Private Kernel Density Estimation without the Curse of Dimensionality

The Gaussian KDE of a dataset \( x_1, \ldots, x_n \in \mathbb{R}^d \) is the function that maps \( y \in \mathbb{R}^d \) to \( \frac{1}{n} \sum_{i=1}^{n} e^{-\|y-x_i\|^2} \).

Differentially private Gaussian KDE:

- **Curator**
  - Has private dataset \( x_1, \ldots, x_n \in \mathbb{R}^d \)
  - Releases a function \( \tilde{K} : \mathbb{R}^d \to \mathbb{R} \)
  - \( \tilde{K} \) must be \( \epsilon \)-DP w.r.t. the dataset
  - \( \tilde{K} \) should approximate the Gaussian KDE

- **Client**
  - Receives \( \tilde{K} \)
  - For each query \( y \in \mathbb{R}^d \), w.h.p.:
    \[
    \tilde{K}(y) \approx \frac{1}{n} \sum_{i=1}^{n} e^{-\|y-x_i\|^2}
    \]

Our results:
- High dimensions: \( \epsilon \)-DP, error \( \sim 1/\sqrt{n} \), runtime linear in \( d \) \( \rightarrow \) no curse of dimensionality
- Low dimensions: \( \epsilon \)-DP, error \( \sim (\log n)^{O(d)}/n \), runtime exp. in \( d \) \( \rightarrow \) near-linear error decay if \( d = O(1) \)

The Technical Stuff:

**Fast Private Kernel Density Estimation via Locality Sensitive Quantization**

**What is LSQ?**
- Expressing a kernel on \( \mathbb{R}^d \) with features that are few, bounded, and sparse.

Formally:
- \( \tilde{k}(x,y) = (Q,R,S)\text{-LSQ} \)able if there is a distribution \( D \) over pairs of functions \( f,g: \mathbb{R}^d \to [{-R,R}]^S \), such that for all \( x,y \in \mathbb{R}^d \):
  - \( f(x) \) and \( g(y) \) have \( \leq S \) non-zeros,
  - \( k(x,y) \approx E_{(f,g) \sim D}[f(x)^Tg(y)] \)

**Theorem:** LSQ \( \Rightarrow \epsilon \)-DP KDE.

And, if \( Q,R,S \) are small, the mechanism has good utility and computational efficiency.

**LSQ Constructions:**
- Random Fourier Features (RFF) [Rahimi-REcht ’07]
- Leads to our high-dimensional result
- Fast Gauss Transform (FGT) [Greengard-Strain ’91]
- Leads to our low-dimensional result
- Locality Sensitive Hashing (LSH) [Indyk-Andoni ’09]
- Recovers prior results of [Coleman-Shrivastava 21]
- LSQ extends LSH to more kernels (e.g., Gaussian)

**Prior work:**

<table>
<thead>
<tr>
<th>Method</th>
<th>Privacy</th>
<th>Error decay</th>
<th>Runtime in ( d )</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Several]</td>
<td>( \epsilon )-DP</td>
<td>( 1/\sqrt{n} )</td>
<td>( \exp(d) )</td>
</tr>
<tr>
<td>Prior [HRW13]</td>
<td>( (\epsilon, \delta) )-DP</td>
<td>( 1/n )</td>
<td>( \exp(d) )</td>
</tr>
<tr>
<td>[CS’21]</td>
<td>( \epsilon )-DP</td>
<td>( 1/\sqrt{n} )</td>
<td>( O(d) )</td>
</tr>
<tr>
<td>Ours</td>
<td>LSQ-RFF</td>
<td>( \epsilon )-DP</td>
<td>( \exp(d) )</td>
</tr>
<tr>
<td>Ours</td>
<td>LSQ-FGT</td>
<td>( \epsilon )-DP</td>
<td>( \exp(d) )</td>
</tr>
</tbody>
</table>

Does it work for other kernels?
Yes, but \( \epsilon \), \( \delta \), \( S \), see paper.

Paper, code, etc.: 

Tal Wagner, Yonatan Naamad, Nina Mishra

Yes, this is a “betterposter”, for #better or worse