# MCgrid

# MCgrid: Projecting cross section calculations on grids☆

Luigi Del Debbio [a], Nathan Hartland [a,b,*], Steffen Schumann [b]

[a] *The Higgs Centre for Theoretical Physics, University of Edinburgh, UK*
[b] *II. Physikalisches Institut, Georg-August-Universität Göttingen, Göttingen, Germany*

## ARTICLE INFO

## ABSTRACT

MCgrid is a software package that provides access to the APPLgrid interpolation tool for Monte Carlo event generator codes, allowing for fast and flexible variations of scales, coupling parameters and PDFs in cutting edge leading- and next-to-leading-order QCD calculations. This is achieved by providing additional tools to the Rivet analysis system for the construction of MCgrid enhanced Rivet analyses. The interface is based around a one-to-one correspondence between a Rivet histogram class and a wrapper for an APPLgrid interpolation grid. The Rivet system provides all of the analysis tools required to project a Monte Carlo weight upon an observable bin, and the MCgrid package provides the correct conversion of the event weight to an APPLgrid fill call. MCgrid has been tested and designed for use with the SHERPA event generator, however as with Rivet the package is suitable for use with any code which can produce events in the HepMC event record format.

**Program summary**

*Program title:* MCgrid

*Catalogue identifier:* AESS_v1_0

*Program summary URL:* http://cpc.cs.qub.ac.uk/summaries/AESS_v1_0.html

*Program obtainable from:* CPC Program Library, Queen's University, Belfast, N. Ireland

*Licensing provisions:* Standard CPC licence, http://cpc.cs.qub.ac.uk/licence/licence.html

*No. of lines in distributed program, including test data, etc.:* 159478

*No. of bytes in distributed program, including test data, etc.:* 2916690

*Distribution format:* tar.gz

*Programming language:* C++, shell, Python.

*Computer:* PC running Linux, Mac.

*Operating system:* Linux, Mac OS.

*RAM:* Varying

*Classification:* 11.2, 11.5, 11.9.

*External routines:* HepMC (http://lcgapp.cern.ch/project/simu/HepMC/),
Rivet (https://rivet.hepforge.org/), APPLgrid (http://applgrid.hepforge.org/).
A Sherpa (https://sherpa.hepforge.org/trac/wiki) installation is also required.

*Nature of problem:*
Efficient filling of cross section grid files from fully exclusive parton-level Monte Carlo events.

*Solution method:*
Analyse Monte Carlo events via the Rivet program, which projects events on discretised cross section tables from APPLgrid.

*Running time:*
Approximately 6 minutes per 1 million Drell–Yan events from Sherpa. This includes both the event generation and the mcgrid computations. The times can vary quite dramatically. The process used in the test case (which is a relatively fast one) takes about 2 minutes 30 seconds for the initial (phase-space fill) run and about 3 minutes for the second and final run. This is for 1 million events on a 2.9GhZ ivy bridge i7.

## 1. Introduction

Measurements at the Large Hadron Collider, operating at the highest centre-of-mass energies ever achieved in accelerator-based experiments, allow for precision studies of a vast range of final states. In particular large final state jet multiplicities become accessible. Such multi-jet events constitute both interesting signals and important backgrounds to new physics searches. Accordingly they must be described with a theoretical precision that either matches or exceeds the experimental accuracy.

The dominant corrections to a given production process at a hadron collider originate from QCD. The demand for precise theoretical predictions describing LHC production processes has therefore triggered an enormous development in the field of next-to-leading-order QCD calculations—making NLO QCD accuracy the new standard. These developments span from the largely automated calculation of NLO QCD cross sections to the combination of NLO matrix elements with parton-shower simulations; see e.g. Ref. [1] and references therein.

The enormous progress recently experienced in the field of NLO QCD calculations was sparked by two important developments. First, the introduction of fast and efficient methods for the calculation of virtual amplitudes; see for instance [2–7]. Second, the organisation and implementation of complete NLO calculations in the framework of parton-level Monte Carlo event generators such as Helac [8], MadGraph [9] or SHERPA [10,11]. All of these approaches rely on using an infrared-subtraction formalism, e.g. the Catani–Seymour [12,13] or the Frixione–Kunszt–Signer [14] method as implemented in automated subtraction term generators; see Refs. [15–20]. The real-emission corrections as well as the phase-space integration are handled by tree-level matrix-element generators such as AMEGIC [21], COMIX [22], MadGraph [9] or Helac [23]. Virtual amplitudes, typically provided by specialised one-loop generators such as BlackHat [7], GoSam [24], Helac-1Loop [25], MadLoop [26], NJET [27], OpenLoops+Collier [28] or Recola [29] can be incorporated via the BLHA interface [30,31]. Examples of recent NLO calculations that have been performed using a combination of the tools listed above include: $W+4$, 5 jets [32,33], $Z+4$ jets [34], $4-$jet and $5-$jet production [35–37], $t\bar{t}+2$ jets [38] and $\gamma\gamma+2$ jets [39]. Most of these new tools are now available to perform NLO QCD event generation and for use in LHC data analysis.

In order to provide a comprehensive analysis of the uncertainty on an NLO QCD prediction, the value of the strong coupling, parton distribution functions, renormalisation and factorisation scales must all be varied to characterise the dependence of the final result upon these uncertain parameters. In particular, the accurate propagation of PDF uncertainty through the calculation requires a great deal of repeated runs. For similar reasons, it is often challenging to include collider processes in global PDF fits beyond the LO approximation. Indeed, for cutting edge high-multiplicity processes the frequent repetition of the full NLO calculation with a modified set of parameters is prohibitive as it requires too much CPU time.

There are two complementary solutions to this problem that allow for an a-posteriori variation of scale choices and parameters. First, parton-level events supplemented with suitable weights can be stored explicitly. One such approach based on ROOT NTuples has recently been presented in [40]. For the price of having large event files and having to loop over many events, one can in principle analyse any final state observable with arbitrary parameter choices for the process under consideration. The second approach is based on cross section interpolation grids that represent a given differential cross section binned in the incoming partonic momentum fractions $x$ and the associated process' scale $Q^2$. Examples of such interpolation tools are APPLgrid [41] and FastNLO [42,43]. While interpolating grids are defined only for a specific final state observable, they allow for a very quick re-evaluation of the respective cross section and have considerably reduced disk space requirements.

In this note we present MCgrid, a tool that provides a direct interface from parton-level Monte Carlo event generators performing LO or NLO QCD calculations to APPLgrid. We utilise the Rivet [44] MC analysis system to provide the experimental analysis tools needed in classifying the event final state into the appropriate observable bins. We convert the event weight information into an appropriate APPLgrid fill call, while correctly taking into account the full PDF dependence. In the implementation we assume that the event weights are generated via the Catani–Seymour dipole subtraction scheme. Furthermore we provide the tools for an automated determination of the breakdown of a process into its contributing subprocesses. We can identify individual initial state flavour channels that can be combined, once their PDF dependence is factored out. MCgrid is implemented as a C++ framework providing standard Rivet analyses with additional interpolation grid functionality.

The present paper is organised as follows. In Section 2 we give a detailed account of the ingredients needed for a consistent parameter variation in QCD calculations, and we describe the basics of interpolation tools such as APPLgrid. In Section 3 we present our implementation of an interface allowing for the fill of APPLgrid cross section tables from fully exclusive parton-level Monte Carlo events in the HepMC format. In Section 4 we give detailed information on how to set up and execute MCgrid. We give examples and describe the validation of our approach in Section 5. The conclusions are presented in Section 6.

## 2. Parameter variation in NLO calculations

In this section we shall perform a brief overview of the methods and techniques available for performing efficient variation of QCD parameters in NLO calculations.

### 2.1. Reweighting leading order MC calculations

QCD computations of final state observables involve detector acceptances, or jet algorithms, in the computation of the perturbative coefficients. The latter are then convoluted with PDFs that encode the nonperturbative information about the partonic content

of hadrons. Let us start by examining the case of such a calculation at leading order in the strong coupling, computing a cross section for the production of some final state $X$,

$$\sigma_{pp \to X}^{\text{LO}} = \int dx_1 \, dx_2 \int d\Phi_n \left( \frac{\alpha_s(\mu_R^2)}{2\pi} \right)^{p_{\text{LO}}}$$
$$\times f_i(x_1, \mu_F^2) f_j(x_2, \mu_F^2) \, |\mathcal{M}_{ij \to X}|^2. \tag{1}$$

The PDFs for the partons inside the nucleons are denoted by $f_i$ and $f_j$, and the sum over all partonic channels is understood. We have explicitly written the dependence of the PDFs on the momentum fractions $x_1$, $x_2$ and on the factorisation scale $\mu_F^2$. The calculation's dependence on the value of the strong coupling constant $\alpha_s$ is also made explicit. $|\mathcal{M}_{ij \to X}|^2$ is the parton-level squared matrix element for the $2 \to n$ process, differential in the final state phase space.

This convolution may be simplified by making use of the initial state flavour symmetries of the parton-level process $\hat{\sigma}$. Grouping the partonic sub channels which differ only by their PDFs into QCD subprocesses, Eq. (1) can be written as

$$\sigma_{pp \to X}^{\text{LO}} = \int dx_1 dx_2 \int d\Phi_n \left( \frac{\alpha_s(\mu_R^2)}{2\pi} \right)^{p_{\text{LO}}} F_l(x_1, x_2, \mu_F^2) \, |\mathcal{M}_{l \to X}|^2, \tag{2}$$

where the subprocess parton density is given by

$$F_l(x_1, x_2, \mu_F^2) = \sum_{i,j=0}^{N_{pdf}} C_{ij}^{(l)} f_i(x_1, \mu_F^2) f_j(x_2, \mu_F^2). \tag{3}$$

The matrix of coefficients $C_{ij}^{(l)}$ is specified by the symmetries of the parton-level cross section, determined by whether or not the partonic channel $ij$ belongs to the subprocess $l$, i.e.

$$C_{ij}^{(l)} = \begin{cases} 1, & ij \in l, \\ 0, & ij \notin l. \end{cases}$$

The integral in Eq. (2) can be computed by Monte Carlo integration:

$$\sigma_{pp \to X}^{\text{LO}} = \sum_{e=1}^{N_{\text{evt}}} \left( \frac{\alpha_s(k_e)}{2\pi} \right)^{p_{\text{LO}}} w_e(k_e) F_{l_e}(k_e), \tag{4}$$

$$w_e(k_e) = |\mathcal{M}_{l_e \to X}(k_e)|^2 \, \Pi_{\text{ps}}(k_e) \Theta(k_e - k_{\text{cuts}}), \tag{5}$$

where the index $e$ runs over the sample of generated MC events, $\Pi_{\text{ps}}(k_e)$ is the corresponding event phase space weight, $F_{l_e}$ is the subprocess density for the event subprocess $l_e$ and

$$k_e = \left\{ p_1, \ldots, p_n, x_1, x_2, \frac{\mu_F^2}{Q^2}, \frac{\mu_R^2}{Q^2} \right\}, \tag{6}$$

are the set of kinematics associated with the event. Eq. (5) shows that the weight $w_e$ is obtained by evaluating the short-distance amplitude $\mathcal{M}$ for the kinematics generated for the given event $e$, taking the kinematic cuts and final state phase space weight into account.

At this order, the procedure for performing a variation of the QCD parameters present in the calculation is fairly straightforward. Provided that the full event record is stored for each entry in the sum in Eq. (4), a different PDF may be used simply by multiplying each weight by factors of $F_{l_e}^{(\text{new})}/F_{l_e}^{(\text{old})}$. A similar procedure may of course be used to reweight the LO power of $\alpha_S$, and to vary the perturbative scales to which the calculation's only dependence is through $\alpha_S$ and the PDFs.

## 2.2. Reweighting NLO event weights

Performing an event weight reweighting at NLO provides more of a challenge. The parton-level cross section develops a dependence upon the perturbative scales used in the calculation, and a subtraction mechanism must be employed in the Monte Carlo integration to ensure the cancellation of singularities in the evaluated integrands.

The use of a subtraction algorithm makes the precise PDF and scale dependence of the event weight considerably more complicated, due to the presence of integrated subtraction terms proportional to Altarelli–Parisi splitting functions. In this section we shall discuss the reweighting of events produced from the SHERPA event generator which utilises the Catani–Seymour dipole subtraction method [12].

In general, a subtraction scheme will separate the differential cross section at NLO into four distinct parts:

$$\sigma_{pp \to X}^{\text{NLO}} = \int d\hat{\sigma}^{\text{B}} + \int d\hat{\sigma}^{\text{V}} + \int d\hat{\sigma}^{\text{I}} + \int d\hat{\sigma}^{\text{RS}}. \tag{7}$$

When attempting to reweight an NLO event sample, the weights must be treated differently according to whether they belong to the Born (B) or Real Subtracted (RS) integrals, or if they correspond to a Virtual (V) or Integrated subtraction (I) event.

In the case of the B or RS weights, their treatment is identical to the leading order case, demonstrated in Eq. (4) and they may be reweighted by multiplication with the appropriate new PDF and strong coupling factors. The B and RS integrals may be performed as

$$\int d\hat{\sigma}^{\text{B/RS}} = \sum_{e=1}^{N_{\text{evt}}} \left( \frac{\alpha_s(\mu_R^2)}{2\pi} \right)^p F_{l_e}(k_e) w_e^{(0)}(k_e), \tag{8}$$

where $p$ is either the leading (B) or next-to-leading (RS) order of $\alpha_S$ and we denote weight contributions with Born-like scale and PDF dependence by $w^{(0)}$.

For the events originating from virtual diagrams, the renormalisation of the matrix element introduces an explicit dependence on the renormalisation scale. In order to accurately reweight the sample, the terms proportional to scale logarithms should be kept track of separately.

The Monte Carlo integral of the virtual contribution for an arbitrary choice of the renormalisation scale $\mu_R$ can then be computed as

$$\int d\hat{\sigma}^{\text{V}} = \sum_{e=1}^{N_{\text{evt}}} \left( \frac{\alpha_s(\mu_R^2)}{2\pi} \right)^{p_{\text{NLO}}}$$
$$\times F_{l_e}(k_e) \left\{ w_e^{(0)}(k_e) + \ell w_e^{(1)}(k_e) + \ell^2 w_e^{(2)}(k_e) \right\}, \tag{9}$$

where $\ell = \log \left( \mu_R^2 / \mu_{R,\text{old}}^2 \right)$. It is therefore clear that to provide an accurate scale variation the additional contributions $w^{(1)}$ and $w^{(2)}$ must be distinguished from the central scale weight $w$ in the event record.

The third case, that of the integrated subtraction, introduces further complexity. Specifically the PDF dependence of each event weight differs considerably from the leading order case. The counter terms introduced in the subtraction algorithm typically take the form of a Born-type matrix element multiplied by a splitting function. These weights must therefore be expanded over many initial state parton flavours.

To correctly perform a PDF reweighting then, the full dependence structure must also be detailed in the event record. If we

assume that all integrated subtraction configurations are represented by just one event, the integral of the I contribution is performed as

$$
\int d\hat{\sigma}^{\mathrm{I}} = \sum_{e=1}^{N_{\mathrm{evt}}} \left( \frac{\alpha_s(\mu_R^2)}{2\pi} \right)^{p_{\mathrm{NLO}}} \left\{ f_1(i, x_1, \mu_F^2) \, w_e^{(0)} f_2(j, x_2, \mu_F^2) \right.
$$
$$
+ \left( \sum_{k=1}^{4} f_1^{(k)}(i, x_1, x_1', \mu_F^2) \, w_{e,k}^{(3)} \right) f_2(j, x_2, \mu_F^2)
$$
$$
\left. + f_1(j, x_1, \mu_F^2) \left( \sum_{k=1}^{4} w_{e,k}^{(4)} f_2^{(k)}(j, x_2, x_2', \mu_F^2) \right) \right\}, \qquad (10)
$$

where $x/x'$ denote the values of parton-$x$ in the integration. The $w_k^{(3)}$ and $w_k^{(4)}$ denote the various contributions to the weight arising from the Altarelli–Parisi splitting functions for a parton splitting in the first beam or second beam respectively. Here we have used for the PDFs ($r = 1$ or $2$)

$$
f_r^{(1)}(i, x, x', \mu_F^2) = \begin{cases} i = \text{quark:} & f_r(i, x, \mu_F^2), \\ i = \text{gluon:} & \sum_q f_r(q, x, \mu_F^2), \end{cases} \qquad (11)
$$

$$
f_r^{(2)}(i, x, x', \mu_F^2) = \begin{cases} i = \text{quark:} & f_r(i, x/x', \mu_F^2)/x', \\ i = \text{gluon:} & \sum_q f_r(q, x/x', \mu_F^2)/x', \end{cases} \qquad (12)
$$

$$
f_r^{(3)}(i, x, x', \mu_F^2) = f_r(g, x, \mu_F^2), \qquad (13)
$$

$$
f_r^{(4)}(i, x, x', \mu_F^2) = f_r(g, x/x', \mu_F^2)/x'. \qquad (14)
$$

The sums over quarks $q$ are taken over the active quark/antiquark flavours at scale $\mu_F^2$. Furthermore, $i, j$ specify the incoming parton flavours in the event $e$ before any splitting, with momentum fractions $x_1$ and $x_2$, respectively.

Projecting each individual partonic channel weight onto the subprocess basis, the same integral can be done as

$$
\int d\hat{\sigma}^{\mathrm{I}} = \sum_{e=1}^{N_{\mathrm{evt}}} \left( \frac{\alpha_s(\mu_R^2)}{2\pi} \right)^{p_{\mathrm{NLO}}} \left\{ F_{l_e}(x_1, x_2, \mu_F^2) \, w_e^{(0)} \right.
$$
$$
+ \sum_s^{N_{\mathrm{sub}}} F_s(x_1/x_1', x_2, \mu_F^2) \, \tilde{w}_{e,s}^{(3)}
$$
$$
\left. + \sum_s^{N_{\mathrm{sub}}} F_s(x_1, x_2/x_2', \mu_F^2) \, \tilde{w}_{e,s}^{(4)} \right\}, \qquad (15)
$$

where the subprocess basis weights $\tilde{w}_s^{(3/4)}$ are obtained from the flavour basis weights $w_k^{(3/4)}$. To be able to perform the PDF reweighting, the values of the additional weights and $x'$ values must be made explicit in the event record.

In summary, there are five classes of contributions that should be distinguished in each event weight.

- $w^{(0)}$: Weights with Born-like PDF and scale dependence.
- $w^{(1)}$: Weights proportional to first order scale logs.
- $w^{(2)}$: Weights proportional to second order scale logs.
- $w^{(3)}$: Weights originating from integration over $x_1$ in I events.
- $w^{(4)}$: Weights originating from integration over $x_2$ in I events.

All of these must be separated into independent initial state flavour contributions, and possibly projected onto a subprocess basis. It can be seen however, that with a central scale choice (i.e neglecting the $w^{(1/2)}$ terms) it is possible to convert the full NLO calculation into a form similar to the leading order case in Eq. (4). By projecting

the $w^{(3/4)}$ weights onto independent events, we may write the full NLO result at central scales as

$$
\sigma_{pp \to X}^{\mathrm{NLO}} = \sum_{e=1}^{N_{\mathrm{evt}}} \left( \frac{\alpha_s(k_e)}{2\pi} \right)^{p_e} F_{l_e}(k_e) w_e(k_e), \qquad (16)
$$

where $p_e$ denotes the order in $\alpha_S$ of the event $e$.

A prescription for identifying all these contributions in NLO records was recently described, along with a procedure and software package for performing the reweighting in the ROOT NTuple[40] format designed by the BlackHat collaboration and implemented in SHERPA. With the various complexities of reweighting NLO events in the Catani–Seymour formalism carefully treated as in the BlackHat NTuple format, all the required information is present to perform an accurate variation of the QCD parameters present in the calculation.

However there remains a difficulty with the event reweighting approach, in that the full event record must be stored and reanalysed for each variation of a parameter in the calculation. In high statistics samples, this can mean storing hundreds of gigabytes of event files. Reanalysing these events may still be a nontrivial computational task simply in the reading and reprocessing of events. Certainly for performance sensitive applications such as PDF fitting, a reweighting approach remains prohibitively expensive.

### 2.3. Interpolation tools

The fundamental difficulty in the event reweighting procedure lies in the sum over events in Eq. (16). In applications where both speed and accuracy are important, such a dependence on the statistics of the event sample is problematic. Possible solutions to this problem have been available for some time in the form of event weight interpolating tools such as the APPLgrid [41] and FastNLO [42,43] projects. The basic principle of these methods is to represent the PDFs used in the product in Eq. (16) upon an interpolating grid in $x$ and $Q^2$.

The number of points in each direction is denoted by $N_x$ and $N_Q$ respectively; therefore

$$
F_l(x_1, x_2, Q^2) = \sum_{\alpha\beta}^{N_x} \sum_{\tau}^{N_Q} F_l(x_\alpha, x_\beta, Q_\tau^2) \, \mathcal{I}^{(\alpha)}(x_1) \mathcal{I}^{(\beta)}(x_2) \mathcal{I}^{(\tau)}(Q^2),
$$
$$
\qquad (17)
$$
$$
= \sum_{\alpha\beta}^{N_x} \sum_{\tau}^{N_Q} F_l(x_\alpha, x_\beta, Q_\tau^2) \, \mathcal{I}^{\alpha\beta,\tau}(x_1, x_2, Q^2). \qquad (18)
$$

If we once again neglect at first the terms in the calculation proportional to logarithms of the chosen scales, the Monte Carlo computation of the cross section in Eq. (16) can be written as

$$
\sigma = \sum_e \sum_{\alpha\beta}^{N_x} \sum_{\tau}^{N_Q} \left( \frac{\alpha_s(Q_\tau^2)}{2\pi} \right)^{p_e} F_{l_e}(x_\alpha, x_\beta, Q_\tau^2) \, \mathcal{I}^{\alpha\beta,\tau}(k_e) w_e(k_e).
$$
$$
\qquad (19)
$$

Subdividing the weights by perturbative order and rearranging the terms in the sum yields

$$
\sigma = \sum_p \sum_l^{N_{\mathrm{sub}}} \sum_{\alpha\beta}^{N_x} \sum_{\tau}^{N_Q} \left( \frac{\alpha_s(Q_\tau^2)}{2\pi} \right)^p F_l(x_\alpha, x_\beta, Q_\tau^2) \, W_{\alpha\beta,\tau}^{(l)(p)}, \qquad (20)
$$

where the sum over the events is performed:

$$
W_{\alpha\beta,\tau}^{(l)(p)} = \sum_e \left[ \delta_{l,l_e} \delta_{p,p_e} \mathcal{I}^{\alpha\beta,\tau}(k_e) \, w_e(k_e) \right]. \qquad (21)
$$

Note that Eq. (20) is a simple sum over the points in the $x, Q^2$ grids. These kinematic points are chosen when defining the grids, and

do not change. Once the interpolated weights $W_{\alpha\beta,\tau}^{(l)(p)}$ for a given process are stored, the computation of the cross section is very fast, since it does not involve a loop over the generated MC events.

If we are interested in performing a calculation differential in some observable or otherwise a quantity that may be represented in a histogram, the event weight final states quantified in the kinematics $k_e$ should be projected onto the relevant observable bin. In this way, the interpolated weight grid is separated into final state observable bins,

$$W_{\alpha\beta,\tau}^{(l)(p)(b)}$$

$$= \sum_e \left[ \delta_{l,l_e} \delta_{p,p_e} \, \ell^{\alpha\beta,\tau}(k_e) w_e(k_e) \Theta(k_b^{\max} - k_e) \Theta(k_e - k_b^{\min}) \right], (22)$$

where the $k_b^{\min}$, $k_b^{\max}$ represent the required kinematic limits for the observable bin $b$. The cross section calculation in Eq. (20) is therefore separated into a differential observable. As the sum over the events has been performed, there is no requirement for the detailed event by event information present in the full record, drastically reducing the storage space requirements. Of course this comes with a caveat. Once the weights are interpolated and stored, the produced grid is restricted to the experimental projection defined by the cuts used when constructing it. As the event final state kinematics are discarded after the projection, each grid is uniquely identified by its projection and binning.

In such an approach the additional scale dependent terms described in the previous section are discarded. Therefore it may at first seem that performing scale variations would be challenging. However, these terms may be inferred by a simple calculation that is made considerably easier by the presence of the full interpolated weight sample, separated by perturbative order.

By demanding that the scale derivative of the cross section result be zero to $\mathcal{O}(\alpha_S^{\mathrm{NNLO}})$ the terms proportional to logarithms of the factorisation and renormalisation scales may be calculated. This procedure is discussed in detail in Ref. [41]. The solution presented there relies on the HOPPET program [45] that provides an implementation of the QCD splitting functions.

## 3. The MCgrid Interface

Access to interpolating tools has so far been restricted to NLO cross section calculators such as MCFM [46] and nlojet++, [47]. Here we shall describe an interface, named MCgrid, that will allow for efficient access to the latest NLO calculations as obtained with fully exclusive parton-level event generators.

We are using the APPLgrid framework to provide the interpolation grid definitions, along with its comprehensive suite of tools for recalculating the stored cross section with varied parameters and scales. The APPLgrid package provides a set of standard methods for the construction and filling of the interpolated weight grids described in Section 2.3.

In order to develop an interface for event generators to the APPLgrid package, a number of issues should be addressed. First the projection of each event weight upon an observable bin, quantified in Eq. (22), must be performed by some set of analysis tools.

Second, the PDF dependence of each event weight from a generator may be complicated by the presence of integrated subtraction terms in the event sample as demonstrated in Section 2.2. Such complex PDF dependence should be removed by converting the single event weight into individual fills, each corresponding to a single pair of initial state partons.

Finally, full Monte Carlo event generators typically differ from NLO cross section integrators in their evaluation of the cross section. While many NLO codes will perform the integration directly in the subprocess PDF basis as in Eq. (2), event generators

will generally produce weights exclusive in the initial state, meaning that the integral in Eq. (1) is performed with the full PDF basis. As the full basis is rather inefficient for the reweighting of a fixed order calculation, the produced parton–parton weights should be converted into subprocess weights while preserving the statistical accuracy of the calculation.

The MCgrid package provides a conversion of NLO event generator weights into a form suitable for interpolation. The projection of each event weight onto an observable bin is performed by the Rivet analysis system. In this section we shall briefly describe the features of the interface along with a description of how it may be used in practice.

### 3.1. Event final state analysis

When developing an interface to process Monte Carlo events into observable bins in an interpolated weight grid, clearly a suite of experimental analysis tools are required in order to perform the projection from the event's final state to the appropriate observable bin. Rather than re-implementing such an analysis suite we have opted to make use of the flexibility of the Rivet analysis system to provide such tools.

The Rivet framework provides a standard set of analysis methods, along with tools for the reading of event records from disk or processed on the fly directly from an event generator. The Rivet system is becoming a standard in Monte Carlo analysis for LHC observables. While it is geared mainly for the analysis of parton showered/hadronised final states, it can also process events from fixed order calculations. A typical Rivet analysis is associated with an experimental measurement, and as such generally includes the experimental data and binning information to provide an automated comparison. However, the inclusion of experimental reference data is not a requirement.

The MCgrid interface provides a set of additional methods to an analysis procedure in Rivet for the generation of APPLgrid interpolated weight grid files. The final state cuts and binnings are performed as usual in a Rivet analysis, with the MCgrid functionality requiring only simple modifications to produce a weight grid. The interface has been designed to follow Rivet standards as much as possible, with each produced APPLgrid file corresponding to a histogram in the analysis.

### 3.2. Interpolating NLO event records

Rivet analyses are based upon event records in the standard HepMC format [48]. By design this format provides mostly final state information, therefore some additional data is required in order to fill the APPLgrid weight grids.

The information on the parton distribution functions and the corresponding $x$ and $Q^2$ values are already hosted by the HepMC:: PDFInfo class. The running coupling evaluated at the process' renormalisation scale is provided via the method HepMC:: GenEvent::alphaQCD(). This information should be correctly filled in the exported event record.

On top of this standard information, all HepMC events that are passed to an MCgrid enabled analysis must also include the power of the strong coupling characterising each event. This is required so that the correct power may be removed from the event weight; it must therefore be the absolute power rather than the power relative to leading order.

Additionally in the filling of an interpolation grid from an NLO event weight the issue of the precise PDF dependence of the weight arises as described in Section 2.1. In MCgrid we assume by default that each integrated subtraction event weight carries this full PDF dependence. The full event weight must therefore be expanded in

a basis of terms corresponding to each application of a splitting function, depending on the factorisation scheme used to define the PDF set. The MCgrid interface expects as inputs the same basis as is used for the PDF reweighting terms in the BlackHat NTuple format described in [40] with the exception of the additional weights accounting for the possible variation of the factorisation scale which are not needed when performing scale variation using the approach described in Ref. [41].

This non-standard information should be appended to each event in the record as additional entries in the HepMC::WeightContainer vector. In this way, no modification to the standard HepMC format is necessary. Explicitly, the full set of HepMC::WeightContainer entries expected by MCgrid is

- WeightContainer[0-3] = [Reserved for Generator use].
- WeightContainer[4] = Event's power of $\alpha_S$.
- WeightContainer[5] = Total event weight $\propto f_{i/H1}f_{j/H2}$, where $ij$ labels the hard event's initial state flavours.
- WeightContainer[6] = Number of additional weights to come.
- WeightContainer[7] = Secondary momentum fraction $x'_1$ for beam 1.
- WeightContainer[8] = Secondary momentum fraction $x'_2$ for beam 2.
- WeightContainer[9+i] = usr_wgts[i], where $i = 0, \ldots, 9$ and with the usr_wgts defined according to the basis in [40].

Here in the HepMC::WeightContainer all powers of $\alpha_S$ should remain present in the specified weights, but the PDF values should not, as is the case for the BlackHat NTuple record. It is worthwhile noting here that for Born-like events, real-emission corrections and real-emission counter-configurations no additional weight information needs to be provided; it suffices to have the HepMC::PDFInfo properly filled, along with the standard event weight and its power of the strong coupling. In this instance the HepMC::WeightContainer entry 6 should be set to zero.

The additional weight information listed above is needed, as we assume that all integrated subtraction configurations get represented by just one event. Accordingly the corresponding event weight carries a dependence on various initial state flavour combinations and PDFs. For a consistent filling of cross section grids we have to disentangle all these contributions and fill the partial event weights in the suitable subprocess grid. Recalling the decomposition of the full event weight from Ref. [40] and described in Eq. (10) the decomposition of the full event weight $w$, allowing for a consistent variations of PDFs, reads

$$
w = \texttt{WeightContainer}[5]
$$
$$
+ \left( \sum_{k=1}^{4} f_1^{(k)}(i, x_1, x'_1, \mu_F^2) \, w_k \right) f_2(j, x_2, \mu_F^2)
$$
$$
+ f_1(j, x_1, \mu_F^2) \left( \sum_{k=1}^{4} f_2^{(k)}(j, x_2, x'_2, \mu_F^2) \, w_{k+4} \right), \tag{23}
$$

where the decomposed weights and $x'$ values must be provided by

$$
x'_1 = \texttt{WeightContainer}[7],
$$
$$
x'_2 = \texttt{WeightContainer}[8],
$$
$$
w_k = \texttt{WeightContainer}[k+8].
$$

In the case where the Monte Carlo code you wish to interface with MCgrid provides also the integrated counter-term weights as independent events, the full PDF dependence of the event sample can be described by Eq. (16). If this is the case, then all event weights may be treated as in the Born or real-emission cases, with the HepMC::WeightContainer entry 6 zeroed.

These modifications have been implemented into the HepMC_Short output of SHERPA as of version 2.0. Therefore MCgrid is able to process SHERPA output without modification.

### 3.3. Automated subprocess determination

Assuming fully exclusive parton-level events as inputs for MCgrid we have to deal with fully exclusive partonic initial states. This corresponds to the maximal dimensionality of the flavour basis for a considered process. Considering proton–proton collisions with $\{u, d, s, c, b, g\}$ and the corresponding anti-quarks as initial state partons we can in principle have 121 different partonic initial states. Obviously for each process there exists an often *much smaller* basis of distinct subprocesses that combine individual channels that are identical up to the initial state PDFs, cf. Section 2.3.

In previous APPLgrid applications these subprocess bases have been identified manually, although a method for automated subprocess identification was sketched in [41]. Here we propose to use process information that is readily available from the event generators themselves, here in particular SHERPA.

At the level of the generation of the matrix elements for a given process individual channels get mapped onto each other. This is achieved by a one-to-one comparison of the transition amplitudes. This procedure guarantees an efficient re-use of matrix element expressions and significantly speeds-up cross section calculations. To give a simple example, the $c\bar{c}$ initiated contribution to inclusive QCD jet production in $pp$ or $p\bar{p}$ collisions is described by the very same matrix elements that account for the $u\bar{u}$ channel. As QCD interactions are flavour-diagonal the difference between these two channels originates from the initial state PDFs only.

We want to use this matrix-element generator internal identification and mapping of equivalent partonic channels to determine the reduced subprocess flavour basis. For SHERPA's matrix-element generators COMIX and AMEGIC the information on mapped channels is written out when the processes get generated for the first time. With MCgrid we now supply python scripts to directly convert these process maps into the lumi_pdf format used by APPLgrid. For that purpose the scripts analyse all channel maps and effectively collect a table of distinct subprocesses with all their contributing flavour channels. While these process maps are generated from SHERPA output, their validity is not limited to SHERPA, rather they can be used with every other event generator delivering events fully exclusive in the initial state flavours.

### 3.3.1. Statistical issues in subprocess identification

It should be noted that there are a number of subtleties involved when using the subprocess identification as described in 2.3. When filling grids with the subprocess identification enabled the cross section calculation applies the weights in the event sample to all the equivalent mapped channels in the same subprocess. As each subprocess can make use of more event weights than any of its component channels, the resulting total cross section enjoys an improved statistical accuracy, albeit mostly benefiting the partonic channels that weigh less in the total calculation. As an explicit example, compare the Monte Carlo sum in the two cases

$$
\sigma = \sum_{e} w_e(i, j) \, f_i f_j, \qquad \sigma_{\text{sub}} = \sum_{e} \sum_{l} w_e(i, j) \, C_{ij}^{(l)} F_l. \tag{24}
$$

While both estimators for the cross section are physically well motivated, it is clear that they differ when operating on a finite event sample. This difference makes a direct comparison of the subprocess-grouped result to the benchmark calculation (where no such grouping is performed) into a statistical exercise, with

differences naturally tending to zero in the limit of very large samples.

In addition to the (slight) statistical advantage available when using a subprocess grouping, the resulting APPLgrid files tend to be considerably smaller, as the number of contributing subprocesses for a particular reaction is typically much smaller than 121. This also impacts the final convolution time when making use of the produced APPLgrid files. Indeed, to be competitive in applications such as PDF fitting, such a grouping is almost a necessity.

Second, in exclusive event generation channels typically get selected with a weight proportional to their relative contribution to the total cross section. In this way a better convergence of the cross section estimate using a finite number of phase space points is achieved. These selection weights need to be accounted for by the respective event weights. Accordingly, the event weights for rather rare processes are significantly enhanced. To be precise, the sampled partonic cross section in Eq. (5) is complemented by a normalisation factor,

$$w_e(i, j, k_e) = \mathcal{N}_{ij} \, d\hat{\sigma}_{l_e \to X}(k_e) \Pi_{\mathrm{ps}}(k_e) \Theta(k_e - k_{\mathrm{cuts}}), \qquad (25)$$

where the normalisation factor

$$\mathcal{N}_{ij} \sim \frac{N_{\mathrm{tot}}}{N_{ij}} \qquad (26)$$

can be accurately approximated by the ratio of the total number of events to the number in the $ij$ channel.

These selection weights render a naive grouping of channels into subprocesses very inefficient, as channels with poor statistics but comparably large weights would dominate the statistical uncertainty of even a well populated subprocess. To avoid this, we need to determine the relative population of all partonic channels in the process to account for the selection weights. In this way, a channel specific event may be converted into a subprocess event while preserving the statistical accuracy of the overall calculation, i.e.

$$W_{\alpha\beta,\tau}^{(l)(p)(b)} = \frac{\mathcal{N}_l}{\mathcal{N}_{ij}} C_{ij}^{(l)} W_{\alpha\beta,\tau}^{(ij)(p)(b)}. \qquad (27)$$

The ratio $\mathcal{N}_l/\mathcal{N}_{ij}$ is determined numerically by monitoring the number of events falling into sub channel $ij$ relative to subprocess $l$. In MCgrid this is done in an initial loop over events, that must anyway be performed in order to determine phase space boundaries for the cross section grids, cf. Section 5.

Let us end by noting, that it is always possible to use the full flavour basis of 121 initial state combinations. In fact, it is only in this basis that we can guarantee to reproduce the input cross section within interpolation accuracy. However, the resulting grids will be significantly larger than corresponding grids produced with subprocess grouping enabled. In particular for applications such as PDF fitting subprocess identification is a must and with MCgrid we provide very efficient and fully automated methods to accomplish this for arbitrary processes.

## 4. Software implementation

In this section we shall briefly describe how the MCgrid tool may be implemented into a Rivet analysis in practice. Here a typical implementation will be broadly sketched; for the detailed technical documentation please refer to the user manual included in the package.

The MCgrid package is supplied as an external library which may be linked to a Rivet analysis at compile time. MCgrid may be configured and installed in the conventional way with the autotools build system. The package may be configured for two main fill modes; the default fill behaviour takes into account the PDF structure of event weights originating from the SHERPA event generator, cf. Section 3.2. For a generic fill mode where the PDF dependence is described fully by Eq. (16), an option is available in the configuration.

There are three main objects that are made available when linking an analysis to MCgrid:

- MCgrid::mcgrid_pdf
  This object inherits from the APPLgrid lumi_pdf class. In addition to the subprocess identification, it provides the required subprocess event counting functionality as described in Section 3.3. Initialised by MCgrid::bookPDF.
- MCgrid::PDFHandler
  This object is used to keep track of the initialised subprocess PDFs, and to pass events to them in the counting phase. Initialised at first use.
- MCgrid::gridPtr
  The primary object in the package, this provides a wrapper for an APPL::grid object. The class performs the conversion of an event generator weight to a suitable APPLgrid fill call. Provided as a
  boost::smart_ptr analogously to the Rivet histogram implementation. Initialised by MCgrid::bookGrid.

We shall now summarise the modifications required to each analysis phase in order to produce an APPLgrid file from MCgrid.

### 4.1. Initialisation phase

Initialising the MCgrid tools in an analysis is a matter of booking the subprocess PDF descriptions for the process in question, and allocating the interpolation grid classes.

This subprocess information is provided by APPLgrid lumi_pdf config files. For the details of how these files may be obtained from SHERPA or constructed by hand, refer to the documentation supplied with MCgrid. To initialise a subprocess config file in MCgrid the following method should be called in the initialisation phase for each process in the analysis:

```
MCgrid::bookPDF(configname, histoDir(),
                beam1Type, beam2Type);
```

where configname is an std::string providing the filename of the subprocess config name. histoDir() is a standard Rivet function which provides the name of the analysis. beam1Type and beam2Type specify whether the beam types used in the config file should be charge conjugated when performing a fill. This accounts for changing quarks to anti-quarks and vice versa in the case of an anti-proton beam.

With the subprocess PDFs initialised it is time to set up the interpolating grids themselves. First the Rivet analysis should be implemented and checked as in a standard analysis using only the histogram classes. Once the user is satisfied with the analysis, they should add to the analysis class their grid classes with the following method:

```
MCgrid::gridPtr MCgrid::bookGrid(
    // Corresponding Rivet histogram
    const Rivet::Histo1DPtr hist,
    // Result of Rivet histoDir() call
    const std::string histoDir,
    // APPLgrid subprocess PDF
    const std::string pdfname,
    // Leading order power of alpha_s for the process
    const int    LOpower,
    // Minimum value of parton x in the event sample
    const double xmin,
```

```
        // Maximum value of parton x in the event sample
        const double xmax,
        // Minimum event scale^2
        const double q2min,
        // Maximum event scale^2
        const double q2max,
        // Grid architecture
        const gridArch arch
    );
```

where the struct `gridArch` specifies the architecture of the APPLgrid interpolation. It can be initialised with the following constructor:

```
gridArch(
    const int nX,    // Number of points in x-grid
    const int nQ2,   // Number of points in Q^2 grid
    const int xOrd,  // Order of interpolation on x-grid
    const int Q2Ord  // Order of interpolation on Q^2-grid
):
```

As an example `init()` phase, consider the construction of a grid for a Drell–Yan $Z$-rapidity analysis where events are generated with a fixed scale of $M_Z^2$ from $p\bar{p}$ beams:

```
// Subprocess PDF
const string PDFname("DYppbar.config");
MCgrid::bookPDF(PDFname, histoDir(),
            MCgrid::BEAM_PROTON,
            MCgrid::BEAM_ANTIPROTON);

// Grid architecture
MCgrid::gridArch arch(50,1,5,0);

/// Book histograms and grids
_h_xsection = bookHisto1D(1, 1, 1);
_a_xsection = MCgrid::bookGrid( _h_xsection,
                                histoDir(), PDFname,
                                0,
                                1E-5, 1,
                                8315.18, 8315.18,
                                arch);
```

### 4.2. Analysis phase

In the analysis phase of the code, the first required modification is that the `MCgrid` event handler must be called for every event passed to `Rivet`. This is done by adding the following line to the very start of the analysis phase, before any selection cuts:

```
MCgrid::PDFHandler::HandleEvent(event);
```

Once the events have been counted, both the histograms and `APPLgrid` classes must be populated after the experimental cuts and analysis tools are applied as usual. Once the required event selection has been performed and the user is ready to fill a histogram, they simply have to fill the corresponding `gridPtr` also:

```
_h_distribution->fill(coord, weight);// Histogram fill
_a_distribution->fill(coord, event); // Grid fill
```

Here `coord` specifies the value of the binned quantity for that event, `weight` is the usual event weight and `event` is the `Rivet::Event` object passed to the `analyse` method.

### 4.2.1. Finalise phase

Finally the normalisation of the grids should be set, and the APPLgrid `.root` files exported for use. This is accomplished in the `finalise` phase of the analysis. For the normalisation the treatment of the grids is once again analogous to that of the histograms. For each histogram/grid pair to be normalised the following should be called:

```
// Histogram normalisation
scale(_h_distribution, normalisation);
// Grid normalisation
_a_distribution->scale(normalisation);
```

And finally the grids should be written to file:

```
_a_distribution->exportgrid();
```

The filename of the grid will be based automatically upon the id of the corresponding histogram.

As the last modification step in the finalise phase, the event handler must be cleared and exported by adding the following as the final line in the finalise phase:

```
MCgrid::PDFHandler::ClearHandler();
```

### 4.3. Executing a `MCgrid`/`Rivet` analysis

As is typical with the `APPLgrid` package, to fill its produced grids two runs of the analysis must be performed. The first, or phase space fill run, determines the relative statistics of each partonic channel in the process such that their statistical samples may be combined correctly, and also establishes the boundaries of the $x, Q^2$ phase space for each of the interpolation grids as explained in [41]. The second run actually populates the grids with the Monte Carlo weights. It is therefore typically sufficient to perform a run with a smaller but representative event sample for the phase space run, and only run the full event sample for the full fill. The modified `Rivet` analysis produced with `MCgrid` utilities can be used as a completely conventional `Rivet` analysis, running over HepMC event record files, or indeed streamed via a FIFO pipe or straight from an event generator. Once the event sample has been run through the modified analysis twice, a standard `APPLgrid` file will be produced.

## 5. `MCgrid` **Validation**

In order to validate the interface and describe some details and options available when implementing an analysis in `Rivet` and `MCgrid` we shall describe an application of the `MCgrid` tool to two processes at hadron colliders; inclusive jet production and the Drell–Yan production of $Z$ bosons.

In the first part of this section, we shall examine the results in the full 121 subprocess basis. In this way a direct comparison to the benchmark cross section is possible. In the second part we examine directly the production of grids with subprocess grouping enabled.

### 5.1. Interpolation accuracy and the ATLAS inclusive jet measurement

The first test to validate the interface must of course test the ability of `MCgrid` to generate a grid file that reproduces the

benchmark result from the direct analysis of the events with `Rivet`.

Having implemented the required additions as described in Section 4, we shall modify an existing `Rivet` analysis corresponding to the ATLAS 2010 inclusive jet measurement[49], double differential in the rapidity and $p_\perp$ of the hardest jet in the event. For the purposes of demonstration we shall consider only the lowest rapidity bin of the $R = 0.4$ measurement.

As in this section we aim to demonstrate the reproduction of the cross section up to the available interpolation accuracy, we shall make use of the ability of `MCgrid` to base multiple grids with different settings on the same `Rivet` histogram instance. Our grids are initialised in the `init` phase as so:

```
// Common grid properties
const string PDFname = "basic.config";
const double xmin    = 1E-5;
const double xmax    = 1;
const double Q2min   = 20;
const double Q2max   = 1E7;
const int LOpower    = 2;

// Book low precision interpolation grid
lowPrec = MCgrid::bookGrid(pTHisto_R04_y1, histoDir(),
                    PDFname, LOpower, xmin, xmax,
                    Q2min, Q2max,
                    MCgrid::gridArch(30,20,5,5)
                    );

// Book medium precision interpolation grid
medPrec = MCgrid::bookGrid(pTHisto_R04_y1, histoDir(),
                    PDFname, LOpower, xmin, xmax,
                    Q2min, Q2max,
                    MCgrid::gridArch(40,25,5,5)
                    );

// Book high precision interpolation grid
highPrec = MCgrid::bookGrid(pTHisto_R04_y1, histoDir(),
                    PDFname, LOpower, xmin, xmax,
                    Q2min, Q2max,
                    MCgrid::gridArch(50,30,5,5)
                    );
```

The three booked grids differ only by the content of their `gridArch` struct which determines their interpolation accuracy. These grids are then filled after experimental cuts as described in Section 4 alongside the corresponding histogram. Running this rivet analysis over a sample of NLO Dijet events generated by SHERPA+BlackHat for the phase space and fill runs, `MCgrid` outputs three APPLgrid ROOT format files.

As in this analysis we have used the `basic_pdf` subprocess PDF which contains all 121 partonic channels as independent subprocesses, a direct comparison to the original `Rivet` calculation is possible. In Fig. 1 we show the result of convoluting the three produced APPLgrid files, produced from 10 million events, with the generating PDF (in this case, the CT10 NLO set [50]) and taking the ratio to the benchmark `Rivet` result.

The results demonstrate that the `MCgrid` interface is able to convert event weights from the Monte Carlo sample into APPLgrid fill calls without any loss of accuracy at the precision of the APPLgrid interpolation result. In all three cases reproduction is at the permille level or better. As detailed above, the accuracy of the APPLgrid calculation may be tuned through the interface to the user's requirements. For a comprehensive discussion of the interpolation accuracy of the APPLgrid framework please refer to [41].

## 5.2. Parameter variation and the CDF Z rapidity measurement

Having verified that the `MCgrid` interface is able to reproduce the benchmark distribution available in SHERPA up to the
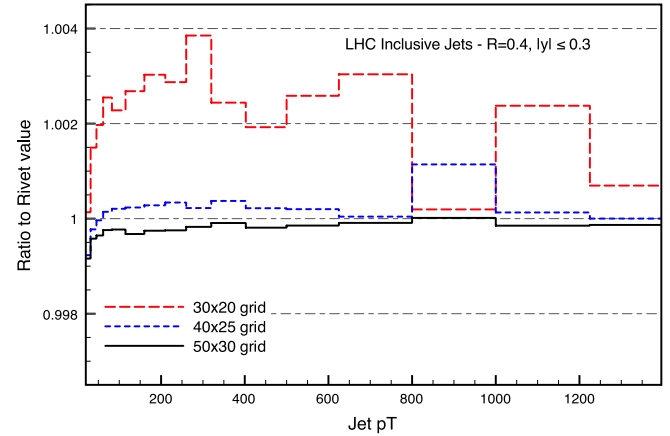


**Fig. 1.** Comparison of the reproduction accuracy for three interpolation grid choices for the ATLAS jet production analysis; see the text for details. The curves show the ratio of the APPLgrid result to the original `Rivet` calculation.

interpolation accuracy available in APPLgrid, we shall now verify the ability of the interface to actually reweight the produced event sample with different PDFs and perturbative scales. With this investigation we shall also demonstrate the interface with a different process at a different collider, modifying the existing `Rivet` analysis for the measurement of the $Z$ boson rapidity distribution at CDF[51]. To test our modifications we generated two Drell–Yan event samples at NLO in QCD with SHERPA, one with 10 million and one with 100 million events. The renormalisation and factorisation scales were fixed to $\mu_F^2 = \mu_R^2 = M_Z^2$. We ran both samples through the `MCgrid` enabled analysis. Additionally the event generation was repeated with the renormalisation and factorisation scales varied to $\mu_F^2 = \mu_R^2 = 2M_Z^2$ and $\mu_F^2 = \mu_R^2 = M_Z^2/2$ such that we can test the reproduction of these runs with the scale variation formula described in Ref. [41].

While the reproduction of the central scale distribution is limited only by interpolation accuracy, the reproduction of the scale variations does have an additional sensitivity to the statistical accuracy of the grid, as the accurate determination of the missing logarithmic terms in the fill weights depends on the interpolation grids being well populated. In particular the accurate reproduction of the scale variation uncertainties is a nontrivial test not only of the APPLgrid interpolation but also of the correct weight conversion and PDF dependence removal performed by `MCgrid`.

In Figs. 2 and 3 the agreement between the interpolated predictions and the benchmark result is demonstrated in the case of central values and scale variations for the two event samples. For both samples, the reproduction of the central value is excellent, with precision only limited by the interpolation accuracy. The reproduction of the scale variations is also very good, with accuracy typically much better than the percent level. The effect of increasing statistics is clear also in the improvement between the two samples for the varied scales. This provides a strong validation of the APPLgrid scale variation formula for Monte Carlo weights arising from a fully exclusive event generator.

The event generation has also been performed under a variation of the PDF used in the calculation. To verify the produced grids under their most typical application, PDF reweighting, the SHERPA/`Rivet` run was repeated with the use of MSTW2008 PDFs [52]. Using the original grid generated with the CT10 distributions, we perform the product with MSTW2008 to test if the produced grids are able to reweight PDFs effectively. The results as demonstrated in Fig. 4 show that the PDF dependence of the event weights has been properly removed, allowing for the correct reweighting when re-convoluted with a different PDF set.
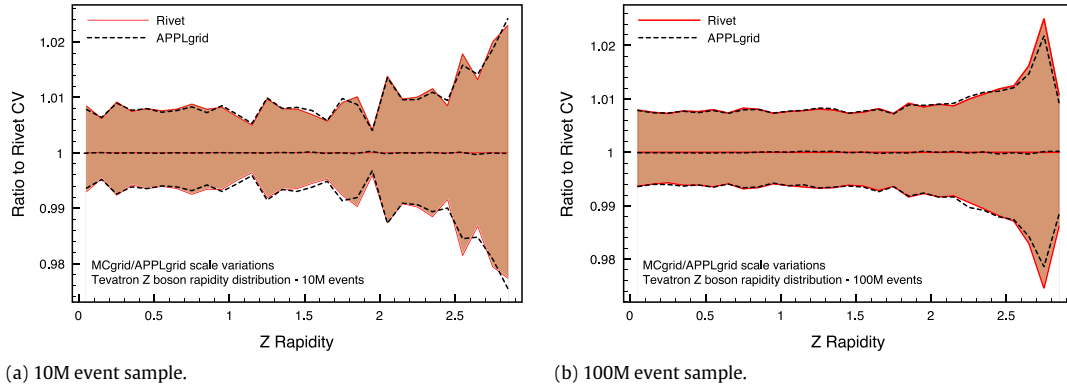
(a) 10M event sample.



(b) 100M event sample.

**Fig. 2.** Comparison of SHERPA/Rivet result to predictions from APPLgrid. Plots shown demonstrate the central value and scale variation reproduction for two event samples, with 10 million and 100 million events respectively. The red band represents the Rivet prediction and scale variation uncertainty, and the dotted black line the APPLgrid prediction. All points are normalised to the Rivet central value.
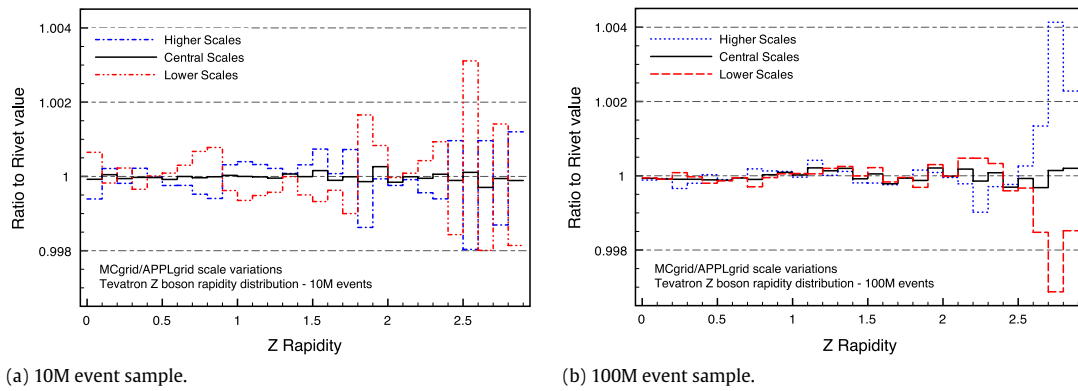


(a) 10M event sample.



(b) 100M event sample.

**Fig. 3.** Comparison of SHERPA/Rivet result to predictions from APPLgrid. Plots shown demonstrate the central value and scale variation reproduction for two event samples, with 10 million and 100 million events respectively. Each histogram represents the ratio of the APPLgrid result to its equivalent result from SHERPA/Rivet.
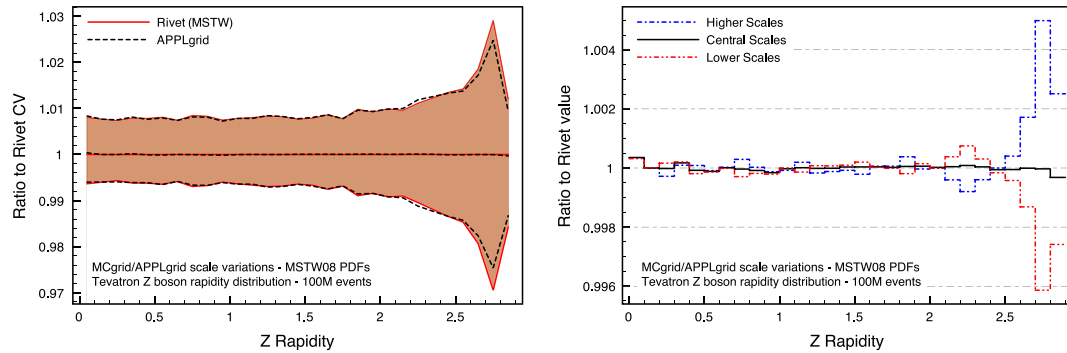




**Fig. 4.** Comparison of SHERPA/Rivet result using MSTW08 PDFs with predictions from an APPLgrid generated with CT10 and convoluted with MSTW08 PDFs. Plots shown demonstrate the central value and scale variation reproduction for the 100 million event sample grid. The red band in the left panel represents the Rivet prediction and scale variation uncertainty, and the dotted black line the APPLgrid prediction. All points are normalised to the Rivet (MSTW) central value.

### 5.3. Grid fills with subprocess identification

As described in Section 3.3.1, the use of a subprocess basis for the incoming PDFs makes a direct comparison to the Rivet result more complicated, as the statistical accuracy of the result is modified. In addition, the MCgrid interface also must perform a tracking of the relative statistical population of the individual partonic channel contributions to a subprocess, such that their selection weights may be corrected to the subprocess' weight.

To test the MCgrid implementation of this tracking, and its generation of subprocess-identified APPLgrid files, we performed the 100M Drell–Yan fill as described in Section 5.2 but this time

utilising the lumi_pdf config file generated by the packaged scripts. In Fig. 5 the ratio of the APPLgrid result to the benchmark distribution is shown, with the basic_pdf result from Section 5.2 for comparison. It is important to note that the reduced agreement between the subprocess APPLgrid and the Rivet benchmark does not imply that the subprocess result is less accurate, rather that here we are not directly comparing like with like.

### 5.4. Demonstration of APPLgrid convolutions

Finally, to provide a demonstration of the performance of the APPLgrid interpolation methods and illustrate the potential
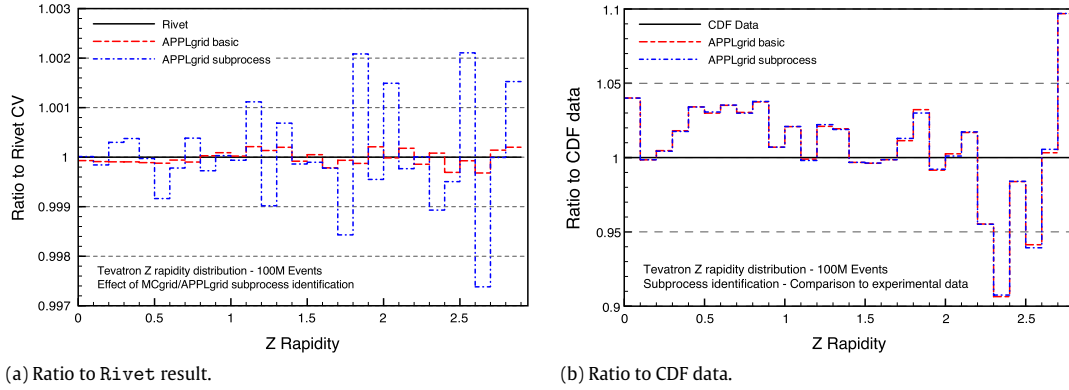
(a) Ratio to `Rivet` result.

(b) Ratio to CDF data.

**Fig. 5.** Illustration of the typical differences expected between produced grid files utilising either the `basic` set of subprocesses or with the subprocesses identified through the scripts included in the `MCgrid` package. The left plot shows the deviation from the benchmark `Rivet` result, and the right demonstrates that the deviation does not correspond to a reduced level of agreement with experimental data.
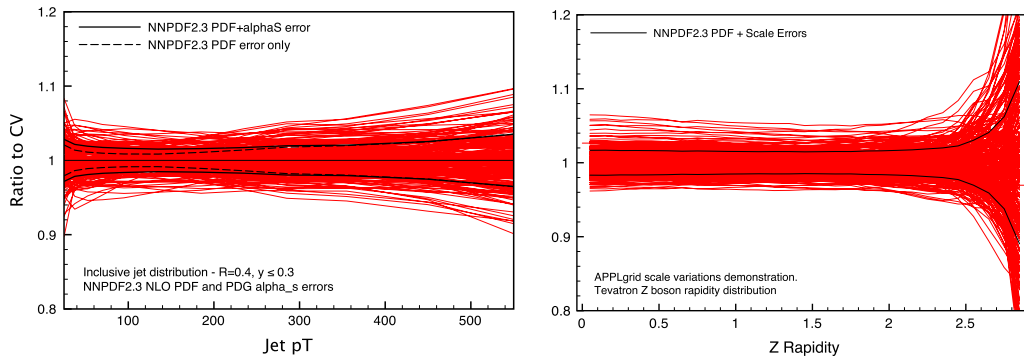


**Fig. 6.** Example application of the `MCgrid` produced `APPLgrid` files. The left figure shows the NNPDF2.3 replica distribution for an inclusive jet $p_\perp$ distribution at the LHC, including the error on $\alpha_s$ via the replica distribution. The figure on the right shows predictions for the CDF Z rapidity measurement, with replica distributions for the central scale, and variations. On each plot, the red lines show individual NNPDF2.3 replicas, the black lines denoting the 1-$\sigma$ contours.

applications of grids produced via the `MCgrid` interface, we shall examine two cases where a large number of repeat calculations must be performed.

Using the grid files produced in Section 5 for the validation of the interface, the uncertainty from perturbative scale variations and $\alpha_s$ may be assessed in a nonlinear fashion by using the representation of probability distributions in the space of PDFs available in NNPDF 2.3 [53].

Taking the PDG reference value of $\alpha_s = 0.1184(7)$ [54], it is possible to generate a distribution of PDFs for this value including the uncertainty by sampling appropriately from the available NNPDF2.3 $\alpha_s$ sets. Assuming a Gaussian uncertainty around $\alpha_s = 0.1184$ and normalising to the maximum number of PDF replicas available in each set, this corresponds to taking 16 replicas at $\alpha_s = 0.117$, 100 at $\alpha_s = 0.118$, 82 at $\alpha_s = 0.119$ and finally 9 replicas at $\alpha_s = 0.120$. Using this total sample of 207 PDF replicas, we plot the combined PDF and $\alpha_s$ uncertainty for the inclusive jet $p_\perp$ distribution in Fig. 6.

Going to an even larger basis of PDF replicas, in the right of Fig. 6 we show the replica distribution for the NNPDF2.3 CDF Z rapidity prediction including scale variation uncertainties. In this instance there are a total of 300 predictions computed and plotted.

Plotting distributions such as these, with very large numbers of predictions leading to accurate assessments of the underlying uncertainties, is only feasible with the use of interpolating tools such as `APPLgrid`. The `MCgrid` interface now greatly increases the number of processes available for such an interpolation.

## 6. Conclusions

In this paper we have presented `MCgrid`, a package for producing `APPLgrid` tables from samples of events produced by Monte Carlo generators. These tables are based on interpolating functions that allow for precise, fast, and flexible computations of scale variations and PDF reweighting. In this way, the storage requirements and the speed of these calculations are greatly improved.

`MCgrid` provides additional methods to be used in conjunction with the `Rivet` programme. In the analysis of a Monte Carlo calculation of a fixed order process it allows for the production of an `APPLgrid` for every considered observable.

The basic idea follows the general `APPLgrid` prescription, whereby grids are computed using some interpolating functions for the PDFs, and separating explicitly the dependence on the perturbative order, the renormalisation and factorisation scales as discussed in Section 3. Note that these interpolation tables are computed by summing over the generated events, for a specific choice of the kinematical variables. The differential cross section is then obtained by contracting these tables with the values of the PDFs at the points chosen in the interpolation grids. The choice of the interpolation grids determines the accuracy of the interpolation tool. Once again, the structure of the interpolating grids has to be decided in advance, and cannot be changed once the tables have been produced.

Using the interpolation tables at LO in perturbation theory is a straightforward exercise. The extension of the tool at NLO is

more subtle, because the precise PDF dependence of the integrated subtraction terms must be taken properly into account. We have detailed our NLO implementation in Section 3.

The details of the software implementation have been presented in Section 4.

The interface has been validated by studying two processes, namely the inclusive jet production at the LHC, and the Drell–Yan production of $Z$ bosons at the Tevatron. As expected, the grids can be tuned to reach an excellent accuracy of order $10^{-3}$ for the computation of the observables. Using the grids we provide an explicit example of the parameter variations mentioned above.

`MCgrid` enables faster studies of scale variations, and PDF variations, without having to perform multiple runs of the Monte Carlo generators. This allows for the determination of reliable uncertainty estimates for arbitrary observables even for very complicated and computationally challenging multi-particle final state calculations. It also provides a solution to the large storage requirements that are necessary for other methods relying on storing explicit events [40]. Because of the increased performance in computing observables, `MCgrid` paves the way for the inclusion of more observables in modern PDF fits. In its present implementation, the full PDF dependence of showered and hadronised events is not fully accounted for. This would require tracing the PDF dependence of the parton shower history of individual events, which is beyond the scope of this publication. While `MCgrid` is able to process such events into the `APPLgrid` format, there is an implicit approximation present in that the reweighting is only performed at the level of the hard process. Therefore full NLO accuracy can only be claimed for fixed order calculations.

`MCgrid` is publicly available and can be downloaded from http://mcgrid.hepforge.org. We make use of the HepMC event record, requiring some additional event information being stored in the `HepMC::WeightContainer`. For the SHERPA event generator, as of version `2.0.0`, this information is provided by default with the `HepMC_Short` output format.

## Acknowledgements

## References

[1] A. Buckley, J. Butterworth, S. Gieseke, D. Grellscheid, S. Höche, H. Hoeth, F. Krauss, L. Lönnblad, et al., Phys. Rep. 504 (2011) 145. arXiv:1101.2599 [hep-ph].
[2] A. Denner, S. Dittmaier, Nuclear Phys. B 734 (2006) 62. [hep-ph/0509141].
[3] G. Ossola, C.G. Papadopoulos, R. Pittau, Nuclear Phys. B 763 (2007) 147. [hep-ph/0609007].
[4] C. Anastasiou, R. Britto, B. Feng, Z. Kunszt, P. Mastrolia, Phys. Lett. B 645 (2007) 213. [hep-ph/0609191].
[5] R.K. Ellis, W.T. Giele, Z. Kunszt, J. High Energy Phys. 0803 (2008) 003. arXiv:0708.2398 [hep-ph].
[6] W.T. Giele, Z. Kunszt, K. Melnikov, J. High Energy Phys. 0804 (2008) 049. arXiv:0801.2237 [hep-ph].
[7] C.F. Berger, Z. Bern, L.J. Dixon, F. Febres Cordero, D. Forde, H. Ita, D.A. Kosower, D. Maitre, Phys. Rev. D 78 (2008) 036003. arXiv:0803.4180 [hep-ph].
[8] G. Bevilacqua, M. Czakon, M.V. Garzelli, A. van Hameren, A. Kardos, C.G. Papadopoulos, R. Pittau, M. Worek, Comput. Phys. Comm. 184 (2013) 986. arXiv:1110.1499 [hep-ph].
[9] J. Alwall, M. Herquet, F. Maltoni, O. Mattelaer, T. Stelzer, J. High Energy Phys. 1106 (2011) 128. arXiv:1106.0522 [hep-ph].
[10] T. Gleisberg, S. Höche, F. Krauss, A. Schälicke, S. Schumann, J.-C. Winter, J. High Energy Phys. 0402 (2004) 056. [hep-ph/0311263].
[11] T. Gleisberg, S. Höche, F. Krauss, M. Schönherr, S. Schumann, F. Siegert, J. Winter, J. High Energy Phys. 0902 (2009) 007. arXiv:0811.4622 [hep-ph].
[12] S. Catani, M.H. Seymour, Nuclear Phys. B 485 (1997) 291. [Erratum-S. Catani, M.H. Seymour, Nuclear Phys. B 510 (1998) 503] [hep-ph/9605323].
[13] S. Catani, S. Dittmaier, M.H. Seymour, Z. Trocsanyi, Nuclear Phys. B 627 (2002) 189. [hep-ph/0201036].
[14] S. Frixione, Z. Kunszt, A. Signer, Nuclear Phys. B 467 (1996) 399. [hep-ph/9512328].
[15] T. Gleisberg, F. Krauss, Eur. Phys. J. C 53 (2008) 501. arXiv:0709.2881 [hep-ph].
[16] M. Czakon, C.G. Papadopoulos, M. Worek, J. High Energy Phys. 0908 (2009) 085. arXiv:0905.0883 [hep-ph].
[17] K. Hasegawa, S. Moch, P. Uwer, Comput. Phys. Comm. 181 (2010) 1802. arXiv:0911.4371 [hep-ph].
[18] R. Frederix, T. Gehrmann, N. Greiner, J. High Energy Phys. 1006 (2010) 086. arXiv:1004.2905 [hep-ph].
[19] S. Plätzer, S. Gieseke, Eur. Phys. J. C 72 (2012) 2187. arXiv:1109.6256 [hep-ph].
[20] G. Bevilacqua, M. Czakon, M. Kubocz, M. Worek, J. High Energy Phys. 1310 (2013) 204. arXiv:1308.5605 [hep-ph].
[21] F. Krauss, R. Kuhn, G. Soff, J. High Energy Phys. 0202 (2002) 044. [hep-ph/0109036].
[22] T. Gleisberg, S. Höche, J. High Energy Phys. 0812 (2008) 039. arXiv:0808.3674 [hep-ph].
[23] A. Cafarella, C.G. Papadopoulos, M. Worek, Comput. Phys. Comm. 180 (2009) 1941. arXiv:0710.2427 [hep-ph].
[24] G. Cullen, N. Greiner, G. Heinrich, G. Luisoni, P. Mastrolia, G. Ossola, T. Reiter, F. Tramontano, Eur. Phys. J. C 72 (2012) 1889. arXiv:1111.2034 [hep-ph].
[25] A. van Hameren, C.G. Papadopoulos, R. Pittau, J. High Energy Phys. 0909 (2009) 106. arXiv:0903.4665 [hep-ph].
[26] V. Hirschi, R. Frederix, S. Frixione, M.V. Garzelli, F. Maltoni, R. Pittau, J. High Energy Phys. 1105 (2011) 044. arXiv:1103.0621 [hep-ph].
[27] S. Badger, B. Biedermann, P. Uwer, V. Yundin, Comput. Phys. Comm. 184 (2013) 1981. arXiv:1209.0100 [hep-ph].
[28] F. Cascioli, P. Maierhofer, S. Pozzorini, Phys. Rev. Lett. 108 (2012) 111601. arXiv:1111.5206 [hep-ph].
[29] S. Actis, A. Denner, L. Hofer, A. Scharf, S. Uccirati, J. High Energy Phys. 1304 (2013) 037. arXiv:1211.6316 [hep-ph].
[30] T. Binoth, F. Boudjema, G. Dissertori, A. Lazopoulos, A. Denner, S. Dittmaier, R. Frederix, N. Greiner, et al., Comput. Phys. Comm. 181 (2010) 1612. arXiv:1001.1307 [hep-ph].
[31] S. Alioli, S. Badger, J. Bellm, B. Biedermann, F. Boudjema, G. Cullen, A. Denner, H. van Deurzen, et al. arXiv:1308.3462 [hep-ph].
[32] C.F. Berger, Z. Bern, L.J. Dixon, F. Febres Cordero, D. Forde, T. Gleisberg, H. Ita, D.A. Kosower, et al., Phys. Rev. Lett. 106 (2011) 092001. arXiv:1009.2338 [hep-ph].
[33] Z. Bern, L.J. Dixon, F. Febres Cordero, S. Höche, H. Ita, D.A. Kosower, D. Maitre, K.J. Ozeren, Phys. Rev. D 88 (2013) 014025. arXiv:1304.1253 [hep-ph].
[34] H. Ita, Z. Bern, L.J. Dixon, F. Febres Cordero, D.A. Kosower, D. Maitre, Phys. Rev. D 85 (2012) 031501. arXiv:1108.2229 [hep-ph].
[35] Z. Bern, G. Diana, L.J. Dixon, F. Febres Cordero, S. Höche, D.A. Kosower, H. Ita, D. Maitre, et al., Phys. Rev. Lett. 109 (2012) 042001. arXiv:1112.3940 [hep-ph].
[36] S. Badger, B. Biedermann, P. Uwer, V. Yundin, Phys. Lett. B 718 (2013) 965. arXiv:1209.0098 [hep-ph].
[37] S. Badger, B. Biedermann, P. Uwer, V. Yundin, arXiv:1309.6585 [hep-ph].
[38] G. Bevilacqua, M. Czakon, C.G. Papadopoulos, M. Worek, Phys. Rev. Lett. 104 (2010) 162002. arXiv:1002.4009 [hep-ph].
[39] T. Gehrmann, N. Greiner, G. Heinrich, arXiv:1308.3660 [hep-ph].
[40] Z. Bern, L.J. Dixon, F.F. Cordero, S. Höche, H. Ita, D.A. Kosower, D. Maitre, arXiv:1310.7439 [hep-ph].
[41] T. Carli, D. Clements, A. Cooper-Sarkar, C. Gwenlan, G.P. Salam, F. Siegert, P. Starovoitov, M. Sutton, Eur. Phys. J. C 66 (2010) 503. arXiv:0911.2985 [hep-ph].
[42] T. Kluge, K. Rabbertz, M. Wobisch, hep-ph/0609285.
[43] M. Wobisch, et al. [fastNLO Collaboration], arXiv:1109.1310 [hep-ph].
[44] A. Buckley, J. Butterworth, L. Lönnblad, H. Hoeth, J. Monk, H. Schulz, J.E. von Seggern, F. Siegert, et al. arXiv:1003.0694 [hep-ph].
[45] G.P. Salam, J. Rojo, Comput. Phys. Comm. 180 (2009) 120. arXiv:0804.3755 [hep-ph].
[46] http://mcfm.fnal.gov/.
[47] Z. Nagy, Phys. Rev. D 68 (2003) 094002. [hep-ph/0307268].
[48] M. Dobbs, J.B. Hansen, Comput. Phys. Comm. 134 (2001) 41. see also: http://lcgapp.cern.ch/project/simu/HepMC/.
[49] G. Aad, et al., [ATLAS Collaboration], Phys. Rev. D 86 (2012) 014022. arXiv:1112.6297 [hep-ex].
[50] H.-L. Lai, M. Guzzi, J. Huston, Z. Li, P.M. Nadolsky, J. Pumplin, C.-P. Yuan, Phys. Rev. D 82 (2010) 074024. arXiv:1007.2241 [hep-ph].
[51] T.A. Aaltonen, et al., [CDF Collaboration], Phys. Lett. B 692 (2010) 232. arXiv:0908.3914 [hep-ex], arXiv:0908.3914 [hep-ex].
[52] A.D. Martin, W.J. Stirling, R.S. Thorne, G. Watt, Eur. Phys. J. C 63 (2009) 189. arXiv:0901.0002 [hep-ph].
[53] R.D. Ball, V. Bertone, S. Carrazza, C.S. Deans, L. Del Debbio, S. Forte, A. Guffanti, N.P. Hartland, et al., Nuclear Phys. B 867 (2013) 244. arXiv:1207.1303 [hep-ph].
[54] J. Beringer, et al., [Particle Data Group Collaboration], Phys. Rev. D 86 (2012) 010001.