Eur. Phys. J. C (2020) 80:595
https://doi.org/10.1140/epjc/s10052-020-8172-7

THE EUROPEAN PHYSICAL JOURNAL C

Regular Article - Theoretical Physics

Jet transition values for the anti-$k_\perp$ algorithm

Zoltán Szőr[a]

PRISMA Cluster of Excellence, Institut für Physik, Johannes Gutenberg-Universität Mainz, 55099 Mainz, Germany

Received: 10 December 2019 / Accepted: 20 June 2020 / Published online: 2 July 2020

Abstract We define jet transition values for the anti-$k_\perp$ algorithm for both hadron and e^+e^- colliders. We show how these transition values can be computed and how they can be used to improve the performance of clusterization when jet resolution parameters are varied over a larger set of values. Finally we present a simple performance test to illustrate the behavior of the new method compared to the original one.

1 Introduction

The production of hadronic jets is a common feature of particle collisions. Jets are widely studied, as they can be used to test the standard model and measure its parameters, they can signal new physics, and provide important background for new physics searches as well.

Jets are defined through jet clustering algorithms: they take final state particles as an input and combine them according to their prescription into larger objects, what we then call jets. The algorithms have a set of resolution parameters, which defines the jet structure: fixing the values of jet algorithm parameters determines what happens in each step of the clusterization, what particles get combined into jets eventually. Although jet algorithms are required in all kind of jet analysis, one particularly important observable is the so-called jet rate. The jet rate, as a function of its parameters, directly connects to the clustering algorithm, as it provides useful information about how the number of jets depends on the choice of parameters.

Jet rate measures the relative production rate of n-jets compared to all hadronic events. It is given by the ratio of the n-jet cross section $\sigma_{\text{n-jet}}$ and the total hadronic cross section σ_{tot} at center-of-mass energy Q^2:

$$R_n(\mathbf{a}) = \frac{\sigma_{\text{n-jet}}(\mathbf{a})}{\sigma_{\text{tot}}}, \tag{1}$$

where $\mathbf{a}$ denotes the set of jet resolution parameters characteristic to a given jet algorithm. Jet rates are mostly studied as a function of one or more of their resolution parameters. This means clustering the same set of momenta with a wide range of chosen values of the jet resolution parameters. This set of momenta might represent a point in the phase space of final state particles or a physical event, but the actual representation is not important in the scope of the paper. Thereby we use the umbrella term ‘partonic event’.

Since repeated clusterization is usually computationally inefficient, in practice one tries to exploit the properties of the algorithm to enhance performance in computations. This is also important on the theory side, since making higher order predictions in perturbation theory typically requires the generation of millions of phase space points, which all need to be clustered individually. Although the bulk of computational cost is coming from the calculation of amplitudes and subtraction terms, slow clusterization might add a non-negligible time contribution as well.

In the case of e^+e^- colliders the most common jet clustering algorithm is the $k_\perp$ (or Durham) algorithm [1], which has auspicious properties to do computations efficiently, illustrated in the next section. Today, in the LHC era, the commonly used jet algorithm is the anti-$k_\perp$ algorithm [2]. Though jet rate studies similar to $k_\perp$ ones are not prevalent, they are also hampered by the lack of properties that the $k_\perp$ has. Hence computations can be slowed down significantly due to clusterization only.

In this paper we present a reformulation of the anti-$k_\perp$ algorithm equivalent to the original, which makes it possible to define transition values in a similar fashion to the $k_\perp$ algorithm. Furthermore we show how these transition values can be computed and used to speed up calculations. Our method can be used for both hadron and e^+e^- colliders.

[a] e-mail: zoltanszoer@uni-mainz.de (corresponding author)

2 The $k_\perp$ algorithm

We start with a short review of the $k_\perp$ algorithm, and discuss how it is used in calculations in practice. The algorithm depends on a single jet resolution parameter y_{cut} and the distance measure is defined as

$$y_{ij} = \frac{2\min(E_i^2, E_j^2)(1-\cos\theta_{ij})}{Q^2}. \tag{2}$$

E_i and E_j denote the energy of particle i and j respectively, while θ_{ij} labels the angle between the three-momenta $\mathbf{p}_i$ and $\mathbf{p}_j$. During clusterization we compute y_{ij} for each pair of particles and find the smallest one $y_{kl} = \min y_{ij}$. If $y_{kl} < y_{\mathrm{cut}}$ holds we combine particles k and l, then start the procedure again with the new list of objects. Otherwise we stop the clusterization and the resulting objects are considered jets.

In the case of the $k_\perp$ algorithm one can uniquely define transition values. Transition values $y_{i-1\leftarrow i}$ are certain values of y_{cut}, where the number of jets changes from i into $i-1$ for a given final state configuration. The distribution of the transition value behaves as an event shape observable. Using the $k_\perp$ algorithm every transition value $y_{i-1\leftarrow i}$ can be computed performing the clusterization only once independently of y_{cut}, such that in every clusterization step the smallest y_{kl} value provides the corresponding $y_{i-1\leftarrow i}$ transition value. We repeat the steps until all particles are clustered into two jets. When jets are defined through the $k_\perp$ algorithm the number of jets is a monotonically decreasing function of y_{cut} for every possible partonic event.

These two properties of the algorithm described previously make possible to connect the $\mathrm{d}\sigma/\mathrm{d}y_{i-1\leftarrow i}$ differential distributions and the $\sigma_{\text{n-jet}}(y_{\mathrm{cut}})$ cross section. For example the three-jet cross section can be computed as

$$\sigma_{3-jet}(y_{\mathrm{cut}}) = \int_{y_{\mathrm{cut}}}^{1} \mathrm{d}y_{2\leftarrow 3}\frac{\mathrm{d}\sigma}{\mathrm{d}y_{2\leftarrow 3}} - \int_{y_{\mathrm{cut}}}^{1} \mathrm{d}y_{3\leftarrow 4}\frac{\mathrm{d}\sigma}{\mathrm{d}y_{3\leftarrow 4}}. \tag{3}$$

The meaning of the two terms are the following: the three-jet cross section for a chosen y_{cut} gets contributions from the $\mathrm{d}\sigma/\mathrm{d}y_{2\leftarrow 3}$ differential cross section for every $y_{2\leftarrow 3}$ value which is greater than y_{cut}. This gives the first term in Eq. (3). However the resulting quantity in itself would include all events with $y_{3\leftarrow 4} \in [0,1]$. Events with $y_{3\leftarrow 4} \in [0, y_{\mathrm{cut}}]$ are indeed events which cluster into three-jets, however for events with $y_{3\leftarrow 4} \in [y_{\mathrm{cut}}, 1]$ clustering stops at four-jets. Thus we need to subtract the integrated $\mathrm{d}\sigma/\mathrm{d}y_{3\leftarrow 4}$ distribution to get the correct three-jet cross section, which gives the second term. Equation (3) provides a very useful relation to speed up numerical calculations. One has to perform the clusterization only once per partonic event, calculate the differential cross sections, then do a simple integration with the desired y_{cut} according to the formula to obtain the n-jet cross section.

3 The anti-$k_\perp$ algorithm

Now we turn our interest towards the anti-$k_\perp$ algorithm and discuss its shortcomings in computational time compared to the $k_\perp$ algorithm. The anti-$k_\perp$ algorithm uses two different measures: a two-particle measure d_{ij} and a beam jet measure d_{iB}. They are defined as

$$\begin{aligned} d_{ij} &= \min(k_{\perp,i}^{2p}, k_{\perp,j}^{2p})\frac{(y_i-y_j)^2+(\phi_i-\phi_j)^2}{R^2}, \\ d_{iB} &= k_{\perp,i}^{2p}, \end{aligned} \tag{4}$$

for hadron colliders, where $k_{\perp,i}$, y_i and ϕ_i denote the transverse momentum, rapidity and azimuth of particle i respectively. In the case of e^+e^- colliders we have

$$\begin{aligned} d_{ij} &= \min(E_i^{2p}, E_j^{2p})\frac{(1-\cos\theta_{ij})}{1-\cos R}, \\ d_{iB} &= E_i^{2p}, \end{aligned} \tag{5}$$

with the notation being identical to the one introduced in the previous section. Choosing $p = -1, 0, 1$ we obtain the anti-$k_\perp$ [2], the Cambridge/Aachen [3] and the inclusive $k_\perp$ [1] algorithms respectively. Collectively they are named as the general inclusive $k_\perp$ algorithm.

The anti-$k_\perp$ algorithm has two jet resolution parameters: R and E_{cut}. During clustering we calculate d_{iB} for every particle i and d_{ij} for every particle pair i, j. If d_{kl} is the smallest measure, we combine particle k, l, but if d_{kB} is the smallest one, particle k is considered a jet candidate, and we remove it from the list of objects. We repeat these steps until every particle becomes part of a jet candidate. Finally we apply energy cut(s), and every jet candidate with $E_i > E_{\mathrm{cut}}$ is a resolved jet.

The anti-$k_\perp$ algorithm has characteristics and properties, which makes it preferable for experimental use [2], for example cone-like jet shapes. However the algorithm has certain other properties, which unfortunately make computational shortcuts like Eq. (3) absent, therefore making clusterization more expensive in the study of the jet rate observable. This is due to the fact that in general the number of jets is not a monotonic function of R^2 or $1-\cos R$ as it can be seen in Fig. 1. The reason is partially the presence of the additional E_{cut} parameter. Although we obtain more and more jet candidates when we increase the spatial resolution, many of them would not survive the last cut on the energy. Furthermore the presence of the beam jet measure, d_{iB} prevents the same definition of jet transition values as in the case of the $k_\perp$ algorithm.

This would leave us in an unfortunate situation where clustering would need to be done for each different choice of the jet resolution parameters, in particular when we vary R. We note that this still can be an issue, even if one uses the improved version of the anti-$k_\perp$ algorithm [4], which scales

$\mathcal{O}(N \log N)$ compared to the $\mathcal{O}(N^3)$ cost of the original formulation. The choice of the efficient method depends on the number of histogram bins and the number of partons to be clustered.

Fortunately we can still define jet transition values, which can be used in calculations.

4 Transition values

We start with an equivalent reformulation of the anti-$k_\perp$ algorithm, which is more suitable to define and find transition values. First we combine the two measures d_{ij} and d_{iB} the following way

$$y_{ijk} \equiv y_{\text{cut}} \frac{\min_{i,j} d_{ij}}{\min_k d_{kB}}, \tag{6}$$

where we define $y_{\text{cut}} \equiv R^2$ and $y_{\text{cut}} \equiv 1 - \cos R$ for hadron and e^+e^- colliders respectively. Note that y_{ijk} is independent of y_{cut}.

Now clusterization is done as it follows: first we calculate y_{ijk}. If $y_{ijk} < y_{\text{cut}}$, we combine particle i, j; otherwise we consider particle k to be a jet candidate and remove it from the list. We repeat the procedure until the list is empty. Finally we apply the energy cuts on our jet candidates.

The clustering procedure is now similar to the $k_\perp$ algorithm, hence we can define jet transition values in a similar fashion. We call $y_t \equiv y_{\text{cut}}$ a transition value when the clustered particle configuration changes. It is important to notice that it does not necessarily imply a change in the number of jet candidates. Two different y_{cut} values can result in the same number of jet candidates, but these candidates may differ in their momenta configuration. As an illustration let us consider the following: we have 4 partons such that they can be separated into two hemispheres and we cluster them into 3 jets. The parton in the first hemisphere is the hardest and is widely separated in angle from the other three, we label it by H. In the second hemisphere one parton is soft and two are hard, labeled as s, h and h' respectively, with the following angle separation $1-\cos\theta_{sh} \sim 1-\cos\theta_{sh'} > 1-\cos\theta_{hh'}$. If we choose y_{cut} such that $1-\cos\theta_{sh} > y_{\text{cut}} > 1-\cos\theta_{hh'}$, then in the clustering process we first remove the soft parton s from the list, then combine partons h and h' together and finally obtain (s), (hh'), (H) as jet candidates. In an other case with $y_{\text{cut}} > 1-\cos\theta_{sh}$ we first combine partons s and h, and obtain (sh), (h'), (H) as jet candidates. Both configurations have the same number of jet candidates, but the way each parton is associated to a jet is different, hence they are in two different regions separated by a transition value. This behavior is due to the presence of the beam jet measure d_{iB}. Then the final number of resolved jets depends on the chosen value of E_{cut} as well.

Using this definition is convenient in practice. The transition values must be calculated only once, then one can apply as many different energy cuts as wanted without repeating the clusterization again. Nevertheless the calculation of y_t values is not straightforward. For the $k_\perp$ algorithm the sequence of clustering is independent of y_{cut} and relevant information can be fully retrieved for any y_{cut} value from one complete clusterization. In contrast, the clusterization sequence of the anti-$k_\perp$ algorithm depends on the actual choice of y_{cut}, due to the presence of the two different distance measures.

It was shown that in the Cambridge algorithm one faces a similar problem, but transition values can still be found systematically [5]. Here we can adopt the method of Ref. [5] as well to find transition values for the anti-$k_\perp$ algorithm in the following way:

1. First set an initial value for y_{ini} and set $y_{\text{cut}} = y_{\text{ini}}$.
2. If y_{cut} is less than some preset lower limit y_{stop}, stop the algorithm.
3. Perform clusterization with the chosen y_{cut}, and find the maximum value of y_{ijk} during the process.
4. Store the transition value $y_t = y_{ijk}^{max}$ and apply energy cuts to obtain the corresponding number of jets.
5. Set $y_{\text{cut}} = y_{ijk}^{max}$ and go to Step 2.

Clusterization between two transition values is completely determined, choosing two different y_{cut} in this set will lead to the same jet configuration. This leads to an improvement in speed in the calculation of jet rates. We can fill histograms more easily between two transition values, we do not have to consider each bin separately and perform clusterization over and over again.

It is worth to mention that the method is independent of the definition of d_{ij} and d_{iB} and also independent of how d_{ij} and d_{iB} are calculated, given that the maximum value of y_{ijk} can be obtained during clusterization. Therefore the transition method can be used both in the hadron and e^+e^- collider version of the anti-$k_\perp$ algorithm and in fact for any version of the general inclusive $k_\perp$ algorithm, that being the original $\mathcal{O}(N^3)$ or the improved $\mathcal{O}(N \log N)$ version.

On Fig. 1 we show the number of jets as a function of y_{cut}. We used a randomly generated partonic event with 10 particles in the final state at $\sqrt{Q^2} = 100\,\text{GeV}$ center-of-mass energy. E_{cut} was chosen 8 GeV. The 10 particle configuration was clustered with the e^+e^- version of the anti-$k_\perp$ algorithm using both approaches: bin-by-bin with 30 y_{cut} values denoted by blue dots and via the transition values method denoted by the red line. Both methods produce identical results, but while the bin-by-bin method required 30 repeated full clusterizations, the red curve was reproduced from 14 transition values. Figure 1 also illustrates the general non-monotonic behavior of the number of jets as a function of y_{cut}.

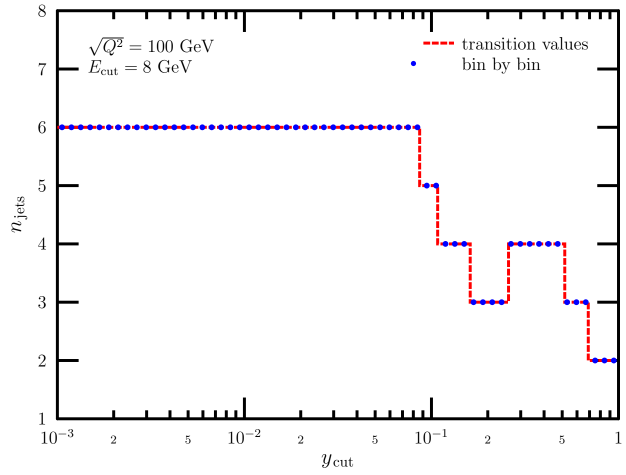

Fig. 1 The number of jets as function of $y_{\rm cut}$ obtained from clustering a randomly generated partonic event with 10 particles in two different ways. The two approaches provide identical results and the non-monotonic behavior of function is also visible

5 Performance

Finally we explore the performance of the new method compared to the traditional approach. We employ three different methods: we name the method computing the number of jets over a wide range of $y_{\rm cut}$ through transition values as *transition*, while the bin-by-bin version is dubbed as *direct*. Both the *transition* and the *direct* method are based on the original formulation of anti-$k_\perp$, which scales as $\mathcal{O}(N^3)$. As third we include the `FJcore` version of the anti-$k_\perp$ algorithm from the `FastJet` package [6], and perform bin-by-bin clustering with it. We call this method *fjcore*. We note that the `FJcore` package provides only a $\mathcal{O}(N^2)$ scaling in contrast to the full `FastJet` version scaling as $\mathcal{O}(N \log N)$. In exchange `FJcore` is easier to integrate and is more likely to be used in cases, where the whole apparatus of `FastJet` is not required. We implemented the *transition* and the *direct* methods in a `Fortran90` program and included the `FJcore` algorithm through the provided wrapper. For simplicity we chose the e^+e^- collider version of the anti-$k_\perp$ algorithm. Using `RAMBO` [7] we generated 1000 partonic events with 5, 10, 15 and 20 particles in the final state, and clustered them with all three methods. We checked that the three methods give the same results, as it is illustrated in Fig. 1. To perform clusterization with the *direct* and *fjcore* methods we selected 30, 60, 90 and 120 bins for $y_{\rm cut}$, the first number of bins being closer to experimental setups, while the last one is more typical for theoretical predictions. In the *transition* method $y_{\rm ini}$ was always set to the largest value of $y_{\rm cut}$ of the histogram, while $y_{\rm stop}$ was chosen to be the smallest. This way we ensured that the range of search for transition values coincides with the range of the histograms.

We summarize our results in Table 1.

The computations were performed on a simple everyday laptop. We emphasize that our numbers in Table 1 are shown just to illustrate the behavior of the new method compared to the usual one, it is not an exhaustive study on performance. For example fluctuations in computational time were not taken into account. Nevertheless Table 1 still provides useful information about how the *transition* method performs.

As we can see the timing of the *direct* and the *fjcore* methods scale with the number of bins, as one would expect it. The numbers indicate a linear relation. The *transition* method depends non-linearly on the number of particles, as more particles introduce more and more possible final jet configurations, hence more transition values to compute. This method also depends on the range of $y_{\rm cut}$ values. Although a large number of particles would mean plenty of transition values, many of them could fall outside of the range of interest, hence they would be not computed in the end. For large number of partons the *fjcore* method is the best, however there is a turnover at 10 partons, where the *transition* method starts to take over and for 5 partons it clearly outperforms the other two methods, an order of magnitude speed up can be achieved. It is even more obvious when the number of bins is larger. Interestingly at low multiplicities *fjcore* is the slowest, which is probably due to the complexity of the algorithm.

Table 1 clearly shows that the *transition* method can be used to improve the speed of clustering in the calculation of fixed order parton level distributions, like jet rates. The calculation of fixed order predictions typically involve only a small number of strongly interacting final state particles, but a large number of bins in order to produce smooth histogram curves. In addition, millions of phase space points are generated, which all require clusterization, therefore faster methods are preferred.

We note that according to Table 1, the *transition* method is always faster than the *direct* method, when the number of bins is large and the multiplicity is moderate or small. As mentioned earlier both methods employ a ‘naive’ $\mathcal{O}(N^3)$ clusterization. Our new method does not depend on whether the clusterization is done via a ‘naive’ $\mathcal{O}(N^3)$ or an improved $\mathcal{O}(N \log N)$ algorithm, given that the value of y_{ijk}^{max} can be tracked during repeated clustering. Hence we expect the integration of the *transition* method into the `FastJet` framework to be possible.

6 Summary

In this paper we defined transition values for the anti-$k_\perp$ algorithm and we presented a way to compute them. The knowledge of these values can speed up computations, which involve large number of variations of the $y_{\rm cut}$ jet parameter. Our simple performance test shows that the new method could be applied best to improve performance significantly in

Table 1 Required time to perform clusterization of 1000 partonic events using the three different methods. Time values are shown is seconds. Various number of particles and bins were used to illustrate performance behavior

Partons	Method	30 bins (s)	60 bins (s)	90 bins (s)	120 bins (s)
5	direct	0.103	0.203	0.285	0.369
	fjcore	0.167	0.287	0.406	0.532
	transition	0.021	0.023	0.023	0.023
10	direct	0.548	1.062	1.578	2.091
	fjcore	0.234	0.421	0.580	0.777
	transition	0.296	0.304	0.299	0.307
15	direct	1.635	3.134	4.532	6.161
	fjcore	0.313	0.601	0.887	1.220
	transition	1.458	1.740	1.722	1.746
20	direct	3.878	6.930	10.219	13.548
	fjcore	0.455	0.896	1.376	1.927
	transition	5.357	5.986	5.924	5.900

the calculation of fixed order predictions for jet rates with the anti-$k_\perp$ algorithm, which might regain interest in the upcoming precision era and future electron-positron colliders. The definition of transition values could also serve as starting point for the development of new observables. Our method can be used both for the hadron and the e^+e^- collider version of the anti-$k_\perp$ algorithm, in fact for any version of the general inclusive $k_\perp$ algorithm. Furthermore the new transition method can be combined either with the 'naive' $\mathcal{O}(N^3)$ or the improved $\mathcal{O}(N \log N)$ clusterization method, hence it is compatible with the `FastJet` framework.

Acknowledgements We would like to thank Stefan Weinzierl for his useful comments on the manuscript.

Data Availability Statement This manuscript has no associated data or the data will not be deposited. [Authors' comment: The manuscript contains no data to be deposited.]

Funded by SCOAP^3.

References

1. S. Catani, Y.L. Dokshitzer, M. Olsson, G. Turnock, B.R. Webber, Phys. Lett. B **269**, 432 (1991). https://doi.org/10.1016/0370-2693(91)90196-W
2. M. Cacciari, G.P. Salam, G. Soyez, JHEP **04**, 063 (2008). https://doi.org/10.1088/1126-6708/2008/04/063
3. Y.L. Dokshitzer, G.D. Leder, S. Moretti, B.R. Webber, JHEP **08**, 001 (1997). https://doi.org/10.1088/1126-6708/1997/08/001
4. M. Cacciari, G.P. Salam, Phys. Lett. B **641**, 57 (2006). https://doi.org/10.1016/j.physletb.2006.08.037
5. S. Bentvelsen, I. Meyer, Eur. Phys. J. C **4**, 623 (1998). https://doi.org/10.1007/s100520050232
6. M. Cacciari, G.P. Salam, G. Soyez, Eur. Phys. J. C **72**, 1896 (2012). https://doi.org/10.1140/epjc/s10052-012-1896-2
7. R. Kleiss, W.J. Stirling, S.D. Ellis, Comput. Phys. Commun. **40**, 359 (1986). https://doi.org/10.1016/0010-4655(86)90119-0