Accelerated Kernel Stein Discrepancy with Rényi Landmark Selection for GAN Training

Michael Carlo **Giovanni Dettori** **Ryan Gumsheimer**

Abstract

Our project investigates replacing the classical adversarial discriminator in GAN training with a kernel-based distance metric, namely Kernel Stein Discrepancy (KSD). We assess whether a kernelized objective can improve training stability and efficiency without compromising sample quality, and we evaluate accelerated Nyström approximations with Rényi landmark selection on CIFAR-10.

1 Project Plan

Our project investigates the replacement of the classical adversarial discriminator in GAN [1] training with a kernel-based distance metric, namely Kernel Stein Discrepancy (KSD). The goal is to assess whether a kernelized objective can provide an alternative that improves training stability and efficiency without compromising sample quality.

We will begin by reproducing the standard GAN setup using benchmark image datasets such as CIFAR-10 to establish a reliable baseline and validate our implementation. Building on this, we will integrate KSD loss with Rényi landmark selection into the training pipeline, using it as a direct substitute for the adversarial loss. Finally, we will compare the two training paradigms (adversarial versus kernel-based) using both quantitative metrics (e.g., Fréchet Inception Distance, FID [2]) and qualitative sample evaluation. All experiments will be implemented in Python, version-controlled via GitHub, and executed with GPU acceleration where appropriate.

2 Background and Motivation

Goodness-of-fit testing and model evaluation are central problems in modern machine learning and statistics. Given samples from an unknown distribution Q and a target model P with density $p(x)$, we often wish to determine how well Q approximates P. Classical divergence measures such as the Kullback–Leibler or Jensen–Shannon divergences are theoretically appealing but typically require access to the density $q(x)$ of the model distribution, which is unavailable in implicit generative models like GANs.

Kernel methods provide a powerful alternative by embedding probability measures into a *reproducing kernel Hilbert space* (RKHS), thereby allowing discrepancies between distributions to be expressed as distances between their embeddings. Formally, every positive-definite kernel $k(x, x')$ defines a unique Hilbert space of functions $\mathcal{H}_k$, where evaluation is continuous and satisfies

$$f(x) = \langle f, k(\cdot, x)\rangle_{\mathcal{H}_k}, \qquad \langle k(\cdot, x), k(\cdot, x')\rangle_{\mathcal{H}_k} = k(x, x').$$

This property allows nonlinear relationships in the data space to appear as linear relations in the induced Hilbert space, so that expectations and variances can be expressed as inner products in $\mathcal{H}_k$.

In this framework, each probability distribution P on $\mathcal{X}$ can be represented by its *mean embedding* in the RKHS,

$$\mu_P := \mathbb{E}_{X\sim P}[\, k(\cdot, X)\,] \in \mathcal{H}_k,$$

Preprint. Under review.

which satisfies, for any $f \in \mathcal{H}_k$,

$$\mathbb{E}_{X\sim P}[f(X)] = \langle f, \mu_P \rangle_{\mathcal{H}_k}.$$

Expectations under P thus become inner products between the test function f and the embedding μ_P. Similarly, comparing two distributions reduces to comparing their embeddings: the closer μ_P and μ_Q are in $\mathcal{H}_k$, the more similar the corresponding distributions are in the input space.

This observation provides a unifying language for defining statistical *discrepancies* between probability measures. A broad family of divergences can be expressed as *integral probability metrics* (IPMs) of the form

$$D_{\mathcal{F}}(P, Q) = \sup_{f\in\mathcal{F}} \Big|\mathbb{E}_{X\sim P}[f(X)] - \mathbb{E}_{Y\sim Q}[f(Y)]\Big|,$$

where $\mathcal{F}$ is a class of discriminating test functions. Choosing $\mathcal{F}$ as the unit ball of the RKHS $\mathcal{H}_k$ yields the *Maximum Mean Discrepancy* (MMD),

$$\mathrm{MMD}(P, Q) = \|\mu_P - \mu_Q\|_{\mathcal{H}_k},$$

which admits a closed-form expression in terms of kernel evaluations and can be efficiently estimated from samples. When the kernel k is *characteristic* meaning that the mean embedding map $P \mapsto \mu_P$ is injective, the MMD uniquely identifies probability distributions, i.e. $\mathrm{MMD}(P, Q) = 0$ if and only if $P = Q$.

However, MMD and related kernel-based divergences still require samples from both P and Q, or explicit knowledge of their normalized densities. This poses a serious limitation when the target distribution P is known only up to a normalizing constant and evaluating $p(x)$ or sampling from P may be computationally intractable. In such cases, standard IPM or embedding-based methods cannot be directly applied.

An elegant solution to this issue arises by exploiting *Stein's identity*, which allows one to construct kernelized discrepancies that depend only on the *score function* of the target distribution, $\nabla_x \log p(x)$. This observation leads directly to the development of the *Kernel Stein Discrepancy* (KSD), a measure of goodness-of-fit that preserves the nonparametric flexibility of kernel methods while not needing a fully normalized target density.

A particularly elegant instance of this framework is provided by *Stein's method*. Let P be a target distribution on $\mathbb{R}^d$ with smooth log-density $\log p(x)$ and score function $\nabla_x \log p(x)$. Stein's identity states that for any sufficiently regular test function $f : \mathbb{R}^d \to \mathbb{R}^d$,

$$\mathbb{E}_{X\sim P}\big[\, \nabla_x \log p(X)^\top f(X) + \nabla_x \cdot f(X) \,\big] = 0.$$

The operator inside the expectation,

$$(T_p f)(x) = \nabla_x \log p(x)^\top f(x) + \nabla_x \cdot f(x),$$

is known as the *Stein operator*. When comparing two distributions, say Q and P, we can evaluate the above expectations under Q: any deviation from zero quantifies a discrepancy between Q and P.

Restricting f to lie in a vector-valued RKHS $\mathcal{H}_k^d$ induced by a kernel k, one therefore obtains the *Kernel Stein Discrepancy* (KSD),

$$\mathrm{KSD}^2(Q, P) = \Big\| \mathbb{E}_{X\sim Q}[T_p k(\cdot, X)] \Big\|^2_{\mathcal{H}_k}.$$

The KSD measures the squared RKHS norm of the Stein witness function, which equals zero if and only if $Q = P$ when k is characteristic and p satisfies mild smoothness and boundary conditions[1]. Unlike divergences that require evaluating $q(x)$, KSD depends only on samples from Q and on the score function of P, making it well suited for implicit models and high-dimensional data. When P is empirical, say P_{data}, its score function $\nabla_x \log p_{\mathrm{data}}(x)$ can be approximated using a pretrained score network, or a kernel density estimate.

[1]For Stein's identity to hold, the target density p must be continuously differentiable and satisfy $p(x)f(x) \to 0$ as $\|x\| \to \infty$ for all admissible test functions f. These assumptions ensure that integration by parts is valid and that the Stein operator yields zero expectation under the true distribution P.

In practice, evaluating the population KSD requires replacing expectations by empirical averages. Given samples $\{x_i\}_{i=1}^n \sim Q$, empirical estimators are obtained by replacing expectations with sample averages. The two standard estimators are the *V-statistic* and the *U-statistic*, respectively defined as

$$\widehat{\mathrm{KSD}}_V^2(Q,P) = \frac{1}{n^2}\sum_{i,j=1}^{n} h_p(x_i,x_j), \qquad \widehat{\mathrm{KSD}}_U^2(Q,P) = \frac{1}{n(n-1)}\sum_{i\neq j} h_p(x_i,x_j),$$

where the *Stein kernel* $h_p(x,x')$ is given by

$$h_p(x,x') = \nabla_x \log p(x)^\top \nabla_{x'} \log p(x')\, k(x,x') + \nabla_x \log p(x)^\top \nabla_{x'} k(x,x') + \nabla_{x'} \log p(x')^\top \nabla_x k(x,x') + \mathrm{tr}\big(\nabla^2_{x,x'} k(x,x')\big)\,.$$

The V-statistic is a biased estimator of the population KSD, while the U-statistic is unbiased. Both estimators have quadratic computational complexity $\mathcal{O}(n^2)$ in the number of samples, which makes them prohibitively expensive for large-scale applications.

To address this limitation, [3] introduced the *Nyström-KSD estimator*, an accelerated approximation that projects the empirical mean embedding $\frac{1}{n}\sum_{i=1}^n h_p(\cdot,x_i)$ onto the subspace spanned by a small set of $m \ll n$ Nyström points $\{\tilde{x}_j\}_{j=1}^m$. The resulting estimator takes the form

$$\widehat{\mathrm{KSD}}_N^2(Q,P) = \beta^\top K_{m,m}^- \beta, \quad \beta = \frac{1}{n} K_{m,n} \mathbf{1}_n,$$

where $K_{m,m}$ and $K_{m,n}$ are Gram matrices built from the Stein kernel h_p, and $K_{m,m}^-$ denotes the Moore–Penrose pseudoinverse. This approach reduces the computational cost to $\mathcal{O}(mn + m^3)$, while preserving the $\mathcal{O}(n^{-1/2})$ statistical rate of convergence, achieving the same asymptotic efficiency as the U- and V-estimators (so called *quadratic*).

Recently it has been shown that this rate cannot be improved. [3] established the $\sqrt{n}$ consistency of both the quadratic and Nyström KSD estimators under a sub-Gaussian assumption on the Stein feature map, and [4] proved that the *minimax lower bound* for KSD estimation is $\mathcal{O}(n^{-1/2})$. Consequently, all known KSD estimators are *minimax optimal*, achieving the fastest possible convergence rate for KSD estimation under standard smoothness assumptions on p and the kernel k.

3 Proposed Research

While the Nyström-KSD estimator achieves the optimal $\mathcal{O}(n^{-1/2})$ convergence rate under *uniform* landmark selection, uniform sampling may not be the most informative strategy in practice. In high-dimensional or heterogeneous datasets, uniformly chosen landmarks often oversample dense regions and underrepresent informative or low-probability areas. To address this, we propose replacing the uniform selection of Nyström landmarks with a ***Rényi sampling scheme***, designed to better capture information in the data distribution.

The proposed sampling criterion is inspired by the quadratic Rényi entropy defined in [5] for a probability density $p(x)$ as

$$H_R = -\log \int p(x)^2\, dx,$$

which measures the spread, or uncertainty, of the distribution. Given an empirical kernel density estimate $\hat{p}(x)$ constructed from data samples $\{x_i\}_{i=1}^N$, this entropy can be approximated as

$$\int \hat{p}(x)^2\, dx = \frac{1}{N^2} \mathbf{1}_N^\top \Omega \mathbf{1}_N,$$

where $\mathbf{1}_N = [1,\ldots,1]^\top$ and Ω is the kernel (Gram) matrix with entries $\Omega_{ij} = k(x_i,x_j)$. Minimizing this quantity corresponds to maximizing the entropy of the selected subset, hence encouraging a set of landmarks that are diverse and informative in the kernel-induced feature space.

In this setting, we aim to optimize the selection of Nyström landmarks $\tilde{X} = \{\tilde{x}_1,\ldots,\tilde{x}_m\}$ by maximizing the entropy of their induced kernel density, or equivalently, by minimizing

$$\mathcal{E}(\tilde{X}) = \frac{1}{m^2} \mathbf{1}_m^\top \Omega(\tilde{X}) \mathbf{1}_m,$$

subject to the constraint that Ω is normalized with respect to the kernel bandwidth. This criterion ensures that the induced subspace $\mathrm{span}\{h_p(\cdot, \tilde{x}_i)\}_{i=1}^m$ better captures the geometry of the Stein feature map and the expressive regions of the data distribution. By promoting landmarks that are both representative and diverse, this strategy is expected to reduce the finite-sample bias of the Nyström-KSD estimator and stabilize its gradients when used as a loss in generative models.

Generative Adversarial Networks (GANs) provide a widely used framework for learning generative models that can synthesize data samples resembling those drawn from an unknown distribution P_{data}. A GAN consists of two components: a generator $G_\theta(z)$, which maps latent variables $z \sim p_z$ (typically standard Gaussian noise) to the data space, and a discriminator $D_\phi(x)$, which aims to distinguish between real samples $x \sim P_{\mathrm{data}}$ and generated samples $x = G_\theta(z)$. The two networks are trained in a minimax game:

$$\min_\theta \max_\phi \; \mathcal{L}_{\mathrm{GAN}}(\theta, \phi) = \mathbb{E}_{x \sim P_{\mathrm{data}}}\big[\log D_\phi(x)\big] + \mathbb{E}_{z \sim p_z}\big[\log(1 - D_\phi(G_\theta(z)))\big].$$

At equilibrium, the generator distribution Q_θ ideally converges to the data distribution P_{data}, achieving a Nash equilibrium in which $D_\phi(x) = \frac{p_{\mathrm{data}}(x)}{p_{\mathrm{data}}(x) + q_\theta(x)}$.

However, the adversarial objective is known to be unstable and provides only an implicit measure of discrepancy between P_{data} and Q_θ. Moreover, the Jensen–Shannon divergence underlying $\mathcal{L}_{\mathrm{GAN}}$ requires that both distributions admit well-defined densities, which is often restrictive in high-dimensional or implicit generative settings.

The *Kernel Stein Discrepancy* offers a non-adversarial alternative that directly quantifies the discrepancy between the model distribution Q_θ and the target P_{data} through their score functions, without needing to train an explicit discriminator. By using the accelerated Nyström-KSD estimator, this discrepancy can be efficiently approximated from mini-batches of generated samples, leading to a tractable training loss

$$\mathcal{L}_{\mathrm{KSD}}(\theta) = \widehat{\mathrm{KSD}}_N^2(Q_\theta, P_{\mathrm{data}}),$$

whose gradients with respect to θ can be computed by automatic differentiation through the Nyström projection. This yields a generator trained to minimize the kernel-based discrepancy to the data distribution, sidestepping adversarial optimization while retaining theoretical convergence guarantees. Furthermore, the proposed ***Rényi Nyström-KSD*** estimator enhances finite-sample efficiency by adaptively selecting informative landmarks, thus combining the statistical rigor of Stein's method with the practical scalability of modern deep generative models.

4 Data and Code Implementation

4.1 Sanity checks on synthetic Gaussian data

Following the same strategy adopted in the Nyström-KSD framework of [3], we run a small set of sanity checks with synthetic data before moving to CIFAR-10 and using the Rényi–Nyström KSD as a training loss for the GAN[2]. These checks are particularly important in our setting because the discrepancy is not only a diagnostic statistic but also the learning signal for the generator. If the KSD computation is mis-scaled, numerically unstable, or inconsistent with the quadratic-time definition, then KSD–GAN training can be unreliable and any behavior may be affected.

All tests use $P = \mathcal{N}(0, I)$ and a Gaussian (RBF) kernel, and verify three basic properties:

- **Stein Gram consistency.** We verify that the Stein kernel Gram matrix produced by our implementation matches the standard one. Concretely, we compute the full Stein matrix H (with entries $H_{ij} = h_p(x_i, x_j)$) and compare it to the matrix returned by the Rényi–Nyström implementation. This check confirms that the Stein kernel construction used downstream by the Nyström estimator is consistent with the standard KSD formulation.

[2]It is important to highlight that these checks are taken from the following repository `https://github.com/FlopsKa/nystroem-ksd`. They are not the result of our original contribution and they have to be intended as part of a preliminary phase.

- **Nyström statistic agrees in scale with the quadratic KSD.** We verify that the Rényi–Nyström estimator produces values consistent with the standard empirical KSD computed from all pairwise Stein kernel evaluations, up to the approximation error induced by using a reduced set of landmarks. Because the quadratic KSD is defined as the average of the Stein kernel Gram matrix, while the Nyström estimator directly approximates this quantity, we compare the appropriately normalized quadratic reference to the Nyström estimate. This check ensures that the accelerated estimator has the correct scale and does not introduce systematic bias due to approximation or normalization.
- **GOF behavior under H_0 and a simple H_1.** Finally, we validate that the hypothesis test based on the Rényi–Nyström KSD behaves sensibly in a controlled scenario: under H_0 (samples from $\mathcal{N}(0, I)$) it yields a non-small p-value, while under a mean-shift alternative H_1 (samples from $\mathcal{N}(3\mathbf{1}, I)$) it yields a small p-value and rejects at the chosen significance level.

These checks are not intended as a full statistical validation, but they provide enough evidence so that it is reasonable to proceed experimenting where the same discrepancy is used as a differentiable objective for GAN training.

4.2 Implementation summary

We evaluate our proposal on CIFAR-10 [6], a standard benchmark of 60,000 32×32 RGB images in 10 classes (50,000 training and 10,000 test images). In line with common practice in generative modeling, we use the training split both to learn the generator and as the reference empirical distribution for discrepancy-based evaluation metrics. The objective is to compare (i) a standard GAN trained with an adversarial discriminator (baseline) to (ii) a KSD-trained generator in which the discriminator is replaced by an accelerated, Nyström-approximated Kernel Stein Discrepancy with Rényi-based landmark selection. Our implementation is designed to (a) be computationally feasible on mini-batches, (b) make the KSD objective differentiable with respect to the generator parameters, and (c) isolate and control sources of instability typical of kernel-based discrepancies, including bandwidth selection, ill-conditioned Gram matrices, and noisy score-function estimates.

The code implements three training pipelines on the same CIFAR-10 batches:

- **KSD-GAN (random-projection):** a generator G_θ is trained by minimizing a *two-sample* Nyström-KSD objective that compares generated samples to real samples, using a Stein kernel in pixel space whose base RBF kernel is defined on a fixed random-projection feature map. The Stein score of the target distribution is approximated via a *pretrained diffusion model* (DDPM). Landmark selection for Nyström acceleration is performed with a Rényi-inspired diversity criterion on real samples only and is explicitly detached from gradients. We refer to this variant as **KSD-GAN (Linear)** because the base kernel is applied to features obtained via a fixed *linear* random projection $z = Px$ from pixel space. This will be clarified in the next section
- **KSD-GAN (ResNet features):** the same training objective is used, but the base kernel is defined on frozen deep features extracted from an ImageNet-pretrained ResNet-18. Feature normalization, bandwidth selection, and landmark selection are again performed using real samples only. Gradient-dependent Stein terms are included only for fake samples to preserve differentiability with respect to the generator.
- **Standard GAN (baseline):** the same generator architecture is trained with a DCGAN-style [7] discriminator using binary cross-entropy, matching the classical adversarial setup.

We then evaluate all generators using (i) FID and KID computed from Inception-v3 features (described in the next section) and (ii) qualitative inspection of generated samples.

4.3 KSD-GAN with Random-projection

We start off by loading a pretrained DDPM for CIFAR-10 via `diffusers` (`google/ddpm-cifar10-32`) and freezing its U-Net parameters. The purpose of this model is not generation, but *score estimation*: we need an approximation of $\nabla_x \log p_t(x)$ to form the Stein kernel.

We implement the forward diffusion marginal

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I),$$

and return both x_t and $\sigma_t = \sqrt{1-\bar{\alpha}_t}$. This is done to push *both* real images and generated images through the same process, ensuring the KSD comparison occurs at the same noise level t.

We then use a U-Net noise predictor $\hat{\varepsilon}_\phi(x_t, t)$ to approximate the score at time t as

$$\nabla_x \log p_t(x_t) \;\approx\; -\frac{\hat{\varepsilon}_\phi(x_t, t)}{\sigma_t}.$$

In practice, we adopt an asymmetric treatment of score evaluations on real and generated samples. The diffusion-based score estimator is kept fixed throughout training and is used only as a reference for the target distribution. Accordingly, score evaluations at real data points do not contribute to gradient propagation, while score evaluations at generated samples are included in the computational graph so that discrepancies measured by the KSD induce gradients with respect to the generator parameters. This choice reflects the role of the diffusion model as a proxy for the unknown data score and ensures that all optimization updates are applied only to the generator.

CIFAR-10 images are converted to tensors and rescaled from $[0, 1]$ to $[-1, 1]$. This matches the generator output which uses `tanh` at the end, and keeps image ranges consistent between real and generated samples.

Directly using a pixel-space RBF kernel in $D = 3 \cdot 32 \cdot 32$ dimensions can be unstable and computationally heavy, while gradients may become poorly scaled. The code therefore defines a fixed, random linear feature map

$$z = Px, \quad P \in \mathbb{R}^{d_{\text{proj}} \times D},$$

implemented by random projection. The projection matrix P is registered as a buffer (not trainable) to ensure the kernel feature space remains fixed across training.

We then compute mean and standard deviation of projected features using only real samples. The reason is that if fake samples influenced normalization, the feature scaling would change as the generator changes, confounding the training signal and potentially creating feedback loops. Using real-only statistics produces a stable reference space.

We proceed by applying the normalization $(z - \mu)/\sigma$ to any batch. This standardization improves numerical conditioning, stabilizes bandwidth selection, and reduces sensitivity to changes in the generator distribution.

We introduce a median heuristic for the RBF bandwidth in the normalized feature space, again computed on real samples only:

$$\sigma^2 \approx \tfrac{1}{2}\,\text{median}\left(\|z_i - z_j\|^2\right).$$

This makes the kernel scale adaptive to the data at each diffusion time t, while avoiding leakage from generator samples into the kernel hyperparameter.

After some unstable runs, we decided to implement an unbiased estimate of MMD^2 [3] between real and fake batches with the same RBF kernel (same normalized z-space and σ^2). This term is optional and is used as a small stabilizer:

$$\mathcal{L} = \text{KSD}^2_{\text{Nyström}} + \lambda\, \text{MMD}^2.$$

[3] To be fully precise, the estimator is clamped in order to maintain non negativity: hence it looses its unbiasedness

The motivation is practical: KSD gradients can be noisy when the score estimate is imperfect or when Rényi landmarks poorly represent the batch. MMD provides a discrepancy that is often smoother and can regularize early training without requiring a discriminator since MMD is purely sample-based.

Through the class `RényiNystroemKSD` we define the accelerated KSD objective. We first construct the *Stein kernel* $h_p(x, y)$ underlying KSD, but with a crucial implementation choice: the base kernel is an RBF in the projected normalized feature space $z = (Px - \mu)/\sigma$. This reduces dimensionality and improves conditioning.

The code computes:

- $K(x, y) = \exp(-\|z_x - z_y\|^2/(2\sigma^2))$,
- $\nabla_x K$ and $\nabla_y K$ using the chain rule through $z = Px$,
- the Stein kernel combination of score-score, score-grad-kernel, and trace Hessian terms.

This matches the standard KSD formulation, with a base kernel defined on a projected feature representation while the Stein operator acts in pixel space. The trace term is implemented analytically for the RBF kernel, which avoids expensive second derivatives.

To ensure the Rényi-Nyström landmark Gram matrix is numerically stable we proceed in symmetrizing and adding a ridge/PSD shift. This is essential: Rényi-Nyström KSD requires solving linear systems in H_{mm}, and if H_{mm} is ill-conditioned, gradients explode or become undefined.

Hence, we implement the two-sample Rényi-Nyström KSD statistic:

$$\widehat{\mathrm{KSD}}_N^2 \;\approx\; \delta^\top H_{mm}^{-1}\delta, \qquad \delta = \beta_f - \beta_r,$$

where β is the Rényi-Nyström feature mean (computed from $H_{m,n}$) referring to fakes (β_f) or reals (β_r). Importantly, real batches and landmark selection are detached: gradients flow through $fake$ samples only. This enforces the intended learning dynamic: the generator moves to reduce discrepancy to the fixed data distribution rather than "chasing" moving kernel parameters. Lastly we provide a diagnostic one-sample KSD estimate for a given batch which will be used later in evaluation across t.

Our Rényi ispired selection is implemented for Nyström acceleration. Conceptually, Nyström requires $m \ll n$ representative points. Uniform random landmarks are simple but can oversample dense regions. Our selection procedure aims to pick a diverse subset (high entropy intuition) using a greedy rule based on the Stein kernel matrix H.

Key implementation constraints are enforced explicitly:

- landmarks are selected from X_t corresponding to real images at time t.
- feature normalization receives the real-only μ, σ so the kernel geometry used for selection matches the geometry used in training.

This design prevents landmark choice from becoming an implicit trainable component and ensures the Rényi-Nyström subspace is directly related to the data distribution.

We use a standard convolutional discriminator ending in a single logit. This enables a direct baseline trained with logit loss. Using a conventional discriminator avoids architectural differences: the comparison focuses on the objective (adversarial vs. KSD-based) rather than a novel discriminator design.

We train the model using a mixture of diffusion timesteps t, motivated by the fact that score estimation and Stein discrepancies vary across noise levels. Early in training, higher-noise timesteps correspond to smoother distributions and provide more stable discrepancy estimates, while later training progressively emphasizes lower-noise timesteps to refine sample quality. The sampling distribution over timesteps evolves over training, inducing a curriculum-like learning dynamics. In our experiments, a single timestep is sampled per iteration and used to compute the training loss.

At each iteration our KSD-GAN training loop:

1. Samples a real batch x_0 and latent z, generates fake $x_0 = G_\theta(z)$.
2. For each selected timestep t: forms x_t for both real and fake using the diffusion forward marginal.
3. Computes real-only feature normalization (μ, σ) and bandwidth σ^2.
4. Selects real-only Rényi-Nyström landmarks.
5. Computes two-sample Rényi-Nyström-KSD where gradients flow through fake samples only.
6. Adds an optional MMD stabilizer.
7. Backpropagates and updates the generator; applies gradient clipping to prevent occasional spikes from ill-conditioned kernel solves.

This pipeline operationalizes the proposal: KSD replaces the discriminator, Nyström makes it efficient, Rényi selection improves landmark quality, and stabilizers address practical training issues.

4.4 Alternative feature map: frozen ResNet embeddings

In addition to the random linear projection used in the main KSD-GAN implementation, we also consider a variant in which the base kernel is defined on deep image features extracted from a frozen ResNet-18 pretrained on ImageNet. This variant is motivated by the fact that convolutional features encode higher-level information and may induce a kernel geometry better aligned with the similarity in images.

In this setting, the feature map $z(x)$ is given by the output of an intermediate ResNet block/layer followed by global average pooling. The ResNet parameters are kept fixed throughout training. Input images are first mapped from $[-1, 1]$ to $[0, 1]$ and normalized using ImageNet mean and standard deviation (and optionally resized to 224×224). As in the random-projection case, the resulting embeddings are standardized using real-only statistics,

$$\tilde{z}(x) = \frac{z(x) - \mu}{\sigma},$$

where (μ, σ) are computed from real samples at the same diffusion time t. The RBF bandwidth σ^2 is selected via a median heuristic on pairwise distances in this normalized feature space, again using real samples only.

The overall KSD objective remains unchanged, but the implementation of the Stein kernel differs from the linear-projection case. Because the ResNet feature map is nonlinear, closed form expressions for kernel gradients are no longer available. Instead, gradients of the base kernel with respect to pixel-space inputs are computed via Jacobian products through the frozen feature extractor.

To preserve computational tractability and the real-only, no-gradient design of kernel hyperparameters and landmarks, we adopt the following strategy:

- For real-only computations (landmark–landmark and landmark–real blocks in the Nyström approximation), only the score–score term $k(x, y)\, s(x)^\top s(y)$ of the Stein kernel is evaluated; gradient-dependent terms are omitted.
- For landmark–fake interactions, gradient-dependent Stein terms are included so that the KSD objective remains differentiable with respect to the generator parameters.
- The mixed Hessian trace term $\mathrm{tr}(\nabla_x \nabla_y k)$ is omitted in this variant to avoid higher-order derivatives through the deep feature map.

All other aspects of the training procedure are unchanged: landmark selection is performed on real samples only and detached from gradients, Nyström Gram matrices are stabilized via symmetrization and ridge regularization, and an optional MMD regularizer in the same feature space is added to improve stability.

4.5 Baseline GAN implementation

The standard GAN trains the same generator architecture `GenX0` against `DCGANDis`. The Discriminator labels use a small amount of label smoothing to reduce discriminator overconfidence, a standard stabilization trick. The generator is trained to maximize discriminator belief in fake images.

4.6 Evaluation metrics and comparison protocol

After training, we compare the two generative models by independently sampling from each generator. In both cases, latent vectors are drawn from a standard Gaussian distribution and passed through the corresponding trained generator to produce clean images in pixel space. No diffusion corruption, kernel statistics, or landmark selection is involved at this stage. The resulting samples are then visualized side by side to provide a qualitative comparison of the image distributions learned by the KSD-trained generator and the standard adversarial GAN.

To compare sample quality with standard practice, we also compute the Fréchet Inception Distance (FID) and the Kernel Inception Distance (KID) [8]. We extract 2048-dimensional Inception-v3 [4] feature embeddings from both real images and generated samples. Let μ_r, Σ_r denote the empirical mean and covariance of the feature embeddings of real images, and μ_f, Σ_f the corresponding quantities for generated images. The FID is then computed as

$$\mathrm{FID} = \|\mu_r - \mu_f\|^2 + \mathrm{tr}\Big(\Sigma_r + \Sigma_f - 2(\Sigma_r\Sigma_f)^{1/2}\Big).$$

KID is computed as an unbiased estimate of the squared Maximum Mean Discrepancy (MMD) between the distributions of real and generated Inception features, using a polynomial kernel

$$k(x, y) = (\gamma\, x^\top y + 1)^d.$$

Let $\{r_i\}_{i=1}^n$ and $\{f_j\}_{j=1}^m$ denote the Inception feature embeddings of real and generated images, respectively. The unbiased KID estimator is given by

$$\mathrm{KID} = \frac{1}{n(n-1)} \sum_{i \neq i'} k(r_i, r_{i'}) + \frac{1}{m(m-1)} \sum_{j \neq j'} k(f_j, f_{j'}) - \frac{2}{nm} \sum_{i,j} k(r_i, f_j).$$

4.7 Qualitative analysis

We present 64 samples generated by KSD-GAN with Random Projection (Linear), KSD-GAN (ResNet), and the Baseline GAN in Figure 1. Both variants of KSD-GAN perform poorly and fail to learn a meaningful representation of the underlying data distribution, whereas the Baseline GAN achieves substantially better results.

The samples generated by KSD-GAN (Linear) (Figure 1, left) are dominated by smooth, amorphous colour blobs with little discernible structure. Visual diversity across samples is minimal, indicating severe mode collapse, where the generator captures only a small subset of the data modes while ignoring others. The absence of sharp edges or coherent spatial patterns further suggests that the model fails to learn even coarse characteristics of the data distribution.

The outputs produced by KSD-GAN (ResNet) (Figure 1, middle) did not improve this behavior. Instead, they exhibit further degradation, with washed-out colours and an almost complete lack of recognizable shapes or structure.

In contrast, the Baseline GAN (Figure 1, right), trained under identical hyperparameters and hardware conditions, performs markedly better. The generated samples display recognizable structure, richer textures, and clear variation across instances, indicating more stable training dynamics and superior generalization.

[4] Inception-v3 is a deep convolutional neural network trained on the ImageNet dataset for large-scale image classification. In FID and KID, a fixed, pretrained Inception-v3 network is used as a feature extractor: images are mapped to a 2048-dimensional embedding obtained from a pooling layer. The network is kept fixed and is used solely for evaluation, not for training.

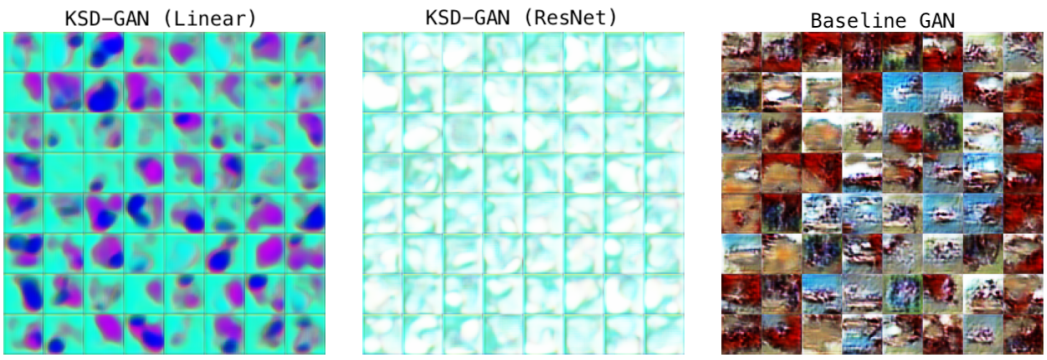

Figure 1: Qualitative comparison of generated samples. Each panel shows 64 samples generated by KSD-GAN (Linear) (left), KSD-GAN (ResNet) (middle), and the Baseline GAN (right).

	KSD-GAN (Linear)	Baseline GAN	SOTA GAN
FID	304.81	104.02	3.19
KID	0.3159	0.0669	N/A

Table 1: Quantitative comparison of generative performance across KSD-GAN (Linear), Baseline GAN, and the state-of-the-art (SOTA) GAN, evaluated using FID and KID metrics. Lower values indicate better sample fidelity and distributional alignment.

4.8 Quantitative analysis

We report the FID and KID scores in Table 1, as defined in Section 4.4, for all evaluated models and compare them against the state-of-the-art (SOTA): Diffusion GAN[5] [9]. Lower FID and KID values indicate higher sample fidelity and closer alignment with the real data distribution.

The KSD-GAN (Linear) exhibits very poor performance on both metrics when compared to the Baseline GAN, confirming its inability to learn a meaningful generative model. Following the qualitative analysis, we chose not to report quantitative results for KSD-GAN (ResNet) in order to conserve computational resources, as its generated samples showed no improvement and were qualitatively worse.

While the Baseline GAN performs substantially better than the KSD-GAN, it is still severely outperformed by the SOTA model. This performance gap is expected, as the Baseline GAN was trained under significantly lower computational constraints appropriate for this project.

Overall, the quantitative results reinforce the qualitative findings: the KSD-based objective fails to produce high-quality and diverse samples, whereas the Baseline GAN achieves more stable training and improved generalization, albeit far from state-of-the-art performance.

4.9 Failure Analysis

Despite extensive tuning efforts, the performance of the proposed KSD-GAN remained consistently poor. We experimented with increasing and decreasing the number of training iterations, varying batch sizes, adjusting the kernel and regularization parameters (including the MMD penalty coefficient), and applying gradient clipping, among other stabilization techniques. None of these interventions led to substantial or sustained improvements.

We suspect that one or several of the following factors contribute to the poor performance observed. First, the training process exhibits highly unstable loss dynamics, which prevents reliable convergence. Second, due to computational constraints, all models were trained for only 1,000 steps; it is therefore plausible that substantially longer training horizons are required for the KSD-based objective to

[5]We include this state-of-the-art (SOTA) FID value only as a rough reference point for contemporary GAN's capability. It is reported from [9] and was obtained under different training settings. Therefore, it should not be interpreted as a strictly comparable baseline to our runs, but rather as evidence of the gap between our lightweight setting and modern high-performance GAN training.

provide a meaningful learning signal. Third, while care was taken during implementation, the possibility of implementation-level errors cannot be fully ruled out.

Finally, and most critically, we believe that the use of a diffusion-based score function constitutes the biggest source of failure in our implementation. The score is evaluated on noised samples x_t and scales inversely with the noise level σ_t, which leads to severe instability as the noise level decreases. In practice, this can be seen from large and highly erratic loss values at small diffusion times, amplifying approximation errors in the score network and producing unreliable gradients for the generator. We believe this issue makes the score-based Stein formulation the most significant obstacle to effective KSD-GAN training in our experiments.

At the same time, we did not identify an alternative way to supply the KSD with a score function without relying on a pretrained score model. A possible direction for future work would be to investigate alternative score estimators or formulations of the Stein discrepancy that avoid explicit dependence on diffusion-based scores.

4.10 Code availability and compute

All code is version-controlled and shared via GitHub `https://github.com/dettorig/RenyiKSDforGANs` for reproducibility and collaboration. Experiments are GPU-accelerated; in particular, both the random-projection and ResNet-feature KSD-GAN implementations were trained and evaluated on NVIDIA A100 GPUs, accessed via rented cloud resources (Google Colab). This choice is motivated by the computational demands of Stein-kernel training with diffusion-based score evaluation and Nyström approximations: the repeated evaluation of score functions and kernel matrices makes training prohibitively slow on significantly less powerful GPUs and effectively infeasible on CPU-only hardware.

References

[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks, 2014.

[2] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[3] Florian Kalinke, Zoltán Szabó, and Bharath K. Sriperumbudur. Nyström Kernel Stein Discrepancy. In *Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 258 of *Proceedings of Machine Learning Research*. PMLR, 2025.

[4] José Cribeiro-Ramallo, Agnideep Aich, Florian Kalinke, Ashit Baran Aich, and Zoltán Szabó. The Minimax Lower Bound of Kernel Stein Discrepancy Estimation. *arXiv preprint*, 2025.

[5] Mark Girolami. Orthogonal Series Density Estimation and the Kernel Eigenvalue Problem. *Neural Computation*, 14(3):669–688, 2002.

[6] Alex Krizhevsky and Geoffrey Hinton. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, 2009.

[7] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, 2016.

[8] Mikołaj Bińkowski, Danica J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs, 2021.

[9] Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Diffusion-GAN: Training GANs with Diffusion, 2023.