Thinformer: Guaranteed Attention Approximation via Low-Rank Thinning

Annabelle Michael Carrell [1] **Albert Gong** [2] **Abhishek Shetty** [3] **Raaz Dwivedi** [2] **Lester Mackey** [4]

Abstract

The goal in thinning is to summarize a dataset using a small set of representative points. Remarkably, sub-Gaussian thinning algorithms can match the quality of uniform subsampling while substantially reducing the number of summary points. However, existing guarantees cover only a restricted range of distributions and kernel-based quality measures and suffer from pessimistic dimension dependence. To address these deficiencies, we introduce a new low-rank analysis of sub-Gaussian thinning that applies to any distribution and any kernel, guaranteeing high-quality compression whenever the kernel or data matrix is approximately low-rank. To demonstrate the broad applicability of the techniques, we design practical sub-Gaussian thinning approaches that improve upon the best known guarantees for approximating attention in transformers.

1. Introduction

This work is about thinning, finding a small set of representative points to summarize a larger dataset. State-of-the-art thinning techniques provably improve upon uniform subsampling but only for restricted classes of kernel-based quality measures and with pessimistic dependence on the data dimension (see, e.g., Harvey & Samadi, 2014; Phillips & Tai, 2020; Alweiss et al., 2021; Dwivedi & Mackey, 2024; 2022; Shetty et al., 2022; Li et al., 2024). We introduce a new analysis for sub-Gaussian thinning algorithms that applies to any kernel and shows that one can efficiently identify a better-than-uniform set of representative points whenever the kernel or data matrix is nearly low-rank. This opens the door to a variety of impactful applications, including approximate dot-product attention in transformers.

[1]University of Cambridge [2]Cornell Tech [3]Massachusetts Institute of Technology [4]Microsoft Research New England. Correspondence to: Annabelle Carrell <ac2411@cam.ac.uk>, Albert Gong <agong@cs.cornell.edu>, Abhishek Shetty <ashetty1995@gmail.com>, Raaz Dwivedi <dwivedi@cornell.edu>, Lester Mackey <lmackey@microsoft.com>.

Notation. For each $n \in \mathbb{N}$ and $a, b \in \mathbb{R}$, we define $[n] \triangleq \{1, \dots, n\}$, $a \wedge b \triangleq \min(a, b)$, and $a \vee b \triangleq \max(a, b)$. We let $\|\mathbf{A}\|_{\mathrm{op}}$, $\|\mathbf{A}\|_{\max}$, and $\|\mathbf{A}\|_{2,\infty}$ respectively represent the maximum singular value, absolute entry, and row Euclidean norm of a matrix $\mathbf{A}$. We also define the Euclidean norm balls $\mathbb{B}^m \triangleq \{\boldsymbol{u} \in \mathbb{R}^m : \|\boldsymbol{u}\|_2 \leq 1\}$ and $\mathbb{B}^m(R) \triangleq R\mathbb{B}^m$ for each $m \in \mathbb{N}$ and $R > 0$. For an event $\mathcal{E}$ and an integrable random variable X, we define $\mathbb{E}_{\mathcal{E}}[X] \triangleq \mathbb{E}[X \cdot \mathbf{1}[\mathcal{E}]]$. We write $a_n \leq \widetilde{O}(b_n)$ to mean $a_n \leq b_n \operatorname{polylog}(n)$.

2. Sub-Gaussian Thinning

Consider a fixed collection of n_{in} input points $\mathcal{X}_{\mathrm{in}}$ belonging to a potentially larger universe of datapoints $\mathcal{X} \triangleq \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}$. The aim of a thinning algorithm is to select n_{out} points from $\mathcal{X}_{\mathrm{in}}$ that together accurately summarize $\mathcal{X}_{\mathrm{in}}$. This is formalized by the following definition.

Definition 1 (Thinning algorithms). *A* thinning algorithm ALG *takes as input $\mathcal{X}_{\mathrm{in}}$ and returns a possibly random subset $\mathcal{X}_{\mathrm{out}}$ of size n_{out}. We denote the input and output empirical distributions by $\mathbb{P}_{\mathrm{in}} \triangleq \frac{1}{n_{\mathrm{in}}} \sum_{\boldsymbol{x} \in \mathcal{X}_{\mathrm{in}}} \boldsymbol{\delta}_{\boldsymbol{x}}$ and $\mathbb{P}_{\mathrm{out}} \triangleq \frac{1}{n_{\mathrm{out}}} \sum_{\boldsymbol{x} \in \mathcal{X}_{\mathrm{out}}} \boldsymbol{\delta}_{\boldsymbol{x}}$ and define the induced probability vectors $\boldsymbol{p}_{\mathrm{in}}, \boldsymbol{p}_{\mathrm{out}} \in \boldsymbol{\Delta}_{n-1}$ over the indices $[n]$ by*

$$\boldsymbol{p}_{\mathrm{in},i} = \tfrac{\mathbf{1}[\boldsymbol{x}_i \in \mathcal{X}_{\mathrm{in}}]}{n_{\mathrm{in}}} \text{ and } \boldsymbol{p}_{\mathrm{out},i} = \tfrac{\mathbf{1}[\boldsymbol{x}_i \in \mathcal{X}_{\mathrm{out}}]}{n_{\mathrm{out}}} \text{ for all } i \in [n].$$

When $\mathcal{X} \subset \mathbb{R}^d$, we use $\mathbf{X} \triangleq [\boldsymbol{x}_1, \dots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n \times d}$ to denote the input point matrix so that

$$\mathbb{E}_{\boldsymbol{x} \sim \mathbb{P}_{\mathrm{in}}}[\boldsymbol{x}] = \mathbf{X}^\top \boldsymbol{p}_{\mathrm{in}} \quad \text{and} \quad \mathbb{E}_{\boldsymbol{x} \sim \mathbb{P}_{\mathrm{out}}}[\boldsymbol{x}] = \mathbf{X}^\top \boldsymbol{p}_{\mathrm{out}}.$$

We will use the following measure of summarization quality.

Definition 2 (Kernel max seminorm). *Given two distributions μ, ν and a reproducing kernel $\mathbf{k}$ (Steinwart & Christmann, 2008, Def. 4.18), and any indices $\mathcal{I} \subseteq [n]$, we further define the* kernel max seminorm (KMS)

$$\|\mathbf{K}(\boldsymbol{p}_{\mathrm{in}} - \boldsymbol{p}_{\mathrm{out}})\|_{\mathcal{I}} \triangleq \max_{i \in \mathcal{I}} |\boldsymbol{e}_i^\top \mathbf{K}(\boldsymbol{p}_{\mathrm{in}} - \boldsymbol{p}_{\mathrm{out}})|. \quad (1)$$

A common strategy for bounding the error of a thinning algorithm is to establish its sub-Gaussianity.

Table 1: **Examples of $(\mathbf{K}, \nu, \delta)$-sub-Gaussian thinning algorithms.** For input size n_{in}, output size $n_{\text{out}} \geq \sqrt{n_{\text{in}}}$, and $\|\mathbf{K}\|_{\max} = 1$ we report each sub-Gaussian parameter ν and runtime up to constants independent of $(n_{\text{in}}, n_{\text{out}}, \delta, \mathbf{K})$.

Algorithm	SUBSAMPLING Prop. B.1	KH(δ) Prop. B.2	KH-COMPRESS(δ) Prop. B.5	GS-THIN Prop. B.6	GS-COMPRESS Prop. B.10
Sub-Gaussian parameter ν	$\frac{1}{\sqrt{n_{\text{out}}}}$	$\frac{\sqrt{\log(n_{\text{out}}/\delta)}}{n_{\text{out}}}$	$\frac{\sqrt{\log(n_{\text{out}})\log(n_{\text{out}}/\delta)}}{n_{\text{out}}}$	$\frac{1}{n_{\text{out}}}$	$\frac{\sqrt{\log(n_{\text{out}})}}{n_{\text{out}}}$
Runtime	n_{out}	n_{in}^2	n_{out}^2	n_{in}^3	n_{out}^3

Definition 3 (Sub-Gaussian thinning algorithm). *We write* ALG $\in \mathcal{G}_{\nu,\delta}(\mathbf{K})$ *and say* ALG *is* $(\mathbf{K}, \nu, \delta)$-sub-Gaussian*, if* ALG *is a thinning algorithm,* $\mathbf{K}$ *is a symmetric positive semidefinite (SPSD) matrix,* $\nu > 0$*,* $\delta \in [0, 1)$*, and there exists an event* $\mathcal{E}$ *with probability at least* $1 - \delta/2$ *such that, the input and output probability vectors satisfy*

$$\mathbb{E}_{\mathcal{E}}[\exp(\langle \boldsymbol{u}, \mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\rangle)] \leq \exp\left(\tfrac{\nu^2}{2}\boldsymbol{u}^\top \mathbf{K}\boldsymbol{u}\right), \forall \boldsymbol{u} \in \mathbb{R}^n.$$

Here, the sub-Gaussian parameter ν controls the summarization quality of the thinning algorithm, and we see from Tab. 1 that a variety of practical thinning algorithms are $(\mathbf{K}, \nu, \delta)$-sub-Gaussian for varying levels of ν.

2.1. Examples of sub-Gaussian thinning algorithms

Perhaps the simplest sub-Gaussian thinning algorithm is *uniform subsampling*: by Prop. B.1, selecting n_{out} points from $\mathcal{X}_{\text{in}}$ uniformly at random (without replacement) is $(\mathbf{K}, \nu, 0)$-sub-Gaussian with $\nu = \sqrt{\|\mathbf{K}\|_{\max}}/\sqrt{n_{\text{out}}}$. Unfortunately, uniform subsampling suffers from relatively poor summarization quality. As we prove in App. B.1.1, its KMS is $\Omega(1/\sqrt{n_{\text{out}}})$ meaning that $n_{\text{out}} = 10000$ points are needed to achieve 1% relative error.

Proposition 1 (Quality of uniform subsampling). *For any* $\mathcal{I} \subseteq [n]$*, a uniformly subsampled thinning satisfies*

$$\mathbb{E}[\|\mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\|_{\mathcal{I}}^2] \geq \tfrac{1}{n_{\text{out}}} \tfrac{n_{\text{in}} - n_{\text{out}}}{n_{\text{in}} - 1} \max_{i \in \mathcal{I}} C_{\mathbf{K}\boldsymbol{e}_i\boldsymbol{e}_i^\top \mathbf{K}}$$

for any SPSD $\mathbf{K}$ *with* $C_{\mathbf{K}} \triangleq \sum_{i=1}^n \boldsymbol{p}_{\text{in},i}\mathbf{K}_{ii} - \boldsymbol{p}_{\text{in}}^\top \mathbf{K}\boldsymbol{p}_{\text{in}}$.

Fortunately, uniform subsampling is not the only sub-Gaussian thinning algorithm available. For example, the Kernel Halving (KH(δ)) algorithm of Dwivedi & Mackey (2024) provides a substantially smaller sub-Gaussian parameter, $\nu = O(\sqrt{\log(n_{\text{out}}/\delta)}/n_{\text{out}})$, at the cost of n_{in}^2 runtime, while the KH-COMPRESS(δ) algorithm of Shetty et al. (2022, Ex. 3) delivers $\nu = O(\sqrt{\log(n_{\text{out}})\log(n_{\text{out}}/\delta)}/n_{\text{out}})$ in only n_{out}^2 time. We derive simplified versions of these algorithms with identical sub-Gaussian constants in Apps. B.2 and B.5 and a linear-kernel variant (LKH(δ)) with $n_{\text{in}}d$ runtime in App. B.3. To round out our set of examples, we show in App. B.6.1 that two new thinning algorithms based on the Gram-Schmidt walk of Bansal et al. (2018) yield even smaller ν at the cost of increased runtime. We call these algorithms Gram-Schmidt Thinning (GS-THIN) and GS-COMPRESS.

3. Low-rank Sub-Gaussian Thinning

One might hope that the improved sub-Gaussian constants of Tab. 1 would also translate into improved quality metrics. Our main result, proved in App. C, shows that this is indeed the case if the inputs are approximately low-rank.

Theorem 1 (Low-rank sub-Gaussian thinning). *Fix any* $\delta' \in (0, 1)$*,* $r \leq n$*, and* $\mathcal{I} \subseteq [n]$*. If* ALG $\in \mathcal{G}_{\nu,\delta}(\mathbf{K})$*, then, with probability at least* $1 - \delta/2 - \delta'$*:*

$$\|\mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\|_{\mathcal{I}} \leq \nu D_{\mathcal{I}} \sqrt{2\log(\tfrac{2|\mathcal{I}|}{\delta'})}. \tag{2}$$

for $D_{\mathcal{I}} \triangleq \max_{i \in \mathcal{I}} \sqrt{\mathbf{K}_{ii}}$.

Suppose that, in addition, $\mathcal{X} \subset \mathbb{R}^d$ *and* $|\mathbf{K}_{il} - \mathbf{K}_{jl}| \leq L_{\mathbf{K}}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2$ *for some* $L_{\mathbf{K}} > 0$ *and all* $i, j \in \mathcal{I}$ *and* $l \in \text{supp}(\boldsymbol{p}_{\text{in}})$*. Then, with probability at least* $1 - \delta/2 - \delta'$*,*

$$\begin{aligned}\|\mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\|_{\mathcal{I}} &\leq \nu D_{\mathcal{I}} \sqrt{2\log(4/\delta')}(1 + \tfrac{32}{\sqrt{3}}) \\ &\quad + \nu D_{\mathcal{I}}\, 32\sqrt{\tfrac{2}{3}\operatorname{rank}(\mathbf{X}_{\mathcal{I}})\log(\tfrac{3e^2 R_{\mathcal{I}} L_{\mathbf{K}}}{D_{\mathcal{I}}^2 \wedge (R_{\mathcal{I}} L_{\mathbf{K}})})}\end{aligned} \tag{3}$$

for $R_{\mathcal{I}} \triangleq \max_{i \in \mathcal{I}} \|\boldsymbol{x}_i\|_2$ *and* $\mathbf{X}_{\mathcal{I}} \triangleq [\boldsymbol{x}_i]_{i \in \mathcal{I}}^\top$.

Let us unpack the two components of this result. First, Thm. 1 provides a high-probability $O(\nu\sqrt{\log(|\mathcal{I}|)})$ bound (2) on the KMS for any kernel and any sub-Gaussian thinning algorithm on any space. In particular, the non-uniform algorithms of Tab. 1 all enjoy $O(\log(n_{\text{out}})\sqrt{\log(|\mathcal{I}|)}/n_{\text{out}})$ KMS, a significant improvement over the $\Omega(1/\sqrt{n_{\text{out}}})$ KMS of uniform subsampling. Second, Thm. 1 provides a refined $O(\nu\sqrt{\operatorname{rank}(\mathbf{X}_{\mathcal{I}})\log(R_{\mathcal{I}}L_{\mathbf{K}})})$ bound (3) on KMS for datapoints in $\mathbb{R}^d$. For bounded data, this trades an explicit dependence on the number of query points $|\mathcal{I}|$ for a rank factor that is never larger (and sometimes significantly smaller) than d. We will make use of these results when approximating dot-product attention in Sec. 4.

4. Approximating Attention

Dot-product attention lies at the heart of the transformer neural network architecture that has revolutionized natural language processing, computer vision, and speech recognition over the last decade (Vaswani et al., 2017; Dosovitskiy et al., 2021; Dong et al., 2018). Given a collection of query, key, and value vectors $(\boldsymbol{q}_i, \boldsymbol{k}_i, \boldsymbol{v}_i)_{i=1}^n$ each in $\mathbb{R}^d$, dot-product attention computes the *softmax matrix*

$$\mathbf{T} \triangleq \textsc{Attention}((\boldsymbol{q}_i)_{i=1}^n, (\boldsymbol{k}_j, \boldsymbol{v}_j)_{j=1}^n) \triangleq \mathbf{D}^{-1}\mathbf{A}\mathbf{V} \quad (4)$$

for $\mathbf{A}_{ij} \triangleq \exp(\frac{\langle \boldsymbol{q}_i, \boldsymbol{k}_j \rangle}{\sqrt{d}})$, $\mathbf{D} = \operatorname{diag}(\mathbf{A}\mathbf{1}_n)$, and $\mathbf{V}_{ij} \triangleq \boldsymbol{v}_{ij}$.

While attention has enjoyed unprecedented success in capturing long-range dependencies amongst datapoints, its computation is expensive, requiring $\Theta(d\,n^2)$ time to construct and multiply the matrix $\mathbf{A}$. This bottleneck has inspired many practical approximate attention mechanisms (e.g., Kitaev et al., 2020; Choromanski et al., 2021; Chen et al., 2021), but, to our knowledge, only two guarantee accurate reconstruction of the softmax matrix $\mathbf{T}$ (Zandieh et al., 2023; Han et al., 2024). In this section, we design a new fast attention approximation based on sub-Gaussian thinning and derive guarantees that improve upon prior art.

4.1. Thinning attention in theory

Algorithm 1: Thinformer

Input: Queries, keys, and values $(\boldsymbol{q}_i, \boldsymbol{k}_i, \boldsymbol{v}_i)_{i=1}^n$ in $\mathbb{R}^d$, n_{out}

// Define key-value attention kernel

$\mathbf{k}_{\text{att}}((\tilde{\boldsymbol{k}}, \tilde{\boldsymbol{v}}), (\tilde{\boldsymbol{k}}', \tilde{\boldsymbol{v}}')) \triangleq \exp\left(\langle \tilde{\boldsymbol{k}}, \tilde{\boldsymbol{k}}' \rangle\right) \langle \tilde{\boldsymbol{v}}, \tilde{\boldsymbol{v}}' \rangle$

// Thin augmented key-value pairs using $\mathbf{k}_{\text{att}}$

$v_{\max} \leftarrow \max_{i \in [n]} \|\boldsymbol{v}_i\|_\infty$; $(\tilde{\boldsymbol{k}}_i, \tilde{\boldsymbol{v}}_i)_{i=1}^n \leftarrow (\boldsymbol{k}_i / d^{\frac{1}{4}}, (\boldsymbol{v}_i, v_{\max}))_{i=1}^n$

$\mathcal{X}_{\text{out}} \leftarrow \text{KH-COMPRESS}(0.5)(\mathcal{X}_{\text{in}} = (\tilde{\boldsymbol{k}}_i, \tilde{\boldsymbol{v}}_i)_{i=1}^n, \mathbf{k}_{\text{att}}, n_{\text{out}})$

// Return exact attention on selected key-value subset

return $\widehat{\mathbf{T}} \triangleq \textsc{Attention}\big((\boldsymbol{q}_i)_{i=1}^n, \{(\boldsymbol{k}, \boldsymbol{v}) : (\tilde{\boldsymbol{k}}, \tilde{\boldsymbol{v}}) \in \mathcal{X}_{\text{out}}\}\big)$

Alg. 1 summarizes our new *Thinformer* module. At its heart is a new key-value attention kernel $\mathbf{k}_{\text{att}}$ that mimics the special structure of the softmax matrix $\mathbf{T}$. Alg. 1 uses the attention kernel and a high-quality thinning algorithm, KH-COMPRESS(0.5), to subselect key-value pairs and then computes exact attention (4) for the key-value subset. In total, this requires only $O(d\,n_{\text{out}}^2)$ time to run KH-COMPRESS(0.5) and $O(d\,n\,n_{\text{out}})$ time to compute ATTENTION with n queries and n_{out} key-value pairs. In contrast, computing the exact softmax matrix $\mathbf{T}$ with standard matrix multiplication requires $\Theta(d\,n^2)$ time. Our next result, proved in App. D, shows that Alg. 1 also admits a strong quality guarantee for approximating $\mathbf{T}$.

Theorem 2 (Quality of Thinformer). *With probability at least $\frac{1}{2}$, Thinformer (Alg. 1) yields*

$$\|\widehat{\mathbf{T}} - \mathbf{T}\|_{\max} \leq \frac{c \exp(\frac{2R^2}{\sqrt{d}}) \|\mathbf{V}\|_{2,\infty} \sqrt{\log_2(n_{\text{out}}) \log(12 n_{\text{out}} \log_2 \frac{n_{\text{in}}}{n_{\text{out}}})}}{n_{\text{out}}}$$

for $c \triangleq \frac{128}{\sqrt{3}} \sqrt{(d+1)\log(3e^2(\frac{R^2}{\sqrt{d}}+2)\|\mathbf{V}\|_{\max})} + \sqrt{\log(8)}(4 + \frac{128}{\sqrt{3}})$ *and* $R \triangleq \max_{i \in [n]} \max(\|\boldsymbol{k}_i\|_2, \|\boldsymbol{q}_i\|_2)$.

Table 2: **Practical approximations with guarantees.** For each approximation $\widehat{\mathbf{T}} \in \mathbb{R}^{n \times d}$ to the softmax matrix $\mathbf{T}$ (4), we report, up to a constant factor, the best worst-case error guarantee for $\|\widehat{\mathbf{T}} - \mathbf{T}\|_{\max}$ given $O(d\,n^{1+a})$ running time and γ-bounded (5) queries and keys. Here, the ratio $\|\mathbf{V}\|_{\text{op}} / \|\mathbf{V}\|_{2,\infty}$ lies in $[1, \sqrt{n}]$ and $\tau = 0.173 + o(1)$.

Approximation	Guarantee
Thinformer	$\frac{n^{2\gamma}\sqrt{d \log(n\|\mathbf{V}\|_{\max})}\log n}{n^a} \cdot \|\mathbf{V}\|_{2,\infty}$
KDEformer	$\frac{n^{2\gamma + \frac{\tau}{2}(1+\frac{\gamma}{2})}}{n^{a/2}} \cdot \|\mathbf{V}\|_{\text{op}}$
HyperAttention	$\frac{n^{\frac{17\gamma}{3}}(\log n)^{\frac{1}{6}}}{n^{a/6}} \cdot \|\mathbf{V}\|_{\text{op}}$

To put this result into context, let us compare with the existing guarantees for practical attention approximation, summarized in Tab. 2. Under the γ-boundedness assumption,

$$\max_{i \in [n]} \max(\|\boldsymbol{k}_i\|_2^2, \|\boldsymbol{q}_i\|_2^2) \leq \gamma \sqrt{d} \log n, \quad (5)$$

the KDEformer approximation $\widehat{\mathbf{T}}_{\text{kde}}$ (Zandieh et al., 2023, Cor. 3.6) with $\tau = 0.173 + o(1)$, the HyperAttention approximation $\widehat{\mathbf{T}}_{\text{hyp}}$ (Han et al., 2024, Thm. 1) with no masking, and the Thinformer approximation $\widehat{\mathbf{T}}_{\text{thin}}$ guarantee

$$\|\widehat{\mathbf{T}}_{\text{kde}} - \mathbf{T}\|_{\max} \leq O\left(\frac{n^{2\gamma + \frac{\tau}{2}(1+\frac{\gamma}{2})}}{n^{a/2}} \cdot \|\mathbf{V}\|_{\text{op}}\right)$$

$$\|\widehat{\mathbf{T}}_{\text{hyp}} - \mathbf{T}\|_{\max} \leq O\left(\frac{n^{\frac{17\gamma}{3}}(\log n)^{\frac{1}{6}}}{n^{a/6}} \cdot \|\mathbf{V}\|_{\text{op}}\right)$$

$$\|\widehat{\mathbf{T}}_{\text{thin}} - \mathbf{T}\|_{\max} \leq O\left(\frac{n^{2\gamma}\sqrt{d \log(n\|\mathbf{V}\|_{\max})}\log n}{n^a} \cdot \|\mathbf{V}\|_{2,\infty}\right)$$

with $O(dn^{1+a})$ runtime and probability at least $\frac{1}{2}$. The Thinformer guarantee exhibits four improvements over its predecessors. First, it establishes a significantly faster error decay rate (n^{-a} versus $n^{-a/2}$ or $n^{-a/6}$) for a given subquadratic runtime n^{1+a}. Second, it reduces the dependence on the error inflation factor γ. Third, like the HyperAttention guarantee, it eliminates all dependence on the KDEformer penalty parameter τ. Finally, it reduces dependence on the value matrix by a factor of $\frac{\|\mathbf{V}\|_{\text{op}}}{\|\mathbf{V}\|_{2,\infty}} \in [1, \sqrt{n}]$.

Put otherwise, with bounded $\|\mathbf{V}\|_{2,\infty}$, $\widehat{\mathbf{T}}_{\text{thin}}$ can provide consistent (i.e., $\|\widehat{\mathbf{T}}_{\text{thin}} - \mathbf{T}\|_{\max} \to 0$ as $n \to \infty$) subquadratic estimation whenever γ is bounded away from $1/2$

Table 3: **Quality of T2T-ViT attention approximations on ImageNet.** We report mean Top-1 accuracy ± 1 standard deviation across five random seeds and mean forward pass runtime ± 1 standard deviation across 50 batches of 64 images.

Attention Algorithm	Top-1 Accuracy (%)	Layer 1 Runtime (ms)	Layer 2 Runtime (ms)
Exact	82.55 ± 0.00	18.48 ± 0.12	1.40 ± 0.01
Performer	80.56 ± 0.30	2.54 ± 0.01	0.60 ± 0.01
Reformer	81.47 ± 0.06	7.84 ± 0.03	1.53 ± 0.01
KDEformer	82.00 ± 0.07	5.39 ± 0.03	2.28 ± 0.03
Scatterbrain	82.05 ± 0.08	6.86 ± 0.02	1.55 ± 0.03
Thinformer (Ours)	82.18 ± 0.05	2.06 ± 0.01	0.54 ± 0.00

Table 4: **Quality of BigGAN attention approximations for image generation.** We report Frechet Inception Distance (FID) with the ImageNet validation set, Inception Scores (IS), and forward pass runtime for computing the approximate softmax matrix (4) on an NVIDIA A100 GPU. A lower FID or higher IS indicates better image generation quality.

Method	FID (↓)	IS (↑)	Runtime (ms)
Exact	23.86	50.30 ± 3.94	5.61
Reformer	75.19	13.14 ± 1.66	7.98
Performer	29.68	29.30 ± 2.18	1.58
KDEformer	21.86	48.82 ± 3.85	6.87
Thinformer (Ours)	21.70	48.96 ± 3.65	2.37

and guarantee, for example, $O(\frac{1}{\sqrt{n}})$ error in $\widetilde{O}(dn^{\frac{3}{2}+2\gamma})$ time. In contrast, the $\widehat{\mathbf{T}}_{\text{kde}}$ and $\widehat{\mathbf{T}}_{\text{hyp}}$ bounds require quadratic runtime to guarantee $O(\frac{1}{\sqrt{n}})$ error in the best case ($\|\mathbf{V}\|_{\text{op}} = O(1)$) and cannot guarantee consistent sub-quadratic estimation in the worst case ($\|\mathbf{V}\|_{\text{op}} = \Omega(\sqrt{n})$).

4.2. Thinning attention in practice

To gauge the practical effectiveness of Alg. 1, we recreate the benchmark Tokens-To-Token Vision Transformer (T2T-ViT) and BigGAN image generation experiments of Zandieh et al. (2023). In the T2T-ViT experiment, attention approximations are scored on their ImageNet classification accuracy and computational expense when used as drop-in replacements for the two most expensive attention layers in a pretrained T2T-ViT neural network (Yuan et al., 2021). In the BigGAN experiment, approximations are scored on their computational expense and two popular measures of image generation quality, the Frechet Inception Distance (FID, Heusel et al., 2017) and Inception Score (IS, Salimans et al., 2016). Using the exact implementations and settings provided by Zandieh et al. (2023), we benchmark our PyTorch implementation of Thinformer against exact attention and four leading attention approximations: Performer (Choromanski et al., 2021), Reformer (Kitaev et al., 2020), ScatterBrain (Chen et al., 2021), and KDEformer.

In Tab. 3, we find that Thinformer provides the highest Top-1 accuracy on the ImageNet 2012 validation set (Russakovsky et al., 2015), while running faster than all of the alternatives. In Tab. 4, Thinformer ($\mathfrak{g} = 2$) yields better FID and IS than all of the alternatives while running significantly faster than exact, KDEformer, and Reformer. Performer runs faster still but at the expense of substantially worse FID and IS. The final attention call of Thinformer can also be combined with optimized attention implementations like FlashAttention (Dao et al., 2022; Dao, 2024) to further reduce the time and memory footprint. See supplementary experiment details in App. E.

5. Conclusion

This work introduced a new analysis of thinning algorithms that adapts to low-rank structures. We exploited this adaptivity to design fast algorithms with strong quality guarantees for dot-product attention in Transformers. More broadly, our techniques provide a general framework for reducing computational resource use in machine learning. Such tools have the potential to reduce energy costs and environmental harms from model training, inference, and evaluation and to improve accessibility in resource-constrained settings, all while provably maintaining high quality.

References

Alweiss, R., Liu, Y. P., and Sawhney, M. Discrepancy minimization via a self-balancing walk. In *Proceedings of the 53rd An-*

nual ACM SIGACT Symposium on Theory of Computing, pp. 14–20, 2021. (Cited on page 1.)

Bansal, N., Dadush, D., Garg, S., and Lovett, S. The gram-schmidt walk: a cure for the banaszczyk blues. In *Proceedings of the 50th annual acm sigact symposium on theory of computing*, pp. 587–597, 2018. (Cited on pages 2, 13, and 17.)

Chen, B., Dao, T., Winsor, E., Song, Z., Rudra, A., and Ré, C. Scatterbrain: unifying sparse and low-rank attention approximation. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NeurIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393. (Cited on pages 3 and 4.)

Choromanski, K. M., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J. Q., Mohiuddin, A., Kaiser, L., Belanger, D. B., Colwell, L. J., and Weller, A. Rethinking attention with performers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Ua6zuk0WRH. (Cited on pages 3 and 4.)

Dao, T. Flashattention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=mZn2Xyh9Ec. (Cited on page 4.)

Dao, T., Fu, D., Ermon, S., Rudra, A., and Ré, C. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022. (Cited on page 4.)

Dong, L., Xu, S., and Xu, B. Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5884–5888, 2018. doi: 10.1109/ICASSP.2018.8462506. (Cited on page 3.)

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy. (Cited on page 3.)

Dwivedi, R. and Mackey, L. Generalized kernel thinning. In *International Conference on Learning Representations*, 2022. (Cited on page 1.)

Dwivedi, R. and Mackey, L. Kernel thinning. *Journal of Machine Learning Research*, 25(152):1–77, 2024. (Cited on pages 1, 2, 9, 10, 11, 12, and 13.)

Han, I., Jayaram, R., Karbasi, A., Mirrokni, V., Woodruff, D., and Zandieh, A. Hyperattention: Long-context attention in near-linear time. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Eh0Od2BJIM. (Cited on page 3.)

Harshaw, C., Sävje, F., Spielman, D. A., and Zhang, P. Balancing covariates in randomized experiments with the gram–schmidt walk design. *Journal of the American Statistical Association*, pp. 1–13, 2024. (Cited on page 16.)

Harvey, N. and Samadi, S. Near-Optimal Herding. In *Proceedings of The 27th Conference on Learning Theory*, volume 35, pp. 1165–1182, 2014. (Cited on page 1.)

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. (Cited on page 4.)

Hoeffding, W. Probability inequalities for sums of bounded random variables. *The collected works of Wassily Hoeffding*, pp. 409–426, 1994. (Cited on page 9.)

Kitaev, N., Kaiser, L., and Levskaya, A. Reformer: The efficient transformer. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkgNKkHtvB. (Cited on pages 3 and 4.)

Li, L., Dwivedi, R., and Mackey, L. Debiased distribution compression. In *Proceedings of the 41st International Conference on Machine Learning*, volume 203 of *Proceedings of Machine Learning Research*. PMLR, 21–27 Jul 2024. (Cited on page 1.)

Markov, A. On certain applications of algebraic continued fractions. *Unpublished Ph. D. thesis, St Petersburg*, 1884. (Cited on page 19.)

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32, 2019. (Cited on page 24.)

Phillips, J. M. and Tai, W. M. Near-optimal coresets of kernel density estimates. *Discrete & Computational Geometry*, 63 (4):867–887, 2020. (Cited on page 1.)

Rudin, W. *Functional Analysis*. International series in pure and applied mathematics. McGraw-Hill, 1991. ISBN 9780070542365. URL https://books.google.com/books?id=Sh_vAAAAMAAJ. (Cited on page 8.)

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y. (Cited on page 4.)

Saadetoglu, M. and Dinsev, S. M. Inverses and determinants of n × n block matrices. *Mathematics*, 11(17), 2023. ISSN 2227-7390. doi: 10.3390/math11173784. URL https://www.mdpi.com/2227-7390/11/17/3784. (Cited on page 17.)

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016. (Cited on page 4.)

Sherman, J. and Morrison, W. J. Adjustment of an Inverse Matrix Corresponding to a Change in One Element of a Given Matrix. *The Annals of Mathematical Statistics*, 21(1):124 – 127, 1950. doi: 10.1214/aoms/1177729893. URL https://doi.org/10.1214/aoms/1177729893. (Cited on page 17.)

Shetty, A., Dwivedi, R., and Mackey, L. Distribution compression in near-linear time. In *International Conference on Learning Representations*, 2022. (Cited on pages 1, 2, 12, 13, 18, and 19.)

Steinwart, I. and Christmann, A. Support vector machines. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1, 2008. URL https://api.semanticscholar.org/CorpusID:661123. (Cited on page 1.)

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964. (Cited on page 3.)

Wainwright, M. J. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge University Press, 2019. (Cited on pages 20 and 21.)

Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z.-H., Tay, F. E., Feng, J., and Yan, S. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 558–567, 2021. (Cited on page 4.)

Zandieh, A., Han, I., Daliri, M., and Karbasi, A. Kdeformer: Accelerating transformers via kernel density estimation. In *International Conference on Machine Learning*, pp. 40605–40623. PMLR, 2023. (Cited on pages 3, 4, and 24.)

Appendix Contents

A. Appendix Notation and Definitions

We often use the shorthand $(a)_+ \triangleq \max(a, 0)$ as well as the shorthand $\mathbf{k}(\mathcal{X}, \mathcal{X})$ to represent the matrix $(\mathbf{k}(\boldsymbol{x}_i, \boldsymbol{x}_j))_{i,j=1}^n$. In addition, for each kernel $\mathbf{k}$, we let $\mathcal{H}_{\mathbf{k}}$ and $\|\cdot\|_{\mathbf{k}}$ represent the associated reproducing kernel Hilbert space (RKHS) and RKHS norm, so that $\mathbb{B}_{\mathbf{k}} = \{f \in \mathcal{H}_{\mathbf{k}} : \|f\|_{\mathbf{k}} \leq 1\}$ and define

$$(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k} \triangleq \frac{1}{n_{\text{in}}} \sum_{x \in \mathcal{X}_{\text{in}}} \mathbf{k}(\boldsymbol{x}, \cdot) - \frac{1}{n_{\text{out}}} \sum_{x \in \mathcal{X}_{\text{out}}} \mathbf{k}(\boldsymbol{x}, \cdot).$$

We also relate our definition of a sub-Gaussian thinning algorithm (Def. 3) to several useful notions of sub-Gaussianity.

Definition A.1 (Sub-Gaussian vector). *We say that a random vector $\boldsymbol{w} \in \mathbb{R}^n$ is* $(\mathbf{K}, \nu)$-sub-Gaussian on an event $\mathcal{E}$ *if* $\mathbf{K}$ *is SPSD and* $\nu > 0$ *satisfies*

$$\mathbb{E}_{\mathcal{E}}\left[\exp(\boldsymbol{u}^\top \mathbf{K} \boldsymbol{w})\right] \leq \exp(\tfrac{\nu^2}{2} \cdot \boldsymbol{u}^\top \mathbf{K} \boldsymbol{u}) \quad \textit{for all} \quad \boldsymbol{u} \in \mathbb{R}^n. \tag{6}$$

If, in addition, the event has probability 1*, we say that $\boldsymbol{w}$ is* $(\mathbf{K}, \nu)$-sub-Gaussian.

Notably, a thinning algorithm is $(\mathbf{K}, \nu, \delta)$-sub-Gaussian if and only if its associated vector $\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}$ is $(\mathbf{K}, \nu)$-sub-Gaussian on an event $\mathcal{E}$ of probability at least $1 - \delta/2$.

Definition A.2 (Sub-Gaussian function). *For a kernel* $\mathbf{k}$*, we say that a random function $\phi \in \mathcal{H}_{\mathbf{k}}$ is* $(\mathbf{k}, \nu)$-sub-Gaussian on an event $\mathcal{E}$ *if* $\nu > 0$ *satisfies*

$$\mathbb{E}_{\mathcal{E}}[\exp(\langle f, \phi \rangle_{\mathbf{k}})] \leq \exp(\tfrac{\nu^2}{2} \cdot \|f\|_{\mathbf{k}}^2) \quad \textit{for all} \quad f \in \mathcal{H}_{\mathbf{k}}. \tag{7}$$

If, in addition, the event has probability 1*, we say that* ϕ *is* $(\mathbf{k}, \nu)$-sub-Gaussian.

Our next two lemmas show that for finitely-supported signed measures like $\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}$, this notion of functional sub-Gaussianity is equivalent to the prior notion of vector sub-Gaussianity, allowing us to use the two notions interchangeably. Hereafter, we say that $\mathbf{k}$ generates a SPSD matrix $\mathbf{K}$ if $\mathbf{k}(\mathcal{X}, \mathcal{X}) = \mathbf{K}$.

Lemma A.1 (Functional sub-Gaussianity implies vector sub-Gaussianity). *In the notation of Def. 3, if* $(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}$ *is* $(\mathbf{k}, \nu)$*-sub-Gaussian on an event* $\mathcal{E}$ *and* $\mathbf{k}$ *generates* $\mathbf{K}$*, then the vector* $\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}$ *is* $(\mathbf{K}, \nu)$*-sub-Gaussian on* $\mathcal{E}$*.*

Proof. Suppose $(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}$ is $(\mathbf{k}, \nu)$-sub-Gaussian on an event $\mathcal{E}$, fix a vector $\boldsymbol{u} \in \mathbb{R}^n$, and define the function

$$f_{\boldsymbol{u}} \triangleq \sum_{i=1}^n u_i \mathbf{k}(\cdot, x_i) \in \mathcal{H}_{\mathbf{k}}.$$

By the reproducing property,

$$\boldsymbol{u}^\top \mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}) = \langle f_{\boldsymbol{u}}, (\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k} \rangle_{\mathbf{k}} \quad \text{and} \quad \|f_{\boldsymbol{u}}\|_{\mathbf{k}}^2 = \boldsymbol{u}^\top \mathbf{K} \boldsymbol{u}. \tag{8}$$

Invoking the representations (8) and the functional sub-Gaussianity condition (7) we therefore obtain

$$\mathbb{E}_{\mathcal{E}}\left[\exp(\boldsymbol{u}^\top \mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\right] = \mathbb{E}_{\mathcal{E}}[\exp(\langle f_{\boldsymbol{u}}, (\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k} \rangle_{\mathbf{k}})] \leq \exp(\|f_{\boldsymbol{u}}\|_{\mathbf{k}}^2 \cdot \tfrac{\nu^2}{2}) = \exp(\boldsymbol{u}^\top \mathbf{K} \boldsymbol{u} \cdot \tfrac{\nu^2}{2}),$$

so that $\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}$ is $(\mathbf{K}, \nu)$-sub-Gaussian on the event $\mathcal{E}$ as claimed. □

Lemma A.2 (Vector sub-Gaussianity implies functional sub-Gaussianity). *In the notation of Def. 3, if* $\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}$ *is* $(\mathbf{K}, \nu)$*-sub-Gaussian on an event* $\mathcal{E}$ *and* $\mathbf{k}$ *generates* $\mathbf{K}$*, then* $(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}$ *is* $(\mathbf{k}, \nu)$*-sub-Gaussian on* $\mathcal{E}$*.*

Proof. Suppose $\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}$ is $(\mathbf{K}, \nu)$-sub-Gaussian on an event $\mathcal{E}$, fix a function $f \in \mathcal{H}_{\mathbf{k}}$, and consider the set

$$\mathcal{L} \triangleq \left\{ f_{\boldsymbol{u}} \triangleq \sum_{i=1}^n u_i \mathbf{k}(\cdot, x_i) : \boldsymbol{u} \in \mathbb{R}^n \right\}.$$

Since $\mathcal{L}$ is a closed linear subspace of $\mathcal{H}_{\mathbf{k}}$, we can decompose f as $f = f_{\boldsymbol{u}} + f_\perp$, where $\boldsymbol{u} \in \mathbb{R}^n$ and $f_\perp$ is orthogonal to $\mathcal{L}$ (Rudin, 1991, Theorem 12.4), so that

$$\|f\|_{\mathbf{k}}^2 = \|f_{\boldsymbol{u}}\|_{\mathbf{k}}^2 + \|f_\perp\|_{\mathbf{k}}^2 \quad \text{and} \quad \|f_{\boldsymbol{u}}\|_{\mathbf{k}}^2 = \boldsymbol{u}^\top \mathbf{K} \boldsymbol{u}. \tag{9}$$

Invoking the orthogonality of $f_\perp$ and $(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k} \in \mathcal{L}$, the reproducing property representations (8), and the vector sub-Gaussianity condition (6), we find that

$$\begin{aligned}\mathbb{E}_{\mathcal{E}}[\exp(\langle f, (\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k} \rangle_{\mathbf{k}})] &= \mathbb{E}_{\mathcal{E}}[\exp(\langle f_{\boldsymbol{u}} + f_\perp, (\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k} \rangle_{\mathbf{k}})] = \mathbb{E}_{\mathcal{E}}\left[\exp(\boldsymbol{u}^\top \mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\right]) \\ &\leq \exp(\boldsymbol{u}^\top \mathbf{K} \boldsymbol{u} \cdot \tfrac{\nu^2}{2}) \overset{(9)}{\leq} \exp(\|f\|_{\mathbf{k}}^2 \cdot \tfrac{\nu^2}{2}),\end{aligned}$$

so that $(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}$ is $(\mathbf{k}, \nu)$-sub-Gaussian on the event $\mathcal{E}$ as claimed. □

We end our discussion about the versions of sub-Gaussianity considered above by presenting the standard fact about the additivity of sub-Gaussianity parameters under summation of independent sub-Gaussian random vectors, adapted to our setting.

Lemma A.3 (Vector sub-Gaussian additivity). *Suppose that, for each* $j \in [m]$*,* $\Delta_j \in \mathbb{R}^n$ *is* $(\mathbf{K}, \nu_j)$ *on an event* $\mathcal{E}_j$ *given* $\Delta_{1:(j-1)} \triangleq (\Delta_1, \ldots, \Delta_{j-1})$ *and* $\mathcal{E}_{\leq j-1} \triangleq \bigcap_{i=1}^{j-1} \mathcal{E}_i$*. Then* $\sum_{j=1}^m \Delta_j$ *is* $(\mathbf{K}, (\sum_{j=1}^m \nu_j^2)^{1/2})$*-sub-Gaussian on* $\mathcal{E}_{\leq m}$*.*

Proof. Let $\mathcal{E}_{\leq s} = \bigcap_{j=1}^s \mathcal{E}_j$ for each $s \in [m]$. We prove the result for $\mathcal{Z}_s = \sum_{i=1}^s \Delta_j$ by induction on $s \in [m]$. The result holds for the base case of $s = 1$ by assumption. For the inductive case, suppose the result holds for $s \in [m-1]$. Fixing $\boldsymbol{u} \in \mathbb{R}^n$, we may apply the tower property, our conditional sub-Gaussianity assumption, and our inductive hypothesis in turn to conclude

$$\begin{aligned}\mathbb{E}\left[\exp(\langle \boldsymbol{u}, \mathbf{K} \textstyle\sum_{j=1}^{s+1} \Delta_j \rangle) \mathbf{1}[\mathcal{E}_{\leq s+1}]\right] &= \mathbb{E}\left[\exp(\langle \boldsymbol{u}, \mathbf{K} \textstyle\sum_{j=1}^{s} \Delta_j \rangle) \mathbf{1}[\mathcal{E}_{\leq s}] \mathbb{E}[\exp(\langle \boldsymbol{u}, \Delta_{s+1} \rangle) \mathbf{1}[\mathcal{E}_{s+1}] \mid \Delta_{1:s}, \mathcal{E}_{\leq s}]\right] \\ &\leq \mathbb{E}\left[\exp(\langle \boldsymbol{u}, \mathbf{K} \textstyle\sum_{j=1}^{s} \Delta_j \rangle) \mathbf{1}[\mathcal{E}_{\leq s}]\right] \exp\left(\tfrac{\nu_{s+1}^2}{2} \cdot \boldsymbol{u}^\top \mathbf{K} \boldsymbol{u}\right) \leq \exp\left(\tfrac{\sum_{j=1}^{s+1} \nu_j^2}{2} \cdot \boldsymbol{u}^\top \mathbf{K} \boldsymbol{u}\right).\end{aligned}$$

Hence, $\mathcal{Z}_{s+1}$ is $(\mathbf{K}, (\sum_{j=1}^{s+1} \nu_j^2)^{1/2})$-sub-Gaussian on $\mathcal{E}_{\leq s+1}$, and the proof is complete. □

B. Proof of Tab. 1: Sub-Gaussian Thinning Examples

This section provides supplementary details for each of the sub-Gaussian thinning algorithms of Tab. 1.

B.1. Subsampling

B.1.1. Proof of Prop. 1: Quality of uniform subsampling

We begin by computing the first and second moments of $\boldsymbol{p}_{\text{out}}$: $\mathbb{E}[\boldsymbol{p}_{\text{out}}] = \boldsymbol{p}_{\text{in}}$ and

$$\mathbb{E}[\boldsymbol{p}_{\text{out}}\boldsymbol{p}_{\text{out}}^\top] = \frac{1}{n_{\text{out}}}\operatorname{diag}(\boldsymbol{p}_{\text{in}}) + \frac{n_{\text{in}}(n_{\text{out}}-1)}{n_{\text{out}}(n_{\text{in}}-1)}(\boldsymbol{p}_{\text{in}}\boldsymbol{p}_{\text{in}}^\top - \frac{1}{n_{\text{in}}}\operatorname{diag}(\boldsymbol{p}_{\text{in}})) = \frac{1}{n_{\text{out}}}(\tfrac{n_{\text{in}}-n_{\text{out}}}{n_{\text{in}}-1})\operatorname{diag}(\boldsymbol{p}_{\text{in}}) + \frac{n_{\text{in}}(n_{\text{out}}-1)}{n_{\text{out}}(n_{\text{in}}-1)}\boldsymbol{p}_{\text{in}}\boldsymbol{p}_{\text{in}}^\top.$$

Hence,

$$\begin{aligned}\mathbb{E}[\mathrm{MMD}^2_{\mathbf{K}}(\boldsymbol{p}_{\text{in}}, \boldsymbol{p}_{\text{out}})] &= \boldsymbol{p}_{\text{in}}^\top\mathbf{K}\boldsymbol{p}_{\text{in}} - 2\boldsymbol{p}_{\text{in}}^\top\mathbf{K}\mathbb{E}[\boldsymbol{p}_{\text{out}}] + \mathbb{E}[\boldsymbol{p}_{\text{out}}^\top\mathbf{K}\boldsymbol{p}_{\text{out}}] = \operatorname{tr}(\mathbf{K}\mathbb{E}[\boldsymbol{p}_{\text{out}}\boldsymbol{p}_{\text{out}}^\top]) - \boldsymbol{p}_{\text{in}}^\top\mathbf{K}\boldsymbol{p}_{\text{in}} \\ &= \frac{1}{n_{\text{out}}}(\tfrac{n_{\text{in}}-n_{\text{out}}}{n_{\text{in}}-1})(\operatorname{tr}(\mathbf{K}\operatorname{diag}(\boldsymbol{p}_{\text{in}})) - \boldsymbol{p}_{\text{in}}^\top\mathbf{K}\boldsymbol{p}_{\text{in}}) = \frac{1}{n_{\text{out}}}(\tfrac{n_{\text{in}}-n_{\text{out}}}{n_{\text{in}}-1})C_{\mathbf{K}}. \end{aligned} \tag{10}$$

To derive the second advertised result, we note that

$$\mathbb{E}[\|\mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\|_{\mathcal{I}}^2] \geq \max_{i\in\mathcal{I}}\mathbb{E}[(\boldsymbol{e}_i^\top\mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}))^2] = \max_{i\in\mathcal{I}}\mathbb{E}[\mathrm{MMD}^2_{\mathbf{K}\boldsymbol{e}_i\boldsymbol{e}_i^\top\mathbf{K}}(\boldsymbol{p}_{\text{in}}, \boldsymbol{p}_{\text{out}})]$$

and invoke the initial result (10) to conclude.

B.1.2. Sub-Gaussianity of subsampling

Proposition B.1 (Sub-Gaussianity of uniform subsampling)**.** *For any SPSD $\mathbf{K} \in \mathbb{R}^{n\times n}$, uniform subsampling (without replacement) is a $(\mathbf{K}, \nu, 0)$-sub-Gaussian thinning algorithm with*

$$\nu \triangleq \frac{\sqrt{\|\mathbf{K}\|_{\max}}}{\sqrt{n_{\text{out}}}}.$$

Proof. Fix any vector $\boldsymbol{u} \in \mathbb{R}^n$, and let $J_1, \ldots, J_{n_{\text{out}}}$ be the random indices in $[n]$ selected by uniform subsampling. Since $\boldsymbol{u}^\top\mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}) = \frac{1}{n_{\text{out}}}\sum_{i=1}^{n_{\text{out}}}\boldsymbol{u}^\top\mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{e}_{J_i})$ is an average of mean-centered scalars drawn without replacement and satisfying

$$|\boldsymbol{u}^\top\mathbf{K}\boldsymbol{e}_{J_i}| \leq \sqrt{\boldsymbol{u}^\top\mathbf{K}\boldsymbol{u}}\sqrt{\boldsymbol{e}_{J_i}^\top\mathbf{K}\boldsymbol{e}_{J_i}} \leq \sqrt{\|\mathbf{K}\|_{\max}}\sqrt{\boldsymbol{u}^\top\mathbf{K}\boldsymbol{u}} \quad \text{with probability 1}$$

by Cauchy-Schwarz, Thm. 4 and equations (1.8) and (4.16) of Hoeffding (1994) imply that

$$\mathbb{E}[\exp(\boldsymbol{u}^\top\mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}))] \leq \exp(\tfrac{\|\mathbf{K}\|_{\max}}{2n_{\text{out}}}\boldsymbol{u}^\top\mathbf{K}\boldsymbol{u}).$$

□

B.2. KH(δ)

In this section, we analyze KH(δ) (Alg. B.1), a variant of the Kernel Halving algorithm (Dwivedi & Mackey, 2024, Alg. 2) with simplified swapping thresholds. Prop. B.2, proved in App. B.2.1, establishes the sub-Gaussianity of KH(δ) and its intermediate iterates.

Proposition B.2 (Sub-Gaussianity of KH(δ))**.** *Suppose $n_{\text{in}} \geq 2$. In the notation of Alg. B.1, on a common event $\mathcal{E}$ of probability at least $1 - \delta/2$, for all $i \in [n_{\text{in}}/2]$, $\frac{1}{2i}\widetilde{\psi}_i$ is $(\mathbf{k}, \nu_i)$-sub-Gaussian with*

$$\begin{aligned}\nu_i &= \mathfrak{b}_{\max,i}\frac{\sqrt{\log(2n_{\text{in}}/\delta)}}{2i} = \frac{\sqrt{\log(2n_{\text{in}}/\delta)}}{2i}\max_{j\in[i]}\mathrm{MMD}_{\mathbf{k}}(\boldsymbol{\delta}_{\boldsymbol{x}_{2j-1}}, \boldsymbol{\delta}_{\boldsymbol{x}_{2j}}) \leq \frac{\sqrt{\log(2n_{\text{in}}/\delta)}}{2i}\max_{j\in[i]}\mathrm{MMD}_{\mathbf{k}}(\boldsymbol{\delta}_{\boldsymbol{x}_{2j-1}}, \boldsymbol{\delta}_{\boldsymbol{x}_{2j}}) \\ &\leq \frac{\sqrt{\log(2n_{\text{in}}/\delta)}}{2i}2\min(\max_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\sqrt{\mathbf{k}(\boldsymbol{x},\boldsymbol{x})}, \max_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\mathrm{MMD}_{\mathbf{k}}(\boldsymbol{\delta}_{\boldsymbol{x}}, \mathbb{P}_{\text{in}})).\end{aligned}$$

Prop. B.2 and the triangle inequality imply that $(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k} = \frac{1}{n_{\text{in}}}\psi_{n_{\text{in}}/2}$ is $(\mathbf{k}, \nu)$-sub-Gaussian on $\mathcal{E}$ with

$$\nu = \mathfrak{b}_{\max,n_{\text{in}}/2}\frac{\sqrt{\log(2n_{\text{in}}/\delta)}}{n_{\text{in}}} \leq \frac{\sqrt{\log(2n_{\text{in}}/\delta)}}{n_{\text{in}}}2\min(\max_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\sqrt{\mathbf{k}(\boldsymbol{x},\boldsymbol{x})}, \max_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\mathrm{MMD}_{\mathbf{k}}(\boldsymbol{\delta}_{\boldsymbol{x}}, \mathbb{P}_{\text{in}})).$$

By Lem. A.1, we thus have that the KH(δ) output $\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}$ is $(\mathbf{K}, \nu)$-sub-Gaussian on $\mathcal{E}$ for $\mathbf{K}$ generated by $\mathbf{k}$ and that KH(δ) $\in \mathcal{G}_{\nu,\delta}(\mathbf{K})$.

Algorithm B.1: KH(δ): Kernel Halving with simplified swapping thresholds and failure probability $\delta/2$

Input: point sequence $\mathcal{X}_{\text{in}} = (\boldsymbol{x}_i)_{i=1}^{n_{\text{in}}}$ with even n_{in}, kernel $\mathbf{k}$

$\mathcal{S}^{(1)}, \mathcal{S}^{(2)} \leftarrow \{\}$; $\widetilde{\psi}_0 \leftarrow \mathbf{0} \in \mathcal{H}_{\mathbf{k}}$ // Initialize empty coresets: $\mathcal{S}^{(1)}, \mathcal{S}^{(2)}$ have size i after round i
$\mathfrak{b}_{\max,i} \leftarrow 0$ // Max function norm so far
for $i = 1, 2, \ldots, n_{\text{in}}/2$ **do**
  // Construct kernel difference function using next two points
  $(\boldsymbol{x}, \boldsymbol{x}') \leftarrow (\boldsymbol{x}_{2i-1}, \boldsymbol{x}_{2i})$; $f_i \leftarrow \mathbf{k}(\boldsymbol{x}_{2i-1}, \cdot) - \mathbf{k}(\boldsymbol{x}_{2i}, \cdot)$; $\eta_i \leftarrow -1$
  // Compute swapping threshold $\mathfrak{a}_i$
  $\mathfrak{b}_i^2 = \|f_i\|_{\mathbf{k}}^2 = \mathbf{k}(\boldsymbol{x}, \boldsymbol{x}) + \mathbf{k}(\boldsymbol{x}', \boldsymbol{x}') - 2\mathbf{k}(x, x')$; $\mathfrak{b}_{\max,i} = \max(\mathfrak{b}_i, \mathfrak{b}_{\max,i-1})$
  $\mathfrak{a}_i \leftarrow \mathfrak{b}_i \mathfrak{b}_{\max,i}(\frac{1}{2} + \log(2n_{\text{in}}/\delta))$
  // Compute RKHS inner product $\left\langle \widetilde{\psi}_{i-1}, f_i \right\rangle_{\mathbf{k}}$, which has a simple form
  $\alpha_i \leftarrow \sum_{j=1}^{2i-2} (\mathbf{k}(\boldsymbol{x}_j, \boldsymbol{x}) - \mathbf{k}(\boldsymbol{x}_j, \boldsymbol{x}')) - 2\sum_{\boldsymbol{z} \in \mathcal{S}^{(1)}} (\mathbf{k}(\boldsymbol{z}, \boldsymbol{x}) - \mathbf{k}(\boldsymbol{z}, \boldsymbol{x}'))$
  // Assign one point to each coreset after probabilistic swapping
  $(x, x') \leftarrow (x', x)$ and $\eta_i \leftarrow 1$ *with probability* $\min(1, \frac{1}{2}(1 - \frac{\alpha_i}{\mathfrak{a}_i})_+)$
  $\mathcal{S}^{(1)}$.append($\boldsymbol{x}$); $\mathcal{S}^{(2)}$.append($\boldsymbol{x}'$); $\widetilde{\psi}_i \leftarrow \widetilde{\psi}_{i-1} + \eta_i f_i$ // $\widetilde{\psi}_i = \sum_{x' \in \mathcal{S}^{(2)}} \mathbf{k}(x', \cdot) - \sum_{x \in \mathcal{S}^{(1)}} \mathbf{k}(x, \cdot)$
end
return $\mathcal{X}_{\text{out}} \triangleq \mathcal{S}^{(1)}$, coreset of size $n_{\text{out}} = n_{\text{in}}/2$

B.2.1. Proof of Prop. B.2: Sub-Gaussianity of KH(δ)

We begin by studying the sub-Gaussian properties of a related algorithm, the self-balancing Hilbert walk (SBHW) of Dwivedi & Mackey (2024, Alg. 3). By Dwivedi & Mackey (2024, Thm. 3(i)), when the SBHW is run on the RKHS $\mathcal{H}_{\mathbf{k}}$ with the same f_i and $\mathfrak{a}_i$ sequences employed in KH(δ), the output ψ_i of each round is $(\mathbf{k}, \sigma_i)$-sub-Gaussian for

$$\sigma_0^2 \triangleq 0 \quad \text{and} \quad \sigma_i^2 \triangleq \sigma_{i-1}^2 + \|f_i\|_{\mathbf{k}}^2 \big(1 + \tfrac{\sigma_{i-1}^2}{\mathfrak{a}_i^2}(\|f_i\|_{\mathbf{k}}^2 - 2\mathfrak{a}_i)\big)_+ \quad \forall i \geq 1. \tag{11}$$

The following lemma bounds the growth of the sub-Gaussian constants σ_i in terms of the swapping thresholds $\mathfrak{a}_i$.

Lemma B.1 (Growth of SBHW sub-Gaussian constants)**.** *For each i, the SBHW sub-Gaussian constants* (11) *satisfy*

$$\sigma_i^2 \leq c_i \quad \textit{for} \quad c_i \triangleq \max_{j \in [i]} \max(\mathfrak{b}_j^2, r_j) \quad \textit{and} \quad r_i \triangleq \tfrac{\mathfrak{a}_i^2}{2\mathfrak{a}_i - \mathfrak{b}_i^2} \leq \tfrac{\mathfrak{a}_i^2}{2\mathfrak{a}_i - \mathfrak{b}_i \mathfrak{b}_{\max,i}}.$$

Proof. We will prove the result by induction on i.

Base case. $\sigma_1^2 = \mathfrak{b}_1^2 \leq c_1$ as desired.

Inductive case. Suppose $\sigma_{i-1}^2 \leq c_{i-1}$. Then $\sigma_i^2 = g(\sigma_{i-1}^2)$ for $g(x) = x + \mathfrak{b}_i^2(1 - x/r_i)_+$. Note that the slope of g is $1 - \mathfrak{b}_i^2/r_i$ for $x < r_i$ and 1 for $x > r_i$. If $1 - \mathfrak{b}_i^2/r_i \geq 0$, then g is increasing and its maximum value over $[0, c_i]$ is at c_i. If, on the other hand, $1 - \mathfrak{b}_i^2/r_i < 0$, then g first decreases and then increases so its maximum value over $[0, c_i]$ is either at 0 or at c. Since $c_i \geq \max(r_i, c_{i-1})$, $\sigma_i^2 \leq \max(g(0), g(c_i)) = \max(\mathfrak{b}_i^2, c_i) = c_i$. The proof is complete. □

Invoking Lem. B.1, the assumption $n_{\text{in}} \geq 2$, and the fact that $\delta \mapsto \frac{\frac{1}{2} + \log(2/\delta)}{\log(2/\delta)}$ is increasing on $(0, 1]$, we find that

$$\sigma_i^2 \leq \mathfrak{b}_{\max,i}^2 \log(2n_{\text{in}}/\delta) \tfrac{(\frac{1}{2} + \log(2n_{\text{in}}/\delta))^2}{2(\log(2n_{\text{in}}/\delta))^2} \leq \mathfrak{b}_{\max,i}^2 \log(2n_{\text{in}}/\delta) \tfrac{(\frac{1}{2} + \log(4))^2}{2(\log(4))^2} \leq \mathfrak{b}_{\max,i}^2 \log(2n_{\text{in}}/\delta). \tag{12}$$

The first inequality in (12) and the definition (11) further imply that

$$\mathfrak{a}_i = \mathfrak{b}_i \mathfrak{b}_{\max,i}(\tfrac{1}{2} + \log(2n_{\text{in}}/\delta)) \geq \sigma_i \mathfrak{b}_i \sqrt{2\log(2n_{\text{in}}/\delta)} \geq \sigma_{i-1} \mathfrak{b}_i \sqrt{2\log(2n_{\text{in}}/\delta)}.$$

Hence, by Dwivedi & Mackey (2024, Thm. 3(iii)), for each $i \in [n_{\text{in}}/2]$, the vector $\widetilde{\psi}_i$ of KH(δ) coincides with the vector ψ_i of SBHW on a common event $\mathcal{E}$ of probability at least $1 - \delta/2$. Therefore, each $\frac{1}{2i}\widetilde{\psi}_i$ is $(\mathbf{k}, \frac{1}{2i}\sigma_i)$-sub-Gaussian on $\mathcal{E}$, implying the result.

Algorithm B.2: LKH(δ): Kernel Halving with linear kernel and failure probability $\delta/2$

Input: point sequence $\mathcal{X}_{\text{in}} = (\boldsymbol{x}_i)_{i=1}^{n_{\text{in}}}$ with even n_{in} and $\boldsymbol{x}_i \in \mathbb{R}^d$

$\mathcal{S}^{(1)}, \mathcal{S}^{(2)} \leftarrow \{\}$; $\psi_0 \leftarrow \mathbf{0} \in \mathbb{R}^d$ // Initialize empty coresets: $\mathcal{S}^{(1)}, \mathcal{S}^{(2)}$ have size i after round i
$\sigma_0 \leftarrow 0$ // Keep track of sub-Gaussian constant
for $i = 1, 2, \ldots, n_{\text{in}}/2$ **do**
  // Consider two points
  $(\boldsymbol{x}, \boldsymbol{x}') \leftarrow (\boldsymbol{x}_{2i-1}, \boldsymbol{x}_{2i})$; $\eta_i \leftarrow -1$
  // Compute swapping threshold $\mathfrak{a}_i$
  $\mathfrak{b}_i^2 = \langle \boldsymbol{x} - \boldsymbol{x}', \boldsymbol{x} - \boldsymbol{x}' \rangle$; $\delta_i = \frac{\delta}{2i(\log(n_{\text{in}}/2)+1)}$
  $(\mathfrak{a}_i, \sigma_i) \leftarrow$ `get_swap_params` $(\sigma_{i-1}, \mathfrak{b}_i, \delta_i)$
  // Compute inner product
  $\alpha_i \leftarrow \langle \psi_{i-1}, \boldsymbol{x} - \boldsymbol{x}' \rangle$
  // Assign one point to each coreset after probabilistic swapping
  $(\boldsymbol{x}, \boldsymbol{x}') \leftarrow (\boldsymbol{x}', \boldsymbol{x})$ and $\eta_i \leftarrow 1$ *with probability* $\min(1, \frac{1}{2}(1 - \frac{\alpha_i}{\mathfrak{a}_i})_+)$
  $\mathcal{S}^{(1)}$`.append(`$\boldsymbol{x}$`)`; $\mathcal{S}^{(2)}$`.append(`$\boldsymbol{x}'$`)`; $\widetilde{\psi}_i \leftarrow \widetilde{\psi}_{i-1} + \eta_i f_i$
end
return $\mathcal{X}_{\text{out}} \triangleq \mathcal{S}^{(1)}$, coreset of size $n_{\text{out}} = n_{\text{in}}/2$

function `get_swap_params` $(\sigma, \mathfrak{b}, \delta)$:
  $\mathfrak{a} \leftarrow \max(\mathfrak{b}\sigma\sqrt{2\log(2/\delta)}, \mathfrak{b}^2)$
  $\sigma^2 \leftarrow \sigma^2 + \mathfrak{b}^2(1 + (\mathfrak{b}^2 - 2\mathfrak{a})\sigma^2/\mathfrak{a}^2)_+$
return $(\mathfrak{a}, \sigma)$;

B.3. LKH(δ)

In this section, we analyze LKH(δ) (Alg. B.2), the Kernel Halving algorithm of (Dwivedi & Mackey, 2024, Alg. 2) with a linear kernel, $\mathbf{k}(\boldsymbol{x}, \boldsymbol{y}) = \langle \boldsymbol{x}, \boldsymbol{y} \rangle$, on $\mathbb{R}^d$ and failure probability $\delta/2$. Notably, Alg. B.2 can be carried out in only $O(nd)$ time thanks to the linear kernel structure. Prop. B.3, proved in App. B.3.1, establishes the sub-Gaussianity of LKH(δ) and its intermediate iterates.

Proposition B.3 (Sub-Gaussianity of LKH(δ)). *Suppose $n_{\text{in}} \geq 2$. In the notation of Alg. B.2, on a common event $\mathcal{E}$ of probability at least $1 - \delta/2$, for all $i \in [n_{\text{in}}/2]$, $\frac{1}{2i}\widetilde{\psi}_i$ is $(\mathbf{k}, \nu_i)$-sub-Gaussian with $\mathbf{k}(\boldsymbol{x}, \boldsymbol{y}) = \langle \boldsymbol{x}, \boldsymbol{y} \rangle$ and*

$$\begin{aligned}\nu_i &= \frac{\sqrt{\log(2n_{\text{in}}(\log(n_{\text{in}}/2)+1)/\delta)}}{2i} \max_{j \in [i]} \|\boldsymbol{x}_{2j-1} - \boldsymbol{x}_{2j}\|_2 \\ &\leq \frac{\sqrt{\log(2n_{\text{in}}(\log(n_{\text{in}}/2)+1)/\delta)}}{2i} 2\min(\max_{\boldsymbol{x} \in \mathcal{X}_{\text{in}}} \sqrt{\|\boldsymbol{x}\|_2}, \max_{\boldsymbol{x} \in \mathcal{X}_{\text{in}}} \|\boldsymbol{x} - \bar{\boldsymbol{x}}\|_2) \quad \textit{for} \quad \bar{\boldsymbol{x}} = \tfrac{1}{n_{\text{in}}} \textstyle\sum_{\boldsymbol{x} \in \mathcal{X}_{\text{in}}} \boldsymbol{\delta}_{\boldsymbol{x}}.\end{aligned}$$

Prop. B.3 and the triangle inequality imply that $(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k} = \frac{1}{n_{\text{in}}}\psi_{n_{\text{in}}/2}$ is $(\mathbf{k}, \nu)$-sub-Gaussian on $\mathcal{E}$ with

$$\begin{aligned}\nu &= \frac{\sqrt{\log(2n_{\text{in}}(\log(n_{\text{in}}/2)+1)/\delta)}}{n_{\text{in}}} \max_{j \in [n_{\text{in}}/2]} \|\boldsymbol{x}_{2j-1} - \boldsymbol{x}_{2j}\|_2 \\ &\leq \frac{\sqrt{\log(2n_{\text{in}}(\log(n_{\text{in}}/2)+1)/\delta)}}{n_{\text{in}}} 2\min(\max_{\boldsymbol{x} \in \mathcal{X}_{\text{in}}} \sqrt{\|\boldsymbol{x}\|_2}, \max_{\boldsymbol{x} \in \mathcal{X}_{\text{in}}} \|\boldsymbol{x} - \bar{\boldsymbol{x}}\|_2) \quad \text{for} \quad \bar{\boldsymbol{x}} = \tfrac{1}{n_{\text{in}}} \textstyle\sum_{\boldsymbol{x} \in \mathcal{X}_{\text{in}}} \boldsymbol{\delta}_{\boldsymbol{x}}.\end{aligned}$$

By Lem. A.1, we thus have that the LKH(δ) output $\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}$ is $(\mathbf{K}, \nu)$-sub-Gaussian on $\mathcal{E}$ for $\mathbf{K}$ generated by $\mathbf{k}$ and that LKH(δ) $\in \mathcal{G}_{\nu,\delta}(\mathbf{K})$.

B.3.1. Proof of Prop. B.3: Sub-Gaussianity of LKH(δ)

We begin by studying the sub-Gaussian properties of a related algorithm, the self-balancing Hilbert walk (SBHW) of Dwivedi & Mackey (2024, Alg. 3). By Dwivedi & Mackey (2024, Thm. 3(i)), when the SBHW is run on the RKHS $\mathcal{H}_{\mathbf{k}}$ with the same f_i and $\mathfrak{a}_i$ sequences employed in LKH(δ), the output ψ_i of each round is $(\mathbf{k}, \sigma_i)$-sub-Gaussian. Moreover, since

$$\mathfrak{a}_i \geq \sigma_{i-1}\mathfrak{b}_i\sqrt{2\log(2/\delta_i)} \quad \text{for each} \quad i \in [n_{\text{in}}/2],$$

Dwivedi & Mackey (2024, Thm. 3(iii)) implies that, for each $i \in [n_{\text{in}}/2]$, the vector $\widetilde{\psi}_i$ of LKH(δ) coincides with the vector ψ_i of SBHW on a common event $\mathcal{E}$ of probability at least $1 - \delta/2$. Therefore, each $\frac{1}{2i}\widetilde{\psi}_i$ is $(\mathbf{k}, \frac{1}{2i}\sigma_i)$-sub-Gaussian on $\mathcal{E}$. Finally, Dwivedi & Mackey (2024, (46)) shows that $\sigma_i \leq \nu_i$ for each $i \in [n_{\text{in}}/2]$, yielding the result.

B.4. RKH(δ)

Algorithm B.3: RKH(δ): Repeated KH(δ)

Input: point sequence $\mathcal{X}_{\text{in}} = (\boldsymbol{x}_i)_{i=1}^{n_{\text{in}}}$, kernel $\mathbf{k}$, output size $n_{\text{out}} \in n_{\text{in}}/2^{\mathbb{N}}$

// Repeatedly divide coreset size in half
$m \leftarrow \log_2(n_{\text{in}}/n_{\text{out}})$
for $\ell = 1, 2, \ldots, m$ **do** $\mathcal{X}_{\text{in}} \leftarrow \text{KH}(\delta/m)(\mathcal{X}_{\text{in}}, \mathbf{k})$;
return $\mathcal{X}_{\text{out}} \triangleq \mathcal{X}_{\text{in}}$, coreset of size $n_{\text{out}} = n_{\text{in}}/2^m$

In this section, we analyze repeated KH(δ) (RKH(δ), Alg. B.3), a variant of the KT-SPLIT algorithm (Dwivedi & Mackey, 2024, Alg. 1a) with simplified swapping thresholds. Our next result, proved in App. B.4.1, establishes the sub-Gaussianity of RKH(δ).

Proposition B.4 (Sub-Gaussianity of RKH(δ)). *If $n_{\text{out}} \in n_{\text{in}}/2^{\mathbb{N}}$ then* RKH(δ) *(Alg. B.3) is $(\mathbf{k}, \nu)$-sub-Gaussian with*

$$\nu = \frac{2}{n_{\text{out}}\sqrt{3}}\sqrt{\log(\frac{6n_{\text{out}}\log_2(n_{\text{in}}/n_{\text{out}})}{\delta})}\min(\max_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\sqrt{\mathbf{k}(\boldsymbol{x},\boldsymbol{x})}, \max_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\text{MMD}_{\mathbf{k}}(\boldsymbol{\delta}_{\boldsymbol{x}}, \mathbb{P}_{\text{in}}))$$

on an event $\mathcal{E}$ of probability at least $1 - \delta/2$.

By Lem. A.1, we thus have that the RKH(δ) output $\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}$ is $(\mathbf{K}, \nu)$-sub-Gaussian on $\mathcal{E}$ for $\mathbf{K}$ generated by $\mathbf{k}$ and that RKH$(\delta) \in \mathcal{G}_{\nu,\delta}(\mathbf{K})$. Finally, $\nu = O(\frac{\sqrt{\log(n_{\text{out}}/\delta)}}{n_{\text{out}}})$ when $n_{\text{out}} \geq \sqrt{n_{\text{in}}}$.

B.4.1. PROOF OF PROP. B.4: SUB-GAUSSIANITY OF RKH(δ)

Let $c = 2\min(\max_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\sqrt{\mathbf{k}(\boldsymbol{x},\boldsymbol{x})}, \max_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\text{MMD}_{\mathbf{k}}(\boldsymbol{\delta}_{\boldsymbol{x}}, \mathbb{P}_{\text{in}}))$, and, for each $\ell \in [m]$, let $\widetilde{\psi}^{(\ell)}$ represent the vector $\widetilde{\psi}_{n_{\text{in}}/2^\ell}$ produced at the end of the ℓ-th call to KH(δ). By the proof of Prop. B.2 and the union bound, on an event $\mathcal{E}$ of probability at least $1 - \delta/2$, $(\widetilde{\psi}^{(\ell)})_{\ell\in[m]} = (\psi^{(\ell)})_{\ell\in[m]}$, where each $\frac{2^{\ell-1}}{n_{\text{in}}}\psi^{(\ell)}$ is $(\mathbf{k}, \nu^{(\ell)})$-sub-Gaussian given $(\psi^{(j)})_{j\in[\ell-1]}$ for

$$\nu^{(\ell)} = c\frac{\sqrt{\log(2n_{\text{in}}m/(2^{\ell-1}\delta))}}{n_{\text{in}}/2^{\ell-1}}.$$

Hence, on $\mathcal{E}$, the weighted sum

$$(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k} = \textstyle\sum_{\ell\in[m]}\frac{2^{\ell-1}}{n_{\text{in}}}\widetilde{\psi}^{(\ell)} = \sum_{\ell\in[m]}\frac{2^{\ell-1}}{n_{\text{in}}}\psi^{(\ell)}$$

is $(\mathbf{k}, \sqrt{\sum_{\ell\in[m]}(\nu^{(\ell)})^2})$-sub-Gaussian by Dwivedi & Mackey (2024, Lem. 14). Finally, by Dwivedi & Mackey (2024, Eq. (63)), $\sqrt{\sum_{\ell\in[m]}(\nu^{(\ell)})^2} \leq \nu$.

B.5. KH-COMPRESS(δ)

In this section, we analyze KH-COMPRESS(δ) (Alg. B.4), a variant of the KT-SPLIT-COMPRESS algorithm (Shetty et al., 2022, Ex. 3) with simplified swapping thresholds.

Proposition B.5 (Sub-Gaussianity of KH-COMPRESS(δ)). *If $n_{\text{out}} \in \sqrt{n_{\text{in}}}\,2^{\mathbb{N}}$ then* KH-COMPRESS(δ) *(Alg. B.4) is $(\mathbf{k}, \nu)$-sub-Gaussian with*

$$\nu = \frac{1}{n_{\text{out}}}\sqrt{\log_2(n_{\text{out}})\log(\frac{4n_{\text{out}}\log_2(n_{\text{in}}/n_{\text{out}})}{\delta})}\max_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\sqrt{\mathbf{k}(\boldsymbol{x},\boldsymbol{x})}$$

on an event $\mathcal{E}$ of probability at least $1 - \delta/2$.

Algorithm B.4: KH-COMPRESS(δ): Compress with KH halving and failure probability δ

Input: point sequence $\mathcal{X}_{\text{in}} = (\boldsymbol{x}_i)_{i=1}^{n_{\text{in}}}$, kernel $\mathbf{k}$, $n_{\text{out}} \in \sqrt{n_{\text{in}}} \cdot 2^{\mathbb{N}}$

$\mathfrak{g} \leftarrow \log_2(n_{\text{out}}/\sqrt{n_{\text{in}}})$ // identify compression level

function `compress` $(\mathcal{S})$ *:*

  if $|\mathcal{S}| = 4^{\mathfrak{g}}$ **then return** $\mathcal{S}$

  Partition $\mathcal{S}$ into four arbitrary subsequences $\{\mathcal{S}_i\}_{i=1}^4$ each of size $|\mathcal{S}|/4$

  for $i = 1, 2, 3, 4$ **do**

    $\widetilde{\mathcal{S}}_i \leftarrow$ `compress` $(\mathcal{S}_i)$ // return coresets of size $2^{\mathfrak{g}} \cdot \sqrt{\frac{|\mathcal{S}|}{4}}$

  end

  $\widetilde{\mathcal{S}} \leftarrow$ CONCATENATE$(\widetilde{\mathcal{S}}_1, \widetilde{\mathcal{S}}_2, \widetilde{\mathcal{S}}_3, \widetilde{\mathcal{S}}_4)$; $\ell \leftarrow 2 \cdot 2^{\mathfrak{g}} \cdot \sqrt{|\mathcal{S}|}$ // coreset of size ℓ

  return $\text{KH}\Big(\frac{\ell^2}{n_{\text{in}} 4^{\mathfrak{g}+1} (\log_4 n_{\text{in}} - \mathfrak{g})}\delta\Big)(\widetilde{\mathcal{S}}, \mathbf{k})$ // coreset of size $2^{\mathfrak{g}}\sqrt{|\mathcal{S}|}$

return `compress` $(\mathcal{X}_{\text{in}})$ // coreset of size $n_{\text{out}} = 2^{\mathfrak{g}}\sqrt{n_{\text{in}}}$

Proof. Since the original Kernel Halving algorithm of Dwivedi & Mackey (2024, Alg. 2) is equal to the KT-SPLIT algorithm of Dwivedi & Mackey (2024, Alg. 1a) with $m = 1$ halving round, KH-COMPRESS(δ) is simply the KT-SPLIT-COMPRESS algorithm of (Shetty et al., 2022, Ex. 3) with KH(δ) of Alg. B.1 substituted for KT-SPLIT$(\delta, m = 1)$. The result now follows immediately from the KH(δ) sub-Gaussian constant of Prop. B.2 and the argument of Shetty et al. (2022, Rem. 2, Ex. 3). □

Now fix any SPSD $\mathbf{K}$ and any kernel $\mathbf{k}$ that generates $\mathbf{K}$. By Lem. A.1, we have that $\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}$ is $(\mathbf{K}, \nu)$-sub-Gaussian on $\mathcal{E}$ and hence that KH-COMPRESS(δ) $\in \mathcal{G}_{\nu,\delta}(\mathbf{K})$. In addition, $\nu = O(\frac{\sqrt{\log(n_{\text{out}})\log(n_{\text{out}}/\delta)}}{n_{\text{out}}})$ when $n_{\text{out}} \geq \sqrt{n_{\text{in}}}$. Furthermore, Shetty et al. (2022, Rem. 1) implies that KH-COMPRESS(δ) has a runtime less than $4^{\mathfrak{g}+1} n_{\text{in}}(\log_4(n_{\text{in}}) - \mathfrak{g}) = 4n_{\text{out}}^2 \log_2(n_{\text{in}}/n_{\text{out}}) = O(n_{\text{out}}^2)$ when $n_{\text{out}} \geq \sqrt{n_{\text{in}}}$.

B.6. GS-THIN

The section introduces and analyzes the Gram-Schmidt Thinning algorithm (GS-THIN, Alg. B.5). GS-THIN repeatedly divides an input sequence in half using, GS-HALVE (Alg. B.6), a symmetrized and kernelized version of the Gram-Schmidt (GS) Walk of Bansal et al. (2018). We will present two different implementations of GS-HALVE: a quartic-time implementation (Alg. B.6) based on the GS Walk description of Bansal et al. (2018) and a cubic-time implementation based on local updates to the matrix inverse (Alg. B.7). While both the algorithms lead to the same output given the same source of randomness, we present the original implementation[1] for conceptual clarity and the optimized implementation for improved runtime. Throughout, for a matrix $\mathbf{Q}$ and vector $\boldsymbol{u}$, we use the notation $\mathbf{Q}_{\mathcal{I}\times\mathcal{J}}$ and $\boldsymbol{u}_{\mathcal{I}}$ to represent the submatrix $(\mathbf{Q}_{ij})_{i\in\mathcal{I}, j\in\mathcal{J}}$ and subvector $(\boldsymbol{u}_i)_{i\in\mathcal{I}}$.

Algorithm B.5: GS-THIN: Gram-Schmidt Thinning

Input: point sequence $\mathcal{X}_{\text{in}} = (\boldsymbol{x}_i)_{i=1}^{n_{\text{in}}}$, kernel $\mathbf{k}$, output size $n_{\text{out}} \in n_{\text{in}}/2^{\mathbb{N}}$, HALVE $\in$ {GS-HALVE, GS-HALVE-CUBIC}

// Repeatedly divide coreset size in half

$m \leftarrow \log_2(n_{\text{in}}/n_{\text{out}})$

for $\ell = 1, 2, \ldots, m$ **do** $\mathcal{X}_{\text{in}} \leftarrow$ HALVE$(\mathcal{X}_{\text{in}}, \mathbf{k})$;

return $\mathcal{X}_{\text{out}} \triangleq \mathcal{X}_{\text{in}}$, coreset of size $n_{\text{out}} = n_{\text{in}}/2^m$

Our first result, proved in App. B.6.1, shows that GS-THIN is a sub-Gaussian thinning algorithm.

Proposition B.6 (GS-THIN sub-Gaussianity). *For* $\mathbf{K}$ *generated by* $\mathbf{k}$*,* GS-THIN *(Alg. B.5) is a* $(\mathbf{K}, \nu, 0)$*-sub-Gaussian thinning algorithm with parameter*

$$\nu \triangleq \frac{2}{\sqrt{3}} \frac{\sqrt{\|\mathbf{K}\|_{\max}}}{n_{\text{out}}}. \tag{13}$$

[1] Towards making this equivalence clear, Alg. B.6 has been expressed with the same variables that Alg. B.7 uses. Alg. B.6 can be slightly simplified if it were to be considered independently.

Algorithm B.6: GS-HALVE: Gram-Schmidt Halving

Input: point sequence $\mathcal{X}_{\text{in}} = (x_i)_{i=1}^{n_{\text{in}}}$ with even n_{in}, kernel $\mathbf{k}$

$\mathcal{X}_{\text{out}} \leftarrow \{\}$ // Initialize empty coreset

// Select one point to keep from each consecutive pair using kernelized GS Walk

$\boldsymbol{z} \leftarrow$ `kernel_gs_walk(`$\mathcal{X}_{\text{in}}$`)`

for $i = 1, \ldots, n_{\text{in}}/2$ **do**

  if $\boldsymbol{z}_i = 1$ **then**

    $\mathcal{X}_{\text{out}}$.append(x_{2i-1})

  else

    $\mathcal{X}_{\text{out}}$.append(x_{2i})

  end

end

return $\mathcal{X}_{\text{out}}$, coreset of size $n_{\text{in}}/2$

---

function `kernel_gs_walk(`$(\boldsymbol{x}_i)_{i=1}^{n_{\text{in}}}$`)`*:*

  $t \leftarrow 1$; $\boldsymbol{z}_t \leftarrow (0, 0, \ldots, 0) \in \mathbb{R}^{n_{\text{in}}/2}$ // Initialize fractional assignment vector

  $\mathcal{A} \leftarrow [n_{\text{in}}/2]$ // Initialize set of active coordinates

  $p \sim \mathcal{A}$ // Select a pivot uniformly at random

  while $\boldsymbol{z}_t \notin \{\pm 1\}^{n_{\text{in}}/2}$ **do**

    $\mathcal{A}' \leftarrow \mathcal{A} \setminus \{ \min (\{i \in [n_{\text{in}}/2] : |\boldsymbol{z}_{ti}| = 1\} \setminus ([n_{\text{in}}/2] \setminus \mathcal{A}))\}$

    // Update set of active coordinates by removing smallest index set to ± 1

    if $p \notin \mathcal{A}'$ **then**

      $p' \sim \text{Unif}(\mathcal{A}')$ // Select a new pivot from $\mathcal{A}'$ uniformly at random

    else

      $p' \leftarrow p$

    end

    // Compute step direction in which to update fractional assignment vector

    $\boldsymbol{u}_t \leftarrow \operatorname{argmin}_{\boldsymbol{u} \in \mathbb{R}^{n_{\text{in}}/2}} \boldsymbol{u}^\top \mathbf{Q} \boldsymbol{u}$ subject to $\boldsymbol{u}_{p'} = 1$ and $\boldsymbol{u}_i = 0$ for all $i \notin \mathcal{A}'$,

      where $\mathbf{Q} \in \mathbb{R}^{(n_{\text{in}}/2) \times (n_{\text{in}}/2)}$ has entries $\mathbf{Q}_{ij} \triangleq \mathbf{k}(x_{2i-1}, x_{2j-1}) + \mathbf{k}(x_{2i}, x_{2j}) - \mathbf{k}(x_{2i-1}, x_{2j}) - \mathbf{k}(x_{2i}, x_{2j-1})$

    $\delta^+ \leftarrow |\max \Delta|$ and $\delta^- \leftarrow |\min \Delta|$, where $\Delta = \left\{\delta \in \mathbb{R} : \boldsymbol{z}_t + \delta \boldsymbol{u}_t \in [-1, +1]^{n_{\text{in}}/2}\right\}$ // Select candidate step sizes

    $\delta_t \leftarrow \delta^+$ with probability $\delta^-/(\delta^+ + \delta^-)$; otherwise $\delta_t \leftarrow -\delta^-$ // Choose step size and sign at random

    $\boldsymbol{z}_{t+1} \leftarrow \boldsymbol{z}_t + \delta_t \boldsymbol{u}_t$ // Update fractional assignments

    $t \leftarrow t + 1$; $\mathcal{A} \leftarrow \mathcal{A}'$; $p \leftarrow p'$

  end

return $\boldsymbol{z}_t$, sign vector in $\{\pm 1\}^{n_{\text{in}}/2}$

Algorithm B.7: GS-HALVE-CUBIC: Gram-Schmidt Halving with cubic runtime

Input: point sequence $\mathcal{X}_{\text{in}} = (x_i)_{i=1}^{n_{\text{in}}}$ with even n_{in}, kernel $\mathbf{k}$ with positive definite $\mathbf{k}(\mathcal{X}_{\text{in}}, \mathcal{X}_{\text{in}})$

$\mathcal{X}_{\text{out}} \leftarrow \{\}$ // Initialize empty coreset

// Select one point to keep from each consecutive pair using kernelized GS Walk

$\boldsymbol{z} \leftarrow$ `kernel_gs_walk_cubic(`$\mathcal{X}_{\text{in}}$`)`

for $i = 1, \ldots, n_{\text{in}}/2$ **do**

  if $\boldsymbol{z}_i = 1$ **then**

    $\mathcal{X}_{\text{out}}$.append(x_{2i-1})

  else

    $\mathcal{X}_{\text{out}}$.append(x_{2i})

  end

end

return $\mathcal{X}_{\text{out}}$, coreset of size $n_{\text{in}}/2$

---

function `kernel_gs_walk_cubic(`$(x_i)_{i=1}^{n_{\text{in}}}$`)`:

  $t \leftarrow 1$; $\boldsymbol{z}_t \leftarrow (0, 0, \ldots, 0) \in \mathbb{R}^{n_{\text{in}}/2}$ // Initialize fractional assignment vector

  $\mathcal{A} \leftarrow [n_{\text{in}}/2]$ // Initialize set of active coordinates

  $p \sim \mathcal{A}$ // Select pivot uniformly at random

  $\mathbf{Q} \leftarrow (\mathbf{k}(x_{2i-1}, x_{2j-1}) + \mathbf{k}(x_{2i}, x_{2j}) - \mathbf{k}(x_{2i-1}, x_{2j}) - \mathbf{k}(x_{2i}, x_{2j-1}))_{i,j=1}^{n_{\text{in}}/2}$ // Form paired difference kernel matrix

  $\mathbf{C} \leftarrow (\mathbf{Q}_{\mathcal{A}\setminus\{p\} \times \mathcal{A}\setminus\{p\}})^{-1}$

  while $\boldsymbol{z}_t \notin \{\pm 1\}^{n_{\text{in}}/2}$ **do**

    $\mathcal{A}' \leftarrow \mathcal{A} \setminus \{ \min (\{i \in [n_{\text{in}}/2] : |\boldsymbol{z}_{ti}| = 1\} \setminus ([n_{\text{in}}/2] \setminus \mathcal{A}))\}$

    // Update set of active coordinates by removing smallest index set to ± 1

    if $p \notin \mathcal{A}'$ **then**

      $p' \sim \text{Unif}(\mathcal{A}')$ // Select a new pivot from $\mathcal{A}'$ uniformly at random

    else

      $p' \leftarrow p$

    end

    $\mathcal{A}_1 \leftarrow \mathcal{A} \setminus \{p\}$

    $\mathcal{A}_2 \leftarrow \mathcal{A}' \setminus \{p'\}$.

    $i \leftarrow \mathcal{A}_1 \setminus \mathcal{A}_2$ // Choose i as the (unique) index that was removed from the active coordinates

    // Compute $(\mathbf{Q}_{\mathcal{A}_2 \times \mathcal{A}_2})^{-1}$ using block matrix inversion and the Sherman-Morrison formula

    $\mathbf{D} \leftarrow \mathbf{C}_{\mathcal{A}_2 \times A_2}$

    $\mathbf{C} \leftarrow \mathbf{D} - \frac{\mathbf{D}\mathbf{Q}_{\mathcal{A}_2 \times \{i\}} \mathbf{Q}_{\{i\} \times \mathcal{A}_2} \mathbf{D}}{\mathbf{Q}_{ii} + \mathbf{Q}_{\{i\} \times \mathcal{A}_2} \mathbf{D} \mathbf{Q}_{\mathcal{A}_2 \times \{i\}}}$

    // Compute step direction in which to update fractional assignment vector

    Compute $\boldsymbol{u}_t$ as $(\boldsymbol{u}_t)_{\mathcal{A}_2} = -\mathbf{C}\mathbf{Q}_{\mathcal{A}_2 \times \{p'\}}$, $\boldsymbol{u}_{tp'} = 1$, and $\boldsymbol{u}_{ti} = 0$ for $i \notin \mathcal{A}'$

    $\delta^+ \leftarrow |\max \Delta|$ and $\delta^- \leftarrow |\min \Delta|$, where $\Delta = \left\{\delta \in \mathbb{R} : \boldsymbol{z}_t + \delta \boldsymbol{u}_t \in [-1, +1]^{n_{\text{in}}/2}\right\}$ // Select candidate step sizes

    $\delta_t \leftarrow \delta^+$ with probability $\delta^-/(\delta^+ + \delta^-)$; otherwise $\delta_t \leftarrow -\delta^-$ // Choose step size and sign at random

    $\boldsymbol{z}_{t+1} \leftarrow \boldsymbol{z}_t + \delta_t \boldsymbol{u}_t$ // Update fractional assignments

    $t \leftarrow t + 1$; $\mathcal{A} \leftarrow \mathcal{A}'$; $p \leftarrow p'$

  end

return $\boldsymbol{z}_t$, sign vector in $\{\pm 1\}^{n_{\text{in}}/2}$

Our second result, proved in App. B.6.2, shows that GS-THIN with the GS-HALVE implementation has $O(n_{\text{in}}^4)$ runtime.

Proposition B.7 (Runtime of GS-THIN with GS-HALVE). *The runtime of* GS-THIN *with implementation* GS-HALVE *(Alg. B.6) is* $O(n_{\text{in}}^4)$.

Our third result, proved in App. B.6.3, establishes the equivalence between GS-HALVE and GS-HALVE-CUBIC. More precisely, we show that the sequence of partial assignment vectors generated by `kernel_gs_walk(·)` of Alg. B.6 and `kernel_gs_walk_cubic(·)` of Alg. B.7 are identical given identical inputs, an invertible induced kernel matrix, and an identical source of randomness.

Proposition B.8 (Agreement of GS-HALVE and GS-HALVE-CUBIC). *Let* $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots$ *be the fractional assignment sequence generated by* `kernel_gs_walk`$((\boldsymbol{x}_i)_{i=1}^{n_{\text{in}}})$ *in Alg. B.6 and* $\boldsymbol{z}'_1, \boldsymbol{z}'_2, \ldots$ *be the fractional assignment sequence generated by* `kernel_gs_walk_cubic`$((\boldsymbol{x}_i)_{i=1}^{n_{\text{in}}})$ *in Alg. B.7 with an identical source of randomness. If the pairwise difference matrix*

$$\mathbf{Q} \triangleq (\mathbf{k}(x_{2i-1}, x_{2j-1}) + \mathbf{k}(x_{2i}, x_{2j}) - \mathbf{k}(x_{2i-1}, x_{2j}) - \mathbf{k}(x_{2i}, x_{2j-1}))_{i,j \in [n_{\text{in}}/2]}$$

is positive definite, then $\boldsymbol{z}_t = \boldsymbol{z}'_t$ *for all* t.

Our fourth result, proved in App. B.6.4, shows that GS-THIN with the GS-HALVE-CUBIC implementation has $O(n_{\text{in}}^3)$ runtime.

Proposition B.9 (Runtime of GS-THIN with GS-HALVE-CUBIC). *The runtime of* GS-THIN *with implementation* GS-HALVE-CUBIC *(Alg. B.7) is* $O(n_{\text{in}}^3)$.

B.6.1. PROOF OF PROP. B.6: GS-THIN SUB-GAUSSIANITY

Our first lemma bounds the sub-Gaussian constant of GS-HALVE (Alg. B.6).

Lemma B.2 (GS-HALVE sub-Gaussianity). *In the notation of Def. 1, consider the input and output vectors* $\boldsymbol{p}_{\text{in}}, \boldsymbol{p}_{\text{out}} \in \mathbb{R}^n$ *of* GS-HALVE *(Alg. B.6) for* $\mathcal{X} \supseteq \mathcal{X}_{\text{in}}$ *with* $|\mathcal{X}| = n \geq n_{\text{in}}$. *If* $\mathbf{K} = \mathbf{k}(\mathcal{X}, \mathcal{X})$, *then* $\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}$ *is* $(\mathbf{K}, \nu)$*-sub-Gaussian with*

$$\nu \triangleq \frac{2\|\mathbf{K}\|_{\max}^{1/2}}{n_{\text{in}}} = \frac{\|\mathbf{K}\|_{\max}^{1/2}}{n_{\text{out}}}.$$

Proof. Since $\mathbf{K}$ is SPSD, there exists a matrix $\boldsymbol{\Phi} \in \mathbb{R}^{n \times d}$ such that $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^\top$. Let $\mathbf{B} \in \mathbb{R}^{d \times (n_{\text{in}}/2)}$ be the matrix with entries

$$\mathbf{B}_{j,i} \triangleq \boldsymbol{\Phi}_{2i-1,j} - \boldsymbol{\Phi}_{2i,j} \quad \text{for} \quad i \in [n_{\text{in}}/2] \quad \text{and} \quad j \in [d].$$

Note that, for each $i \in [n_{\text{in}}/2]$,

$$\textstyle\sum_{j \in [d]} \mathbf{B}_{j,i}^2 = \mathbf{K}_{2i-1,2i-1} + \mathbf{K}_{2i,2i} - \mathbf{K}_{2i-1,2i} - \mathbf{K}_{2i,2i-1} \leq 4\|\mathbf{K}\|_{\max}.$$

Hence, by Harshaw et al. (2024, Thm. 6.6), $\frac{1}{n_{\text{in}}}\mathbf{B}\boldsymbol{z}$ is $(\mathbf{I}, \nu)$-sub-Gaussian where $\mathbf{I}$ is the identity matrix in $\mathbb{R}^{d \times d}$.

Now fix any $\boldsymbol{u} \in \mathbb{R}^d$. Since $\frac{1}{n_{\text{in}}}\mathbf{B}\boldsymbol{z} = -\boldsymbol{\Phi}^\top(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})$ by construction,

$$\mathbb{E}\big[\exp\big(\boldsymbol{u}^\top \mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\big)\big] \leq \mathbb{E}\Big[\exp\Big(-\langle \boldsymbol{\Phi}^\top \boldsymbol{u}, \tfrac{1}{n_{\text{in}}}\mathbf{B}\boldsymbol{z}\rangle\Big)\Big] \leq \exp\Big(\tfrac{\nu^2}{2} \cdot \|\boldsymbol{\Phi}^\top \boldsymbol{u}\|_2^2\Big) = \exp\Big(\tfrac{\nu^2}{2} \cdot \boldsymbol{u}^\top \mathbf{K} \boldsymbol{u}\Big).$$

□

Now, for $\ell \in [m]$, let $\boldsymbol{p}_\ell \in \mathbb{R}^n$ denote the output probability vector produced by the ℓ-th call to GS-HALVE. Defining $\boldsymbol{p}_0 \triangleq \boldsymbol{p}_{\text{in}}$ and $\boldsymbol{p}_{\text{out}} \triangleq \boldsymbol{p}_m$, we have

$$\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}} = \textstyle\sum_{i=1}^m \Delta_i, \quad \text{for} \quad \Delta_i \triangleq \boldsymbol{p}_{i-1} - \boldsymbol{p}_i \quad \text{for} \quad i \in [m].$$

By Lem. B.2, each $\boldsymbol{p}_{i-1} - \boldsymbol{p}_i$ is $(\mathbf{K}, \frac{2\|\mathbf{K}\|_{\max}^{1/2}}{n_{\text{in}}/2^{i-1}})$-sub-Gaussian conditional on $(\Delta_1, \ldots, \Delta_{i-1})$. Applying Lem. A.3 to the sequence $(\Delta_j)_{j=1}^m$, we find that $\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}}$ is $(\mathbf{K}, \nu)$-sub-Gaussian with parameter

$$\nu = \Big(\textstyle\sum_{j=1}^m \frac{4\|\mathbf{K}\|_{\max}}{(n_{\text{in}}/2^{j-1})^2}\Big)^{1/2} = \frac{2\|\mathbf{K}\|_{\max}^{1/2}}{n_{\text{in}}}\Big(\textstyle\sum_{j=1}^m 4^j\Big)^{1/2} \leq \frac{\|\mathbf{K}\|_{\max}^{1/2}}{n_{\text{in}}}\sqrt{\tfrac{4}{3}4^m}.$$

Simplifying the above using the fact that $n_{\text{out}} = n_{\text{in}}/2^m$ yields our desired result (13).

B.6.2. Proof of Prop. B.7: Runtime of GS-Thin with GS-Halve

We essentially reproduce the argument from Bansal et al. (2018) for the runtime of the GS-Halve algorithm in our kernelized context.

The main computational cost of GS-Halve is the execution of the `kernel_gs_walk(·)` subroutine in Alg. B.6. The number of iterations in while loop for $\boldsymbol{z}_t$ is at most $n_{\text{in}}/2$. This is due to the fact that in each iteration, at least one new variable is set to $\{\pm 1\}$. Further, in each iteration, the main computational cost is the computation of

$$\boldsymbol{u}_t \leftarrow \operatorname{argmin}_{\boldsymbol{u}\in\mathbb{R}^{n_{\text{in}}/2}} \boldsymbol{u}^\top \mathbf{Q}\boldsymbol{u}$$

under the constraints that $\boldsymbol{u}_p = 1$ and $\boldsymbol{u}_i = 0$ for all $i \notin \mathcal{A}$. Since this can be implemented in $O(n_{\text{in}}^3)$ time using standard convex optimization techniques, GS-Halve has total runtime

$$r_{\text{H}}(\ell) \le C\ell^4$$

for an input sequence of size ℓ and a constant C independent of ℓ. Now, note that GS-Thin calls GS-Halve iteratively on inputs of size $n_{\text{in}}2^{-i}$ for $i = 0, 1, \ldots, m-1$ where $m = \log_2(n_{\text{in}}/n_{\text{out}})$. Thus, GS-Thin has runtime

$$\textstyle\sum_{i=0}^{m-1} r_{\text{H}}(n_{\text{in}}/2^i) \le \sum_{i=0}^{m-1} C(n_{\text{in}}/2^i)^4 = O(n_{\text{in}}^4).$$

B.6.3. Proof of Prop. B.8: Agreement of GS-Halve and GS-Halve-Cubic

We want to reason that any round of partial coloring leads to the same output across the two algorithms. Fix any fractional assignment update round. Recall that $\mathcal{A}_1 = \mathcal{A}\backslash\{p\}$ and $\mathcal{A}_2 = \mathcal{A}' \setminus \{p'\}$. These represent the active set coordinates without the pivot before and after the update respectively.

The main difference between Algs. B.6 and B.7 is in the computation of the step direction $\boldsymbol{u}_t$, which is the solution of the program

$$\boldsymbol{u}_t \leftarrow \operatorname{argmin}_{\boldsymbol{u}\in\mathbb{R}^n} \boldsymbol{u}^\top \mathbf{Q}\boldsymbol{u} \quad \text{subject to} \quad \boldsymbol{u}_{p'} = 1 \quad \text{and} \quad \boldsymbol{u}_i = 0 \quad \text{for all} \quad i \notin \mathcal{A}'.$$

$\boldsymbol{u}_t$ has a closed form with entries

$$(\boldsymbol{u}_t)_{\mathcal{A}_2} = -(\mathbf{Q}_{\mathcal{A}_2\times\mathcal{A}_2})^{-1} \cdot \mathbf{Q}_{\mathcal{A}_2\times\{p'\}}.$$

Note that the invertibility of $\mathbf{Q}_{\mathcal{A}_2\times\mathcal{A}_2}$ follows from the positive-definiteness of $\mathbf{Q}$, as, for any $\boldsymbol{w} \in \mathbb{R}^{|\mathcal{A}_2|}$,

$$\boldsymbol{w}^\top \mathbf{Q}_{\mathcal{A}_2\times\mathcal{A}_2}\boldsymbol{w} = \tilde{\boldsymbol{w}}^\top \mathbf{Q}\tilde{\boldsymbol{w}} > 0$$

for a second vector $\tilde{\boldsymbol{w}}$ with $\tilde{\boldsymbol{w}}_{\mathcal{A}_2} = \boldsymbol{w}$ and all other entries equal to zero. Therefore, to compute $\boldsymbol{u}_t$, it suffices to keep track of the inverse of $\mathbf{Q}_{\mathcal{A}_2\times\mathcal{A}_2}$ as $\mathcal{A}'$ across iterations.

Let i be the unique element in $\mathcal{A}_1\backslash\mathcal{A}_2$. Writing $\mathbf{Q}_{\mathcal{A}_1\times\mathcal{A}_1}$ in block form, we have

$$\mathbf{Q}_{\mathcal{A}_1\times\mathcal{A}_1} = \begin{bmatrix} \mathbf{Q}_{\mathcal{A}_2\times\mathcal{A}_2} & \mathbf{Q}_{\mathcal{A}_2\times\{i\}} \\ \mathbf{Q}_{\{i\}\times\mathcal{A}_2} & \mathbf{Q}_{ii} \end{bmatrix}.$$

By block matrix inversion (see, e.g., Saadetoglu & Dinsev, 2023, Thm. 2), the leading size $|\mathcal{A}_2| \times |\mathcal{A}_2|$ principal submatrix of $(\mathbf{Q}_{\mathcal{A}_1\times\mathcal{A}_1})^{-1}$ equals

$$\mathbf{D} \triangleq \left(\mathbf{Q}_{\mathcal{A}_2\times\mathcal{A}_2} - \frac{\mathbf{Q}_{\mathcal{A}_2\times\{i\}}\mathbf{Q}_{\{i\}\times\mathcal{A}_2}}{\mathbf{Q}_{ii}}\right)^{-1}.$$

Thus, by the Sherman-Morrison formula (Sherman & Morrison, 1950),

$$(\mathbf{Q}_{\mathcal{A}_2\times\mathcal{A}_2})^{-1} = \left(\mathbf{D}^{-1} + \frac{\mathbf{Q}_{\mathcal{A}_2\times\{i\}}\mathbf{Q}_{\{i\}\times\mathcal{A}_2}}{\mathbf{Q}_{ii}}\right)^{-1} = \mathbf{D} - \frac{\mathbf{D}\mathbf{Q}_{\mathcal{A}_2\times\{i\}}\mathbf{Q}_{\{i\}\times\mathcal{A}_2}\mathbf{D}}{\mathbf{Q}_{ii}+\mathbf{Q}_{\{i\}\times\mathcal{A}_2}\mathbf{D}\mathbf{Q}_{\mathcal{A}_2\times\{i\}}}. \tag{14}$$

Hence, if we already have access to a matrix $\mathbf{C} = (\mathbf{Q}_{\mathcal{A}_1\times\mathcal{A}_1})^{-1}$, we can compute $\mathbf{D}$ by dropping the row and column of $\mathbf{C}$ corresponding to i and then compute $(\mathbf{Q}_{\mathcal{A}_2\times\mathcal{A}_2})^{-1}$ using (14). Since in Alg. B.7 we begin by explicitly computing the inverse of $\mathbf{Q}_{\mathcal{A}'\times\mathcal{A}'}$, the update step in Alg. B.7 maintains the required inverse and thus its partial assignment updates match those of Alg. B.6.

Algorithm B.8: GS-COMPRESS: Compress with GS-HALVE-CUBIC halving

Input: point sequence $\mathcal{X}_{\text{in}} = (\boldsymbol{x}_i)_{i=1}^{n_{\text{in}}}$, kernel $\mathbf{k}$, $n_{\text{out}} \in \sqrt{n_{\text{in}}} \cdot 2^{\mathbb{N}}$

$\mathfrak{g} \leftarrow \log_2(n_{\text{out}}/\sqrt{n_{\text{in}}})$ // identify compression level

function `compress` $(\mathcal{S})$ *:*

  if $|\mathcal{S}| = 4^{\mathfrak{g}}$ **then return** $\mathcal{S}$

  Partition $\mathcal{S}$ into four arbitrary subsequences $\{\mathcal{S}_i\}_{i=1}^4$ each of size $|\mathcal{S}|/4$

  for $i = 1, 2, 3, 4$ **do**

    $\widetilde{\mathcal{S}}_i \leftarrow$ `compress` $(\mathcal{S}_i)$ // return coresets of size $2^{\mathfrak{g}} \cdot \sqrt{\frac{|\mathcal{S}|}{4}}$

  end

  $\widetilde{\mathcal{S}} \leftarrow$ CONCATENATE$(\widetilde{\mathcal{S}}_1, \widetilde{\mathcal{S}}_2, \widetilde{\mathcal{S}}_3, \widetilde{\mathcal{S}}_4)$; $\ell \leftarrow 2 \cdot 2^{\mathfrak{g}} \cdot \sqrt{|\mathcal{S}|}$ // coreset of size ℓ

  return GS-HALVE-CUBIC$(\widetilde{\mathcal{S}}, \mathbf{k})$ // coreset of size $2^{\mathfrak{g}}\sqrt{|\mathcal{S}|}$

return `compress` $(\mathcal{X}_{\text{in}})$ // coreset of size $n_{\text{out}} = 2^{\mathfrak{g}}\sqrt{n_{\text{in}}}$

B.6.4. PROOF OF PROP. B.9: RUNTIME OF GS-THIN WITH GS-HALVE-CUBIC

We begin by establishing the runtime of `kernel_gs_walk_cubic(·)`.

Lemma B.3 (Running time of `kernel_gs_walk_cubic(·)`). *The routine* `kernel_gs_walk_cubic(·)` *runs in* $O(\ell^3)$ *time given a point sequence of size* ℓ*.*

Proof. First, the initialization of $\mathbf{C}$ costs $O(\ell^3)$ time using standard matrix inversion algorithms. Second, the number of iterations in the while loop is at most $\ell/2$ since, in each iteration, at least one new variable is assigned a permanent sign in $\{\pm 1\}$. In each while loop iteration, the main computational costs are the update of $\mathbf{C}$ and the computation of the step direction $\boldsymbol{u}_t$, both of which cost $O(\ell^2)$ time using standard matrix-vector multiplication. Hence, together, all while loop iterations cost $O(\ell^3)$ time. □

Given the above lemma, we have that GS-HALVE-CUBIC, on input of size ℓ, has a running time

$$r_{\text{H}}(\ell) \leq C\ell^3$$

for some C independent of ℓ. When used in GS-THIN this yields the runtime

$$\sum_{i=0}^{m-1} r_{\text{H}}(n_{\text{in}}/2^i) = \sum_{i=0}^{m-1} C(n_{\text{in}}/2^i)^3 = O(n_{\text{in}}^3).$$

B.7. GS-COMPRESS

This section introduces and analyzes the new GS-COMPRESS algorithm (Alg. B.8) which combines the COMPRESS meta-algorithm of Shetty et al. (2022) with the GS-HALVE-CUBIC halving algorithm (Alg. B.7). The following result bounds the sub-Gaussian constant and runtime of GS-COMPRESS.

Proposition B.10 (GS-COMPRESS sub-Gaussianity and runtime). *If* $\mathbf{K}$ *is generated by* $\mathbf{k}$*, then* GS-COMPRESS *is* $(\mathbf{K}, \nu, 0)$*-sub-Gaussian with*

$$\nu \triangleq \tfrac{1}{n_{\text{out}}} \sqrt{\log_2(n_{\text{out}}) \|\mathbf{K}\|_{\max}}.$$

Moreover, GS-COMPRESS *has an* $O(n_{\text{out}}^3)$ *runtime.*

Proof. By Lem. B.2 and Prop. B.8, GS-HALVE-CUBIC is $(\mathbf{K}, \nu_{\text{H}}(\ell))$-sub-Gaussian for an input point sequence of size ℓ and $\nu_{\text{H}}(\ell) = 2\sqrt{\|K\|_{\max}}/\ell$. Hence, by Lem. A.2, GS-HALVE-CUBIC is also $\nu_{\text{H}}(\ell)$ f-sub-Gaussian in the sense of Shetty et al. (2022, Def. 2) for each $f \in \mathcal{H}_{\mathbf{k}}$. By Shetty et al. (2022, Rmk. 2), GS-COMPRESS is therefore f-sub-Gaussian with parameter

$$\nu \leq \sqrt{\log_2(n_{\text{in}}/n_{\text{out}})}\nu_{\text{H}}(2n_{\text{out}}) \leq \sqrt{\log_2(n_{\text{out}})}\tfrac{\|\mathbf{K}\|_{\max}^{1/2}}{n_{\text{out}}}$$

for each $f \in \mathcal{H}_{\mathbf{k}}$. Hence, Lem. A.1 implies that GS-COMPRESS is a $(\mathbf{K}, \nu, 0)$-sub-Gaussian thinning algorithm.

Furthermore, Shetty et al. (2022, Thm. 1) implies that GS-COMPRESS has a runtime of

$$\sum_{i=0}^{\log_2(n_{\text{in}}/(2n_{\text{out}}))} 4^i \cdot r_{\text{H}}(2n_{\text{out}}2^{-i}).$$

where the GS-HALVE-CUBIC runtime $r_{\text{H}}(\ell) \leq C\ell^3$ for C independent of the input size ℓ by Lem. B.3. Therefore, the GS-COMPRESS runtime is bounded by

$$\sum_{i=0}^{\log_2(n_{\text{in}}/(2n_{\text{out}}))} 4^i \cdot (2n_{\text{out}})^3 2^{-3i} = O(n_{\text{out}}^3).$$

□

Remark 1 (COMPRESS with GS-HALVE). If the GS-HALVE implementation were used in place of GS-HALVE-CUBIC, parallel reasoning would yield an $O(n_{\text{out}}^4)$ runtime for GS-COMPRESS.

C. Proof of Thm. 1: Low-rank sub-Gaussian thinning

We establish the first kernel max seminorm bound (2) in App. C.1 and the Lipschitz kernel max seminorm bound (3) in App. C.2. Throughout, we use the notation $\mathbb{P}_{\mathcal{E}}(\mathcal{E}') \triangleq \mathbb{P}(\mathcal{E}, \mathcal{E}')$ for events $(\mathcal{E}, \mathcal{E}')$.

C.1. Proof of kernel max seminorm bound (2)

We begin by establishing a general bound on the maximum discrepancy between input and output expectations over a collection of test functions admitting a finite cover.

Lemma C.1 (Discrepancy cover bound). *Fix any kernel* $\mathbf{k}$*, subset* $\mathcal{F} \subset \mathcal{H}_{\mathbf{k}}$*, and scalars* $\varepsilon \geq 0$ *and* $\delta' \in (0, 1)$*. Define*

$$a \triangleq \sup_{f \in \mathcal{F}} \|f\|_{\mathbf{k}} \quad \text{and} \quad \mathbb{B}_{\mathcal{F}} \triangleq \{f \in \mathcal{H}_{\mathbf{k}} : \|f\|_{\mathbf{k}} \leq a\},$$

and let $\mathcal{C}_{\epsilon,\mathcal{F}}$ *be a set of minimum cardinality satisfying*

$$\mathcal{C}_{\epsilon,\mathcal{F}} \subset \mathbb{B}_{\mathcal{F}} \quad \text{and} \quad \sup_{f \in \mathcal{F}} \min_{f' \in \mathcal{C}_{\epsilon,\mathcal{F}}} \max_{\boldsymbol{x} \in \mathcal{X}_{\text{in}}} |f(\boldsymbol{x}) - f'(\boldsymbol{x})| \leq \varepsilon. \tag{15}$$

If $(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}$ *is* $(\mathbf{k}, \nu)$*-sub-Gaussian on an event* $\mathcal{E}$ *(Def. A.2), then, on* $\mathcal{E}$*,*

$$\|\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}\|_{\mathcal{F}} \triangleq \sup_{f \in \mathcal{F}} (\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}) f \leq 2\epsilon + \nu a \sqrt{2 \log(|\mathcal{C}_{\epsilon,\mathcal{F}}|/\delta')} \quad \textit{with probability at least } 1 - \delta'.$$

Proof. The triangle inequality and the covering property (15) together imply that, with probability 1,

$$\begin{aligned}
(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}) f &\leq \min_{f' \in \mathcal{C}_{\epsilon,\mathcal{F}}} (\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}) f' + |(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})(f - f')| \\
&\leq \|\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}\|_{\mathcal{C}_{\epsilon,\mathcal{F}}} + \min_{f' \in \mathcal{C}_{\epsilon,\mathcal{F}}} |\mathbb{P}_{\text{in}}(f - f')| + |\mathbb{P}_{\text{out}}(f - f')| \\
&\leq \|\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}\|_{\mathcal{C}_{\epsilon,\mathcal{F}}} + 2 \min_{f' \in \mathcal{C}_{\epsilon,\mathcal{F}}} \max_{\boldsymbol{x} \in \mathcal{X}_{\text{in}}} |f(\boldsymbol{x}) - f'(\boldsymbol{x})| \\
&\leq \|\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}\|_{\mathcal{C}_{\epsilon,\mathcal{F}}} + 2\varepsilon
\end{aligned} \tag{16}$$

for each $f \in \mathcal{F}$. Since $s \mapsto e^{ts}$ is increasing, the bound (16), the assumed sub-Gaussianity (Def. A.2), and the fact that $\mathcal{C}_{\epsilon,\mathcal{F}}$ belongs to $\mathbb{B}_{\mathcal{F}}$ imply that

$$\begin{aligned}
\mathbb{E}_{\mathcal{E}}[\exp(t\|\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}\|_{\mathcal{F}})] &\leq e^{2t\varepsilon} \mathbb{E}_{\mathcal{E}}[\exp(t\|\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}\|_{\mathcal{C}_{\epsilon,\mathcal{F}}})] \\
&\leq \textstyle\sum_{f' \in \mathcal{C}_{\epsilon,\mathcal{F}}} e^{2t\varepsilon} \mathbb{E}_{\mathcal{E}}[\exp(t(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}) f')] \\
&\leq \textstyle\sum_{f' \in \mathcal{C}_{\epsilon,\mathcal{F}}} \exp(\frac{t^2 \nu^2 \|f'\|_{\mathbf{k}}^2}{2} + 2t\epsilon) \leq |\mathcal{C}_{\epsilon,\mathcal{F}}| \exp(\frac{t^2 \nu^2 a^2}{2} + 2t\epsilon).
\end{aligned}$$

Now, by Markov's inequality (Markov, 1884), for any $\alpha > 0$,

$$\begin{aligned}
\mathbb{P}_{\mathcal{E}}(\sup_{f \in \mathcal{F}} (\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}) f > \alpha + 2\epsilon) &\leq \inf_{t > 0} \mathbb{E}_{\mathcal{E}}[\exp(t\|\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}\|_{\mathcal{F}})] / \exp(t(\alpha + 2\epsilon)) \\
&\leq |\mathcal{C}_{\epsilon,\mathcal{F}}| \inf_{t > 0} \exp(\tfrac{t^2 \nu^2 a^2}{2} - t\alpha) = |\mathcal{C}_{\epsilon,\mathcal{F}}| \exp(\tfrac{-\alpha^2}{2\nu^2 a^2}).
\end{aligned}$$

Finally, choosing $\alpha = \nu a \sqrt{2 \log(|\mathcal{C}_{\epsilon,\mathcal{F}}|/\delta')}$ yields the desired claim. □

Now fix any $\epsilon \geq 0$, $\delta' \in (0,1)$, and kernel $\mathbf{k}$ that generates $\mathbf{K}$, and consider the subset $\mathcal{F} = \{\pm \mathbf{k}(\boldsymbol{x}_i, \cdot) : i \in \mathcal{I}\}$. Since $\|\mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\|_{\mathcal{I}} = \|\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}}\|_{\mathcal{F}}$ and $\sup_{f \in \mathcal{F}} \|f\|_{\mathbf{k}} = D_{\mathcal{I}}$, Lem. C.1 implies that, on the event $\mathcal{E}$,

$$\|\mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\|_{\mathcal{I}} \leq 2\epsilon + \nu D_{\mathcal{I}} \sqrt{2 \log(|\mathcal{C}_{\epsilon,\mathcal{F}}|/\delta')} \quad \text{with probability at least } 1 - \delta'.$$

Since $\mathbb{P}(\mathcal{E}^c) \leq \delta/2$ and $|\mathcal{F}| \leq 2|\mathcal{Z}|$, we use the estimate $|\mathcal{C}_{0,\mathcal{F}}| \leq 2|\mathcal{I}|$ with $\epsilon = 0$ to obtain the advertised bound (2).

C.2. Proof of Lipschitz kernel max seminorm bound (3)

Introduce the query point set $\mathcal{Z} \triangleq \{\boldsymbol{x}_i : i \in \mathcal{I}\}$, fix any $\delta' \in (0,1)$ and $\boldsymbol{z}_0 \in \mathcal{Z}$, and define the symmetrized seminorm

$$\|(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}\|_{\mathcal{Z},\mathcal{Z}} \triangleq \sup_{\boldsymbol{z},\boldsymbol{z}' \in \mathcal{Z}} |(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}(\boldsymbol{z}) - (\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}(\boldsymbol{z}')|.$$

By the triangle inequality and the derivation of App. C.1, we have, on the event $\mathcal{E}$,

$$\begin{aligned}\|\mathbf{K}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\|_{\mathcal{I}} &\leq \|(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}\|_{\mathcal{Z},\mathcal{Z}} + |(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}(\boldsymbol{z}_0)| \\ &\leq \|(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}\|_{\mathcal{Z},\mathcal{Z}} + \nu \sqrt{\mathbf{k}(\boldsymbol{z}_0, \boldsymbol{z}_0)} \sqrt{2 \log(4/\delta')} \quad \text{with probability at least } 1 - \delta'/2. \end{aligned} \tag{17}$$

Since $\mathbb{P}(\mathcal{E}^c) \leq \delta/2$, it only remains to upper bound $\|(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}\|_{\mathcal{Z},\mathcal{Z}}$ on $\mathcal{E}$ with probability at least $1 - \delta'/2$.

To this end, we first establish that $((\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}(\boldsymbol{z}))_{\boldsymbol{z} \in \mathcal{Z}}$ is a sub-Gaussian process on $\mathcal{E}$ with respect to a particular bounded-Hölder metric ρ.

Definition C.1 (Sub-Gaussian process on an event)**.** *We say an indexed collection of random variables $(X_\theta)_{\theta \in \Theta}$ is a* sub-Gaussian process with respect to ρ on an event $\mathcal{E}$ *if ρ is a metric on Θ and*

$$\mathbb{E}_{\mathcal{E}}\left[\exp\left(\tfrac{(X_\theta - X'_\theta)^2}{\rho(\theta,\theta')^2}\right)\right] \leq 2 \quad \textit{for all} \quad \theta, \theta' \in \Theta.$$

Lemma C.2 (Bounded-Hölder sub-Gaussian process)**.** *Consider a kernel $\mathbf{k}$ on $\mathcal{X} = \mathbb{R}^d$ satisfying $|\mathbf{k}(\boldsymbol{z}, \boldsymbol{x}) - \mathbf{k}(\boldsymbol{z}', \boldsymbol{x})| \leq L_{\mathbf{k}} \|\boldsymbol{z} - \boldsymbol{z}'\|_2$ for all $\boldsymbol{z}, \boldsymbol{z}' \in \mathcal{Z} \subset \mathcal{X}$ and $\boldsymbol{x} \in \mathcal{X}_{\text{in}}$. If $(\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}$ is $(\mathbf{k}, \nu)$-sub-Gaussian on an event $\mathcal{E}$ (Def. A.2), then $((\mathbb{P}_{\text{in}} - \mathbb{P}_{\text{out}})\mathbf{k}(\boldsymbol{z}))_{\boldsymbol{z} \in \mathcal{Z}}$ is a sub-Gaussian process on $\mathcal{E}$ with respect to the metric*

$$\rho(\boldsymbol{z}, \boldsymbol{z}') \triangleq \nu \sqrt{8/3} \min(2 \sup_{\boldsymbol{z} \in \mathcal{Z}} \sqrt{\mathbf{k}(\boldsymbol{z}, \boldsymbol{z})}, \sqrt{2 L_{\mathbf{k}} \|\boldsymbol{z} - \boldsymbol{z}'\|_2}). \tag{18}$$

The proof of Lem. C.2 can be found in App. C.3. Our next lemma, a slight modification of Wainwright (2019, Thm. 5.36), bounds the suprema of symmetrized sub-Gaussian processes on an event in terms of covering numbers.

Lemma C.3 (Sub-Gaussian process tails)**.** *Suppose $(X_\theta)_{\theta \in \Theta}$ is a sub-Gaussian process with respect to ρ on an event $\mathcal{E}$, and define the diameter* $\operatorname{diam}(\Theta, \rho) \triangleq \sup_{\theta,\theta' \in \Theta} \rho(\theta, \theta')$*, the covering number*

$$\mathcal{N}(u; \Theta, \rho) \triangleq \min\{|\mathcal{C}_u| : \mathcal{C}_u \subseteq \Theta, \max_{\theta \in \Theta} \min_{\theta' \in \mathcal{C}_u} \rho(\theta, \theta') \leq u\} \quad \textit{for all } u > 0,$$

and the entropy integral $\mathcal{J}(\Theta, \rho) \triangleq \int_0^{\operatorname{diam}(\Theta,\rho)} \sqrt{\log(1 + \mathcal{N}(u; \Theta, \rho))}\, du$. Then,

$$\mathbb{P}_{\mathcal{E}}(\sup_{\theta,\theta' \in \Theta} |X_\theta - X_{\theta'}| \geq 8(\mathcal{J}(\Theta, \rho) + t)) \leq 2 \exp(-t^2/\operatorname{diam}(\Theta, \rho)^2) \quad \textit{for all} \quad t > 0.$$

Proof. Since $\sqrt{\log(1 + xy)} \leq \sqrt{\log((1+x)(1+y))} \leq \sqrt{\log(1+x)} + \sqrt{\log(1+y)}$ for all $x, y > 0$, the proof is identical to that of Wainwright (2019, Thm. 5.36) with $c_1 = 8$ and $(\mathbb{E}_{\mathcal{E}}, \mathbb{P}_{\mathcal{E}})$ substituted for $(\mathbb{E}, \mathbb{P})$. □

Our final lemma bounds the diameter, covering numbers, and entropy integral of $\mathcal{Z}$ using the metric ρ.

Lemma C.4 (Covering properties of bounded-Hölder metric)**.** *Consider the bounded-Hölder metric ρ (18) for a kernel $\mathbf{k}$ on $\mathcal{X} = \mathbb{R}^d$ and a finite set $\mathcal{Z} \subset \mathcal{X}$. If $\mathbf{Z}$ is a matrix with one row corresponding to each element of $\mathcal{Z}$, $r = \operatorname{rank}(\mathbf{Z})$, and $R = \max_{\boldsymbol{z} \in \mathcal{Z}} \|\boldsymbol{z}\|_2$, then, in the notation of Lem. C.3,*

$$\mathcal{N}(u; \mathcal{Z}, \rho) \leq (1 + c^2/u^2)^r \quad \textit{for} \quad c \triangleq \nu \sqrt{\tfrac{32}{3} R L_{\mathbf{k}}} \quad \textit{and all} \quad u > 0, \tag{19}$$

$$\operatorname{diam}(\mathcal{Z}, \rho) \leq D \triangleq \min(c, \nu \sqrt{\tfrac{32}{3}} \max_{\boldsymbol{z} \in \mathcal{Z}} \sqrt{\mathbf{k}(\boldsymbol{z}, \boldsymbol{z})}), \quad \textit{and} \tag{20}$$

$$\mathcal{J}(\mathcal{Z}, \rho) \leq D \sqrt{2r \log(\sqrt{3} e c / D)}.$$

Proof. The diameter bound (20) follows directly from the definition of ρ (18) and the fact $\max_{\boldsymbol{z},\boldsymbol{z}'\in\mathcal{Z}} \|\boldsymbol{z}-\boldsymbol{z}'\|_2 \le 2R$.

To establish the covering number bound (19), we let $\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ be a compact singular value decomposition of $\mathbf{Z}$ so that

$$\mathbf{V}\in\mathbb{R}^{d\times r}, \quad \mathbf{Z}=\mathbf{Z}\mathbf{V}\mathbf{V}^\top, \quad \text{and} \quad \max_{\boldsymbol{z}\in\mathcal{Z}} \|\mathbf{V}^\top\boldsymbol{z}\|_2 = \max_{\boldsymbol{z}\in\mathcal{Z}} \|\boldsymbol{z}\|_2 = R.$$

Fix any $\epsilon>0$, and let $\mathcal{C}$ and $\mathcal{C}_{\text{ext}}$ be a sets of minimum cardinality satisfying

$$\begin{aligned} \mathcal{C}\subset\mathbb{B}^r(R), \qquad & \max_{\boldsymbol{v}\in\mathbb{B}^r(R)} \min_{\boldsymbol{v}'\in\mathcal{C}} \|\boldsymbol{v}'-\boldsymbol{v}\|_2 \le \epsilon^2/2, \\ \mathcal{C}_{\text{ext}}\subset\mathbb{B}^d(R), \quad \text{and} \quad & \max_{\boldsymbol{z}\in\mathcal{Z}} \min_{\boldsymbol{z}'\in\mathcal{C}_{\text{ext}}} \|\boldsymbol{z}'-\boldsymbol{z}\|_2 \le \epsilon^2/2. \end{aligned} \tag{21}$$

Since $\mathbf{V}^\top\boldsymbol{z}\in\mathbb{B}^r(R)$ for each $\boldsymbol{z}\in\mathcal{Z}$ and $\mathbf{V}\boldsymbol{v}'\in\mathbb{B}^d$ for each $\boldsymbol{v}'\in\mathbb{B}^r$, we have

$$\begin{aligned} \max_{\boldsymbol{z}\in\mathcal{Z}} \min_{\boldsymbol{v}'\in\mathcal{C}} \|\mathbf{V}\boldsymbol{v}'-\boldsymbol{z}\|_2 &= \max_{\boldsymbol{z}\in\mathcal{Z}} \min_{\boldsymbol{v}'\in\mathcal{C}} \|\mathbf{V}(\boldsymbol{v}'-\mathbf{V}^\top\boldsymbol{z})\|_2 \\ &= \max_{\boldsymbol{z}\in\mathcal{Z}} \min_{\boldsymbol{v}'\in\mathcal{C}} \|\boldsymbol{v}'-\mathbf{V}^\top\boldsymbol{z}\|_2 \le \epsilon^2/2, \end{aligned}$$

so that $\mathbf{V}\mathcal{C}$ satisfies the criteria of (21). Since $|\mathbf{V}\mathcal{C}|\le|\mathcal{C}|\le(1+4R/\epsilon^2)^r$ by Wainwright (2019, Lem. 5.2), we must also have $|\mathcal{C}_{\text{ext}}|\le(1+4R/\epsilon^2)^r$.

Now, since $\mathcal{C}_{\text{ext}}$ has minimum cardinality amongst sets satisfying (21), for each $\boldsymbol{z}'\in\mathcal{C}_{\text{ext}}$, there is some $\boldsymbol{z}\in\mathcal{Z}$ satisfying $\|\boldsymbol{z}'-\boldsymbol{z}\|_2\le\epsilon^2/2$ (or else $\boldsymbol{z}'$ would be superfluous). Hence, there exists a set $\mathcal{C}_{\text{int}}\subseteq\mathcal{Z}$ satisfying

$$|\mathcal{C}_{\text{int}}|\le|\mathcal{C}_{\text{ext}}|\le(1+4R/\epsilon^2)^r \quad \text{and} \quad \max_{\boldsymbol{z}\in\mathcal{Z}} \min_{\boldsymbol{z}'\in\mathcal{C}_{\text{int}}} \|\boldsymbol{z}'-\boldsymbol{z}\|_2 \le \epsilon^2.$$

Moreover, by our metric definition (18),

$$\max_{\boldsymbol{z}\in\mathcal{Z}} \min_{\boldsymbol{z}'\in\mathcal{C}_{\text{int}}} \rho(\boldsymbol{z},\boldsymbol{z}') \le \tfrac{c}{2\sqrt{R}} \max_{\boldsymbol{z}\in\mathcal{Z}} \min_{\boldsymbol{z}'\in\mathcal{C}_{\text{int}}} \sqrt{\|\boldsymbol{z}-\boldsymbol{z}'\|_2} \le \tfrac{c\epsilon}{2\sqrt{R}}.$$

Hence, for $u=\frac{c\epsilon}{2\sqrt{R}}$, $\mathcal{N}(u;\mathcal{Z},\rho)\le|\mathcal{C}_{\text{int}}|\le(1+c^2/u^2)^r$. Since $\epsilon>0$ was arbitrary, we have established (19).

Finally, we bound the entropy integral using the inequality $1\le c^2/u^2$ for $u\in[0,D]$, the concavity of the square-root function, and Jensen's inequality:

$$\begin{aligned} \mathcal{J}(\mathcal{Z},\rho) &\le \int_0^D \sqrt{\log(1+(1+c^2/u^2)^r)}\,du \le \int_0^D \sqrt{\log((3c^2/u^2)^r)}\,du = \int_0^D \sqrt{2r\log(\sqrt{3}c/u)}\,du \\ &\le D\sqrt{\tfrac{1}{D}\int_0^D 2r\log(\sqrt{3}c/u)\,du} = D\sqrt{2r\log(\sqrt{3}ec/D)}. \end{aligned}$$

□

Together, Lems. C.2, C.3, and C.4 imply that, in the notation of Lem. C.4,

$$\|(\mathbb{P}_{\text{in}}-\mathbb{P}_{\text{out}})\mathbf{k}\|_{\mathcal{Z},\mathcal{Z}} \le 8D\sqrt{2r\log(\sqrt{3}ec/D)} + 8D\sqrt{\log(4/\delta')}$$

on $\mathcal{E}$ with probability at least $1-\delta'/2$. Combining this bound with the inequality (17) yields the result.

C.3. Proof of Lem. C.2: Bounded-Hölder sub-Gaussian process

Define $X_{\boldsymbol{z}}=(\mathbb{P}_{\text{in}}-\mathbb{P}_{\text{out}})\mathbf{k}(\boldsymbol{z})$ for each $\boldsymbol{z}\in\mathcal{Z}$, and fix any $\boldsymbol{z},\boldsymbol{z}'\in\mathcal{Z}$. Our sub-Gaussianity assumption implies

$$\mathbb{E}_{\mathcal{E}}[\exp(\lambda(X_{\boldsymbol{z}}-X_{\boldsymbol{z}'})] \le \exp(\tfrac{\nu^2\lambda^2}{2}\|\mathbf{k}(\boldsymbol{z},\cdot)-\mathbf{k}(\boldsymbol{z}',\cdot)\|_{\mathbf{k}}^2) \quad \text{for all} \quad \lambda\in\mathbb{R}.$$

Moreover, by our Lipschitz assumption,

$$\|\mathbf{k}(\boldsymbol{z},\cdot)-\mathbf{k}(\boldsymbol{z}',\cdot)\|_{\mathbf{k}}^2 = \mathbf{k}(\boldsymbol{z},\boldsymbol{z})-\mathbf{k}(\boldsymbol{z},\boldsymbol{z}')+\mathbf{k}(\boldsymbol{z}',\boldsymbol{z}')-\mathbf{k}(\boldsymbol{z}',\boldsymbol{z}) \le \min(4\max_{\boldsymbol{z}\in\mathcal{Z}}\mathbf{k}(\boldsymbol{z},\boldsymbol{z}), 2L_{\mathbf{k}}\|\boldsymbol{z}-\boldsymbol{z}'\|_2).$$

Finally, Lem. C.5 shows that $\mathbb{E}_{\mathcal{E}}[\exp(\frac{(X_{\boldsymbol{z}}-X_{\boldsymbol{z}'})^2}{\rho(\boldsymbol{z},\boldsymbol{z}')^2}]\le 2$ so that $(X_{\boldsymbol{z}})_{\boldsymbol{z}\in\mathcal{Z}}$ is a sub-Gaussian process on $\mathcal{E}$ with respect to ρ.

Lemma C.5 (Squared exponential moment bound)**.** *If* $\mathbb{E}_{\mathcal{E}}[\exp(\lambda X)]\le\exp(\frac{\nu^2\lambda^2}{2})$ *for all* $\lambda\in\mathbb{R}$*, then* $\mathbb{E}_{\mathcal{E}}[\exp(\frac{3X^2}{8\nu^2})]\le 2$.

Proof. The proof is identical to that in Wainwright (2019, Sec. 2.4) with $\mathbb{E}_{\mathcal{E}}$ substituted for $\mathbb{E}$. □

D. Proof of Thm. 2: Quality of Thinformer

Throughout we will make use of the convenient representation

$$\widehat{\mathbf{T}} = \widehat{\mathbf{D}}^{-1}\widehat{\mathbf{A}}\mathbf{V} \text{ for } \mathcal{I}_{\text{out}} \triangleq \{i \in [n] : (\tilde{\boldsymbol{k}}_i, \tilde{\boldsymbol{v}}_i) \in \mathcal{X}_{\text{out}}\}, \quad \widehat{\mathbf{A}} \triangleq \frac{n}{n_{\text{out}}}(\exp(\frac{\langle \boldsymbol{q}_i, \boldsymbol{k}_j\rangle}{\sqrt{d}})\mathbf{1}[j \in \mathcal{I}_{\text{out}}])_{i,j=1}^n, \text{ and } \widehat{\mathbf{D}} \triangleq \widehat{\mathbf{A}}\mathbf{1}_n. \quad (22)$$

Our proof makes use of three lemmas. The first, proved in App. D.1, bounds the approximation error for the attention matrix $\mathbf{T}$ in terms of the approximation error for $\mathbf{AV}$ and $\mathbf{A}\mathbf{1}_n$.

Lemma D.1 (Decomposing attention approximation error)**.** *In the notation of Alg. 1 and (22),*

$$\|\widehat{\mathbf{D}}^{-1}\widehat{\mathbf{A}}\mathbf{V} - \mathbf{D}^{-1}\mathbf{A}\mathbf{V}\|_{\max} \leq \min\left(\|(\tfrac{1}{n}\mathbf{D})^{-1}\|_{\max}, \|(\tfrac{1}{n}\widehat{\mathbf{D}})^{-1}\|_{\max}\right)(\tfrac{1}{n}\|\widehat{\mathbf{A}}\mathbf{V} - \mathbf{A}\mathbf{V}\|_{\max} + \tfrac{1}{n}\|\mathbf{A}\mathbf{1}_n - \widehat{\mathbf{A}}\mathbf{1}_n\|_\infty \|\mathbf{V}\|_{\max}).$$

The second, proved in App. D.2, bounds the approximation error for $\mathbf{AV}$ and $\mathbf{A}\mathbf{1}_n$ in terms of the KMS (1) for a specific choice of attention kernel matrix.

Lemma D.2 (KMS bound on attention approximation error)**.** *Instantiate the notation of Alg. 1 and (22) and define the query set*

$$\mathcal{X}' \triangleq \{\boldsymbol{x}_{i+nj} \triangleq (\tilde{\boldsymbol{q}}_i, \boldsymbol{e}_j^{d+1}) : i \in [n], j \in [d+1]\} \quad \textit{where} \quad \tilde{\boldsymbol{q}}_i \triangleq \boldsymbol{q}_i/d^{\frac{1}{4}}$$

and $\boldsymbol{e}_j^{d+1}$ is the j-th standard basis vector in $\mathbb{R}^{d+1}$. If $\mathbf{K}_{\text{att}} \triangleq \mathbf{k}_{\text{att}}(\mathcal{X}, \mathcal{X})$ for $\mathcal{X} \triangleq \mathcal{X}' \cup \mathcal{X}_{\text{in}}$, then

$$\max\left(\tfrac{1}{n}\|(\widehat{\mathbf{A}} - \mathbf{A})\mathbf{V}\|_{\max}, \tfrac{1}{n}\|(\widehat{\mathbf{A}} - \mathbf{A})\mathbf{1}_n\|_\infty \|\mathbf{V}\|_{\max}\right) = \|\mathbf{K}_{\text{att}}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\|_{\mathcal{I}} \quad \textit{for} \quad \mathcal{I} \triangleq [n(d+1)].$$

Our third lemma, proved in App. D.3, bounds the size of key parameters of the thinned attention problem.

Lemma D.3 (Thinned attention problem parameters)**.** *Instantiate the notation of Lem. D.2, and define $R \triangleq \max_{i\in[n]} \max(\|\boldsymbol{q}_i\|_2, \|\boldsymbol{k}_i\|_2)$. Then, for all $i, j \in \mathcal{I}$ and $l \in \text{supp}(\boldsymbol{p}_{\text{in}})$,*

$$\begin{aligned}
&\|(\tfrac{1}{n}\mathbf{D})^{-1}\|_{\max} \leq \exp(\tfrac{R^2}{\sqrt{d}}), \quad \max_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}} \sqrt{\mathbf{k}_{\text{att}}(\boldsymbol{x}, \boldsymbol{x})} \leq \exp(\tfrac{R^2}{2\sqrt{d}})\sqrt{\|\mathbf{V}\|_{2,\infty}^2 + \|\mathbf{V}\|_{\max}^2},\\
&R_{\mathcal{I}} \triangleq \max_{i\in\mathcal{I}} \|\boldsymbol{x}_i\|_2 \leq \sqrt{\tfrac{R^2}{\sqrt{d}} + 1}, \quad D_{\mathcal{I}} \triangleq \max_{i\in\mathcal{I}} \sqrt{\mathbf{K}_{\text{att},ii}} \leq \exp(\tfrac{R^2}{2\sqrt{d}}),\\
&\text{rank}(\mathbf{X}_{\mathcal{I}}) \leq d+1 \quad \textit{for} \quad \mathbf{X}_{\mathcal{I}} \triangleq [\boldsymbol{x}_i]_{i\in\mathcal{I}}^\top, \quad \textit{and}\\
&|\mathbf{K}_{\text{att},il} - \mathbf{K}_{\text{att},jl}| \leq L_{\mathbf{K}_{\text{att}}}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2 \quad \textit{for} \quad L_{\mathbf{K}_{\text{att}}} \triangleq \exp(\tfrac{R^2}{\sqrt{d}})\sqrt{\tfrac{R^2}{\sqrt{d}} + 2}\|\mathbf{V}\|_{\max}.
\end{aligned}$$

Now instantiate the notation of Lem. D.2, and define the coefficient

$$c \triangleq 2\sqrt{2}\Big(32\sqrt{\tfrac{2}{3}(d+1)\log(3e^2(\tfrac{R^2}{\sqrt{d}} + 2)\|\mathbf{V}\|_{\max})} + \sqrt{2\log(8)}(1 + \tfrac{32}{\sqrt{3}})\Big).$$

Together, Lem. D.3, the KMS quality bound of Thm. 1, and the KH-COMPRESS(0.5) sub-Gaussian constant ν of Prop. B.5 imply that, with probability at least $\frac{1}{2}$,

$$\|\mathbf{K}_{\text{att}}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\|_{\mathcal{I}} \leq \tfrac{c}{2\sqrt{2}}\exp(\tfrac{R^2}{\sqrt{d}})\sqrt{\|\mathbf{V}\|_{2,\infty}^2 + \|\mathbf{V}\|_{\max}^2}\,\frac{\sqrt{\log_2(n_{\text{out}})\log(8n_{\text{out}}\log_2\frac{n_{\text{in}}}{n_{\text{out}}})}}{n_{\text{out}}}.$$

Hence, by Lems. D.1 and D.2, with probability at least $\frac{1}{2}$,

$$\begin{aligned}
\|\widehat{\mathbf{D}}^{-1}\widehat{\mathbf{A}}\mathbf{V} - \mathbf{D}^{-1}\mathbf{A}\mathbf{V}\|_{\max} &\leq \tfrac{c}{\sqrt{2}}\exp(\tfrac{2R^2}{\sqrt{d}})\sqrt{\|\mathbf{V}\|_{2,\infty}^2 + \|\mathbf{V}\|_{\max}^2}\,\frac{\sqrt{\log_2(n_{\text{out}})\log(8n_{\text{out}}\log_2\frac{n_{\text{in}}}{n_{\text{out}}})}}{n_{\text{out}}}\\
&\leq c\exp(\tfrac{2R^2}{\sqrt{d}})\|\mathbf{V}\|_{2,\infty}\frac{\sqrt{\log_2(n_{\text{out}})\log(8n_{\text{out}}\log_2\frac{n_{\text{in}}}{n_{\text{out}}})}}{n_{\text{out}}}.
\end{aligned}$$

D.1. Proof of Lem. D.1: Decomposing attention approximation error

By the triangle inequality, we have

$$\|\widehat{\mathbf{D}}^{-1}\widehat{\mathbf{A}}\mathbf{V} - \mathbf{D}^{-1}\mathbf{A}\mathbf{V}\|_{\max} \leq \|\widehat{\mathbf{D}}^{-1}\widehat{\mathbf{A}}\mathbf{V} - \widehat{\mathbf{D}}^{-1}\mathbf{A}\mathbf{V}\|_{\max} + \|\widehat{\mathbf{D}}^{-1}\mathbf{A}\mathbf{V} - \mathbf{D}^{-1}\mathbf{A}\mathbf{V}\|_{\max}.$$

We bound the first term on the right-hand side using the submultiplicativity of the max norm under diagonal rescaling:

$$\|\widehat{\mathbf{D}}^{-1}\widehat{\mathbf{A}}\mathbf{V} - \widehat{\mathbf{D}}^{-1}\mathbf{A}\mathbf{V}\|_{\max} \leq \|\widehat{\mathbf{D}}^{-1}\|_{\max}\|\widehat{\mathbf{A}}\mathbf{V} - \mathbf{A}\mathbf{V}\|_{\max} = \|(\tfrac{1}{n}\widehat{\mathbf{D}})^{-1}\|_{\max}\tfrac{1}{n}\|\widehat{\mathbf{A}}\mathbf{V} - \mathbf{A}\mathbf{V}\|_{\max}.$$

To bound the second term we use the same submultiplicativity property and the fact that each entry of $\mathbf{D}^{-1}\mathbf{A}\mathbf{V}$ is the average of values in $\mathbf{V}$:

$$\begin{aligned}\|\widehat{\mathbf{D}}^{-1}\mathbf{A}\mathbf{V} - \mathbf{D}^{-1}\mathbf{A}\mathbf{V}\|_{\max} &= \|\widehat{\mathbf{D}}^{-1}(\mathbf{D} - \widehat{\mathbf{D}})\mathbf{D}^{-1}\mathbf{A}\mathbf{V}\|_{\max} \leq \|\widehat{\mathbf{D}}^{-1}\|_{\max}\|\mathbf{D} - \widehat{\mathbf{D}}\|_{\max}\|\mathbf{D}^{-1}\mathbf{A}\mathbf{V}\|_{\max} \\ &= \|(\tfrac{1}{n}\widehat{\mathbf{D}})^{-1}\|_{\max}\tfrac{1}{n}\|\mathbf{A}\mathbf{1}_n - \widehat{\mathbf{A}}\mathbf{1}_n\|_\infty\|\mathbf{V}\|_{\max}.\end{aligned}$$

An identical argument reversing the roles of $(\mathbf{D}, \mathbf{A})$ and $(\widehat{\mathbf{D}}, \widehat{\mathbf{A}})$ yields the second bound.

D.2. Proof of Lem. D.2: KMS bound on attention approximation error

Define the augmented value matrix $\widetilde{\mathbf{V}} = [\mathbf{V}, \|\mathbf{V}\|_{\max}\mathbf{1}_n] \in \mathbb{R}^{d+1}$. By the definition of $\mathbf{K}_{\text{att}}$ and $\widehat{\mathbf{A}}$,

$$\|\mathbf{K}_{\text{att}}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})\|_{\mathcal{I}} = \max_{i\in[n],j\in[d+1]} |\textstyle\sum_{\ell\in[n]} \mathbf{A}_{i\ell}\widetilde{\mathbf{V}}_{\ell j}(\boldsymbol{p}_{\text{in}} - \boldsymbol{p}_{\text{out}})_\ell| = \tfrac{1}{n}\|(\mathbf{A} - \widehat{\mathbf{A}})\widetilde{\mathbf{V}}\boldsymbol{e}_j^d\|_\infty = \tfrac{1}{n}\|(\mathbf{A} - \widehat{\mathbf{A}})\widetilde{\mathbf{V}}\|_{\max}.$$

D.3. Proof of Lem. D.3: Thinned attention problem parameters

First, by the Cauchy-Schwarz inequality and the nonnegativity of $\mathbf{D} = \mathbf{A}\mathbf{1}_n$ we have

$$\|(\tfrac{1}{n}\mathbf{D})^{-1}\|_{\max} = \frac{1}{\min_{i\in[n]}\frac{1}{n}\sum_{j\in[n]}\mathbf{A}_{ij}} \leq \frac{1}{\min_{i\in[n],j\in[n]}\exp(\frac{\langle \boldsymbol{q}_i, \boldsymbol{k}_j\rangle}{\sqrt{d}})} \leq \frac{1}{\min_{i\in[n],j\in[n]}\exp(\frac{-\|\boldsymbol{q}_i\|_2\|\boldsymbol{k}_j\|_2}{\sqrt{d}})} \leq \exp(\tfrac{R^2}{\sqrt{d}}).$$

Second, the $\max_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\sqrt{\mathbf{k}_{\text{att}}(\boldsymbol{x},\boldsymbol{x})}$ inequality follows as

$$\mathbf{k}_{\text{att}}((\tilde{\boldsymbol{k}}_i, \tilde{\boldsymbol{v}}_i), (\tilde{\boldsymbol{k}}_i, \tilde{\boldsymbol{v}}_i)) = \exp(\tfrac{\|\boldsymbol{k}_i\|_2^2}{\sqrt{d}})(\|\boldsymbol{v}_i\|_2^2 + \|\mathbf{V}\|_{\max}^2) \leq \exp(\tfrac{R^2}{\sqrt{d}})(\|\mathbf{V}\|_{2,\infty}^2 + \|\mathbf{V}\|_{\max}^2).$$

Third, the $R_{\mathcal{I}}$ inequality follows as

$$\|(\tilde{\boldsymbol{q}}_i, \boldsymbol{e}_j^{d+1})\|_2 = \sqrt{\|\tilde{\boldsymbol{q}}_i\|_2^2 + 1} \leq \sqrt{\tfrac{R^2}{\sqrt{d}} + 1} \quad \text{for all} \quad i \in [n], j \in [d+1].$$

Fourth, the $D_{\mathcal{I}}$ inequality follows as

$$\max_{i\in\mathcal{I}}\mathbf{K}_{\text{att},ii} = \max_{i\in[n]}\exp(\tfrac{\|\boldsymbol{q}_i\|_2^2}{\sqrt{d}}) \leq \exp(\tfrac{R^2}{\sqrt{d}}).$$

Fifth, the rank inequality follows as $\boldsymbol{x}_i \in \mathbb{R}^{d+1}$ for $i \in \mathcal{I}$. Finally, the Lipschitz inequality follows as, for any $i, k, l \in [n]$ and $j, m \in [d+1]$,

$$\begin{aligned}
&|\exp(\tfrac{\langle \boldsymbol{q}_i, \boldsymbol{k}_l\rangle}{\sqrt{d}})\langle \boldsymbol{e}_j^{d+1}, \tilde{\boldsymbol{v}}_l\rangle - \exp(\tfrac{\langle \boldsymbol{q}_k, \boldsymbol{k}_l\rangle}{\sqrt{d}})\langle \boldsymbol{e}_m^{d+1}, \tilde{\boldsymbol{v}}_l\rangle| \\
&\leq \exp(\tfrac{\langle \boldsymbol{q}_i, \boldsymbol{k}_l\rangle}{\sqrt{d}})|\tilde{\boldsymbol{v}}_{lj} - \tilde{\boldsymbol{v}}_{lm}| + |\exp(\tfrac{\langle \boldsymbol{q}_i, \boldsymbol{k}_l\rangle}{\sqrt{d}}) - \exp(\tfrac{\langle \boldsymbol{q}_k, \boldsymbol{k}_l\rangle}{\sqrt{d}})||\tilde{\boldsymbol{v}}_{lm}| \\
&\leq \exp(\tfrac{\|\boldsymbol{q}_i\|_2\|\boldsymbol{k}_l\|_2}{\sqrt{d}})\|\boldsymbol{e}_j^{d+1} - \boldsymbol{e}_m^{d+1}\|_2\tfrac{|\tilde{\boldsymbol{v}}_{lj} - \tilde{\boldsymbol{v}}_{lm}|}{\sqrt{2}} + \exp(\tfrac{\max(\|\boldsymbol{q}_i\|_2, \|\boldsymbol{q}_k\|_2)\|\boldsymbol{k}_l\|_2}{\sqrt{d}})|\tfrac{\langle \boldsymbol{q}_i - \boldsymbol{q}_k, \boldsymbol{k}_l\rangle}{\sqrt{d}}||\tilde{\boldsymbol{v}}_{lm}| \\
&\leq \exp(\tfrac{R^2}{\sqrt{d}})\|\boldsymbol{e}_j^{d+1} - \boldsymbol{e}_m^{d+1}\|_2\tfrac{|\tilde{\boldsymbol{v}}_{lj} - \tilde{\boldsymbol{v}}_{lm}|}{\sqrt{2}} + \exp(\tfrac{R^2}{\sqrt{d}})\tfrac{\|\boldsymbol{q}_i - \boldsymbol{q}_k\|_2 R}{\sqrt{d}}|\tilde{\boldsymbol{v}}_{lm}| \\
&\leq \exp(\tfrac{R^2}{\sqrt{d}})\|\boldsymbol{e}_j^{d+1} - \boldsymbol{e}_m^{d+1}\|_2\sqrt{2}\|\mathbf{V}\|_{\max} + \exp(\tfrac{R^2}{\sqrt{d}})\tfrac{\|\boldsymbol{q}_i - \boldsymbol{q}_k\|_2 R}{\sqrt{d}}\|\mathbf{V}\|_{\max} \\
&\leq \exp(\tfrac{R^2}{\sqrt{d}})\sqrt{\tfrac{R^2}{\sqrt{d}} + 2}\|\mathbf{V}\|_{\max}\|(\tilde{\boldsymbol{q}}_i, e_j^{d+1}) - (\tilde{\boldsymbol{q}}_k, \boldsymbol{e}_m^{d+1})\|_2
\end{aligned}$$

by the triangle inequality, multiple applications of Cauchy-Schwarz, and the mean-value theorem applied to $x \mapsto e^x$.

E. Supplementary Experiment Details

The T2T-ViT experiment of Sec. 4.2 was carried out using Python 3.12.9, PyTorch 2.8.0.dev20250407+cu128 (Paszke et al., 2019), and an Ubuntu 22.04.5 LTS server with an AMD EPYC 7V13 64-Core Processor, 220 GB RAM, and a single NVIDIA A100 GPU (80 GB memory, CUDA 12.8, driver version 570.124.04). For reference, attention layer 1 has $(n, d) = (3136, 64)$ and attention layer 2 has $(n, d) = (784, 64)$. For each layer and each of the first 50 ImageNet 2012 validation set batches of size 64, we measured the time required to complete a forward pass through the layer using CUDA events following 10 warm-up batches to initialize the GPU. Tab. E.1 provides the hyperparameter settings for each attention approximation in Tab. 3. The settings and implementations for all methods other than Thinformer were provided by Zandieh et al. (2023), and our experiment code builds on their open-source repository `https://github.com/majid-daliri/kdeformer`.

Table E.1: **Configurations for the attention approximation methods of Tab. 3.**

Attention Algorithm	Layer 1 Configuration	Layer 2 Configuration
Performer	`num_features=49`	`num_features=12`
Reformer	`bucket_size=49` `n_hashes=2`	`bucket_size=12` `n_hashes=2`
ScatterBrain	`local_context=49` `num_features=48`	`local_context=12` `num_features=6`
KDEformer	`sample_size=64` `bucket_size=32`	`sample_size=56` `bucket_size=32`
Thinformer (Ours)	`g=2`	`g=4`