Supplementary material for: Rehashing Kernel Evaluation in High Dimensions

Paris Siminelakis [*1] **Kexin Rong** [*1] **Peter Bailis** [1] **Moses Charikar** [1] **Philip Levis** [1]

Outline In the first two sections, we give proofs for all our formal results while restating them for convenience. In Section 2, we give the precise description of our Hashing-Based-Sketch and its theoretical analysis. In Section 3, we present our diagnostic and visualization procedures in more detail. In Section 4, we explain our design decisions and the procedure for generating a fair synthetic benchmark. In Section 5, we provide additional setup details and results for the experimental evaluation.

1. Proofs

1.1. Basic inequalities

We first state without proof some well known inequalities that we will use in the proofs.

Lemma 1 (Chebyshev's and Paley-Zygmund inequalities)**.** *For a non-negative random variable Z and parameters $t > 0$, $\theta \in [0,1]$, we have*

$$\mathbb{P}[Z \geq (t+1)\cdot \mathbb{E}[Z]] \leq \frac{1}{t^2}\cdot \mathrm{RelVar}[Z], \tag{1}$$

$$\mathbb{P}[Z > (1-\theta)\mathbb{E}[Z]] \geq \frac{1}{1+\frac{1}{\theta^2}\cdot \mathrm{RelVar}[Z]}. \tag{2}$$

Theorem 1 (Chernoff bounds)**.** *Let $X = \sum_{i=1}^n X_i$, where $X_i = 1$ with probability p_i and $X_i = 0$ with probability $1-p_i$, and all X_i are independent. Let $v = \mathbb{E}[X] = \sum_{i=1}^n p_i$. Then for $\delta > 0$*

$$\mathbb{P}[X \geq (1+\delta)v] \leq e^{-\frac{\delta^2}{2+\delta}v}, \tag{3}$$

$$\mathbb{P}[X \leq (1-\delta)v] \leq e^{-\frac{1}{2}\delta^2 v}. \tag{4}$$

1.2. Median-trick to boost success probability

The median-trick is based on concentration of sums of independent binary random variables. If we define binary random variables appropriately we can obtain bounds for the concentration of the median of i.i.d. random variables around their expectation.

Lemma 2. *Let $Z_1, \ldots, Z_L$ be $L \geq 1$ i.i.d. copies of a non-negative random variable with $\mathrm{RelVar}[Z] \leq \frac{\epsilon^2}{6}$ then:*

$$\mathbb{P}\left[\mathrm{median}\{Z_1,\ldots,Z_L\} \geq (1+\epsilon)\mathbb{E}[Z]\right] \leq e^{-\frac{L}{6}},$$
$$\mathbb{P}\left[\mathrm{median}\{Z_1,\ldots,Z_L\} \leq (1-\epsilon)\mathbb{E}[Z]\right] \leq e^{-\frac{L}{4}}.$$

Proof of Lemma 2. Let

$$X_i = \mathbb{I}[Z_i \geq (1+\epsilon)\mathbb{E}[Z]],$$
$$Y_i = \mathbb{I}[Z_i \leq (1-\epsilon)\mathbb{E}[Z]].$$

By Lemma 1, we have that

$$a_i = \mathbb{E}[X_i] \leq \frac{1}{\epsilon^2}\frac{\epsilon^2}{6} \leq \frac{1}{6}, \quad b_i = \mathbb{E}[Y_i] \leq \frac{1}{7}.$$

We get the following upper bounds

$$\mathbb{P}[\mathrm{median}\{Z_1,\ldots,Z_L\} \geq (1+\epsilon)\mathbb{E}[Z]] \leq \mathbb{P}[\sum_{i=1}^L X_i \geq \frac{L}{2}].$$

$$\mathbb{P}[\mathrm{median}\{Z_1,\ldots,Z_L\} \geq (1+\epsilon)\mathbb{E}[Z]] \leq \mathbb{P}[\sum_{i=1}^L Y_i \geq \frac{L}{2}],$$

that along with Chernoff bounds will give us our result. We only show the first inequality as the second one follows similarly. Let $A = \sum_{i=1}^L a_i \leq L/6$, the first event is bounded by $\exp(-\frac{(\frac{L/2}{A}-1)^2}{2+(\frac{L/2}{A}-1)}A) \leq \exp(-L/6)$. □

1.3. Moments of Hashing-Based-Estimators

Lemma 3. *Assuming that $\forall i \in [n], p(x_i, q) > 0$ then*

$$\mathbb{E}[Z_h] = \sum_{i=1}^n u_i k(x, x_i), \tag{5}$$

$$\mathbb{E}[Z_h^2] = \sum_{i,j=1}^n k^2(q, x_i)\frac{u_i \mathbb{P}[i,j \in H(q)] u_j}{p^2(q,x_i)}. \tag{6}$$

[*]Equal contribution [1]Stanford University, Stanford, California, US. Correspondence to: Paris Siminelakis <psimin@stanford.edu>, Kexin Rong <krong@stanford.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

Proof of Lemma 3. We start with the expectation:

$$\begin{aligned}\mathbb{E}_{h,X}[\frac{k(q,X)}{p(q,X)}u_{H(q)}] &= \mathbb{E}_h[\mathbb{E}_X[\frac{k(q,X)}{p(q,X)}]u_{H(q)}] \\ &= \mathbb{E}_h[\sum_{i\in H(q)}\frac{u_i}{u_{H(q)}}\frac{k(q,x_i)}{p(q,x_i)}u_{H(q)}] \\ &= \sum_{i=1}^{n} u_i\mathbb{E}[\mathbb{I}[h(x_i)=h(q)]]\frac{k(x_i,q)}{p(x_i,q)} \\ &= \sum_{i=1}^{n} u_i k(x_i,q)\end{aligned}$$

We proceed with the second moment:

$$\begin{aligned}\mathbb{E}_{h,X}[\frac{k^2(q,X)}{p^2(q,X)}u^2_{H(q)}] &= \mathbb{E}_h[\mathbb{E}_X[\frac{k^2(q,X)}{p^2(q,X)}]u^2_{H(q)}] \\ &= \mathbb{E}_h[\sum_{i\in H(q)}\frac{u_i}{u_{H(q)}}\frac{k^2(q,x_i)}{p^2(q,x_i)}u^2_{H(q)}] \\ &= \mathbb{E}_h[\sum_{i\in H(q)} u_i\frac{k^2(q,x_i)}{p^2(q,x_i)}u_{H(q)}] \\ &= \mathbb{E}_h[\sum_{i,j\in H(q)} u_iu_j\frac{k^2(q,x_i)}{p^2(q,x_i)}] \\ &= \sum_{i,j=1}^{n} k^2(x_i,q)\frac{u_i\mathbb{P}[i,j\in H(q)]u_j}{p^2(x_i,q)}\end{aligned}$$

□

1.4. Refined Variance bound

Here, we derive our new inequality bounding the variance of HBE and RS. Let $\mu \le \lambda \le L \le 1$ and define:

$$S_1 = \{i\in[n] : L \le w_i \le 1\} \tag{7}$$
$$S_2 = \{i\in[n]\setminus S_1 : \lambda \le w_i \le L\} \tag{8}$$
$$S_3 = \{i\in[n]\setminus(S_2\cup S_1) : \mu \le w_i \le \lambda\} \tag{9}$$
$$S_4 = \{i\in[n] : w_i < \mu\} \tag{10}$$

as well as $\mu_\ell = \sum_{i\in S_\ell} u_i w_i \le \mu$. The intuition behind the definition of the sets is that for radial decreasing kernels they correspond to spherical annuli around the query (Figure 1).

Lemma 4. *For non-negative weights $w_1,\dots,w_n$, vector $u\in\Delta_n$ and sets $S_1,\dots,S_4\subseteq[n]$ as above it holds*

$$\begin{aligned}\sum_{i,j\in[n]} w_i^2\{u_iV_{ij}u_j\} \le &\sum_{\ell\in[3],\ell'\in[3]}\sup_{\substack{i\in S_\ell,\\ j\in S_{\ell'}}}\left\{\frac{V_{ij}w_i}{w_j}\right\}\mu_\ell\mu_{\ell'} \\ &+ u_{S_4}\sum_{\ell\in[3]}\sup_{\substack{i\in S_\ell,\\ j\in S_4}}\left\{V_{ij}\frac{w_i}{\mu}\right\}\mu_\ell\mu \\ &+ \sup_{i\in S_4,j\in[n]}\{V_{ij}w_i\}\cdot\mu_4 \end{aligned}\tag{11}$$

where $u_S := \sum_{j\in S} u_j \le 1$.

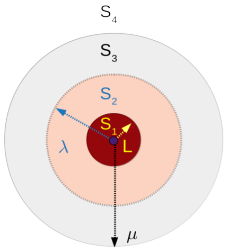


Figure 1. Depiction of the sets that appear in Lemma 4

Proof of Lemma 4. First we observe that $S_1\uplus S_2\uplus S_3\uplus S_4 = [n]$ forms a partition:

$$\begin{aligned}\sum_{i,j\in[n]} u_iu_jV_{ij}w_i^2 = &\sum_{\ell,\ell'\in[3]}\sum_{i\in S_\ell,j\in S_{\ell'}} u_iu_jV_{ij}w_i^2 \\ &+ \sum_{\ell\in[3]}\sum_{i\in S_\ell,j\in S_4} u_iu_jV_{ij}w_i^2 \\ &+ \sum_{i\in S_4,j\in[n]} u_iu_jV_{ij}.w_i^2\end{aligned}\tag{12}$$

For the first three sets we have some bounds on the ration $\frac{w_i}{w_j}$ whereas for the last set we have a bound on the w_i. We utilize these by:

$$\sum_{\substack{i\in S_\ell,\\ j\in S_{\ell'}}}\frac{V_{ij}w_i}{w_j}u_iw_iu_jw_j \le \sup_{\substack{i\in S_\ell,\\ j\in S_{\ell'}}}\{\frac{V_{ij}w_i}{w_j}\}\sum_{i\in S_\ell} w_iu_i\sum_{j\in S_{\ell'}} w_ju_j,$$

$$\sum_{\substack{i\in S_\ell,\\ j\in S_4}}\frac{V_{ij}w_i}{\mu}w_iu_iu_j\mu \le u_{S_4}\sup_{\substack{i\in S_\ell,\\ j\in S_4}}\left\{\frac{V_{ij}w_i}{\mu}\right\}\mu\sum_{i\in S_\ell} w_iu_i,$$

$$\sum_{\substack{i\in S_4,\\ j\in[n]}}\{V_{ij}w_i\}u_iu_jw_i \le \sup_{i\in S_4,j\in[n]}\{V_{ij}w_i\}\|u\|_1\sum_{j\in S_4} w_iu_i.$$

Identifying μ_i in the above expressions and substituting the bounds in (12) completes the proof. □

1.5. Adaptive procedure

Theorem 2. *Given an (a,β,γ)-regular estimator $\mathcal{Z}$, the AMR procedure outputs a number $\hat{Z}$ such that*

$$\mathbb{P}[|\hat{Z}-\mu| \le \epsilon\cdot\max\{\mu,\tau\}] \ge \frac{2}{3} - O_{\gamma,\alpha}(\epsilon^2)$$

and with the same probability uses $O_\gamma(\frac{1}{\epsilon^2}\frac{1}{\mu^\beta})$ samples.

Proof of Theorem 2. Recall that $\mu_t = (1+\gamma)^{-t}$ and let $t_0 := t_0(\mu)\in\mathbb{Z}$ such that:

$$\mu_{t_0+1} \le \mu \le \mu_{t_0} \tag{13}$$

We consider two cases $t_0 < T$ or $t_0 \ge T$.

Case I ($t_0 < T$)**.** In this case, we want to show that our algorithm with constant probability does not terminate before t_0 and not after $t_0 + 1$.

Let $\bar{Z}_t$ be the mean of m_t i.i.d. samples $Z_t^{(i)} \sim \mathcal{Z}(t, \gamma)$ with mean $\mathbb{E}[Z_t^{(i)}] = \mu$ and $\mathrm{RelVar}[Z_t^{(i)}] \le V_t(\mu)$. Then,

$$\mathrm{RelVar}[\bar{Z}_t] \le \frac{\epsilon^2}{6} \frac{V_t(\mu)}{V_t(\mu_{t+1})}. \tag{14}$$

Let A_0 be the event that the algorithm terminates before t_0.

$$\begin{aligned}
\mathbb{P}[A_0] &= \mathbb{P}[\exists t < t_0, \bar{Z}_t \ge \mu_t] && (15)\\
&\le \sum_{t<t_0} \mathbb{P}[\bar{Z}_t \ge \left(\frac{\mu_t}{\mu}\right)\mu] && (16)\\
&\le \frac{\epsilon^2}{6} \sum_{t=1}^{t_0-1} \frac{\mu^2}{(\mu_t-\mu)^2} \frac{V_t(\mu)}{V_t(\mu_{t+1})} && (17)\\
&\le \frac{\epsilon^2}{6} \sum_{t=1}^{t_0-1} \frac{\mu^2}{(\mu_t-\mu)^2} (\frac{\mu_{t+1}}{\mu})^{2-\alpha}. && (18)
\end{aligned}$$

where in (16) we use union bound, in (17) we use the first part of Lemma 1 and in (18) property (B) of a regular estimator. In the next three inequalities we use (13), $t \le t_0 - 1$ and $\sum_{t=0}^{s} x^s \le (1-x)^{-1}$ for $x < 1$.

$$\begin{aligned}
\mathbb{P}[A_0] &\le \frac{\epsilon^2}{6} \sum_{t=1}^{t_0-1} \frac{1}{(1-\frac{\mu_{t_0}}{\mu_t})^2} \frac{\mu_{t+1}^2}{\mu_t^2} (\frac{\mu_{t_0}}{\mu_{t+1}})^{\alpha} && (19)\\
&\le \frac{\epsilon^2}{6} \frac{1}{\gamma^2} \mu_{t_0}^{\alpha} \sum_{t=1}^{t_0-1} (1+\gamma)^{-\alpha(t_0-t-1)} && (20)\\
&\le \frac{\epsilon^2}{6} \frac{1}{\gamma^2} \mu_{t_0}^{\alpha} \frac{1}{1-(1+\gamma)^{-\alpha}}. && (21)
\end{aligned}$$

Furthermore, let A_1 be the event that the algorithm terminates after $t > t_0 + 1$.

$$\begin{aligned}
\mathbb{P}[A_1] &= \mathbb{P}[\forall t \le t_0 + 1, \bar{Z}_t < \mu_t] && (22)\\
&\le \mathbb{P}[\bar{Z}_{t_0+1} < \mu_{t_0+1}] && (23)\\
&= 1 - \mathbb{P}[\bar{Z}_{t_0+1} \ge \mu_{t_0+1}]. && (24)
\end{aligned}$$

Using the second part of Lemma 1 (Paley-Zygmund)

$$\begin{aligned}
\mathbb{P}[\bar{Z}_{t_0+1} \ge \mu_{t_0+1}] &\ge \frac{1}{1 + \frac{(\gamma+1)^2}{\gamma^2} \frac{\epsilon^2}{6} \frac{V_{t_0+1}(\mu)}{V_{t_0+1}(\mu_{t_0+1})}} && (25)\\
&\ge \left(1 + \frac{(\gamma+1)^2}{\gamma^2} \frac{\epsilon^2}{6}\right)^{-1}. && (26)
\end{aligned}$$

Therefore, $\mathbb{P}[A_1] \le 1 - \left(1 + \frac{(\gamma+1)^2}{\gamma^2} \frac{\epsilon^2}{6}\right)^{-1} \le \frac{(\gamma+1)^2}{\gamma^2} \frac{\epsilon^2}{6}$. Finally, let t^* be the (random) level where the algorithm terminates and A_2 be the event that $|\bar{Z}_{t^*} - \mu| > \epsilon\mu$. If any of the three events happen we say that the procedure fails. We can bound the failure probability by:

$$\begin{aligned}
\mathbb{P}[F] &= \mathbb{P}[A_0 \vee A_1 \vee A_2]\\
&= \mathbb{P}[A_0 \vee A_1 \vee A_2 \wedge A_0] + \mathbb{P}[(A_0 \vee A_1 \vee A_2) \wedge A_0^c]\\
&\le \mathbb{P}[A_0] + \mathbb{P}[A_1 \wedge A_0^c] + \mathbb{P}[A_2 \wedge A_0^c]. && (27)
\end{aligned}$$

To bound the last term we use:

$$\begin{aligned}
\mathbb{P}[A_2 \wedge A_0^c] &= \mathbb{P}[A_2 \wedge A_0^c \wedge A_1] + \mathbb{P}[A_2 \wedge A_0^c \wedge A_1^c]\\
&\le \mathbb{P}[A_1] + \mathbb{P}[A_2 \wedge A_0^c \wedge A_1^c].
\end{aligned}$$

and

$$\begin{aligned}
\mathbb{P}[A_2 \wedge A_0^c \wedge A_1^c] &= \sum_{t \in \{t_0, t_0+1\}} \mathbb{P}[|\bar{Z}_t - \mu| > \epsilon\mu \wedge t^* = t]\\
&\le \sum_{t \in \{t_0, t_0+1\}} \mathbb{P}[|\bar{Z}_t - \mu| > \epsilon\mu]\\
&\le \frac{1}{\epsilon^2} \sum_{t \in \{t_0, t_0+1\}} \mathrm{RelVar}[\bar{Z}_t]\\
&\le \frac{1}{\epsilon^2} \sum_{t \in \{t_0, t_0+1\}} \frac{\epsilon^2}{6} \frac{V_t(\mu)}{V_t(\mu_{t+1})}.
\end{aligned}$$

By definition $\mu \ge \mu_{t+1}$ for all $t \ge t_0$, thus by (B) and (28):

$$\mathbb{P}[A_2 \wedge A_0^c \wedge A_1^c] \le \frac{2}{6} = \frac{1}{3}. \tag{28}$$

Hence, the overall probability failure is bounded by:

$$\begin{aligned}
\mathbb{P}[F] &\le \mathbb{P}[A_0] + 2\mathbb{P}[A_1] + \mathbb{P}[A_2 \wedge A_0^c \wedge A_1^c]\\
&\le \frac{\epsilon^2}{6} \frac{1}{\gamma^2} \mu_{t_0}^{\alpha} \frac{1}{1-(1+\gamma)^{-\alpha}} + 2\frac{(\gamma+1)^2}{\gamma^2} \frac{\epsilon^2}{6} + \frac{1}{3}.
\end{aligned}$$

When the algorithm succeeds the total number of samples is bounded by

$$\begin{aligned}
\sum_{t=1}^{t_0+1} \lceil \frac{6}{\epsilon^2} V_t(\mu_{t+1}) \rceil &\le (t_0+1) + \frac{6C}{\epsilon^2} \sum_{t=1}^{t_0+1} (1+\gamma)^{\beta(t+1)}\\
&\le (t_0+1) + \frac{6C}{\epsilon^2} (1+\gamma)^{2\beta} \frac{(1+\gamma)^{\beta t_0}}{\gamma}\\
&\le (t_0+1) + \frac{6C}{\epsilon^2} \frac{(1+\gamma)^{\beta}}{\gamma} \frac{1}{\mu^{\beta}}.
\end{aligned}$$

Case II ($t_0 \ge T$)**.** In this case $\mu \le \mu_T \le \frac{1}{1+\gamma}\epsilon\tau$. By the same arguments as in the case $t_0 < T$ we get that the probability terminates before $t < t_0$ is at most $\frac{\epsilon^2}{6\gamma^2} \mu_{t_0}^{\alpha} \frac{1}{1-(1+\gamma)^{-\alpha}}$. If the condition $\bar{Z}_T \ge \mu_T$ is satisfied then:

$$\mathbb{P}[|\bar{Z}_T - \mu| > \epsilon\mu] \le \frac{1}{\epsilon^2} \mathrm{RelVar}[\bar{Z}_T] \le \frac{1}{6} \tag{29}$$

If $\bar{Z}_T < \mu_T$ then:

$$|0 - \mu| \le \mu \le \mu_T \le \frac{1}{1+\gamma}\epsilon\tau \le \epsilon \max\{\mu, \tau\} \tag{30}$$

Conclusion. Thus, overall if $\hat{Z}$ is the output of AMR:

$$\mathbb{P}[|\hat{Z}-\mu|>\epsilon\max\{\mu,\tau\}] \le \frac{\epsilon^2}{6}\frac{1}{\gamma^2}\mu_{t_0}^{\alpha}\frac{1}{1-(1+\gamma)^{-\alpha}} + 2\frac{(\gamma+1)^2}{\gamma^2}\frac{\epsilon^2}{6}+\frac{1}{3}$$

As we see in the above expression the failure probability is dominated by the $\frac{1}{3}$ term. For example for $\gamma=1,\epsilon=0.2$, $\alpha=1$ we have that the extra term is less than 0.0667. □

1.6. Regular estimator for Gaussian Kernel

Theorem 3. *$\mathcal{Z}_{\text{Gauss}}$ is $(1,\frac{3}{4},\gamma)$-regular and takes preprocessing time/space bounded by $O_{d,\kappa_T,\gamma}(\epsilon^{-3+\frac{1}{4}}\tau^{-\frac{3}{4}}\cdot n)$.*

Proof of Theorem 3. By Lemma 3 and Theorem 1 (Section 2.3 in main paper), (A) holds with $V_t(\mu) := \frac{4e^{\frac{3}{2}}}{\mu}e^{r_t^2-r_t\sqrt{\log(\frac{1}{\mu})}}$. Moreover, since $\forall x\ge y>0$

$$\frac{V_t(y)}{V_t(x)} = \frac{x}{y}e^{-r_t(\sqrt{\log(\frac{1}{y})}-\sqrt{\log(\frac{1}{x})})} \le \left(\frac{x}{y}\right)^{2-1} \tag{31}$$

and $V_t'(x) = -\frac{4e^{\frac{3}{2}}}{x}e^{r_t^2-r_t\sqrt{\log(\frac{1}{\mu})}}(\frac{1}{x}+\frac{r_t}{2\sqrt{\log(\frac{1}{x})}}) < 0$, property (B) holds with $\alpha=1$. Finally,

$$V_t(\mu_{t+1}) = 4e^{\frac{3}{2}}e^{\{\frac{1}{4}-\frac{1}{2}\sqrt{\frac{t+1}{t}}+(1+\frac{1}{t})\}t\log(1+\gamma)} \tag{32}$$

$$= 4e^{\frac{3}{2}}\left(\frac{1}{\mu_t}\right)^{\frac{1}{4}-\frac{1}{2}\sqrt{\frac{t+1}{t}}+(1+\frac{1}{t})} \tag{33}$$

$$\le 4e^{\frac{3}{2}}(1+\gamma)^{1-\frac{1}{\sqrt{2}}}\cdot\left(\frac{1}{\mu_t}\right)^{\frac{3}{4}}, \tag{34}$$

and consequently (C) holds with $\beta=\frac{3}{4}$. Finally, the estimator uses at most $O(\frac{1}{\epsilon^2}V_T(\mu_{T+1}))$ hash tables each taking preprocessing time/space $O_{d,q_T,\gamma}(n)$ space. □

2. Sketching

For any hash table H and a vector $u\in\Delta_n$ (simplex), let $B=B(H)$ denote the number of buckets and $u_{\max}=u_{\max}(H):=\max\{u_{H_i}:i\in[B]\}$ the maximum weight of any hash bucket of H. The precise definition of our Hashing-Based-Sketch is given below in Algorithm 1.

For a fixed H, we can obtain the following bounds on the first two moments of our sketch (S_m,w).

Lemma 5 (Moments)**.** *For the sketch (S_m,w) produced by the HBS procedure it holds that*

$$\mathbb{E}[\text{KDE}_{S_m}^w|H] = \text{KDE}_P^u(q),$$

$$\text{Var}[(\text{KDE}_{S_m}^w)^2|H] \le \frac{1}{m}(Bu_{\max})^{1-\gamma^*}\sum_{i=1}^n k^2(x_i,q)u_i.$$

Algorithm 1 Hashing-Based-Sketch (HBS)

Input: set P, size m, hashing scheme $\mathcal{H}_\nu$, threshold $\tau\in(0,1)$, $u\in\Delta_n$
Sample $h\sim\mathcal{H}_\nu$ and create hash tabel $H=h(P)$.
Set γ according to (35)
$S_m\leftarrow\emptyset$, $w\leftarrow 0\cdot\mathbf{1}_m$, $B\leftarrow B(H)$
for $j=1,\dots,m$ **do**
  Sample hash bucket H_i with probability $\propto u_{H_i}^\gamma$
  Sample a point X_j from H_i with probability $\propto u_j$
  $S_m\leftarrow S_m\cup\{X_j\}$
  $w_j(\gamma,m)\leftarrow\frac{u_{H_i}}{m}\frac{\sum_{i'=1}^B u_{H_{i'}}^\gamma}{u_{H_i}^\gamma}$
Output: (S_m,w)

The above analysis shows that the sketch is always unbiased and that the variance depends on the hash function H only through $(Bu_{\max})^{1-\gamma^*}\ge 1$. We postpone the proof of this lemma after showing how it implies the following theorem.

Theorem 4. *Let H be the hash function sampled by the HBS procedure. For $\epsilon>0$ and $\delta\in[e^{-\frac{6}{\epsilon^2}\frac{u_{\max}}{n\tau}},e^{-\frac{6}{\epsilon^2}})$, let:*

$$\gamma^* = \left\{1-\frac{\log(\frac{\epsilon^2}{6}\log(1/\delta))}{\log(\frac{u_{\max}}{\tau})}\right\}^{\mathbb{I}[B\le(\frac{1}{2})^{\frac{1}{6}}\frac{1}{\tau}]}, \tag{35}$$

$$m = \frac{6}{\epsilon^2}\frac{1}{\tau}(Bu_{\max})^{1-\gamma^*} < \frac{\log(\frac{1}{\delta})}{\tau}. \tag{36}$$

Then (S_m,w) is an $(\epsilon,\frac{1}{6},\tau)$-sketch and if $B\le(\frac{1}{2})^{\frac{1}{6}}\frac{1}{\tau}$ any hash bucket with weight at least τ will have non empty intersection with S_m with probability at least $1-\delta$.

Proof of Theorem 4. Given a hash bucket with weight at least τ, the probability that we sample a point from that bucket is at least:

$$\rho \ge \frac{\tau^\gamma}{B^{1-\gamma}} = \tau\frac{1}{(B\tau)^{1-\gamma}} \tag{37}$$

The probability that we see no point after m independent samples is less than $(1-\rho)^m \le e^{-m\frac{\tau^\gamma}{B^{1-\gamma}}}$ For $m\ge\frac{\log(1/\delta)}{\tau}(B\tau)^{1-\gamma}$ this probability is at most δ. On the other hand by Lemma 5 if $m\ge\frac{6}{\epsilon^2}\frac{1}{\tau}(Bu_{\max})^{1-\gamma}$ we have that $\text{Var}[\text{KDE}_{S_m}^w]\le\frac{\epsilon^2}{6}\mu\tau$. The case $B>2^{-\frac{1}{6}}\frac{1}{\tau}$ is trivial as $\gamma^*=1$. For $B\le 2^{-\frac{1}{6}}\frac{1}{\tau}\Rightarrow u_{\max}\ge\frac{1}{B}\ge\tau 2^{\frac{1}{6}}$. We set γ to make the two lower bounds on m equal,

$$\frac{6}{\epsilon^2}\frac{1}{\tau}(Bu_{\max})^{1-\gamma} = \frac{\log(1/\delta)}{\tau}(B\tau)^{1-\gamma} \tag{38}$$

$$\Leftrightarrow \left(\frac{u_{\max}}{\tau}\right)^{1-\gamma} = \frac{\epsilon^2\log(1/\delta)}{6} \tag{39}$$

$$\Leftrightarrow \gamma = 1-\frac{\log(\frac{\epsilon^2}{6}\log(1/\delta))}{\log(\frac{u_{\max}}{\tau})}. \tag{40}$$

This is strictly less than one for $\log(1/\delta)\frac{\epsilon^2}{6} > 1 \Rightarrow \delta < e^{-\frac{6}{\epsilon^2}}$, and more than zero for $\delta \geq e^{-\frac{6}{\epsilon^2}\frac{u_{\max}}{\tau}}$. Since $u_{\max} \geq \tau 2^{1/6}$ the two inequalities are consistent. Furthermore,

$$
\begin{aligned}
m &= \frac{6}{\tau\epsilon^2} \cdot (Bu_{\max})^{1-\gamma^*} && (41)\\
&= \frac{6}{\tau\epsilon^2} \cdot (Bu_{\max})^{\log(\frac{\epsilon^2\log(1/\delta)}{6})\frac{1}{\log(\frac{u_{\max}}{\tau})}} && (42)\\
&= \frac{6}{\tau\epsilon^2} \cdot e^{\log(\log(1/\delta)\frac{\epsilon^2}{6})\frac{\log(Bu_{\max})}{\log(\frac{u_{\max}}{\tau})}} && (43)\\
&\leq \frac{6}{\tau\epsilon^2} \cdot \left(\log(1/\delta)\frac{\epsilon^2}{6}\right)^{(1-\frac{1}{6}\frac{\log 2}{\log(\frac{u_{\max}}{\tau})})} && (44)\\
&< \frac{\log(1/\delta)}{\tau}. && (45)
\end{aligned}
$$

□

Remark 1. *Observe that $\frac{\log\frac{1}{\delta}}{\tau}$ is the number of samples that random sampling would require in order to have the same property for any bucket with $u_{H_i} \geq \tau$. When $\gamma^* < 1$, our scheme always uses less samples by a factor of $\left(\log(1/\delta)\frac{\epsilon^2}{6}\right)^{\frac{\log(B\tau)}{\log(\frac{u_{\max}}{\tau})}} < 1$.*

Thus, our sketch will have similar variance with random sampling in dense regions of the space but will have better performance for relatively "sparse" regions.

2.1. Proof of Lemma 5

Proof of Lemma 5. Let I be the random hash bucket and X_I the corresponding random point, then for a single point:

$$
\begin{aligned}
\mathbb{E}[\mathrm{KDE}^{w_1}_{\{X_I\}}] &= \mathbb{E}_I[\mathbb{E}_{X_I}[\frac{u_{H_I}}{m}\frac{\sum_{i'=1}^B u_{H_{i'}}^\gamma}{u_{H_I}^\gamma}k(X_I,q)]]\\
&= \mathbb{E}_I[\sum_{j\in H_I}\frac{u_{H_I}}{m}\frac{\sum_{i'=1}^B u_{H_{i'}}^\gamma}{u_{H_I}^\gamma}k(x_j,q)\frac{u_j}{u_{H_I}}]\\
&= \frac{1}{m}\mathbb{E}_I[\frac{\sum_{i'=1}^B u_{H_{i'}}^\gamma}{u_{H_I}^\gamma}\sum_{j\in H_I}k(x_j,q)u_j]\\
&= \frac{1}{m}\sum_{i\in[B]}\sum_{j\in H_i}k(x_j,q)u_j\\
&= \frac{1}{m}\mathrm{KDF}^u_P(q).
\end{aligned}
$$

The first part follows by linearity of expectation. Similarly,

$$
\mathbb{E}[(\mathrm{KDF}^w_{S_m})^2] \leq \sum_{j=1}^m \mathbb{E}[(\mathrm{KDF}^{w_j}_{\{x_j\}})^2] + (\mathrm{KDF}^u_P(q))^2.
$$

By linearity we only have to bound the first term

$$
\begin{aligned}
\mathbb{E}[(\mathrm{KDE}^{w_1}_{\{X_I\}})^2] &= \mathbb{E}_I[\mathbb{E}_{X_I}[(\frac{u_{H_I}}{m}\frac{\sum_{i'=1}^B u_{H_{i'}}^\gamma}{u_{H_I}^\gamma}k(X_I,q))^2]]\\
&= \mathbb{E}_I[\sum_{j\in H_I}(\frac{u_{H_I}}{m}\frac{\sum_{i'=1}^B u_{H_{i'}}^\gamma}{u_{H_I}^\gamma}k(x_j,q))^2\frac{u_j}{u_{H_I}}]\\
&= \mathbb{E}_I[(\frac{\sum_{i'=1}^B u_{H_{i'}}^\gamma}{mu_{H_I}^\gamma})^2 u_{H_I}\sum_{j\in H_I}k^2(x_j,q)u_j]\\
&= \frac{\sum_{i'=1}^B u_{H_{i'}}^\gamma}{m^2}\sum_{i\in[B]}u_{H_I}^{1-\gamma}\sum_{j\in H_I}k^2(x_j,q)u_j\\
&\leq \frac{\sum_{i'=1}^B u_{H_{i'}}^\gamma}{m^2}u_{\max}^{1-\gamma}\sum_{i\in[B]}\sum_{j\in H_I}k^2(x_j,q)u_j\\
&\leq \frac{(Bu_{\max})^{1-\gamma}}{m^2}\sum_{j=1}^n k^2(x_j,q)u_j.
\end{aligned}
$$

The last inequality follows by applying Hölder's inequality with $p = \frac{1}{\gamma}$ and $q = \frac{1}{1-\gamma}$, and due to $u \in \Delta_n$. □

3. Diagnostic and Visualization procedures

In this section, we show how our refined variance bounds along with the adaptive procedure lead to a diagnostic procedure estimating the variance of RS and HBE, as well as to a visualization procedure that gives intuition about the "local structure" of the queries in a given dataset.

3.1. Diagnostic procedure

In order to go beyond worst-case bounds (typically expressed as a function of the query density μ) and provide dataset specific bounds on the variance of different methods (RS and HBE) we use Lemma 4. By setting the coefficients V_{ij} appropriately, we can bound the variance of RS ($V_{ij} = 1$) and HBE ($V_{ij} = \frac{\min\{p(q,x_i),p(q,x_j)\}}{p(q,x_i)^2}$). Unfortunately, evaluating the bound (11) directly over the whole dataset for a single query is no cheaper than evaluating the methods directly.

At a high level, our diagnostic procedure goes around this by evaluating the upper bound for each query q on 'representative sample" $\tilde{S}_0(q)$ instead of P . By doing this for a number T of random queries picked uniformly from the dataset P, we get an estimate of the average relative variance for different methods.

Specifically, given $\tau \in (0,1)$ and $\epsilon \in (0,1)$, for a single query let $\tilde{S}_0$ be the random set produced by the AMR procedure (Algorithm 1, Section 4.2) called with random sampling and define the sets $\tilde{S}_\ell = \tilde{S}_0 \cap S_\ell$ for $\ell \in [4]$ and

their corresponding "densities" $\tilde{\mu}_\ell = \sum_{i\in\tilde{S}_\ell} u_i w_i$. Let

$$\lambda_\epsilon := \arg\max_{\tilde{\mu}_0 \le \lambda \le 1} \left\{ \tilde{\mu}_3 \le \frac{1}{2}(\epsilon\tilde{\mu}_0 - \tilde{\mu}_4) \right\} \tag{46}$$

$$L_\epsilon := \arg\min_{\tilde{\mu}_0 \le L \le 1} \left\{ \tilde{\mu}_1 \le \frac{1}{2}(\epsilon\tilde{\mu}_0 - \tilde{\mu}_4) \right\} \tag{47}$$

be such that $\tilde{\mu}_2 \ge (1-\epsilon)\tilde{\mu}_0$, i.e. most of the mass is captured by the set $\tilde{S}_2$ (that is an spherical annulus for kernels that are decreasing with distance). Since Lemma 4 holds for all $\mu \le \lambda \le L \le 1$, $L_\epsilon, \lambda_\epsilon$ complete the definition of four sets $\tilde{S}_1, \ldots, \tilde{S}_4$ which we use to evaluate (11), and denote by $V_{\text{method}}(q)$ the corresponding bound used with V_{ij} corresponding to a certain estimator, e.g. $\text{method} \in \{\text{RS}, \text{HBE}\}$. Below we give the procedure in pseudo-code.

Algorithm 2 Diagnostic

Input: set P, threshold $\tau \in (\frac{1}{n}, 1)$, accuracy ϵ, $T \ge 1$, collision probability $p(x,y)$ of the hashing scheme $\mathcal{H}_\nu$.
for $t = 1, \ldots, T$ **do**
  $q \leftarrow \text{Random}(P)$ ▷ For each random query
  $(\tilde{S}_0, \tilde{\mu}_0) \leftarrow \text{AMR}(\mathcal{Z}_{RS}(q), \epsilon, \tau)$
  Set $\lambda_\epsilon, L_\epsilon$ using (46) and (47)
  Let V_{RS} be the r.h.s of (11) for $\tilde{S}_0$ and $V_{ij} = 1$.
  Let V_{HBE} be the r.h.s of (11) for $\tilde{S}_0$ and

$$V_{ij} = \frac{\min\{p(q,x_i), p(q,x_j)\}}{p(q,x_i)^2} \tag{48}$$

  $\text{rV}_{\text{RS}}(t) \leftarrow V_{\text{RS}}/\max\{\hat{\mu}, \tau\}^2$
  $\text{rV}_{\text{HBE}}(t) \leftarrow V_{\text{HBE}}/\max\{\hat{\mu}, \tau\}^2$.
Output: $(\text{mean}_T(rV_{\text{RS}}), \text{mean}_T(rV_{\text{HBE}}))$

Remark 2. *We only show the procedure for choosing between RS and HBE with a specific hashing scheme. The same procedure can be used to evaluate a multitude of hashing schemes to select the best one for a given dataset.*

3.2. Visualization procedure

We can use the information from our diagnostics to visualize what is the data set like by aggregating local information for random queries. For a set S, let $r_S = \min_{i\in S} \log(\frac{1}{k(x_i,q)})$ and $R_S = \max_{j\in S} \log(\frac{1}{k(x_j,q)})$. The basis of our visualization is the following fact:

Lemma 6. *Let X be a random sample from S, then $E[k^2(X,q)] \le \exp(R_S - r_S) \cdot \mu_S^2$.*

Proof of Lemma 6. We have that $e^{-R_S} \le k(x_i,q) \le e^{-r_S}$, therefore

$$\mathbb{E}[k^2(X,q)] = \sum_{i\in S} k^2(x_i,q)u_i \tag{49}$$

$$\le e^{-r_S} \sum_{i\in S} k(x_i,q)u_i \frac{\mu_S}{\mu_S} \tag{50}$$

$$\le e^{R_S - r_S}\mu_S^2 \tag{51}$$

where in the last part we used $\mu_S \ge e^{-R_S}$. □

Thus if we plot an annulus of width $w_S = R_S - r_s$ then e^{w_S} is an estimate of the relative variance for RS! The visualization procedure when given a sequence of T pairs of numbers (λ_t, L_t) for $t \in [T]$ (produced by the diagnostic procedure) plots overlapping annuli around the origin representing the queries. Since often the ratio $\max_{i,j\in S} \frac{k(x_i,q)}{k(x_j,q)}$ is refered to as the *condition number* of the set S, we call our procedure the Log-Condition plot.

Algorithm 3 Log-Condition Plot

Input: $\{(\lambda_t, L_t)\}_{t\in[T]}$.
for $t = 1, \ldots, T$ **do** ▷ For each query
  $r_t \leftarrow \log(1/L_t)$,
  $R_t \leftarrow \log(1/\lambda_t)$
  draw 2D-annulus(r_t, R_t)
Output: figure with overlapping annuli.

Remark 3. *In the specific case of the Laplace (exponential) kernel, the radii we are plotting correspond to actual distances.*

4. Synthetic benchmarks

In this section, we introduce a general procedure to create tunable synthetic datasets that exhibit different local structure around the query. We then show how to use this procedure as a building block to create two different family of instances with specific characteristics aimed to test kernel density evaluation methods.

4.1. $(\mu, D, n, s, d, \sigma)$-Instance

Since the problem of kernel density is query dependent and the kernel typically depends only on the distance, we shall always assume that the query point is at the origin $q = 0 \in \mathbb{R}^d$.

We further assume that the kernel is an invertible function of the distance $K(r) \in [0,1]$ and let $K^{-1}(\mu) \in [0,\infty)$ be the inverse function. For example, the exponential kernel is given by $K(r) = e^{-r}$ and the inverse function is given by $K^{-1}(\mu) = \log(\frac{1}{\mu})$.

The dataset is created with points lying in D different directions and s distance scales (equally spaced between 0 and

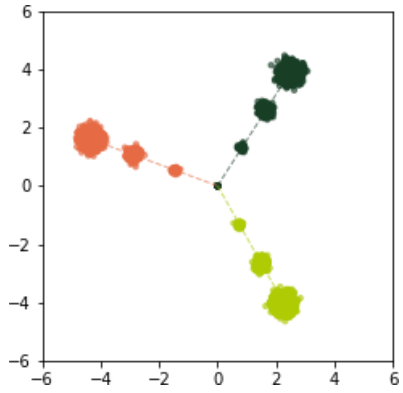


Figure 2. ($\mu = 0.01$, $D = 3$, $s = 4$, $d = 2$, $\sigma = 0.05$)-Instance. Each of the $D = 3$ directions is coded with a different color.

$R = K^{-1}(\mu)$) such that the *contribution from each direction and scale* to the kernel density at the origin is *equal*. To achieve this the number of points n_j placed a at the j-th distance scale r_j is given by

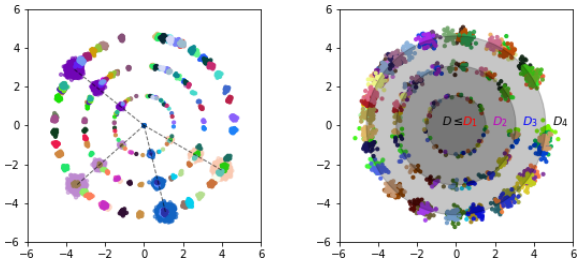
$$n_\ell := \lfloor n \frac{\mu}{K(r_j)} \rfloor. \tag{52}$$

The reasoning behind this design choice is to make sure that we have diversity in the distance scales that matter in the problem, so not to favor a particular class of methods (e.g. random sampling , nearest-neighbor based). Also, placing the points on the same direction makes the instance more difficult for HBE as the variance in (6) increases with the ratio $\frac{\mathbb{P}[h(i)=h(j)=h(q)]}{\mathbb{P}[h(i)=h(q)]^2}$, that expresses how correlated the values $\{h(i), h(j), h(q)\}$ are. We give an example visualization of such data sets in 2 dimensions in Figure 2. The detailed procedure is described below (Algorithm 4).

Algorithm 4 $(\mu, D, n, s, d, \sigma)$-Instance

Input: $\mu \in [\frac{1}{n}, 1]$, $D \geq 1$, $n \geq 1$, $s \geq 2$, $d \geq 1$, $\sigma \geq 0$, kernel K, inverse K^{-1}.
$R \leftarrow K^{-1}(\mu)$, $r_0 \leftarrow K^{-1}(1)$, $P \leftarrow \emptyset$.
for $j = 0, \ldots, s-1$ **do**
  $r_{j+1} \leftarrow \frac{R-r_0}{s-1} j + r_0$ ▷ distances for each D
  $n_{j+1} \leftarrow \lfloor n \frac{\mu}{K(r_{j+1})} \rfloor$ ▷ points at each distance
for $i = 1, \ldots, D$ **do**
  $v_i \leftarrow \frac{g_i}{\|g_i\|}$ with $g_i \sim \mathcal{N}(0, I_d)$. ▷ random direction
  for j=1,..., s **do** ▷ For each distance scale
    for $\ell = 1, \ldots, n_j$ **do** ▷ generate a "cluster"
      $g_{ij\ell} \sim \mathcal{N}(0, I_d)$
      $x_{ij\ell} \leftarrow s_j v_i + \frac{\sigma}{\sqrt{d}} s_j g_{ij\ell}$
      $P \leftarrow P \cup \{x_{ij\ell}\}$
Output: Set of points P

Remark 4. *If $D \ll n$ this class of instances becomes highly structured with a small number of tightly knit "clusters" (Figure 2). One would expect in this case, space-partioning methods to perform well. At the same time by Lemma 4, this type of instances are the ones that maximize the variance of both HBE $(s \geq 1)$ and RS $(s > 1)$.*



(a) "worst-case" instance (b) D-structured instance

Figure 3. The two family of instances for $d = 2$.

Remark 5. *On the other hand if $D \gg n$ the instances become spread out (especially in high dimensions). This type of instances are ideal for sampling based methods when $s = 1$, and difficult for space-partitioning methods.*

Based on the above remarks we propose the following subclass of instances.

4.2. "Worst-case" instance

In order to create an instance that is *hard for all methods* we take a union of the two extremes $D \ll n$ and $D \gg n$. We call such instances "worst-case" as there does not seem to be a single type of structure that one can exploit, and these type of instances realize the worst-case variance bounds for both HBE and RS. In particular, if we want to generate an instance with N points, we first set D a small constant and $n = \Theta(N)$ and take the union of such a dataset with another using $D = \Omega(N^{1-o(1)})$ and $n = O(N^{o(1)})$. An example of such a dataset is given in 3(a).

4.3. D-structured instance

"Worst-case" instances are aimed to be difficult for any kernel evaluation method. In order to create instances that have more varied structure, we use our basic method to create a single parameter family of instance by fixing N, μ, σ, s, d and setting $n = \frac{N}{D}$. We call this family of instances as *D-structured*. As one increases D, two things happen:

- The number of directions (clusters) increases.
- $n = \frac{N}{D}$ decreases and hence certain distance scales disappear. By (52), if $n\mu < K(r_j) \Rightarrow D_j > \frac{N\mu}{K(r_j)}$ then distance scale j will have no points assigned to it.

Hence, for this family when $D \ll \frac{N}{D} \leftrightarrow D \ll \sqrt{N}$ the instances are highly structured and we expect space-

Table 1. Preprocessing time ($init$) and total query time ($query$) on 10K random queries for additional datasets. All runtime measurements are reported in seconds.

Dataset	Time	RS	HBE	ASKIT	FigTree
higgs	$init$	0	141	25505	$>$ 1day
	$query$	**6**	18	1966	$>$ 1day
hep	$init$	0	138	23421	$>$ 20 hours
	$query$	**6**	11	1581	$>$ 1day
susy	$init$	0	67	5326	3245
	$query$	18	**12**	$>$ 9756	5392
home	$init$	0	11	237	7
	$query$	2369	**17**	376	33
mnist	$init$	0	211	14	437
	$query$	**168**	389	?	1823

Table 2. Preprocessing time (in seconds) for clustering test.

n	D	HBE	FigTree	ASKIT
$500K$	1	192	2	113
$50K$	10	20	3	105
$5K$	100	16	16	105
500	1000	19	174	104
50	10000	39	1516	102
5	100000	334	0.3	101

partitioning methods to perform well. On the other extreme as D increases and different distance scales start to die out (Figure 3(b)) the performance of random sampling keeps improving until there is only one (the outer) distance scale, where random sampling will be extremely efficient. On the other hand HBE's will have roughly similar performance on the two extremes as both correspond to worst-case datasets for scale-free estimators with $\beta = 1/2$, and will show slight improvement in between $1 \ll D \ll n$. This picture is confirmed by our experiments.

5. Experiments

5.1. Datasets

We provide detailed descriptions of the datasets as well as the bandwidth used for the kernel density evaluation in Table 3. We also include specifications for the additional datasets acquired from LIBSVM (Chang & Lin, 2011) and the UCI Machine Learning Repository (Dheeru & Karra Taniskidou, 2017) that were used to evaluate the accuracy of the diagnostic procedure.

We selected the top eight datasets in Table 3 for density evaluation in the main paper as they are the largest, most complex datasets in our collection. We provide additional density evaluation results in Table 1 for datasets with comparable sizes or dimensions to the ones reported in the main paper. For higgs and hep, FigTree failed to finish the evaluation within a day. Given the performance of RS on these datasets, we don't expect FigTree to achieve better performance even if the query returns successfully. For mnist, we were not able to get ASKIT to achieve relative error below 1 even after trying parameters that span a few orders of magnitude; this is potentially caused by the high-dimensionality and sparsity of this dataset.

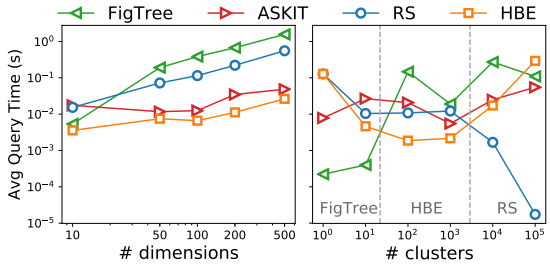


Figure 4. Results from the synthetic experiment (repeat of results in the main paper for easy reference).

5.2. Synthetic Experiment

For the clustering test, we set $\mu = 0.001, s = 4, d = 100, \sigma = 0.01, N = 500K$. The varying parameters are the number of clusters (D) and the number of points per cluster n. We report preprocessing time (in seconds) for all methods in Table 2. The ordering of methods according to preprocessing time largely follows the that of query time.

As discussed in Section 4.3, for the D-structured instances as the number of points per cluster n decreases, smaller distance scales start to disappear due to (52). Let D_i be the threshold such that for $D > D_i$, there are no-points in scale i. The corresponding numbers for our experiment is roughly $D1 = 500$ (at distance 0) , $D2 = 1077$, $D3 = 10K$, $D4 = N = 500K$. In particular, only a single distance scale remains when $D = 100K > D3$, a set up in which RS is orders of magnitude more efficient than alternative methods (Figure 4 right).

5.3. Sketching Experiment

In this section, we evaluate the quality of the proposed hashing-based sketch. As baselines, we compare against sparse kernel approximation (SKA) (Cortes & Scott, 2017), kernel herding algorithm (Herding) (Chen et al., 2010) and uniform sampling. To control for the difference in the complexity (Table 4), we compare the approximation error achieved by sketches of the same size (s) under the same compute budget ($2n$, where n is dataset size). We describe the detailed setup below, including necessary modifications to meet the computational constraints.

Table 3. Specifications of real-world datasets.

Dataset	N	d	σ	Description
census	$2.5M$	68	3.46	Samples from 1900 US census.
TMY3	$1.8M$	8	0.43	Hourly energy load profiles for US references buildings.
TIMIT	$1M$	440	10.97	Speech data for acoustic-phonetic studies. First 1M data points used.
SVHN	$630K$	3072	28.16	Google Street View house numbers. Raw pixel values of 32x32 images.
covertype	$581K$	54	2.25	Cartographic variables for predicting forest cover type.
MSD	$463K$	90	4.92	Audio features of popular songs.
GloVe	$400K$	100	4.99	Pre-trained word vectors from Wikipedia 2014 + Giga 5 word. 5. 6B tokens, 400K vocab.
ALOI	$108K$	128	3.89	Color image collection of 1000 small objects. Each image is represented by a 128 dimensional SIFT feature vector.
higgs	$11M$	28	3.41	Signatures of Higgs bosons from Monte Carlo simulations.
hep	$10.5M$	27	3.36	Signatures of high energy physics particles (Monte Carlo simulations).
susy	$5M$	18	2.24	Signatures of supersymmetric particles (Monte Carlo simulations).
home	$969K$	10	0.53	Home gas sensor measurements.
skin	$245K$	3	0.24	Skin Segmentation dataset.
ijcnn	$142K$	22	0.90	IJCNN 2001 Neural Network Competition.
acoustic	$79K$	50	1.15	Vehicle classification in distributed sensor networks.
mnist	$70K$	784	11.15	28x28 images of handwritten digits.
corel	$68K$	32	1.04	Image dataset, with color histograms as features.
sensorless	$59K$	48	2.29	Dataset for sensorless drive diagnosis.
codrna	$59K$	8	1.13	Detection of non-coding RNAs.
shuttle	$44K$	9	0.62	Space shuttle flight sensors.
poker	$25K$	10	1.37	Poker hand dataset.
cadata	$21K$	8	0.62	California housing prices.

Table 4. Overview of algorithm complexity and parameter choice for the sketching experiment (n: dataset size, s: sketch size, T: number of hash tables, m: sample size for herding).

Algorithm	Complexity	Parameters
HBS	$O(n'T + s)$	$n' = \frac{2n}{5}, T = 5$
SKA	$O(n_c s_c + s_c^3)$	$s_c = n^{\frac{1}{3}}, n_c = n^{\frac{2}{3}}$
Herding	$O(n_h m + n_h s)$	$m = s, n_h = \frac{n}{s}$

HBS. For HBS, we used 5 hash tables, each hashing a subset of $\frac{2}{5}n$ points in the dataset. In practice, we found that varying this small constant on the number of hash tables does not have a noticeable impact on the performance.

SKA. SKA (Algorithm 5) produces the sketch by greedily finding s points in the dataset that minimizes the maximum distance. The associated weights are given by solving an equation that involves the kernel matrix of the selected points. SKA's complexity $O(ns + s^3)$ is dominated by the matrix inversion procedure used to solve the kernel matrix equation. To ensure that SKA is able to match the sketch size of alternative methods under the compute budget of $2n$, we augment SKA with random samples when necessary:

- If the target sketch size is smaller than $n^{\frac{1}{3}}$ ($s < n^{\frac{1}{3}}$), we use SKA to produce a sketch of size s from a subsample of n/s data points.
- For $s > n^{\frac{1}{3}}$, we use SKA to produce a sketch of size $s_c = n^{\frac{1}{3}}$ from a subsample of $n_c = n^{\frac{2}{3}}$ data points. We match the difference in sketch size by taking an additional $s - s_c$ random samples from the remaining $n - n_c$ data points that were not used for the SKA sketch. The final estimate is a weighted average between the SKA sketch and the uniform sketch: $\frac{1}{s_c}$ for SKA and $(1 - \frac{1}{s_c})$ for uniform, where the weights are determined by the size of the two sketches.

The modification uses SKA as a form of regularization on random samples. Since SKA iteratively selects points that are farthest away from the current set, the resulting sketch is helpful in predicting the "sparser" regions of the space. These sparser regions, in turn, are the ones that survive in the $n^{2/3}$ random sample of the dataset (sub-sampling with probability $n^{-1/3}$), therefore SKA naturally includes points from "sparse" clusters of size $\Omega(n^{1/3})$ in the original set.

Herding. The kernel Herding algorithm (Algorithm 6) first estimates the density of the dataset via random sam-

pling; the sketch is then produced by iteratively selecting points with maximum residual density. The algorithm has a complexity of $O(nm + ns)$, where m stands for the sample size used to produce the initial density estimates.

To keep Herding under the same $2n$ compute budget, we downsample the dataset to size $n_h = \frac{n}{s}$, and use $m = s$ samples to estimate the initial density. This means that, the larger the sketch size is, the less accurate the initial density estimate is. As a result, we observe degrading performance at larger sketch sizes $s = \Omega(\sqrt{n})$.

Algorithm 5 Sparse Kernel Approximation (SKA)

Input: set P, kernel K, size s.
$S = \{x_1, \dots, x_s\} \leftarrow$ Greedy-kcenter(P, s)
$K \in \mathbb{R}^{s \times s}$ with $K_{ij} \leftarrow k(x_i, x_j)$ for $x_i, x_j \in S$.
$y \in \mathbb{R}^s$ with $y_i \leftarrow \mathrm{KDE}_P^w(x_i)$ for $x_i \in S$.
Let $\hat{w}$ be a solution to $K\hat{w} = y$.
Output: $(S, \hat{w})$

Algorithm 6 Approximate Kernel Herding (AKH)

Input: set P, kernel K, size s, samples m.
for $i = 1, \dots, |P|$ **do**
  $P_i \leftarrow \mathrm{Random}(P, m)$. ▷ random set of m points
  $d_i \leftarrow \mathrm{KDE}_{P_i}(x_i)$ ▷ estimate of the density
$S_0 \leftarrow \emptyset$ ▷ initialization
for $t = 1, \dots, s$ **do**
  $j^* \leftarrow \arg\max_{i \in [n]} \{d_i - \mathrm{KDE}_{S_{t-1}}(x_i)\}$ ▷ greedy
  $S_t \leftarrow S_{t-1} \cup \{x_j^*\}$ ▷ add point to the set
Output: $(S_s, \frac{1}{s}\mathbf{1}_s))$ ▷ return the sketch

Results. Figure 5 reports the relative error on random queries (left), i.e. uniformly random points from the dataset, and low-density queries (right), uniformly random points of the dataset with density around a threshold τ. Uniform, SKA and HBS achieve similar mean error on random queries, while the latter two have improved performance on low-density queries, with HBS performing slightly better than SKA. By design, HBS has similar performance with random sampling on average, but performs better on relatively "sparse" regions due to the theoretical guarantee that buckets with weight at least τ are sampled with high probability. Combining SKA with random samples initially results in performance degradation but eventually acts as a form of regularization improving upon random. Kernel Herding is competitive only for a small number of points in the sketch.

Overhead reduction. In Table 5, we report on the estimated preprocessing runtime reduction enabled by HBS for the density estimation results reported in Table 1 of the main paper. The estimates are calculated by the dividing the number of data points hashed according to the original

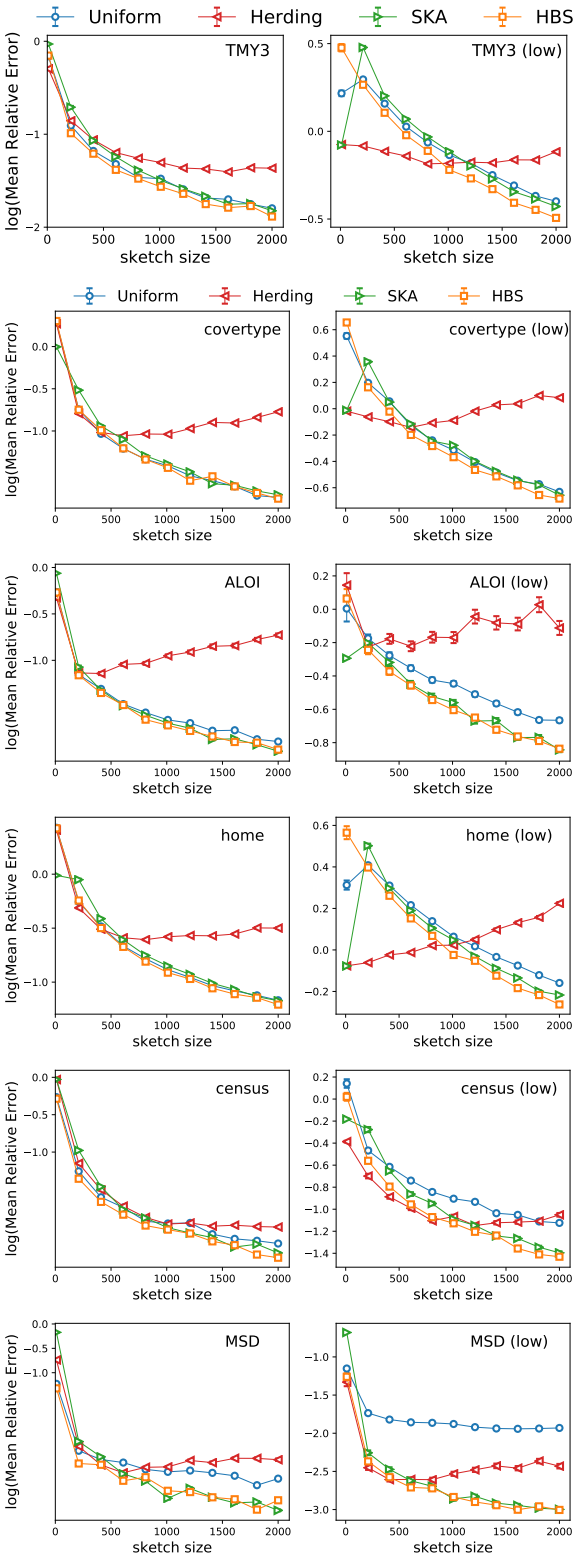


Figure 5. Sketching results on random and low-density queries.

Table 5. Estimated overhead reduction enabled by HBS.

	census	TMY3	TIMIT	SVHN	covertype	MSD	GloVe	ALOI
Reduction (est.)	958×	755×	821×	659×	554×	607×	595×	303×

HBE procedure by the number of data points hashed after enabling HBS.

5.4. Visualizations of real-world data sets

Our Log-Condition plots use circles with radius r to represent points with weights roughly e^{-r} (roughly at distance $\sqrt{r}$ for the Gaussian kernel). The visualizations are generated by plotting overlapping annuli around the origin that represent a random queries from the dataset, such that the width of the annulus roughly corresponds to the log of the relative variance of random sampling.

We observe two distinctive types of visualizations. Datasets like census exhibit dense inner circles, meaning that a small number of points close to the query contribute significantly towards the density. To estimate the density accurately, one must sample from these small clusters, which HBE does better than RS. In contrast, datasets like MSD exhibit more weight on the outer circles, meaning that a large number of "far" points is the main source of density. Random sampling has a good chance of seeing these "far" points, and therefore, tends to perform better on such datasets. The top two plots in Figure 6 amplify these observations on synthetic datasets with highly clustered/scattered structures. RS performs better for all datasets in the right column except for SVHN.

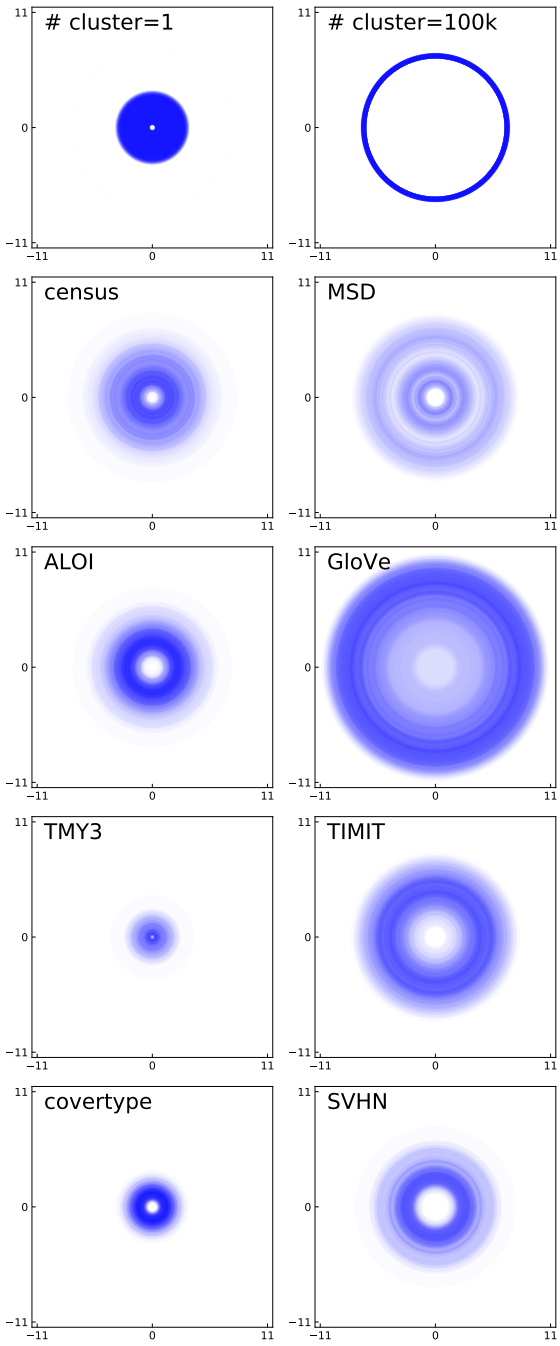


Figure 6. Visualizations of datasets. The top row shows two extreme cases of highly clustered (# cluster=1) versus highly scattered (#cluster=100k) datasets. RS performs better for all datasets in the right column expect for SVHN.

References

Chang, C.-C. and Lin, C.-J. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Chen, Y., Welling, M., and Smola, A. Super-samples from Kernel Herding. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, UAI'10, pp. 109–116, Arlington, Virginia, United States, 2010. AUAI Press. ISBN 978-0-9749039-6-5.

Cortes, E. C. and Scott, C. Sparse Approximation of a Kernel Mean. *Trans. Sig. Proc.*, 65(5):1310–1323, March 2017. ISSN 1053-587X.

Dheeru, D. and Karra Taniskidou, E. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.