# Quantum Algorithms lecture

## Quantum algorithm for Topological data analysis

Casper Gyurik

Leiden Institute of Advanced Computer Science

November 11th, 2019

Universiteit Leiden
The Netherlands

What is topological data analysis?

The topological data analysis pipeline

- Going from a dataset to a topological object
- Studying topological object using homology

What can quantum computers bring to the table?

The Lloyd, Garnerone & Zanardi algorithm

# What is topological data analysis?

studying the shape of a dataset

Topological data analysis studies the *topology* of your dataset.

- I.e., features that are invariant under *continuous* deformations.

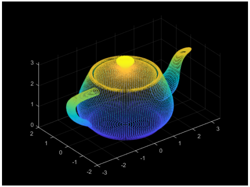
# What is topological data analysis?

studying the shape of a dataset

Topological data analysis studies the *topology* of your dataset.

- I.e., features that are invariant under *continuous* deformations.

Typically, the dataset is a point cloud in a $d$-dimensional space such as $\mathbb{R}^d$.

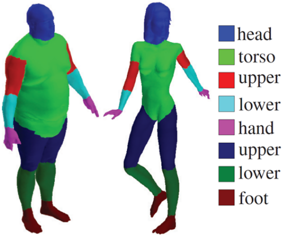


- E.g, a point cloud sampled from a 3d object.

# What can topological data analysis be used for?

an example of an application of topological data analysis

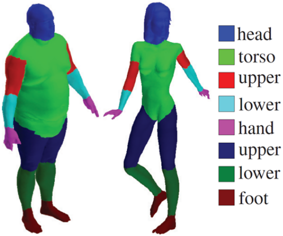
Suppose we would like to classify parts of the human body.



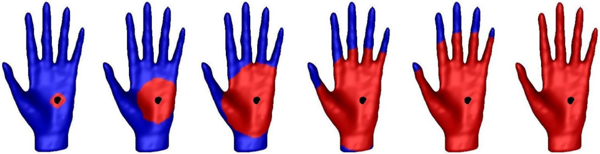
# What can topological data analysis be used for?

an example of an application of topological data analysis

Suppose we would like to classify parts of the human body.




TDA does this by studying *persistent topological* features.



- I.e., features of neighbourhoods around points at different scales.

# What topological features do we study? How do we study them?

an example of an important topological feature

Important topological feature to study: *the number of holes*.

(a) One hole

(b) Two holes

# What topological features do we study? How do we study them?

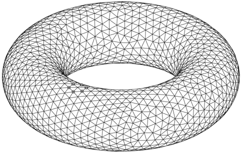
an example of an important topological feature

Important topological feature to study: *the number of holes*.

(a) One hole

(b) Two holes

**Algebraic topology:** use algebra to compute topological features.



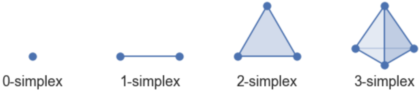
- Triangulate object and study connections between triangles using algebra.

# Going from a dataset to a simplicial complex

## The Vietoris-Rips complex

**1st step TDA pipeline:** converting point-cloud to a *simplicial complex*.

- A collection of points, lines, triangles and their $k$-dimensional counterparts.



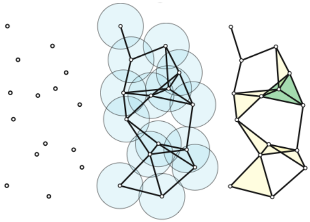
# Going from a dataset to a simplicial complex

## The Vietoris-Rips complex

**1st step TDA pipeline:** converting point-cloud to a *simplicial complex*.

- A collection of points, lines, triangles and their $k$-dimensional counterparts.


**Vietoris-Rips complex**: draw circles with radius $\epsilon > 0$ around data-points and connect data-points if their circles overlap.



- *Persistent topology:* how does topology change if we vary $\epsilon$?

# Studying topology of simplicial complex using homology

turning simplicial complex into a complex vector space

**2nd step TDA pipeline:** study topology of Vietoris-Rips complex using *homology*.

- A tool from algebraic topology to extract topological features.

# Studying topology of simplicial complex using homology

turning simplicial complex into a complex vector space

**2nd step TDA pipeline:** study topology of Vietoris-Rips complex using *homology*.

- A tool from algebraic topology to extract topological features.

First, for a point cloud $\mathcal{D} = \{v_i\}_{i=1}^n$ encode the $k$-simplices of the Vietoris-Rips complex as $n$-bit strings.

$\mathrm{VR}(\mathcal{D}, \epsilon)_k := \{j \in \{0,1\}^n \mid j$ has $k$ ones and $\forall j_i, j_l = 1$ we have $||v_i - v_l|| < \epsilon\}$.

# Studying topology of simplicial complex using homology

turning simplicial complex into a complex vector space

**2nd step TDA pipeline:** study topology of Vietoris-Rips complex using *homology*.

- A tool from algebraic topology to extract topological features.

First, for a point cloud $\mathcal{D} = \{v_i\}_{i=1}^{n}$ encode the $k$-simplices of the Vietoris-Rips complex as $n$-bit strings.

$\mathsf{VR}(\mathcal{D}, \epsilon)_k := \{j \in \{0,1\}^n \mid j$ has $k$ ones and $\forall j_i, j_l = 1$ we have $||v_i - v_l|| < \epsilon\}$.

Next, we turn this into a complex vector space

$$\mathcal{H}_k^{\epsilon} := \mathsf{span}_{\mathbb{C}}\{|j\rangle \mid j \in \mathsf{VR}(\mathcal{D}, \epsilon)_k\} \subset \left(\mathbb{C}^2\right)^{\otimes n}.$$

# Studying topology of simplicial complex using homology

turning simplicial complex into a complex vector space

**2nd step TDA pipeline:** study topology of Vietoris-Rips complex using *homology*.

- A tool from algebraic topology to extract topological features.

First, for a point cloud $\mathcal{D} = \{v_i\}_{i=1}^n$ encode the $k$-simplices of the Vietoris-Rips complex as $n$-bit strings.

$\mathsf{VR}(\mathcal{D}, \epsilon)_k := \{j \in \{0,1\}^n \mid j$ has $k$ ones and $\forall j_i, j_l = 1$ we have $||v_i - v_l|| < \epsilon\}$.

Next, we turn this into a complex vector space

$$\mathcal{H}_k^\epsilon := \mathsf{span}_{\mathbb{C}}\{|j\rangle \mid j \in \mathsf{VR}(\mathcal{D}, \epsilon)_k\} \subset \left(\mathbb{C}^2\right)^{\otimes n}.$$

We now have an algebraic object to study!

# Studying topology of simplicial complex using homology



In order to study $\mathcal{H}_k^\epsilon$ we consider the so-called $k$*-th boundary map*

$$\partial_k^\epsilon : \mathcal{H}_k^\epsilon \to \mathcal{H}_{k-1}^\epsilon$$

$$|j\rangle \mapsto \sum_{i=1}^{k} (-1)^i \, |j(\hat{i})\rangle \,,$$

where $j(\hat{i})$ is obtained from $j$ by setting $i$-th 1 to 0.

# Studying topology of simplicial complex using homology

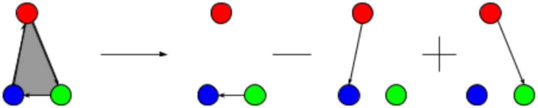
## the boundary operator

In order to study $\mathcal{H}_k^\epsilon$ we consider the so-called $k$*-th boundary map*

$$\partial_k^\epsilon : \mathcal{H}_k^\epsilon \to \mathcal{H}_{k-1}^\epsilon$$
$$|j\rangle \mapsto \sum_{i=1}^{k} (-1)^i \, |j(\hat{i})\rangle \, ,$$

where $j(\hat{i})$ is obtained from $j$ by setting $i$-th 1 to 0.

Maps triangle to alternating sum of its edges and tetrahedron to sum of its faces.



# Studying topology of simplicial complex using homology

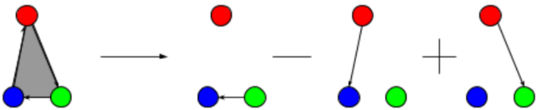
## the boundary operator

In order to study $\mathcal{H}_k^\epsilon$ we consider the so-called $k$*-th boundary map*

$$\partial_k^\epsilon : \mathcal{H}_k^\epsilon \to \mathcal{H}_{k-1}^\epsilon$$
$$|j\rangle \mapsto \sum_{i=1}^{k} (-1)^i \, |j(\hat{i})\rangle \,,$$

where $j(\hat{i})$ is obtained from $j$ by setting $i$-th 1 to 0.

Maps triangle to alternating sum of its edges and tetrahedron to sum of its faces.



Using this map we can study how simplices are connected to eachother!

# Studying topology of simplicial complex using homology

## the homology group

Important object to study is the so-called *$k$-th homology group* defined as

$$H_k^\epsilon = \left(\ker\, \partial_k^\epsilon\right) / \left(\mathrm{im}\, \partial_{k+1}^\epsilon\right).$$

- Quotient of vector spaces $\leftrightarrow$ substracting bases of vector spaces.

# Studying topology of simplicial complex using homology

the homology group

Important object to study is the so-called $k$*-th homology group* defined as

$$H_k^\epsilon = \left(\text{ker } \partial_k^\epsilon\right) / \left(\text{im } \partial_{k+1}^\epsilon\right).$$

- Quotient of vector spaces $\leftrightarrow$ substracting bases of vector spaces.

Its dimension, called the $k$*-th betti-number*, is an important topological feature.

$$\beta_k^\epsilon = \dim H_k^\epsilon.$$

- Turns out: $\beta_k^\epsilon$ is equal to number of $k$-dimensional holes at scale $\epsilon$.

# Studying topology of simplicial complex using homology

## the homology group

Important object to study is the so-called *$k$-th homology group* defined as

$$H_k^\epsilon = (\text{ker } \partial_k^\epsilon) \,/\, (\text{im } \partial_{k+1}^\epsilon)\,.$$

- Quotient of vector spaces $\leftrightarrow$ substracting bases of vector spaces.

Its dimension, called the *$k$-th betti-number*, is an important topological feature.

$$\beta_k^\epsilon = \dim H_k^\epsilon.$$

- Turns out: $\beta_k^\epsilon$ is equal to number of $k$-dimensional holes at scale $\epsilon$.

**Goal in TDA:** compute $\beta_k^\epsilon$ to extract topological features of your point cloud.

To compute $\beta_k^\epsilon$ we consider the so-called $k$*-th combinatorial Laplacian*

$$\Delta_k^\epsilon := \partial_k^T \partial_k + \partial_{k+1} \partial_{k+1}^T : \mathcal{H}_k^\epsilon \to \mathcal{H}_k^\epsilon.$$

- Hermitian matrix!

To compute $\beta_k^\epsilon$ we consider the so-called *$k$-th combinatorial Laplacian*

$$\Delta_k^\epsilon := \partial_k^T \partial_k + \partial_{k+1} \partial_{k+1}^T : \mathcal{H}_k^\epsilon \to \mathcal{H}_k^\epsilon.$$

- Hermitian matrix!

We do so because it can be shown that the following holds

$$\beta_k^\epsilon = \dim \left( \ker \Delta_k^\epsilon \right).$$

To compute $\beta_k^\epsilon$ we consider the so-called *$k$-th combinatorial Laplacian*

$$\Delta_k^\epsilon := \partial_k^T \partial_k + \partial_{k+1}\partial_{k+1}^T : \mathcal{H}_k^\epsilon \to \mathcal{H}_k^\epsilon.$$

- Hermitian matrix!

We do so because it can be shown that the following holds

$$\beta_k^\epsilon = \dim\left(\ker \Delta_k^\epsilon\right).$$

**Question:** can we use quantum computers to efficiently compute this dimension?

# What can quantum computers bring to the table?

Quantum linear algebra: Hamiltonian simulation and quantum phase estimation

Quantum computers are generally very good at doing linear algebra.

- $\Delta_k^\epsilon$ is Hermitian $\implies$ Hamiltonian simulation can implement $U = e^{i\Delta_k^\epsilon}$!
- Also, can use quantum phase estimaton to investigate eigenvalues of $U$.

# What can quantum computers bring to the table?

## Quantum linear algebra: Hamiltonian simulation and quantum phase estimation

Quantum computers are generally very good at doing linear algebra.

- $\Delta_k^\epsilon$ is Hermitian $\implies$ Hamiltonian simulation can implement $U = e^{i\Delta_k^\epsilon}$!
- Also, can use quantum phase estimaton to investigate eigenvalues of $U$.

We are interested in the number of eigenvalues that are equal to 0 since

$$\beta_k^\epsilon = \dim\left(\ker \Delta_k^\epsilon\right) = \#\{j : \lambda_j(\Delta_k^\epsilon) = 0\},$$

where $\lambda_1(\Delta_k^\epsilon), \ldots, \lambda_N(\Delta_k^\epsilon)$ denote the eigenvalues of $\Delta_k^\epsilon$.

Quantum computers are generally very good at doing linear algebra.

- $\Delta_k^\epsilon$ is Hermitian $\implies$ Hamiltonian simulation can implement $U = e^{i\Delta_k^\epsilon}$!
- Also, can use quantum phase estimaton to investigate eigenvalues of $U$.

We are interested in the number of eigenvalues that are equal to 0 since

$$\beta_k^\epsilon = \dim\left(\ker \Delta_k^\epsilon\right) = \#\{j : \lambda_j(\Delta_k^\epsilon) = 0\},$$

where $\lambda_1(\Delta_k^\epsilon), \ldots, \lambda_N(\Delta_k^\epsilon)$ denote the eigenvalues of $\Delta_k^\epsilon$.

**LGZ algorithm:** combine Hamiltonian simulation and QPE to estimate number of eigenvalues that are equal to 0.

# The Lloyd, Garnerone & Zanardi algorithm

an overview of the algorithm

**Goal:** estimate $\beta_k^\epsilon = \dim\left(\ker \Delta_k^\epsilon\right) = \#\{j : \lambda_j(\Delta_k^\epsilon) = 0\}$.

To do so, the LGZ algorithm takes the following steps:

# The Lloyd, Garnerone & Zanardi algorithm

an overview of the algorithm

**Goal:** estimate $\beta_k^\epsilon = \dim(\ker \Delta_k^\epsilon) = \#\{j : \lambda_j(\Delta_k^\epsilon) = 0\}$.

To do so, the LGZ algorithm takes the following steps:

1. Use Grover's algorithm to prepare the *simplex-state* given by

$$|\psi_k\rangle := \frac{1}{\sqrt{\dim \mathcal{H}_k^\epsilon}} \sum_{j \in \mathsf{VR}(\mathcal{D},\epsilon)_k} |j\rangle \,.$$

   - I.e., the uniform superposition over the complex vector space $\mathcal{H}_k^\epsilon$.

# The Lloyd, Garnerone & Zanardi algorithm

an overview of the algorithm

**Goal:** estimate $\beta_k^\epsilon = \dim(\ker \Delta_k^\epsilon) = \#\{j : \lambda_j(\Delta_k^\epsilon) = 0\}$.

To do so, the LGZ algorithm takes the following steps:

1. Use Grover's algorithm to prepare the *simplex-state* given by

$$|\psi_k\rangle := \frac{1}{\sqrt{\dim \mathcal{H}_k^\epsilon}} \sum_{j \in \mathsf{VR}(\mathcal{D},\epsilon)_k} |j\rangle .$$

   - I.e., the uniform superposition over the complex vector space $\mathcal{H}_k^\epsilon$.

2. Use Hamiltonian simulation to implement $U = e^{i\Delta_k^\epsilon}$.
   - Actually, since $\Delta_k^\epsilon$ is not sparse we implement $e^{iB_k}$, for $B_k \approx \sqrt{\Delta_k^\epsilon}$.

# The Lloyd, Garnerone & Zanardi algorithm

an overview of the algorithm

**Goal:** estimate $\beta_k^\epsilon = \dim(\ker \Delta_k^\epsilon) = \#\{j : \lambda_j(\Delta_k^\epsilon) = 0\}$.

To do so, the LGZ algorithm takes the following steps:

1. Use Grover's algorithm to prepare the *simplex-state* given by

$$|\psi_k\rangle := \frac{1}{\sqrt{\dim \mathcal{H}_k^\epsilon}} \sum_{j \in \mathsf{VR}(\mathcal{D},\epsilon)_k} |j\rangle .$$

   - I.e., the uniform superposition over the complex vector space $\mathcal{H}_k^\epsilon$.
2. Use Hamiltonian simulation to implement $U = e^{i\Delta_k^\epsilon}$.
   - Actually, since $\Delta_k^\epsilon$ is not sparse we implement $e^{iB_k}$, for $B_k \approx \sqrt{\Delta_k^\epsilon}$.
3. Use quantum phase estimation to sample from eigenvalues of $U = e^{i\Delta_k^\epsilon}$.
   - By last weeks tutorial we can one-to-one relate these to eigenvalues of $\Delta_k^\epsilon$.

# The Lloyd, Garnerone & Zanardi algorithm

an overview of the algorithm

**Goal:** estimate $\beta_k^\epsilon = \dim(\ker \Delta_k^\epsilon) = \#\{j : \lambda_j(\Delta_k^\epsilon) = 0\}$.

To do so, the LGZ algorithm takes the following steps:

1. Use Grover's algorithm to prepare the *simplex-state* given by

$$|\psi_k\rangle := \frac{1}{\sqrt{\dim \mathcal{H}_k^\epsilon}} \sum_{j \in \mathsf{VR}(\mathcal{D},\epsilon)_k} |j\rangle \,.$$

   - I.e., the uniform superposition over the complex vector space $\mathcal{H}_k^\epsilon$.
2. Use Hamiltonian simulation to implement $U = e^{i\Delta_k^\epsilon}$.
   - Actually, since $\Delta_k^\epsilon$ is not sparse we implement $e^{iB_k}$, for $B_k \approx \sqrt{\Delta_k^\epsilon}$.
3. Use quantum phase estimation to sample from eigenvalues of $U = e^{i\Delta_k^\epsilon}$.
   - By last weeks tutorial we can one-to-one relate these to eigenvalues of $\Delta_k^\epsilon$.
4. Estimate the number of eigenvalues $\lambda_j(\Delta_k^\epsilon) = 0$.
   - Either using quantum counting or Monte-Carlo estimation.

# The Lloyd, Garnerone & Zanardi algorithm

a Grover-step to prepare simplex state

**Goal:** Prepare the *simplex-state* given by $|\psi_k\rangle := \frac{1}{\sqrt{\dim \mathcal{H}_k^\epsilon}} \sum_{j \in \mathsf{VR}(\mathcal{D},\epsilon)_k} |j\rangle$.

# The Lloyd, Garnerone & Zanardi algorithm

a Grover-step to prepare simplex state

**Goal:** Prepare the *simplex-state* given by $|\psi_k\rangle := \frac{1}{\sqrt{\dim \mathcal{H}_k^\epsilon}} \sum_{j \in \mathsf{VR}(\mathcal{D},\epsilon)_k} |j\rangle$.

For $j \in \{0,1\}^n$ we can efficiently check whether $j \in \mathsf{VR}(\mathcal{D},\epsilon)_k$.

- First, check whether $j$ contains exactly $k$ ones.
- Afterwards, check pairwise distance between all $v_i$ for which $j_i = 1$.
  - Requires a total of $O(k^2)$ computations of pairwise distances $||v_i - v_j||$.

# The Lloyd, Garnerone & Zanardi algorithm

a Grover-step to prepare simplex state

**Goal:** Prepare the *simplex-state* given by $|\psi_k\rangle := \frac{1}{\sqrt{\dim \mathcal{H}_k^\epsilon}} \sum_{j \in \mathsf{VR}(\mathcal{D},\epsilon)_k} |j\rangle$.

For $j \in \{0,1\}^n$ we can efficiently check whether $j \in \mathsf{VR}(\mathcal{D}, \epsilon)_k$.

- First, check whether $j$ contains exactly $k$ ones.
- Afterwards, check pairwise distance between all $v_i$ for which $j_i = 1$.
  - Requires a total of $O(k^2)$ computations of pairwise distances $||v_i - v_j||$.

Allows us to efficiently implement a quantum oracle to the membership function

$$f : \{0,1\}^n \to \{0,1\}$$

$$f(j) = \begin{cases} 1 \text{ if } j \in \mathsf{VR}(\mathcal{D},\epsilon)_k, \\ 0 \text{ else.} \end{cases}$$

# The Lloyd, Garnerone & Zanardi algorithm

a Grover-step to prepare simplex state

**Goal:** Prepare the *simplex-state* given by $|\psi_k\rangle := \frac{1}{\sqrt{\dim \mathcal{H}_k^\epsilon}} \sum_{j \in \mathsf{VR}(\mathcal{D},\epsilon)_k} |j\rangle$.

For $j \in \{0,1\}^n$ we can efficiently check whether $j \in \mathsf{VR}(\mathcal{D},\epsilon)_k$.

- First, check whether $j$ contains exactly $k$ ones.
- Afterwards, check pairwise distance between all $v_i$ for which $j_i = 1$.
  - Requires a total of $O(k^2)$ computations of pairwise distances $||v_i - v_j||$.

Allows us to efficiently implement a quantum oracle to the membership function

$$f : \{0,1\}^n \to \{0,1\}$$

$$f(j) = \begin{cases} 1 \text{ if } j \in \mathsf{VR}(\mathcal{D},\epsilon)_k, \\ 0 \text{ else.} \end{cases}$$

Therefore, Grover's algorithm allows us to implement the mapping

$$\frac{1}{\sqrt{2^n}} \sum_{j \in \{0,1\}^n} |j\rangle \mapsto |\psi_k\rangle = \frac{1}{\sqrt{\dim \mathcal{H}_k^\epsilon}} \sum_{j \in \mathsf{VR}(\mathcal{D},\epsilon)_k} |j\rangle .$$

# The Lloyd, Garnerone & Zanardi algorithm

using Hamiltonian simulation and QPE to uniformly sample from the eigenvalues of $\Delta_k^\epsilon$

We would like to use HS and QPE to sample from the eigenvalues of $e^{i\Delta_k^\epsilon}$.

- Similar to Problem 3 of last tutorial, this gives us the eigenvalues of $\Delta_k^\epsilon$.
- We don't know eigenvectors of $\Delta_k^\epsilon$. What input quantum state do we use?

# The Lloyd, Garnerone & Zanardi algorithm

using Hamiltonian simulation and QPE to uniformly sample from the eigenvalues of $\Delta_k^\epsilon$

We would like to use HS and QPE to sample from the eigenvalues of $e^{i\Delta_k^\epsilon}$.

- Similar to Problem 3 of last tutorial, this gives us the eigenvalues of $\Delta_k^\epsilon$.
- We don't know eigenvectors of $\Delta_k^\epsilon$. What input quantum state do we use?

From the simplex-state $|\psi_k\rangle$ we can efficiently prepare the mixed state

$$P_{\mathcal{H}_k^\epsilon} = \frac{1}{\dim \mathcal{H}_k^\epsilon} \sum_{j \in \mathsf{VR}(\mathcal{D},\epsilon)_k} |j\rangle \langle j| = \text{"Projector onto } \mathcal{H}_k^\epsilon\text{"}.$$

# The Lloyd, Garnerone & Zanardi algorithm

using Hamiltonian simulation and QPE to uniformly sample from the eigenvalues of $\Delta_k^\epsilon$

We would like to use HS and QPE to sample from the eigenvalues of $e^{i\Delta_k^\epsilon}$.

- Similar to Problem 3 of last tutorial, this gives us the eigenvalues of $\Delta_k^\epsilon$.
- We don't know eigenvectors of $\Delta_k^\epsilon$. What input quantum state do we use?

From the simplex-state $|\psi_k\rangle$ we can efficiently prepare the mixed state

$$P_{\mathcal{H}_k^\epsilon} = \frac{1}{\dim \mathcal{H}_k^\epsilon} \sum_{j \in \mathsf{VR}(\mathcal{D},\epsilon)_k} |j\rangle\langle j| = \text{"Projector onto } \mathcal{H}_k^\epsilon\text{"}.$$

We can rewrite this mixed state in terms of the eigenvectors of $\Delta_k^\epsilon$

$$P_{\mathcal{H}_k^\epsilon} = \frac{1}{\dim \mathcal{H}_k^\epsilon} \sum_{j=1}^{\dim \mathcal{H}_k^\epsilon} |\psi_j\rangle\langle \psi_j|.$$

# The Lloyd, Garnerone & Zanardi algorithm

using Hamiltonian simulation and QPE to uniformly sample from the eigenvalues of $\Delta_k^\epsilon$

We would like to use HS and QPE to sample from the eigenvalues of $e^{i\Delta_k^\epsilon}$.

- Similar to Problem 3 of last tutorial, this gives us the eigenvalues of $\Delta_k^\epsilon$.
- We don't know eigenvectors of $\Delta_k^\epsilon$. What input quantum state do we use?

From the simplex-state $|\psi_k\rangle$ we can efficiently prepare the mixed state

$$P_{\mathcal{H}_k^\epsilon} = \frac{1}{\dim \mathcal{H}_k^\epsilon} \sum_{j \in \mathsf{VR}(\mathcal{D},\epsilon)_k} |j\rangle \langle j| = \text{"Projector onto } \mathcal{H}_k^\epsilon\text{"}.$$

We can rewrite this mixed state in terms of the eigenvectors of $\Delta_k^\epsilon$

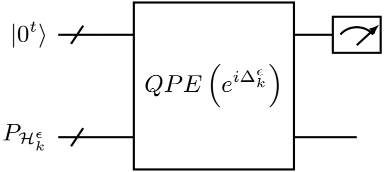
$$P_{\mathcal{H}_k^\epsilon} = \frac{1}{\dim \mathcal{H}_k^\epsilon} \sum_{j=1}^{\dim \mathcal{H}_k^\epsilon} |\psi_j\rangle \langle \psi_j| .$$

So, the mixed state $P_{\mathcal{H}_k^\epsilon}$ is the $j$-th eigenstate $|\psi_j\rangle$ with probability $\frac{1}{\dim \mathcal{H}_k^\epsilon}$.
Let's use this state as input to the HS + QPE routine!

# The Lloyd, Garnerone & Zanardi algorithm

the circuit and the output distribution

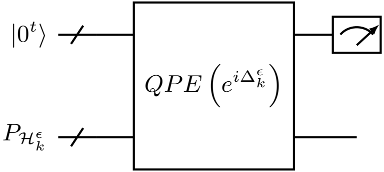
Altogether, after the Grover-step the LGZ algorithm runs the following circuit



# The Lloyd, Garnerone & Zanardi algorithm

the circuit and the output distribution

Altogether, after the Grover-step the LGZ algorithm runs the following circuit



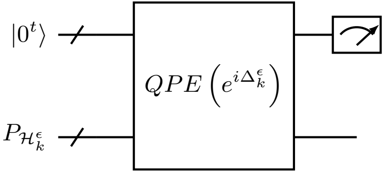

$P_{\mathcal{H}_k^\epsilon}$ is in an eigenstate $|\psi_j\rangle$ with uniform probability, thus the probability of measuring an eigenvalue $\lambda$ is

$$\mathsf{Pr}\left(\lambda\right) = \frac{\#\{j \mid \lambda_j = \lambda\}}{\dim \mathcal{H}_k^\epsilon}.$$

# The Lloyd, Garnerone & Zanardi algorithm

the circuit and the output distribution

Altogether, after the Grover-step the LGZ algorithm runs the following circuit




$P_{\mathcal{H}_k^\epsilon}$ is in an eigenstate $|\psi_j\rangle$ with uniform probability, thus the probability of measuring an eigenvalue $\lambda$ is

$$\mathsf{Pr}\left(\lambda\right) = \frac{\#\{j \mid \lambda_j = \lambda\}}{\dim \mathcal{H}_k^\epsilon}.$$

In particular, the probability of measuring $\lambda = 0$ is given bu

$$\mathsf{Pr}\left(0\right) = \frac{\#\{j \mid \lambda_j = 0\}}{\dim \mathcal{H}_k^\epsilon} = \frac{\beta_k^\epsilon}{\dim \mathcal{H}_k^\epsilon}.$$

Using quantum counting or Monte-Carlo estimation, we can now estimate $\beta_k^\epsilon$.

# References

where to find the papers

- Nature: https://www.nature.com/articles/ncomms10138
- Arxiv: https://arxiv.org/pdf/1408.3106.pdf
- Nice review paper: https://arxiv.org/pdf/1906.07673.pdf

# My research

why am I looking at this algorithm? what are the interesting questions?

- **Motivation:** dequantizations of Ewin Tang et al.

# My research

why am I looking at this algorithm? what are the interesting questions?

- **Motivation:** dequantizations of Ewin Tang et al.
- **Question:** can we show classical/quantum seperation?

# My research

why am I looking at this algorithm? what are the interesting questions?

- **Motivation:** dequantizations of Ewin Tang et al.
- **Question:** can we show classical/quantum seperation?
  - Does it solve a hard problem for complexity class with (conjectured) quantum advantage?

# My research

why am I looking at this algorithm? what are the interesting questions?

- **Motivation:** dequantizations of Ewin Tang et al.
- **Question:** can we show classical/quantum seperation?
  - Does it solve a hard problem for complexity class with (conjectured) quantum advantage?
- **Question:** can we use samples from the eigenvalues of $\Delta_k^\epsilon$ for something else than computing Betti numbers?

# My research

why am I looking at this algorithm? what are the interesting questions?

- **Motivation:** dequantizations of Ewin Tang et al.
- **Question:** can we show classical/quantum seperation?
  - Does it solve a hard problem for complexity class with (conjectured) quantum advantage?
- **Question:** can we use samples from the eigenvalues of $\Delta_k^\epsilon$ for something else than computing Betti numbers?
  - So-called applications of "simplicial spectral theory".