FriendlyCore: Tool for Differentially Private Aggregation

Eliad Tsfadia

Joint with: Edith Cohen, Haim Kaplan, Yishay Mansour, Uri Stemmer

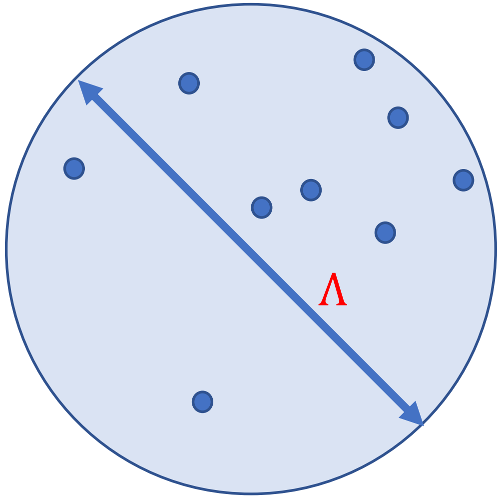



DP Averaging

DP Averaging

Input: Points $D \in \left(\mathbb{R}^d\right)^n$ in a ball of diameter Λ
Output: $\text{Avg}(D) + Noise$

$Noise \propto \Lambda$

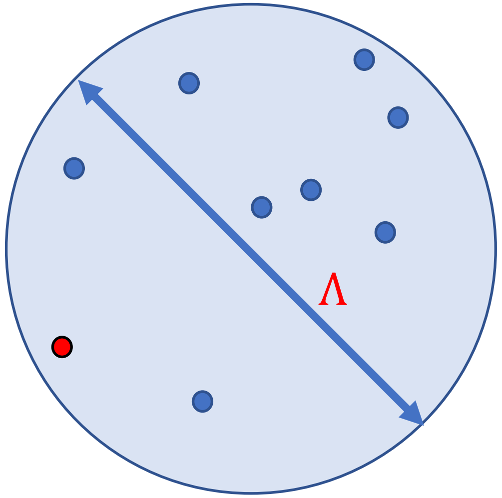


DP Averaging

Input: Points $D \in \left(\mathbb{R}^d\right)^n$ in a ball of diameter Λ

Output: $\mathrm{Avg}(D) + Noise$

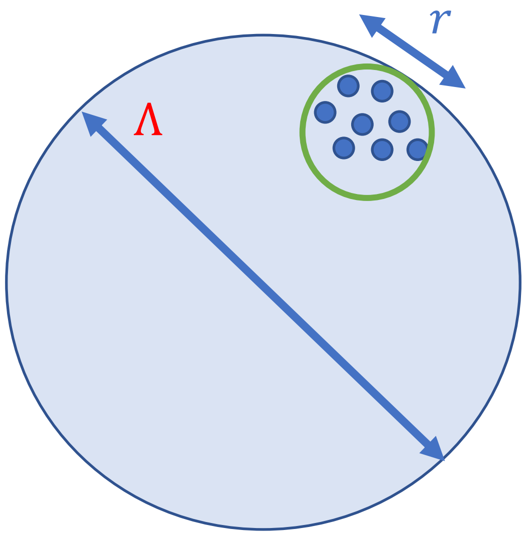
$Noise \propto \Lambda$



DP Averaging

Suppose D has diameter $r \ll \Lambda$

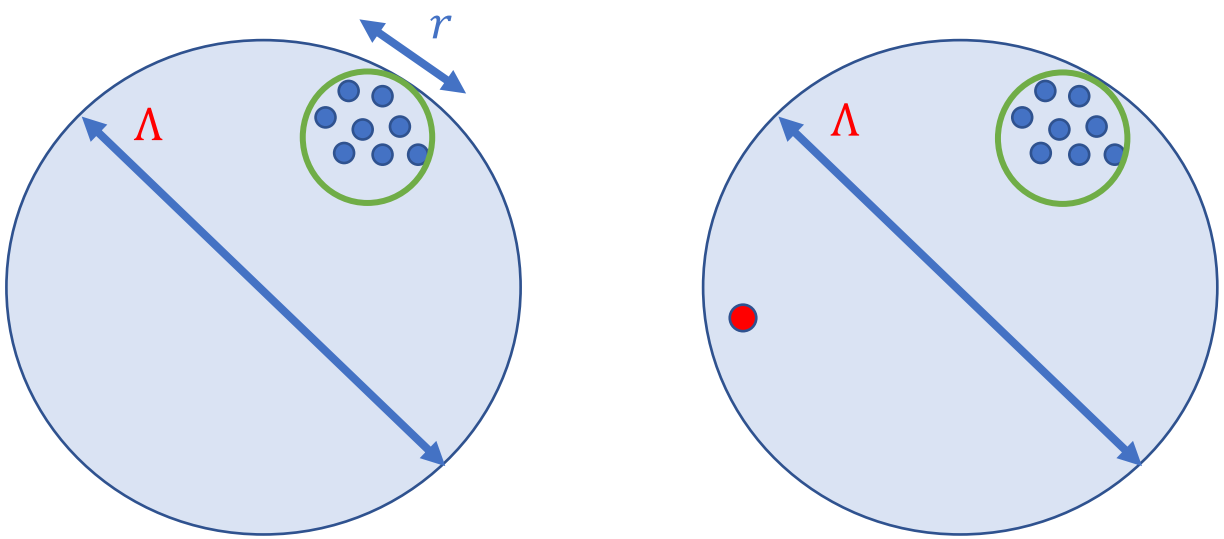
Wish: Replace $\Lambda \leftarrow r$ in $Noise$ $\Rightarrow$ $\times$ Λ / r to gain in accuracy



DP Averaging

Suppose D has diameter $r \ll \Lambda$

Wish: Replace $\Lambda \leftarrow r$ in $Noise$ $\Rightarrow$ $\times$ Λ/r to gain in accuracy



With DP: (almost) same output

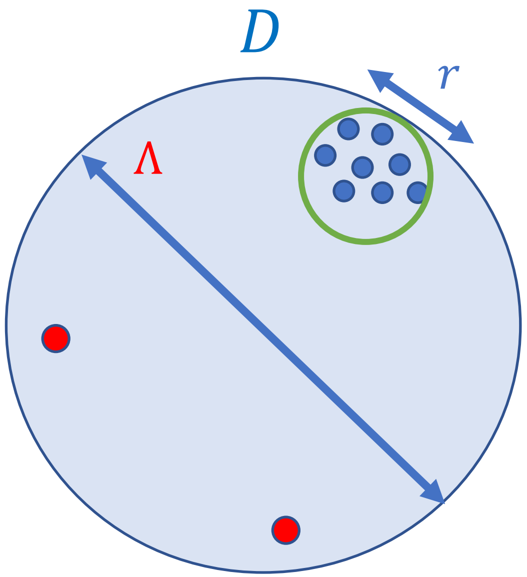
FriendlyCore Paradigm

Input:

- Dataset D of points

Operation:

$C \leftarrow \text{FriendlyCore}(D) \qquad (C \subseteq D)$

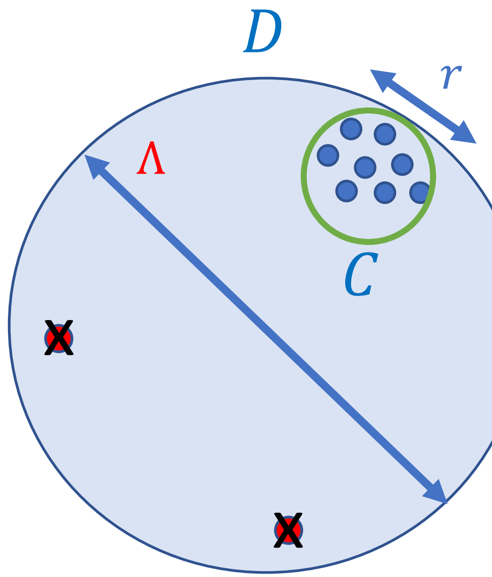


FriendlyCore Paradigm

Input:
- Dataset D of points

Operation:

$C \leftarrow \text{FriendlyCore}(D) \quad (C \subseteq D)$



FriendlyCore Paradigm

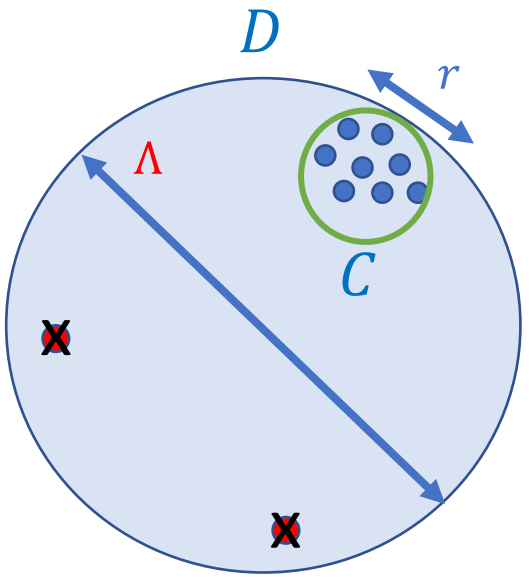
Input:

- Dataset D of points

Operation:

$C \leftarrow \text{FriendlyCore}(D) \qquad (C \subseteq D)$

C is``friendly''



FriendlyCore Paradigm

Input:

- Dataset D of points

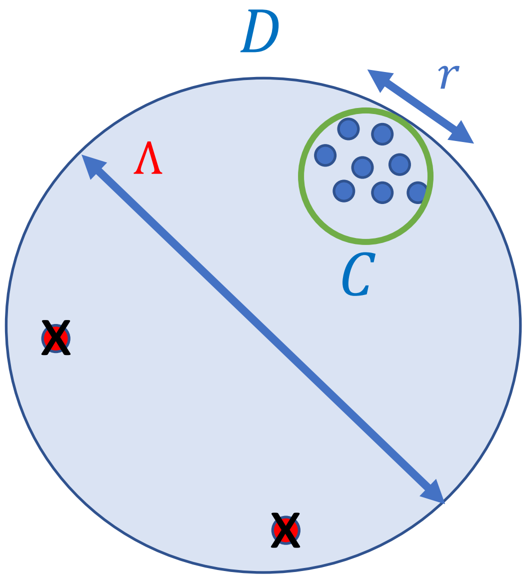
Operation:

$C \leftarrow \text{FriendlyCore}(D) \qquad (C \subseteq D)$

Output: $A(C)$

A is ``friendly'' DP algorithm
(weaker notion of privacy)

C is``friendly''



FriendlyCore Paradigm

Input:
- Dataset D of points

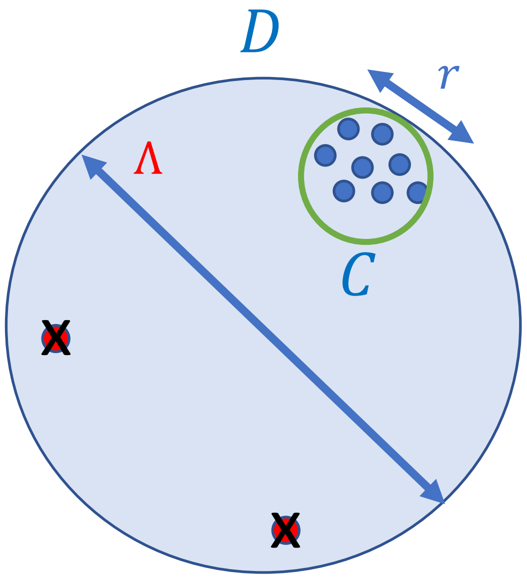
Operation:

$C \leftarrow \text{FriendlyCore}(D) \qquad (C \subseteq D)$

Output: $A(C)$

A is ``friendly'' DP algorithm
(weaker notion of privacy)

C is``friendly''



Averaging example:

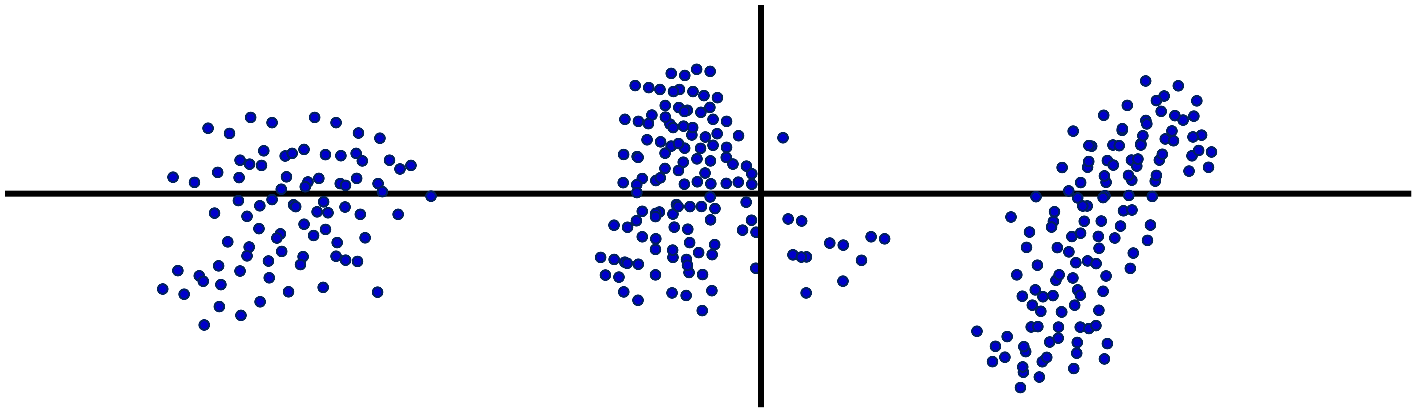
friendly C $\Rightarrow$ has diameter $r \ll \Lambda$

friendly DP A $\Rightarrow$ add noise $\propto r$

Clustering

Clustering

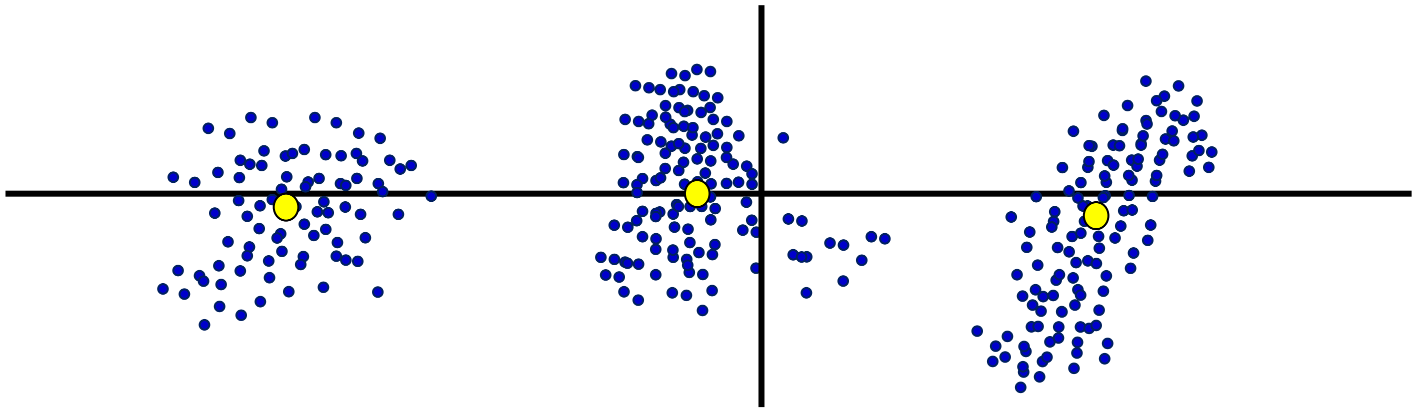
Input: Data points $D = \{x_1, \ldots, x_n\} \in \left(\mathbb{R}^d\right)^n$ and parameter k



Clustering

Input: Data points $D = \{x_1, \dots, x_n\} \in \left(\mathbb{R}^d\right)^n$ and parameter k

Goal: Output centers $C = (c_1, \dots, c_k)$ (e.g., minimize the k -means cost):



Clustering

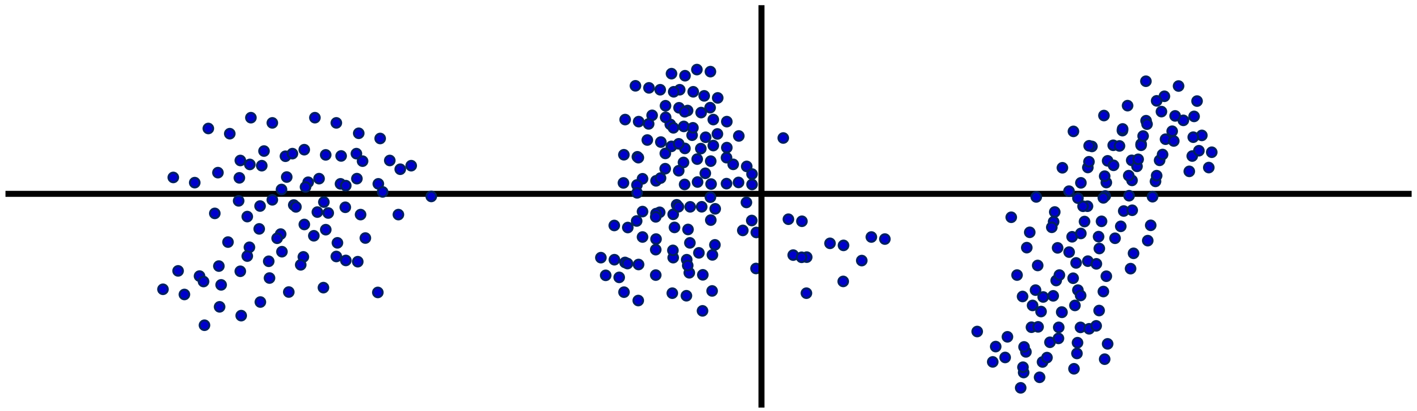
Input: Data points $D = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_n}\} \in (\mathbb{R}^{\boldsymbol{d}})^{\boldsymbol{n}}$ and parameter $\boldsymbol{k}$

Sample and Aggregate:

[Nissim, Raskhodnikova, Smith 07]

(1) Randomly split D into m subsets

(2) Execute some non-private algorithm in each subset.



Clustering

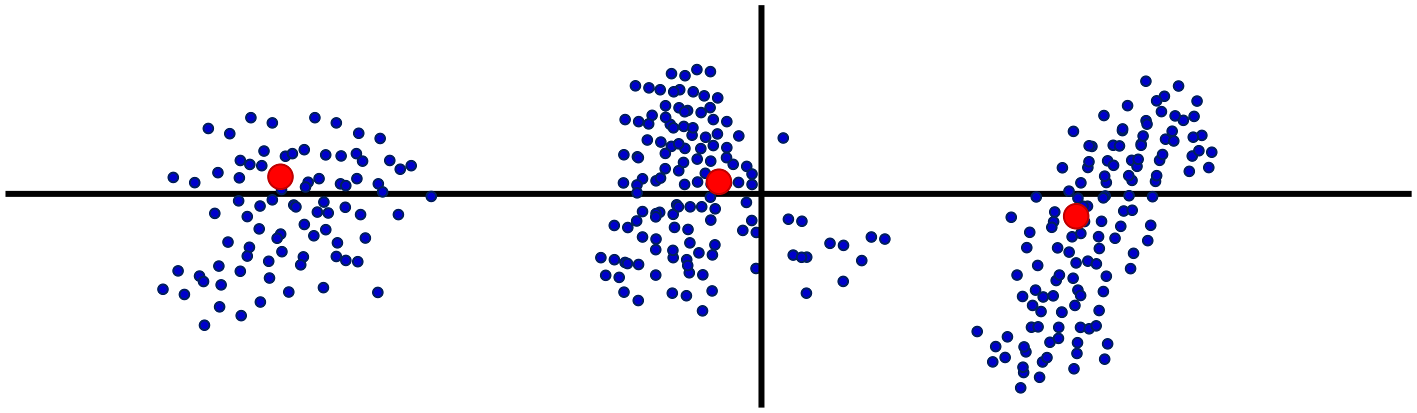
Input: Data points $D = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_n}\} \in \left(\mathbb{R}^{\boldsymbol{d}}\right)^{\boldsymbol{n}}$ and parameter $\boldsymbol{k}$

Sample and Aggregate:

[Nissim, Raskhodnikova, Smith 07]

(1) Randomly split D into m subsets

(2) Execute some non-private algorithm in each subset.



Clustering

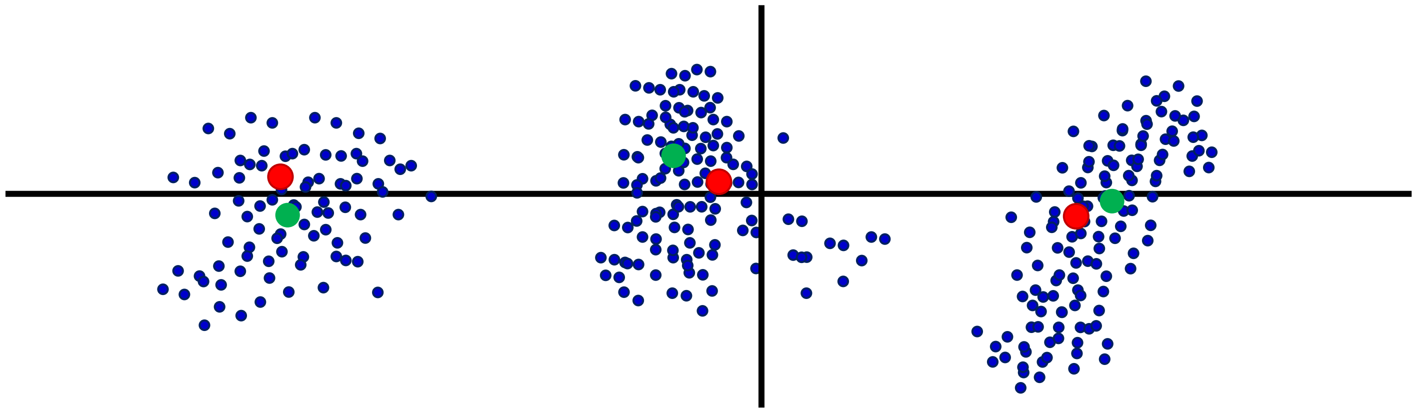
Input: Data points $D = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_n}\} \in \left(\mathbb{R}^{\boldsymbol{d}}\right)^{\boldsymbol{n}}$ and parameter $\boldsymbol{k}$

Sample and Aggregate:

[Nissim, Raskhodnikova, Smith 07]

(1) Randomly split D into m subsets

(2) Execute some non-private algorithm in each subset.



Clustering

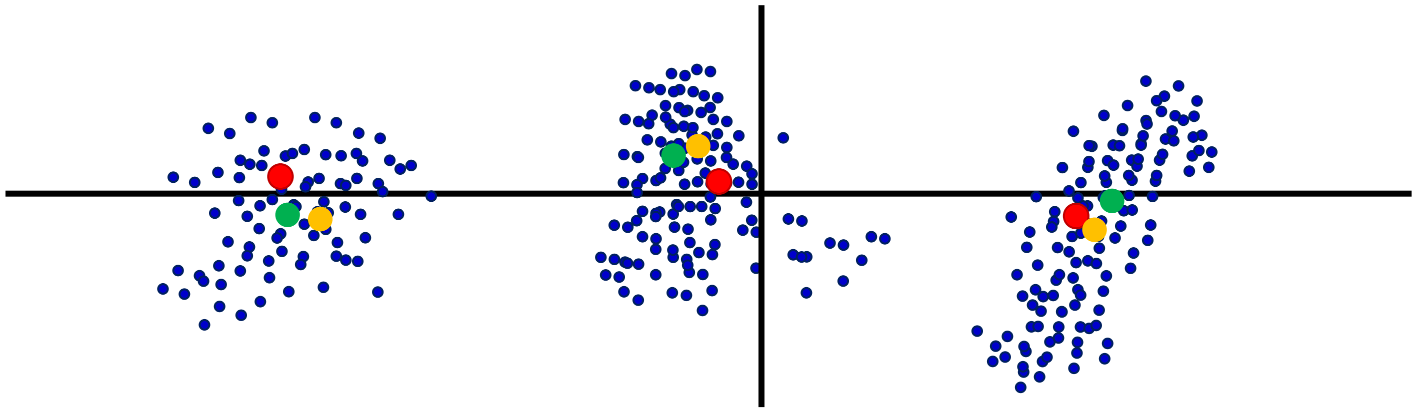
Input: Data points $D = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_n}\} \in (\mathbb{R}^{\boldsymbol{d}})^{\boldsymbol{n}}$ and parameter $\boldsymbol{k}$

Sample and Aggregate:

[Nissim, Raskhodnikova, Smith 07]

(1) Randomly split D into m subsets

(2) Execute some non-private algorithm in each subset.



Clustering

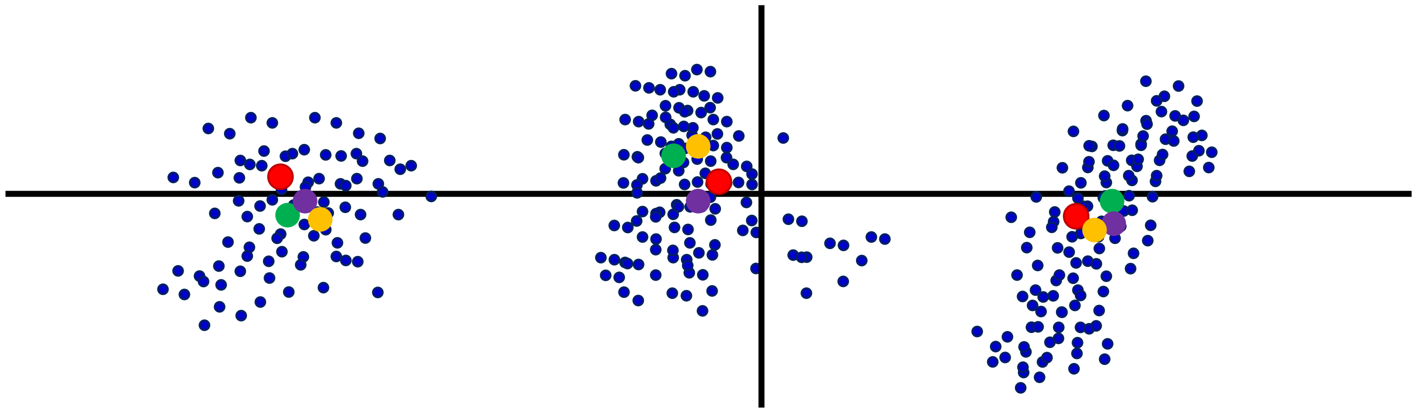
Input: Data points $D = \{\boldsymbol{x_1}, \ldots, \boldsymbol{x_n}\} \in (\mathbb{R}^{\boldsymbol{d}})^{\boldsymbol{n}}$ and parameter $\boldsymbol{k}$

Sample and Aggregate:

[Nissim, Raskhodnikova, Smith 07]

(1) Randomly split D into m subsets

(2) Execute some non-private algorithm in each subset.



Clustering

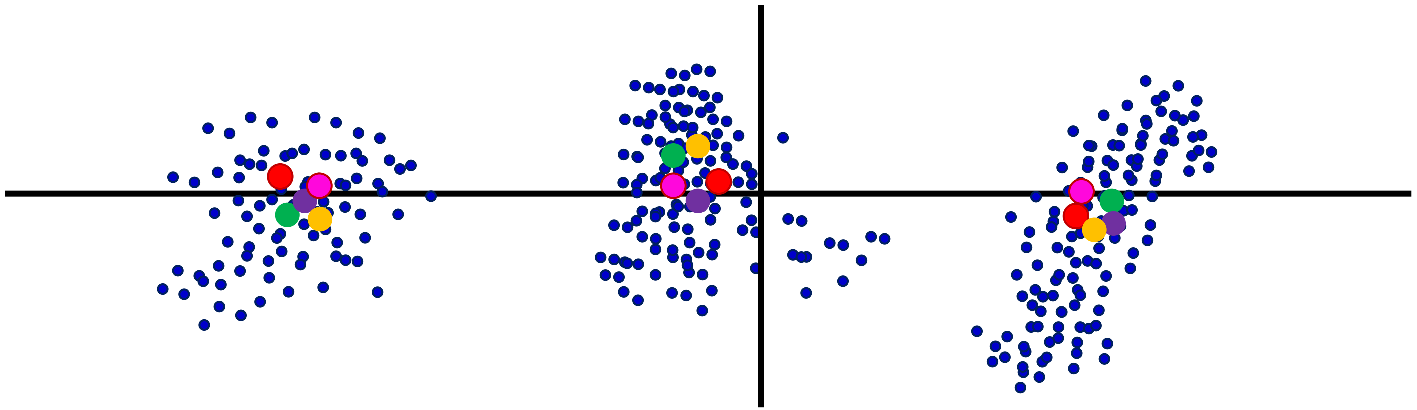
<u>Input</u>: Data points $D = \{x_1, \ldots, x_n\} \in \left(\mathbb{R}^d\right)^n$ and parameter k

<u>Sample and Aggregate</u>:

[Nissim, Raskhodnikova, Smith 07]

(1) Randomly split D into m subsets

(2) Execute some non-private algorithm in each subset.



Clustering

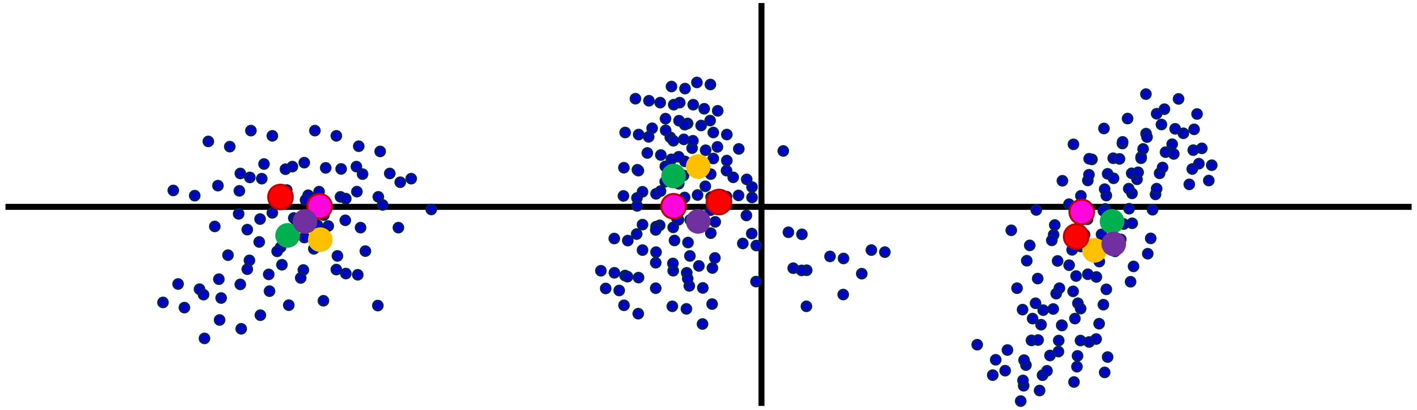
Input: Data points $D = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_n}\} \in \left(\mathbb{R}^{\boldsymbol{d}}\right)^{\boldsymbol{n}}$ and parameter $\boldsymbol{k}$

Sample and Aggregate:

[Nissim, Raskhodnikova, Smith 07]

(1) Randomly split D into m subsets

(2) Execute some non-private algorithm in each subset.



k-tuple Clustering [Cohen et al. 21]

Clustering

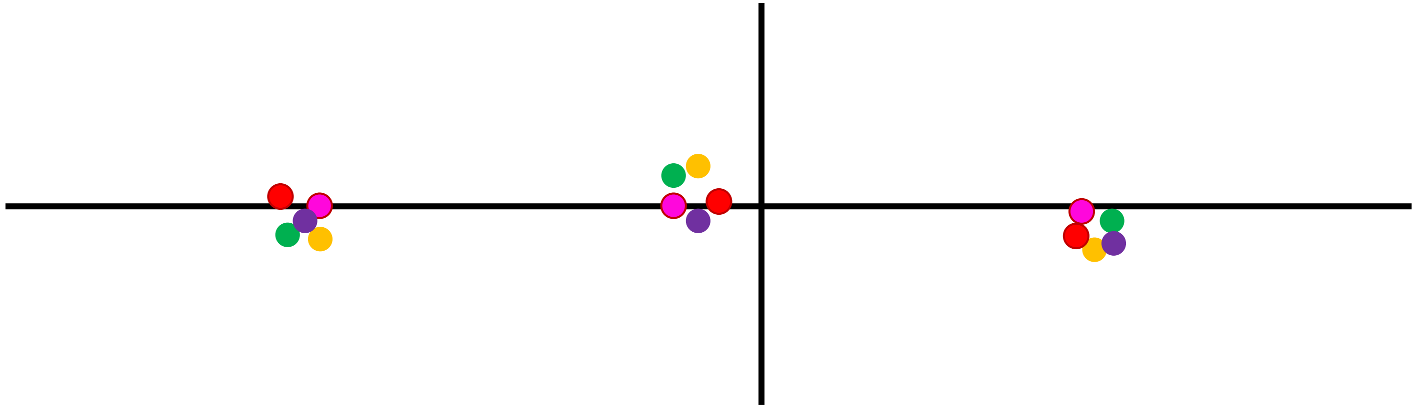
Input: Data points $D = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_n}\} \in \left(\mathbb{R}^{\boldsymbol{d}}\right)^{\boldsymbol{n}}$ and parameter $\boldsymbol{k}$

Sample and Aggregate:

[Nissim, Raskhodnikova, Smith 07]

(1) Randomly split D into m subsets

(2) Execute some non-private algorithm in each subset.



k-tuple Clustering [Cohen et al. 21]

Clustering

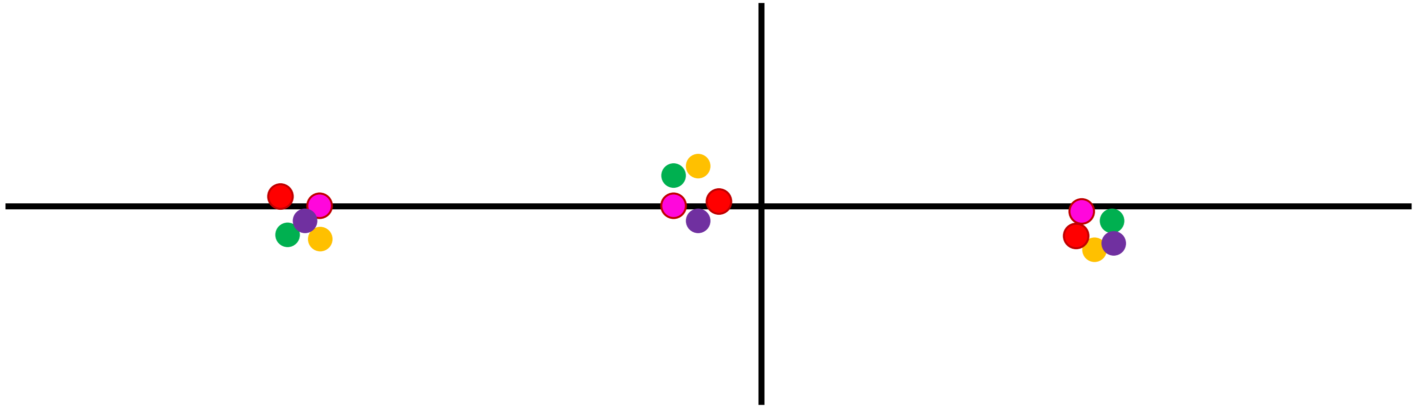
Input: Data points $D = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_n}\} \in (\mathbb{R}^d)^n$ and parameter $\boldsymbol{k}$

Sample and Aggregate:

[Nissim, Raskhodnikova, Smith 07]

(1) Randomly split D into m subsets

(2) Execute some non-private algorithm in each subset.



k-tuple Clustering [Cohen et al. 21]

Input: *unordered* k-tuples $\{\boldsymbol{Y_1}, \dots, \boldsymbol{Y_m}\} \in \left((\mathbb{R}^d)^k\right)^m$

Goal: Privately identify a new k-tuple that is ``close'' to them.

Clustering

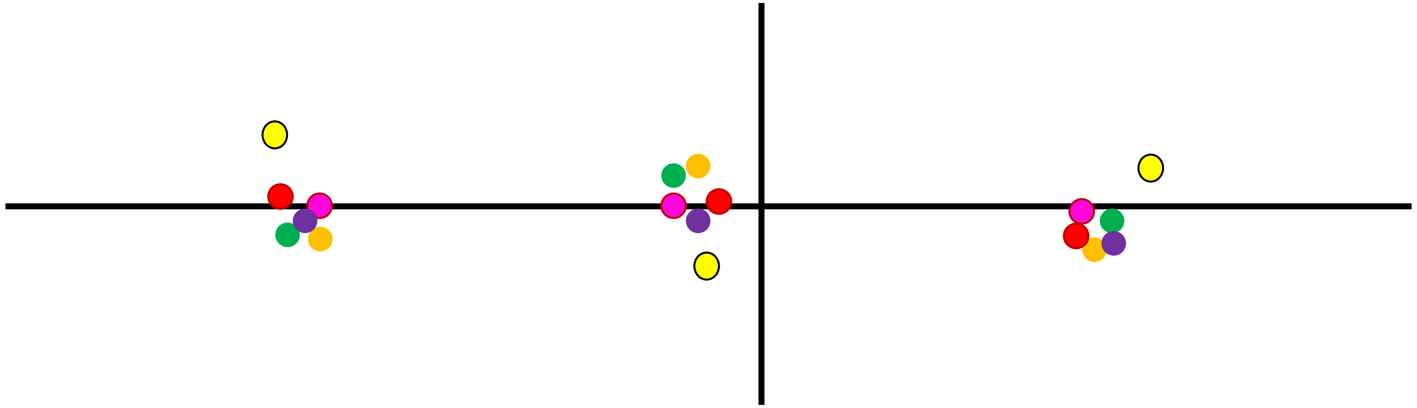
Input: Data points $D = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_n}\} \in \left(\mathbb{R}^{\boldsymbol{d}}\right)^{\boldsymbol{n}}$ and parameter $\boldsymbol{k}$

Sample and Aggregate:

[Nissim, Raskhodnikova, Smith 07]

(1) Randomly split D into m subsets

(2) Execute some non-private algorithm in each subset.



k-tuple Clustering [Cohen et al. 21]

Input: *unordered* k-tuples $\{\boldsymbol{Y_1}, \dots, \boldsymbol{Y_m}\} \in \left(\left(\mathbb{R}^{\boldsymbol{d}}\right)^{\boldsymbol{k}}\right)^{\boldsymbol{m}}$

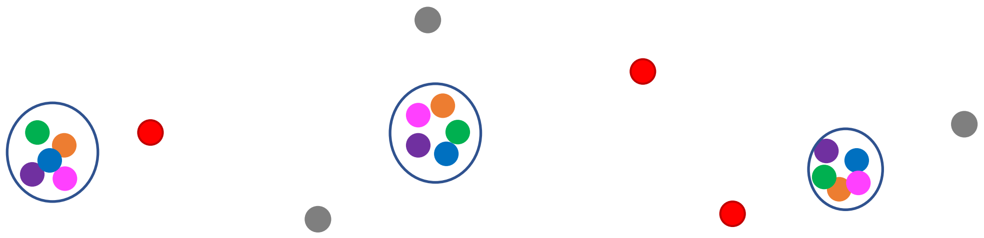
Goal: Privately identify a new k-tuple that is ``close'' to them.

FriendlyCore Paradigm

Input:
- Dataset D of k-tuples

Operation:

$C \leftarrow \text{FriendlyCore}(D) \qquad (C \subseteq D)$

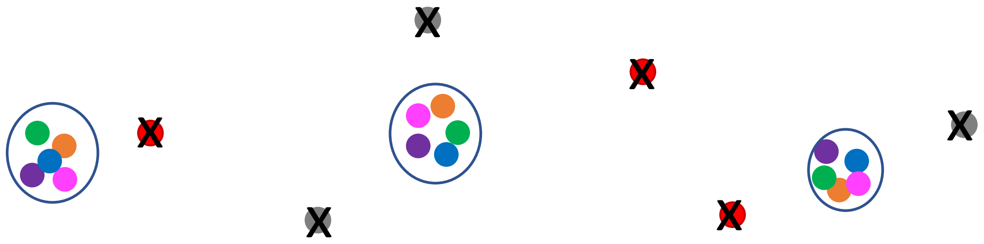


FriendlyCore Paradigm

Input:
- Dataset D of k-tuples

Operation:

$C \leftarrow \text{FriendlyCore}(D) \quad (C \subseteq D)$



friendly $C \Rightarrow$ tuples close to each other

FriendlyCore Paradigm

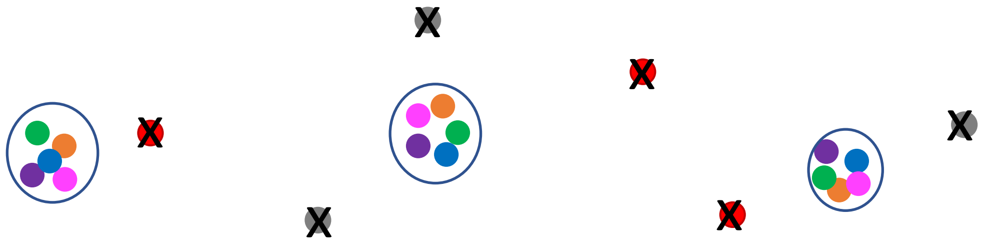
Input:

- Dataset D of k-tuples

Operation:

$C \leftarrow \text{FriendlyCore}(D) \quad (C \subseteq D)$

Output: $A(C)$ (*friendly* DP A)



friendly C ⇒ tuples close to each other

FriendlyCore Paradigm

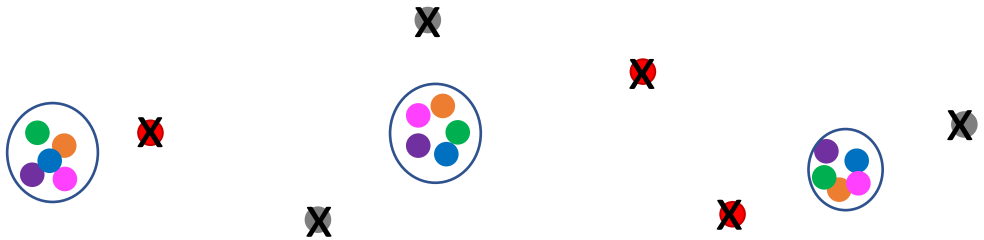
Input:

- Dataset D of k-tuples

Operation:

$C \leftarrow \text{FriendlyCore}(D) \qquad (C \subseteq D)$

Output: $A(C)$ (*friendly* DP A)



friendly $C \Rightarrow$ tuples close to each other

friendly DP $A \Rightarrow$ very simple clustering

FriendlyCore

- Simple, Generic and Practical Algorithm

FriendlyCore

- Simple, Generic and Practical Algorithm

Utility:

- If all elements in D are “close” to each other:

$$\text{FriendlyCore}(D) = D$$

FriendlyCore

- Simple, Generic and Practical Algorithm

Utility:

- If all elements in D are "close" to each other:

$$\mathrm{FriendlyCore}(D) = D$$

Privacy:

- If A is *friendly* (ε, δ)-DP, then
  $A\big(\mathrm{FriendlyCore}(\cdot)\big)$ is $\approx (2\varepsilon, 2e^{3\varepsilon}\delta)$-DP

FriendlyCore

- Simple, Generic and Practical Algorithm

Utility:

- If all elements in D are “close” to each other:

$$\mathrm{FriendlyCore}(D) = D$$

Privacy:

- If A is *friendly* (ε, δ)-DP, then
  $A\big(\mathrm{FriendlyCore}(\cdot)\big)$ is $\approx (2\varepsilon, 2e^{3\varepsilon}\delta)$-DP

  - Also, zCDP version

Summary

Summary

- FriendlyCore: tool for private aggregation tasks
  - ➢ Example Applications: averaging (optimal asymptotic) and clustering
    Also, learning *unrestricted* covariance matrix of a Gaussian.

Summary

- FriendlyCore: tool for private aggregation tasks
  - ➢ Example Applications: averaging (optimal asymptotic) and clustering
    Also, learning *unrestricted* covariance matrix of a Gaussian.

- Empirical evaluations:
  - Averaging: Comparison with CoinPress [Biswas et al. 20]
  - Clustering: Comparison with [Chang Kamath 21]

Summary

- FriendlyCore: tool for private aggregation tasks
  - ➢ Example Applications: averaging (optimal asymptotic) and clustering

    Also, learning *unrestricted* covariance matrix of a Gaussian.

- Empirical evaluations:
  - Averaging: Comparison with CoinPress [Biswas et al. 20]
  - Clustering: Comparison with [Chang Kamath 21]

Thank you!