Eur. Phys. J. C (2023) 83:1051
https://doi.org/10.1140/epjc/s10052-023-12243-x

THE EUROPEAN PHYSICAL JOURNAL C

Letter

A new way of reducing negative weights in MC@NLO

Rikkert Frederix[1,a], **Paolo Torrielli**[2,b]

[1] Division of Particle and Nuclear Physics, Department of Physics, Lund University, Box 118, 221 00 Lund, Sweden
[2] Dipartimento di Fisica, Università di Torino and INFN, Sezione di Torino, 10125 Turin, Italy

Received: 20 October 2023 / Accepted: 6 November 2023 / Published online: 17 November 2023

Abstract We introduce a new technique, that we dub *Born spreading*, aimed at reducing the number of negative-weight $\mathbb{S}$ events in the MC@NLO matching of NLO calculations with parton-shower simulations. We show that such a technique, based on a re-distribution of Born matrix elements in the radiative phase space, achieves a sizeable reduction of negative-weight events at little computational cost. The method does not induce any biases in physical distributions.

1 Introduction

Modern high-energy particle phenomenology requires Monte Carlo simulations featuring steadily increasing computational costs, both in terms of running times and of computing resources [1–3]. The bulk of the simulations carried out by the experimental collaborations are ones where next-to-leading order (NLO) predictions in QCD are matched with parton-shower event generators. Since NLO cross sections are in general not positive-definite, it is not uncommon that some of the events generated by means of such simulations are associated to a negative weight.

As far as resources are concerned, the presence of negative weights may represent a serious bottleneck. The computational cost of a simulation, for a given target accuracy (Monte Carlo error), scales as $1/(1-2f)^2$, with f being the total fraction of negative-weight events in the sample [4]. Such an issue may be severe for the MC@NLO matching scheme [5], where particularly complex LHC processes may produce up to $O(30-40\%)$ negative-weight events, whereas alternative matching schemes [6–9] are less affected.

Various techniques have been proposed aiming at the reduction of negative weights, either modifying the event generation [4,10], or as a post-processing step on existing event samples [11–14]. In the context of MC@NLO, and in particular in its implementation within the MadGraph5_aMC@NLO code [15] (abbreviated as MG5_aMC henceforth), negative weights have two distinct sources. Negative $\mathbb{H}$ events have a physical origin in the way the MC@NLO matching is defined, i.e. they depend on whether the radiation probability, as predicted by the parton shower one is interfacing to, is bigger than the NLO real matrix element for the considered process. As such, the fraction of these negative-weight events can be reduced only with a modification of the matching prescription, e.g. the one that has been proposed in [4] and dubbed MC@NLO-Δ matching.

On the other hand, negative $\mathbb{S}$ events are largely caused by the fact that, although having Born kinematics, the short-distance cross section associated with them has support in the radiative phase space. Such a cross section is not positive-definite locally in the radiative phase space, resulting in negative-weight events after the unweighting procedure. This is usually tackled by means of the folding procedure [4,16–18], where at a given Born phase-space point one averages over many triplets (*folds*) of variables associated to the radiated particle, thereby efficiently compensating local $\mathbb{S}$-integrand negativities before the unweighting step is taken. Although this is a systematic working solution, it typically entails a significant runtime increase before a reduction of negative weights is obtained, roughly proportional to the number of folds, so that the final effective gain in terms of computational costs is not always a priori obvious.

In this paper we propose a novel alternative strategy, that we dub *Born spreading*, to reduce negative $\mathbb{S}$ events in MG5_aMC at very little CPU cost with respect to the standard MC@NLO implementation. We present the ideas behind the method in Sect. 2, we detail its implementation in Sect. 3 and its performances with respect to folding in Sect. 4. In Sect. 5 we discuss a number of possible extensions of the method, and draw our conclusions in Sect. 6.

[a] e-mail: rikkert.frederix@hep.lu.se
[b] e-mail: paolo.torrielli@unito.it (corresponding author)

2 Born spreading

The $\mathbb{S}$-event generating functional in MC@NLO can be schematically written as

$$\mathcal{F}^{(\mathbb{S})} = \int \left[\frac{B(\Phi_B)}{\int d\Phi_r} + \frac{V(\Phi_B)}{\int d\Phi_r} + K_{\rm MC}(\Phi_B, \Phi_r) \right] d\Phi_r \times \mathcal{F}_{\rm MC}^{(B)}, \quad (1)$$

where $d\Phi_B$ and $d\Phi_r$ are the Born and radiative phase-space measures, respectively (the former including Bjorken fractions in case of hadronic collisions, as well as flux factors). $B(\Phi_B)$ is the Born matrix-element squared, whereas $K_{\rm MC}(\Phi_B, \Phi_r)$ is the so-called Monte Carlo counterterm, i.e. the $O(\alpha_s)$ emission probability as approximated by the shower (a local subtraction of infrared and collinear singularities from this term, e.g. by means of the FKS procedure [19], is understood). The term labelled as $V(\Phi_B)$ is the finite part of the renormalised virtual contribution, including remnants of the integrated (FKS) subtraction terms, as well as mass factorisation finite contributions in case of hadronic collisions. Finally, $\mathcal{F}_{\rm MC}^{(B)}$ is the standard generating functional of the parton shower, where the label (B) indicates that the partonic multiplicity the shower evolution starts from is the Born-level one. Equation (1) makes it clear that, although $\mathbb{S}$-event kinematics is Born-like, the support of the integrand in square brackets is in the full Born+radiative phase space.

In Eq. (1), the origin of negative $\mathbb{S}$-event contributions is twofold.[1] The first one is due to the FKS subtraction in $K_{\rm MC}(\Phi_B, \Phi_r)$: even though the subtraction cancels when integrated over Φ_r, locally it might over-compensate, resulting in a negative contribution. The second source is that for certain Φ_B configurations, the virtual contribution, $V(\Phi_B)$, can be negative. Both these sources can be reduced by the following procedure.

Since the dependence of the Born contribution $B(\Phi_B)$ upon the radiative variables Φ_r is immaterial, the expression in Eq. (1) can be equivalently rewritten as

$$\mathcal{F}^{(\mathbb{S})} = \int \left[\frac{B(\Phi_B)\, F(\Phi_r)}{\int F(\Phi_r)\, d\Phi_r} + \frac{V(\Phi_B)}{\int d\Phi_r} + K_{\rm MC}(\Phi_B, \Phi_r) \right] d\Phi_r \times \mathcal{F}_{\rm MC}^{(B)}, \quad (2)$$

where an arbitrary integrable *spreading function* $F(\Phi_r)$ has been introduced as a Born multiplier. It is clear that the functional form of $F(\Phi_r)$ does not affect physical distributions stemming from the $\mathbb{S}$-event generating functional, as those just depend on Born-level kinematics, and the spreading function is normalised so as to preserve the Born weight locally in Φ_B. Therefore, one can leverage the ample freedom in the definition of $F(\Phi_r)$, and find one that minimises the presence of negative weights in the $\mathbb{S}$ events. The idea is that of defining $F(\Phi_r)$ so that the Born contribution, now non-uniformly distributed over Φ_r, is accumulated more where the rest of the $\mathbb{S}$-event integrand is more negative, as to rescue as much as possible of the negatively-contributing phase space.

If we define

$$W(\Phi_B, \Phi_r) = \left[\frac{V(\Phi_B)}{\int d\Phi_r} + K_{\rm MC}(\Phi_B, \Phi_r) \right] \frac{1}{B(\Phi_B)}, \quad (3)$$

namely the $\mathbb{S}$-event contribution of non-Born origin with the Born divided out, a spreading function that satisfies the above requirements is

$$F(\Phi_r) = \max \left[0, -\int W(\Phi_B, \Phi_r)\, d\Phi_B \right]. \quad (4)$$

We use this as our baseline choice to study the Born-spreading performances throughout the paper.

We conclude this Section by mentioning a subtlety in the Born-spreading procedure that specifically arises in the context of the MC@NLO matching. In MC@NLO one needs to assign to each generated event a shower starting scale, representing the phase-space boundary for subsequent shower activity. Such a scale can be chosen arbitrarily to a large extent, with the sole constraint that the Monte Carlo counterterms used in the definition of the short-distance MC@NLO cross section must feature the very same boundary, in order not to introduce double counting at $O(\alpha_s)$. As the contributions to $\mathcal{F}^{(\mathbb{S})}$ in Eq. (1) have support in the radiative phase space, the starting scale assigned to $\mathbb{S}$ events may in principle depend on Φ_r. Were this the case, any spreading function $F(\Phi_r) \neq 1$ would cause a distortion in the starting-scale distribution and, in turn, a bias in the prediction for physical observables through showering effects. We stress that this bias would not represent a higher-order effect, namely it would spoil the formal $O(\alpha_s)$ accuracy of the prediction. Two main strategies can be adopted to correct for this feature. On the one hand, one can encode $F(\Phi_r)$ in the definition of the Monte Carlo counterterms: however, on top of requiring a thorough validation of the latter, this solution would also modify the functional form of $W(\Phi_B, \Phi_r)$ and in turn of the spreading function itself, resulting in a circular approach. A simple alternative solution is to assign to $\mathbb{S}$ events a starting scale that just depends on Φ_B: this option is not only formally allowed, but also more natural from the physical viewpoint, as it potentially reduces spurious correlations between radiative and non-radiative configurations. We have opted for this solution in our implementation of Born spreading in the context of MG5_aMC, which has entailed minimal modifications to the default scale assignment for events in the Monte Carlo dead zones.

[1] There is, potentially, also a third small contribution to the negative weights coming from non-positive parton density functions (PDFs).

3 Implementation

Our implementation of the Born-spreading idea in MG5_aMC builds upon the default integration and event-generation strategies of the latter, summarised in the following. A preliminary stage (dubbed *step-0*) is executed solely for the integration routines to iteratively adapt the multi-dimensional grids to the shape of the integrands at hand, while the integration results are discarded. Based on such grids, in a subsequent *step-1* the proper integration is performed, together with the evaluation of the upper bounds necessary for event unweighting. Finally, *step-2* deals with the generation of the unweighted (up to a sign) event sample.

If Born spreading is active, step-0 is initially run as usual, with the $\mathbb{S}$-integrand defined as in Eq. (1); after the integration grids are set up, the code turns to sample the spreading function $F(\Phi_r)$ in the radiative phase space. The latter is parametrised in terms of the dimensionless FKS variables ξ, y, ϕ [19], related to the energy, polar and azimuthal angle of the radiated parton. The azimuthal modulation of the integrands is discarded at all stages of the Born-spreading procedure, which is expected to have little numerical effect, as ϕ is not associated to any physical phase-space singularities. Then a two-dimensional $N_\xi \times N_y$ grid (by default 40×40) is defined, and the code throws N_{spread} random points (by default 10^6) to sample the spreading function.[2] In each of the $N_\xi \times N_y$ bins, the piecewise value of the spreading function F_{ij} (with $i\,(j) = 1, ..., N_{\xi\,(y)}$) is obtained upon averaging over all of the sampled underlying Born kinematics. At this point, the $\mathbb{S}$-integrand is redefined as in Eq. (2) (with the formal replacement $F(\Phi_r) \to F_{ij}$), and another instance of step-0 is run, in order to let the integration grids re-adapt to the newly-defined integrand. Finally, once the new grids have been obtained, the subsequent integration and event-generation steps proceeds as in the default code.

From the above discussion, one can anticipate that in presence of Born spreading the time spent by the code in the step-0 phase will be significantly larger than in the default operational mode: this is mainly driven by the sampling of the spreading function, and to a lesser extent by the subsequent grid setup. However, commonly step-0 is (by far) the fastest step in the whole integration and event-generation procedure, hence a time penalty at this stage is not particularly worrisome; moreover, such a penalty just depends on the considered process, and does not scale with the number of events produced, nor with the accuracy required for the integration result, thereby representing a mere constant time offset. Conversely, the subsequent step-1 and step-2 are expected to perform by construction as well as in the default code: this represents a considerable potential advantage in terms of runtimes with respect to folding, which was mentioned above to scale linearly with the number of employed folds.

[2] The virtual component in the spreading function is evaluated only approximately (using $\tilde{V}_k$ instead of V, as defined in Ref. [15]), i.e. no loop matrix-element routine is actually called in the spreading procedure.

4 Results

In this Section we present results for the Born-spreading procedure applied to several LHC processes, covering a variety of physical situations including colour-singlet production ($pp \to e^+e^-$, and $pp \to H$), processes with jets at Born level ($pp \to W^+j$), and processes with massive coloured final states ($pp \to t\bar{t}$, $pp \to W^+t\bar{t}$, and $pp \to Hb\bar{b}$). Our simulations refer to the LHC operated at 13 TeV, with the following Standard Model parameters: $m_t = 173$ GeV, $m_W = 80.385$ GeV, $\Gamma_W = 2.047600$ GeV, $m_H = 125$ GeV, $m_Z = 91.188$ GeV, $\Gamma_Z = 2.441404$ GeV, $G_F = 1.16639 \cdot 10^{-5}$ GeV^2, $\alpha = 1/132.507$. We employ the central replica of the NNPDF2.3 PDF set [20], which sets the strong coupling value as $\alpha_s(m_Z) = 0.118$. The central values of renormalisation and factorisation scales are chosen to be $\mu_{R,F} = \frac{1}{2}\sum_i (m_i^2 + p_{T,i}^2)^{1/2}$, with the sum running over all final-state Born-level particles. All processes are simulated in the five-flavour scheme, with the exception of $pp \to Hb\bar{b}$, where a bottom mass $m_b = 4.7$ GeV is assumed. For neutral Drell-Yan $pp \to e^+e^-$ we require a threshold for the lepton-pair invariant mass, $m_{e^+e^-} \geq 30$ GeV, while for $pp \to W^+j$ we impose a $p_T \geq 10$ GeV generation cut on the hardest jet in the event, where jets are reconstructed with FASTJET [21] using an $R = 0.7$ k_T algorithm.

In Table 1, Born spreading is compared against two folding setups, with $n_\xi \times n_y \times n_\phi = 2 \times 2 \times 1$ and $4 \times 4 \times 1$ folds, respectively. Results of the default MG5_aMC baseline (formally corresponding to a $1 \times 1 \times 1$ folding), are also displayed for reference. The comparison focuses on the reduction of negative-weight $\mathbb{S}$ events with respect to the $1 \times 1 \times 1$ baseline, as well as on time performances relevant to the generation of 1 M events per process on a desktop machine. Runtimes are broken up in their different integration and event-generation stages. We have extensively verified that, upon addressing the subtlety presented at the end of Sect. 2, physical distributions obtained with Born spreading are statistically identical to those obtained with the default MG5_aMC code and with folding, hence we refrain from showing the corresponding plots.

As far as the first four listed processes are concerned, Born spreading proves an excellent compromise between speed and negative-weight reduction. The latter is substantial in comparison with the default code, with residual negative fraction at the level of $O(2\%)$. Correspondingly, as anticipated, Born spreading induces a significant runtime increase only

Table 1 Runtimes and fraction of negative-weight $\mathbb{S}$ events for various LHC processes with default MG5_aMC code, two folding setups, and Born spreading

	Step-0 (s) (grid setup)	Step-1 (s) (integration)	Step-2 (s) (generation)	Negative $\mathbb{S}$ weights (%)
$pp \to e^+e^-$				
Default	1	14	147	7.1
$2 \times 2 \times 1$ folding	1	33	258	2.1
$4 \times 4 \times 1$ folding	1	114	781	1.8
Born spreading	113	30	189	2.0
$pp \to H$				
Default	1	121	187	10.6
$2 \times 2 \times 1$ folding	1	115	399	2.7
$4 \times 4 \times 1$ folding	1	228	1190	0.6
Born spreading	82	122	203	1.1
$pp \to t\bar{t}$				
Default	2	132	455	8.6
$2 \times 2 \times 1$ folding	2	262	1005	2.2
$4 \times 4 \times 1$ folding	2	1092	3189	1.2
Born spreading	199	137	448	2.1
$pp \to W^+t\bar{t}$				
Default	5	346	1511	4.2
$2 \times 2 \times 1$ folding	2	661	2938	2.2
$4 \times 4 \times 1$ folding	2	2605	10020	1.7
Born spreading	202	741	2138	2.6
$pp \to W^+j$				
Default	10	604	2013	24.2
$2 \times 2 \times 1$ folding	10	1265	5160	13.2
$4 \times 4 \times 1$ folding	7	2803	16020	9.0
Born spreading	355	645	2226	18.8
$pp \to Hb\bar{b}$				
Default	77	1311	19440	27.3
$2 \times 2 \times 1$ folding	39	4320	16380	22.4
$4 \times 4 \times 1$ folding	48	17220	34260	20.9
Born spreading	578	1263	20760	24.7

in the grid-setup phase, which in any case is reasonably fast and, importantly, does not scale with the number of generated events; time performances remain comparable with the baseline at all subsequent stages, which represents a strong benefit of the Born-spreading procedure. For these four processes, Born spreading is better than or comparable with $2 \times 2 \times 1$ folding in terms of negative-weight reduction, and is significantly faster. $4 \times 4 \times 1$ folding is systematically more efficient than Born spreading in reducing negative $\mathbb{S}$ weights, however it typically takes a factor 3–5 longer. We argue that once the residual fraction of negative weights f is at the level of few percent, reducing it further is barely justified, since in that case the computational-cost reduction scales linearly with f, while inducing a time penalty that scales with the number of events. In this respect, Born spreading represents an interesting alternative to folding.

As for processes with light QCD final states, $pp \to W^+j$ and $pp \to Hb\bar{b}$, in general we note a sizeable fraction of negative $\mathbb{S}$ weights. Born spreading is able to systematically reduce their impact with respect to the default MG5_aMC, maintaining comparable runtimes. In this case already $2 \times 2 \times 1$ folding outperforms Born spreading as for negative-weight reduction, and runtimes are only moder-

ately increased at this level.[3] Being the fraction of negative weights still considerably large, the actual benefits of folding with respect to Born spreading have to be carefully assessed taking into account a realistic number of generated events, as well as time and CPU cost spent in the showering and detector-simulation phase.

5 Possible extensions of the method

The Born-spreading strategy introduced in this paper aims at alleviating the impact of negative $\mathbb{S}$ weights in MC@NLO simulations. Its goals are the same as the well-established folding procedure, with which it can be naturally compared. On one hand, folding guarantees a progressive reduction of negative weights which scales with the number of employed folds. This means that in principle a minimum fraction of negative weights can be reached. However, the resources needed to achieve a significant reduction may well outweigh the advantages of the reduction itself, which requires case-by-case assessment. On the other hand, the Born-spreading technique on paper has a more limited scope than folding: being based on a re-distribution of the Born contribution in the radiative phase space, it does not fully capture how the distribution of negative weights is correlated with the Born phase-space variables. In this sense, once a spreading function has been defined, the procedure is not parametrically improvable. Nevertheless, the strategy is computationally so much cheaper than folding to warrant consideration as the new default operational mode in MG5_aMC, especially for the production of large event samples. We note incidentally that the negative-weight events in the POWHEG approach [6] have the same origin as the negative $\mathbb{S}$ weights in MC@NLO, whence Born spreading could prove beneficial with that approach as well.

Although the Born-spreading results shown in Sect. 4 look encouraging, we stress that the analysis of the method is still at a preliminary stage, and much is to be done to fully exploit its potential. The first immediate direction would be that of optimising the involved parameters (N_ξ, N_y, N_{spread}), in order to potentially reduce the time offset in step-0 without degrading the negative-weight reduction. Furthermore, the functional form of the spreading function itself should be more carefully assessed: if the one defined in Eq. (4) is quite natural, we notice that it just exploits the region in which the no-Born $\mathbb{S}$ cross section is locally negative. One could for instance explore spreading functions that erode the positive integrand in order to fill more efficiently the negative region, achieving a further compensation of negative weights. Furthermore, the ample freedom in the choice of spreading function makes this problem an ideal test ground for neural-network algorithms, which would then have to learn the optimal spreading function minimising a loss function related to the fraction of negative weights. This method could also be expanded to capture some of the correlations between negative weights and Born variables that are lost in the current implementation of spreading. Another way to extend the method is to include the virtual contribution in the spreading. It can either be spread together with the Born contribution, or, a separate spreading function can be defined for the virtual corrections. For the latter case, the Born and virtual spreading functions would be in competition and have to be determined with some iterative procedure, or learning algorithm. Moreover, Born spreading is also compatible with folding, namely one could envisage to act with spreading in step-0, and subsequently fold the spread sample, thereby reducing the number of required folds for a given target negative fraction. Finally, Born spreading achieves a reduction of negative weights which is naturally complementary to that of MC@NLO-Δ, and their joint effect should be thoroughly assessed in realistic event generation.

6 Conclusions

We have presented a new method, dubbed *Born spreading*, for the reduction of negative-weight $\mathbb{S}$ events in the MC@NLO matching formalism. The method is based on the consideration that the Born contribution, usually integrated along with all the other components of the $\mathbb{S}$ integrand, by its own nature just depends on non-radiative variables, hence it can be re-distributed (*spread*) arbitrarily in the radiative phase space. This freedom is exploited to find a spreading function that maximally reduces the fraction of negative $\mathbb{S}$ weights.

We have described the implementation of this idea in the context of MadGraph5_aMC@NLO, and characterised its performances for a variety of physical processes at the LHC in terms of running times and negative-weight reduction. A detailed comparison has been performed both against the default MadGraph5_aMC@NLO mode, and against the folding method. Born spreading is particularly efficient in achieving moderate to sizeable (but not extreme) negative-weight reductions at negligible extra CPU cost, at variance with folding, which can achieve more consistent reductions, but typically requiring deployment of significant computing resources.

We have finally indicated several possible ways to extend the Born-spreading idea, in the hope of optimising its perfor-

[3] Notably, step-2 in $pp \to Hb\bar{b}$ with $2 \times 2 \times 1$ folding is even faster than the default: most of the integration time in this case is spent in the evaluation of the virtual contribution, which is not affected by folding. Hence the enhanced stability of the integrand achieved with folding is sufficient to guarantee smaller numerical error, and an increased unweighting efficiency.

mances and of making it an established valuable tool for the reduction of negative-weight events.

Acknowledgements RF has been partially supported by the Swedish Research Council contract numbers 2016-05996 and 2020-04423. PT has been partially supported by the Italian Ministry of University and Research (MUR) through Grant PRIN 2022BCXSW9 and by Compagnia di San Paolo through Grant TORP_S1921_EX-POST_21_01.

Data Availability Statement This manuscript has no associated data or the data will not be deposited. [Authors' comment: This is a theoretical study, and it has no experimental data.]

References

1. T. Aarrestad, et al., in *Snowmass* 2021, ed. by P. Canal, et al. (2020). https://doi.org/10.5281/zenodo.4009114
2. ATLAS Software and Computing HL-LHC Roadmap. Tech. rep., CERN, Geneva (2022). https://cds.cern.ch/record/2802918
3. CMS Phase-2 Computing Model: Update Document. Tech. rep., CERN, Geneva (2022). https://cds.cern.ch/record/2815292
4. R. Frederix, S. Frixione, S. Prestel, P. Torrielli, JHEP **07**, 238 (2020). https://doi.org/10.1007/JHEP07(2020)238
5. S. Frixione, B.R. Webber, JHEP **06**, 029 (2002). https://doi.org/10.1088/1126-6708/2002/06/029
6. P. Nason, JHEP **11**, 040 (2004). https://doi.org/10.1088/1126-6708/2004/11/040
7. S. Frixione, P. Nason, C. Oleari, JHEP **11**, 070 (2007). https://doi.org/10.1088/1126-6708/2007/11/070
8. S. Jadach, W. Płaczek, S. Sapeta, A. Siódmok, M. Skrzypek, JHEP **10**, 052 (2015). https://doi.org/10.1007/JHEP10(2015)052
9. P. Nason, G.P. Salam, JHEP **01**, 067 (2022). https://doi.org/10.1007/JHEP01(2022)067
10. K. Danziger, S. Höche, F. Siegert, arXiv:2110.15211 [hep-ph]
11. J.R. Andersen, C. Gütschow, A. Maier, S. Prestel, Eur. Phys. J. C **80**(11), 1007 (2020). https://doi.org/10.1140/epjc/s10052-020-08548-w
12. B. Nachman, J. Thaler, Phys. Rev. D **102**(7), 076004 (2020). https://doi.org/10.1103/PhysRevD.102.076004
13. J.R. Andersen, A. Maier, Eur. Phys. J. C **82**(5), 433 (2022). https://doi.org/10.1140/epjc/s10052-022-10372-3
14. J.R. Andersen, A. Maier, D. Maître, Eur. Phys. J. C **83**(9), 835 (2023). https://doi.org/10.1140/epjc/s10052-023-11905-0
15. 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
16. P. Nason, (2007). arXiv:0709.2085 [hep-ph]
17. S. Frixione, P. Nason, G. Ridolfi, (2007). arXiv:0707.3081 [hep-ph]
18. S. Alioli, P. Nason, C. Oleari, E. Re, JHEP **06**, 043 (2010). https://doi.org/10.1007/JHEP06(2010)043
19. S. Frixione, Z. Kunszt, A. Signer, Nucl. Phys. B **467**, 399 (1996). https://doi.org/10.1016/0550-3213(96)00110-1
20. R.D. Ball et al., Nucl. Phys. B **867**, 244 (2013). https://doi.org/10.1016/j.nuclphysb.2012.10.003
21. M. Cacciari, G.P. Salam, G. Soyez, Eur. Phys. J. C **72**, 1896 (2012). https://doi.org/10.1140/epjc/s10052-012-1896-2