Simplified fast detector simulation in MADANALYSIS 5

Jack Y. Araz¹,²a, Benjamin Fuks³,⁴b and Georgios Polykratis³c

¹ School of Physics & Astronomy, University of Glasgow, United Kingdom
² Concordia University 7141 Sherbrooke St. West, Montréal, QC, Canada H4B 1R6
³ Laboratoire de Physique Théorique et Hautes Energies (LPTHE), UMR 7589, Sorbonne Université et CNRS, 4 place Jussieu, 75252 Paris Cedex 05, France
⁴ Institut Universitaire de France, 103 boulevard Saint-Michel, 75005 Paris, France

Abstract. We introduce a new simplified fast detector simulator in the MADANALYSIS 5 platform. The Python-like interpreter of the programme has been augmented by new commands allowing for a detector parametrisation through smearing and efficiency functions. On run time, an associated C++ code is automatically generated and executed to produce reconstructed-level events. In addition, we have extended the MADANALYSIS 5 recasting infrastructure to support our detector emulator and provide predefined LHC detector configurations. We have compared predictions obtained with our approach to those from the DELPHES 3 software, both for Standard Model processes and a few new physics signals. Results generally agree to a level of about 10% or better, although DELPHES 3 is 30% to 50% slower and requires 100 times more disk space.

1 Introduction

The discovery of the last missing particle of the Standard Model at the Large Hadron Collider (LHC) at CERN has opened a new era in our understanding of the fundamental laws of nature. However, the concrete mechanism behind electroweak symmetry breaking is still today a mystery and there is no sign of phenomenon beyond the Standard, despite the wealth of data currently available. As a consequence, the experimental LHC search results are interpreted as stronger and stronger constraints on a large set of new physics models, those constraints being obtained by comparing associated predictions with data.

Those reinterpretations can be achieved in two ways. First, they can rely on experimental data from which the detector effects have been unfolded, i.e. by considering LHC data as observed by a perfect detector with an infinite resolution and an ideal calibration. This requires an excellent understanding of the background and is a complex, ill-defined and time-consuming problem, as there is no unique solution to the inversion of the convolution of the detector response [1]. Consequently, reinterpretations are usually performed by adopting a second approach in which the detector effects are included, or folded forward, in the simulation of the new physics signals to be confronted to data. This folding is expected to appropriately capture the impact of the inner working of the detector and the inefficiency of the reconstruction of the event record. This has the advantage of being computationally much more acceptable.

The most accurate forward folding method relies on the implementation of the exact details of the detector functioning in a framework based on the GEANT 4 package [2]. The latter allows for the modelling of the detector geometry and material interactions, from which one could then deal with the simulation of the electronic response of the detector. As a final step, it is required to reproduce the impact of the reconstruction details associated with any given experiment, which yields appropriate definitions for the various physics objects used in an analysis. This is, however, in practice not achievable by anyone outside the collaborations, by virtue of the lack of publicly available information. In addition, even if feasible, running such a chain of tools for the plethora of new physics model potentially interesting is likely to be computationally unfeasible, as this requires several minutes of computing time for a single simulated event.

The resolution, reconstruction and identification efficiencies corresponding to all final-state objects relevant for a physics analysis are however often publicly available under the form of functions of standard object properties (like the transverse momentum or pseudo-rapidity). This, therefore, opens the door to a well-motivated and computationally much more efficient approach, as is implemented in packages like the DELPHES 3 software [3] or the RIVET toolkit [4].

In DELPHES 3, the simulation of the detector approximates the steps followed in a GEANT-based approach. It relies on approximate experiment geometries and particle propagation models, that are combined with tabu-
lated reconstruction and identification efficiencies to yield the reconstructed physics objects to use in an analysis. Moreover, the programme allows for the optional simulation of the pile-up. In Rivet, a lighter approach is implemented [5]. Effective transfer functions, including the smearing of the object kinematics properties and reconstruction efficiencies, are used to connect the Monte Carlo representation of any physics object (i.e. at the truth level) to their reconstructed representation.

In this paper, we report on the design of an extension of the capabilities of the MadAnalysis 5 framework [6–8] to handle the simulation of the response of a detector. MadAnalysis 5 is a general platform for beyond the Standard Model phenomenology. It can deal both with the development of an analysis of any given collider signal (together with its associated background) via a user-friendly Python-based command-line interface and a developer-friendly C++ core. MadAnalysis 5 can also be used for the (automated) reinterpretation of existing LHC results [9,10].

The response of a typical detector can be emulated in MadAnalysis 5 in two ways. The user has first the option to rely on the interface of the code to the FastJet package [11]. Whilst fast and efficient, this way of proceeding is restricted to the simulation of a detector with an infinite resolution. It indeed leads to the sole application of a jet clustering algorithm (as available from FastJet) to reconstruct the event, possibly together with flat parametrisations to model some specific reconstruction effects like b-tagging or tau-tagging. Nevertheless, this procedure is useful for studies dedicated to a particular effect or at the Monte Carlo truth level.

As a simple application of a jet algorithm is known to be quite unrealistic in many contexts for which detector effects matter, MadAnalysis 5 has been interfaced to the Delphes 3 package for a more detailed simulation of the detector effects. This interface drives the run of Delphes 3, and additionally switches on the usage of new modules that are specific to MadAnalysis 5. These are for instance related to a better handling of object isolation or the shrinking of the output Root file. The price to pay to employ Delphes 3 instead of simply FastJet is obviously a much slower run of the code, as stemming from the larger complexity of Delphes 3 with respect to FastJet.

In the following, we detail several improvements that have been added to the MadAnalysis 5 interface to FastJet. These allow the user to include a lightweight, therefore computationally cheap, and realistic detector simulation when Monte Carlo event reconstruction (with FastJet) is in order.

Thanks to those developments, MadAnalysis 5 offers now the flexibility to rely on any specific set of transfer functions to effectively map the Monte Carlo objects to their reconstructed counterparts (i.e. reconstruction efficiencies, tagging efficiencies and the corresponding mistagging rates). Moreover, the properties of the different objects can be smeared to mimic detector resolution degradations. Our changes in the C++ core of the programme have been combined with new functionalities at the level of the Python command-line interface, so any user can straightforwardly implement his/her own detector parametrisation in a simple and user-friendly manner. In addition, the code is shipped with predefined sets of commands allowing to automatically load detector configurations associated with the ATLAS and CMS detectors.

The new features introduced in this work are available from version 1.8.51 of the code, that can be downloaded from LaunchPad1. As said above, this upcoming release of the programme allows for the simulation of the impact of a detector in a way that combines realism and efficiency. In addition, it allows for dedicated studies singling out specific detector effects, that could potentially be applied directly at the particle level. The information included in this paper is summarised on the MadAnalysis 5 website2.

The rest of this paper has been prepared as follows. In section 2, we briefly review how to reconstruct a sample of Monte Carlo events in MadAnalysis 5 with the help of its interface to FastJet. In section 3, we detail the new features that have been developed in order to allow the user to combine event reconstruction with a fast and realistic detector simulation. In section 4, we compare predictions obtained with our new method with those arising from the usage of Delphes 3. We moreover quantitatively assess the differences between the results returned by both detector simulators relatively to the Monte Carlo truth. We then present, in section 5, how the new MadAnalysis 5 fast and simplified detector simulator can be used for the reinterpretation of the results of two specific LHC analyses. In this context, a comparison with a more usual method relying on Delphes 3 is performed. We summarise our work and conclude in section 6.

2 Event reconstruction with MadAnalysis 5

2.1 Running FastJet from MadAnalysis 5

Thanks to its interface to FastJet, MadAnalysis 5 allows for the reconstruction of hadron-level Monte Carlo events through the application of a jet-clustering algorithm. To proceed, the programme has to be started in the reconstructed mode, ./bin/ma5 -R
and the FastJet package has to be present on the user system. If this is not the case, it is sufficient to type, in the MadAnalysis 5 command-line interface,

install fastjet
to trigger a local installation of FastJet, in the subfolder tools/fastjet.

The MadAnalysis 5 run has then to be configured following the specifications of the user. First, one must switch event reconstruction on by typing

1 See the webpage https://launchpad.net/madanalysis5.
2 See the webpage http://madanalysis.irmp.ucl.ac.be/wiki/SFS.
set main.fastsim.package = fastjet

This turns on the usage of FASTJET for event reconstruction and indicates to the code that the input hadron-level event sample(s) are encoded following the HepMC event file format [12].

By default, jets are reconstructed by making use of the anti-$k_t$ algorithm [13], with a radius parameter set to $R = 1$. This behaviour can be modified by typing in

set main.fastsim.algorithm = <algo>
set main.fastsim.<property> = <value>

where <algo> represents the keyword associated with the adopted jet algorithm, and <property> generically denotes any of its property. MadAnalysis 5 can employ the longitudinally invariant $k_t$ algorithm [14, 15] ($kt$), the Cambridge/Aachen algorithm [16, 17] ($cambridge$), the anti-$k_t$ algorithm ($antikt$), the generalised $k_t$ algorithm [11] ($genkt$), as well as the seedless infrared-safe cone algorithm [18] ($siscone$). We refer to table 1 for the list of available jet algorithms, the corresponding options and their default values.

Among all the jet algorithm properties that can be tuned, three of them are common to all algorithms. The user can define a transverse momentum threshold ($ptmin$) so that any softer jet is filtered out, and fix the jet radius $R$ ($radius$) that dictates how distant the constituents of a given jet can be. Moreover, he/she can decide whether the algorithm should be exclusive or inclusive ($exclusive_id$), i.e. whether the leptons and photons originating from hadron decays have to be included in their respective collections in addition to be considered as constituents of the reconstructed jets ($exclusive_id = false$), or not ($exclusive_id = true$).

Furthermore, the generalised $k_T$ algorithm involves a distance measure depending on a continuous parameter $p$ ($p$), and the cone algorithm depends on the fraction of overlapping momentum above which two protojets are combined ($overlap$), on the maximum number of passes the algorithm should be carried out ($npassmax$), and on a threshold driving the removal of too soft objects ($input_ptmin$). Those protojets, on which jet reconstruction relies in general, are either final-state hadrons or objects that have been already combined. For all algorithms, the combination process obeys to the $E$-scheme [11], i.e. the momentum of the combined object equals the sum of the initial momenta.

In practice, the code starts by filtering the input particles (as provided in the event record), restraining the analyses list to all visible final-state particles. This corresponds to the entire set of objects not explicitly tagged as invisible. The tagging of any object as invisible can be achieved by typing in

define invisible = invisible <pdg-code>

where the $<pdg-code>$ value corresponds to the Particle Data Group (PDG) identifier [19] of a new invisible state. The above command results in adding this new code to the list of invisible particles stored in the invisible container, which includes by default the Standard Model neutrinos and antineutrinos, as well as the supersymmetric lightest neutralino and gravitino.

Similarly, the user can inform the code about the existence of a new strongly-interacting particle, which hence participates to the hadronic activity in the event and has to be accounted for by the clustering algorithm. The information is provided by superseding the definition of the hadronic container.

define hadronic = hadronic <new-pdg-code>

After having defined the characteristics of the reconstruction, event files have to be imported, either one by one or simultaneously by using wildcards. This is achieved through the standard command

import <path-to-hepmc-files> as <set>

where the user-defined label $<set>$ allows one to group several event files (assumed to describe the same physics process) into a single set. This line can be repeated as much as needed. The reconstructed events are saved on disk and stored in an event file encoded in the LHE event format [20,21] by typing, in the interpreter,

set main.outputfile = <name-of-an-LHE-file>

where $<name-of-an-LHE-file>$ is a filename carrying an .lhe or .lhe.gz extension. This file will be created in the $<dir>/Output/SAF/<set>/lheEvents0_0$ directory during the MadAnalysis 5 run ($<dir>$ denoting the run working directory).

The clustering is finally started after the

| Algorithm  | ptmin | exclusive_id | radius | p | overlap | npassmax | input_ptmin |
|------------|-------|--------------|--------|---|---------|----------|-------------|
| $kt$       | 5     | true         | 1      | - | -       | -        | -           |
| $cambridge$| 5     | true         | 1      | - | -       | -        | -           |
| $antikt$   | 5     | true         | 1      | - | -       | -        | -           |
| $genkt$    | 5     | true         | 1      | - | -       | -        | -           |
| $siscone$  | 5     | true         | 1      | 1 | -       | -        | -           |

Table 1. Jet algorithms available in MadAnalysis 5, shown together with the properties that can be changed by the user on run time. The default values are indicated.

3 The deprecated STDHEP format is still supported in MadAnalysis 5 and can therefore possibly be used as well.

4 While other cone algorithms are interfaced, they are not infrared-safe and will therefore not be further discussed. We refer to ref. [8] for more information.
The platform first allows for the implementation of a simple b-tagging procedure, relying on flat efficiencies and mistagging efficiencies. The decision behind the (mis)tagging of any given jet involves the angular distance between the reconstructed object and a true B-hadron (as read from the input event). Each property defining this procedure is entered in the interpreter as follows,

\[ \text{set main.fastsim.bjet_id.<property> = <value>} \]

the list and default values being given in table 2. Behind the scenes, the algorithm matches each B-hadron present in the analysed event to all reconstructed jets lying at an angular distance smaller than a threshold fixed by the user (\texttt{matching_dr}). Those jets are then considered as b-tagged with a probability entered by the user (\texttt{efficiency}). b-jet identification can be restricted to the closest jet (\texttt{exclusive = true}), or to all matched jets (\texttt{exclusive = false}). The mistagging of charm and light jets as b-jets is performed similarly, the decision being taken following flat probabilities that are provided by the user (\texttt{misid_cjet} and \texttt{misid_ljet} respectively).

Next, a simple tau identification procedure can be employed, its properties being set by typing, in the interpreter,

\[ \text{set main.fastsim.tau_id.<property> = <value>} \]

The list of available options, together with their default values, is given in table 3. The user has the possibility to fix the probability with which a jet originating from the hadronic decay of a tau lepton will be correctly tagged as a hadronic tau object (\texttt{efficiency}), as well as the probability with which a light jet will be incorrectly tagged as a hadronic tau (\texttt{misid_ljet}).

### 3 A simplified and realistic fast detector simulator

#### 3.1 Generalities

In the present work, we have improved the way in which detector effects can be included when the interface of \texttt{MadAnalysis 5} with \texttt{FastJet} is used for event reconstruction. Our modifications allow for the post-processing of the \texttt{FastJet} output to model detector effects on the basis of transfer functions. Those functions are provided by the user directly in the \texttt{Python} interpreter and could depend on various object kinematical properties. When the C++ code is generated, the transfer functions are converted into a new C++ module that is called at the end of the \texttt{FastJet} run. This enables the modification of the properties of the reconstructed objects according to various experimental biases. In contrast to relying on a complex programme like Delphes 3 that acts at the hadronic level, our setup acts at the time of the reconstruction, which results in a faster event reconstruction process and lighter output files.

We consider three classes of effects. Firstly, the kinematical properties of any given reconstructed objects could be smeared to account for the detector resolution. Secondly, each object has a given probability of being effectively reconstructed, depending on its kinematics. Finally, object identification can be more or less successful, depending again on the kinematics, and leads to a potential misidentification. Whilst the latter effects could already be accounted for in the previous version of the code (see section 2.2), it was not possible to include any dependence on the object kinematics. This limitation has been alleviated.

In order to handle functions at the level of the command-line interface, we rely on abstract syntax trees to decode the information provided by the user and store it internally. Those trees can then be converted into C++ (or any other programming language) and merged with the code generated by the \texttt{MadAnalysis 5} interpreter, that is then compiled and run on the input events. This allows for a very flexible definition of any transfer function. The latter is allowed to depend on any of the observables supported by \texttt{MadAnalysis 5}, the complete list of them being available from the manual [6] or the normal mode.

| Properties | Default value | type   |
|------------|---------------|--------|
| matching_dr | 0.5           | Float  |
| efficiency  | 1             | Float in [0,1] |
| exclusive   | true          | Boolean |
| misid_cjet  | 0             | Float in [0,1] |
| misid_ljet  | 0             | Float in [0,1] |

**Table 2.** Properties defining the crude b-tagging algorithm available in older versions of \texttt{MadAnalysis 5}, shown together with their default values. These properties can be modified by typing in the interpreter set main.fastsim.bjet_id.<property> = <value>.

| Properties | Default value | type   |
|------------|---------------|--------|
| efficiency  | 1             | Float in [0,1] |
| misid_ljet  | 0             | Float in [0,1] |

**Table 3.** Properties defining the crude tau-tagging algorithm available in older versions of \texttt{MadAnalysis 5}, shown together with their default values. These properties can be modified by typing in the interpreter set main.fastsim.tau_id.<property> = <value>.
define reco_efficiency <obj> <func> [<dom>]

Defines the efficiency associated with the reconstruction of an object <obj>. The efficiency is provided as a piecewise function whose each component <func> has an optional domain of definition <dom>.

define smearer <obj> with <obs> <func> [<dom>]

Defines the standard deviation of a Gaussian of vanishing mean that is relevant for the smearing of the property <obs> of the object <obj>. The width of the Gaussian is provided as a piecewise function whose each component <func> has an optional domain of definition <dom>.

set main.fastsim.jetrecomode = <value>

Allows to switch between jet-based jet smearing (<value> = jets, default) and constituent-based jet smearing (<value> = constituents).

define tagger <obj> as <reco> <func> [<dom>]

Defines the efficiency of tagging the Monte Carlo truth object <obj> as a reconstructed <reco> object. The efficiency is provided as a piecewise function whose each component <func> has an optional domain of definition <dom>.

display reco_efficiency

display smearer

display tagger

Displays the different modules of the implemented fast detector simulation.

Table 4. The subroutines allowing for the definition of a simplified fast detector simulation in MadAnalysis 5. Reconstruction efficiencies, smearing and object identification are further detailed in sections 3.2, 3.3 and 3.4 respectively.

| Object        | PDG code | Label |
|---------------|----------|-------|
| Electron      | 11       | e     |
| Muon          | 13       | mu    |
| Tau           | 15       | ta    |
| Jet           | 21       | j     |
| Photon        | 22       | a     |
| c-jet         | 4        | c     |
| b-jet         | 5        | b     |

Table 5. List of reconstructed objects supported by the code. They are given together with their generalised PDG code and the corresponding MadAnalysis 5 label through which they can be referred to in the code. b-jets and c-jets cannot be used for reconstruction efficiencies and smearing.

| Properties         | Symbol | Label      |
|--------------------|--------|------------|
| x momentum component | $p_x$  | PX         |
| y momentum component | $p_y$  | PY         |
| z momentum component | $p_z$  | PZ         |
| Transverse momentum | $p_T$  | PT         |
| Transverse energy   | $E_T$  | ET         |
| Energy              | $E$    | E          |
| Pseudo-rapidity     | $\eta$ (or $|\eta|$) | ETA or (ABSETA) |
| Azimuthal angle (in $[0, 2\pi]$) | $\phi$ | PHI |
| Number of tracks    | $n_{tr}$ | NTRACKS |

Table 6. Main object properties entering the different functions relevant for a simplified fast detector simulation in MadAnalysis 5. The number of tracks $n_{tr}$ can only be used for jet and hadronic tau (mis)tagging, when defining the domain of application of the corresponding efficiency piecewise function. The complete list of observables can be obtained from the manual [6] or the normal mode reference card (see App. A of Ref. [8]).

reconstruction efficiencies from the interpreter. The code will then generate on run time a probability distribution indicating whether an object should be reconstructed, according to its properties.

This is achieved by means of the define keyword,

define reco_efficiency <obj> <func> [<dom>]

the first argument (reco_efficiency) indicating that one deals with the definition of a new reconstruction efficiency. In the above syntax, <obj> stands for the object under consideration, <func> for the functional form of the efficiency and <dom> consists in an optional attribute relevant for piecewise functions. In the latter case, it is indeed necessary to define, through (in)equalities, the domain of application of each piece of the full function. In the case where the user would input an ill-defined function for which the different kinematics subdomains are not disjoint (which is not recommended), the code considers a weighted sum of the corresponding efficiency functions.

3.2 Reconstruction efficiencies

The granularity of a typical high-energy physics detector, together with the lack of precision in its data acquisition system, implies that it is not always possible to fully reconstruct every single object that leaves hits in it. This can be embedded in the MadAnalysis 5 machinery by defining reference card (see App. A of Ref. [8]). Moreover, they can involve a variety of standard mathematical functions like trigonometric, cyclometric or hyperbolic functions.

In the rest of this section, we discuss reconstruction efficiencies, smearing and object identification in sections 3.2, 3.3 and 3.4 respectively. The main commands to be typed in the command-line interface are collected in table 4. At any time, the user can display the currently implemented detector simulator modules by means of the display command (see the table).

As the event file import, the generation of the working directory and the execution of the code are unchanged, we refer to section 2.1. Details on the usage of the MadAnalysis 5 simplified fast detector simulation in the expert mode of the programme are provided in section 3.5, and section 3.6 describes how to make use of the CMS and ATLAS detector parametrisation built in the MadAnalysis 5 platform.
In addition, if this domain $<\text{dom}>$ is not provided, the efficiency is understood as applicable over the entire kinematical regime.

The list of available reconstructed objects is given in table 5 and can be referred to either through a dedicated label, or through a so-called ‘generalised PDG code’. The latter extends the traditional usage of PDG codes in high-energy physics software in the sense that the code refers here to reconstructed objects instead of the corresponding Standard Model particles. As shown in the table, reconstruction efficiencies can be defined for jets ($j$), hadronic taus ($\tau$), electrons ($e$), muons ($\mu$) and photons ($\gamma$). At this stage of the detector simulation, the distinction between heavy-flavour and light jets has not been implemented. This is left for the particle identification module (see section 3.4).

The function representing the efficiency and its corresponding domain of application (if relevant) have to be provided as valid formulas in Python. They can involve any observable supported by the MadAnalysis 5 interpreter, the mostly relevant ones being the transverse momentum $p_T$ (PT), the $x$, $y$ and $z$ components of the momentum ($PX$, $PY$, and $PZ$), the pseudo-rapidity $\eta$ (ETA) or its absolute value $|\eta|$ (ABSETA), the energy $E$ ($E$), the transverse energy $E_T$ (ET) and the azimuthal angle $\phi$ $\in [0, 2\pi]$ (PHI). All these properties are collected in table 6.

For instance, in the following (toy) snippet of code, we define a flat photon reconstruction efficiency of 99% provided that the photon energy $E$ is larger than 2 GeV and its pseudo-rapidity satisfies $|\eta| < 0.88$. When the energy $E > 2$ GeV and $0.88 < |\eta| < 3$, the efficiency is of 98%, and it finally vanishes otherwise.

\begin{verbatim}
define reco_efficiency a 0 \\ [E<=2. or ABSETA>=3.]
define reco_efficiency a 0.99 \\ [E>2. and ABSETA<0.88]
define reco_efficiency a 0.98 \\ [E>2. and ABSETA>0.88 and ABSETA<3.]
\end{verbatim}

As can be seen, the usage of the or and and keywords is supported (and needed) to define the different parts of the domain of definition of the piecewise function.

### 3.3 Smearing

The impact of the detector resolution is performed via the smearing formalism. The particle momenta, as returned by the Monte Carlo simulations, are smeared by the MadAnalysis 5 smearing formalism. The particle momenta, as returned by the Monte Carlo simulations, are smeared by the MadAnalysis 5 smearing formalism. The particle momenta, as returned by the Monte Carlo simulations, are smeared by the MadAnalysis 5 smearing formalism.

The standard deviation of those Gaussian functions $\sigma$ directly depends on the object properties. It is often parametrised, taking the example of energy smearing, as originating from three distinct components [22],

\[
\sigma(E, \eta, \phi) = \frac{N(\eta, \phi) \oplus S(\eta, \phi)}{E} \oplus C(\eta, \phi) = \sqrt{N^2(\eta, \phi) + \frac{S^2(\eta, \phi)}{E} + C^2(\eta, \phi)}. \tag{1}
\]

In this expression, $N$ (in GeV) corresponds to the so-called noise term describing both the imperfections in the readout electronics, and the fluctuations arising from the simultaneous energy deposits of uncorrelated pile-up jets. This component of the resolution dominates for low-energy objects. The second term, $S$ (in $\sqrt{\text{GeV}}$), represents the stochastic contribution related to statistical random fluctuations in the physical evolution of the shower in the detector, whereas the energy-independent last term $C$ (therefore more relevant for high-energy objects) is associated with imperfections in the calorimeter geometry, anomalies in signal collection uniformities, as well as with intercalibration errors and fluctuations in the longitudinal energy content.

Several examples for the energy resolution dependence on the object energy are shown in figure 1, for three set of noise, stochastic and constant factors. We first consider a case in which the resolution is essentially dominated by its noise and stochastic terms (blue), so that it is especially poor in the low-energy regime. We then focus on a setup on which the noise factor is negligible with respect to a constant contribution (red), resulting to a quite similar overall resolution. Finally, a more realistic situation (green) in which all three components are contributing is presented, the total resolution being globally worse over the entire energy range.

In all those examples, the energy $E$ is smeared by a quantity distributed according to a Gaussian of vanishing mean, and of standard deviation given by eq. (1),

\[
E \rightarrow E' = E + N_\varepsilon [E; 0, \sigma(E, \eta, \phi)]. \tag{2}
\]

The smeared energy $E'$ is in addition enforced to be positive and the quantity $N_\varepsilon$ is a random parameter sampled
according to the standard normal distribution \( N \),

\[
N(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp \left[ -\frac{1}{2} \left( \frac{x - \mu}{\sigma} \right)^2 \right].
\]  

(3)

Smearing can be enabled in MadAnalysis 5 through the `define` command, that takes as first argument the `smearer` keyword indicating that a smearer is being defined. The complete syntax reads

\[
\text{define smearer } <\text{obj}> \text{ with } <\text{obs}> <\text{func}> \text{ [}<\text{dom}>]\n\]

The object whose properties have to be smeared is denoted by `<obj>` and has to be provided either through its generalised PDG code, or its label (see table 5). The property that has to be smeared is provided through the `<obs>` argument, the list of observables available being shown in table 6. Finally, the function `<func>` (and its corresponding domain of application `<dom>`) represents the resolution \( \sigma \), that can be provided as a piecewise function depending on any of the object properties.

After the smearing of some object properties, the missing energy is always recalculated accordingly.

As an example, we present below a snippet of code that could be relevant for the smearing of the jet transverse momentum at a typical LHC detector [23],

\[
\text{define smearer } j \text{ with } PT \ \sqrt{(0.0036 + (0.0013*PT)^2)} \ \text{[}\text{ABSETA}<0.5 \text{ and } PT > 0.1\text{]} \n\]

\[
\text{define smearer } j \text{ with } PT \ \sqrt{(0.01 + (0.0017*PT)^2)} \ \text{[}\text{ABSETA}>0.5 \text{ and } \text{ABSETA}<1.5 \text{ and } PT > 0.1\text{]} \n\]

\[
\text{define smearer } j \text{ with } PT \ \sqrt{(0.625 + (0.0031*PT)^2)} \ \text{[}\text{ABSETA}>1.5 \text{ and } \text{ABSETA}<2.5 \text{ and } PT > 0.1\text{]} \n\]

the piecewise function being only defined for cases in which the jets can effectively be reconstructed. We indeed implicitly assume that the jet reconstruction efficiency vanishes anywhere outside the considered domain. The resolution \( \sigma(PT) \) corresponds to

\[
\sigma(PT) = \begin{cases} 
\sqrt{0.0036 + (0.0013PT)^2} & \text{for } PT > 0.1 \text{ GeV and } |\eta| \leq 0.5, \\
\sqrt{0.01 + (0.0017PT)^2} & \text{for } PT > 0.1 \text{ GeV and } 0.5 < |\eta| \leq 1.5, \\
\sqrt{0.0625 + (0.0031PT)^2} & \text{for } PT > 0.1 \text{ GeV and } 1.5 < |\eta| \leq 2.5.
\end{cases}
\]  

(4)

Whilst the methods that are presented above hold for any class of object, jet smearing can be implemented in a second manner. Instead of smearing the properties of each reconstructed jet, treating it as a whole, the code offers the option to smear instead the properties of each constituent of the jet. The resulting jet four-momentum is then evaluated in a second step. For most cases, a jet-based smearing is sufficient to emulate most detector environments and there is no need to rely on any constituent-based smearing. However, this by definition renders the reconstruction process blind to any hadronic effect occurring inside a jet. The latter may be relevant for very specific studies, like when jet substructure is in order [5]. In this case, hard QCD radiation can for instance activate detector cells around those defining the jet and leads to the presence of two overlapping jets sharing the energy deposits of the surrounding cells. Such an effect can be covered by constituent-based jet smearing.

In order to activate constituent-based or jet-based jet smearing, it is sufficient to type, in the MadAnalysis 5 command-line interface,

\[
\text{set main.fastsim.jetrecomode = } <\text{val}>
\]

where `<val>` can take either the `jets` value (default), or the `constituents` value.

### 3.4 Object identification - taggers

In its simplified detector simulation, MadAnalysis 5 allows the user to input a large set of tagging efficiencies and related mistagging rates. The list of available pairs of reconstructed and truth-level objects is shown in table 7.

| Reconstructed object | Truth object                                            |
|----------------------|---------------------------------------------------------|
| b-jet                | b-jet, c-jet, light jet                                 |
| c-jet                | b-jet, c-jet, light jet                                 |
| Hadronic tau         | Hadronic tau, light jet                                 |
| Photon               | Light jet, electron, muon                              |
| Electron             | Light jet, muon, photon                                 |
| Muon                 | Electron, photon                                        |

Table 7. List of tagging efficiencies and mistagging rates that are supported by the code.

Jets can be (correctly and incorrectly) tagged as b-jet, c-jet or lighter jets. Both light jets and true hadronic taus can be identified as taus. In addition, jets faking electrons and photons, as well as electrons, muons and photons faking each other can be implemented. While the efficiencies relevant for b-tagging and tau-tagging can be provided as well (those consist in ‘taggers’ strictly speaking), the situation is slightly different for electrons, muons and photons. Here, the tagging efficiencies simply consist in the corresponding reconstruction efficiencies (see section 3.2) and should be implemented as such. The efficiency for correctly identifying a jet as a jet, a photon as a photon, an electron as an electron and a muon as a muon cannot thus be implemented as taggers, and the user has to enter them by means of the definition of reconstruction efficiencies.

A (mis)tagger can be defined by typing, in the command-line interface,

\[
\text{define tagger } <\text{true}> \text{ as } <\text{reco}> <\text{func}> \text{ [}<\text{dom}>]\n\]

Once again, this relies on the `define` command, that takes as first argument the keyword `tagger` indicating to the code that a tagger is being defined. The keyword `<true>` represents the label of the truth-level object (see table 5)
that can be tagged as a reconstructed object denoted by `<reco>` (see table 5 as well). As in the previous sections, the efficiency is provided as the function `<func>`, to be given as a valid PYTHON formula, and `<dom>` consists in its domain of application that is relevant when the (mis)tagging efficiency is a piecewise function. For any tagger of a jet or a hadronic tau as any given object, it is possible to make use of the number of tracks associated with the truth-level object to define the domain of application of the different elements of the piecewise function `<func>`.

The definition of a tagger leads to the generation of a probability distribution that indicates, at the time that the C++ code is executed, whether a reconstructed object (a photon, a jet, etc.) or one of its properties (b-tag, c-tag, etc.) has to be modified.

As an illustrative example, b-tagging performances that are typical of an LHC detector could be entered as

```python
define tagger b as b \ \ \ tanh(PT/400)*(85/(4+0.252*PT)) \ \ \ [ABSETA <= 2.5] define tagger b as b 0 [ABSETA > 2.5]
define tagger c as b \ \ \ tanh(0.018*PT)*(1/(4+0.0052*PT))
define tagger j as b 0.01+3.8e-05*PT
```

In this snippet of code, the b-tagging efficiency $\varepsilon_{bjb}$ is parametrised by [24]

$$\varepsilon_{bjb}(p_T) = \tanh \left[ \frac{p_T}{400} \right] \frac{85}{4 + 0.252 p_T} \theta \left[ 2.5 - |\eta| \right],$$

whilst the mistagging rate of a charmed jet ($\varepsilon_{bjc}$) as a b-jet and the one of a light jet as a b-jet ($\varepsilon_{bjl}$) are given by

$$\varepsilon_{bjc}(p_T) = \tanh \left[ 0.018 p_T \right] \frac{1}{4 + 0.0052 p_T},$$

$$\varepsilon_{bjl}(p_T) = 0.01 + 3.8 \cdot 10^{-5} p_T.$$

### 3.5 Expert mode

Whereas the metalanguage defining the MADANALYSIS 5 interpreter is rich, it is limited by its own definition. The user can circumvent this inherent limitation by using the platform in its so-called expert mode. Analyses are here implemented directly in C++, bypassing the MADANALYSIS 5 command-line interface. In this way, the user can benefit from all capabilities of the programme (readers, observables, etc.), and focus on implementing only the non-standard routines necessary for his/her own purpose.

The analysis skeleton generated automatically when the expert mode is switched on can incorporate a simplified detector simulation as defined in the beginning of this section. To this aim, it is sufficient to initiate MADANALYSIS 5 by typing in a shell,

```
./bin/ma5 -Re <wdir> <label> <sfs>
```

where `<dir>` stands for the working directory in which the analysis template is generated, `<label>` refers to the analysis name that is used throughout the entire analysis, and `<sfs>` is the (optional) configuration file, written as a set of normal MADANALYSIS 5 commands, defining how the simplified fast detector simulation should be run. Providing such a file is optional, so that if it is absent, the code runs as in previous versions of the programme. We refer to the manual [7] of the expert mode and ref. [8] for more information.

### 3.6 Standard LHC detector parametrisation

MADANALYSIS 5 is shipped with two predefined detector parametrisations, that are respectively related to the ATLAS and CMS detectors. Those cards are available from the `madanalysis/input` subdirectory, and have been validated through a comparison with the standard ATLAS and CMS DELPHES 3 detector configuration files. The example of the CMS validation is presented in section 4.

In order to load those cards when running MADANALYSIS 5, the user has to start the code by providing the path to the detector card as an argument of the `bin/ma5` command. This would give, from a shell,

```
./bin/ma5 -R madanalysis/input/<card>.ma5
```

where `<card>` has to be replaced by `CMS_default` and `ATLAS_default` for the CMS and ATLAS detector parametrisations respectively.

### 4 Comparison with DELPHES 3 and the Monte Carlo truth

In this section, we validate the implementation of our simplified fast detector simulation in MADANALYSIS 5. To this aim, we perform a comparison between predictions relying on our module and predictions relying instead on DELPHES 3, for a variety of Standard Model processes. Both results are moreover confronted to the expectation of the Monte-Carlo truth, where events are reconstructed as such (without any smearing and tagging). The general design of our simulation framework is depicted in section 4.1, whilst sections 4.2, 4.3, 4.4 and 4.5 focus on jet, lepton, hadronic tau and photon properties respectively.

### 4.1 Simulation framework

In order to validate our implementation, we generate several samples of Monte Carlo events describing various Standard Model processes relevant for proton-proton collisions at a centre-of-mass energy of 14 TeV. Hard scattering events are produced with MG5_AMC [25], that we use to convolute leading-order matrix elements with the leading-order set of NNPDF 2.3 parton densities [26]. Those events are matched with parton showering as modelled by the PYTHIA 8 package [27], that is also employed to simulate
hadronisation. As we focus on Standard Model processes, the typical energy scales under consideration are not so hard, ranging between a few tens of GeV to 100–200 GeV. Section 5 will be instead dedicated to new physics, so that this section will be relevant for a study of the features of our fast detector simulation framework for much larger scales.

In our comparison, we include detector effects in four different ways. First, we consider an ideal detector, or the so-called Monte Carlo truth. In practice, reconstruction is performed as described in section 2, with all detector effects being switched off. We rely on the anti-\(k_T\) algorithm with a radius parameter set to \(R = 0.5\). Second, we make use of the complex detector machinery as implemented in the Delphes 3 package, using the standard CMS detector parametrisation shipped with the programme. Finally, we consider the simplified fast detector simulation introduced in this work. We use of the complex detector machinery as implemented in an Intel(R) Xeon(R) Core E3-1271v3 CPU with a 3.6 GHz clock speed.

### Table 8. Comparison of the reconstruction of \(N_{\text{events}}\) top–antitop events with Delphes 3 and the simplified fast detector simulation introduced in this work. We consider jet smearing based either on the reconstructed jets themselves (SFS [Jets]) or on their constituents (SFS [Const.]). The results depict the size of the output file (top panel) and the time needed for the run (lower panel), as obtained by using a machine with an Intel(R) Xeon(R) Core E3-1271v3 CPU with a 3.6 GHz clock speed.

| \(N_{\text{events}}\) | Delphes 3 | SFS [Jets] | SFS [Const.] |
|----------------------|----------|------------|-------------|
| 10,000               | 821.0 Mb | 6.7 Mb     | 6.7 Mb      |
| 50,000               | 4.1 Gb   | 34.0 Mb    | 34.0 Mb     |
| 100,000              | 8.1 Gb   | 67.0 Mb    | 68.0 Mb     |
| 200,000              | 17 Gb    | 133 Mb     | 134 Mb      |

In figure 2, we present and compare various distributions when an ideal detector is considered (filled areas, truth), when a CMS-like detector is considered and modelled in Delphes 3 (solid) and when the simplified fast detector simulation (SFS) introduced in this work, parametrised to model a CMS-like detector and run both when jet-based (dotted) and constituent-based (dashed) jet smearing, is switched on.

In the upper line of the figure, we begin with studying the distribution in the number of jets \(N_j\) (upper left) and \(b\)-jets (upper right). As can be noticed by inspecting

### 4.2 Multijet production

In order to investigate the differences in the jet properties that would arise from using different reconstruction methods, we consider hard-scattering di-jet production, \(pp \rightarrow jj\). In our simulation process, we impose, at the generator level, that each jet features a transverse momentum of at least 20 GeV and a pseudo-rapidity smaller than 5. Moreover, the invariant mass of the di-jet system is enforced to be larger than 100 GeV.

In our simplified fast detector simulation (SFS), the transverse momentum \(p_T\) and energy \(E\) of the jets are smeared according to the transfer functions provided in ref. [23]. The resolution on the jet transverse momentum \(\sigma(p_T)\) is given by eq. (4), and the one on the jet energy \(\sigma(E)\) reads

\[
\sigma(E) = \begin{cases} 
0.05 \oplus 1.5\sqrt{E} & \text{for } |\eta| \leq 3, \\
0.13 \oplus 2.7\sqrt{E} & \text{for } 3 < |\eta| \leq 5.
\end{cases}
\]  

Moreover, we implement the \(b\)-tagging performance defined by eqs. (5) and (6), keeping in mind that \(b\)-jet identification is ineffective for \(|\eta| > 2.5\), by virtue of the CMS tracker geometry. Finally, we include the following jet reconstruction efficiency \(\varepsilon_j(\eta)\), that depends on the jet pseudo-rapidity,

\[
\varepsilon_j(\eta) = \begin{cases} 
92.5\% & \text{for } |\eta| \leq 1.5, \\
87.5\% & \text{for } 1.5 < |\eta| \leq 2.5, \\
80\% & \text{for } |\eta| > 2.5.
\end{cases}
\]  

This allows for the simulation of most tracker effects on the jets, with the limitation that charged and neutral jet components are equally considered. Our detector simulation also impacts all the other objects potentially present in the final state. However, we refer to the next sections for details on the simulation of the corresponding detector effects.

At the analysis level, we select as jet and lepton candidates those jets and leptons with a transverse momentum greater than 20 GeV and 10 GeV respectively. We moreover impose simple isolation requirements by ignoring any lepton lying at an angular distance smaller than 0.2 of any reconstructed jet (\(\Delta R_{\ell j} < 0.2\)), and then ignore any jet lying at a distance smaller than 0.4 of any of the remaining leptons (\(\Delta R_{j\ell} < 0.4\)). We finally require at least 20 GeV of missing transverse hadronic energy.

In figure 2, we present and compare various distributions when an ideal detector is considered (filled areas, truth), when a CMS-like detector is considered and modelled in Delphes 3 (solid) and when the simplified fast detector simulation (SFS) introduced in this work, parametrised to model a CMS-like detector and run both when jet-based (dotted) and constituent-based (dashed) jet smearing, is switched on.
Fig. 2. Jet properties when reconstruction is considered with various options for the detector simulation: an ideal detector (truth, filled area), Delphes 3 (solid) and the MadAnalysis 5 simplified fast detector simulation with either jet-based (SFS [Jets], dotted) or constituent-based (SFS [Constituents], dashed) jet smearing. We show distributions in the jet (upper left) and b-jet (upper right) multiplicity, leading light jet $p_T$ (centre left) and $|\eta|$ (centre right), as well as leading b-jet $p_T$ (lower left) and $|\eta|$ (lower right). In the lower insets, the distributions are normalised to the truth results.
jets with a moderately small more and more jets (with a greater and greater number of bins, that correspond to summing over contributions for predictions accumulate and therefore impact the largest of agreement between the three detector simulations. The Figure 3, namely the scalar sum of the transverse momentum of charged and neutral hadrons. Those difference included in Delphes 3 (see eq. (8)). The latter, as implemented in the simplified are found to originate from the reconstruction efficiency between Delphes 3 and the other fast simulation results are found to originate from the reconstruction efficiency (see eq. (8)). The latter, as implemented in the simplified fast detector simulation of MadAnalysis 5, are expected to mimic, but only to some extent, the tracker effects included in Delphes 3 that is much more complex and distinguish charged and neutral hadrons. Those difference however only impact the soft parts of the spectrum.

Finally, we consider a more inclusive variable in figure 3, namely the scalar sum of the transverse momentum of all reconstructed jets \( H_T \). We again get a nice level of agreement between the three detector simulations. The differences between Delphes 3-based and SFS-based predictions accumulate and therefore impact the largest \( H_T \) bins, that correspond to summing over contributions for more and more jets (with a greater and greater number of jets with a moderately small \( p_T \)).

4.3 Electrons and muons

In this section, we perform on a comparison of electron and muon properties when these are reconstructed from the different methods considered in this work. To this aim, we generate a sample of Drell-Yan events, \( pp \rightarrow \ell^+\ell^- \) with \( \ell = e, \mu \). In our simulations, we impose, at the Monte Carlo generator level, the dilepton invariant mass to be at least 50 GeV and constrain each individual lepton to feature a transverse momentum greater than 10 GeV.

The electron and muon reconstruction efficiencies \( \varepsilon_e \) and \( \varepsilon_\mu \) are extracted from ref. [28], and depend both on the electron and muon transverse momentum \( p_T \) and pseudorapidity \( \eta \). They respectively read

\[
\varepsilon_e(p_T, \eta) = \begin{cases} 
0.7 & \text{for } p_T > 10 \text{ GeV and } |\eta| \leq 1.5 , \\
0.525 & \text{for } p_T > 10 \text{ GeV and } 1.5 < |\eta| \leq 2.5 , \\
0 & \text{otherwise} , 
\end{cases}
\]

\[
\varepsilon_\mu(p_T, \eta) = \begin{cases} 
0.891 B(p_T) & \text{for } p_T > 10 \text{ GeV and } |\eta| \leq 1.5 , \\
0.882 B(p_T) & \text{for } p_T > 10 \text{ GeV and } 1.5 < |\eta| \leq 2.4 , \\
0 & \text{otherwise} , 
\end{cases}
\]

with

\[
B(p_T) = \theta \left[ 1 - \frac{p_T}{\text{TeV}} \right] - e \left[ \frac{1}{2} \left( \frac{1 - p_T}{\text{TeV}} \right) \right] \theta \left[ \frac{p_T}{\text{TeV}} - 1 \right] .
\]

In addition, we model the effects stemming from the electromagnetic calorimeter by smearing the electron energy in a Gaussian way, with a standard deviation \( \sigma_e(E, \eta) \) defined by [28, 29]

\[
\frac{\sigma_e(E, \eta)}{E} = \begin{cases} 
1 + 0.64|\eta|^2 \left[ \frac{0.4}{E} \oplus \frac{0.11}{\sqrt{E}} \oplus 0.008 \right] & \text{for } |\eta| \leq 1.5 , \\
2.16 + 5.6(|\eta| - 2)^2 \left[ \frac{0.4}{E} \oplus \frac{0.11}{\sqrt{E}} \oplus 0.008 \right] & \text{for } 1.5 < |\eta| \leq 2.5 , \\
2.08 \frac{1}{\sqrt{E}} \oplus 0.107 & \text{for } 2.5 < |\eta| \leq 5 .
\end{cases}
\]

At the analysis level, we implement similar selections as in section 4.2. We consider as jets and leptons those jets and leptons with \( p_T > 20 \) GeV and 10 GeV respectively, and require all studied objects to be isolated. Any lepton too close to a jet is discarded (\( \Delta R_{\ell j} < 0.2 \)), and any jet too close to any of the remaining leptons is discarded too (\( \Delta R_{\ell j} < 0.4 \)). Moreover, we impose a selection on the invariant mass of the lepton pair, \( m_{\ell\ell} > 55 \) GeV, for events featuring at least two reconstructed leptons of opposite electric charges.

In figure 4, we compare various lepton-related distributions after using the different considered options for the simulation of the detector effects. We start by considering the distributions in the electron (upper left) and muon (upper right) multiplicity, which show that the loss in leptons relatively to the Monte truth is similar for all three
Fig. 4. Same as figure 2, but for the electron and muon multiplicity distribution (upper row), and the pseudo-rapidity (central row) and transverse momentum (lower row) spectrum of the leading electron (left) and muon (right).
detector simulations. This is not surprising as the efficiencies have been tuned accordingly.

For similar reasons, the detector effects impacting the leading electron (centre left) and muon (centre right) pseudo-rapidity distributions are similarly handled in all three setups. Relatively to the Monte-Carlo truth, not a single electron and muon is reconstructed for pseudo-rapidities larger than 2.5 and 2.4 respectively, and the fraction of lost leptons is larger for $|\eta| > 1.5$ than for $|\eta| < 1.5$. This directly stems from the detector geometry and design, that make it impossible to reconstruct any non-central lepton and lead to degraded performance for larger pseudo-rapidities (as implemented in all detector simulator reconstruction efficiencies).

The various options for modelling the detector effects however yield important differences for the leading electron (lower left) and leading muon (lower right) transverse momentum distributions. While we obtain a good agreement between SFS-based and DELPHES 3-based results for any lepton exhibiting a transverse momentum larger than 40 GeV, DELPHES 3 predicts a smaller number of lepton featuring $p_T < 40$ GeV than in the SFS case. The difference ranges up to 30% at the kinematical selection threshold of $p_T \approx 10$ GeV. This originates from the inner machineries implemented in the various codes. In DELPHES 3, hadron-level objects are first converted into tracks and calorimetric deposits, which involves efficiencies provided by the user. Next, momenta and energies are smeared, before that the code reconstructs the objects to be used at the analysis level. This latter step relies again on user-defined reconstruction efficiencies, this time specific to each class of reconstructed objects. In the MADANALYSIS 5 SFS, such a splitting of the reconstruction efficiency into two components has not been implemented, for the purpose of keeping the detector modelling simple. This could introduce significant shifts in the distributions describing the properties of the reconstructed objects, especially for softer objects where those details matter.

This feature consequently impacts any observable, potentially more inclusive, that would depend on the properties of the various leptons. This is illustrated in figure 5 where we present the missing transverse energy ($E_T$) spectrum. In this case, the differences are even more pronounced than for the lepton $p_T$ spectra, due to the particle flow method used in DELPHES 3 to reconstruct the missing energy of each event. In contrast, MADANALYSIS 5 only sums vectorially the momentum of all visible reconstructed objects.

Most differences occur in the low-energy region of the distribution, that is largely impacted by the differently modelled softer objects. In the larger $E_T$ regime, a constituent-based jet smearing turns out to absorb a good part of the missing energy degradation modelled in DELPHES 3. In this configuration, more objects are smeared so that any given effect gets eventually compensated by the smearing of the numerous other objects. In contrast, jet-based smearing yields predictions agreeing quite well with those obtained with DELPHES 3.

![Fig. 5. Same as figure 2, but for the missing transverse energy spectrum.](image)

4.4 Tau pair-production

We now move on with a comparison of the detector performance for the reconstruction of hadronic taus, for which DELPHES 3 and our simplified detector simulation follow quite different methods. DELPHES 3 uses a cone-based algorithm to identify hadronic taus from the jet collection, whereas in the SFS approach, hadronic taus are tagged through the matching of a reconstructed jet with a hadron-level object. Such differences can cause large variations in the properties of the reconstructed taus, as shown in the rest of this section.

To quantify the impact of this difference, we produce a sample of di-tau events, $pp \rightarrow \tau^+ \tau^-$, that includes a 10 GeV selection on the tau transverse momentum at the generator level. In the SFS case, we reconstruct jets, electrons and muons as described in sections 4.2 and 4.3. We moreover implement a tau reconstruction efficiency $\varepsilon_{\text{tracks}}$ aiming at reproducing tracker effects,

$$\varepsilon_{\text{tracks}}(p_T) = 0.7 \, \theta \left( \frac{p_T}{\text{GeV}} - 20 \right),$$

as well as a tau tagging efficiency $\varepsilon_{\tau_{ij}}$ that is identical to the one embedded in the standard DELPHES 3 CMS detector parametrisation, and that is independent of the tau transverse momentum,

$$\varepsilon_{\tau_{ij}}(\eta) = 0.6 \, \theta (2.5 - |\eta|).$$

The corresponding mistagging rate of a light jet as a tau $\varepsilon_{\tau_{ij}}$ is flat and independent of the kinematics,

$$\varepsilon_{\tau_{ij}} = 0.01.$$
by selecting only taus whose transverse momentum is larger than 20 GeV. Furthermore, to remove any potential overlap between the tau and the jet collection, we ignore any jet that is too close to a tau ($\Delta R_{\tau j} < 0.2$) and impose that the missing hadronic activity $H_T$, defined as the missing transverse energy stemming from the jet activity only, is larger than 20 GeV.

In figure 6, we compare predictions obtained with the SFS approach, DELPHES 3 and the Monte Carlo truth (i.e. for an ideal detector).

We start by considering the tau multiplicity spectrum (upper left). As expected from the imperfections of the tagger of eqs. (13) and (14), a large number of true tau objects are not tagged as such, for all three detector simulation setups. We observe a quite good agreement between SFS-based and DELPHES 3-based results, despite the above-mentioned differences between the two approaches for tau tagging.

As for the electron and muon case, predictions for the pseudo-rapidity spectrum of the leading tau (lower left) are comparable, regardless of the adopted detector simulator. The main differences arise in the $|\eta| > 2.5$ forward regime. For SFS-based results, forward taus are stemming from the misidentification of a light jet as a tau, whereas in the DELPHES 3 case, the entire jet collection (including $c$-jets and $b$-jets) is used. This different treatment, on top of the above-mentioned differences inherent to the whole tau-tagging method, leads to the observed deviations between the predictions for $|\eta| > 2.5$.

In the lower right panel of figure 6, we present the distribution in the transverse momentum of the leading tau. As for the other considered observables, the detector effects are pretty important when compared with the Monte Carlo truth. For hard taus with $p_T \gtrsim 50$ GeV, DELPHES 3 and SFS-based predictions are in very good agreement. However, the different treatment in the two simulators sig-
nificantly impacts the lower $p_T$ regime, leading in a very different behaviour.

In the upper right panel, we show how those effects impact a more global observable. This is illustrated with the missing transverse energy $E_T$ (that is reconstructed with a particle flow algorithm in DELPHES 3). The discrepancies between DELPHES 3 and SFS results are not so drastic as for the $p_T(\tau_1)$ distribution in the soft regime, but impact instead the entire spectrum with a shift of $\mathcal{O}(10\%)$ in one way or the other. When $E_T > 100$ GeV (not shown on the figure), however, we enter the hard regime where the exact details of the detector simulation matter less and a good agreement is obtained between all predictions.

4.5 Photon production

This subsection is dedicated to the last class of objects that could be reconstructed in a detector, namely photons. To compare the expectation from predictions made with DELPHES 3 to those made with the simplified fast detector simulator of MADANALYSIS 5, we generate a sample of diphoton events, $pp \rightarrow \gamma\gamma$. In our simulations, we impose a generator level selection of 10 GeV on the photon $p_T$, as its pseudo-rapidity is constrained to be below 5 in absolute value.

The SFS simulation includes a reconstruction efficiency $\varepsilon_\gamma$ given by

\[
\varepsilon_\gamma(p_T, \eta) = \begin{cases} 
0.96 & \text{for } p_T > 10 \text{ GeV and } |\eta| \leq 1.5 , \\
0.875 & \text{for } p_T > 10 \text{ GeV and } 1.5 < |\eta| \leq 2.5 , \\
0 & \text{otherwise ,} \end{cases}
\]

(15)

and the photon energy is smeared as in eq. (11), this smearing originating from the same electronic calorimeter effects as in the electron case.

At the analysis level, we follow the same strategy as in the previous section, on top of which we enforce the reconstructed photons to have a transverse momentum of at least 10 GeV. Moreover, we remove, from the photon collection, any jet that is too close to a photon ($\Delta R_{\gamma j} < 0.4$).

In order to assess the impact of the detector on the photons, we compare in figure 7 the Monte Carlo truth predictions (ideal detector) with the results obtained with the MADANALYSIS 5 SFS simulation and with DELPHES 3. In all cases, the detector capabilities in reconstructing the photons are quite good, only a few photons being lost, and their properties are nicely reproduced. We nevertheless observe that the energy spectrum turns to be softer after the reconstruction process, as illustrated in the lower panel of the figure. This mainly results from the smearing effects.

**Fig. 7.** Same as figure 2, but for the reconstructed photon multiplicity distribution (upper), and for the pseudo-rapidity (centre) and energy (lower) spectrum of the leading reconstructed photon.
5 Reinterpreting the results of the LHC

5.1 Generalities

The LHC collaborations usually interpret their results for a specific set of selected models. This hence leaves the task of the reinterpretation in other theoretical frameworks to studies that have to be carried out outside the collaborations. The most precise method that is available to theorists in this context relies on the Monte Carlo simulation of any new physics signal of interest. The resulting events are first reconstructed by including a detector simulation mimicking the ATLAS or CMS detector, and next studied in order to see to which extent the signal regions of a given analysis are populated. From those predictions, it is then possible to conclude about the level of exclusion of the signal, from a comparison with data and the Standard Model expectation.

Such an analysis framework is available within MADAnalysis 5 for half a decade [8, 9]. In practice, it makes use of the MADAnalysis 5 interface to Delphes 3 to handle the simulation of the detector.

In this work, we have extended this infrastructure, so that the code offers, from version 1.8.5i onwards, the choice to employ either Delphes 3 or the SFS to deal with the simulation of the detector response. Similarly to what is done in Rivet [4] or ColliderBit [30], the simulation of the detector can now be handled together with a simple event reconstruction to be performed with FastJet, through transfer functions embedding the various reconstruction and tagging efficiencies (see section 3).

The estimation of the LHC potential relatively to any given new physics signal can be achieved by typing, in the MADAnalysis 5 interpreter (after having started the programme in the reconstruction-level mode),

```
set main.recast = on
import <events.hepmc.gz>
submit
```

The above set of commands turns on the recasting module of the platform and allows for the reinterpretation of the results of all reimplemented LHC analyses available on the user system. The signal to test is described by the `<events.hepmc.gz>` hadron-level event sample. After submission, MADAnalysis 5 generates a recasting card requiring to switch on or off any of all reimplemented analyses, regardless that they use Delphes 3 or the SFS for the simulation of the detector response. On run time, the code automatically chooses the way to handle it, so that the user does not have to deal with it by himself/herself. We refer to refs. [8, 10] for more information on LHC recasting in the MADAnalysis 5 context, the installation of the standard Public Analysis Database analyses (PAD) that rely on Delphes 3, as well as for a detailed list with all available options.

All analyses that have been reimplemented and validated within the SFS context can be downloaded from the internet and locally installed by typing, in the command-line interface,

```
install PADForSFS
```

Up to now, this command triggers the installation of two analyses, namely the ATLAS-SUSY-2016-07 search for gluinos and squarks in the multi-jet plus missing transverse energy channel [31] and the CMS-SUS-16-048 search for charginos and neutralinos through a signature comprised of soft leptons and missing transverse energy [32]. Details about the validation of the two SFS reimplementations are provided in sections 5.2 and 5.3 respectively. The list of all validated analyses available for LHC recasting with an SFS detector simulation will be maintained up-to-date on the MadAnalysis 5 website.

In the rest of this section, we compare the SFS-based predictions with those resulting from the usage of the Delphes 3 software for the simulation of the response of the LHC detectors. Additionally to a direct comparison of a Delphes 3-based and transfer-function-based approach for LHC recasting, this allows one to assess the capabilities of the SFS approach as compared with Delphes 3 in the case of events featuring very hard objects that are typical of most searches for new physics at the LHC. Very hard jets are in particular considered in the ATLAS-SUSY-2016-07 analysis, which contrasts with the study of the jet properties achieved in section 4 that solely covers objects featuring a moderate transverse momentum of 10–100 GeV.

5.2 Recasting a multi-jet plus missing energy ATLAS search for squarks and gluinos

In the ATLAS-SUSY-2016-07 analysis, the ATLAS collaboration searches for squarks and gluinos through a signature comprised of 2 to 6 jets and a large amount of missing transverse energy. A luminosity of 36.1 fb$^{-1}$ of proton-proton collisions at a centre-of-mass energy of 13 TeV is analysed. This search includes two classes of signal regions. The first one relies on the effective mass variable $M_{\text{eff}}(N)$, defined as the scalar sum of the transverse momentum of the $N$ leading jets and the missing transverse energy, and the second one on the recursive jigsaw reconstruction technique. However, only the former region can be recasted due to the lack of public information in the jigsaw case.

Consequently, only signal regions depending on the $M_{\text{eff}}(N)$ quantity have been reimplemented in the MADAnalysis 5 framework, as detailed in ref. [33]. In this implementation, the simulation of the detector is handled with Delphes 3 and an appropriately tuned parameter card. This Delphes 3-based recast code has been validated by reproducing public information provided by the ATLAS collaboration, so that we use it as a reference below. In the following, we denote predictions obtained with it as ‘PAD’ results, the acronym PAD referring to the traditional Public Analysis Database of MADAnalysis 5 that relies on Delphes 3 for the simulation of the detector. In contrast, results obtained by using an SFS

---

5 See the webpage http://madanalysis.irmp.ucl.ac.be/wiki/SFS.
detector simulation are tagged as ‘SFS’ or ‘PADForSFS’ results.

In order to illustrate the usage of the SFS framework for LHC recasting, we have modified the original ATLAS-SUSY-2016-07 analysis reimplementation and included it in the PADForSFS database, together with an appropriate ATLAS SFS detector parametrisation. The user has thus the choice to use either DELPHES 3 or the SFS framework for the reinterpretation of the results of this ATLAS analysis.

The validation of our reimplementation has been achieved by comparing predictions for a well-defined benchmark scenario with those obtained with the reference version of the reimplementation based on DELPHES 3. We have adopted a simplified model setup inspired by the Minimal Supersymmetric Standard Model in which all superpartners are decoupled, with the exception of the gluino and the lightest neutralino. Their masses have been fixed to 1 TeV and 825 GeV respectively, and the gluino is enforced to decay into a di-jet plus neutralino system with a branching ratio of 1.

We have generated 200,000 new physics Monte Carlo events by matching leading-order matrix elements convoluted with the leading order set of NNPDF 2.3 parton densities [26] as generated by MG5_aMC [25], with the parton shower machinery of PYTHIA 8 [27]. Gluino decays are handled with the MADSpin [34] and MadWidth [35] packages, and we have relied on PYTHIA 8 for the simulation of the hadronisation processes. Those events have then been analysed automatically in MadAnalysis 5, both in the ‘PAD’ context with a DELPHES 3-based detector simulation and in the SFS context with an SFS-based detector simulation.

We have compared the two sets of results and found that they deviate by at most 10% for all signal regions populated by at least 20 events (out of the 200,000 simulated events). In addition, we have verified that enforcing a constituent-based or jet-based jet smearing had little impact on the results. We have observed that this choice indeed leads to a modification of the SFS predictions of about 1%, so that the jet-smearing choice is irrelevant. Jet-based jet smearing is therefore used below. Furthermore, in terms of performance, the run of DELPHES 3 has been found 50% slower than the SFS one.

We present a subset of the results in table 9, focusing on three of the ATLAS-SUSY-2016-07 signal regions that are among the most populated ones by the considered 1 TeV gluino signal. We show predictions obtained by using DELPHES 3 for the simulation of the detector (PAD), and depict them both in terms of the number of events \( n_i \) surviving a cut \( i \) and of the related cut efficiency

\[
\varepsilon_i = n_i / n_{i-1} .
\]

We additionally display SFS-based predictions (SFS), showing again both the number of events and the various cut efficiencies.

| Cuts | \( n_i^{PAD} \) | \( \varepsilon_i^{PAD} \) | \( n_i^{SFS} \) | \( \varepsilon_i^{SFS} \) | \( \delta_i \) |
|------|----------------|-----------------|--------------|-----------------|--------|
| Initial | 7262.2 | - | 7262.2 | - | - |
| Preselection | 1129.8 | 0.156 | 1092.0 | 0.150 | 3.3% |
| At least 2 jets | 1125.0 | 0.996 | 1087.5 | 0.996 | 0 |
| At least 3 jets | 1073.0 | 0.954 | 1032.9 | 0.950 | 0.4% |
| \( \Delta \varphi(j_i - p_T^\text{jet}) > 0.4 \) | 848.5 | 0.791 | 821.6 | 0.796 | 0.6% |
| \( \Delta \varphi(j_i - p_T^\text{jet}) > 0.2 \) | 767.6 | 0.905 | 743.9 | 0.905 | 0.1% |
| \( p_T(j_i) > 700 \text{ GeV} \) | 53.9 | 0.070 | 54.8 | 0.074 | 4.9% |
| \( p_T(j_i) > 50 \text{ GeV} \) | 53.9 | 1 | 54.8 | 1 | 0 |
| \( M_\text{eff}(3) > 1300 \text{ GeV} \) | 43.0 | 0.798 | 44.4 | 0.812 | 1.7% |
| Preselection | 7262.2 | - | 7262.2 | - | - |
| At least 2 jets | 1129.8 | 0.156 | 1092.0 | 0.150 | 3.3% |
| At least 4 jets | 857.4 | 0.762 | 821.2 | 0.755 | 0.9% |
| \( \Delta \varphi(j_i - p_T^\text{jet}) > 0.4 \) | 677.1 | 0.790 | 652.3 | 0.794 | 0.6% |
| \( \Delta \varphi(j_i - p_T^\text{jet}) > 0.2 \) | 522.0 | 0.771 | 500.8 | 0.768 | 0.4% |
| \( p_T(j_i) > 100 \text{ GeV} \) | 113.5 | 0.218 | 112.5 | 0.225 | 3.3% |
| \( |\eta(j_i)| < 2 \) | 91.1 | 0.803% | 89.0 | 0.791 | 1.5% |
| Aplanarity > 0.04 | 54.5 | 0.598 | 52.9 | 0.594 | 0.7% |
| \( E_T/\sqrt{M_T} > 16 \text{ GeV} \) | 49.2 | 0.902 | 48.2 | 0.911 | 1.0% |
| \( M_\text{eff}(4) > 1400 \text{ GeV} \) | 27.0 | 0.549 | 27.5 | 0.570 | 3.8% |

Table 9. Cut-flow charts associated with three signal regions of the ATLAS-SUSY-2016-07 analysis, among the most populated ones by a 1 TeV gluino signal. We focus on regions relevant for events featuring at least 3 jets (top), 4 jets (middle) or 6 jets (lower). In our notation, \( j_k \) denotes any of the three leading jets, and \( j_i \) any jet from the third one.
The deviations $\delta_i$ between the DELPHES 3 and SFS predictions are evaluated at the level of the efficiencies,

$$\delta_i = \left| 1 - \epsilon_i^{\text{SFS}} / \epsilon_i^{\text{PAD}} \right|.$$  \hspace{1cm} \hspace{1cm} (17)

As above-mentioned, the deviations at any cut-level are smaller than 10%, and often lie at the level of 1%. This is in particular the case for the cuts relevant to the other 19 (not shown) signal regions. A more complete set of results, including predictions for all signal regions, is available online.6

5.3 Recasting a CMS search for compressed electroweakinos with soft leptons and missing energy

The CMS-SUS-16-048 analysis is an unusual search for the supersymmetric partners of the Standard Model gauge and Higgs bosons. It relies on the reconstruction of soft leptons with a transverse momentum smaller than 30 GeV, and mainly targets the associated production of a neutralino and an off-shell gauge boson, possibly carrying the same flavour (SF), and some missing transverse momentum carried away by the two produced lightest neutralinos.

The CMS collaboration has also designed a series of signal regions dedicated to the search for electroweakino production from stop decays,

$$pp \rightarrow \tilde{\chi}_1^{\pm} \tilde{\chi}_1^0 \rightarrow (W^* \chi_1^0) (Z^* \chi_1^0).$$ \hspace{1cm} \hspace{1cm} (18)

The considered signal is thus potentially comprised of a pair of opposite-sign (OS) soft leptons (arising from the off-shell gauge bosons), possibly carrying the same flavour (SF), and some missing transverse momentum carried away by the two produced lightest neutralinos.

The CMS-SUS-16-048 analysis has been implemented in the MADAnalysis 5 framework and validated in the context of the last Les Houches workshop on TeV colliders [36], both for a simulation of the detector effects relying on DELPHES 3 and on the SFS infrastructure. This analysis is therefore both included in the standard PAD and new PADForSFS database.

---

6 See the PADForSFS webpage http://madanalysis.irmp.ucl.ac.be/wiki/SFS.
As in section 5.2, the recast code based on DELPHES 3 has been first validated against publicly available material provided by the CMS collaboration. The SFS-based version of the analysis has been developed next, using the DELPHES 3-based one as a reference for the validation procedure. Whereas both reimplementations were validated in ref. [36], we report in the following on the comparison of DELPHES 3-based and SFS-based predictions for a given signal, as this fits in the spirit of this work aiming at documenting the performance of the new SFS module of MADANALYSIS 5 and comparing with a more standard DELPHES 3-based detector simulation.

For our comparison, we again consider a simplified model inspired by the Minimal Supersymmetric Standard Model. The Standard Model is extended by the three electroweakino states relevant for the signal of eq. (18). We make of one of the next-to-minimal simplified scenarios proposed in ref. [37] and that is studied in the considered CMS analysis, and set the mass of the lightest neutralino to 142.5 GeV and the ones of the other states to 150 GeV.

We generate 300,000 signal events in the same simulation chain as the one described in the previous section. When comparing the recasting results, we observe a good agreement between DELPHES 3-based and SFS-based predictions. Ignoring signal regions featuring a poor statistics (less than 10 events out of the 300,000 generated ones), the cut efficiencies are found to differ by at most 50%, although the deviations are found to lie at the percent level for a large majority of cuts from all signal regions. In addition, using jet-based or constituent-based jet smearing does not change the conclusions, the deviations $\delta_i$ from DELPHES 3 being impacted by about 1%.

We illustrate our results in table 10 for two of the CMS-SUS-16-048 signal regions that are among the most populated ones by the considered compressed electroweakino signal. The complete set of results can be obtained online, on the SFS wiki page.

6 Conclusions

We have discussed the implementation of a new detector emulator in the MADANALYSIS 5 framework. This extends the capacities of the platform when FASTJET is used for event reconstruction.

Our work includes two components. First, the PYTHON-like interpreter of MADANALYSIS 5 has been extended by new commands. Those allow for an intuitive parametrisation of a high-energy physics detector through user-defined smearing, reconstruction and identification/mistagging efficiency functions. Next, the C++ core of the programme has been modified so that it could handle those user-defined functions, that are converted from the PYTHON-like MADANALYSIS 5 metalanguage to C++ by the interpreter.

On run time, MADANALYSIS 5 automatically generates a full C++ code that allows for event reconstruction with FASTJET and that incorporates the input detector effects. The code is then automatically compiled and run. Correspondingly, we have extended the recasting infrastructure of MADANALYSIS 5. The user has now the option to rely on our fast simplified detector simulator instead of DELPHES 3 to simulate the ATLAS and CMS detectors. Two LHC analyses are currently available and have been validated. Moreover, the programme is shipped with two configuration scripts reproducing the response of the ATLAS and CMS detectors.

In order to validate our implementation, we have compared the associated predictions with those obtained when the DELPHES 3 package is used instead for the simulation of the detector effects. We have considered Standard Model processes as well as new physics signals, the latter being investigated in the context of LHC recasting. In all our comparisons, we have found that the variations between the two approaches are of at most about 10%. The main advantage of our emulator is, however, the speed and the disk space needed to store the output file. Our simplified fast detector simulation is indeed 30%-50% faster than DELPHES 3, and requires 100 times less disk space. Moreover, with such a machinery at hand, it becomes possible to use the code to single out a particular detector effect and hence determine the performance of any given (future) detector.

With this work, we have hence augmented the capabilities of the MADANALYSIS 5 package, offering a new fast and efficient way to emulate the response of a typical high-energy physics detector. Predictions compatible with those obtained when DELPHES 3 is used have been found, the difference lying at the level of the uncertainties inherent to a fast detector simulation carried out outside the experimental collaborations.

In a future release, we plan to further extend both our package and the normal mode of MADANALYSIS 5 to include the handling of vertex position and impact parameter variables. Those heavier changes will consequently allow for an appropriate modelling of long-lived particles in the context of our simplified fast detector simulation.

7 Acknowledgements

The authors are grateful to Eric Conte, Mariana Frank and Sabine Kraml for useful comments on the manuscript. JYA acknowledges the hospitality of the LPTHE (Sorbonne University), IPPP (Durham University) and LNF (INFN) where parts of this work have been completed during his visits. JYA has received funding from the European Union’s Horizon 2020 research and innovation programme as part of the Marie Sklodowska-Curie Innovative Training Network MChitITN3 (grant agreement no. 722104).

References

1. F. Spano, Unfolding in particle physics: a window on solving inverse problems, EPJ Web Conf. 55 (2013) 03002.
2. GEANT4 collaboration, S. Agostinelli et al., GEANT4: A Simulation toolkit, Nucl. Instrum. Meth. A 506 (2003) 250–303.

3. DELPHES 3 collaboration, J. de Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens et al., DELPHES 3, A modular framework for fast simulation of a generic collider experiment, JHEP 02 (2014) 057, [1307.6346].

4. A. Buckley, J. Butterworth, L. Lonnblad, D. Grellscheid, H. Hoeth, J. Monk et al., FastJet user manual, Comput. Phys. Commun. 184 (2013) 2803–2819, [1003.0694].

5. A. Buckley, D. Kar and K. Nordström, Fast simulation of detector effects in Rivet, SciPost Phys. 8 (2020) 025, [1910.01657].

6. E. Conte, B. Dumont, B. Fuks and C. Wymant, Designing and recasting LHC analyses with MadAnalysis 5, Eur. Phys. J. C 74 (2014) 3103, [1405.3982].

7. E. Conte and B. Fuks, Confronting new physics theories to LHC data with MadAnalysis 5, Int. J. Mod. Phys. A33 (2018) 1830027, [1808.00480].

8. B. Dumont, B. Fuks, S. Kraml, S. Bein, G. Chalons, E. Conte et al., Toward a public analysis database for LHC new physics searches using MADANALYSIS 5, Eur. Phys. J. C 75 (2015) 56, [1407.3278].

9. J. Y. Araz, M. Frank and B. Fuks, Reinterpreting the results of the LHC with MadAnalysis 5: uncertainties and higher-luminosity estimates, 2019.11418.

10. M. Cacciari, G. P. Salam and G. Soyez, FastJet User Manual, Eur. Phys. J. C72 (2012) 1856, [1111.6097].

11. M. Dobbs and J. B. Hansen, The HepMC C++ Monte Carlo event record for High Energy Physics, Comput. Phys. Commun. 134 (2001) 41–46.

12. M. Cacciari, G. P. Salam and G. Soyez, The Anti-k(T) jet clustering algorithm, JHEP 04 (2008) 063, [0802.1189].

13. S. Catani, Y. L. Dokshitzer, M. H. Seymour and B. R. Webber, Longitudinally invariant Kt clustering algorithms for hadron hadron collisions, Nucl. Phys. B406 (1993) 187–224.

14. S. Ellis and D. E. Soper, Successive combination jet algorithm for hadron collisions, Phys. Rev. D48 (1993) 3160–3166, [hep-ph/9305266].

15. Y. L. Dokshitzer, G. D. Leder, S. Moretti and B. R. Webber, Better jet clustering algorithms, JHEP 08 (1997) 001, [hep-ph/9707323].

16. M. Wobisch and T. Wengler, Hadronization corrections to jet cross-sections in deep inelastic scattering, in Monte Carlo generators for HERA physics. Proceedings, Workshop, Hamburg, Germany, 1998-1999, pp. 270–279, 1999, [hep-ph/9907280].

17. G. P. Salam and G. Soyez, A Practical Seedless Infrared-Safe Cone jet algorithm, JHEP 05 (2007) 086, [0704.0292].

18. Particle Data Group collaboration, M. Tanabashi et al., Review of Particle Physics, Phys. Rev. D 98 (2018) 030001.

19. E. Boos et al., Generic User Process Interface for Event Generators, in 2nd Les Houches Workshop on Physics at TeV Colliders, 9, 2001. [hep-ph/0109068].

20. J. Alwall et al., A Standard format for Les Houches event files, Comput. Phys. Commun. 176 (2007) 300–304, [hep-ph/0609017].

21. C. Fabjan and F. Gianotti, Calorimetry for particle physics, Rev. Mod. Phys. 75 (2003) 1243–1286.

22. CMS collaboration, S. Chatrchyan et al., Description and performance of track and primary-vertex reconstruction with the CMS tracker, JINST 9 (2014) P10009, [1405.6559].

23. CMS collaboration, S. Chatrchyan et al., Identification of b-Quark Jets with the CMS Experiment, JINST 8 (2013) P04013, [1211.4462].

24. J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer et al., The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations, JHEP 07 (2014) 079, [1405.0301].

25. R. D. Ball et al., Parton distributions with LHC data, Nucl. Phys. B 867 (2013) 244–289, [1207.1303].

26. T. Sjöstrand, S. Ask, J. R. Christiansen, R. Corke, N. Desai, P. Iten et al., An Introduction to PYTHIA 8.2, Comput. Phys. Commun. 191 (2015) 159–177, [1410.3012].

27. CMS collaboration, V. Khachatryan et al., Performance of Electron Reconstruction and Selection with the CMS Detector in Proton-Proton Collisions at $\sqrt{s} = 8$ TeV, JINST 10 (2015) P06005, [1502.02701].

28. CMS collaboration, S. Chatrchyan et al., Energy Calibration and Resolution of the CMS Electromagnetic Calorimeter in pp Collisions at $\sqrt{s} = 7$ TeV, JINST 8 (2013) P09009, [1306.1818].

29. GAMBIT collaboration, C. Balázs et al., ColliderBit: a GAMBIT module for the calculation of high-energy collider observables and likelihoods, Eur. Phys. J. C77 (2017) 795, [1705.07919].

30. ATLAS collaboration, M. Aaboud et al., Search for squarks and gluinos in final states with jets and missing transverse momentum using 36 fb$^{-1}$ of $\sqrt{s} = 13$ TeV pp collision data with the ATLAS detector, Phys. Rev. D97 (2018) 112001, [1712.02332].

31. CMS collaboration, M. Sirunyan et al., Search for new physics in events with two soft oppositely charged leptons and missing transverse momentum in proton-proton collisions at $\sqrt{s} = 13$ TeV, Phys. Lett. B782 (2018) 440–467, [1801.01846].

32. G. Chalons and H. Reyes-Gonzalez, MadAnalysis 5 implementation of ATLAS-SUSY-16-07 (arXiv:1712.03329), 10.7484/INSPIREHEP.DATA.56DC.PPE2.

33. P. Artoisenet, R. Frederix, O. Mattelaer and R. Rietkerk, Automatic spin-entangled decays of heavy resonances in theories with FeynRules and MadGraph5 aMC@NLO, Comput. Phys. Commun. 197 (2015) 312–323, [1402.1178].

34. Brooijmans et al., Les Houches 2019 Physics at TeV Colliders: New Physics Working Group Report, in 11th Les Houches Workshop on Physics at TeV Colliders: PhysTeV Les Houches, 2, 2020. 2002.12220.
37. B. Fuks, M. Klasen, S. Schmiemann and M. Sunder, 
Realistic simplified gaugino-higgsino models in the 
MSSM, Eur. Phys. J. C 78 (2018) 209, [1710.09941].