Density Approximation for Kinetic Groups

Max van Mulken Bettina Speckmann Kevin Verbeek
TU Eindhoven

Algorithms Geometry & Applications

Groups

Photos by T.R. Shankar Raman and M.M. Karim

Groups

Finding groups in...

Static point sets:

- K-means clustering
- DBSCAN
- ...

Moving point sets:

- Grouping structure [Buchin et al., 2013]

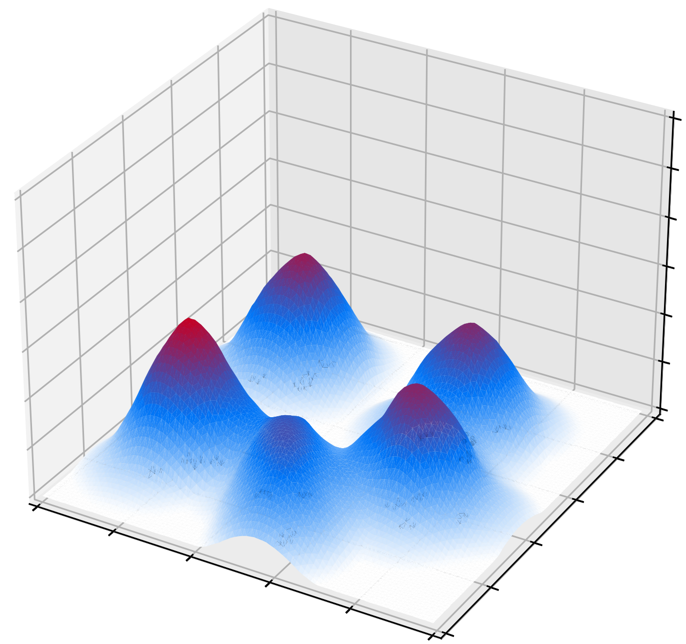
Photos by T.R. Shankar Raman and M.M. Karim

Group density

Group density contains information about underlying behavior

Predator proximity, feeding sites, environmental factors, ...

Function $f(x, y)$ that estimates the density at (x, y)



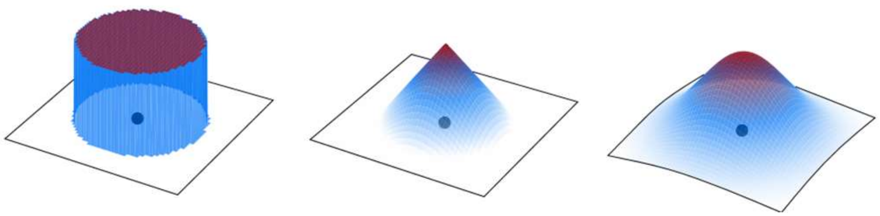
Goal: Maintain $f(x, y)$ as the points move

$\downarrow$

Maintain approximation of $f(x, y)$

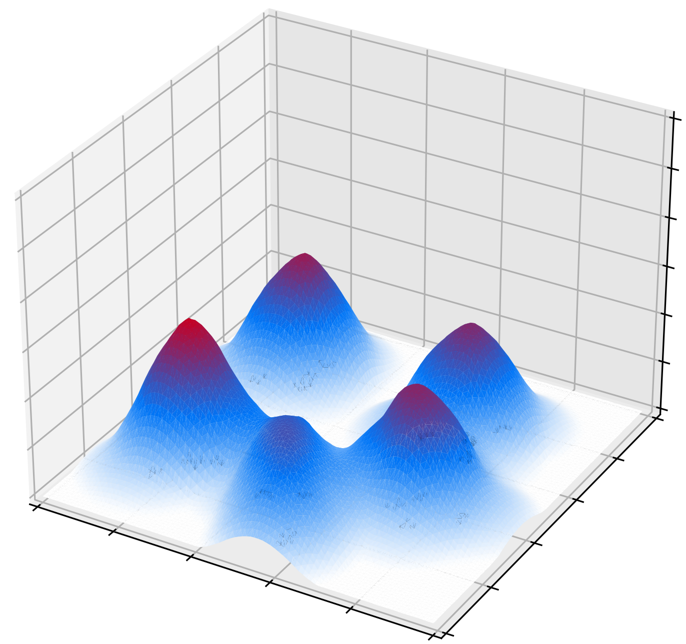
Kernel Density Estimation

Estimate the density function using kernel function K



The density function is then estimated by:

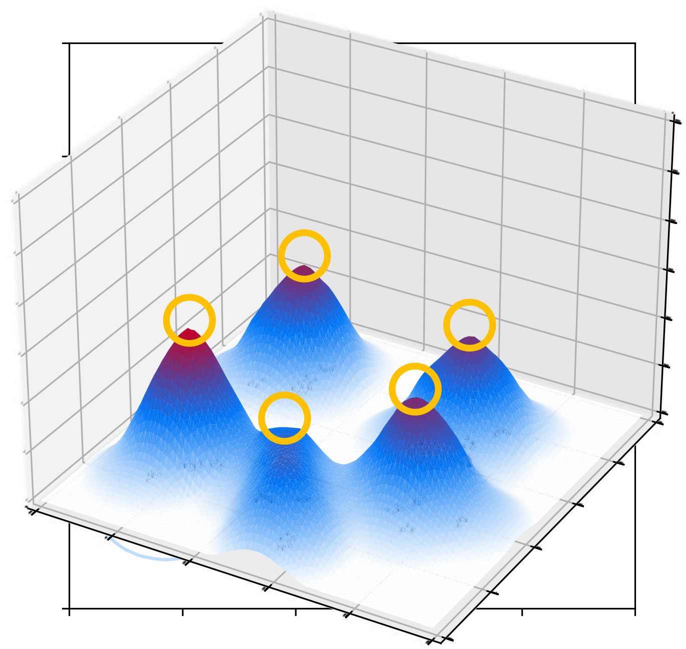
$$KDE_P(x, y) = \frac{1}{n} \sum_{p \in P} K(x - x(p), y - y(p))$$



Kernel Density Estimation

Use $KDE_P(x, y)$ to describe group characteristics:

- Local maxima → Dense clusters
- Contour lines → Group shape
- Kernel radius → Spatial scales



Kernel Density Estimation

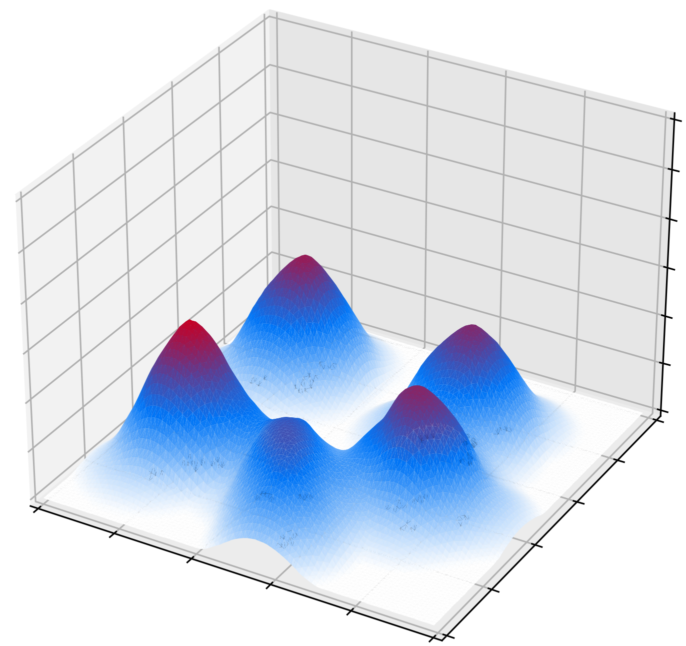
Use $KDE_P(x, y)$ to describe group characteristics:

Useful in practice...

but difficult to give general theoretical guarantees

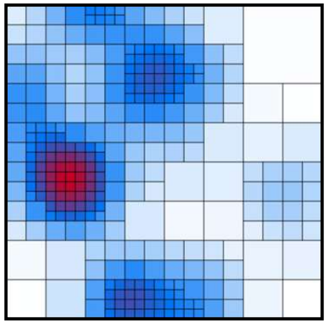
Assumptions:

- Trajectories of points are known
- Piece-wise linear movement
- Within a bounding box

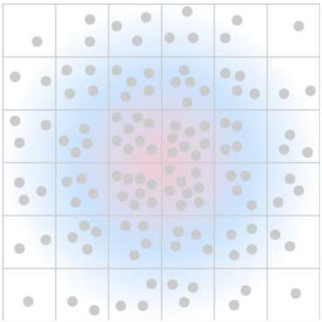


Overview

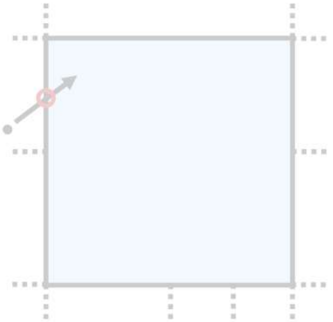
Volume-based quadtree



Discretization

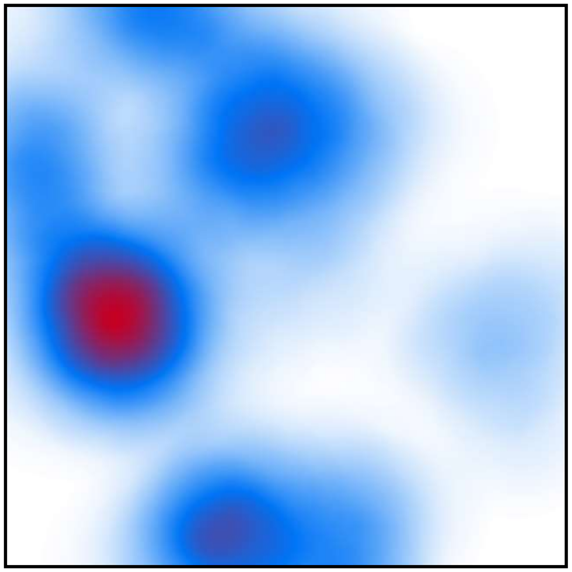


Kinetic Data Structure



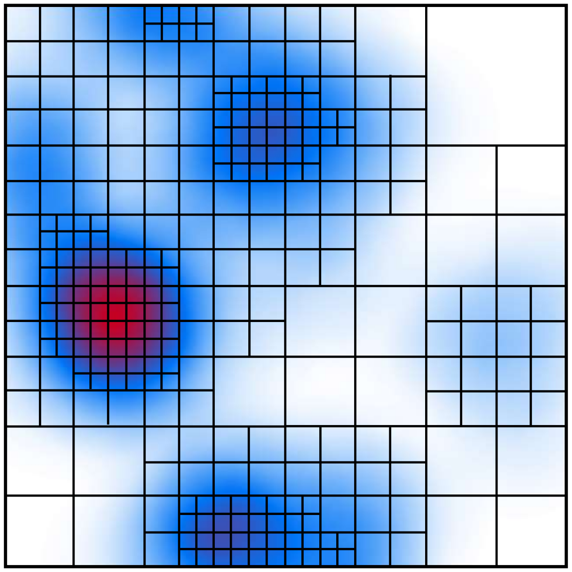
Volume-based Quadtree

Construct quadtree on volume under KDE_P



Volume-based Quadtree

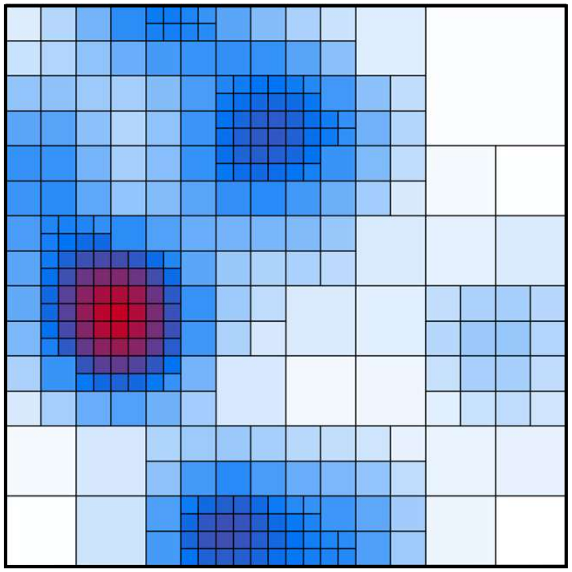
Subdivide cell v if the volume under KDE_P in v exceeds threshold $\rho > 0$



Volume-based Quadtree

Subdivide cell v if the volume under KDE_P in v exceeds threshold $\rho > 0$

Value of density approximation in v equals average value of KDE_P in v



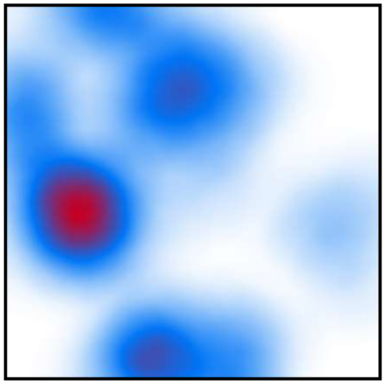
Volume-based Quadtree

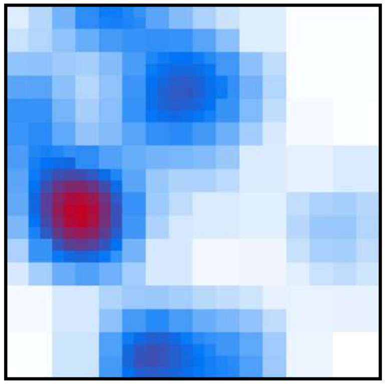
Subdivide cell v if the volume under KDE_P in v exceeds threshold $\rho > 0$

Value of density approximation in v equals average value of KDE_P in v

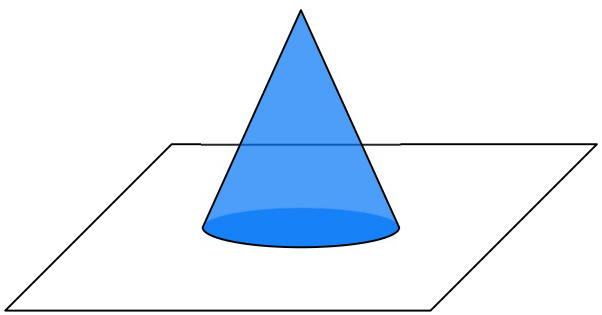
Measure the quality of approximation f_T of KDE_P as

$$|KDE_P(x,y) - f_T(x,y)| \text{ for all } (x,y) \in \mathcal{D}$$

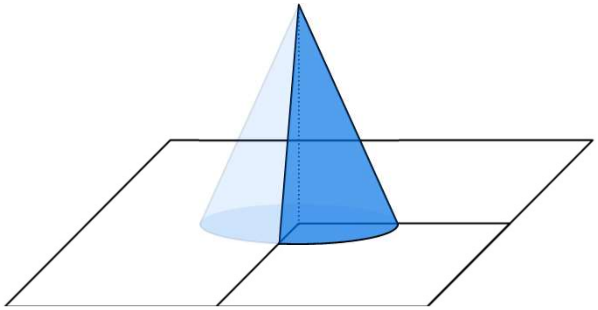




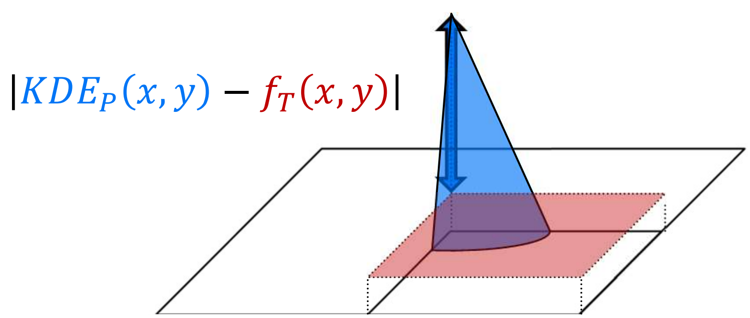
Volume-based Quadtree



Volume-based Quadtree



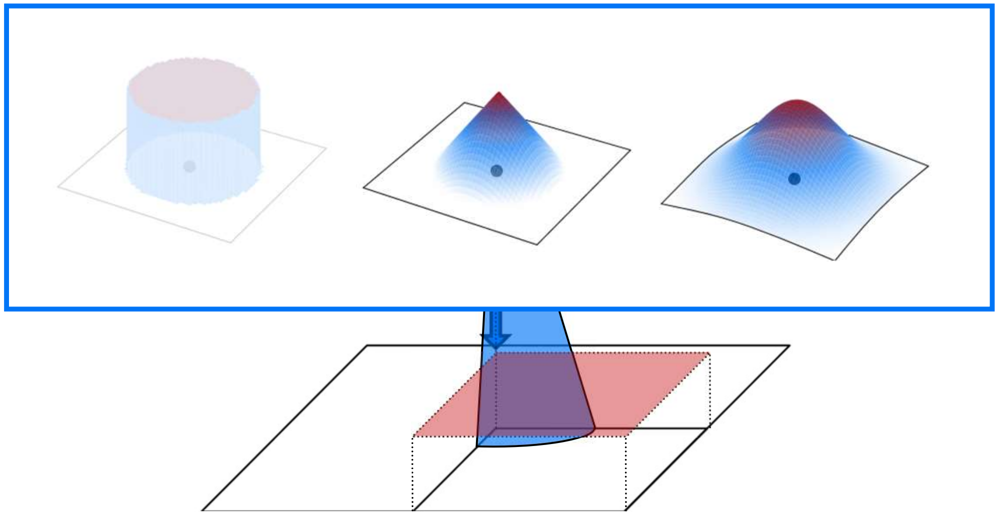
Volume-based Quadtree



Volume-based Quadtree

Bound the maximum slope and height of KDE_P

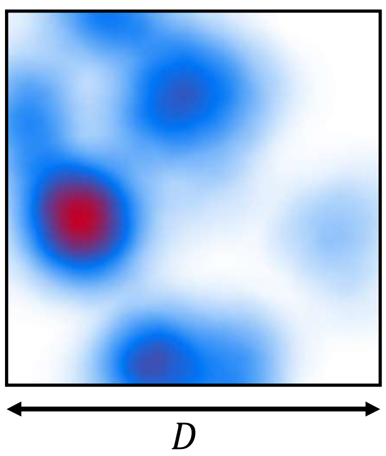
- Lipschitz constant λ, maximum absolute slope in any direction
- Maximum height z^*



Volume-based Quadtree

Bound the maximum slope and height of KDE_P

- Lipschitz constant λ, maximum absolute slope in any direction
- Maximum height z^*
- Domain size D

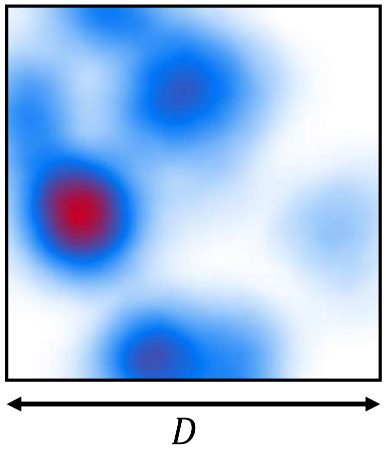


Volume-based Quadtree

For Lipschitz constant λ, maximum height z^* and domain size D:

$$|KDE_P(x,y) - f_T(x,y)| < \varepsilon \text{ for } \rho = O(\varepsilon^3)$$

with polynomial bounds on the size of the quadtree



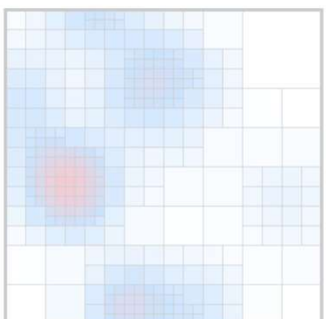
Towards moving points

Difficult to maintain as underlying points move...

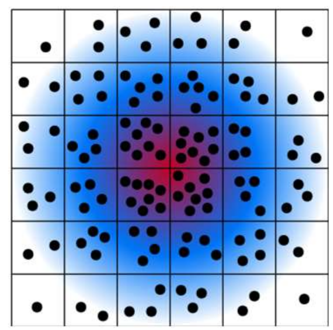
Solution: Approximate volume under KDE_P using a large set of moving points

Overview

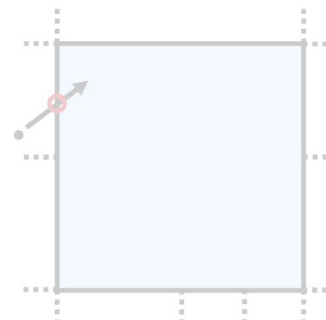
Volume-based quadtree



Discretization



Kinetic Data Structure



Discretization

Construct moving point set $\mathcal{S}$ as a stand-in for the volume under KDE_P

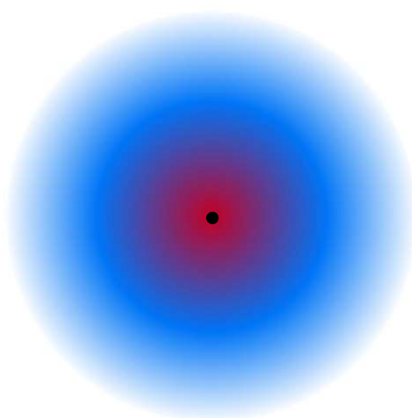
Such that at any time and for any quadtree cell v:

volume under KDE_P in v $\approx$ number of points from $\mathcal{S}$ in v

Discretization

Construct moving point set $\mathcal{S}$ as a stand-in for the volume under KDE_P:

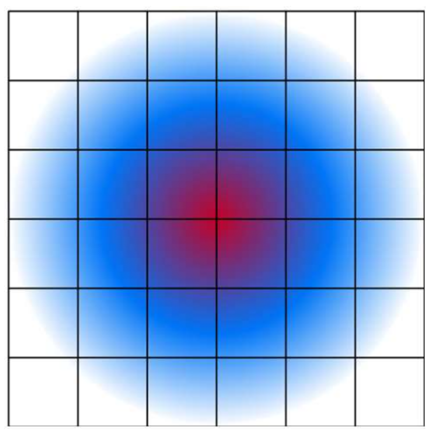
1. Approximate a single kernel K with a set of points S



Discretization

Construct moving point set $\mathcal{S}$ as a stand-in for the volume under KDE_P:

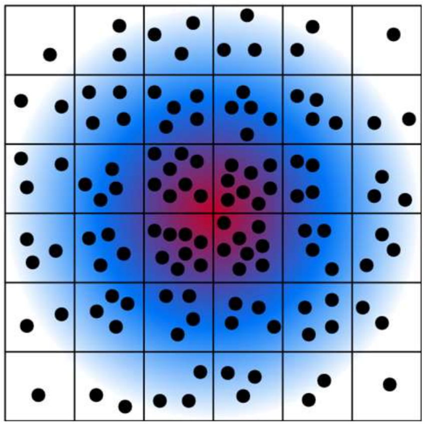
1. Approximate a single kernel K with a set of points S
   a. Overlay $r \times r$ grid



Discretization

Construct moving point set $\mathcal{S}$ as a stand-in for the volume under KDE_P:

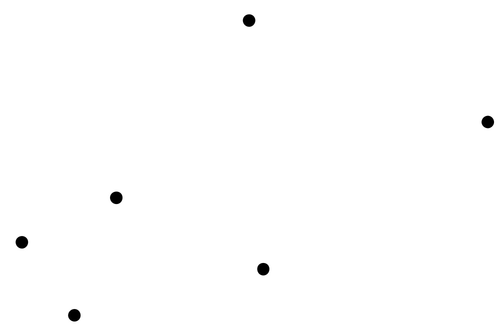
1. Approximate a single kernel K with a set of points S
   a. Overlay $r \times r$ grid
   b. Sample random points in each grid cell proportional to kernel value



Discretization

Construct moving point set $\mathcal{S}$ as a stand-in for the volume under KDE_P:

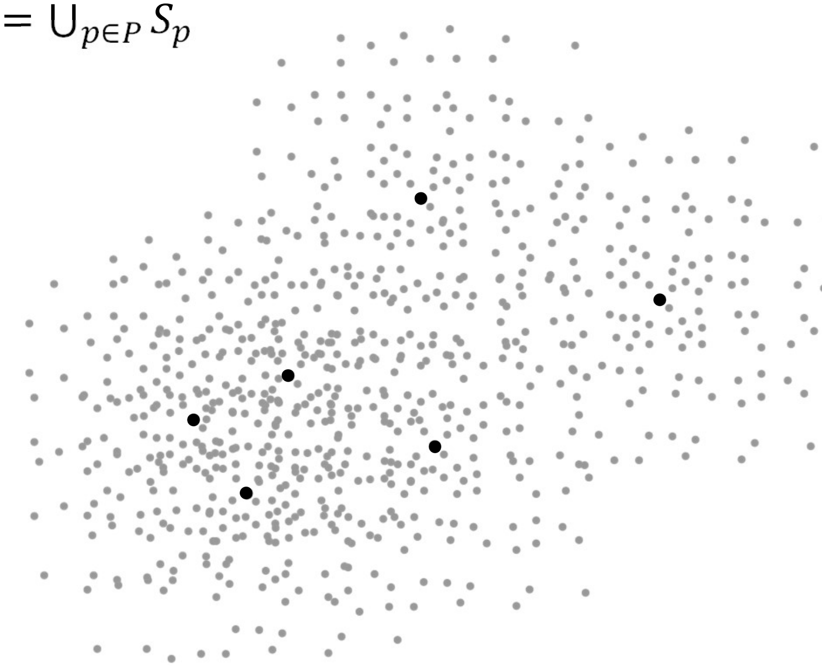
1. Approximate a single kernel K with a set of points S
2. Approximate KDE_P with the union of kernels $\mathcal{S} = \bigcup_{p \in P} S_p$



Discretization

Construct moving point set $\mathcal{S}$ as a stand-in for the volume under KDE_P:

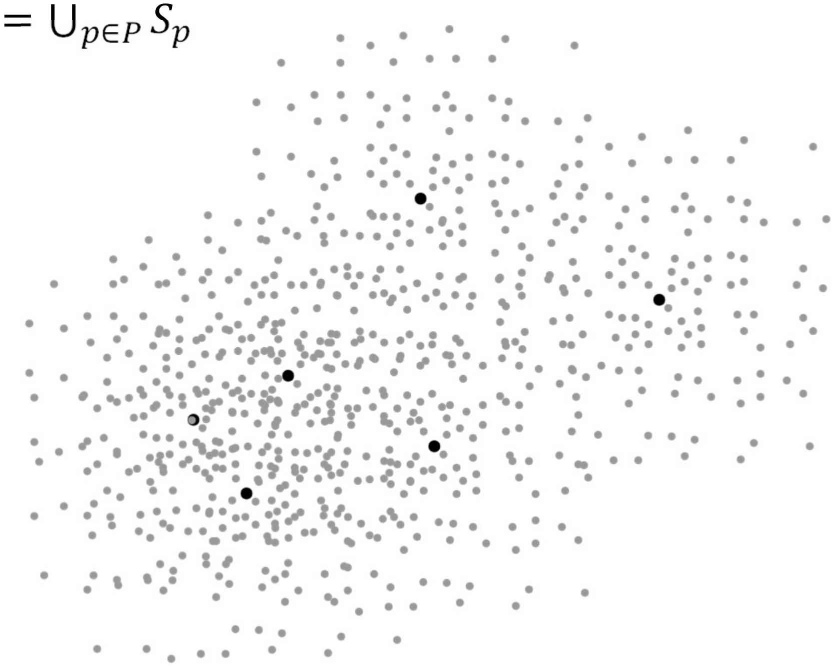
1. Approximate a single kernel K with a set of points S
2. Approximate KDE_P with the union of kernels $\mathcal{S} = \bigcup_{p \in P} S_p$
   a. Assign each input point $p \in P$ a set S_P



Discretization

Construct moving point set $\mathcal{S}$ as a stand-in for the volume under KDE_P:

1. Approximate a single kernel K with a set of points S
2. Approximate KDE_P with the union of kernels $\mathcal{S} = \bigcup_{p \in P} S_p$
   a. Assign each input point $p \in P$ a set S_P
   b. S_P copies the movement of p



Discretization

Construct moving point set $\mathcal{S}$ as a stand-in for the volume under KDE_P:

1. Approximate a single kernel K with a set of points S
2. Approximate KDE_P with the union of kernels $\mathcal{S} = \bigcup_{p \in P} S_p$
3. Reduce the size of $\mathcal{S}$
   a. Take coreset of $\mathcal{S}$ [Agarwal *et al.*, 2005]

Volume-based Quadtree

Construct volume-based quadtree on KDE_P:

Subdivide cell v if the volume under KDE_P in v exceeds threshold $\rho > 0$

Value of density approximation in v equals average value of KDE_P in v

Weight-based Quadtree

Construct volume-based quadtree on KDE_P:

Subdivide cell v if the volume under KDE_P in v exceeds threshold $\rho > 0$

Value of density approximation in v equals average value of KDE_P in v

Construct weight-based quadtree on $\mathcal{S}$:

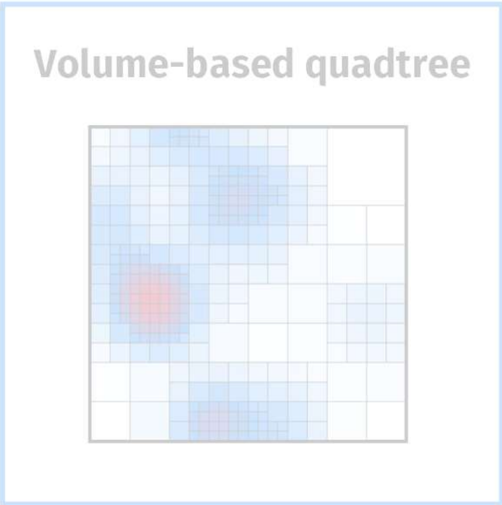
Subdivide cell v if the number of points from $\mathcal{S}$ in v exceeds threshold $\rho > 0$

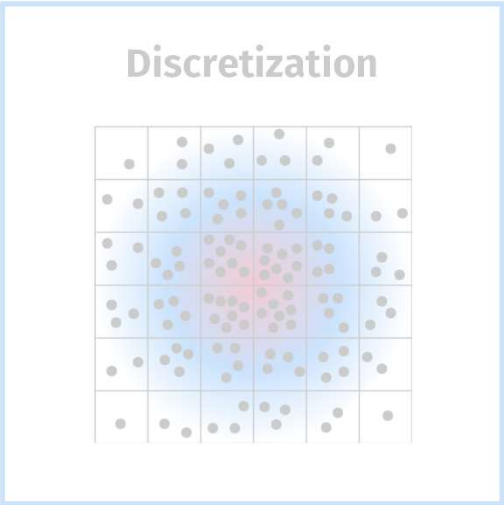
Value of density approximation in v proportional to the number of points in v

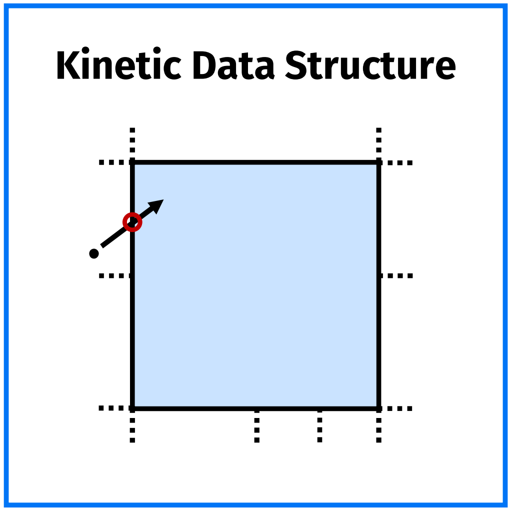
$|KDE_P(x, y) - f_{\tilde{T}}(x, y)| < \varepsilon$ for $\rho = O(\varepsilon^3)$

with $|\mathcal{S}| = O\left(\frac{1}{\varepsilon^8} \log\left(\frac{1}{\varepsilon}\right)\right)$

Overview


Volume-based quadtree


Discretization

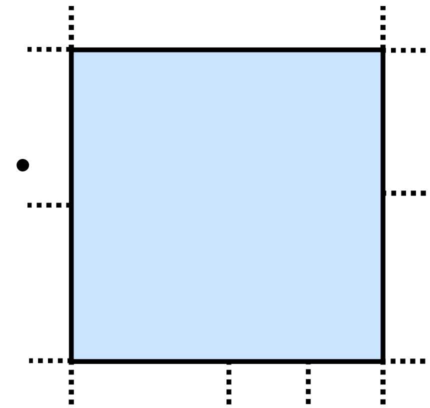

Kinetic Data Structure

Kinetic Data Structure

Compute weight-based quadtree $\tilde{T}$

Goal: Maintain correctness of $\tilde{T}$ as the points move

Observe: $\tilde{T}$ depends only on distribution of P into cells

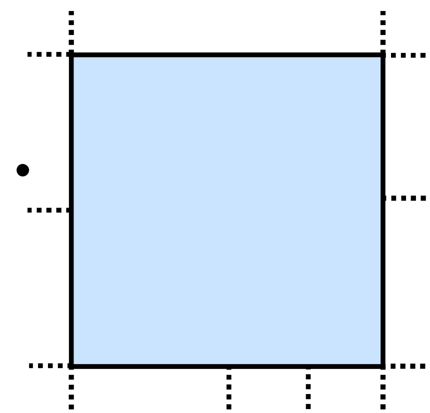


Kinetic Data Structure

Compute weight-based quadtree $\tilde{T}$

Goal: Maintain correctness of $\tilde{T}$ as the points move

Observe: $\tilde{T}$ changes only when a point from $\mathcal{S}$ crosses a cell boundary



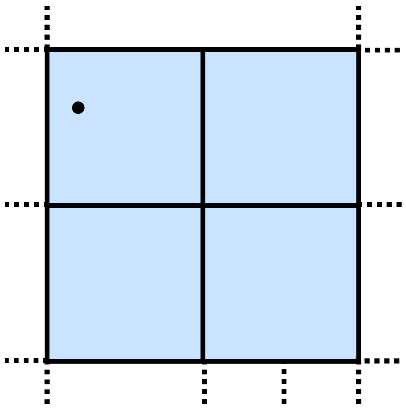
Kinetic Data Structure

Compute weight-based quadtree $\tilde{T}$

Goal: Maintain correctness of $\tilde{T}$ as the points move

Observe: $\tilde{T}$ changes only when a point from $\mathcal{S}$ crosses a cell boundary

- Update cell boundaries
- Update cell heights



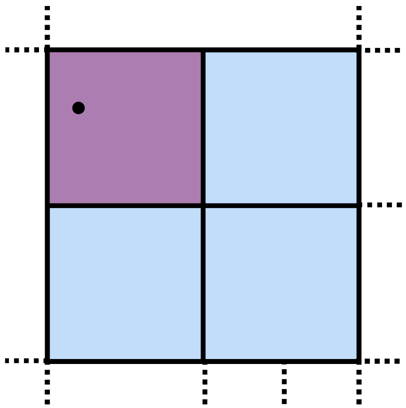
Kinetic Data Structure

Compute weight-based quadtree $\tilde{T}$

Goal: Maintain correctness of $\tilde{T}$ as the points move

Observe: $\tilde{T}$ changes only when a point from $\mathcal{S}$ crosses a cell boundary

- Update cell boundaries
- Update cell heights
- Update local maxima



Sufficient to maintain $\tilde{T}$ ...

But what about its local maxima?

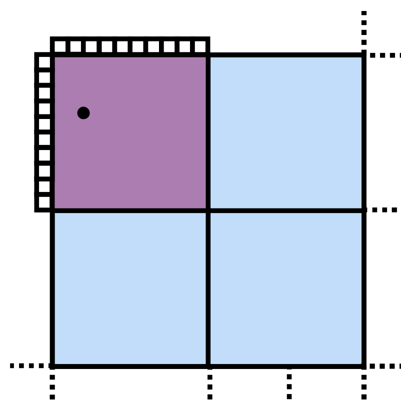
Kinetic Data Structure

Compute weight-based quadtree $\tilde{T}$

Goal: Maintain correctness of $\tilde{T}$ as the points move

Observe: $\tilde{T}$ changes only when a point from $\mathcal{S}$ crosses a cell boundary

- Update cell boundaries
- Update cell heights
- Update local maxima ...



Sufficient to maintain $\tilde{T}$ ...

But what about its local maxima?

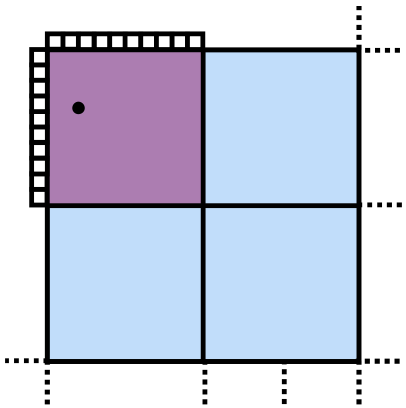
Kinetic Data Structure

Compute weight-based quadtree $\tilde{T}$

Goal: Maintain correctness of $\tilde{T}$ as the points move

Observe: $\tilde{T}$ changes only when a point from $\mathcal{S}$ crosses a cell boundary

- Update cell boundaries
- Update cell heights
- Update local maxima ...



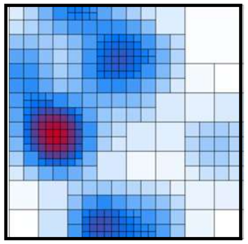
Small cells must have high values ...

Local maxima cannot have small neighbors

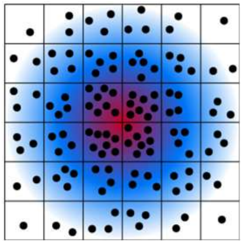
Summary

Theorem. *Let $f = KDE_P$ be a KDE function on a set P of n linearly moving points in $\mathbb{R}^2$. For any $\varepsilon > 0$, there exists a kinetic data structure that maintains an ε-approximation of f.*

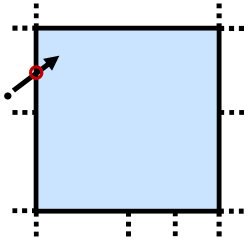
Volume-based quadtree



Discretization



Kinetic Data Structure



Summary

Theorem. *Let $f = KDE_P$ be a KDE function on a set P of n linearly moving points in $\mathbb{R}^2$. For any $\varepsilon > 0$, there exists a kinetic data structure that maintains an ε-approximation of f.*

Future work:

- Engineer practical approach

  Deal with assumptions

  (known trajectories, piece-wise linear, bounding box)

- Use density surface to find other characteristics
  - Group shape, clustering, ...