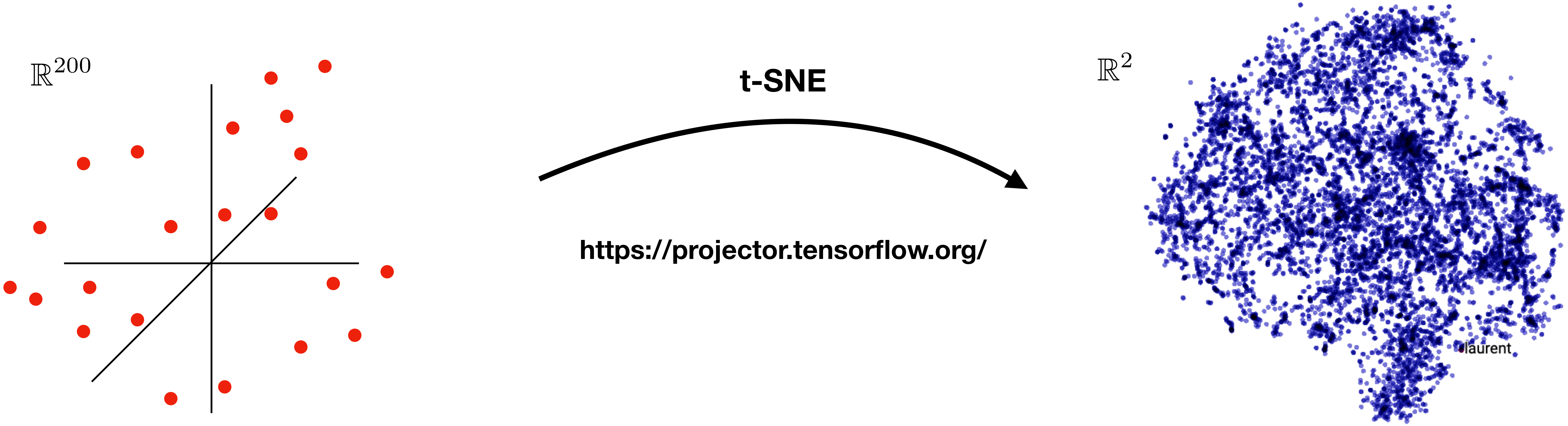
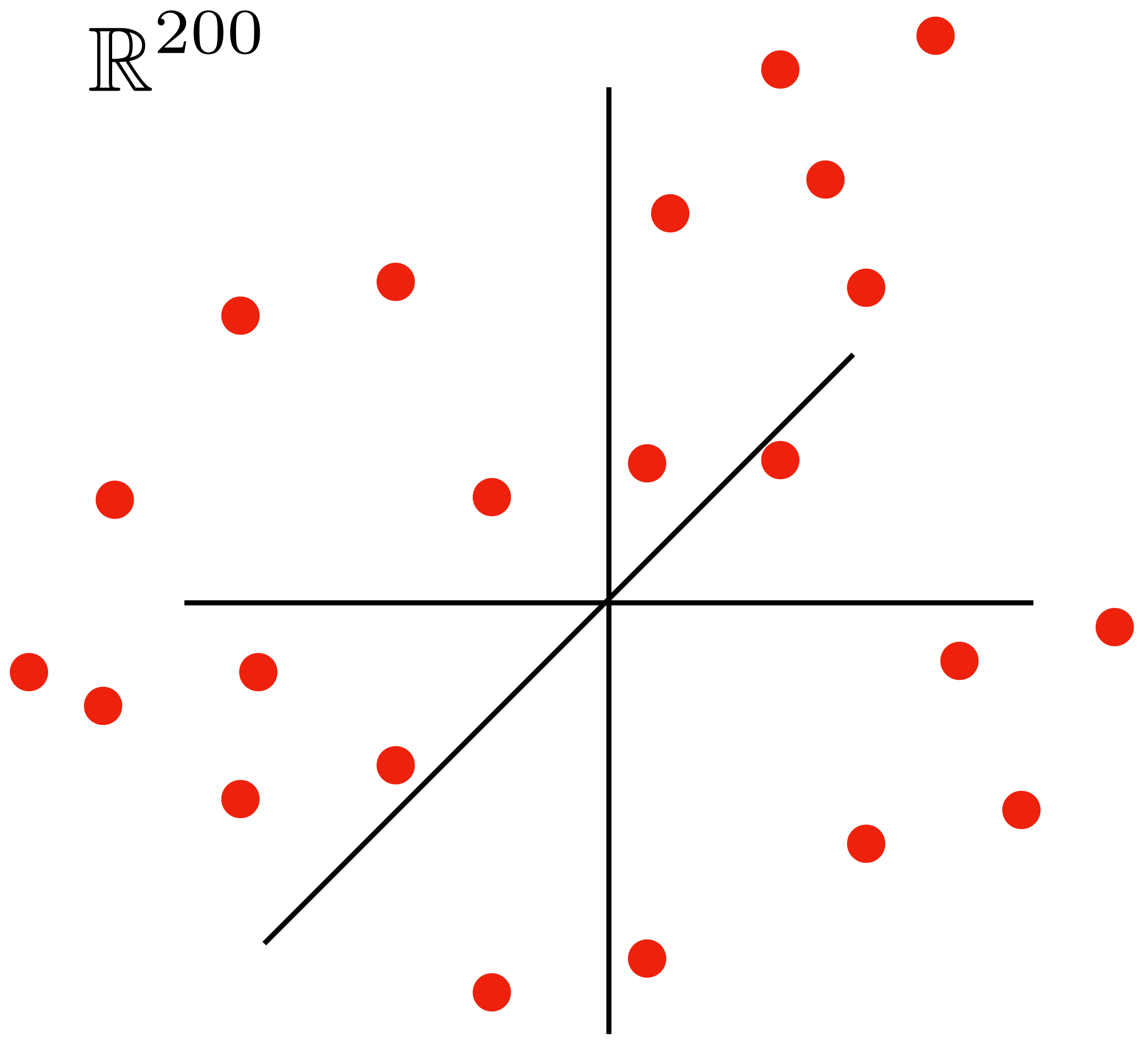
A Quasi-Monte Carlo Data Structure for Smooth Kernel Evaluation

Moses Charikar (Stanford)
Michael Kapralov (EPFL)
Erik Waingarten (Penn)

t-SNE: Visualization technique via "matching" geometric graphs.

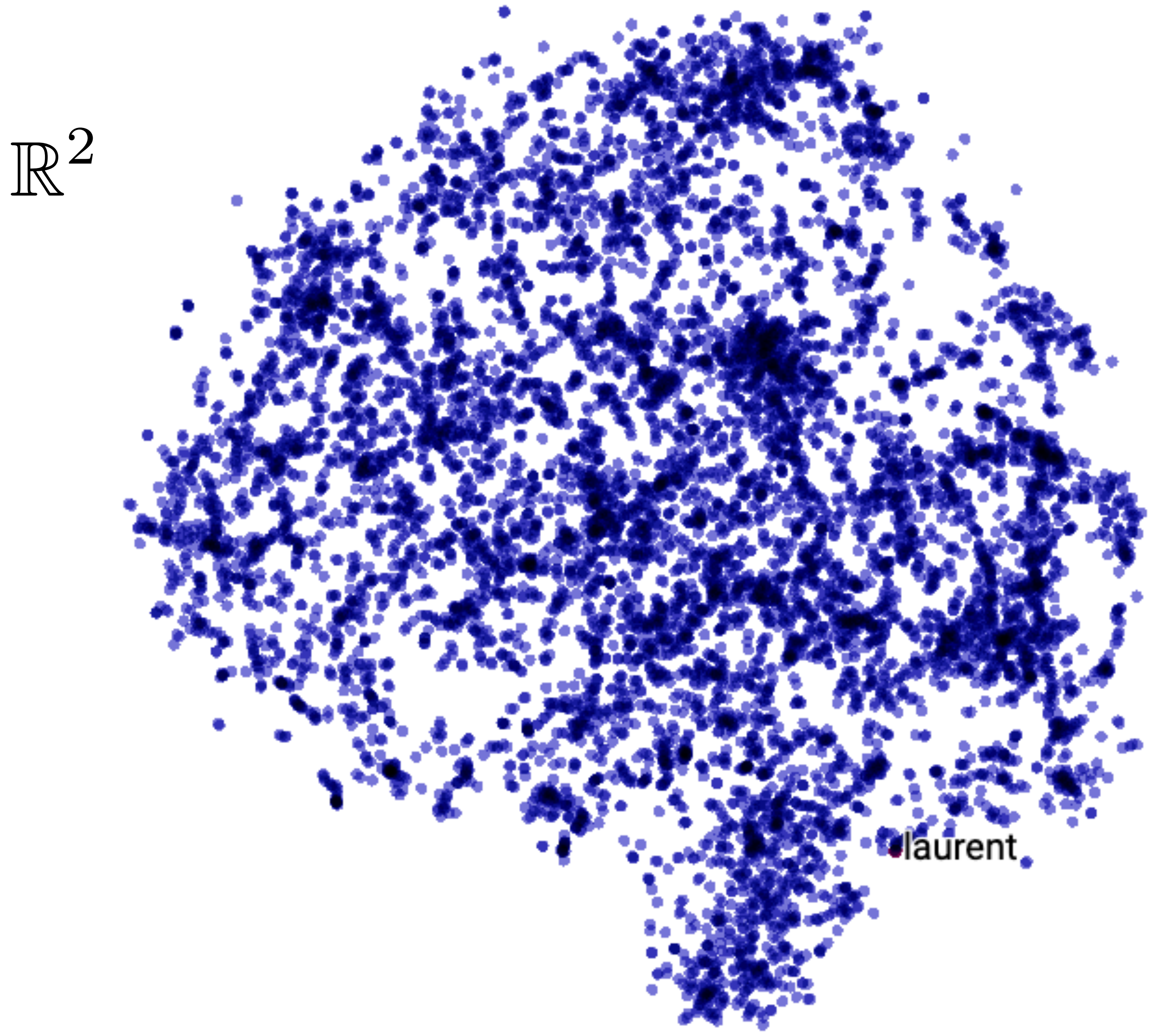


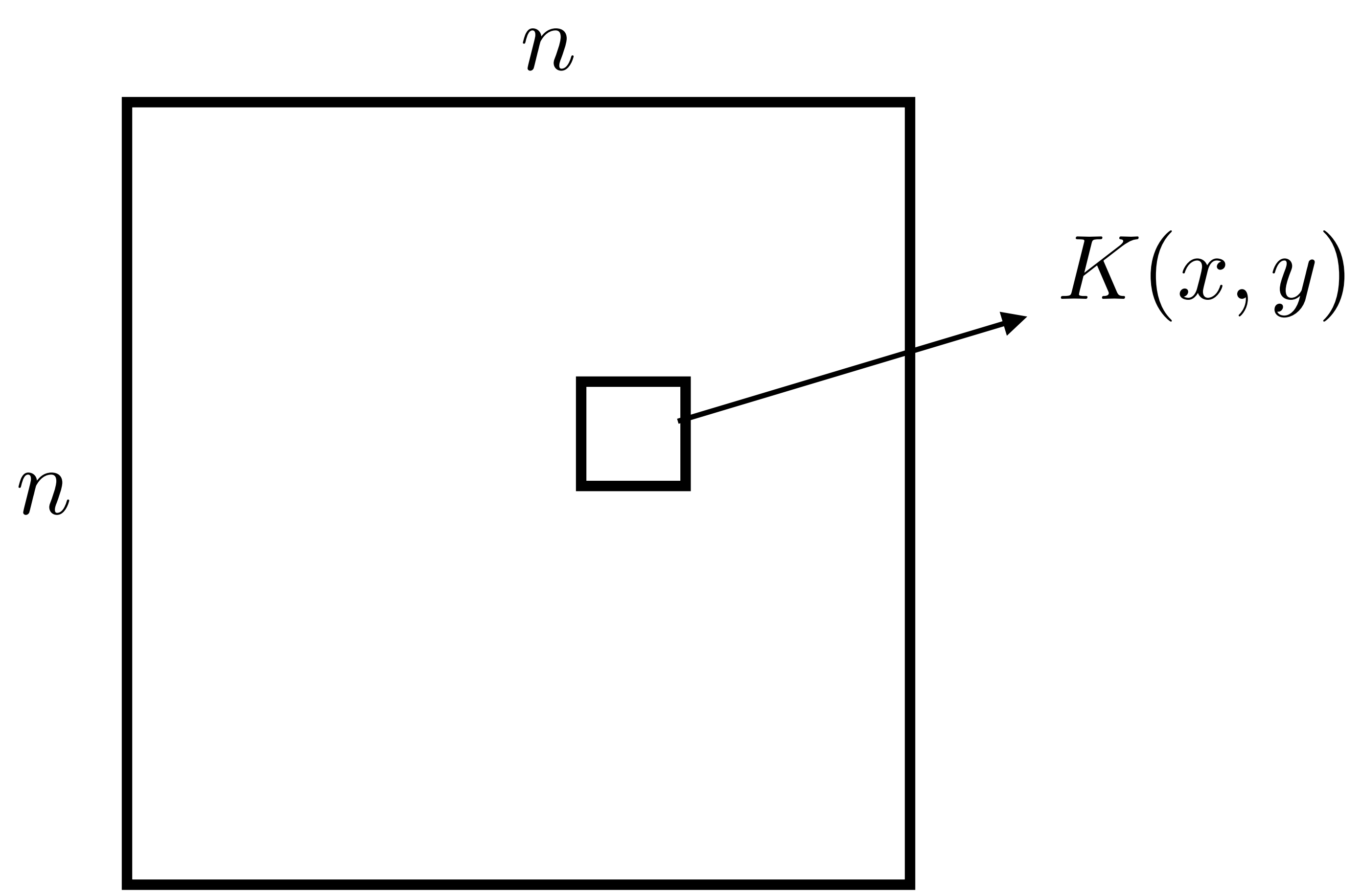
t-SNE: Visualization technique via “matching” geometric graphs.



t-SNE

https://projector.tensorflow.org/

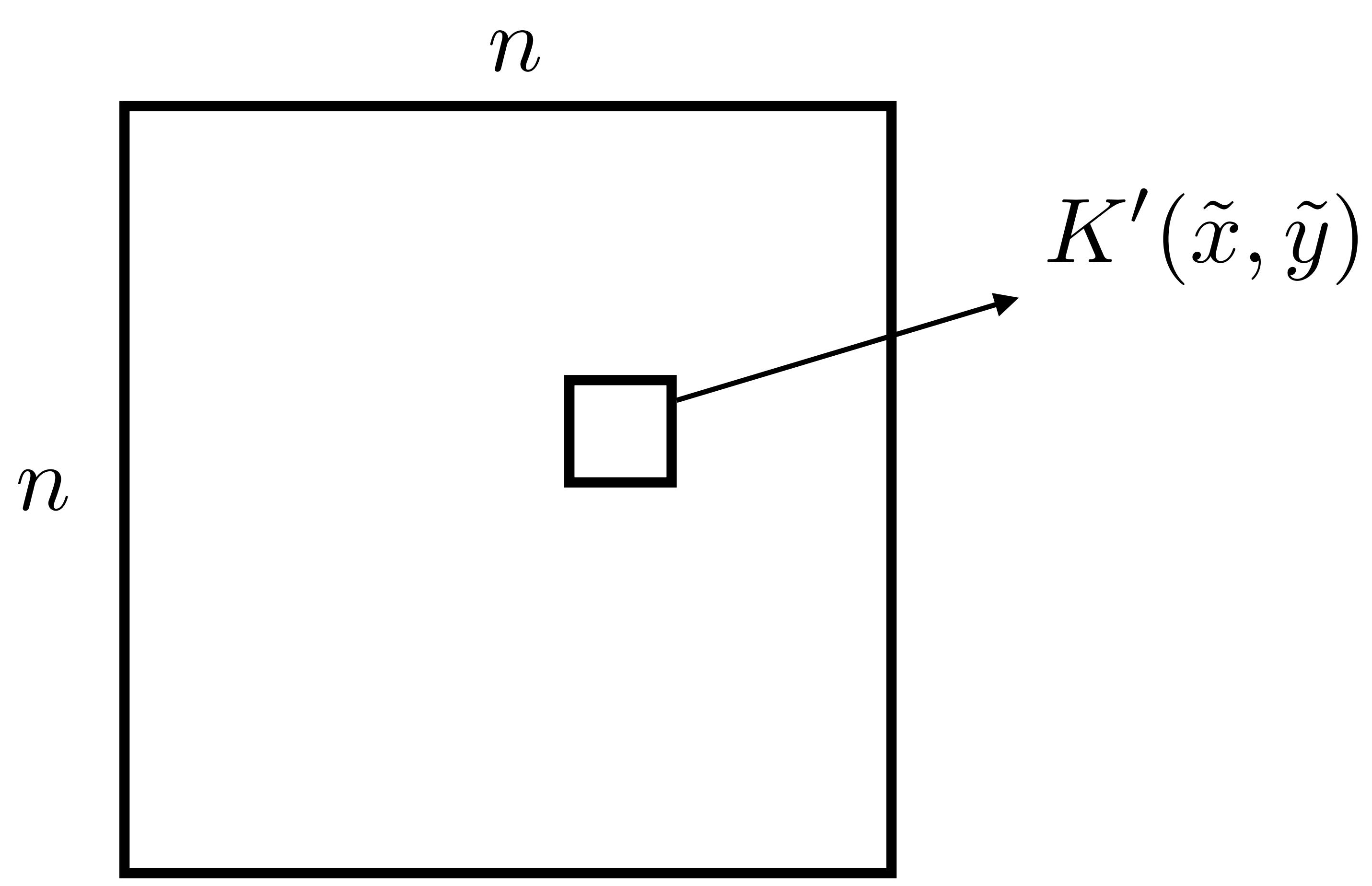




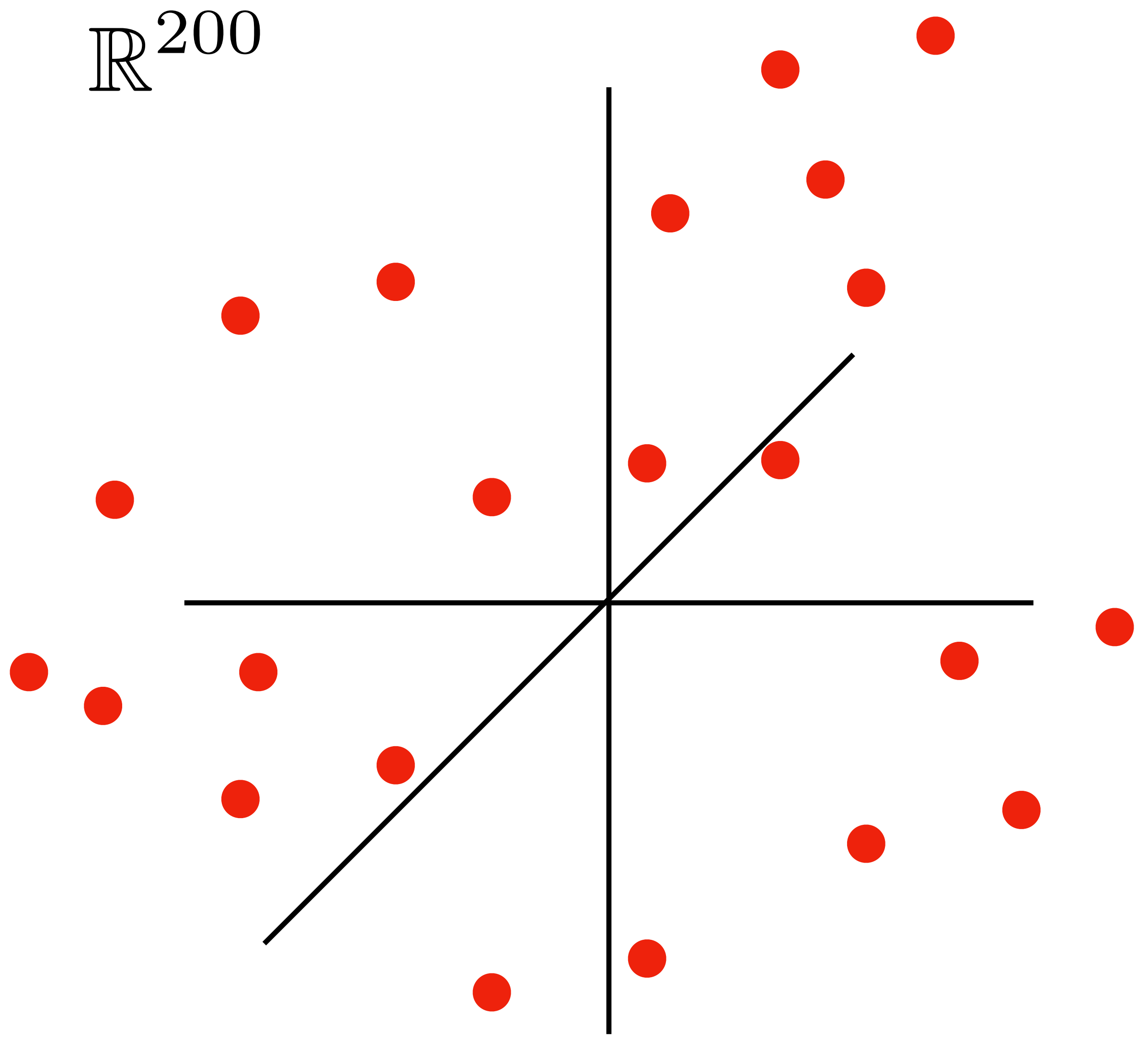
Kernel Function

$$K(x, y) \in [0, 1]$$

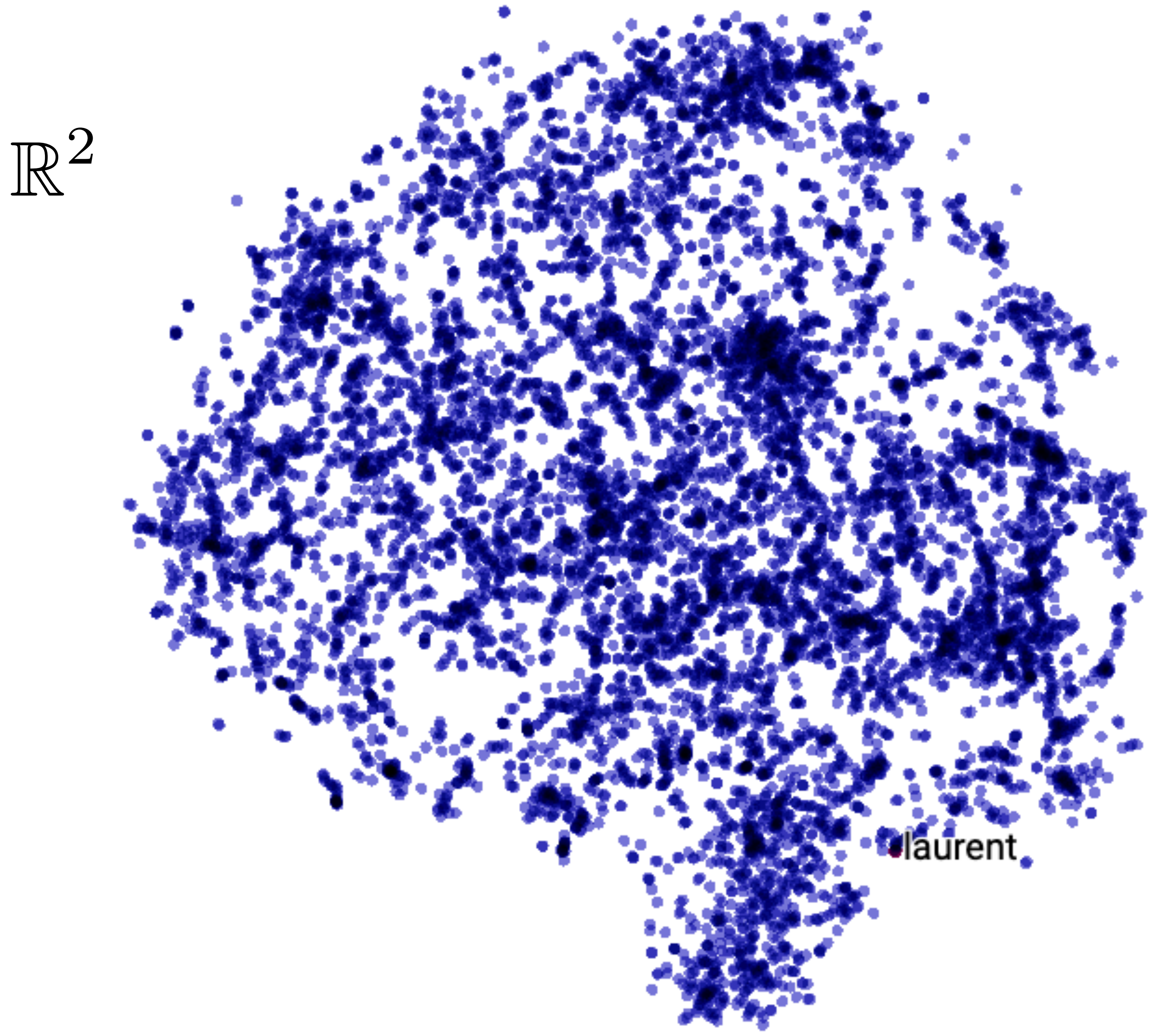
$$K(x, y) \to 0 \text{ as } \|x - y\|_2 \to \infty$$



t-SNE: Visualization technique via “matching” geometric graphs.



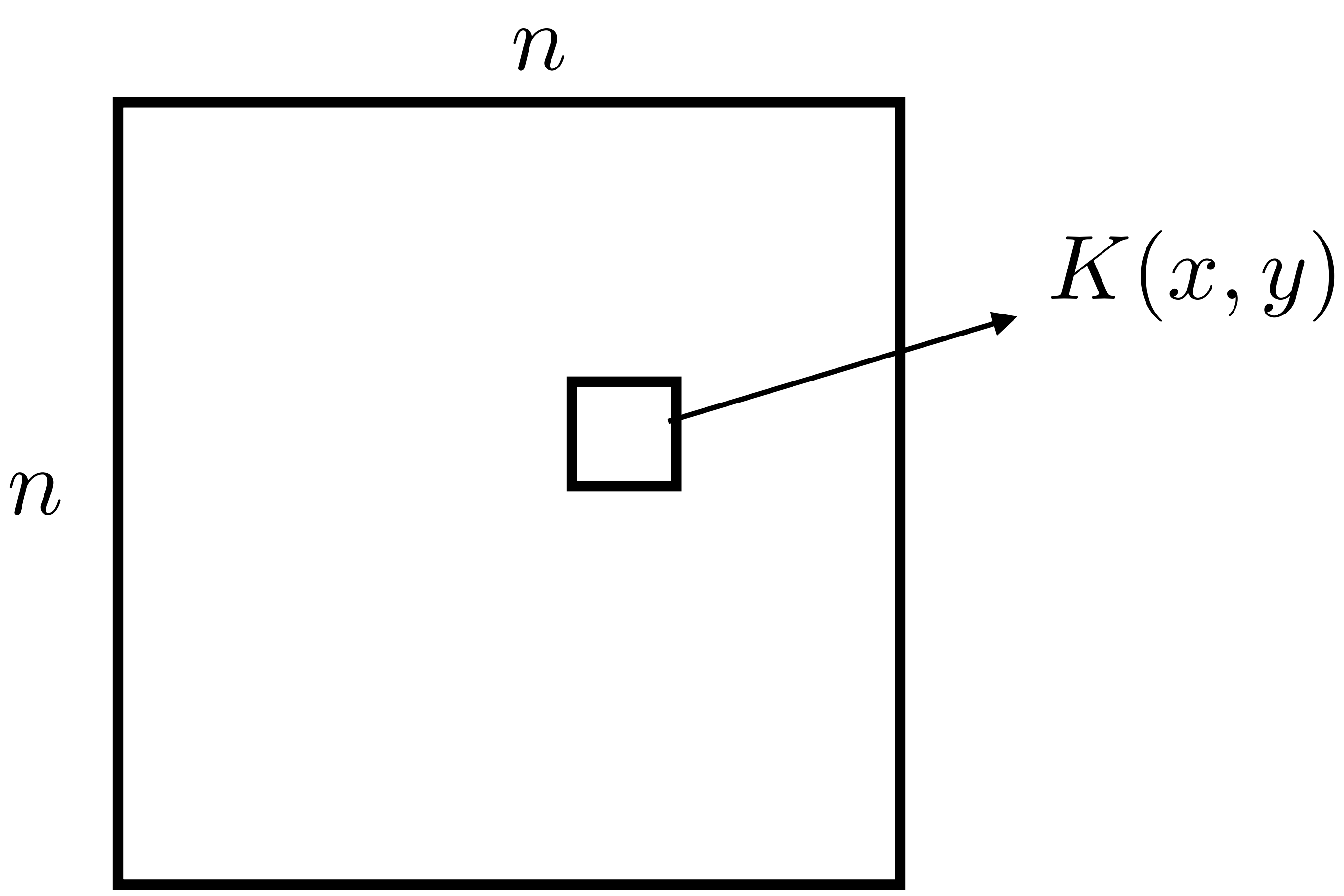
t-SNE



https://projector.tensorflow.org/

Ex: $K(x,y) = \exp\left(-\frac{\|x-y\|_2^2}{2\sigma^2}\right)$

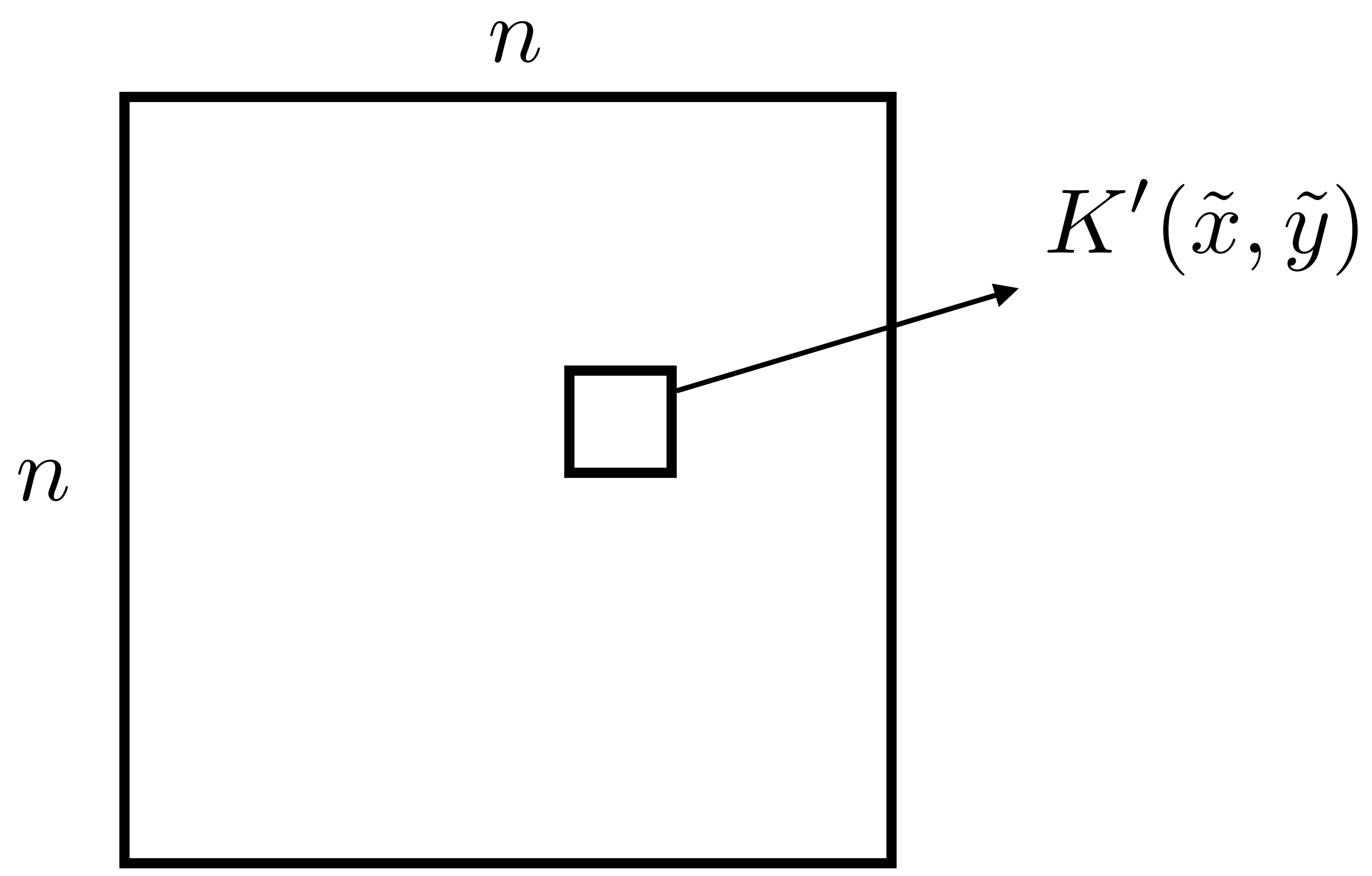
$$K(x,y) = \left(1 + \|x-y\|_2^2\right)^{-1}$$



<u>Kernel Function</u>

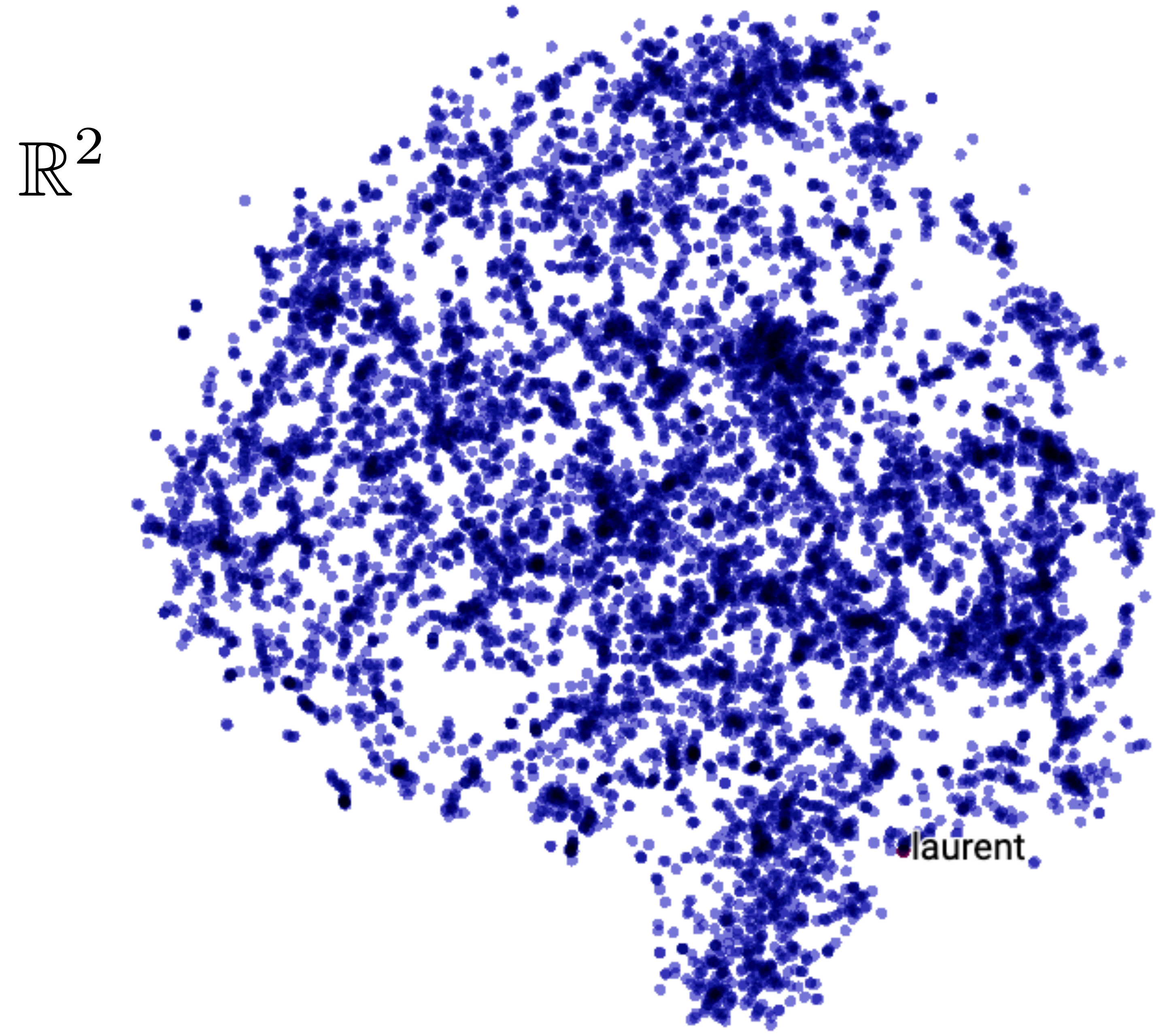
$$K(x,y) \in [0,1]$$

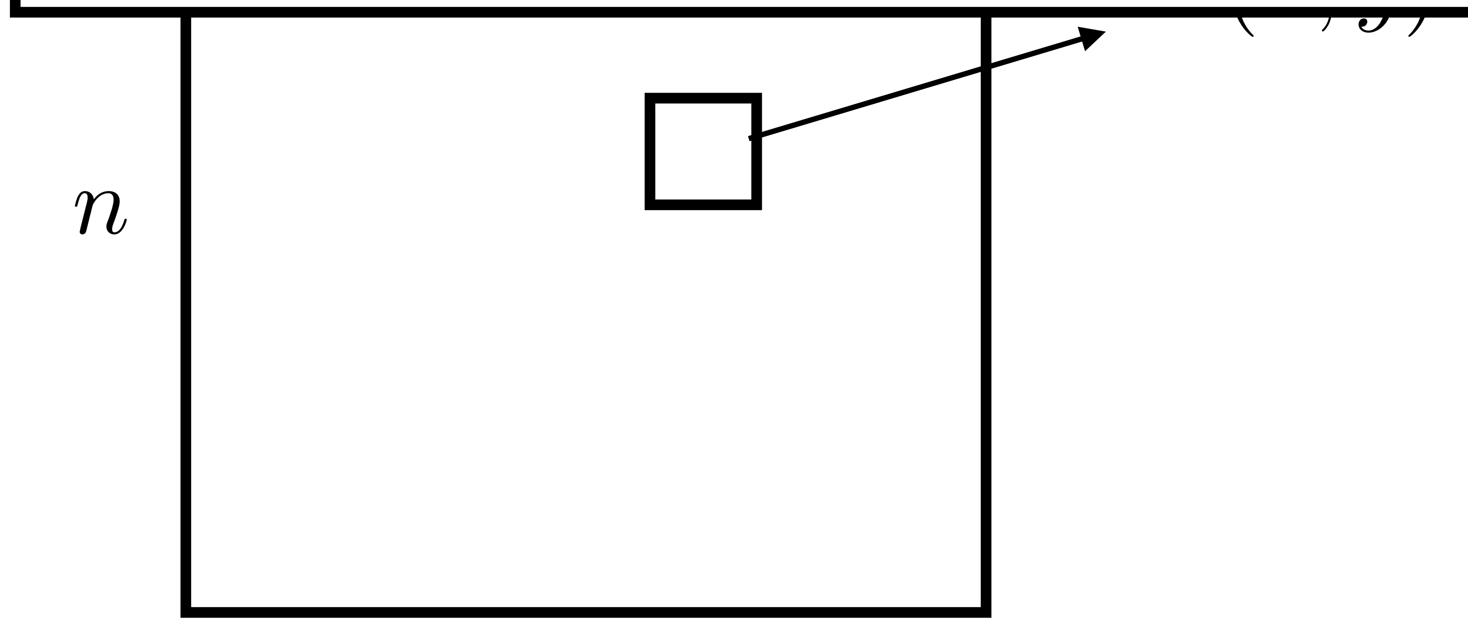
$$K(x,y) \to 0 \text{ as } \|x-y\|_2 \to \infty$$



t-SNE: Visualization technique via "matching" geometric graphs.

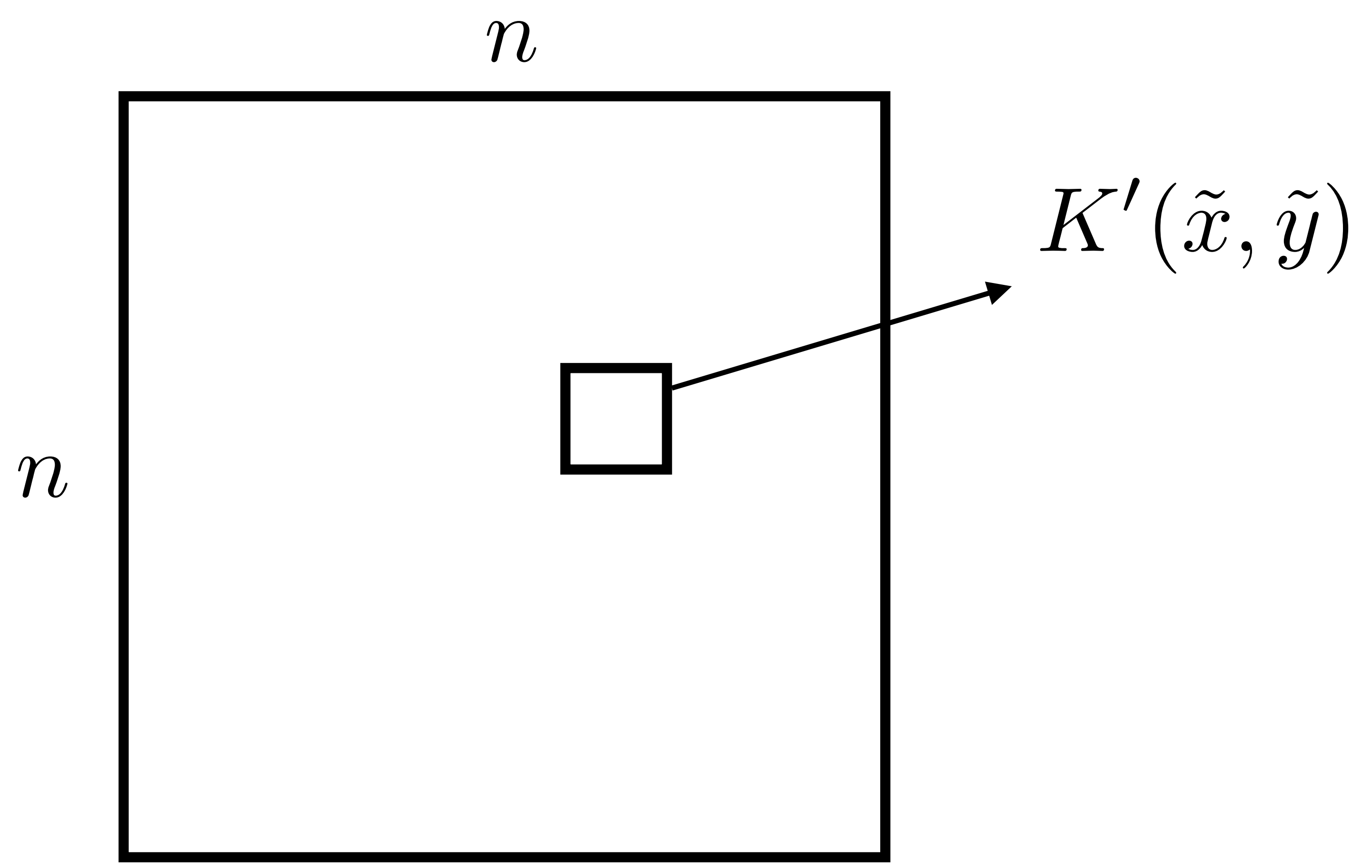
- In low-dimensional space:
  - Computational geometry — kd-trees, ball-trees, vantage point trees, …, etc.
  - Scientific computing — fast multipole methods, n-body simulations, …, etc.



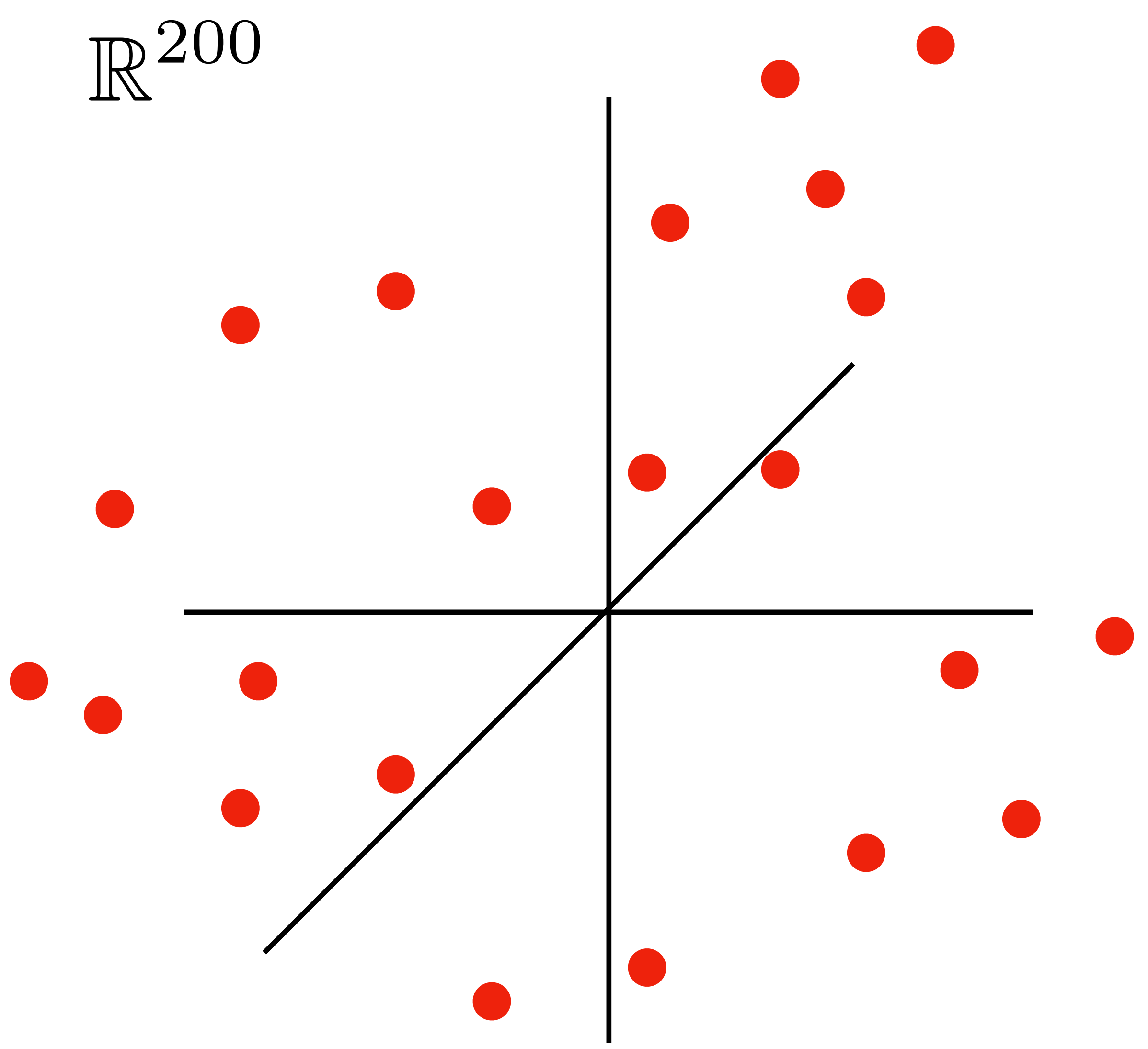


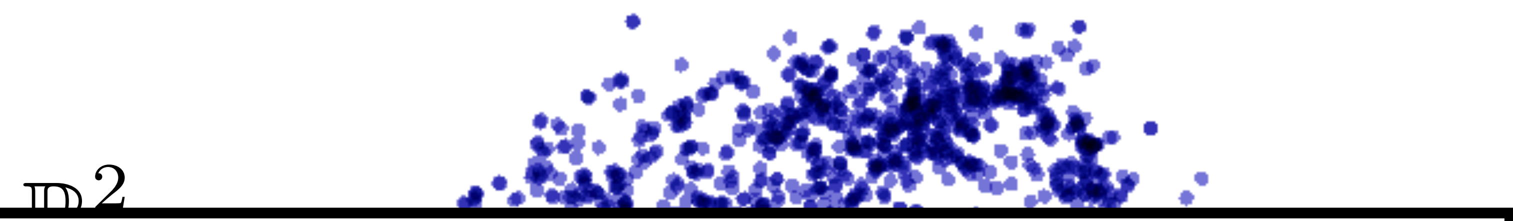
$$K(x, y) \in [0, 1]$$

$$K(x, y) \to 0 \text{ as } \|x - y\|_2 \to \infty$$



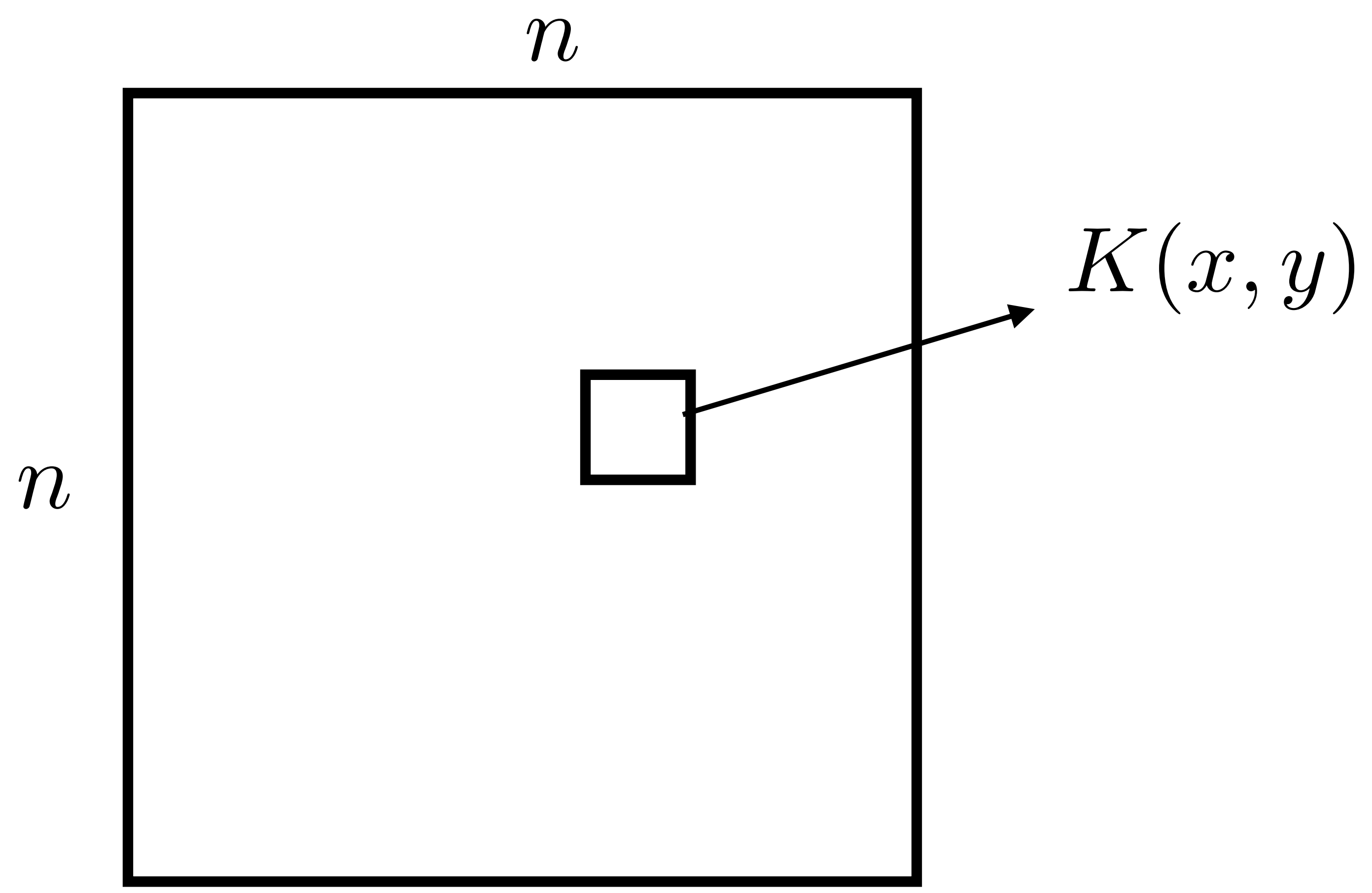
t-SNE: Visualization technique via "matching" geometric graphs.





- In high-dimensional space:

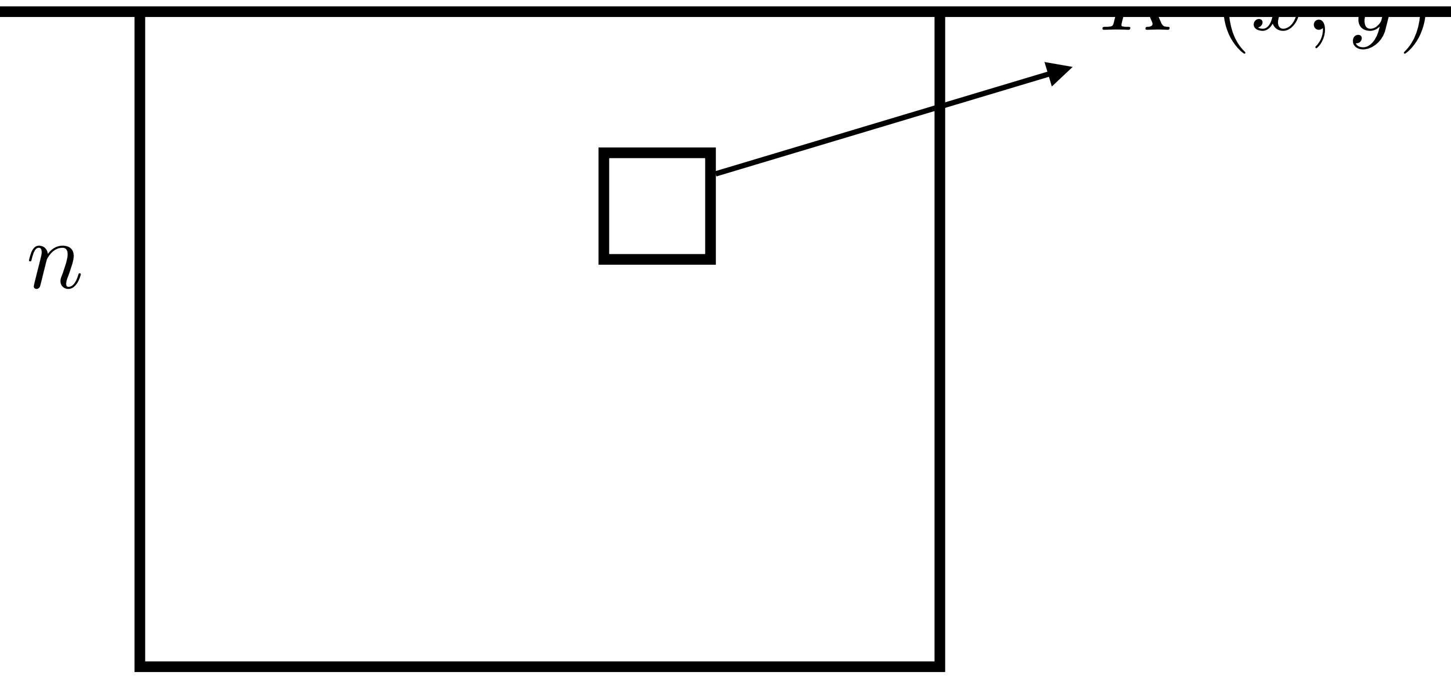
  - Currently, explicitly working with nxn matrix
  - Incorporate tools from TCS:
    - LSH, sketching, random projections, coresets, …, etc.



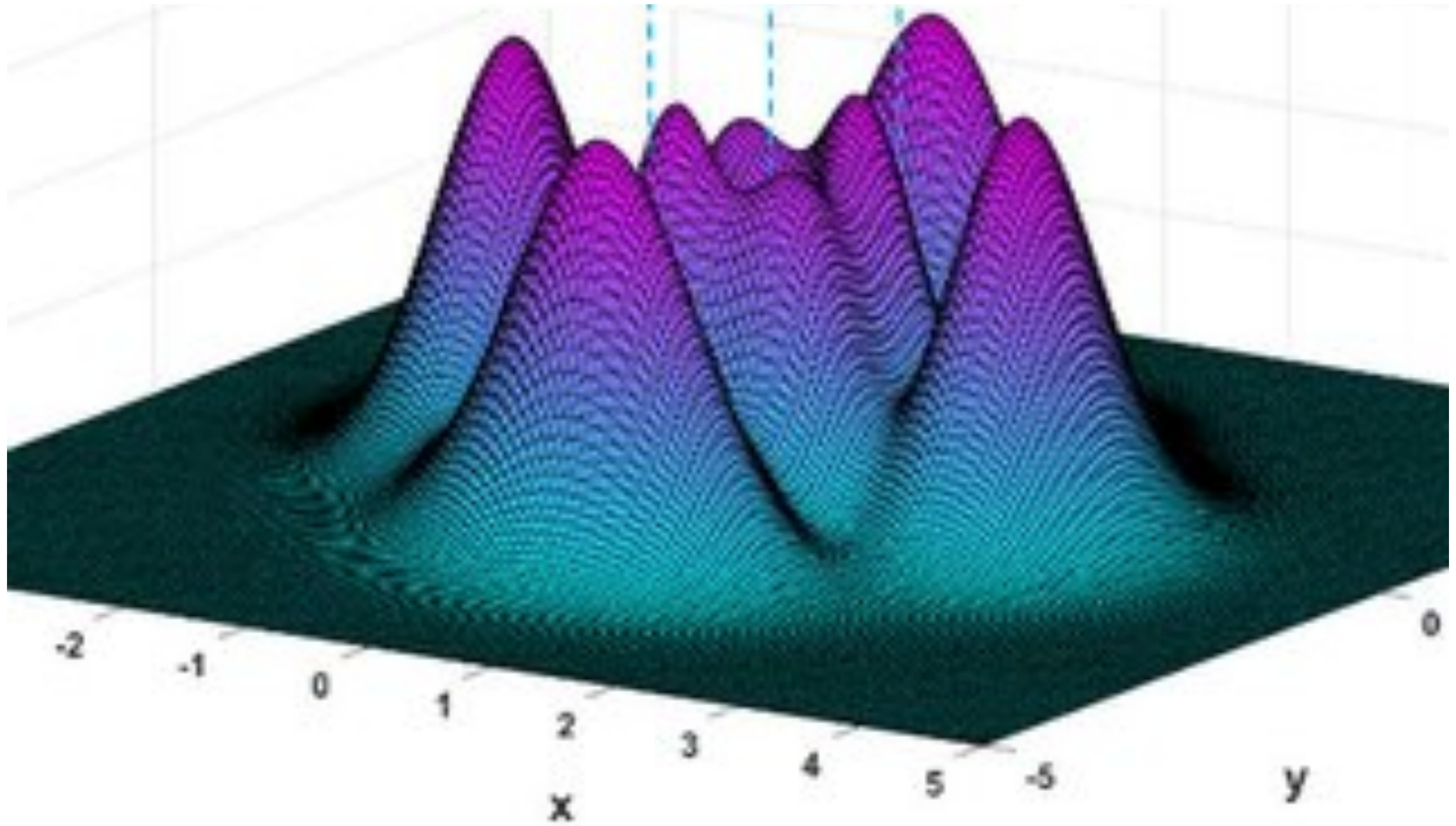
Kernel Function

$$K(x, y) \in [0, 1]$$

$$K(x, y) \to 0 \text{ as } \|x - y\|_2 \to \infty$$



Kernel Density Evaluation: Estimating "Degree" in Geometric Graph



$$K(P,q) = \frac{1}{|P|}\sum_{p \in P} K(p,q) \stackrel{\text{def}}{=} \mu$$

$$K(x,y) \in [0,1]$$

$$K(x,y) \to 0 \text{ as } \|x-y\|_2 \to \infty$$

Kernel Density Estimation (KDE)

Preprocess: a dataset $P \subset \mathbb{R}^d$

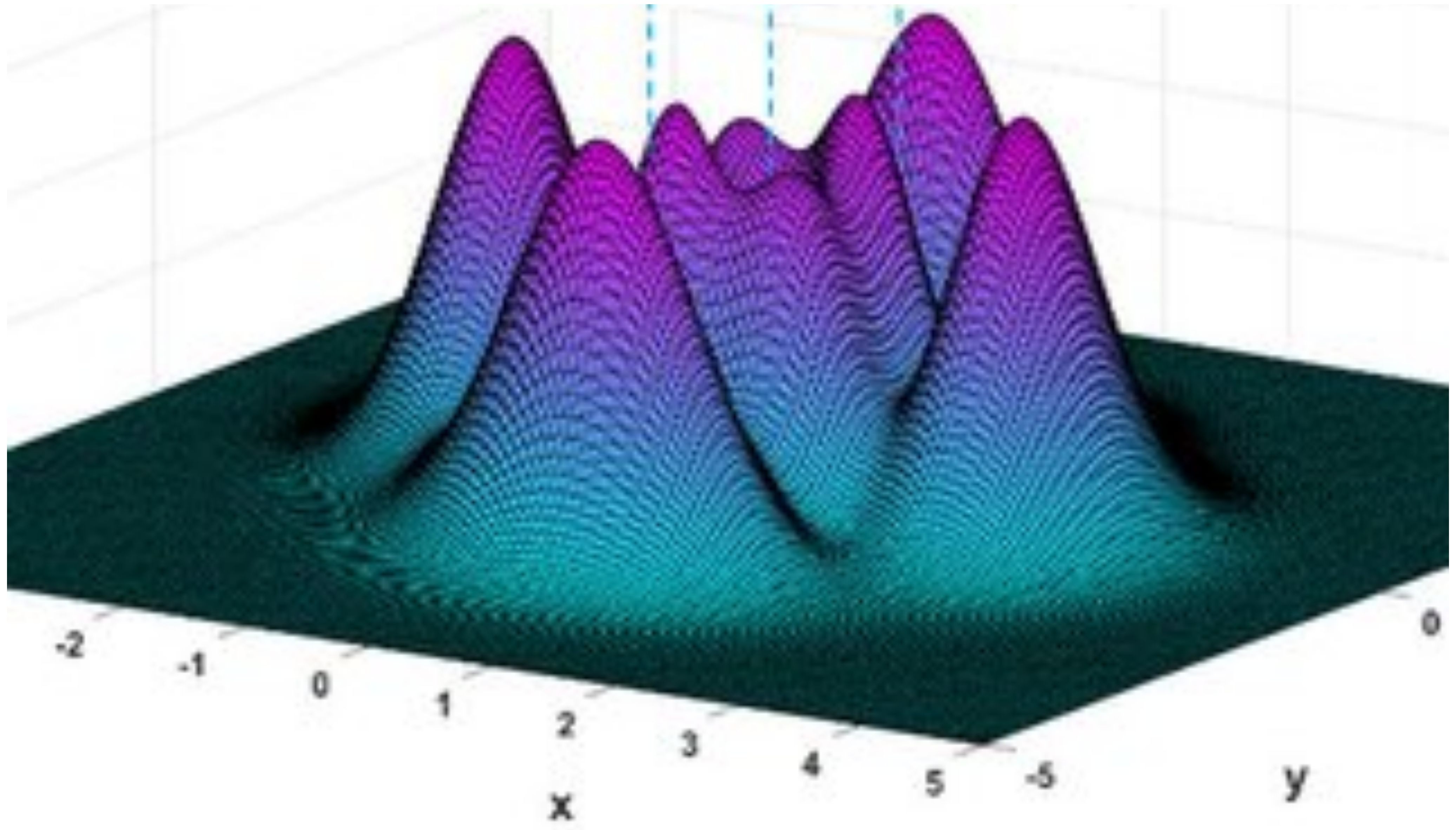
Query: a point $q \in \mathbb{R}^d$

Compute $K(P,q)$ (approximately)

Time-Space Tradeoffs
for $(1+\epsilon)$-approx w.h.p

[Charikar-Siminelakis '17,
Backurs-Charikar-Indyk-Siminelakis '18,
Alman, Chu, Schild, Song '20, …, etc]

Kernel Density Evaluation: Estimating “Degree” in Geometric Graph



$$K(P,q) = \frac{1}{|P|}\sum_{p\in P} K(p,q) \stackrel{\text{def}}{=} \mu$$

$$K(x,y) \in [0,1]$$

$$K(x,y) \to 0 \text{ as } \|x-y\|_2 \to \infty$$

Kernel Density Estimation (KDE)

Preprocess: a dataset $P \subset \mathbb{R}^d$

Query: a point $q \in \mathbb{R}^d$

Compute $K(P,q)$ (approximately)

Time-Space Tradeoffs
for $(1+\epsilon)$-approx w.h.p

Trivial: $O(nd)$ query time.

[Charikar-Siminelakis ’17,
Backurs-Charikar-Indyk-Siminelakis ’18,
Alman, Chu, Schild, Song ’20, …, etc]

Two approaches for "beating $O(nd)$ query time"

$$K(P,q) = \frac{1}{|P|}\sum_{p\in P} K(p,q) \stackrel{\text{def}}{=} \mu$$

$$K(x,y) \in [0,1]$$

Uniform Random Sampling:

$$O\left(\frac{1}{\mu\epsilon^2}\right)$$

Two approaches for "beating $O(nd)$ query time"

$$K(P,q) = \frac{1}{|P|}\sum_{p\in P} K(p,q) \stackrel{\text{def}}{=} \mu$$

$$K(x,y) \in [0,1]$$

Uniform Random Sampling:

$$O\left(\frac{1}{\mu\epsilon^2}\right)$$

Our work:

$$\frac{\text{polylog}(1/\mu)}{\epsilon}$$

(smooth)

LSH Sampling
[Charikar, Siminelakis '17]

$O\left(\frac{1}{\sqrt{\mu}\epsilon^2}\right)$, $\frac{\text{polylog}(1/\mu)}{\epsilon^2}$ (smooth)

[Backurs, Charikar, Indyk, Siminelakis '18]

Coresets via Discrepancy
[Phillips-Tai '20]

$$O\left(\frac{1}{\mu\epsilon}\right)$$

Random/LSH Sampling

Store $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_t \sim P \subset \mathbb{R}^d$

Query: $\frac{1}{t}\sum_{i=1}^{t} K(\boldsymbol{p}_i, q)$

Random

$$\mathbf{E}\left[\frac{1}{t}\sum_{i=1}^{t} K(\boldsymbol{p}_i, q)\right] = \mu$$

Chebyshev:

$$\mathbf{Pr}\left[\begin{array}{c}\text{accurate}\\ \text{estimate}\end{array}\right] \leq \frac{1}{(\epsilon\mu)^2} \cdot \boxed{\mathbf{Var}}$$

LSH

Random/LSH Sampling

Store $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_t \sim P \subset \mathbb{R}^d$

Query: $\frac{1}{t}\sum_{i=1}^{t} K(\boldsymbol{p}_i, q)$

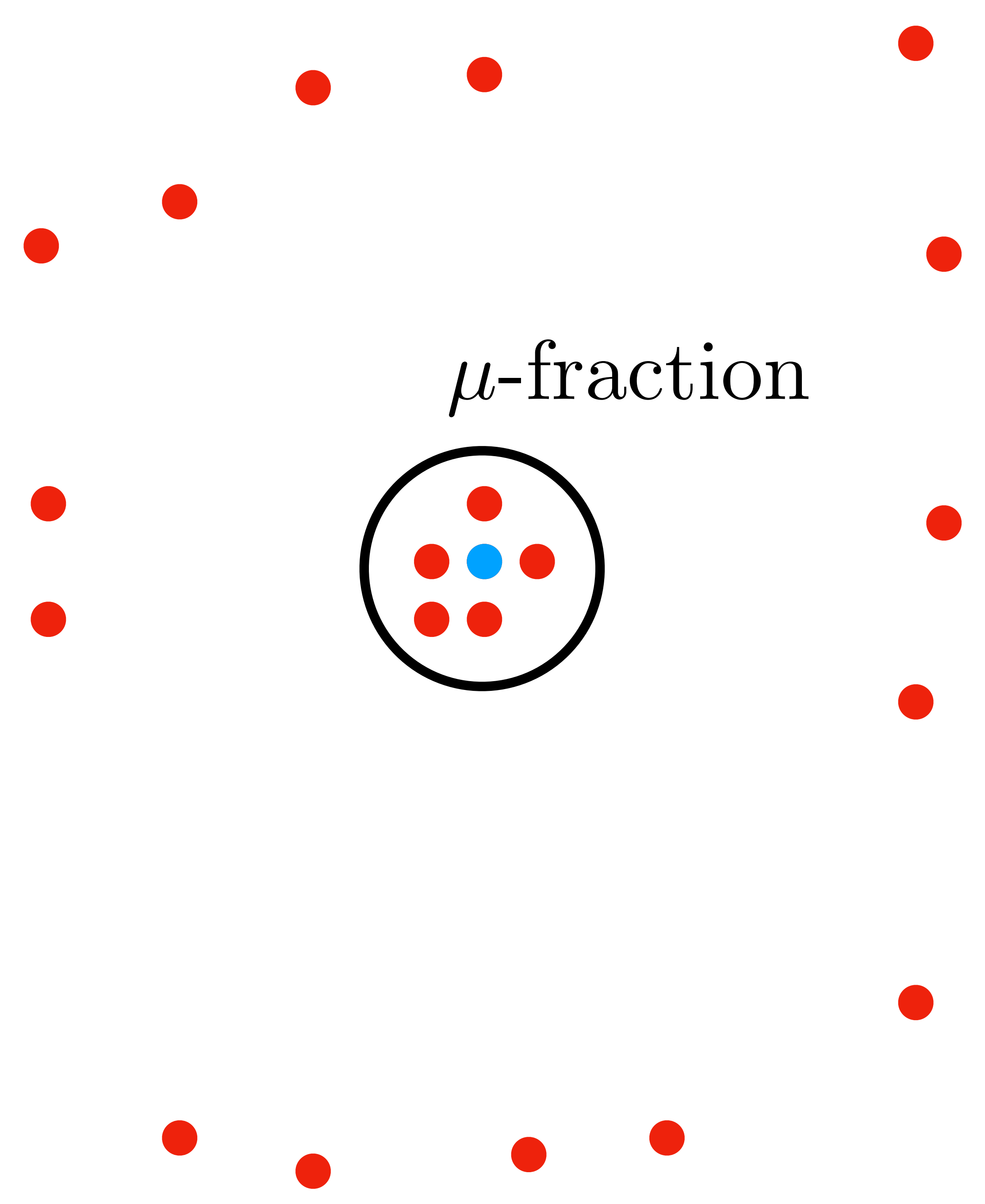
Random

$$\mathbf{E}\left[\frac{1}{t}\sum_{i=1}^{t} K(\boldsymbol{p}_i, q)\right] = \mu$$

Chebyshev:

$$\mathbf{Pr}\left[\begin{array}{c}\text{accurate}\\ \text{estimate}\end{array}\right] \leq \frac{1}{(\epsilon\mu)^2} \cdot \boxed{\mathbf{Var}}$$

LSH



Random/LSH Sampling

Store $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_t \sim P \subset \mathbb{R}^d$

Query: $\frac{1}{t} \sum_{i=1}^{t} K(\boldsymbol{p}_i, q)$

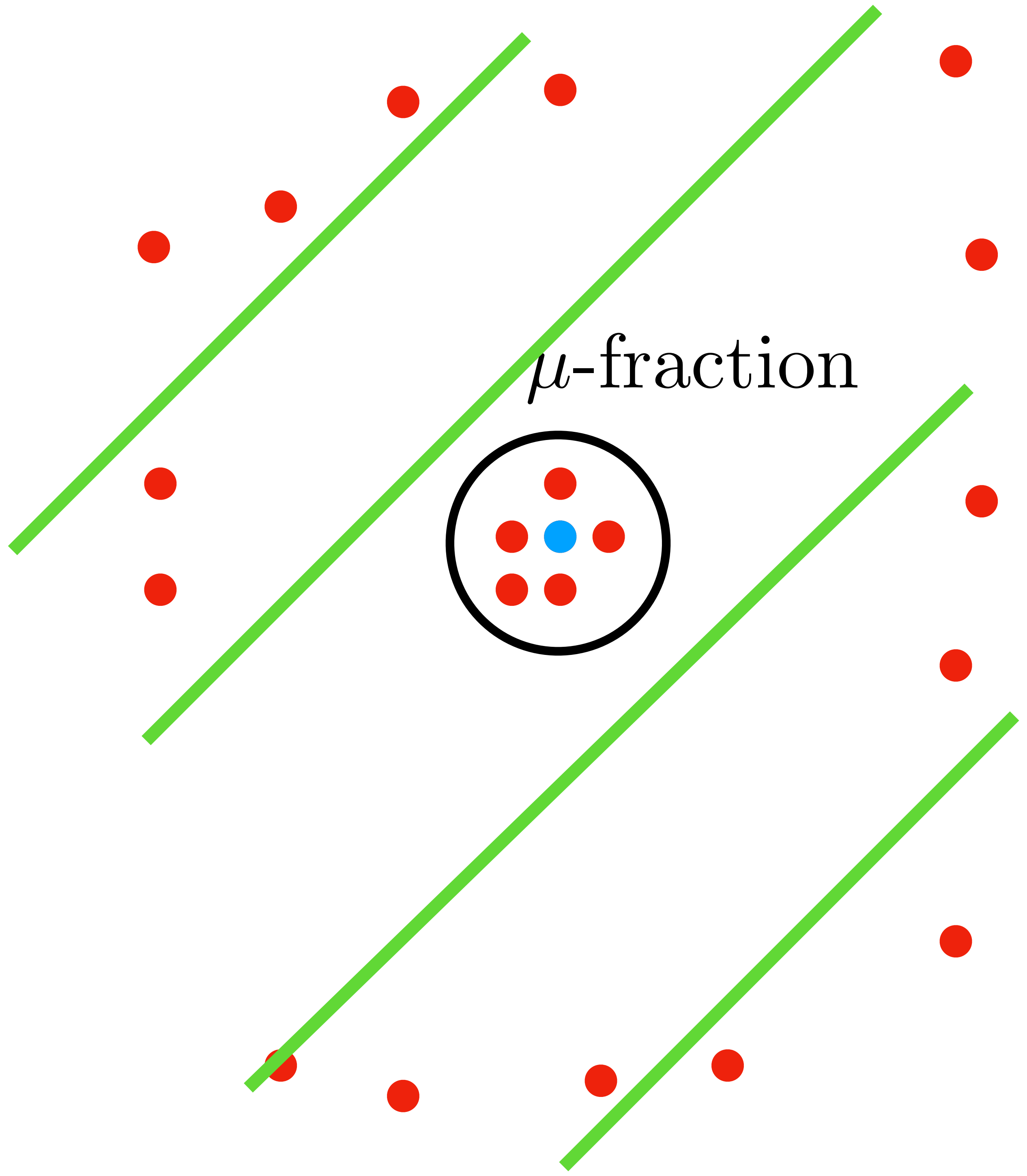
Random

$$\mathbf{E}\left[\frac{1}{t}\sum_{i=1}^{t} K(\boldsymbol{p}_i, q)\right] = \mu$$

Chebyshev:

$$\mathbf{Pr}\left[\begin{array}{c}\text{accurate}\\ \text{estimate}\end{array}\right] \leq \frac{1}{(\epsilon\mu)^2} \cdot \boxed{\mathbf{Var}}$$

LSH



Random/LSH Sampling

Store $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_t \sim P \subset \mathbb{R}^d$

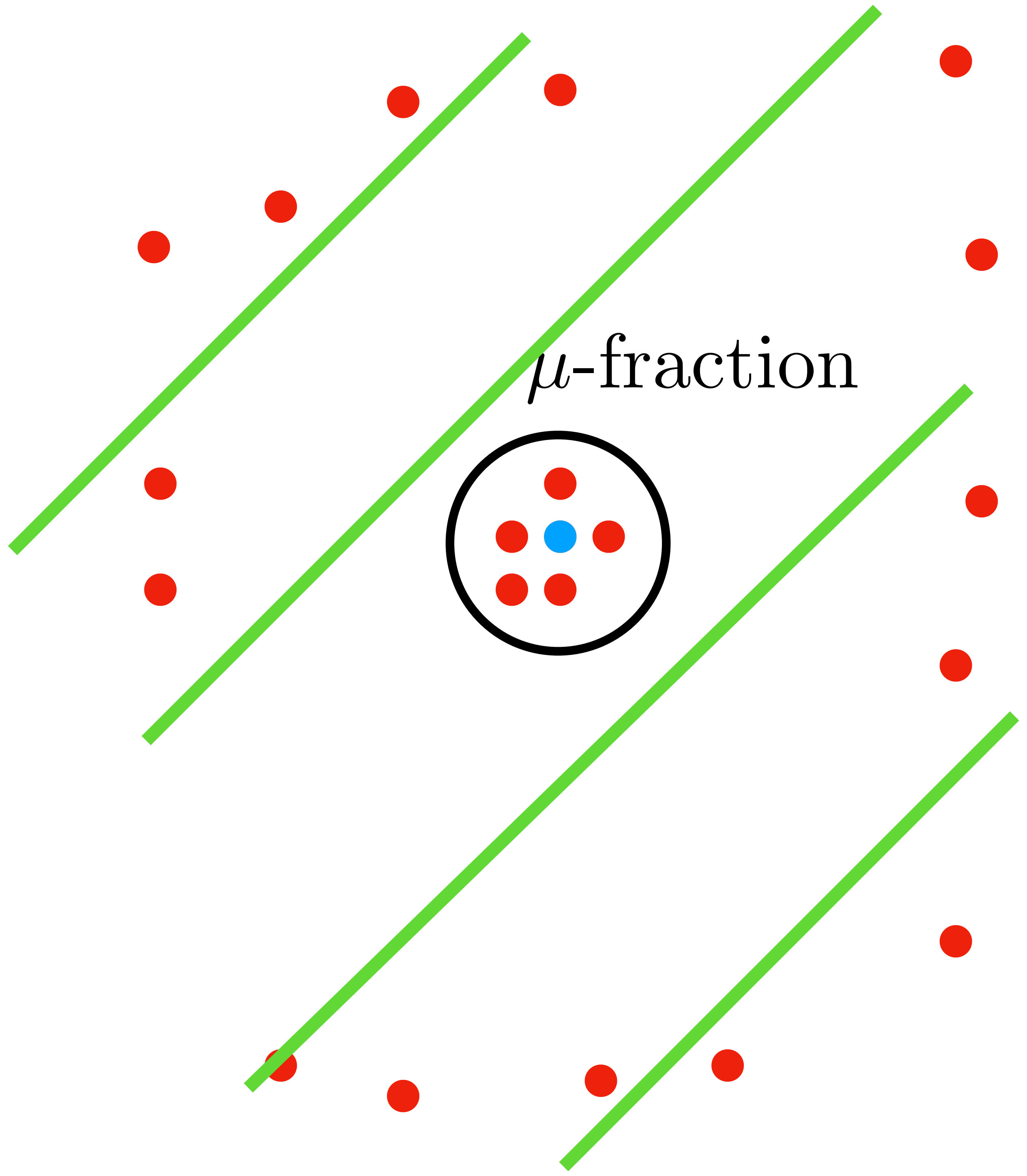
Query: $\frac{1}{t}\sum_{i=1}^{t} K(\boldsymbol{p}_i, q)$

Random

$$\mathbf{E}\left[\frac{1}{t}\sum_{i=1}^{t} K(\boldsymbol{p}_i, q)\right] = \mu$$

Chebyshev:

$$\mathbf{Pr}\left[\begin{matrix}\text{accurate}\\ \text{estimate}\end{matrix}\right] \le \frac{1}{(\epsilon\mu)^2} \cdot \boxed{\mathbf{Var}}$$



LSH

[Charikar, Siminelakis '17]

$$O\left(\frac{1}{\sqrt{\mu}\epsilon^2}\right)$$

[Backurs, Charikar Indyk, Siminelakis '18] (smooth)

$$\frac{\text{polylog}(1/\mu)}{\epsilon^2}$$

<u>Coresets via Discrepancy</u> (PSD)

[Phillips-Tai '20]

Find "important" $p_1, \ldots, p_t \in P$

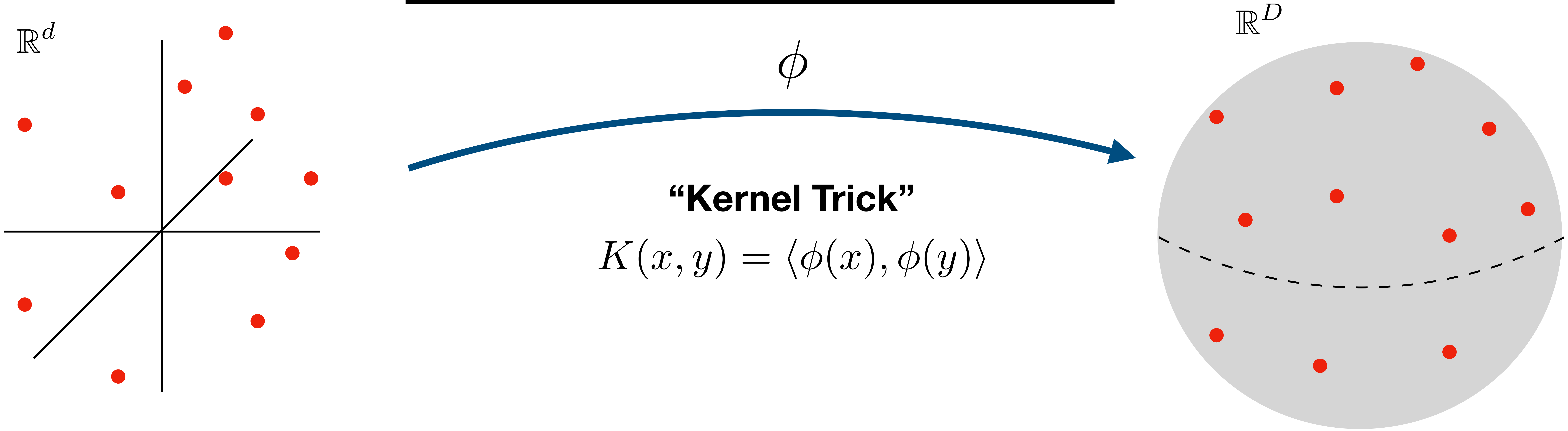
Query: $\frac{1}{t}\sum_{i=1}^{t} K(p_i, q).$

Coresets via Discrepancy (PSD)

[Phillips-Tai '20]

Find "important" $p_1, \ldots, p_t \in P$

Query: $\frac{1}{t} \sum_{i=1}^{t} K(p_i, q)$.

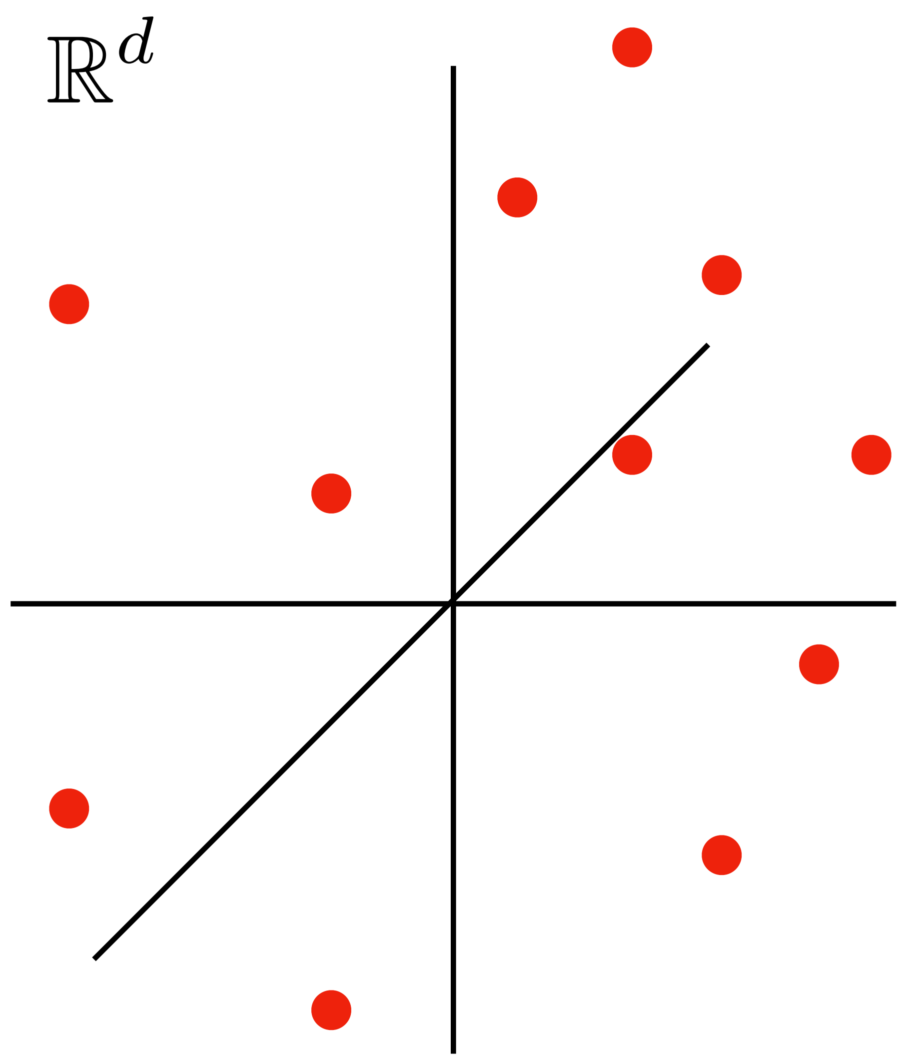


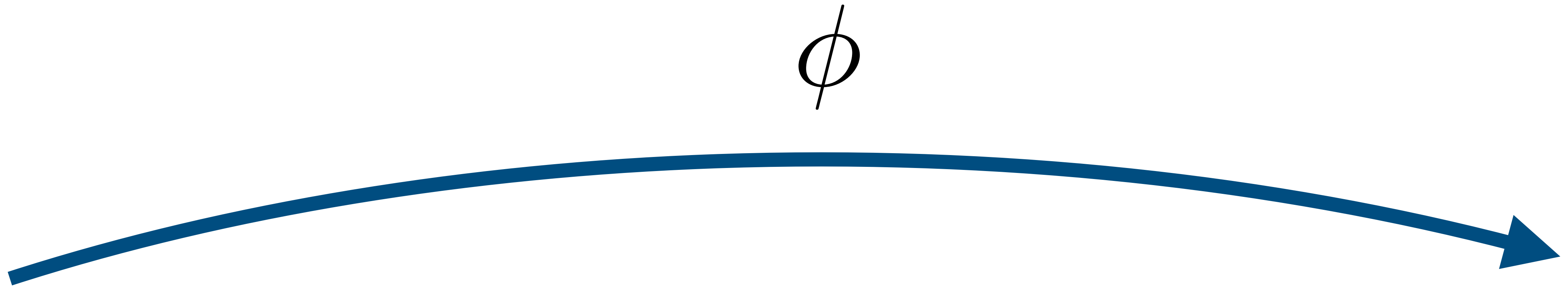
Coresets via Discrepancy (PSD)

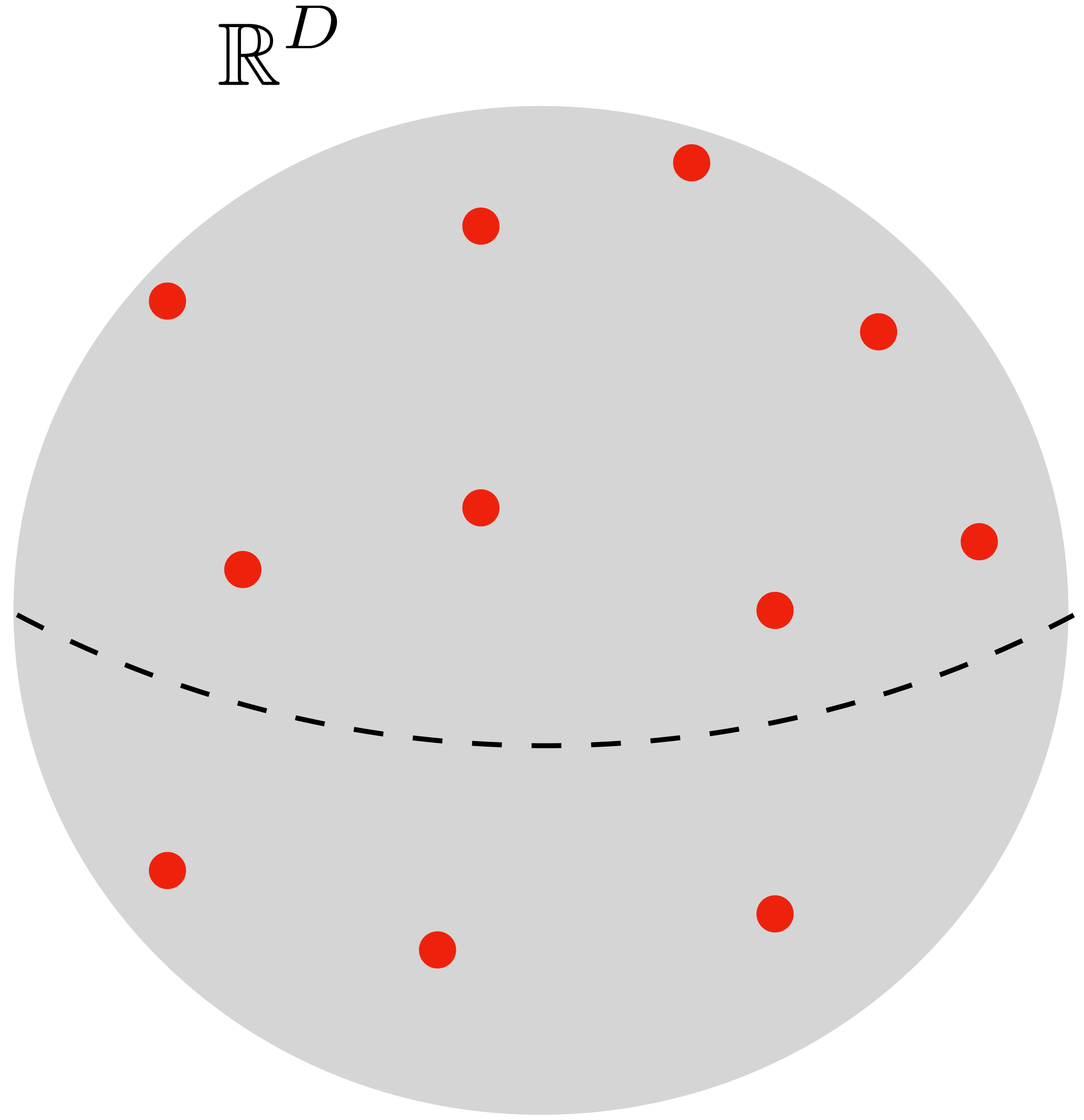
[Phillips-Tai ’20]

Find “important” $p_1, \ldots, p_t \in P$

Query: $\frac{1}{t} \sum_{i=1}^{t} K(p_i, q)$.







“Kernel Trick”

$$K(x, y) = \langle \phi(x), \phi(y) \rangle$$

“Halving Technique”

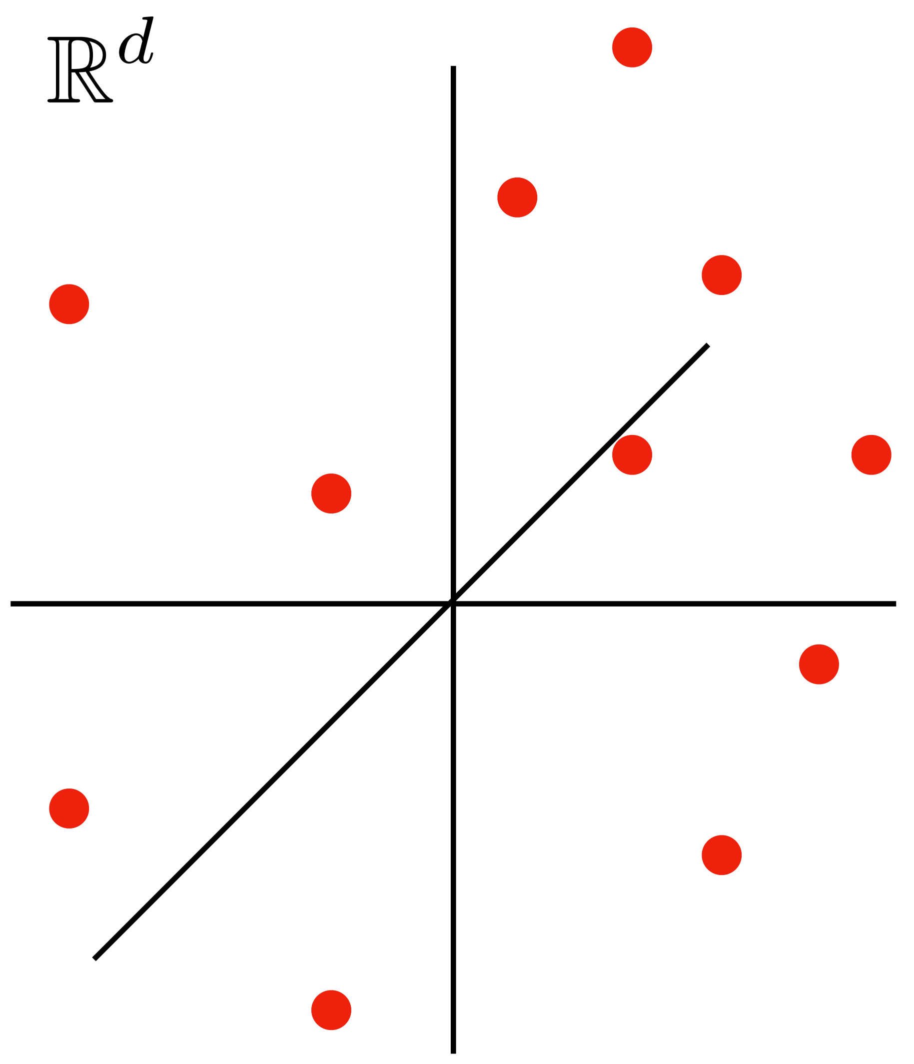
How to decrease your dataset by a factor of 2?

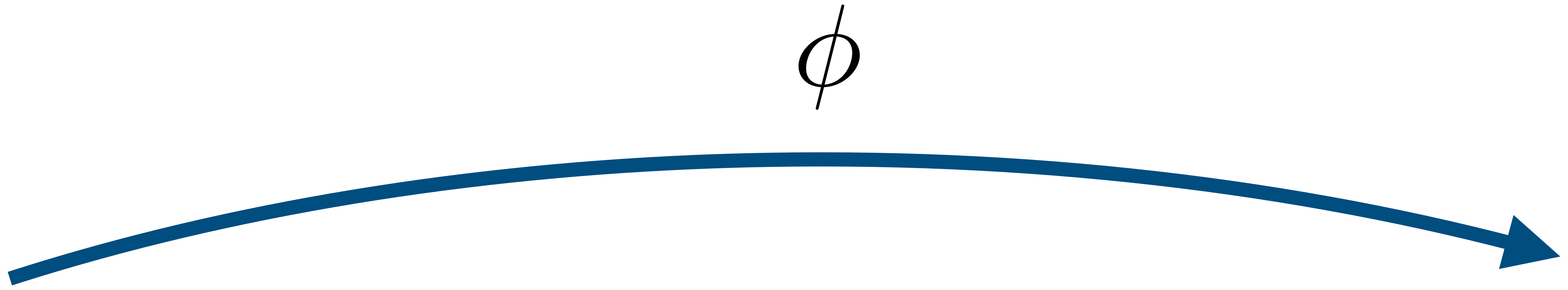
Coresets via Discrepancy (PSD)

[Phillips-Tai '20]

Find "important" $p_1, \dots, p_t \in P$

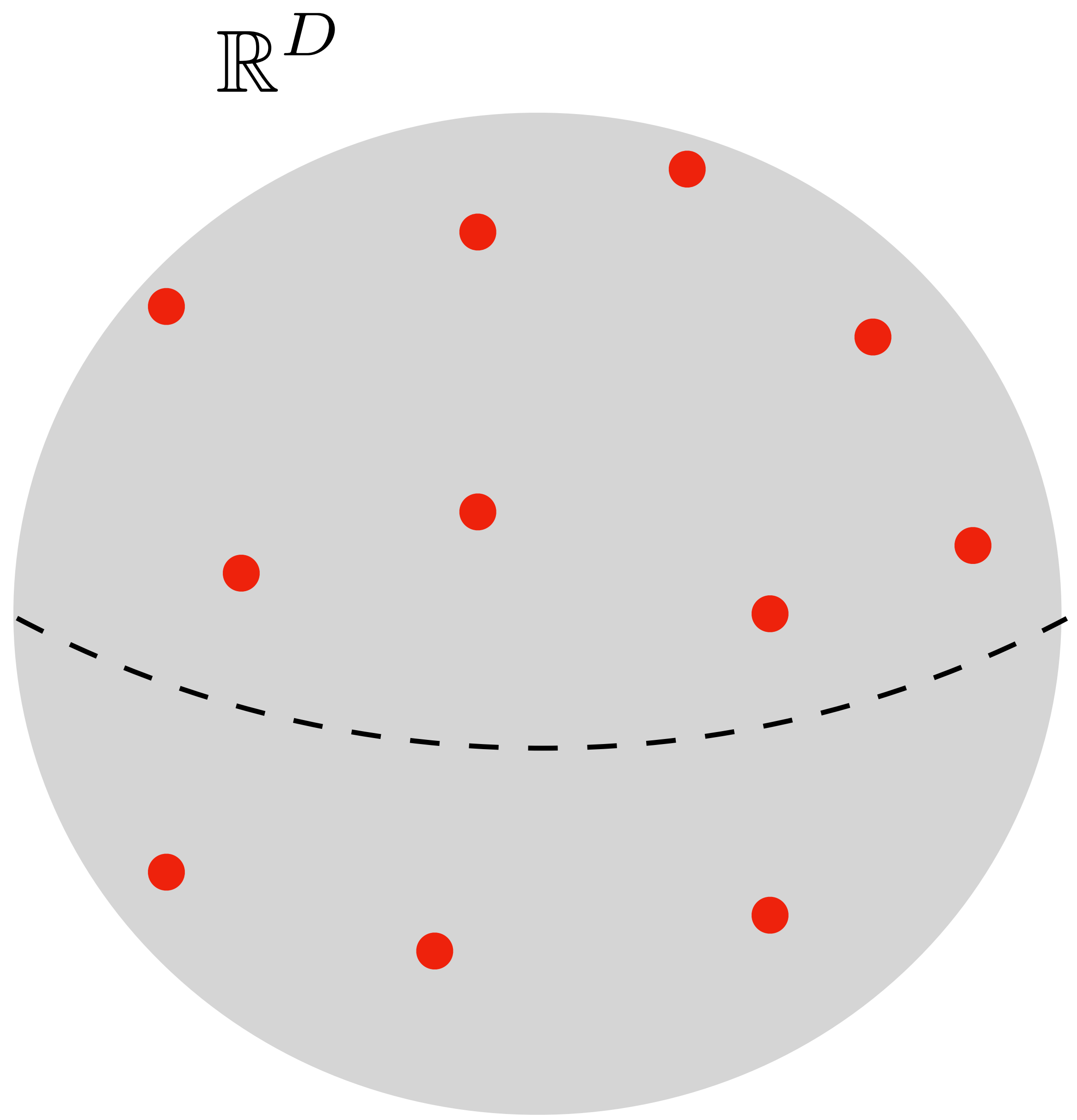
Query: $\frac{1}{t} \sum_{i=1}^{t} K(p_i, q)$.





"Kernel Trick"

$$K(x, y) = \langle \phi(x), \phi(y) \rangle$$



Vector Balancing:

Find a setting of signs $a_1, \dots, a_n \in \{-1, 1\}$ such that for (unknown) query q we minimize

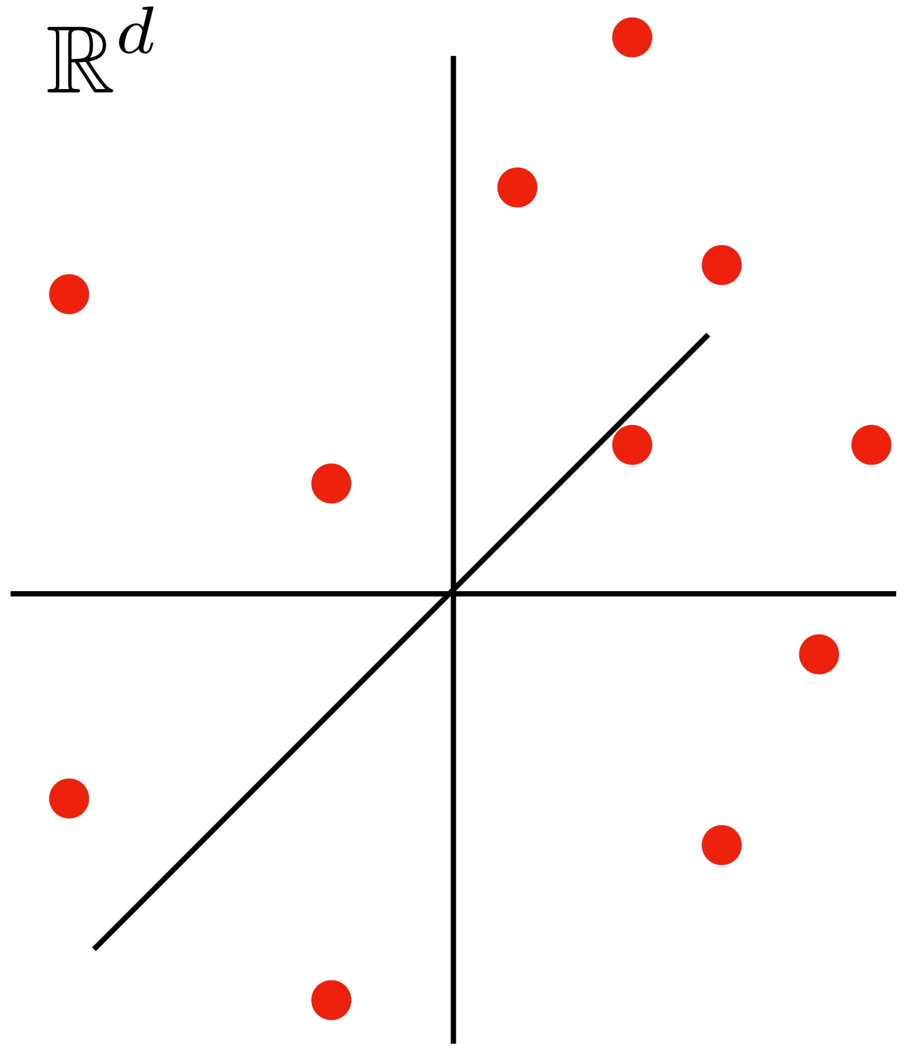
$$\left| \sum_{a_i = 1} \langle \phi(p_i), \phi(q) \rangle - \sum_{a_i = -1} \langle \phi(p_i), \phi(q) \rangle \right|$$

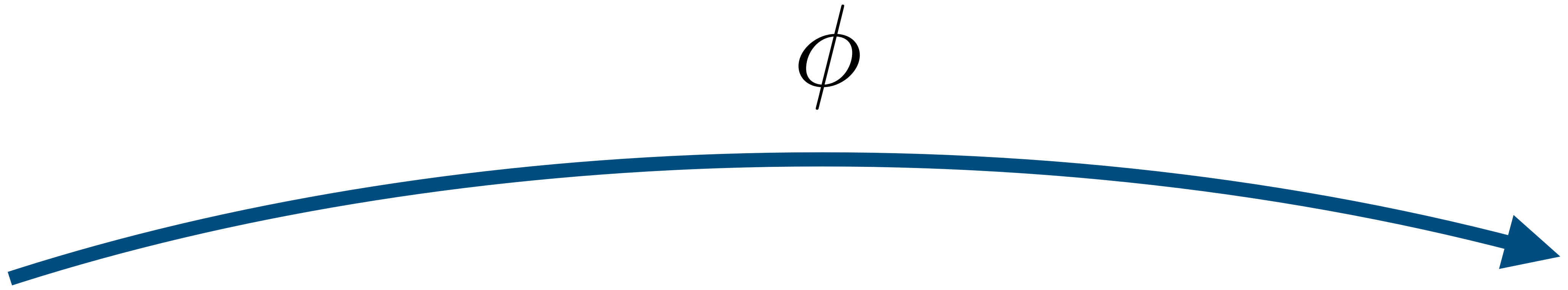
Coresets via Discrepancy (PSD)

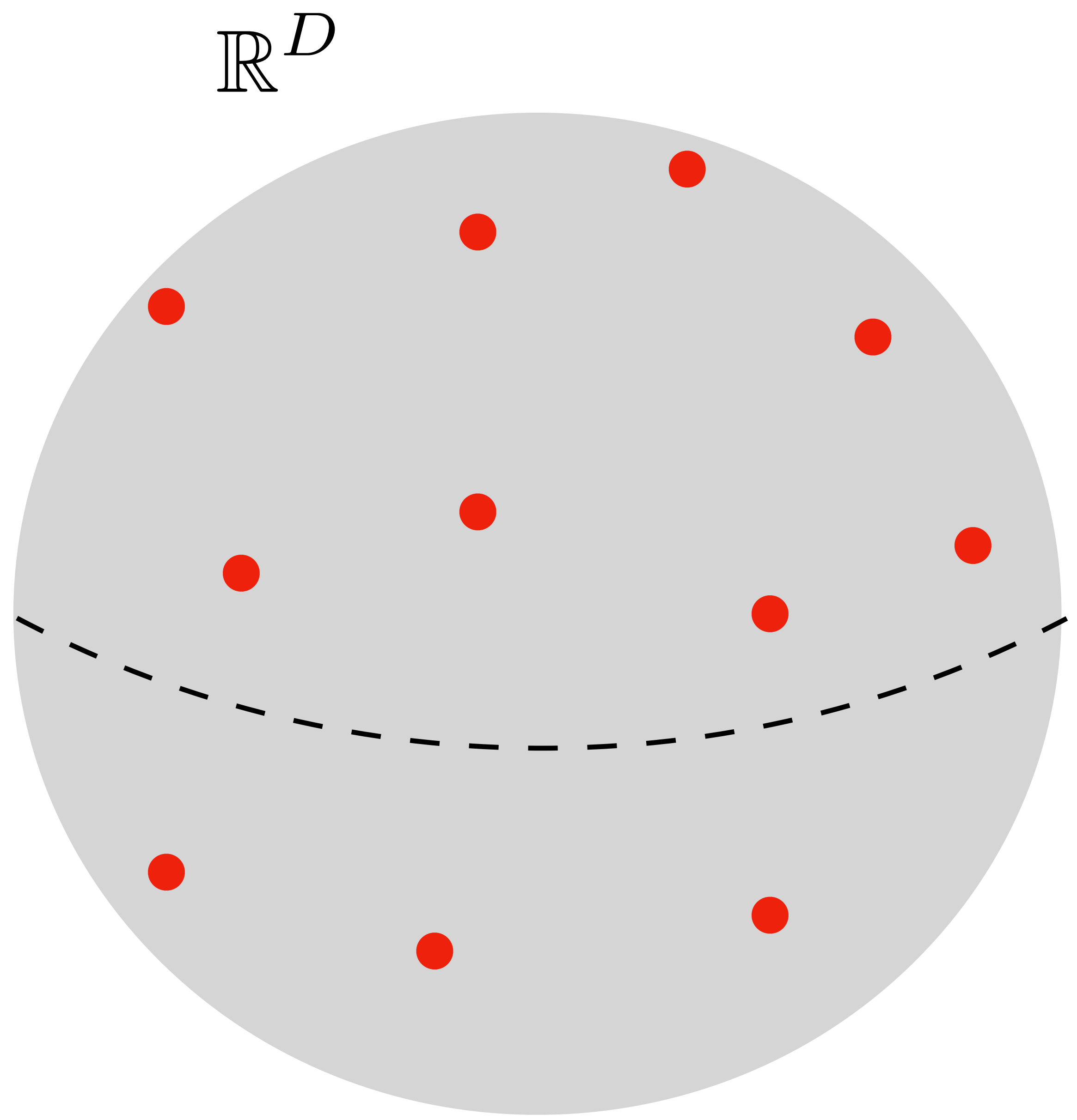
[Phillips-Tai '20]

Find "important" $p_1, \ldots, p_t \in P$

Query: $\frac{1}{t}\sum_{i=1}^{t} K(p_i, q).$







"Kernel Trick"

$$K(x, y) = \langle \phi(x), \phi(y) \rangle$$

Vector Balancing:

Find a setting of signs $a_1, \ldots, a_n \in \{-1, 1\}$ such that for (unknown) query q we minimize

$$\left| \sum_{a_i = 1} \langle \phi(p_i), \phi(q) \rangle - \sum_{a_i = -1} \langle \phi(p_i), \phi(q) \rangle \right| \leq \tilde{O}(1)$$

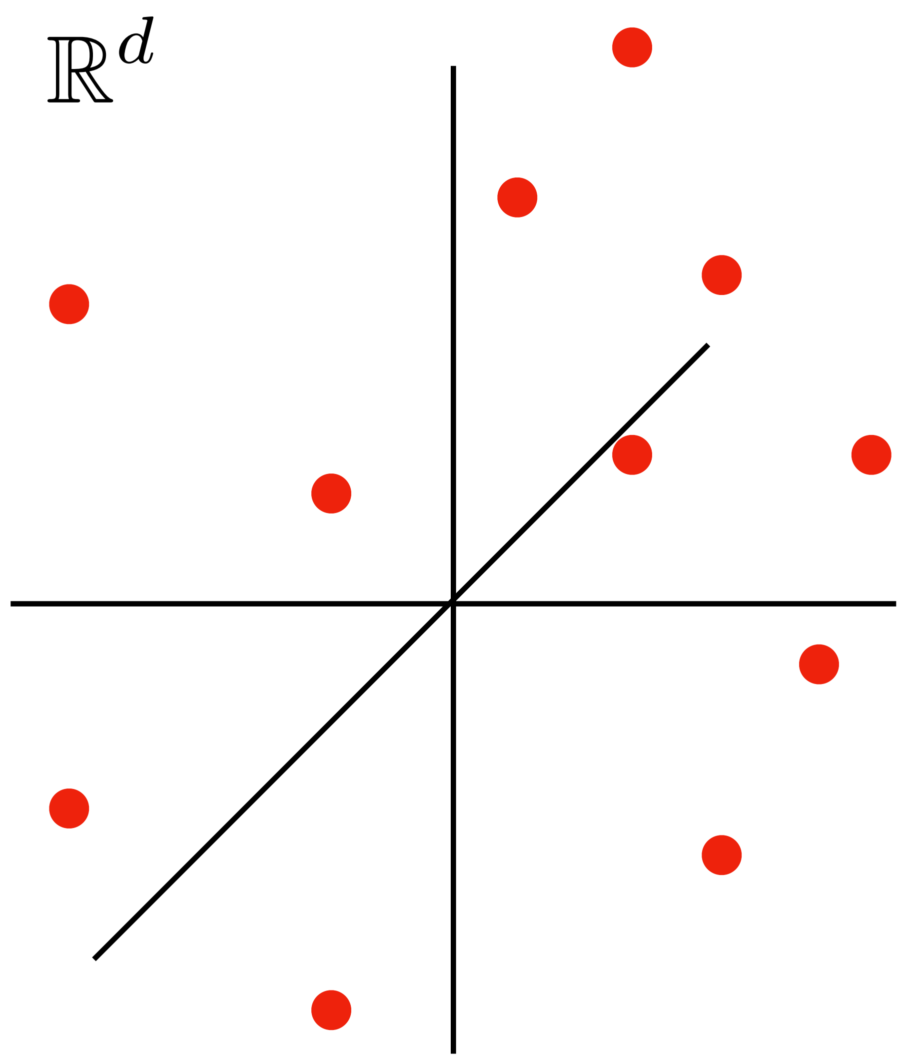
[PT'20]

Coresets via Discrepancy (PSD)

[Phillips-Tai '20]

Find "important" $p_1, \ldots, p_t \in P$

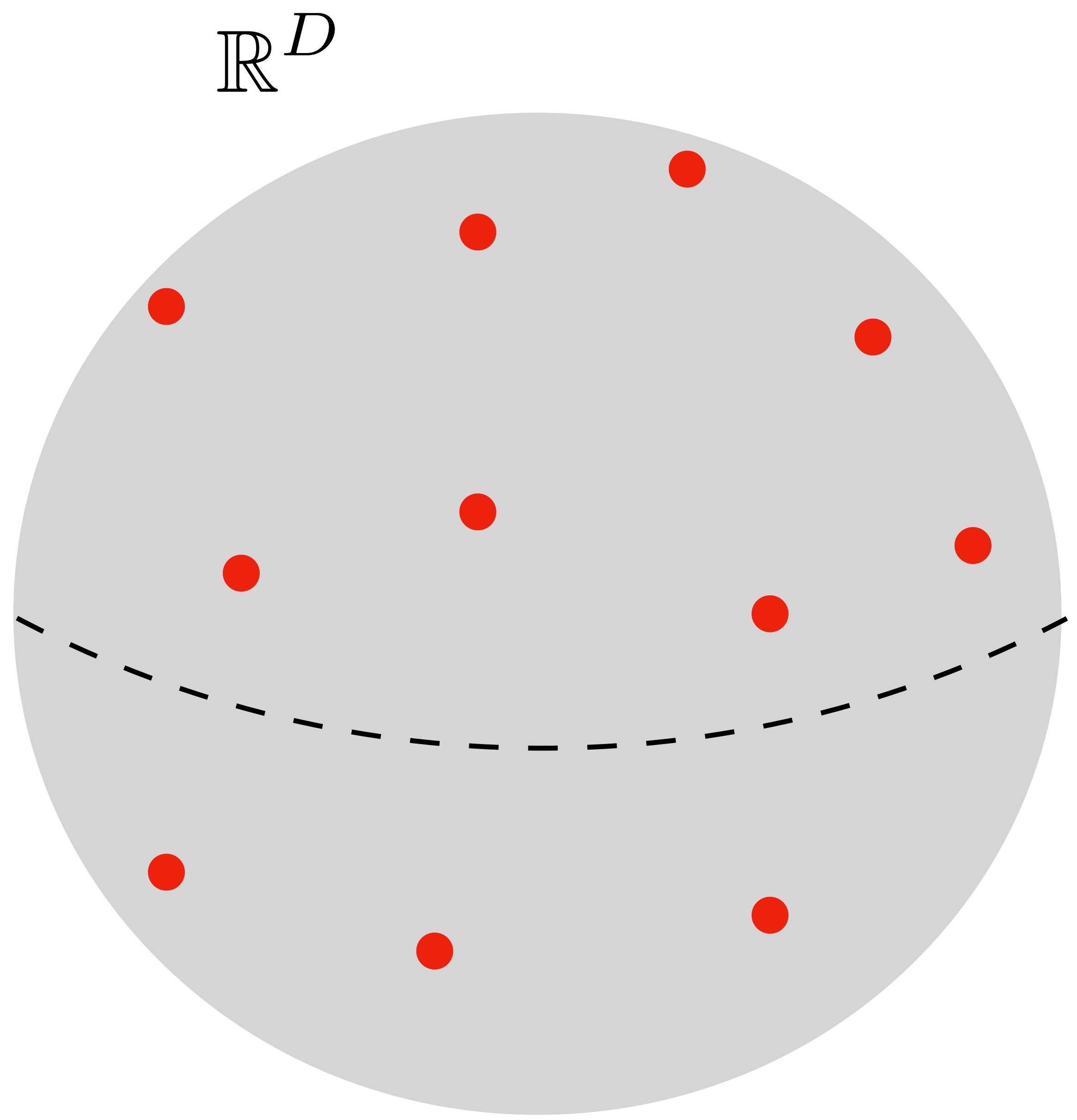
Query: $\frac{1}{t} \sum_{i=1}^{t} K(p_i, q)$.



ϕ

"Kernel Trick"

$$K(x, y) = \langle \phi(x), \phi(y) \rangle$$



Vector Balancing:

Find a setting of signs $a_1, \ldots, a_n \in \{-1, 1\}$ such that for (unknown) query q we minimize

$$\left| \sum_{a_i = 1} \langle \phi(p_i), \phi(q) \rangle - \sum_{a_i = -1} \langle \phi(p_i), \phi(q) \rangle \right| \leq \tilde{O}(1)$$

[PT'20]

As long as dataset is large, error incurred is at most $1/|P|$

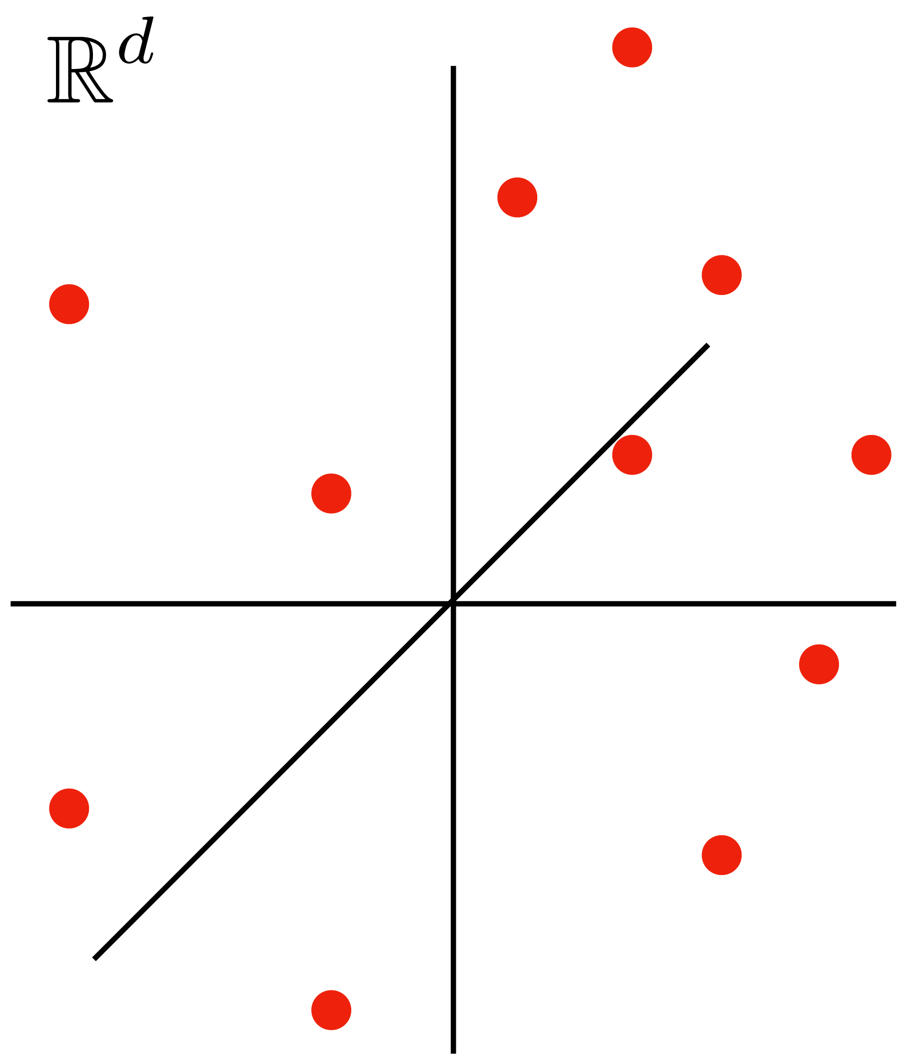
$\longrightarrow \tilde{O}\left(\frac{1}{\mu \epsilon}\right)$

Coresets via Discrepancy (PSD)

[Phillips-Tai '20]

Find "important" $p_1, \dots, p_t \in P$

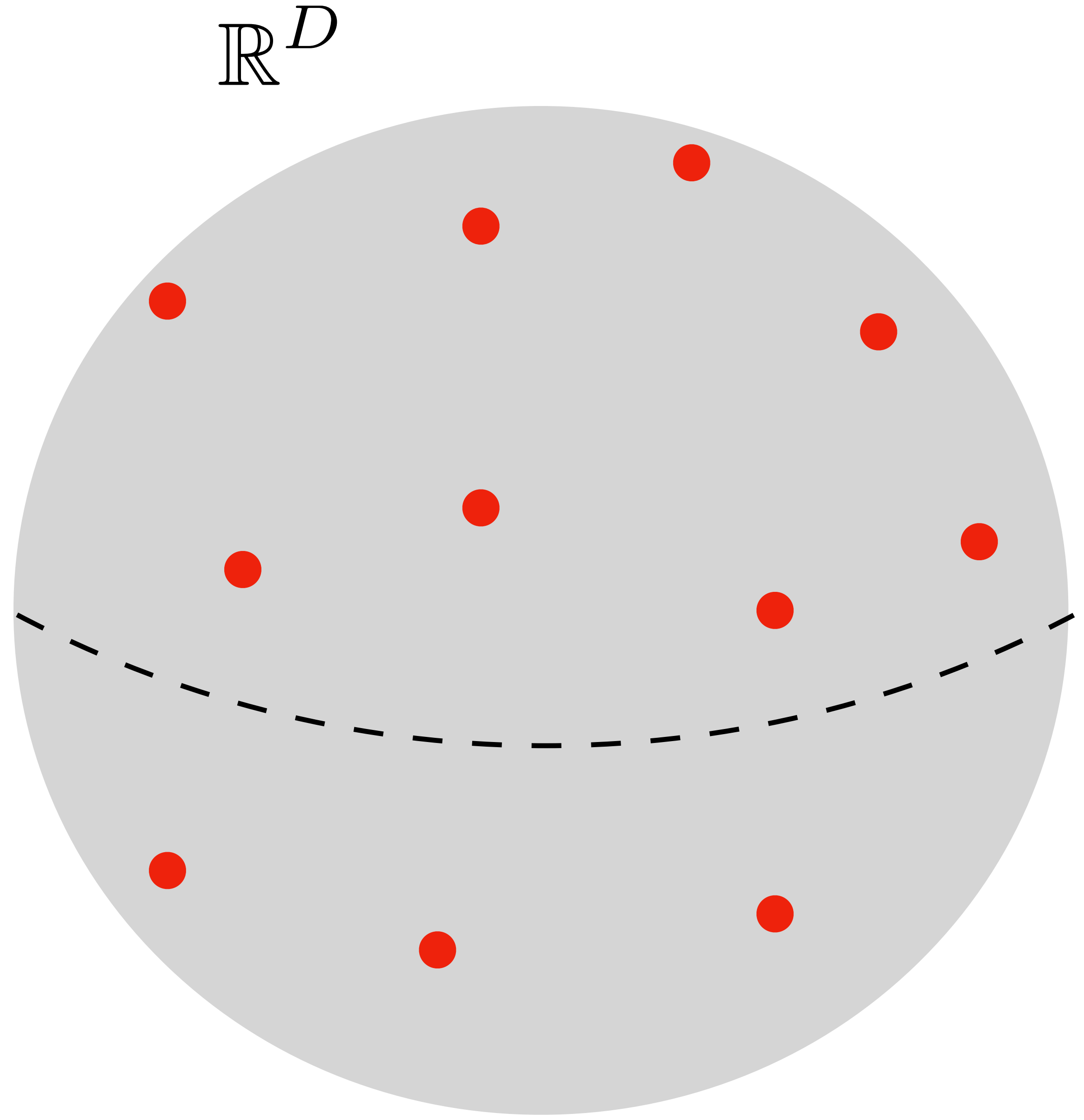
Query: $\frac{1}{t}\sum_{i=1}^{t} K(p_i, q)$.



ϕ

"Kernel Trick"

$$K(x,y) = \langle \phi(x), \phi(y) \rangle$$



Vector Balancing:

Find a setting of signs $a_1, \dots, a_n \in \{-1, 1\}$ such that for (unknown) query q we minimize

$$\left| \sum_{a_i=1} \langle \phi(p_i), \phi(q) \rangle - \sum_{a_i=-1} \langle \phi(p_i), \phi(q) \rangle \right| \leq \tilde{O}(1)$$

[PT'20]

As long as dataset is large, error incurred is at most $1/|P|$

$\longrightarrow \tilde{O}\left(\frac{1}{\mu\epsilon}\right)$

Main Idea?

1. Apply LSH.
2. Run discrepancy after.

Basics of Discrepancy Minimization

"Simplest" PSD Kernel:

- Linear kernel: $K(x, y) = x^\intercal y$

- 1-dimensional :-): $K(x, y) = x \cdot y$

Basics of Discrepancy Minimization

"Simplest" PSD Kernel:

- Linear kernel: $K(x, y) = x^\intercal y$

- 1-dimensional :-): $K(x, y) = x \cdot y$

- "Discrepancy Minimization Task:"

  - Input: $p_1, \ldots, p_n \in \mathbb{R}$

  - Find signs $a_1, \ldots, a_n \in \{-1, 1\}$

    to minimize $\left| q \sum_{i=1}^{n} a_i \cdot p_i \right|$.

Basics of Discrepancy Minimization

"Simplest" PSD Kernel:

- Linear kernel: $K(x, y) = x^{\intercal} y$

- 1-dimensional :-): $K(x, y) = x \cdot y$

- "Discrepancy Minimization Task:"
  - Input: $p_1, \ldots, p_n \in \mathbb{R}$
  - Find signs $a_1, \ldots, a_n \in \{-1, 1\}$

    to minimize $\left| q \sum_{i=1}^{n} a_i \cdot p_i \right|$.

- **You know** $p_1, \ldots, p_n \in \mathbb{R}$
- **You don't know** q

Basics of Discrepancy Minimization

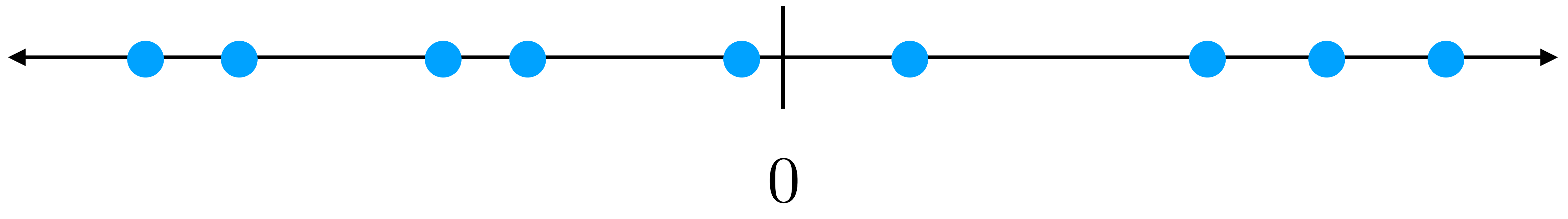
"Simplest" PSD Kernel:

- Linear kernel: $K(x, y) = x^\intercal y$

- 1-dimensional :-): $K(x, y) = x \cdot y$

- "Discrepancy Minimization Task:"
  - Input: $p_1, \ldots, p_n \in \mathbb{R}$
  - Find signs $a_1, \ldots, a_n \in \{-1, 1\}$

    to minimize $\left| q \sum_{i=1}^{n} a_i \cdot p_i \right|$.

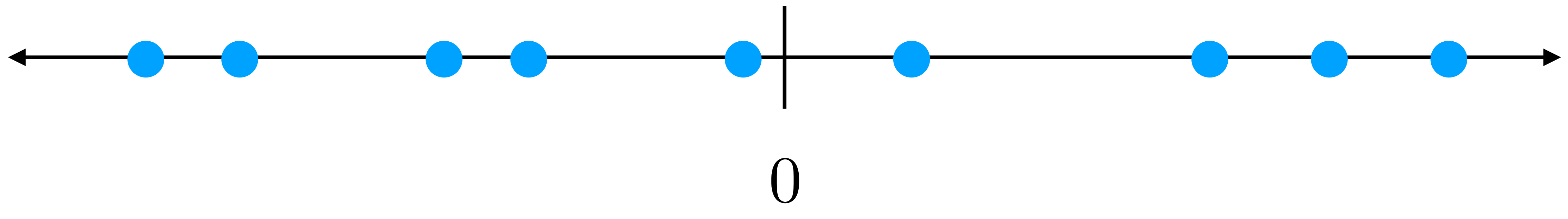


Basics of Discrepancy Minimization

"Simplest" PSD Kernel:

- Linear kernel: $K(x, y) = x^\intercal y$
- 1-dimensional :-): $K(x, y) = x \cdot y$
- "Discrepancy Minimization Task:"
  - Input: $p_1, \ldots, p_n \in \mathbb{R}$
  - Find signs $a_1, \ldots, a_n \in \{-1, 1\}$

    to minimize $\left| q \sum_{i=1}^{n} a_i \cdot p_i \right|$.



Random signs: $|q| \cdot \left(\sum_{i=1}^{n} p_i^2 \right)^{1/2}$

Basics of Discrepancy Minimization

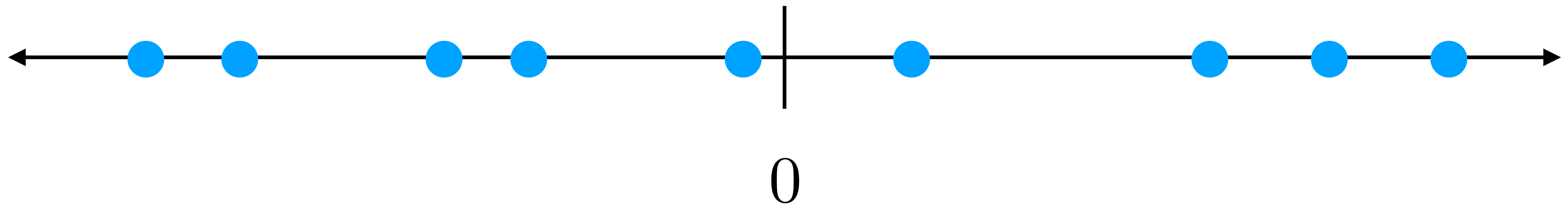
"Simplest" PSD Kernel:

- Linear kernel: $K(x, y) = x^\intercal y$

- 1-dimensional :-): $K(x, y) = x \cdot y$

- "Discrepancy Minimization Task:"
  - Input: $p_1, \ldots, p_n \in \mathbb{R}$
  - Find signs $a_1, \ldots, a_n \in \{-1, 1\}$

    to minimize $\left| q \sum_{i=1}^{n} a_i \cdot p_i \right|$.



Random signs: $|q| \cdot \left(\sum_{i=1}^{n} p_i^2 \right)^{1/2}$

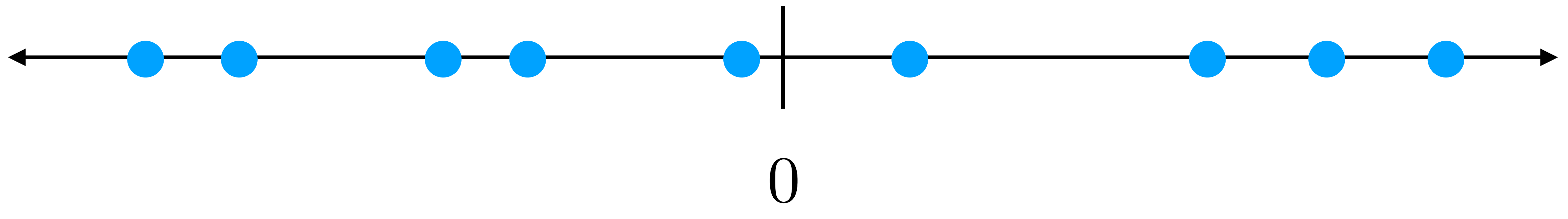
Greedy balance:

Basics of Discrepancy Minimization

"Simplest" PSD Kernel:

- Linear kernel: $K(x, y) = x^\intercal y$

- 1-dimensional :-): $K(x, y) = x \cdot y$



- "Discrepancy Minimization Task:"
  - Input: $p_1, \ldots, p_n \in \mathbb{R}$
  - Find signs $a_1, \ldots, a_n \in \{-1, 1\}$

    to minimize $\left| q \sum_{i=1}^{n} a_i \cdot p_i \right|$.

Random signs: $|q| \cdot \left(\sum_{i=1}^{n} p_i^2 \right)^{1/2}$

Greedy balance: $|q| \cdot \max_i |p|$

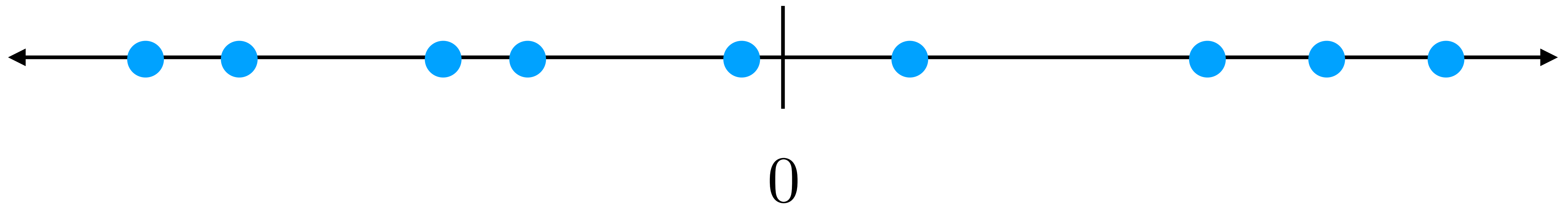
Basics of Discrepancy Minimization

General

~~"Simplest"~~ PSD Kernel:

- ~~Linear kernel~~: $K(x,y) = \phi(x)^\intercal \phi(y)$
- ~~1-dimensional :-): $K(x,y) = x \cdot y$~~
- "Discrepancy Minimization Task:"
  - Input: $\phi(p_1), \ldots, \phi(p_n) \in \mathbb{R}^D$
  - Find signs $a_1, \ldots, a_n \in \{-1, 1\}$

    to minimize $\left| \phi(q)^\intercal \left(\sum_{i=1}^{n} a_i \phi(p_i) \right) \right|$.

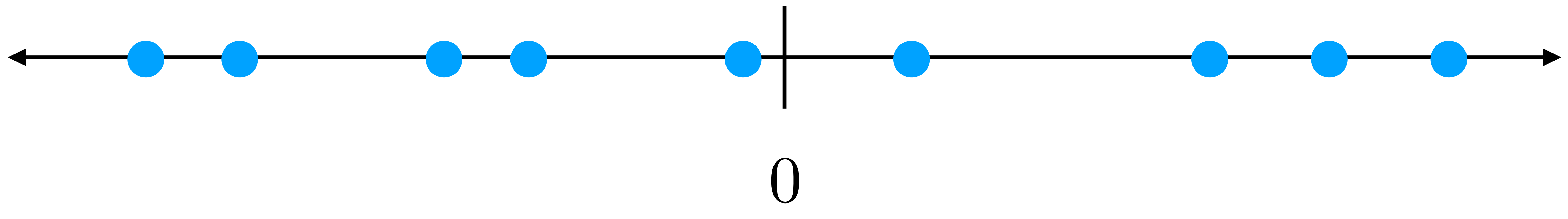


Basics of Discrepancy Minimization

General
~~“Simplest”~~ PSD Kernel:

- ~~Linear kernel~~: $K(x,y) = \phi(x)^\intercal \phi(y)$
- ~~1-dimensional :-): $K(x,y) = x \cdot y$~~
- “Discrepancy Minimization Task:”
  - Input: $\phi(p_1), \ldots, \phi(p_n) \in \mathbb{R}^D$
  - Find signs $a_1, \ldots, a_n \in \{-1, 1\}$

    to minimize $\left|\phi(q)^\intercal \left(\sum_{i=1}^{n} a_i \phi(p_i)\right)\right|$.



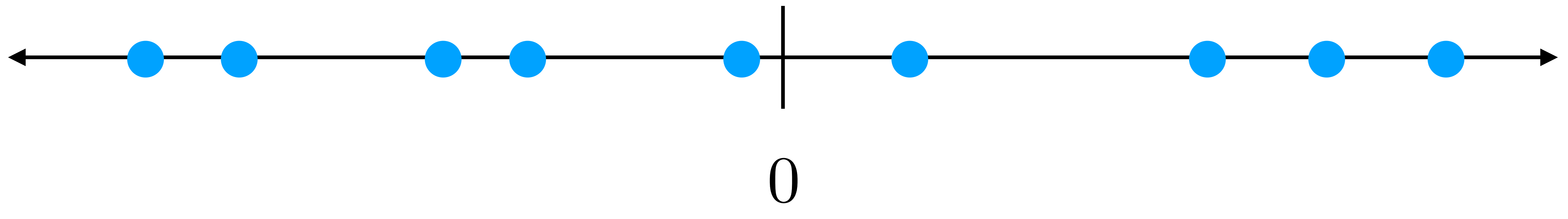
Random signs: $\|\phi(q)\|_2 \cdot \left(\sum_{i=1}^{n} \|\phi(p_i)\|_2^2\right)^{1/2}$

Basics of Discrepancy Minimization

General

~~"Simplest"~~ PSD Kernel:

- ~~Linear kernel~~: $K(x,y) = \phi(x)^\intercal \phi(y)$
- ~~1-dimensional :-): $K(x,y) = x \cdot y$~~



- "Discrepancy Minimization Task:"
  - Input: $\phi(p_1), \ldots, \phi(p_n) \in \mathbb{R}^D$
  - Find signs $a_1, \ldots, a_n \in \{-1, 1\}$

    to minimize $\left| \phi(q)^\intercal \left(\sum_{i=1}^{n} a_i \phi(p_i) \right) \right|$.

Random signs: $\|\phi(q)\|_2 \cdot \left(\sum_{i=1}^{n} \|\phi(p_i)\|_2^2 \right)^{1/2}$

"Gram-Schmidt Walk"

[Bansal, Dadush, Garg, Lovett '17]

"Self-balancing Walk"

[Alweiss, Liu, Sawhney '20]

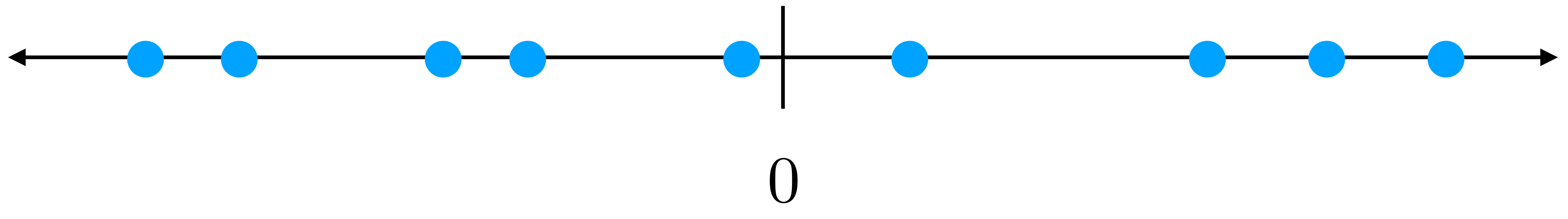
$$\tilde{O}(1) \cdot \|\phi(q)\|_2 \cdot \max_i \|\phi(p_i)\|_2$$

Basics of Discrepancy Minimization

General
~~"Simplest"~~ PSD Kernel:

- ~~Linear kernel~~: $K(x, y) = \phi(x)^\intercal \phi(y)$

- ~~1-dimensional :-): $K(x, y) = x \cdot y$~~



- "Discrepancy Minimization Task:"
  - Input: $\phi(p_1), \ldots, \phi(p_n) \in \mathbb{R}^D$
  - Find signs $a_1, \ldots, a_n \in \{-1, 1\}$

    to minimize $\left| \phi(q)^\intercal \left(\sum_{i=1}^{n} a_i \phi(p_i) \right) \right|$.

Random signs: $\|\phi(q)\|_2 \cdot \left(\sum_{i=1}^{n} \|\phi(p_i)\|_2^2 \right)^{1/2}$

"Gram-Schmidt Walk"
[Bansal, Dadush, Garg, Lovett '17]
"Self-balancing Walk"
[Alweiss, Liu, Sawhney '20]

γ_2-norm ← $\tilde{O}(1) \cdot \|\phi(q)\|_2 \cdot \max_i \|\phi(p_i)\|_2$

Basics of Discrepancy Minimization

General
~~"Simplest"~~ PSD Kernel:

- ~~Linear kernel~~: $K(x, y) = \phi(x)^\intercal \phi(y)$
- ~~1-dimensional :-): $K(x, y) = x \cdot y$~~
- "Discrepancy Minimization Task:"
  - Input: $\phi(p_1), \ldots, \phi(p_n) \in \mathbb{R}^D$
  - Find signs $a_1, \ldots, a_n \in \{-1, 1\}$

    to minimize $\left| \phi(q)^\intercal \left(\sum_{i=1}^{n} a_i \phi(p_i) \right) \right|$.

$$\gamma_2(K) = \min_{\phi} \max_{q,p} \|\phi(q)\|_2 \|\phi(p)\|_2$$

Random signs: $\|\phi(q)\|_2 \cdot \left(\sum_{i=1}^{n} \|\phi(p_i)\|_2^2 \right)^{1/2}$

"Gram-Schmidt Walk"
[Bansal, Dadush, Garg, Lovett '17]
"Self-balancing Walk"
[Alweiss, Liu, Sawhney '20]

γ_2-norm ← $\tilde{O}(1) \cdot \|\phi(q)\|_2 \cdot \max_i \|\phi(p_i)\|_2$

[Phillips-Tai '20]
PSD of Bounded Kernel Directly Applies

$K \colon \mathbb{R}^d \times \mathbb{R}^d \to [0,1]$

PSD and Radially Decaying

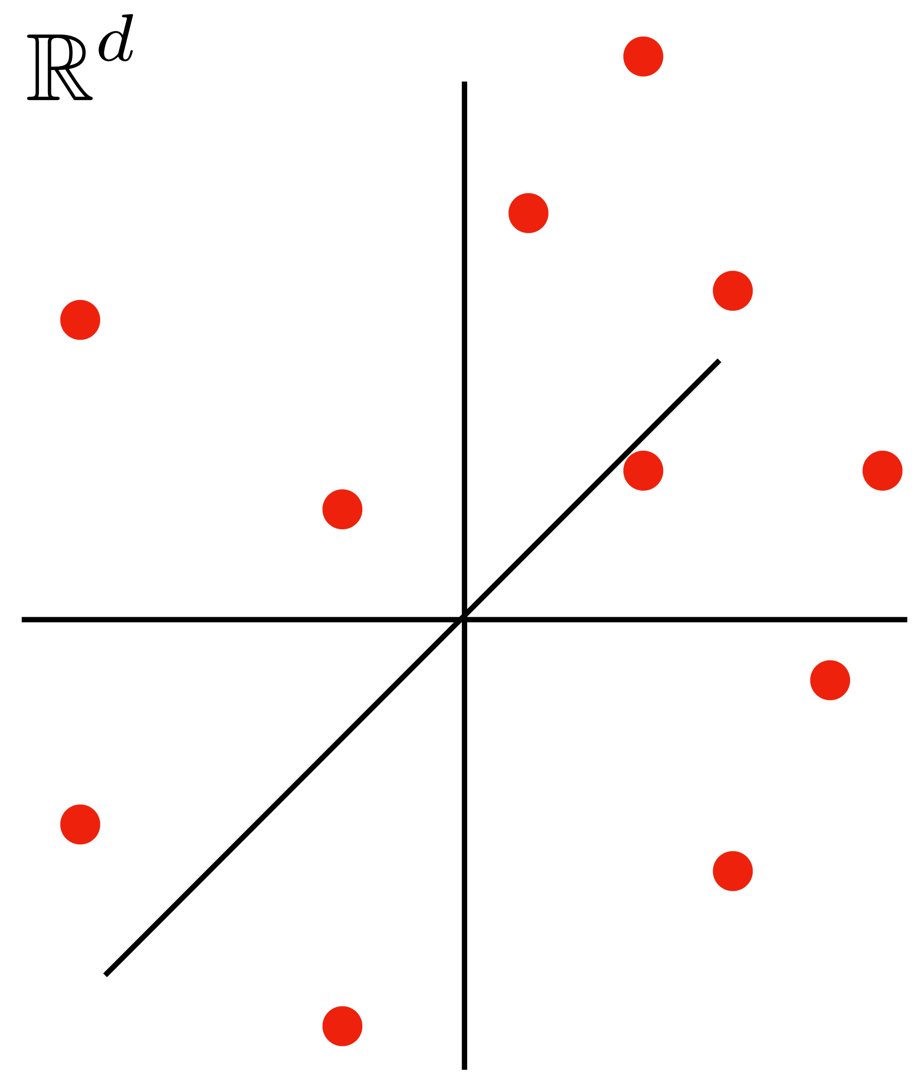
$$\tilde{O}(1) \cdot \underbrace{\max\{\|\phi(q)\|_2 \cdot \|\phi(p)\|_2\}}_{\gamma_2\text{-norm}}$$

$$\begin{aligned}\|\phi(p)\|_2^2 &= \langle \phi(p), \phi(p) \rangle \\ &= K(p,p) = 1\end{aligned}$$

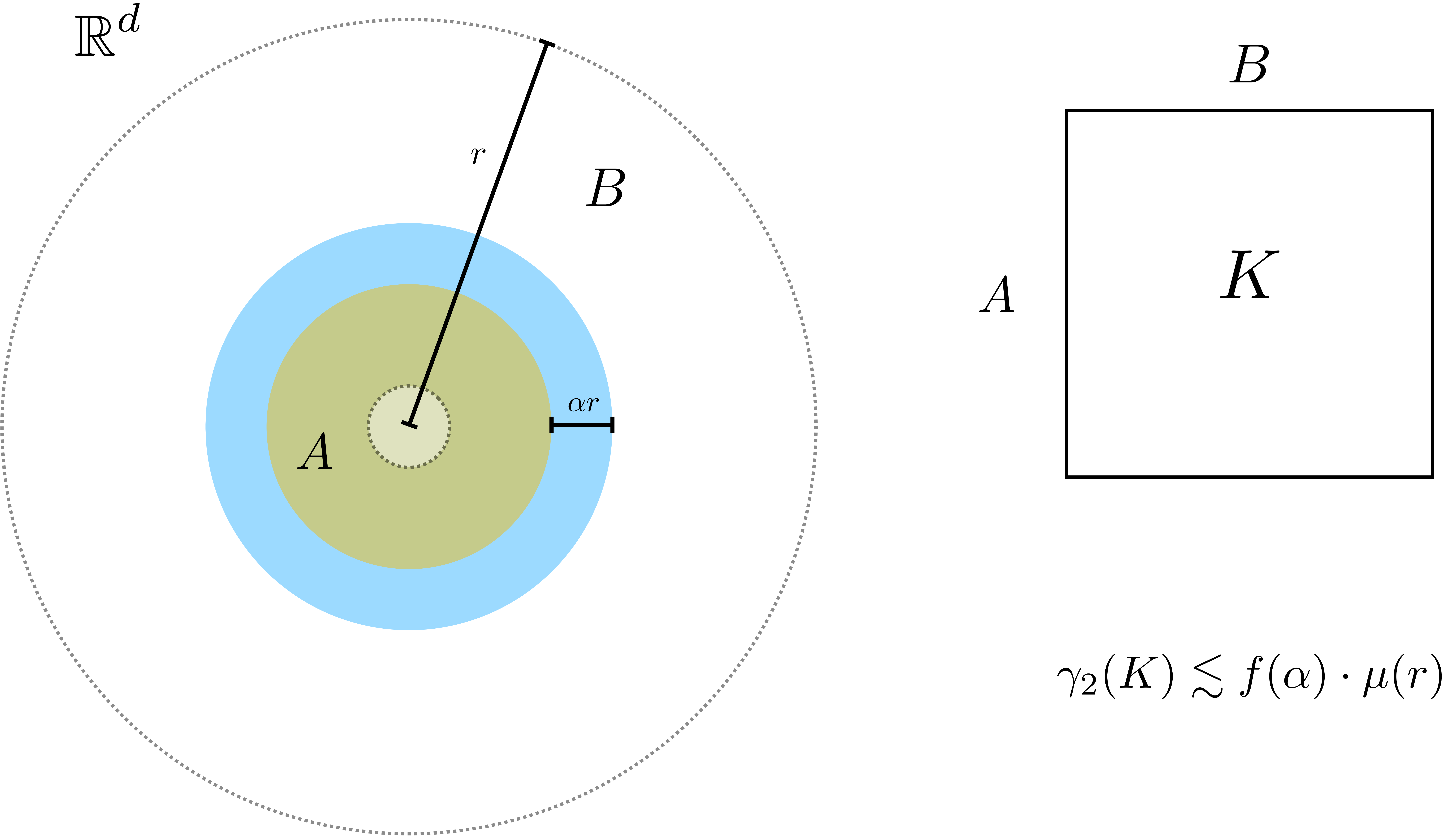
$$\implies \gamma_2\text{-norm} \leq 1$$

What if we could…

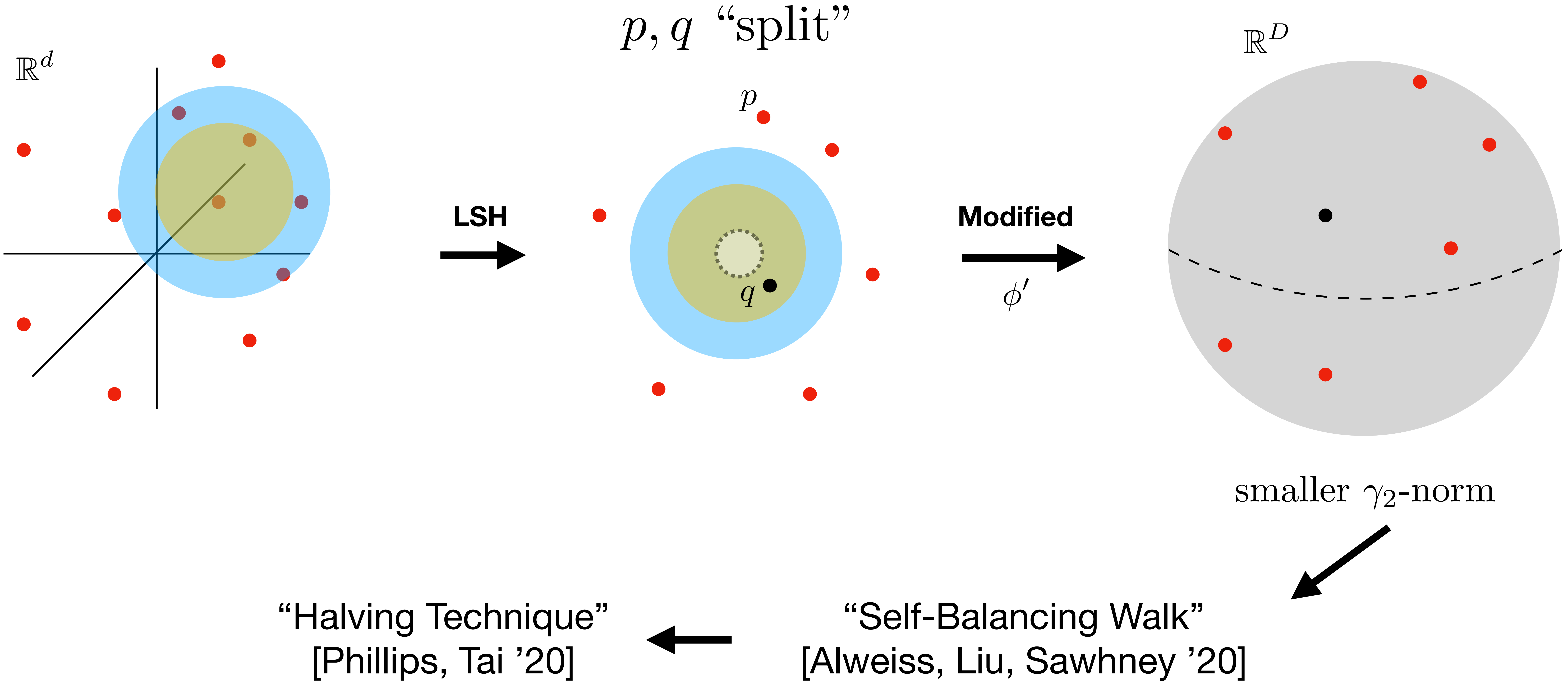
1. Decompose dataset efficiently (LSH)
2. For each part, find *better* embedding ϕ with smaller γ_2-norm.



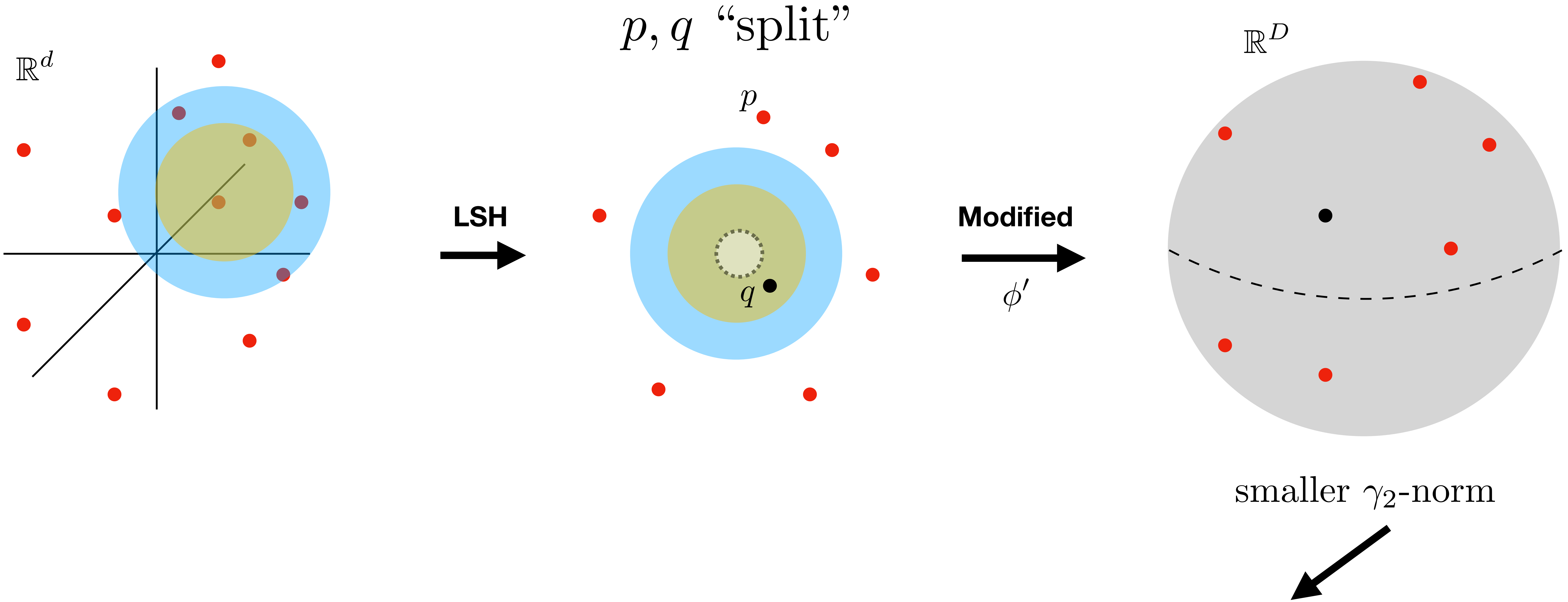
$$K : \mathbb{R}^d \times \mathbb{R}^d \to [0, 1]$$


$\mathbb{R}^d$
r
B
αr
A
B
K
A
$\gamma_2(K) \lesssim f(\alpha) \cdot \mu(r)$

At “scale” R:


$\mathbb{R}^d$
LSH
p, q “split”
p
q
Modified
ϕ'
$\mathbb{R}^D$
smaller γ_2-norm
“Self-Balancing Walk”
[Alweiss, Liu, Sawhney ’20]
“Halving Technique”
[Phillips, Tai ’20]

At “scale” R:

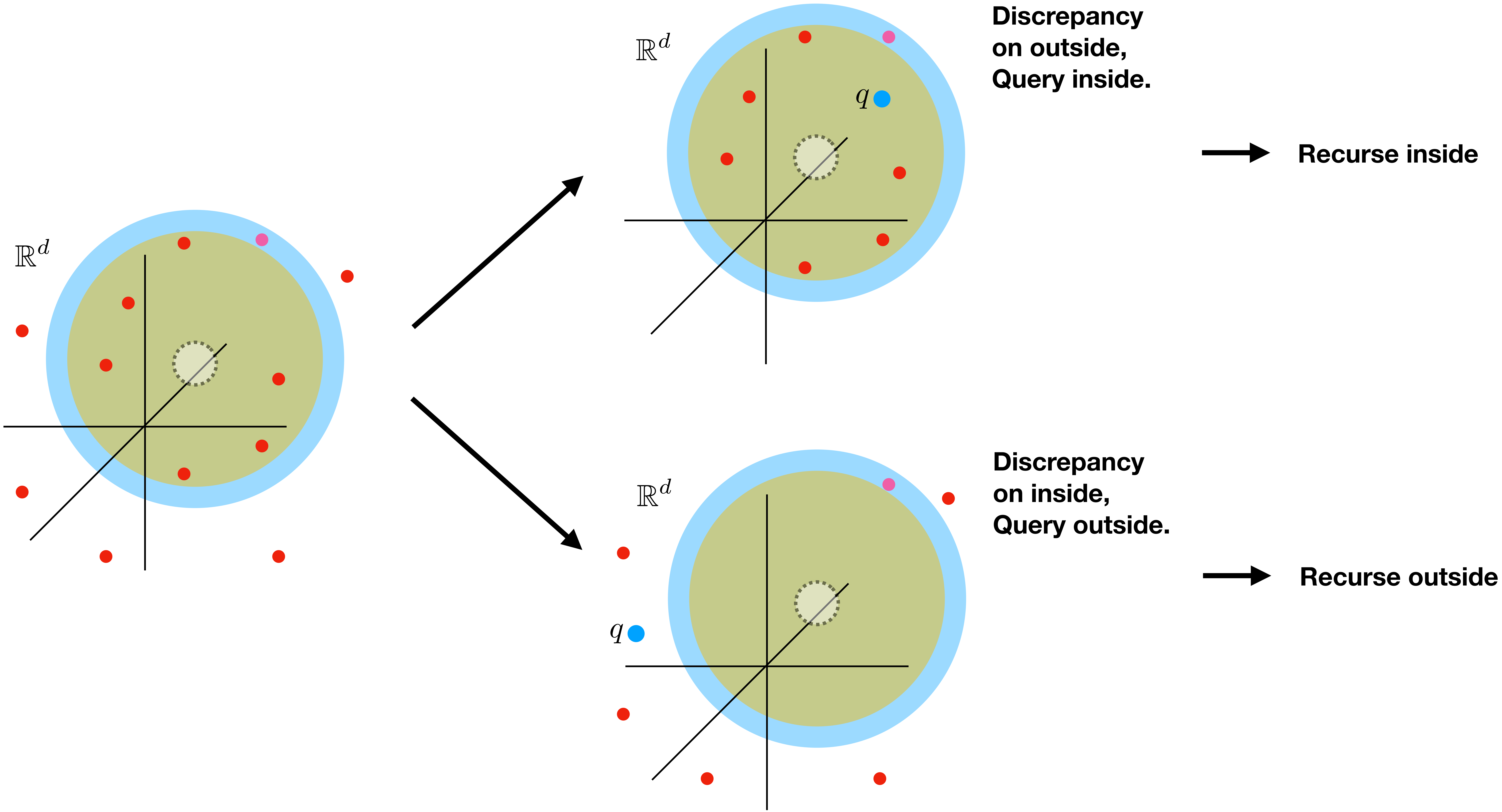


“Self-Balancing Walk”
[Alweiss, Liu, Sawhney ’20]

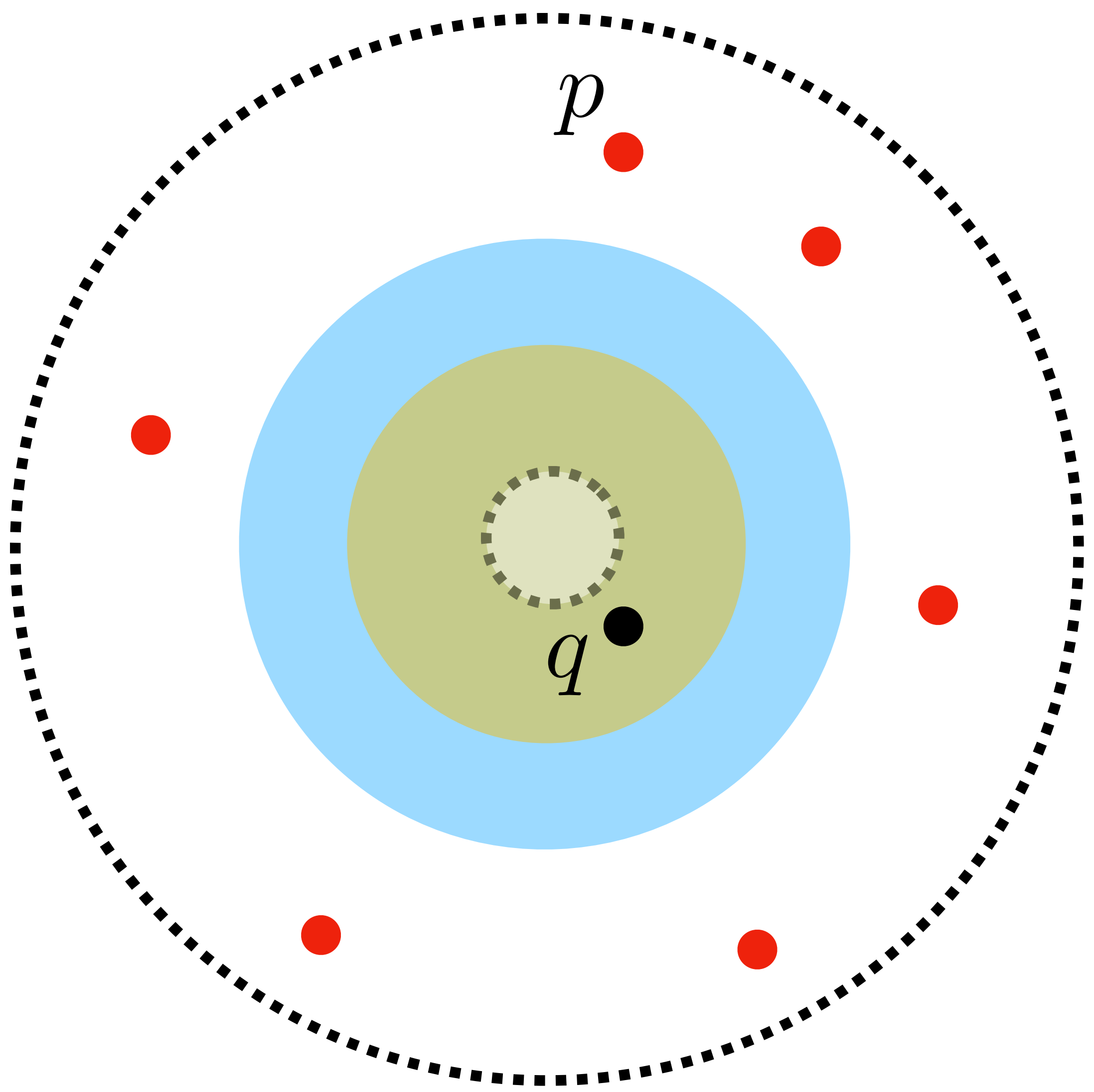
“Halving Technique”
[Phillips, Tai ’20]

$$\underbrace{O(\log \Phi)}_{\text{scales}} \times \underbrace{O(\log n)}_{\text{“caught”}} \times \underbrace{O(\gamma_2/(\epsilon\mu(R)))}_{\text{coreset size}}$$

Data Structure



Why we need smoothness:



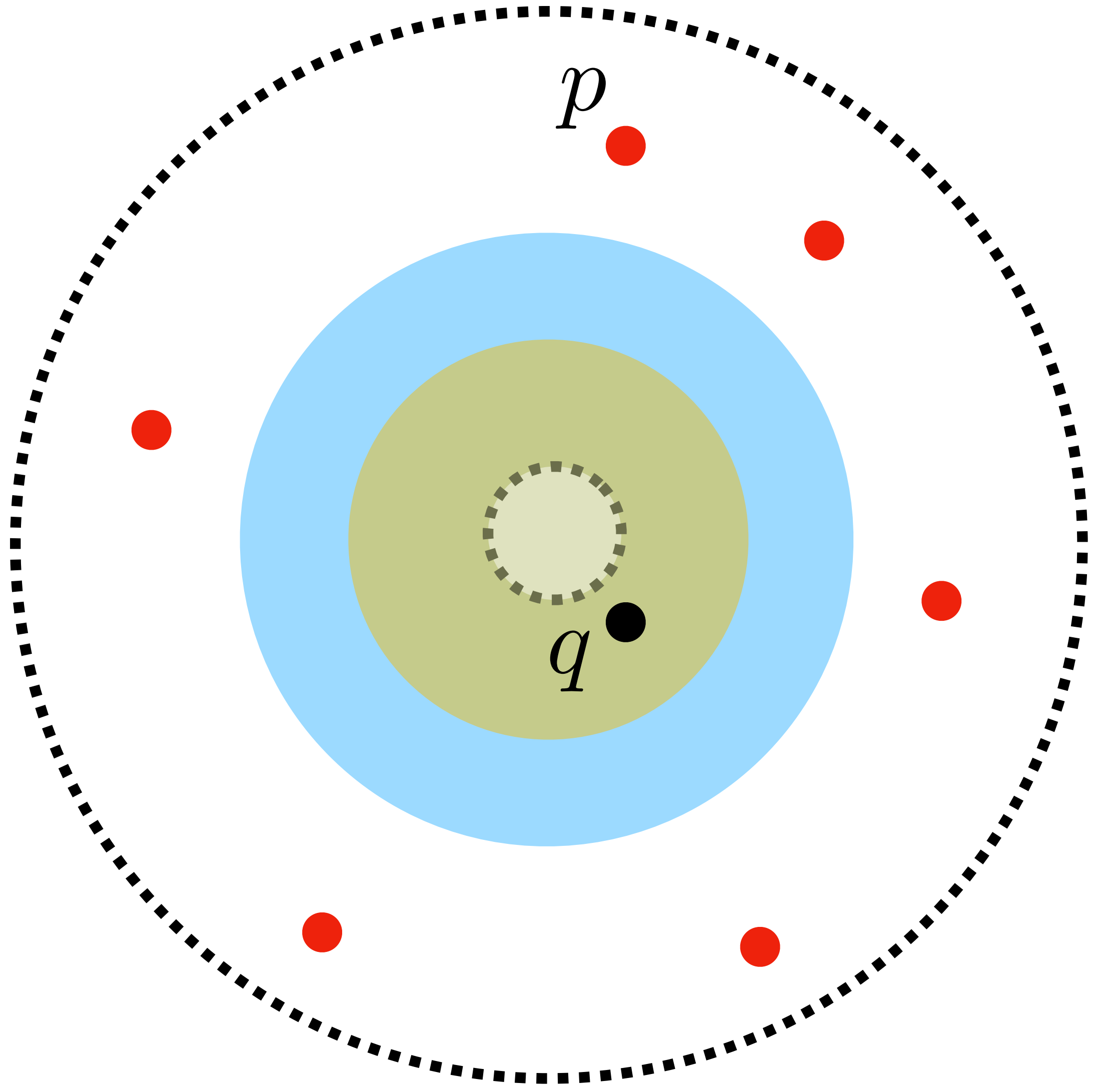
$$\|q - p\|_2 \approx R$$

$$K(p,q) \approx \mu(R)$$

Tension:

- How thick of a shell can I fit between a query and many dataset points at particular length?

Why we need smoothness:



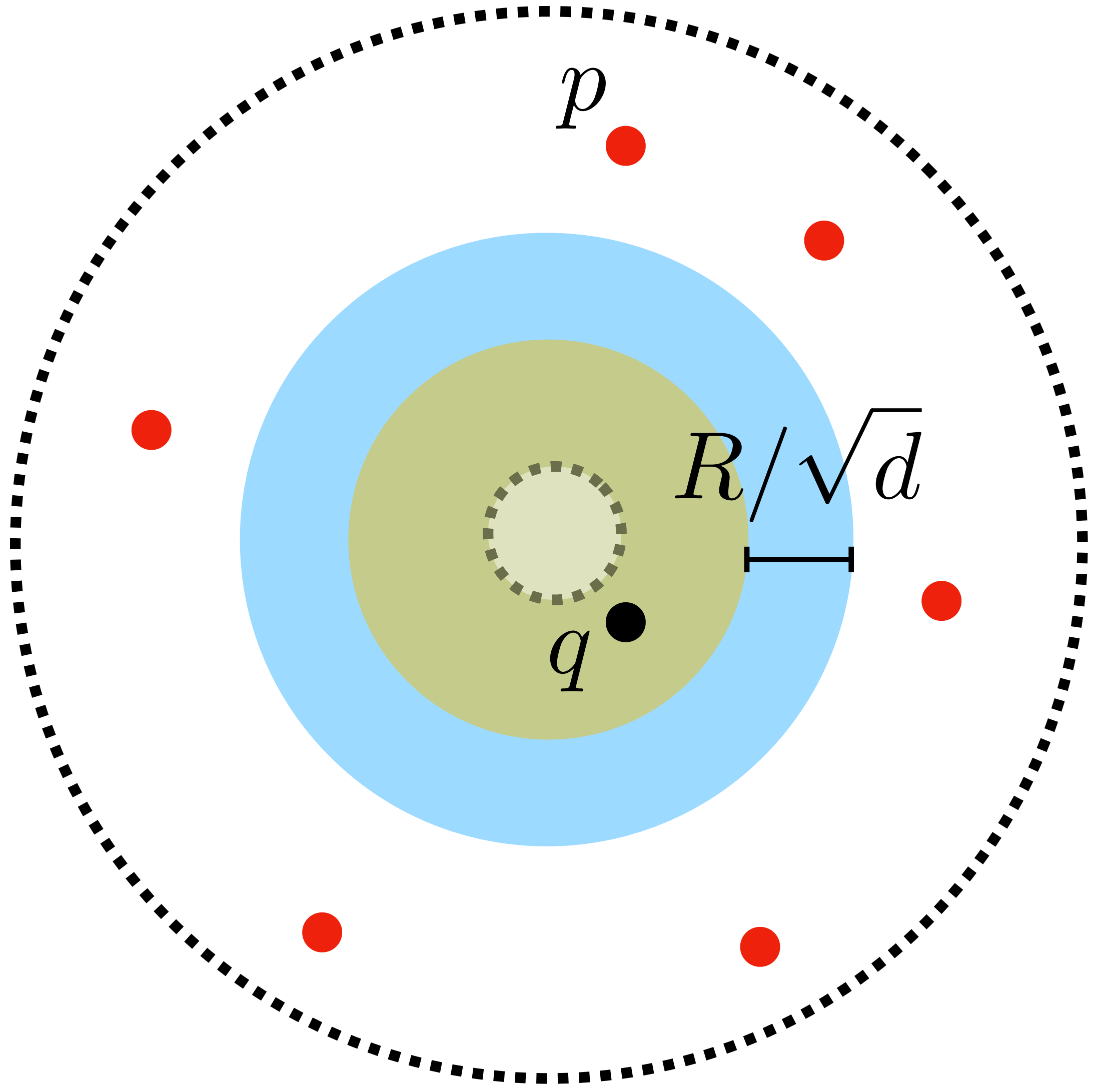
$$\|q - p\|_2 \approx R$$

$$K(p, q) \approx \mu(R)$$

Tension:

- How thick of a shell can I fit between a query and many dataset points at particular length?
  - Don’t know query location. Fix random shell and “get lucky” and split.
  - Suggest shell should be thin.

Why we need smoothness:



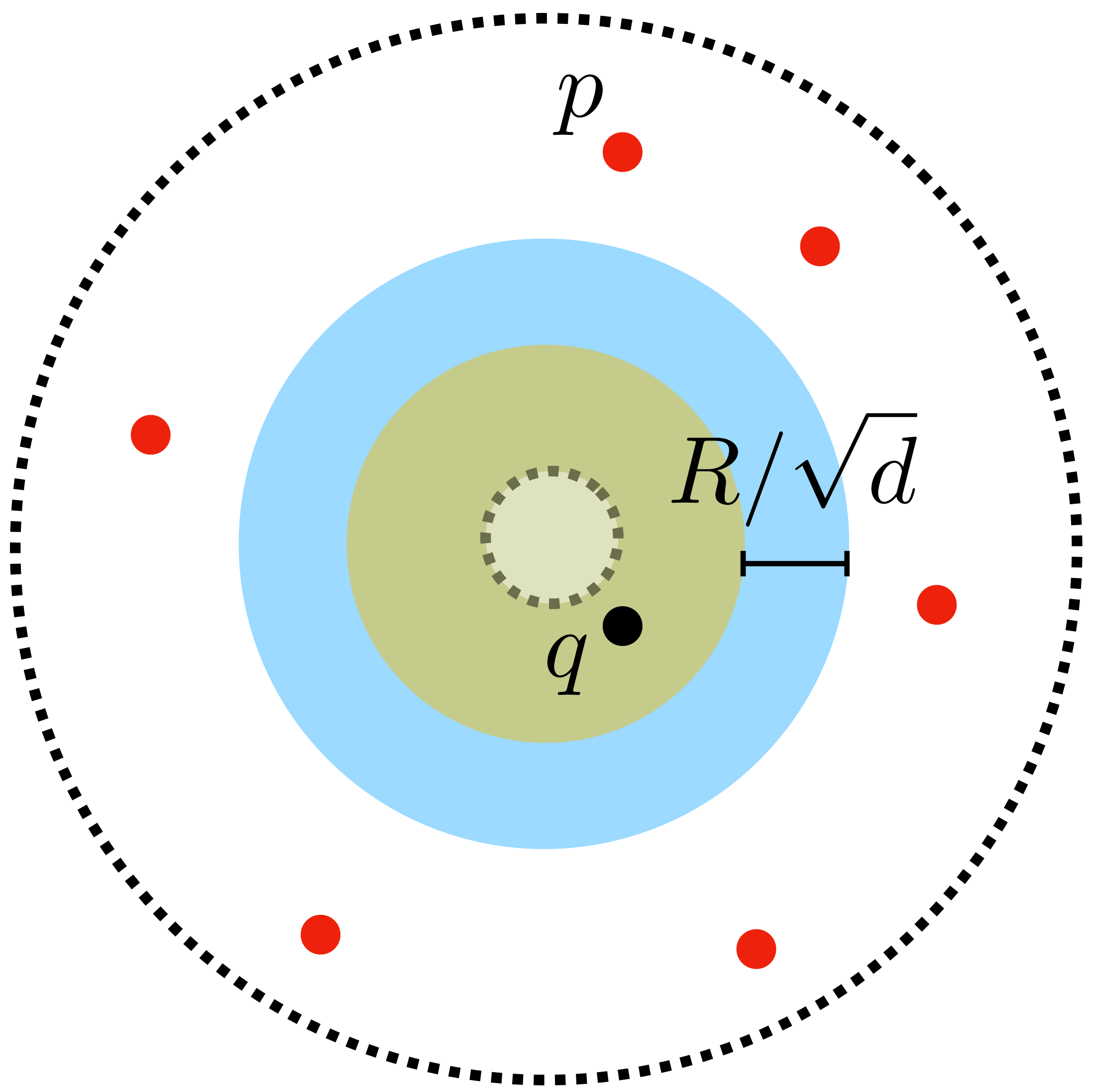
$$\|q - p\|_2 \approx R$$

$$K(p, q) \approx \mu(R)$$

Tension:

- How thick of a shell can I fit between a query and many dataset points at particular length?
  - Don’t know query location. Fix random shell and “get lucky” and split.
  - Suggest shell should be thin.

Why we need smoothness:



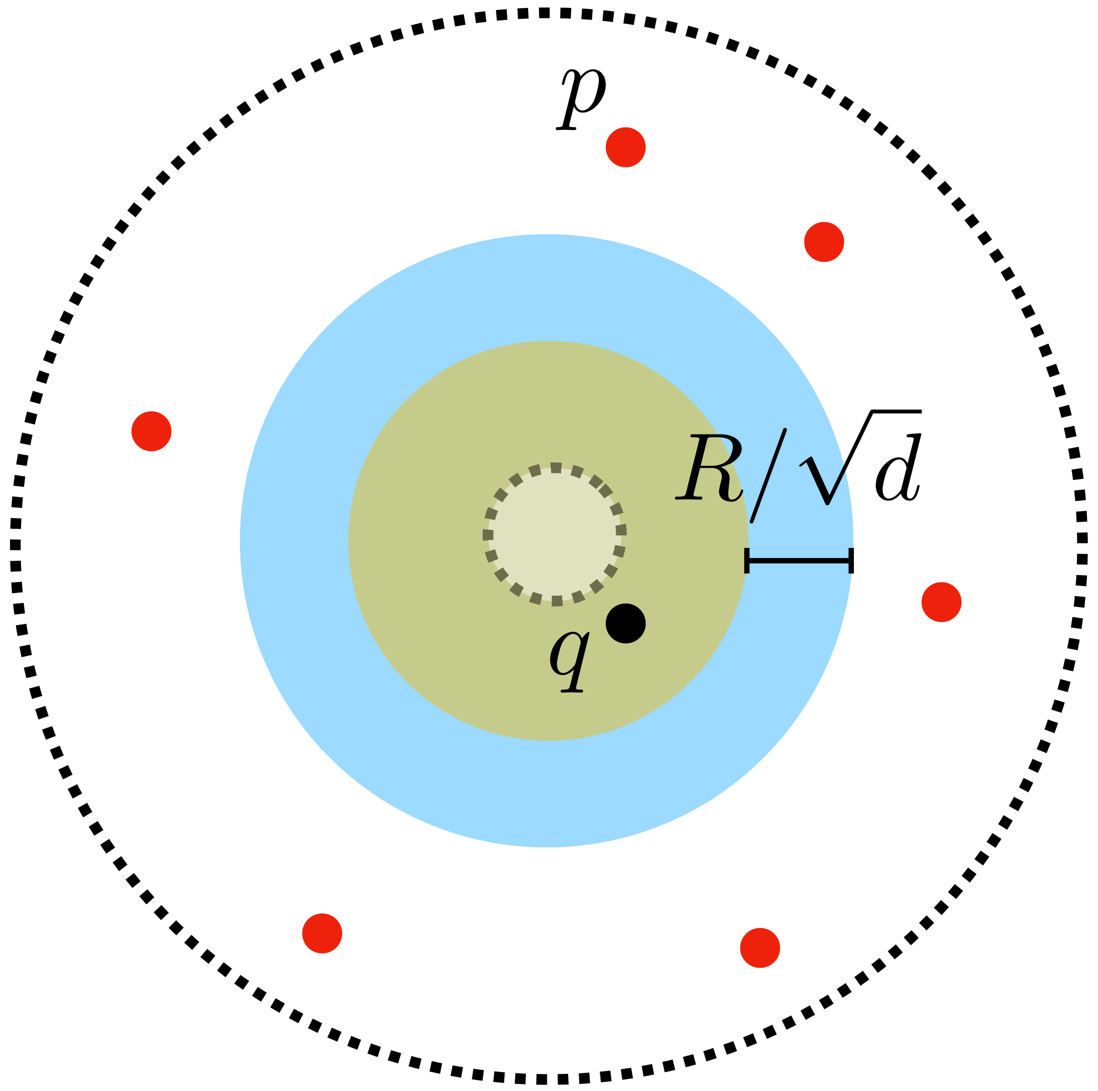
$$\|q - p\|_2 \approx R$$

$$K(p, q) \approx \mu(R)$$

Tension:

- How thick of a shell can I fit between a query and many dataset points at particular length?
  - Don't know query location. Fix random shell and "get lucky" and split.
  - Suggest shell should be thin.
- Split by thick shell lower bounds distance, certifies upper bound on $K(p, q)$ so γ_2-norm gets better ("errors" pay at most $\mu(R/\sqrt{d})$)

Why we need smoothness:



$$\|q - p\|_2 \approx R$$

$$K(p, q) \approx \mu(R)$$

$$\mu(R) \text{ vs. } \mu(R/\sqrt{d})?$$

Tension:

- How thick of a shell can I fit between a query and many dataset points at particular length?
  - Don't know query location. Fix random shell and "get lucky" and split.
  - Suggest shell should be thin.
- Split by thick shell lower bounds distance, certifies upper bound on $K(p, q)$ so γ_2-norm gets better ("errors" pay at most $\mu(R/\sqrt{d})$)

Open Problems

1. Non-smooth kernels?

2. Preprocessing is slower.