Eur. Phys. J. C (2022) 82:1144
https://doi.org/10.1140/epjc/s10052-022-11078-2

THE EUROPEAN PHYSICAL JOURNAL C

Special Article - Tools for Experiment and Theory

Improving colour computations in MadGraph5_aMC@NLO and exploring a $1/N_c$ expansion

Andrew Lifson[1,2,a], **Olivier Mattelaer**[2]

[1] Department of Astronomy and Theoretical Physics, Lund University, Sölvegatan 14A, 223 62 Lund, Sweden
[2] Centre for Cosmology, Particle Physics and Phenomenology (CP3), Université catholique de Louvain, Chemin du Cyclotron 2, 1348 Louvain-la-Neuve, Belgium

Received: 19 October 2022 / Accepted: 24 November 2022 / Published online: 19 December 2022
© The Author(s) 2022

Abstract In this paper, we present an extension of MadGraph5_aMC@NLO which is able to evaluate tree-level QCD matrix-elements up to $2 \to 6$ (one more particle than before). To achieve this, we implemented Berends–Giele-like recursion, and re-implemented the way colour is computed such that we can now expand the colour matrix in powers of $1/N_c$ and truncate this expansion to a chosen order. For high multiplicity samples, even without truncating the colour matrix, the new implementation offers a speed gain compared to the previous MadGraph5_aMC@NLO code.

Contents

1 Introduction . . . 1
2 Background theory . . . 2
2.1 Colour ordering and the $1/N_c$ expansion . . . 2
2.1.1 Colour ordering in the fundamental basis 2
2.1.2 $1/N_c$ expansion . . . 3
2.2 Berends–Giele recursions . . . 4
3 Technical implementation . . . 4
3.1 The MadGraph5_aMC@NLO event generator . 4
3.2 Implementation of colour computation . . . 5
3.3 Implementation of Berends–Giele recursion . . 6
3.4 Sources of speed differences . . . 6
4 Validation and results . . . 7
4.1 Accuracy and precision of colour approximation 7
4.2 Speed gain . . . 11
5 Conclusion . . . 13
Appendix A: Manual . . . 13
Appendix B: Accuracy and speed of additional processes 14
Appendix C: Subprocess cross-sections in multi-jet production . . . 16
Appendix D: Modified colour expansion for multiquark amplitudes . . . 16
References . . . 17

[a] e-mail: andrew.lifson@thep.lu.se (corresponding author)

1 Introduction

Accurate and efficient calculation of the hard matrix element is at the core of most predictions in high-energy physics, with many tools currently available to automate these calculations [1–26]. However, due to a much faster growth in the number of events required for the high-luminosity LHC compared to the growth of the LHC cpu-hour budget, the efficiency of such programs needs to be improved by at least 20% and ideally by a factor of two [27,28]

At the high-luminosity LHC, we expect to see and generate many processes with multiple well-separated jets. There are two challenges to calculating the hard matrix element for these types of processes, even at tree level. The first is to quickly calculate the Lorentz part of the amplitude (often called the kinematic part), typically calculated by summing Feynman diagrams which grow roughly factorially with the number of external particles; while the second is to calculate the colour algebra, which typically grows like a factorial squared with the number of external particles. Indeed, as the multiplicity increases, the colour takes up a larger percentage of a MadGraph5_aMC@NLO (MG5aMC) calculation, with the colour taking up about 60% of the time required to calculate the cross section of $gg \to t\bar{t}ggg$ [29].

There have been several attempts to speed up the Feynman diagrams used to calculate the kinematics [29–34]. However, an alternative method to speed up the kinematics, is to use recursions such as the off-shell Berends–Giele (BG) recursion instead of Feynman diagrams [35]. These recursions sum up multiple Feynman diagrams into a single term,

thus decreasing the required amount of computation, and have already been implemented in e.g. [5,11]. Other recursive methods can include other off-shell recursions such as [36,37], or on-shell recursion relations such as [38–40], though past studies have shown that BG recursions are typically quicker [41–43].

For the colour, research has mainly focused on two directions. The first is to diagonalise the colour matrix, thus severely reducing the number of elements in the colour matrix. This is mostly realised in the multiplet basis [44–46]. The second approach is to use the large-N_c limit [47], and expand the colour matrix in a power series in $1/N_c$. For the most relevant processes, each order of the expansion is separated by two powers of $1/N_c$, making the expansion about as accurate as the expansion in α_s.

In this paper, we implement both BG recursion and a colour expansion in MG5aMC for tree-level Standard Model processes. In Sect. 2, we summarise colour ordering and the $1/N_c$ expansion, as well as BG recursions. Next, in Sect. 3, we describe and profile their implementations in the MG5aMC event generator. We show our results for pure QCD processes in Sect. 4, showing the accuracy and speed of the colour expansion. We conclude in Sect. 5. A small user manual is described in Appendix A. In Appendix B, we briefly show our results for some additional processes including those with an electroweak boson. We study the relative importance of different subprocesses in a typical QCD cross section in Appendix C. Finally, in Appendix D, we describe a proposed modified definition of the colour expansion in multiquark amplitudes.

2 Background theory

In this section, we describe the two main ideas we implemented in this paper, colour ordering in the fundamental basis and its expansion in powers of $1/N_c$, and the use of Berends–Giele recursion to calculate colour-ordered kinematic amplitudes.

2.1 Colour ordering and the $1/N_c$ expansion

2.1.1 Colour ordering in the fundamental basis

A trick which is often used in QCD calculations is to factorise the colour part of an amplitude from the kinematics [48–51]

$$\mathcal{M}(1,\dots,n) = \sum_{\sigma} F_\sigma(\mathrm{su}(N_c)) M_\sigma(p_1,h_1;\dots;p_n,h_n), \tag{1}$$

where, for e.g. the fundamental (also called the trace) or colour-flow bases, σ is a given permutation of colour orderings, F_σ is a function of the gauge algebra $\mathrm{su}(N_c)$, and M_σ is the kinematic (colour ordered) amplitude, which is a function of the momenta and helicities of the particles. Depending on the basis in colour space, there may be different forms of F_σ and M_σ, and different sets of permutations σ.

The squared matrix-element is then given by,

$$|\mathcal{M}(1,\dots,n)|^2 = \sum_{\sigma,\sigma'} M_\sigma \underbrace{F_\sigma F^*_{\sigma'}}_{C_{\sigma\sigma'}} M^*_{\sigma'}, \tag{2}$$

where we have dropped all functional dependence on the right hand side; σ, σ' are two sets of colour-ordering permutations; and the product $F_\sigma F^*_{\sigma'} \equiv C_{\sigma\sigma'}$ is called the colour matrix, which in e.g. the fundamental or colour flow bases is a square matrix with size growing factorially with the particle multiplicity. The colour matrix typically contains polynomials in N_c, the number of colours, and is calculated using the following colour-algebra relations:

$$\begin{aligned}
&\mathrm{Tr}(t^a) = 0, && \mathrm{Tr}(t^a t^b) = T_R \delta^{ab},\\
&if^{abc} = \frac{1}{T_R}\mathrm{Tr}(t^a[t^b,t^c]), && if^{abc}t^c = [t^a,t^b],\\
&\delta_{ii} = N_c, && \delta^{aa} = N_c^2 - 1,\\
&t^a_{ij}t^a_{kl} = T_R\left(\delta_{il}\delta_{jk} - \frac{1}{N_c}\delta_{ij}\delta_{kl}\right).
\end{aligned} \tag{3}$$

Here, $i,j,k,l = 1,2,\dots,N_c$ are (anti)fundamental indices, $a,b,c = 1,2,\dots,N_c^2-1$ are adjoint indices, all repeated indices are summed, t^a_{ij} is a generator of $\mathrm{su}(N_c)$ in the fundamental representation, f^{abc} the structure constants (or equivalently the generators of $\mathrm{su}(N_c)$ in the adjoint representation), and T_R is a normalisation factor, in MG5aMC set to $1/2$ (though in the literature it is often set to one).

In MG5aMC, the fundamental basis is used to calculate the colour matrix. In this basis, all colour factors are written as strings of fundamental matrices t^a_{ij}. For example, the all-gluon amplitude is written as

$$\mathcal{M}(ng) = \sum_{P(2,\dots,n)} \mathrm{Tr}(t^1\dots t^n) M(1,\dots,n), \tag{4}$$

where, $P(2,\dots,n)$ indicates the sum of all permutations of particles $2\dots n$ (particle 1 is fixed to not double count, since the trace is cyclic).

This gives a colour matrix $C^{ng}_{\sigma\sigma'}$

$$C^{ng}_{\sigma\sigma'} = \mathrm{Tr}(t^{\sigma_1}\dots t^{\sigma_n})\mathrm{Tr}(t^{\sigma'_n}\dots t^{\sigma'_1}), \tag{5}$$

which can be written as a polynomial in N_c using Eq. (3).

Similarly, the amplitude with a single quark line is given by[1]

$$\mathcal{M}(q\bar{q}+ng) = \sum_{P(1,\dots,n)} (t^1\dots t^n)_{q\bar{q}} M(1,\dots,n), \tag{6}$$

[1] We consider all particles as outgoing, so each quark line has a quark and an antiquark.

with colour matrix $C^{q\bar{q}+ng}_{\sigma\sigma'}$

$$C^{q\bar{q}+ng}_{\sigma\sigma'} = (t^{\sigma_1}\dots t^{\sigma_n})_{q\bar{q}}(t^{\sigma'_n}\dots t^{\sigma'_1})_{\bar{q}q}, \tag{7}$$

while the amplitude with two distinct quark lines is given by

$$\begin{aligned}\hat{\mathcal{M}}(q\bar{q}Q\bar{Q}+ng) &= \sum_{i=0,n}\sum_{P(1,\dots,i)}\sum_{P(i+1,\dots,n)}\Big[(t^1\dots t^i)_{q\bar{Q}}(t^{i+1}\dots t^n)_{Q\bar{q}} \\ &\quad\times M(q,1,\dots,i,\bar{Q},Q,i+1,\dots,n,\bar{q}) \\ &\quad-\frac{1}{N_c}(t^1\dots t^i)_{q\bar{q}}(t^{i+1}\dots t^n)_{Q\bar{Q}} \\ &\quad\times M(q,1,\dots,i,\bar{q},Q,i+1,\dots,n,\bar{Q})\Big].\end{aligned} \tag{8}$$

Here, the first sum allows the gluons to be emitted by either fundamental colour line, and the second and third sums permute the gluons on each fundamental colour line. If there are no gluons in a string of t-matrices ($i = 0$ or $i = n$), then that string should be replaced by a Kronecker delta with the relevant (anti)fundamental indices.

The reason to have two strings of t-matrices, is that we have used the Fierz identity (last equation of Eq. (3)) to remove the repeating colour index of the internal gluon connecting the two quark lines. This leaves us with two terms, the first (second line of Eq. (8)) is called the $u(N_c)$ term, while the second (fourth line of Eq. (8)) is called the $u(1)$ term, and is $1/N_c$ suppressed.

If the two quark lines have the same flavour we use (see e.g. [52,53])

$$\begin{aligned}\mathcal{M}(q_1\bar{q}_1q_2\bar{q}_2+ng) &= \hat{\mathcal{M}}(q_{\sigma(1)}\bar{q}_1q_{\sigma(2)}\bar{q}_2+ng) \\ &\quad-\hat{\mathcal{M}}(q_{\sigma(2)}\bar{q}_1q_{\sigma(1)}\bar{q}_2+ng),\end{aligned} \tag{9}$$

where σ is a permutation of the quarks, and $\hat{\mathcal{M}}$ is the distinct-flavour amplitude from Eq. (8).

2.1.2 $1/N_c$ expansion

In the fundamental basis, each term in the colour matrix $C_{\sigma\sigma'}$ is a polynomial in N_c. One possible definition of the colour expansion in this basis, is to keep polynomials of the highest degree at leading colour (LC), keep polynomials with at most two degrees smaller at next-to-leading colour (NLC), and so on. In this definition, each kept polynomial is retained in full, i.e. we do not truncate the individual polynomials in the colour matrix. We now go through the expansion for different types of tree-level amplitudes.

Amplitudes with at most one quark line: For these amplitudes, the polynomial has the form [54]

$$C_{\sigma\sigma'} = a_nN_c^n + a_{n-2}N_c^{n-2} + \dots + a_mN_c^m, \quad \text{for}\begin{cases} n = n_g, & \text{all-gluon amplitudes} \\ n = n_g+1, & \text{single-quark amplitudes}\end{cases} \tag{10}$$

where each term in the expansion is two powers of N_c smaller than the previous term, each a_i is some constant, n_g is the number of gluons and m is an integer with $m \le n-2$. This motivates expanding the colour matrix in powers of N_c, such that the LC terms are those with $a_n \neq 0$, the NLC terms are those with $a_n = 0$, $a_{n-2} \neq 0$, and so on.

Looking at the colour matrices themselves (Eqs. (5) and (7)), and using the colour algebra relations (3), it is easy to prove that $a_n = 0$ only if $\sigma \neq \sigma'$, $a_{n-2} = 0$ except on the diagonal and some off-diagonal terms, and so on.

Modified leading colour for all-gluon amplitudes: The LC all-gluon amplitude can be modified and made more accurate by using [48,54]

$$\sum_{\text{colours}}|\mathcal{M}(ng)|^2 = T_R^nN_c^{n-2}(N_c^2-1) \times\sum_{P(2,\dots,n)}\Big[|M(1,\dots,n)|^2+\mathcal{O}(N_c^{-2})\Big], \tag{11}$$

as the LC definition. Note that in this definition we do not keep the full LC polynomial, but rather truncate it due to relations between colour-ordered amplitudes.

Unfortunately, the authors only know the $\mathcal{O}(N_c^{-2})$ terms in this version of the expansion for 6 or less gluons [48], but not in full generality. This leads to the strange effect that the default 'leading colour' amplitude Eq. (11) is more accurate than the NLC amplitude which uses the standard $1/N_c$ expansion in the fundamental basis (cf Sect. 4.1). For this reason, we label the default LC matrix element as modified LC, or 'modLC'. We leave to future work a program which calculates the modified off-diagonal terms for an arbitrary number of gluons.

Amplitudes with two quark lines: The colour expansion for these amplitudes suffers from two problems, one which occurs when the quarks have the same flavour, and another when they have distinct flavours. First, unlike Eq. (10), the same-flavour colour matrix has the form

$$C_{\sigma\sigma'} = a_nN_c^n + a_{n-1}N_c^{n-1} + a_{n-2}N_c^{n-2} + \dots + a_mN_c^m, \quad n = n_g+2, \tag{12}$$

so that, at a given order of the expansion, we have corrections of $\mathcal{O}(1/N_c) \sim 0.33$, not of $\mathcal{O}(1/N_c^2) \sim 0.11$ as before. Due to this, we do not expect as precise an expansion as the previous cases.

Despite this, we still define the expansion in powers of $1/N_c^2$, and not as powers of $1/N_c$. Therefore, the LC terms are those with $a_n \neq 0$, the NLC terms are those with $a_{n-1} \neq 0$ and/or $a_{n-2} \neq 0$ but $a_n = 0$, and so on.

If the quarks have distinct flavours, the colour matrix once again follows Eq. (10) with $n = n_g + 2$, but this time a different problem arises. In this case, at LC we only include the first three lines of Eq. (8), missing entirely all of the kinematic amplitudes in the last line of this equation. Since these kinematic amplitudes could contain terms much larger than $1/N_c^2$ we expect the expansion to be poor at LC. In appendix D we show an attempt to solve this second problem by redefining the colour expansion.

2.2 Berends–Giele recursions

The basic idea of these recursions is to calculate an off-shell current $J_n(1, \ldots, n)$ with n particles on shell and a single particle off shell. The $(n+1)$-particle colour-ordered amplitude is given by J_n with its off-shell propagator amputated, and the result contracted with the wavefunction for particle $n+1$ [35].

Gluon currents: The base ingredients of the gluon off-shell currents are the one- and two-particle currents J_1^μ and J_2^μ

$$\begin{aligned} J_1^\mu(1) &= \epsilon^\mu(1), \\ J_2^\mu(1,2) &= \frac{-i}{(p_1+p_2)^2} V_3^{\mu\mu_1\mu_2}(p_1,p_2) J_{1,\mu_1}(1) J_{1,\mu_2}, \end{aligned} \tag{13}$$

where $\epsilon^\mu(1)$ is the gluon polarisation vector with momentum p_1, and $V_3^{\mu\mu_1\mu_2}(p_1, p_2)$ the colour-ordered three-gluon vertex.

Using these ingredients as input, together with the colour-ordered four-point vertex $V_4^{\mu_1\mu_2\mu_3\mu_4}$, a generic n-point current J_n^μ is

$$\begin{aligned} J_n^\mu(1,\ldots,n) &= \frac{-i}{P_{1,n}^2} \\ &\times \Bigg\{ \sum_{i=1}^{n-1} V_3^{\mu\nu\rho}(P_{1,i}, P_{i+1,n}) J_\nu(1,\ldots,i) J_\rho(i+1,\ldots,n) \\ &+ \sum_{i=1}^{n-2} \sum_{j=i+1}^{n-1} V_4^{\mu\nu\rho\sigma} J_\nu(1,\ldots,i) J_\rho(i+1,\ldots,j) J_\sigma(j+1,\ldots,n) \Bigg\}, \end{aligned} \tag{14}$$

where we use the shorthand $P_{1,n}^2 = (p_1 + \cdots + p_n)^2$, drop the number of particles n in J_n^μ where convenient, and use all outgoing momenta.

To obtain the $(n+1)$-point amplitude it remains to amputate the propagator, and contract this current with an (on-shell) external gluon,

$$M(1,\ldots,n+1) = i P_{1,n}^2 \epsilon_\mu(n+1) J_n^\mu(1,\ldots,n)|_{p_1+\cdots+p_{n+1}=0}\,. \tag{15}$$

Quark currents: The base ingredients for the quark current is a single on-shell quark, and an on-shell quark which radiated a gluon i.e.

$$\begin{aligned} J_1(q) &= \bar{u}(q), \\ J_2(q;1) &= \bar{u}(q) i \gamma^\mu \epsilon_\mu(1) i \frac{\not{q} + \not{p}_1 + m}{(q+p_1)^2 - m^2} \\ &\equiv -J_1(q) \not{J}_1(1) \frac{\not{q} + \not{p}_1 + m}{(q+p_1)^2 - m^2}, \end{aligned} \tag{16}$$

where if the current J has a q in its arguments, then it is a quark current, otherwise it is a gluon current.

For an arbitrary number of gluons, the quark current is

$$\begin{aligned} J(q;1,\ldots,n) = &-\sum_{i=0}^{n-1} J(q;1,\ldots,i) \not{J}(i+1,\ldots,n) \\ &\times \frac{\not{q} + \not{P}_{1,n} + m}{(q+P_{1,n})^2 - m^2}, \end{aligned} \tag{17}$$

while the amplitude is found by contracting with the inverse propagator and the antispinor, and putting the anti-spinor on shell

$$\begin{aligned} M(q;1,\ldots,n;\overline{q}) = &J(q;1,\ldots,n)(-i)(\not{q} + \not{P}_{1,n} - m) \\ &\times v(\overline{q})|_{q+P_{1,n}+\overline{q}=0}\,, \end{aligned} \tag{18}$$

where again $P_{i,j} = p_i + \cdots + p_j$ and all momenta are outgoing.

3 Technical implementation

In this section, we will go through some of the details of our implementation of the colour matrix and its expansion, as well as of the BG recursions. First, in Sect. 3.1, we recall the main features of the event generator used throughout the paper, `MadGraph5_aMC@NLO` (MG5aMC). Then, we will give some details of how we implemented the colour expansion (Sect. 3.2) and the BG recursions (Sect. 3.3). Finally, in Sect. 3.4, we discuss in detail the sources of speed difference between the old and new codes using $gg \to 5g$ as a test case.

3.1 The `MadGraph5_aMC@NLO` event generator

MG5aMC is a metacode which writes a program in the user's preferred language to calculate either the squared matrix element (standalone mode) or cross section/event generation (MadEvent mode) of a chosen process within a chosen model

at either leading order (LO) or next-to-leading order (NLO). For example, choosing the default language of Fortran, the default model of the Standard Model (SM), and $gg \to gg$ at LO as a process, MG5aMC will first generate the four Feynman diagrams in this process, then write a Fortran program which either calculates its squared matrix element or cross section. The user then runs the program to get their result.

The most common usage of MG5aMC (at LO) is the MadEvent mode, which returns the cross section for a given process, including all cuts required to compare to experiment. This requires both calculating the hard matrix element, and sampling phase space efficiently to obtain an accurate cross section.

On the other hand, the standalone version of MG5aMC calculates matrix elements at a specific, given, phase-space point. It allows to isolate the speed of a matrix element computation, since we do not have to worry about the convergence speed of the integral. If many phase-space points are required, it uses RAMBO [55] to do a flat scan of the phase space.

In this paper, we use the standalone version to better isolate the speed of the matrix element calculation and to validate that the Berends–Giele recursions are correctly implemented.

3.2 Implementation of colour computation

In standard MG5aMC (also referred to below as the old code), the colour matrix is written explicitly as a square matrix of floats with size growing factorially with the particle multiplicity, and Eq. (2) is calculated by using two for loops to do the explicit matrix multiplication. All of the Feynman diagrams appear on equal footing, and are only calculated once each using the helicity amplitude formalism. In pseudocode, it looks like this:

```
# store colour matrix
C_ij = ...

# calculate all wavefunctions and propagators
wf_i = ...

# calculate all Feynman diagrams
amp_i = ...

# combine Feynman diagrams into colour-ordered amplitudes
coamp_i = ...

# sum over colours
full_amp^2 = 0
loop over rows
|   tmp = SUM over columns (C_ij * coamp_j)
|   full_amp^2 += conj(coamp_i) * tmp
```

In the new BG/colour ordered code (from now on referred to as the new code) each of these steps are done differently. One big difference occurs for multiquark amplitudes. For these, we do not simply calculate all kinematic (BG or Feynman) diagrams once. Instead, we separate the kinematic amplitude into multiple calculations of different flows corresponding to: (i) whether the colour ordering belongs to a u(N_c) or u(1) gluon; and (ii) how many gluons are on each colour line. This makes it easy to combine partial graphs into BG currents, but has the disadvantage that the same kinematic diagrams are calculated multiple times.

Additionally, instead of writing the full colour matrix explicitly, we take the first row (if multiquark the first row for each flow) of the colour sum and separate it into contributions at LC, NLC, N2LC, etc. For each colour order and flow, we write the kinematic amplitudes for that row times the relevant colour matrix entries times the conjugate amplitude. That is, we have something of the form

$$M^*_{\sigma_1}\left(\sum_{j\in \mathrm{N}^k\mathrm{LC}} C_{\sigma_1\sigma_j} M_{\sigma_j}\right). \tag{19}$$

To loop over all rows, we keep the values of colour factors in Eq. (19) the same, and permute the colour-ordered amplitude indices σ_j. This requires a permutation matrix of the same size as the original colour matrix, but which, unlike the colour matrix, has integer components, so uses only half the size in memory (and it can technically be reduced even further). This is a feasible solution for the multiplicities we wish to probe, but for higher multiplicities the factorial-squared growth will quickly become a problem (expanding in powers of $1/N_c$ offers one possible solution depending on the accuracy desired).

A pseudocode of the new program (for a given flow) is:

```
# calculate kinematics
...

# store permutation matrix
perm[i,j] = k

# sum over colours
full_amp^2 = 0
loop over rows of permutation matrix (nrow)
|  onerow = 0
|  loop over column j, filtered for 1/Nc
|  |   onerow+= C_1j* coamp[perm[nrow,j]]
|  onerow*= conj(coamp[perm[nrow,1])
|  full_amp^2+= onerow
```

Note that in both methods of computation, one can use the fact that the colour-matrix is symmetric to further optimise the computation.[2]

[2] This optimisation will be added to MG5aMC version 3.5.0 within the old code and within 3.5.1 for the new code.

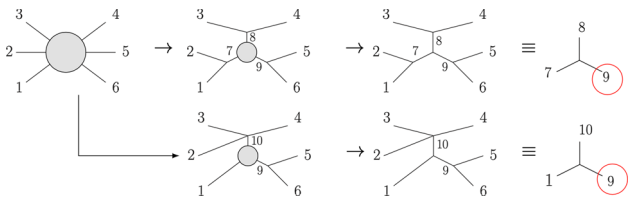

Fig. 1 Recycling in MG5aMC. Starting from out to in, MG5aMC calculates propagators as off-shell currents which it then caches. When two diagrams share an off-shell current such as current 9, there is no need to recalculate this current, and the cached version is instead used. The value of the Feynman diagram is then calculated by contracting the remaining, often internal, currents

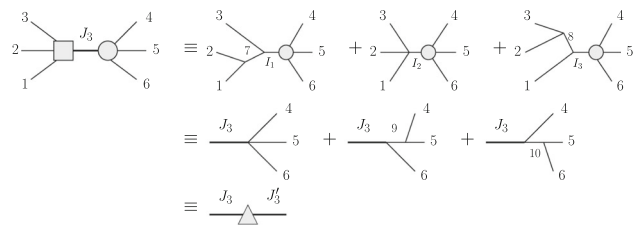

Fig. 2 An example of BG recursion in MG5aMC. Each of the square and circular blobs represent three possible diagrams. The current J_3 is created by combining the three off-shell currents I_1, I_2, and I_3 into a single off-shell current (first line), which is then used in three graphs (second line). In this way, 9 Feynman diagrams becomes 3 graphs. A future optimisation would be to also combine particles 4, 5, and 6 into another three-particle current J_3', which would then have its propagator amputated and be contracted with J_3 (third line)

3.3 Implementation of Berends–Giele recursion

In MG5aMC, multiple Feynman diagrams are calculated efficiently by recycling three- and four-point off-shell currents when they belong to multiple diagrams (see Fig. 1). This allows to reduce the total number of calculations required, making a simpler and faster program.

While the version of BG recursion given in Sect. 2.2 builds currents by always adding a single extra particle until all particles have been used, MG5aMC does not do this. This is because MG5aMC uses multiple small BG currents in parallel (for different external particles), before eventually contracting these currents together in a trivalent or four-valent vertex (see first two lines of Fig. 2). One consequence of this choice is that BG recursions lead to a speed gain only for multiplicity greater than or equal to six, since below that the recycling algorithm reaches the same efficiency.

We stress that our new code is less optimal to compute the kinematics part than standard MG5aMC, both with and without using BG recursion. The reason for this is that we have not implemented all possible optimisations (many such optimisations are well known, and are left to future work). Nevertheless, the BG recursions compile far quicker than the old code at high multiplicity, allowing to generate and study processes with higher multiplicity than before. Also, the new code is faster to run than the old code at high multiplicities, even with the slower kinematics (see Sect. 4.2).

3.4 Sources of speed differences

As seen in the pseudocode in Sect. 3.2, we can, loosely speaking, divide a MG5aMC calculation into four parts:

(i) Calculate wavefunctions (`WF`s), both external and internal (i.e. propagators or off-shell BG currents)
(ii) Calculate the amplitudes (`AMP`s), i.e. completed Feynman or BG graphs
(iii) Sum up the `AMP`s into the colour ordered amplitudes (M_σ)
(iv) Loop over the colour matrix, calculating Eq. (2).

Table 1 The number of instructions to calculate $gg \to 5g$ for 10 phase-space points at full colour in standard MG5aMC standalone and in our new code (at N6LC, i.e. full colour and using BG recursions). In addition to the total number of instructions required to do the calculation (Full ME), we have broken down the calculation into four steps: calculating internal and external wavefunctions (`WF`s), calculating completed graphs (`AMP`s), putting these graphs into colour ordered amplitudes (M_σ), and summing over colours (col sum). The number in brackets is the percentage of the total number of instructions required to calculate the Full ME. In the right-hand column we compare the old code and the new one, and use red when the new code is worse than the old one

	MG5aMC	New code	$\frac{\text{MG5aMC}}{\text{new code}}$
Full ME	41G	31G	1.3
(i) `WF`s	0.99G (2.4%)	0.59G (1.9%)	1.7
(ii) `AMP`s	6.6G (16%)	16G (51%)	*0.42*
(iii) M_σ	3.2G (9.3%)	1.1G (3.6%)	2.9
(iv) col sum	30G (72%)	14G (44%)	2.2

In the new code, all four of these steps are changed. To understand the effect of each change, we profiled the process $gg \to 5g$ for both standard MG5aMC and for the new code, with results summarised in Table 1.

For steps (i) and (ii), our BG recursion misses many optimisations included in standard MG5aMC, so even though we use BG recursions, we actually have more `WF`s at low multiplicity but less at high multiplicity, and have many more `AMP`s in the new code than the old code. Improving this is left for future work, but for now we are mostly interested in high multiplicity processes where the colour sum dominates. As seen in Sect. 4.2 below, at low multiplicity the missed optimisations cause the new program to be slower than the standard MG5aMC one, but at high multiplicity the new program is significantly faster.

An effect of the BG recursions is to reduce how many `AMP`s go into the individual colour-ordered amplitudes M_σ. Though this part of the code was not a bottleneck, using BG recursions can improve this part of the calculation signifi-

cantly at high multiplicity, e.g. by about a factor three for $gg \to 5g$.

The biggest improvement of the new code is in the colour sum. In standard MG5aMC, the colour matrix is stored as a matrix of real numbers of double precision. The colour sum is then just the matrix multiplication of Eq. (2).

In contrast to this, the new code only explicitly stores the first row of the colour sum (for each flow). We then have a single loop over all rows using a permutation matrix of integer numbers (see Sect. 3.2 for more details). This simple change appears to more than halve the work of the colour sum, which is vital because as seen in Table 1 and Ref. [29], the colour sum in MG5aMC is one of the main bottlenecks for going to higher multiplicities. While this change definitely helps, we remind that this optimisation doesn't change the factorial-squared growth of the colour sum. On the other hand, truncating the expansion in powers of $1/N_c$ helps this issue.

4 Validation and results

Now we turn our attention to the results of this paper. We will first look at the accuracy of the $1/N_c$ colour expansion for various processes, and validate this expansion by showing that it converges to the full colour result. Next, we will consider the speed of the program and compare it with the standard version of MG5aMC.

We checked the accuracy and speed process by process in both pure QCD and mixed QCD/EW theories,[3] with a representative subset of QCD processes shown below (the mixed QCD/EW results are given in Appendix B).

As will be seen below, the LC amplitudes are in general not good enough to be used in practical purposes, the NLC amplitudes can be used to speed up phase-space integration but require special tricks/correction factors [7,56,57], while all processes studied have good accuracy already at NNLC. For the speed, we will find that the new code is faster than the old one at high enough multiplicity, but slower for low multiplicities (where the computation is not dominated by the colour-matrix).

4.1 Accuracy and precision of colour approximation

All-gluon amplitudes: We begin by considering the accuracy of the $gg \to (n-2)g$ all-gluon amplitudes, as shown in Fig. 3. In the top panel we see the average value of N^{k}LC/FC over a flat scan of phase space (using RAMBO [55]), i.e. for each phase-space point we divide the colour-truncated squared matrix element by the full squared matrix element calculated by MG5aMC, and average this over phase space.

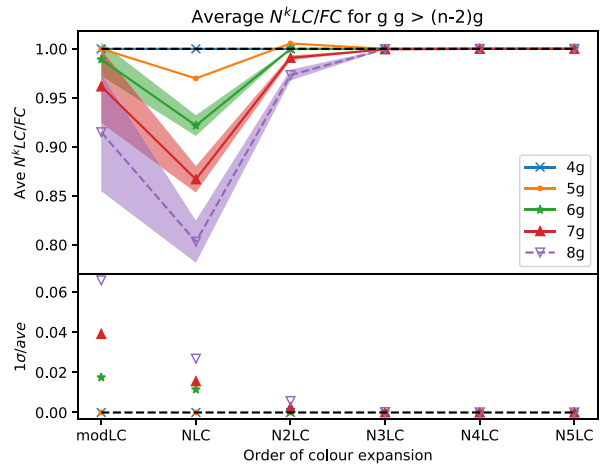

Fig. 3 Accuracy of expansion in colours $1/N_c$ in the process $gg \to (n-2)g$ compared to the value calculated in standard MG5aMC (FC). modLC is described in Eq. (11) and is a modified LC value, hence it is more accurate than NLC which is unmodified. The 8g line is dotted since the accuracy was compared to N5LC rather than FC due to MG5aMC not being able to calculate this process. The top panel shows, for a given colour order N^{k}LC, the average N^{k}LC/FC value, with the standard deviation in the shaded region. The bottom panel shows the relative error

For up to 6 gluons, the average is taken using 100,000 phase-space points, for 7 gluons using 10,000 points, and for 8 gluons using 1000 points. All processes were calculated at $\sqrt{s} = 1$ TeV. The 8g version is dotted since we could not compile the FC process in standard MG5aMC, therefore we took the N5LC value to approximate FC. Since the 8g N4LC and N5LC results already agree for the first four significant figures, this should not affect any conclusions. Such convergence is also a good validation of our colour expansion.

The shaded regions correspond to the standard deviation of the N^{k}LC/FC ratios, while the bottom panel is the percentage uncertainty, i.e. the standard deviation divided by the average. We assume a roughly Gaussian distribution,[4] and study the phase-space dependence of the accuracy and precision later in this section.

From Fig. 3, we conclude that modified LC, Eq. (11), is more accurate but less precise than NLC. Additionally, for 8 gluons the colour expansion converges by N3LC, at per-mil-level accuracy. Also, by NLC the relative precision of the expansion is much smaller than the average offset from the true value, allowing to systematically correct results if desired. We stress that when computing cross-sections and/or generating events, precise but inaccurate results can help speed up the code. This can be achieved by avoiding to compute the full matrix-element for all phase-space points, but still guaranteeing no bias after phase-space integration [3,7].

[3] Note that the decay-chain syntax is not supported.

[4] We have made some basic checks that a Gaussian assumption is reasonable.

Table 2 The average modLC/FC, LC/FC, and NLC/FC in the all-gluon colour expansion in the fundamental basis

	modLC/FC	LC/FC	NLC/FC
4g	1	0.704	1
5g	1	0.624	0.970
6g	0.989	0.549	0.922
7g	0.962	0.474	0.867
8g	0.915	0.401	0.803

To quantify the effect of modified LC, Eq. (11), we show in Table 2 the average values of both the standard LC/FC and the modified LC/FC. The NLC/FC value is also used for comparison, confirming that it is far more accurate than the true LC amplitudes, even if it is less accurate than the modLC results. Since the only difference between modLC and LC is changing the colour factor in Eq. (11), the relative (but not absolute) precision of LC and modLC are the same.

Table 2 shows that a true LC all-gluon amplitude in the fundamental basis is a very poor description of the full amplitude, being about 60% too small for the 8 gluon amplitude. The reason is likely that we are using fundamental matrices (i.e. the colour matrices of quarks) to describe the colour of gluons. Therefore, we expect e.g. the colour flow expansion to be more suited to the all gluon amplitude, since a pure gluon amplitude can be fully described with $U(3)$ gluons. Alternatively, the modLC description works very well since it uses more than just a strict expansion in colour to calculate the colour factor.

Amplitudes with a single quark pair: Next we consider QCD processes with a single quark pair using $u\bar{u} \to ng$ as a test process (see Fig. 4). We used 100,000 phase-space points for up to 5 gluons and 10,000 points for 6 gluons. In this case, the LC approximation is neither particularly accurate nor precise. At low multiplicity, it over-estimates the amplitude, while it increasingly under-estimates it starting from four gluon multiplicity. Similar to the all-gluon amplitudes in Fig. 3, the NLC relative precision is around a few percent. However, unlike the all-gluon case, the NLC amplitude is already quite accurate, being on average percent-level accurate or better for 5 or less gluons, and about 3% accurate for 6 gluons. Both the accuracy and relative precision of N2LC is at or better than about 0.1% for all studied processes.

Amplitudes with two quark pairs: To complete the pure massless QCD analysis, we study processes with two quark pairs. We take two test cases, $u\bar{u} \to d\bar{d} + ng$ and $u\bar{u} \to u\bar{u} + ng$, again using 100,000 phase-space points for all multiplicities except for the largest one, which was calculated using 10,000 points. We use two test cases here in order to study the effects of quark interference on accuracy and precision.

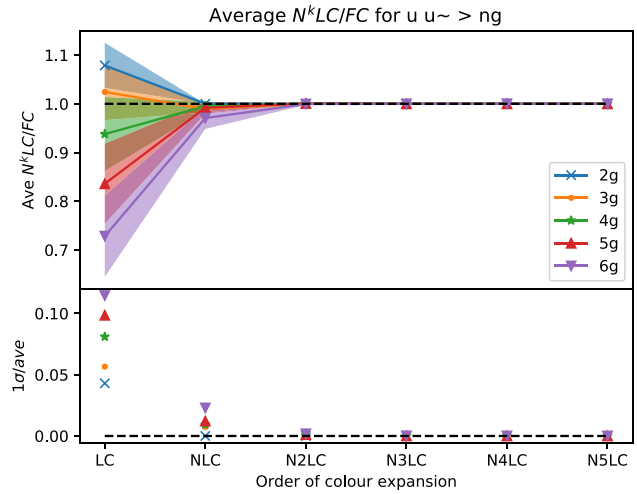

Fig. 4 Same as Fig. 3 but for the process $u\bar{u} \to ng$

As we see in Fig. 5, LC has only about 20–30% relative precision, and that for distinct quark flavours the LC value again decreases with increasing gluons. The same-flavour LC amplitudes are more precise than the distinct-flavour ones, possibly due to all kinematic amplitudes being included already at LC for the same flavour case (cf Eqs. (8) and (9) and the discussion around (12)). By NLC, the accuracy is already very good, around the percent level, with precision about 5% or better. Once again, by NNLC, the accuracy is around 0.1% or better, with precision around 0.5% or better.

Amplitudes with a top quark pair: An important process in QCD is the production of a top pair. MG5aMC can now calculate this production using the new code. As we see in Fig. 6, the LC values for $t\bar{t}$ production become very inaccurate at high multiplicity, with the $gg \to t\bar{t}4g$ matrix element being just 56% of its required value on average. However, the relative precision of about 8.7% allows this value to be systematically corrected. Indeed, such a correction for gluon-induced top production appears well motivated already for two or more final-state gluons. If the process is quark-induced, the LC relative precision is around or above 20% depending on the multiplicity.

At NLC, the results are also quite different depending on the subprocess. For the gluon-induced process, the NLC result is only 9% accurate for 4 gluons with a relative precision of about 2.9%, while the second process is accurate to within a few percent for all processes studied but has a slightly worse relative precision of up to 3.5%.

Like the previous processes, NNLC describes the results to a high accuracy and precision. All processes are described to an accuracy and precision of about a half of a percent or better for all multiplicities.

Accuracy in different parts of phase space: While Figs. 3, 4, 5 and 6 show the average accuracy of the expansion in a flat

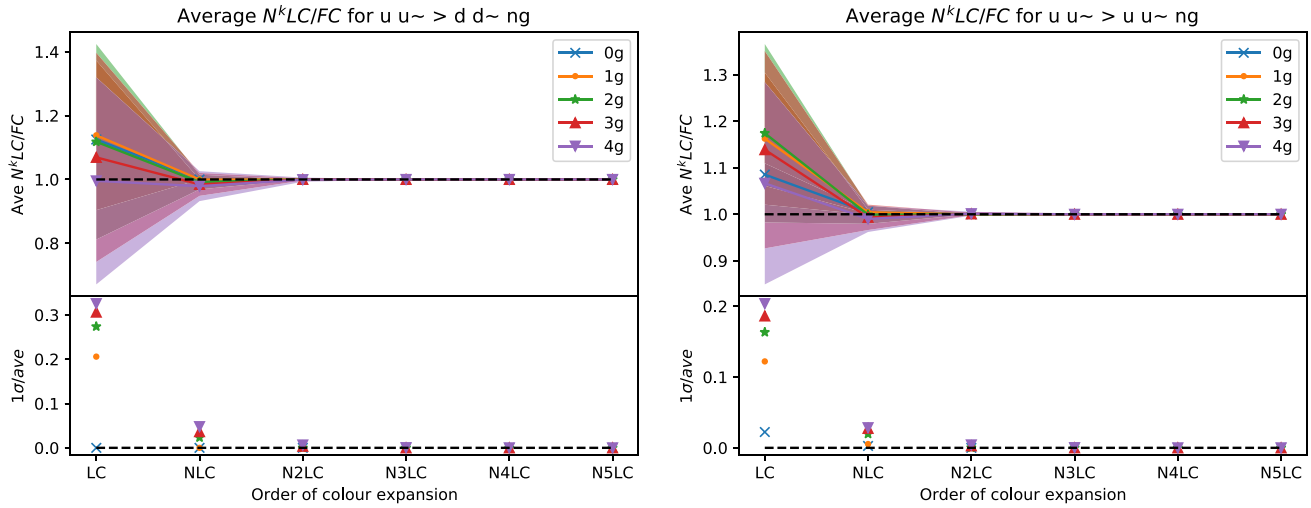

Fig. 5 Same as Fig. 3 but for the processes $u\bar{u} \to d\bar{d} + ng$ and $u\bar{u} \to u\bar{u} + ng$

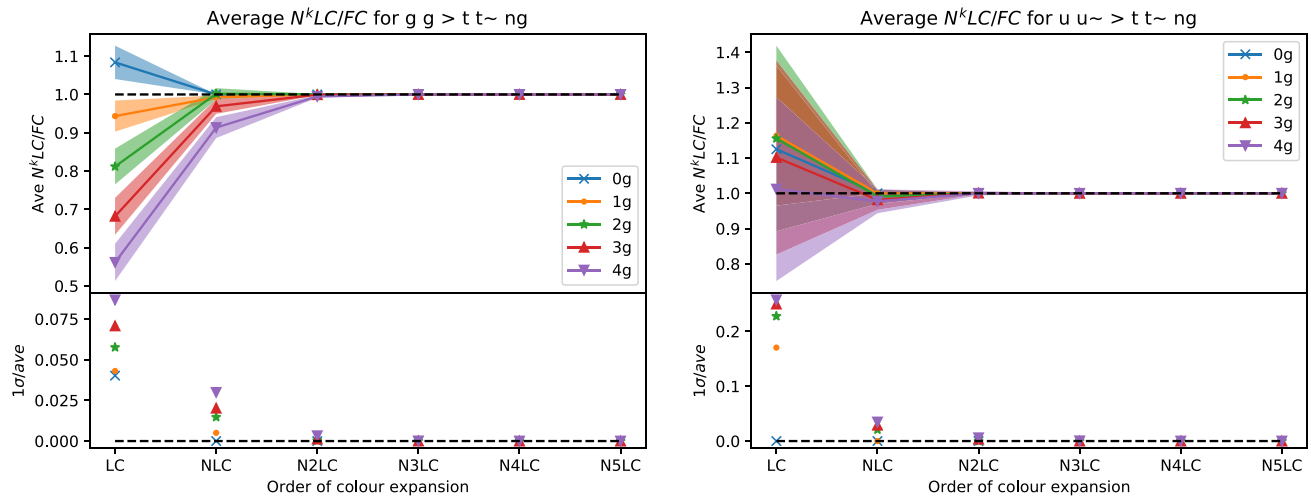

Fig. 6 Same as Fig. 3 but for the processes $gg \to t\bar{t} + ng$ and $u\bar{u} \to t\bar{t} + ng$

phase-space scan, it is also good to know if the accuracy and precision are dependent on the phase-space region. In order to check this, we looked at the processes $u\bar{u} \to 3g$ and $u\bar{u} \to 4g$ for 10^7 phase-space points produced by RAMBO. For each point, we calculated the energy fractions $x_i = 2E_i/E_{cm}$ of each particle, storing the minimum one; and calculated the cosine of the opening angle between each particle, $\cos(\theta_{ij})$, storing the maximum value (minimum angle) for each point.

As is shown in Figs. 7 and 8, the accuracy and precision, especially at LC, depends strongly on whether all particles are well-separated or not in angle. On the other hand, the energy of the softest particle appears to have little importance on the accuracy of the colour expansion. Since the accuracy and precision of LC appears to depend too much on the phase-space point, we think that LC is too crude to be used. On the other hand, NLC can be used, but may vary slightly with the opening angle of two particles, which might create an issue depending on the multiplicity and how the approximation is used.

In addition to this general scan over phase-space, it is useful to confirm that each of the colour-ordered amplitudes has the expected soft and collinear limit [48]. To do this, we created around a thousand $u\bar{u} \to 3g$ Born phase-space points, and added a fourth soft or collinear gluon. The added gluon was then made more and more soft or collinear to another parton. As we see in Fig. 9, the accuracy and precision of the colour expansion are not changed in the deeply soft or collinear limits. Therefore, the inclusion of the pole in the squared matrix element does not depend on the terms included in the colour expansion.

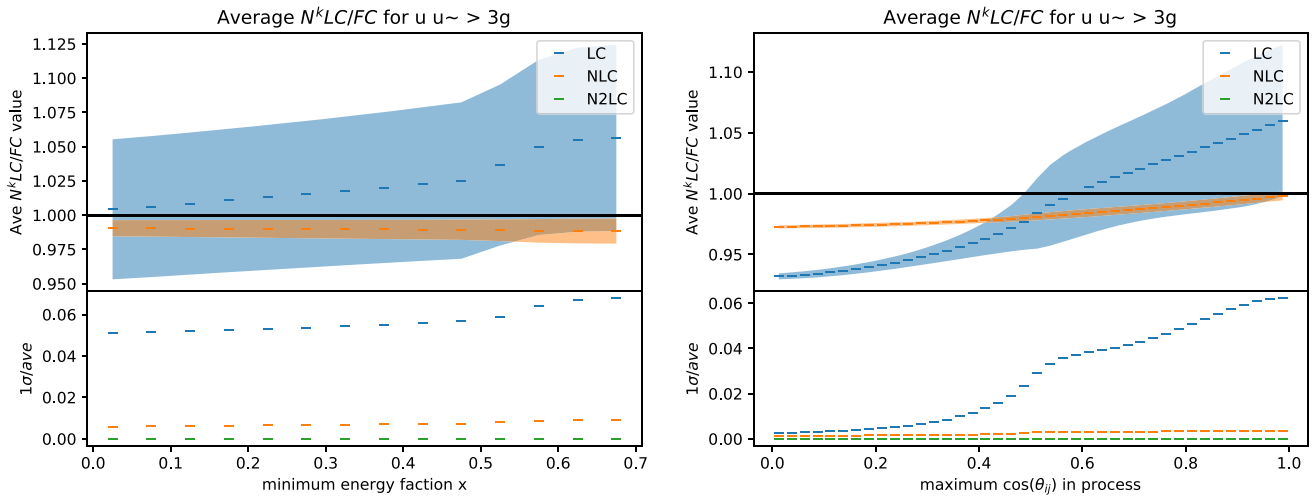

Fig. 7 Accuracy and precision of expansion in colours $1/N_c$ in the process $u\bar{u} \to 3g$ as a function of the minimum energy fraction $x = \min(2E_i/E_{cm})$ of a given particle (left) and maximum $\cos(\theta_{ij})$ between two particles (right)

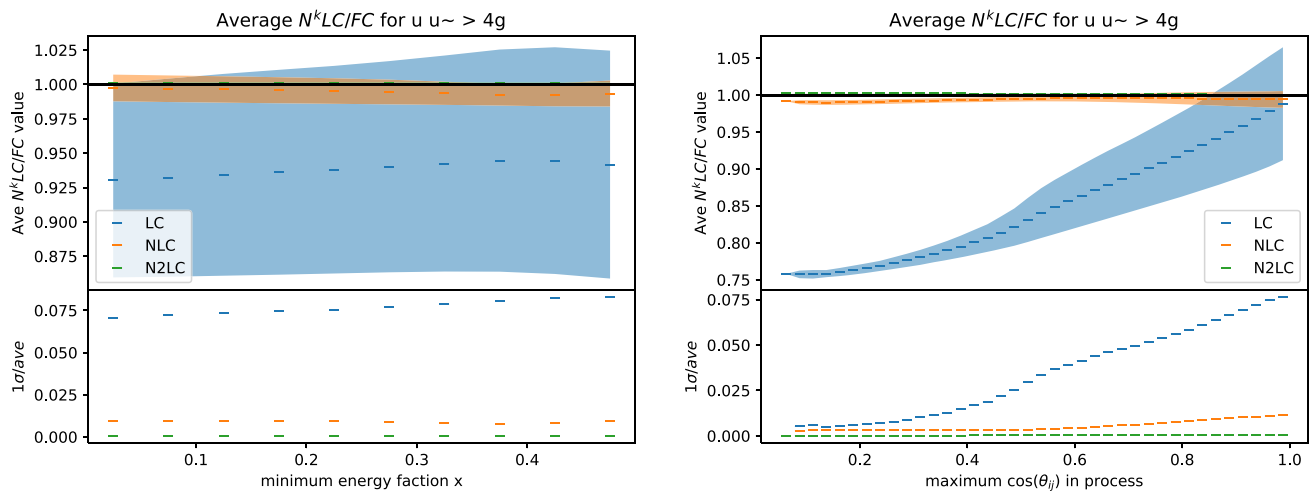

Fig. 8 Same as Fig. 7 but for the process $u\bar{u} \to 4g$

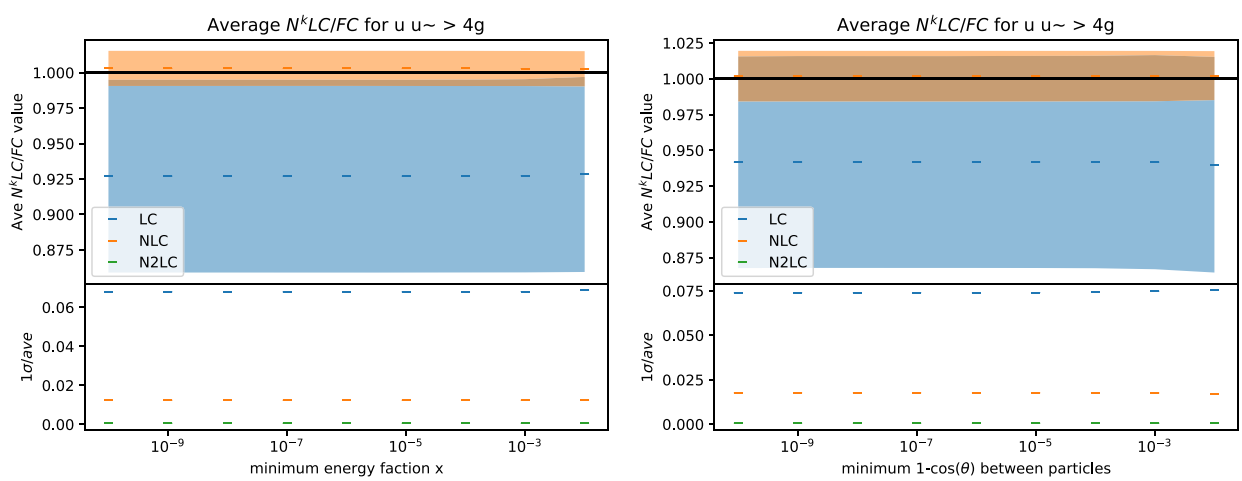

Fig. 9 The accuracy and precision of the colour expansion in the soft and collinear limits for $u\bar{u} \to 4g$

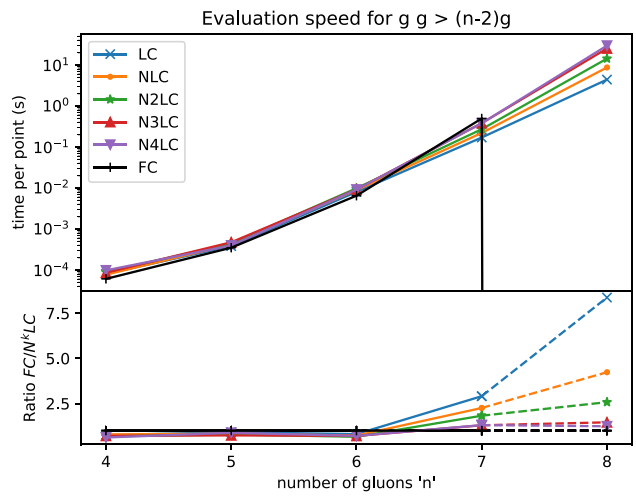

Fig. 10 Top: Speed of the process $gg \to (n-2)g$ for each colour ordering and gluon multiplicity. FC corresponds to standard MG5aMC (version 2.9.2, for standalone essentially equivalent to the latest version 3.4.1). Bottom: Ratio of the speed using standard MG5aMC to using the new code for each colour ordering. Standard MG5aMC cannot calculate $gg \to 6g$, so the right-most ratio speeds instead contains the N5LC BG speed on the denominator to show the effects of colour ordering and obtain an estimate for the true speed increase. Speed tests done on a MacBook Pro 2020 CPU i5-8257U

4.2 Speed gain

In this section we compare the speed of this new code with that of standard MG5aMC. To do this, we compare the time taken to evaluate the same matrix elements in the new and old codes (for the different sources of speed gain (and loss), see Sect. 3.4). Note that these comparisons ignore the time taken to generate and compile the code in the new and old way.[5]

All-gluon amplitudes: First we describe in detail the speed of the all-gluon amplitudes, shown in Fig. 10. The top panel of this figure shows the average time it takes to calculate a single phase-space point at each gluon multiplicity and each order of the colour expansion. The bottom panel shows the ratio

$$\frac{t_{FC}}{t_{N^kLC}^{new\,code}},$$

where t_{FC} is the time taken using standard MG5aMC, and $t_{N^kLC}^{new\,code}$ is the time using the new code (with BG recursions) with the colour matrix expanded to include all terms up to N^kLC. It allows to quantify the speed gain or loss from using the new code and truncating the colour expansion. When the order in $1/N_c$ is high enough, both the old and new codes are evaluating the same matrix element and there is no speed gain due to truncating the expansion.

At low gluon multiplicity, the new code is actually slower than standard MG5aMC, but at seven gluons the colour sum dominates sufficiently such that the new code is between 1.2 and 2.9 times faster than MG5aMC depending on the truncation of the colour expansion, and at eight gluons, we can only use the new code. We therefore significantly speed up the slowest processes, even though we slow down some faster ones.

There are several options to address the speed loss. The first is to optimise the BG recursions. As discussed in Sect. 3.3, there are many possible optimisations not yet used in the BG recursion, and implementing them should help alleviate this problem. A second option is to import the colour computation from the new code into standard MG5aMC and ignore BG recursions completely. A third option is to use some optimised BG recursions and the new colour computation at high multiplicity, and use standard MG5aMC together with the new colour computation at low multiplicity. Since BG recursions are expected to bring gains at high multiplicity, this may create a best of both worlds scenario. Exploring these options is left for future work.

Since we cannot use standard MG5aMC for 8 gluons, the speed increase for this process is compared to the N5LC BG recursion in the ratio plot at the bottom of Fig. 10, i.e. the increase shown is purely due to truncating the colour matrix. This is almost certainly an underestimate of the speed increase.

It is worth noting that since the colour matrix has size $(n-1)! \times (n-1)!$, the effect of truncating the matrix leads to larger speed gain for larger gluon multiplicity. By 8 gluons, the LC amplitude is over 8 times faster than the full answer calculated with BG recursions, while the N2LC result is over twice as fast (recall that the 8 gluon N2LC amplitude is accu-

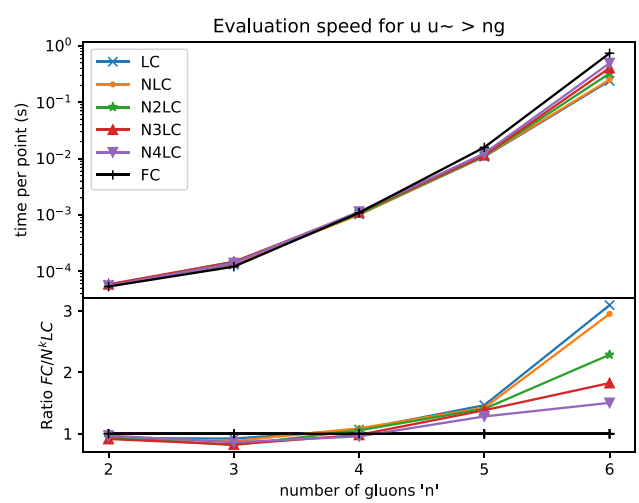

Fig. 11 Same as Fig. 10 but for $u\bar{u} \to ng$

[5] The new code is generated and compiled much faster at high multiplicity, and both codes take a similar time to generate and compile at low multiplicity.

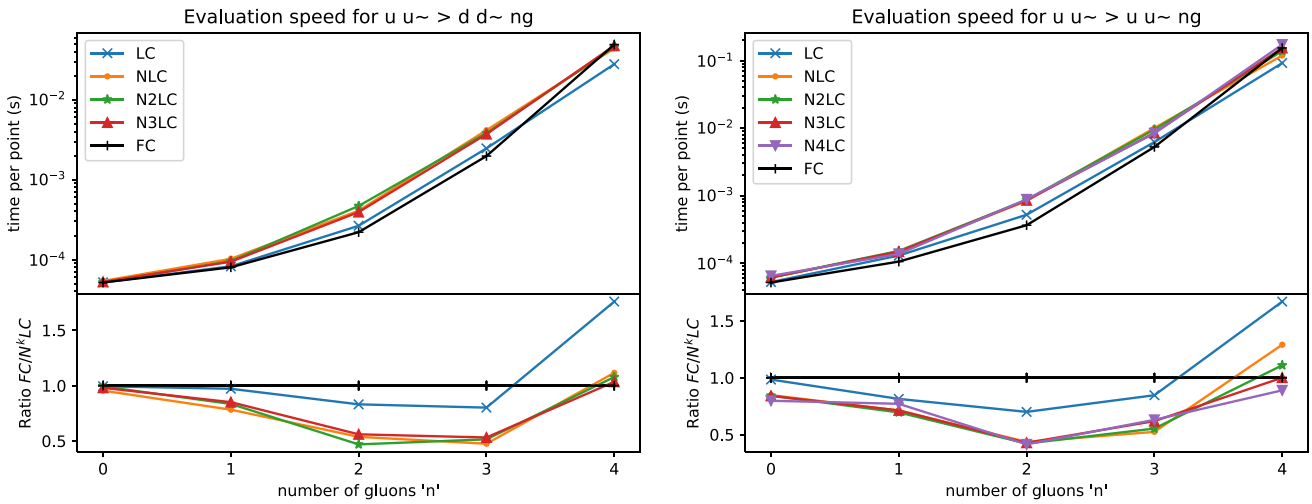

Fig. 12 Same as Fig. 10 but for $u\bar{u} \to d\bar{d} + ng$ (left) and $u\bar{u} \to u\bar{u} + ng$ (right)

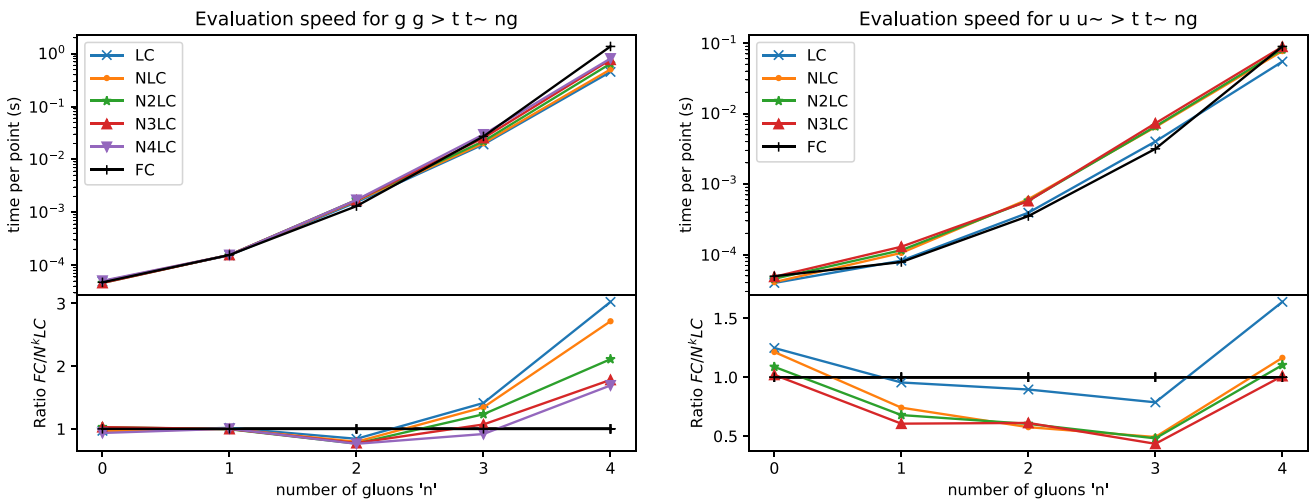

Fig. 13 Same as Fig. 10 but for $gg \to t\bar{t} + ng$ and $u\bar{u} \to t\bar{t} + ng$

rate to within a few percent and has a precision of about half a percent, see Fig. 3). At 7 gluons the LC result is about 2.4 times faster than the FC result when FC is calculated using the new code (i.e. when the only difference is the truncated colour matrix).

Amplitudes with a single quark pair: Next, we consider QCD processes with a single quark pair, again using $u\bar{u} \to ng$ as a test process (see Fig. 11). We again see that the new code is much faster at high gluon multiplicity, and a bit slower at low gluon multiplicity. This amplitude is about a factor 10 faster than the all-gluon amplitude, and has a similar level of importance (see Appendix C, Fig. 17). The 6g amplitude at N2LC is about 2.3 times faster than standard MG5aMC with an accuracy of around 0.1% and precision of around 0.5% (see Fig. 4).

Amplitudes with two quark pairs: To complete the pure massless QCD analysis, we again study $u\bar{u} \to d\bar{d} + ng$ and $u\bar{u} \to u\bar{u} + ng$ as shown in Fig. 12. This time the new code is significantly slower than standard MG5aMC for low gluon multiplicity, but again starts to become faster at high multiplicity. However, as one can seen in appendix C (Fig. 17), this process is less significant than the other massless QCD processes. Further, comparing Fig. 12 to Figs. 10 and 11, we see that multiquark amplitudes are also quicker than most other massless QCD processes, hence a speed gain or loss here is not so significant.

Amplitudes with a top quark pair: Finally, in Fig. 13, we consider the speed of pure QCD processes with a top pair. Once again, at high multiplicity (in this case four or more gluons in the final state) we see the new code becomes faster than standard MG5aMC. For less final-state gluons the old code is quicker.

5 Conclusion

In this paper, we have re-implemented the colour computation of MG5aMC and implemented BG-like recursions within MG5aMC. We now have both a more efficient way to generate QCD amplitudes, as well as a faster matrix-element evaluation at high multiplicity. In particular, MG5aMC can for the first time generate and evaluate matrix elements for $gg \to 6g$ and some other high multiplicity processes.

For the colour computation, we defined an expansion of the colour-matrix as a function of the highest power of N_c, and studied the accuracy and relative precision of the expansion for various processes. In general the LC approximation does not provide either an accurate or precise value of the full matrix-element squared, and therefore is barely usable for any practical application. The situation radically improves for NLC accuracy where the precision is typically at the percent level, even if the computation can be affected by a large bias. This approximation should be enough to speed up phase-space integration, thanks to various phase-space integration methods based on having access to fast matrix-elements [7,56,57]. For the all-gluon amplitude, the N2LC approximation is also affected by a bias. However, all other processes are precise at the per-mil level at N2LC and do not have any significant bias. In all cases, the N3LC amplitudes are extremely precise and accurate, and should be usable without corrections in many applications.

Importantly, the novel implementation of the colour sum in the new code improves the evaluation time of high-multiplicity matrix elements, even without truncating the colour expansion. If truncating the colour expansion, we can further gain in the evaluation time by using phase-space symmetry to limit the number of colour orderings required [58]. At low multiplicity, the computation of the colour-matrix is not critical, and since our implementation of the BG relation is not as optimised as standard MG5aMC, the new code is slower than the old code at these multiplicities. Such optimisation is left for future work. Additionally, like done in [58], it would be beneficial to know in advance which terms of the colour matrix contribute to which order of the expansion. This would greatly help speed up the generation of the code, allowing to go to even higher multiplicity.

This paper is an important milestone for the MG5aMC code, both by allowing higher multiplicity, and by allowing more control on the colour treatment of the computation. Now such improvement needs to be incorporated within the other types of computation offered by MG5aMC, in particular for LO/NLO cross-section/event generation for merged generation. The best approach here would require some deep change within the phase-space integrator since it is not compatible with BG recursions [59]. Independently of making these deep changes, importing the new colour computation into the main code should be fairly straightforward. This optimisation should allow to have, for high multiplicity, code faster by around 30%, thus allowing us to meet the requirement needed for HL-LHC [27,28].

Acknowledgements We would like to thank Rikkert Frederix, Malin Sjödahl, and Timea Vitos for a thorough reading of, and comments on, the manuscript. We would also like to thank Johan Alwall, Stefano Frixione, and Fabio Maltoni for discussions related to this paper. AL would like to additionally thank Malin Sjödahl for the encouragement to branch out during his PhD and do this project. This work has received funding from the European Union's Horizon 2020 research and innovation programme as part of the Marie Skłodowska-Curie Innovative Training Network MCnetITN3 (Grant agreement no. 722104). In addition, AL would like to thank his funding from the Swedish Research Council (contract number 2016-05996, as well as European Union's Horizon 2020 research and innovation programme (Grant agreement no. 668679). OM received funding from FRS-FNRS agency via the IISN maxlhc convention (4.4503.16). Computational resources have been provided by the supercomputing facilities of the Université catholique de Louvain (CISM/UCL) and the Consortium des Équipements de Calcul Intensif en Fédération Wallonie Bruxelles (CÉCI) funded by the Fond de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under convention 2.5020.11 and by the Walloon Region.

Data Availability Statement This manuscript has no associated data or the data will not be deposited. [Authors' comment: There are no associated data available.]

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.
Funded by SCOAP^3. SCOAP^3 supports the goals of the International Year of Basic Sciences for Sustainable Development.

Appendix A: Manual

In this section, we describe how to use the new code. This short manual will assume the reader is familiar with and already knows how to run MG5aMC (the unfamiliar reader is directed to [7] for the structure and main commands of MG5aMC). We remind that the recursions and the colour ordering are only available in standalone mode, such that we can only calculate squared matrix elements, and not cross sections (which would require a dedicated phase-space integrator). Both the BG recursions and new colour implementation can only be used within a colour expansion. If the user wants the full colour amplitude using the new code, they simply need to choose a high enough colour order such that the expansion finishes. For example, if the expansion natu-

rally terminates at NLC, then choosing colour order NLC, N2LC, N3LC etc. will all give the full colour result. There is no known time penalty for choosing e.g. N3LC when the expansion naturally terminates at NLC.

To switch on the new code we use the `set color_ordering` command, where `color_ordering 0` means using normal MG5aMC and is the default. If `color_ordering` is set to $k \geq 1$, then MG5aMC will calculate the $\mathrm{N}^{k-1}\mathrm{LC}$ amplitude. Only *after* setting the colour ordering to a non-zero value is it possible to use `set optimization` to toggle between BG recursion (default, `optimization 3`) and standard Feynman diagrams (`optimization 1`). Note that although `optimization 1` uses Feynman diagrams, it does *not* use the optimised version from standard MG5aMC. Therefore, using `optimization 1` will be slower than using the default BG recursions. Finally, if we want to use the modified definition of multiquark colour (see Appendix D), we can change the `LC_defn` from its default value of `fund` to `modLC`.

To make the instructions more explicit, we write here a sample card (assumed to be called `example.txt`), which will instruct MG5aMC to generate and calculate the process $pp \to 5j$ at NLC using BG recursion for the kinematics. To use it, type `./bin/mg5aMC example.txt` in the MG5aMC directory.

```
set ignore_six_quark_processes j # avoid 6 quark process
set color_ordering 2
set LC_defn fund #default: other option modLC
set optimization 3 #default: other option 1
generate p p > 5j
output standalone
launch
```

Appendix B: Accuracy and speed of additional processes

In this section we briefly repeat the analysis of Sect. 4 for QCD processes with three quark pairs and QCD processes with the addition of an electroweak boson.

Processes with three quark pairs: Similar to the two-quark amplitudes, we here distinguish between whether the quarks all have the same flavour, whether two quark lines have the same flavour, or all quarks are distinct, with the first and last of these cases shown in Fig. 14. For up to 1 gluon in the final state, we used 100,000 phase-space points. For 2 gluons we used 10,000 phase-space points. Like in the two-quark-line case (cf Fig. 5), the LC accuracy and precision is rather poor, NLC provides a good approximation, and by N2LC the approximation is very close to exact for the multiplicities studied.

Also, since we optimised for multigluon amplitudes and not for multiquark amplitudes, we found the new code to be slower than the old one for this type of process. This is unlikely to be an issue however, since processes with three quark pairs are typically very sub-leading, so this matrix element is calculated far less often compared to those in the main text (see Fig. 17).

Processes with an EW boson: As a first test case we look at Z production, using the process $u\bar{u} \to Z + ng$ (see Fig. 15). Comparing to Fig. 4, we see a similar accuracy and relative precision when the number of gluons are the same. On the other hand, comparing to Fig. 11 and in particular looking at the high multiplicity end, we see a greater improvement in speed when adding gluons, but a slightly lower speed gain if comparing overall particle multiplicity.

As a second test case we consider Z boson production with an additional quark pair (see Fig. 16). Comparing to Fig. 5 we again see that the accuracy and precision is largely driven

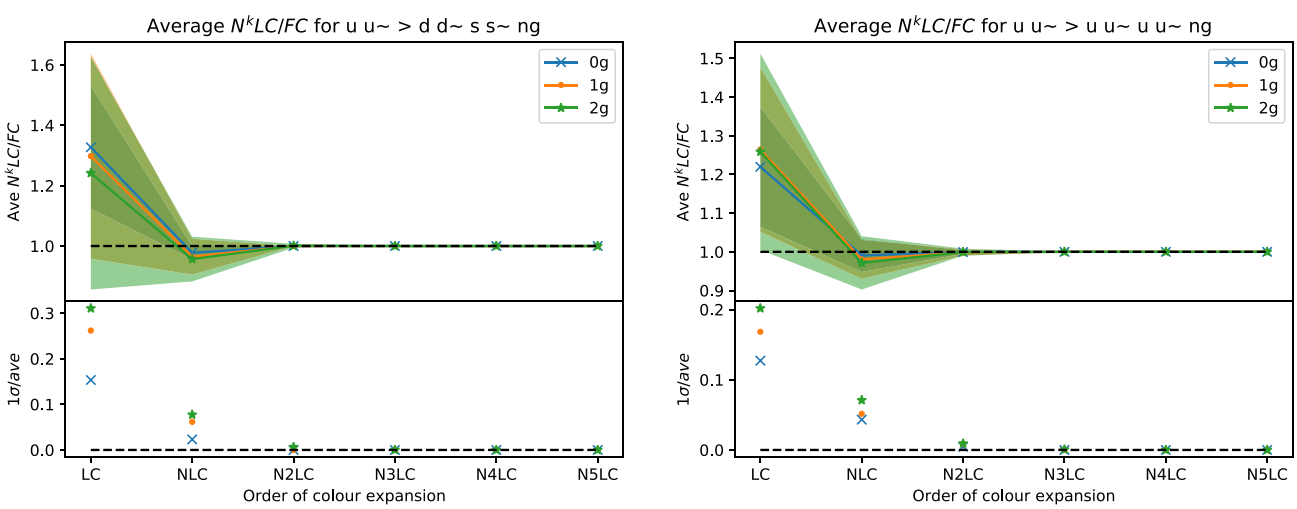

Fig. 14 Same as Fig. 3 but for the processes $u\bar{u} \to d\bar{d}s\bar{s} + ng$ and $u\bar{u} \to u\bar{u}u\bar{u} + ng$

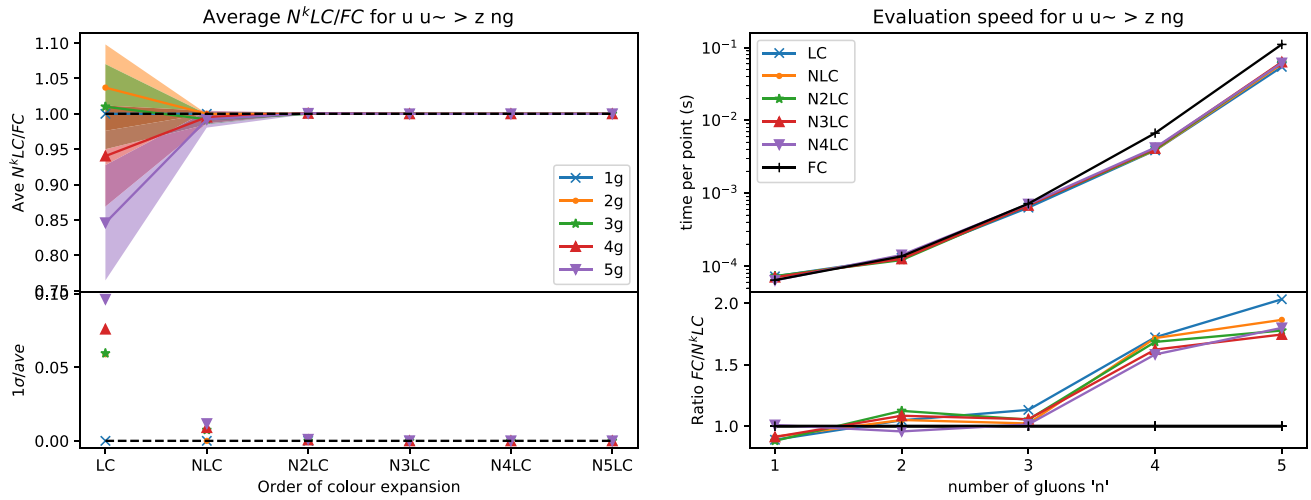

Fig. 15 Same as Figs. 3 and 10 but for $u\bar{u} \to Z + ng$

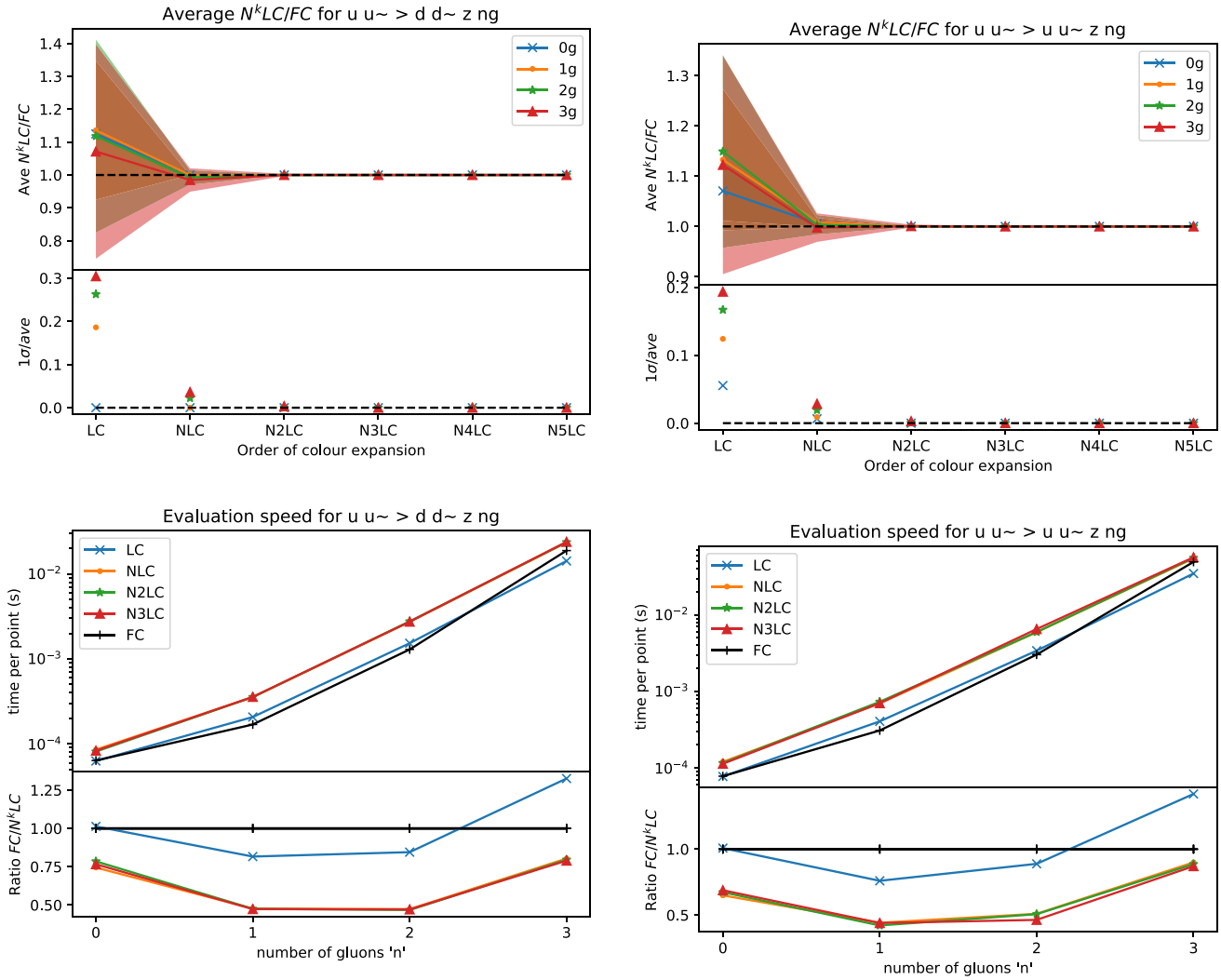

Fig. 16 Same as Figs. 3 and 10 but for $u\bar{u} \to Z\, d\bar{d} + ng$ (left) and $u\bar{u} \to Z\, u\bar{u} + ng$ (right)

by the number and nature of the QCD particles involved. Instead comparing the speed to Fig. 12, we see that this time we have a worse speed performance when adding a Z boson compared to the pure QCD multiquark case.

We conclude that the Z boson has little effect on the accuracy and precision, and that it is the QCD part of the process which is important for this. On the other hand, the Z boson has a large role to play in evaluation speed.

Similar to the Z boson, we tested the addition of a W boson by testing the processes $u\bar{d} \to W^+ + ng$, $u\bar{d} \to W^+ s\bar{s} + ng$ and $u\bar{d} \to W^+ u\bar{u} + ng$. The conclusions stated in the previous paragraph about the Z boson were found to be equally applicable to the W boson, with the exception that the W boson was found to play a larger role in evaluation speed.

Appendix C: Subprocess cross-sections in multi-jet production

In this appendix, we study the relative importance of various types of subprocesses classified by the number of quark lines present in the sub-process. Such information advises how critical it is to optimise the speed of the various contributions.

In Fig. 17, we present the tree-level cross-sections for multi-jet production, grouped via the number of quark lines present within the associated subprocess. The cross-sections are computed at partonic level as they would be within the MLM mode of MG5aMC.

The code used to perform his is the following:

```
generate p p > 2j
add process p p > 3j
add process p p > 4j
add process p p > 5j
output
launch
shower=OFF
set xqcut 10
```

As can be seen from the above set of commands, everything is default except for the value of xqcut, and therefore the only additional cut is the maximum rapidity of the jet which is set at 5. Additionally, the PDF is NNPDF 2.3 (lhaid=247000) [60]. We stress that within this procedure, no Sudakovs are included at parton-level. Those factors are normally included after the running of the parton-shower by vetoing some of the generated events. Therefore the reported cross-sections contain double-counting and should not be compared to experimental results. However, these values dictate how many events need to be generated within each category and therefore indicate the relative-importance of each category for a typical multi-jet calculation.

From Fig. 17, we can conclude that, like in any matched-merged computation, the cross-section is dominated by the lowest multiplicity, which is fast and easy to compute. However, the overall computation time is dominated by the highest multiplicity sample due to lower event generation efficiency and slower matrix-element evaluation.

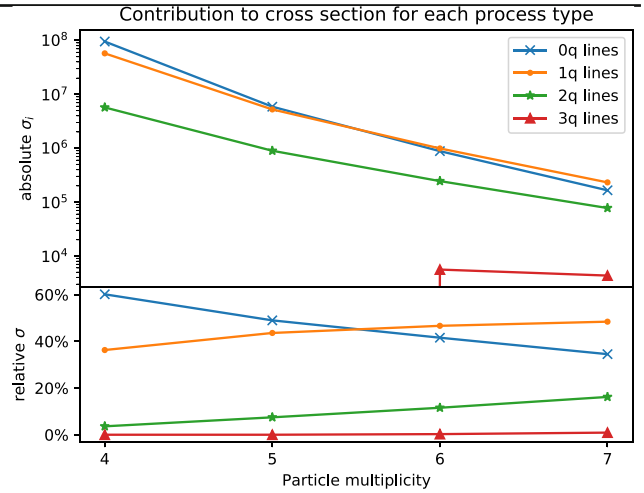

Fig. 17 MLM cross-section for each multiplicity and amplitude type without the inclusion of the Sudakov form-factor ($xqcut = 10$ GeV)

At high multiplicity, the full gluon amplitude is second to (but basically on par with) the single quark line (that includes $gg \to q\bar{q}(n-4)g$, $q\bar{q} \to (n-2)g$ and $q/\bar{q}g \to q/\bar{q}(n-3)g$). Higher numbers of quark lines are suppressed at such multiplicities, with the three-quark line being completely negligible. By extrapolating the plot for higher multiplicity, one can guess that cross sections with two quark lines will surpass the full gluon amplitude at either multiplicity 8 ($2 \to 6$ process) or 9.

Appendix D: Modified colour expansion for multiquark amplitudes

Here we describe an attempt to modify the colour expansion in multiquark amplitudes, for reasons outlined at the end of Sect. 2.1.2. It was found that this modified colour expansion does not overly help the accuracy or precision of the colour expansion, but we leave it here for the interested reader.

For two quark pairs, a strict colour expansion includes the $1/N_c$ from the $u(1)$ gluon in the expansion. This implies that only the second and third lines of Eq. (8) are included at LC if the quark lines have different flavour. Therefore, unlike for single-quark or all-gluon amplitudes, we do not include all kinematic amplitudes at least once at LC.

We therefore propose a modified colour or 'modN^{k}LC' expansion, which does not count the $1/N_c$ terms coming from the $u(1)$ gluon in the expansion, but rather includes it in the definition of the kinematic amplitude. The remaining rules of the expansion continue as before.

In other words, Eq. (8) is changed to

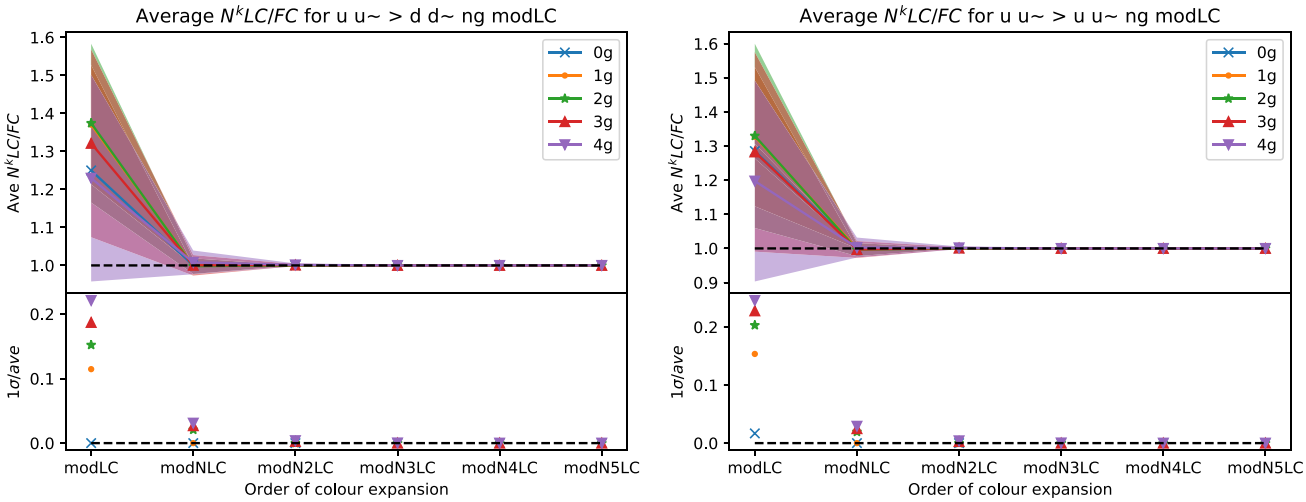

Fig. 18 Same as Fig. 5 but at modified colour

$$\hat{\mathcal{M}}(q\bar{q}Q\bar{Q}+ng) = \sum_{i=0,n}\sum_{P(1,\ldots,i)}\sum_{P(i+1,\ldots,n)}\Big[(t^1\ldots t^i)_{q\bar{Q}}(t^{i+1}\ldots t^n)_{Q\bar{q}} \times M(q,1,\ldots,i,\bar{Q},Q,i+1,\ldots,n,\bar{q}) - (t^1\ldots t^i)_{q\bar{q}}(t^{i+1}\ldots t^n)_{Q\bar{Q}} \times M^*(q,1,\ldots,i,\bar{q},Q,i+1,\ldots,n,\bar{Q})\Big], \quad (20)$$

where we defined $M^* = \frac{1}{N_c}M$. In this way, the colour matrix and expansion ignores the colour suppression of the $u(1)$ gluon and includes all kinematic amplitudes already at LC.

As we see by comparing Fig. 18 to Fig. 5, modified colour decreases the accuracy, but has up to half the relative uncertainty at LC if the quarks have different flavours. Despite this positive effect, the modLC amplitudes are not precise enough for practical corrections. In addition, the same-flavour precision actually gets worse using this colour expansion.

The speed of the modified expansion was found to be slower at each order but the same at the end of the expansion as expected.

References

1. A. Kanaki, C.G. Papadopoulos, Comput. Phys. Commun. **132**, 306 (2000). https://doi.org/10.1016/S0010-4655(00)00151-X
2. M. Moretti, T. Ohl, J. Reuter, O'Mega: an optimizing matrix element generator (2001). arXiv:hep-ph/0102195
3. F. Krauss, R. Kuhn, G. Soff, JHEP **02**, 044 (2002). https://doi.org/10.1088/1126-6708/2002/02/044
4. M.L. Mangano, M. Moretti, F. Piccinini, R. Pittau, A.D. Polosa, JHEP **07**, 001 (2003). https://doi.org/10.1088/1126-6708/2003/07/001
5. T. Gleisberg, S. Hoeche, JHEP **12**, 039 (2008). https://doi.org/10.1088/1126-6708/2008/12/039
6. C.F. Berger, Z. Bern, L.J. Dixon, F. Febres Cordero, D. Forde, H. Ita, D.A. Kosower, D. Maitre, Phys. Rev. D **78**, 036003 (2008). https://doi.org/10.1103/PhysRevD.78.036003
7. J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer, H.S. Shao, T. Stelzer, P. Torrielli, M. Zaro, JHEP **07**, 079 (2014). https://doi.org/10.1007/JHEP07(2014)079
8. W. Kilian, T. Ohl, J. Reuter, Eur. Phys. J. C **71**, 1742 (2011). https://doi.org/10.1140/epjc/s10052-011-1742-y
9. A. Belyaev, N.D. Christensen, A. Pukhov, Comput. Phys. Commun. **184**, 1729 (2013). https://doi.org/10.1016/j.cpc.2013.01.014
10. T. Hahn, Comput. Phys. Commun. **140**, 418 (2001). https://doi.org/10.1016/S0010-4655(01)00290-9
11. A. Cafarella, C.G. Papadopoulos, M. Worek, Comput. Phys. Commun. **180**, 1941 (2009). https://doi.org/10.1016/j.cpc.2009.04.023
12. J. Bellm et al., Eur. Phys. J. C **76**(4), 196 (2016). https://doi.org/10.1140/epjc/s10052-016-4018-8
13. F. Cascioli, P. Maierhofer, S. Pozzorini, Phys. Rev. Lett. **108**, 111601 (2012). https://doi.org/10.1103/PhysRevLett.108.111601
14. G. Bevilacqua, M. Czakon, M.V. Garzelli, A. van Hameren, A. Kardos, C.G. Papadopoulos, R. Pittau, M. Worek, Comput. Phys. Commun. **184**, 986 (2013). https://doi.org/10.1016/j.cpc.2012.10.033
15. S. Badger, B. Biedermann, P. Uwer, V. Yundin, Comput. Phys. Commun. **184**, 1981 (2013). https://doi.org/10.1016/j.cpc.2013.03.018
16. G. Cullen et al., Eur. Phys. J. C **74**(8), 3001 (2014). https://doi.org/10.1140/epjc/s10052-014-3001-5
17. S. Actis, A. Denner, L. Hofer, J.N. Lang, A. Scharf, S. Uccirati, Comput. Phys. Commun. **214**, 140 (2017). https://doi.org/10.1016/j.cpc.2017.01.004
18. A. Denner, J.N. Lang, S. Uccirati, Comput. Phys. Commun. **224**, 346 (2018). https://doi.org/10.1016/j.cpc.2017.11.013
19. S. Honeywell, S. Quackenbush, L. Reina, C. Reuschle, Comput. Phys. Commun. **257**, 107284 (2020). https://doi.org/10.1016/j.cpc.2020.107284
20. F. Buccioni, J.N. Lang, J.M. Lindert, P. Maierhöfer, S. Pozzorini, H. Zhang, M.F. Zoller, Eur. Phys. J. C **79**(10), 866 (2019). https://doi.org/10.1140/epjc/s10052-019-7306-2
21. V. Hirschi, R. Frederix, S. Frixione, M.V. Garzelli, F. Maltoni, R. Pittau, JHEP **05**, 044 (2011). https://doi.org/10.1007/JHEP05(2011)044
22. A. Denner, S. Dittmaier, L. Hofer, Comput. Phys. Commun. **212**, 220 (2017). https://doi.org/10.1016/j.cpc.2016.10.013

23. V. Hirschi, T. Peraro, JHEP **06**, 060 (2016). https://doi.org/10.1007/JHEP06(2016)060
24. G. Ossola, C.G. Papadopoulos, R. Pittau, JHEP **03**, 042 (2008). https://doi.org/10.1088/1126-6708/2008/03/042
25. A. van Hameren, Comput. Phys. Commun. **182**, 2427 (2011). https://doi.org/10.1016/j.cpc.2011.06.011
26. R.K. Ellis, G. Zanderighi, JHEP **02**, 002 (2008). https://doi.org/10.1088/1126-6708/2008/02/002
27. T. Aarrestad et al., HL-LHC computing review: common tools and community software (2020). https://doi.org/10.5281/zenodo.4009114
28. A. Collaboration, (2022) ATLAS software and computing HL-LHC roadmap. Technical report, CERN, Geneva. http://cds.cern.ch/record/2802918
29. O. Mattelaer, K. Ostrolenk, Eur. Phys. J. C **81**(5), 435 (2021). https://doi.org/10.1140/epjc/s10052-021-09204-7
30. A. Lifson, C. Reuschle, M. Sjodahl, Eur. Phys. J. C **80**(11), 1006 (2020). https://doi.org/10.1140/epjc/s10052-020-8260-8
31. J. Alnefjord, A. Lifson, C. Reuschle, M. Sjodahl, Eur. Phys. J. C **81**(4), 371 (2021). https://doi.org/10.1140/epjc/s10052-021-09055-2
32. A. Lifson, M. Sjodahl, Z. Wettersten, Eur. Phys. J. C **82**(6), 535 (2022). https://doi.org/10.1140/epjc/s10052-022-10455-1
33. D. Maître, H. Truong, JHEP **11**, 066 (2021). https://doi.org/10.1007/JHEP11(2021)066
34. A. Ballestrero, E. Maina, Phys. Lett. B **350**, 225 (1995). https://doi.org/10.1016/0370-2693(95)00351-K
35. F.A. Berends, W.T. Giele, Nucl. Phys. B **306**, 759 (1988). https://doi.org/10.1016/0550-3213(88)90442-7
36. D.A. Kosower, Nucl. Phys. B **335**, 23 (1990). https://doi.org/10.1016/0550-3213(90)90167-C
37. C. Schwinn, S. Weinzierl, JHEP **05**, 006 (2005). https://doi.org/10.1088/1126-6708/2005/05/006
38. R. Britto, F. Cachazo, B. Feng, Nucl. Phys. B **715**, 499 (2005). https://doi.org/10.1016/j.nuclphysb.2005.02.030
39. R. Britto, F. Cachazo, B. Feng, E. Witten, Phys. Rev. Lett. **94**, 181602 (2005). https://doi.org/10.1103/PhysRevLett.94.181602
40. F. Cachazo, P. Svrcek, E. Witten, JHEP **09**, 006 (2004). https://doi.org/10.1088/1126-6708/2004/09/006
41. M. Dinsdale, M. Ternick, S. Weinzierl, JHEP **03**, 056 (2006). https://doi.org/10.1088/1126-6708/2006/03/056
42. S. Badger, B. Biedermann, L. Hackl, J. Plefka, T. Schuster, P. Uwer, Phys. Rev. D **87**(3), 034011 (2013). https://doi.org/10.1103/PhysRevD.87.034011
43. T. Gleisberg, S. Hoeche, F. Krauss, R. Matyszkiewicz, How to calculate colourful cross sections efficiently (2008). arXiv:0808.3672 [hep-ph]
44. S. Keppeler, M. Sjodahl, JHEP **09**, 124 (2012). https://doi.org/10.1007/JHEP09(2012)124
45. M. Sjodahl, J. Thorén, JHEP **09**, 055 (2015). https://doi.org/10.1007/JHEP09(2015)055
46. M. Sjodahl, J. Thorén, JHEP **11**, 198 (2018). https://doi.org/10.1007/JHEP11(2018)198
47. G. 't Hooft, Nucl. Phys. B **72**, 461 (1974). https://doi.org/10.1016/0550-3213(74)90154-0
48. M.L. Mangano, S.J. Parke, Z. Xu, Nucl. Phys. B **298**, 653 (1988). https://doi.org/10.1016/0550-3213(88)90001-6
49. V. Del Duca, A. Frizzo, F. Maltoni, Nucl. Phys. B **568**, 211 (2000). https://doi.org/10.1016/S0550-3213(99)00657-4
50. V. Del Duca, L.J. Dixon, F. Maltoni, Nucl. Phys. B **571**, 51 (2000). https://doi.org/10.1016/S0550-3213(99)00809-3
51. F. Maltoni, K. Paul, T. Stelzer, S. Willenbrock, Phys. Rev. D **67**, 014026 (2003). https://doi.org/10.1103/PhysRevD.67.014026
52. S. Weinzierl, Eur. Phys. J. C **45**, 745 (2006). https://doi.org/10.1140/epjc/s2005-02467-6
53. C. Reuschle, S. Weinzierl, Phys. Rev. D **88**(10), 105020 (2013). https://doi.org/10.1103/PhysRevD.88.105020
54. M.L. Mangano, S.J. Parke, Phys. Rep. **200**, 301 (1991). https://doi.org/10.1016/0370-1573(91)90091-Y
55. R. Kleiss, W.J. Stirling, S.D. Ellis, Comput. Phys. Commun. **40**, 359 (1986). https://doi.org/10.1016/0010-4655(86)90119-0
56. K. Danziger, T. Janßen, S. Schumann, F. Siegert, SciPost Phys. **12**, 164 (2022). https://doi.org/10.21468/SciPostPhys.12.5.164
57. S. Weinzierl, Introduction to Monte Carlo methods (2000). arXiv:hep-ph/0006269
58. R. Frederix, T. Vitos, JHEP **12**, 157 (2021). https://doi.org/10.1007/JHEP12(2021)157
59. F. Maltoni, T. Stelzer, JHEP **02**, 027 (2003). https://doi.org/10.1088/1126-6708/2003/02/027
60. R.D. Ball, V. Bertone, S. Carrazza, L. Del Debbio, S. Forte, A. Guffanti, N.P. Hartland, J. Rojo, Nucl. Phys. B **877**, 290 (2013). https://doi.org/10.1016/j.nuclphysb.2013.10.010