

Quantum clustering and jet reconstruction at the LHC

Jorge J. Martínez de Lejarza
Jorge.M.Lejarza@ific.uv.es

Based on: J. J. M. de Lejarza, L. Cieri and G. Rodrigo, [arxiv:2204.06496](https://arxiv.org/abs/2204.06496)

IFIC-Universitat de València/CSIC

9th July 2022, International Conference on High Energy Physics



VNIVERSITAT
ID VALÈNCIA



CSIC
CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS

Outline

- 1 Motivation
- 2 Quantum algorithms
 - Quantum subroutine to compute a Minkowski-type distance
 - Quantum maximum search by amplitude encoding
- 3 Quantum clustering algorithms
 - Quantum K-means
 - Quantum Affinity Propagation
 - Quantum k_T jet algorithm
- 4 Conclusions

Outline

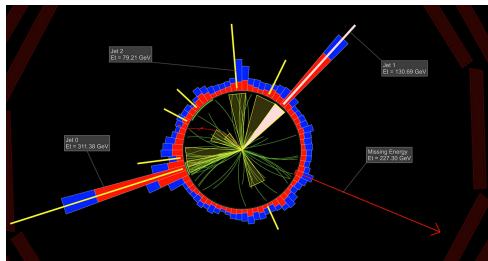
- 1 Motivation
- 2 Quantum algorithms
 - Quantum subroutine to compute a Minkowski-type distance
 - Quantum maximum search by amplitude encoding
- 3 Quantum clustering algorithms
 - Quantum K-means
 - Quantum Affinity Propagation
 - Quantum k_T jet algorithm
- 4 Conclusions

Motivation

Current status of jet clustering in High Energy Physics

Situation

- Analysing HEP collisions \rightarrow one of the most computationally demanding activities
- Identifying **jets** formed \rightarrow daunting task which consumes a great deal of resources \rightarrow ML and Classical algorithms
 - Difficulty is expected to increase \rightarrow with the High Luminosity LHC (HL-LHC)

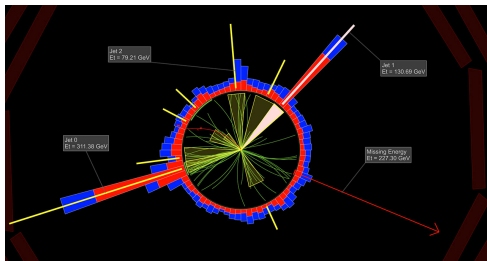


Motivation

Current status of jet clustering in High Energy Physics

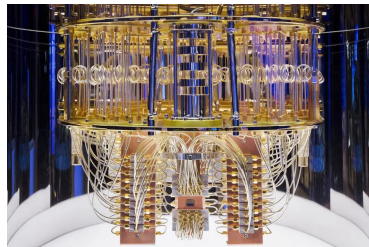
Situation

- Analysing HEP collisions \rightarrow one of the most computationally demanding activities
- Identifying **jets** formed \rightarrow daunting task which consumes a great deal of resources \rightarrow ML and Classical algorithms
 - Difficulty is expected to increase \rightarrow with the High Luminosity LHC (HL-LHC)



Possible solution

- What if we might speed up jet clustering algorithms using:
Quantum Computing?



Outline

1 Motivation

2 Quantum algorithms

- Quantum subroutine to compute a Minkowski-type distance
- Quantum maximum search by amplitude encoding

3 Quantum clustering algorithms

- Quantum K-means
- Quantum Affinity Propagation
- Quantum k_T jet algorithm

4 Conclusions

Quantum Algorithms

Quantum subroutine to compute a Minkowski-type distance

Previous approach

Euclidean distance

- For computing the quantum Euclidean distance between two d -dimensional vectors \mathbf{x}_1 , \mathbf{x}_2 , classical information must be encoded:

$$|x_i\rangle = |\mathbf{x}_i|^{-1} \sum_{\mu=1}^d x_{i,\mu} |\mu\rangle, \quad i = 1, 2$$

Quantum Algorithms

Quantum subroutine to compute a Minkowski-type distance

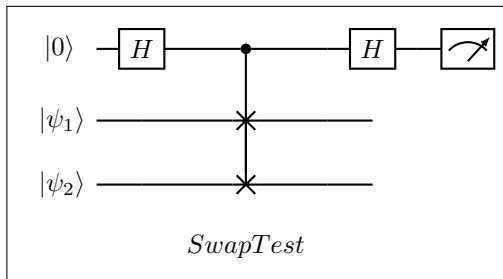
Previous approach

Euclidean distance

- For computing the quantum Euclidean distance between two d -dimensional vectors \mathbf{x}_1 , \mathbf{x}_2 , classical information must be encoded:

$$|x_i\rangle = |\mathbf{x}_i|^{-1} \sum_{\mu=1}^d x_{i,\mu} |\mu\rangle, \quad i = 1, 2$$

- Then, we can use the following quantum circuit:



[Buhrman, Cleve, Watrous, de Wolf \(2001\)](#)

Quantum Algorithms

Quantum subroutine to compute a Minkowski-type distance

Previous approach

Euclidean distance

- Where define the following states:

$$|\psi_1\rangle = \frac{1}{\sqrt{2}} (|0, x_1\rangle + |1, x_2\rangle) \quad |\psi_2\rangle = \frac{1}{\sqrt{Z_{12}}} (|\mathbf{x}_1||0\rangle - |\mathbf{x}_2||1\rangle)$$

$$|\psi'_1\rangle = \frac{1}{\sqrt{2}} (|x_1, 0\rangle + |x_2, 1\rangle), \quad Z_{12} = |\mathbf{x}_1|^2 + |\mathbf{x}_2|^2$$

- We can compute:

$$\langle\psi'_1|\psi_2\rangle\langle\psi_2|\psi_1\rangle = \frac{1}{2Z_{12}} |\mathbf{x}_1 - \mathbf{x}_2|^2$$

Quantum Algorithms

Quantum subroutine to compute a Minkowski-type distance

Previous approach

Euclidean distance

Flow chart:

1. Initialization:

$$|\Psi_0\rangle = |0\rangle \otimes |\psi_1\rangle \otimes |\psi_2\rangle = |0, \psi_1, \psi_2\rangle$$

2. Applying Hadamard gate H :

$$|\Psi_1\rangle = \left(H \otimes I^{\otimes n+1}\right) |\Psi_0\rangle = \frac{1}{\sqrt{2}} (|0, \psi_1, \psi_2\rangle + |1, \psi_1, \psi_2\rangle)$$

3. Applying the CSWAP gate:

$$|\Psi_2\rangle = \text{CSWAP} |\Psi_1\rangle = \frac{1}{\sqrt{2}} (|0, \psi_1, \psi_2\rangle + |1, \psi_2, \psi'_1\rangle)$$

4. Applying Hadamard gate H :

$$|\Psi_3\rangle = \left(H \otimes I^{\otimes n+1}\right) |\Psi_2\rangle = \frac{1}{2} (|0\rangle (|\psi_1, \psi_2\rangle + |\psi_2, \psi'_1\rangle) + |1\rangle (|\psi_1, \psi_2\rangle - |\psi_2, \psi'_1\rangle))$$

5. Measurement:

$$P_{\Psi_3}(|0\rangle) = |\langle 0 | \Psi_3 \rangle|^2 = \frac{1}{2} + \frac{1}{2} \langle \psi'_1 | \psi_2 \rangle \langle \psi_2 | \psi_1 \rangle$$

Quantum Algorithms

Quantum subroutine to compute a Minkowski-type distance

Previous approach

Euclidean distance

- Once the probability $P_{\Psi_3}(|0\rangle)$ has been estimated, we can combine:

$$\left\{ \begin{array}{l} \langle \psi'_1 | \psi_2 \rangle \langle \psi_2 | \psi_1 \rangle = \frac{1}{2Z_{12}} |\mathbf{x}_1 - \mathbf{x}_2|^2 \\ P_{\Psi_3}(|0\rangle) = |\langle 0 | \Psi_3 \rangle|^2 = \frac{1}{2} + \frac{1}{2} \langle \psi'_1 | \psi_2 \rangle \langle \psi_2 | \psi_1 \rangle \end{array} \right.$$

Quantum Algorithms

Quantum subroutine to compute a Minkowski-type distance

Previous approach

Euclidean distance

- Once the probability $P_{\Psi_3}(|0\rangle)$ has been estimated, we can combine:

$$\left\{ \begin{array}{l} \langle \psi'_1 | \psi_2 \rangle \langle \psi_2 | \psi_1 \rangle = \frac{1}{2Z_{12}} |\mathbf{x}_1 - \mathbf{x}_2|^2 \\ P_{\Psi_3}(|0\rangle) = |\langle 0 | \Psi_3 \rangle|^2 = \frac{1}{2} + \frac{1}{2} \langle \psi'_1 | \psi_2 \rangle \langle \psi_2 | \psi_1 \rangle \end{array} \right.$$



$$d_E^{(Q)}(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{2Z_{12}(2P_{\Psi_3}(|0\rangle) - 1)}$$

Quantum Algorithms

Quantum subroutine to compute a Minkowski-type distance

Previous approach

Euclidean distance

- Once the probability $P_{\Psi_3}(|0\rangle)$ has been estimated, we can combine:

$$\begin{cases} \langle \psi'_1 | \psi_2 \rangle \langle \psi_2 | \psi_1 \rangle = \frac{1}{2Z_{12}} |\mathbf{x}_1 - \mathbf{x}_2|^2 \\ P_{\Psi_3}(|0\rangle) = |\langle 0 | \Psi_3 \rangle|^2 = \frac{1}{2} + \frac{1}{2} \langle \psi'_1 | \psi_2 \rangle \langle \psi_2 | \psi_1 \rangle \end{cases}$$



$$d_E^{(Q)}(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{2Z_{12}(2P_{\Psi_3}(|0\rangle) - 1)}$$

Could this
provide a
speed-up?

Quantum Algorithms

Quantum subroutine to compute a Minkowski-type distance

Previous approach

Euclidean distance

- Once the probability $P_{\Psi_3}(|0\rangle)$ has been estimated, we can combine:

$$\begin{cases} \langle \psi'_1 | \psi_2 \rangle \langle \psi_2 | \psi_1 \rangle = \frac{1}{2Z_{12}} |\mathbf{x}_1 - \mathbf{x}_2|^2 \\ P_{\Psi_3}(|0\rangle) = |\langle 0 | \Psi_3 \rangle|^2 = \frac{1}{2} + \frac{1}{2} \langle \psi'_1 | \psi_2 \rangle \langle \psi_2 | \psi_1 \rangle \end{cases}$$



$$d_E^{(Q)}(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{2Z_{12}(2P_{\Psi_3}(|0\rangle) - 1)}$$

Could this
provide a
speed-up?

QRAM

Quantum Random Access Memory

Classical computation: $\mathcal{O}(d)$
Quantum computation: $\mathcal{O}(\log d)$

Quantum Algorithms

Quantum subroutine to compute a Minkowski-type distance

Our approach

Invariant sum squared

[JML, Cieri, Rodrigo \(2022\)](#)

- The invariant sum squared (a.k.a invariant mass squared) is:

$$s_{12} = (x_{0,1} + x_{0,2})^2 - |\mathbf{x}_1 + \mathbf{x}_2|^2$$

- We have to use the SwapTest twice (spatial and temporal part):

Spatial part

$$|\psi_2\rangle \longrightarrow |\psi_2\rangle = \frac{1}{\sqrt{Z_{12}}} (|\mathbf{x}_1||0\rangle + |\mathbf{x}_2||1\rangle)$$

Temporal part

$$\left\{ \begin{array}{l} |\varphi_1\rangle = H|0\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \\ |\varphi_2\rangle = \frac{1}{\sqrt{Z_0}} (x_{0,1}|0\rangle + x_{0,2}|1\rangle) \\ Z_0 = x_{0,1}^2 + x_{0,2}^2 \end{array} \right.$$

Quantum Algorithms

Quantum subroutine to compute a Minkowski-type distance

Our approach

Invariant sum squared

[JML, Cieri, Rodrigo \(2022\)](#)

- The invariant sum squared (a.k.a invariant mass squared) is:

$$s_{12} = (x_{0,1} + x_{0,2})^2 - |\mathbf{x}_1 + \mathbf{x}_2|^2$$

- We have to use the SwapTest twice (spatial and temporal part):

Spatial part

$$|\psi_2\rangle \longrightarrow |\psi_2\rangle = \frac{1}{\sqrt{Z_{12}}} (|\mathbf{x}_1||0\rangle + |\mathbf{x}_2||1\rangle)$$



$$|\mathbf{x}_1 + \mathbf{x}_2|^2 = 2Z_{12}(2P_{\Psi_3}(|0\rangle)_{\text{spatial}}) - 1$$

Temporal part

$$\begin{cases} |\varphi_1\rangle = H|0\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \\ |\varphi_2\rangle = \frac{1}{\sqrt{Z_0}} (x_{0,1}|0\rangle + x_{0,2}|1\rangle) \\ Z_0 = x_{0,1}^2 + x_{0,2}^2 \end{cases}$$



$$(x_{0,1} + x_{0,2})^2 = 2Z_0(2P_{\Psi_3}(|0\rangle)_{\text{time}}) - 1$$

Quantum Algorithms

Quantum subroutine to compute a Minkowski-type distance

Our approach

Invariant sum squared

[JML, Cieri, Rodrigo \(2022\)](#)

- The invariant sum squared (a.k.a invariant mass squared) is:

$$s_{12} = (x_{0,1} + x_{0,2})^2 - |\mathbf{x}_1 + \mathbf{x}_2|^2$$

- We have to use the SwapTest twice (spatial and temporal part):

Spatial part

$$|\psi_2\rangle \longrightarrow |\psi_2\rangle = \frac{1}{\sqrt{Z_{12}}} (|\mathbf{x}_1\rangle|0\rangle + |\mathbf{x}_2\rangle|1\rangle)$$



Temporal part

$$\begin{cases} |\varphi_1\rangle = H|0\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \\ |\varphi_2\rangle = \frac{1}{\sqrt{Z_0}} (x_{0,1}|0\rangle + x_{0,2}|1\rangle) \\ Z_0 = x_{0,1}^2 + x_{0,2}^2 \end{cases}$$



$$|\mathbf{x}_1 + \mathbf{x}_2|^2 = 2Z_{12}(2P_{\Psi_3}(|0\rangle|_{\text{spatial}}) - 1) \quad (x_{0,1} + x_{0,2})^2 = 2Z_0(2P_{\Psi_3}(|0\rangle|_{\text{time}}) - 1)$$

$$s_{12}^{(Q)} = 2(Z_0(2P_{\Psi_3}(|0\rangle|_{\text{time}}) - 1) - Z_{12}(2P_{\Psi_3}(|0\rangle|_{\text{spatial}}) - 1))$$

Quantum Algorithms

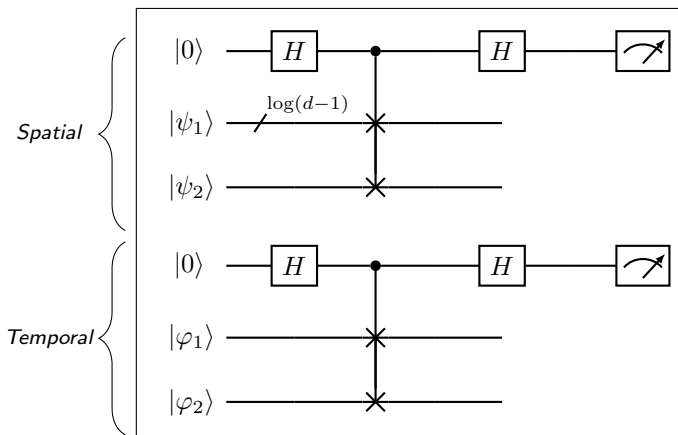
Quantum subroutine to compute a Minkowski-type distance

Our approach

Invariant sum squared

[JML, Cieri, Rodrigo \(2022\)](#)

- The general quantum circuit to compute the invariant sum squared is:



Quantum Algorithms

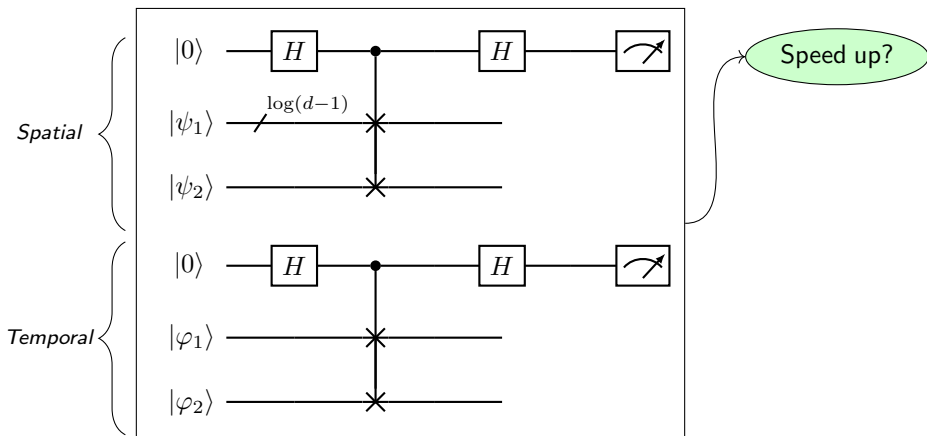
Quantum subroutine to compute a Minkowski-type distance

Our approach

Invariant sum squared

[JML, Cieri, Rodrigo \(2022\)](#)

- The general quantum circuit to compute the invariant sum squared is:



Quantum Algorithms

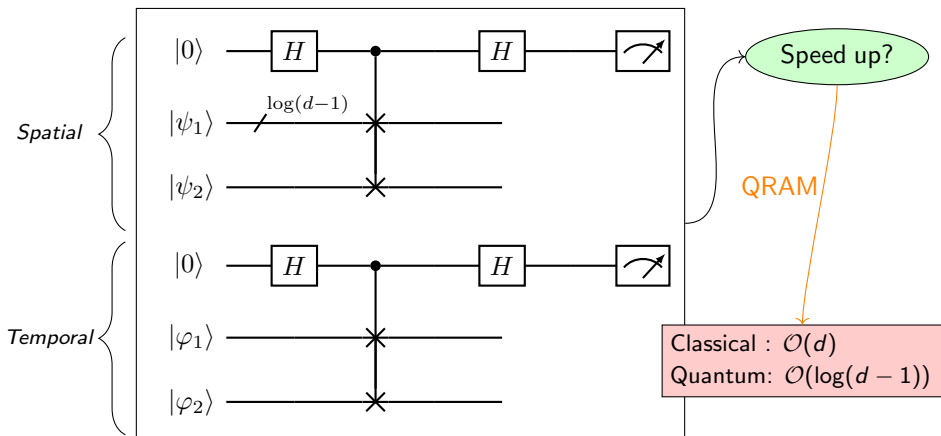
Quantum subroutine to compute a Minkowski-type distance

Our approach

Invariant sum squared

[JML, Cieri, Rodrigo \(2022\)](#)

- The general quantum circuit to compute the invariant sum squared is:



Quantum Algorithms

Quantum maximum search by amplitude encoding

[JML, Cieri, Rodrigo \(2022\)](#)

Let $L[0, \dots, N-1]$ be an unsorted list of N items. The quantum algorithm to find the maximum using amplitude encoding proceeds in two steps:

- 1 The list of N elements is encoded into a $\log_2(N)$ qubits state as follows:

$$|\Psi\rangle = \frac{1}{\sqrt{L_{sum}}} \sum_{j=0}^{N-1} L[j] |j\rangle ,$$

where $L_{sum} = \sum_{j=0}^{N-1} L[j]^2$ is a normalization constant.

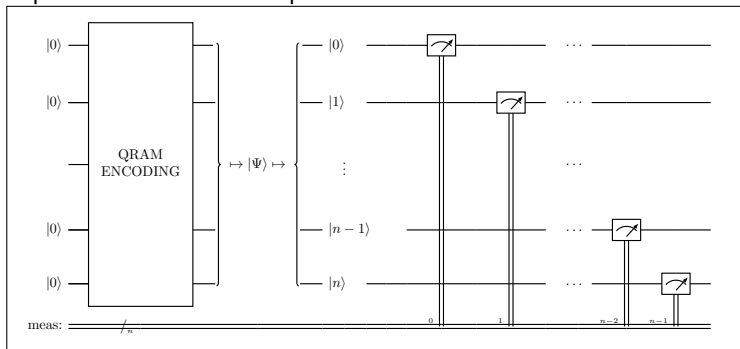
- 2 The final state is measured. This step is rerun several times to reduce the statistical uncertainty. Once done, the most repeated state gives us the maximum.

Quantum Algorithms

Quantum maximum search by amplitude encoding

[JML, Cieri, Rodrigo \(2022\)](#)

- The quantum circuit of this procedure is:

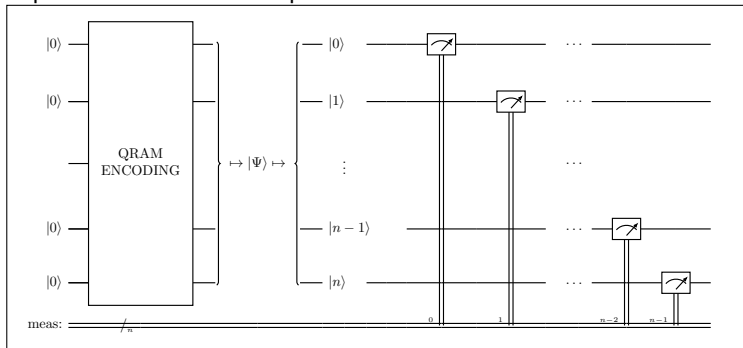


Quantum Algorithms

Quantum maximum search by amplitude encoding

[JML, Cieri, Rodrigo \(2022\)](#)

- The quantum circuit of this procedure is:



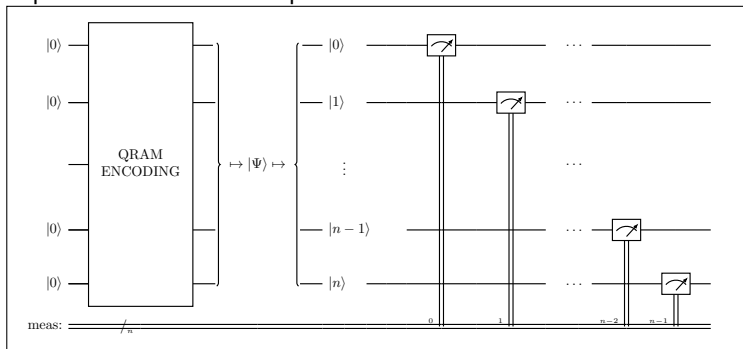
Speed up?

Quantum Algorithms

Quantum maximum search by amplitude encoding

[JML, Cieri, Rodrigo \(2022\)](#)

- The quantum circuit of this procedure is:



Speed up?

GRAM

Classical : $\mathcal{O}(N)$

Quantum: $\mathcal{O}(\log N)$

Outline

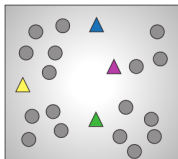
- 1 Motivation
- 2 Quantum algorithms
 - Quantum subroutine to compute a Minkowski-type distance
 - Quantum maximum search by amplitude encoding
- 3 Quantum clustering algorithms
 - Quantum K-means
 - Quantum Affinity Propagation
 - Quantum k_T jet algorithm
- 4 Conclusions

Quantum clustering algorithms

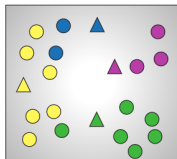
Quantum K-means

K-means workflow

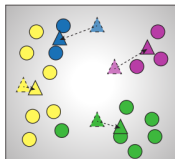
[MacQueen \(1967\)](#)



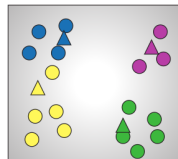
1. Randomly generate K initial centroids within the data domain (here $K=4$, represented by triangles).



2. Assign every point (represented by circles) to the corresponding nearest centroid (assignment represented through colors).



3. Recalculate the new K centroids by computing the mean of each cluster of points.



4. Repeat steps 2 and 3 until centroids stabilize, and convergence has been reached.

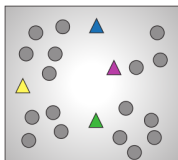
[Pires, Bargassa, Seixas, Omar \(2021\)](#)

Quantum clustering algorithms

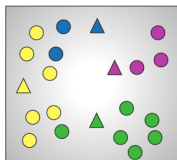
Quantum K-means

K-means workflow

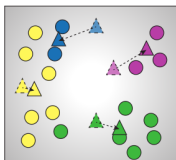
[MacQueen \(1967\)](#)



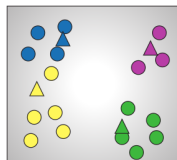
1. Randomly generate K initial centroids within the data domain (here $K=4$, represented by triangles).



2. Assign every point (represented by circles) to the corresponding nearest centroid (assignment represented through colors).



3. Recalculate the new K centroids by computing the mean of each cluster of points.



4. Repeat steps 2 and 3 until centroids stabilize, and convergence has been reached.

[Pires, Bargassa, Seixas, Omar \(2021\)](#)

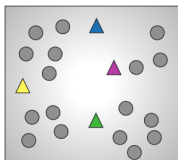
- Step 2 includes two procedures that might be speed up [JML, Cieri, Rodrigo \(2022\)](#)
 - Computing the distances \rightarrow Quantum invariant sum squared \rightarrow From $\mathcal{O}(d)$ to $\mathcal{O}(\log(d-1))$
 - Assigning the nearest centroid (obtaining a minimum) \rightarrow Quantum maximum search \rightarrow From $\mathcal{O}(K)$ to $\mathcal{O}(\log K)$

Quantum clustering algorithms

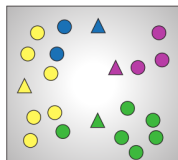
Quantum K-means

K-means workflow

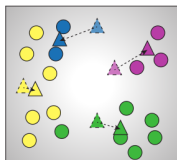
[MacQueen \(1967\)](#)



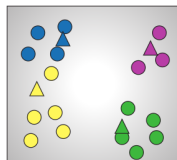
1. Randomly generate K initial centroids within the data domain (here $K=4$, represented by triangles).



2. Assign every point (represented by circles) to the corresponding nearest centroid (assignment represented through colors).



3. Recalculate the new K centroids by computing the mean of each cluster of points.



4. Repeat steps 2 and 3 until centroids stabilize, and convergence has been reached.

[Pires, Bargassa, Seixas, Omar \(2021\)](#)

- Step 2 includes two procedures that might be speed up

[JML, Cieri, Rodrigo \(2022\)](#)

- Computing the distances \rightarrow Quantum invariant sum squared \rightarrow

From $\mathcal{O}(d)$ to $\mathcal{O}(\log(d-1))$

- Assigning the nearest centroid (obtaining a minimum) \rightarrow Quantum maximum search \rightarrow From $\mathcal{O}(K)$ to $\mathcal{O}(\log K)$

The total speed-up

Classical: $\mathcal{O}(NKd)$
Quantum: $\mathcal{O}(N \log K \log(d-1))$

Quantum clustering algorithms

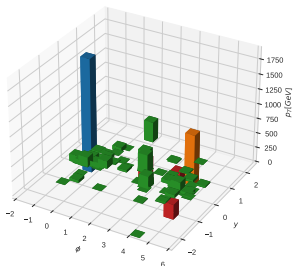
Quantum K-means

[JML, Cieri, Rodrigo \(2022\)](#)

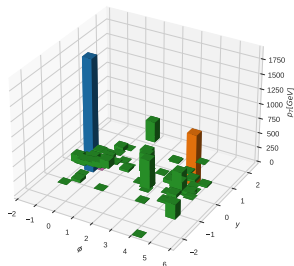
K-means quantum simulations



- LHC simulated-data:



Classical K-means



Quantum K-means, $\varepsilon_c = 0.94$

$$\varepsilon_c = \frac{\# \text{ part. classified as the classical algorithm}}{\# \text{ part. in total}}$$

Quantum clustering algorithms

Quantum Affinity Propagation

Affinity Propagation algorithm

[Frey, Dueck \(2007\)](#)

- Main ideas:
 - Does **not need** the number of clusters to be defined **beforehand**
 - Consider all data points as exemplars → they are reduced until reaching the optimal number

Quantum clustering algorithms

Quantum Affinity Propagation

Affinity Propagation algorithm

[Frey, Dueck \(2007\)](#)

- Main ideas:
 - Does **not need** the number of clusters to be defined **beforehand**
 - Consider all data points as exemplars \rightarrow they are reduced until reaching the optimal number
- Input \rightarrow similarity matrix \rightarrow metric $\left\{ \begin{array}{l} \text{Most cases: Euclidean distance} \\ \text{Our case: Invariant sum squared} \end{array} \right.$

Quantum clustering algorithms

Quantum Affinity Propagation

Affinity Propagation algorithm

[Frey, Dueck \(2007\)](#)

- Main ideas:
 - Does **not need** the number of clusters to be defined **beforehand**
 - Consider all data points as exemplars \rightarrow they are reduced until reaching the optimal number
- Input \rightarrow similarity matrix \rightarrow metric $\left\{ \begin{array}{l} \text{Most cases: Euclidean distance} \\ \text{Our case: Invariant sum squared} \end{array} \right.$
- Quantum advantage? \rightarrow computing Quantum invariant sum squared \rightarrow
From $\mathcal{O}(d)$ to $\mathcal{O}(\log(d-1))$

[JML, Cieri, Rodrigo \(2022\)](#)

Quantum clustering algorithms

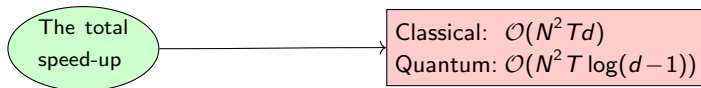
Quantum Affinity Propagation

Affinity Propagation algorithm

[Frey, Dueck \(2007\)](#)

- Main ideas:
 - Does **not need** the number of clusters to be defined **beforehand**
 - Consider all data points as exemplars \rightarrow they are reduced until reaching the optimal number
- Input \rightarrow similarity matrix \rightarrow metric
 - Most cases: Euclidean distance
 - Our case: Invariant sum squared
- Quantum advantage? \rightarrow computing Quantum invariant sum squared \rightarrow
 From $\mathcal{O}(d)$ to $\mathcal{O}(\log(d-1))$

[JML, Cieri, Rodrigo \(2022\)](#)

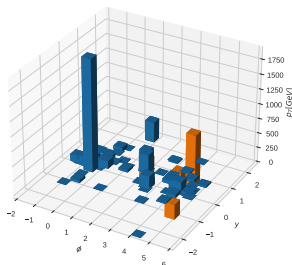


Quantum clustering algorithms

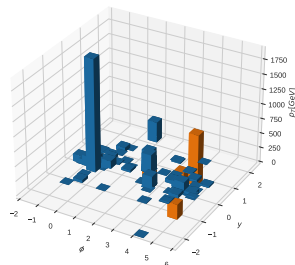
Quantum Affinity propagation

[JML, Cieri, Rodrigo \(2022\)](#) **Affinity Propagation quantum simulations**  

- LHC simulated-data:




Classical Affinity Propagation



Quantum Affinity Propagation, $\epsilon_c = 1.00$

Quantum clustering algorithms

Quantum Affinity propagation

[JML, Cieri, Rodrigo \(2022\)](#) **Affinity Propagation quantum simulations**  

- LHC simulated-data:



Quantum clustering algorithms

Quantum k_T jet algorithm

k_T jet algorithm

[Catani, Dokshitzer, Olsson, Turnock, Webber \(1991\)](#)

[Cacciari, Salam, Soyez \(2008\)](#)

- 1 • For each pair of partons i, j compute:

$$d_{ij} = \min(p_{T,i}^{2p}, p_{T,j}^{2p}) \Delta R_{ij}^2 / R^2, \text{ with } \Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j)^2$$

where $p_{T,i}$, y_i and ϕ_i are the transverse momentum (with respect to the beam direction), rapidity and azimuth of particle i .

- For each particle i the beam distance is $d_{iB} = p_{T,i}^{2p}$.
- 2 Find d_{min} amongst d_{ij} , d_{iB} .
 - If d_{ij} , the particles i and j are merged
 - If d_{iB} , declare i as a final jet and remove it from the list of particles
 - 3 Repeat from step 1 until no particles left.

Quantum clustering algorithms

Quantum k_T jet algorithm

k_T jet algorithm

[Catani, Dokshitzer, Olsson, Turnock, Webber \(1991\)](#)

[Cacciari, Salam, Soyez \(2008\)](#)

- 1 • For each pair of partons i, j compute:

$$d_{ij} = \min(p_{T,i}^{2p}, p_{T,j}^{2p}) \Delta R_{ij}^2 / R^2, \text{ with } \Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j)^2$$

where $p_{T,i}$, y_i and ϕ_i are the transverse momentum (with respect to the beam direction), rapidity and azimuth of particle i .

- For each particle i the beam distance is $d_{iB} = p_{T,i}^{2p}$.
- 2 Find d_{min} amongst d_{ij} , d_{iB} .
 - If d_{ij} , the particles i and j are merged
 - If d_{iB} , declare i as a final jet and remove it from the list of particles
- 3 Repeat from step 1 until no particles left.
- Step 2 includes finding a minimum in a list of order $N \rightarrow$ Quantum maximum search \rightarrow From $\mathcal{O}(N)$ to $\mathcal{O}(\log N)$

[JML, Cieri, Rodrigo \(2022\)](#)

Quantum clustering algorithms

Quantum k_T jet algorithm

k_T jet algorithm

[Catani, Dokshitzer, Olsson, Turnock, Webber \(1991\)](#)

[Cacciari, Salam, Soyez \(2008\)](#)

- 1 • For each pair of partons i, j compute:

$$d_{ij} = \min(p_{T,i}^{2p}, p_{T,j}^{2p}) \Delta R_{ij}^2 / R^2, \text{ with } \Delta R_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j)^2$$

where $p_{T,i}$, y_i and ϕ_i are the transverse momentum (with respect to the beam direction), rapidity and azimuth of particle i .

- For each particle i the beam distance is $d_{iB} = p_{T,i}^{2p}$.
- 2 Find d_{min} amongst d_{ij} , d_{iB} .
 - If d_{ij} , the particles i and j are merged
 - If d_{iB} , declare i as a final jet and remove it from the list of particles
 - 3 Repeat from step 1 until no particles left.
 - Step 2 includes finding a minimum in a list of order $N \rightarrow$ Quantum maximum search \rightarrow From $\mathcal{O}(N)$ to $\mathcal{O}(\log N)$

[JML, Cieri, Rodrigo \(2022\)](#)



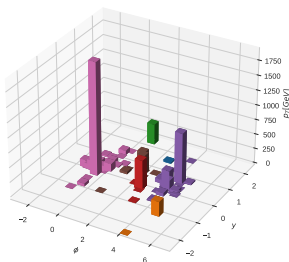
Quantum clustering algorithms

Quantum k_T jet algorithm

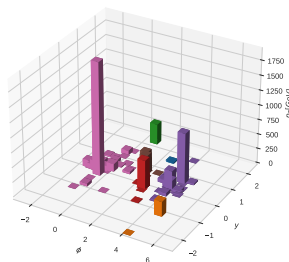
[JML, Cieri, Rodrigo \(2022\)](#) k_T -jet algorithm quantum simulations



- LHC simulated-data, $p = 1$, k_T :



Classical k_T , $R = 1$



Quantum k_T , $R = 1$, $\epsilon_c = 0.98$

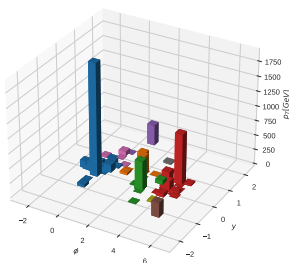
Quantum clustering algorithms

Quantum k_T jet algorithm

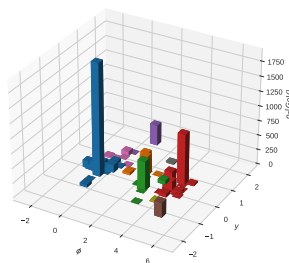
[JML, Cieri, Rodrigo \(2022\)](#) k_T -jet algorithm quantum simulations



- LHC simulated-data, $p = -1$, anti- k_T :



Classical anti- k_T , $R = 1$



Quantum anti- k_T , $R = 1$, $\varepsilon_c = 0.99$

Quantum clustering algorithms

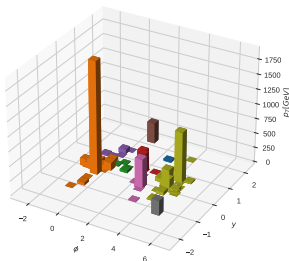
Quantum k_T jet algorithm

[JML, Cieri, Rodrigo \(2022\)](#)

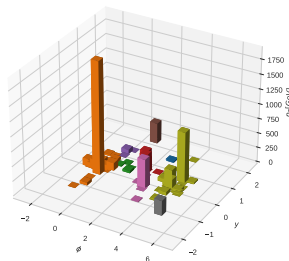
k_T -jet algorithm quantum simulations



- LHC simulated-data, $p = 0$, Cam/Aachen:



Classical Cam/Aachen, $R = 1$



Quantum Cam/Aachen, $R = 1$, $\epsilon_c = 0.98$

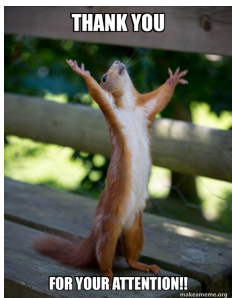
Outline

- 1 Motivation
- 2 Quantum algorithms
 - Quantum subroutine to compute a Minkowski-type distance
 - Quantum maximum search by amplitude encoding
- 3 Quantum clustering algorithms
 - Quantum K-means
 - Quantum Affinity Propagation
 - Quantum k_T jet algorithm
- 4 Conclusions

Conclusions

- Quantum computing to **speed-up** jet clustering algorithms
- Two procedures:
 - Quantum **distance** \rightarrow invariant sum (mass) squared \rightarrow by **SwapTest**
 - Quantum **maximum search** \rightarrow by **Amplitude Encoding**
- Proven achievements in LHC simulated data:
 - Quantum algorithms **as good as** classical
- When **QRAM devices exist** one would obtain
 - Quantum K-means \rightarrow From $\mathcal{O}(NKd)$ to $\mathcal{O}(N \log K \log(d-1))$
 - Quantum Affinity Propagation \rightarrow From $\mathcal{O}(N^2 Td)$ to $\mathcal{O}(N^2 T \log(d-1))$
 - Quantum $k_T \rightarrow \begin{cases} \text{From } \mathcal{O}(N^2) \text{ to } \mathcal{O}(N \log N) & \text{(without Voronoi diagrams)} \\ \text{From } \mathcal{O}(N \log N) \text{ to } \mathcal{O}(N \log N) & \text{(with Voronoi diagrams)} \end{cases}$
- What if **QRAM never exists** \rightarrow other data loading methods
 - Cut-off of Grover-Rudolph From $\mathcal{O}(2^n)$ to $\mathcal{O}(2^{k_0(\epsilon)})$ [Marin, Gonzalez-Conde, Sanz \(2021\)](#)
 - qGANs From $\mathcal{O}(2^n)$ to $\mathcal{O}(\text{poly}(n))$ [Zoufal, Lucchi, Woerner \(2019\)](#)

Thank you for your attention!!



Backup slides

- Data generation

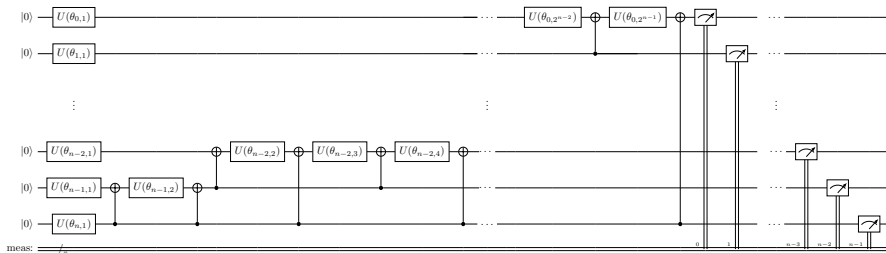
- C++ code based on ROOT \rightarrow generates n -particle events $\left\{ \begin{array}{l} \text{Massive} \\ \text{Massless} \end{array} \right.$
- Precision $\rightarrow 10^{-2}$
- Proton-proton $s = \sqrt{14}\text{TeV}$
- $p_T \geq 10 \text{ GeV}$
- $n = 128$ massless particles

- Efficiencies w.r.t the number of shots:

$$\left(d_{ij}^{-1}\right)^a$$

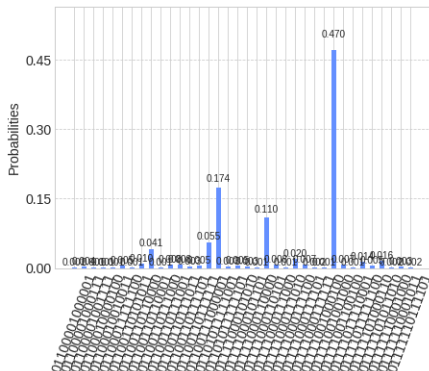
a	Efficiency anti- k_T	<i>Shots</i> anti- k_T	Efficiency k_T	<i>Shots</i> k_T	Efficiency Cam/Aachen	<i>Shots</i> Cam/Aachen
1	0.96	50	0.98	50	0.96	70
2	0.99	40	0.99	45	0.98	60
3	1.00	25	0.98	20	0.97	40
4	1.00	15	0.95	15	1.00	20
5	0.99	5	1.00	8	0.98	10

- How have we loaded the state in our quantum simulations?
-We have used the Grover-Rudolph algorithm:



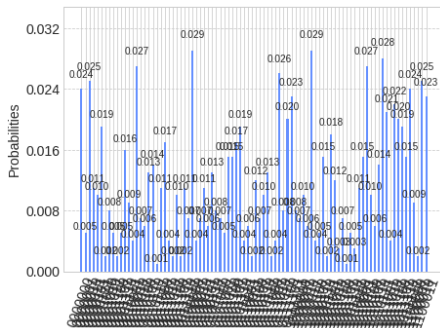
- Gates time execution/propagation delay in "early stages"
 - Classical gates propagation delay $\sim 100\text{ns}$ (1980s)
 - Quantum gates time execution $\sim 100\text{ ns}$ (IBMQ Melbourne) for CNOTs (2022)
- Gates time execution/propagation delay when tech is consolidated
 - Classical gates propagation delay $\sim 100\text{ps}$ (2022)
 - Quantum gates time execution ??? (2060s)

- Quantum maximum search algorithm:



Real distances to be analysed

$N = 8201$, 14 qubits, 1000 shots



Random distribution from (1,1000)

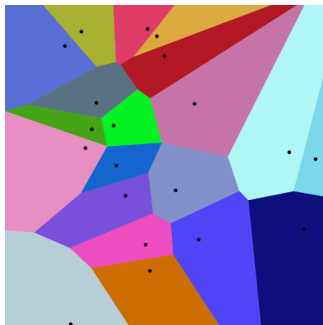
$N = 100$, 7 qubits, 1000 shots

Failing in obtaining the minimum

- Failing in getting the closest centroid
 - The particle is assigned to other cluster (not the nearest centroid)
 - This problem will be solved with more iterations → it will finally converge
- Failing in getting the smallest distance
 - Flip in the order in which two particles merge
 - The final result will in many cases be independent of this permutation

Voronoi diagrams

- As a simple illustration, consider a group of shops in a city. Suppose we want to estimate the number of customers of a given shop.
 - With all else being equal (price, products, quality of service, etc)
 - Reasonable to assume that customers choose their preferred shop simply by distance considerations
 - The Voronoi cell R_k of a given shop P_k can be used for giving a rough estimate on the number of potential customers going to this shop



Usefulness of K-means and Affinity Propagation

- Both use invariant mass squared s_{12} as a metric \rightarrow Lorentz Invariant quantity \rightarrow does not change from one inertial frame to another.

- K-means:

- [Chekanov 2006](#)

Leads to 25% and 40%

improvement of the top-quark and W **mass resolution**, respectively, compared to the k_T algorithm. Nevertheless it is **3 times slower**.

- [Thaler, Van Tilburg \(2011\)](#)
 - [Stewart, Tackmann, Thaler, Vermillion, Wilkason \(2016\)](#)

- Affinity Propagation:

- [Leone, Sumedha, Weigt \(2007\)\)](#)

Biological application \rightarrow cancer datasets

- [Bailly-Bechet et al. \(2009\)](#)

Biological/medical datasets

- [González-Martín et al. \(2017\)](#)

Astrophysical datasets

- Not** been applied **yet in HEP**
 \rightarrow computationally demanding. On the other hand, it could overcome K-means \rightarrow does **not need** the number of clusters to be **pre-defined**.