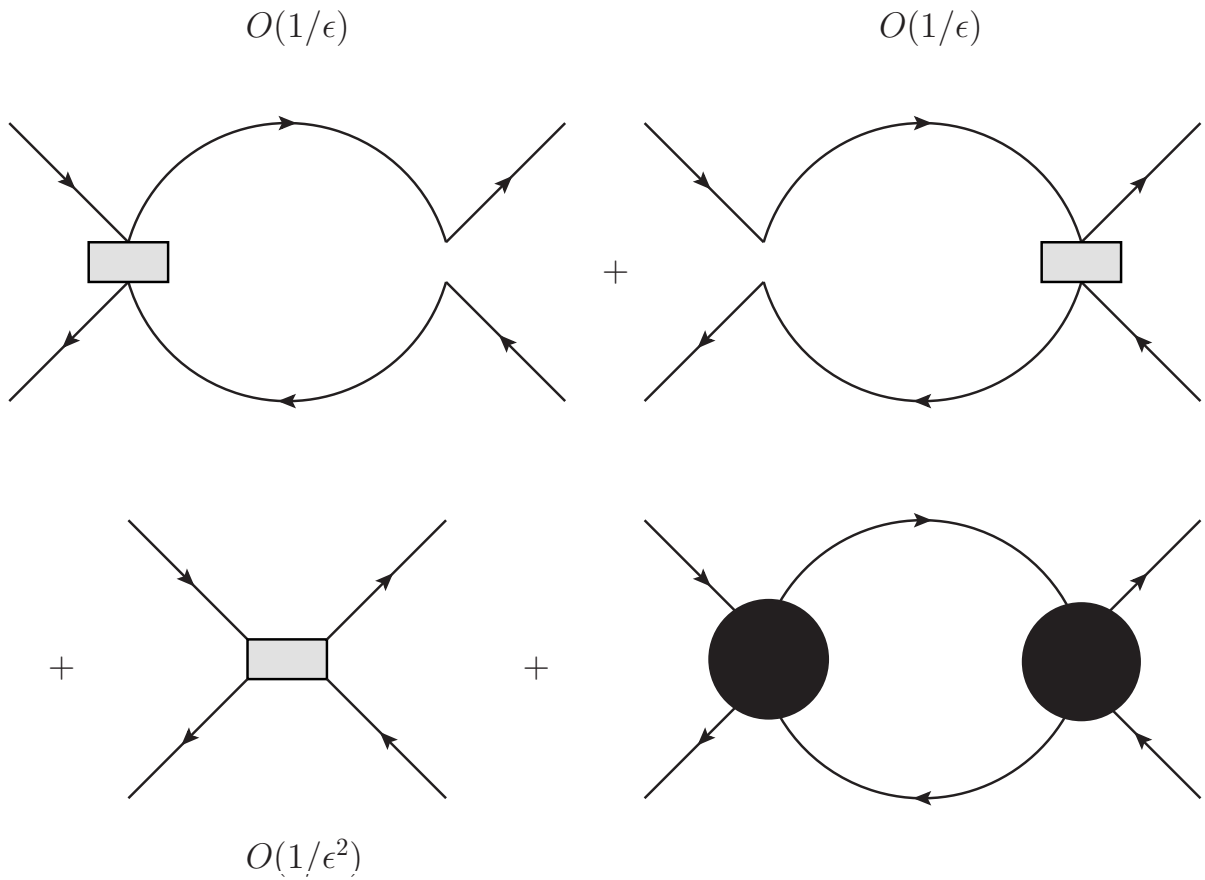


Figure: Counterterm insertions required to cancel divergences. Grey rectangles denote local counterterm insertions and black circles represent one-gluon corrections to the $\Delta S = 1$ operators.

# Automating diagram evaluation with Claude code

## Agentic AI strategy

1. To ensure correctness, offload as much as possible to verified tools: Mathematica, FeynCalc, LiteRed etc.
2. Break the problem into SMALL steps, build up layers of functionality that have been verified.
3. Apply verified solution chain to all diagrams.

## Key takeaways

1. Claude is extremely effective IF the user understand how to solve the problem.
2. Essentially no transcription errors, errors due to Claude making incorrect assumptions.
3. Once all the functionality has been verified, Claude can rip through all the relevant diagrams very quickly (one vs two loops).

# Important takeaways

Build up VERY DETAILED prompts in the following manner

- EL: “DON’T CODE YET! (initial prompt for small task) ”
- EL: “Is anything unclear about the task I have given you? Please ask me clarifying questions.”
- EL: “Please explain to me in detail what it is you think I want you to do.”
- A lot of this functionality has become part of the default Claude code.

# Claude strengths

Claude is an incredible Feynman diagram machine

- Give Claude a hand-drawn diagram with external momenta labels.
- I encountered NO transcription errors from input diagram $\rightarrow$ Feynman rules $\rightarrow$ Mathematica code.
- Claude writes FeynCalc code without making mistakes.
- $\Rightarrow$ NO worrying about algebraic mistakes or missing factors of $1/2$.

# Claude weaknesses

Claude will make mistakes when you give AMBIGUOUS prompts. It will make assumptions which are often incorrect.

1. Four- vs. d-dimensional Dirac algebras.
2. Naive dimensional regularization vs 't Hooft Veltman schemes.
3. Will try to write its own code to evaluate scalar master integrals.
4. Won't recognize when to appropriately make Fierz transformations.

# QCD corrections to $\Delta S = 1$ four-quark operators

(Cross checking with Christoph Lehner)

# Four-quark effective operators

$$Q_1^c = (\bar{s}_i c_j)_{V-A}(\bar{c}_j d_i)_{V-A} \qquad Q_2^c = (\bar{s}c)_{V-A}(\bar{c}d)_{V-A},$$

$$Q_1^u = (\bar{s}_i u_j)_{V-A}(\bar{u}_j d_i)_{V-A}, \qquad Q_2^u = (\bar{s}u)_{V-A}(\bar{u}d)_{V-A},$$

$$Q_3 = (\bar{s}d)_{V-A}\sum_q(\bar{q}q)_{V-A}, \qquad Q_4 = (\bar{s}_i d_j)_{V-A}\sum_q(\bar{q}_j q_i)_{V-A},$$

$$Q_5 = (\bar{s}d)_{V-A}\sum_q(\bar{q}q)_{V+A}, \qquad Q_6 = (\bar{s}_i d_j)_{V-A}\sum_q(\bar{q}_j q_i)_{V+A},$$

$$Q_7 = \frac{3}{2}(\bar{s}d)_{V-A}\sum_q e_q(\bar{q}q)_{V+A}, \qquad Q_8 = \frac{3}{2}(\bar{s}_i d_j)_{V-A}\sum_q e_q(\bar{q}_j q_i)_{V+A},$$

$$Q_9 = \frac{3}{2}(\bar{s}d)_{V-A}\sum_q e_q(\bar{q}q)_{V-A}, \qquad Q_{10} = \frac{3}{2}(\bar{s}_i d_j)_{V-A}\sum_q e_q(\bar{q}_j q_i)_{V-A}.$$

Where $(\bar{q}q)_{V\pm A} = (\bar{q}(1 \pm \gamma_5)q)$ and the subscripts $i, j$ are colour indices.

# Diagrams

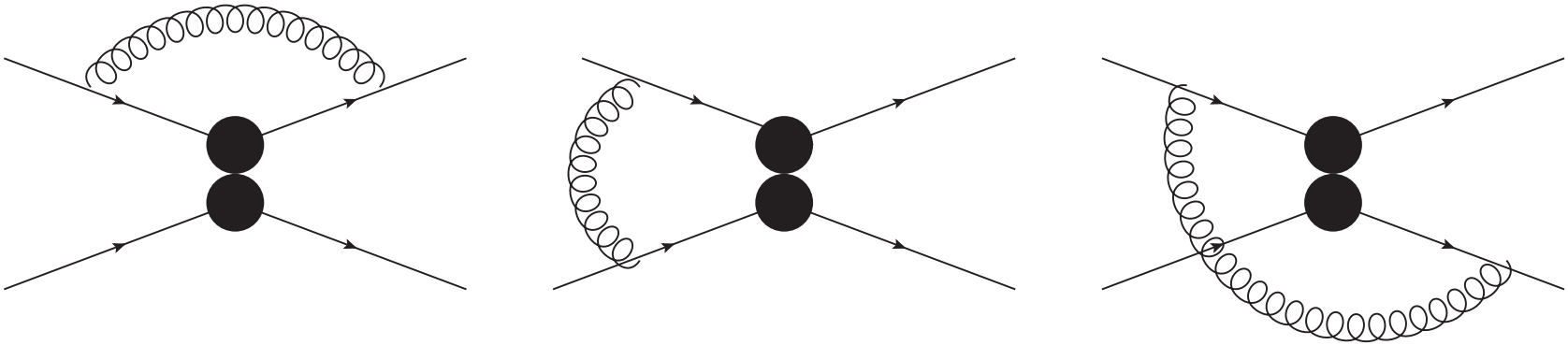

Figure: One-loop QCD corrections to the $\Delta S = 1$ four-quark operators.

Gluonic corrections mix up the colours:

$$\langle Q_1 \rangle = C_1 Q_1 + C_2 Q_2. \tag{2}$$

# Training and checking Claude

# Process

I want to build up a set of verified results that I can call in later functionality. My strategy is as follows, I will discuss these in more detail in what follows.

1. Identify and evaluate all required tensor integrals, save the results.
2. For each diagram 1-6, identify the Dirac structures in the numerator and combine with integral results.
3. Project the many resulting structures onto the basis defined by C. Lehner and C. Sturm [Phys.Rev.D 84 (2011) 014001].
4. Combine diagrams 1-6 as required to find the one-loop correction to each operator $Q_1 - Q_{10}$.
5. Perform $\overline{\text{MS}}$ renormalization.

# Process

Note on file structure

- Verify each step in an example Mathematica file.
- CLAUDE.md file instructs Claude to reference and use verified example codes.
- (Opposed to making new code from scratch/memory)
- Write out intermediate verified results to disk and load back in for subsequent steps.

# Integral evaluation and check

An example integral is

$$\int \frac{d^4k}{(2\pi)^4}\frac{1}{k^2}\left(\eta^{\mu\nu}-(1-\xi)\frac{k^\mu k^\nu}{k^2}\right)\gamma^\mu\frac{(\not{p}_1+\not{k})}{(p_1+k)^2}P_L\frac{(\not{p}_2+\not{k})}{(p_2+k)^2}\gamma^\nu \qquad (3)$$

- Remove all the Dirac structures.
- Evaluate the integrals with the numerator structures $\{1, k^\nu, k^\nu k^\mu, k^\nu k^\mu k^\rho, k^\nu k^\mu k^\rho k^\sigma\}$.
- Easy to verify input Mathematica code.
- I know FeynCalc will correctly evaluate the integrals.
- Therefore I can trust the results of this step are correct.

# Dirac structures

Continuing with the previous example integral

$$\int \frac{d^4k}{(2\pi)^4}\frac{1}{k^2}\left(\eta^{\mu\nu}-(1-\xi)\frac{k^\mu k^\nu}{k^2}\right)\gamma^\mu\frac{(\not{p}_1+\not{k})}{(p_1+k)^2}P_L\frac{(\not{p}_2+\not{k})}{(p_2+k)^2}\gamma^\nu \tag{4}$$

- Isolate the Dirac structure using $\not{k}=\gamma^\mu k_\mu$.
- Identify the $k^\mu k^\nu\ldots$ pieces with cached integrals from the previous step.
- I am left with a LOT of Dirac structures. For example

# Dirac Structures

For example

$$\left(\bar{s}\gamma^{\mu}P_L d\right)\left(\bar{u}\gamma_{\mu}P_L u\right) \qquad \left(\bar{s}\not{p}_1\not{p}_2\gamma^{\mu}P_L d\right)\left(\bar{u}\gamma_{\mu}P_L u\right)$$

$$\left(\bar{s}\not{p}_1 P_L d\right)\left(\bar{u}\not{p}_1 P_L u\right) \qquad \left(\bar{s}\not{p}_1\gamma^{\nu}\gamma^{\mu}P_L d\right)\left(\bar{u}\gamma_{\mu}\not{p}_2\gamma_{\nu}P_L u\right)$$

$$\left(\bar{s}\not{p}_1 P_L d\right)\left(\bar{u}\not{p}_2 P_L u\right) \qquad \left(\bar{s}\gamma^{\sigma}\gamma^{\nu}\gamma^{\mu}P_L d\right)\left(\bar{u}\gamma_{\mu}\gamma_{\sigma}\gamma_{\nu}P_L u\right)$$

Next, I need to reduce all of these pieces to the basis given in C. Lehner and C. Sturm [Phys.Rev.D 84 (2011) 014001].

# Dirac Structures

I use a projection onto the basis in [Phys.Rev.D 84 (2011) 014001].
Checking for correctness:

- I (pen and paper) use Dirac matrix identities to simplify 5 of the Dirac structures that appear.
- Claude writes Mathematica code to perform the projection.
- I make Claude verify the Mathematica code against my correct pen and paper results.
- Now I have a verified projection code to use on the hundreds of Dirac structures that appear.

This same code is used for all six diagrams. These diagrams in different combinations create the full amplitudes for all ten operators $Q_1, ...., Q_{10}$.

# Stop to check Physics

Now I want to run some overall tests to confirm my results for $Q_1, ..., Q_{10}$. My tests are as follows

1. Identify the $1/\epsilon$ pole associated with the quark wavefunction renormalization factor $Z_q^{-2}$ in each of the final expressions.
2. Check that any term containing $\not{p}_{1,2}$ in the numerator has no $1/\epsilon$ pole.
3. Check that, after removing the pole associated with $Z_q^{-2}$ , the remaining poles are gauge-independent.
4. Finally, I can check that the poles I find agree with the results in [Phys.Rev.D 84 (2011)].

The above tests seem to be strong constraints, and once they were all satisfied I went to check the finite amplitudes for three flavors against [Phys.Rev.D 84 (2011)] and found them to be correct.

# QCD corrections to $\Delta S = 2$ bi-local operators

(In progress)

# QCD corrections to bi-local operators

- Necessary input for lattice QCD calculation of the long-distance contribution to the CP-violating parameter $\epsilon_K$.
- RI/SMOM matching factors are known only to $O(\alpha_s^0)$, and so this calculation will eliminate a large systematic uncertainty [Phys.Rev.D 109 (2024) 5, 054501].
- $\sim 250$ diagrams reducing to $\sim 150$ master integrals.

# QCD corrections to bi-local operators

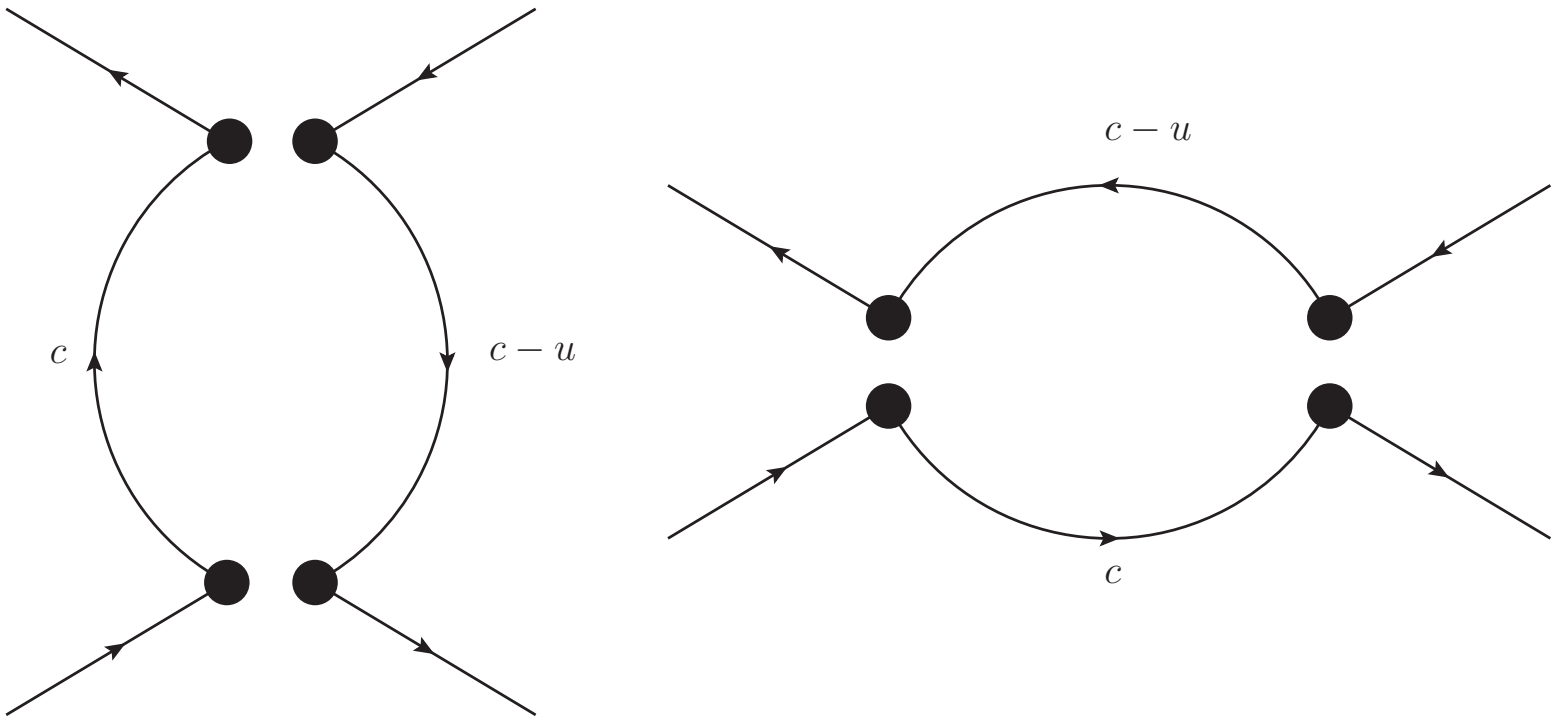


Figure: One-loop diagrams contributing to the bi-local operator at $O(\alpha_s^0)$. Quark flavors running around the loops are labelled on the diagram.

# Physics Strategy

Extend strategy used by Gorbahn *et al.* [JHEP 09 (2025) 011] for the $\Delta S = 2$ RI/SMOM matching factors.

1. Identify tree-level spinor structures that will contribute to RI/SMOM matching factors.
2. Define projectors onto these relevant structures e.g.

$$\begin{aligned} \Pi_{\mu}(\Gamma_1 P_L \otimes \Gamma_2 P_L) &= \mathrm{tr}\, \gamma^{\mu} \Gamma_1 \gamma_{\mu} \Gamma_2, \\ \Pi_{ij}(\Gamma_1 P_L \otimes \Gamma_2 P_L) &= \mathrm{tr}\, \not{p}_i \Gamma_1 \not{p}_j \Gamma_2, \end{aligned} \tag{5}$$

   Notice that $P_L$ are omitted $\Rightarrow$ avoid $\gamma_5$ NDR ambiguities.
3. Apply projectors to the loop integrands BEFORE performing loop integrals $\Rightarrow$ removes spin structures and avoid tensor reduction.

# Claude strategy

We apply the same strategy as for the one-loop case: verify functionality on test cases then let Claude automatically generate Mathematica code that applies the strategy to each diagram.

1. Give Claude a hand-drawn set of two-loop integrals representing the different types of topologies with external momentum routing.
2. Claude automatically generates the integrand for the diagram and applied projectors.
3. Projected integrands are fed to LiteRed2 [arXiv:1212.2685 [hep-ph]] which reduces them to a set of master scalar integrals.
4. pySecDec [Comput.Phys.Commun. 295 (2024) 108956] is used to numerically evaluate the resulting master scalar integrals.

# Physics checks

1. Check for cancellation of non-local divergences with one-loop counterterm diagrams.
2. Coefficients of local two-loop counterterms are constrained by the value of one-loop counterterms and we can therefore check their value [Nucl.Phys.B 61 (1973) 455-468].
3. Match conventions and normalizations by comparing one-loop amplitudes to known values [Phys.Rev.D 109 (2024) 5, 054501].
4. Finally, the same calculation has been performed at vanishing external momenta, against which we can compare [Phys.Rev.D 82 (2010) 094026].

# Summary and results

- Successfully calculated the $\Delta S = 1$ four-flavor RI/SMOM - $\overline{\text{MS}}$ matching factors using Claude code.
- Two-loop bi-local operators calculation will soon be complete.
- Claude code makes one- and two-loop perturbative QCD calculations a breeze IF the user understands the details.
- Agentic AI strategy:
  1. Rely as much as possible on existing code bases where correctness is assured.
  2. Break the problem into many SMALL steps, each of which can be verified.
  3. Make very detailed prompts by asking Claude what is unclear / it doesn't understand.
  4. Claude quickly generates code for all diagrams.

# Bonus slide: Claude specifications

One can choose various models and intelligence settings for the agent. I use

- Claude Pro ($20 per month tier).
- Claude Opus 4.7 (highest performance model at the time).
- Highest thinking effort (Highest token usage).
- Claude code v2.1.128
- All communication with Claude via CLI.
- I don't use any skills functionality or pre-trained tools.
- Mathematica 14.0
- FeynCalc 10.2.0

These setting consume the most tokens! Allocation was the main bottleneck.

# Bonus slide: Training Claude

I asked "How would you summarize my strategy for interacting with you?" Claude's paraphrased response is

1. **You retain conceptual control; you delegate the mechanical encoding.** You decide the physics — operator basis, flavor routing, ... — and hand me the FeynCalc encoding.
2. **Mirror-and-perturb.** Almost every new calculation is "do this exactly like the analogous case you already did, but change X."
3. **Plan-then-execute.** Your repeated "tell me how you'll do this before you do it" forces me to declare the algebraic plan up front so you can correct misunderstandings before they're encoded into Mathematica. You confirm or correct, then I run
4. **Continuous cross-validation against literature.** Every result gets compared to a reference ...
5. **Incremental, persistent scaffolding.** Results aren't one-offs — they accumulate as named bundles ... that the next calculation imports. CLAUDE.md is the running ledger of what's verified, so context survives across sessions.