Bounded Random Features for Grassmannian Kernels with Rank-One Projections

Rémi Delogne and Laurent Jacques

December 2026

Introduction

Why subspaces?

Many high-dimensional signals (images videos) live in *low-dimensional linear subspaces*.

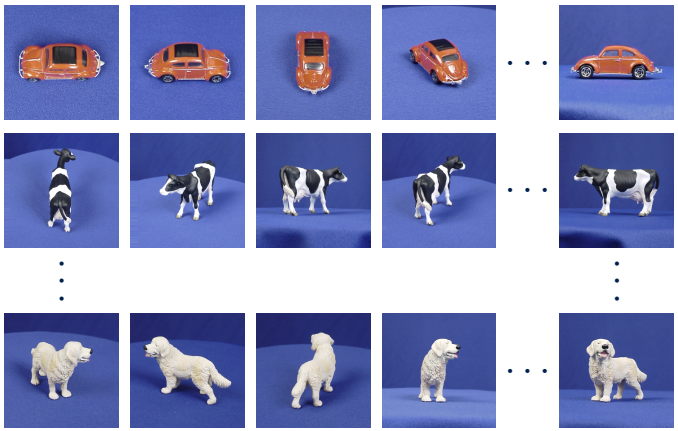
Examples include image illumination changes, video motion, signal processing, etc.

Subspaces capture *global structure* rather than characteristics of individual samples.

This leads naturally to working on the Grassmann manifold: $\mathcal{G}(k, n)$.

Goal: accelerate and develop existing kernel methods with random features.

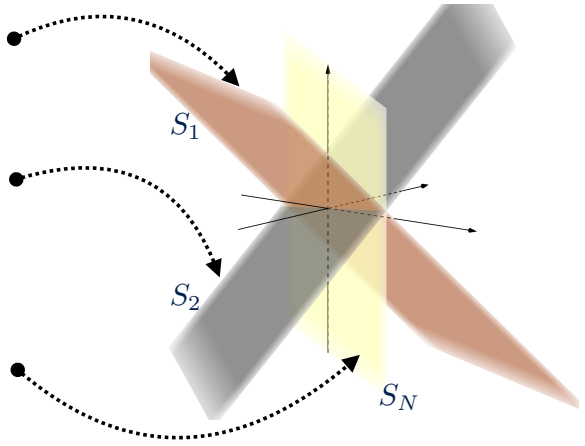
From points to subspaces



$$\left\{\boldsymbol{x}_j^{(1)}\right\}_{j=1}^{M} \subset S_1$$

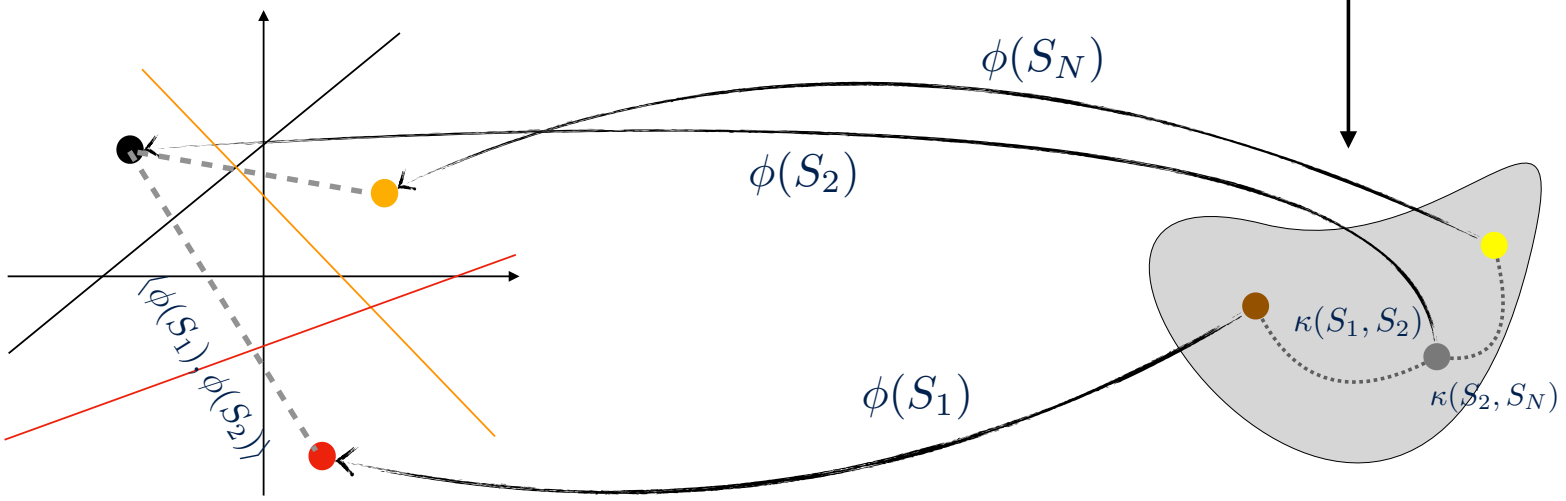
$$\left\{\boldsymbol{x}_j^{(2)}\right\}_{j=1}^{M} \subset S_2$$

$$\left\{\boldsymbol{x}_j^{(N)}\right\}_{j=1}^{M} \subset S_N$$



Images in $\mathbb{R}^n$

Subspaces of $\mathbb{R}^n$



Euclidean Space $\mathbb{R}^m$ Embedding Grassmannian $\mathcal{G}(k, n)$

Grassmannian kernels are powerful...

Classical Grassmanian kernels[1]:

Projection kernel κ^{pro} and Binet-Cauchy kernel κ^{bc}.

Both depend only on principal angles between subspaces.

Invariant under rotation, elegant, and effective for classification, clustering, and detection tasks.

However: in practice kernel methods such as this do not scale well (Gram matrix), computing them can be expensive.

[1] Mehrtash T Harandi et al. "Expanding the family of grassmannian kernels: An embedding perspective". In: *European conference on computer vision*. Springer. 2014, pp. 408–423.

. . . but they do not scale

For a dataset of size N, Gram matrices require $\mathcal{O}(N^2)$ memory and evaluations.

Projection kernel evaluation involves $\mathcal{O}(nk^2)$ computations.

Binet-Cauchy kernel evaluation costs $\mathcal{O}(nk^2 + k^3)$

For high-dimensional data, this quickly becomes impractical.

We need scalable, geometry-aware Grassmann kernels.

Our goal

Use random projections on projectors to embed subspaces. In particular, use rank-one projections (AROP).

Apply bounded non-linear functions (sign, complex exponential) to stabilise the features.

Obtain Grassmannian kernels that admit random feature approximations.

Structured and hardware variants (FFT based, OPU) further reduce cost.

Random features + boundedness (+ structure) = scalable Grassmann kernels.

Main results

- Two Grassmannian kernels from bounded AROP features: sign and complex exponential.
- Uniform approximation guarantees for the corresponding random feature estimators.
- FFT-based structured sketches to reduce storage and computational cost.
- ETH-80 experiments to evaluate the approximations.

Background

The Grassmannian $\mathcal{G}(k, n)$

Elements are k-dimensional linear subspaces of $\mathbb{R}^n$.

Any subspace is represented by a matrix $\boldsymbol{U} \in \mathbb{R}^{n \times k}$ with orthonormal columns.

Projector representation: $\boldsymbol{P} = \boldsymbol{U}\boldsymbol{U}^\top$, which is invariant to changes of basis $\boldsymbol{U} \to \boldsymbol{U}\boldsymbol{R}$ for a rotation matrix $\boldsymbol{R}$.

Projectors make kernel design easier: linear, basis-free, geometric.

Notation: We use $\boldsymbol{P}$ and $\boldsymbol{U}$ interchangeably. In computations only $\boldsymbol{U}$ is used for computational efficiency.

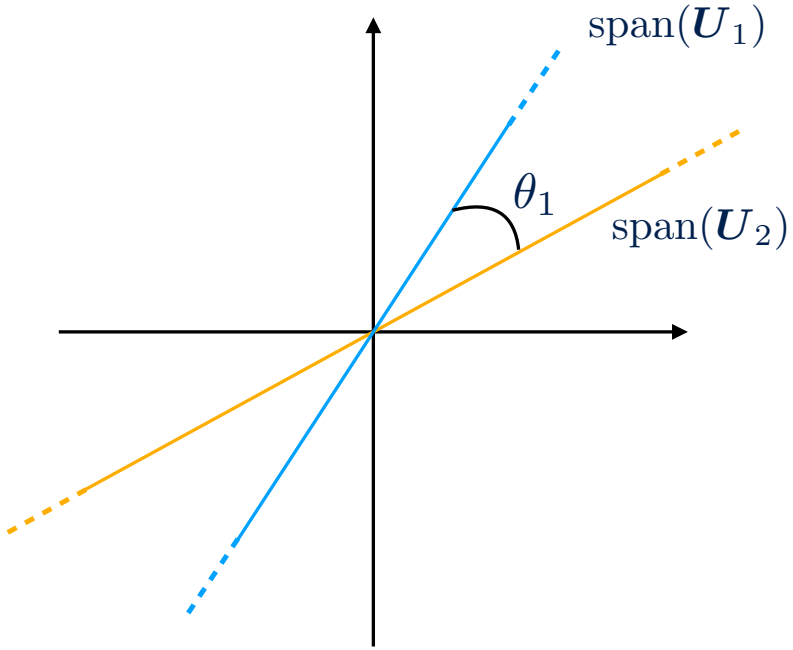
Principal angles

Principal angles measure how much two subspaces "align".

Given $\boldsymbol{U}_1$ and $\boldsymbol{U}_2$ orthonormal, the k-th principal angle θ_k is:

$$\cos\theta_k = \max_{\substack{\boldsymbol{u}_k \in \mathrm{span}(\boldsymbol{U}_1),\, \|\boldsymbol{u}_k\|=1 \\ \boldsymbol{v}_k \in \mathrm{span}(\boldsymbol{U}_2),\, \|\boldsymbol{v}_k\|=1}} \boldsymbol{u}_k^\top \boldsymbol{v}_k$$

with $\boldsymbol{u}_k \perp \{\boldsymbol{u}_i\}_{i<k}, \boldsymbol{v}_k \perp \{\boldsymbol{v}_i\}_{i<k}$.



Classical Grassmannian kernels

Projection kernel:

$$\kappa^{\mathrm{pro}}(\boldsymbol{U}_1, \boldsymbol{U}_2) = \|\boldsymbol{U}_1^\top \boldsymbol{U}_2\|_F^2 = \sum_{j=1}^{k} \cos^2(\theta_j).$$

Binet-Cauchy kernel:

$$\kappa^{\mathrm{bc}}(\boldsymbol{U}_1, \boldsymbol{U}_2) = \det(\boldsymbol{U}_1^\top \boldsymbol{U}_2)^2 = \prod_{j=1}^{k} \cos^2(\theta_j).$$

Why are they kernels

Kernels $\leftrightarrow$ inner products after embedding.

- Projection kernel: $\phi(\boldsymbol{U}) = \boldsymbol{U}\boldsymbol{U}^\top = \boldsymbol{P}$.
- Binet-Cauchy kernel: $\phi(\boldsymbol{U}) = \text{Plücker}(\boldsymbol{P})$.

Random features for scalability

Random features approximate a kernel via an explicit m-dimensional embedding:

$$\kappa(\boldsymbol{P}_1, \boldsymbol{P}_2) \approx \langle \phi(\boldsymbol{P}_1), \phi(\boldsymbol{P}_2) \rangle.$$

Linear classification becomes possible, thus avoid storing the full Gram matrix.

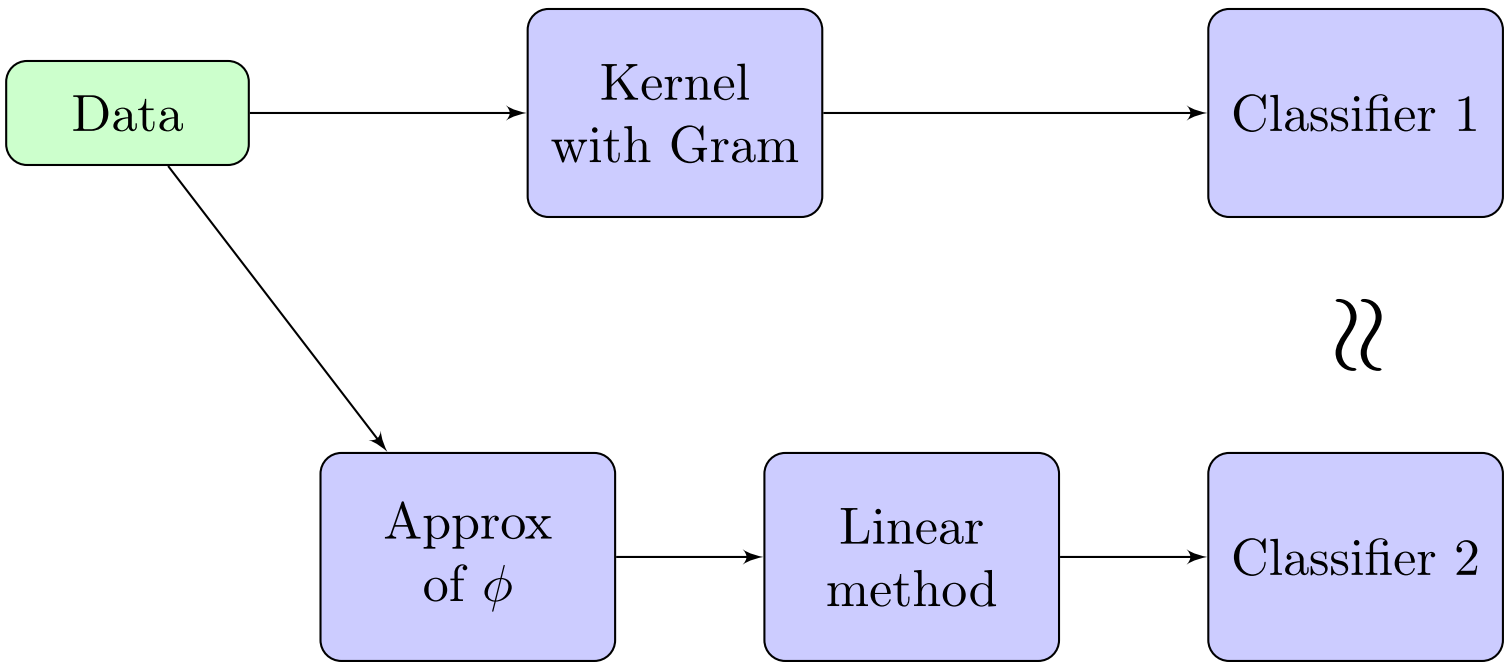
Successful for many kernels in Euclidean spaces[2].

Our aim

Extend this idea to Grassmannian kernels while preserving geometry.

[2] A. Rahimi and B. Recht. "Random Features for Large-Scale Kernel Machines". In: *Advances in Neural Information Processing Systems*. Vol. 20. 2007, pp. 1177–1184.

Approximating κ



Projectors make random features simple

Projectors $\boldsymbol{P}$ live in a linear space (with scalar products distances etc).

Random features can be built from random linear projections with inner products applied to $\boldsymbol{P}$.

Idea: use random objects to probe $\boldsymbol{P}$.

Notation

Expressing elements of $\mathcal{G}(k, n)$ as projectors $\boldsymbol{P}$ will often make the theory work nicer, but we use $\boldsymbol{P}$ and an arbitrary corresponding orthonormal basis $\boldsymbol{U}$ interchangeably.

Random embeddings

Asymmetric rank-one projections (AROP)

Goal: No storage of Gram matrix by transforming the dataset

Probe projectors $\boldsymbol{P} = \boldsymbol{U}\boldsymbol{U}^\top$ with rank-one matrices.

$$\boldsymbol{A}_i = \boldsymbol{a}_i \boldsymbol{b}_i^\top, \quad \boldsymbol{a}_i, \boldsymbol{b}_i \sim \mathcal{N}(0, \boldsymbol{I}_n).$$

Define the AROP sketch of a projector $\boldsymbol{P}$ as

$$\mathcal{A}_i(\boldsymbol{P}) = \langle \boldsymbol{A}_i, \boldsymbol{P} \rangle = \boldsymbol{a}_i^\top \boldsymbol{P} \boldsymbol{b}_i = \boldsymbol{a}_i^\top \boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{b}_i.$$

This is linear in $\boldsymbol{P}$ and invariant to the choice of basis $\boldsymbol{U}$.

Storage scales as $\mathcal{O}(mn)$ with computational complexity as $\mathcal{O}(nkm)$

AROP recovers the projection kernel in expectation

For two projectors $\boldsymbol{P}_1, \boldsymbol{P}_2$, define

$$\widehat{\kappa}^{\mathrm{pro}}(\boldsymbol{P}_1, \boldsymbol{P}_2) = \frac{1}{m} \sum_{i=1}^{m} \left(\boldsymbol{a}_i^\top \boldsymbol{P}_1 \boldsymbol{b}_i\right)\left(\boldsymbol{a}_i^\top \boldsymbol{P}_2 \boldsymbol{b}_i\right).$$

One can show

$$\mathbb{E}\left[\widehat{\kappa}^{\mathrm{pro}}(\boldsymbol{P}_1, \boldsymbol{P}_2)\right] = \langle \boldsymbol{P}_1, \boldsymbol{P}_2 \rangle = \kappa^{\mathrm{pro}}(\boldsymbol{P}_1, \boldsymbol{P}_2).$$

So AROP is a random feature estimator of κ^{pro}.

The heavy-tail problem

Each $Z_i = \left(\boldsymbol{a}_i^\top \boldsymbol{P}_1 \boldsymbol{b}_i\right)\left(\boldsymbol{a}_i^\top \boldsymbol{P}_2 \boldsymbol{b}_i\right)$ involves products of Gaussian random variables.

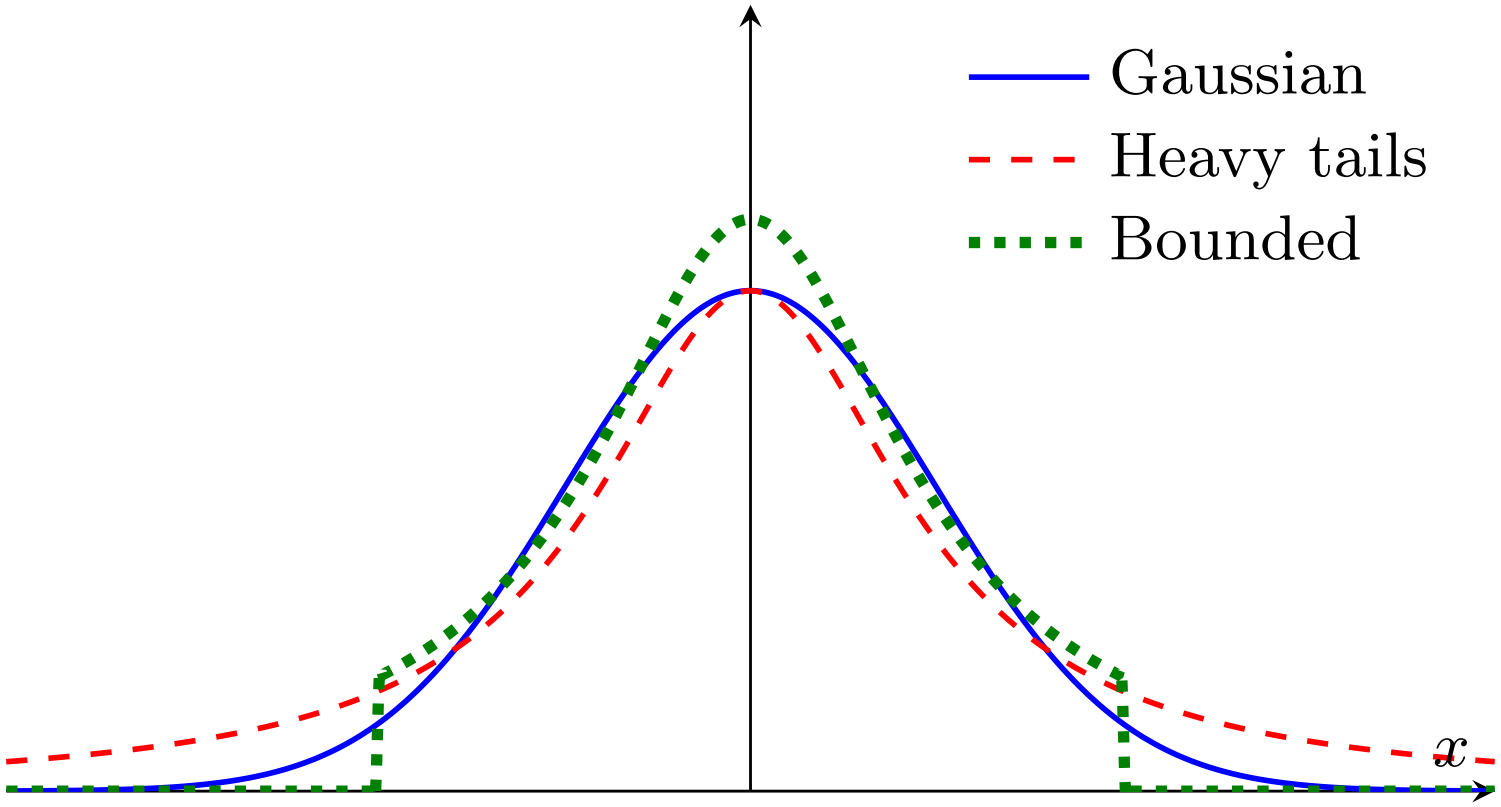
$\implies$ Z_i has heavy tails, $\implies$ $\widehat{\kappa}^{\mathrm{pro}}$ concentrates slowly $\implies$ uniform approximation over $\mathcal{G}(k, n)$ requires **very large** m.

We need to *tame the tails* with bounded functions

- Sign function $\operatorname{sign}(\mathcal{A}(\boldsymbol{P}))$.
- Complex exponential $\exp(i\omega\mathcal{A}(\boldsymbol{P}))$.

Bounded random variables $\implies$ nice concentration theorems.

Tails taming



Bounded functions

Binary AROP features

Define the binary sketch of a projector $\boldsymbol{P}$:

$$\mathcal{A}_i^{\pm}(\boldsymbol{P}) = \operatorname{sign}\left(\boldsymbol{a}_i^{\top} \boldsymbol{P} \boldsymbol{b}_i\right).$$

Kernel:

$$\kappa^{\pm}(\boldsymbol{P}, \boldsymbol{Q}) = \mathbb{E}\left[\mathcal{A}_i^{\pm}(\boldsymbol{P}) \mathcal{A}_i^{\pm}(\boldsymbol{Q})\right].$$

Values lie in $\{-1, +1\}$, making the sketch very light.

Random feature approximation:

$$\widehat{\kappa}^{\pm} = \tfrac{1}{m} \langle \mathcal{A}^{\pm}(\boldsymbol{P}), \mathcal{A}^{\pm}(\boldsymbol{Q}) \rangle.$$

Properties of $\kappa^{\pm}$

$\kappa^{\pm}$ has the properties of being a Grassmannian kernel.

Unfortunately, no closed-form for $k > 1$, but boundedness $\implies$ if

$$m \gtrsim \delta^{-2}\left(\mathcal{O}(nk) + \log\left(\tfrac{1}{\eta}\right)\right),$$

then

$$\sup_{\boldsymbol{P},\boldsymbol{Q}} |\widehat{\kappa}^{\pm} - \kappa^{\pm}| \leq \delta$$

with probability at least $1 - \eta$.

Exponential AROP features

Define the complex sketch:

$$\Psi_j(\boldsymbol{P}) = \exp\left(i\omega\mathcal{A}_j(\boldsymbol{P})\right), \omega \in \mathbb{R}.$$

Grassmannian Kernel with random approximation

$$\kappa^\circ(\boldsymbol{P}, \boldsymbol{Q}) = \mathbb{E}\left[\Psi_i(\boldsymbol{P})\overline{\Psi_i(\boldsymbol{Q})}\right]\widehat{\kappa}^\circ = \frac{1}{m}\langle\Psi(\boldsymbol{P}), \Psi(\boldsymbol{Q})\rangle.$$

Bounded in modulus: $|\Psi_j(\boldsymbol{P})| \leq 1$.

Closed form, for $\omega \in \mathbb{R}$

$$\kappa^\circ(\boldsymbol{P}, \boldsymbol{Q}) = \prod_{j=1}^{k}\left(1 + \omega^2 \sin^2\theta_j\right)^{-1}.$$

Concentration

Similar to $\kappa^{\pm}$, if

$$m \gtrsim \delta^{-2}\left(\mathcal{O}(nk) + \log\left(\tfrac{1}{\eta}\right)\right), \text{ then}$$

$$\sup_{\boldsymbol{P},\boldsymbol{Q}} |\widehat{\kappa}^{\circ} - \kappa^{\circ}| \leq \delta \text{ with probability at least } 1 - \eta.$$

Summary

Two new kernels with bounded functions:

- Good properties
- Good concentration

Efficient sketching

Goal

Goal: replace unstructured Gaussian vectors by structured transforms.

Aim to keep similar statistical behaviour while making computation cheaper.

Structured AROP: FFT-based random probes

Replace Gaussian vectors $(\boldsymbol{a}_i, \boldsymbol{b}_i)$ by *structured* random vectors.

Define $\boldsymbol{T} = \boldsymbol{F}\boldsymbol{D}_K\boldsymbol{F}\boldsymbol{D}_{K-1}\cdots\boldsymbol{F}\boldsymbol{D}_1, \qquad \boldsymbol{T}' = \boldsymbol{F}\boldsymbol{D}'_K\boldsymbol{F}\cdots\boldsymbol{F}\boldsymbol{D}'_1,$
where

- $\boldsymbol{F}$ is a fast transform (FFT-type),
- $\boldsymbol{D}_j, \boldsymbol{D}'_j$ are random diagonal ± 1 matrices,
- only the diagonal signs are stored ($\mathcal{O}(n)$ memory).

Applying the FFT-sign-FFT series to all k columns of $\boldsymbol{U}$ (without forming $\boldsymbol{T}$ explicitly) costs $\mathcal{O}(nk \log n)$.

The i-th AROP measurement is then

$$\mathcal{A}_i(\boldsymbol{P}) = (\boldsymbol{T}\boldsymbol{U})_{i,:}\,(\boldsymbol{T}'\boldsymbol{U})_{i,:}^{\top},$$

Practical benefits

Memory

- Unstructured AROP: store $\mathcal{O}(mn)$ floats.
- Structured AROP: store only $\mathcal{O}(n)$ random signs (the diagonal matrices).

Computation

- Unstructured AROP: $\mathcal{O}(mnk)$ for all $\boldsymbol{a}_i^\top \boldsymbol{P} \boldsymbol{b}_j$.
- Structured AROP: $\underbrace{\mathcal{O}(nk \log n)}_{\text{apply transforms to } \boldsymbol{U}} + \underbrace{\mathcal{O}(mk)}_{\text{final dot products}}$

Total cost

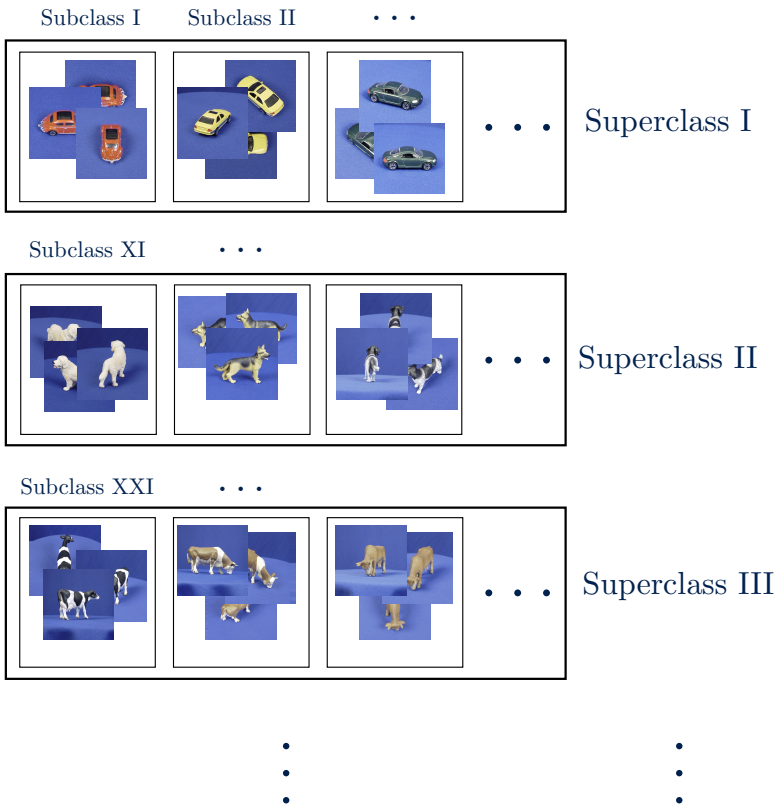
$$\mathcal{O}(nk \log n + mk)$$

Complexity summary (per subspace)

Method	Compute cost	Sketch storage
Dense projections	$\mathcal{O}(mn^2k)$	$\mathcal{O}(mn^2)$
Unstructured AROP	$\mathcal{O}(mnk)$	$\mathcal{O}(mn)$
Structured AROP	$\mathcal{O}(nk\log n + mk)$	$\mathcal{O}(n)$
OPU	$\mathcal{O}(k)$	$\mathcal{O}(1)$

Experiments

ETH-80 dataset and setup



- Dataset: 80 objects in 8 superclasses (car, dog, ...).
- 41 images per object, resized to 32×32 greyscale ($n = 1024$).
- Classify in superclasses
- Classify in subclasses

Superclass classification: experimental protocol

Goal. Classify each object into one of the 8 superclasses.

Setup

- For all 80 objects, stack 41 images $\boldsymbol{X} \in \mathbb{R}^{1024\times 41}$.
- Orthonormal basis $\boldsymbol{U} \in \mathbb{R}^{1024\times 9}$ from the SVD of $\boldsymbol{X}$.
- $\boldsymbol{U} \in \mathcal{G}(9, 1024)$.
- For each superclass: choose 7 objects for training, 3 for testing
  $\Longrightarrow$ 56 train subspaces, 24 test subspaces.

Kernels evaluated.

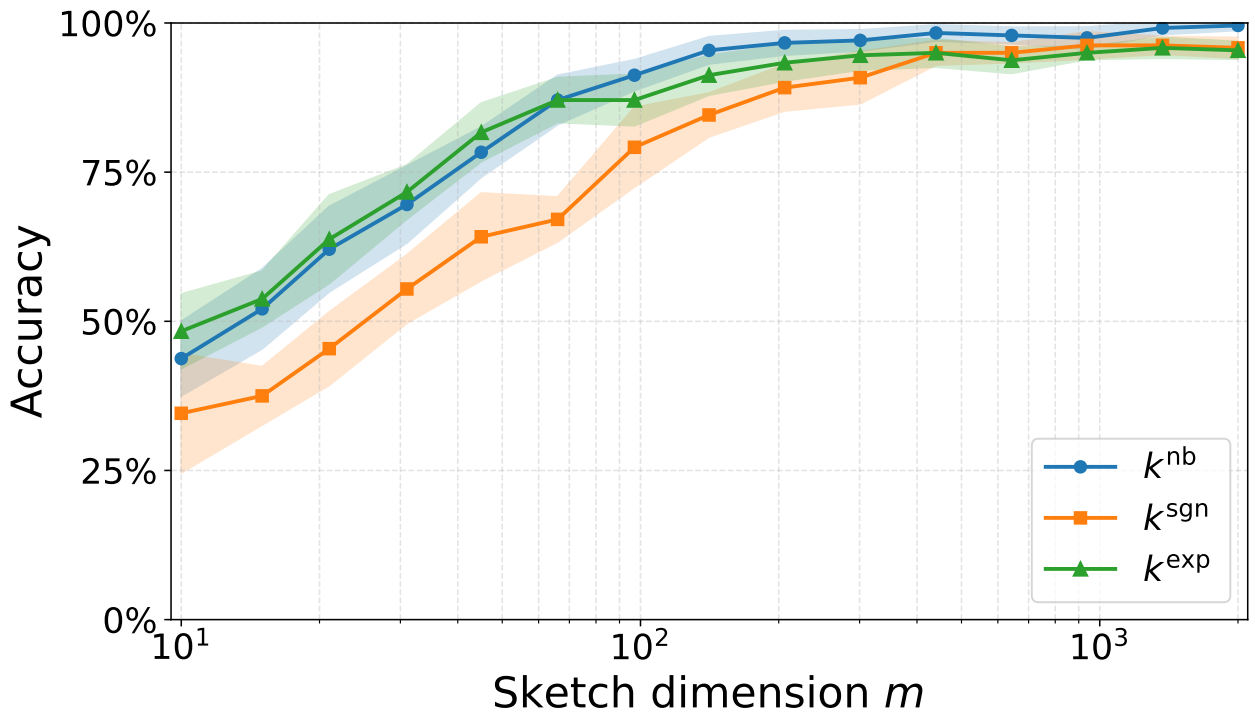
- Exact: κ^{pro}, κ° as ground truth (GT).
- Random features: $\widehat{\kappa}^{\mathrm{pro}}$, $\widehat{\kappa}^{\pm}$, $\widehat{\kappa}^{\circ}$.
- Structured version (S).

Oversampling ratio: $\rho = m/(nk)$, with m the embedding dimension.

Superclass classification (8-way): unstructured sketches

Accuracy (%) / total time (s)

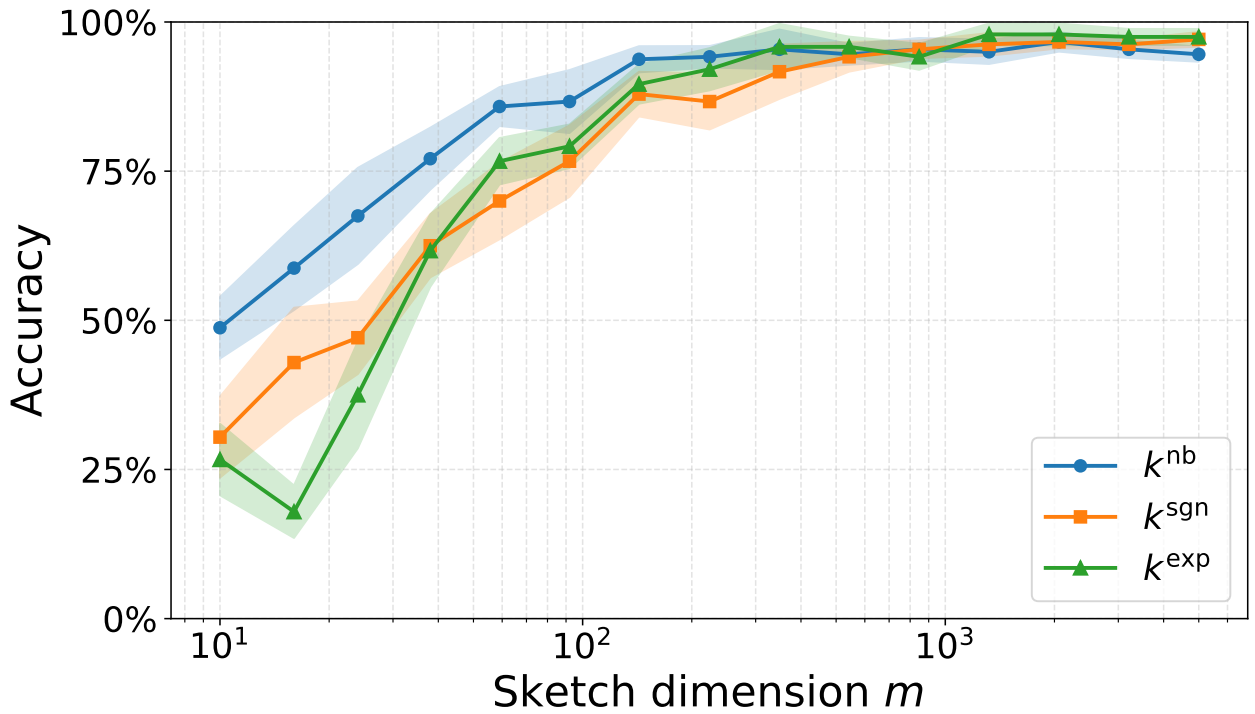
Method	$\widehat{\kappa}^{\mathrm{pro}}$	$\widehat{\kappa}^{\pm}$	$\widehat{\kappa}^{\circ}$
Exact (GT)	100.00 / 0.26	–	100.00 / 0.24
$\rho = 0.05$	98.12 / 1.32	93.12 / 1.32	93.54 / 1.32
$\rho = 0.20$	99.79 / 5.31	96.04 / 5.31	93.75 / 5.31



Superclass classification (8-way): structured sketches

Accuracy (%) / total time (s)

Method	$\widehat{\kappa}^{\mathrm{pro}}$	$\widehat{\kappa}^{\pm}$	$\widehat{\kappa}^{\circ}$
Exact (GT)	100.00 / 0.26	–	100.00 / 0.24
$\rho = 0.05$	94.79 / 0.10	92.29 / 0.10	94.79 / 0.11
$\rho = 0.20$	93.54 / 0.20	96.46 / 0.20	93.54 / 0.20



Subclass classification: experimental protocol

Goal: Classify each object among 80 subclasses.

Setup

- Split 41 images in 28 train and 13 test.
- Create several training subspaces per object by randomly selecting 15 images:

$$\boldsymbol{X} \in \mathbb{R}^{1024\times 15}, \quad \boldsymbol{U} = \text{top-9 singular vectors.}$$

- Repeat 10 times $\rightarrow$ 10 training subspaces per object.
- Test subspace: build from the remaining 13 images $\implies$ 800 train subspaces, 80 test subspaces.

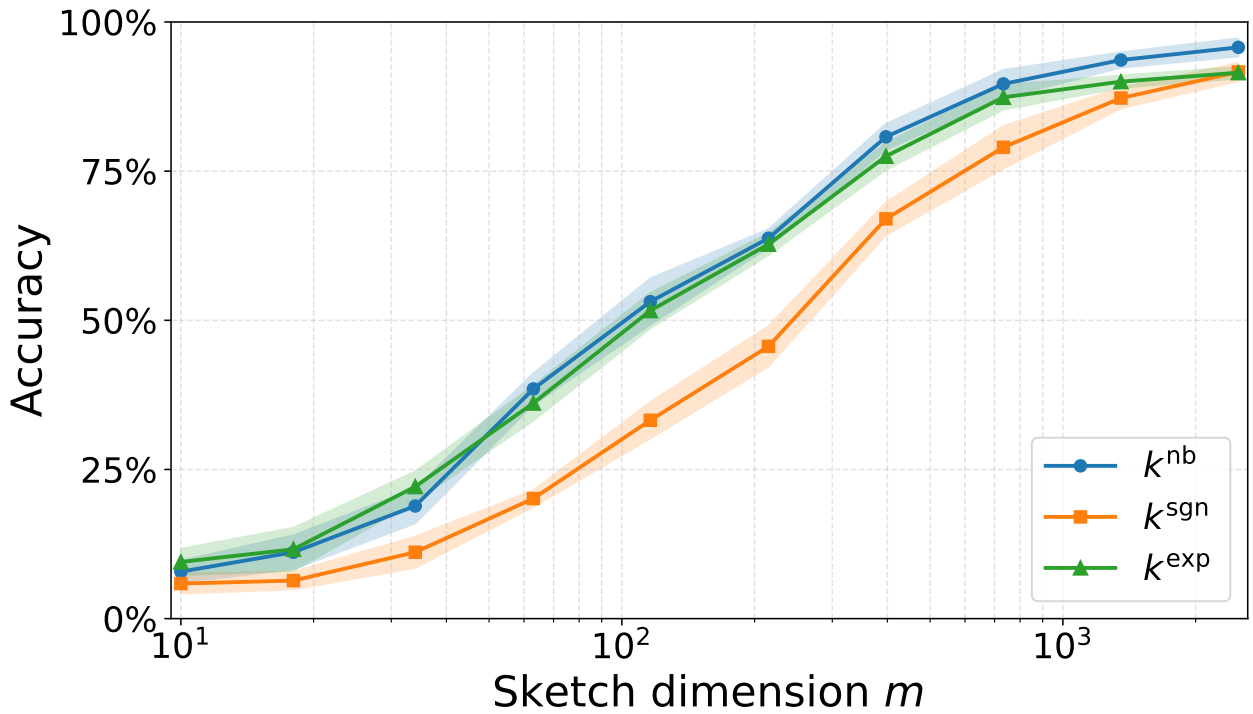
Approximated kernels

κ^{pro} and κ° as ground truth (GT), unstructured: $\widehat{\kappa}^{\mathrm{pro}}$, $\widehat{\kappa}^{\pm}$, $\widehat{\kappa}^{\circ}$ and structured version

Subclass classification (80-way): unstructured sketches

Accuracy (%) / total time (s)

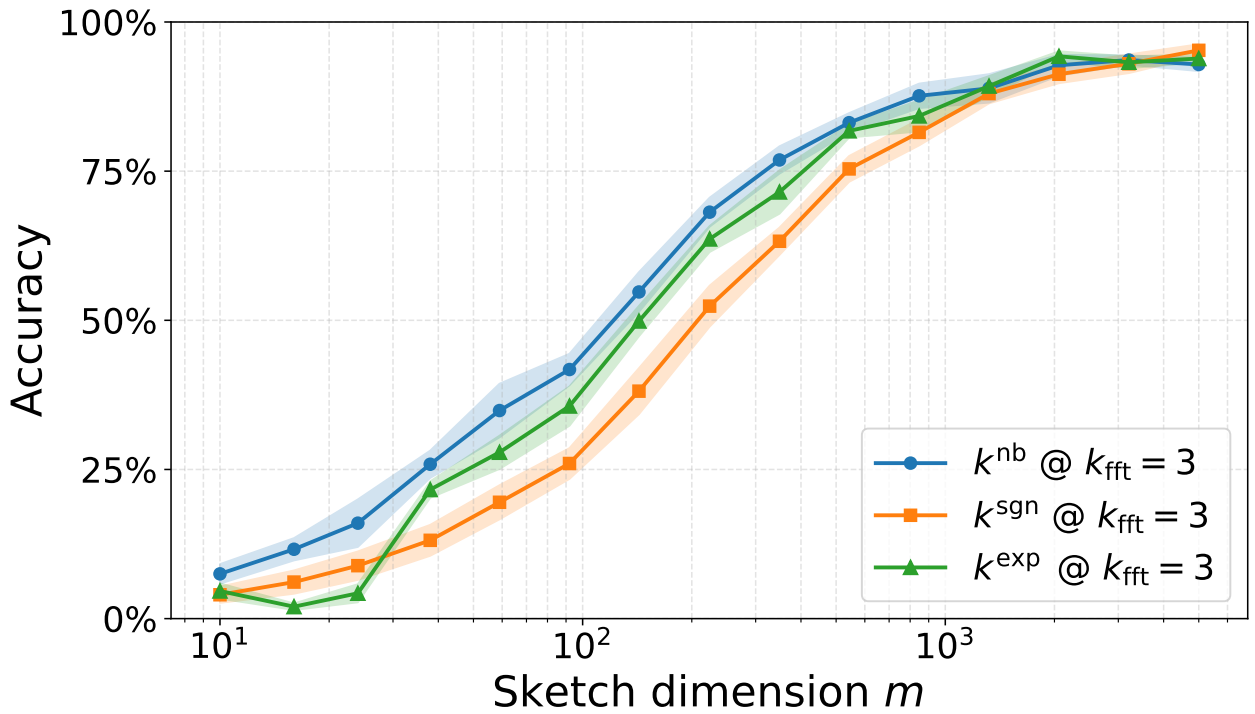
Method	$\widehat{\kappa}^{\mathrm{pro}}$	$\widehat{\kappa}^{\pm}$	$\widehat{\kappa}^{\circ}$
Exact (GT)	98.75 / 52.89	–	90.00 / 58.79
$\rho = 0.05$	84.19 / 24.19	72.25 / 24.20	81.38 / 24.38
$\rho = 0.20$	94.06 / 85.21	91.75 / 85.24	92.31 / 86.41



Subclass classification (80-way): structured sketches

Accuracy (%) / total time (s)

Method	$\widehat{\kappa}^{\mathrm{pro}}$	$\widehat{\kappa}^{\pm}$	$\widehat{\kappa}^{\circ}$
Exact (GT)	98.75 / 52.89	–	90.00 / 58.79
$\rho = 0.05$	83.00 / 1.87	72.13 / 1.89	83.00 / 2.00
$\rho = 0.20$	92.38 / 4.03	91.25 / 4.08	92.38 / 5.24



Experimental takeaways

Superclass classification (8-way)

- Random feature approximations reach high accuracy; this small setting mainly validates approximation behaviour.
- Exact kernels take 0.24–0.26 s; structured sketches 0.10–0.20 s, versus 1.32–5.31 s unstructured.

Subclass classification (80-way)

- Structured sketches achieve strong accuracy in 1.87–5.24 s, versus 52.89–58.79 s for exact kernels.
- Unstructured sketches take 24–86 s: increasing m can erase their computational advantage.

Different features target different kernels: accuracy measures both approximation quality and kernel suitability. Total time includes sketch computation and SVM training/evaluation; averages over 20 runs.

Conclusion

Conclusion

We built **random feature approximations** for several Grassmannian kernels, both classical and new ones.

We provided **theoretical guarantees**: concentration, uniform control, and explicit dependence on m.

We extended the approach to **structured sketches** to reduce memory and computation.

ETH-80 experiments **illustrate the approximation results**.

Random features + boundedness (+ structure) = scalable Grassmann kernels.

Thank you for your attention!

References



Mehrtash T Harandi et al. "Expanding the family of grassmannian kernels: An embedding perspective". In: *European conference on computer vision*. Springer. 2014, pp. 408–423.



A. Rahimi and B. Recht. "Random Features for Large-Scale Kernel Machines". In: *Advances in Neural Information Processing Systems*. Vol. 20. 2007, pp. 1177–1184.