Heat Kernels in Graphs:
A Journey from Random Walks to Geometry, and Back

He Sun
University of Bristol
Let $G$ be an undirected $d$-regular graph with $n$ vertices.
Let $G$ be an undirected $d$-regular graph with $n$ vertices.

**Laplacian Matrix**

The **normalised Laplacian matrix** of $G$ is defined by

$$
\mathcal{L} \triangleq \mathbf{I} - \frac{1}{d} \cdot \mathbf{A},
$$

where $\mathbf{A}$ is the adjacency matrix of $G$. 

---

Notation

Let $G$ be an undirected $d$-regular graph with $n$ vertices.
Let $G$ be an undirected $d$-regular graph with $n$ vertices.

The normalised Laplacian matrix of $G$ is defined by

$$L \triangleq I - \frac{1}{d} \cdot A,$$

where $A$ is the adjacency matrix of $G$.

Example:

$$L_G = \begin{pmatrix}
1 & -1/3 & -1/3 & -1/3 \\
-1/3 & 1 & -1/3 & -1/3 \\
-1/3 & -1/3 & 1 & -1/3 \\
-1/3 & -1/3 & -1/3 & 1
\end{pmatrix}$$
Let $G$ be an undirected $d$-regular graph with $n$ vertices.

The normalised Laplacian matrix of $G$ is defined by

$$\mathcal{L} \triangleq \mathbf{I} - \frac{1}{d} \cdot \mathbf{A},$$

where $\mathbf{A}$ is the adjacency matrix of $G$.

Example:

Matrix $\mathcal{L}$ has eigenvalues $0 = \lambda_1 \leq \ldots \leq \lambda_n$ with corresponding eigenvectors $f_1, \ldots, f_n$. 

$$\mathcal{L}_G = \begin{pmatrix}
1 & -1/3 & -1/3 & -1/3 \\
-1/3 & 1 & -1/3 & -1/3 \\
-1/3 & -1/3 & 1 & -1/3 \\
-1/3 & -1/3 & -1/3 & 1 \\
\end{pmatrix}$$
Heat Kernel: a Fundamental Solution of a PDE

Let $\mathcal{M}$ be a compact Riemannian manifold, and

$$u : \mathcal{M} \times [0, \infty) \to \mathbb{R}$$

be a smooth function describing the temperature at a point in $\mathcal{M}$ and time $t$. 
Let $\mathcal{M}$ be a compact Riemannian manifold, and

$$u : \mathcal{M} \times [0, \infty) \to \mathbb{R}$$

be a smooth function describing the temperature at a point in $\mathcal{M}$ and time $t$. Then the heat kernel is the fundamental solution of the following PDE:

$$\frac{\partial u}{\partial t} + \Delta u = 0.$$
Let $\mathcal{M}$ be a compact Riemannian manifold, and

$$u : \mathcal{M} \times [0, \infty) \to \mathbb{R}$$

be a smooth function describing the temperature at a point in $\mathcal{M}$ and time $t$. Then the heat kernel is the fundamental solution of the following PDE:

$$\frac{\partial u}{\partial t} + \Delta u = 0.$$
When $\Delta$ is the Laplacian matrix $\mathcal{L}$ of graph $G$, for any $t \geq 0$ the heat kernel of $G$ can be written as

$$H_t = e^{-t\mathcal{L}} = \sum_{k=0}^{\infty} \frac{t^k e^{-t}}{k!} P^k,$$

where $P$ is the random walk matrix of $G$. 
Heat Kernel Defines a Continuous-Time Random Walk

Heat Kernel in Graphs

When $\Delta$ is the Laplacian matrix $\mathcal{L}$ of graph $G$, for any $t \geq 0$ the heat kernel of $G$ can be written as

$$H_t = e^{-t\mathcal{L}} = \sum_{k=0}^{\infty} \frac{t^k e^{-t}}{k!} P^k,$$

where $P$ is the random walk matrix of $G$.

Heat kernel defines a continuous-time random walk:

- Vertices choose a neighbour according to $P$;
- Jumps occur after $\text{Poison}(1)$ waiting times.
Heat Kernel in Graphs

When $\Delta$ is the Laplacian matrix $\mathcal{L}$ of graph $G$, for any $t \geq 0$ the heat kernel of $G$ can be written as

$$H_t = e^{-tL} = \sum_{k=0}^{\infty} \frac{t^k e^{-t}}{k!} P^k,$$

where $P$ is the random walk matrix of $G$.

Heat kernel defines a continuous-time random walk:

- Vertices choose a neighbour according to $P$;
- Jumps occur after $\text{Poison}(1)$ waiting times.

**Continuous-time Random Walks $\approx$ Discrete-time Random Walks!**
Heat Kernel in Graphs

When $\Delta$ is the Laplacian matrix $\mathcal{L}$ of graph $G$, for any $t \geq 0$ the heat kernel of $G$ can be written as

$$H_t = e^{-t\mathcal{L}} = \sum_{k=0}^{\infty} \frac{t^k e^{-t}}{k!} P^k,$$

where $P$ is the random walk matrix of $G$.

Heat kernel defines a continuous-time random walk:

- Vertices choose a neighbour according to $P$;
- Jumps occur after $\text{Poison}(1)$ waiting times.

**Continuous-time Random Walks $\approx$ Discrete-time Random Walks!**

The heat kernel defines a semi-group, i.e.,

$$H_{t+s} = H_t \cdot H_s, \forall t, s \geq 0 \quad \text{and} \quad \lim_{t \to 0} H_t = I.$$
For any time-step $t \geq 0$, define an embedding $\psi_t : V \mapsto \mathbb{R}^n$ by

$$
\psi_t(v) = \left( e^{-t\lambda_1} f_1(v), e^{-t\lambda_2} f_2(v), \ldots, e^{-t\lambda_n} f_n(v) \right).
$$
For any time-step $t \geq 0$, define an embedding $\psi_t : V \hookrightarrow \mathbb{R}^n$ by

$$
\psi_t(v) = \left( e^{-t\lambda_1} f_1(v), e^{-t\lambda_2} f_2(v), \ldots, e^{-t\lambda_n} f_n(v) \right).
$$

Let the heat kernel distance between vertices $u$ and $v$ be

$$
d_t(u, v) = \|\psi_t(u) - \psi_t(v)\|^2.
$$
For any time-step $t \geq 0$, define an embedding $\psi_t : V \mapsto \mathbb{R}^n$ by

$$
\psi_t(v) = \left( e^{-t\lambda_1} f_1(v), e^{-t\lambda_2} f_2(v), \ldots, e^{-t\lambda_n} f_n(v) \right).
$$

Let the heat kernel distance between vertices $u$ and $v$ be

$$
d_t(u, v) = \|\psi_t(u) - \psi_t(v)\|^2.
$$

A simple calculation shows that $d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$. 

Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$$
Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$$
Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$$

edge $\{u, v\}$ is along a sparse cut
Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$$

edge $\{u, v\}$ is along a sparse cut
Meaning of the heat kernel distance, with a proper choice of $t$:

\[ d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2 \]

edge \( \{u, v\} \) is along a sparse cut

- One of the two walks needs to go across a sparse cut.
Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$$

edge $\{u, v\}$ is along a sparse cut

- One of the two walks needs to go across a sparse cut.
- For any vertex $w$, the value of $(H_t(w, u) - H_t(w, v))^2$ is big.
 Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$$

- One of the two walks needs to go across a sparse cut.
- For any vertex $w$, the value of $(H_t(w, u) - H_t(w, v))^2$ is big.
- Hence, $d_t(u, v)$ is big.
Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$$

- edge $\{u, v\}$ is along a sparse cut

- One of the two walks needs to go across a sparse cut.
- For any vertex $w$, the value of $(H_t(w, u) - H_t(w, v))^2$ is big.
- Hence, $d_t(u, v)$ is big.
Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$$

- One of the two walks needs to go across a sparse cut.
- For any vertex $w$, the value of $(H_t(w, u) - H_t(w, v))^2$ is big.
- Hence, $d_t(u, v)$ is big.
Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} \left( H_t(w, u) - H_t(w, v) \right)^2$$

- One of the two walks needs to go across a sparse cut.
- For any vertex $w$, the value of $(H_t(w, u) - H_t(w, v))^2$ is big.
- Hence, $d_t(u, v)$ is big.
Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$$

- One of the two walks needs to go across a sparse cut.
- For any vertex $w$, the value of $(H_t(w, u) - H_t(w, v))^2$ is big.
- Hence, $d_t(u, v)$ is big.
Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$$

- One of the two walks needs to go across a sparse cut.
- For any vertex $w$, the value of $(H_t(w, u) - H_t(w, v))^2$ is big.
- Hence, $d_t(u, v)$ is big.

- The values of two $H_t(w, \cdot)$s are close to each other.
Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$$

- One of the two walks needs to go across a sparse cut.
- For any vertex $w$, the value of $(H_t(w, u) - H_t(w, v))^2$ is big.
- Hence, $d_t(u, v)$ is big.

- The values of two $H_t(w, \cdot)$s are close to each other.
- Hence, $(H_t(w, u) - H_t(w, v))^2$ is small for any vertex $w$. 

edge $\{u, v\}$ is along a sparse cut

edge $\{u, v\}$ is at one side a sparse cut
Heat Kernel Distance: From Geometry to Random Walks

Meaning of the heat kernel distance, with a proper choice of $t$:

$$d_t(u, v) = \sum_{w \in V} (H_t(w, u) - H_t(w, v))^2$$

- One of the two walks needs to go across a sparse cut.
- For any vertex $w$, the value of $(H_t(w, u) - H_t(w, v))^2$ is big.
- Hence, $d_t(u, v)$ is big.

- The values of two $H_t(w, \cdot)$s are close to each other.
- Hence, $(H_t(w, u) - H_t(w, v))^2$ is small for any vertex $w$.
- Hence, $d_t(u, v)$ is small.
Key Questions

- Are our intuitions based on random walks correct?
Key Questions

- Are our intuitions based on random walks correct?
- How do we apply these intuitions to design algorithms?
Key Questions

- Are our intuitions based on random walks correct?

- How do we apply these intuitions to design algorithms?

- Do PDEs lead to an entirely new technique to design algorithms for large datasets?
Applications in clustering:
Graph Clustering

Applications in clustering:
Graph Clustering

Applications in clustering:
The conductance of a set $S$ is defined by

$$
\phi_G(S) \triangleq \frac{|E(S, V \setminus S)|}{d \cdot |S|}.
$$
The conductance of a set $S$ is defined by

$$\phi_G(S) \triangleq \frac{|E(S, V \setminus S)|}{d \cdot |S|}.$$
Graph Conductance

The conductance of a set $S$ is defined by

$$
\phi_G(S) \triangleq \frac{|E(S, V \setminus S)|}{d \cdot |S|}.
$$

The conductance of a graph $G$ is defined by

$$
\phi_G \triangleq \min_{S: |S| \leq |V|/2} \phi_G(S).
$$

Cheeger's Inequality

$$
\phi_G(S) = \frac{2}{4.6} = \frac{1}{12}
$$
Graph Conductance

The conductance of a set $S$ is defined by

$$\phi_G(S) \triangleq \frac{|E(S, V \setminus S)|}{d \cdot |S|}.$$

The conductance of a graph $G$ is defined by

$$\phi_G \triangleq \min_{S: |S| \leq |V|/2} \phi_G(S).$$

Cheeger's Inequality

$$\frac{\lambda_2}{2} \leq \phi_G \leq \sqrt{2\lambda_2}.$$

$$\phi_G(S) = \frac{2}{4.6} = \frac{1}{12}$$
The $k$-way expansion constant is defined by

$$
\rho(k) = \min_{\text{partition } A_1, \ldots, A_k} \max_{1 \leq i \leq k} \phi_G(A_i).
$$
$k$-Way Expansion

The $k$-way expansion constant is defined by

$$\rho(k) = \min_{\text{partition } A_1, \ldots, A_k} \max_{1 \leq i \leq k} \phi_G(A_i).$$

Higher-Order Cheeger's Inequality

$$\frac{\lambda_k}{2} \leq \rho(k) \leq O(k^3) \sqrt{\lambda_k}.$$
The $k$-way expansion constant is defined by

$$\rho(k) = \min_{\text{partition } A_1, \ldots, A_k} \max_{1 \leq i \leq k} \phi_G(A_i).$$

Higher-Order Cheeger's Inequality

$$\frac{\lambda_k}{2} \leq \rho(k) \leq O(k^3) \sqrt{\lambda_k}.$$ 

A large gap between $\lambda_{k+1}$ and $\rho(k)$ implies that
- existence of a $k$-way partition with bounded $\rho(k)$. 
The $k$-way expansion constant is defined by

$$
\rho(k) = \min_{\text{partition } A_1, \ldots, A_k} \max_{1 \leq i \leq k} \phi_G(A_i).
$$

Higher-Order Cheeger's Inequality

$$
\frac{\lambda_k}{2} \leq \rho(k) \leq O(k^3) \sqrt{\lambda_k}.
$$

A large gap between $\lambda_{k+1}$ and $\rho(k)$ implies that

- existence of a $k$-way partition with bounded $\rho(k)$.
- any $(k + 1)$-way partition contains a set with conductance at least $\lambda_{k+1}/2$. 
The $k$-way expansion constant is defined by

$$\rho(k) = \min_{\text{partition } A_1, \ldots, A_k} \max_{1 \leq i \leq k} \phi_G(A_i).$$

Higher-Order Cheeger's Inequality

$$\frac{\lambda_k}{2} \leq \rho(k) \leq O(k^3) \sqrt{\lambda_k}.$$ 

A large gap between $\lambda_{k+1}$ and $\rho(k)$ implies that

- existence of a $k$-way partition with bounded $\rho(k)$.
- any $(k+1)$-way partition contains a set with conductance at least $\lambda_{k+1}/2$.
- Graph $G$ has exactly $k$ clusters.
The $k$-way expansion constant is defined by

$$
\rho(k) = \min_{\text{partition } A_1, \ldots, A_k} \max_{1 \leq i \leq k} \phi_G(A_i).
$$

Higher-Order Cheeger's Inequality

$$
\frac{\lambda_k}{2} \leq \rho(k) \leq O(k^3) \sqrt{\lambda_k}.
$$

A large gap between $\lambda_{k+1}$ and $\rho(k)$ implies that

- existence of a $k$-way partition with bounded $\rho(k)$.
- any $(k + 1)$-way partition contains a set with conductance at least $\lambda_{k+1}/2$.
- Graph $G$ has exactly $k$ clusters.

The key parameter: $\Upsilon \triangleq \frac{\lambda_{k+1}}{\rho(k)}$. 
The Structure Theorem

Let $G$ be a $d$-regular graph with $k$ disjoint components $S_1, \ldots, S_k$. 

$\Upsilon = \Omega(k)$ implies that $\text{span}\{f_1, \ldots, f_k\} \approx \text{span}\{\chi_1, \ldots, \chi_k\}$. 

Lemma (Peng-S.-Zanetti, 2017) Define $F(v) = (f_1(v), \ldots, f_k(v))$. 

There are points $p(1), \ldots, p(k)$, s.t. cluster $S_i$ is concentrated around $p(i)$. 

PDEs for Large Data

He Sun
The Structure Theorem

Let $G$ be a $d$-regular graph with $k$ disjoint components $S_1, \ldots, S_k$. For any $1 \leq i \leq k$ let

$$
\chi_i(v) = \begin{cases} 
1 & \text{if } v \in S_i, \\
0 & \text{otherwise.}
\end{cases}
$$
The Structure Theorem

Let $G$ be a $d$-regular graph with $k$ disjoint components $S_1, \ldots, S_k$. For any $1 \leq i \leq k$ let

$$
\chi_i(v) = \begin{cases} 
1 & \text{if } v \in S_i, \\
0 & \text{otherwise.}
\end{cases}
$$

Then

$$\text{span } \{f_1, \ldots, f_k\} = \text{span } \{\chi_1, \ldots, \chi_k\}.$$
The Structure Theorem

Let $G$ be a $d$-regular graph with $k$ disjoint components $S_1, \ldots, S_k$. For any $1 \leq i \leq k$ let

$$
\chi_i(v) = \begin{cases} 
1 & \text{if } v \in S_i, \\
0 & \text{otherwise.}
\end{cases}
$$

Then

$$
\text{span} \{ f_1, \ldots, f_k \} = \text{span} \{ \chi_1, \ldots, \chi_k \}.
$$

**Lemma (Peng-S.-Zanetti, 2017)**

$\Upsilon = \Omega(k)$ implies that $\text{span} \{ f_1, \ldots, f_k \} \approx \text{span} \{ \chi_1, \ldots, \chi_k \}$. 
Let $G$ be a $d$-regular graph with $k$ disjoint components $S_1, \ldots, S_k$. For any $1 \leq i \leq k$ let

$$
\chi_i(v) = \begin{cases} 
1 & \text{if } v \in S_i, \\
0 & \text{otherwise}.
\end{cases}
$$

Then

$$
\text{span } \{f_1, \ldots, f_k\} = \text{span } \{\chi_1, \ldots, \chi_k\}.
$$

Lemma (Peng-S.-Zanetti, 2017)

$\Upsilon = \Omega(k)$ implies that $\text{span } \{f_1, \ldots, f_k\} \approx \text{span } \{\chi_1, \ldots, \chi_k\}$.

Define $F(v) = (f_1(v), \ldots, f_k(v))$. 
The Structure Theorem

Let $G$ be a $d$-regular graph with $k$ disjoint components $S_1, \ldots, S_k$. For any $1 \leq i \leq k$ let

$$
\chi_i(v) = \begin{cases} 
1 & \text{if } v \in S_i, \\
0 & \text{otherwise.}
\end{cases}
$$

Then

$$\text{span} \{f_1, \ldots, f_k\} = \text{span} \{\chi_1, \ldots, \chi_k\}.$$

Lemma (Peng-S.-Zanetti, 2017)

$\Upsilon = \Omega(k)$ implies that $\text{span} \{f_1, \ldots, f_k\} \approx \text{span} \{\chi_1, \ldots, \chi_k\}$.

Define $F(v) = (f_1(v), \ldots, f_k(v))$.

There are points $p^{(1)}, \ldots, p^{(k)}$, s.t. cluster $S_i$ is concentrated around $p^{(i)}$. 
Well-Separation Property of the Embedding

\[ \sum_{i=1}^{k} \sum_{u \in S_i} \left\| F(u) - p^{(i)} \right\|^2 \leq \frac{k^2}{\Upsilon}. \]

Points from \( S_i \) concentrate around \( p^{(i)} \).
Well-Separation Property of the Embedding

Distance between different clusters inversely\[\approx\] the smaller cluster.

\[
\sum_{i=1}^{k} \sum_{u \in S_i} \left\| F(u) - p^{(i)} \right\|^2 \leq \frac{k^2}{\Upsilon}.
\]

Points from \( S_i \) concentrate around \( p^{(i)} \)s.

\[
\left\| p^{(i)} \right\|^2 \in \left( \frac{9}{10}, \frac{11}{10} \right) \cdot \frac{1}{|S_i|}
\]

“Bigger” clusters are closer to the origin.
Well-Separation Property of the Embedding

\[
\sum_{i=1}^{k} \sum_{u \in S_i} \left\| F(u) - p^{(i)} \right\|^2 \leq \frac{k^2}{\Upsilon}.
\]

Points from \( S_i \) concentrate around \( p^{(i)} \)s.

\[
\left\| p^{(i)} \right\|^2 \in \left( \frac{9}{10}, \frac{11}{10} \right) \cdot \frac{1}{|S_i|}
\]

“Bigger” clusters are closer to the origin.

\[
\left\| p^{(i)} - p^{(j)} \right\|^2 \geq \frac{1}{k \min\{|S_i|, |S_j|\}}
\]

Distance between different clusters inversely \( \approx \) the smaller cluster.
A Simple Algorithm For Graph Clustering

ASSUME we know the pairwise distances of the points for free!
ASSUME we know the pairwise distances of the points for free!

1. Obtain a set $C$ of candidate centres.

Algorithm

\[
\text{for } i = 1 \text{ to } K = \Theta(k \log k) \text{ do} \\
\quad \text{set } c_i = v \text{ with prob. proportional to } \|F(v)\|^2. \\
\text{return } C \triangleq \{c_1, \ldots, c_K\}.
\]
A Simple Algorithm For Graph Clustering

**ASSUME** we know the pairwise distances of the points for free!

1. Obtain a set $C$ of candidate centres.

   **Algorithm**

   $\textbf{for } i = 1 \textbf{ to } K = \Theta(k \log k) \textbf{ do}$

   $\quad \textbf{set } c_i = v \textbf{ with prob. proportional to } \|F(v)\|^2.$

   $\textbf{return } C \triangleq \{c_1, \ldots, c_K\}.$

   *With const. prob., each $S_i$ has at least one vertex sampled.*
ASSUME we know the pairwise distances of the points for free!

1. Obtain a set $C$ of candidate centres.

   **Algorithm**
   
   for $i = 1$ to $K = \Theta(k \log k)$ do
   
   set $c_i = v$ with prob. proportional to $\|F(v)\|^2$.
   
   return $C \triangleq \{c_1, \ldots, c_K\}$.
   
   *With const. prob., each $S_i$ has at least one vertex sampled.*

2. Delete points in $C$ “close” to each other, until $|C| = k$. 

PDEs for Large Data

He Sun
A Simple Algorithm For Graph Clustering

**ASSUME** we know the pairwise distances of the points for free!

1. Obtain a set \( C \) of candidate centres.

   **Algorithm**

   ```
   for \( i = 1 \) to \( K = \Theta(k \log k) \) do
     set \( c_i = v \) with prob. proportional to \( \|F(v)\|^2 \).
   return \( C \triangleq \{c_1, \ldots, c_K\} \).
   ```

   *With const. prob., each \( S_i \) has at least one vertex sampled.*

2. Delete points in \( C \) “close” to each other, until \( |C| = k \).

   *With const. prob., each \( S_i \) has exactly one vertex remaining in \( C \).*
**A Simple Algorithm For Graph Clustering**

**ASSUME** we know the pairwise distances of the points for free!

1. Obtain a set $C$ of candidate centres.

   **Algorithm**
   
   ```
   for $i = 1$ to $K = \Theta(k \log k)$ do 
     set $c_i = v$ with prob. proportional to $\|F(v)\|^2$. 
   return $C \triangleq \{c_1, \ldots, c_K\}$.
   ```

   *With const. prob., each $S_i$ has at least one vertex sampled.*

2. Delete points in $C$ “close” to each other, until $|C| = k$.

   *With const. prob., each $S_i$ has exactly one vertex remaining in $C$.*

3. The other $n - k$ vertices find their closest neighbours in $C$. 

---

PDEs for Large Data

He Sun
**A Simple Algorithm For Graph Clustering**

**ASSUME** we know the pairwise distances of the points for free!

1. Obtain a set $C$ of candidate centres.

```
for $i = 1$ to $K = \Theta(k \log k)$ do
    set $c_i = v$ with prob. proportional to $\|F(v)\|^2$.
return $C \triangleq \{c_1, \ldots, c_K\}$.
```

*With const. prob., each $S_i$ has at least one vertex sampled.*

2. Delete points in $C$ “close” to each other, until $|C| = k$.

*With const. prob., each $S_i$ has exactly one vertex remaining in $C$.*

3. The other $n - k$ vertices find their closest neighbours in $C$.

*apply approximate nearest neighbour data structures.*
ASSUME we know the pairwise distances of the points for free!

1. Obtain a set $C$ of candidate centres.

Algorithm

\[
\text{for } i = 1 \text{ to } K = \Theta(k \log k) \text{ do }
\]

\[
\text{set } c_i = v \text{ with prob. proportional to } \|F(v)\|^2.
\]

\[
\text{return } C \triangleq \{c_1, \ldots, c_K\}.
\]

With const. prob., each $S_i$ has at least one vertex sampled.

2. Delete points in $C$ “close” to each other, until $|C| = k$.

With const. prob., each $S_i$ has exactly one vertex remaining in $C$.

3. The other $n - k$ vertices find their closest neighbours in $C$.

apply approximate nearest neighbour data structures.

Runtime is $O(n \cdot \text{poly log } n)$, even for a large value of $k$!
Obtaining the Pairwise Distances via Heat Kernels

Recall the two embeddings discussed so far:

- $F(v) = (f_1(v), \ldots, f_k(v))$
- $\psi_t(v) = (e^{-t\lambda_1} f_1(v), \ldots, e^{-t\lambda_n} f_n(v))$
Recall the two embeddings discussed so far:

- \( F(v) = (f_1(v), \ldots, f_k(v)) \)
- \( \psi_t(v) = (e^{-t \lambda_1} f_1(v), \ldots, e^{-t \lambda_n} f_n(v)) \)

We can compute in \( O(nd \cdot \log^{10} n) \) time an embedding such that, with high probability, it holds that

\[
(1 - \varepsilon)\|F(u) - F(v)\|^2 \leq \|\psi_t(u) - \psi_t(v)\|^2 \leq \|F(u) - F(v)\|^2 + n^{-10}.
\]
Obtaining the Pairwise Distances via Heat Kernels

Recall the two embeddings discussed so far:

- $F(v) = (f_1(v), \ldots, f_k(v))$
- $\psi_t(v) = (e^{-t\lambda_1}f_1(v), \ldots, e^{-t\lambda_n}f_n(v))$

We can compute in $O(nd \cdot \log^{10} n)$ time an embedding such that, with high probability, it holds that

$$(1 - \varepsilon)\|F(u) - F(v)\|^2 \leq \|\psi_t(u) - \psi_t(v)\|^2 \leq \|F(u) - F(v)\|^2 + n^{-10}.$$ 

**Lemma (Peng-S.-Zanetti, 2017)**

**Proof Sketch**

- Johnson-Lindenstrauss transformation
- Algorithm for approximating matrix exponential.
Main Result

Theorem (Peng-S.-Zanetti, 2017)

There is a linear-time algorithm that, for a graph $G$ with $k$ clusters $S_1, \ldots, S_k$ and $\Upsilon = \Omega(k^3)$, outputs a partition $A_1, \ldots, A_k$ such that

$$|A_i \triangle S_i| = O(k^3 \cdot \Upsilon^{-1} \cdot |S_i|).$$
Main Result

**Theorem (Peng-S.-Zanetti, 2017)**

There is a linear-time algorithm that, for a graph $G$ with $k$ clusters $S_1, \ldots, S_k$ and $\Upsilon = \Omega(k^3)$, outputs a partition $A_1, \ldots, A_k$ such that

$$|A_i \triangle S_i| = O(k^3 \cdot \Upsilon^{-1} \cdot |S_i|).$$

- The heat kernel distances

  $$d_t(u, v) = \sum_w (H_t(w, u) - H_t(w, v))^2$$

  do behave differently among edges inside a cluster and edges crossing different clusters.
Main Result

Theorem (Peng-S.-Zanetti, 2017)

There is a linear-time algorithm that, for a graph $G$ with $k$ clusters $S_1, \ldots, S_k$ and $\Upsilon = \Omega(k^3)$, outputs a partition $A_1, \ldots, A_k$ such that

$$|A_i \triangle S_i| = O(k^3 \cdot \Upsilon^{-1} \cdot |S_i|).$$

- The heat kernel distances

$$d_t(u, v) = \sum_w (H_t(w, u) - H_t(w, v))^2$$

  do behave differently among edges inside a cluster and edges crossing different clusters.

- This gives us the first linear-time algorithm for graph clustering.
Main Result

**Theorem (Peng-S.-Zanetti, 2017)**

There is a linear-time algorithm that, for a graph $G$ with $k$ clusters $S_1, \ldots, S_k$ and $\Upsilon = \Omega(k^3)$, outputs a partition $A_1, \ldots, A_k$ such that

$$|A_i \triangle S_i| = O(k^3 \cdot \Upsilon^{-1} \cdot |S_i|).$$

- The heat kernel distances

  $$d_t(u, v) = \sum_w (H_t(w, u) - H_t(w, v))^2$$

  do behave differently among edges inside a cluster and edges crossing different clusters.

- This gives us the first linear-time algorithm for graph clustering.

- Our intuitions are from random walk theory, but our analysis is based on geometry.
Main Result

**Theorem (Peng-S.-Zanetti, 2017)**

There is a linear-time algorithm that, for a graph $G$ with $k$ clusters $S_1, \ldots, S_k$ and $\Upsilon = \Omega(k^3)$, outputs a partition $A_1, \ldots, A_k$ such that

$$|A_i \Delta S_i| = O(k^3 \cdot \Upsilon^{-1} \cdot |S_i|).$$

- The heat kernel distances
  
  $$d_t(u, v) = \sum_w (H_t(w, u) - H_t(w, v))^2$$

  do behave differently among edges inside a cluster and edges crossing different clusters.

- This gives us the first linear-time algorithm for graph clustering.

- Our intuitions are from random walk theory, but our analysis is based on geometry.

- A direct proof based on random walks?
Beyond Graph Clustering

What is the limit of this technique?
Revisit the Graph Expansion Problem

Graph Expansion

Given a $d$-regular graph $G = (V, E)$ as input, find a set $S \subseteq V$ of size $|S| \leq n/2$ of minimum conductance, i.e.,

$$\phi_G(S) = \min_{S': |S'| \leq n/2} \phi_G(S').$$
Revisit the Graph Expansion Problem

Graph Expansion

Given a \( d \)-regular graph \( G = (V, E) \) as input, find a set \( S \subseteq V \) of size \( |S| \leq n/2 \) of minimum conductance, i.e.,

\[
\phi_G(S) = \min_{S' : |S'| \leq n/2} \phi_G(S').
\]

- This is the simplified version of graph clustering (\( k = 2 \) clusters).
Revisit the Graph Expansion Problem

Graph Expansion

Given a $d$-regular graph $G = (V, E)$ as input, find a set $S \subseteq V$ of size $|S| \leq n/2$ of minimum conductance, i.e.,

$$\phi_G(S) = \min_{S': |S'| \leq n/2} \phi_G(S').$$

- This is the simplified version of graph clustering ($k = 2$ clusters).
- NP-hard to approximate, and there is no constant-factor approximation algorithms assuming the small-set expansion conjecture holds.

Arora-Rao-Vazirani, JACM, 2009

Improve the state-of-the-art algorithm by heat kernels?
Revisit the Graph Expansion Problem

Graph Expansion

Given a $d$-regular graph $G = (V, E)$ as input, find a set $S \subseteq V$ of size $|S| \leq n/2$ of minimum conductance, i.e.,

$$\phi_G(S) = \min_{S' : |S'| \leq n/2} \phi_G(S').$$

- This is the simplified version of graph clustering ($k = 2$ clusters).
- NP-hard to approximate, and there is no constant-factor approximation algorithms assuming the small-set expansion conjecture holds.
- The current best approximation algorithm is based on SDP + geometric embedding.  
  
  *Arora-Rao-Vazirani, JACM, 2009*
Revisit the Graph Expansion Problem

Graph Expansion

Given a \( d \)-regular graph \( G = (V, E) \) as input, find a set \( S \subseteq V \) of size \( |S| \leq n/2 \) of minimum conductance, i.e.,

\[
\phi_G(S) = \min_{S': |S'| \leq n/2} \phi_G(S').
\]

- This is the simplified version of graph clustering \((k = 2 \text{ clusters})\).
- NP-hard to approximate, and there is no constant-factor approximation algorithms assuming the small-set expansion conjecture holds.
- The current best approximation algorithm is based on SDP + geometric embedding. \( \text{Arora-Rao-Vazirani, JACM, 2009} \)

Improve the state-of-the-art algorithm by heat kernels?
Grid Graphs

We define a family of graphs \( \{G\}_n \) as follows:

- Every \( G_n \) has \( 3n \) vertices, which form a grid of size \( \sqrt{n} \times 3\sqrt{n} \).
- The weight of every edge in the middle row has weight \( \frac{1}{\sqrt{n}} \), and all the other edges have weight 1.
Grid Graphs

We define a family of graphs \( \{G\}_n \) as follows:

- Every \( G_n \) has \( 3n \) vertices, which form a grid of size \( \sqrt{n} \times 3\sqrt{n} \).
- The weight of every edge in the middle row has weight \( \frac{1}{\sqrt{n}} \), and all the other edges have weight 1.
Grid Graphs

We define a family of graphs \( \{G\}_n \) as follows:

- Every \( G_n \) has \( 3n \) vertices, which form a grid of size \( \sqrt{n} \times 3\sqrt{n} \).
- The weight of every edge in the middle row has weight \( 1/\sqrt{n} \), and all the other edges have weight \( 1 \).
Heat Kernel Distances in the Grid Graphs

\[ \sqrt{n} \text{ rows} \]

\[ 3\sqrt{n} \text{ coloums} \]

the sparest cut
Summary

- Heat kernel is a basic notion in PDEs.
Summary

- Heat kernel is a basic notion in PDEs.

- We studied its connections to random walks and geometry, which allows us to design the first linear-time algorithm for graph clustering.
Summary

- Heat kernel is a basic notion in PDEs.

- We studied its connections to random walks and geometry, which allows us to design the first linear-time algorithm for graph clustering.

- This leaves us a number of interesting questions, including the powers and limits of PDEs for processing large datasets.
Summary

- Heat kernel is a basic notion in PDEs.

- We studied its connections to random walks and geometry, which allows us to design the first linear-time algorithm for graph clustering.

- This leaves us a number of interesting questions, including the powers and limits of PDEs for processing large datasets.

THANK YOU!