SUPER_QUANTA Software Documentation

Version 1.1

Prepared by:

M. Costa, C. Costa, S. Dimou, P. K. Chrysanthis, H. Panagopoulos

Rinnoco Ltd and University of Cyprus

Email: info@rinnoco.com

April 28, 2026

This document provides comprehensive documentation for the use of the `SUPER_QUANTA` software, which facilitates the calculation of Feynman diagrams in lattice quantum field theory. It includes detailed guidance on installation, operational use, worked examples, and troubleshooting procedures. The `SUPER_QUANTA` software derives its name from the research project **EXCELLENCE/0524/0208**, jointly implemented by *Rinnoco Ltd* and the *University of Cyprus*. It constitutes a continuation and substantial refinement of the earlier `FEDILA` software, originally developed under the project **CONCEPT/0823/0052** as a proof of concept. This upgraded version introduces enhanced computational efficiency and integrates the **overlap fermion action**, thus broadening its applicability to supersymmetric theories and chirally invariant formulations. Additionally, Rinnoco Ltd's expertise in data storage and compression has enabled a streamlined computational workflow by incorporating **two tools, compression and decompression** of the data produced. This not only reduces redundancy but also facilitates the reuse of intermediate results, addressing two key limitations of the previous implementation. These improvements mark a significant advancement over the original `FEDILA` framework and position `SUPER_QUANTA` as a powerful tool for research in lattice perturbation theory and quantum field theory.

A Guide to Calculating Feynman Diagrams with SUPER_QUANTA software

M. Costa,[1,2,*] C. Costa,[1,†] S. Dimou,[1,‡] P.K. Chrysanthis,[1,§] and H. Panagopoulos[2,¶]

[1] *Rinnoco Ltd, Limassol, CY-3047, Cyprus*

[2] *Department of Physics, University of Cyprus, Nicosia, CY-1678, Cyprus*

SUPER_QUANTA is a pioneering software solution designed to revolutionize analytic computations in quantum field theory, targeting both the continuum and discretized (lattice) formulations of the fundamental forces of nature. Its architecture includes support for advanced formulations such as the overlap fermion formalism. This documentation and tutorial series provides comprehensive guidance for utilizing SUPER_QUANTA to compute Feynman diagrams. The software is implemented as a flexible Mathematica package that supports a variety of renormalization schemes, including the continuum Gauge Invariant Renormalization Scheme (GIRS), and is tailored for extracting meaningful results from supersymmetric quantum field theories. In addition, SUPER_QUANTA includes functionality for efficient compression of large algebraic expressions, enabling users to store intermediate results and recover them for future use without loss of information. This guide presents both conceptual explanations of the theoretical foundations and step-by-step tutorials designed to help users fully exploit each Mathematica command included in the package.

1. INTRODUCTION

SUPER_QUANTA is a state-of-the-art software package for perturbative calculations, designed for the in-depth exploration of physical properties at the microscopic scale, particularly within the framework of Quantum Field Theories (QFTs). Developed using the symbolic high-level Mathematica language, SUPER_QUANTA provides a versatile and robust environment for theoretical physicists and researchers engaged in the study of QFTs. This documentation serves as both an introduction to the software and a guide to its various features and functionalities.

One of the central challenges in quantum field theory is managing the infinities that arise in perturbative calculations. **Renormalization** is the process of redefining physical quantities to make these divergences manageable and physically meaningful. SUPER_QUANTA supports several **renormalization schemes** and **regularization techniques**, enabling the handling of divergent integrals:

- **Gauge Invariant Renormalization Scheme (GIRS)**: A continuum-based approach that preserves gauge invariance, ensuring that the physical predictions of the theory remain consistent with its fundamental symmetries. GIRS is particularly advantageous in non-Abelian gauge theories, where it plays a critical role in maintaining the integrity and consistency of renormalized quantities.
- **Dimensional Regularization**: SUPER_QUANTA incorporates dimensional regularization, a technique that analytically continues the number of space-time dimensions to regulate divergent integrals. By performing computations in non-integer dimensions, this method allows for a systematic treatment of ultraviolet divergences and is widely used in perturbative QFT.
- **Lattice Regularization**: This non-perturbative method discretizes space-time into a finite lattice, enabling the formulation of QFTs in a computationally tractable form. Lattice regularization is essential for numerical simulations and for investigating strongly coupled quantum field theories where perturbative methods fail.

The specific technological objectives of the SUPER_QUANTA project include the following key developments:

- **Mathematica Routine for Stout Smearing**: Develop a Mathematica routine that applies the stout smearing procedure iteratively to gluon links. Both isotropic and non-isotropic smearing methods will be supported.
- **Statistical Inference Integration**: Incorporate state-of-the-art numerical integration techniques into the Fortran codes (generated via Medacode from Mathematica). These techniques will provide high-precision results with accurate error estimates, particularly in the context of the extrapolation procedure.

*Electronic address: marios.c@rinnoco.com
†Electronic address: costa.c@rinnoco.com
‡Electronic address: salomi@rinnoco.com
§Electronic address: panos@rinnoco.com
¶Electronic address: haris@ucy.ac.cy

- **Multi-loop Diagram Evaluation Framework**: Design an algorithmic framework for evaluating multi-loop Feynman diagrams using concepts from Graph Theory and Computer Science.
- **Automated GIRS Calculations**: Develop a software package for automated perturbative calculations in both lattice and continuum formulations, specifically within the Gauge Invariant Renormalization Scheme (GIRS). This will significantly improve the efficiency and versatility of such computations.
- **Supersymmetric QCD Capabilities**: Extend the software to support calculations in supersymmetric QCD, including gluino and squark fields and their interactions. This will enable detailed investigations into supersymmetric extensions of the Standard Model.
- **Overlap Fermion Implementation**: Incorporate support for overlap fermions into the computational framework. This addition ensures chiral symmetry at finite lattice spacing and is essential for accurately exploring supersymmetric models and related phenomena.
- **Evaluation of Divergent Integrals**: Evaluate a basis of superficially divergent Feynman integrals arising in multi-loop computations, extending beyond leading-order lattice artifacts. This is essential for precision lattice perturbation theory and beneficial to the wider scientific community.
- **Intermediate Data Storage**: Utilize our efficient symbolic and numerical data storage tool to archive intermediate results from Feynman diagram computations. This enables reproducibility, data reuse, and streamlined workflows.

2. INSTALLATION AND SETUP

Since SUPER_QUANTA is a Mathematica package, you must first install Mathematica. We provide a link to the instructions for installing the Wolfram Mathematica platform.

To begin using SUPER_QUANTA, follow these steps:

1. **System Requirements**: Ensure that your system meets the necessary requirements for running Mathematica and that you have the appropriate version installed.
2. **Installation**: Download the SUPER_QUANTA package from the official repository and install it within your Mathematica environment. Detailed installation instructions are provided in the package's README file.

   To install the package, copy the tar file `SUPER_QUANTA_software-main.zip` to your disk and unpack it with:

   ```
   >> unzip SUPER_QUANTA_software-main.zip
   ```

   Change to the SUPER_QUANTA directory:

   ```
   >> cd SUPER_QUANTA_software-main
   ```

   Open the file `input.m` and replace `"username"` with your own username. After making the changes, save the file.

   ```
   >> emacs input.m
   [Replace "username" with your username]
   ```

   For better organization of the calculations, we suggest creating two additional directories for each project:

   ```
   >> mkdir fortranfiles
   >> mkdir outfiles
   ```

   Open a Mathematica kernel and you can read the input.m. To read input.m, you need to provide the path where the SUPER_QUANTA_software-main folder is saved. The reader can follow the inputs In[1] to In[129] in Mathematica to become more familiar with this software calculating a Feynman diagram on the lattice.

```
In[1]:= << /home/username/SUPER_QUANTA_software-main/input.m
```

Now, all the commands needed to calculate a Feynman diagram can be used in your Mathematica kernel. Note that these new Mathematica commands start with a lowercase letter, in contrast to the commands from the Mathematica library, which start with an uppercase letter.

Getting Started

The software is designed to be user-friendly, with a command structure that is intuitive for those familiar with QFT and Mathematica. After launching Mathematica and loading the SUPER_QUANTA package by reading the appropriate `input.m` file, users should become familiar with the software's notation. In general, our commands use the following notations:

- nDim is the number of dimensions of the theory.
- e is related to the dimension (nDim) of spacetime, through: nDim = 4 - 2 e.
- Nc is the number of colors of the theory.
- Nf is the number of flavors of the theory.
- im is the imaginary unit.
- hv = 0 (1) in naive ('t Hooft-Veltman) definition of γ_5.
- k[i] is the momentum of the i$^{\text{th}}$ leg of the vertex.
- q[i] is the momentum of the i$^{\text{th}}$ external leg of the Feynman diagram.
- p[i] is the loop momenta of the Feynman diagram.
- a[i] is the i$^{\text{th}}$ color external index in the adjoint representation.
- c[i] is the i$^{\text{th}}$ color internal index in the adjoint representation.
- af[i] is the i$^{\text{th}}$ color external index in the fundamental representation.
- cf[i] is the i$^{\text{th}}$ color internal index in the fundamental representation.
- mu[i] (or nu[i]) is the i$^{\text{th}}$ external Lorentz index.
- rho[i] is the i$^{\text{th}}$ Lorentz index to be summed over.
- fnu[i] is the i$^{\text{th}}$ Dirac index.
- fl[i] is the i$^{\text{th}}$ flavor index.
- f[a,b,c] are the antisymmetric structure constants.
- epsilon[mu,nu,rho,sigma] is the Levi-Civita tensor.
- d[a,b,c] is the symmetric trace of a 3-generator product: d[a,b,c] = $2Tr(T^aT^bT^c + T^aT^cT^b)$, where T is the generator of the gauge group. This notation takes into account the cyclicity of the trace.
- traceGen[a,b,...,c,d] is the trace of the product of generators: $Tr(T^aT^b...T^cT^d)$. For example, traceGen[a,b] = $\delta_{ab}/2$ and traceGen[ ] = Nc (`traceGen` can also be called as `spur`.).
- productGen[i,a,b,...,c,d,j] is the product of generators, where the first and the last arguments denote color indices in the fundamental representation: $(T^aT^b...T^cT^d)_{ij}$ (`productGen` can also be called as `openspur`.).
- delm[a,b] is the Kronecker delta for Lorentz indices.
- delc[a,b] is the Kronecker delta for color indices.
- delf[a,b] is the Kronecker delta for flavor indices.
- delta[a + b + ...] is the Dirac delta.
- deltaL[a,b,...,c] is a Kronecker delta whose 2 or more arguments are Lorentz indices.
- deltaColor[a,b] is a Kronecker delta of two color indices in the adjoint representation.
- traceDirac[a,b,...,c,d] is the trace of a product of Dirac matrices. The arguments of traceDirac can be: gamma5, nu[_] (an external Lorentz index taking values: 1,2,3,4), rho[_] (a dummy Lorentz index taking values: 1,2,3,4,...,nDim) or X, Y (unspecified Dirac matrices; each may appear at most once). For example: traceDirac[gamma5,nu[3],X,rho[2],gamma5,Y] = $\mathrm{Tr}(\gamma_5\gamma_{\nu_3}\mathrm{X}\gamma_{\rho_2}\gamma_5\mathrm{Y})$ (traceDirac can also be called as `gtrace`.).

- productDirac[i,a,b,...,c,d,j] is a product of Dirac matrices, where the first and the last arguments denote the spinor indices. For example: productDirac[i,gamma5,nu[3],X,rho[2],gamma5,Y,j] = $(\gamma_5\gamma_{\nu_3}\mathrm{X}\gamma_{\rho_2}\gamma_5\mathrm{Y})_{ij}$ (productDirac can also be called as `dtrace`.).

Further notations are provided in Section 3.5.

3. COMMANDS OF THE SOFTWARE

3.1. Creation of vertices

The initial step in evaluating each Feynman diagram involves determining the algebraic expressions of the vertices. Each calculation focuses on evaluating Feynman diagrams that encompass various unique vertices. These vertices have distinct characteristics as they are derived from different actions and operators, such as improved gluon and fermion actions/operators, the Supersymmetric QCD (SQCD) action/operators, actions incorporating "background" and "quantum" gauge fields, and actions using stout links, among others.

The primary task in this section is to describe the Mathematica commands for calculating these vertices, which includes both pure gluon vertices and those involving fermions. The procedures of computing the lattice vertices involve the manipulation of products of the link variables (like Wilson loops or Wilson lines), such as expanding the links in terms of the gluon field up to the required order, and performing Fourier transformations of fields to represent vertices in momentum space.

The conventions for the Fourier transformations are:

$$\text{Quark fields:}\quad \tilde{\psi}(k) = \int d^4x\, e^{-ik\cdot x}\,\psi(x), \tag{1}$$

$$\text{Squark fields:}\quad \tilde{A}_{\pm}(k) = \int d^4x\, e^{\mp ik\cdot x}\,A_{\pm}(x), \tag{2}$$

$$\text{Gluon fields:}\quad \tilde{u}_{\mu}(k) = \int d^4x\, e^{-ik\cdot x}\,u_{\mu}(x), \tag{3}$$

$$\text{Gluino fields:}\quad \tilde{\lambda}(k) = \int d^4x\, e^{-ik\cdot x}\,\lambda(x), \tag{4}$$

$$\text{Ghost fields:}\quad \tilde{c}(k) = \int d^4x\, e^{-ik\cdot x}\,c(x). \tag{5}$$

In general, an n-point vertex can be expressed as:

$$\frac{1}{(2\pi)^{4n}}\int d^4k_1\ldots d^4k_n \sum_{\substack{\mu_1,\ldots,\mu_n\\ a_1,\ldots,a_n}} \left(\prod_{i=1}^{n}\tilde{X}^{a_i}_{\mu_i}(k_i)\right) V^{a_1,\ldots,a_n}_{\mu_1,\ldots,\mu_n}(k_1,\ldots,k_n) \tag{6}$$

where k_j denote momenta; a_j are color indices either in the adjoint or in the fundamental representation; μ_j are Lorentz indices. $\tilde{X}^{a_i}_{\mu_i}(k_i)$ are fields of the theory, which appear in the vertex, with momentum k and Lorentz (color) index μ (a). Note that only gluon fields have Lorentz indices.

As an example, we provide a lattice discretized SQCD action where we extend Wilson's formulation of the QCD action to encompass SUSY partner fields as well. In this standard discretization quarks, squarks and gluinos are defined on the lattice sites, while gluons are defined on the links of the lattice: $U_\mu(x) = e^{igaT^\alpha u^\alpha_\mu(x+a\hat{\mu}/2)}$; α is a color index in the adjoint representation of the gauge group. This formulation leaves no SUSY generators intact, and it also breaks chiral symmetry; thus, the need for fine-tuning will arise in numerical simulations of SQCD. For Wilson-type quarks and gluinos, the Euclidean action $\mathcal{S}^L_{\rm SQCD}$ on the lattice becomes (in the Wess-Zumino gauge):

$$\begin{aligned}
\mathcal{S}^L_{\rm SQCD} \;=\;& a^4\sum_x\Big[\frac{N_c}{g^2}\sum_{\mu,\,\nu}\left(1-\frac{1}{N_c}\,\mathrm{Tr}U_{\mu\,\nu}\right)+\sum_\mu \mathrm{Tr}\left(\bar{\lambda}\gamma_\mu\mathcal{D}_\mu\lambda\right)-a\frac{r}{2}\mathrm{Tr}\left(\bar{\lambda}\mathcal{D}^2\lambda\right)\\
&+\;\sum_\mu\Big(\mathcal{D}_\mu A^\dagger_+\mathcal{D}_\mu A_+ + \mathcal{D}_\mu A_-\mathcal{D}_\mu A^\dagger_- + \bar{\psi}\gamma_\mu\mathcal{D}_\mu\psi\Big)-a\frac{r}{2}\bar{\psi}\mathcal{D}^2\psi\\
&+\;i\sqrt{2}g\big(A^\dagger_+\bar{\lambda}^\alpha T^\alpha P_+\psi-\bar{\psi}P_-\lambda^\alpha T^\alpha A_+ + A_-\bar{\lambda}^\alpha T^\alpha P_-\psi-\bar{\psi}P_+\lambda^\alpha T^\alpha A^\dagger_-\big)\\
&+\;\frac{1}{2}g^2(A^\dagger_+T^\alpha A_+ - A_-T^\alpha A^\dagger_-)^2 - m(\bar{\psi}\psi - mA^\dagger_+A_+ - mA_-A^\dagger_-)\Big],
\end{aligned} \tag{7}$$

where: a is the lattice spacing, $U_{\mu\nu}(x) = U_\mu(x)U_\nu(x+a\hat{\mu})U^\dagger_\mu(x+a\hat{\nu})U^\dagger_\nu(x)$, and a summation over flavors is understood in the last three lines of Eq. (7). The 4-vector x is restricted to the values $x = na$, with n being an integer 4-vector.

The terms proportional to the Wilson parameter, r, eliminate the problem of fermion doubling, at the expense of breaking chiral invariance. In the limit $a \to 0$ the lattice action reproduces the continuum one. The bare couplings for the Yukawa and quartic terms (last two lines of Eq.(7)) need not coincide with the gauge coupling g; this requirement is imposed on the respective renormalized values.

The definitions of the covariant derivatives are as follows:

$$\mathcal{D}_\mu \lambda(x) \equiv \frac{1}{2a}\Big[U_\mu(x)\lambda(x+a\hat{\mu})U_\mu^\dagger(x) - U_\mu^\dagger(x-a\hat{\mu})\lambda(x-a\hat{\mu})U_\mu(x-a\hat{\mu})\Big] \tag{8}$$

$$\mathcal{D}^2 \lambda(x) \equiv \frac{1}{a^2}\sum_\mu \Big[U_\mu(x)\lambda(x+a\hat{\mu})U_\mu^\dagger(x) - 2\lambda(x) + U_\mu^\dagger(x-a\hat{\mu})\lambda(x-a\hat{\mu})U_\mu(x-a\hat{\mu})\Big] \tag{9}$$

$$\mathcal{D}_\mu \psi(x) \equiv \frac{1}{2a}\Big[U_\mu(x)\psi(x+a\hat{\mu}) - U_\mu^\dagger(x-a\hat{\mu})\psi(x-a\hat{\mu})\Big] \tag{10}$$

$$\mathcal{D}^2 \psi(x) \equiv \frac{1}{a^2}\sum_\mu \Big[U_\mu(x)\psi(x+a\hat{\mu}) - 2\psi(x) + U_\mu^\dagger(x-a\hat{\mu})\psi(x-a\hat{\mu})\Big] \tag{11}$$

$$\mathcal{D}_\mu A_+(x) \equiv \frac{1}{a}\Big[U_\mu(x)A_+(x+a\hat{\mu}) - A_+(x)\Big] \tag{12}$$

$$\mathcal{D}_\mu A_+^\dagger(x) \equiv \frac{1}{a}\Big[A_+^\dagger(x+a\hat{\mu})U_\mu^\dagger(x) - A_+^\dagger(x)\Big] \tag{13}$$

$$\mathcal{D}_\mu A_-(x) \equiv \frac{1}{a}\Big[A_-(x+a\hat{\mu})U_\mu^\dagger(x) - A_-(x)\Big] \tag{14}$$

$$\mathcal{D}_\mu A_-^\dagger(x) \equiv \frac{1}{a}\Big[U_\mu(x)A_-^\dagger(x+a\hat{\mu}) - A_-^\dagger(x)\Big] \tag{15}$$

Note that in Eqs. (12)-(15), in order not to involve more than two lattice points, we do not use the symmetric derivative.

By analogy to the continuum case, a discrete version of a gauge-fixing term, together with the compensating ghost field term, must be added to the action, in order to avoid divergences from the integration over gauge orbits; these terms are the same as in the non-supersymmetric case. Although these terms can be found in literature, we present them here for the sake of completeness:

$$S_{GF}^L = \frac{1}{2\alpha}a^2 \sum_x \sum_\mu \mathrm{Tr}\left(u_\mu(x+a\hat{\mu}/2) - u_\mu(x-a\hat{\mu}/2)\right)^2. \tag{16}$$

$$\begin{aligned} S_{Ghost}^L = {} & 2a^2 \sum_x \sum_\mu \mathrm{Tr}\{(\bar{c}(x+a\hat{\mu}) - \bar{c}(x))(c(x+a\hat{\mu}) - c(x) \\ & + ig[u_\mu(x+a\hat{\mu}/2), c(x)] + \frac{1}{2}ig[u_\mu(x+a\hat{\mu}/2), c(x+a\hat{\mu}) - c(x)] \\ & - \frac{1}{12}g^2[u_\mu(x+a\hat{\mu}/2), [u_\mu(x+a\hat{\mu}/2), c(x+a\hat{\mu}) - c(x)]])\} + \mathcal{O}(g^3). \end{aligned} \tag{17}$$

Similarly, a standard "measure" term must be added to the action, in order to account for the Jacobian in the change of integration variables: $U_\mu \to u_\mu$:

$$S_M^L = \frac{g^2 N_c}{12}a^2 \sum_x \sum_\mu \mathrm{Tr}\left(u_\mu(x+a\hat{\mu}/2)^2\right) + \mathcal{O}(g^4). \tag{18}$$

It is important to assign unique dummy indices and momenta to the different powers of the fields; this is the only part out of many that is straightforward by hand, but not in an automated computer evaluation. We write V in the form

$$\begin{aligned} V = {} & C_1\left(L_1(E_1 + E_2 + \ldots) + L_2(E_3 + E_4 + \ldots) + \ldots\right) \\ & + C_2\left(L_3(E_5 + E_6 + \ldots) + L_4(E_7 + E_8 + \ldots) + \ldots\right) + \ldots \end{aligned} \tag{19}$$

where C_i are 'colour structures', L_i are 'Lorentz structures' and E_i are monomials in trigonometric functions of

momentum components. For instance, the 4-point vertex of gluons of QCD can be expressed as:

$$
\begin{aligned}
&V(k_1,k_2,k_3,k_4) = -g^2(2\pi)^4\delta(k_1+k_2+k_3+k_4)\mathrm{Tr}(T^{\alpha_1}T^{\alpha_2}T^{\alpha_3}T^{\alpha_4})\times \\
&\Big[\delta_{\mu_1\mu_2\mu_3\mu_4}\left(\frac{2}{3}-\frac{2}{3}\sum_\rho \cos\left(k_{1\,\rho}\right)+\frac{1}{2}\sum_\rho \cos\left(k_1+k_2\right)_\rho\right) \\
&+\delta_{\mu_1\mu_2\mu_3}\left(-\frac{4}{3}\sin\left(\frac{k_{4\,\mu_1}}{2}\right)\sin\left(\frac{k_{4\,\mu_4}}{2}\right)+2\sin\left(\frac{k_{4\,\mu_1}}{2}\right)\sin\left(\frac{(2k_1+k_4)_{\mu_4}}{2}\right)+2\sin\left(\frac{k_{4\,\mu_1}}{2}\right)\sin\left(\frac{(2k_3+k_4)_{\mu_4}}{2}\right)\right) \\
&+\delta_{\mu_1\mu_2}\delta_{\mu_3\mu_4}\left(\cos\left(\frac{(k_3+k_4)_{\mu_1}}{2}\right)\cos\left(\frac{(k_3+k_4)_{\mu_3}}{2}\right)-2\cos\left(\frac{(k_3-k_4)_{\mu_1}}{2}\right)\cos\left(\frac{(k_3+k_4)_{\mu_3}}{2}\right)\right) \\
&+\delta_{\mu_1\mu_3}\delta_{\mu_2\mu_4}\left(\cos\left(\frac{(k_1-k_3)_{\mu_2}}{2}\right)\cos\left(\frac{(k_2-k_4)_{\mu_1}}{2}\right)\right)\Big],
\end{aligned}
\tag{20}
$$

where T^a is a generator of the gauge group and ρ is an 'internal' Lorentz index, appearing in structures which break rotational symmetry on the lattice. It is essential to take advantage of the fact that all vertices are completely symmetric with respect to the interchange of external lines. We use this symmetry to simplify the expressions for the vertices in three steps:

1. Minimize the color structures, for example, transform $\sum_c f^{a_1a_3c}f^{a_2a_4c}$ to the form $\sum_c f^{a_1a_2c}f^{a_3a_4c}$ using color algebra. The software includes a function, which is called 'colorExpand' and performs color simplifications in the expression in an efficient way, without expanding subexpressions which do not contain color indices.

2. Utilize the residual symmetry within each color structure to reduce all associated Lorentz structures to their simplest forms.

3. For each color-times-Lorentz structure, use its residual symmetry to minimize the number of accompanying monomials.

It is worth mentioning that for any given theory, once we calculate the vertices, we save them, and we can subsequently use them. The argument "vertices" in the command "contract" (see Sec. 3.2) will be of the form: {{Ngluon, Nghostpair, Ngluinobar, Ngluino, Nquarkbar, Nquark, NAplusdag, NAplus, NAminusdag, NAminus}, expression}, where: Ngluon is the number of gluon legs, and similarly for all other Nxxx (dag stands for dagger), and expression is the expression for the vertex itself. Therefore, the Mathematica expression for the 4-gluon vertex is:

```
{{4, 0, 0, 0, 0, 0, 0, 0, 0, 0}
- g^2 spur[c[1],c[2],c[3],c[4]] * delta[k[1]+k[2]+k[3]+k[4]]*
    (delm[mu[1],mu[2],mu[3],mu[4]] *
        ( 2/3 -2/3 c2[2k[1],rho[1]] +1/2 c2[2k[1]+2k[2],rho[1]])
 + delm[mu[1],mu[2],mu[3]] *
        (-4/3 s2[k[4],mu[1]] s2[k[4],mu[4]] +2 s2[k[4],mu[1]] s2[2k[1]+k[4],mu[4]]
        +2 s2[k[4],mu[1]] s2[2k[3]+k[4],mu[4]])
+ delm[mu[1],mu[2]]delm[mu[3],mu[4]] *
                 (c2[k[3]+k[4],mu[1]] c2[k[3]+k[4],mu[3]]
                  -2   c2[k[3]-k[4],mu[1]] c2[k[3]+k[4],mu[3]])
+ delm[mu[1],mu[3]]delm[mu[2],mu[4]] *
                 (c2[k[1]-k[3],mu[2]] c2[k[2]-k[4],mu[1]]))}
```

The convention for assigning indices and momenta to different fields in a vertex follows the order: gluon, gluino, antiquark, quark, antisquark (+), antisquark (−), squark (+), squark (−), antighost, ghost.

The first command presented in this tutorial is how to make fermionic (quark) vertices particular for stout links[1]

---

[1] Stout smearing according to Ref. [14]: Each link $U_\mu(x) = exp(i\,g\,a\,u_\mu(x+a\hat{\mu}/2))$ is replaced by a stout link $\widetilde{U}_\mu(x)$ defined as [7]

$$\widetilde{U}_\mu(x) = e^{i\,Q_\mu(x)}\,U_\mu(x)\,, \tag{21}$$

where the definition of $Q_\mu(x)$ is

$$Q_\mu(x) = \frac{\omega}{2\,i}\left[V_\mu(x)U_\mu^\dagger(x) - U_\mu(x)V_\mu^\dagger(x) - \frac{1}{3}\mathrm{Tr}\left(V_\mu(x)U_\mu^\dagger(x) - U_\mu(x)V_\mu^\dagger(x)\right)\right]. \tag{22}$$

ω is a tunable parameter, called a stout smearing parameter, and $V_\mu(x)$ represents the sum over all staples associated with the link, $U_\mu(x)$.

in the fermionic part: $\psi(x)\widetilde{U}_\mu(x)\psi(x)$ for one smearing step, the same command contains also the Background Field (BF) technique [2]:

```
makestoutvertexU[nQ_,nB_]
```

where 'nQ' is the number of quantum gluons and 'nB' is the number of background gluons of the vertex. It also returns an expression multiplyed with the parameter 'w' that corresponds to the omega parameter (ω) of the stout step, which can be set to 0 if no stout smearing is needed.

Let us extract the lattice vertex with gluon-antiquark-quark fields. There are two contributions. The first comes from $\sum_x \sum_\mu \bar{\psi}(x)\gamma_\mu \mathcal{D}_\mu \psi(x) = \frac{1}{2}\sum_x \sum_\mu \bar{\psi}(x)\Big[U_\mu(x)\psi(x+\hat{\mu}) - U^\dagger_\mu(x-\hat{\mu})\psi(x-\hat{\mu})\Big] = \frac{1}{2}\sum_x \sum_\mu \Big[\bar{\psi}(x)U_\mu(x)\psi(x+\hat{\mu}) - \bar{\psi}(x+\hat{\mu})U^\dagger_\mu(x)\psi(x)\Big]$, where we omit powers of the lattice spacing, which can be recovered through dimensional analysis. The second contribution is coming from the Wilson term: $-\frac{r}{2}\sum_x \bar{\psi}(x)\mathcal{D}^2\psi(x) = -\frac{r}{2}\sum_x \sum_\mu \bar{\psi}(x)\Big[U_\mu(x)\psi(x+\hat{\mu}) - 2\psi(x) + U^\dagger_\mu(x-\hat{\mu})\psi(x-\hat{\mu})\Big] = -\frac{r}{2}\sum_x \sum_\mu \Big[\bar{\psi}(x)U_\mu(x)\psi(x+\hat{\mu}) - 2\bar{\psi}(x)\psi(x) + \bar{\psi}(x+\hat{\mu})U^\dagger_\mu(x)\psi(x)\Big]$.

Figure 1 shows this vertex. Using some commands, we are able to construct the vertex from this part:

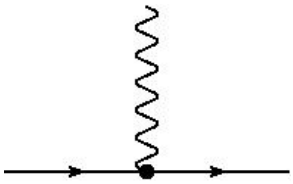


FIG. 1: Interaction lattice vertex with gluon-antiquark-quark fields. A wavy (solid) line represents gluons (quarks).

```
In[2]:=  makestoutvertexU[1,0] /. w -> 0 ;

In[3]:= 1/2 dtrace[fmu[2],mu[1],fmu[3]] * (% * phi2[2k[3], mu[1]] - (% * phi2[-2k[2], mu[1]] /.
                            im g -> -im g)) +
        (-r/2)* (% * phi2[2k[3], mu[1]] + (% * phi2[-2k[2], mu[1]] /.
                            im g -> -im g)) ;

In[4]:= % /. phi2[a_,b_]*phi2[c_,b_] :> phi2[a+c,b] ;

In[5]:= % /. delta[k[1]-k[2]+k[3]] phi2[k[1]+a_,b_] :> delta[k[1]-k[2]+k[3]] phi2[k[2]-k[3]+a,b] ;

In[6]:= % /. phi2[a_,b_] :> c2[a,b] + im s2[a,b] ;

In[7]:= (Expand[%] /. im^2 -> -1) // canonical // simplifydelm ;

In[8]:= Vertex[value] = { {1, 0, 0, 0, 1, 1, 0, 0, 0, 0}, %} ;
```

Note that $phi2[k,\mu] = \exp(ik_\mu/2)$, $c2[k,\mu] = \cos(k_\mu/2)$ and $s2[k,\mu] = \sin(k_\mu/2)$. Furthermore, in the above we also use the command 'simplifydelm' so as to simplify contractions in Lorentz indices.

```
simplifydelm[expression]
```

---

[2] On the lattice, the background field technique [6, 10] can be approached in more than one way. Different lattice actions may be chosen and the precise way in which the background field is introduced is arbitrary to some extent. However, the differences between the choices of lattice actions should be irrelevant in the continuum limit. The background field on the lattice is introduced by decomposing the gauge link variable as follows [6] (Q_μ: quantum field, B_μ: background field):

$$\begin{aligned} U_\mu(x) &= U^Q_\mu(x)U^B_\mu(x), \\ U^Q_\mu(x) &\equiv e^{ig_0 Q_\mu(x)}, \\ U^B_\mu(x) &\equiv e^{iag_0 B_\mu(x)} \end{aligned} \tag{23}$$

where $Q_\mu(x) = Q^a_\mu(x)T^a$ is the quantum field, $B_\mu(x) = B^a_\mu(x)T^a$ is the background field, and T^a are the generators of SU(N) with the convention of $\mathrm{Tr}(T^aT^b) = \delta^{ab}/2$.

The command 'canonical' processes an expression by checking the function f when it has the argument $-x$. If $f(-x)$ appears, the function applies one of the following rules: If f is even, it simplifies $f(-x)$ to $f(x)$, meaning the function's value remains the same when its argument changes sign. If f is odd, it simplifies $f(-x)$ to $-f(x)$, meaning the function's value is negated when its argument changes sign.

```
canonical[expression]
```

We also present a command for clover-like vertices in which we need to put the list of the links that connect quark and antiquark fields in the clover term of the fermion action: $\sigma_{\mu\nu}\psi(x)U_{\mu\nu}(x)\psi(x)$; this command has the following form:

```
makecloververtex[listOFlinks_, nQ_, nB_]
```

where the first argument is the list of links, and by definition, the 'listOFlinks' for $U_{\mu\nu}$ is {1,2,-1,-2}. By making the appropriate substitutions, one can obtain pure gluonic vertices; we leave this exercise to the reader to reproduce, for example, Eq. (20).

Several key interaction vertices required for our supersymmetric lattice perturbation theory framework are already implemented and stored in the software. These include both action and operator vertices incorporating stout smearing links. The main lattice action vertices in standard discretization (without stout links) are defined in `SUSY_LATTaction_vertices.m`, while vertices associated with composite operators—such as the supercurrent S_μ, the gauge-invariant operator T_μ, and gluino-glue structures—are found in `SUSY_LATToperators_vertices.m` and `SUSY_LATT_gluino_glue_vertices.m`. All of these expressions follow consistent conventions for color and Lorentz index contractions.

In order to capture the effects of non-isotropic stout smearing, we utilize dedicated routines such as `makestoutvertex_nonisotropic.m`, which automates the construction of smeared vertices based on the link structure . These routines generate analytic expressions involving gluon fields smeared via exponentials of nested Wilson loops [14] and commutator structures $Q_{\mu\rho}$, ensuring the inclusion of all relevant gauge interactions to high order in g. The stout smearing is controlled by the parameter w, which encodes the strength of the smearing applied to the gauge links.

To incorporate chiral symmetry on the lattice, our framework includes support for overlap fermions following the Ginsparg-Wilson relation. The relevant initialization and symbolic structures for the overlap action and its associated composite operators are defined in the file `SUSY_vertices_intro_overlap.m`. This file contains vertices involving overlap fermions and their supersymmetric extensions. These include the interactions of quark-antiquark-gluon and gluino-gluino-gluon vertices. The vertices defined in this file are compatible with the rest of the lattice software, and their formulation and definitions are shown in Refs. [3, 4].

3.2. Contraction among vertices

The next step in evaluating Feynman diagrams is the contraction among vertices, which is done automatically once the algebraic expressions of the vertices and the "incidence matrix", of the diagram are specified. This incidence matrix is used to systematically organize and track the contractions between pairs of field operators. This matrix is a square $n \times n$ matrix for each type of contraction where n is the number of the used vertices.

The result of the contraction is a preliminary expression for the diagram. This is followed by simplifications of the color dependence, Dirac matrices, and tensor structures. Additionally, symmetries of the theory, such as periodicity, reflection, charge conjugation, parity and hypercubic symmetry, are fully utilized to reduce the complexity and proliferation of the algebraic expressions.

Note that before contraction, we must ensure the symmetrization of the vertices and the proper renaming of indices (Lorentz, color) and momenta. We have the functions 'symmetrizeauto', 'symmetrize', 'symmetrizepartially' which are used by the function 'contract'; given a vertex which contains several identical fields (typically gluons), these functions symmetrize the vertex w.r.t. these fields (only to the extent that is needed, for the sake of economy). As a side note, we have also the 'antisymmetrize' function in order to antisymmetrize expressions over some indices. The function 'contract' applies contractions leading to a Feynman diagram and have the following arguments:

- 'vertices' which is a list of N vertices, which may contain the fields of the theory (gluons, gluinos, ghosts, quarks and/or squarks (plus/minus)).
- 'contractionsG' an $N \times N$ incidence matrix whose $(ij)^{th}$ element indicates the number of gluons joining the i^{th} to the j^{th} vertex.
- 'contractionsGh' an $N \times N$ incidence matrix whose $(ij)^{th}$ element indicates the number of contractions among antighosts from the i^{th} and ghosts from the j^{th} vertex.

- 'contractionsQ' an $N \times N$ incidence matrix whose $(ij)^{th}$ element indicates the number of contractions among antiquarks from the i^{th} and quarks from the j^{th} vertex.
- 'contractionsg' an $N \times N$ incidence matrix whose $(ij)^{th}$ element indicates the number of contractions among antigluino from the i^{th} and gluino from the j^{th} vertex.
- 'contractionsgbgb' an $N \times N$ incidence matrix whose $(ij)^{th}$ element indicates the number of contractions among antigluinos from the i^{th} and the j^{th} vertex.
- 'contractionsgg' an $N \times N$ incidence matrix whose $(ij)^{th}$ element indicates the number of contractions among gluinos from the i^{th} and the j^{th} vertex.
- 'contractionsAp' an $N \times N$ incidence matrix whose $(ij)^{th}$ element indicates the number of contractions among squark $A_+^\dagger$ from the i^{th} and A_+ from the j^{th} vertex.
- 'contractionsAm' an $N \times N$ incidence matrix whose $(ij)^{th}$ element indicates the number of contractions among squark $A_-^\dagger$ from the i^{th} and A_- from the j^{th} vertex.

All these matrices, except the incidence matrix of gluons, are not symmetric; in all these cases, the function 'contract' should be called separately for each version of the matrices. The function 'contract' must be called for each Feynman diagram.

Firstly, the function 'contract' detects how many fields are contained in each vertex and how many of them remain external after contraction. We have to note that the indices are assigned to external fields without any symmetrization; such a symmetrization (over indices of same-type external fields) will have to be performed by hand after the command 'contract' has finished. This will amount to adding together various mirror diagrams. However, the function 'contract' totally symmetrizes the vertex list.

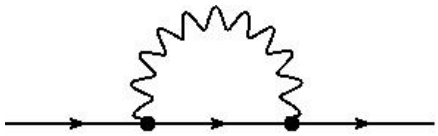


FIG. 2: One-loop Feynman diagrams contributing to the 2-pt Green's function $\langle \psi(x)\bar{\psi}(y)\rangle$.

Specifically, we assume that a vertex could contain more than one gluon or squark field and we symmetrize over them. If two vertices are connected by n lines, only one of the two need be symmetrized with respect to those lines. The aforementioned command combines all symmetrized vertices by renaming all indices (and momenta); indices assigned to contracted legs are converted to internal ones. Afterward, both internal and external indices are arranged in ascending order so as to have unique names. This command performs self-contractions of vertices, and contractions of the i^{th} and j^{th} vertex; these two operations are performed simultaneously, in order to respect the order of indices. In particular, for each contraction, we substitute the related tree-level propagator of each pair of fields.

SUPER_QUANTA features the following contract command that automates the process of applying Wick's theorem. It involves contracting pairs of fields and replacing them with propagators. It has the following form:

```
contract[vertices_, contractionsG_, contractionsGh_, contractionsg_,
         contractionsgbgb_, contractionsgg_, contractionsQ_,
  contractionsAp_, contractionsAm_]
```

Taking the part of the total vertex that we created earlier (Vertex[value]), we apply Wick's theorem to generate our first diagram, as shown in Fig. 2.

```
In[9]:= contract[{Vertex[value],Vertex[value]},
                 {{0,1},{1,0}},
                 {{0,0},{0,0}},
                 {{0,0},{0,0}},{{0,0},{0,0}},{{0,0},{0,0}},
                 {{0,1},{0,0}},
                 {{0,0},{0,0}},{{0,0},{0,0}}] ;
```

The final expression of the Green's function will depend on (after the integration of the loop momentum):

- q[i] (external momenta),
- nu[i] (Lorentz index),

- a[i] (color adjoint index),
- fnu[i] (Dirac index),
- af[i] (color fundamental index),

where the first values of the index 'i' refer to external gluons, the next values to external antigluinos, and so on, in the order: {gluon, antigluino, gluino, antighost, ghost, antiquark, quark, $A^{\dagger}_{+}$, A_{+}, $A^{\dagger}_{-}$, A_{-}}, i=1,2,...,n (n: number of external fields). We have to take into account that in case the vertices of the Green's function contain "open" indices, these indices must be given values which do not clash with the values assigned to external fields by "contract". Note that combinatorial factors, or extra factors of (-1) for fermion closed loops, are not included in this function.

The expression of Green's functions may also depend on:

- p[i] (internal loop momenta)

and on

- rho[i] (Lorentz index with a summation).

3.3. Simplifications of the color dependence

We can simplify color structures by using the identity:

$$T^a_{ij}T^a_{kl} = \frac{1}{2}\delta_{il}\delta_{kj} - \frac{1}{2N}\delta_{ij}\delta_{kl}, \tag{24}$$

which is valid for $SU(N)$. Using this identity, all internal color indices are completely eliminated. The algorithms used to implement this simplification are the same as those employed in the continuum [5]. The function 'colorExpand' takes an expression as an argument and performs color simplifications on it, without expanding subexpressions which do not contain color indices.

```
In[10]:=  (% /. openspur -> productGen /. spur -> traceGen) // colorExpand ;

In[11]:= % //. im^i_ :> -im^(i-2) ;
```

Another simplification can involve solving the delta function, which can be done as shown below.

```
In[12]:= Variables[%] // Sort

In[13]:= (#/(# /. delta[__]->1 ))& /@ List @@ %% // Union

In[14]:= simplifydelta[Expand[%%%],p[2]] // canonical;

In[15]:= (#/(# /. delta[__]->1 ))& /@ List @@ % // Union

In[16]:= replace[Expand[%%], {q[2],q[1]}] /. delta[0] -> 1 ;

In[17]:= canonical[%];

In[18]:= Variables[%] // Sort
```

The following command was used:

```
replace[expresion, {a_, b_}]
```

to replace occurrences of a specific part of an expression (a) with another expression (b).

It is useful to observe the Dirac indices:

```
In[19]:= (#/(# /. dtrace[__]->1 /. propF[___] -> 1))& /@ List @@ %% // Union
```

At this point, we can replace the expression for all the tree-level Wilson propagators appearing in the diagrams. For completeness, we provide the propagators corresponding to the action in Eq. (7).

$$\begin{aligned}
&\text{Quark propagator}: \quad \frac{1}{i\mathring{\not{q}} + \frac{2r}{a}\sum_\mu \sin^2(aq_\mu/2) - m}, \qquad \text{where}: \quad \mathring{\not{q}} = \frac{1}{a}\sum_\mu \gamma_\mu \sin(aq_\mu) \\
&\text{Gluon Propagator}: \quad \frac{1}{\hat{q}^2}\left(\delta_{\mu\nu} - (1-\alpha)\frac{\hat{q}_\mu \hat{q}_\nu}{\hat{q}^2}\right), \qquad \text{where}: \quad \hat{q}_\mu = \frac{2}{a}\sin\frac{aq_\mu}{2}\,, \quad \hat{q}^2 = \sum_\mu \hat{q}_\mu^2 \\
&\text{Ghost propagator}: \quad \frac{1}{\hat{q}^2} \\
&\text{Squark propagator}: \quad \frac{1}{\hat{q}^2 + m^2} \\
&\text{Gluino propagator}: \quad \frac{2}{i\mathring{\not{q}} + \frac{2r}{a}\sum_\mu \sin^2(aq_\mu/2)}
\end{aligned} \tag{25}$$

In the following equations we present the Mathematica expressions for these tree-level propagators, where we define the gauge parameter $\beta = 1 - \alpha$:

```
In[20]:= %%%  /.
propF[a_, b_, c_, d_] :>
fhat[a,m] ((-m + 2 r s2sq[a]) dtrace[c, b]  - im dtrace[c, d, b] s2[2 a, d]) /.
prop[a_, b_, c_] :> hat2[a] (delm[b, c] - 4 beta hat2[a] s2[a, b] s2[a, c]) /.
propG[a_] :> hat2[a] /.
propA[a_] :> hat2[a, m] /.
propg[a_, b_, c_, d_] :> 2 fhat[a] ( 2 r s2sq[a] dtrace[c, b] - im dtrace[c, d, b] s2[2 a, d]) ;
```

As an exercise, the definitions of s2sq[a], hat2[a], hat2[a,m] can be derived by comparing the expressions above with Eq. (25). You can also see these definitions explicitly in Sections 3.4 and 3.5.

3.4. Gamma matrix algebra

In this section, we mainly describe how to simplify the Dirac matrices. We have the function 'traceDirac[a,b,...,c,d]' which is the trace of a product of Dirac matrices. The arguments of traceDirac can be: gamma5, nu[_] (an external Lorentz index taking values: 1,2,3,4), rho[_] (a dummy Lorentz index taking values: 1,2,3,4,...,nDim) or X, Y (unspecified Dirac matrices; each may appear at most once). For example: traceDirac[gamma5,nu[3],X,rho[2],gamma5,Y] = $\mathrm{Tr}(\gamma_5\,\gamma_{\nu_3}\,\mathrm{X}\,\gamma_{\rho_2}\,\gamma_5\,\mathrm{Y})$. There is another function called 'productDirac[i,a,b,...,c,d,j]' with the same possible arguments as 'traceDirac[a,b,...,c,d]'. The function 'productDirac[i,a,b,...,c,d,j]' is a product of Dirac matrices. For example: productDirac[i,gamma5,nu[3],X,rho[2],gamma5,Y,j] = $(\gamma_5\,\gamma_{\nu_3}\,\mathrm{X}\,\gamma_{\rho_2}\,\gamma_5\,\mathrm{Y})_{ij}$.

```
In[21]:= Expand[%] //. dtrace[a__, i_]*dtrace[i_, b__] :> dtrace[a, b] ;

In[22]:= (#/(# /. dtrace[__]->1 ))& /@ List @@ % // Union

In[23]:= Expand[%%] /.
        {dtrace -> productDirac,
         gtrace -> traceDirac} ;
```

There are the following substitutions in order to simplify the gamma matrix algebra:

- productDiracRules4dim is a set of substitution rules which may be applied repeatedly in order to simplify products of Dirac matrices in 4 dimensions.
- productDiracRulesDdim is a set of substitution rules which may be applied repeatedly in order to simplify products of Dirac matrices in nDim = 4-2e dimensions.
- traceDiracRules4dim is a set of substitution rules which may be applied repeatedly in order to simplify traces of Dirac matrices in 4 dimensions.

- traceDiracRulesDdim is a set of substitution rules which may be applied repeatedly in order to simplify traces of Dirac matrices in nDim = 4-2e dimensions.

```
In[24]:= % //. productDiracRules4dim  //. traceDiracRules4dim;

In[25]:= (Expand[%] // simplifydelm) // reducerho // rorder ;
```

The commands 'reducerho' ('rorderall') and 'rorder' do the reordering, in the momenta, given the symmetry properties, and in the rho indices, respectively. The function 'reducerho' reduces rho indices after applying the 'simplifydelm' function.

There is also the function 'reducegammaproductDirac' ('reducegammatraceDirac') which operates on a sum of terms, each of which is a product. For each term, which may contain up to one "productDirac" ("traceDirac"), it checks all pairs of consecutive indices of gamma matrices: If the rest of the term is symmetric under the interchange of a pair of two consecutive indices, then the term gets multiplied by a Kronecker delta of these two indices.

```
In[26]:= % // reducegammaproductDirac // reducegammatraceDirac;

In[27]:= lb[%]

In[28]:= Expand[%%] //. productDiracRules4dim  //. traceDiracRules4dim;

In[29]:= (Expand[%] // simplifydelm) //reducerho // rorder ;

In[30]:= Variables[%] // Sort

In[31]:= bl[%]

In[32]:= (Expand[%%%] // simplifydelm // reducerho) //.
productDiracRules4dim  //. traceDiracRules4dim // simplifydelm // reducerho;

In[33]:= {lb[%], Variables[%] // Sort}
```

The function 'lb' takes an expression as an argument and gives the length and the number of bytes of the expression. Similarly, the function 'bl' takes a list as an argument and gives the length and the number of bytes of the list.

In case you need to collect the expression with respect to specific variables, you can use the following:

```
In[34]:= collect[%%, {fhat[_,_], hat2[_]}]
```

The function 'collect' depends on the list given as the last argument. For example, one can collect all of the powers of the gauge coupling, g, the lattice spacing, a, and the number of colors of the theory, Nc. It also collects delm[ ], epsilon[ ], hat2[ ], etc. Note that the list of elements to collect may contain also Heads of variables as well as patterns, e.g. collect[...,g,s2[q[_],_],c2,...].

Note that Mathematica also has a command named Collect which can be used as follows:

```
Collect[expression, {fhat[_,_], hat2[_]} ]
```

or even better:

```
Collect[expression, {fhat[_,_], hat2[_]}, Factor ]
```

3.5. Trigonometric manipulations

In this section, we give details on how the trigonometric simplifications are systematically applied to ensure all terms are in canonical form, making identifications and cancellations straightforward and automatic.

We use the following trigonometric notation:

- s2[a_,b_] = Sin[a[b]/2]
- c2[a_,b_] = Cos[a[b]/2]
- s2sq[a_] = $\sum_\rho$ Sin[a[ρ]/2]^2
- sisq[a_] = $\sum_\rho$ Sin[a[ρ]]^2

- s2qu[a_] = $\sum_\rho$ Sin[a[ρ]/2]^4
- phi2[a_,b_] = exp[$I * a[b]/2$], where I is the imaginary unit
- s2overs2[d_,a_,b_] = Sin[d*a[b]/2] / Sin[a[b]/2]
- s2sqdiff[a_,b_] = s2sq[a+b] - s2sq[b]
- hat2[p_] = 1/4 / Sum[Sin[p[i]/2]^2, {i,4}]
- fhat[p_,mass_] = 1 / (Sum[Sin[p[i]]^2,{i,4}] + (mass + 2r Sum[Sin[p[i]/2]^2,{i,4}])^2), where r is the Wilson parameter
- fhat[p_] = fhat[p,0]
- m[p_] = mass + 2r Sum[Sin[p[i]/2]^2,{i,4}], where 'mass' refers to the fermion mass. Note that the sign of the mass term differs from that in the quark propagator in Eq. (25).

For calculations in dimensional regularization, we use some of these symbols with different meanings. For example:

- s2[2 p_, b_] represents p_b,
- sisq[p_] represents p^2,
- hat2[p_] represents $\frac{1}{p^2}$.

When calculating Feynman diagrams, it is often convenient to perform a shift in the loop momentum, such as $p \to -p - q$, where p is the loop momentum and q is some external momentum. This shift is particularly useful when simplifying integrals over loop momenta or when aligning terms in the integrand to match a standard form. The integration measure d^4p remains invariant under such linear transformations, ensuring the validity of the shift. These shifts are most convenient in cases where the loop integrals become easier to evaluate after the redefinition of the loop variables, or when symmetries in the diagram allow for further simplifications. Such momentum shifts do not affect the physical content of the calculation, provided that regularization is handled consistently in the case of divergent integrals.

In the software package, there is a function called 'count[expr_,a_]', acting on an expression 'expr' representing a product and it counts the factors in the expression which match the pattern 'a'. If a factor matches 'a^i', it will contribute 'i' to the total count. If 'expr' is not a product, then the result is 1 if 'expr' matches 'a' as a whole, else it will be 0. An extension of this function is 'countlist[a_, b_]'; given a list 'b' of n doublets (where the first member of the doublet is a pattern, and the second member is a score): 'countlist[a_,b_]' provides a total score for product 'a', depending on the number of factors which match the patterns in 'b': countlist[a_, b_] = count[a,b[[1,1]]*b[[1,2]]] + ... + count[a,b[[n,1]]*b[[n,2]]]. If b is not a list, then countlist[a_,b_] = count[a,b]. Another useful function is 'hattosine', which rewrites hats in terms of sine and cosine functions.

It is convenient to define two lists to track the superficial degree of divergences (list1) and the overall powers (list2) of the lattice spacing:

```
In[35]:= list1 = {{m,1},{s2[_,_],1},{hat2[__],-2},{prop[__],-2},

{fhat[__],-2},{s2sq[_],2},{sisq[_],2},{s2qu[_],4}} ;

In[36]:= list2 = {{m,1},{s2[a_. q[1],_],1}} ;
```

The two lists can be used in combination with the function countlist, as follows:

```
In[37]:= {countlist[#,list1], countlist[#,list2]}& /@ List @@ Expand[%%%] // Union
```

The diagram in Fig. 2 contributes to the inverse fermion propagator, computed at one-loop order in perturbation theory. It is important to note that, for dimensional reasons, a global prefactor of $\frac{1}{a}$ multiplies our expressions for the inverse propagator. Consequently, the $\mathcal{O}(a^0)$ correction is obtained by including all terms up to $\mathcal{O}(a^1)$.

There are also functions which can be applied to an expression in order to simplify cosines; the Head of the expression must be Plus in order to maximize the instances in which such simplifications will take place. For example, 'replacec2All[expr]' acts on expression 'expr', replacing cosines by use of (for explanatory reasons we do not keep the second argument for the Lorentz index):

$$c2(x) = 1 - 2\, s2(x/2)^2$$
$$c2(x)^2 = 1 - s2(x)^2,$$

provided that arguments of trigonometric functions remain integer multiples of momenta. The function 'replacec2q[expr]' acts on expression 'expr', replacing cosine terms using the following rule (for explanatory purposes, we omit the second argument for the Lorentz index), keeping up to $O(q^2)$ terms, by use of:

$$c2(aq) = 1 - 2\,a^2\,s2(q)^2 + O(q^4)$$
$$c2(aq)^2 = 1 - a^2\,s2(q)^2 + O(q^4).$$

Terms of $O(q^4)$ are ignored. The function 'replacec2qexact[expr]' acts on expression 'expr', replacing cosines of (a multiple of) external momentum q by use of the exact formulae:

$$c2(aq)^2 = 1 - s2(aq)^2$$
$$c2(aq) = 1 - 2\,s2(aq/2)^2.$$

A similar function is used for the internal momenta p, called 'replacec2p[expr]'.

In all these cases, implicit summation over the directions of the arguments of the cosines is considered. Furthermore, the expression 'expr' must have Head Plus to ensure maximum efficiency.

The function 'makes2sq' takes a typical integrand: prefactor * (sum of terms) and substitutes, wherever legitimate, s2[a_,rho[_]]^2 by s2sq[a], s2[2a_,rho[_]]^2 by sisq[a] and s2[a_,rho[_]]^4 by s2qu[a]. This reduces the number of rho indices, thus moderating the increase in the size of the expression once the rho's are made to be different. The propagator momenta must be p[1],p[2],p[3],p[1]+p[2],p[1]+p[3],p[3]-p[2] in order for this command to make the maximum amount of substitutions. To use the aforementioned function, one simply says (before rendering rho's different):

```
In[38]:= makes2sq[%%%%];

In[39]:= reducerho[%];

In[40]:= rorder[%];
```

The function can also be used for the external momenta, in the following naive way (given that p[2] does not appear in the expression):

```
makes2sq[Expand[%%] /. q[1]->p[2]] /. p[2] -> q[1] ;
Expand[%] // simplifydelm // reducerho // rorder ;
```

3.6. Extraction of the dependence on external momenta

Determining the precise analytic momentum dependence of an n-point function as $a \to 0$ is one of the most challenging tasks, both conceptually and algorithmically.

The above simplifications are followed by the extraction of all forms of functional dependence on the external momentum q (logarithmically divergent, Lorentz non-invariant, polynomial terms) and the lattice spacing (terms of order a^0, a^1, a^2, and possibly $\ln a$).

As a first task we want to reduce the number of infrared divergent integrals to a minimal set. To do this, we use two kinds of subtractions among the propagators, using the simple equalities:

$$\frac{1}{\tilde{k}^2} \;=\; \frac{1}{\hat{k}^2} + \left\{ \frac{4\sum_\mu \sin^4(k_\mu/2) - 4\left(\sum_\mu \sin^2(k_\mu/2)\right)^2}{\tilde{k}^2\,\hat{k}^2} \right\} \tag{26}$$

$$\begin{aligned} D(k) \;&=\; D_{plaq}(k) + \Big\{ D(k) - D_{plaq}(k) \Big\} \\ &=\; D_{plaq}(k) + D_{plaq}(k) \Big\{ D^{-1}_{plaq}(k) - D^{-1}(k) \Big\} D(k) \end{aligned} \tag{27}$$

where k stands for p or $p + a\,q$, and p (q) is the loop (external) momentum.

```
prop[p[i_],b_,c_]:> dprop[p[i],b,c] + delm[b,c] hat2[p[i]] ;
```

The denominator of the fermion propagator, $\tilde{k}^2$, is defined as

$$\tilde{k}^2 = \sum_\mu \sin^2(k_\mu) + \left(m_f + \frac{r}{2}\hat{k}^2 \right)^2, \quad \hat{k}^2 = 4\sum_\mu \sin^2(\frac{k_\mu}{2}) \tag{28}$$

For the present work, one sets $m_f = 0$ and $r = 1$, as used in Eq. (26); D is the 4×4 Symanzik gluon propagator; the expression for the matrix $\left(D^{-1}_{plaq}(k) - D^{-1}(k)\right)$, which is $\mathcal{O}(k^4)$, is independent of the gauge parameter, β, and it can be easily obtained in closed form. Moreover, we have

$$\left(D_{plaq}(k)\right)_{\mu\nu} = \frac{\delta_{\mu\nu}}{\hat{k}^2} - \beta \frac{\hat{k}_\mu \, \hat{k}_\nu}{(\hat{k}^2)^2} \tag{29}$$

Terms in curly brackets of Eqs. (26) and (27) are less IR divergent than their unsubtracted counterparts, by two powers in the momentum. These subtractions are performed iteratively until all primitively divergent integrals (initially depending on the fermion and the Symanzik propagator) are expressed in terms of the Wilson gluon propagator.

Having reduced the number of distinct divergent integrals down to a minimum, the most laborious task is the computation of these integrals, which is performed in a noninteger number of dimensions $D > 4$. Ultraviolet divergences are explicitly isolated à la Zimmermann and evaluated as in the continuum. The remainders are D-dimensional, parameter-free, zero external momentum lattice integrals which can be recast in terms of Bessel functions, and finally expressed as sums of a pole part plus numerical constants. We analytically evaluate an extensive basis of superficially divergent loop integrals, listed in Eqs. (C1) - (C10) of Appendix B; a few of these were calculated in Ref. [16]. The integrals of Eqs. (C1), (C2), (C3), are the most demanding ones in the list; they must be evaluated to two further orders in a, beyond the order at which an IR divergence initially sets in. As a consequence, their evaluation requires going to $D > 6$ dimensions. Fortunately, they are a sufficient basis for all massless integrals which can appear in any $\mathcal{O}(a^2)$ one-loop calculation; that is, any such computation can be recast in terms of (C1), (C2), (C3), plus other integrals which are more readily handled. A correct way to evaluate (C1), (C2), (C3) has not been presented previously in the literature, despite their central role in $\mathcal{O}(a^2)$ calculations, and this has prevented one-loop computations to $\mathcal{O}(a^2)$ thus far. The calculation of such an integral is given in detail in the next section.

Terms which are IR convergent can be treated by Taylor expanding in aq to the desired order. Alternatively, the extraction of the aq dependence may be performed using iterative subtractions of the form

$$f(p + a\,q) = f(p) + \Big[f(p + a\,q) - f(p) \Big] \tag{30}$$

In the following exact substitutions we perform a different type of subtraction, adding and subtracting the term $\frac{1}{\hat{k}^2 + m^2}$.

```
If[countlist[#, list1] <= -3,
  (# /. c_ fhat[a_, b_]^(i_.):> c fhat[a, b]^(i - 1) hat2[a, b] +
      c fhat[a, b]^i hat2[a, b] * (4 s2qu[a] - 4 s2sq[a]^2 + 4 r b s2sq[a])
  ), #] & /@ Expand[%] ;
```

Similarly, with the term $\frac{1}{\hat{k}^2}$:

```
If[countlist[#, list1] <= -3,
  (# /. c_ fhat[a_]^(i_.):> c fhat[a]^(i - 1) hat2[a] +
      c fhat[a]^i hat2[a] * (4 s2qu[a] - 4 s2sq[a]^2)
  ), #] & /@ Expand[%] ;
```

Eq. 30 leads to exact relations such as the following ones

$$\begin{aligned} \frac{1}{\widetilde{p + a\,q}^{\,2}} &= \frac{1}{\tilde{p}^2} - \frac{\sum_\mu \sin(2p_\mu + a\,q_\mu)\sin(a\,q_\mu)}{\widetilde{p + a\,q}^{\,2}\,\tilde{p}^2} \\ &- \frac{\sum_\mu \sin(p_\mu + \frac{a\,q_\mu}{2})\sin(\frac{a\,q_\mu}{2})\left(\hat{p}^2 + \widehat{p + a\,q}^{\,2}\right)}{\widetilde{p + a\,q}^{\,2}\,\tilde{p}^2} \end{aligned} \tag{31}$$

$$\frac{1}{\widehat{p + a\,q}^{\,2}} = \frac{1}{\hat{p}^2} - \frac{4\sum_\mu \sin(p_\mu + \frac{a\,q_\mu}{2})\sin(\frac{a\,q_\mu}{2})}{\widehat{p + a\,q}^{\,2}\,\hat{p}^2} \tag{32}$$

In these relations the exact $a\,q$ dependence of the remainders is under full control; this type of subtraction is especially useful when applied to the Symanzik propagator.

When applicable, we can use expansions of the cosine terms. Furthermore, we may also expand certain sines avoiding IR divergences. Below is the commands for this process:

```
ExpandAll[%] //. s2[a_, b_]^i_. c2[a_, b_]^j_. :> s2[a, b]^(i-1) c2[a, b]^(j-1) * (1/2) * s2[2a, b] ;

Expand[%] /. c_ sisq[q[1] + p[1]]^(i_.) -> c * sisq[q[1] + p[1]]^(i-1) *
s2[2q[1] + 2p[1], rho[21]]^2 ;

Expand[% //. s2[a_ + b_, c_] :> s2[a, c] * c2[b, c] + c2[a, c] * s2[b, c]] ;

Expand[% //. c2[a_ + b_, c_] :> c2[a, c] * c2[b, c] - s2[a, c] * s2[b, c]] ;

Expand[%] // reducerho // rorder ;
```

All cosine functions that depend solely on q can be replaced exactly with equivalent expressions involving sine functions. This can be achieved by repeatedly using the 'replacec2exact' command.

```
replacec2exact[replacec2exact[Expand[%]]] ;
```

To prevent an explosion in the number of terms, one can omit the terms that are convergent and of higher order than the required power of the lattice spacing.

```
If[(countlist[#, list2] > 0) && (countlist[#, list1] > -3), 0, #] & /@ Expand[%] ;
```

Furthermore, substitutions for the cosine of the loop momentum p can be made to express the integrals in a form suitable for extracting the divergent part of the expressions.

```
If[countlist[#, list1] <= -3, replacec2p[#], #] & /@ Expand[%] ;
```

These commands can be followed by simplifications such as:

```
makes2sq[Expand[%] /. q[1] -> p[2]] /. p[2] -> q[1];
Expand[%] // simplifydelm // reducerho // rorder;
% //. s2sq[a_]^i_. hat2[a_]^j_. :> (1/4) * s2sq[a]^(i-1) * hat2[a]^(j - 1) ;
{Length[%], Variables[%]}
```

After the divergent terms have been brought to the appropriate form, the expression can be separated into two parts: the divergent and the convergent terms. This separation is achieved by evaluating the superficial degree of divergence.

```
(* Extract the divergent terms (superficial divergence number <= -4) *)
divergent = If[countlist[#, list1] <= -3, #, 0] & /@ Expand[%%] ;
{Length[%], Variables[%]}

(* Extract the convergent terms (superficial divergence number > -4) *)
convergent = If[countlist[#, list1] > -3, #, 0] & /@ Expand[%%%%] ;
{Length[%], Variables[%]}
```

For the convenience of the user, we have created a new command, 'subtractionIRdiv', to automate this procedure. This command is specifically designed for use with one-loop expressions. In particular, it simplifies one-loop expressions by isolating infrared (IR) divergent terms. It takes as input the full expression and the power of the powers of the lattice spacing (n) that we want to achieve. The function processes the input to separate and tag divergent terms with "div", while finite (convergent) terms remain untagged.

The command works only for mass independent renormalization schemes, so that $m = 0$. This simplifies the algebraic expressions, but at the same time requires special treatment when it comes to IR singularities.

```
In[41]:= Expand[%] /. m -> 0 /. hat2[a_,0] :> hat2[a] /. fhat[b_,0] -> fhat[b] ;

In[42]:= subtractionIRdiv[Expand[%],1] ;

In[43]:= convergent = % /. div -> 0 ;

In[44]:= divergent = %% - % ;
```

The divergent integrals can be replaced by the expressions included in this software by following the commands shown below:

```
In[45]:= {countlist[#, list1], countlist[#, list2]} & /@ List @@ divergent // Union

In[46]:= {Length[%%], Variables[%%]}

In[47]:= (#/(# /. hat2[__]->1 /. fhat[__] -> 1 /. s2[__]->1 /. c2[__]->1 /.
sisq[__] ->1 /. s2sq[__] -> 1 /. s2qu[__] -> 1 ))& /@ List @@
ExpandAll[divergent] // (reducerho /@ #)& // (rorder /@ #)& // Union

In[48]:= IntegralsResults

In[49]:=  DIVpart = divergent /. IntegralsResults /.  Log[q1^2] -> Log[q[1]^2];

In[50]:= {Length[%], Variables[%]}

In[51]:= Save["MyFirstDiagram.m", DIVpart]
```

For a more complete list of primitive divergent integrals, including contributions from massive propagators, we refer to the substitution rules given below. As an illustrative example, the last entry is:

```
IntegralsRules[[-1]]

hat2[p[1] + q[1], M]^3 s2sq[p[1]] ->
0.00521172353266 - Log[M^2]/(64Pi^2) + q[1]^2/(128M^2Pi^2)
```

For the convergent part, we can manipulate it using a naive Taylor expansion. Additionally, for these convergent terms, we can proceed to render rho's different between the external momentum q[1] and the loop momentum p[1]. The function 'matchindices' takes an integrand with propagators independent of external momenta, and matches the directions of sines, so as to lead to an even integrand under p[i] → -p[i]. Aside from a prefactor, the numerator is a sum of terms, where each term may be a function of p's times a function of external momenta. There is also a function, which is called 'matchsemiperiodic' and takes an integrand over p[1] whose denominators are invariant under: $p[1]_\mu \to p[1]_\mu + \pi$ (where μ is any particular direction), and whose numerators have p-dependent parts made up exclusively of s2[2p[1] + a_., b_] and/or c2[2p[1] + a_., b_] and matches the directions of sines and/or cosines, so as to lead to an even integrand under $p[1]_\mu \to p[1]_\mu + \pi$.

In some cases, for the convergent terms, we can set q[1]=0, or we can Taylor expand the expression to the appropriate powers. In the first case:

```
In[52]:= {countlist[#, list1], countlist[#, list2]} & /@ List @@ convergent // Union

In[53]:= {Length[convergent], Variables[convergent]}
```

In order to carefully extract the external momentum, we must meticulously apply the following procedure. However, in many cases, we can set q to zero.

```
replace[ExpandAll[convergent], {q[1],0}] ;
{Length[%], Variables[%]}

In[54]:= {count[#,s2sq[q[1]+p[1]]],count[#,s2qu[q[1]+p[1]]]}& /@ List @@ convergent // Union

In[55]:= convergent /. s2sq[a_+b_]^i_. :> s2sq[a+b]^(i-1)*s2[a+b,rho[20]]^2 ;

In[56]:= % /. s2sq[a_+b_]^i_. :> s2sq[a+b]^(i-1)*s2[a+b,rho[21]]^2 ;

In[57]:= % /. s2qu[a_+b_]^i_. :> s2qu[a+b]^(i-1)*s2[a+b,rho[26]]^4 ;

In[58]:= Select[Variables[%],FreeQ[#,productDirac]&]

In[59]:= Module[{expr = ExpandAll[%%], exprtemp = 0, sum = 0, i=1},
       While[i<=Length[expr],
             exprtemp = Take[expr,{i,Min[i+99,Length[expr]]}] ;
      exprtemp = exprtemp //. s2[a_+ b_,c_] :> s2[a,c] c2[b,c] + c2[a,c] s2[b,c] // Expand ;
      exprtemp = exprtemp //. c2[a_ + b_,c_]:> c2[a,c] c2[b,c] - s2[a,c] s2[b,c] // Expand ;
             exprtemp = Select[exprtemp,(countlist[#,list2] <= 1)&];
```

```
            exprtemp = If[countlist[#,list2]>=1,simplifydelm[# /.c2[a_.q[1],b_]:>delm[b,b]],#]& /@ expr
            exprtemp = FixedPoint[replacec2exact, exprtemp];
            sum = sum + Select[exprtemp,(countlist[#,list2] <= 1)&];
            Print["{i=",i,", length[sum]=", Length[sum],"}"];
            i = i + 100]; sum];

In[60]:= {Length[%], Variables[%]}
```

In the convergent terms, when the appropriate powers of the external momentum are present in the remaining expression, we can set q to zero:

```
In[61]:= If[countlist[#, list2] == 1, canonical[#/. {hat2[a_ + b_. q[1], c___] :> hat2[a],

hat2[a_, c__] :> hat2[a], fhat[a_ + b_. q[1], c__] :> fhat[a], fhat[a_, c__] :> fhat[a],

s2[a_ + b_. q[1], c_] :> s2[a, c], s2qu[a_ + q[1]] :> s2qu[a], s2sq[a_ + q[1]] :> s2sq[a]}],#] & /@ %%;

In[62]:= {Length[%], Variables[%]}

In[63]:= rorder[reducerho[%%]];

In[64]:= Length[%]
```

For completeness, we provide the expressions for the massive propagators:

```
In[65]:= If[countlist[#,list2] < 1, # /. fhat[a_+q[1],m]^i_. :>
fhat[a+q[1],m]^(i-1) (fhat[a]-
(s2[4a+2q[1],rho[20]] s2[2q[1],rho[20]] + m^2 + 4 m r s2sq[a+q[1]]+
  4r^2 s2[2a+q[1],rho[20]] s2[q[1],rho[20]] *(s2sq[a]+s2sq[a+q[1]]) )*fhat[a]*fhat[a+q[1],m]) //
Expand // reducerho, #]& /@ Expand[%%];

In[66]:= {ByteCount[%],Length[%]}

In[67]:= If[countlist[#,list2] < 1, # /. hat2[a_+q[1],m]^i_. :>
hat2[a+q[1],m]^(i-1) (hat2[a]-
(4s2[q[1],rho[20]]*s2[2a+q[1],rho[20]] + m^2 )*hat2[a]*hat2[a+q[1],m]) //
Expand // reducerho, #]& /@ %% ;

In[68]:= {ByteCount[%],Length[%]}

In[69]:= If[countlist[#,list2] < 1, # /. fhat[a_+q[1]]^i_. :>
fhat[a+q[1]]^(i-1) (fhat[a]-
(s2[4a+2q[1],rho[20]] s2[2q[1],rho[20]] +
  4r^2 s2[2a+q[1],rho[20]] s2[q[1],rho[20]] *(s2sq[a]+s2sq[a+q[1]]) )*fhat[a]*fhat[a+q[1]]) //
Expand // reducerho, #]& /@ %%;

In[70]:= {ByteCount[%],Length[%]}

In[71]:= If[countlist[#,list2] < 1, # /. hat2[a_+q[1]]^i_. :>
hat2[a+q[1]]^(i-1) (hat2[a]-
(4s2[q[1],rho[20]]*s2[2a+q[1],rho[20]])*hat2[a]*hat2[a+q[1]]) //
Expand // reducerho, #]& /@ %% ;

In[72]:= {ByteCount[%],Length[%]}

In[73]:= If[countlist[#,list2] < 1, # /. hat2[a_+q[1]]^i_. :>
hat2[a+q[1]]^(i-1) (hat2[a]-
(4s2[q[1],rho[20]]*s2[2a+q[1],rho[20]])*hat2[a]*hat2[a+q[1]]) //
Expand // reducerho, #]& /@ %% ;

In[74]:= {ByteCount[%],Length[%]}
```

```
In[75]:= Variables[%%] // Sort

In[76]:= If[countlist[#,list2] == 1 , canonical[#/.
{hat2[a_+b_. q[1],c___]:>hat2[a],
 hat2[a_,c__]:>hat2[a], fhat[a_+b_. q[1],c__]:>fhat[a],
 fhat[a_,c__]:>fhat[a], s2[a_+b_. q[1],c_]:>s2[a,c],
 hat2[a_+b_. q[1]]:>hat2[a],fhat[a_+b_. q[1]]:>fhat[a],
 s2qu[a_+q[1]]:>s2qu[a], s2sq[a_+q[1]]:>s2sq[a]}], #]& /@ %%% ;

In[77]:= {Length[%], Variables[%]}
```

At any time, we can verify the term count in our expression.

```
In[78]:= {countlist[#, list1], countlist[#, list2]} & /@ List @@ %% // Union
```

Since q is now separate from p, we can use the function 'matchindices'.

```
In[79]:= matchindices[%%%];

In[80]:= FreeQ[%, matchindices]

In[81]:= simplifydelm[%%] // reducerho // rorder ;

In[82]:= {Length[%], Variables[%]}
```

Another command, 'makeindependent', can be used to simplify symbolic expressions by isolating terms, identifying common elements, and applying targeted substitutions to ensure that q and p have different Lorentz indices. This command is included in the module, where it can also be combined with some previous commands to further simplify the expression. This task is needed in order to isolate the part to be integrated and to make it independent of the external momentum components.

```
In[83]:= Module[{expr = %%, exprtemp = 0, sum = 0, i=1},
       While[i<=Length[expr],
    temp = Take[expr,{i,Min[i+99,Length[expr]]}];
    temp = ((Expand[makeindependent[temp]] // simplifydelm // Expand // reducerho // rorder ));
    Print["makeindependent done!"];
    temp = temp //. s2[a_,b_]^ii_. c2[a_,b_]^j_. :> s2[a,b]^(ii-1) c2[a,b]^(j-1) 1/2 s2[2a,b];
    temp = makes2sq[temp]//. s2sq[a_]^ii_. hat2[a_]^j_. :> 1/4 s2sq[a]^(ii-1) hat2[a]^(j-1) ;
    temp = rorder[reducerho[makes2sq[temp/.q[1]->p[2]]  /.p[2]->q[1]]] ;
    temp = temp // reducerho // rorder ;
    sum = sum + temp;
    Print["{i=",i,", length[sum]=", Length[sum],"}"];
    i = i + 100];
        sum  ];

In[84]:= Variables[%]
```

Some additional simplifications include:

```
In[85]:= %% /. s2[b_. q[1],a_] :> S2[2q[1],a] b/2  // Expand;

In[86]:= % /. S2 :> s2 ;

In[87]:= % //. im^i_ :> -im^(i-2) ;

In[88]:= {Length[%],Variables[%]}

In[89]:= %% //. productDiracRules4dim //. traceDiracRules4dim;

In[90]:= (Expand[%] // simplifydelm) //reducerho // rorder ;

In[91]:= % // reducegammaproductDirac // reducegammatraceDirac;
```

```
In[92]:= lb[%]

In[93]:= Expand[%%] //. productDiracRules4dim //. traceDiracRules4dim;

In[94]:= (Expand[%] // simplifydelm) //reducerho // rorder ;

In[95]:= Variables[%] // Sort

In[96]:= bl[%]

In[97]:= (Expand[%%%] // simplifydelm // reducerho) //.
productDiracRules4dim //. traceDiracRules4dim // simplifydelm // reducerho;

In[98]:= {lb[%], Variables[%] // Sort}
```

At this point, we can prepare the expression for numerical integration by extracting the prefactors and saving them separately. These prefactors consist of the variables that multiply the integrand.

```
In[99]:= out99 = (ExpandAll[%%] // reducerho // rorder // rorderall) /.
r -> 1 /. nDim -> 4;

In[100]:= Select[Variables[out99],(FreeQ[#,p] && FreeQ[#,cprop0] && FreeQ[#,cprop1])&]

In[101]:= ((#/(# /. Table[%[[i]] -> 1,{i,Length[%]}]))& /@ (List @@ out99)) // Union

In[102]:= varlist = Table[var[i] -> %[[i]],{i,Length[%]}]

In[103]:= (* Check *)

In[104]:= Reverse[Table[dummy %%%[[i]]-> var[i],{i,Length[%%%]}]];

In[105]:= ((dummy #)& /@ out100) /. %;

In[106]:= SameQ[% /. varlist, out99]

In[107]:= intexpr = %%;

In[108]:= % // reducerho // rorder // rorderall;

In[109]:= Variables[%]

In[110]:= Length[%%]

In[111]:= intexpr = Expand[%%%];
```

We perform the explicit summations over rho indices:

```
In[112]:= FreeQ[intexpr,rho[#]]& /@ {1,2,3,4,5,6,7,8,9,10}

In[113]:= (If[FreeQ[#,rho[1]],#,4 #]& /@ (%% + dummy)) /. dummy -> 0 ;

In[114]:= Length[%]
```

For completeness, we also provide the additional summations over rho[i], where the indices for $i > 5$ are treated the same as for $i = 5$.

```
(If[FreeQ[#,rho[2]],#, (# /. {rho[2] -> rho[1]}) + 3 #]& /@ (%% + dummy)) /. dummy -> 0 ;
Length[%]
(If[FreeQ[#,rho[3]],#, (# /. {rho[3] -> rho[1]}) + (# /. {rho[3] -> rho[2]}) + 2 #]& /@
(%% + dummy)) /. dummy -> 0 ;
Length[%]
```

```
(If[FreeQ[#,rho[4]],#, (# /. {rho[4] -> rho[1]}) + (# /. {rho[4] -> rho[2]}) +
(# /. {rho[4] -> rho[3]}) + #]& /@ (%% + dummy)) /. dummy -> 0 ;
Length[%]
(If[FreeQ[#,rho[5]],#, (# /. {rho[5] -> rho[1]}) + (# /. {rho[5] -> rho[2]}) +
(# /. {rho[5] -> rho[3]}) + (# /. {rho[5] -> rho[4]}) ]& /@ (%% + dummy)) /. dummy -> 0 ;
Length[%]
```

3.7. Numerical integration over loop momenta

Having reached this point, the only remaining task is the numerical evaluation of loop integrals that do not depend on external momenta.

Let us briefly outline some characteristics of our procedure for integration over loop momenta. Standard routines for numerical integration cannot be directly applied for a number of reasons, primarily stemming from the presence of poles in the integrands. The poles exist due to not only to the gluon massless propagators, but also the fermion propagators, given that the latter are Taylor expanded with respect to the mass, leading to massless denominators; in fact, Taylor expansion exacerbates this situation further, leading to double-pole contributions from fermion propagators.

Having converted all integrands in the form of expressions which contain at worst double poles, and are therefore integrable, we apply a naïve discretization of the Brillouin zone, in a way corresponding to a finite hypercubic lattice of size L: i.e., in the case of a two loop calculation namely for all eight components of the two momentum four-vector dimensionless integration variables p_1 and p_2, integration over the interval $[0,\, 2\pi)$ is converted into a summation over the values $p_{i,\mu} = 2\pi\, j/L$ $(j = 0, \ldots, L-1)$. Momentum values corresponding to propagator poles: $p_1 = 0$, $p_2 = 0$ and/or $p_1 \pm p_2 = 0$ are excluded from the summation. This procedure will lead to the exact result when extrapolated to an infinite lattice, $L \to \infty$; see below for a description of the extrapolation method and the estimate of its associated error. Note in passing that, for finite L, this procedure is not equivalent to finite-lattice perturbation theory, because zero modes, and their associated non-Gaussian functional integration, are simply left out. Having cast the momentum integrals in the form of eightfold nested sums, a number of essential simplifications must be applied before carrying out the summations:

• First, all trigonometric functions forming the integrands can now be preevaluated, since their arguments only take a finite, and very limited, set of values. This feature, which is not directly applicable in adaptive algorithms, reduces execution time significantly.

• Second, by virtue of a number of potential symmetries, such as reflections and changing the order of nested sums, the size of the integrands is reduced and the domain of summation is shrunk to a small hypertriangular region of the original domain. (A small complication for finite lattices is that the boundaries of this region are sets of nonzero measure, and must therefore be properly accounted for.)

• Third, typical integrands contain sums of $\sim 10^6$ terms (summands). Each summand is a product of trigonometric factors and each such factor may depend on some, or all, of the eight momentum components. A substantial optimization is obtained by organizing the summands in an inverse tree structure: that is, summands whose dependence on the innermost summation index is contained inside an identical factor are grouped together, so that the factor need be evaluated only once; the same procedure is applied iteratively to outer summations.

We have incorporated the above optimizations in a *metacode* which converts the original *Mathematica* expressions to Fortran code for the evaluation of each diagram at various values of L.

Once results are obtained for several hypercubic lattice sizes L^4 (typically up to $L = 128$ for 1-loop integrals and $L = 32$ for 2-loop integrals is sufficient), they must be extrapolated to $L \to \infty$. Extrapolation to infinite lattice size is of course a source of systematic error, indeed the only such source. To estimate this error, we proceed as follows: First, different extrapolations of our numerical results, the latter generically denoted as r_L, are performed using a broad spectrum of functional forms $f^k(L)$ (around 20 of them) of the type:

$$f^k(L) = \sum_{i,j} e^{(k)}_{i,j}\, L^{-i}\, (\ln L)^j \tag{33}$$

These forms encode the expected large-L behavior of lattice sums (see, e.g., [9]). A total of N_k coefficients $(e^{(k)}_{i,j})$ appear in the k^{th} functional form; these coefficients are determined uniquely using the results on N_k lattices of consecutive size L.

For the k^{th} such extrapolation, a deviation d_k is calculated using a range of criteria for quality of fit. One possible criterion, as an example, is the difference: $d_k = \left| f^k(L^*) - r_{L^*} \right|$, where L^* is the largest lattice size which was not used in the determination of $e^{(k)}_{i,j}$. Finally, these deviations are used to assign weights ("grades") $d_k^{-2}/(\sum_k d_k^{-2})$ to each extrapolation; averaging over all extrapolations $e^{(k)}_{0,0}$ with these weights produces a final value for r_∞, together with the error estimate. We can check the reliability of our error estimates in a number of ways: In those cases where

the result for a diagram is also known analytically, this result coincides with the numerical one within the systematic error; also, extrapolations which incorporate new data from larger lattices are compatible with previous results, again within systematic error.

After performing the summation over rho[i], we use the following commands to further reduce the number of terms. For convenience, these commands are organized into a module, where we save the variables and the integrand.

The user should also select the integrator to be used. Note that different integrators are available, for example for Wilson-like fermions and for overlap fermions. If the theory contains Wilson-like fermions, the following integrator should be used:

```
<< integratorOI1loop.m ;

In[115]:= Module[{expr = Expand[%%%%], exprtemp = 0, sum = 0, i=1},
       While[i<=Length[expr],
    exprtemp = Take[expr,{i,Min[i+1999,Length[expr]]}];
    exprtemp = Expand[exprtemp] // reducerho // rorder ;
    sum = sum + exprtemp;
    Print["{i=",i,", length[sum]=", Length[sum],"}"];
    i = i + 2000];
       sum  ];

In[116]:= intexpr = % ;

In[117]:= Save["MyFirstDiagram.m",varlist, intexpr]
```

The required numerical integrations of the algebraic expressions for the loop integrands are performed by highly optimized Fortran programs; these are generated by our Mathematica 'integrator1' routine. Each integral is expressed as a sum over the discrete Brillouin zone of finite lattices, with varying size L ($4^4 \leq L^4 \leq 128^4$ for one-loop integrals and $4^4 \leq L^4 \leq 32^4$ for two-loop integrals), and evaluated for given values of the Symanzik coefficients of the gluon action.

```
In[118]:= Module[{i=1,temp,filename,expr=intexpr},
While[i<=Length[expr], Print[i]; temp=Take[expr,{i,Min[i+39999,Length[expr]]}];
      filename = StringJoin["fortranfiles/namefile_part",ToString[(i+39999)/40000],".f"];
      integrator1[temp,1,filename];
      Close[filename]; i = i + 40000]]
```

Note that for the overlap formulation, a new command has been created. It can be used as follows:

```
<< integrator1loop_overlap.m ;

integrator1[intexpr, 1, "namefile.f"]

Close["namefile.f"];
```

After generating the Fortran files, they can be executed, and the output will provide the integral values for each lattice size. In order to do this, GCC has to be installed:

```
>> /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
>> brew install gcc
```

These values must then be extrapolated to an infinite lattice and multiplied by the appropriate variable.

```
>> gfortran @ofastG namefile_part1.f -o namefile_part1.x
>> ./namefile_part1.x < input1loop > ./outfiles/namefile_part1.out
```

Instead of the first command, we can use the following command:

```
 >> gfortran -Ofast namefile_part1.f -o namefile_part1.x
```

For overlap fermions, there are overlap parameters for both quark and gluino fields. For convenience, we provide a new file, `inputO1loop`, which inputs the values of these parameters, with ρ varying from 0.1 to 1.9 in steps of 0.1.

3.8. Extrapolation to infinite lattice size

The final part of the evaluation of Feynman diagrams is the extrapolation of the numerical results calculated in the previous sections (which apply to finite lattice size) to an infinite lattice size. This process introduces a systematic error, which is reliably estimated using a sophisticated inference technique. Drawing from our previous experience, we aim for a fractional error smaller than 10^{-8} for one-loop quantities and smaller than 10^{-4} for two-loop quantities.

For the extrapolation, we create a command 'extrapolate', which uses the variables that multiply each integral and the output file containing the results from the executed Fortran files.

When the files *namefile_parti.out* are ready, you can extrapolate each integral as follows (in our case $i = 1$):

```
In[119]:= extrapolate[Table[StringJoin["outfiles/namefile_part",ToString[i],".out"],{i,1}],varlist]
```

Here, we also provide a new version of the overlap formulation:

```
<< extrapolate0_automatic.m;
```

This must be loaded separately from the package, as it is not included in `input.m`.

```
In[120]:= % /. result[{{{a__},{b__}}}] :> Around[b] ;

In[121]:= Variables[%] // Sort

In[122]:= %% /. Around[a_,b_] :> a // Expand ;

In[123]:= Variables[%] // Sort

In[124]:= CONVpart = %% ;

In[125]:= Save["MyFirstDiagram.m", CONVpart]
```

Lastly, we add the divergent parts to the extrapolated convergent results to obtain the total expression of the diagram. We also include the combinatorial factor for each diagram (calculated by hand), multiplying it with the appropriate result.

```
In[126]:= TotalExpression = Combinatorial * (DIVpart + CONVpart) /. Combinatorial -> 1;

In[127]:=  Save["MyFirstDiagram.m", TotalExpression]

In[128]:= Quit
```

3.9. Data Storage Commands

In perturbative calculations with stout smeared links, intermediate symbolic expressions can contain millions of terms and are often reused across several downstream stages. The current storage workflow is implemented through the Mathematica package `shrinker.m`, which provides a uniform interface for native `.mx` storage and for compressed cache files.

Prerequisites

- Wolfram Language / Mathematica with `wolframscript` available.
- The package file `shrinker/shrinker.m` accessible from the current working directory or via an absolute path.

1) Load the package

If the Mathematica session starts inside the `shrinker/` directory:

```
<< shrinker.m
```

Otherwise load it explicitly using the full path:

```
Get["/.../shrinker/shrinker.m"]
```

2) Create an `.mx` cache from a source file

Single output:

```
shrink["SYM_GFSGg_lattice.m3", "SYM_GFSGg_lattice.mx", intexpr]
```

Multiple outputs stored in the same `.mx` file:

```
shrink["SYM_GFSGg_lattice.m3", "SYM_GFSGg_lattice.mx", {intexpr, varlistGFSGg}]
```

If the symbol list is omitted, all newly defined outputs from the source file are exported as an `Association`:

```
shrink["SYM_GFSGg_lattice.m3", "SYM_GFSGg_lattice.mx"]
```

By default, `shrink` performs a raw export with no simplification pipeline:

```
shrink["SYM_GFSGg_lattice.m3", "SYM_GFSGg_lattice.mx", intexpr, "RunPipeline" -> False]
```

To enable the configured simplification stages:

```
shrink["SYM_GFSGg_lattice.m3", "SYM_GFSGg_lattice.mx", intexpr, "RunPipeline" -> True]
```

3) Reload stored expressions from `.mx`

For a single stored output:

```
loadShrink["operator_stout_vertices.mx"];
Qnu1nu2result[[1]]
```

For files containing several outputs, plain `loadShrink` restores all of them into the current Mathematica session:

```
loadShrink["SYM_GFSGg_lattice.mx"];
varlistGFSGg[[1]]
```

Explicit selections are also supported:

```
loadShrink["SYM_GFSGg_lattice.mx", intexpr];
loadShrink["SYM_GFSGg_lattice.mx", {intexpr, varlistGFSGg}];
```

Definitions with arguments can also be restored directly:

```
loadShrink["SYM_contractions_intro_supercurrent_lattice.mx", {operatorGGg[value], diagram[oneloop, GGg,
operatorGGg[value]
```

4) Reconstruct Mathematica source from `.mx`

Single output:

```
unShrink["SYM_GFSGg_lattice.mx", "SYM_GFSGg_lattice_restored.m3", intexpr]
```

Multiple outputs:

```
unShrink["SYM_GFSGg_lattice.mx", "SYM_GFSGg_lattice_restored.m3", intexpr, varlistGFSGg}]
```

`unShrink` writes wrapped multi-line `InputForm`, so the restored file remains readable Mathematica source rather than a single long line.

5) Compress already loaded symbols

Default scheme (`Compress`):

```
loadShrink["SYM_GFSGg_lattice.mx"];
compress[{intexpr, varlistGFSGg}, "SYM_GFSGg_lattice_cache"]
```

Supported file-based schemes are:

- `Compress` → `.wlc`
- `GZIP` → `.gz`
- `BZIP2` → `.bz2`
- `ZSTD` → `.zst`

Examples:

```
compress[intexpr, "SYM_GFSGg_lattice_cache", "Scheme" -> "GZIP"]
compress[intexpr, "SYM_GFSGg_lattice_cache", "Scheme" -> "BZIP2"]
compress[intexpr, "SYM_GFSGg_lattice_cache", "Scheme" -> "ZSTD"]
```

6) Restore compressed caches

Single or multi-output caches can be restored directly into the Mathematica session:

```
uncompress["SYM_GFSGg_lattice_cache.wlc"];
uncompress["SYM_GFSGg_lattice_cache.gz"];
uncompress["SYM_GFSGg_lattice_cache.bz2"];
uncompress["SYM_GFSGg_lattice_cache.zst"];
```

After restoration:

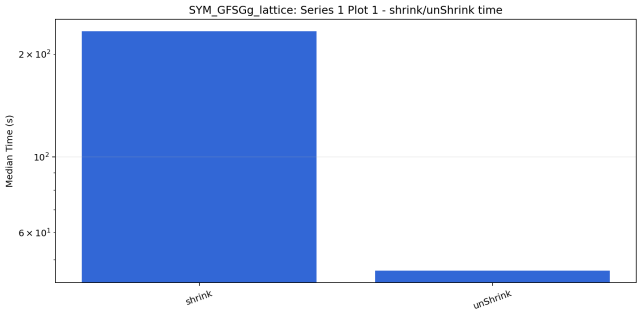
```
varlistGFSGg[[1]]
```

Explicit selections are also allowed:

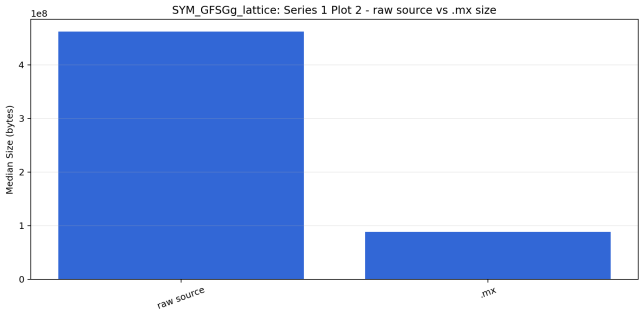
```
uncompress[intexpr, "SYM_GFSGg_lattice_cache.wlc"];
uncompress[{intexpr, varlistGFSGg}, "SYM_GFSGg_lattice_cache.wlc"];
```

7) Notes on the benchmark plots

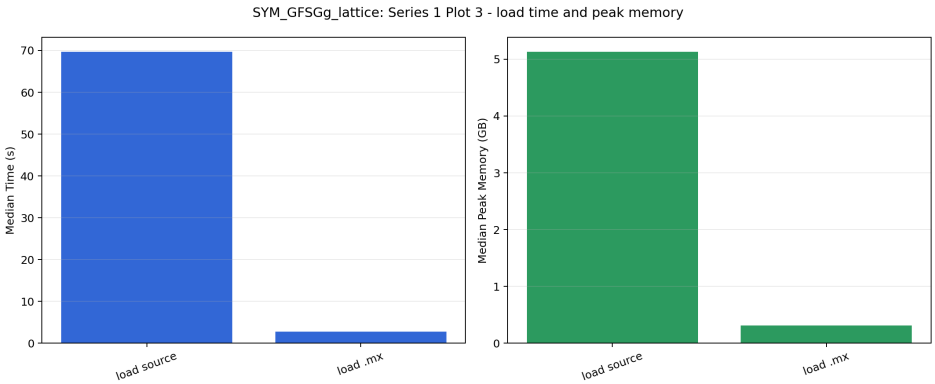
The benchmark figures were generated using the `SYM_GFSGg_lattice.m3` dataset with the symbol set {`intexpr`, `varlistGFSGg`}. Time plots use a logarithmic vertical axis because storage and restore stages differ by more than one order of magnitude.



(a) `shrink` versus `unShrink` time



(b) Raw source versus `.mx` size

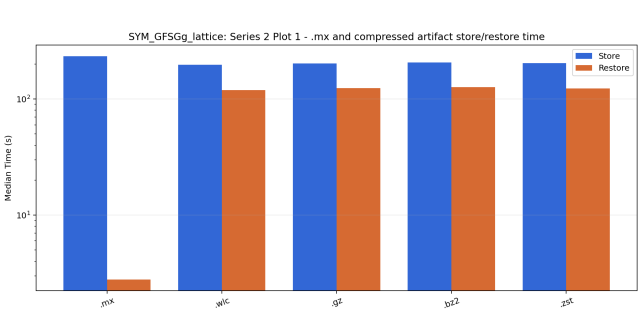


(c) Raw-source load versus `loadShrink["...mx"]`: time and peak memory

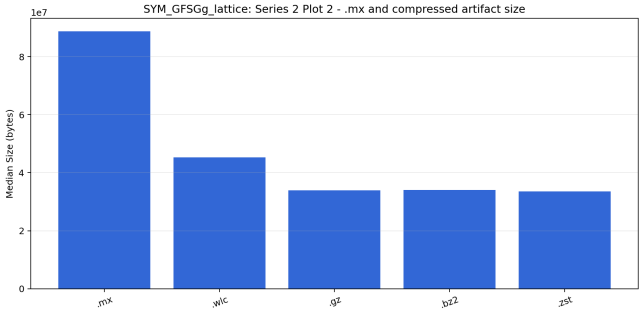
FIG. 3: Series 1 benchmark for the native `.mx` workflow.

Main observations.

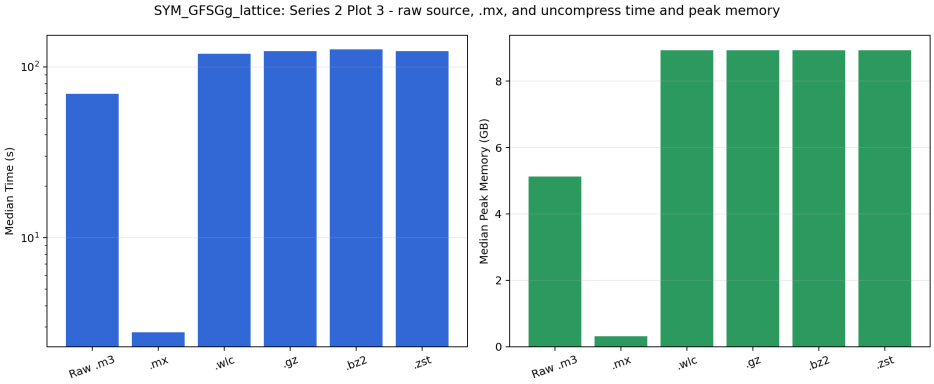
- Compared to the raw `.m3` source, `.mx` is approximately $\sim 5.2\times$ smaller on disk, $\sim 25\times$ faster to reload, and $\sim 16.2\times$ lower in peak memory during reload.
- Among the compressed cache formats, `.zst` and `.gz` achieve the smallest artifacts (~ 33.6 – 34.0 MB), which is about $\sim 2.6\times$ smaller than `.mx`.



(a) `.mx` and compressed artifact store/restore time



(b) `.mx`, `.wlc`, `.gz`, `.bz2`, and `.zst` artifact size



(c) Raw `.m3`, `.mx`, and compressed-cache restore time and peak memory

FIG. 4: Series 2 benchmark for compressed caches derived from the serialized payload.

TABLE I: Summary of the `SYM_GFSGg_lattice` benchmark campaign.

Artifact	Size (MB)	Store (s)	Load/Restore (s)	Peak Memory (GB)
Raw `.m3`	461.9	–	69.68	5.13
`.mx`	88.8	233.66	2.79	0.317
`.wlc`	45.3	197.81	119.64	8.93
`.gz`	34.0	202.15	123.96	8.94
`.bz2`	34.1	206.44	126.72	8.94
`.zst`	33.6	203.30	123.72	8.94

- The compact compressed caches trade disk reduction for restore cost: their restore times are $\sim 120 - 127$ s and their peak memory is close to 8.94 GB, substantially higher than direct `.mx` reloads.
- `.wlc` serves as the no-extra-file-compression baseline for the serialized payload and preserves the full raw serialized size.

Interpretation. `.mx` is the preferred format for iterative interactive work and repeated reloads inside Mathematica, because it minimizes both reload time and peak memory. The file-compressed caches (`.gz`, `.bz2`, `.zst`) are useful when disk footprint is the dominant concern and a slower reconstruction step is acceptable.

Appendix A: Evaluation of Continuum Integrals

There are three main commands: 'reducediamond', 'ChetyrkinFormula' and 'FourierTransformIntegral', which can be use to calculate loop integrals in the continuum. We use dimensional regularization in order to calculate the integrals in the continuum, in $D = 4 - 2\epsilon$ dimensions.

Firstly, the 'reducediamond' is designed to transform diamond-topology to simpler topologies. Its primary purpose is to process "diamond"-type diagrams, which commonly appear in higher loop calculations.

```
reducediamond[expression]
```

This function makes use of the recursion relation, given in Chetyrkin et al., N.P. B.192, 159, for scalar two-loop integrals of the "diamond" type, i.e., with propagators of the form:

$$(\widehat{p_1}^{\,2})^a(-\widehat{p_1+q_1}^{\,2})^b(-\widehat{p_1+p_2}^{\,2})^l(\widehat{p_1}^{\,2})^m(-\widehat{p_2+q_1}^{\,2})^n \tag{A1}$$

The function can work also for tensor two-loop integrals, using a generalized recursion relation given in Ref. [15].

Note that before applying 'reducediamond' in an expression, the latter must be expanded, or the part of the expression that depends on both denominators and p_1 must be collected. Below are the commands that you can use to evaluate such two-loop integrals:

```
In[1]:= << /home/username/SUPER_QUANTA_software-main/input.m

In[2]:= = hat2[p[1]] hat2[-p[1]+q[1]]^2 hat2[-p[1]+p[2]]^2 hat2[p[2]] hat2[-p[2]+q[1]]

In[3]:= makedenoms[%]

In[4]:= reducediamond[%]

In[5]:= unmakedenoms[%]
```

Secondly, the 'ChetyrkinFormula' function automates the application of a standard one-loop formula for Feynman integrals, a cornerstone in the renormalization of operators in QFT, given in Ref.[1]. It is particularly suited for dimensional regularization techniques and higher-loop corrections, enabling symbolic and numerical evaluations of Feynman integrals. This function performs integration over the momentum p[i], using the one-loop formula.

```
ChetyrkinFormula[expression, p[i_]]
```

For example:

```
In[6]:= hat2[p[1]] hat2[p[1]+q[1]]^2 s2[2 p[1],rho[1]] hat2[p[2]] hat2[p[2]+q[1]]

In[7]:= ChetyrkinFormula[%, p[1]]

In[8]:= ChetyrkinFormula[%, p[2]]

In[9]:= % /. grule
```

Lastly, the 'FourierTransformIntegrals' function handles the symbolic evaluation of Fourier transform integrals, a crucial task in converting position-space quantities into momentum-space representations in QFT. In particular, it calculates one-loop integrals of p[1] having the form:

$$\exp[i\,z\,p]/(p^2)^a(p_{\mu_1}\cdots p_{\mu_n}) \tag{A2}$$

```
FourierTransformIntegral[expression]

In[10]:= exp[im z p[1]] hat2[p[1]] s2[2p[1],rho[1]]

In[11]:= FourierTransformIntegral[%] /.  oneminusx -> (1-x) /. hold[a_] :> a ;

In[12]:= FullSimplify[%]
```

Furthermore, we have extended our software to evaluate integrals with massive denominators using the 't Hooft procedure of Ref. [19]. After introducing Feynman parameters (see, e.g., Ref. [17]) and combining denominators, loop integrals can be reduced to a single quadratic denominator.

For example, suppose we have two denominators,

$$A = p^2 + m_1^2, \qquad B = (p+q)^2 + m_2^2, \tag{A3}$$

and powers $i, j > 0$, one uses

$$\frac{1}{A^{\,i}B^{\,j}} = \frac{\Gamma(i+j)}{\Gamma(i)\Gamma(j)}\int_0^1 dx\; x^{j-1}(1-x)^{i-1}\;\frac{1}{\big[(1-x)A + xB\big]^{i+j}}. \tag{A4}$$

In Euclidean space,

$$(1-x)a + xb = p^2 + 2\,p{\cdot}q\,x + (1-x)m_1^2 + xm_2^2 + x\,q^2. \tag{A5}$$

In our code we reduce denominators to:

$$\texttt{dentHooft}[k, m^2] \;\equiv\; p^2 + 2\,k{\cdot}p + m^2 = p^2 + 2\,p{\cdot}\underbrace{q\,x}_{k} + \underbrace{(1-x)m_1^2 + x m_2^2 + x\,q^2}_{m^2}\,. \tag{A6}$$

As an example, we present the list of integrals appearing in our expression. Let the full expression be denoted by exprtHooft:

```
In[13] := exprtHooft = (g^2*n2m1on*delf[afl[2], afl[1]]*hat2[p[1]]*openspur[af[1], af[2]]*
    (-2*im*productDirac[fnu[1], rho[1], fnu[2]]*hat2[p[1] + q[1], m[1]]*s2[2*p[1], rho[1]]
     +2*im*productDirac[fnu[1], gamma5, rho[1], gamma5, fnu[2]]*hat2[p[1] + q[1], m[1]]*s2[2*p[1], rho[1
     + m*productDirac[fnu[1], fnu[2]]*hat2[p[1] + q[1], m[1]]
     + im*productDirac[fnu[1], rho[1], rho[2], rho[1], fnu[2]]*hat2[p[1] + q[1], m[1]]*s2[2*q[1], rho[2]
     - beta*m*productDirac[fnu[1], rho[1], rho[2], fnu[2]]*hat2[p[1]]*hat2[p[1] + q[1], m[1]]*
         s2[2*p[1], rho[1]]*s2[2*p[1], rho[2]]
     - beta*im*productDirac[fnu[1], rho[1], rho[2], rho[3], fnu[2]]*hat2[p[1]]*hat2[p[1] + q[1], m[1]]*
         s2[2*p[1], rho[1]]*s2[2*p[1], rho[2]]*s2[2*p[1], rho[3]]
     - beta*im*productDirac[fnu[1], rho[1], rho[2], rho[3], fnu[2]]*hat2[p[1]]*hat2[p[1] + q[1], m[1]]*
         s2[2*p[1], rho[1]]*s2[2*p[1], rho[2]]*s2[2*q[1], rho[2]]))/2;


In[14] := intlList = ((#/(# /. hat2[__]->1 /. s2[2p[1],_]-> 1 /. s2[p[1],_]-> 1 ))& /@
List @@ Expand[%]) // Union

Out[14] := {
  hat2[p[1]] hat2[p[1] + q[1], m[1]],

  hat2[p[1]] hat2[p[1] + q[1], m[1]] s2[2 p[1], rho[1]],

  hat2[p[1]]^2 hat2[p[1] + q[1], m[1]] s2[2 p[1], rho[1]] s2[2 p[1], rho[2]],

  hat2[p[1]]^2 hat2[p[1] + q[1], m[1]] s2[2 p[1], rho[1]] s2[2 p[1], rho[2]] s2[2 p[1], rho[3]]
}
```

When a momentum-dependent factor appears with a power greater than 2 (in particular, whenever you encounter a squared factor, you should first linearize it by introducing an auxiliary dummy index and an explicit Kronecker delta contraction. Concretely, perform the replacement below and then simplify the result:

```
In[15]:= % /. a_ s2[2p[1],rho[1]]^2 :> a delm[rho[22],rho[1]] s2[2p[1],rho[1]] s2[2p[1],rho[22]];

In[16]:= % /. hat2[p[1]] -> inv[p^2]
           /. hat2[p[1]+q[1]] -> inv[(p+q)^2]
           /. hat2[p[1]+q[1],m] -> inv[(p+q)^2 + m^2]
           /. hat2[p[1],m_] :> inv[p^2 + m^2] ;

In[17]:= Variables[%]

In[18]:= %% /. inv[a_]^i_. inv[b_]^j_. :>
             intx1 (1-x1)^(i-1) x1^(j-1) Gamma[i+j]/Gamma[i]/Gamma[j]/
               (den[collect[Expand[(1-x1) a + x1 b],p]])^(i+j)/.
               inv[a_]^i_. :> den[Expand[a]]^(-i);

In[19]:= Variables[%]

In[20]:= %% /. p[1] -> p ;

In[21]:= % /. den[p^2 + p b_ + c_] :> dentHooft[Expand[b/2], Expand[c]];

In[22]:= % /. den[p^2 + c_] :> dentHooft[0, Expand[c]];

In[23]:= Variables[%]
```

```
In[24]:= %%

In[25]:= tHooftIntegrals ;

In[26]:= tHooftIntegrals /. p[2] -> p

In[27]:= %%% /. % ;

In[28]:= Variables[%] // Sort
```

In the above commands, we have used the Eqs. (A8), (A9), (A11) and (A12). To elucidate the procedure, with the measure:

$$\int_p \equiv \int \frac{d^n p}{(2\pi)^n}, \qquad \Delta \equiv m^2 - k^2, \tag{A7}$$

the basic integrals (Euclidean versions of Eqs. (B.5)–(B.10) in Ref. [19]) read:

$$\int_p \frac{1}{\left(p^2 + 2k\cdot p + m^2\right)^\alpha} = \frac{1}{(4\pi)^{n/2}} \frac{\Gamma\left(\alpha - \frac{n}{2}\right)}{\Gamma(\alpha)} \frac{1}{\Delta^{\,\alpha - n/2}}, \tag{A8}$$

$$\int_p \frac{p_\mu}{\left(p^2 + 2k\cdot p + m^2\right)^\alpha} = \frac{1}{(4\pi)^{n/2}} \frac{\Gamma\left(\alpha - \frac{n}{2}\right)}{\Gamma(\alpha)} \frac{(-k_\mu)}{\Delta^{\,\alpha - n/2}}, \tag{A9}$$

$$\int_p \frac{p^2}{\left(p^2 + 2k\cdot p + m^2\right)^\alpha} = \frac{1}{(4\pi)^{n/2}} \frac{1}{\Gamma(\alpha)} \frac{1}{\Delta^{\,\alpha - n/2}} \left[\Gamma\left(\alpha - \frac{n}{2}\right) k^2 + \Gamma\left(\alpha - 1 - \frac{n}{2}\right) \frac{n}{2}\, \Delta\right], \tag{A10}$$

$$\int_p \frac{p_\mu p_\nu}{\left(p^2 + 2k\cdot p + m^2\right)^\alpha} = \frac{1}{(4\pi)^{n/2}} \frac{1}{\Gamma(\alpha)} \frac{1}{\Delta^{\,\alpha - n/2}} \left[\Gamma\left(\alpha - \frac{n}{2}\right) k_\mu k_\nu + \Gamma\left(\alpha - 1 - \frac{n}{2}\right) \frac{1}{2}\, \delta_{\mu\nu}\, \Delta\right], \tag{A11}$$

$$\int_p \frac{p_\mu p_\nu p_\lambda}{\left(p^2 + 2k\cdot p + m^2\right)^\alpha} = \frac{1}{(4\pi)^{n/2}} \frac{1}{\Gamma(\alpha)} \frac{1}{\Delta^{\,\alpha - n/2}} \Bigg[- \Gamma\left(\alpha - \frac{n}{2}\right) k_\mu k_\nu k_\lambda \\ - \Gamma\left(\alpha - 1 - \frac{n}{2}\right) \frac{1}{2} (\delta_{\mu\nu} k_\lambda + \delta_{\mu\lambda} k_\nu + \delta_{\nu\lambda} k_\mu) \Delta \Bigg], \tag{A12}$$

$$\int_p \frac{p^2\, p_\mu}{\left(p^2 + 2k\cdot p + m^2\right)^\alpha} = \frac{1}{(4\pi)^{n/2}} \frac{1}{\Gamma(\alpha)} \frac{(-k_\mu)}{\Delta^{\,\alpha - n/2}} \left[\Gamma\left(\alpha - \frac{n}{2}\right) k^2 + \Gamma\left(\alpha - \frac{n}{2} - 1\right) \frac{n+2}{2}\, \Delta\right]. \tag{A13}$$

The formulas generalize to arbitrary tensor rank; in our implementation we provide explicit reduction rules up to rank 5, i.e. up to five powers of the loop momentum in the numerator using the list provided in tHooftIntegrals.

Then we can multiply the expression by μ^{2e}, since there is an overall g^2 factor and we are working in the continuum.

```
In[29]:= Expand[mu^(2 e) %%];

In[30]:= % /. factor -> Factor

In[31]:= % /. nDim -> 4 - 2 e // FullSimplify;

In[32]:= (* Check that no terms like e * x1^(-1+e) appear: *)

In[33]:= %% /. 1 - x1 -> onemx /. x1 - 1 -> -onemx // InputForm

In[34]:= % //. (a_ b_)^c_ :> a^c b^c
          //. a_ (-2 + e) :> a factor[-2 + e]
          //. a_ (-1 + e) :> a factor[-1 + e];

In[35]:= Expand /@ %;

In[36]:= (List @@ (# + dummy)) & /@ %;
```

```
In[37]:= Union @@ %;

In[38]:= Select[%, (FreeQ[#, dummy]) &]

In[39]:= Select[%%, (FreeQ[#, dummy]) &];

In[40]:= Table[
           Select[%[[i]], (!FreeQ[#, x1] || !FreeQ[#, onemx]) &],
           {i, Length[%]}
         ] // Union

In[41]:= (* There no negative powers of x1 or of onemx, and thus the integration will not give 1/e *)

In[42]:= Series[Expand[%%%%%%%%%%], {e, 0, 0}] // Normal;

In[43]:= Variables[%] // Sort

In[44]:= %% // Expand // Simplify;

In[45]:= Table[
           Integrate[%[[i]], {x1, 0, 1},
             Assumptions -> {x1 >= 0 && x1 < 1 && m > 0 && q > 0}
           ],
           {i, Length[%]}
         ]

In[46]:= % /. intx1 -> 1;

In[47]:= % /. Log[2 mu] :> Log[2] + Log[mu];

In[48]:= % /. Log[mu] -> Log[mubar] + EulerGamma/2 - 1/2 Log[4 Pi] // Expand;

In[49]:= % /. Log[4 Pi/q^2] -> Log[4 Pi] - 2 Log[q];

In[50]:= % /. Log[4 Pi/m^2] -> Log[4 Pi] - 2 Log[m];

In[51]:= % /. Log[m] :> 1/2 Log[m^2/mubar^2] + 1/2 Log[mubar^2]
           /. Log[q[1]] :> Log[q[1]/mubar] + Log[mubar];

In[52]:= Expand[%] /. Log[2] -> 1/2 Log[4] /. Log[16] -> 2 Log[4] /.
                 Log[64] -> 3 Log[4] /. Log[4] -> Log[4 Pi] - Log[Pi] // Expand;

In[53]:= Simplify[%];

In[54]:= (* Simplify the above before substituting in exprtHooft:
   e.g., look at all Log[_], and make the arguments simpler: *)

In[55]:= %% /. Log[-q + Sqrt[4 m^2 + q^2]] -> -Log[q + Sqrt[4 m^2 + q^2]] + Log[4 m^2]
           /. ArcTanh[x__] :> 1/2 Log[(x + 1)/(1 - x)];

In[56]:= % /. Log[4 m^2] :> Log[4] + Log[m^2];

In[57]:= Factor[%];

In[58]:= IntegralList = Table[intlList[[i]] -> %[[i]], {i, Length[%]}];

In[59]:= % // Sort // Reverse

In[60]:= Expand[Expand[exprtHooft] /. %] /. 0. :> 0;

In[61]:= {Variables[%] // Sort, lb[%]}
```

```
In[62]:= %% /. s2[2 q, a_] :> s2[2 q[1], a];

In[63]:= {Variables[%] // Sort, lb[%]}

In[64]:= simplifydelm[Expand[%%]] // reducerho;

In[65]:= {Variables[%] // Sort, lb[%]}

In[66]:= exprtHooft = %% // FullSimplify

In[67]:= Quit
```

Appendix B: Evaluation of a primitively divergent lattice integral

Divergent integrals which appear in calculations up to $\mathcal{O}(a^1)$ may be evaluated using the standard procedure of Kawai et al. [8], in which one subtracts and adds to the original integrand its naive Taylor expansion, to the appropriate order with respect to a, in $D \to 4^+$ dimensions: The subtracted integrand, being UV convergent, is calculated in the continuum limit $a \to 0$, using the methods of Ref. [1], while the Taylor expansion terms are recast in terms of Bessel functions and are evaluated in the limit $\epsilon \to 0$ ($\epsilon \equiv (4-D)/2$).

In contrast to the above, some of the integrals in the section, given that they must be evaluated to $\mathcal{O}(a^2)$, have Taylor expansions which remain IR divergent all the way up to $D \leq 6$ dimensions. A related difficulty regards Kawai's procedure: Subtracting from the original integral its Taylor expansion in D-dimensions to the appropriate order, the UV-convergent subtracted expression at which one arrives can no longer be evaluated in the continuum limit by naively setting $a \to 0$, because there will be $\mathcal{O}(a^2)$ corrections which must not be neglected. These novel difficulties plague integrals C1, C2, C3, of Appendix B. Using a combination of momentum shifts, integration by parts and trigonometric identities, one may express C2 and C3 in terms of C1 and other less divergent integrals. Thus, it suffices to address the evaluation of C1

$$A1(p) = \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{1}{\hat{k}^2 \, \widehat{k+a\,p}^{\,2}} \tag{B1}$$

This is a prototype case of an integral which is IR divergent in $D \leq 6$ dimensions; in fact, all other integrals encountered in the present calculation may be expressed in terms of $A1(p)$ plus other integrals which are IR convergent at $D > 4$ (and are thus amenable to a more standard treatment).

First we split the original integrand I into two parts

$$I \equiv \frac{1}{\hat{k}^2 \, \widehat{k+a\,p}^{\,2}} = I_1 + I_2 \tag{B2}$$

where I_2 is obtained from I by a series expansion, with respect to the arguments of all trigonometric functions, to subleading order; I_1 is simply the remainder $I - I_2$

$$\begin{aligned} I_1 \;=\; & \frac{k^2 - \frac{k^4}{12} - \hat{k}^2}{k^2 \, \hat{k}^2 \, \widehat{k+a\,p}^{\,2}} + \frac{k^4 \left(k^2 - \hat{k}^2\right)}{12 \, (k^2)^2 \, \hat{k}^2 \, \widehat{k+a\,p}^{\,2}} + \frac{k^4 \left((k+a\,p)^2 - \widehat{k+a\,p}^{\,2}\right)}{12 \, (k^2)^2 \, (k+a\,p)^2 \, \widehat{k+a\,p}^{\,2}} \\ & + \frac{(k+a\,p)^2 - \frac{(k+a\,p)^4}{12} - \widehat{k+a\,p}^{\,2}}{k^2 \, (k+a\,p)^2 \, \widehat{k+a\,p}^{\,2}} + \frac{(k+a\,p)^4 \left((k+a\,p)^2 - \widehat{k+a\,p}^{\,2}\right)}{12\,k^2 \, ((k+a\,p)^2)^2 \, \widehat{k+a\,p}^{\,2}} \end{aligned} \tag{B3}$$

$$I_2 \;=\; \frac{1}{k^2 \, (k+a\,p)^2} + \left[\frac{(k+a\,p)^4}{12\,k^2 \, ((k+a\,p)^2)^2} + \frac{k^4}{12 \, (k^2)^2 \, (k+a\,p)^2} \right] \tag{B4}$$

($q^4 \equiv \sum_\mu q_\mu^4$). I_2 is free of trigonometric functions, while I_1 is naively Taylor expandable to $\mathcal{O}(a^2)$; its integral equals

$$\int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} I_1 = 0.004210419649(1) + a^2\,p^2\,0.0002770631001(3) + \mathcal{O}(a^4, a^4 \ln a) \tag{B5}$$

The errors appearing in the above equation come from extrapolations to infinite lattice size.

To evaluate the integral of I_2 we split the hypercubic integration region into a sphere of arbitrary radius μ about the origin ($\mu \leq \pi$) plus the rest

$$\int_{-\pi}^{\pi} = \int_{|k| \leq \mu} + \left(\int_{-\pi}^{\pi} - \int_{|k| \leq \mu} \right) \tag{B6}$$

The integral outside the sphere is free of IR divergences and is thus Taylor expandable to any order, giving[3] (for $\mu = 3.14155$)

$$\left(\int_{-\pi}^{\pi} - \int_{|k| \leq \mu} \right) \frac{d^4k}{(2\pi)^4} I_2 = 6.42919(3)\,10^{-3} + a^2\,p^2\,6.2034(1)\,10^{-5} + \mathcal{O}(a^4) \tag{B7}$$

---

[3] Due to its peculiar domain, this integral has been evaluated by a Monte Carlo routine, rather than as a sum over lattice points. The errors in Eq. (B7) are thus Monte Carlo errors.

We are now left with the integral of I_2 over a sphere. The most infrared divergent part of I_2 is $1/(k^2\,(k+a\,p)^2)$, with IR degree of divergence -4, and can be integrated *exactly*, giving

$$\int_{|k|\leq\mu}\frac{d^4k}{(2\pi)^4}\frac{1}{k^2\,(k+a\,p)^2}=\frac{1}{16\pi^2}\left(1-\ln(\frac{a^2\,p^2}{\mu^2})\right) \tag{B8}$$

The remaining two terms comprising I_2 have IR degree of divergence -2, thus their calculation to $\mathcal{O}(a^2)$ can be performed in D-dimensions, with D slightly greater that 4. Let us illustrate the procedure with one of these terms: $k^4/(\left(k^2\right)^2\,(k+a\,p)^4)$. By appropriate substitutions of

$$\frac{1}{(k+\bar{p})^2}=\frac{1}{k^2}+\frac{-2(k\cdot\bar{p})-\bar{p}^2}{k^2\,(k+\bar{p})^2}\qquad(\bar{p}\equiv a\,p) \tag{B9}$$

we split this term as follows

$$\begin{aligned}\frac{k^4}{\left(k^2\right)^2\,(k+\bar{p})^2} &= \left[\frac{k^4}{\left(k^2\right)^3}+\frac{k^4\,(-2(k\cdot\bar{p})-\bar{p}^2)}{\left(k^2\right)^4}+\frac{4\,k^4(k\cdot\bar{p})^2}{\left(k^2\right)^5}\right]\\ &+\left(\frac{k^4\left(4(k\cdot\bar{p})\bar{p}^2+(\bar{p}^2)^2\right)}{\left(k^2\right)^4\,(k+\bar{p})^2}+\frac{4\,k^4(k\cdot\bar{p})^2\left(-2(k\cdot\bar{p})-\bar{p}^2\right)}{\left(k^2\right)^5\,(k+\bar{p})^2}\right)\end{aligned} \tag{B10}$$

The part in square brackets is polynomial in a and can be integrated easily, using D-dimensional spherical coordinates. The remaining part is UV-convergent; thus the integration domain can now be recast in the form

$$\int_{|k|\leq\mu}=\int_{|k|<\infty}-\int_{\mu\leq|k|<\infty} \tag{B11}$$

The integral over the whole space can be performed using the methods of Ref. [1], whereas the integral outside the sphere of radius μ is $\mathcal{O}(a^3)$ and may be safely dropped. The same procedure is applied to the last term of I_2. Adding the contributions from all the steps described above, we check that the result is independent of μ.

Appendix C: A basis of divergent integrals

The most challenging aspect of this calculation, which demands careful attention, is isolating the dependence on the external momentum p and the lattice spacing a from the divergent terms. The singularities can be systematically extracted, and below we provide a list of primitively divergent integrals that arise in our algebraic expressions.

In the following integrals we define

$$\begin{aligned}\hat{k}_\mu &= 2\,\sin(\frac{k_\mu}{2})\\ \hat{k}^2 &= 4\sum_\mu\sin^2(\frac{k_\mu}{2})\\ \mathring{k}_\mu &= \sin(k_\mu)\end{aligned}$$

In addition, $(\ )_S$ means sum over inequivalent permutations. No summation over the indices μ, ν, ρ, σ is implied, unless otherwise stated.

$$\bullet\int_{-\pi}^{\pi}\frac{d^4k}{(2\pi)^4}\,\frac{1}{\hat{k}^2\,\widehat{k+a\,p}^{\,2}} \;=\; 0.036678329075-\frac{\ln(a^2p^2)}{16\pi^2}$$
$$+\;0.0000752406(3)\,a^2\,p^2+a^2\,\frac{\sum_\mu p_\mu^4}{384\pi^2\,p^2}+\mathcal{O}(a^4\,p^4) \tag{C1}$$

$$\bullet\int_{-\pi}^{\pi}\frac{d^4k}{(2\pi)^4}\,\frac{\mathring{k}_\mu}{\hat{k}^2\,\widehat{k+a\,p}^{\,2}} \;=\; a\,p_\mu\Big[-0.008655827648+\frac{\ln(a^2p^2)}{32\pi^2}$$
$$-\,0.0005107825(2)\,a^2\,p^2+0.001171329715\,a^2\,p_\mu^2$$
$$-\,a^2\,\frac{\sum_\mu p_\mu^4}{768\pi^2\,p^2}+a^2\,\frac{\ln(a^2p^2)}{384\pi^2}\left(\frac{p^2}{2}-p_\mu^2\right)\Big]$$
$$+\mathcal{O}(a^5\,p^5) \tag{C2}$$

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\overset{\circ}{k}_\mu \overset{\circ}{k}_\nu}{(\hat{k}^2)^2\, \widehat{k+a\,p}^{\,2}} = \delta_{\mu\nu}\Big[0.004327913824 - \frac{\ln(a^2p^2)}{64\pi^2} + 0.00025539124(8)\, a^2\, p^2 - 0.000135654113\, a^2\, p_\mu^2 + a^2\, \frac{\sum_\mu p_\mu^4}{1536\pi^2\, p^2} + a^2\, \frac{\ln(a^2p^2)}{768\pi^2} \left(p_\mu^2 - \frac{p^2}{2} \right) \Big] + a^2\, p_\mu\, p_\nu \Big[\frac{1}{32\pi^2\, a^2\, p^2} - 0.0003788538(2) + \frac{\sum_\mu p_\mu^4}{768\pi^2\, (p^2)^2} - \frac{(p_\mu^2 + p_\nu^2)}{384\pi^2\, p^2} + \frac{\ln(a^2p^2)}{768\pi^2} \Big] + \mathcal{O}(a^4\, p^4) \tag{C3}$$

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\overset{\circ}{k}_\mu \overset{\circ}{k}_\nu}{\hat{k}^2\, \widehat{k+a\,p}^{\,2}} = \delta_{\mu\nu}\Big[0.014966695116 - 0.001256484446\, a^2\, p^2 - 0.001027789631\, a^2\, p_\mu^2 + \frac{a^2\, p^2\, \ln(a^2p^2)}{192\pi^2} \Big] + a^2\, p_\mu\, p_\nu \Big[0.003970508789 - \frac{\ln(a^2p^2)}{48\pi^2} \Big] + \mathcal{O}(a^4\, p^4) \tag{C4}$$

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{(\overset{\circ}{k}_\mu)^3}{\hat{k}^2\, \widehat{k+a\,p}^{\,2}} = a\, p_\mu \Big[-0.006184131744 + 0.001102333439\, a^2\, p^2 - 0.000174224479\, a^2\, p_\mu^2 + a^2\, \frac{\ln(a^2p^2)}{64\pi^2} \left(p_\mu^2 - \frac{p^2}{2} \right) \Big] + \mathcal{O}(a^5 p^5) \tag{C5}$$

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\overset{\circ}{k}_\mu \overset{\circ}{k}_\nu \overset{\circ}{k}_\rho}{(\hat{k}^2)^2\, \widehat{k+a\,p}^{\,2}} = (\delta_{\nu\rho}\, a\, p_\mu)_S \Big[-0.000728769948 + \frac{\ln(a^2p^2)}{192\pi^2} \Big] + 0.001027789631 \delta_{\mu\nu\rho}\, a\, p_\mu - a\, \frac{p_\mu\, p_\nu\, p_\rho}{48\pi^2\, p^2} + \mathcal{O}(a^3\, p^3) \tag{C6}$$

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\sum_\mu \hat{k}_\mu^4}{16(\hat{k}^2)^2\, \widehat{k+a\,p}^{\,2}} = 0.004050096698 - 0.000107954163\, a^2\, p^2 + a^2\, \frac{\sum_\mu p_\mu^4}{1024\pi^2\, p^2} + \mathcal{O}(a^4\, p^4) \tag{C7}$$

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\mathring{k}_\mu\, \mathring{k}_\nu\, \mathring{k}_\rho\, \mathring{k}_\sigma}{(\hat{k}^2)^2\, \widehat{k+a\,p}^{\,2}} = 0.001589337971\, (\delta_{\mu\nu}\, \delta_{\rho\sigma})_S - 0.001675948042\, \delta_{\mu\nu\rho\sigma} - 0.000372782983(\delta_{\mu\nu\rho}\, a^2\, p_\mu\, p_\sigma)_S - 0.000062130497(\delta_{\mu\nu}\, \delta_{\rho\sigma}\, a^2\, p_\mu^2)_S + \delta_{\mu\nu\rho\sigma}\, \big(\, 0.000186391491\, a^2\, p^2 + 0.000410290033\, a^2\, p_\mu^2\big) + (\delta_{\mu\nu}\, a^2\, p_\rho\, p_\sigma)_S \left(\, 0.000227848225 - \frac{\ln(a^2p^2)}{384\pi^2}\right) + (\delta_{\mu\nu}\, \delta_{\rho\sigma})_S\, a^2\, p^2 \left(-0.000245852737 + \frac{\ln(a^2p^2)}{768\pi^2}\right) + a^2\, \frac{p_\mu\, p_\nu\, p_\rho\, p_\sigma}{64\pi^2\, p^2} + \mathcal{O}(a^4\, p^4) \tag{C8}$$

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\mathring{k}_\nu \sum_\mu \hat{k}_\mu^4}{16(\hat{k}^2)^2\, \widehat{k+a\,p}^{\,2}} = a\, p_\nu \Big[-0.000800034900 + 0.000069705553\, a^2\, p^2 + 0.000107082394\, a^2\, p_\nu^2 - a^2\, \frac{\sum_\rho p_\rho^4}{1280\pi^2\, p^2} - a^2\, \frac{\ln(a^2p^2)}{2560\pi^2} \left(\frac{p^2}{2} - p_\nu^2\right)\Big] + \mathcal{O}(a^5\, p^5) \tag{C9}$$

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\mathring{k}_\nu\, \mathring{k}_\rho \sum_\mu k_\mu \widehat{+a}\, p_\mu^{\,4}}{16(\hat{k}^2)^2\, (\widehat{k+a\,p}^{\,2})^2} = \delta_{\nu\rho}\Big[0.000400017450 - 0.000034852777\, a^2\, p^2 + a^2\, \frac{\sum_\mu p_\mu^4}{2560\pi^2\, p^2} + 0.000105349447\, a^2\, p_\nu^2 + a^2\, \frac{\ln(a^2p^2)}{5120\pi^2} (\frac{p^2}{2} - 3p_\nu^2)\Big] + a^2\, p_\nu\, p_\rho \Big[\, 0.000006643045 - \frac{p_\nu^2 + p_\rho^2}{2560\pi^2\, p^2} + \frac{\sum_\mu p_\mu^4}{5120\pi^2\, (p^2)^2} + \frac{\ln(a^2p^2)}{5120\pi^2}\Big] + \mathcal{O}(a^4\, p^4) \tag{C10}$$

Appendix D: Well-Known One-Loop Lattice Integrals P_1, P_2

Some one-loop and two-loop integrals can be found in Ref [11]. We list below the most well-known of them. Eqs. (D1)-(D5) refer to one-loop integrals with zero external momentum.

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{1}{\hat{k}^2} = P_1 \tag{D1}$$

with the numerical value: $P_1 = 0.15493339023106021(1)$.

$$\bullet P_2 = \lim_{m\to 0} \left(\frac{1}{(4\pi)^2} \ln(m^2) + \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{1}{(\hat{k}^2 + m^2)^2} \right) \tag{D2}$$

with the numerical value: $P_2 = 0.02401318111946489(1)$.

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\sum_\mu \hat{k}_\mu^4}{(\hat{k}^2)^2} = 1 - 4P_1 \tag{D3}$$

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\sum_\mu \hat{k}_\mu^4}{(\hat{k}^2)^3} = \frac{1}{2}P_1 - \frac{1}{8\pi^2} \tag{D4}$$

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\sum_\mu \hat{k}_\mu^6}{(\hat{k}^2)^3} = 1 - 5P_1 - \frac{1}{2\pi^2} \tag{D5}$$

$$\bullet \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{1}{\hat{k}^2 \widehat{k + ap}^2} = A(ap), \tag{D6}$$

where $A(p) = \frac{1}{(4\pi)^2}\left(-\ln(a^2p^2) + 2\right) + P_2 + O(a^2p^2)$ and Eq. (C1) gives also the $O(a^2p^2)$ contribution.

Appendix E: Evaluation of a basis of nontrivial divergent two-loop Feynman diagrams

In this appendix we present the procedure that we used to evaluate nontrivial divergent integrals which appeared in our two-loop computation using staggered fermions. In the Wilson case, the two-loop divergent integrals which appeared can be expressed in terms of a basis of standard integrals found in Ref. [11], along with manipulations found in Ref. [18]. However, in the staggered case, the divergent integrals are not related to those standard integrals in an obvious way. Some further steps are needed to this end.

In our computations, there appeared 4 types of nontrivial divergent 2-loop integrals; they are listed below:

$$I_{1\mu\nu} = \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\mathring{k}_\mu \, \mathring{k}_\nu}{(\widehat{k}^2)^2 \, (\widehat{k+aq})^2} \int_{-\pi}^{\pi} \frac{d^4p}{(2\pi)^4} \frac{1}{\mathring{p}^2 \, (\mathring{p+k})^2} \tag{E1}$$

$$I_{2\mu\nu} = \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\mathring{k}_\mu \, \sin(aq_\nu)}{(\widehat{k}^2)^2 \, (\widehat{k+aq})^2} \int_{-\pi}^{\pi} \frac{d^4p}{(2\pi)^4} \frac{1}{\mathring{p}^2 \, (\mathring{p+k})^2} \tag{E2}$$

$$I_{3\mu\nu\rho\sigma} = \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\mathring{k}_\mu \, \mathring{k}_\nu}{(\widehat{k}^2)^2 \, (\widehat{k+aq})^2} \int_{-\pi}^{\pi} \frac{d^4p}{(2\pi)^4} \frac{\sin(2p_\rho) \, \sin(2p_\sigma)}{(\mathring{p}^2)^2 \, (\mathring{p+k})^2} \tag{E3}$$

$$I_{4\mu\nu\rho\sigma} = \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\mathring{k}_\mu \, \sin(aq_\nu)}{(\widehat{k}^2)^2 \, (\widehat{k+aq})^2} \int_{-\pi}^{\pi} \frac{d^4p}{(2\pi)^4} \frac{\sin(2p_\rho) \, \sin(2p_\sigma)}{(\mathring{p}^2)^2 \, (\mathring{p+k})^2} \tag{E4}$$

where $\widehat{p}^2 = \sum_\mu \widehat{p}_\mu^2$, $\widehat{p}_\mu = 2\sin(p_\mu/2)$, $\mathring{p}^2 = \sum_\mu \mathring{p}_\mu^2$, $\mathring{p}_\mu = \sin(p_\mu)$ and q is an external momentum. The crucial point is the presence of expressions like $\mathring{p}^2$ or $(\mathring{p+k})^2$ rather than $\widehat{p}^2$ or $(\widehat{p+k})^2$ in the denominators of the above integrals. This behaviour comes from the tree-level staggered fermion propagator. Also, the other crucial point is the fact that we cannot manipulate these integrals via subtractions of the form:

$$\frac{1}{\mathring{p}^2} = \frac{1}{\widehat{p}^2} + \Big(\frac{1}{\mathring{p}^2} - \frac{1}{\widehat{p}^2}\Big) \tag{E5}$$

in order to express them in terms of a standard tabulated integral plus additional terms which are more convergent; such a procedure is applicable, e.g., in the case of the Wilson fermion propagator $\Big[1/\Big(\mathring{p}^2 + r^2(\widehat{p}^2)^2/4\Big)\Big]$ or in other less divergent integrals with staggered fermion propagators. The reason for which such a subtraction cannot be applied is the existence of potential IR singularities at all corners of the Brillouin zone (not only at zero momentum), in the staggered fermion propagator. Therefore, such a subtraction will not alleviate the divergent behaviour at the remaining corners of the Brillouin zone.

For the above integrals we followed a different approach. At first, we perform the substitution $p_\mu \to p'_\mu + \pi \, C_\mu$, where $-\pi/2 < p'_\mu < \pi/2$ and $C_\mu \in \{0, 1\}$, which is the same substitution we applied to the external momenta. Now the integration region for the innermost integral breaks up into 16 regions with range $[-\pi/2, \pi/2]$; the contributions from these regions are identical. To restore the initial range $[-\pi, \pi]$, we apply the following change of variables: $p'_\mu \to p''_\mu = 2p'_\mu$. Then we obtain:

$$I_{1\mu\nu} = 16 \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\mathring{k}_\mu \, \mathring{k}_\nu}{(\widehat{k}^2)^2 \, (\widehat{k+aq})^2} \int_{-\pi}^{\pi} \frac{d^4p}{(2\pi)^4} \frac{1}{\widehat{p}^2 \, (\widehat{p+2k})^2} \tag{E6}$$

$$I_{2\mu\nu} = 16 \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\mathring{k}_\mu \, \sin(aq_\nu)}{(\widehat{k}^2)^2 \, (\widehat{k+aq})^2} \int_{-\pi}^{\pi} \frac{d^4p}{(2\pi)^4} \frac{1}{\widehat{p}^2 \, (\widehat{p+2k})^2} \tag{E7}$$

$$I_{3\mu\nu\rho\sigma} = 64 \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\mathring{k}_\mu \, \mathring{k}_\nu}{(\widehat{k}^2)^2 \, (\widehat{k+aq})^2} \int_{-\pi}^{\pi} \frac{d^4p}{(2\pi)^4} \frac{\mathring{p}_\rho \, \mathring{p}_\sigma}{(\widehat{p}^2)^2 \, (\widehat{p+2k})^2} \tag{E8}$$

$$I_{4\mu\nu\rho\sigma} = 64 \int_{-\pi}^{\pi} \frac{d^4k}{(2\pi)^4} \frac{\mathring{k}_\mu \, \sin(aq_\nu)}{(\widehat{k}^2)^2 \, (\widehat{k+aq})^2} \int_{-\pi}^{\pi} \frac{d^4p}{(2\pi)^4} \frac{\mathring{p}_\rho \, \mathring{p}_\sigma}{(\widehat{p}^2)^2 \, (\widehat{p+2k})^2} \tag{E9}$$

where we omit the double prime from p. The above integrals are similar to standard divergent integrals, computed in Ref. [11]. The only difference is the presence of a factor of 2 in the denominators, i.e. $1/(\widehat{p+2k})^2$. This can be

treated via subtraction methods. We define:

$$A(k) = \int_{-\pi}^{\pi} \frac{d^4p}{(2\pi)^4} \frac{1}{\widehat{p}^2\ \widehat{(p+k)^2}} \tag{E10}$$

$$A_{\rm as}(k) \equiv \frac{1}{(4\pi)^2}[-\ln(k^2)+2] + P_2 \tag{E11}$$

$$B_{\rho\sigma}(k) = \int_{-\pi}^{\pi} \frac{d^4p}{(2\pi)^4} \frac{\overset{\circ}{p}_\rho\ \overset{\circ}{p}_\sigma}{(\widehat{p}^2)^2\ \widehat{(p+k)^2}} \tag{E12}$$

$$\widetilde{B}_{\rho\sigma}(2k) \equiv \frac{1}{2(4\pi)^2} \frac{\overset{\circ}{k}_\rho\ \overset{\circ}{k}_\sigma}{\widehat{k}^2} + \delta_{\rho\sigma}[\frac{1}{4}A(2k) - \frac{1}{32}P_1] \tag{E13}$$

where the values of the numerical constants P_1 and P_2 are noted in Ref. [11]. $A_{\rm as}(k)$ and $\widetilde{B}_{\rho\sigma}(2k)$ are asymptotic values of $A(k)$ and $B_{\rho\sigma}(2k)$, respectively:

$$A(k) = A_{\rm as}(k) + \mathcal{O}(k^2), \qquad B_{\rho\sigma}(2k) = \widetilde{B}_{\rho\sigma}(2k) + \mathcal{O}(k^2) \tag{E14}$$

The first two integrals $I_{1\mu\nu}$ and $I_{2\mu\nu}$ contain the quantity $A(2k)$ and the remaining two integrals $I_{3\mu\nu\rho\sigma}$ and $I_{4\mu\nu\rho\sigma}$ the quantity $B_{\rho\sigma}(2k)$. We apply the following subtractions:

$$A(2k) = A(k) + [A_{\rm as}(2k) - A_{\rm as}(k)] + [A(2k) - A(k) - A_{\rm as}(2k) + A_{\rm as}(k)] \tag{E15}$$

$$B_{\rho\sigma}(2k) = \widetilde{B}_{\rho\sigma}(2k) + [B_{\rho\sigma}(2k) - \widetilde{B}_{\rho\sigma}(2k)] \tag{E16}$$

Integrals $I_{1\mu\nu}$ and $I_{2\mu\nu}$ separate into 3 sub-integrals. The first sub-integral with the quantity $A(k)$ is already computed in Ref. [11] (for $I_{1\mu\nu}$) or can be converted into standard divergent integrals of Ref. [11] using integration by parts (for $I_{2\mu\nu}$). The second sub-integral with the quantity $[A_{\rm as}(2k) - A_{\rm as}(k)] = -\ln 4/(4\pi)^2$ is a one loop divergent integral computed in Ref. [11] or [2]. The third sub-integral with the quantity $[A(2k) - A(k) - A_{\rm as}(2k) + A_{\rm as}(k)] = \mathcal{O}(k^2)$ is convergent and so we can integrate it numerically for $a \to 0$ (In particular, it gives zero for $I_{2\mu\nu}$). Also, integrals $I_{3\mu\nu\rho\sigma}$ and $I_{4\mu\nu\rho\sigma}$ separate into 2 sub-integrals. The first sub-integral with the quantity $\widetilde{B}_{\rho\sigma}(2k)$ gives expressions which can be converted into standard integrals of Refs. [2, 11] or into the above $I_{1\mu\nu}$, $I_{2\mu\nu}$ integrals. The second sub-integral with the quantity $[B_{\rho\sigma}(2k) - \widetilde{B}_{\rho\sigma}(2k)] = \mathcal{O}(k^2)$ is convergent and so we can integrate it numerically for $a \to 0$ (In particular, it gives zero for $I_{4\mu\nu\rho\sigma}$). Therefore, according to the above manipulations, the final expressions for the four integrals are given by:

$$\begin{aligned} I_{1\mu\nu} &= \Big\{\frac{2}{(2\pi)^4}\Big[-\ln(a^2q^2) + \frac{3}{2} - \ln 4\Big] + \frac{1}{2\pi^2}P_2\Big\}\frac{q_\mu q_\nu}{q^2} \\ &+ \delta_{\mu\nu}\Big\{\frac{2}{(4\pi)^4}\Big[\ln(a^2q^2)\Big]^2 - \frac{1}{4\pi^2}\Big[P_2 + \frac{1}{(4\pi)^2}\Big(\frac{5}{2} - \ln 4\Big)\Big]\ln(a^2q^2) - \frac{1}{4\pi^2}\Big[P_2 + \frac{3}{2(4\pi)^2}\ln 4\Big] + 4X_2 + G_1\Big\} \\ &+ \mathcal{O}(a^2q^2) \end{aligned} \tag{E17}$$

$$I_{2\mu\nu} = \Big\{\frac{1}{(2\pi)^4}\Big[\ln(a^2q^2) - 2 + \ln 4\Big] - \frac{1}{\pi^2}P_2\Big\}\frac{q_\mu q_\nu}{q^2} + \mathcal{O}(a^2q^2) \tag{E18}$$

$$\begin{aligned} I_{3\mu\nu\rho\sigma} &= \frac{1}{3(2\pi)^4}\frac{q_\mu q_\nu q_\rho q_\sigma}{q^4} + \delta_{\rho\sigma}\Big\{\frac{2}{(2\pi)^4}\Big[-\ln(a^2q^2) + \frac{5}{3} - \ln 4\Big] - \frac{1}{(4\pi)^2}(P_1 - 8P_2)\Big\}\frac{q_\mu q_\nu}{q^2} \\ &+ \frac{1}{12(2\pi)^4}\Big\{\delta_{\mu\nu}\frac{q_\rho q_\sigma}{q^2} + \delta_{\mu\rho}\frac{q_\nu q_\sigma}{q^2} + \delta_{\mu\sigma}\frac{q_\nu q_\rho}{q^2} + \delta_{\nu\rho}\frac{q_\mu q_\sigma}{q^2} + \delta_{\nu\sigma}\frac{q_\mu q_\rho}{q^2}\Big\} \\ &+ \delta_{\mu\nu}\delta_{\rho\sigma}\Big\{\frac{2}{(4\pi)^4}\Big[\ln(a^2q^2)\Big]^2 - \frac{1}{4\pi^2}\Big[P_2 - \frac{1}{8}P_1 + \frac{1}{(4\pi)^2}\Big(\frac{51}{2} - \ln 4\Big)\Big]\ln(a^2q^2) \\ &\qquad -\frac{1}{4\pi^2}\Big[\Big(\frac{1}{3} - \ln 4\Big)P_2 - \frac{11}{144}P_1 + \frac{3}{2(4\pi)^2}\Big(\frac{1}{27} - \ln 4\Big)\Big] - \frac{1}{2}P_1\ P_2 + 4X_2 + G_1 + G_3\Big\} \\ &+ (\delta_{\mu\rho}\delta_{\nu\sigma} + \delta_{\mu\sigma}\delta_{\nu\rho})\Big\{\frac{1}{(12\pi)^4}\Big[-\ln(a^2q^2) + \frac{1}{6}\Big] + \frac{1}{6\pi^2}(P_1 + 3P_2) + G_2\Big\} \\ &+ \delta_{\mu\nu\rho\sigma}\Big\{\frac{1}{(2\pi)^4} + \frac{1}{2(4\pi)^2} - \frac{1}{3\pi^2}P_1 + G_4\Big\} + \mathcal{O}(a^2q^2) \end{aligned} \tag{E19}$$

$$
\begin{aligned}
I_{4\mu\nu\rho\sigma} &= -\frac{1}{2(2\pi)^4}\frac{q_\mu q_\nu q_\rho q_\sigma}{q^4} - \frac{4}{(4\pi)^4}\Big\{\delta_{\mu\rho}\frac{q_\nu q_\sigma}{q^2} + \delta_{\mu\sigma}\frac{q_\nu q_\rho}{q^2}\Big\} \\
&+ \delta_{\rho\sigma}\Big\{\frac{1}{(2\pi)^4}\Big[\ln(a^2q^2) - \frac{9}{4}\Big] - \frac{1}{2(2\pi)^2}(P_1 - 8P_2)\Big\}\frac{q_\mu q_\nu}{q^2} + \mathcal{O}(a^2q^2)
\end{aligned} \tag{E20}
$$

where P_1, P_2, X_2 are given in Ref. [11] and $G_1 - G_4$ are given below:

$$G_1 = 0.000803016(6) \tag{E21}$$
$$G_2 = -0.0006855532(7) \tag{E22}$$
$$G_3 = 0.00098640(7) \tag{E23}$$
$$G_4 = 0.00150252(2) \tag{E24}$$

Appendix F: Introduction to Overlap fermions for SQCD

In this appendix, we present the overlap lattice formulation of the SQCD action introduced in Ref. [13]. The theory is defined with gauge group $SU(N_c)$, where the gauge sector described by the Wilson plaquette action constructed from the gluon fields u_μ. Overlap fermions are employed to discretize both the gluino fields λ and the quark fields ψ, whereas the squark fields $A_\pm$ are defined using a naive lattice discretization. Ensuring chiral invariance of the Yukawa interactions requires a modified construction, which leads to additional terms in the lattice action, as detailed below.

In our conventions the SQCD lattice action reads[4]:

$$
\begin{aligned}
S^{\rm Total}_{\rm SQCD} = &\frac{N_c}{g^2}\sum_{x,\mu,\nu}\Big(1 - \frac{1}{N_c}{\rm Tr}\, U_{\mu\nu}(x)\Big) - a^8\sum_{x,y}{\rm Tr}\big[\lambda^T(x)\,\mathcal{C}\,\mathcal{D}^{\rm adj}_{\rm ov}(x,y)\lambda(y)\big] \\
&+ a^8\sum_{x,y}\bar\psi(x)\Big[\Big(1 - \frac{a\,m}{2}\Big)\mathcal{D}_{\rm ov}(x,y) + m\,\frac{1}{a^4}\delta_{x,y}\Big]\psi(y) \\
&+ a^4\sum_x\Big[(\mathcal{D}_\mu A_+(x))^\dagger\mathcal{D}_\mu A_+(x) + \mathcal{D}_\mu A_-(x)\,\mathcal{D}_\mu A^\dagger_-(x) + m^2\Big(A^\dagger_+(x)\,A_+(x) + A_-(x)\,A^\dagger_-(x)\Big) \\
&\qquad + \frac{1}{2}g^2\Big(A^\dagger_+(x)\,T^\alpha A_+(x) - A_-(x)\,T^\alpha A^\dagger_-(x)\Big)^2\Big] \\
&+ i\sqrt{2}g\,a^4\sum_x\Big[A^\dagger_+(x)\,\bar\lambda^\alpha(x)\,T^\alpha P_+\psi(x) - \bar\psi(x)\,P_-\lambda^\alpha(x)\,T^\alpha A_+(x) \\
&\qquad + A_-(x)\,\bar\lambda^\alpha(x)\,T^\alpha P_-\psi(x) - \bar\psi(x)\,P_+\lambda^\alpha(x)\,T^\alpha A^\dagger_-(x)\Big] \\
&+ \frac{a\,g^2}{2}a^4\sum_x\bar\lambda^\alpha(x)\Big[\Big(A^\dagger_+(x)\{T^\alpha,T^\beta\}A^\dagger_-(x)\Big)P_+ + \big(A_-(x)\{T^\alpha,T^\beta\}A_+(x)\big)P_-\Big]\lambda^\beta(x) \\
&- \sum_x\Big({\rm Tr}\log(a\,\Xi_+(x)) + {\rm Tr}\log(a\,\Xi_-(x))\Big) \\
&+ \frac{1}{4}\,a^4\sum_x\Big(\bar\Psi^\alpha(x) - (\Psi^T)^\alpha(x)\,\mathcal{C}\Big)\left((\mathcal{C}^T\,\Xi(x))^{-1}\right)^{\alpha\beta}\Big((\bar\Psi^T)^\beta(x) + \mathcal{C}\,\Psi^\beta(x)\Big).
\end{aligned} \tag{F1}
$$

[4] The trace in the first term is taken in the fundamental representation, whereas the traces in the second term (first line) and in the penultimate term are taken in the adjoint representation.

where[5]:

$$U_{\mu\nu}(x) = U_\mu(x)U_\nu(x+a\hat{\mu})U_\mu^\dagger(x+a\hat{\nu})U_\nu^\dagger, \qquad U_\mu(x) = \exp(iag\,u_\mu(x+a\hat{\mu}/2))$$

$$\mathcal{D}_{\rm ov}^{\rm adj}(x,y) = \frac{\rho^\lambda}{2a}\left[\frac{\delta_{x,y}}{a^4} + \left(X^{\rm adj}(X^{\rm adj\dagger}X^{\rm adj})^{-1/2}\right)_{x,y}\right],$$

$$X^{\rm adj}(x,y) = -\frac{\rho^\lambda}{a}\frac{\delta_{x,y}}{a^4} + \frac{1}{2}\sum_\mu\left[\gamma_\mu(\nabla_\mu^{\rm adj}(x,y) + \nabla_\mu^{*\rm adj}(x,y)) - a\,r^\lambda a^4\sum_z \nabla_\mu^{*\rm adj}(x,z)\nabla_\mu^{\rm adj}(z,y)\right],$$

$$\nabla_\mu^{\rm adj}(x,y) = \frac{1}{a}\left(U_\mu^{\rm adj}(x)\,\frac{1}{a^4}\,\delta_{x+a\hat{\mu},y} - \mathbb{1}\,\frac{1}{a^4}\,\delta_{x,y}\right), \quad \nabla_\mu^{*\,\rm adj}(x,y) = \frac{1}{a}\left(\mathbb{1}\,\frac{1}{a^4}\,\delta_{x,y} - U_\mu^{\rm adj\,\dagger}(x-a\hat{\mu})\,\frac{1}{a^4}\,\delta_{x-a\hat{\mu},y}\right),$$

$$\left[U_\mu^{\rm adj}(x)\right]^{\alpha\beta} = 2\,{\rm Tr}\left(T^\alpha\,U_\mu(x)\,T^\beta\,U_\mu^\dagger(x)\right),$$

$$\mathcal{D}_{\rm ov}(x,y) = \frac{\rho^\psi}{a}\left[\frac{\delta_{x,y}}{a^4} + \left(X(X^\dagger X)^{-1/2}\right)_{x,y}\right],$$

$$X = -\frac{\rho^\psi}{a}\frac{\delta_{x,y}}{a^4} + \frac{1}{2}\sum_\mu\left[\gamma_\mu(\nabla_\mu(x,y) + \nabla_\mu^*(x,y)) - a\,r^\psi a^4\sum_z \nabla_\mu^*(x,z)\nabla_\mu(z,y)\right],$$

$$\nabla_\mu(x,y) = \frac{1}{a}\left(U_\mu(x)\,\frac{1}{a^4}\,\delta_{x+a\hat{\mu},y} - \mathbb{1}\,\frac{1}{a^4}\,\delta_{x,y}\right), \quad \nabla_\mu^*(x,y) = \frac{1}{a}\left(\mathbb{1}\,\frac{1}{a^4}\,\delta_{x,y} - U_\mu^\dagger(x-a\hat{\mu})\,\frac{1}{a^4}\,\delta_{x-a\hat{\mu},y}\right),$$

$$\mathcal{D}_\mu A_+(x) = \frac{1}{a}[U_\mu(x)\,A_+(x+a\hat{\mu}) - A_+(x)], \quad \mathcal{D}_\mu A_+^\dagger(x) = \frac{1}{a}\left[A_+^\dagger(x+a\hat{\mu})\,U_\mu^\dagger(x) - A_+^\dagger(x)\right],$$

$$\mathcal{D}_\mu A_-(x) = \frac{1}{a}\left[A_-(x+a\hat{\mu})\,U_\mu^\dagger(x) - A_-(x)\right], \quad \mathcal{D}_\mu A_-^\dagger(x) = \frac{1}{a}\left[U_\mu(x)\,A_-^\dagger(x+a\hat{\mu}) - A_-^\dagger(x)\right],$$

$$\Psi^\alpha = \quad i\sqrt{2}g(A_+^\dagger P_+ + A_- P_-)T^\alpha\psi + \frac{a\,g^2}{2}Q^{\alpha\beta}\lambda^\beta, \quad \bar{\Psi}^\alpha = -i\sqrt{2}g\,\bar{\psi}T^\alpha(A_+P_- + A_-^\dagger P_+) + \frac{a\,g^2}{2}\bar{\lambda}^\beta Q^{\beta\alpha},$$

$$Q^{\alpha\beta} = (A_+^\dagger\{T^\alpha,T^\beta\}A_-^\dagger)P_+ + (A_-\{T^\alpha,T^\beta\}A_+)P_-, \quad \Xi \equiv \Xi_+\,P_+ + \Xi_-\,P_-, \quad \Xi^{\alpha\beta} = -\frac{\rho^\lambda}{a}\,\left(\delta^{\alpha\beta} - \frac{a^2\,g^2}{2\,\rho^\lambda}\,Q^{\alpha\beta}\right).$$

Note that the bare couplings g appearing in Eq. (F1) are not identical on the lattice. The coupling in the covariant derivatives and in the gluon action (coupling to the gluon fields) is the gauge coupling. The quartic terms in the fourth line define the quartic coupling, while the fifth line defines the Yukawa coupling. The Yukawa coupling also appears in the last three lines, which arise from functional integration over the auxiliary fields following the procedure of Ref. [13], introduced to render the Yukawa sector chiral invariant as well.

The total SQCD action in the overlap lattice formulation contains these three additional terms, as shown in Ref. [13], and reads:

$$\mathcal{S}_{\rm SQCD}^{\rm Total} = \mathcal{S}_{\rm SQCD}^{L} + \mathcal{S}_{\lambda\lambda} + \mathcal{S}_{\rm Pf} + \mathcal{S}_{\Psi\Psi}\,. \tag{F2}$$

Each of the additional contributions can be expanded perturbatively in the Yukawa coupling. The first contribution, $\mathcal{S}_{\lambda\lambda}$, is a monomial in g and is thus already in a form suitable for perturbation theory:

$$\mathcal{S}_{\lambda\lambda} \equiv a^4\sum_x \frac{a\,g^2}{2}\,\bar{\lambda}^\alpha\left[(A_+^\dagger\{T^\alpha,T^\beta\}A_-^\dagger)P_+ + (A_-\{T^\alpha,T^\beta\}A_+)P_-\right]\lambda^\beta\,. \tag{F3}$$

---

[5] Note that the overlap and the Wilson parameters for the gluino fields, ρ^λ and r^λ respectively, are the same as $a\,\mu$ in Ref. [12]

The Pfaffian contribution admits the expansion:

$$\begin{aligned}\mathcal{S}_{\rm Pf} \;\equiv\; & -\sum_x \Big(\mathrm{Tr}\log(a\,\Xi_+(x)) + \mathrm{Tr}\log(a\,\Xi_-(x))\Big) = \sum_x \frac{a^2\,g^2}{2\rho^\lambda}\Big(\frac{N_c^2-1}{N_c}\Big)\Big((A_+^{\dagger f}A_-^{\dagger f}) + (A_-^f A_+^f)\Big) \\ & + \frac{a^4\,g^4}{16(\rho^\lambda)^2}\sum_x\Big[\Big((A_+^{\dagger f}A_-^{\dagger f})(A_+^{\dagger f'}A_-^{\dagger f'}) + (A_-^f A_+^f)(A_-^{f'}A_+^{f'})\Big)\Big(\frac{N_c^2+2}{N_c^2}\Big) \\ & + \Big((A_+^{\dagger f}A_-^{\dagger f'})(A_+^{\dagger f'}A_-^{\dagger f}) + (A_-^f A_+^{f'})(A_-^{f'}A_+^f)\Big)\Big(\frac{N_c^2-4}{N_c}\Big)\Big] + \mathcal{O}(g^6)\,. \end{aligned} \tag{F4}$$

Further, in perturbation theory, it is necessary to introduce a discrete version of the gauge-fixing term into the action. This term is expressed in terms of the gauge field, $u_\mu(x)$:

$$S_{GF}^L = \frac{1}{2\alpha}a^2\sum_x\sum_\mu \mathrm{Tr}\left(u_\mu(x+a\hat{\mu}/2) - u_\mu(x-a\hat{\mu}/2)\right)^2. \tag{F5}$$

In addition, covariant gauge fixing produces the following action for the ghost fields c and $\bar{c}$:

$$\begin{aligned} S_{Ghost}^L \;=\; & 2a^2\sum_x\sum_\mu \mathrm{Tr}\{(\bar{c}(x+a\hat{\mu}) - \bar{c}(x))(c(x+a\hat{\mu}) - c(x) \\ & + ig[u_\mu(x+a\hat{\mu}/2), c(x)] + \frac{1}{2}ig[u_\mu(x+a\hat{\mu}/2), c(x+a\hat{\mu}) - c(x)] \\ & - \frac{1}{12}g^2[u_\mu(x+a\hat{\mu}/2), [u_\mu(x+a\hat{\mu}/2), c(x+a\hat{\mu}) - c(x)]])\} + \mathcal{O}(g^3). \end{aligned} \tag{F6}$$

These two terms must be added to the action, in order to avoid divergences from the integration over gauge orbits; they are the same as in the non-supersymmetric case. Furthermore, an additional term must be added to the action, Eq. (F1), in order to account for the Jacobian in the change of integration variables: $U_\mu \to u_\mu$. This term is the standard "measure" term S_M^L in the action and, to lowest order in g, it reads:

$$S_M^L = \frac{g^2 N_c}{12}a^2\sum_x\sum_\mu \mathrm{Tr}\left(u_\mu(x+a\hat{\mu}/2)\right)^2 + \mathcal{O}(g^4). \tag{F7}$$

In this work we introduce the overlap parameters ρ^λ, ρ^ψ and the Wilson parameters r^λ, r^ψ for the gluino and quarks, respectively. The range of admissible values for ρ^λ is: $0 < \rho^\lambda < 2r^\lambda$, and similarly for ρ^ψ: $0 < \rho^\psi < 2r^\psi$. Smaller values of of ρ^λ, ρ^ψ are not allowed, since then the propagator would have no poles; also, larger values of ρ^λ, ρ^ψ are forbidden, since they would lead to spurious poles at momenta $q = \pi/a$. We will later specialize to the conventional choices $r^\lambda = \rho^\lambda$ (denoted as μ in [12]) and $r^\psi = 1$.

The third contribution, $\mathcal{S}_{\Psi\Psi}$, is defined in the last line of Eq. (F1). To obtain its perturbative form (up to terms of $\mathcal{O}(g^6)$), it suffices to insert $\Xi^{-1} = (\Xi_+)^{-1}P_+ + (\Xi_-)^{-1}P_-$ into $\mathcal{S}_{\Psi\Psi}$ and expand to first subleading order:

$$\left((\Xi_+)^{-1}\right)^{\alpha\beta} = -\frac{1}{\mu}\,\delta^{\alpha\beta} - \frac{a\,g^2}{2\mu^2}\,A_+^\dagger\{T^\alpha, T^\beta\}A_-^\dagger + \mathcal{O}(g^4)\,, \quad \left((\Xi_-)^{-1}\right)^{\alpha\beta} = -\frac{1}{\mu}\,\delta^{\alpha\beta} - \frac{a\,g^2}{2\mu^2}\,A_-\{T^\alpha, T^\beta\}A_+ + \mathcal{O}(g^4)\,. \tag{F8}$$

In the following, we present the overlap actions for the quark fields. A more detailed formulation in the case of Supersymmetric QCD (SQCD) is available in our paper. The main drawback of the overlap action is its intrinsically non-ultralocal character. As a consequence, both numerical simulations and perturbative studies become significantly more challenging and computationally intensive, requiring considerable human effort and computing resources.

For simplicity and pedagogical clarity, we focus on the overlap action for massless quarks to illustrate how the vertices are extracted in this case:

$$S_{\rm overlap} = a^8\sum_{x,y}\bar{\Psi}(x)\,D_N(x,y)\,\Psi(y) \tag{F9}$$

where $D_N(x,y)$ is the overlap-Dirac operator

$$D_N(x,y) \;=\; \rho\left[\frac{\delta_{x,y}}{a^4} - \left(X\frac{1}{\sqrt{X^\dagger X}}\right)_{xy}\right], \qquad X = \frac{1}{a^4}\left(D_W - \rho\right) \tag{F10}$$

and D_W is the Wilson-Dirac operator

$$D_{\rm W} = \frac{1}{2}\left[\gamma_\mu\left(\nabla_\mu^* + \nabla_\mu\right) - a\nabla_\mu^*\nabla_\mu\right], \qquad \nabla_\mu\psi(x) = \frac{1}{a}\left[U(x,\mu)\psi(x+a\hat{\mu}) - \psi(x)\right] \tag{F11}$$

The overlap parameter ρ is restricted by the condition $0 < \rho < 2$ to guarantee the correct pole structure of D_N. The coupling constant is included in the link variables, present in the definition of X, and one must take the perturbative expansion of X in powers of g_0. This expansion in momentum space takes the form

$$X(p',p) = \underbrace{\chi_0(p)(2\pi)^4\delta_P(p'-p)}_{tree-level} + \underbrace{X_1(p',p) + X_2(p',p)}_{1-loop} + \underbrace{X_3(p',p) + X_4(p',p)}_{2-loop} + O(g_0^5) \tag{F12}$$

where χ_0 is the inverse fermion propagator and X_i are the vertices of the Wilson fermion action (p (p'): fermion (antifermion) momentum). The construction of overlap vertices (up to two-loops) make use of χ_0 and $X_1 - X_4$; these quantities can be written in the compact form:

$$\begin{aligned}
\chi_0(p) &= \frac{i}{a}\sum_\mu \gamma_\mu \sin(ap_\mu) + \frac{r}{a}\sum_\mu \Big(1-\cos(ap_\mu)\Big) - \frac{\rho}{a} \\
X_1(p',p) &= g_0 \int d^4k \delta(p'-p-k) \sum_\mu A_\mu(k) V_{1,\mu}\Big(\frac{p'+p}{2}\Big) \\
X_2(p',p) &= \frac{g_0^2}{2}\int \frac{d^4k_1 d^4k_2}{(2\pi)^4}\delta(p'-p-k_1-k_2)\sum_\mu A_\mu(k_1)A_\mu(k_2)V_{2,\mu}\Big(\frac{p'+p}{2}\Big) \\
X_3(p',p) &= \frac{g_0^3}{3!}\int \frac{d^4k_1 d^4k_2 d^4k_3}{(2\pi)^8}\delta(p'-p-\sum_{i=1}^3 k_i)\sum_\mu \prod_{i=1}^3 A_\mu(k_i)\Big[-a^2 V_{1,\mu}\Big(\frac{p'+p}{2}\Big)\Big] \\
X_4(p',p) &= \frac{g_0^4}{4!}\int \frac{d^4k_1 d^4k_2 d^4k_3 d^4k_4}{(2\pi)^{12}}\delta(p'-p-\sum_{i=1}^4 k_i)\sum_\mu \prod_{i=1}^4 A_\mu(k_i)\Big[-a^2 V_{2,\mu}\Big(\frac{p'+p}{2}\Big)\Big]
\end{aligned} \tag{F13}$$

where

$$V_{1,\,\mu}(p) = i\,\gamma_\mu \cos(ap_\mu) + r\,\sin(ap_\mu), \quad V_{2,\,\mu}(p) = -i\,\gamma_\mu a\,\sin(ap_\mu) + a\,r\,\cos(ap_\mu) \tag{F14}$$

A_μ represents a gluon field; later on we will have to generalize Eqs. (F13) to the case where both background and quantum gluon fields are present, see Eq. (F22).

At this point we can proceed with the perturbative expansion of D_N in powers of g_0. This leads to the propagator of zero mass fermions and to gluon-fermion-antifermion vertices (with up to 4 gluons). After analytical manipulations (an essential step is the expansion of $1/\sqrt{X^\dagger X}$ using complex analysis, which is presented in Ref. [3]), the overlap-Dirac operator is expanded into terms with up to 4 gluons as:

$$D_N(k_1,k_2) = D_0(k_1)\,(2\pi)^4\,\delta^4(k_1-k_2) + \Sigma(k_1,k_2) \tag{F15}$$

$D_0(k_1)$ is the inverse propagator,

$$D_0(k_1) = 1 + \frac{\chi_0(k_1)}{\omega(k_1)}, \qquad \omega(p) = \sqrt{\Big(\sum_\mu \sin^2(p_\mu)\Big) + \Big(\rho - 2r\sum_\mu \sin^2(p_\mu/2)\Big)^2} \tag{F16}$$

and

$$\begin{aligned}
\frac{\Sigma(k_1,k_2)}{\rho} &= \underbrace{V_1^1(k_1,k_2)}_{1-\text{gluon vertex}} + \underbrace{V_1^2(k_1,k_2) + V_2^2(k_1,k_2)}_{2-\text{gluon vertex}} \\
&+ \underbrace{V_1^3(k_1,k_2) + V_2^3(k_1,k_2) + V_3^3(k_1,k_2)}_{3-\text{gluon vertex}} \\
&+ \underbrace{V_1^4(k_1,k_2) + V_2^4(k_1,k_2) + V_3^4(k_1,k_2) + V_4^4(k_1,k_2)}_{4-\text{gluon vertex}} + \mathcal{O}(g_0^5)
\end{aligned} \tag{F17}$$

where we have set $a = 1$. V_1^i,V_2^i, V_3^i are given below as well as the expression for V_4^4.

$$V_1^i(k_1 \quad, k_2) = \frac{1}{\omega(k_1)+\omega(k_2)}\left[X_i(k_1,k_2) - \frac{1}{\omega(k_1)\omega(k_2)}\chi_0(k_1)\,X_i^\dagger(k_1,k_2)\,\chi_0(k_2)\right] \tag{F18}$$

$$\begin{aligned} V_2^i(k_1 \quad, k_2) = &\int \frac{d^4k_3}{(2\pi)^4}\,\frac{1}{\omega(k_1)+\omega(k_3)}\,\frac{1}{\omega(k_1)+\omega(k_2)}\,\frac{1}{\omega(k_2)+\omega(k_3)}\,\times \\ &\sum_{\substack{\{j>0,k>0\}\\ \{j+k=i\}}}\Big[-X_j(k_1,k_3)\,\chi_0^\dagger(k_3)\,X_k(k_3,k_2) \\ &-X_j(k_1,k_3)\,X_k^\dagger(k_3,k_2)\,\chi_0(k_2) \\ &-\chi_0(k_1)\,X_j^\dagger(k_1,k_3)\,X_k(k_3,k_2) \\ &+\frac{\omega(k_1)+\omega(k_2)+\omega(k_3)}{\omega(k_1)\omega(k_2)\omega(k_3)}\chi_0(k_1)X_j^\dagger(k_1,k_3)\chi_0(k_3)X_k^\dagger(k_3,k_2)\chi_0(k_2)\Big] \end{aligned} \tag{F19}$$

$$\begin{aligned} V_3^i(k_1 \quad, k_2) = &\int\int \frac{d^4k_3}{(2\pi)^4}\,\frac{d^4k_4}{(2\pi)^4}\,\frac{1}{4}\left(\prod_{p\epsilon S_4}\frac{1}{\omega(k_{p_1})+\omega(k_{p_2})}\right)\times \\ &\sum_{\substack{\{j>0,k>0,l>0\}\\ \{j+k+l=i\}}}\Big[-\frac{1}{6}\left(\sum_{p\epsilon S_4}\omega(k_{p_1})\omega(k_{p_2})\omega(k_{p_3})\right)X_j(k_1,k_3)X_k^\dagger(k_3,k_4)X_l(k_4,k_2) \\ &+\Big(\omega(k_1)+\omega(k_3)+\omega(k_4)+\omega(k_2)\Big)\times \\ &\Big[\chi_0(k_1)X_j^\dagger(k_1,k_3)X_k(k_3,k_4)X_l^\dagger(k_4,k_2)\chi_0(k_2)+\chi_0(k_1)X_j^\dagger(k_1,k_3)X_k(k_3,k_4)\chi_0^\dagger(k_4)X_l(k_4,k_2) \\ &+\chi_0(k_1)X_j^\dagger(k_1,k_3)\chi_0(k_3)X_k^\dagger(k_3,k_4)X_l(k_4,k_2)+X_j(k_1,k_3)\chi_0^\dagger(k_3)X_k(k_3,k_4)X_l^\dagger(k_4,k_2)\chi_0(k_2) \\ &+X_j(k_1,k_3)\chi_0^\dagger(k_3)X_k(k_3,k_4)\chi_0^\dagger(k_4)X_l(k_4,k_2)+X_j(k_1,k_3)X_k^\dagger(k_3,k_4)\chi_0(k_4)X_l^\dagger(k_4,k_2)\chi_0(k_2)\Big] \\ &-\left(\sum_{p\epsilon S_4}\frac{\omega(k_{p_1})\omega(k_{p_2})\Big(\omega(k_{p_1})/2+\omega(k_{p_3})/3\Big)}{\omega(k_1)\omega(k_3)\omega(k_4)\omega(k_2)}\right)\times \\ &\chi_0(k_1)X_j^\dagger(k_1,k_3)\chi_0(k_3)X_k^\dagger(k_3,k_4)\chi_0(k_4)X_l^\dagger(k_4,k_2)\chi_0(k_2)\Big] \end{aligned} \tag{F20}$$

(S_4: permutation group of the numbers $\{1,2,3,4\}$)

The use of the background field technique implies that instead of the generic gluonic fields (appearing in X_i's, Eq. (F13)), one must consider all possible combinations of background (A) and quantum (Q) fields which originate in the links of Eqs. (F11), (F21). In the lattice version of the background field technique, the link variable takes the form:

$$U_\mu(x) = e^{iag_0Q_\mu(x)}\cdot e^{iaA_\mu(x)} \tag{F21}$$

Hence,

$$\begin{aligned} X_1(p',p) &= X_1^{Q}(p',p) + X_1^{A}(p',p) \\ X_2(p',p) &= X_2^{QQ}(p',p) + X_2^{QA}(p',p) + X_2^{AA}(p',p) \\ X_3(p',p) &= X_3^{QQQ}(p',p) + X_3^{QQA}(p',p) + X_3^{QAA}(p',p) + X_3^{AAA}(p',p) \\ X_4(p',p) &= X_4^{QQQQ}(p',p) + X_4^{QQQA}(p',p) + X_4^{QQAA}(p',p) + X_4^{QAAA}(p',p) + X_4^{AAAA}(p',p) \end{aligned} \tag{F22}$$

As an example, let us write the expression for X_3^{QQA}

$$X_3^{QQA}(p', \quad p) = \frac{g_0^2}{4}\int \frac{d^4k_1 d^4k_2 d^4k_3}{(2\pi)^8}\delta(p'-p-k_1-k_2-k_3)\times$$
$$\sum_\mu \Bigg[-a^2 V_{1,\mu}\Big(\frac{p'+p}{2}\Big)\Big(Q_\mu(k_1)Q_\mu(k_2)A_\mu(k_3)+A_\mu(k_3)Q_\mu(k_2)Q_\mu(k_1)\Big)$$
$$+ iaV_{2,\mu}\Big(\frac{p'+p}{2}\Big)\Big(Q_\mu(k_1)Q_\mu(k_2)A_\mu(k_3)-A_\mu(k_3)Q_\mu(k_2)Q_\mu(k_1)\Big)\Bigg] \tag{F23}$$

Upon substituting the expression for X_i's in the overlap vertices, the latter become extremely lengthy and complicated.

For further information, here we will explain the basic idea of expanding the expressions $1/\sqrt{X^\dagger X}$, appearing in the overlap-Dirac operator, in powers of g_0, using a procedure introduced by Y. Kikukawa and A. Yamada. We also provide the expression for $V_4^4(k_1,k_2)$ further below.

In an integral representation, the combination $\dfrac{1}{\sqrt{X^\dagger X}}$ can be written as

$$\frac{1}{\sqrt{X^\dagger X}} = \int_{-\infty}^{\infty}\frac{dt}{\pi}\frac{1}{t^2+X^\dagger X} \tag{F24}$$

In fact, Eq. (F24) is valid for any operator X provided that $X^\dagger X$ has no vanishing eigenvalues. We begin the desired expansion of the overlap-Dirac operator in powers of g_0, by setting

$$X^\dagger X = \underbrace{X_0^\dagger X_0}_{\mathcal{O}(g_0^0)} + Z \tag{F25}$$

The first term of Eq. (F25) corresponds to the inverse fermionic propagator, while Z leads to the vertices; for our 2-loop calculation we need to write Z up to $\mathcal{O}(g_0^4)$

$$Z = \underbrace{(X_0^\dagger X_1 + X_1^\dagger X_0)}_{\mathcal{O}(g_0^1)} + \underbrace{(X_0^\dagger X_2 + X_1^\dagger X_1 + X_2^\dagger X_0)}_{\mathcal{O}(g_0^2)} + \underbrace{(X_0^\dagger X_3 + X_1^\dagger X_2 + X_2^\dagger X_1 + X_3^\dagger X_0)}_{\mathcal{O}(g_0^3)}$$
$$+ \underbrace{(X_0^\dagger X_4 + X_1^\dagger X_3 + X_2^\dagger X_2 + X_3^\dagger X_1 + X_4^\dagger X_0)}_{\mathcal{O}(g_0^4)} + \mathcal{O}(g_0^5) \tag{F26}$$

Using the above equations, we write the denominator on the r.h.s. of Eq. (F24) as

$$\frac{1}{t^2+X^\dagger X} = \frac{1}{t^2+X_0^\dagger X_0}\Big(1 - Z\frac{1}{t^2+X_0^\dagger X_0} + Z\frac{1}{t^2+X_0^\dagger X_0}Z\frac{1}{t^2+X_0^\dagger X_0} + ...\Big) \tag{F27}$$

From this point forward it is easier to work in momentum space since, taking the Fourier transform, the denominator becomes diagonal

$$F.T.[\frac{1}{t^2+X_0^\dagger X_0}] = \frac{1}{t^2+\omega^2(p)} \tag{F28}$$

($\omega^2(p)$ defined in Eq. (F16)). Combining Eqs. (F24) and (F27) we derive the Taylor expansion of $\dfrac{1}{\sqrt{X^\dagger X}}$ in momentum space

$$\frac{1}{\sqrt{X^\dagger X}}_{F.T.}(p',p) = \int_{-\infty}^{\infty}\frac{dt}{\pi}\frac{2\pi\delta(p'-p)}{t^2+\omega^2(p')} - \int_{-\infty}^{\infty}\frac{dt}{\pi}\frac{1}{t^2+\omega^2(p')}Z(p',p)\frac{1}{t^2+\omega^2(p)}$$
$$+ \int_{-\infty}^{\infty}\frac{dt}{\pi}\int_{-\infty}^{\infty}\frac{dk}{(2\pi)^4}\frac{1}{t^2+\omega^2(p')}Z(p',k)\frac{1}{t^2+\omega^2(k)}Z(k,p)\frac{1}{t^2+\omega^2(p)} + ... \tag{F29}$$

The integral over t can now be performed by closing the contour around the upper complex t-plane. Considering, as an example the third term on the r.h.s. of Eq. (F29), the result is

$$\int_{-\infty}^{\infty}\frac{dt}{\pi}\frac{1}{t^2+\omega^2(p')}\frac{1}{t^2+\omega^2(k)}\frac{1}{t^2+\omega^2(p)} = \frac{\omega(p')+\omega(k)+\omega(p)}{\omega(p')\omega(k)\omega(p)\,[\omega(p')+\omega(k)]\,[\omega(k)+\omega(p)]\,[\omega(p)+\omega(p')]}$$

Similarly we integrate all terms of Eq. (F29) over t and this leads to Eqs. (F16)-(F20) and to the expression for $V_4^4(k_1,k_2)$ which is presented below

$$
\begin{aligned}
V_4^4 \quad (k_1,k_2) = & \int\int\int \frac{d^4k_3}{(2\pi)^4}\frac{d^4k_4}{(2\pi)^4}\frac{d^4k_5}{(2\pi)^4}\frac{1}{12}\left(\prod_{p\epsilon S_5}\frac{1}{\omega(k_{p_1})+\omega(k_{p_2})}\right)\times \\
& \Bigg[\left(\sum_{p\epsilon S_5}\omega(k_{p_1})\omega(k_{p_2})\omega(k_{p_3})\omega(k_{p_4})\Big(\omega(k_{p_1})/6+\omega(k_{p_5})/30\Big)\right)\times \\
& \Big[(X_1(k_1,k_3)X_1^\dagger(k_3,k_4)X_1(k_4,k_5)X_1^\dagger(k_5,k_2)\chi_0(k_2) \\
& +X_1(k_1,k_3)X_1^\dagger(k_3,k_4)X_1(k_4,k_5)\chi_0^\dagger(k_5)X_1(k_5,k_2) \\
& +X_1(k_1,k_3)X_1^\dagger(k_3,k_4)\chi_0(k_4)X_1^\dagger(k_4,k_5)X_1(k_5,k_2) \\
& +X_1(k_1,k_3)\chi_0^\dagger(k_3)X_1(k_3,k_4)X_1^\dagger(k_4,k_5)X_1(k_5,k_2) \\
& +\chi_0(k_1)X_1^\dagger(k_1,k_3)X_1(k_3,k_4)X_1^\dagger(k_4,k_5)X_1(k_5,k_2)\Big] \\
& -\frac{1}{6}\left(\sum_{p\epsilon S_5}\omega(k_{p_1})\omega(k_{p_2})\Big(\omega(k_{p_1})+\omega(k_{p_3})\Big)\right)\times \\
& \Big[X_1(k_1,k_3)\chi_0^\dagger(k_3)X_1(k_3,k_4)X_1^\dagger(k_4,k_5)\chi_0(k_5)X_1^\dagger(k_5,k_2)\chi_0(k_2) \\
& +X_1(k_1,k_3)\chi_0^\dagger(k_3)X_1(k_3,k_4)\chi_0^\dagger(k_4)X_1(k_4,k_5)\chi_0^\dagger(k_5)X_1(k_5,k_2) \\
& +X_1(k_1,k_3)\chi_0^\dagger(k_3)X_1(k_3,k_4)\chi_0^\dagger(k_4)X_1(k_4,k_5)X_1^\dagger(k_5,k_2)\chi_0(k_2) \\
& +X_1(k_1,k_3)X_1^\dagger(k_3,k_4)\chi_0(k_4)X_1^\dagger(k_4,k_5)\chi_0(k_5)X_1^\dagger(k_5,k_2)\chi_0(k_2) \\
& +\chi_0(k_1)X_1^\dagger(k_1,k_3)\chi_0(k_3)X_1^\dagger(k_3,k_4)\chi_0(k_4)X_1^\dagger(k_4,k_5)X_1(k_5,k_2) \\
& +\chi_0(k_1)X_1^\dagger(k_1,k_3)\chi_0(k_3)X_1^\dagger(k_3,k_4)X_1(k_4,k_5)\chi_0^\dagger(k_5)X_1(k_5,k_2) \\
& +\chi_0(k_1)X_1^\dagger(k_1,k_3)\chi_0(k_3)X_1^\dagger(k_3,k_4)X_1(k_4,k_5)X_1^\dagger(k_5,k_2)\chi_0(k_2) \\
& +\chi_0(k_1)X_1^\dagger(k_1,k_3)X_1(k_3,k_4)\chi_0^\dagger(k_4)X_1(k_4,k_5)\chi_0^\dagger(k_5)X_1(k_5,k_2) \\
& +\chi_0(k_1)X_1^\dagger(k_1,k_3)X_1(k_3,k_4)\chi_0^\dagger(k_4)X_1(k_4,k_5)X_1^\dagger(k_5,k_2)\chi_0(k_2) \\
& +\chi_0(k_1)X_1^\dagger(k_1,k_3)X_1(k_3,k_4)X_1^\dagger(k_4,k_5)\chi_0(k_5)X_1^\dagger(k_5,k_2)\chi_0(k_2)\Big] \\
& +\left(\sum_{p\epsilon S_5}\frac{\omega^2(k_{p_1})\omega(k_{p_2})\omega(k_{p_3})}{\omega(k_1)\omega(k_2)\omega(k_3)\omega(k_4)\omega(k_5)}\times\right. \\
& \left.\Big(\omega(k_{p_2})[\omega(k_{p_1})/2+\omega(k_{p_3})/6]+\omega(k_{p_4})[\omega(k_{p_1})/3+\omega(k_{p_2})+\omega(k_{p_5})/3]\Big)\right)\times \\
& \chi_0(k_1)X_1^\dagger(k_1,k_3)\chi_0(k_3)X_1^\dagger(k_3,k_4)\chi_0(k_4)X_1^\dagger(k_4,k_5)\chi_0(k_5)X_1^\dagger(k_5,k_2)\chi_0(k_2)\Bigg]
\end{aligned}
\tag{F30}
$$

(S_5: permutation group of the numbers 1 through 5)

At this point, we present the tree-level propagators, including the massive quark and squark propagators. Note that the bare quark and squark masses, denoted by m, need not coincide; only the renormalized masses are required to match. A similar comment applies to the coupling constants: gauge invariance requires the equality of the gauge coupling appearing in the gluon action and in all covariant derivatives, whereas the Yukawa and four-squark couplings are, *a priori*, independent. Since our one-loop calculations involve only the tree-level values of these couplings, we will denote all coupling constants by the same symbol, g.

The tree-level propagators corresponding to all fields in the SQCD action are as follows:

$$\text{Quark propagator:} \quad D^{\psi}(q) = \frac{a}{\rho^{\psi}\left(1-\frac{am}{2}\right)} \frac{\phi\,\omega^{\psi}(q) - i\,\mathring{\not{q}} + b^{\psi}(q)}{(1+\phi^2)\,\omega^{\psi}(q) + 2\phi\,b^{\psi}(q)}, \tag{F31}$$

$$\text{Gluon propagator:} \quad D^{u}_{\mu\nu}(q) = \frac{1}{\hat{q}^2}\left(\delta_{\mu\nu} - (1-\alpha)\frac{\hat{q}_\mu \hat{q}_\nu}{\hat{q}^2}\right), \tag{F32}$$

$$\text{Ghost propagator:} \quad D^{c}(q) = \frac{1}{\hat{q}^2}, \tag{F33}$$

$$\text{Squark propagator:} \quad D^{A}(q) = \frac{1}{\hat{q}^2 + m^2}, \tag{F34}$$

$$\text{Gluino propagator:} \quad D^{\lambda}(q) = \frac{2a}{\rho^{\lambda}}\left(\frac{1}{2} - \frac{i\,\mathring{\not{q}}}{2\big(\omega^{\lambda}(q) + b^{\lambda}(q)\big)}\right). \tag{F35}$$

where:

$$\omega^{\psi}(q) = \frac{1}{a}\left[\sum_{\mu}\sin^2(aq_\mu) + \left(r^{\psi}\sum_{\mu}(1-\cos(aq_\mu)) - \rho^{\psi}\right)^2\right]^{1/2}, \quad b^{\psi}(q) = \frac{r^{\psi}}{a}\sum_{\mu}(1-\cos(aq_\mu)) - \frac{\rho^{\psi}}{a}, \tag{F36}$$

$$\phi = 1 - \frac{ma}{\rho^{\psi}\left(1-\frac{am}{2}\right)}, \qquad \mathring{\not{q}} = \frac{1}{a}\sum_{\mu}\gamma_\mu \sin(aq_\mu), \qquad \hat{q}_\mu = \frac{2}{a}\sin\Big(\frac{aq_\mu}{2}\Big), \qquad \hat{q}^2 = \sum_{\mu}\hat{q}^2_\mu, \tag{F37}$$

$$\omega^{\lambda}(q) = \frac{1}{a}\left[\sum_{\mu}\sin^2(aq_\mu) + \left(r^{\lambda}\sum_{\mu}(1-\cos(aq_\mu)) - \rho^{\lambda}\right)^2\right]^{1/2}, \quad b^{\lambda}(q) = \frac{r^{\lambda}}{a}\sum_{\mu}(1-\cos(aq_\mu)) - \frac{\rho^{\lambda}}{a}. \tag{F38}$$

Also, note that including the Wilson parameter for quarks r^{ψ} and for gluinos r^{λ} in the overlap operator means that the term $\sum_{\mu}(1-\cos(aq_\mu))$ in $\omega(q)$ and $b(q)$ is multiplied by r. This means that the propagator still has only one pole at $q=0$, with the correct residue (continuum limit). However, the range of values for ρ would be $0<\rho<2r$. Furthermore, the choice in Ref. [12] of using $r^{\lambda}=\rho^{\lambda}$ automatically satisfies this condition, since $r<2r$. By the way, let us recall that values of $\rho<0$ are not allowed, since then the propagator would have no poles ($\omega(q)+b(q)>0$ always); also $\rho>2r$ is not allowed since $\omega(q)+b(q)$ would have spurious poles at $q=\pi/a$.

The vertices coming from the action in Eq. (F2) can be found in the following file. For convenience, this file also includes the contractions needed to calculate the one-loop self-energies of the gluon, gluino, quark, and squark fields using the overlap action. If one is interested only in the SQCD sector (gluons, quarks, ghosts), one can calculate only the diagrams involving these fields. The file can be read as follows:

```
<< SUSY_vertices_overlap.m ;
```

Acknowledgments

The project (EXCELLENCE/0524/0208) is implemented under the programme of social cohesion "THALIA 2021-2027" co-funded by the European Union through the Research and Innovation Foundation (RIF).

[1] Chetyrkin, K. G., and Tkachov, F. V. Integration by parts: The algorithm to calculate beta-functions in 4 loops. *Nucl. Phys. B 192* (1981), 159.

[2] Chetyrkin, K. G., and Tkachov, F. V. Integration by parts: The algorithm to calculate beta-functions in 4 loops. *Nucl. Phys. B 192* (1981), 159–204.

[3] Constantinou, M., and Panagopoulos, H. Qcd with overlap fermions: Running coupling and the 3-loop beta-function. *Phys. Rev. D 76* (2007), 114504.

[4] Constantinou, M., and Panagopoulos, H. Gauge theories with overlap fermions in an arbitrary representation: Evaluation of the 3-loop beta-function. *Phys. Rev. D 77* (2008), 057503.

[5] Cvitanovic, P. Group theory for feynman diagrams in non-abelian gauge theories. *Phys. Rev. D 14* (1976), 1536–1553.

[6] Ellis, R. K., and Martinelli, G. Two loop corrections to the lambda parameters of one-plaquette actions. *Nucl. Phys. B 235* (1984), 93–114.

[7] Horsley, R., Perlt, H., Rakow, P. E. L., Schierholz, G., and Schiller, A. Perturbative determination of c(sw) for plaquette and symanzik gauge action and stout link clover fermions. *Phys. Rev. D 78* (2008), 054504.

[8] Kawai, H., Nakayama, R., and Seo, K. Comparison of the lattice lambda parameter with the continuum lambda parameter in massless qcd. *Nucl. Phys. B 189* (1981), 40.

[9] Luscher, M., and Weisz, P. Efficient Numerical Techniques for Perturbative Lattice Gauge Theory Computations. *Nucl. Phys. B 266* (1986), 309.

[10] Luscher, M., and Weisz, P. Background field technique and renormalization in lattice gauge theory. *Nucl. Phys. B 452* (1995), 213–233.

[11] Luscher, M., and Weisz, P. Two-loop relation between the bare lattice coupling and the ms coupling in pure su(n) gauge theories. *Nucl. Phys. B 452* (1995), 234–260.

[12] M. Clancy, D. B. K., and Singh, H. Generalized ginsparg-wilson relations. *Phys. Rev. D 109* (2024), 014502.

[13] M. Costa, E. I., and Panagopoulos, H. N=1 supersymmetric qcd on the lattice using overlap fermions. *Phys. Rev. D 112* (2025), 094506.

[14] Morningstar, C., and Peardon, M. J. Analytic smearing of su(3) link variables in lattice qcd. *Phys. Rev. D 69* (2004), 054501.

[15] Panagopoulos, G., Panagopoulos, H., and Spanoudes, G. Two-loop renormalization and mixing of gluon and quark energy-momentum tensor operators. *Phys. Rev. D 103*, 1 (2021), 014515.

[16] Panagopoulos, H., and Vicari, E. The trilinear gluon condensate on the lattice. *Nucl. Phys. B 332* (1990), 261.

[17] Peskin, M. E., and Schroeder, D. V. *An Introduction to Quantum Field Theory*. Addison-Wesley, Reading, MA, 1995.

[18] Skouroupathis, A., and Panagopoulos, H. Two-loop renormalization of vector, axial-vector and tensor fermion bilinears on the lattice. *Phys. Rev. D 79* (2009), 094508.

[19] 't Hooft, G., and Veltman, M. J. G. DIAGRAMMAR. *NATO Sci. Ser. B 4* (1974), 177–322.