Eur. Phys. J. C (2021) 81:1117
https://doi.org/10.1140/epjc/s10052-021-09885-0

THE EUROPEAN
PHYSICAL JOURNAL C

Special Article - Tools for Experiment and Theory

Accelerating LHC phenomenology with analytic one-loop amplitudes

A C++ interface to MCFM

John M. Campbell[1,a], **Stefan Höche**[1,b], **Christian T. Preuss**[2,c]

[1] Fermi National Accelerator Laboratory, Batavia, IL 60510, USA
[2] School of Physics and Astronomy, Monash University, Wellington Road, Clayton, VIC 3800, Australia

Received: 21 July 2021 / Accepted: 27 November 2021 / Published online: 19 December 2021

Abstract The evaluation of one-loop matrix elements is one of the main bottlenecks in precision calculations for the high-luminosity phase of the Large Hadron Collider. To alleviate this problem, a new C++ interface to the MCFM parton-level Monte Carlo is introduced, giving access to an extensive library of analytic results for one-loop amplitudes. Timing comparisons are presented for a large set of Standard Model processes. These are relevant for high-statistics event simulation in the context of experimental analyses and precision fixed-order computations.

1 Introduction

Many measurements at particle colliders can only be made with the help of precise Standard Model predictions, which are typically derived using fixed-order perturbation theory at the next-to-leading order (NLO) or next-to-next-to-leading order (NNLO) in the strong and/or electroweak coupling. Unitarity-based techniques and improvements in tensor reduction during the past two decades have enabled the computation of many new one-loop matrix elements, often using fully numeric techniques [1–19]. The algorithmic appeal and comparable simplicity of the novel approaches has also led to the partial automation of the computation of one-loop matrix elements in arbitrary theories, including effective field theories that encapsulate the phenomenology of a broad range of additions to the Standard Model [20,21]. With this "NLO revolution" precision phenomenology has entered a new era.

It has become clear, however, that the fully numeric computation of one-loop matrix elements is not without its drawbacks, the most relevant being a relatively large computational complexity. While the best methods exhibit good scaling with the number of final-state particles and are the only means to perform very high multiplicity calculations, it is prudent to resort to known analytic results whenever they are available and computational resources are scarce. The problem has become pressing due to the fact that the computing power on the Worldwide LHC Computing Grid (WLCG) is projected to fall short of the demand by at least a factor two in the high-luminosity phase of the Large Hadron Collider (LHC) [22–25]. Moreover, most techniques for fully differential NNLO calculations rely on the fast and numerically stable evaluation of one-loop results in infrared-singular regions of phase space, further increasing the demand for efficient one-loop computations [26,27].

In this letter, we report on an extension of the well-known NLO parton-level program MCFM [27–30], which allows the one-loop matrix elements in MCFM to be accessed using the Binoth Les Houches Accord (BLHA) [31,32] via a direct C++ interface.[1] This is in the same spirit as the BLHA interface to the BLACKHAT library [1], which gives access to analytic matrix elements for $V+$ jet(s), $\gamma\gamma$ (+jet) and di-(tri-)jet production. We have constructed the new interface for the most relevant Standard-Model processes available in MCFM, representing a selection of $2 \to n$ processes with $n \leq 4$. As a proof of generality, we have implemented it in the SHERPA [33] and PYTHIA [34] event generation frameworks.[2] We test the newly developed methods in both a stand-alone setup and

[1] The source code is available at gitlab.com/mcfm-team/releases.

[2] The PYTHIA version has been tested in the context of NLO matrix-element corrections (cf. [35,36]) in the VINCIA shower [37]. The implementation of NLO MECs in VINCIA and the MCFM interface are planned to be made public in a future PYTHIA 8.3 release.

[a] e-mail: johnmc@fnal.gov
[b] e-mail: shoeche@fnal.gov
[c] e-mail: christian.preuss@monash.edu (corresponding author)

a typical setup of the SHERPA event generator, and summarize the speed gains in comparison to automated one-loop programs.

2 Available processes

The Standard Model processes currently available through the MCFM one-loop interface are listed in Table 1, with additional processes available in the Higgs effective theory shown in Table 2. All processes are implemented in a crossing-invariant fashion. As well as processes available in the most recent version of the MCFM code (v10.0), the interface also allows access to previously unreleased matrix elements for $pp \to \gamma jj$ [38] and di-jet production. Further processes listed in the MCFM manual [39] may be included upon request.

Table 1 Processes available in the Standard Model

Process	Order EW	Order QCD	References
$pp \to \ell^+\ell^-$	2	1	–
$pp \to \ell^+\ell^- j$	2	2	[42,43]
$pp \to \ell^+\ell^- jj$	2	3	[42,43]
$pp \to \ell^\pm \nu_\ell$	2	1	–
$pp \to \ell^\pm \nu_\ell j$	2	2	[42,43]
$pp \to \ell^\pm \nu_\ell jj$	2	3	[42,43]
$pp \to h_0$	1	2	–
$pp \to h_0 j$	1	3	[44]
$pp \to h_0 jj$	1	4	[45,46]
$pp \to h_0 h_0$	2	2	[47]
$pp \to \ell^+\ell^- h_0$	3	1	–
$pp \to \ell^+\ell^- h_0 j$	3	2	[48]
$pp \to \ell^\pm \nu_\ell h_0$	3	1	–
$pp \to \ell^\pm \nu_\ell h_0 j$	3	2	[48]
$pp \to \gamma j$	1	2	[49,50]
$pp \to \gamma jj$	1	3	[38]
$pp \to \gamma\gamma$	2	1	–
$gg \to \gamma\gamma$	2	2	[51]
$pp \to \gamma\gamma j$	2	2	[52]
$pp \to \gamma\gamma\gamma$	3	1	[52]
$pp \to \gamma\gamma\gamma\gamma$	4	1	[53]
$pp \to \ell^+\ell^-\gamma$	3	1	[28,54]
$pp \to \ell^\pm \nu_\ell \gamma$	3	1	[28,54]
$pp \to \nu_\ell \bar{\nu}_\ell \gamma$	3	1	[28,54]
$pp \to \ell^+\ell'^- \nu_\ell \bar{\nu}_{\ell'}$	4	1	[28,54]
$pp \to \ell^+\ell^- \nu_{\ell'} \bar{\nu}_{\ell'}$	4	1	[28,54]
$pp \to \ell^+\ell^-\ell'^+\ell'^-$	4	1	[28,54]
$pp \to \ell^+\ell^+\ell^-\ell^-$	4	1	[28,54]
$pp \to \ell^+\ell^-\ell'^\pm \nu_{\ell'}$	4	1	[28,54]
$pp \to \ell^\pm \nu_\ell \nu_{\ell'} \bar{\nu}_{\ell'}$	4	1	[28,54]
$pp \to t\bar{t}$	0	3	[55]
$pp \to jj$	0	3	[56]

Table 2 Processes available in the Higgs EFT

Process	Order EW	Order QCD	References
$pp \to h_0$	1	2	–
$pp \to h_0 j$	1	3	[57]
$pp \to h_0 jj$	1	4	[58–65]

In assembling the interface we have modified the original MCFM routines such that, as far as possible, overhead associated with the calculation of all partonic channels – as required for the normal operation of the MCFM code – is avoided, and only the specific channel that is requested is computed. Additionally, all matrix elements are calculated using the complex-mass scheme [40,41] and a non-diagonal form of the CKM matrix may be specified in the interface. In general, effects due to loops containing a massive top quark are fully taken into account, with the additional requirement that the width of the top quark is set to zero.[3] The intent is that the interface can therefore be used as a direct replacement for a numerical one-loop provider (OLP). We have checked, on a point-by-point basis, that the one-loop matrix elements returned by the interface agree perfectly with those provided by OPENLOOPS2, RECOLA2 and MADLOOP5. A brief overview of the structure of the interface is given in Appendix B.

3 Timing benchmarks

To gauge the efficiency gains compared to automated one-loop providers, we compare the evaluation time in MCFM against OPENLOOPS2, RECOLA2, and MADLOOP5 using their default settings. In particular, we neither tune nor deactivate their stability systems. The tests are conducted in three stages. First, we test the CPU time needed for the evaluation of loop matrix elements at single phase space points; in a second stage, we test the speedup in the calculation of Born-plus-virtual contributions of NLO calculations using realistic setups; lastly, we compare the CPU time of the different OLPs in a realistic multi-jet merged calculation. In all cases, we estimate the dependence on the computing hardware by running all tests on a total of four different CPUs, namely

[3] An approximate form for top-quark loops is used for the processes $qq' \to \gamma qq'$, $qg \to \gamma qg$, $q\bar{q} \to e^-e^+hg$ and $gg \to gg$, so that strict agreement with other OLPs for these processes requires the removal of the top-quark loops in those.

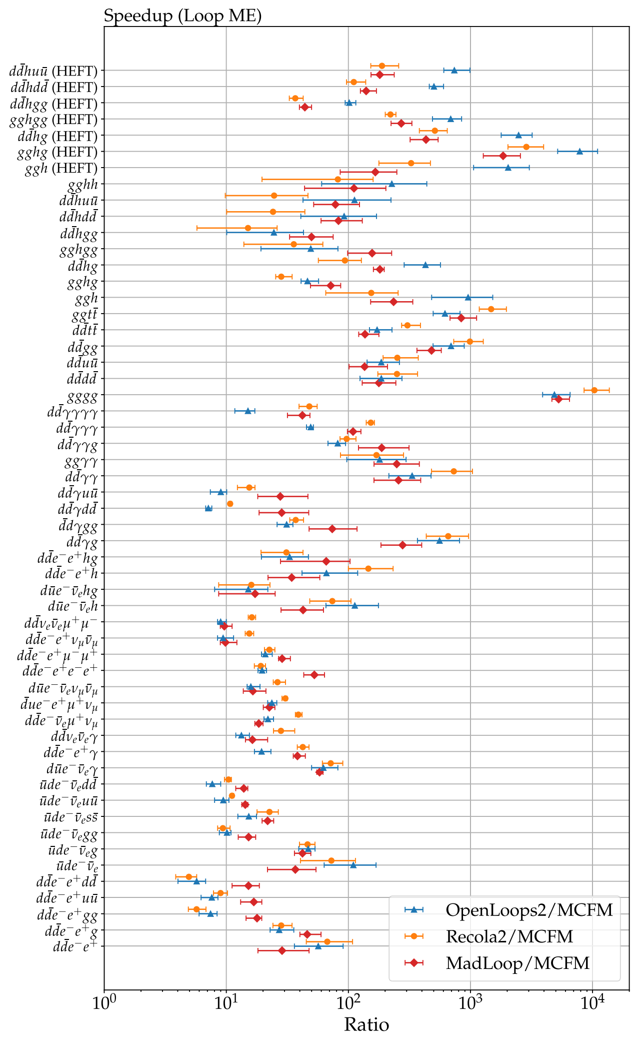


Fig. 1 CPU time ratio of OpenLoops, Recola, and MadLoop5 to Mcfm at the level of loop matrix elements

– Intel® Xeon® E5-2650 v2 (2.60GHz, 20MB)
– Intel® Xeon® Gold 6150 (2.70GHz, 24.75MB)
– Intel® Xeon® Platinum 8260 (2.40GHz, 35.75MB)
– Intel® Xeon Phi™ 7210 (1.30GHz, 32MB)

For the timing tests at matrix-element level, we use standalone interfaces to the respective tools and sample phase space points flatly using the Rambo algorithm [66]. We do not include the time needed for phase-space point generation in our results, and we evaluate a factor 10 more phase-space points in Mcfm in order to obtain more accurate timing measurements at low final-state multiplicity. The main programs and scripts we used for this set of tests are publicly available[1]. The results are collected in Fig. 1, where we show all distinct partonic configurations that contribute to the processes listed in Tables 1 and 2. We use the average across the different CPUs as the central value, while the error bars range from the minimal to the maximal value. The interface to Mcfm typically evaluates matrix elements a factor 10–100 faster than the numerical one-loop providers, although for a handful of (low multiplicity) cases this factor can be in the 1,000–10,000 range.

We perform a second set of tests, using the Sherpa event generator [33,67], its existing OLP interfaces to OpenLoops2 and Recola2[4],[5] [68], and a dedicated interface to MadLoop5[6]. With these interfaces we test the speedup in the calculation of the Born-like contributions to a typical NLO computation for the LHC at $\sqrt{s} = 14$ TeV, involving the loop matrix elements in Tables 1 and 2. The scale choices and phase-space cuts used in these calculations are listed in Appendix A. Figure 2 shows the respective timing ratios. It is apparent that the large gains observed in Fig. 1 persist in this setup, because the Born-like contributions to the NLO cross section consist of the Born, integrated subtraction terms, collinear mass factorization counterterms and virtual corrections (BVI), and the timing is dominated by the loop matrix elements if at least one parton is present in the final state at Born level. The usage of Mcfm speeds up the calculation by a large factor compared to the automated OLPs, with the exception of very simple processes, such as $pp \to \ell\bar{\ell}$, $pp \to h$, etc., where the overhead from process management and integration in Sherpa dominates. To assess this overhead we also compute the timing ratios after subtracting the time that the Sherpa computation would take without a loop matrix element. The corresponding results are shown in a lighter shade and confirm that the Sherpa overhead is significant at low multiplicity and becomes irrelevant at higher multiplicity.

In the final set of tests we investigate a typical use case in the context of parton-level event generation for LHC experiments. We use the Sherpa event generator in a multi-jet merging setup for $pp \to W$+jets and $pp \to Z$+jets [69] at $\sqrt{s} = 8$ TeV, with a jet separation cut of $Q_{\text{cut}} = 20$ GeV, and a maximum number of five final state jets at the matrix-element level. Up to two-jet final states are computed at NLO accuracy. In this use case, the gains observed in Figs. 1 and 2 will be greatly diminished, because the timing is dominated by the event generation efficiency for the highest multiplicity tree-level matrix elements [70] and influenced by particle-level event generation as well as the clustering algorithm needed for multi-jet merging.[7] We make use of the efficiency improvements described in Ref. [73], in particular neglect-

[4] At the time of this study, Sherpa provided an interface to Recola's Standard Model implementation only.

[5] For $V + 2j$ processes, we use Recola1 due to compatibility issues with Recola2.

[6] We thank Valentin Hirschi for his help in constructing a dedicated MadLoop5 interface to Sherpa. This interface will be described in detail elsewhere.

[7] In this study we do not address the question of additional timing overhead due to NLO electroweak corrections or PDF reweighting [71],

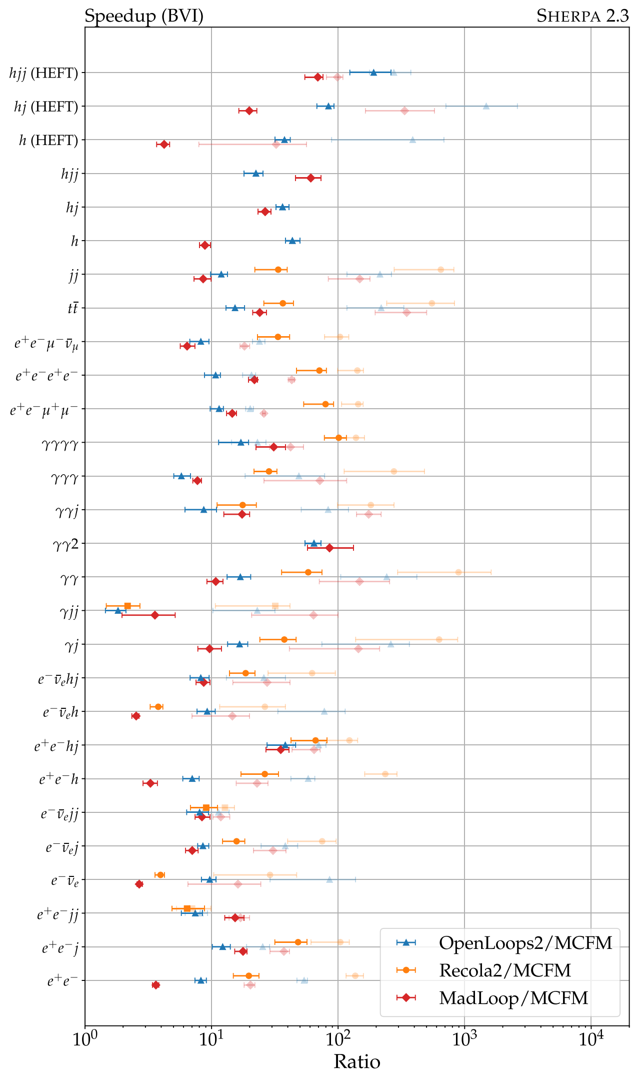


Fig. 2 CPU time ratio of OpenLoops2, Recola2 [5], and MadLoop5 to Mcfm at the level of Born-like contributions to the NLO cross section (BVI)

ing color and spin correlations in the S-MC@NLO matching procedure [74]. We do not include underlying event simulation or hadronization. The results in Table 3 still show a fairly substantial speedup when using Mcfm. We point out that a higher gain could be achieved by also making use of Mcfm's implementation of analytic matrix elements for real-emission corrections and Catani-Seymour dipole terms.

We close this section with a direct comparison of the CPU time needed for the calculation of Drell-Yan processes with one and two jets using Sherpa and Mcfm, up to a target precision on the integration of 0.1% (one jet) or 0.3% (two jets). The center-of-mass energy is $\sqrt{s} = 14$ TeV, and the scale choices and cuts are listed in Appendix A. The results are shown in Table 4. As might be expected when comparing a dedicated parton-level code with a general-purpose particle-level generator, Mcfm is substantially faster than Sherpa for the evaluation of all contributions to the NLO calculation. These results indicate a few avenues for further improvements of general-purpose event generators. With the efficient evaluation of virtual contributions in hand, attention should now turn to the calculation of real-radiation configurations – that represent the bottleneck for both Sherpa and Mcfm. In the simplest cases with up to 5 partons, the real radiation and dipole counterterms could be evaluated using analytic rather than numerical matrix elements, by a suitable extension of the interface we have presented here. In addition, the form of the phase-space generation may be improved for Born-like phase-space integrals. Table 4 lists the number of phase-space points before cuts that are required to achieve the target accuracy. We find that Mcfm uses fewer than half of the points needed by Sherpa in the Born-like phase-space integrals, while Sherpa uses fewer points than Mcfm in the real-emission type integrals but at a much higher computational cost. This confirms that Sherpa's event generation is indeed impaired by the slow evaluation of real-emission type matrix elements, and by the factorial scaling of the diagram-

Table 3 CPU time ratios in an NLO multi-jet merged setup using Sherpa

Merged process	Sherpa+	Sherpa+
$n \leq 2$ @ NLO	OpenLoops2/Mcfm	MadLoop5/Mcfm
$n \leq 5$ @ LO		
$pp \to Z + nj$	$1.83^{+0.20}_{-0.12}$	$3.01^{+0.26}_{-0.18}$
$pp \to W^+ + nj$	$1.34^{+0.06}_{-0.07}$	$1.36^{+0.03}_{-0.03}$
$pp \to W^- + nj$	$1.38^{+0.06}_{-0.04}$	$1.38^{+0.07}_{-0.11}$

Table 4 Comparison of integration times using Sherpa and Mcfm

Process		Sherpa	Mcfm
MC accuracy		Time/#pts	Time/#pts
$pp \to Zj$	Born-like	76.8 m/11.3M	13.6 m /4.5M
0.1%	Real-like	38 h/33.1M	51.5 m/22.5M
$pp \to Zjj$	Born-like	96.0 h/22.4M	19.6 h/4.5M
0.3%	Real-like	830.4 h/58.7M	62.9 h/83.8M
$pp \to W^+j$	Born-like	40.5 m/12.8M	7.37 m/4.5M
0.1%	Real-like	16.9 h/38.3M	59.4 m/36.0M
$pp \to W^+jj$	Born-like	14.1 h/20.3M	9.32 h/7.2M
0.3%	Real-like	222.1 h/38.9M	54.4 h/119.8M
$pp \to W^-j$	Born-like	34.1 m/11.0M	7.46 m/4.5M
0.1%	Real-like	15.9 h/40.5M	47.2 m/28.1M
$pp \to W^-jj$	Born-like	12.8 h/20.0M	7.34 h/5.6M
0.3%	Real-like	281.1 h/52.0M	38.8 h/83.8M

Footnote 7 continued
which could both be relevant in practice. It has recently been shown that in good implementations of the reweighting and EW correction algorithm, the additional overhead will not be sizable [72].

based phase-space integration technique [75,76] used in its calculations.[8]

4 Numerical stability

As alluded to above, the numerical stability of one-loop amplitudes is of vital importance for both NLO and NNLO calculations, where the latter case necessitates a stable evaluation in single-unresolved phase-space regions. Here we wish to limit the discussion to this case and estimate the accuracy that can be expected from the one-loop amplitudes with an additional parton with respect to the Born multiplicity, i.e., those processes that correspond to the real-virtual contribution in an NNLO calculation. To this end, we generate trajectories into the singular limits according to dipole kinematics, rescaling the Catani-Seymour variables [77] of an initially hard configuration as[9]

$$y_{ijk} \to \begin{cases} \lambda \frac{s}{Q^2} & \text{final} - \text{final} \\ \frac{1}{1+\frac{Q^2(1-y_{ijk})}{\lambda s}} & \text{final} - \text{initial} \\ -z_i \frac{\lambda}{1-z_i \lambda} & \text{initial} - \text{initial} \end{cases}, \quad z_i \to z_i \tag{1}$$

in the collinear limit, and

$$\begin{aligned} y_{ijk} &\to \begin{cases} \frac{C(1-z_i)}{z_i} & \text{final} - \text{final} \\ \text{sign}(y_{ijk}) C \frac{1-z_i}{z_i} & \text{initial} - \text{initial} \end{cases}, \\ z_i &\to \begin{cases} 1 - 2\frac{\lambda}{1+C} & \text{final} - \text{final} \\ \frac{1}{1+2\frac{\lambda^2-\lambda\sqrt{1+\lambda^2}}{1+C}} & \text{initial} - \text{initial} \end{cases}, \\ C &= \frac{y_{ijk} z_i}{1-z_i} \end{aligned} \tag{2}$$

in the soft limit. To assess the stability of the loop-amplitude evaluation, we calculate the number of exact digits as

$$N_{\text{sd}} = -\log_{10}\left(\frac{|V - V'|}{V}\right), \tag{3}$$

where V and V' denote the finite parts of the one-loop amplitude evaluated on two phase-space points that are rotated with respect to each other.

We consider crossings of the processes listed in Tables 1 and 2 such that only final-state singularities are considered. We have validated that the numerical accuracy is generally worse when approaching final-state singularities, so that we deem this simplification sufficient. For each singular limit, we generate 10^4 hard phase-space points with $\sqrt{s} = 1$ TeV using SHERPA and, depending on the singular limit of interest, rescale the momenta according to Eq. 1 or Eq. 2 with $\lambda \in \{10^{-3}, 10^{-6}, 10^{-9}\}$. The results are collected in Figs. 3, 4 and 5, where each point corresponds to the average numerical accuracy according to Eq. 3 and the solid error bars indicate the 25% quantiles of the median. The lighter-shaded error bands span from the worst to the best result in each run. In cases where the two results agree perfectly within machine precision,[10] we set the number of stable digits to 16.

For all processes of interest, the numerical evaluation is sufficiently stable even in the deep infrared regions. Although not shown in the figures, we have checked that the stability of our interface is comparable to that of the other OLPs shown in Figs. 1 and 2 using appropriate settings.

5 Conclusions

We have presented a novel C++ interface to the well-known MCFM parton-level Monte Carlo generator, giving access to its extensive library of analytical one-loop amplitudes. The interface is generic and not tied to any specific Monte Carlo event generation tool. As a proof of its generality, we have implemented the interface in both, the SHERPA and PYTHIA event generators. The SHERPA interface will become public with version 3.0.0, and the PYTHIA interface is foreseen to become public in a future release of the 8.3 series[2]. It should be straightforward to adapt our code to the needs of other event generators.

We expect the interface to be valuable in two respects. First, for many of the processes considered here the speedup over other OLPs is substantial; accessing these matrix elements via this interface rather than an automated tool will therefore provide an immediate acceleration of event generation for many processes of high phenomenological interest. Second, the speed comparisons presented here highlight processes that are particularly computationally intensive for automated tools. Further improvements to the efficiency of these codes may be possible, with potential gains across a wider range of processes.

The structure of the interface allows for simple extensions. Further one-loop matrix elements in MCFM, implemented either currently or in the future, may become accessible in a straightforward manner. In the same spirit, the interface could also be extended to provide tree-level or two-loop matrix elements included in MCFM as the need arises. Further extensions to the interface, for instance to provide finer control

[8] We do not make use of SHERPA's recursive phase-space generator [78], because it is available for color-sampled matrix element evaluation only. Color sampling would further reduce the efficiency of the integration, because the processes at hand involve a relatively small number of QCD partons.

[9] We use the definition of y_{ijk} and $\tilde{z}_i$ according to the final-state emitter case with a final-state spectator, and treat initial-state momenta as outgoing. This allows to reconstruct the kinematics using a single, unified mapping procedure.

[10] This is only the case for $pp \to h_0 + j$.

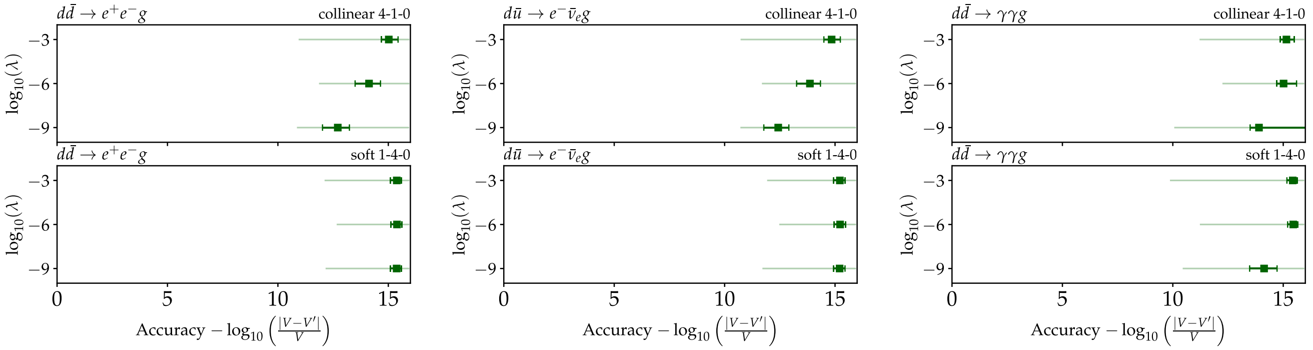


Fig. 3 Test of the numerical accuracy of standard-model three-parton one-loop amplitudes in the soft and collinear limits

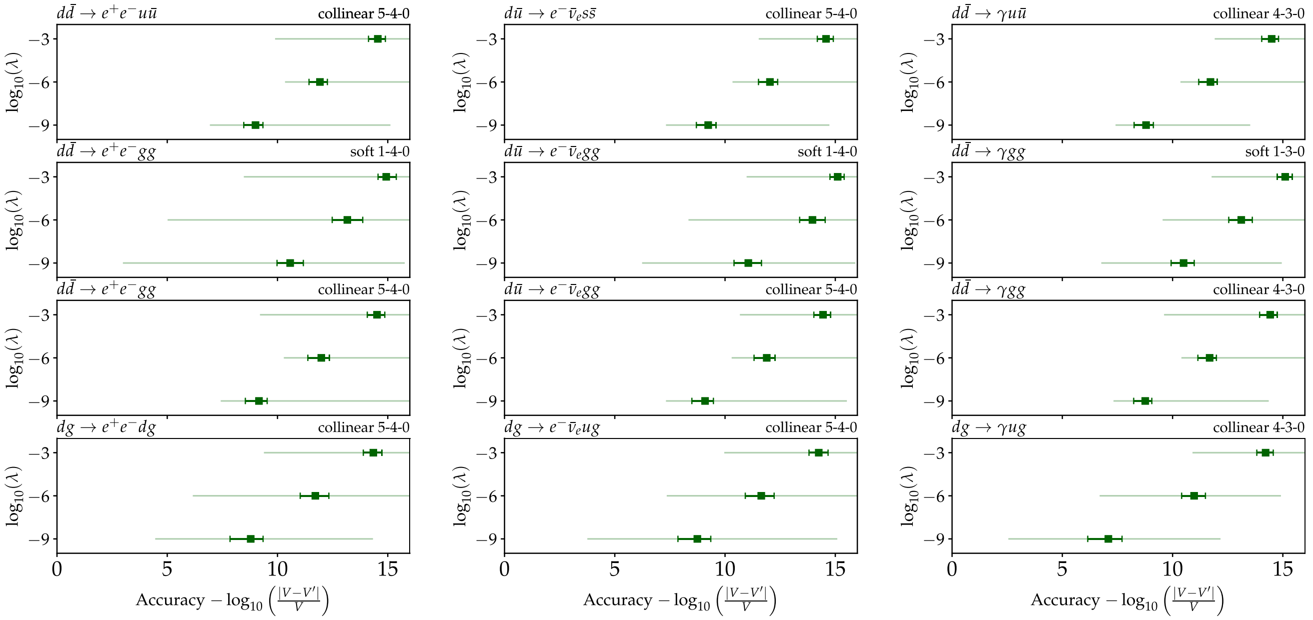
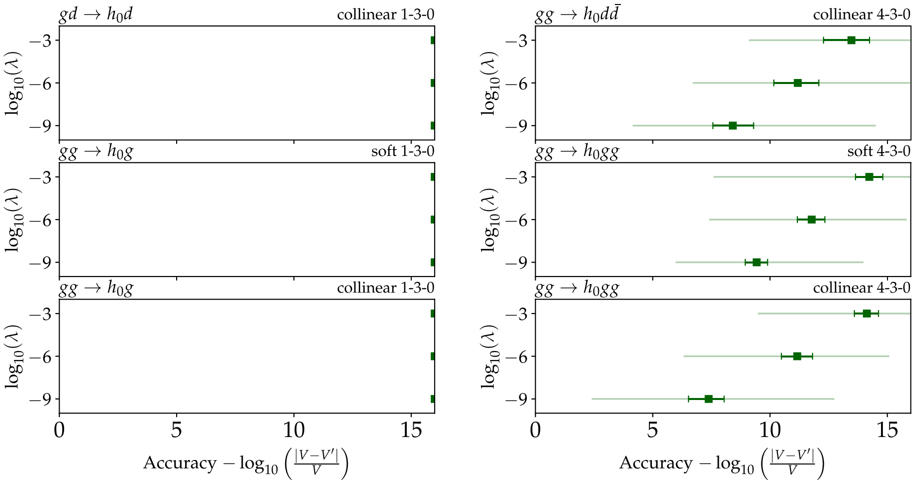


Fig. 4 Test of the numerical accuracy of standard-model four-parton one-loop amplitudes in the soft and collinear limits

Fig. 5 Test of the numerical accuracy of HEFT three- and four-parton one-loop amplitudes in the soft and collinear limits. Note that the accuracy is set to 16 digits in the $h_0 + j$ case, where the two results agree perfectly



over the one-loop matrix elements via the selection of helicities or color configurations, would also be possible.

Given that we have interfaced three popular automated OLPs within the generator-agnostic structure of the new MCFM interface, it is natural to envision the future development of a hybrid program that makes use of the fastest matrix element library for each process. Thinking further ahead, it may be worthwhile to reconsider a streamlined event generation framework, combining different (dedicated) parton-level and particle-level tools. This idea has been pursued with THEPEG [79], but so far rarely deployed. Apart from obvious efficiency improvements through the use of dedicated tools for different applications, such a framework enables previously unavailable methods for systematics studies. In view of both the faster integration in MCFM over SHERPA and the magnitude of uncertainties pertaining to theoretical modeling of collider observables, this is becoming an increasingly important avenue for future work.

We want to close by highlighting that only a relatively small number of analytical amplitudes has to be known in order to cover a wide range of physical processes. When judiciously assembled, many parts of the calculations can be recycled in a process-independent way, with only charge and coupling factors being process-specific. Compared to other efforts to increase the efficiency of event generators, swapping automated for analytical matrix elements is straightforward and simple. Analytical matrix element libraries provide a so-far little explored path towards higher-efficiency event generation for the (high-luminosity) LHC and future colliders.

Acknowledgements We are grateful to Frank Krauss and Marek Schönherr for many fruitful discussions. We would also like to thank Stefano Pozzorini, Jonas Lindert, and Jean-Nicolas Lang for help with OPENLOOPS, as well as Valentin Hirschi for help with MADLOOP. CTP thanks Peter Skands for support. CTP is supported by the Monash Graduate Scholarship, the Monash International Postgraduate Research Scholarship, and the J. L. William Scholarship. This research was supported by the Fermi National Accelerator Laboratory (Fermilab), a U.S. Department of Energy, Office of Science, HEP User Facility. Fermilab is managed by Fermi Research Alliance, LLC (FRA), acting under Contract no. DE-AC02-07CH11359. We further acknowledge support from the Monash eResearch Centre and eSolutions—Research Support Services through the MonARCH HPC Cluster. This work was also supported in part by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Grant agreement no. 722104—MCnetITN3.

Data Availability Statement This manuscript has no associated data or the data will not be deposited. [Authors' comment: All programs used to reproduce the data shown in this publication are publicly available.]

Funded by SCOAP^3.

Appendix A: Parameters and cuts for timing comparisons

In order to perform the timing comparisons shown in Fig. 2 and Table 4. we employ the following scale choices and phase-space cuts:

- $\Delta R_{\ell,\gamma} > 0.4$
- $\Delta R_{\gamma,\gamma} > 0.4$
- $p_{\mathrm{T},\gamma} > 30$ GeV
- $p_{\mathrm{T},j} > 30$ GeV
- 66 GeV $< m_{\ell\bar{\ell}} <$ 116 GeV

We reconstruct jets using the anti-k_T algorithm [80] in the implementation of FastJet [81] with an R parameter of 0.4. For the di-jet process we require $p_{\mathrm{T},j}$ >80 GeV. Photons are isolated from QCD activity based on Ref. [82] with $\delta_0 = 0.4$, $n = 2$ and $\epsilon_\gamma = 2.5\%$.

Appendix B: Structure of the interface

The MCFM C++ interface is constructed as a C++ class

```
CXX_Interface mcfm;
```

included in the header:

```
#include "MCFM/CXX_Interface.h"
```

It must be initialized on a `std::map` of `std::strings`, containing all (standard-model) parameters:

```
bool CXX_Interface::Initialize
(std::map<std::string,std::
string>& parameters);
```

Prior to use, each process has to be initialized in the interface:

```
int CXX_Interface::
InitializeProcess(const
Process_Info &pi);
```

which takes a `Process_Info` object as input, which in turn contains the defining parameters of a given process, i.e., the PDG IDs, number of incoming particles, and QCD and EW coupling orders:

```
Process_Info(const std::vector
<int> &ids, const int nin,
const int oqcd, const int oew);
```

Phase space points are defined using the `FourVec` struct, which represents four-vectors in the ordering (E, p_x, p_y, p_z).

```
FourVec(double e, double px,
double py, double pz);
```

Given a list of four-vectors in this format, one-loop matrix elements can be calculated either using the process ID returned by the `InitializeProcess` method

```
void CXX_Interface::Calc(int
procID, const std::vector
<FourVec> &p, int oqcd);
```

or using a `Process_Info` struct:

```
void CXX_Interface::Calc(const
Process_Info &pi, const
std::vector<FourVec> &p,int oqcd);
```

In the same way, the result of this calculation can be accessed either via the process ID

```
const std::vector<double>
&CXX_Interface::GetResult
(int procID);
```

or using the `Process_Info` struct:

```
const std::vector<double>
&CXX_Interface::GetResult(const
Process_Info &pi)
```

The result is returned as a list of Laurent series coefficients in the format $(\mathcal{O}\left(\varepsilon^0\right), \mathcal{O}\left(\varepsilon^{-1}\right), \mathcal{O}\left(\varepsilon^{-2}\right)$, Born). However, by default only the $\mathcal{O}\left(\varepsilon^0\right)$ coefficient, i.e., the finite part, is returned. The calculation of the pole terms and the Born can be enabled by setting the following switch to 1:

```
void CXX_Interface::SetPoleCheck
(int check);
```

An example code showing the basic usage of the interface as well as a function filling the complete list of parameters with default values is publicly available[1].

References

1. C.F. Berger, Z. Bern, L.J. Dixon, F. Febres Cordero, D. Forde, H. Ita, D.A. Kosower, D. Maître, Phys. Rev. D **78**, 036003 (2008). https://doi.org/10.1103/PhysRevD.78.036003
2. C.F. Berger, Z. Bern, L.J. Dixon, F. Febres Cordero, D. Forde, T. Gleisberg, H. Ita, D.A. Kosower, D. Maître, Phys. Rev. D **80**, 074036 (2009). https://doi.org/10.1103/PhysRevD.80.074036
3. C.F. Berger, Z. Bern, L.J. Dixon, F. Febres Cordero, D. Forde, T. Gleisberg, H. Ita, D.A. Kosower, D. Maître, Phys. Rev. D **82**, 074002 (2010). https://doi.org/10.1103/PhysRevD.82.074002
4. C.F. Berger, Z. Bern, L.J. Dixon, F. Febres Cordero, D. Forde, T. Gleisberg, H. Ita, D.A. Kosower, D. Maître, Phys. Rev. Lett. **106**, 092001 (2011). https://doi.org/10.1103/PhysRevLett.106.092001
5. H. Ita, Z. Bern, L.J. Dixon, F. Febres Cordero, D.A. Kosower, D. Maître, Phys. Rev. D **85**, 031501 (2012). https://doi.org/10.1103/PhysRevD.85.031501
6. Z. Bern, L.J. Dixon, F. Febres Cordero, S. Höche, H. Ita, D.A. Kosower, D. Maître, K.J. Ozeren, Phys. Rev. D **88**(1), 014025 (2013). https://doi.org/10.1103/PhysRevD.88.014025
7. 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
8. 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
9. F. Cascioli, P. Maierhöfer, S. Pozzorini, Phys. Rev. Lett. **108**, 111601 (2012). https://doi.org/10.1103/PhysRevLett.108.111601
10. F. Buccioni, S. Pozzorini, M. Zoller, Eur. Phys. J. C **78**(1), 70 (2018). https://doi.org/10.1140/epjc/s10052-018-5562-1
11. 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
12. S. Badger, B. Biedermann, P. Uwer, Comput. Phys. Commun. **182**, 1674 (2011). https://doi.org/10.1016/j.cpc.2011.04.008
13. S. Badger, B. Biedermann, P. Uwer, V. Yundin, Comput. Phys. Commun. **184**, 1981 (2013). https://doi.org/10.1016/j.cpc.2013.03.018
14. G. Cullen, N. Greiner, G. Heinrich, G. Luisoni, P. Mastrolia, G. Ossola, T. Reiter, F. Tramontano, Eur. Phys. J. C **72**, 1889 (2012). https://doi.org/10.1140/epjc/s10052-012-1889-1
15. G. Cullen et al., Eur. Phys. J. C **74**(8), 3001 (2014). https://doi.org/10.1140/epjc/s10052-014-3001-5
16. S. Actis, A. Denner, L. Hofer, A. Scharf, S. Uccirati, JHEP **04**, 037 (2013). https://doi.org/10.1007/JHEP04(2013)037
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, JHEP **07**, 087 (2017). https://doi.org/10.1007/JHEP07(2017)087
19. A. Denner, J.N. Lang, S. Uccirati, Comput. Phys. Commun. **224**, 346 (2018). https://doi.org/10.1016/j.cpc.2017.11.013
20. C. Degrande, Comput. Phys. Commun. **197**, 239 (2015). https://doi.org/10.1016/j.cpc.2015.08.015
21. C. Degrande, G. Durieux, F. Maltoni, K. Mimasu, E. Vryonidou, C. Zhang, Phys. Rev. D **103**(9), 096024 (2021)
22. J. Albrecht et al., Comput. Softw. Big Sci. **3**(1), 7 (2019). https://doi.org/10.1007/s41781-018-0018-8
23. S. Alioli, et al., (2019). arXiv:1902.01674
24. S. Amoroso et al., Comput. Softw. Big Sci. **5**(1), 12 (2021). https://doi.org/10.1007/s41781-021-00055-1
25. T. Aarrestad, et al., in *2022 Snowmass Summer Study*, ed. by P. Canal, et al. (2020). https://doi.org/10.5281/zenodo.4009114
26. R. Boughezal, J.M. Campbell, R.K. Ellis, C. Focke, W. Giele, X. Liu, F. Petriello, C. Williams, Eur. Phys. J. C **77**(1), 7 (2017). https://doi.org/10.1140/epjc/s10052-016-4558-y
27. J. Campbell, T. Neumann, JHEP **12**, 034 (2019). https://doi.org/10.1007/JHEP12(2019)034
28. J.M. Campbell, R.K. Ellis, Phys. Rev. D **60**, 113006 (1999). https://doi.org/10.1103/PhysRevD.60.113006
29. J.M. Campbell, R.K. Ellis, C. Williams, JHEP **07**, 018 (2011). https://doi.org/10.1007/JHEP07(2011)018
30. J.M. Campbell, R.K. Ellis, W.T. Giele, Eur. Phys. J. C **75**(6), 246 (2015). https://doi.org/10.1140/epjc/s10052-015-3461-2
31. T. Binoth et al., Comput. Phys. Commun. **181**, 1612 (2010). https://doi.org/10.1016/j.cpc.2010.05.016
32. S. Alioli et al., Comput. Phys. Commun. **185**, 560 (2014). https://doi.org/10.1016/j.cpc.2013.10.020
33. E. Bothmann et al., SciPost Phys. **7**(3), 034 (2019). https://doi.org/10.21468/SciPostPhys.7.3.034

34. T. Sjöstrand, S. Ask, J.R. Christiansen, R. Corke, N. Desai, P. Ilten, S. Mrenna, S. Prestel, C.O. Rasmussen, P.Z. Skands, Comput. Phys. Commun. **191**, 159 (2015). https://doi.org/10.1016/j.cpc.2015.01.024
35. L. Hartgring, E. Laenen, P. Skands, JHEP **10**, 127 (2013). https://doi.org/10.1007/JHEP10(2013)127
36. N. Baberuxki, C.T. Preuss, D. Reichelt, S. Schumann, JHEP **04**, 112 (2020). https://doi.org/10.1007/JHEP04(2020)112
37. H. Brooks, C.T. Preuss, P. Skands, JHEP **07**, 032 (2020). https://doi.org/10.1007/JHEP07(2020)032
38. J.M. Campbell, R.K. Ellis, C. Williams, Phys. Rev. Lett. **118**(22), 259901 (2017). https://doi.org/10.1103/PhysRevLett.118.222001 [Erratum: Phys. Rev. Lett. 124, 259901 (2020)]
39. J. Campbell, K. Ellis, W. Giele, T. Neumann, C. Williams, https://www.mcfm.fnal.gov/
40. A. Denner, S. Dittmaier, M. Roth, D. Wackeroth, Nucl. Phys. B **560**, 33 (1999). https://doi.org/10.1016/S0550-3213(99)00437-X
41. A. Denner, S. Dittmaier, M. Roth, L.H. Wieders, Nucl. Phys. B **724**, 247 (2005). https://doi.org/10.1016/j.nuclphysb.2011.09.001 [Erratum: Nucl. Phys. B 854, 504-507 (2012)]
42. Z. Bern, L.J. Dixon, D.A. Kosower, Nucl. Phys. B **513**, 3 (1998). https://doi.org/10.1016/S0550-3213(97)00703-7
43. J.M. Campbell, R.K. Ellis, JHEP **01**, 020 (2017). https://doi.org/10.1007/JHEP01(2017)020
44. R.K. Ellis, I. Hinchliffe, M. Soldate, J.J. van der Bij, Nucl. Phys. B **297**, 221 (1988). https://doi.org/10.1016/0550-3213(88)90019-3
45. R.K. Ellis, S. Seth, JHEP **11**, 006 (2018). https://doi.org/10.1007/JHEP11(2018)006
46. L. Budge, J.M. Campbell, G. De Laurentis, R.K. Ellis, S. Seth, JHEP **05**, 079 (2020). https://doi.org/10.1007/JHEP05(2020)079
47. E.W.N. Glover, J.J. van der Bij, Nucl. Phys. B **309**, 282 (1988). https://doi.org/10.1016/0550-3213(88)90083-1
48. J.M. Campbell, R.K. Ellis, C. Williams, JHEP **06**, 179 (2016). https://doi.org/10.1007/JHEP06(2016)179
49. R.K. Ellis, D.A. Ross, A.E. Terrano, Nucl. Phys. B **178**, 421 (1981). https://doi.org/10.1016/0550-3213(81)90165-6
50. P. Aurenche, R. Baier, A. Douiri, M. Fontannaz, D. Schiff, Nucl. Phys. B **286**, 553 (1987). https://doi.org/10.1016/0550-3213(87)90453-6
51. J.M. Campbell, R.K. Ellis, Y. Li, C. Williams, JHEP **07**, 148 (2016). https://doi.org/10.1007/JHEP07(2016)148
52. J.M. Campbell, C. Williams, Phys. Rev. D **89**(11), 113001 (2014). https://doi.org/10.1103/PhysRevD.89.113001
53. T. Dennen, C. Williams, Phys. Rev. D **91**(5), 054012 (2015). https://doi.org/10.1103/PhysRevD.91.054012
54. L.J. Dixon, Z. Kunszt, A. Signer, Nucl. Phys. B **531**, 3 (1998). https://doi.org/10.1016/S0550-3213(98)00421-0
55. P. Nason, S. Dawson, R.K. Ellis, Nucl. Phys. B **327**, 49 (1989). https://doi.org/10.1016/0550-3213(89)90286-1 (**[Erratum: Nucl. Phys. B 335, 260-260 (1990)]**)
56. R.K. Ellis, J.C. Sexton, Nucl. Phys. B **269**, 445 (1986). https://doi.org/10.1016/0550-3213(86)90232-4
57. C.R. Schmidt, Phys. Lett. B **413**, 391 (1997). https://doi.org/10.1016/S0370-2693(97)01102-7
58. L.J. Dixon, E.W.N. Glover, V.V. Khoze, JHEP **12**, 015 (2004). https://doi.org/10.1088/1126-6708/2004/12/015
59. R.K. Ellis, W.T. Giele, G. Zanderighi, Phys. Rev. D **72**(2005). https://doi.org/10.1103/PhysRevD.72.054018 [Erratum: Phys. Rev. D 74, 079902 (2006)]
60. S.D. Badger, E.W.N. Glover, Nucl. Phys. B Proc. Suppl. **160**, 71 (2006). https://doi.org/10.1016/j.nuclphysbps.2006.09.030
61. S.D. Badger, E.W.N. Glover, K. Risager, JHEP **07**, 066 (2007). https://doi.org/10.1088/1126-6708/2007/07/066
62. E.W.N. Glover, P. Mastrolia, C. Williams, JHEP **08**, 017 (2008). https://doi.org/10.1088/1126-6708/2008/08/017
63. S. Badger, E.W. Nigel Glover, P. Mastrolia, C. Williams, JHEP **01**, 036 (2010). https://doi.org/10.1007/JHEP01(2010)036
64. L.J. Dixon, Y. Sofianatos, JHEP **08**, 058 (2009). https://doi.org/10.1088/1126-6708/2009/08/058
65. S. Badger, J.M. Campbell, R.K. Ellis, C. Williams, JHEP **12**, 035 (2009). https://doi.org/10.1088/1126-6708/2009/12/035
66. R. Kleiss, W.J. Stirling, S.D. Ellis, Comput. Phys. Commun. **40**, 359 (1986). https://doi.org/10.1016/0010-4655(86)90119-0
67. T. Gleisberg, S. Höche, F. Krauss, M. Schönherr, S. Schumann, F. Siegert, J. Winter, JHEP **02**, 007 (2009). https://doi.org/10.1088/1126-6708/2009/02/007
68. B. Biedermann, S. Bräuer, A. Denner, M. Pellen, S. Schumann, J.M. Thompson, Eur. Phys. J. C **77**, 492 (2017). https://doi.org/10.1140/epjc/s10052-017-5054-8
69. S. Höche, F. Krauss, M. Schönherr, F. Siegert, JHEP **04**, 027 (2013). https://doi.org/10.1007/JHEP04(2013)027
70. S. Höche, S. Prestel, H. Schulz, Phys. Rev. D **100**(1), 014024 (2019). https://doi.org/10.1103/PhysRevD.100.014024
71. E. Bothmann, M. Schönherr, S. Schumann, Eur. Phys. J. C **76**(11), 590 (2016). https://doi.org/10.1140/epjc/s10052-016-4430-0
72. E. Bothmann, S. Höche, M. Schönherr, (2021) (**in prep**)
73. K. Danziger, *Master's thesis. Number CERN-THESIS-2020-024* (2020). https://www.cds.cern.ch/record/2715727
74. S. Höche, F. Krauss, M. Schönherr, F. Siegert, JHEP **09**, 049 (2012). https://doi.org/10.1007/JHEP09(2012)049
75. E. Byckling, K. Kajantie, Nucl. Phys. B **9**, 568 (1969)
76. E. Byckling, K. Kajantie, Phys. Rev. **187**, 2008 (1969)
77. S. Catani, M.H. Seymour, Nucl. Phys. B **485**, 291-419 (1997). (Erratum: Nucl. Phys. B **510**, 503-504 (1998))
78. T. Gleisberg, S. Höche, JHEP **12**, 039 (2008). https://doi.org/10.1088/1126-6708/2008/12/039
79. L. Lönnblad, Nucl. Instrum. Methods A **559**, 246 (2006). https://doi.org/10.1016/j.nima.2005.11.143
80. M. Cacciari, G.P. Salam, G. Soyez, JHEP **04**, 063 (2008). https://doi.org/10.1088/1126-6708/2008/04/063
81. M. Cacciari, G.P. Salam, G. Soyez, Eur. Phys. J. C **72**, 1896 (2012). https://doi.org/10.1140/epjc/s10052-012-1896-2
82. S. Frixione, Phys. Lett. B **429**, 369 (1998). https://doi.org/10.1016/S0370-2693(98)00454-7