Differentially Private Histograms for Range-Sum Queries: A Modular Approach

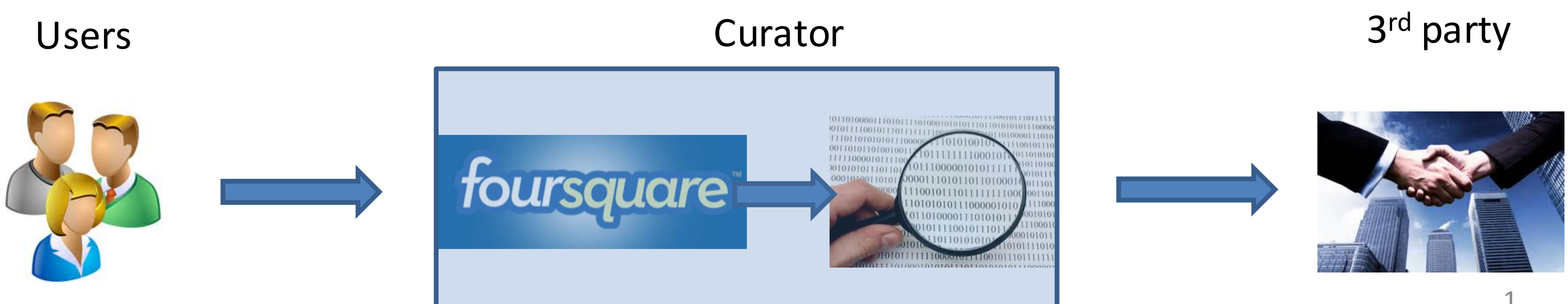
Georgios Kellaris

Stavros Papadopoulos

Dimitris Papadias

Privacy-preserving data publishing

- A **curator** (e.g., a company, a hospital, an institution, etc.) gathers data about its **users**
- **Third-parties** (e.g., research labs, advertising agencies, etc.) wish to learn **statistical facts** about the collected data
- How can the curator release **useful** statistics *while* preserving **user privacy**?



ϵ-differential privacy [Dwork, ICALP'06]

- Publishing statistics can reveal potentially private information

ϵ-differential privacy [Dwork, ICALP'06]

- Publishing statistics can reveal potentially private information

D ⟵ the database is hidden

Name	Age	HIV+
Alice	42	Yes
Bob	31	Yes
Carol	32	Yes
Dave	36	No
Ellen	43	Yes
Frank	41	Yes
Grace	26	Yes
...	...	...

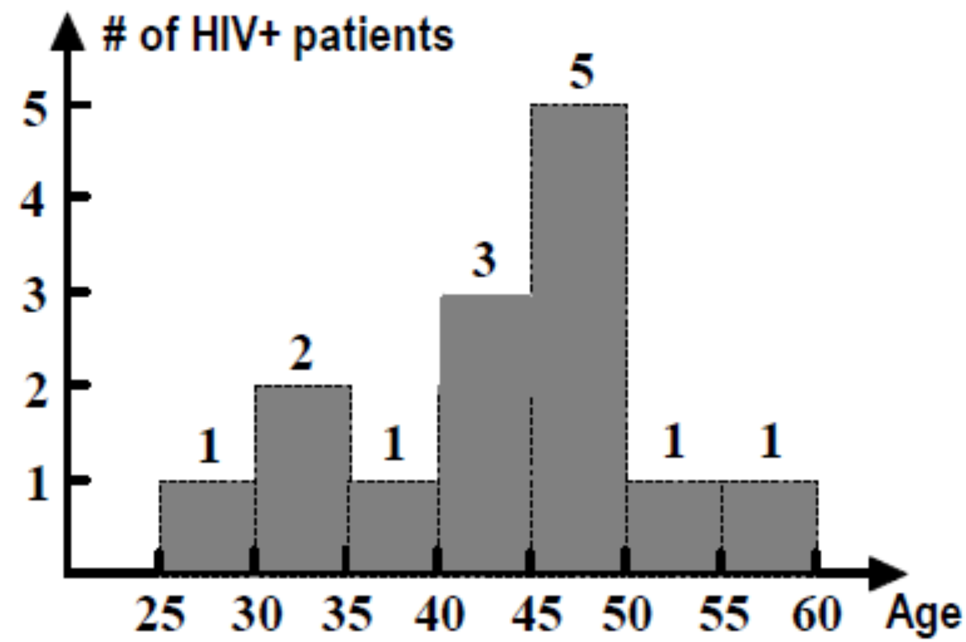
Curator

ϵ-differential privacy [Dwork, ICALP'06]

- Publishing statistics can reveal potentially private information

D ← the database is hidden

Name	Age	HIV+
Alice	42	Yes
Bob	31	Yes
Carol	32	Yes
Dave	36	No
Ellen	43	Yes
Frank	41	Yes
Grace	26	Yes
...	...	...



Curator

ϵ-differential privacy [Dwork, ICALP'06]

- Publishing statistics can reveal potentially private information

D'

Name	Age	HIV+
Alice	[redacted]	
Bob	31	Yes
Carol	32	Yes
Dave	36	No
Ellen	43	Yes
Frank	41	Yes
Grace	26	Yes
...	...	...

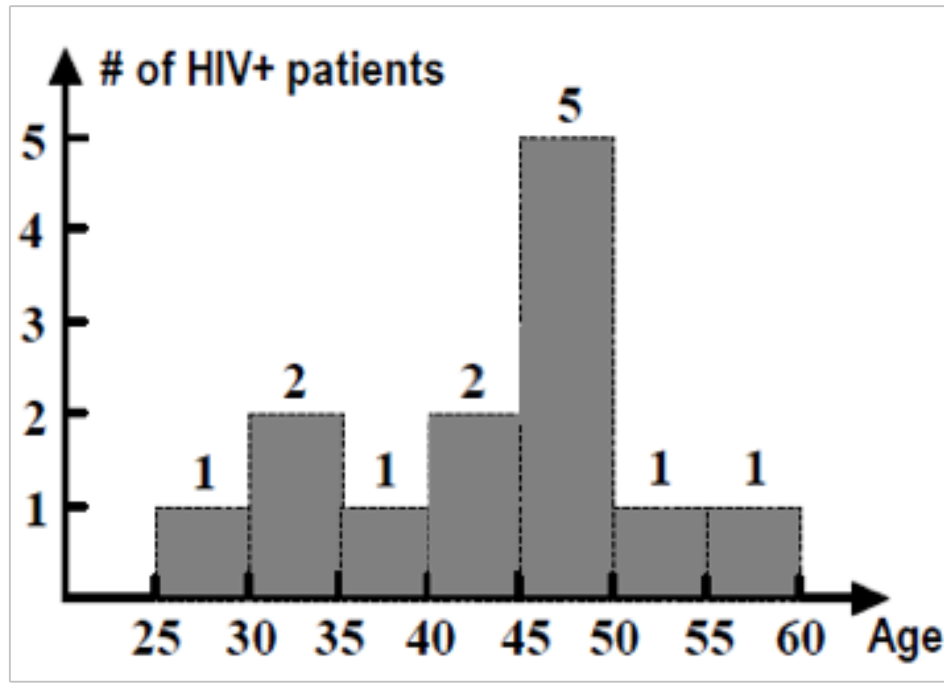
Before publishing, the adversary happens to know everything, except for *Alice*

ϵ-differential privacy [Dwork, ICALP'06]

- Publishing statistics can reveal potentially private information

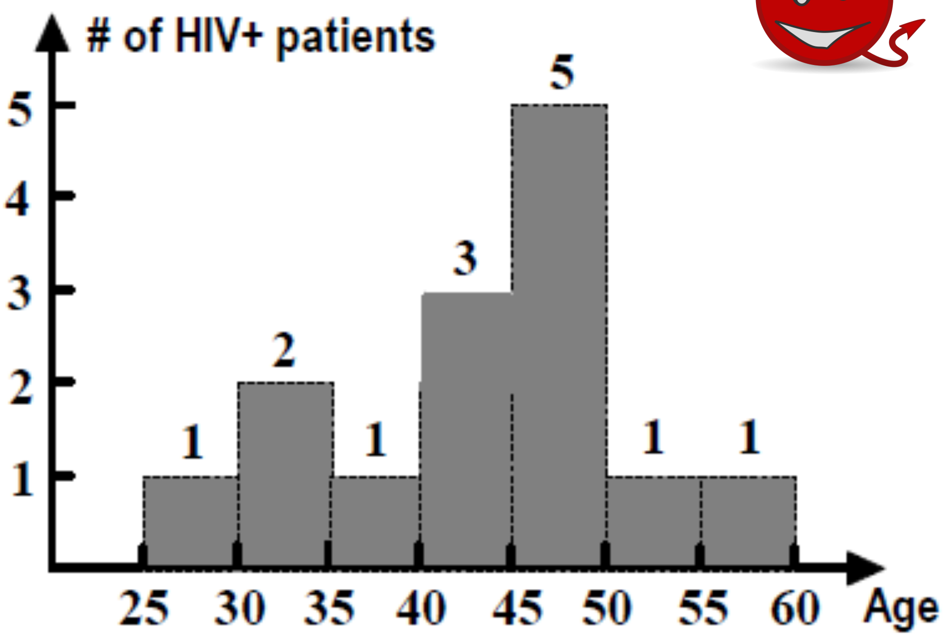
D'

Name	Age	HIV+
Alice	[redacted]	
Bob	31	Yes
Carol	32	Yes
Dave	36	No
Ellen	43	Yes
Frank	41	Yes
Grace	26	Yes
...	...	...

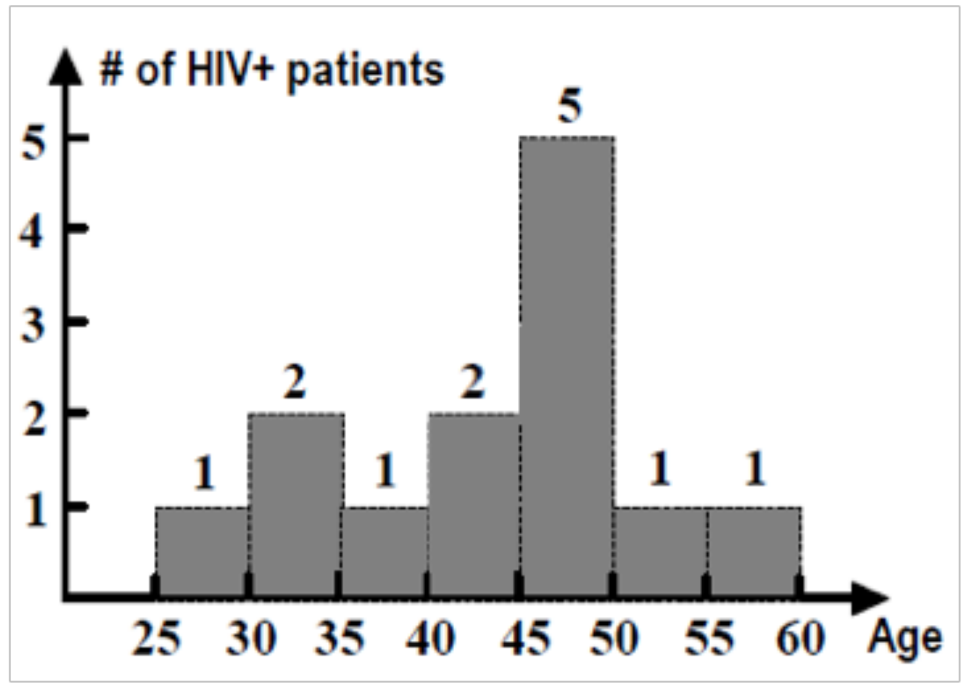


ϵ-differential privacy [Dwork, ICALP'06]

- Publishing statistics can reveal potentially private information



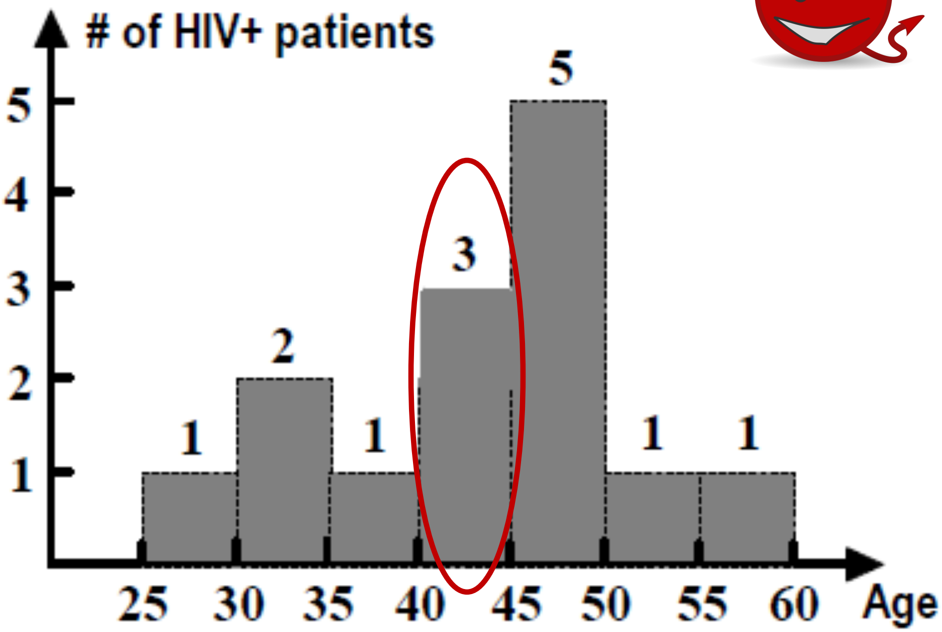
Published data



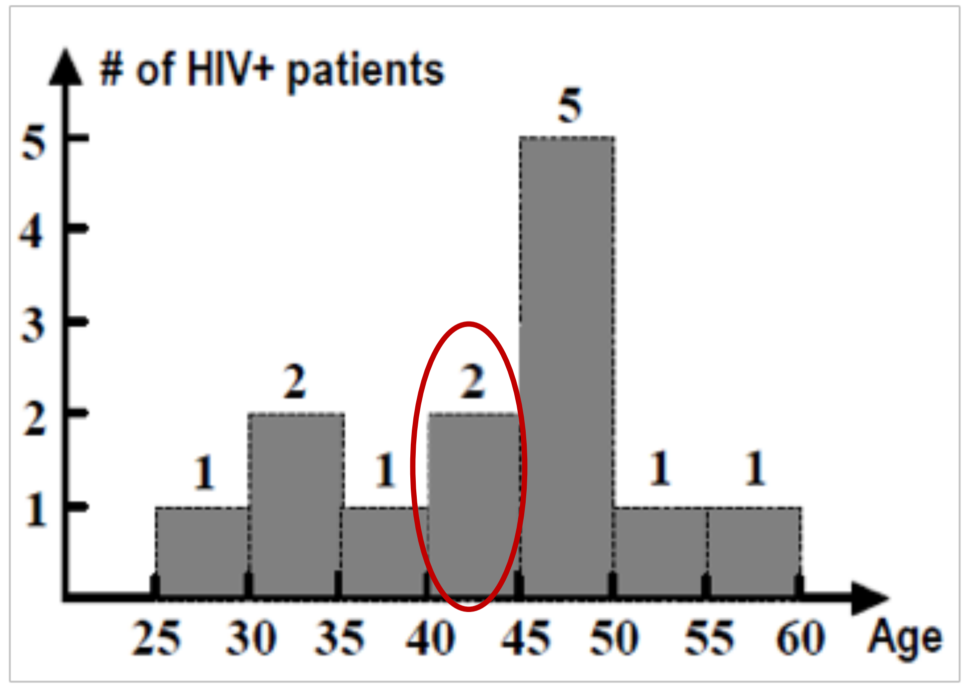
Adversary's view

ϵ-differential privacy [Dwork, ICALP'06]

- Publishing statistics can reveal potentially private information



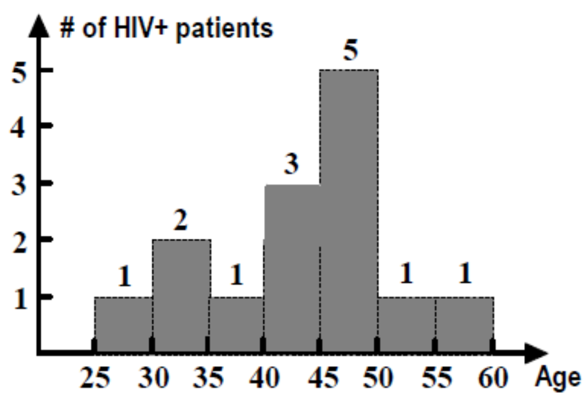
Published data



Adversary's view

ϵ-differential privacy [Dwork, ICALP'06]

- Main idea
  - Randomized Mechanism: Hide the presence of any user, by hiding the effect he has on the published statistics

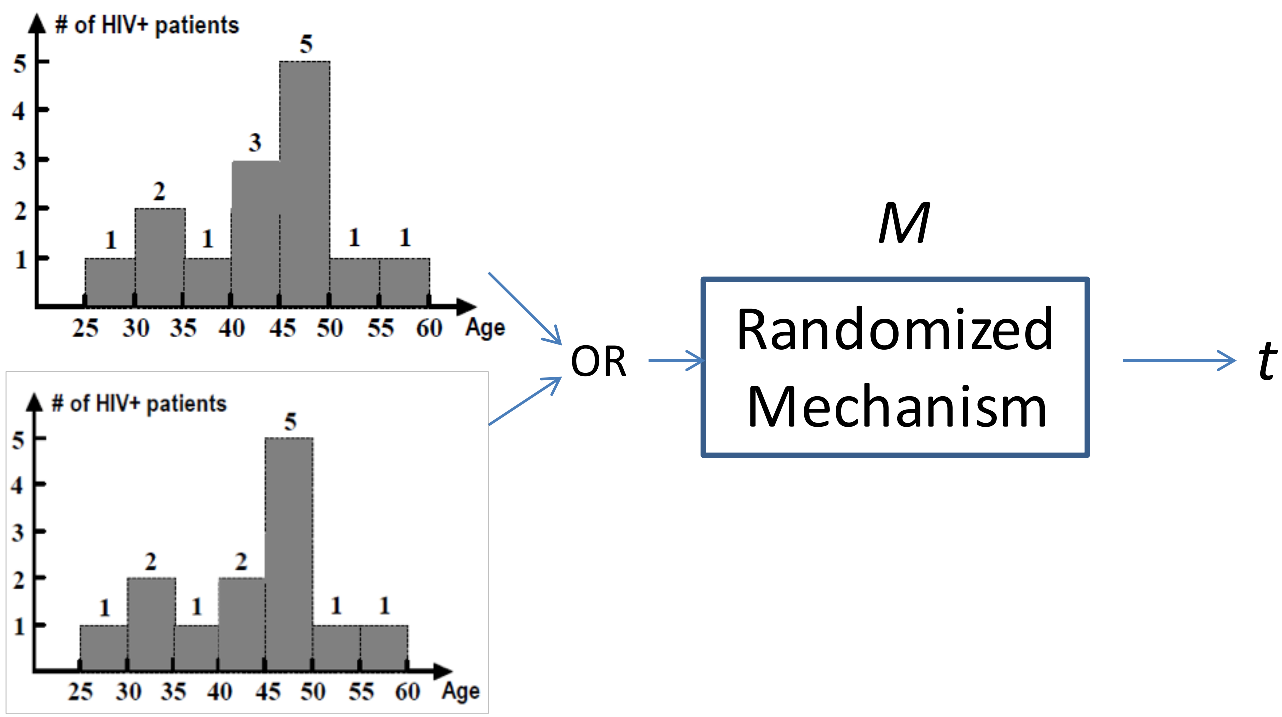


M

Randomized Mechanism → t

ϵ-differential privacy [Dwork, ICALP'06]

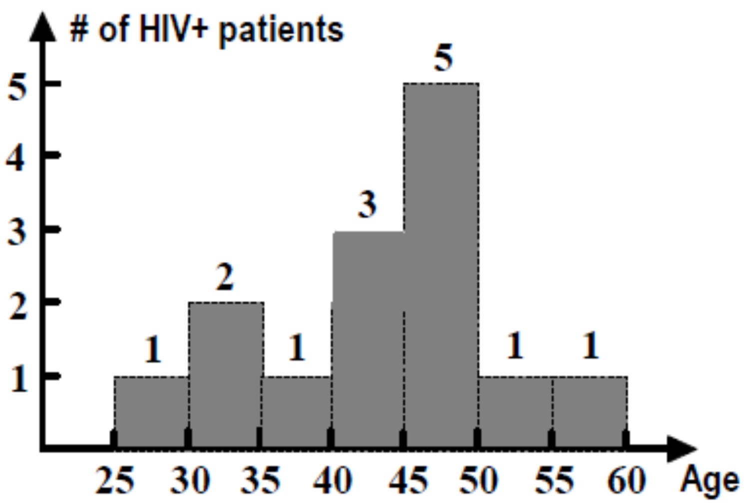
- Main idea
  - Any output (called *transcript*) of *M* is produced with almost the *same* probability, whether any single user was in the database (*D*) or not (*D*')

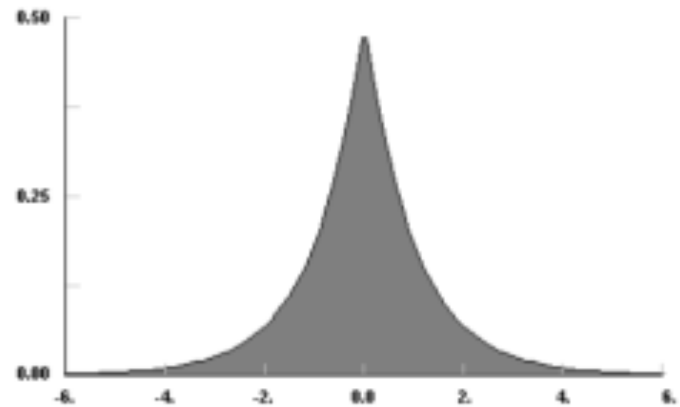


Laplace Perturbation Algorithm (LPA)

[Dwork et al., TCC'06]

- Add noise drawn from the Laplace distribution with mean 0

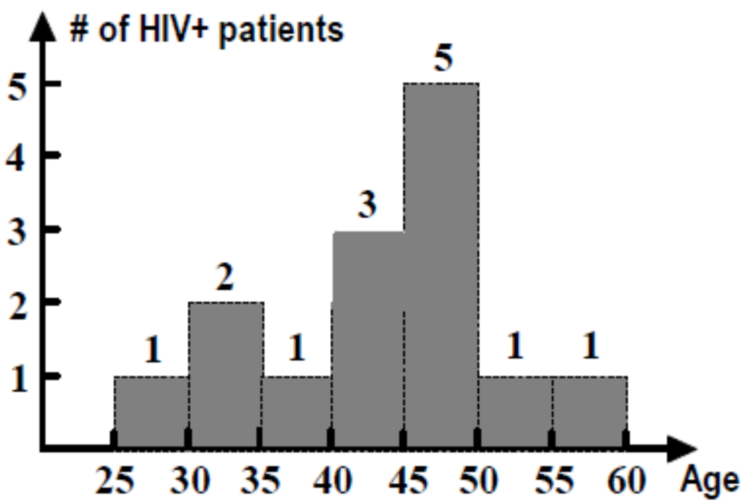


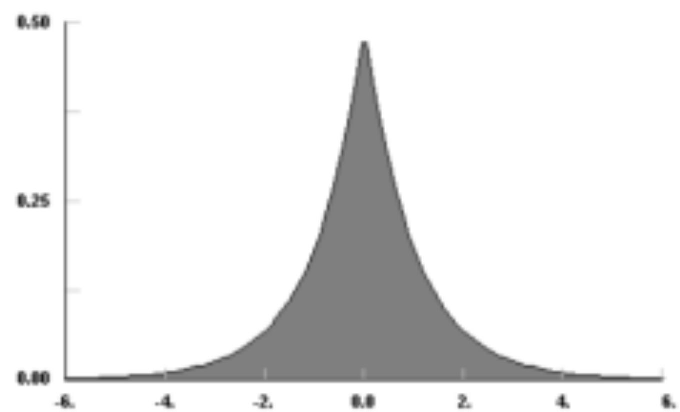


Laplace Perturbation Algorithm (LPA)

[Dwork et al., TCC'06]

- Add noise drawn from the Laplace distribution with mean 0

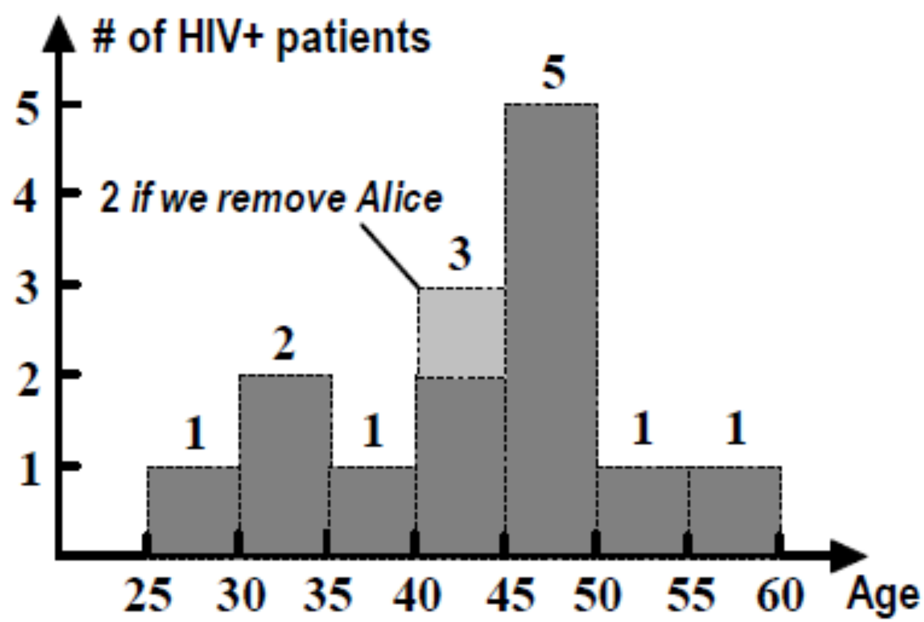




How much noise?

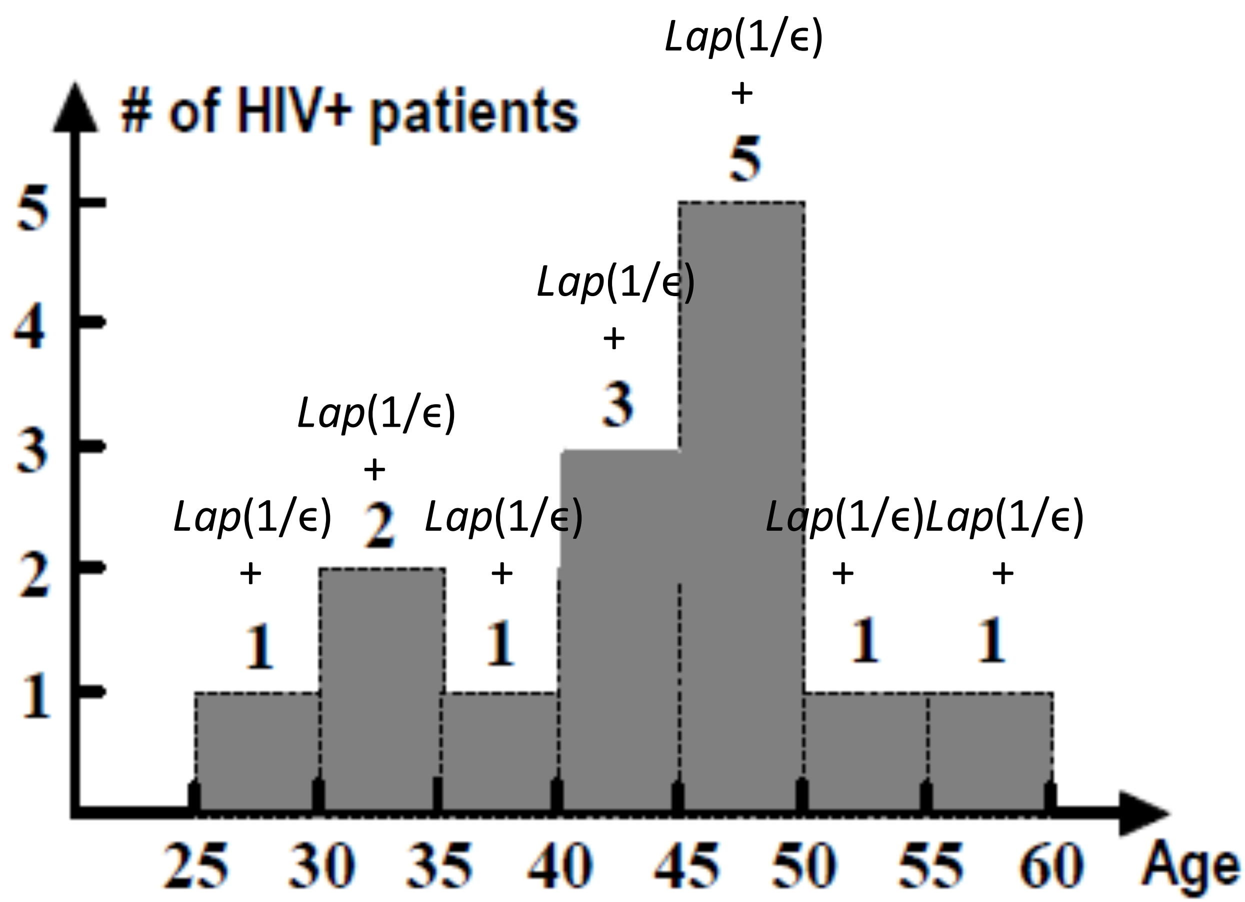
Laplace Perturbation Algorithm (LPA)
[Dwork et al., TCC'06]

- The *scale* of the distribution depends on the sensitivity *Δ*
  - *Δ*: **maximum** amount of statistical information that can be affected by *any single* user
    - I.e. how much the statistics will change if we remove any single user



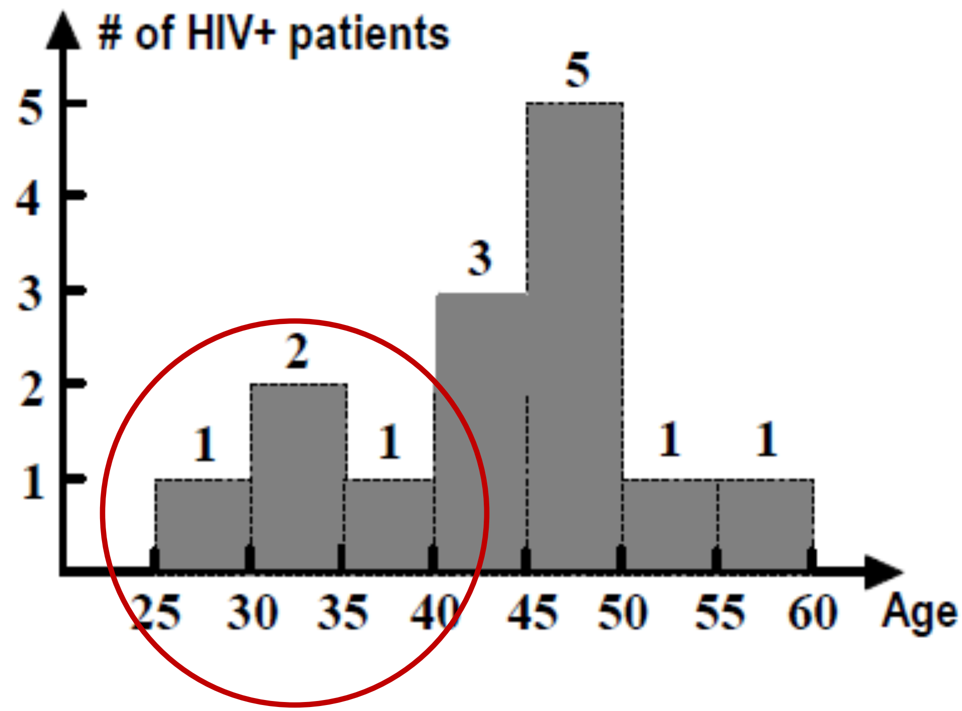
Laplace Perturbation Algorithm (LPA)

[Dwork et al., TCC'06]



Setting: Range queries on histograms

Name	Age	HIV+
Alice	42	Yes
Bob	31	Yes
Carol	32	Yes
Dave	36	No
Ellen	43	Yes
Frank	41	Yes
Grace	26	Yes
...	...	...



- Range query:
  - Give me the number of HIV+ patients with age range 25-40

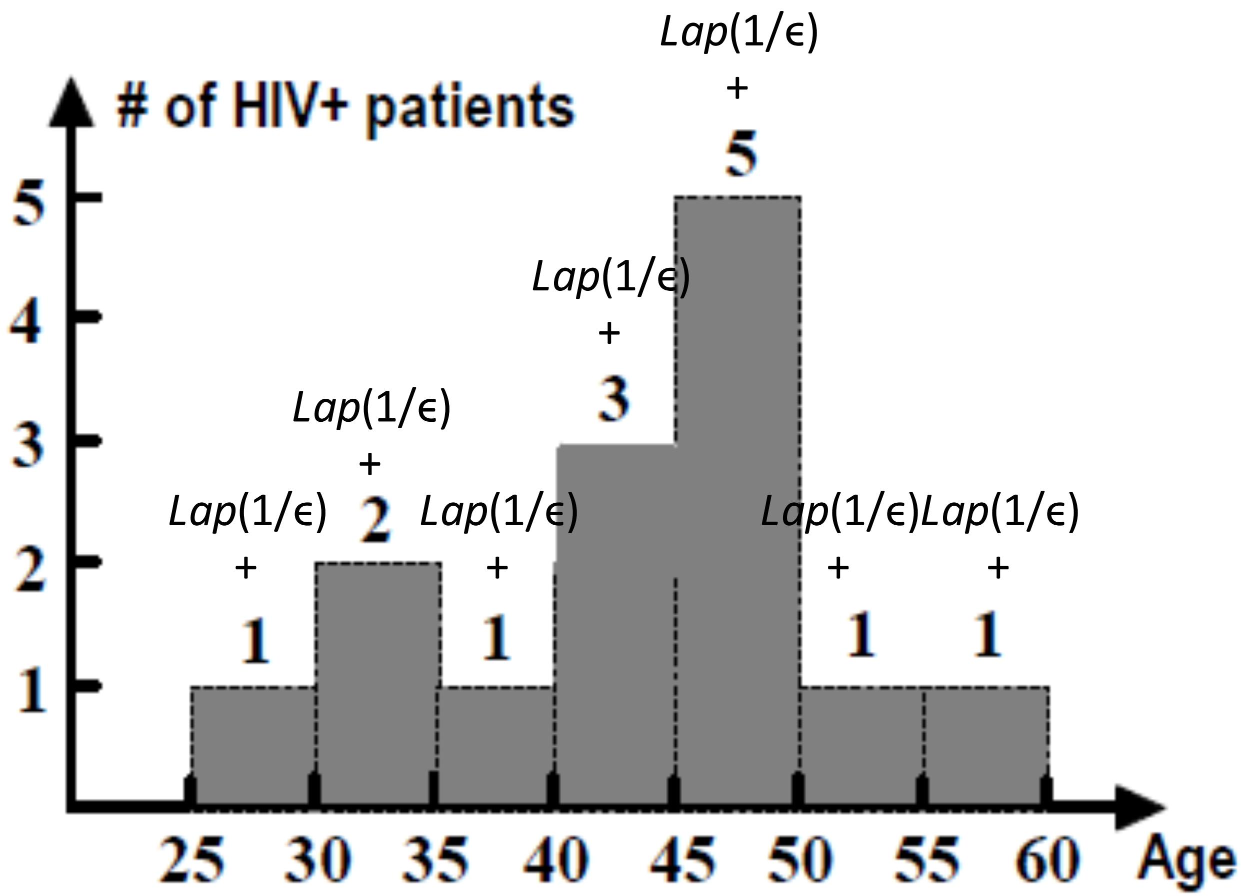
Problem definition

- Publish a *differentially private* histogram
- Any *range* query (<u>not</u> known a priori) on the released histogram should get an answer **close** to the real one
- Focus on both *accuracy* and *time efficiency*

Laplace Perturbation Algorithm (LPA)

Pros

- Each output bin value very close to the original (small error per bin)
- Very fast (O(*n*))



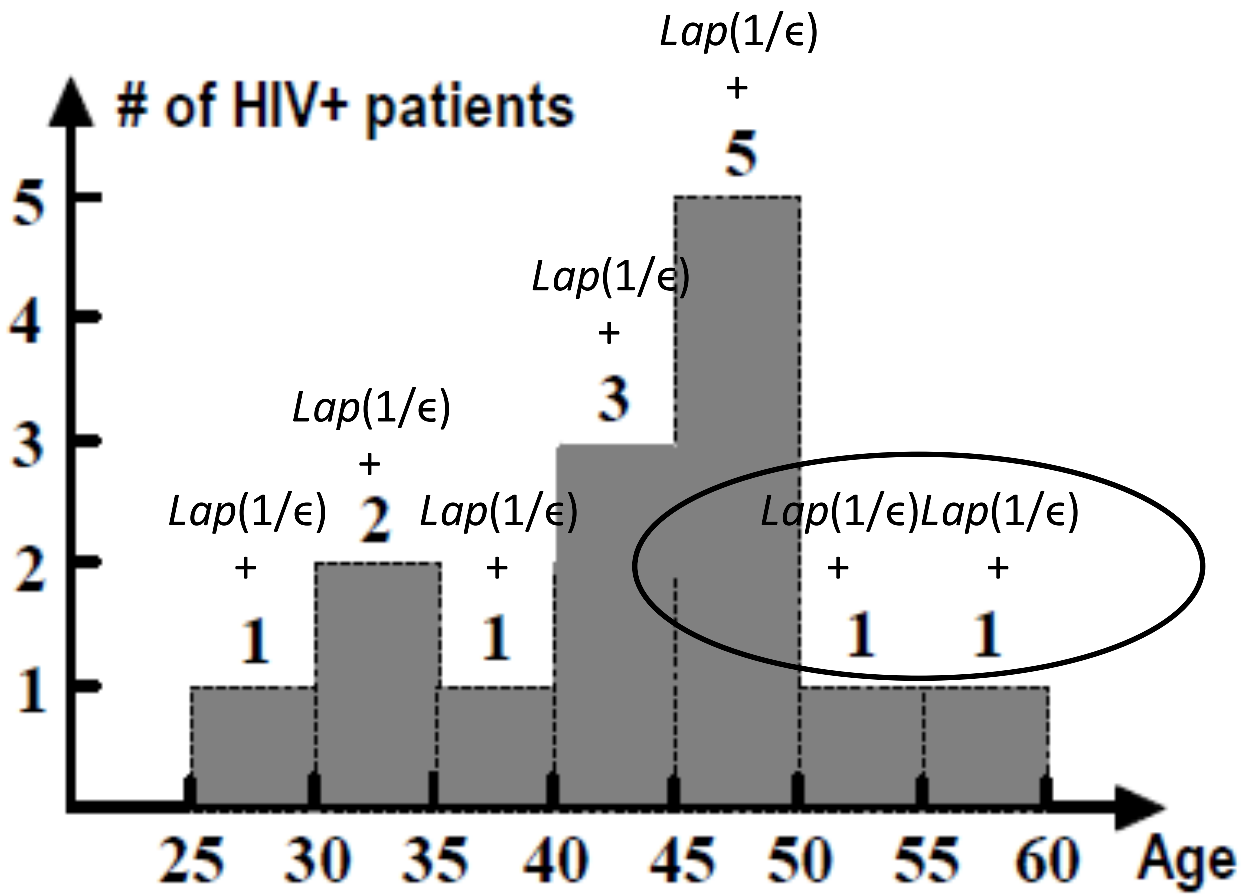
Laplace Perturbation Algorithm (LPA)

<u>Pros</u>

- Each output bin value very close to the original (small error per bin)
- Very fast (O(*n*))

<u>Cons</u>

- When computing ranges, the error due to noise accumulates (error proportional to range size)



Related Work

Data Aware

Take advantage of the distribution of the bin values

Data Oblivious

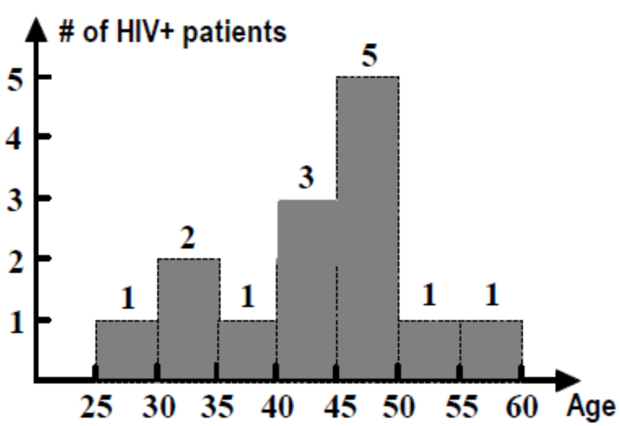
Oblivious to the bin values

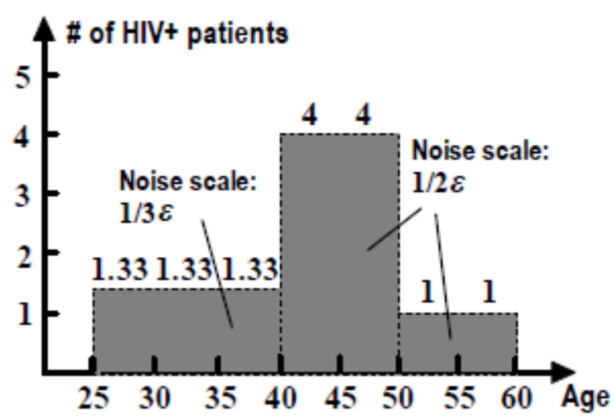
Related Work

Related Work

Group and average consecutive bins before the LPA

- Reduces the sensitivity of the grouped bins
- Reduces the required noise
- Introduces approximation error





Related Work

Data Aware → Smoothing

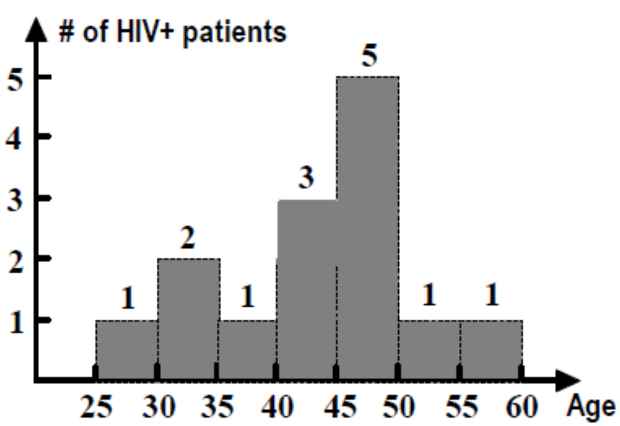
Data Oblivious

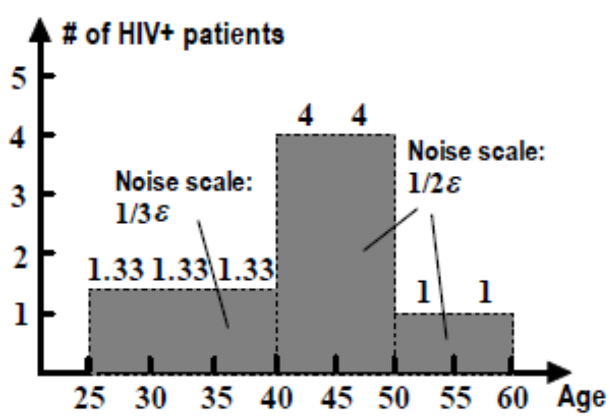
Group and average consecutive bins before the LPA

- Reduces the sensitivity of the grouped bins
- Reduces the required noise
- Introduces approximation error

Find the best way to merge the bins

- Explore all possible groups (count: $O(n^2)$)
- Choose the groups that minimize the **total error** of the *approximation* and the *noise addition*

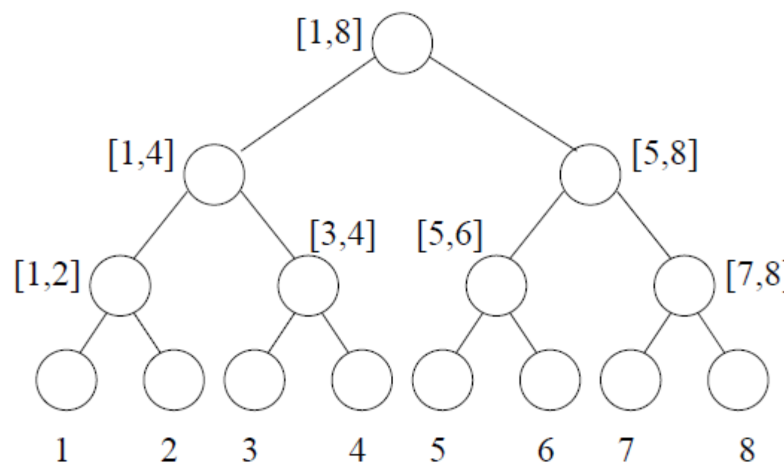




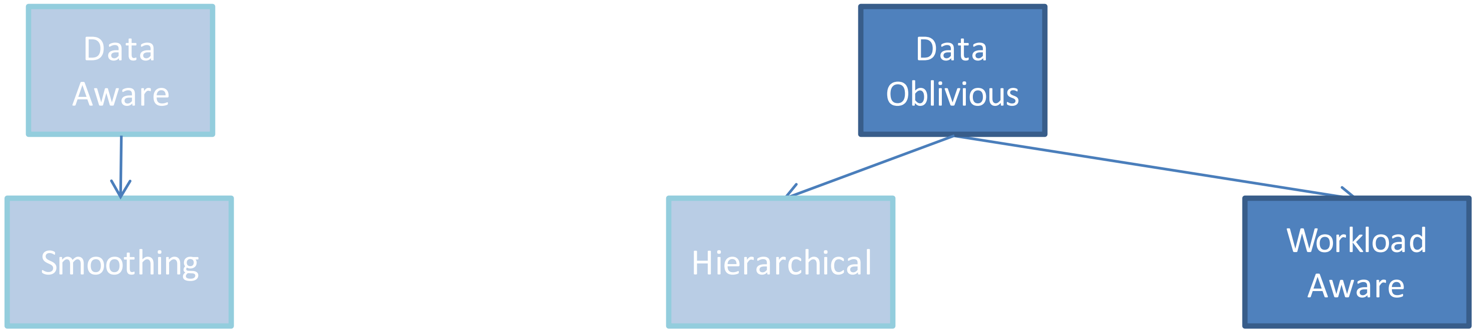
Related Work

Build an aggregate tree over the bins – each node holds the sum of its children

- Sensitivity: log*n*
- Compute range using the sub-trees that contain the query



Related Work



Given the range queries

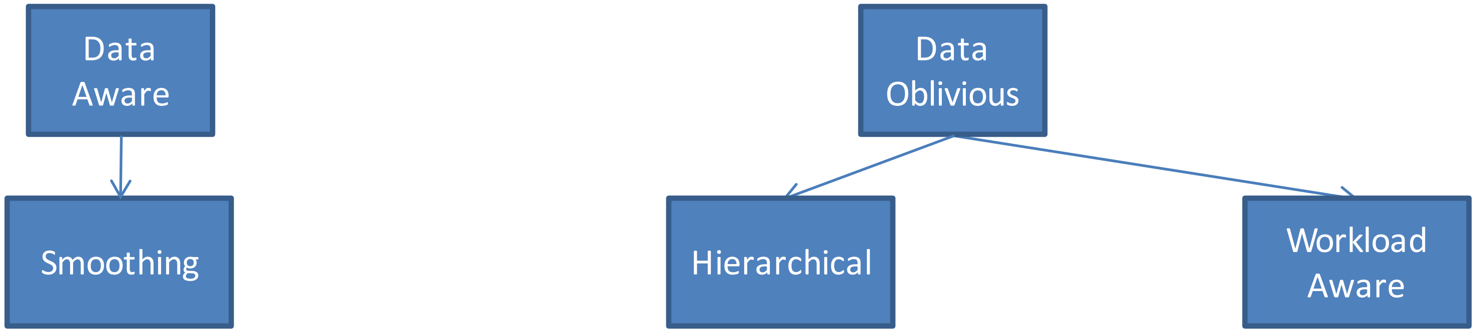
- Add noise with different scale to each query answer
- Combine query answers that have overlaps to get more accurate results

Applies to our setting by fixing all possible range queries

Modular Approach: Motivation

- Every method can be decomposed into primitive components/modules
- Benefits
  - Better *understanding* of each technique
  - Easy to *discover* performance **bottlenecks** and apply optimizations
  - Easy to *combine* different **components** to *design* **new methods** that benefit from the merits of different approaches

Modules


Data Aware
Smoothing
Data Oblivious
Hierarchical
Workload Aware

Modules

Smoothing

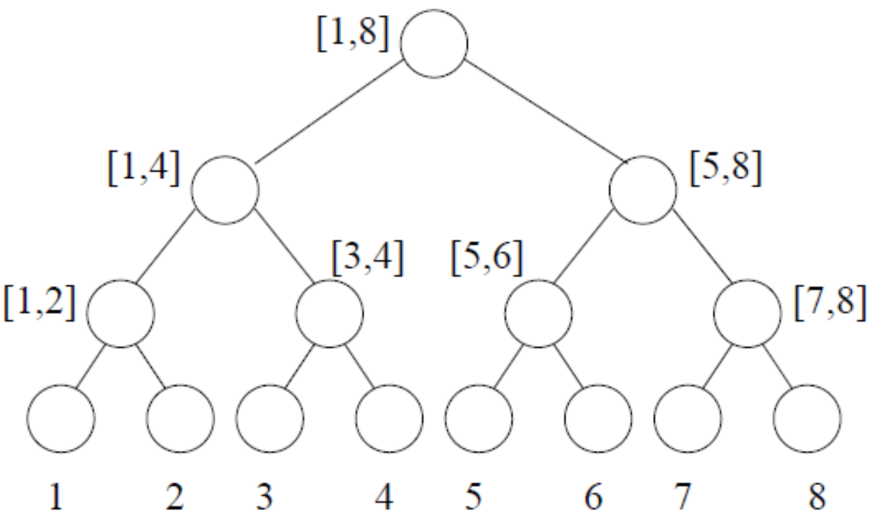
Hierarchical

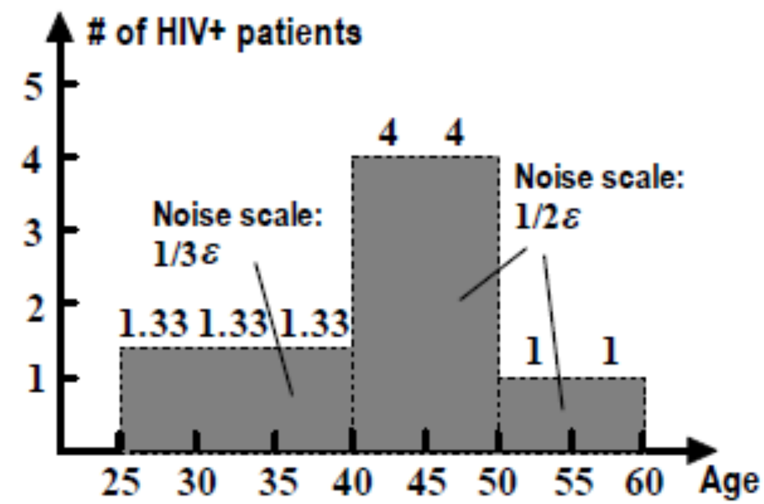
Workload Aware

Every existing method can be reproduced from these modules by parameterizing them

New Scheme: Subtree Smoothing

- Combine the merits of Hierarchical and Smoothing

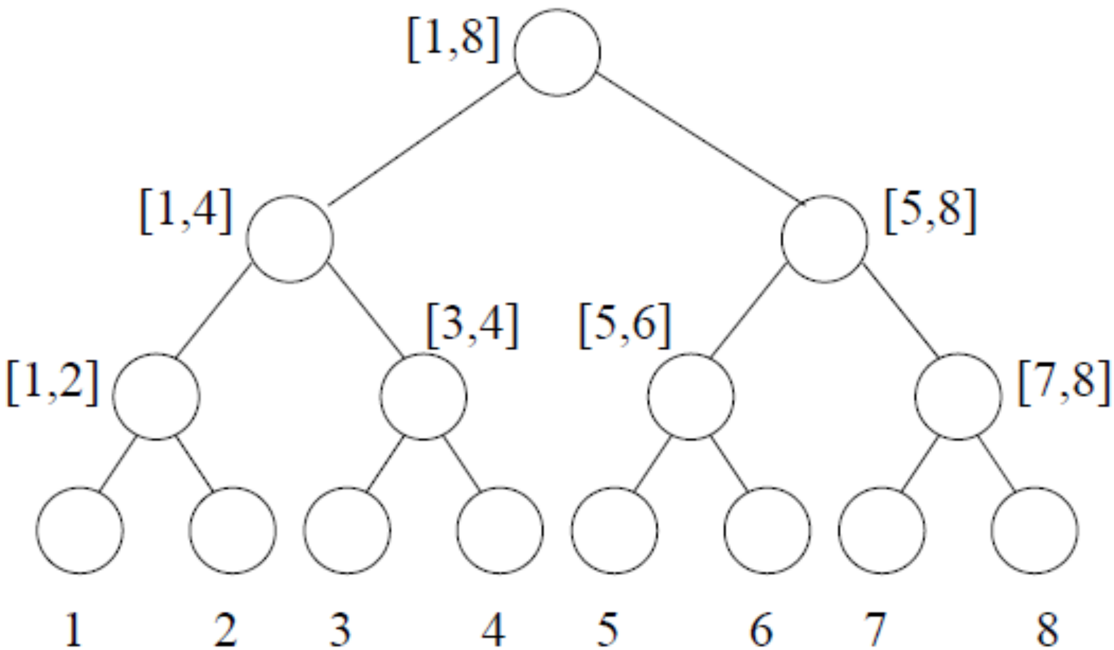




Fast (O(n)); accurate for large ranges

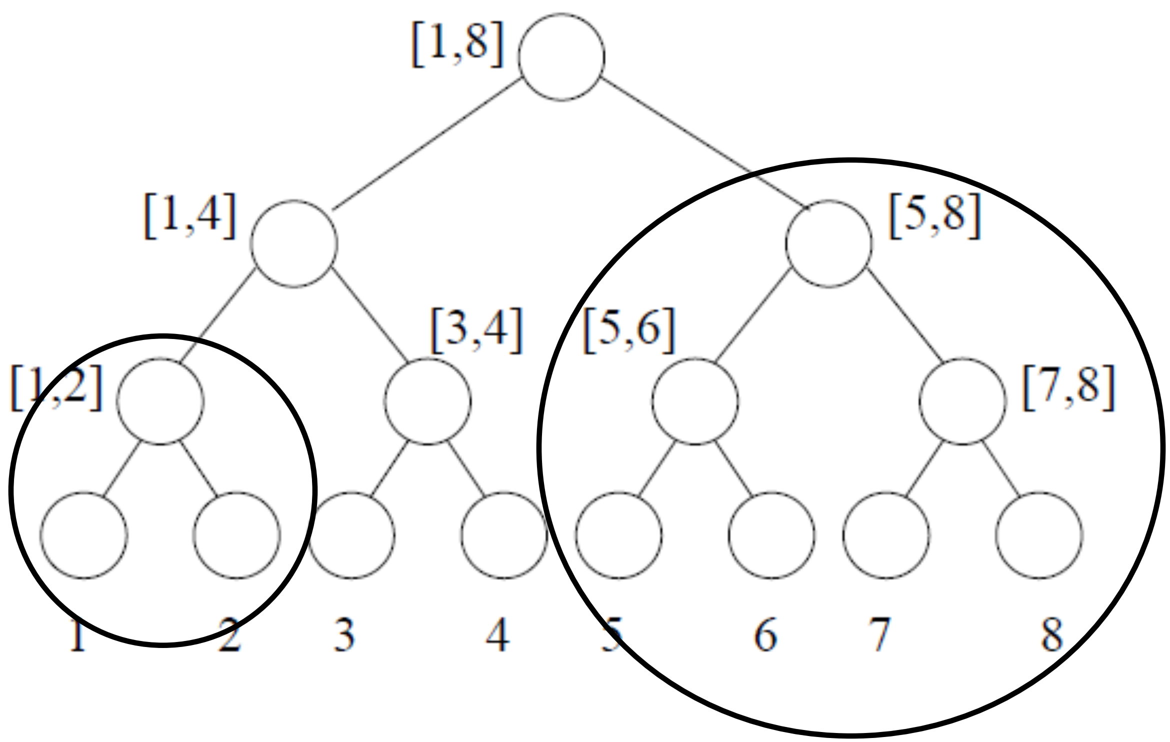
Slow (O(n^2)); accurate for small ranges

New Scheme: Subtree Smoothing



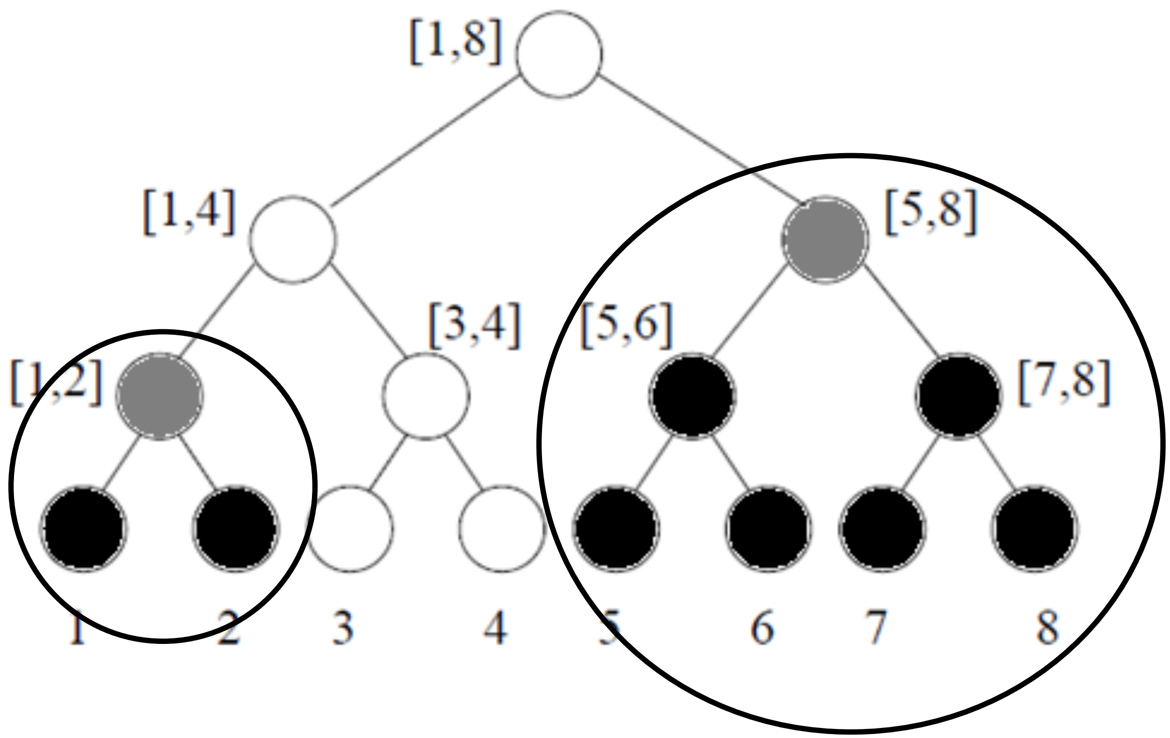
New Scheme: Subtree Smoothing

- Prune subtrees with similar values

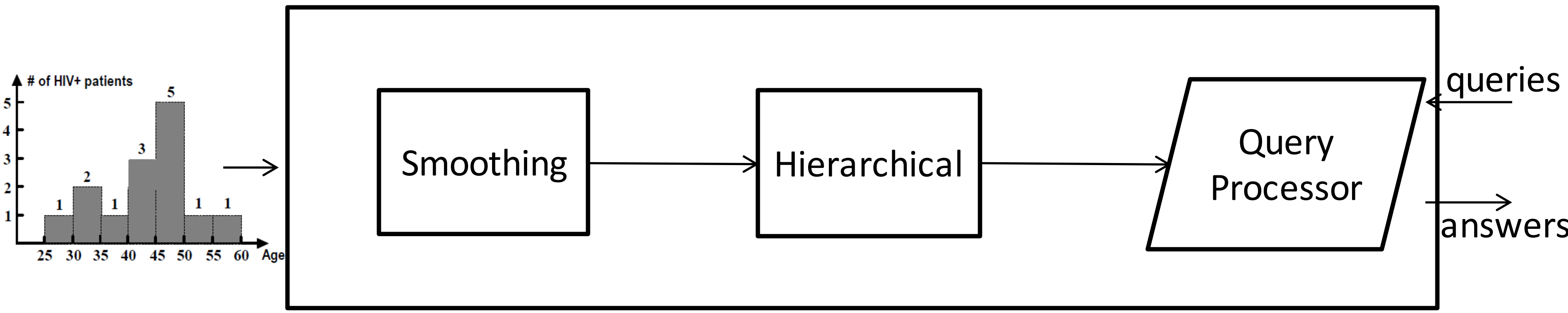


New Scheme: Subtree Smoothing

- Approximate the pruned nodes from the subtree root



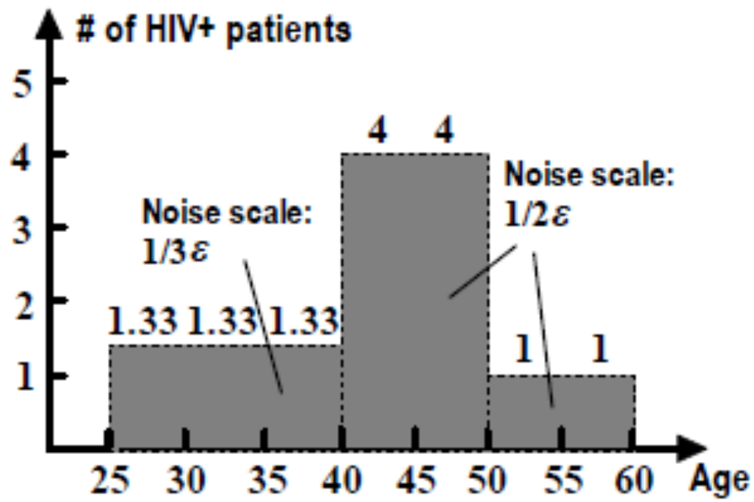
New Scheme: Subtree Smoothing



- Very fast method (O(n))
- Data-aware (applies smoothing)
- Accurate for large ranges (utilizes tree structure)

New Scheme: Smoothed Prefix Sums

- Combine the merits of Smoothing and Workload Aware



+

Workload Aware

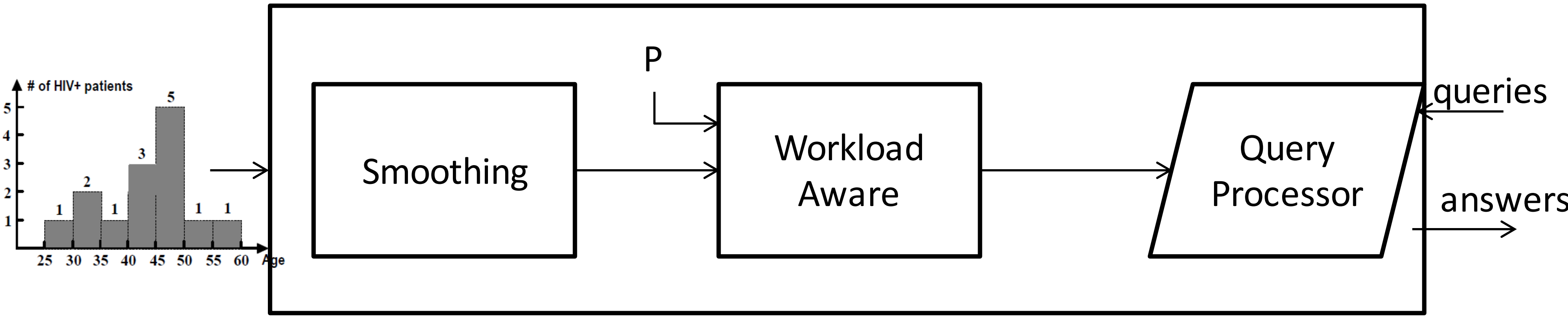
Accurate for small ranges

Very accurate; very slow (O(n^3log$n$))

New Scheme: Smoothed Prefix Sums

- Setting all possible queries (O(n^2)) as input to Workload Aware schemes, their running time is prohibitive
- Instead, use the prefix sums
  - P[1]=h[1], P[2]=h[1]+h[2], …, P[n]=h[1]+…+h[n]
  - O(n) possible queries
  - Any range can be computed by subtracting two prefix sums

New Scheme: Smoothed Prefix Sums



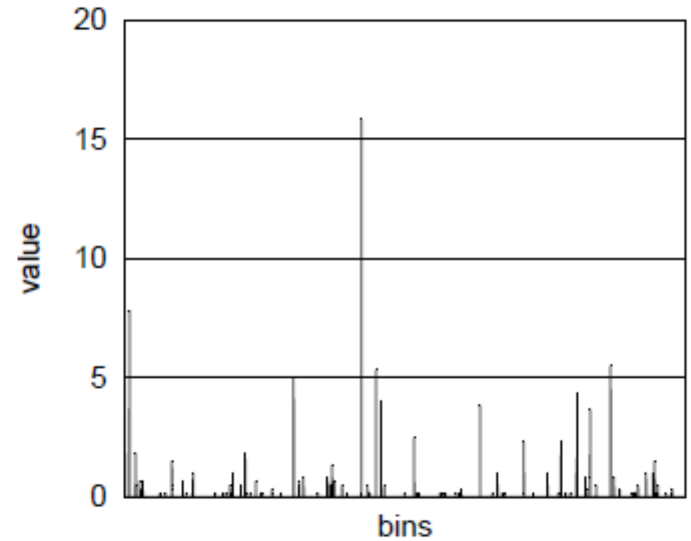
Running time of Workload Aware drops by an n factor ($O(n^2 \log n)$)

Experiments

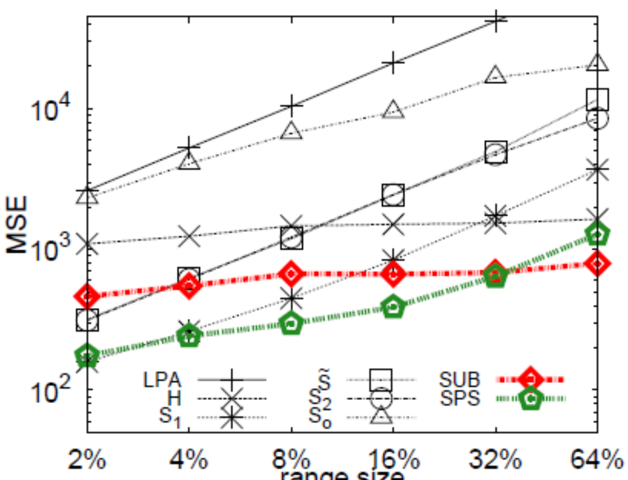
- Compare all new methods and previous ones that are not subsumed by others

Scheme	Abv.	Time
Laplace Perturbation Algorithm	LPA	$O(n)$
Hierarchical	H	$O(n)$
Smoothing	S_1, S_2, S_o, $\hat{S}$	$O(n^2 \log n)$, $O(n^2)$, $O(n^2)$, $O(n \log^2 n)$
Subtree Smoothing	SUB	$O(n)$
Smoothed Prefix Sums	SPS	$O(n^2 \log n)$

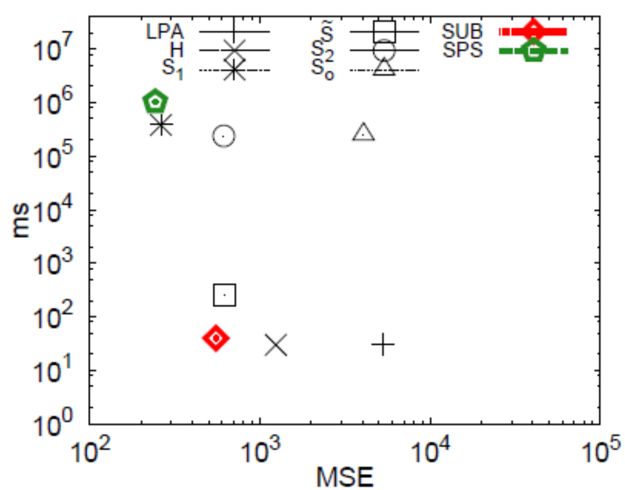
Net (64K bins)



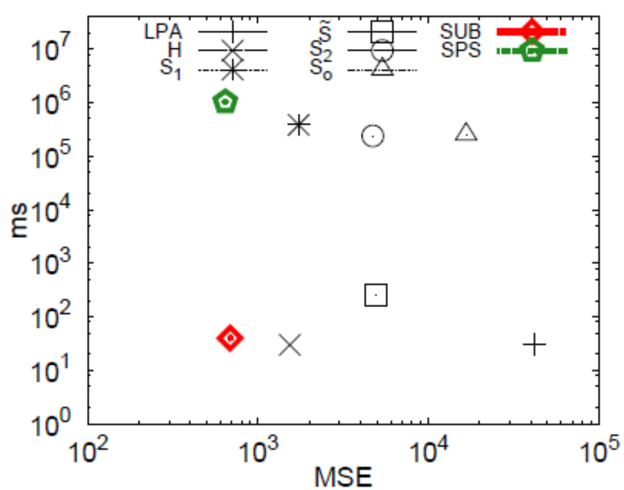
Error vs range



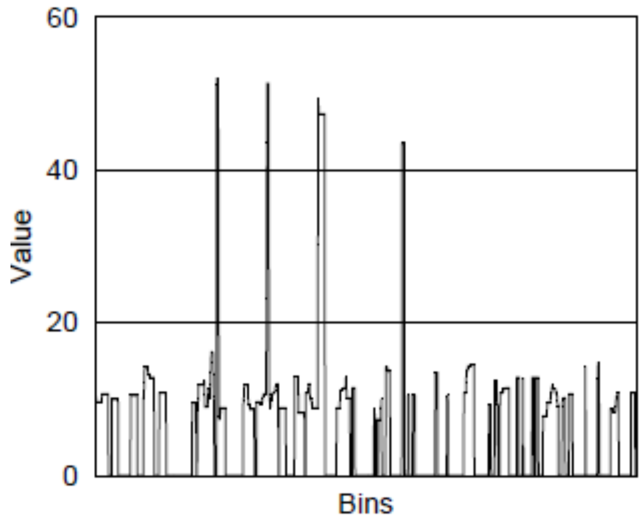
Skyline – small ranges



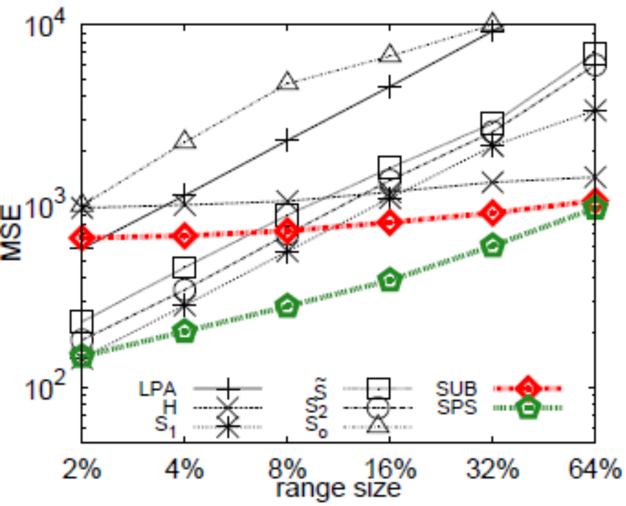
Skyline – large ranges



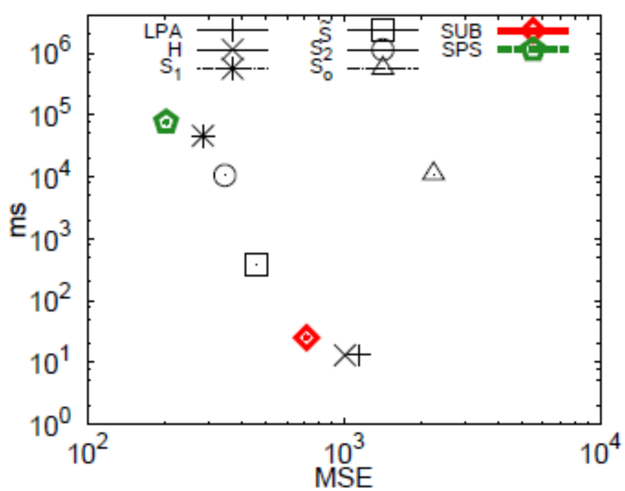
Rome (14K bins)



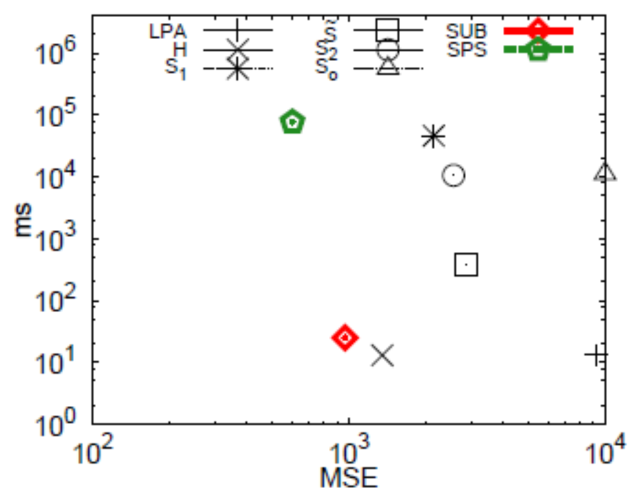
Error vs range



Skyline – small ranges



Skyline – large ranges



Challenges

- Modularize differentially private methods for other settings
- Is it possible to combine differentially private modules with cryptographic modules?
- Differential Privacy + Cryptography = ?

Thank you!