ColorMath - A package for color summed calculations in SU($\boldsymbol{N_c}$)

Malin Sjödahl

Dept. of Astronomy and Theoretical Physics, Lund University
Sölvegatan 14A, 223 62 Lund, Sweden

Abstract

A Mathematica package for color summed calculations in QCD (SU(N_c)) is presented. Color contractions of any color amplitude appearing in QCD may be performed, and the package uses a syntax which is very similar to how color structure is written on paper. It also supports the definition of color vectors and bases, and special functions such as scalar products are defined for such color tensors.

1. Introduction

With the LHC follows an increased demand of exact calculations in QCD involving many color charged partons. Due to the non-abelian nature of QCD this poses a nontrivial computational problem.

This is the issue the Mathematica® package ColorMath[1] is set out to tackle. The main feature of ColorMath is thus the ability to automatically perform color summed calculations starting from a QCD color structure which is expressed using a syntax very similar to how the color structure would have been written on paper.

ColorMath allows both partial and full dummy index contractions, and thus deals with color structures having an arbitrary set of free indices. In this sense it is thus more general than [1], on the other hand it is a pure SU(N_c) tool, rather than a high energy physics general purpose tool, such as [2, 3]. ColorMath works for an arbitrary $N_c \geq 2$, and for arbitrary trace convention of the squared SU(N_c) generators, T_R. It may be used by just giving the color structure in the appropriate form and afterwards running **CSimplify** to contract all repeated indices. For example, consider $q_1 \bar{q}_2 \to q_3 \bar{q}_4$ via gluon exchange in the s- and t-channels. Writing down the amplitude as

$$\mathbf{Amp} = \mathbf{S}\,\mathbf{t}^{\{g\}q1}{}_{q2}\mathbf{t}^{\{g\}q4}{}_{q3} + \mathbf{T}\,\mathbf{t}^{\{g\}q1}{}_{q3}\mathbf{t}^{\{g\}q4}{}_{q2} \tag{1}$$

where **S** and **T** represent some s- and t-channel kinematics, and $\mathbf{t}^{\{g\}q1}{}_{q2}$ the SU(N_c) generator in the fundamental representation, otherwise typically denoted $(t^g)^{q1}{}_{q2}$, we may calculate the squared amplitude (defined as in the scalar product eq. (12)) using

$$\mathbf{CSimplify}[\mathbf{Conjugate}[\mathbf{Amp}]\,\mathbf{ReplaceDummyIndices}[\mathbf{Amp}]], \tag{2}$$
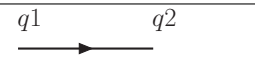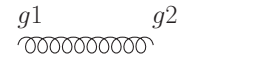
immediately giving the answer

$$T_R^2 \frac{(N_c^2 - 1)}{N_c}[(N_c S - T)S^* + (N_c T - S)T^*]. \tag{3}$$

Each squared amplitude can in principle can be calculated like this. However, ColorMath also facilitates the usage of color vectors and bases and has special functions for the calculations of scalar products and gluon exchanges.

---

[1] ColorMath is available at `www.thep.lu.se/~malin/ColorMath.html`.

This paper, which is intended to be the main reference, is organized as follows: First a general introduction to the basic color building blocks is given in section 2. In section 3 the computational strategy is presented, whereas basic examples are given in section 4, and the usage of vectors and their functions are presented in section 5. In section 6 some remarks concerning validation and scalability are made and in section 7 conclusions are drawn.
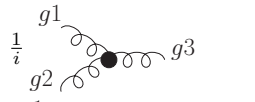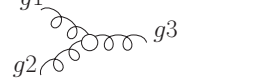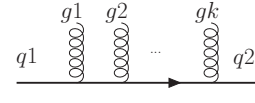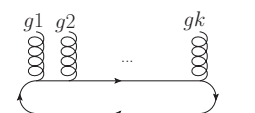
2. QCD building blocks

From a color perspective, the QCD Lagrangian is built out of

$$\text{quark-gluon vertices,} \qquad \qquad = (t^a)^i{}_j, \qquad\qquad (4)$$

and

$$\text{triple-gluon vertices,} \qquad \qquad = if^{abc}, \qquad\qquad (5)$$

where we follow the convention of reading of the fully antisymmetric structure constant indices in counterclockwise order. The four gluon vertices can be rewritten in terms of (one gluon) contracted triple-gluon vertices, and thus need no special treatment.

Due to confinement, we never observe individual colors and it therefore suffices to calculate color summed/averaged quantities for making predictions in quantum chromodynamics. We may thus constrain ourselves to treat QCD amplitudes carrying a set of external indices with values that need never be specified, as they are always summed over in the end.

In principle, the $SU(N_c)$ generators, along with a delta function for indicating a quark and an antiquark color singlet, $\delta^{q1}{}_{q2}$, constitute a minimal set of objects for treating the color structure in QCD[2]. For convenience, and for performance reasons, it is, however, useful to define a larger set of objects. The complete set of basic building blocks for carrying color structure used by the ColorMath package is given in table 1.

Apart from a delta function in quarks indices, $\delta^{q1}{}_{q2}$, a delta function in gluon indices, denoted by $\Delta^{\{g1,g2\}}$, is also defined. Note the Mathematica **List** brackets in $\{g1, g2\}$. The gluon delta function can alternatively be expressed using

$$\overset{g1}{} \overset{g2}{} = T_R \overset{g1}{} \overset{g2}{} \quad \Leftrightarrow \quad \text{tr}[t^{g1}t^{g2}] = T_R\,\delta^{g1g2} \quad \Leftrightarrow \quad \mathbf{o}^{\{g1,g2\}} = \mathbf{TR}\,\Delta^{\{g1,g2\}} \qquad (6)$$

where T_R (**TR**) comes from the normalization of the $SU(N_c)$ generators, and is typically taken to be $1/2$ (the Gell-Mann normalization) or 1. A rescaling of the normalization of the $SU(N_c)$ generators can always be absorbed into a normalization of the strong coupling constant. To allow for arbitrary normalization T_R is kept as a free parameter, denoted by **TR**, and may be defined by the user. Also the number of colors, denoted **Nc** in ColorMath, may be set by the user; by default both **Nc** and **TR** are kept as free parameters. Note the ColorMath notation for a trace over two gluons $\mathbf{o}^{\{g1,g2\}}$ in eq. (6). Similarly, a general trace over k gluons $g1, g2, ..gk$ is denoted $\mathbf{o}^{\{g1,g2,...,gk\}}$, and may be thought of as a closed quark-line with k gluons attached.

The totally antisymmetric structure constants, which – along with an extra i – define the triple gluon vertices, are denoted $\mathbf{f}^{\{g1,g2,g3\}}$. Similarly, the totally symmetric "structure constants" are defined as $\mathbf{d}^{\{g1,g2,g3\}}$. Recall that, starting from the commutation (anticommutation) relations

$$[t^{g1}, t^{g2}] = if^{g1\,g2\,g3}\,t^{g3}\ , \qquad \{t^{g1}, t^{g2}\} = d^{g1\,g2\,g3}\,t^{g3}, \qquad\qquad (7)$$

---

[2]To enhance the similarity with usage inside Mathematica, we here use the somewhat unorthodox notation $q1$ etc. for single quark and gluon indices.

Table 1: Along with the number of colors, **Nc**, and the trace of an SU(N_c) generator squared, **TR**, the basic building blocks are as below. Note that $\mathbf{o}^{\{g1,g2,...,gk\}}$ represents a trace over gluons with indices $g1...gk$, $\mathrm{tr}[t^{g1}t^{g2}...t^{gk}]$, and that $\mathbf{t}^{\{g1,g2,...,gk\}q1}{}_{q2}$ represent the $q1,q2$-component in a trace over generators that has been cut open, $(t^{g1}t^{g2}...t^{gk})^{q1}{}_{q2}$. For convenience, also the totally symmetric "structure constants" $\mathbf{d}^{\{g1,g2,g3\}}$ are defined. Note the Mathematica **FullForm**. ColorMath is built on pattern matching, and it is therefore essential that the expressions have the correct **FullForm**. In particular, **Power** may not be used instead of **Superscript**. To get the right **FullForm** it is recommended to use the function form, which is just a function returning the corresponding ColorMath object.

Pictorial representation	ColorMath	Function form	Mathematica **FullForm**
	$\delta^{q1}{}_{q2}$	$\delta[\mathbf{q1},\mathbf{q2}]$	**Subscript[Superscript[\[Delta], q1], q2]**
	$\Delta^{\{g1,g2\}}$	$\Delta[\mathbf{g1},\mathbf{g2}]$	**Superscript[\[CapitalDelta], List[g1, g2]]**
	$\mathbf{f}^{\{g1,g2,g3\}}$	$\mathbf{f}[\mathbf{g1},\mathbf{g2},\mathbf{g3}]$	**Superscript[f, List[g1, g2, g3]]**
	$\mathbf{d}^{\{g1,g2,g3\}}$	$\mathbf{d}[\mathbf{g1},\mathbf{g2},\mathbf{g3}]$	**Superscript[d ,List[g1, g2, g3]]**
	$\mathbf{t}^{\{g1,...,gk\}q1}{}_{q2}$	$\mathbf{t}[\{\mathbf{g1},\ldots,\mathbf{gk}\},\mathbf{q1},\mathbf{q2}]$	**Subscript[Superscript[Superscript[t, List[g1,...,gk]], q1], q2]**
	$\mathbf{o}^{\{g1,...,gk\}}$	$\mathbf{o}[\{\mathbf{g1},\ldots,\mathbf{gk}\}]$	**Superscript[o, List[g1, ..., gk]]**

the structure constants can be rewritten in terms of traces over SU(N_c) generators,

$$
\begin{aligned}
if^{g1\,g2\,g3}/d^{g1\,g2\,g3} &= \frac{1}{T_R}\big[\mathrm{tr}(t^{g1}t^{g2}t^{g3}) \mp \mathrm{tr}(t^{g2}t^{g1}t^{g3})\big] \\
&= \frac{1}{T_R}\big[(t^{g1})^{q1}{}_{q2}(t^{g2})^{q2}{}_{q3}(t^{g3})^{q3}{}_{q1} \mp (t^{g2})^{q1}{}_{q2}(t^{g1})^{q2}{}_{q3}(t^{g3})^{q3}{}_{q1}\big].
\end{aligned}
\tag{8}
$$

In ColorMath notation $\mathbf{t}$ is used to denote an open quark-line, and the above expression is written similarly,

$$
\begin{aligned}
\mathbf{I}\,\mathbf{f}^{\{g1,g2,g3\}}/\mathbf{d}^{\{g1,g2,g3\}} &= \frac{1}{\mathbf{TR}}\big(\mathbf{o}^{\{g1,g2,g3\}} \mp \mathbf{o}^{\{g2,g1,g3\}}\big) \\
&= \frac{1}{\mathbf{TR}}\big(\mathbf{t}^{\{g1\}q1}{}_{q2}\,\mathbf{t}^{\{g2\}q2}{}_{q3}\,\mathbf{t}^{\{g3\}q3}{}_{q1} \mp \mathbf{t}^{\{g2\}q1}{}_{q2}\,\mathbf{t}^{\{g1\}q2}{}_{q3}\,\mathbf{t}^{\{g3\}q3}{}_{q1}\big).
\end{aligned}
\tag{9}
$$

Pictorially this represents

$$
\text{} = \frac{1}{T_R}\left[\;\text{} - \text{}\;\right], \qquad \text{} = \frac{1}{T_R}\left[\;\text{} + \text{}\;\right], \tag{10}
$$

where i is included in the triple gluon vertex. The rationale for putting the gluon index in the SU(N_c) generators $\mathbf{t}^{\{g2\}q1}{}_{q2}$ inside a Mathematica **List** in eq. (9) is to allow for the natural extension of having many gluons attached to an open quark-line, thus

3

$$\mathbf{t}^{\{g1,g2,\ldots,gk\}q1}{}_{q2} = \mathbf{t}^{\{g1\}q1}{}_{d2}\mathbf{t}^{\{g2\}d2}{}_{d3}\ldots\mathbf{t}^{\{gk\}dk}{}_{q2}. \tag{11}$$

The left hand side has several advantages compared to the right hand side. An open quark-line with an arbitrary number of gluon indices can be written in a compact form with no dummy indices. This is not only more human readable, but also superior from a computational point of view, as it avoids the contraction of unnecessary dummy indices. A third advantage with the above notation is its direct correspondence to the trace type bases [4–12]. A basis (or spanning set) for the color space for a fixed set of external quarks, anti-quarks and gluons can always be taken to be a sum of products of open and closed quark-lines.

In this context we also remark that the color tensors defined in table 1 are color scalars, i.e., they are invariant under $\mathrm{SU}(N_c)$ transformations. This imposes no restriction for our purposes as, for any QCD amplitude, the overall color structure, including *both* incoming and outgoing particles, always is a color singlet. As the basic building blocks are invariant, each tensor built out of these objects, i.e., each tensor needed for color summed calculations in perturbative QCD is a actually a color scalar.

The scalar product on this vector space is given by summing over all external color indices, i.e.

$$\langle \mathbf{c}_1 | \mathbf{c}_2 \rangle = \sum_{a_1, a_2, \ldots} \mathbf{c}_1^{*a_1 a_2 \ldots} \mathbf{c}_2^{a_1 a_2 \ldots} \tag{12}$$

with $a_i = 1, \ldots, N_c$ if parton i is a quark or anti-quark and $a_i = 1, \ldots, N_c^2 - 1$ if parton i is a gluon. As long as the color structures in table 1 are multiplied by real coefficients the scalar product is actually real, which is easy to prove using the computational rules in the next section.

3. Basic computational strategy

Having defined all the color carrying objects, we turn to describing the basic strategy for carrying out calculations.

Again we note that we need not treat the four gluon vertex as this can be rewritten in terms of three gluon vertices. To treat an arbitrary color structure in QCD we may always compute any squared amplitude in the following way:

(i) Rewrite the triple gluon vertices using eq. (8). This results in a color structure which is a sum of products of open and closed quark-lines, connected to each other via repeated gluon indices.

(ii) Contract all internal gluon indices using the Fierz or completeness relation

$$\tag{13}$$

(iii) If present, remove Kronecker deltas (for quarks and gluons), and internal quark indices using

$$\tag{14}$$

(iv) Use the tracelessness of the $\mathrm{SU}(N_c)$ generators, and contract quark and gluon delta functions with repeated indices

4

$$\text{} \quad = \quad N_c^2 - 1. \tag{15}$$

After this, all dummy indices have been contracted. A squared amplitude is thus represented by a rational function in T_R and N_c, whereas a general amplitude is represented as a sum of products of closed and open quark-lines, i.e., as sums of products of $\mathbf{t}^{\{g1,g2,\ldots,gk\}q1}{}_{q2}$, including $\mathbf{t}^{\{\}q1}{}_{q2} = \delta^{q1}{}_{q2}$, and $\mathbf{o}^{\{g1,g2,\ldots,gk\}}$, which for two gluons may be rewritten as $\mathbf{o}^{\{g1,g2\}} = \mathbf{TR}\,\Delta^{\{g1,g2\}}$.

By applying these rules, *any* amplitude square and any interference term appearing in QCD can be calculated [6, 11]. To successfully square an arbitrary QCD amplitude using the above set of rules we see that steps (i-ii) have to be performed while keeping the relative order, i.e., by first applying rule (i) and then rule (ii). We also note that these rules increase the number of terms, whereas the rules (iii-iv) decrease the number of terms or keep it fixed. In order not to unnecessarily inflate an expression it may therefore be useful to apply the latter rules at any time during the computation. ColorMath utilizes this and tries to contract indices using (iii-iv) at any time, while rules (i-ii) are used only when needed, i.e., when the non-expanding rules fail.

On top of the rules (iii-iv) there are other cases in which the contraction of a gluon results in at most one term. This will happen if

(v) Two neighboring gluons attached to the same (closed or open) quark-line are contracted. In this case the result is simply $c_F = T_R(N_c^2 - 1)/N_c$ times the color structure where the two involved gluons have been contracted,

$$\text{} \quad = T_R \frac{N_c^2 - 1}{N_c} \quad \text{}. \tag{16}$$

(vi) Similarly, it is easy to show, using the Fierz identity, eq. (13), that the contraction of two next to neighboring gluons results in only one term

$$\text{} \quad = -T_R \frac{1}{N_c} \quad \text{}. \tag{17}$$

While the above rules are sufficient for squaring color amplitudes, it is sometimes advantageous to contract indices directly using the structure constants $f^{g1\,g2\,g3}$ and $d^{g1\,g2\,g3}$. Therefore rule (i) is *not* always automatically used for simplifying expressions. Instead, in addition to the above set of rules, a limited set of rules for contraction of repeated gluon indices occurring in the standard and the symmetric structure constants are implemented. More specifically, all rules for contracting two gluons in products of two symmetric or antisymmetric structure constants, for example,

$$f^{g1\,i1\,i2} f^{g2\,i1\,i2} = 2N_c\,T_R\,\delta^{g1\,g2} \tag{18}$$

and all rules for contracting three gluons in products of three symmetric or antisymmetric structure constants, such as

$$d^{g1\,i1\,i2} d^{g2\,i1\,i3} f^{g3\,i2\,i3} = T_R \frac{N_c^2 - 4}{N_c} f^{g1\,g2\,g3} \tag{19}$$

are implemented. For more intricate gluon contractions involving structure constants ColorMath can contract indices by first applying rule (i).

4. Basic calculations

In this section, basic functionality of ColorMath is explored. As always with Mathematica, the package has to be loaded before it can be used. With

$$\mathbf{Get}[\text{"/full/path/to/ColorMath1.0.m"}] \tag{20}$$

where the version number is adjusted to the version in question, this can be done. To start using the package, it is, however, recommended to start from (a version of) the tutorial "ColorMathTutorial.nb" and modify its content according to the desired usage.

The color contractions corresponding to the basic manipulations from section 3 are carried out using Mathematica **Rules**, i.e., a set of replacement rules based on pattern matching. As always, the rules may be applied using "**Expr/.TheRules**", and may be applied repeatedly using "**Expr//.TheRules**".

The rules described in (iii-iv) are, along with eq. (6) contained in a set of rules called **SimpleRules**, as they keep the expression at least as simple (in a term counting sense) as it initially was. Applying these to an expression thus tend to simplify it, for example

$$\mathbf{t}^{\{g1\}q1}{}_{q2}\,\mathbf{t}^{\{g2\}q2}{}_{q1}\,\Delta^{\{g1,g2\}}/.\mathbf{SimpleRules} \tag{21}$$

results in $\mathbf{TR}(\Delta^{\{g1,g2\}})^2$.

To fully utilize the rules we (may) need to apply them repeatedly,

$$\mathbf{t}^{\{g1\}q1}{}_{q2}\,\mathbf{t}^{\{g2\}q2}{}_{q1}\,\Delta^{\{g1,g2\}}//.\mathbf{SimpleRules} \tag{22}$$

giving the fully contracted expression $(-1 + \mathbf{Nc}^2)\mathbf{TR}$.

The rules defined in (v-vi), acting on $\mathbf{t}$ and $\mathbf{o}$, are similarly contained in **OTSimpleRules**, and the union of **SimpleRules** and **OTSimpleRules** and a few rules for rewriting closed quark-lines with zero to two gluons, are contained in **AllSimpleRules**.

The special rules for gluon index contraction in structure constants, exemplified in equation (18-19) are defined in **FDRules**. The complete set of rules are stated in table A.2.

Rather than thinking about how individual rules have to be applied, it is convenient to have a standard procedure for contracting color indices. This is embodied in the function **CSimplify**, which does what its name suggests; simplifies the color structure as far as possible. For color structures which do not contain structure constants this always implies contracting all repeated indices.

For color structure involving the structure constants **CSimplify** first attempts simplification using the **FDRules**. If, after this, the expression still contains structure constants, the structure constants are by default rewritten in terms of traces using eq. (9), and the indices are fully contracted, resulting in a sum of products of open and closed quark-lines. Sometimes it may, however, be desirable *not* to rewrite the structure constants, as expressions may be more compact if they are kept. This can be achieved by using the option **RemoveFD→ False**.

The most useful set of functions are given in table A.3. Apart from **CSimplify** we especially note the function **ReplaceDummyIndices** for replacing all repeated indices in a color structure with a new unique set of color indices.

For the purpose of calculating amplitudes square, we need, as in eq. (12), the complex conjugated version of the color structures. We note that

$$\mathbf{Conjugate}[\delta^{q1}{}_{q2}] = \delta^{q2}{}_{q1}$$
$$\mathbf{Conjugate}[\mathbf{t}^{\{g1,\ldots,gk\}q1}{}_{q2}] = \mathbf{t}^{\{gk,\ldots,g1\}q2}{}_{q1}$$
$$\mathbf{Conjugate}[\mathbf{o}^{\{g1,\ldots,gk\}}] = \mathbf{o}^{\{gk,\ldots,g1\}} \tag{23}$$

whereas $\Delta^{\{g1,g2\}}$, $\mathbf{f}^{\{g1,g2,g3\}}$ and $\mathbf{d}^{\{g1,g2,g3\}}$ are real. This is implemented in ColorMath via redefinition of the Mathematica's built in function **Conjugate**. With this in mind, we are ready to perform calculations of the type in eq. (2).

Finally we remark that to Mathematica $\Delta^{\{g1,g2\}} \neq \Delta^{\{g2,g1\}}$, and similarly, $\mathbf{o}^{\{g2,g3,g4,g1\}} \neq \mathbf{o}^{\{g1,g2,g3,g4\}}$ etc. For this reason a function **SortIndices**, which writes indices in Mathematica default order is defined, s.t. for example

$$\mathbf{SortIndices}[\mathbf{o}^{\{g2,g3,g4,g1\}}] = \mathbf{o}^{\{g1,g2,g3,g4\}}. \tag{24}$$

This function is used by **CSimplify**, and by some of the rules in table A.2.

Sometimes more detailed control over the calculation may be desired. For this purpose, and for internal usage, functions manipulating indices and probing the color structure are given in table A.4, in Appendix A.

While each squared amplitude in principle can be calculated as outlined above, ColorMath also offers more efficient tools for dealing with vectors in color space.

5. Defining and using vectors

For the purpose of studying color space it is often convenient to define a basis for the color space, and sometimes also projection operators. Both of these are examples of color (singlet) tensors, and ColorMath has a set of tools for working directly with such tensors.

As an example, let us consider the color structure for $q_1 \bar{q}_2 \to q_3 \bar{q}_4 g_5$[3]. A basis for the color space may be written as [13]

$$\mathbf{Vector181}_{\{q1\_,q2\_,q3\_,q4\_,g5\_\}} := \delta^{q1}{}_{q2}\, \mathbf{t}^{\{g5\}q4}{}_{q3}$$
$$\mathbf{Vector818}_{\{q1\_,q2\_,q3\_,q4\_,g5\_\}} := \mathbf{t}^{\{g5\}q1}{}_{q2}\, \delta^{q4}{}_{q3}$$
$$\mathbf{Vector888s}_{\{q1\_,q2\_,q3\_,q4\_,g5\_\}} := \mathbf{Module}[\{i1,i2\}, \mathbf{t}^{\{i1\}q1}{}_{q2}\, \mathbf{t}^{\{i2\}q4}{}_{q3}\, \mathbf{d}^{\{i1,i2,g5\}}]$$
$$\mathbf{Vector888a}_{\{q1\_,q2\_,q3\_,q4\_,g5\_\}} := \mathbf{Module}[\{i1,i2\}, \mathbf{t}^{\{i1\}q1}{}_{q2}\, \mathbf{t}^{\{i2\}q4}{}_{q3}\, \mathbf{I}\, \mathbf{f}^{\{i1,i2,g5\}}] \tag{25}$$

where the basis vectors are labeled using first the overall multiplet of $q_1\bar{q}_2$, then the overall multiplet of $q_3\bar{q}_4$, and finally the overall multiplet of $q_3\bar{q}_4 g_5$, which – due to color conservation – must equal the multiplet of $q_1\bar{q}_2$, (implying that the notation is somewhat redundant).

From a Mathematica perspective we note a few things. First, on the left hand side, we see that the indices inside the **List** in the **Subscript** are followed by underscore to indicate pattern matching. This is standard in Mathematica and makes it possible to use any symbol to denote the indices in later calculations. Then we note that the last two tensors in eq. (25) are defined using **Module**. This is to ensure that each time the tensor is used, it comes with a fresh set of dummy indices. This is also the reason why set delayed ":=" is used.

This basis is orthogonal since at least one set of partons transform under different representations in the various tensors. It is, however, not normalized. Finding the normalization is easy using ColorMath. To calculate for example **Vector181** squared we could enter

$$\mathbf{CSimplify}[\mathbf{Conjugate}[\mathbf{Vector181}_{\{q1,q2,q3,q4,g5\}}]\, \mathbf{Vector181}_{\{q1,q2,q3,q4,g5\}}], \tag{26}$$

but it is yet much easier to use the tensor functions for calculating scalar products. Instead we could simply write

$$\mathbf{CDot}[\mathbf{Vector181}, \mathbf{Vector181}] \tag{27}$$

---

[3]In this user guide indices representing incoming quarks and outgoing anti-quarks are placed upstairs, whereas outgoing quarks and incoming anti-quarks are placed downstairs. Note, however, that we could as well have used the opposite convention.

resulting in $\mathrm{Nc}\left(-1+\mathrm{Nc}^2\right)\mathrm{TR}$. Having a basis we naturally want to calculate all norms (square). ColorMath has special functions for this as well. If we define

$$\mathbf{OurBasis} = \{\mathbf{Vector181}, \mathbf{Vector818}, \mathbf{Vector888s}, \mathbf{Vector888a}\} \qquad (28)$$

we may calculate the squares of the basis vectors using

$$\mathbf{CDot[OurBasis]}. \qquad (29)$$

This results in a **List** containing the scalar products between each basis vector and itself

$$\left\{\mathrm{Nc}\left(-1+\mathrm{Nc}^2\right)\mathrm{TR}, \ \mathrm{Nc}\left(-1+\mathrm{Nc}^2\right)\mathrm{TR}, \ \frac{2\left(4-5\mathrm{Nc}^2+\mathrm{Nc}^4\right)\mathrm{TR}^3}{\mathrm{Nc}}, \ 2\mathrm{Nc}\left(-1+\mathrm{Nc}^2\right)\mathrm{TR}^3\right\}. \qquad (30)$$

Often non-orthogonal bases are used, in particular, this is the case for the trace type bases. In this case, to square a general amplitude, all the scalar products between all the basis vectors are needed. For calculating the scalar product matrix the command **CDotMatrix** may be used. This returns a **List** of **List**s, i.e. a matrix, where the ij-th element is the scalar product between the i-th and j-th vector in the list of (basis) vectors. For larger vector spaces with complicated scalar products, it may be desirable to get progress information on the calculations. This can be obtained by setting the option **Verbose $\to$ True** for **CDotMatrix**,

$$\mathbf{CDotMatrix[OurBasis, Verbose \to True]}. \qquad (31)$$

Similarly, it may be useful to be able to simplify potential (normalization) roots assuming a large **Nc**. This can be done by using the option **NcMin**. The scalar product functions, along with their options, are listed in table A.5.

Often it is also of interest to investigate the effect of gluon exchange on the color structure expressed in a basis, i.e. starting in a basis vector j, what is the effect on the basis vector of exchanging a gluon between parton1 and parton2. This is useful both for soft gluon resummation and for one-loop corrections via gluon exchange. The result can be expressed in terms of a matrix whose element ij is the i-th component resulting after such an exchange in the initial vector j. This is calculated by the function

$$\mathbf{CGamma[Basis, parton1, parton2]} \qquad (32)$$

where **Basis** is a **List** of basis vectors, defined using the syntax in eq. (25) and parton1 and parton2 are the numbers of the partons in the basis vectors, i.e., in this case numbers between 1 and 5. The sign conventions are such that quark-gluon vertex always comes without additional signs, and the triple-gluon vertex have the indices appearing in the order: external index, internal dummy index and index of the exchanged gluon[4]. For example, we may calculate the effect of gluon exchange between parton 1 and 3 in the above basis using

$$\mathbf{CGamma[OurBasis, 1, 3, Verbose \to False, BasisType \to OrthogonalBasis]}. \qquad (33)$$

Here we have supplied optional information about the basis type, that the basis is orthogonal, to speed up the calculations. By default **CGamma** does a few consistency checks. It checks that the vector resulting after gluon exchange, when squared, has the same value as the basis decomposed version. For orthonormal bases (**BasisType $\to$ OrthonormalBasis**) it is also checked that the resulting matrix is symmetric [14]. These checks may, however be turned off (**MakeChecks $\to$ False**). The set of options, with default values are listed in table A.5.

---

[4]The sign conventions thus differ from the typical eikonal choice.

6. Validation and scalability

The computational rules and functions in this package have been used for calculating the three gluon projection operators presented in [15]. This imposes highly nontrivial consistency checks, as it is verified that every projector square equals itself, which at intermediate steps often involve several ten thousand terms. Additional consistency checks on the color contraction rules have been made by using the **CGamma** function which checks that vectors square and basis decomposed vectors square agree, and by changing the order in which the color structure contraction rules are applied. The scalar product matrices have also been compared to the ColorFull code [16] for tree level trace bases with up to six partons out of which one parton is a quark and one an anti-quark. Selected results have been compared against [13], and the package has been tested both in Mathematica 7, 8 and 9.

The computational effort needed for exact treatment of the color space grows very quickly with the number of partons. The dimension of the vector space grows roughly as a factorial in the number of gluons plus $q\bar{q}$-pairs [15] (strictly speaking an exponential for finite N_c in a multiplet basis). The computational effort for ColorMath, or any program operating by direct manipulation of quark-lines, tend to grow roughly as the square of this, as the quark-lines are non-orthogonal. ColorMath (in its current form) is thus rather intended to be an easy to use package for calculations of low and intermediate complexity than a competitive tool for processes with very many colored partons.

7. Conclusion

In this paper a Mathematica package ColorMath for performing color summed calculations in QCD is presented. This package allows for simple evaluation of QCD color amplitudes which are expressed in a format which very much resembles how the color structure would have been written on paper, see table 1. The idea is that the user – for simple cases – just should give the expression, and then run **CSimplify**[**Expr**] rather than **Simplify**[**Expr**]. The package is based on advanced pattern matching rules, and a list of rules is given in table A.2, whereas functions acting on color structures are given in table A.3.

For calculations of intermediate or high complexity it is often beneficial to use a basis for performing color space calculations. ColorMath allows for definition of color tensors of form $\mathbf{C1}_{\{i1\_,i2\_,...,ik\_\}} := ...$, carrying an arbitrary set of quark, anti-quark and gluon indices. Special functions for calculating scalar products, and investigating the effect of gluon exchange, are given in table A.5. ColorMath is, however, *not* intended for high speed calculations involving many colored partons. For this purpose a separate C++ package is written [16].

Appendix A. Rules and functions

In this appendix rules and functions for index and color structure probing are stated. First the basic rules for manipulations are stated in table A.2, then the most important functions, operating directly on the basic color objects from table 1 are listed in table A.3, whereas special functions for indices and color structure probing are stated in table A.4, and the functions dealing with color vector objects are stated in table A.5.

Table A.2: The complete set of rules used by ColorMath to contract indices. These rules are applied as always in Mathematica, by using "**Expr**/.**TheRules**". To ensure that all indices which can be contracted by the rules actually are contracted, apply the rules repeatedly,"**Expr**//.**TheRules**". Note that all rules have names ending with **Rules**.

Rule	Effect
AllSimpleRules	**AllSimpleRules** is the set of all rules involving $\Delta^{\{g1,g2\}}$, $\delta^{q1}{}_{q2}$, $\mathbf{o}^{\{g1,...,gk\}}$ and $\mathbf{t}^{\{g1,...,gk\}q1}{}_{q2}$ which do not increase the number of terms, i.e., the union of **SimpleRules** and **OTSimpleRules**.
ExpandThenRules	The rules **ExpandThenRules** first uses **Expand**, then applies **OTThenAllSimpleRules** and finally restores default index order with the **SortIndices** function.
FDRules	Rules for gluon contraction for terms involving up to three structure constants, $\mathbf{f}^{\{g1,g2,g3\}}$ or $\mathbf{d}^{\{g1,g2,g3\}}$, i.e., expressions of form eq. (18) and eq. (19). The terms resulting after contraction are expressed in terms of structure constants and gluon deltas, $\Delta^{\{g1,g2\}}$.
FDToORules	Rules for replacing structure constants, $\mathbf{f}^{\{g1,g2,g3\}}$ and $\mathbf{d}^{\{g1,g2,g3\}}$, with sums of closed quark-lines $\mathbf{o}^{\{g1,g2,g3\}}$ using the intermediate expression in eq. (9).
FDToTRules	Replaces structure constants, $\mathbf{f}^{\{g1,g2,g3\}}$ and $\mathbf{d}^{\{g1,g2,g3\}}$, with a sum of products of $SU(N_c)$ generators $\mathbf{t}^{\{g1\}q1}{}_{q2}$ using the last form in eq. (9).
OTGluonRules	Rules for contracting repeated gluon indices in $\mathbf{o}^{\{g1,...,gk\}}$ and $\mathbf{t}^{\{g1,...,gk\}q1}{}_{q2}$. The Fierz identity, eq. (13), is included in these rules.
OTSimpleRules	Rules for contracting neighboring and next to neighboring gluons in closed and open quark-lines, $\mathbf{o}^{\{g1,...,gk\}}$ and $\mathbf{t}^{\{g1,...,gk\}q1}{}_{q2}$, eq. (16) and eq. (17).
OTThenAllSimpleRules	The rules **OTThenAllSimpleRules** first applies **OTGluonRules** and then repeatedly **AllSimpleRules**.
OTToTRules	Rules for replacing open and closed quark-lines, $\mathbf{o}^{\{g1,...,gk\}}$ and $\mathbf{t}^{\{g1,...,gk\}q1}{}_{q2}$, with products of $SU(N_c)$ generators $\mathbf{t}^{\{g1\}q1}{}_{q2}$.
Remove0ORules	Rule for simplifying closed quark-lines with 0 gluons, $\mathbf{o}^{\{\}} = \mathbf{Nc}$.
Remove0To1ORules	Rules for simplifying closed quark-lines with 0 or 1 gluons, $\mathbf{o}^{\{\}} = \mathbf{Nc}$, $\mathbf{o}^{\{g1\}} = 0$.
Remove0To2ORules	Rules for simplifying closed quark-lines with 0 to 2 gluons, $\mathbf{o}^{\{\}} = \mathbf{Nc}$, $\mathbf{o}^{\{g1\}} = 0$ and $\mathbf{o}^{\{g1,g2\}} = \mathbf{TR}\,\Delta^{\{g1,g2\}}$.
SimpleRules	Basic rules for quark and gluon contraction. These rules involve $\Delta^{\{g1,g2\}}$, $\delta^{q1}{}_{q2}$ or quark contraction, and never increase the number of terms. These rules thus contain the rules in (iii-iv) as well as in eq. (6).

Table A.3: The most useful functions, to be used on any expression carrying color structure of the form given in table 1. Recall to relabel dummy indices, manually, or by using **ReplaceDummyIndices**, and to place quark indices that are to be contracted such that one index sits upstairs and one downstairs. Furthermore, check that the correct **FullForm** is used, cf. table 1. The function **WhatIsWrong** may be useful for identifying common input mistakes in the basic color objects.

Function with usage	Effect
CSimplify[Expr] **Options** with default value: **RemoveFD → True**	The most general function for simplifying color structure. If **Expr** contains structure constants, $\mathbf{f}^{\{g1,g2,g3\}}$ or $\mathbf{d}^{\{g1,g2,g3\}}$, **FDRules** are first applied. If, after this, the expression still contains structure constants they are – by default – removed using FDToORules, and all repeated indices are subsequently contracted by repeatedly using **AllSimpleRules**, then **OTThenAllSimpleRules**, and finally **ExpandThenRules**. For color structure containing structure constants, it could happen that it is desired not to replace the structure constants. This can be achieved by setting the option **RemoveFD → False**.
GluonContract[Expr, Gs]	Contracts a set (**List**) of gluons **Gs** = $\{\mathbf{g1}, ..., \mathbf{gn}\}$ or a single gluon **Gs** = $\mathbf{g1}$, in the expression **Expr**, while leaving other indices uncontracted. This function is intended for quark-lines and will replace structure constants with quark-lines.
ReplaceDummyIndices[Expr]	Replaces the dummy indices in **Expr** with a new set of unique dummy indices.
SortIndices[Expr]	Sorts the gluon indices appearing in $\Delta^{\{g1,g2\}}$, $\mathbf{f}^{\{g1,g2,g3\}}$ or $\mathbf{d}^{\{g1,g2,g3\}}$ and $\mathbf{o}^{\{g1,...,gk\}}$ such that they stand in Mathematica default order. This is needed to ensure that one color structure only is represented in one form.
SplitConstAndColor[Expr]	Splits an expression **Expr** into a **List** of **List**s of color structures and corresponding multiplicative factors {{constants 1, color structure 1},{constants 2, color structure 2},...}. This is done by first expanding the expression and then splitting the terms separately.
WhatIsWrong[Expr]	Checks if anything is obviously wrong with an expression, for example if **Power** is used instead of **Superscript** or if gluon indices are placed downstairs. The check is performed by first expanding the expression, and then checking each term. Read error messages from above.

Table A.4: The below set of functions are used for probing the color structure, and identifying indices. Concerning the convention of what (lower or upper) quark-indices are referred to as incoming quarks and outgoing anti-quarks, as opposed to incoming anti-quarks and outgoing quarks, ColorMath does not specify any preference, but simply refers to the indices as upper quarks and lower quarks when needed.

Function	Effect
AllIndices[Expr]	Returns a **List** of all (external and dummy) indices in **Expr**.
ContainsColor[Expr]	Returns **True** if the expression **Expr** contains any color structure, i.e., any of the terms in table 1, and **False** otherwise.
ContainsFD[Expr]	Returns **True** if the expression **Expr** contains structure constants and **False** otherwise.
ContainsGluonDelta[Expr]	Returns **True** if the expression **Expr** contains a gluon delta-function, $\Delta^{\{g1,g2\}}$, and **False** otherwise.
ContainsO[Expr]	Returns **True** if the expression **Expr** contains closed quark-lines, $o^{\{g1,\dots,gk\}}$, and **False** otherwise.
ContainsQuarkDelta[Expr]	Returns **True** if the expression **Expr** contains a quark delta-function, $\delta^{q1}{}_{q2}$, and **False** otherwise.
ContainsT[Expr]	Returns **True** if the expression **Expr** contains open quark-lines, $t^{\{g1,\dots,gk\}q1}{}_{q2}$, and **False** otherwise.
DummyIndices[Expr]	Finds the dummy indices in an expression **Expr**, by first expanding it and then finding all dummy indices in all terms.
GluonIndices[Expr]	Returns a **List** of all gluon indices (external and dummy) in the expression **Expr**.
LowerQuarkIndices[Expr]	Returns a **List** of all (external and dummy) quark-type indices placed downstairs, i.e., $q2$ in $t^{\{g1,\dots,gk\}q1}{}_{q2}$ and $\delta^{q1}{}_{q2}$.
UpperQuarkIndices[Expr]	Returns a **List** of all (external and dummy) quark-type indices placed upstairs, i.e., $q1$ in $t^{\{g1,\dots,gk\}q1}{}_{q2}$ and $\delta^{q1}{}_{q2}$.

Table A.5: Special functions for color structures expressed in the form of eq. (25), i.e., the corresponding tensors need to be defined using the pattern matching underscores, potential dummy indices should be hidden inside **Module**s, and the evaluation should be delayed for expressions containing dummy indices, i.e. ":=" should be used. The vector indices should sit inside a **List** in the **Subscript**. Additional options may be supplied to these functions to control for example the verbosity. Thus, **CNorm** may for example be called using **CDot**[{**C1**,..., **Cn**}, **Verbose** → **True**], to get progress information or simply as **CDot**[{**C1**,...,**Cn**}] to use the default options.

Vector function with usage	Effect
CDot[**C1**, **C2**, **Options**] **Options** with default value: **NcMin** → **3**	Calculates the scalar product between two color tensors **C1** and **C2**. Before using **CDot** the color tensors **C1**$_{\{i1\_,i2\_,...,ik\_\}}$ and **C2**$_{\{i1\_,i2\_,...,ik\_\}}$ should have been defined. For simplifications (in particular of roots) it is by default assumed that **Nc** is at least **3**. However, this may be manually changed by setting the option **NcMin**, for example **CDot**[**C1**, **C2**, **NcMin** → **100**].
CDot[{**C1**,..., **Cn**}, **Options**] **Options** with default value: **NcMin** → **3** **Verbose** → **False**	Similar to **CDot**[**C1**, **C2**] but calculates the scalar product of a set of color tensors and themselves and returns the result as a **List** where the ith element is **CDot**[**Ci**, **Ci**]. In addition to assuming a minimal value of **Nc** for simplifications, it is also possible to get progress information by setting the option **Verbose** to **True**.
CDotMatrix[{**C1**,..., **Cn**}, **Options**] **Options** with default value: **NcMin** → **3** **Verbose** → **False**	Calculates the scalar product matrix given a **List** of color vectors, i.e., element ij is **CDot**[**Ci**, **Cj**]. If the list contains a basis, this function thus returns the scalar product matrix.
CNorm[**C1**, **Options**] **Options** with default value: **NcMin** → **3**	Calculates the norm of a color tensors using **CDot**. For simplifying the result, it may be useful to set the option **NcMin** to a large value.
CNorm[{**C1**,...,**Cn**}, **Options**] **Options** with default value: **NcMin** → **3** **Verbose** → **False**	Similar to **CNorm** above, but calculates the norms of a **List** of color vectors {**C1**,...,**Cn**}, and returns a **List** containing the norms. Progress information is available by supplying the option **Verbose**→ **True**.
CGamma[{**C1**,...,**Cn**}, **p1**, **p2**, **Options**] **Options** with default value: **NcMin** → **3** **Verbose** → **True** **MakeChecks** → **True** **BasisType** → **GeneralBasis**	Describes the effect of gluon exchange between partons **p1** and **p2**, on basis vector **Cj** as a column vector j in a resulting matrix (technically **List** of **List**), i.e., element ij is the resulting color structure's i-th component. The **BasisType** option should assume one of the values **GeneralBasis**, **OrthogonalBasis**, **OrthonormalBasis** or **TraceBasis**. The latter is defined to be any basis where the basis vectors consist of *one* product of closed and open quark-lines, not a sum. By default it is checked that the color tensor, resulting after gluon exchange, is the same when squared directly, as when basis decomposed, and then squared. This checks that the basis is complete and that the basis decomposition is correct. For orthonormal bases, it is also checked that the resulting matrix is symmetric. Using **MakeChecks** → **False** these checks are turned off. By default, progress information is also written out, which can be changed with **Verbose** → **False**.

References

[1] J. Hakkinen and H. Kharraziha, *COLOR: A Computer program for QCD color factor calculations*, *Comput.Phys.Commun.* **100** (1997) 311–321, [`hep-ph/9603229`].

[2] J. Alwall, M. Herquet, F. Maltoni, O. Mattelaer, and T. Stelzer, *MadGraph 5 : Going Beyond*, *JHEP* **1106** (2011) 128, [`arXiv:1106.0522`].

[3] J. Kuipers, T. Ueda, J. Vermaseren, and J. Vollinga, *FORM version 4.0*, `arXiv:1203.6543`.

[4] J. E. Paton and H.-M. Chan, *Generalized Veneziano model with isospin*, *Nucl. Phys. B* **10** (1969) 516–520.

[5] P. Dittner, *Invariant tensors in SU(3). II*, *Commun. Math. Phys.* **27** (1972) 44–52.

[6] P. Cvitanović, *Group theory for Feynman diagrams in non-Abelian gauge theories*, *Phys. Rev. D* **14** (1976) 1536–1553.

[7] P. Cvitanović, P. Lauwers, and P. Scharbach, *Gauge invariance structure of quantum chromodynamics*, *Nucl. Phys. B* **186** (1981) 165–186.

[8] M. L. Mangano, S. J. Parke, and Z. Xu, *Duality and multi-gluon scattering*, *Nucl. Phys. B* **298** (1988) 653.

[9] M. L. Mangano, *The color structure of gluon emission*, *Nucl. Phys. B* **309** (1988) 461.

[10] Z. Nagy and D. E. Soper, *Parton showers with quantum interference*, *JHEP* **09** (2007) 114, [`arXiv:0706.0017`].

[11] M. Sjodahl, *Color structure for soft gluon resummation – a general recipe*, *JHEP* **0909** (2009) 087, [`arXiv:0906.1121`].

[12] S. Platzer and M. Sjodahl, *Subleading N_c improved parton showers*, *JHEP* **1207** (2012) 042, [`arXiv:1201.0260`].

[13] M. Sjodahl, *Color evolution of $2 \to 3$ processes*, *JHEP* **12** (2008) 083, [`arXiv:0807.0555`].

[14] M. H. Seymour and M. Sjodahl, *Symmetry of anomalous dimension matrices explained*, *JHEP* **12** (2008) 066, [`arXiv:0810.5756`].

[15] S. Keppeler and M. Sjodahl, *Orthogonal multiplet bases in SU(Nc) color space*, *JHEP* **1209** (2012) 124, [`arXiv:1207.0609`].

[16] M. Sjodahl, *ColorFull – A C++ package for color space calculations*, *(work in preparation)*.