EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

MATHEMATISCH-
NATURWISSENSCHAFTLICHE FAKULTÄT
- Medical Data Privacy and Privacy Preserving Machine Learning
- Institute for Bioinformatics and Medical Informatics

Dynamic k-anonymity: A Topological Framework

Arjhun Swaminathan, Mete Akgün

International Workshop on Data Privacy Management (DPM 2024) , September 19, 2024

Outline

- k-anonymity
- Topology informed k-anonymity
  - Čech Complexes
  - Persistence Barcodes
  - Weighted Persistence Barcodes
- Dynamic k-anonymity using Persistence Homology
  - Addition
  - Deletions
  - Updates

Introduction

The goal of k-anonymity is to protect data prior to publishing.

Name	Admission Date	Age	Blood Pressure	Diagnosis
Maria	02.10.2022	23	121 mm Hg	Anxiety
Priya	05.10.2022	44	97 mm Hg	UTI
Ahmed	03.01.2023	21	95 mm Hg	–
Aiden	05.02.2023	41	100 mm Hg	Asthma

Identifiers | Quasi-identifiers | Sensitive Data

Table 1: Table illustrating the classification of data attributes into identifiers (to be de-identified prior to publication), quasi-identifiers, and sensitive data.

Problem: Quasi-identifier data can collectively identify an individual.

k-anonymity

How do we solve the problem?

Make k-individuals look alike.

T		
02.10.2022	23	121
05.10.2022	44	97
03.01.2023	21	95
05.02.2023	41	100

$\bar{T}$		
2022	$20-50$	$95-125$
2022	$20-50$	$95-125$
2023	$*1$	$70-100$
2023	$*1$	$70-100$

$\bar{T}^*$		
$****$	$**$	$***$
$****$	$**$	$**$
$****$	$**$	$**$
$****$	$**$	$***$

Data privacy vs. Data Utility.

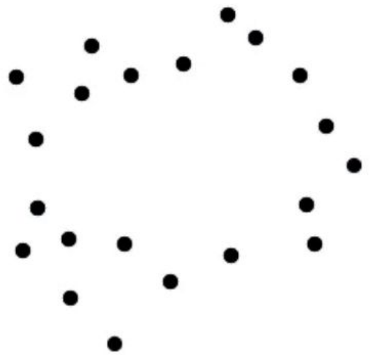
Topology-informed k-anonymity [1]

Pro: Can compute multiple generalizations for varied k-anonymity requirements in a single computation.

Con: Is restricted to static data. Needs complete recomputation for any changes to data - expensive.

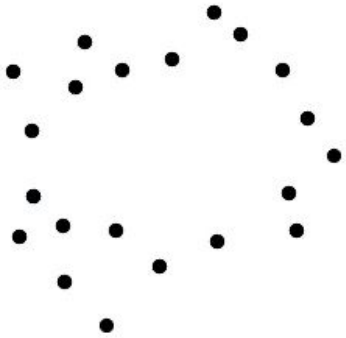
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



Topology-informed k-anonymity [1]

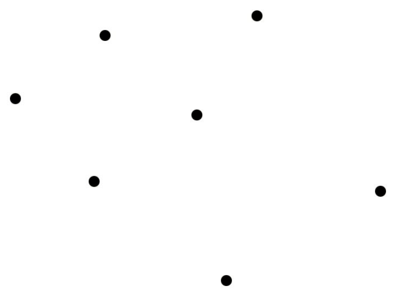
1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



Arthur Jaffe, *"VR Polygons: Non-Euclidean Virtual Reality,"* stat.berkeley.edu.

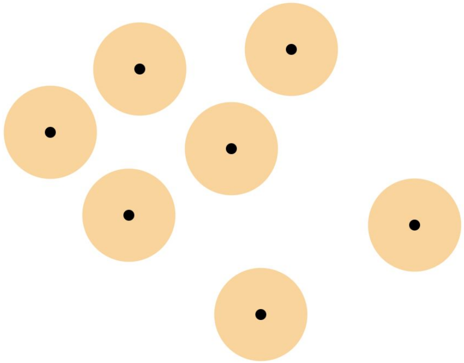
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



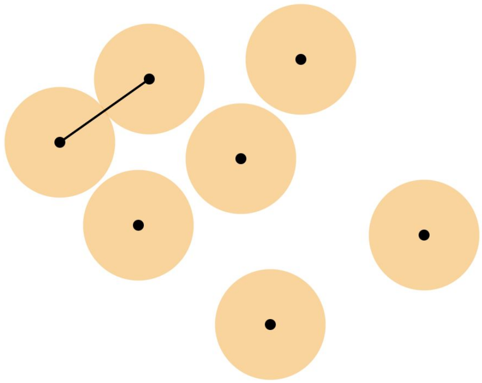
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



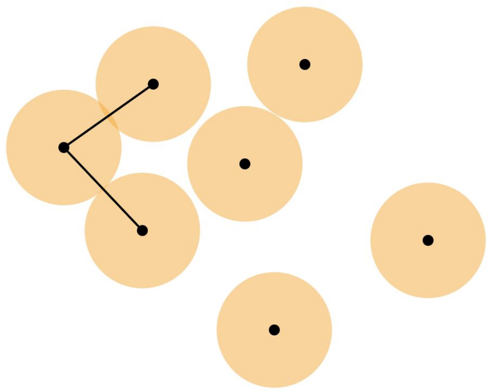
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



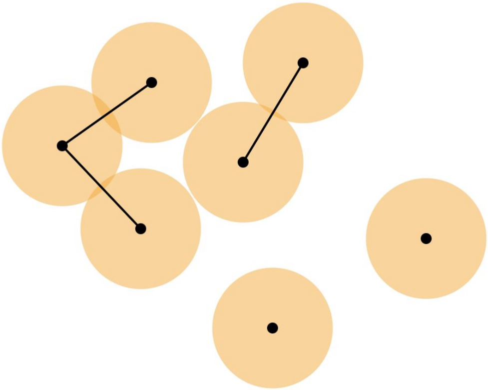
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



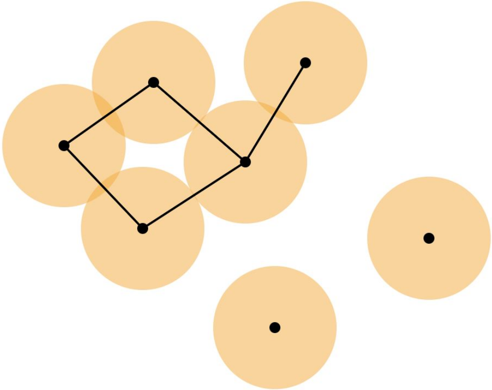
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



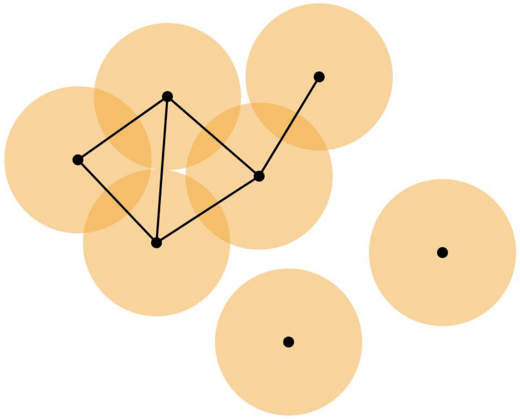
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



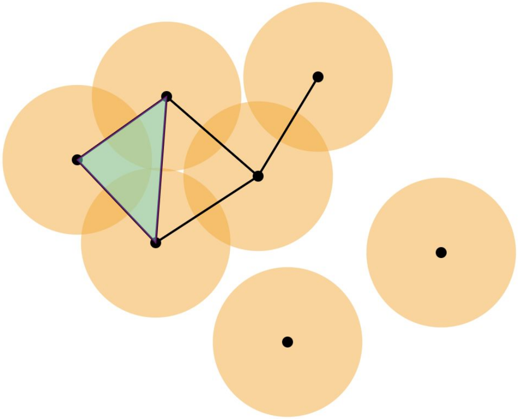
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



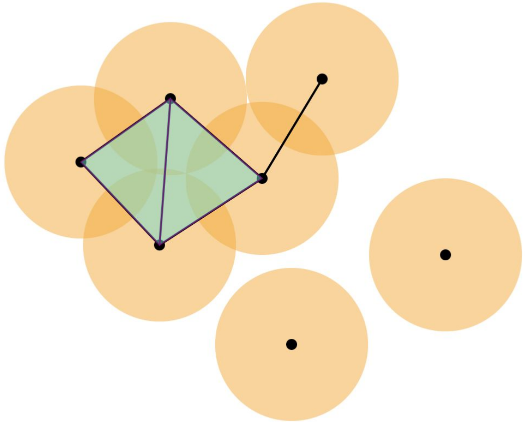
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



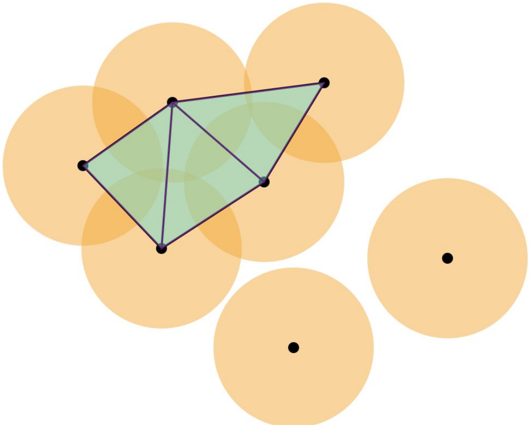
Topology-informed k-anonymity [1]

1. Make a point cloud
2. **Build a Čech complex**
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



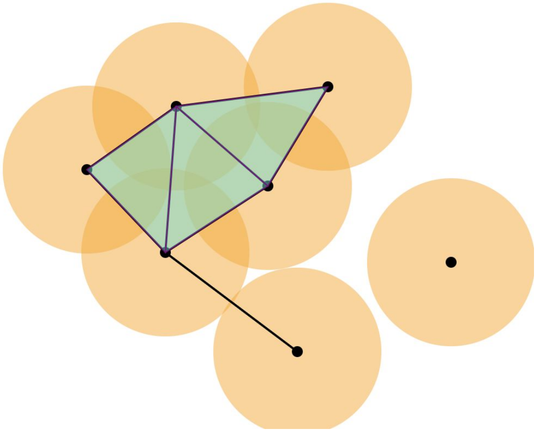
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



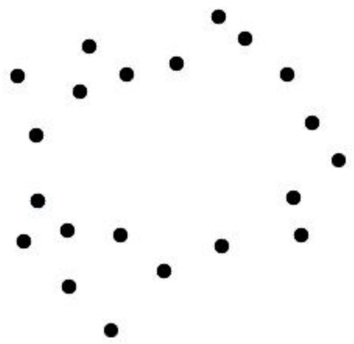
Topology-informed k-anonymity [1]

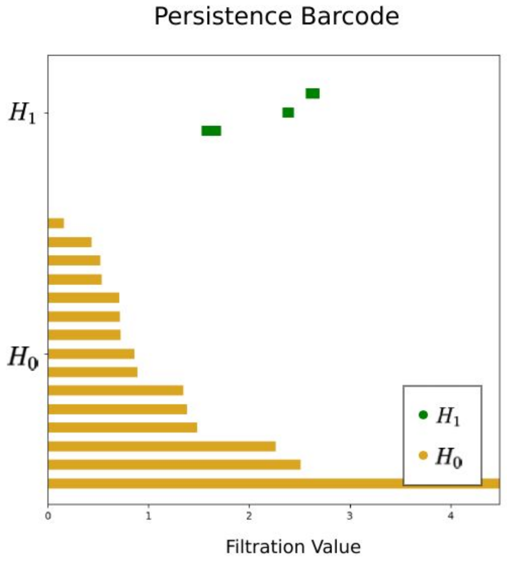
1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



Topology-informed k-anonymity [1]

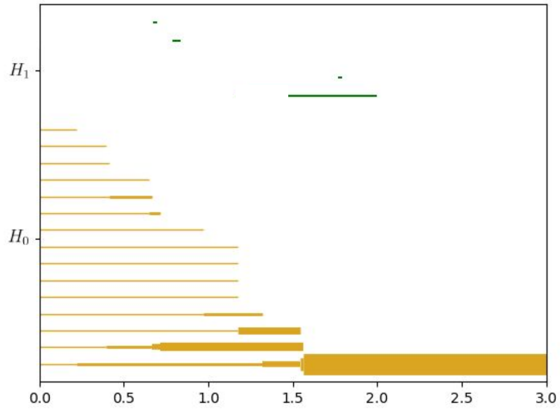
1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode





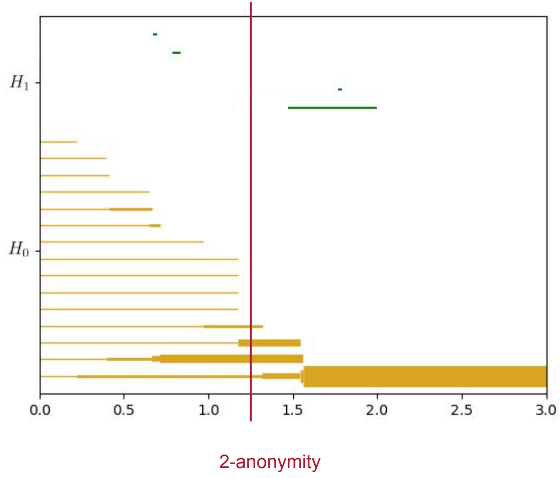
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



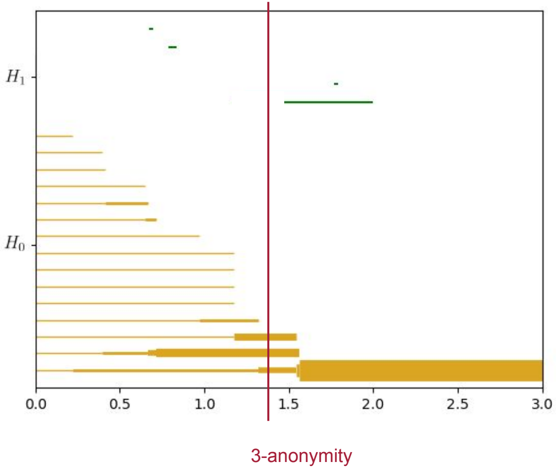
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. **Build Weighted Persistence Barcode**



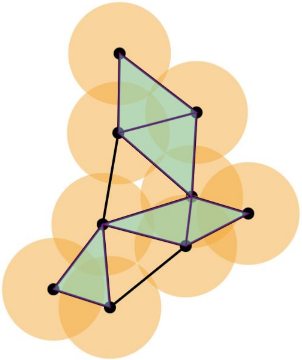
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. **Build Weighted Persistence Barcode**



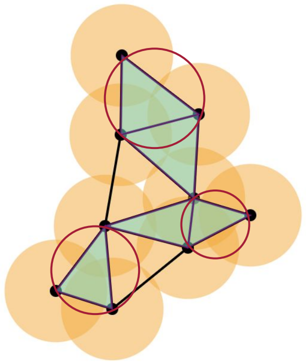
Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



Topology-informed k-anonymity [1]

1. Make a point cloud
2. Build a Čech complex
3. Compute the Persistence Barcode
4. Build Weighted Persistence Barcode



Topology-informed k-anonymity [1]

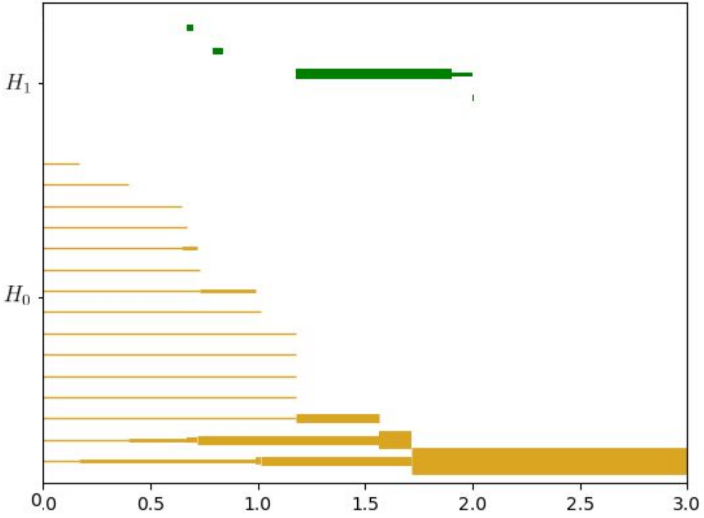
Pro: Can compute multiple generalizations for varied k-anonymity requirements in a single computation.

Con: Is restricted to static data. Needs complete recomputation for any changes to data - expensive.

Dynamic k-anonymity

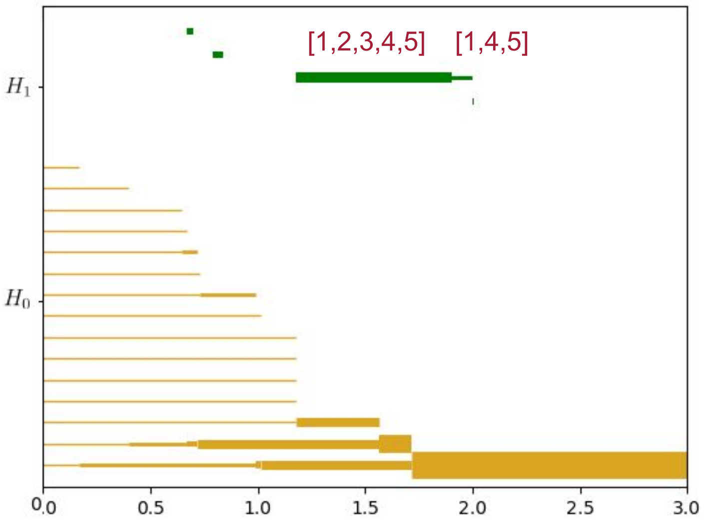
1. Introduce Hole-Weighted Persistence Barcodes
2. Data Removal
3. Data Addition
4. Data Updates

We do this using a breadth-first search (BFS).



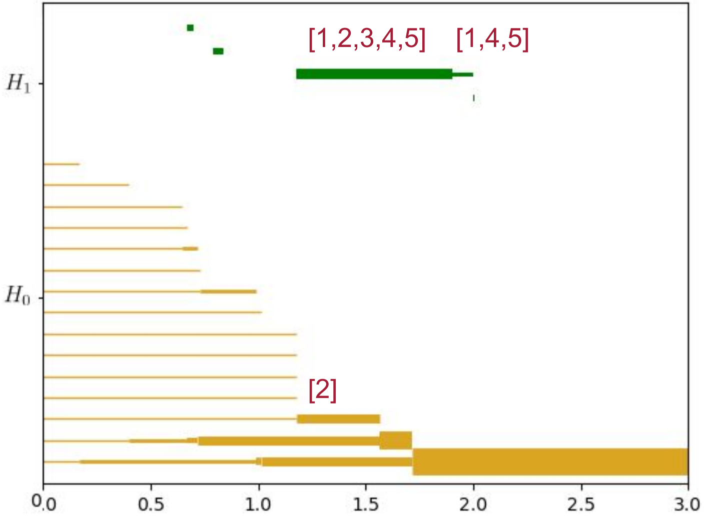
Dynamic k-anonymity

1. Introduce Hole-Weighted Persistence Barcodes
2. Data Removal
3. Data Addition
4. Data Updates



Dynamic k-anonymity

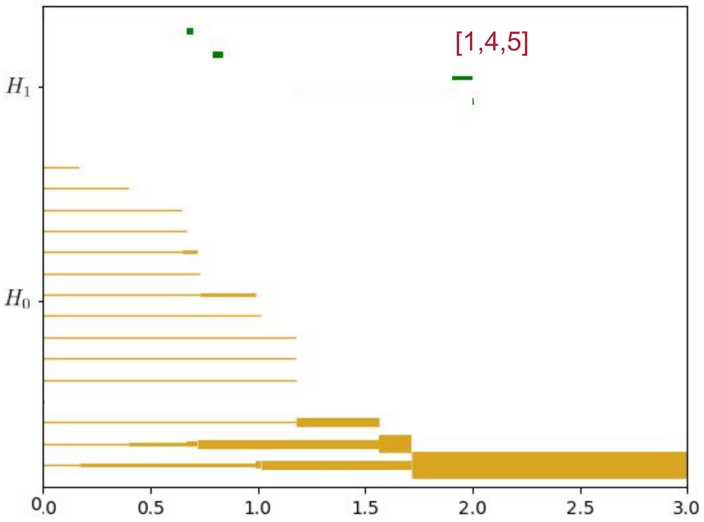
1. Introduce Hole-Weighted Persistence Barcodes
2. **Data Removal**
3. Data Addition
4. Data Updates



Vertex 2 is removed?

Dynamic k-anonymity

1. Introduce Hole-Weighted Persistence Barcodes
2. **Data Removal**
3. Data Addition
4. Data Updates

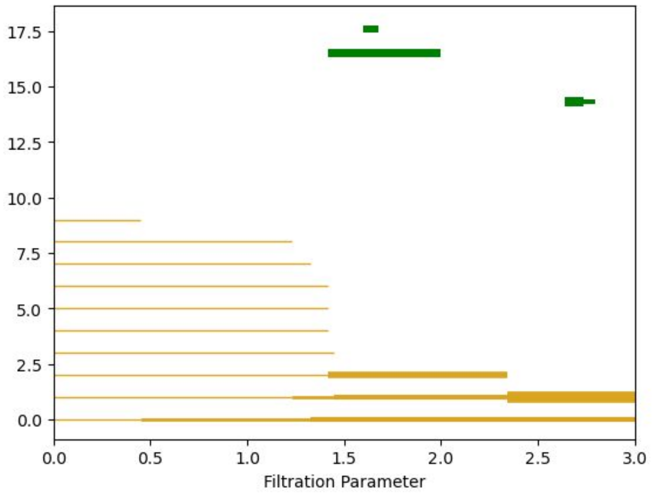


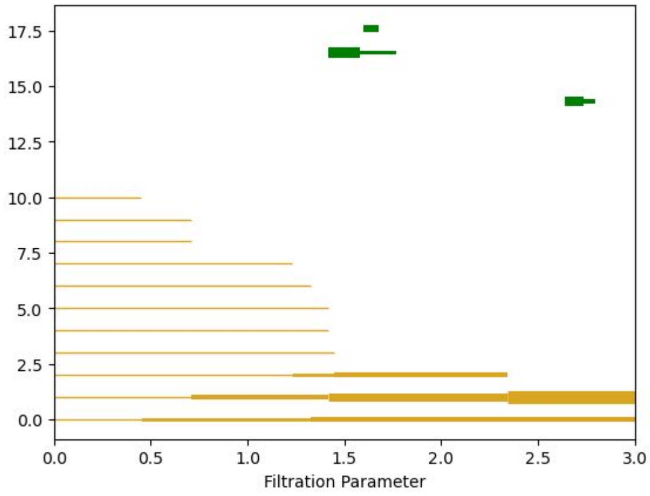
No problem.

Dynamic k-anonymity

1. Introduce Hole-Weighted Persistence Barcodes
2. Data Removal
3. **Data Addition**
4. Data Updates

Not a lot of changes occur when data is added. They are primarily local.

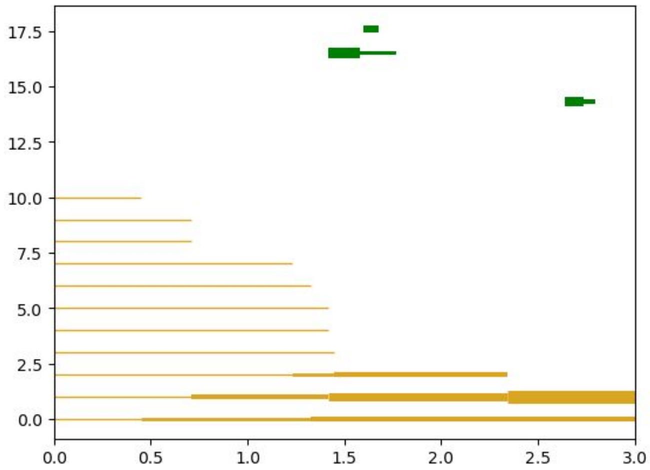




Dynamic k-anonymity

1. Introduce Hole-Weighted Persistence Barcodes
2. Data Removal
3. **Data Addition**
4. Data Updates

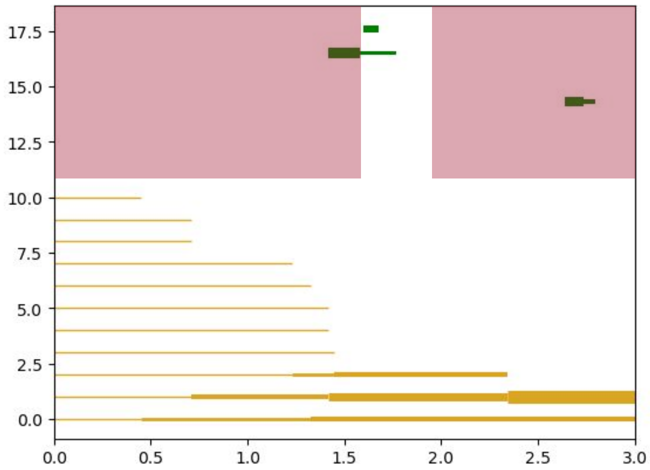
We introduce filtration trimming - where we find the radii where the changes occur, and only compute homology there.



Dynamic k-anonymity

1. Introduce Hole-Weighted Persistence Barcodes
2. Data Removal
3. **Data Addition**
4. Data Updates

We introduce filtration trimming - where we find the radii where the changes occur, and only compute homology there.



Dynamic k-anonymity

Data Points	Added Points	Filtration Length	Trimmed Length
10	1	231	19
10	2	298	20
10	5	575	45
20	1	1561	347
20	5	2625	386
20	10	4525	1115
50	1	22151	3301
50	5	27775	3792
50	10	36050	3374
100	1	171801	15379
100	5	193025	17263
100	10	221925	18760
100	25	325625	19242

Table 3: Filtration Lengths and Trimmed Filtration Lengths for Simulated Data with 2 Quasi-identifiers.

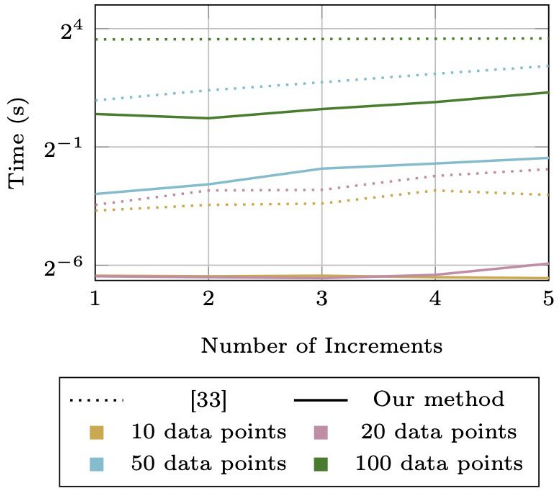
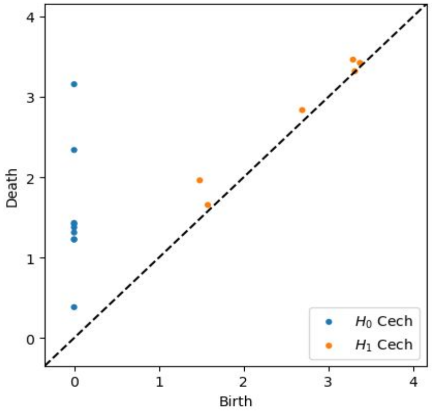


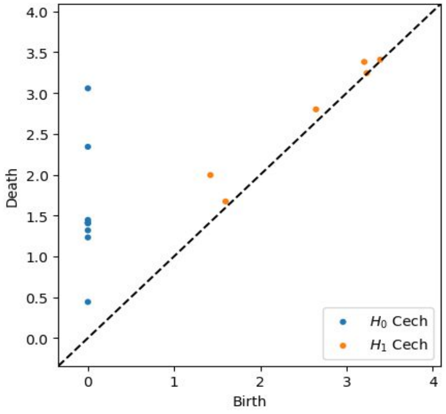
Fig. 5: Comparison of methods when data points are increased by 10% of the original dataset at each step. The time required to compute persistent homology on full and trimmed filtration lengths is plotted.

Dynamic k-anonymity

1. Introduce Hole-Weighted Persistence Barcodes
2. Data Removal
3. Data Addition
4. **Data Updates**

Persistence information is stable - minor changes in the data doesn't affect persistence information much.

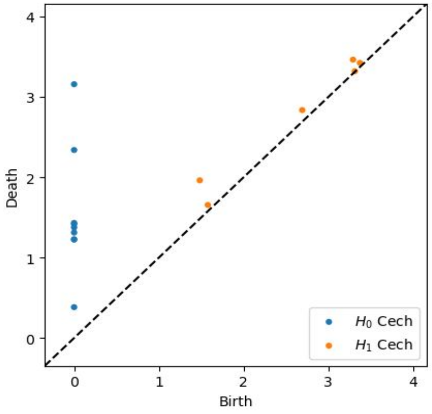


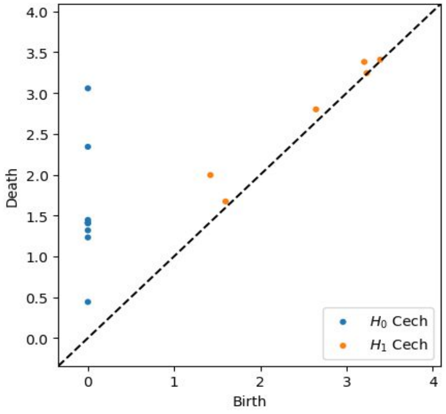


Dynamic k-anonymity

1. Introduce Hole-Weighted Persistence Barcodes
2. Data Removal
3. Data Addition
4. Data Updates

If current anonymized table already meets k-anonymity requirement, just edit the hole-weighted persistence barcode appropriately.
Else, use the removal and addition algorithms.





Computational Complexity

	Previous work [1]	Our method
Persistence information	$\mathcal{O}(\sum_i^M ({}^N C_i)^3)$	$\mathcal{O}(\sum_i^M ({}^N C_i)^3)$
Hole-weighted persistence barcode computation	-	$\mathcal{O}(\sum_i^M ({}^N C_i)^3)$
'K' removals	$\mathcal{O}(\sum_{J=N-K}^N \sum_i^M ({}^J C_i)^3)$	$\mathcal{O}(2\sum_i^M ({}^N C_i)^3 + KN)$
Additions	$\mathcal{O}(\sum_i^M ({}^N C_i)^3)$	$\mathcal{O}\left({}^{\bar{T}} C_{\bar{T}/2}(t/2)\right)$

*for N samples with M quasi-identifiers

*here, $\bar{T}$ represents the number of local t-dimensional simplices around the added point

Future Work

- Extending to categorical data
- Incorporating more robust privacy requirements

References

[1] Speranzon, A., Bopardikar, S.D.: An algebraic topological perspective to privacy. In: 2016 American Control Conference (ACC). pp. 2086–2091. IEEE (2016)

[2] Saul Nunes, *"A Nerve Playground,"* sauln.github.io.

[3] LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Mondrian multidimensional k-anonymity. In: 22nd International conference on data engineering (ICDE’06). pp. 25–25. IEEE (2006)

Thanks for listening!