Indian Institute of Technology Hyderabad

Clustering Algorithms for Jet Reconstruction in HEP

Abhishek Agarwal
EP17BTECH11001

Under the guidance of Professor Priyotosh Bandhyopadhyay

July 2020

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Clustering Algorithms for Jet Reconstruction in HEP

Abhishek Agarwal
ep17btech11001@iith.ac.in[a,b]

[a]*Department of Physics, Indian Institute of Technology*
[b]*Kandi, Sangareddy, Telangana, India*

Abstract: Hadronic collisions are extremely hard to analyse since they result in jet formation and smear out the final state at LHC like colliders in various ways. For a very long time now, reconstruction of these hadronic jets and their distributions have been vital to uncover the physics of these processes. Here we compare the jet reconstruction results from today's standard and well accepted algorithm to a new algorithm, QuickShift++ (also referred to as QuickShift throughout this report), a graph based clustering method.

1. Introduction

Today's High Energy Particle Physics is driven by simulations and Monte Carlo Event generators. There are a few alternatives which are widely used to simulate and observe the physics of Standard Model and Beyond it as well. In any QCD event generation, quarks from stable hadrons are made to interact with one another at very high energies. The resulting state of particles interacts in complicated ways and forms showers of hadronic particles. These showers are arranged in particular structures or clusters that we call jets. Jets are very vaguely defined so far in the literature, however they give us access to some important observable quantities which characterises the physical process that took place in the collision.

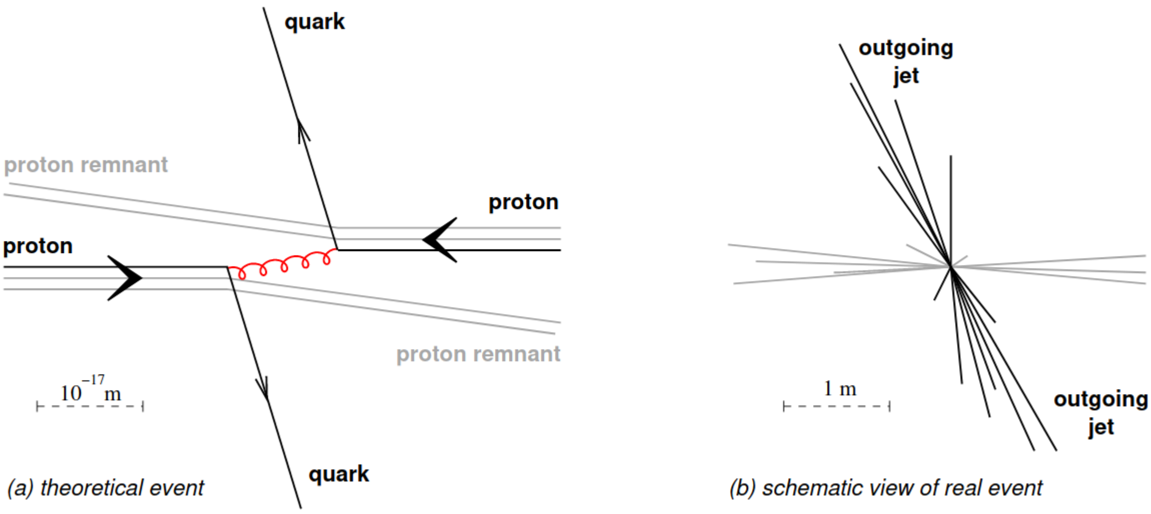


Figure 1: A schematic of a Hard QCD process taken from [1]

1.1. The Standard Model[2]

The Standard Model is the accepted frontier of our understanding of the universe. It is a very successful theory describing the relation and existence of three out of the four fundamental forces of nature. It aims to identify the most elementary particles of the universe and understand how they interact with one another. The Higgs Boson[3] is the latest addition to this theory and explains why and how massive particles are massive. The stunning success of the Standard Model has convinced us that it is not only an important path towards a Grand Unified Theory, but it that it is the correct one.

1.1.1. Elementary Particles

The Standard Model elucidates two kinds of particles that exist at an elementary scale. One are $fermions$, carrying half integer spin and the other are *bosons*, carrying a full integer spin. With the Spin Statistics Theorem, it has been shown that two identical fermions cannot be put into the same space together, they repel by nature. However bosons can be put. This is the primary difference in the behaviour of these particles. Fermions take up space, and are like the bricks of the universe. The bosons do not take up such space and more appropriately are said to govern the interactions between all the particles.

There are four forces in nature, gravitational force, electromagnetic force, the strong nuclear force and weak nuclear force. The Standard Model describes and combines electromagnetic force and weak nuclear force into a single Electro-Weak Force which is mediated by the γ photon, W^+/W^- Bosons, and a Z Boson. It also describes two quarks that comes in three generations which become more massive as we move down the generations. This gives us six quarks. The heaviest of the quarks is a Top Quark. Due to it's special place in the standard model[4], the Top Quark physics provides us insight into physics beyond the Standard Model.

1.1.2. A brief on Detector Phenomenology

In a detector, two particles to be collided are sent at desired energies along the Z axis. The XY plane becomes what is called the transverse plane, and instead of measuring the x and y positions and momenta of the particles we measure the η (pseudorapidity) and ϕ (azimuthal angle), which are all related by the equations:

$$p_x = p_t cos\phi$$
$$p_y = p_t sin\phi$$
$$p_z = p_t sinh\eta$$

Where p_t is the transverse momenta of a particle. These, along with the mass of the particle are related by the relativistic equation $E^2 = |\vec{p}|^2 + m^2$ in the natural units. Energy of particles is measured by layers upon layers of calorimeters, allowing us to estimate the particles total four momenta. In addition to this, very powerful magnetic field curve charges particles allowing us to measure their charge. All of this information is collected for every collision happening at a particle collider and is called an event.

Added to this, heavy and highly energetic particles undergo radiation emission, maybe giving off photons. Quarks undergo hadronization in which heavy and energetic quarks radiate gluons and lighter hadrons. This means that instead of seeing one hadron, we see a bunch of hadrons huddled together in places where we expect a hard quark to land. This bunch of particles forms a shower like canopy and is called a jet. We can most often look at event data and identify these jets. Due to conservation of momentum, the total four momentum of all of the constituents of the jet tells us the four momentum of the initial hard quark correctly. In order to separate these jets in the event from the rest of the event data, we use jet clustering algorithms.

1.2. The Top Quark

The Top quark is very heavy and a very large amount of energy is required to create top quarks in pair. It has spin equal to $\frac{1}{2}$ and electric charge of $+\frac{3}{2}$. Due to it's charge it interacts with the Photon. Being a quark, it can carry color charge and takes part in Strong Nuclear Interaction by mediating gluons. It also changes flavour to a significantly less massive bottom quark while producing a W^+ Boson. This is the primary mode of decay of the top quark.

As can be seen in Figure 2, the Top quark shows up as three distinct high energy jets. The harder the top quark, the closer will the three sub-jets be. The standard model predicts top decay as $t \rightarrow W^+\ b$ with a branching ratio of 99.8%. The W boson further decays either into two leptons, in which case the the entire top quark decays into only one jet without a substructure. The W can also decay into two lighter quarks, in which case the typical three prong jet structure is seen in the data.

2. Jet Clustering in FastJet

The FastJet algorithm, written by Matteo Cacciari, Gavin Salam and Gregory Soyez [5] marked a very important point in Simulation for Physics because it delivered a truly real time and reliable algorithm to construct jets in simulation as well as experimental events. Before them jet clustering was known to be either fast and unreliable, or slow but robust. Their pioneering work in 2006 changed the landscape by demonstrating that Jet clustering could be fast[6].

FastJet comes in built with three jet clustering algorithms. All of them are the same in spirit, with very subtle differences in the metric they use. The metric here is a function of position of two partons which can be used to calculate a distance between them. Formally a metric is:

$$g : \boldsymbol{O_1} \times \boldsymbol{O_2} \to \mathbb{R} \tag{1}$$

where $\boldsymbol{O_i}$ is the set of all observable properties of the i'th parton. Using this metric function, we can measure which particles are "closer" and attach a physical context to this "closeness".

The basic algorithm is outlined here. The inputs to this algorithm are a list of particles in the four-momenta vector space, and an additional jet size parameter R. The first step is to sort the particles in order of pT, call this set $\boldsymbol{P}$. We also define a placeholder particle called beam particle to simplify our notation, $b_i : pT = pT(p_i)$ & η, ϕ $s.t.$ $((\eta(p_i) - \eta(b_i))^2 + (\phi(p_i) - \phi(b_i))^2) = 1$ where $p_i \in \boldsymbol{P}$. Then we iterate until no particles are left:

- Define set $\boldsymbol{D} = \{g(p_i, p_j) : p_i, p_j \in \boldsymbol{P}; i \neq j\} \cup \{g(p_i, b_i) : p_i \in \boldsymbol{P}\}$ and take $d = min(\boldsymbol{D})$.
- If $d = g(p_i, p_j) : p_i, p_j \in \boldsymbol{P}$ and $i \neq j$, recombine p_i & p_j into a particle and reinsert into $\boldsymbol{P}$.
- If $d = pT(p_i)^2 : p_i \in \boldsymbol{P}$, remove p_i from $\boldsymbol{P}$ as a jet J_t.

At the end $\boldsymbol{P}$ is left empty and we have a set of jets $\{J_1, J_2 ...\}$

2.1. $\boldsymbol{K_t}$ Clustering

K_t clustering uses the metric:

$$g(1,2) = min(pT(1)^2, pT(2)^2)((\eta(1) - \eta(2))^2 + (\phi(1) - \phi(2))^2) \tag{2}$$

2.2. anti-$\boldsymbol{K_t}$ Clustering[7]

Anti-K_t clustering uses the metric:

$$g(1,2) = min(pT(1)^{-2}, pT(2)^{-2})((\eta(1) - \eta(2))^2 + (\phi(1) - \phi(2))^2) \tag{3}$$

2.3. Cambridge/Aachen Clustering

Cambridge/Aachen Clustering uses the metric:

$$g(1,2) = (\eta(1) - \eta(2))^2 + (\phi(1) - \phi(2))^2 \tag{4}$$

3. Jet Clustering with newer algorithms

Here we present various alternative clustering algorithms. In this section, each algorithm attempts to assign a list of data points spanning some space with a metric.

3.1. K Means Clustering

K Means clustering is perhaps the oldest clustering algorithm discussed here. It is reasonably fast and reliable. It's drawback is that it requires the number of clusters to be known beforehand. This means that in $eta \times \phi$ space, it needs to know the number of jets before starting.

It starts by assuming that the clusters are randomly spread out. Then it iteratively assigns each data point to the nearest cluster, and the cluster point is shifted to the mean position of the cluster. The iteration stops when there no cluster point has moved since the last iteration.

3.2. Mean Shift Clustering

Mean Shift Clustering is a more complicated and slower version of K Means Clustering. It has been around since at least 1995 when it's application to find modes and clusters in given data were discussed [8]. Since then a lot of research has gone into it but it was deemed to be too slow. Finally in 2018 a hybrid variant of Mean Shift called QuickShift was demonstrated to be fast and reliable [9].

It starts by spreading evenly spreading many jet centres across the space. Then iteratively all jet centres are shifted to the mean position of data points their neighbourhood, combining two centres if they get too close. The iteration ends when no jet centre has moved since the last iteration.

3.3. QuickShift++

The QuickShift++ is an optimised Mean Shift algorithm, which instead of starting with uniformly spaced out jet centres, estimates cluster location using a graph partition algorithm. In 2018 it was demonstrated that using these starting locations, the algorithm performs faster and better clustering. [9]

4. QuickShift++ Algorithm

The fist part of the algorithm is just to estimate the cluster cores using a special and fast density approximation to the actual density of data points. The exact and entire algorithm has been formally defined and analysed[9]. It begins with defined the inputs as a set $X_{[n]} := \{x_1, ..., x_n\}$ having n elements scattered in $\mathbb{R}^d$ on whom a distance can be defined which is at least as good as a Lebesgue measure (A Lebesgue measure of a set is the minimum of sum of all intervals whose union covers the set). This imposes a minimum ordering on the input data. We then need to find the points of highest density in X. To do this we use a special density estimator function:

$$
\begin{aligned}
f_k(x) :&= \frac{k}{n \times v_d \times r_k(x)^d} \ where \\
v_d &= Volume\ of\ a\ unit\ ball\ in\ \mathbb{R}^d \\
r_k(x) :&= Distance\ of\ x\ from\ its\ k'th\ nearest\ neighbour
\end{aligned}
$$

We define cluster cores using the $f_k(x)$ as a density function, every significant the local maxima of $f_k(x)$ is a cluster core. The significance of a cluster core here is characterised by another parameter β. $M \subset X$ is a cluster core if it is a connected component of:

$$\{x \in X : f_k(x) \geq (1-\beta) max(\{f_k(x\prime)\ \forall\ x\prime \in M\})\} \tag{5}$$

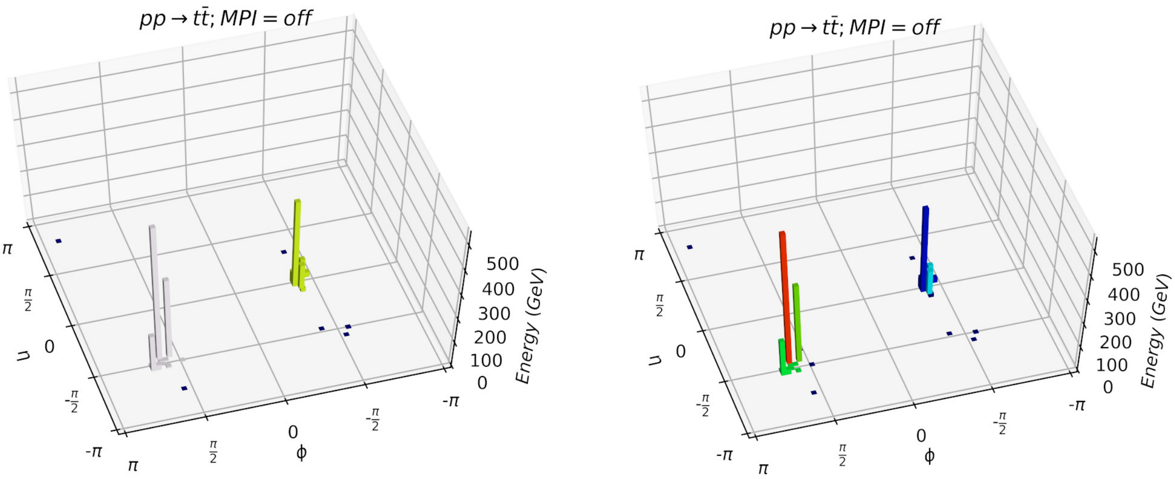
That is M is a cluster core in X if it is itself as ordered as X, and the density $(f_k(x))$ at all points in M is more than or equal to some fraction $(1-\beta)$ of the highest density of all points in it. To assist the process of estimating cluster cores in X, we also define an undirected graph $G(\lambda)$ with vertices $\{x \in X : f_k(x) \geq \lambda\}$ and edges between two points $x, x\prime$ if and only if $g(x, x\prime) \leq min(r_k(x), r_k(x\prime)$. We then sort X in order of $f_k(x)$ and start with an empty set of cluster cores M. Then:

- For each $x_i \in X_{sorted}$, let A be the cluster core that x_i belongs to in $G(\lambda = (1-\beta)(f_k(x_i))$,
- If A is disjoint from every cluster core in M, add A to M.

At the end of this procedure we are left with M, a set of set of points which is our set of estimated cluster cores in X. Then to cluster the remaining points we do a hill climbing step:

- For each point x not already in any cluster, make a path $x \to x\prime$ where $x\prime$ is defined by $g(x, x\prime) = min(g(x, x_i)\ \forall\ x_i \in X)\ and\ f_k(x\prime) > f_x(x)$

This makes a path of increasing density form all points not already in a cluster to a point that lies in some cluster. Thus all the points in QuickShift++ are assigned some cluster.



(a) FastJet anti-k_t at $R = 0.5$

(b) QuickShift++ with $k, \beta = 17, 0.75$

Figure 2: A comparison of FastJet and QuickShift++ Algorithms in a $pp \to t\bar{t}$; $E_cm = 14TeV$, $|\eta| \leq 2.5$, ISR, FSR and Hadronisaiton is on. MPI is off. An initial parton cut of $E_p > 1.5$ GeV and $Jet_{pT} > 20$ GeV was used.

5. Comparison

FastJet takes a single R parameter. It roughly corresponds to the jet size. At higher energies, jets gets more collimated and all the jet constituents are more recognizable at lower R values. The algorithm itself is a simple nearest neighbour clustering with a the metrics discussed above. It runs in < 0.1 *sec* and has been demonstrated to have a time complexity $O(NlogN)$. QuickShift on the other hand is a peak estimating algorithm over a density space formed from the given metric. It runs in < 0.3 *sec*. It has been demonstrated to have a time complexity of $O(kN\alpha(n))$ where α is the Inverse Ackermann function[10]. This time complexity might look like it is faster than FastJet, however given the order of particles in any reasonable jet, k is usually near 20 meaning QuickShift will actually run a little bit slower than FastJet.

The k parameter roughly corresponds to the number of constituents and β corresponds to the densities of particles to be classified as different jets. The QuickShift implementation uses the standard Euclidean Metric over $\eta \times \phi$ which is the same as Cambridge/Aachen Metric, but with a key difference. The C/A algorithm knows that ϕ is circular and posses an extra symmetry given by $f(x + 2\pi) = f(x)$, which however is not built into the QuickShift metric.

Another important difference in these algorithms is the fact that FastJet uses all four components of the four momenta to discern useful information in order to perform robust jet reconstruction. However the QuickShift algorithm only performs clustering in $\eta \times \phi$ space and does not know about the energy and invariant mass of the particles. Despite having less information about the final state of the event and not knowing that ϕ is circular, QuickShift is able to resolve the fat Top Jets into three different sub-jets at 14 TeV. In order of their energy, the sub-jets physically correspond to a b jet that is the direct child of the *top* quark, and two lighter jets coming from the decay of the W boson which is also a direct child of the *top* quark[11].

To do a more direct analysis of jet counting by both the algorithms I simulated $N = 10,000$ $pp \to t\bar{t}$ events at $E_{cm} = 14$ TeV in PYTHIA[12] version 8.24 built with FastJet 3.3.3 version. I also set up the forced decay modes such that the $BranchRatio(t \to bW) = 1.0$ and $BranchRatio(W \to 2\ leptons) = 0.0$. This way we ensure that the system always contains 2 hard and 4 light jets. η and ϕ are rounded off to one decimal place in order to emulate detector level least counts. FastJet at $R = 0.5$ recorded only $23,758$ out of $60,000$ jets while QuickShift++ recorded at least $24,816$ jets in events with less than or equal to 6 jets. Due to using an unphysical metric, QuickShift also reports $4,686$ jets lie in events with more than 6 jets. It is to be noted here that both the algorithms had a $jet_{pT} > 20$ GeV cut as well.

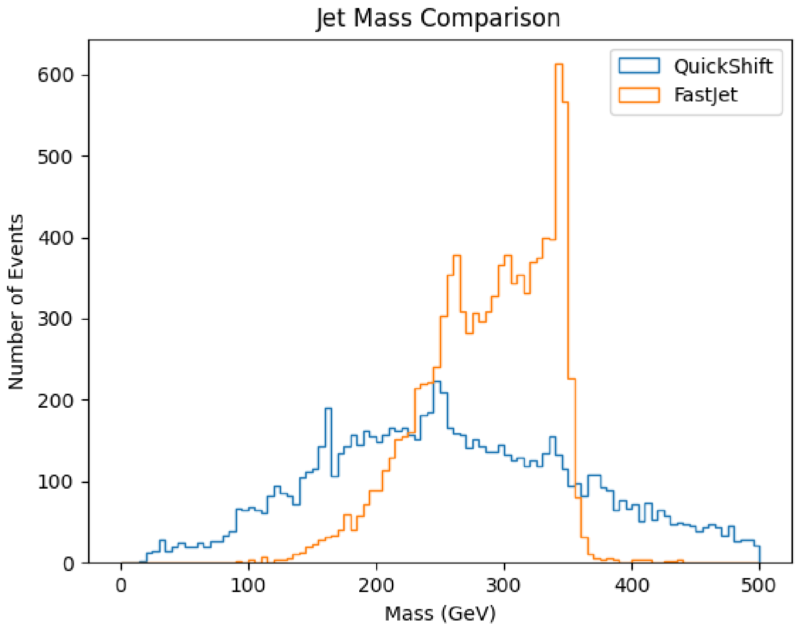


Figure 3: A comparison of FastJet and QuickShift++ Algorithms in terms of jet masses. The orange histogram shows the arithmetic sum of mass of first two jets in order of pT found by FastJet $R = 0.5$. The mass cutoff at $350 GeV$ is unmistakable. The blue histogram is the arithmetic sum of masses of the two ahrdest jets found by QuickShift. The peaks at $m = 170, 250, 345 GeV$ are quite clear. They correspond to a single top quark, one top quark with a W Boson and finally two top quarks. The unusually high spread of the blue curve is due to the following reasons. Data is smeared out by loss precision due to porting event data from C++ code into python code. It is also smeared by the loss of energy as the subject are very often clustered as separate jets by QuickShift.

6. IR Safety

The question of determining whether a jet clustering algorithm is IR Safe or not has been resolved[13]. To test the IR safety of QuickShift clustering, we remove the cut $E_{parton} >= 1.5\ GeV$ and run it on the $10,000$ $pp \to t\bar{t}$ events again. This time QuickShift reported $25,127$ jets in the events with less than or equal to 6 jets. Again it also reports that $5,453$ jets come from events with more than 6 jets. Overall there is a 1.25% increase in jets counted due to soft radiation. While this is not a conclusive proof neither is it a formal demonstration of IR safety, however, it is a strong indication that the QuickShift algorithm if implemented in a way specific to jet finding, would be demonstrably IR safe.

The proof of IR Safety relies on two factors, when testing the algorithm with and without soft particles:

1. All jets found in the event without soft particles will be found also in the event with the soft particles.
2. Any extra jets found in the event with soft particles will themselves be soft, i.e. they will not contain any of the hard particles.

While the first property is not hard to show for QuickShift, the second property becomes a problem. Since the density estimator $f_k(x)$ does not know how hard or soft a point really is, regions with a lot of soft particles will have a higher density than regions with a few hard particles. To some extent this is offset by the fact that any hard particles will in turn produce a large number of softer particles, in real physical events, harder particles will naturally have a higher density of particles. In a real world implementation however we should be able to use a custom density function with a pT or energy term which will help QuickShift separate soft particles from hard particles.

7. Conclusion

FastJet with k_t, anti-k_t and C/A jet finding algorithms has been for a very long time been the standard of jet finding in HEP. However it has been quite some time since then. The last decade has changed the scenario, computation is more affordable than it ever has been and computer science has been making progress with new clustering algorithms taking over. With the advent of more advanced algorithms and the progress in Data Structures, it is incumbent that we use this knowledge and further our efficiency. Investigating newer methods of jet clustering is a relatively cheap and effective way to boost the efficiency of state of the art event classifiers. Pre built and generic implementations of QuickShift provide reasonably good jet finding results as has been demonstrated above. Writing this algorithm to be commonly compatible, faster, more robust and tailoring it to suit the specific needs of jet finding might result in an algorithm which can do better.

References

[1] G. P. Salam, M. Cacciari, Jet clustering in particle physics, via a dynamic nearest neighbour graph implemented with cgal. URL `http://www.lpthe.jussieu.fr/ salam/repository/docs/kt-cgta-v2.pdf`

[2] S. F. Novaes, Standard model: An introduction (2000). arXiv:hep-ph/0001283.

[3] D. Anselmi, On the nature of the higgs boson, Modern Physics Letters A 34 (16) (2019) 1950123. doi:10.1142/s0217732319501232. URL `http://dx.doi.org/10.1142/S0217732319501232`

[4] G. Perez, Top quark theory and the new physics searches frontier, Phys. Scripta T 158 (2013) 014008. doi:10.1088/0031-8949/2013/T158/014008.

[5] M. Cacciari, G. P. Salam, G. Soyez, Fastjet user manual, The European Physical Journal C 72 (3). doi:10.1140/epjc/s10052-012-1896-2. URL `http://dx.doi.org/10.1140/epjc/s10052-012-1896-2`

[6] M. Cacciari, G. P. Salam, Dispelling the n^3 myth for the k_t jet-finder, Physics Letters B 641 (1) (2006) 57–61. doi:10.1016/j.physletb.2006.08.037. URL `http://dx.doi.org/10.1016/j.physletb.2006.08.037`

[7] M. Cacciari, G. P. Salam, G. Soyez, The anti-ktjet clustering algorithm, Journal of High Energy Physics 2008 (04) (2008) 063–063. doi:10.1088/1126-6708/2008/04/063. URL `http://dx.doi.org/10.1088/1126-6708/2008/04/063`

[8] Yizong Cheng, Mean shift, mode seeking, and clustering, IEEE Transactions on Pattern Analysis and Machine intelligence 17 (8) (1995) 790–799.

[9] H. Jiang, J. Jang, S. Kpotufe, Quickshift++: Provably good initializations for sample-based mean shift (2018). arXiv:1805.07909.

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, Introduction to algorithms, MIT press, 2009.

[11] D. E. Kaplan, K. Rehermann, M. D. Schwartz, B. Tweedie, Top tagging: A method for identifying boosted hadronically decaying top quarks, Physical Review Letters 101 (14). doi:10.1103/physrevlett.101.142001. URL `http://dx.doi.org/10.1103/PhysRevLett.101.142001`

[12] T. Sjöstrand, S. Ask, J. R. Christiansen, R. Corke, N. Desai, P. Ilten, S. Mrenna, S. Prestel, C. O. Rasmussen, P. Z. Skands, An introduction to pythia 8.2, Computer Physics Communications 191 (2015) 159–177. doi:10.1016/j.cpc.2015.01.024. URL `http://dx.doi.org/10.1016/j.cpc.2015.01.024`

[13] G. P. Salam, G. Soyez, A practical seedless infrared-safe cone jet algorithm, Journal of High Energy Physics 2007 (05) (2007) 086–086. doi:10.1088/1126-6708/2007/05/086. URL `http://dx.doi.org/10.1088/1126-6708/2007/05/086`