Pairwise Statistics

- Problems involving pairwise distance computation
Pairwise Statistics

- Problems involving pairwise distance computation
  - All nearest-neighbor search
Pairwise Statistics

- Problems involving pairwise distance computation
  - All nearest-neighbor search
  - All point kernel density estimation / kernel machines
Pairwise Statistics

- Problems involving pairwise distance computation
  - All nearest-neighbor search
  - All point kernel density estimation / kernel machines

- Single-query case: Cumulative effect of a reference set $\mathcal{R}$ with $N$ points in Euclidean space on a query point $q$
Pairwise Statistics

- Problems involving pairwise distance computation
  - All nearest-neighbor search
  - All point kernel density estimation / kernel machines

- Single-query case: Cumulative effect of a reference set $\mathcal{R}$ with $N$ points in Euclidean space on a query point $q$

- Multiple queries case (all-query): Combined effects of $\mathcal{R}$ on each point $q$ in a query set $Q$ of size $O(N)$. 
Pairwise Statistics

- Problems involving pairwise distance computation
  - All nearest-neighbor search
  - All point kernel density estimation / kernel machines

- Single-query case: Cumulative effect of a reference set $\mathcal{R}$ with $N$ points in Euclidean space on a query point $q$

- Multiple queries case (all-query): Combined effects of $\mathcal{R}$ on each point $q$ in a query set $\mathcal{Q}$ of size $O(N)$.

- Naïve Solution: Quadratic $O(N^2)$ runtime
Pairwise Statistics

- Problems involving pairwise distance computation
  - All nearest-neighbor search
  - All point kernel density estimation / kernel machines

- Single-query case: Cumulative effect of a reference set $\mathcal{R}$ with $N$ points in Euclidean space on a query point $q$

- Multiple queries case (all-query): Combined effects of $\mathcal{R}$ on each point $q$ in a query set $Q$ of size $O(N)$.

- Naïve Solution: Quadratic $O(N^2)$ runtime

- Sub-problem: magnitude of effect of point $r$ on point $q$ inversely proportional to their distance $d(q, r)$.
Pairwise Statistics

- Problems involving pairwise distance computation
  - All nearest-neighbor search
  - All point kernel density estimation / kernel machines

- Single-query case: Cumulative effect of a reference set \( \mathcal{R} \) with \( N \) points in Euclidean space on a query point \( q \)

- Multiple queries case (all-query): Combined effects of \( \mathcal{R} \) on each point \( q \) in a query set \( Q \) of size \( O(N) \).

- Naïve Solution: Quadratic \( O(N^2) \) runtime

- Sub-problem: magnitude of effect of point \( r \) on point \( q \) inversely proportional to their distance \( d(q, r) \).

- Solution method used: \textit{spatial-partitioning tree datastructures} built on \( \mathcal{R} \)
Existing Algorithms: Single-query Case

Existing Algorithms: All-query

Contributions

Trees: Structure
Trees: Properties
Tree NN: Algorithm and Analysis
Dual-tree NN: Algorithm
Dual-tree NN: Analysis
Single-tree KDE: Algorithm
Single-tree KDE (contd.)
Tree Approximate KDE: Approximate potential summation
Multiple queries on a single reference tree
Existing Algorithms: Single-query Case

- Multiple queries on a single reference tree
  - $kd$-tree (Freidman, Bentley, Finkel, '77): Expected $O(\log N)$ for NN of a single query, also used for KDE (no runtime bounds).
Multiple queries on a single reference tree

- *kd*-tree (Freidman, Bentley, Finkel, ’77): Expected $O(\log N)$ for NN of a single query, also used for KDE (no runtime bounds).

- Barnes & Hut, ’86: $O(\log N)$ bound on approximate potential computation, no error guarantees.
Existing Algorithms: Single-query Case

- Multiple queries on a single reference tree
  - \( kd \)-tree (Freidman, Bentley, Finkel,’77): Expected \( O(\log N) \) for NN of a single query, also used for KDE (no runtime bounds).
  - Barnes & Hut,’86: \( O(\log N) \) bound on approximate potential computation, no error guarantees.
  - Karger & Ruhl,’02: Randomized algorithm for \( O(\log N) \) single NN query for low-intrinsic-dimensional data.
Existing Algorithms: Single-query Case

- Multiple queries on a single reference tree
  - $kd$-tree (Freidman, Bentley, Finkel, '77): Expected $O(\log N)$ for NN of a single query, also used for KDE (no runtime bounds).
  - Barnes & Hut, '86: $O(\log N)$ bound on approximate potential computation, no error guarantees.
  - Karger & Ruhl, '02: Randomized algorithm for $O(\log N)$ single NN query for low-intrinsic-dimensional data.
  - Navigating Nets data structure (Krauthgamer & Lee, '04) for $O(\log N)$ NN query for low-intrinsic-dimensional data (more robust notion)
Existing Algorithms: Single-query Case

- Multiple queries on a single reference tree
  - *kd*-tree (Freidman, Bentley, Finkel, ’77): Expected $O(\log N)$ for NN of a single query, also used for KDE (no runtime bounds).
  - Barnes & Hut, ’86: $O(\log N)$ bound on approximate potential computation, no error guarantees.
  - Karger & Ruhl, ’02: Randomized algorithm for $O(\log N)$ single NN query for low-intrinsic-dimensional data.
  - Navigating Nets data structure (Krauthgamer & Lee, ’04) for $O(\log N)$ NN query for low-intrinsic-dimensional data (more robust notion)
  - Cover trees (Beygelzimer, Kakade & Langford, ’06) for $O(\log N)$ NN query (algorithm efficient in practice too).
Existing Algorithms: Single-query Case

- Multiple queries on a single reference tree
  - \textit{kd}-tree (Freidman, Bentley, Finkel, ’77): Expected $O(\log N)$ for NN of a single query, also used for KDE (no runtime bounds).
  - Barnes & Hut, ’86: $O(\log N)$ bound on approximate potential computation, no error guarantees.
  - Karger & Ruhl, ’02: Randomized algorithm for $O(\log N)$ single NN query for low-intrinsic-dimensional data.
  - Navigating Nets data structure (Krauthgamer & Lee, ’04) for $O(\log N)$ NN query for low-intrinsic-dimensional data (more robust notion)
  - Cover trees (Beygelzimer, Kakade & Langford, ’06) for $O(\log N)$ NN query (algorithm efficient in practice too).

- All-query case: Single-tree algorithms improves $O(N^2)$ to at best $O(N \log N)$
Existing Algorithms: All-query Case

- Existing all-query algorithm/analysis: Only monochromatic case
Existing Algorithms: All-query Case

- Existing all-query algorithm/analysis: Only monochromatic case
  - Fast Multipole Method (FMM) (Greengard & Rokhlin, ’87) - algorithm for $N$-body particle simulation with non-rigorous $O(N)$ runtime bound
Existing Algorithms: All-query Case

- Existing all-query algorithm/analysis: Only monochromatic case
  - Fast Multipole Method (FMM) (Greengard & Rokhlin,'87) - algorithm for $N$-body particle simulation with non-rigorous $O(N)$ runtime bound
  - Methods based on well-separated-pair-decomposition (WSPD) (Callahan & Kosaraju,’95) for all-NN and particle simulation with $O(N)$ runtime bound - inefficient in practice, not clearly extendible to bichromatic case
Existing Algorithms: All-query Case

- Existing all-query algorithm/analysis: Only monochromatic case
  - Fast Multipole Method (FMM) (Greengard & Rokhlin,’87) - algorithm for $N$-body particle simulation with non-rigorous $O(N)$ runtime bound
  - Methods based on well-separated-pair-decomposition (WSPD) (Callahan & Kosaraju,’95) for all-NN and particle simulation with $O(N)$ runtime bound - inefficient in practice, not clearly extendible to bichromatic case

- Dual tree algorithms: Generalization to bichromatic sets
Existing Algorithms: All-query Case

- Existing all-query algorithm/analysis: Only monochromatic case
  - Fast Multipole Method (FMM) (Greengard & Rokhlin, ’87) - algorithm for $N$-body particle simulation with non-rigorous $O(N)$ runtime bound
  - Methods based on well-separated-pair-decomposition (WSPD) (Callahan & Kosaraju, ’95) for all-NN and particle simulation with $O(N)$ runtime bound - inefficient in practice, not clearly extendible to bichromatic case

- Dual tree algorithms: Generalization to bichromatic sets
  - builds tree also on $Q$
Existing Algorithms: All-query Case

- Existing all-query algorithm/analysis: Only monochromatic case
  - Fast Multipole Method (FMM) (Greengard & Rokhlin,’87) - algorithm for $N$-body particle simulation with non-rigorous $O(N)$ runtime bound
  - Methods based on well-separated-pair-decomposition (WSPD) (Callahan & Kosaraju,’95) for all-NN and particle simulation with $O(N)$ runtime bound - inefficient in practice, not clearly extendible to bichromatic case

- Dual tree algorithms: Generalization to bichromatic sets
  - builds tree also on $Q$
  - descends both trees simultaneously
Existing Algorithms: All-query Case

- Existing all-query algorithm/analysis: Only monochromatic case
  - Fast Multipole Method (FMM) (Greengard & Rokhlin,’87) - algorithm for $N$-body particle simulation with non-rigorous $O(N)$ runtime bound
  - Methods based on well-separated-pair-decomposition (WSPD) (Callahan & Kosaraju,’95) for all-NN and particle simulation with $O(N)$ runtime bound - inefficient in practice, not clearly extendible to bichromatic case

- Dual tree algorithms: Generalization to bichromatic sets
  - builds tree also on $Q$
  - descends both trees simultaneously
  - significantly more efficient in practice than single-tree algorithms
Existing Algorithms: All-query Case

- Existing all-query algorithm/analysis: Only monochromatic case
  - Fast Multipole Method (FMM) (Greengard & Rokhlin,’87) - algorithm for $N$-body particle simulation with non-rigorous $O(N)$ runtime bound
  - Methods based on well-separated-pair-decomposition (WSPD) (Callahan & Kosaraju,’95) for all-NN and particle simulation with $O(N)$ runtime bound - inefficient in practice, not clearly extendible to bichromatic case

- Dual tree algorithms: Generalization to bichromatic sets
  - builds tree also on $Q$
  - descends both trees simultaneously
  - significantly more efficient in practice than single-tree algorithms
  - conjectured to be $O(N)$ with cover trees
Our Contributions

We provide $O(N)$ runtime bounds for the following problems:
Our Contributions

We provide $O(N)$ runtime bounds for the following problems:

- **All Nearest-neighbors**: $\forall q \in Q$, find

$$r^*(q) = \arg \min_{r \in R} d(q, r).$$
Our Contributions

We provide $O(N)$ runtime bounds for the following problems:

- **All Nearest-neighbors:** $\forall q \in Q$, find
  
  $$r^*(q) = \arg \min_{r \in \mathcal{R}} d(q, r).$$

- **Kernel summations:** Given a kernel function $K(\cdot)$, $\forall q \in Q$, compute the kernel summations
  
  $$f(q) = \sum_{r \in \mathcal{R}} K(d(q, r)), \ \forall q \in Q.$$
Our Contributions

We provide $O(N)$ runtime bounds for the following problems:

- **All Nearest-neighbors:** $\forall q \in Q$, find
  \[ r^*(q) = \arg \min_{r \in \mathcal{R}} d(q, r). \]

- **Kernel summations:** Given a kernel function $K(\cdot)$, $\forall q \in Q$, compute the kernel summations
  \[ f(q) = \sum_{r \in \mathcal{R}} K(d(q, r)), \forall q \in Q. \]

- **N-body potential calculation:** $\forall q \in Q$ compute the net electrostatic or gravitational potential
  \[ f(q) = \sum_{r \in \mathcal{R}, r \neq q} d(q, r)^{-1}. \]
Cover trees: Structure

- Tree $T$ stores $\mathcal{R}$ in the form of a levelled tree. Each level is indexed by an integer scale $i$ which decreases as the tree is descended. Let $C_i$ denote the set of nodes at scale $i$:
Cover trees: Structure

- Tree $T$ stores $\mathcal{R}$ in the form of a levelled tree. Each level is indexed by an integer scale $i$ which decreases as the tree is descended. Let $C_i$ denote the set of nodes at scale $i$:
  - nesting invariant: $C_i \subset C_{i-1}$
Cover trees: Structure

- Tree $T$ stores $\mathcal{R}$ in the form of a levelled tree. Each level is indexed by an integer scale $i$ which decreases as the tree is descended. Let $C_i$ denote the set of nodes at scale $i$:
  - nesting invariant: $C_i \subset C_{i-1}$
  - covering invariant: For every $p \in C_{i-1}$, there exists a $q \in C_i$ satisfying $d(p, q) \leq 2^i$, and exactly one such $q$ is a parent of $p$. 
Cover trees: Structure

- Tree $T$ stores $\mathcal{R}$ in the form of a levelled tree. Each level is indexed by an integer scale $i$ which decreases as the tree is descended. Let $C_i$ denote the set of nodes at scale $i$:
  - nesting invariant: $C_i \subset C_{i-1}$
  - covering invariant: For every $p \in C_{i-1}$, there exists a $q \in C_i$ satisfying $d(p, q) \leq 2^i$, and exactly one such $q$ is a parent of $p$.
  - separation invariant: For all $p, q \in C_i$, $d(p, q) > 2^i$. 
Cover trees: Structure

- Tree $T$ stores $\mathcal{R}$ in the form of a levelled tree. Each level is indexed by an integer scale $i$ which decreases as the tree is descended. Let $C_i$ denote the set of nodes at scale $i$:
  - nesting invariant: $C_i \subset C_{i-1}$
  - covering invariant: For every $p \in C_{i-1}$, there exists a $q \in C_i$ satisfying $d(p, q) \leq 2^i$, and exactly one such $q$ is a parent of $p$.
  - separation invariant: For all $p, q \in C_i$, $d(p, q) > 2^i$.

- Space requirement: $O(N)$
Cover trees: Structure

- Tree $T$ stores $\mathcal{R}$ in the form of a levelled tree. Each level is indexed by an integer scale $i$ which decreases as the tree is descended. Let $C_i$ denote the set of nodes at scale $i$:
  - nesting invariant: $C_i \subseteq C_{i-1}$
  - covering invariant: For every $p \in C_{i-1}$, there exists a $q \in C_i$ satisfying $d(p, q) \leq 2^i$, and exactly one such $q$ is a parent of $p$.
  - separation invariant: For all $p, q \in C_i$, $d(p, q) > 2^i$.

- Space requirement: $O(N)$
- Construction time: $O(N \log N)$
Cover trees: Structure

- Tree \( T \) stores \( \mathcal{R} \) in the form of a levelled tree. Each level is indexed by an integer scale \( i \) which decreases as the tree is descended. Let \( C_i \) denote the set of nodes at scale \( i \):
  - nesting invariant: \( C_i \subset C_{i-1} \)
  - covering invariant: For every \( p \in C_{i-1} \), there exists a \( q \in C_i \) satisfying \( d(p, q) \leq 2^i \), and exactly one such \( q \) is a parent of \( p \).
  - separation invariant: For all \( p, q \in C_i \), \( d(p, q) > 2^i \).

- Space requirement: \( O(N) \)
- Construction time: \( O(N \log N) \)

- Representations
Cover trees: Structure

- Tree $T$ stores $\mathcal{R}$ in the form of a levelled tree. Each level is indexed by an integer scale $i$ which decreases as the tree is descended. Let $C_i$ denote the set of nodes at scale $i$:
  - nesting invariant: $C_i \subset C_{i-1}$
  - covering invariant: For every $p \in C_{i-1}$, there exists a $q \in C_i$ satisfying $d(p, q) \leq 2^i$, and exactly one such $q$ is a parent of $p$.
  - separation invariant: For all $p, q \in C_i$, $d(p, q) > 2^i$.

- Space requirement: $O(N)$
- Construction time: $O(N \log N)$

- Representations
  - Implicit: Infinitely many levels $C_i$ with the level $C_\infty$ containing a single node which is the root and the level $C_{-\infty}$ containing every point in the dataset as a node.
Cover trees: Structure

- **Tree** $T$ stores $\mathcal{R}$ in the form of a levelled tree. Each level is indexed by an integer scale $i$ which decreases as the tree is descended. Let $C_i$ denote the set of nodes at scale $i$:
  - nesting invariant: $C_i \subseteq C_{i-1}$
  - covering invariant: For every $p \in C_{i-1}$, there exists a $q \in C_i$ satisfying $d(p, q) \leq 2^i$, and exactly one such $q$ is a parent of $p$.
  - separation invariant: For all $p, q \in C_i$, $d(p, q) > 2^i$.

- **Space requirement**: $O(N)$
- **Construction time**: $O(N \log N)$

- **Representations**
  - Implicit: Infinitely many levels $C_i$ with the level $C_{\infty}$ containing a single node which is the root and the level $C_{-\infty}$ containing every point in the dataset as a node.
  - Explicit: Collapses all nodes whose only child is the self-child. (provides the $O(N)$ space bound)
Cover Tree: Properties

- Intrinsic dimension: **Expansion constant** of $\mathcal{R}$ is

\[
\arg\min_{c \geq 2} |B_{\mathcal{R}}(p, 2\rho)| \leq c |B_{\mathcal{R}}(p, \rho)| \quad \forall p \in \mathcal{R}, \forall \rho > 0.
\]

\[
d_{KR}(\mathcal{R}) = \log c.
\]
Cover Tree: Properties

- Intrinsic dimension: **Expansion constant** of $\mathcal{R}$ is

$$\arg \min_{c \geq 2} |B_{\mathcal{R}}(p, 2\rho)| \leq c |B_{\mathcal{R}}(p, \rho)| \quad \forall p \in \mathcal{R}, \forall \rho > 0.$$

$$d_{KR}(\mathcal{R}) = \log c.$$

- Properties of the trees
Cover Tree: Properties

- Intrinsic dimension: **Expansion constant** of $\mathcal{R}$ is

$$\arg \min_{c \geq 2} |B_{\mathcal{R}}(p, 2\rho)| \leq c |B_{\mathcal{R}}(p, \rho)| \ \forall p \in \mathcal{R}, \ \forall \rho > 0.$$ 

$$d_{KR}(\mathcal{R}) = \log c.$$

- Properties of the trees
  - **Width bound**: The number of children of any node $p$ is bounded by $c^4$. 

\[ \text{Intrinsic dimension: Expansion constant of } \mathcal{R} \text{ is} \]

\[ \arg \min_{c \geq 2} |B_{\mathcal{R}}(p, 2\rho)| \leq c |B_{\mathcal{R}}(p, \rho)| \ \forall p \in \mathcal{R}, \ \forall \rho > 0. \]

\[ d_{KR}(\mathcal{R}) = \log c. \]

- **Properties of the trees**
  - **Width bound**: The number of children of any node $p$ is bounded by $c^4$. 

Intrinsic dimension: **Expansion constant** of $\mathcal{R}$ is

$$\arg\min_{c \geq 2} |B_\mathcal{R}(p, 2\rho)| \leq c |B_\mathcal{R}(p, \rho)| \quad \forall p \in \mathcal{R}, \forall \rho > 0.$$  

$$d_{KR}(\mathcal{R}) = \log c.$$  

Properties of the trees

- **Width bound:** The number of children of any node $p$ is bounded by $c^4$.
- **Growth bound:** For all $p \in \mathcal{R}$ and $\rho > 0$, if there exists a point $r \in \mathcal{R}$ such that $2\rho < d(p, r) \leq 3\rho$, then

  $$|B(p, 4\rho)| \geq \left(1 + \frac{1}{c^2}\right) |B(p, \rho)|.$$
Cover Tree: Properties

- Intrinsic dimension: **Expansion constant** of $\mathcal{R}$ is
  \[
  \arg \min_{c \geq 2} |B_{\mathcal{R}}(p, 2\rho)| \leq c |B_{\mathcal{R}}(p, \rho)| \quad \forall p \in \mathcal{R}, \forall \rho > 0.
  \]

  
  \[d_{KR}(\mathcal{R}) = \log c.\]

- Properties of the trees
  - **Width bound**: The number of children of any node $p$ is bounded by $c^4$.
  - **Growth bound**: For all $p \in \mathcal{R}$ and $\rho > 0$, if there exists a point $r \in \mathcal{R}$ such that $2\rho < d(p, r) \leq 3\rho$, then
    \[|B(p, 4\rho)| \geq \left(1 + \frac{1}{c^2}\right) |B(p, \rho)|.
  
  - **Depth bound**: The maximum depth of any point $p$ in the explicit representation is $O(c^2 \log N)$. 
Single-tree NN: Algorithm and analysis

**FindNN**($\mathcal{R}$-Tree $T$, query $q$)

1. **Initialize** $R_\infty = C_\infty$.
2. **for** $i = \infty$ to $-\infty$ **do**
   3. $R = \{\text{Children}(r) : r \in R_i\}$
   4. $R_{i-1} = \{r \in R : d(q, r) \leq d(q, R) + 2^i\}$
   **end for**
3. **return** $\arg \min_{r \in R_{-\infty}} d(q, r)$
Single-tree NN: Algorithm and analysis

\textbf{FindNN}(\mathcal{R}\text{-Tree } T, \text{ query } q)

\textbf{Initialize} \quad R_\infty = C_\infty.

\textbf{for} \ i = \infty \ \textbf{to} \ -\infty \ \textbf{do}

3: \quad R = \{Children(r): r \in R_i\}

\quad R_{i-1} = \{r \in R: d(q, r) \leq d(q, R) + 2^i\}

\textbf{end for}

6: \quad \textbf{return} \ \arg \min_{r \in R_{-\infty}} d(q, r)

- If the dataset \( \mathcal{R} \cup \{q\} \) has expansion constant \( c \), the nearest neighbor of \( q \) can be found in time \( O(c^{12} \log N) \).
Dual-tree NN: Algorithm

\textbf{FindAllNN}(Q-subtree } q_j \text{, } \mathcal{R} \text{-cover set } R_i \text{)}

1. if \( i = -\infty \) then
   \( \forall q \in L(q_j) \) \textbf{return} \( \arg \min_{r \in R_{-\infty}} d(q, r) \).
   \( // L(q_j) \) is the set of all the leaves of the subtree \( q_j \).

3. \textbf{else if } j < i \textbf{ then}
   \( R = \{ \text{Children}(r): r \in R_i \} \)
   \( R_{i-1} = \{ r \in R: d(q_j, r) \leq d(q_j, R) + 2^i + 2^{j+2} \} \)

6. \textbf{FindAllNN} \( (q_j, R_{i-1}) \)

9. \textbf{else}
   \( \forall p_{j-1} \in \text{Children}(q_j) \) \textbf{FindAllNN}(\( p_{j-1}, R_i \))

9. \textbf{end if}
$S'$ - query cover tree, $T$ - reference cover tree. The degree of bichromaticity $\kappa$ of the query-reference pair $(Q, R)$ is the maximum number of descends in $S'$ between any two descends in $T$. 
Dual-tree NN: Analysis

- $S$ - query cover tree, $T$ - reference cover tree. The **degree of bichromaticity** $\kappa$ of the query-reference pair $(Q, R)$ is the maximum number of descends in $S$ between any two descends in $T$.

- $S$ descended completely before $T$ is descended even once $\Rightarrow$ runtime no better than $O(N \log N)$. 
Dual-tree NN: Analysis

- $S'$ - query cover tree, $T$ - reference cover tree. The **degree of bichromaticity** $\kappa$ of the query-reference pair $(Q, R)$ is the maximum number of descends in $S'$ between any two descends in $T$.
  - $S'$ descended completely before $T$ is descended even once $\Rightarrow$ runtime no better than $O(N \log N)$.
  - $\kappa$ quantifies this difference in scale.
**Dual-tree NN: Analysis**

- $S$ - query cover tree, $T$ - reference cover tree. The **degree of bichromaticity** $\kappa$ of the query-reference pair $(Q, R)$ is the maximum number of descends in $S$ between any two descends in $T$.
  - $S$ descended completely before $T$ is descended even once $\Rightarrow$ runtime no better than $O(N \log N)$.
  - $\kappa$ quantifies this difference in scale
  - For monochromatic case, $\kappa = 1$. 
Dual-tree NN: Analysis

- \( S \) - query cover tree, \( T \) - reference cover tree. The **degree of bichromaticity** \( \kappa \) of the query-reference pair \((Q, R)\) is the maximum number of descends in \( S \) between any two descends in \( T \).
  - \( S \) descended completely before \( T \) is descended even once \( \Rightarrow \) runtime no better than \( O(N \log N) \).
  - \( \kappa \) quantifies this difference in scale
  - For monochromatic case, \( \kappa = 1 \).

- \( R \) of size \( N \) and expansion constant \( c_R \), \( Q \) of size \( O(N) \) and expansion constant \( c_Q \), and bounded \( \kappa \) for the \((Q, R)\) pair, **FindAllNN** computes the nearest neighbor in \( R \) of each point in \( Q \) in \( O(c_R^{12} c_Q^{4\kappa} N) \) time.
Dual-tree NN: Analysis

- $S'$ - query cover tree, $T$ - reference cover tree. The **degree of bichromaticity** $\kappa$ of the query-reference pair $(Q, R)$ is the maximum number of descends in $S'$ between any two descends in $T$.
  - $S$ descended completely before $T$ is descended even once $\Rightarrow$ runtime no better than $O(N \log N)$.
  - $\kappa$ quantifies this difference in scale
  - For monochromatic case, $\kappa = 1$.

- $R$ of size $N$ and expansion constant $c_R$, $Q$ of size $O(N)$ and expansion constant $c_Q$, and bounded $\kappa$ for the $(Q, R)$ pair, **FindAllNN** computes the nearest neighbor in $R$ of each point in $Q$ in $O(c_{R}^{12} c_{Q}^{4\kappa} N)$ time.

- **Monochromatic case:** $R$ of size $N$ with expansion constant $c$, **FindAllNN** has a runtime bound of $O(c^{16} N)$. 
Approximation necessary
- Approximation necessary

- Forms of approximation:
Approximation necessary

- Forms of approximation:
  - \( \epsilon \) absolute error bound, if for each exact value \( f(q_i) \) for \( q_i \in Q \), \( \hat{f}(q_i) \) is computed such that
    \[
    \left| \hat{f}(q_i) - f(q_i) \right| \leq N\epsilon.
    \]
Single-tree KDE: Algorithm

- Approximation necessary

- Forms of approximation:
  - $\epsilon$ absolute error bound, if for each exact value $f(q_i)$ for $q_i \in Q$, $\hat{f}(q_i)$ is computed such that $|\hat{f}(q_i) - f(q_i)| \leq N\epsilon$.
  - $\epsilon$ relative error bound, if for each exact value $f(q_i)$ for $q_i \in Q$, $\hat{f}(q_i) \in \mathbb{R}$ is computed such that $|\hat{f}(q_i) - f(q_i)| \leq \epsilon |f(q_i)|$. 
Single-tree KDE: Algorithm

- Approximation necessary
- Forms of approximation:
  - \( \epsilon \) absolute error bound, if for each exact value \( f(q_i) \) for \( q_i \in Q \), \( \hat{f}(q_i) \) is computed such that \( |\hat{f}(q_i) - f(q_i)| \leq N\epsilon \).
  - \( \epsilon \) relative error bound, if for each exact value \( f(q_i) \) for \( q_i \in Q \), \( \hat{f}(q_i) \in \mathbb{R} \) is computed such that \( |\hat{f}(q_i) - f(q_i)| \leq \epsilon |f(q_i)| \).

**Algorithm for \( \epsilon \)-absolute error**

**KernelSum(\( R \)-tree \( T \), query \( q \))**

1. Initialize \( R_\infty = C_\infty, \hat{f}(q) = 0 \)
2. for \( i = \infty \) to \( -\infty \) do
   3. \( R = \{ \text{Children}(r) : r \in R_i \} \)
   4. \( R_{i-1} = \{ r \in R : K_h(d(q, r) - 2^i) - K_h(d(q, r) + 2^i) > \epsilon \} \)
   5. \( \hat{f}(q) = \hat{f}(q) + \sum_{r \in \{ R - R_{i-1} \}} K_h(d(q, r)) \cdot |L(r)| \)
5. end for
6. return \( \hat{f}(q) = \hat{f}(q) + \sum_{r \in R_{-\infty}} K_h(d(q, r)) \)
Single-tree KDE (contd.)

- Given $\mathcal{R}$ of size $N$ with expansion constant $c$, error value $\epsilon$, and a monotonically decreasing smooth non-negative kernel function $K(\cdot)$ concave for $x \in [0, h]$ and convex for $x \in (h, \infty)$ for some bandwidth $h > 0$, $\text{KernelSum}$ computes the $\epsilon$-absolute approximate kernel summation at a query $q$ with a runtime bound of $O\left(c f(\epsilon, K(\cdot), K'(\cdot), K^{(-1)}(\cdot), h) \log N\right)$.
Given $\mathcal{R}$ of size $N$ with expansion constant $c$, error value $\epsilon$, and a monotonically decreasing smooth non-negative kernel function $K(\cdot)$ concave for $x \in [0, h]$ and convex for $x \in (h, \infty)$ for some bandwidth $h > 0$, $\text{KernelSum}$ computes the $\epsilon$-absolute approximate kernel summation at a query $q$ with a runtime bound of $O(c f(\epsilon, K(\cdot), K'(\cdot), K^{(-1)}(\cdot), h) \log N)$.

**Relative Error:**
Single-tree KDE (contd.)

- Given $\mathcal{R}$ of size $N$ with expansion constant $c$, error value $\epsilon$, and a monotonically decreasing smooth non-negative kernel function $K(\cdot)$ concave for $x \in [0, h]$ and convex for $x \in (h, \infty)$ for some bandwidth $h > 0$, $\text{KernelSum}$ computes the $\epsilon$-absolute approximate kernel summation at a query $q$ with a runtime bound of $O\left(cf(\epsilon,K(\cdot),K'(\cdot),K^{(-1)}(\cdot),h) \log N\right)$

- Relative Error:
  - Algorithm: Same as $\text{KernelSum}$ except that the definition of $R_{i-1}$ is changed to:

$$R_{i-1} = \{ r \in R : K(d(q, r) - 2^i) - K(d(q, r) + 2^i) > \frac{\epsilon f(q)}{N} \}$$
Given $\mathcal{R}$ of size $N$ with expansion constant $c$, error value $\epsilon$, and a monotonically decreasing smooth non-negative kernel function $K(\cdot)$ concave for $x \in [0, h]$ and convex for $x \in (h, \infty)$ for some bandwidth $h > 0$, $\text{KernelSum}$ computes the $\epsilon$-absolute approximate kernel summation at a query $q$ with a runtime bound of $O\left(c f(\epsilon, K(\cdot), K'(\cdot), K^{(-1)}(\cdot), h) \log N\right)$.

**Relative Error:**
- Algorithm: Same as $\text{KernelSum}$ except that the definition of $R_{i-1}$ is changed to:

$$R_{i-1} = \{ r \in R : K(d(q, r) - 2^i) - K(d(q, r) + 2^i) > \frac{\epsilon f(q)}{N}\}$$

- Analysis: Relative error is an instance of absolute error by upperbounding the maximum possible relative error, hence can be computed in $O(\log N)$ time.
Dual tree KDE: Algorithm

Initialize $\Delta_f(q) \leftarrow 0 \forall q \in q_\infty$

AllKernelSum($Q$-subtree $q_j$, $R$-cover set $R_i$)

if $i = -\infty$ then
    for $\forall q \in L(q_j)$ do
        $\hat{f}(q) = \hat{f}(q) + \sum_{r \in R_{-\infty}} K(d(q, r)) + \Delta_f(q_j)$
    end for
    $\Delta_f(q_j) = 0$

6: else

if $j < i$ then
    $R = \{ \text{Children}(r) : r \in R_i \}$

9: $R_{i-1} = \{ r \in R : K(d(q_j, r) - 2^i - 2^{j+1}) - K(d(q_j, r) + 2^i + 2^{j+1}) > \epsilon \}$

$\Delta_f(q_j) = \Delta_f(q_j) + \sum_{r \in R \setminus R_{i-1}} K(d(q_j, r)) \cdot |L(r)|$

AllKernelSum($q_j$, $R_{i-1}$)

12: else

for $\forall p_{j-1} \in \text{Children}(q_j)$ do
    $\Delta_f(p_{j-1}) = \Delta_f(p_{j-1}) + \Delta_f(q_j)$
end for

15: AllKernelSum($p_{j-1}$, $R_i$)

end if

$\Delta_f(q_j) = 0$

18: end if

end if
Dual tree Approximate KDE: Analysis

- Absolute error: $O(N)$
Dual tree Approximate KDE: Analysis

- Absolute error: $O(N)$
- Relative error: $O(N)$
Kernel function: \( K(d) = 1/d(q, r) \)
N-body approximate potential summation

- **Kernel function:** \( K(d) = 1/d(q, r) \)

- \( C^2 \) continuous construction of this kernel:

\[
K_e(d) = \begin{cases} 
\frac{1}{d_{\text{min}}} \left( \frac{15}{8} - \frac{5}{4} \left( \frac{d}{d_{\text{min}}} \right)^2 + \frac{3}{8} \left( \frac{d}{d_{\text{min}}} \right)^4 \right), & d < d_{\text{min}} \\
\frac{1}{d}, & d \geq d_{\text{min}}
\end{cases}
\]

where \( d_{\text{min}} = \min_{r \in \mathcal{R}, q \neq r} d(q, r) \)
N-body approximate potential summation

- Kernel function: $K(d) = 1/d(q, r)$
- $C^2$ continuous construction of this kernel:

$$K_e(d) = \begin{cases} \frac{1}{d_{min}} \left( \frac{15}{8} - \frac{5}{4} \left( \frac{d}{d_{min}} \right)^2 + \frac{3}{8} \left( \frac{d}{d_{min}} \right)^4 \right), & d < d_{min} \\ \frac{1}{d}, & d \geq d_{min} \end{cases}$$

where $d_{min} = \min_{r \in \mathcal{R}, q \neq r} d(q, r)$

- Consider $d_{min}$ equivalent to the bandwidth $h$. Algorithmic runtimes follows from the approximate KDE analysis.
Thank You

Any questions