Space and Time Efficient Kernel Density Estimation in High Dimensions

<table>
<thead>
<tr>
<th>Name</th>
<th>Institution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arturs Backurs</td>
<td>TTIC</td>
</tr>
<tr>
<td>Piotr Indyk</td>
<td>MIT</td>
</tr>
<tr>
<td>Tal Wagner</td>
<td>MIT</td>
</tr>
</tbody>
</table>
Background:
Density Estimation

**Problem**: Given a dataset $x_1, \ldots, x_n \in \mathbb{R}^d$, estimate density at a query point $y \in \mathbb{R}^d$.

*How to formalize this?*
**Background:**

**Kernel Similarity Measures**

**Method:** Define a similarity measure ("kernel"):

\[ k: \mathbb{R}^d \times \mathbb{R}^d \rightarrow [0,1] \]

such that the more similar \( x, y \) are, the closer \( k(x, y) \) to 1.

Examples of popular kernels:

- "Exponential": \( k(x, y) = \exp \left( -\frac{\|x-y\|_2}{\sigma} \right) \)
- "Laplacian": \( k(x, y) = \exp \left( -\frac{\|x-y\|_1}{\sigma} \right) \)
- "Gaussian": \( k(x, y) = \exp \left( -\frac{\|x-y\|_2^2}{\sigma} \right) \)

\( \sigma \) is a parameter called *bandwidth*.
Background:

Kernel Density Estimation

• **Definition**: The Kernel Density Estimation of a query $y$ in a dataset $X = \{x_1, \ldots, x_n\}$ is defined as

$$KDE_X(y) = \frac{1}{n} \sum_{i=1}^{n} k(x_i, y)$$
Fast KDE

• Exact naïve computation: $\Omega(n)$ time, too slow
  • Typically there are multiple query points

• Can we estimate $KDE_x(y)$ efficiently?
Fast KDE: Uniform Sampling

• Suppose we have the promise: $KDE_X(y) \geq \tau$ for some small $\tau > 0$
  - i.e.: the query $y$ is not too unrelated to the dataset $X$

• We want a $(1 \pm \varepsilon)$ relative approximation of $KDE_X(y)$

• **Uniform sampling**: If we choose $O\left(\frac{1}{\tau \cdot \varepsilon^2}\right)$ random points $\tilde{X} \subset X$, then
  $$KDE_{\tilde{X}}(y) = (1 \pm \varepsilon)KDE_X(y)$$

• Running time: $O\left(\frac{1}{\tau \cdot \varepsilon^2}\right)$. Can we do better?
Fast KDE: Hashing-Based Estimators (HBE) [Charikar & Siminelakis 2017]

- Method based on **Locality-Sensitive Hashing (LSH)** [Indyk & Motwani 98]

- **Definition:** The kernel $k$ is **LSHable** if there exists a distribution $\mathcal{H}$ over hash functions $h: \mathbb{R}^d \rightarrow \{0,1\}^*$, such that for every $x, y \in \mathbb{R}^d$,
  \[
  \Pr_{h \sim \mathcal{H}} [h(x) = h(y)] \approx \sqrt{k(x,y)}
  \]
  (hash collision probability)

- **Theorem** [Charikar & Siminelakis 2017]: If $k$ is LSHable, we can estimate KDE in time $O\left(\frac{1}{\sqrt{\tau} \cdot \epsilon^2}\right)$.

Improvement of $1/\sqrt{\tau}$ over random sampling; matters in practice [Siminelakis et al. 2019]

Exponential and Laplacian kernels are LSHable
Fast KDE: Our Results

- **Drawback of HBE:** Requires super-linear preprocessing time and storage space: \( O \left( n \cdot \frac{1}{\sqrt{\tau \cdot \varepsilon^2}} \right) = O \left( \frac{1}{\tau \sqrt{\tau \cdot \varepsilon^4}} \right) \)
  - Burdens practical implementation [Siminelakis et al. 2019]

- **This work:** We modify HBE to get the best of both worlds:
  - Preprocessing time and storage space: \( O \left( \frac{1}{\tau \cdot \varepsilon^2} \right) \) (same as uniform sampling)
  - Query KDE estimation time: \( O \left( \frac{1}{\sqrt{\tau \cdot \varepsilon^2}} \right) \) (same as HBE)
HBE Scheme

Preprocessing step

Query step

Dataset

Hash table

Hash table

Hash table

Query point

KDE estimate

Estimator
Space-Efficient HBE Scheme

Preprocessing step

Query step

Dataset

Hash table

Hash table

Hash table

Query point

KDE estimate

Estimator

Random dropout
Representative Experiments (more in paper)

- Forest cover type dataset*, Laplacian kernel

* Jock A Blackard and Denis J Dean, Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables, Computers and electronics in agriculture 24 (1999), no. 3, 131–151

Our query time is similar to HBE and better than uniform sampling (RS)

Our storage space and preprocessing time improve over HBE
References

• Moses Charikar, Paris Siminelakis. **Hashing-based-estimators for kernel density in high dimensions.** FOCS 2017

• Piotr Indyk, Rajeev Motwani. **Approximate nearest neighbors: towards removing the curse of dimensionality.** STOC 1998

• Paris Siminelakis, Kexin Rong, Peter Bailis, Moses Charikar, Philip Levis. **Rehashing kernel evaluation in high dimensions.** ICML 2019