Learning to Remove Pileup at the LHC with Jet Images

Patrick T. Komiske,[a] **Eric M. Metodiev,**[a] **Benjamin Nachman,**[b] **and Matthew D. Schwartz**[c]

[a] Center for Theoretical Physics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA
[b] Physics Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA
[c] Department of Physics, Harvard University, Cambridge, MA 02138, USA

E-mail: pkomiske@mit.edu, metodiev@mit.edu, bpnachman@lbl.gov, schwartz@physics.harvard.edu

Abstract. We present the Pileup Mitgation with Machine Learning (PUMML) algorithm for pileup removal at the Large Hadron Collider (LHC) based on the jet images framework using state-of-the-art machine learning techniques. We demonstrate that our algorithm outperforms existing methods on a wide range of jet observables up to pileup levels of 140 collisions per bunch crossing. We also investigate what aspects of the event our algorithms are utilizing by understanding the learned parameters of a simplified version of the model.

1. Introduction

The Large Hadron Collider (LHC) is operated at very high instantaneous luminosities to achieve the large statistics required to search for exotic Standard Model (SM) or beyond the SM processes as well as for precision SM measurements. Most collisions are soft, with the protons dissolving into mostly low-energy pions that disperse throughout the detector. A typical collision of this sort at the LHC will contribute about 0.6 GeV/rad^2 of energy [1, 2]. Occasionally, one pair of protons within a bunch crossing collides head-on, producing hard, high-energy radiation of interest. At high luminosity, this hard collision, or leading vertex (LV), is always accompanied by soft proton-proton collisions called pileup. The data collected thus far by ATLAS and CMS have approximately 20 pileup collisions per bunch crossing on average ($\langle \text{NPU} \rangle \sim 20$); the data in Run 3 are expected to contain $\langle \text{NPU} \rangle \sim 80$; and the HL-LHC in Runs 4-5 will have $\langle \text{NPU} \rangle \sim 200$. Mitigating the impact of this extra energy on physical observables is one of the biggest challenges for data analysis at the LHC.

The Pileup Mitigation with Machine Learning (PUMML) algorithm was introduced in Ref. [3], building on the jet images paradigm [4, 5] which has found successful application in jet tagging with convolutional neural networks [6, 7] and generation [8, 9]. To apply the convolutional neural network paradigm to cleaning an image itself, we exploit the finer resolution of the tracking detectors relative to the calorimeters of ATLAS and CMS and the fact that tracking information allows charged particles to be identified as coming from either pileup or the leading vertex. We give as input to our network three-channel jet images: one channel for the charged LV particles, one channel for the charge pileup particles, and one channel, at slightly

lower resolution, for the total neutral particles. We then train the network to reconstruct the unknown image for LV neutral particles.

We apply the algorithm to $R = 0.4$ anti-k_t jets. The jet image inputs are square grids in pseudorapidity-azimuth (η, ϕ) space of size 0.9×0.9 centered on the charged leading vertex transverse momentum (p_T)-weighted centroid of the jet. To simulate the detector resolutions of charged and neutral calorimeters, charged images are discretized into $\Delta\eta \times \Delta\phi = 0.025 \times 0.025$ pixels and neutral images are discretized into $\Delta\eta \times \Delta\phi = 0.1 \times 0.1$ pixels. We use the following three input channels:

`red` = the transverse momenta of all neutral particles

`green` = the transverse momenta of charged pileup particles

`blue` = transverse momenta of charged leading vertex particles

The output of our network is also an image:

`output` = the transverse momenta of neutral leading vertex particles.

Only charged particles with $p_T > 500$ MeV were included in the green or blue channels. Charged particles not passing this charged reconstruction cut were treated as if they were neutral particles. The different resolutions for charged and neutral particles initially present a challenge, since standard architectures assume identical resolution for each color channel. To avoid this issue, we perform a direct upsampling of each neutral pixel to 4×4 pixels of size $\Delta\eta \times \Delta\phi = 0.025 \times 0.025$ and divide each pixel value by 16 such that the total momentum in the image is unchanged.

The convolutional neural net architecture used in this study took as input 36×36 pixel, three-channel pileup images. Two convolutional layers, each with 10 filters of size 6×6 with 2×2 strides, were used after zero-padding the input images and first convolutional layer with a 2-pixel buffer on all sides. The output of the second layer has size $9 \times 9 \times 10$, with the 9×9 part corresponding to the size of the target output and the 10 corresponding to the number of filters in the second layer. In order to project down to a $9 \times 9 \times 1$ output, a third convolution layer with filter size 1×1 is used. This last 1×1 convolutional layer is a standard scheme for dimensionality reduction. A rectified linear unit (ReLU) activation function was applied at each stage. A schematic of the framework and architecture is shown in Fig. 1. All neural network implementation and training was performed with the python deep learning libraries Keras [10] and Theano [11]. The dataset consisted of 56k pileup images, with a 90%/10% train/test split. He-uniform initialization [12] was used to initialize the model weights. The neural network was trained using the Adam [13] algorithm with a batch size of 50 over 25 epochs with a learning rate of 0.001. The following loss function was used to train PUMML was a modified per-pixel logarithmic squared loss with a value of $\bar{p} = 10$ GeV:

$$\ell = \left\langle \log \left(\frac{p_T^{(\text{pred})} + \bar{p}}{p_T^{(\text{true})} + \bar{p}} \right)^2 \right\rangle . \tag{1}$$

The PUMML architecture is *local* in that the rescaling of a neutral pixel is a function solely of the information in a patch in (η, ϕ)-space around that pixel. The size of this patch can be controlled by tuning the filter sizes and number of layers in the architecture. Further, due to weight-sharing in convolutional layers, the same function is applied for all pixels. Building this locality and translation invariance into the architecture ensures that the algorithm learns a universal pileup mitigation technique, while carrying the benefit of drastically reducing the number of model parameters. Indeed, the PUMML architecture used in this study has only 4,711 parameters, which is small on the scale of deep learning architectures, but serves to highlight the effectiveness of using modern machine learning techniques (such as convolutional layers) in high energy physics without necessarily using large or deep networks.

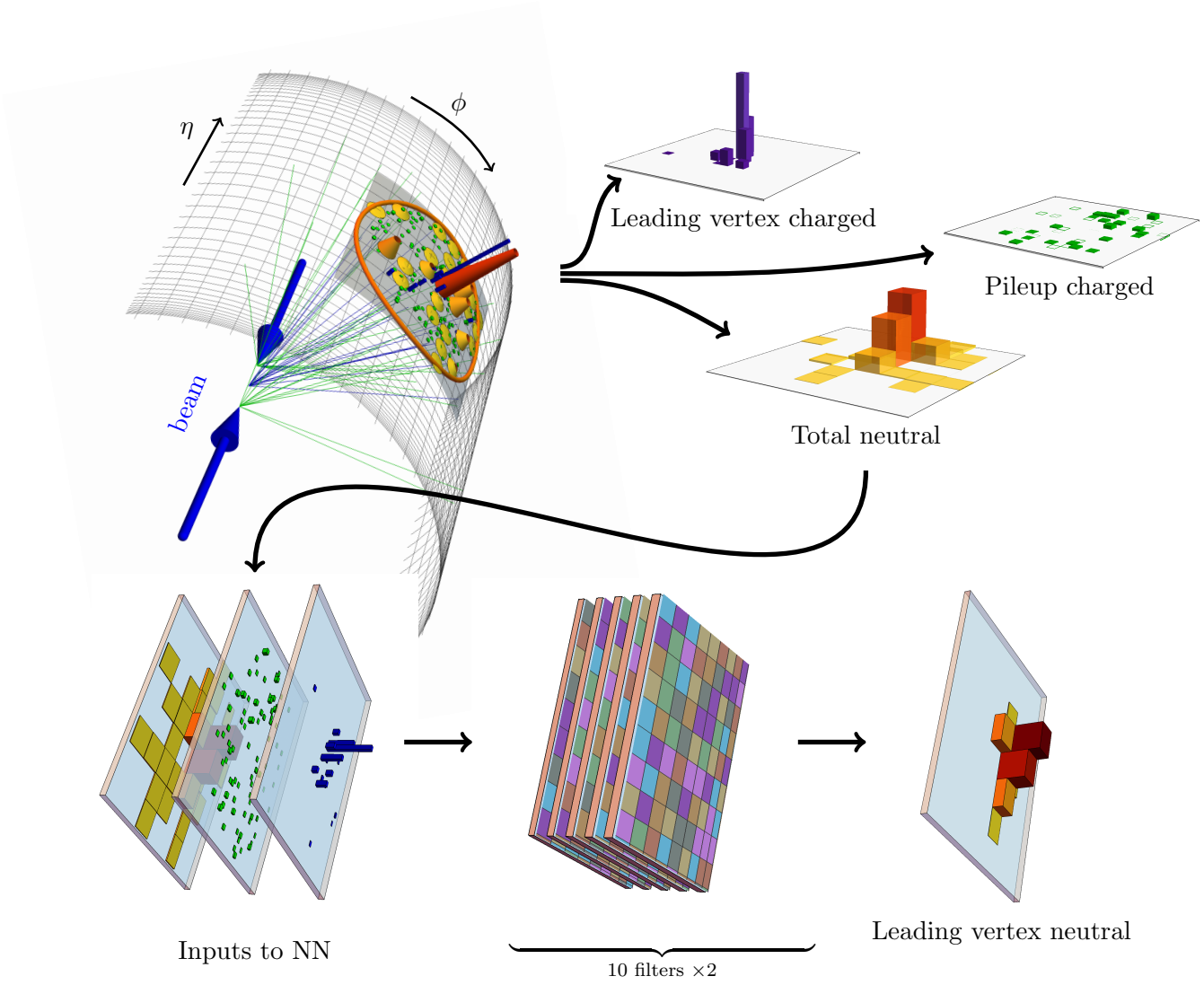

Figure 1: An illustration of the PUMML framework. The input is a three-channel image: blue/purple represents charged radiation from the leading vertex, green is charged pileup radiation, and yellow/orange/red is the total neutral radiation. The resolution of the charged images is higher than for the neutral one. These images are fed into a convolutional layer with several filters whose value at each pixel is a function of a patch around that pixel location in the input images. The output is an image combining the pixels of each filter to one output pixel.

2. Performance

To test the PUMML algorithm, we consider $q\bar{q}$ light-quark-initiated jets coming from the decay of a scalar with mass $m_\phi = 500$ GeV. Events were generated using Pythia 8.183 [14] with the default tune for pp collisions at $\sqrt{s} = 13$ TeV. Pileup was generated by overlaying soft QCD processes onto each event. Final state particles except muons and neutrinos were kept. The events were clustered with FastJet 3.1.3 [15] using the anti-k_t algorithm [16] with a jet radius of $R = 0.4$. A parton-level p_T cut of 95 GeV was applied and up to two leading jets with $p_T > 100$ GeV and $\eta \in [-2.5, 2.5]$ were selected from each event. All particles were taken to be massless.

Samples were generated with the number of pileup vertices ranging from 0 to 180. Since the model must be trained to fix its parameters, the learned model depends on the pileup distribution used for training. For our pileup simulations, we trained on a Poisson distribution of NPUs with mean $\langle \text{NPU} \rangle = 140$. For robustness studies, we also tried training with NPU= 140 for each event or NPU= 20 for each event. For comparison, we show the performance of two powerful and widely used constituent-based pileup mitigation methods: PUPPI [17] and SoftKiller [18]. In both cases, default parameter values were used: $R_0 = 0.3$, $R_{\text{min}} = 0.02$, $w_{\text{cut}} = 0.1$, $p_{\text{T}}^{\text{cut}}(\text{NPU}) = 0.1 + 0.007 \times \text{NPU}$ (PUPPI), grid size = 0.4 (SoftKiller). Variations in the PUPPI parameters did not yield a large difference in performance. Both PUPPI and SoftKiller were implemented at the particle level and then discretized for comparison with PUMML.

To evaluate the performance of different pileup mitigation techniques, we compute several observables and compare the true values to the corrected values of the observables. To facilitate a comparison with PUMML, which outputs corrected neutral calorimeter cells rather than lists of particles, a detector discretization is applied to the true and reconstructed events. Our comparisons focus on the following four jet observables:

- **Jet Mass**: Invariant mass of the leading jet.
- **Dijet Mass**: Invariant mass of the two leading jets.
- **Energy Correlation Functions**, $\text{ECF}_N^{(\beta)}$ [19]: Specifically, we consider the logarithm of the two- and three-point ECFs with $\beta = 4$.

As a measure of pileup mitigation effectiveness, we show the distributions of the per-event percent error in reconstructing the true values of the observables in Fig. 2. A distribution more peaked around zero indicates better reconstruction performance. PUMML outperforms the other pileup mitigation techniques by this metric, successfully reconstructing jet substructure observables such as the jet mass and the energy correlation functions.

3. Opening the black box

While it is generally very difficult to determine what a network is learning, one possible probe is to examine the weights of the filter layers in the convolutional network. For our full network, these weights are complicated and the subtractor that the network learns is difficult to probe analytically. Instead, we trained a simplified PUMML network with a single 12×12 pixel filter, which spans 3×3 neutral pixels, with no bias term. The different channels of this filter are shown in Fig. 3. The neutral filter clearly selects the relevant neutral pixel for subtraction, while the charged pileup filter is approximately uniform (with the value dependent on the specific choice of loss function and activation function), and the charged leading vertex filter does not significantly contribute.

The filter values motivate the following parameterization of what PUMML is learning:

$$p_T^{\text{N,LV}} = 1.0\, p_T^{\text{N,total}} - \beta\, p_T^{\text{C,PU}} + 0.0\, p_T^{\text{C,LV}}, \tag{2}$$

for some $\mathcal{O}(1)$ constant β, where $p_T^{\text{N,LV}}$, $p_T^{\text{N,total}}$, $p_T^{\text{C,PU}}$, and $p_T^{\text{C,LV}}$ are the neutral-pixel-level transverse momenta of the neutral leading-vertex particles, all neutral particles, charged pileup

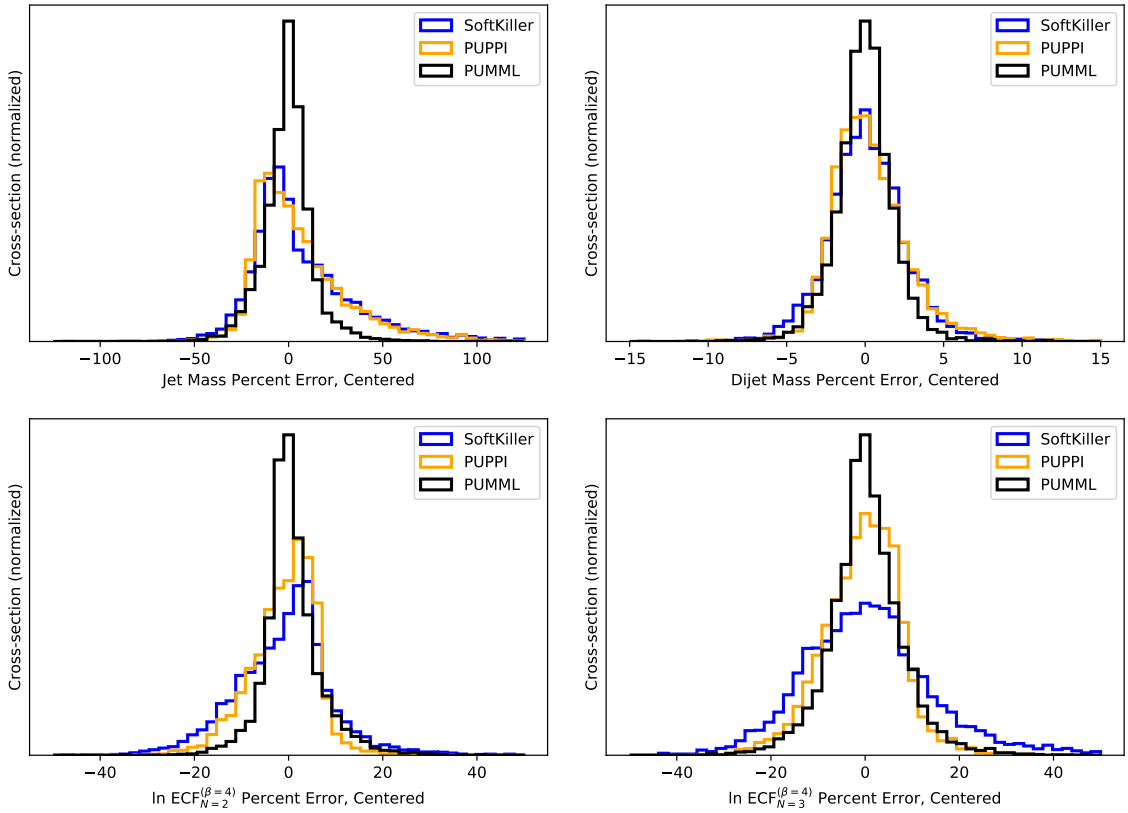

Figure 2: Distributions of the percent error between reconstructed and true values for leading jet mass (top left), dijet mass (top right), ln $\mathrm{ECF}^{(\beta=4)}_{N=2}$ (bottom left), and ln $\mathrm{ECF}^{(\beta=4)}_{N=3}$ (bottom right) for the considered pileup subtraction methods with Poissonian $\langle \mathrm{NPU} \rangle = 140$ pileup. All distributions are centered to have median at 0. The improved reconstruction performance of PUMML is highlighted by its taller and narrower peaks.

particles, and charged leading-vertex particles, respectively. The values 1.0 and 0.0 in Eq. 2 are stable (to the 0.05 level) under variations in the loss and activation functions. This is reassuring as the learned subtractor is thereby robust in the NPU $\to$ 0 limit despite begin trained on $\langle \mathrm{NPU} \rangle = 140$.

Eq. 2 is remarkably similar to the physically-motivated formula used in Jet Cleansing [20]. Cleansing is built on the observation that since pileup is the incoherent sum of many separate scattering events, its variance is smaller than the variance of the radiation from the leading-vertex. Thus, it is better to estimate $p_T^{\mathrm{N,PU}}$ from $p_T^{\mathrm{C,PU}}$ than to estimate $p_T^{\mathrm{N,LV}}$ from $p_T^{\mathrm{C,LV}}$. The simplest form of Cleansing (Linear Cleansing) gives the formula:

$$p_T^{\mathrm{N,LV}} = p_T^{\mathrm{N,tot}} - \left(\frac{1}{\overline{\gamma_0}} - 1 \right) p_T^{\mathrm{C,PU}}, \tag{3}$$

where $\overline{\gamma_0}$ is the average ratio of charged p_T to total p_T in a subjet. Thus this simple one 12×12 filter PUMML network is learning a subtractor of the same parametric form as Linear Cleansing.

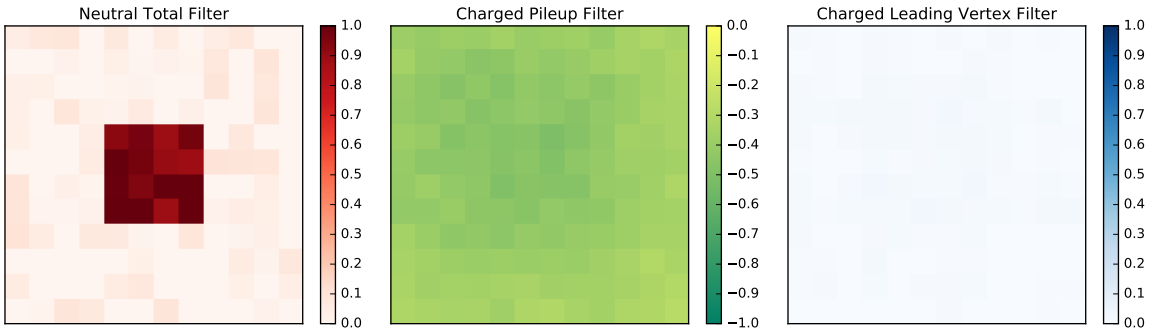

Figure 3: Filter weights for a simple PUMML network with a single 12×12 filter and a ReLU activation function trained with $\langle \text{NPU} \rangle = 140$. The network has selected the relevant neutral pixel, turned off the charged leading vertex contribution, and is using the charged pileup information uniformly.

4. Conclusions

In this paper, we have introduced the first application of machine learning to the critically important problem of pileup mitigation at hadron colliders. We have phrased the problem of pileup mitigation in the language of a machine learning regression problem. The method we introduced, PUMML, takes as input the transverse momentum distribution of charged leading-vertex, charged pileup, and all neutral particles, and outputs the corrected leading vertex neutral energy distribution. We demonstrated that PUMML works at least as well as, and often better than, the competing algorithms PUPPI and SoftKiller in their default implementations.

To prevent the model from learning simulation artifacts, it is preferable to train on actual data rather than simulation. In many machine learning applications in collider physics, obtaining truth-level training samples in data is a substantial challenge. To overcome this challenge in classification tasks, Refs. [21] and [22] introduce an approach to train from mixed samples with different class proportions. For PUMML and pileup mitigation more broadly, a more direct method to train on data is possible. To simulate pileup, we overlay soft QCD events on top of a hard scattering process, both generated with Pythia. Experimentally, there are large samples of minimum bias and zero-bias (i.e. randomly triggered) data. There are also samples of relatively pileup-free events from low luminosity runs. Thus we can construct high-pileup samples using purely data. This kind of *data overlay* approach, which has already been used by experimental groups in other contexts [23, 24], could be perfect for training PUMML with data. Therefore, an implementation of ML-based pileup mitigation in an actual experimental setting could avoid mis-modeling artifacts during training, thus adding more robustness and power to this new tool.

Acknowledgments

The authors would like to thank Philip Harris, Francesco Rubbo, Ariel Schwartzman, Nhan Tran, and Jesse Thaler for helpful conversations. PTK and EMM would like to thank the MIT Physics Department for its support. Computations in this paper were run on the Odyssey cluster supported by the FAS Division of Science, Research Computing Group at Harvard University. This work was supported by the U.S. Department of Energy, Office of Science under contracts DE-AC02-05CH11231 and DE-SC0013607. Additional support was provided by the Harvard Data Science Initiative. Cloud computing resources were provided through a Microsoft Azure for Research award.

References

[1] Khachatryan V *et al.* (CMS) 2017 *JINST* **12** P02014 (*Preprint* 1607.03663)
[2] Aaboud M *et al.* (ATLAS) 2017 (*Preprint* 1703.09665)
[3] Komiske P T, Metodiev E M, Nachman B and Schwartz M D 2017 (*Preprint* 1707.08600)
[4] Cogan J, Kagan M, Strauss E and Schwarztman A 2015 *JHEP* **02** 118 (*Preprint* 1407.5675)
[5] de Oliveira L, Kagan M, Mackey L, Nachman B and Schwartzman A 2016 *JHEP* **07** 069 (*Preprint* 1511.05190)
[6] Komiske P T, Metodiev E M and Schwartz M D 2017 *JHEP* **01** 110 (*Preprint* 1612.01551)
[7] Kasieczka G, Plehn T, Russell M and Schell T 2017 *JHEP* **05** 006 (*Preprint* 1701.08784)
[8] de Oliveira L, Paganini M and Nachman B 2017 (*Preprint* 1701.05927)
[9] Paganini M, de Oliveira L and Nachman B 2017 (*Preprint* 1705.02355)
[10] Chollet F 2015 Keras
[11] Bergstra J, Breuleux O, Bastien F, Lamblin P, Pascanu R, Desjardins G, Turian J, Warde-Farley D and Bengio Y 2010 *Proc. 9th Python in Science Conf* pp 1–7
[12] He K, Zhang X, Ren S and Sun J 2015 *Proceedings of the IEEE international conference on computer vision* pp 1026–1034
[13] Kingma D and Ba J 2014 *arXiv preprint arXiv:1412.6980*
[14] Sjöstrand T, Ask S, Christiansen J R, Corke R, Desai N, Ilten P, Mrenna S, Prestel S, Rasmussen C O and Skands P Z 2015 *Comput. Phys. Commun.* **191** 159–177 (*Preprint* 1410.3012)
[15] Cacciari M, Salam G P and Soyez G 2012 *Eur. Phys. J.* **C72** 1896 (*Preprint* 1111.6097)
[16] Cacciari M, Salam G P and Soyez G 2008 *JHEP* **04** 063 (*Preprint* 0802.1189)
[17] Bertolini D, Harris P, Low M and Tran N 2014 *JHEP* **10** 059 (*Preprint* 1407.6013)
[18] Cacciari M, Salam G P and Soyez G 2015 *Eur. Phys. J.* **C75** 59 (*Preprint* 1407.0408)
[19] Larkoski A J, Salam G P and Thaler J 2013 *JHEP* **06** 108 (*Preprint* 1305.0007)
[20] Krohn D, Schwartz M D, Low M and Wang L T 2014 *Phys. Rev.* **D90** 065020 (*Preprint* 1309.4777)
[21] Dery L M, Nachman B, Rubbo F and Schwartzman A 2017 *JHEP* **05** 145 (*Preprint* 1702.00414)
[22] Metodiev E M, Nachman B and Thaler J 2017 (*Preprint* 1708.02949)
[23] Marshall Z, Collaboration A *et al.* 2014 *Journal of Physics: Conference Series* vol 513 (IOP Publishing) p 022024
[24] Haas A (ATLAS) 2017 Atlas simulation using real data: Embedding and overlay Tech. rep.