LINEAR COMBINATIONS OF GAUSSIAN LATENTS IN GENERATIVE MODELS: INTERPOLATION AND BEYOND

Anonymous authors
Paper under double-blind review

ABSTRACT

Sampling from generative models has become a crucial tool for applications like data synthesis and augmentation. Diffusion, Flow Matching and Continuous Normalizing Flows have shown effectiveness across various modalities, and rely on Gaussian latent variables for generation. For search-based or creative applications that require additional control over the generation process, it has become common to manipulate the latent variable directly. However, existing approaches for performing such manipulations (e.g. interpolation or forming low-dimensional representations) only work well in special cases or are network or data-modality specific. We propose Combination of Gaussian variables (COG) as a general purpose method to form linear combinations of latent variables while adhering to the assumptions of the generative model. COG is easy to implement yet outperforms recent sophisticated methods for interpolation. As COG naturally addresses the broader task of forming linear combinations, new capabilities are afforded, including the construction of subspaces of the latent space, dramatically simplifying the creation of expressive low-dimensional spaces of high-dimensional objects.

1 INTRODUCTION

Generative models are a cornerstone of machine learning, with diverse applications including image synthesis, data augmentation, and creative content generation. Diffusion models (Ho et al., 2020; Song et al., 2020a;b) have emerged as a particularly effective approach to generative modeling for various modalities, such as for images (Ho et al., 2020), audio (Kong et al., 2020), video (Ho et al., 2022), and 3D models (Luo & Hu, 2021). A yet more recent approach to generative modelling is Flow Matching (FM) (Lipman et al., 2022), built upon Continuous Normalizing Flows (Chen et al., 2018), generalising the diffusion approach to allow for different probability paths between data and latent distribution, e.g. through Optimal Transport (Gulrajani et al., 2017; Villani et al., 2009).

As well as generation, these models allow inversion where, by running the generative procedure in the opposite direction, data objects can be transformed deterministically into a corresponding realization in the latent space. Such invertible connections between latent and data space provides a convenient mechanism for controlling generated objects by manipulating their latent vectors. The most common manipulation is to attempt semantically meaningful interpolation of two generated objects (Song et al., 2020a;b; Luo & Hu, 2021) by interpolating their corresponding latent vectors. However, the optimal choice of interpolant remains an open question, with simple approaches like linear interpolation leading to intermediates for which the model fails to generate plausible objects.

White (2016) argue that poor quality generation under linear interpolation is due to mismatch between the norms of the intermediate vectors and those of the Gaussian vectors that the model has been trained to expect. Indeed, it is well-known that the squared norm of a D-dimensional unit Gaussian follows the chi-squared distribution. Consequently, likely samples are concentrated in a tight annulus around a radius $\sqrt{D}$ — a set which is not closed under linear interpolation. Because of this it is common to rely instead on spherical interpolation (Shoemake, 1985) (SLERP) that maintain similar norms as the endpoints. Motivated by poor performance of interpolation between the latent vectors provided by inverting natural images, alternatives to SLERP have been recently proposed (Samuel et al., 2023; Zheng et al., 2024). However, these procedures are expensive, require tuning, and — in common with SLERP — are difficult to generalize to other kinds of manipulations beyond interpolation, such as creating subspaces exploited by latent space optimisation methods (Gómez-Bombarelli et al., 2018).

In this work, we demonstrate that successful generation from latent vectors is strongly dependent on their broader statistical characteristics matching those of Gaussian samples — a stronger condition than just having likely norms. We show that effective manipulation of latent spaces can be achieved by following the simple guiding principle of *adhering to the modelling assumptions of the generation process*. Our primary contributions are as follows:

- We show that even if a latent leads to a valid object upon generation, e.g. reconstructs an inverted image, it can fail to provide effective interpolation if it lacks Gaussian characteristics — as diagnosed by statistical normality tests.
- We introduce Combination of Gaussian variables (COG) — a simple scheme for ensuring that interpolation intermediates continue to match the latent distribution.
- We demonstrate that COG can be applied to create general linear combinations beyond interpolations, including centroid calculation (Figure 1) and, in particular, to define meaningful low-dimensional latent representations (Figure 2) — a goal achieved previously under significant expense and only for specific architectures (Kwon et al., 2022; Haas et al., 2024).
- We show that COG is easy to implement through a closed-form expression that makes no assumptions about the network structure or data modality, and demonstrate that it outperforms or match state-of-the-art baselines when applied to popular generative models.

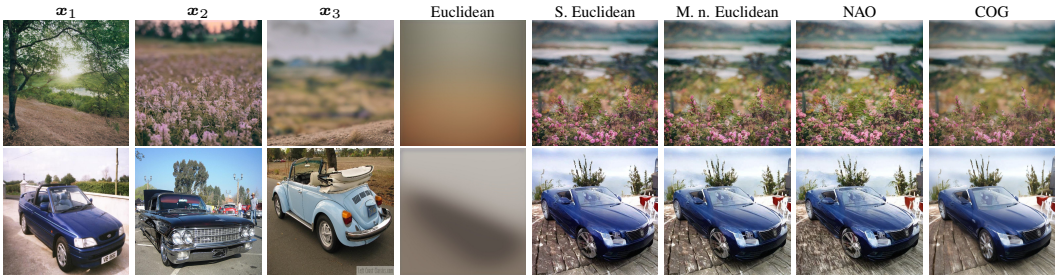


Figure 1: **Centroid determination**. Generation using Stable Diffusion 2.1 (Rombach et al., 2022) from the centroid of the latents corresponding to images $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, $\boldsymbol{x}_3$ using different methods. Note that our proposed method removes several artifacts, such as unrealistic headlights and chassi texture.

Figure 2: **Low-dimensional subspaces**. A 5-dimensional subspace from the flow matching model Stable Diffusion 3 (Esser et al., 2024) extracted using COG (left) from the latents corresponding to images $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_5$. The left plot show generations from uniform grid points across an axis-aligned slice of the subspace coordinate system, centered around the coordinate for $\boldsymbol{x}_1$. Each coordinate in the subspace correspond to a linear combination of latents, which define basis vectors. The right plot show the corresponding subspace *without* the proposed COG transformation. See Figure 6, Figure 7 and Section F in the appendix for additional examples.

2 BACKGROUND

2.1 GENERATIVE MODELLING WITH GAUSSIAN LATENT VARIABLES

The methodology presented in this paper is applicable to any generative model that generates objects through transformation of Gaussian samples. For clarity of exposition, we will focus primarily on popular Diffusion and Flow Matching models.

Diffusion models learn a process that reverses the effect of gradually adding noise to training data (Ho et al., 2020). The noise level is governed by a schedule designed so that the noised samples at the final time index T follow the latent distribution $\boldsymbol{x}^{(T)} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. To sample from the diffusion model, one starts by drawing a sample from the latent distribution before iteratively evaluating the reverse process with learnt parameters $\boldsymbol{\theta}$, denoising step-by-step, generating a sequence $\{\boldsymbol{x}^{(T)}, \boldsymbol{x}^{(T-1)}, \ldots, \boldsymbol{x}^{(0)}\}$

$$\boldsymbol{x}^{(t-1)} \sim p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(t-1)} \mid \boldsymbol{x}^{(t)}),$$

finally producing a generated object $\boldsymbol{x}^{(0)}$. Diffusion has also been extended to the continuous time setting by Song et al. (2020b), where the diffusion is expressed as a stochastic differential equation. Note that, by using the Denoising Diffusion Implicit Model (DDIM) (Song et al., 2020a) or the probability flow formulation in Song et al. (2020b), the generation process can be made deterministic, i.e. the latent representation $\boldsymbol{x}^{(T)} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ completely specifies the generated object $\boldsymbol{x}^{(0)}$.

Flow matching Lipman et al. (2022) is an efficient approach to train Continuous Normalizing Flows (CNF) Chen et al. (2018) — an alternative class of generative models that builds complicated distributions from simple latent distributions using differential equations. CNFs model the evolution of data points over continuous time using an ordinary differential equation (ODE) parameterized by a neural network, providing a deterministic relationship between latent and object space. Mathematically, the transformation of a latent sample $\boldsymbol{x}^{(T)} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ at time $t = T$ to $\boldsymbol{x}^{(t)}$ at time t is governed by $\boldsymbol{f}(\boldsymbol{x}^{(t)}, t; \boldsymbol{\theta})$, where $\boldsymbol{f}(\cdot; \boldsymbol{\theta})$ is a neural network with parameters $\boldsymbol{\theta}$. Flow matching allow the transformation dynamics of the CNF to be learnt without needing to simulate the entire forward process during training, which improves stability and scalability.

Note that both diffusion and flow matching models can generate in a deterministic manner, where the realization of the Gaussian latent variable completely specify the generated data object, e.g. the image. Moreover, by running their deterministic generation formulation in reverse, we can obtain the corresponding latent representation associated with a known object, which is called inversion. In this paper we focus on the effect of properties of the latent vectors, and their manipulation, on the generation process. For simplicity of notation, henceforth we drop the time index and refer to the latent variable $\boldsymbol{x}^{(T)}$ as $\boldsymbol{x}$, and use subscript indexing to denote realizations of the latent variable.

2.2 INTERPOLATION OF LATENTS

The most common latent space manipulation is interpolation, where we are given two latent vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, referred to as seeds, and obtain intermediate latent vectors. The simplest approach is to interpolate linearly between the seeds to get intermediates

$$\boldsymbol{y}^t_{\text{lin}} = t\boldsymbol{x}_1 + (1-t)\boldsymbol{x}_2 \quad \text{for} \quad t \in [0, 1].$$

However, as discussed in (White, 2016) — originally in the context of Variational Autoencoders (Kingma, 2013) and Generative Adversarial Networks (Goodfellow et al., 2014) — this yields intermediates with unlikely norms, and generation from these lead to highly implausible generated objects. To address this they propose spherical interpolation (SLERP) (Shoemake, 1985), which instead builds interpolants via

$$\boldsymbol{y}^t_{\text{SLERP}} = \frac{\sin t\theta}{\sin \theta}\boldsymbol{x}_1 + \frac{\sin((1-t)\theta)}{\sin \theta}\boldsymbol{x}_2 \quad \text{for} \quad t \in [0, 1] \quad \cos\theta = \frac{\langle \boldsymbol{x}_1, \boldsymbol{x}_2 \rangle}{||\boldsymbol{x}_1|| ||\boldsymbol{x}_2||},$$

to maintain similar norms for the intermediates as the endpoints. SLERP has become popular and is the standard method also in the context of diffusion models (Song et al., 2020a;b).

Motivated by degrading performance of SLERP when interpolating between the latent vectors corresponding to inverted natural images, Zheng et al. (2024) propose to add additional Gaussian

noise to interpolants via the inversion procedure, which is run for each interpolation. The noise level is tuned to trade-off between generation quality and adhering to the seed images. Alternatively, also motivated by observed degradation of SLERP performance on inverted images, Samuel et al. (2023) advocate for Norm-Aware Optimisation (NAO) which uses back-propagation to identify interpolation paths $\gamma : [0, 1] \to \mathbb{R}^d$ that solve

$$\inf_{\gamma} - \int \log \mathbb{P}(\gamma(s))ds \quad \text{s.t.} \quad \gamma(0) = \boldsymbol{x}_1, \gamma(1) = \boldsymbol{x}_2,$$

where $\log \mathbb{P} : \mathbb{R}^d \to \mathbb{R}$ is the log likelihood of the squared norm under its sampling distribution $\chi^2(D)$ for unit Gaussian samples. NAO can also calculate centroids of K latents by simultaneous optimization of paths between the centroid and each respective latent.

All the proposed interpolation methods above aim to make the intermediates adhere to statistics of Gaussian samples. However, these recent approaches require significant computation, have hyperparameters to tune, and lack theoretical guarantees. Morover — in common with spherical interpolation — they are difficult to generalize to more general expressions of the latents beyond interpolants, such as building subspaces based on their span, which we will show enables expressive low-dimensional representations.

In this work we will propose a simple technique that guarantees that interpolations follow the latent distribution given that the original (seed) latents do, and which lets us go beyond interpolation. Moreover, we will address how to assess the distributional assumption of the latents, to simplify the diagnosis of errors at the source.

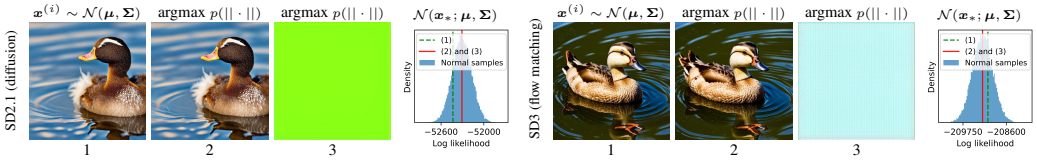


Figure 3: **Likelihood and norm insufficient**. Column (1) of each panel shows an image generated using a random sample from the associated latent distribution for the diffusion model of Rombach et al. (2022) (left) and the flow matching model of Esser et al. (2024) (right). The columns (2) and (3) both show images generated from latents with the most likely norm for their respective latent distribution. Columns (2) use the same normal samples as in columns (1) but rescaled to have this norm, which like in columns (1) yields realistic images. Meanwhile, columns (3) show the failed generation from constant vectors $s\boldsymbol{I}$ that although also are scaled to have the most likely norm according to the latent distribution, lacks other characteristics that the network were trained to accept, even though its likelihood $\mathcal{N}(s\boldsymbol{I}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is typical of real samples. Moreover, the distribution mode $\boldsymbol{\mu}$, which also lacks needed characteristics to generate plausible images, has vastly higher log likelihood than any realistic sample; -33875 and -135503 for the two models, respectively. See Table 2 for an example of failed generation using the mode.

3 Assessing the Normality of Latents

Before introducing our proposed interpolation method, we first explore evidence for our central hypothesis that tackles a misconception underpinning current interpolation methods:

Having a latent vector with a norm that is likely for a sample from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is not a sufficient condition for plausible sample generation. Rather, plausible generation requires a latent vector that match the characteristics of a sample from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ more generally (as evaluated by normality tests), with a likely norm being only one such characteristic.

The characteristics of the random variable $\boldsymbol{x}$ may be described by a collection of statistics, e.g. its mean, variance or norm. Above, we hypothesise that if a specified latent $\boldsymbol{x}_*$ has an extremely unlikely value for a characteristic for which the network has come to rely on, then implausible generation will follow. A norm is unlikely if it has a low likelihood under its sampling distribution, see Section 1. While the norm has been shown to be an important statistic (Samuel et al., 2023), our Figure 3 illustrates on a popular diffusion model (Rombach et al., 2022) and a flow matching model (Esser

et al., 2024) that a latent with a likely norm — even the most likely — can still induce failure in the generative process if some other critical characteristics are unmet.

In general, neural networks have many degrees of freedom, and so it is difficult to determine and list the input characteristics that a specific model relies on. Therefore, rather than trying to determine all necessary statistics for each specific model, we propose instead to rely on standard methods for normality testing, a classic area of statistics (Razali et al., 2011; Kolmogorov, 1933; Shapiro & Wilk, 1965) — exploring the (null) hypothesis that a latent vector $\boldsymbol{x}_* \in \mathbb{R}^D$ is drawn from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

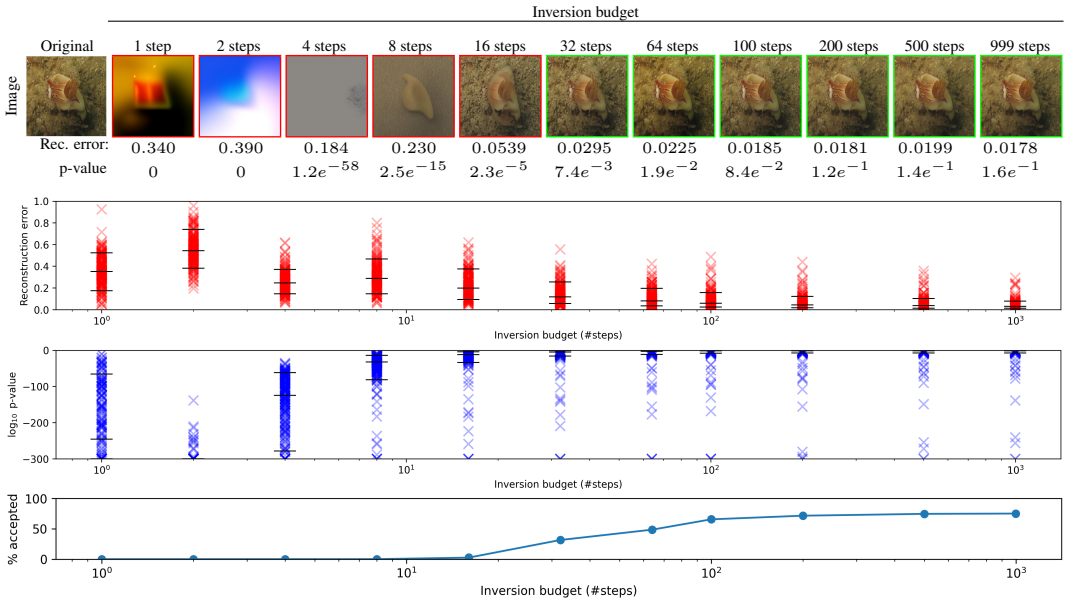


Figure 4: **Normality testing of latent vectors obtained from inversion**. The plot in the second row shows the LPIPS (Zhang et al., 2018) reconstruction errors of 200 inverted randomly selected images across 50 random classes from ImageNet1k (Deng et al., 2009) under various inversion step budgets for the diffusion model of Rombach et al. (2022). The third row plot shows the p-values of the Kolmogorov-Smirnov normality test of the corresponding latents obtained from the inversions. The three horizontal black lines per budget indicate the 10th, 50th and 90th percentiles. In the top row the reconstruction obtained from each inversion budget is shown for a randomly selected image, indicated with green when its latent were accepted (with a p-value of at least $1e^{-3}$), together with its reconstruction error and p-value. The number of steps used at generation is 999 in all cases.

3.1 Normality testing

Popular normality tests consider broad statistics associated with normality. For example, the classic Kolmogorov–Smirnov (Kolmogorov, 1933) test considers the distribution of the largest absolute difference between the empirical and the theoretical cumulative density function across all values. In Appendix D we investigate the effectiveness of a range of classic normality tests and tests based on the likelihood $\mathcal{N}(\boldsymbol{x}_*; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ as well as the likelihood of the norm statistic. We assess these methods across a suite of test cases including for vectors with unlikely characteristics and which we know, through failed generation, violate the assumptions of the model. In order to improve statistical power and detect even small deviations from normality of a single instance $\boldsymbol{x}_* \in \mathbb{R}^D$, we **equivalently** investigate the collection of D univariate samples $\{\epsilon_i\}_{i=1}^D$ under the hypothesis $\epsilon_1, \ldots, \epsilon_d \sim \mathcal{N}(0, 1)$ where $\boldsymbol{\epsilon} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}), \boldsymbol{\epsilon} = [\epsilon_1, \ldots, \epsilon_d]$. We find that Kolmogorov-Smirnov and Cramér–von Mises both correctly rejects the failure cases (by assigning low p-values) while reporting calibrated p-values for real samples, with **Kolmogorov-Smirnov** doing so with the lowest p-values for all failure cases.

3.2 Normality testing on inversions

The most common way to obtain a seed latent $\boldsymbol{x}_*$ is by inverting a particular data example (as described in Section 2). However, as inversion is subject to a finite computation budget (i.e. finite

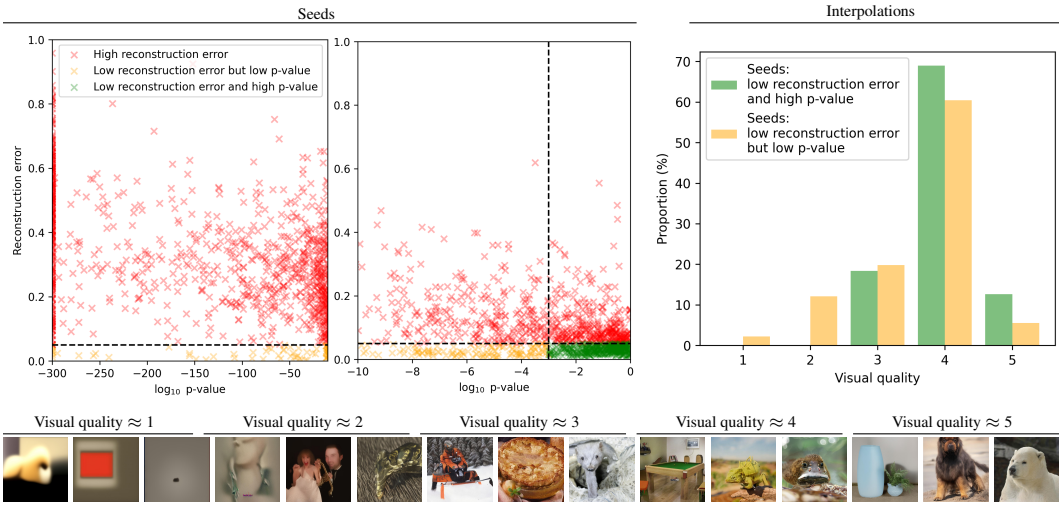


Figure 5: **Normality rejection linked to failure of interpolants** Shown in the left (plus middle) plot is the LPIPS (Zhang et al., 2018) reconstruction error and the Kolmogorov-Smirnov p-values for all inversions across the various inversion budgets shown in Figure 4. Note that although latents with high p-values tend to have low reconstruction errors, there are many latents with low reconstruction errors that have low p-values. Shown in the right plot is the visual quality determined by the Q-Align score (Wu et al., 2023) of spherical interpolations between pairs of inversions of same-class as seeds, considering seeds with low and high p-values respectively among the reconstructing seeds. We note that if additionally considering the p-value we are able to avoid seed latents that, when interpolated, will yield low visual quality. In the bottom row we show randomly selected examples of interpolations of each respective visual quality for reference. Low reconstruction error is defined as below 0.05, and high p-values as greater than $1e^{-3}$ which is a level expected for real samples from the latent distribution. The number of steps used at generation is 999 (maximum for the model) from all latents.

inversion steps), numerical imprecision, and because the particular data instance may not be drawn from the data distribution that the model approximates, the resulting latent vector $\boldsymbol{x}_*$ may not have characteristics matching samples from $\boldsymbol{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. We will now assess whether normality testing can be used to predict if latent vectors obtained through inversion are sufficiently sample-like to yield plausible generations. For simplicity in what follows, we will colloquially refer to a failure to reject the hypothesis that the latent follows $\boldsymbol{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ as acceptance of the latent.

Figure 4 demonstrate that high p-values are strongly linked to latents obtained from inversions reproducing the image with good quality. The p-value indicate the probability of observing the latent vector by chance *given* that it is drawn from the latent distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. If this value is extremely low that is a strong indicator that the latent lack characteristics expected of samples from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. In the figure we see that at low inversion budgets (with high reconstruction errors) most p-values are $1e^{-50}$ or lower, and then reach values you expect to see by chance for real samples (around $1e^{-3}$) at budgets where the reconstruction errors tend to be small. However, we will show that the p-value provide additional information over the reconstruction error about the quality of the latent.

We now examine how the p-values of two inversions relate to our ability to interpolate them with good quality. Figure 5 demonstrates that although latents with high p-values tend to have low reconstruction errors, there are many latents with low reconstruction errors that have low p-values. This shows that just because the process of going from object to latent and back reproduces the object that does not necessarily mean that the latent acquired is well characterised as a sample from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. The lack of such characteristics can be inherited when manipulating the latent; in Appendix 8 we show that interpolants typically have low p-values if any of its seed latents do. Figure 5 also demonstrates that if we beyond object reconstruction require the p-values of latents to be high — at levels expected by real samples — we are able to avoid using seeds that, when interpolated, yield low visual quality.

In this section we have presented evidence for our central hypothesis —that it is critical for latents to adhere to a broad range of Gaussian statistics in order to yield plausible generations. Inversion,

which is a common way to obtain the latents used as seeds, is a source of error which can prevent successful interpolation as their lack of Gaussian characteristics — as we show is indicated by low p-values using a classic normality test — is inherited by the interpolants. We have demonstrated that low p-values of seed latents is associated with lower quality interpolants even when the seed latents reproduces the original image. Therefore we propose normality testing of latents, a practice which we have not yet seen adopted in the generative modelling literature. In the next section we will propose a method where guarantees are provided for interpolants of Gaussian seed latents, removing the interpolation itself — and a wider class of latent space manipulations — as a source of error.

4 COG: LINEAR COMBINATIONS OF GAUSSIAN LATENTS

Now, equipped with the knowledge that matching broad characteristics of a sample from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is critical, we propose a simple scheme for forming linear combinations of seed latents that maintain their Gaussian characteristics. In all that follows, we assume that these seed latents are sampled from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. In this section we change the notation slightly, where a seed latent $\boldsymbol{x}_k \in \mathbb{R}^D$ rather than being a realization of a random variable is itself a random variable, following the predetermined Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. We assume access to K such seed latent variables $\{\boldsymbol{x}_k\}_{k=1}^K$ and attempt to form new variables following the same distribution.

Let $\boldsymbol{y}$ be a linear combination of the K latent variables

$$\boldsymbol{y} := \sum_{k=1}^{K} w_k \boldsymbol{x}_k = \boldsymbol{w}^T \boldsymbol{X}, \tag{1}$$

where $w_k \in \mathbb{R}$, $\boldsymbol{w} = [w_1, w_2, \ldots, w_K]$ and $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K]$. Then we have that $\boldsymbol{y}$ is also a Gaussian random variable, with mean and covariance

$$\boldsymbol{y} \sim \mathcal{N}(\alpha\boldsymbol{\mu}, \beta\boldsymbol{\Sigma}) \qquad \alpha = \sum_{k=1}^{K} w_k \qquad \beta = \sum_{k=1}^{K} w_k^2. \tag{2}$$

In other words, $\boldsymbol{y}$ is only distributed as $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ in the specific case where (a) $\alpha\boldsymbol{\mu} = \boldsymbol{\mu}$ and (b) $\beta\boldsymbol{\Sigma} = \boldsymbol{\Sigma}$ — an observation which we now use to explain the empirically observed behaviour of existing interpolation methods. Firstly, for linear interpolation, where $\boldsymbol{w} = [v, 1-v], v \in [0, 1]$, (b) holds only for the endpoints $v = \{0, 1\}$, and so leads to implausible generations for interpolants (as demonstrated empirically in Figure 11). In contrast, in the popular case of high-dimensional unit Gaussian latent vectors, spherical interpolants have $\beta \approx 1,\ \forall v \in [0, 1]$, as proven in Appendix B, and (a) is met as $\alpha\mathbf{0} = \mathbf{0}$, which is consistent with plausible interpolations (see Figure 11).

In this work, we instead propose transforming linear combinations such that $\alpha = \beta = 1$, for any $\boldsymbol{w} \in \mathbb{R}^K$, thus **exactly meeting the criteria** for **any** linear combination and **any** choice of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. We define a transformed random variable substituting the latent linear combination $\boldsymbol{y}$

$$\boldsymbol{z} = \boldsymbol{a} + \boldsymbol{B}\boldsymbol{y} \qquad \boldsymbol{a} = (1 - \frac{\alpha}{\sqrt{\beta}})\boldsymbol{\mu} \qquad \boldsymbol{B} = \frac{1}{\sqrt{\beta}}\boldsymbol{I} \tag{3}$$

for which it holds that $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ given latent variables $\boldsymbol{x}_k \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. See Section A for the derivations of Equation 2 and 3 and accompanying proofs. The weights $\boldsymbol{w}$, which via α and β together with the set of K original (seed) latents specify the transformed linear combination $\boldsymbol{z}$, depend on the operation, represented as particular linear combinations. Below are a few examples of popular operations (linear combinations) to form $\boldsymbol{y}$; these are used as above to, for the corresponding weights $\boldsymbol{w}$, obtain $\boldsymbol{z}$ following the distribution expected by the generative model.

- **Interpolation**: $\boldsymbol{y} = v\boldsymbol{x}_1 + (1-v)\boldsymbol{x}_2$, where $\boldsymbol{w} = [v, 1-v]$ and $v \in [0, 1]$.
- **Centroid Determination**: $\boldsymbol{y} = \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{x}_k = \sum_{k=1}^{K} \frac{1}{K}\boldsymbol{x}_k$, where $\boldsymbol{w} = [\frac{1}{K}]^K$.
- **Subspace Projection**: Suppose we wish to build a navigable subspace spanned by linear combinations of K latent variables. By performing the QR decomposition of $\boldsymbol{A} := [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K] \in \mathbb{R}^{D \times K}$ to produce a semi-othonormal matrix $\boldsymbol{U} \in \mathbb{R}^{D \times K}$ (as the Q-matrix), we can then define a subspace projection of any new $\boldsymbol{x}$ into the desired

subspace via $s(\boldsymbol{x}) := \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{x} = \boldsymbol{U}\boldsymbol{h} \in \mathbb{R}^D$. The weights $\boldsymbol{w}$ for a given point in the subspace $s(\boldsymbol{x})$ are given by $\boldsymbol{w} = \boldsymbol{A}_\dagger s(\boldsymbol{x}) = \boldsymbol{A}_\dagger \boldsymbol{U}\boldsymbol{h} \in \mathbb{R}^K$ where $\boldsymbol{A}_\dagger$ is the Moore–Penrose inverse of $\boldsymbol{A}$. See the derivation of the weights and proof in Appendix C. One can directly pick coordinates $\boldsymbol{h} \in \mathbb{R}^K$ and convert to (yet uncorrected) subspace latents by $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{h}$, which are subsequently corrected via Equation 3. In Figure 2 we use grids in $\mathbb{R}^K$ to set $\boldsymbol{h}$.

5 EXPERIMENTS

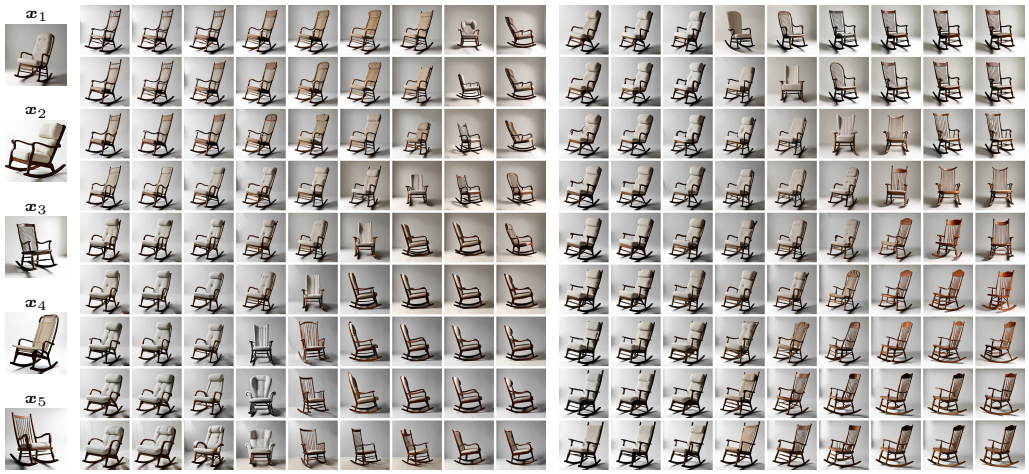


Figure 6: **Low-dimensional subspaces**. The latents $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_5$ (corresponding to images) are converted into basis vectors and used to define a 5-dimensional subspace. The grids show generations from uniform grid points in the subspace coordinate system, where the left and right grids are for the dimensions $\{1, 2\}$ and $\{3, 4\}$, respectively, centered around the coordinate for $\boldsymbol{x}_1$. Each coordinate in the subspace correspond to a linear combination of the basis vectors, which through COG yield valid latents. The flow matching model Stable Diffusion 3 (Esser et al., 2024) is used in this example.

We now assess our proposed transformation scheme experimentally, performing qualitative and quantitative comparisons to the respective baselines for interpolation and centroid determination. We then demonstrate several examples of low-dimensional subspaces on popular diffusion models and a flow matching model.

5.1 LINEAR INTERPOLATION AND CENTROID DETERMINATION

For the application of interpolation, we compare our proposed COG to linear interpolation (LERP), spherical linear interpolation (SLERP), and Norm-Aware Optimization (NAO) Samuel et al. (2023).

In contrast to the all the other considered interpolation methods (including COG) which only involve analytical expressions, NAO requires a numerical optimization scheme based on a discretization of a line integral. We closely follow the evaluation protocol in Samuel et al. (2023), basing the experiments on Stable Diffusion (SD) 2.1 (Rombach et al., 2022) and inversions of random images from 50 random classes from ImageNet1k (Deng et al., 2009), and assess visual quality and preservation of semantics using FID scores and class prediction accuracy, respectively. For the interpolation we (randomly without replacement) pair the 50 images per class into 25 pairs, forming 1250 image pairs in total.

Interpolation			
Method	**Accuracy**	**FID**	**Time**
LERP	3.92%	199	$6e^{-3}$s
SLERP	64.6%	42.6	$9e^{-3}$s
NAO	62.1%	46.0	30s
COG (ours)	**67.4%**	**38.9**	$6e^{-3}$s
Centroid determination			
Method	**Accuracy**	**FID**	**Time**
Euclidean	0.286%	310	$4e^{-4}$s
Standardized Euclidean	44.6%	88.8	$1e^{-3}$s
Mode norm Euclidean	44.6%	88.4	$1e^{-3}$s
NAO	44.0%	93.0	90s
COG (ours)	**46.3%**	**87.7**	$6e^{-4}$s

Table 1: Quantitative comparisons of baselines.

For the centroid determination we compare to NAO, the Euclidean centroid $\bar{\boldsymbol{x}} = \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{x}_k$ and two transformations thereof; "standardized Euclidean", where $\bar{\boldsymbol{x}}$ is subsequently standardized to have mean zero and unit variance (as SD 2.1 assumes), and "mode norm Euclidean", where $\bar{\boldsymbol{x}}$ is rescaled for its norm to equal the maximum likelihood norm $\sqrt{D}$. For each class, we form 10 3-groups, 10 5-groups, 4 10-group and 1 25-group, sampled without replacement per group, for a total of 1250 centroids per method. For more details on the experiment setup and settings, see Appendix E.

In Table 1 we show that our method outperforms the baselines in terms of FID scores and accuracy, as calculated using a pre-trained classifier following the evaluation methodology of Samuel et al. (2023). For an illustration of centroids and interpolations see Figure 1 and Figure 16, respectively. The evaluation time of an interpolation path and centroid, shown with one digit of precision, illustrate that the analytical expressions are significantly faster than NAO. Surprisingly, NAO did not perform as well as spherical interpolation and several other baselines, despite that we used their implementation and it was outperforming these methods in Samuel et al. (2023). We note that one discrepancy is that we report FID scores using the standard number of features (2048), while in their paper they are using fewer (64), which in Seitzer (2020) is not recommended since it does not necessarily correlate with visual quality. In the appendix we include results using FID scores using all settings of features. We note that, in our setup, the baselines performs substantially better than reported in their setup - including NAO in terms of FID scores using the 64 feature setting (1.30 vs 6.78), and class accuracy during interpolation (62% vs 52%). See Section G in the appendix for more details and ablations.

5.2 Low-dimensional subspaces

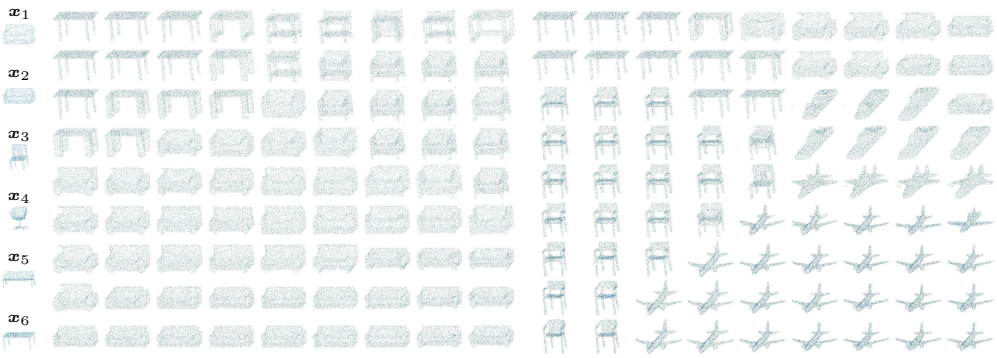


Figure 7: **Model-agnostic subspace definitions**. The latents $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_6$ (which via generation corresponds to point clouds) are converted into basis vectors and used to define a 6-dimensional subspace. The grids show generations from uniform grid points in the subspace coordinate system, where the left and right grids show dimensions $\{1, 2\}$ centered around the coordinate for $\boldsymbol{x}_1$ and $\boldsymbol{x}_3$, respectively. The diffusion model DDMI (Park et al., 2024) trained on ShapeNet (Chang et al., 2015) data is used this example. COG allows subspaces to be defined without any model-specific treatment.

In Figure 2 and Figure 6 we illustrate slices of a 5-dimensional subspace of the latent space of a flow matching model, indexing high-dimensional images of sports cars and rocking chairs, respectively. The subspaces here are defined using five images (one per desired dimension), formed using our COG method described in Section 4 to transform the linear combinations of the corresponding projections. Our approach is dimensionality and model-agnostic, which we illustrate in Figure 7, where the same procedure is used on a completely different model without adaptations; where we define a 6-dimensional subspace of the latent space of a diffusion model for point clouds. In Figure 9 in the appendix we show corresponding grids to the sports cars in Figure 2 without the proposed COG transformation, which either leads to implausible images or the same image for each coordinate, depending on the model. In the appendix (Section F) we also include more slices and examples; including subspaces using the diffusion model Stable Diffusion 2.1 (Rombach et al., 2022).

6 RELATED WORK

6.1 GENERATIVE MODELS WITH NON-GAUSSIAN PRIORS

In Fadel et al. (2021), in the context of normalizing flows and Davidson et al. (2018) in the context of VAEs, Dirichlet and von Mises-Fisher prior distributions are explored with the goal of improving interpolation performance. However, these method requires training the model with the new latent distribution, which is impractical and untested for the large pretrained models we consider in this work.

6.2 CONDITIONAL DIFFUSION

An additional way to control the generation process of a diffusion model is to guide the generative process with additional information, such as text and labels, to produce outputs that meet specific requirements, such as where samples are guided towards a desired class (Dhariwal & Nichol, 2021; Ho & Salimans, 2022), or to align outputs with textual descriptions (Radford et al., 2021). Conditioning is complementary to latent space manipulation. For example, when making Figure 2 we used conditioning (a prompt) to force the generation of sports cars, however the variation of images fulfilling this constraint is encoded in the latent space.

6.3 LOW-DIMENSIONAL REPRESENTATIONS

We have shown that COG can provide expressive low-dimensional representations of latent spaces of generative models. To the best of our knowledge, the most closely related line of work was initiated in Kwon et al. (2022), where it was shown that semantic edit directions can recovered from activations of a UNet Ronneberger et al. (2015) denoiser network during generation. Using a pretrained CLIP (Contrastive Language-Image Pre-Training) (Radford et al., 2021) model to define directions within the inner-most feature map of the UNet architecture, named h-space, they show that some semantic edits, such as adding glasses to an image of a person, corresponds to linear edits in the h-space. More recently, Haas et al. (2024) demonstrate that h-space edit directions can also be found through Principle Component Analysis and Park et al. (2023) propose a pullback metric that transfers edits in the h-space into the original latent space. However, while these three approaches demonstrate impressive editing capabilities on images, they all require a modification to the generative process and are limited to diffusion models built with UNet architectures. In our work low-dimensional representations are instead formed analytically as linear subspaces based on a (free) choice of latent basis vectors. These basis vectors could be obtained through various methodologies, including via the pull-back metric in Park et al. (2023), which may be interesting to explore in future work.

7 CONCLUSION

In this paper we propose COG, a general and simple scheme to create linear combinations of latents that maintain a prespecified Gaussian distribution and demonstrate its effectiveness for interpolation and defining subspaces within the latent spaces of generative models. Of course, these linear combinations only follow the desired distribution if the original (seed) latents do. Therefore, we propose the adoption of normality tests to assess the validity of latents obtained from e.g. inversions. Still, existing methods for normality testing may not align perfectly with the characteristics dependent on by the network, and it may be an interesting direction of future work to develop tailored methods, along with more extensive comparisons of existing normality tests in the context of generative models.

REFERENCES

Robert B Ash. *Information theory*. Courier Corporation, 2012.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Harald Cramér. On the composition of elementary errors: First paper: Mathematical deductions. *Scandinavian Actuarial Journal*, 1928(1):13–74, 1928.

Tim R Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M Tomczak. Hyperspherical variational auto-encoders. In *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*, pp. 856–865. Association For Uncertainty in Artificial Intelligence (AUAI), 2018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024.

Samuel G Fadel, Sebastian Mair, Ricardo da S. Torres, and Ulf Brefeld. Principled interpolation in normalizing flows. In *Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part II 21*, pp. 116–131. Springer, 2021.

Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

René Haas, Inbar Huberman-Spiegelglas, Rotem Mulayoff, Stella Graßhof, Sami S Brandt, and Tomer Michaeli. Discovering interpretable directions in the semantic latent space of diffusion models. In *2024 IEEE 18th International Conference on Automatic Face and Gesture Recognition (FG)*, pp. 1–9. IEEE, 2024.

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. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in Neural Information Processing Systems*, 35:8633–8646, 2022.

Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

An Kolmogorov. Sulla determinazione empirica di una legge didistribuzione. *Giorn Dell'inst Ital Degli Att*, 4:89–91, 1933.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

Mingi Kwon, Jaeseok Jeong, and Youngjung Uh. Diffusion models already have a semantic latent space. *arXiv preprint arXiv:2210.10960*, 2022.

Richard J Larsen and Morris L Marx. *An introduction to mathematical statistics*. Prentice Hall Hoboken, NJ, 2005.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2837–2845, 2021.

Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, and Balaji Lakshminarayanan. Detecting out-of-distribution inputs to deep generative models using typicality. *arXiv preprint arXiv:1906.02994*, 2019.

Dogyun Park, Sihyeon Kim, Sojin Lee, and Hyunwoo J Kim. Ddmi: Domain-agnostic latent diffusion models for synthesizing high-quality implicit neural representations. *arXiv preprint arXiv:2401.12517*, 2024.

Yong-Hyun Park, Mingi Kwon, Jaewoong Choi, Junghyo Jo, and Youngjung Uh. Understanding the latent space of diffusion models through the lens of riemannian geometry. *Advances in Neural Information Processing Systems*, 36:24129–24142, 2023.

Karl Pearson. X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50(302):157–175, 1900.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Nornadiah Mohd Razali, Yap Bee Wah, et al. Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of statistical modeling and analytics*, 2(1): 21–33, 2011.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pp. 234–241. Springer, 2015.

Dvir Samuel, Rami Ben-Ari, Nir Darshan, Haggai Maron, and Gal Chechik. Norm-guided latent space exploration for text-to-image generation. *Advances in Neural Information Processing Systems*, 36, 2023.

Maximilian Seitzer. pytorch-fid: FID Score for PyTorch. `https://github.com/mseitzer/pytorch-fid`, August 2020. Version 0.3.0.

Samuel Sanford Shapiro and Martin B Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3-4):591–611, 1965.

Ken Shoemake. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pp. 245–254, 1985.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020a.

Yang Song, Jascha Sohl-Dickstein, Diederik Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

Michel Talagrand. Concentration of measure and isoperimetric inequalities in product spaces. *Publications Mathématiques de l'Institut des Hautes Etudes Scientifiques*, 81:73–205, 1995.

Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxvit: Multi-axis vision transformer. In *European conference on computer vision*, pp. 459–479. Springer, 2022.

Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

Tom White. Sampling generative networks. *arXiv preprint arXiv:1609.04468*, 2016.

Haoning Wu, Zicheng Zhang, Weixia Zhang, Chaofeng Chen, Liang Liao, Chunyi Li, Yixuan Gao, Annan Wang, Erli Zhang, Wenxiu Sun, et al. Q-align: Teaching lmms for visual scoring via discrete text-defined levels. *arXiv preprint arXiv:2312.17090*, 2023.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

PengFei Zheng, Yonggang Zhang, Zhen Fang, Tongliang Liu, Defu Lian, and Bo Han. Noisediffusion: Correcting noise for image interpolation with diffusion models beyond spherical linear interpolation. *arXiv preprint arXiv:2403.08840*, 2024.

A GAUSSIAN LINEAR COMBINATIONS

In Section 4 we introduced COG, which we use to transform latent variables arising from linear combinations — for example interpolations or subspace projections — such that they maintain the same distribution as the seed latents; the latent distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. We will now derive the distribution of a (uncorrected) linear combination $\boldsymbol{y}$ — which we show does not match the latent distribution — followed by the distribution of the transformed variable $\boldsymbol{z}$, which we show does.

Lemma 1. *Let $\boldsymbol{y}$ be a linear combination of K i.i.d. random variables $\boldsymbol{x}_k \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where $\boldsymbol{y}$ is defined as*

$$\boldsymbol{y} := \sum_{k=1}^{K} w_k \boldsymbol{x}_k = \boldsymbol{w}^T \boldsymbol{X},$$

with $w_k \in \mathbb{R}$, $\boldsymbol{y} \in \mathbb{R}^D$, $\boldsymbol{x}_k \in \mathbb{R}^D$, $\boldsymbol{w} = [w_1, w_2, \ldots, w_K]$ and $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K]$. Then $\boldsymbol{y}$ is a Gaussian random variable with the distribution

$$\boldsymbol{y} \sim \mathcal{N}(\alpha\boldsymbol{\mu}, \beta\boldsymbol{\Sigma}),$$

where

$$\alpha = \sum_{k=1}^{K} w_k \quad \textit{and} \quad \beta = \sum_{k=1}^{K} w_k^2.$$

Proof. Given $\boldsymbol{x}_k \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, we know that each $\boldsymbol{x}_k$ has mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

Define $\boldsymbol{y} = \boldsymbol{w}^T \boldsymbol{X} = \sum_{k=1}^{K} w_k \boldsymbol{x}_k$.

First, we calculate the mean of $\boldsymbol{y}$:

$$\mathbb{E}[\boldsymbol{y}] = \mathbb{E}\left[\sum_{k=1}^{K} w_k \boldsymbol{x}_k\right] = \sum_{k=1}^{K} w_k \mathbb{E}[\boldsymbol{x}_k] = \sum_{k=1}^{K} w_k \boldsymbol{\mu} = \left(\sum_{k=1}^{K} w_k\right) \boldsymbol{\mu} = \alpha\boldsymbol{\mu}.$$

Next, we calculate the covariance of $\boldsymbol{y}$:

$$\text{Cov}(\boldsymbol{y}) = \text{Cov}\left(\sum_{k=1}^{K} w_k \boldsymbol{x}_k\right) = \sum_{k=1}^{K} w_k^2 \text{Cov}(\boldsymbol{x}_k),$$

since the $\boldsymbol{x}_k$ are i.i.d. and thus $\text{Cov}(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}) = 0$ for $i \neq j$.

Given that $\boldsymbol{x}_k \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, we have $\text{Cov}(\boldsymbol{x}_k) = \boldsymbol{\Sigma}$. Therefore,

$$\text{Cov}(\boldsymbol{y}) = \sum_{k=1}^{K} w_k^2 \boldsymbol{\Sigma} = \left(\sum_{k=1}^{K} w_k^2\right) \boldsymbol{\Sigma} = \beta \boldsymbol{\Sigma}.$$

Hence, $\boldsymbol{y} \sim \mathcal{N}(\alpha \boldsymbol{\mu}, \beta \boldsymbol{\Sigma})$ with $\alpha = \sum_{k=1}^{K} w_k$ and $\beta = \sum_{k=1}^{K} w_k^2$.

□

Lemma 2. *Let $\boldsymbol{z}$ be defined as*

$$\boldsymbol{z} = \boldsymbol{a} + \boldsymbol{B}\boldsymbol{y},$$

where

$$\boldsymbol{a} = \left(1 - \frac{\alpha}{\sqrt{\beta}}\right) \boldsymbol{\mu} \quad \textit{and} \quad \boldsymbol{B} = \frac{1}{\sqrt{\beta}} \boldsymbol{I},$$

with $\boldsymbol{y} \sim \mathcal{N}(\alpha \boldsymbol{\mu}, \beta \boldsymbol{\Sigma})$. Then $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Proof. Given $\boldsymbol{y} \sim \mathcal{N}(\alpha \boldsymbol{\mu}, \beta \boldsymbol{\Sigma})$, we need to show that $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

First, we calculate the mean of $\boldsymbol{z}$:

$$\mathbb{E}[\boldsymbol{z}] = \mathbb{E}\left[\boldsymbol{a} + \boldsymbol{B}\boldsymbol{y}\right] = \boldsymbol{a} + \boldsymbol{B}\mathbb{E}[\boldsymbol{y}] = \left(1 - \frac{\alpha}{\sqrt{\beta}}\right) \boldsymbol{\mu} + \frac{1}{\sqrt{\beta}} \alpha \boldsymbol{\mu} = \boldsymbol{\mu}.$$

Next, we calculate the covariance of $\boldsymbol{z}$:

$$\text{Cov}(\boldsymbol{z}) = \text{Cov}\left(\boldsymbol{a} + \boldsymbol{B}\boldsymbol{y}\right) = \boldsymbol{B}\text{Cov}(\boldsymbol{y})\boldsymbol{B}^T = \frac{1}{\sqrt{\beta}} \beta \boldsymbol{\Sigma} \frac{1}{\sqrt{\beta}} = \boldsymbol{\Sigma}.$$

Since $\boldsymbol{z}$ has mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$, we conclude that

$$\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

□

B Spherical interpolants are approximately i.i.d. for high dimensional unit Gaussians

Spherical linear interpolation (SLERP) (Shoemake, 1985) is defined as

$$\boldsymbol{y} = w_1 \boldsymbol{x}_1 + w_2 \boldsymbol{x}_2 \tag{4}$$

where

$$w_i := \frac{\sin v_i \theta}{\sin \theta}, \tag{5}$$

$v_i \in [0, 1]$ and $v_2 = 1 - v_1$ and $\cos\theta = \frac{\langle \boldsymbol{x}_1, \boldsymbol{x}_2 \rangle}{||\boldsymbol{x}_1|| ||\boldsymbol{x}_2||}$, where $\cos\theta$ is typically referred to as the *cosine similarity* of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$.

As such, using Equation 2, we obtain

$$\alpha = \sum_{k=1}^{K} w_k = \frac{\sin v_1 \theta}{\sin \theta} + \frac{\sin v_2 \theta}{\sin \theta} \quad \text{and} \quad \beta = \frac{\sin^2 v_1 \theta}{\sin^2 \theta} + \frac{\sin^2 v_2 \theta}{\sin^2 \theta} \tag{6}$$

As discussed in Section 4, for a linear combination $\boldsymbol{y}$ to be a random variable following distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, given that $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ do, it must be true that $\alpha\boldsymbol{\mu} = \boldsymbol{\mu}$ and $\beta\boldsymbol{\Sigma} = \boldsymbol{\Sigma}$.

A common case is using unit Gaussian latents (as in e.g. the models Esser et al. (2024) and Rombach et al. (2022) used in this paper), i.e. where $\boldsymbol{\mu} = \mathbf{0}, \boldsymbol{\Sigma} = \boldsymbol{I}$. In this case it trivially follows that $\alpha\boldsymbol{\mu} = \boldsymbol{\mu}$ since $\alpha\mathbf{0} = \mathbf{0}$. We will now show that $\beta \approx 1$ in this special (i.e. unit Gaussian) case.

Lemma 3. *Let*

$$\beta = \frac{\sin^2 v\theta}{\sin^2 \theta} + \frac{\sin^2(1-v)\theta}{\sin^2 \theta},$$

where $\cos\theta = \frac{\langle \boldsymbol{x}_1, \boldsymbol{x}_2 \rangle}{\|\boldsymbol{x}_1\|\|\boldsymbol{x}_2\|}$ *and* $\boldsymbol{x}_1, \boldsymbol{x}_2 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. *Then* $\beta \approx 1$ *for large* D, $\forall v \in [0, 1]$.

Proof. Since $\boldsymbol{x}_1, \boldsymbol{x}_2 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, each component x_{1j} and x_{2j} for $j = 1, \ldots, D$ are independent standard normal random variables. The inner product $\langle \boldsymbol{x}_1, \boldsymbol{x}_2 \rangle$ is given by:

$$\langle \boldsymbol{x}_1, \boldsymbol{x}_2 \rangle = \sum_{j=1}^{D} x_{1j} x_{2j}.$$

The product $x_{1j}x_{2j}$ follows a distribution known as the standard normal product distribution. For large D, the sum of these products is approximately normal due to the Central Limit Theorem (CLT), with:

$$\langle \boldsymbol{x}_1, \boldsymbol{x}_2 \rangle \sim \mathcal{N}(0, D).$$

Next, consider the norms $\|\boldsymbol{x}_1\|$ and $\|\boldsymbol{x}_2\|$. Each $\|\boldsymbol{x}_i\|^2 = \sum_{j=1}^{D} x_{ij}^2$ is a chi-squared random variable with D degrees of freedom. For large D, by the central limit theorem, $\|\boldsymbol{x}_i\|^2 \sim \mathcal{N}(D, 2D)$, and therefore $\|\boldsymbol{x}_i\|$ is approximately $\sqrt{D}$.

Thus, for large D,

$$\cos(\theta) = \frac{\langle \boldsymbol{x}_1, \boldsymbol{x}_2 \rangle}{\|\boldsymbol{x}_1\|\|\boldsymbol{x}_2\|} \approx \frac{\mathcal{N}(0, D)}{\sqrt{D} \cdot \sqrt{D}} = \frac{\mathcal{N}(0, D)}{D} = \mathcal{N}\left(0, \frac{1}{D}\right).$$

Thus, $\theta \approx \pi/2$, which implies $\sin(\theta) \approx 1$. Therefore:

$$\beta = \frac{\sin^2(v\theta)}{\sin^2 \theta} + \frac{\sin^2((1-v)\theta)}{\sin^2 \theta} \approx \sin^2(v\theta) + \sin^2((1-v)\theta).$$

Using the identity $\sin^2(a) + \sin^2(b) = 1 - \cos^2(a - b)$,

$$\beta \approx 1 - \cos^2(v\theta - (1-v)\theta) = 1 - \cos^2(\theta)$$

Given $v \in [0, 1]$ and $\theta \approx \pi/2$ for large D, the argument of $\cos$ remains small, leading to $\cos(\cdot) \approx 0$. Hence,

$$\beta \approx 1.$$

Therefore, for large D, $\beta \approx 1$.

□

Empirical verification To confirm the effectiveness of this approximation for dimensional D typical of the popular generative models that use unit Gaussian latents, we estimate the confidence interval of β using 10k samples of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, respectively, where $\boldsymbol{x}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. For Stable Diffusion 3 (Esser et al., 2024), a flow matching model with $D = 147456$, the estimated 99% confidence interval of β is $[0.9934, 1.0067]$ for $v = 0.5$ where the error is largest. For Stable Diffusion 2.1 (Rombach et al., 2022), a diffusion model with $D = 36864$, the corresponding confidence interval of β is $[0.9868, 1.014]$.

C LINEAR COMBINATION WEIGHTS OF SUBSPACE PROJECTIONS

In Section 4 we introduced COG, which we use to transform latent variables arising from linear combinations such that they maintain the the latent distribution. This transformation depends on the weights $\boldsymbol{w}$ which specify the linear combination. We will now derive the linear combination weights for subspace projections.

Lemma 4. *Let $\boldsymbol{U} \in \mathbb{R}^{D\times K}$ be a semi-orthonormal matrix. For a given point $\boldsymbol{x} \in \mathbb{R}^D$, the subspace projection is $s(\boldsymbol{x}) = \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{x}$. The weights $\boldsymbol{w} \in \mathbb{R}^K$ such that $s(\boldsymbol{x})$ is a linear combination of $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K$ (columns of $\boldsymbol{A} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K] \in \mathbb{R}^{D\times K}$) can be expressed as $\boldsymbol{w} = \boldsymbol{A}_\dagger s(\boldsymbol{x})$, where $\boldsymbol{A}_\dagger$ is the Moore-Penrose inverse of $\boldsymbol{A}$.*

Proof. The subspace projection $s(\boldsymbol{x})$ of $\boldsymbol{x} \in \mathbb{R}^D$ is defined as:

$$s(\boldsymbol{x}) = \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{x}.$$

We aim to express $s(\boldsymbol{x})$ as a linear combination of the columns of $\boldsymbol{A} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K] \in \mathbb{R}^{D\times K}$. That is, we seek $\boldsymbol{w} \in \mathbb{R}^K$ such that:

$$s(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{w}.$$

By definition, the Moore-Penrose inverse $\boldsymbol{A}_\dagger$ of $\boldsymbol{A}$ satisfies the following properties:

1. $\boldsymbol{A}\boldsymbol{A}_\dagger\boldsymbol{A} = \boldsymbol{A}$
2. $\boldsymbol{A}_\dagger\boldsymbol{A}\boldsymbol{A}_\dagger = \boldsymbol{A}_\dagger$
3. $(\boldsymbol{A}\boldsymbol{A}_\dagger)^T = \boldsymbol{A}\boldsymbol{A}_\dagger$
4. $(\boldsymbol{A}_\dagger\boldsymbol{A})^T = \boldsymbol{A}_\dagger\boldsymbol{A}$

Since $s(\boldsymbol{x})$ is in the subspace spanned by the columns of $\boldsymbol{A}$, there exists a $\boldsymbol{w}$ such that:

$$s(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{w}.$$

Consider a $\boldsymbol{w}' \in \mathbb{R}^K$ constructed using the Moore-Penrose inverse $\boldsymbol{A}_\dagger$:

$$\boldsymbol{w}' = \boldsymbol{A}_\dagger s(\boldsymbol{x}).$$

We now verify that this $\boldsymbol{w}'$ satisfies the required equation. Substituting back

$$\boldsymbol{A}\boldsymbol{w}' = \boldsymbol{A}(\boldsymbol{A}_\dagger s(\boldsymbol{x}))$$

and using the property of the Moore-Penrose inverse $\boldsymbol{A}\boldsymbol{A}_\dagger\boldsymbol{A} = \boldsymbol{A}$, we get:

$$\boldsymbol{A}\boldsymbol{A}_\dagger s(\boldsymbol{x}) = s(\boldsymbol{x}).$$

Thus:

$$\boldsymbol{A}\boldsymbol{w}' = s(\boldsymbol{x}),$$

which shows that $\boldsymbol{w}' = \boldsymbol{A}_\dagger s(\boldsymbol{x})$ is indeed the correct expression for the weights.

From uniqueness of $\boldsymbol{w}$ for a given set of columns $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K$ (see Lemma 5), we have proven that the weights $\boldsymbol{w}$ for a given point in the subspace $s(\boldsymbol{x})$ are given by:

$$\boldsymbol{w} = \boldsymbol{A}_\dagger s(\boldsymbol{x}).$$

□

Lemma 5. *The weights $\boldsymbol{w} \in \mathbb{R}^K$ such that $s(\boldsymbol{x})$ is a linear combination of $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K$ (columns of $\boldsymbol{A} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K] \in \mathbb{R}^{D\times K}$) are unique.*

Proof. Suppose there exist two different weight vectors $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ such that both satisfy the equation:

$$s(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{w}_1 = \boldsymbol{A}\boldsymbol{w}_2.$$

Then, subtracting these two equations gives:

$$\boldsymbol{A}\boldsymbol{w}_1 - \boldsymbol{A}\boldsymbol{w}_2 = \boldsymbol{0}.$$

This simplifies to:

$$\boldsymbol{A}(\boldsymbol{w}_1 - \boldsymbol{w}_2) = \boldsymbol{0}.$$

Let $\boldsymbol{v} = \boldsymbol{w}_1 - \boldsymbol{w}_2$. Then:

$$\boldsymbol{A}\boldsymbol{v} = \boldsymbol{0}.$$

Since $\boldsymbol{v} \in \mathbb{R}^K$, this equation implies that $\boldsymbol{v}$ lies in the null space of $\boldsymbol{A}$. However, the assumption that $\boldsymbol{A}$ has full column rank (since $\boldsymbol{A}$ is used to represent a linear combination for $s(\boldsymbol{x})$) implies that $\boldsymbol{A}$ has no non-zero vector in its null space, i.e., $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{0}$ only when $\boldsymbol{v} = \boldsymbol{0}$.

Therefore:

$$\boldsymbol{v} = \boldsymbol{0} \implies \boldsymbol{w}_1 = \boldsymbol{w}_2.$$

This shows that the weights $\boldsymbol{w}$ are unique, and there cannot be two distinct sets of weights $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ that satisfy the equation $s(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{w}$.

Hence, we conclude that the weights $\boldsymbol{w}$ such that $s(\boldsymbol{x})$ is a linear combination of $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K$ are unique. □

D NORMALITY TEST COMPARISON

We compare popular alternatives for normality testing given known parameters which are applicable to very large sample sizes, including Kolmogorov-Smirnov (Kolmogorov, 1933), Shapiro-Wilk (Shapiro & Wilk, 1965), Chi-square (Pearson, 1900), and Cramér–von Mises (Cramér, 1928). We assess these methods on a suite of test cases of vectors which are extremely unlikely under the model's latent distribution and which we know through failed generation violates the characteristic assumptions of the diffusion model, as well as the positive test case where $\boldsymbol{x}_* \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. We report the $[0.1\%, 99.9\%]$-confidence interval of the p-value produced by each respective method on $1e^4$ random samples in the stochastic test cases. We also include the likelihood $\mathcal{N}(\boldsymbol{x}_*; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ and the likelihood of the norm statistic.

The results are reported in Table 2. We find that Kolmogorov-Smirnov and Cramér–von Mises both succeed in reporting a lower 99.9th percentile p-value for each tested failure case than the 0.1th percentile assigned to real samples. Kolmogorov-Smirnov did so with the largest gap, reporting extremely low p-values for all failure cases while reporting calibrated p-values for the positive case. Shapiro-Wilk, which in the literature often is regarded as one of the most powerful tests (Razali et al., 2011), did not work well in our context in contrast to the other normality testing methods, as it produced high p-values also for several of the failure cases. An explanation may be that in our context we have sample sizes of tens of thousands (same as the dimensionality of the latent D), while comparisons in the literature typically focuses on smaller sample sizes, such as up to hundreds or a few thousand (Razali et al., 2011). The Chi-square test is a test for discrete distributions. However, it is commonly used on binned data (Larsen & Marx, 2005) to approximate a continuous distribution as discrete. We do this, using histograms of 30 bins. This test successfully produced very low p-values for each failure case, but also did so for many valid samples, yielding a 0.1% of 0. The likelihood $\mathcal{N}(\boldsymbol{x}_*; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ assigns high likelihoods to vectors $\boldsymbol{x}_*$ near the mode irrespective of its characteristics. We note that it fails to distinguish to failure cases from real samples, and in some cases assigns much higher likelihood to the failure cases than real data. The insufficiency of the likelihood to describe the feasibility of samples is a phenomenon present in cases of high-dimensionality and low joint dependence between these dimensions; this is due to the quickly shrinking concentration of points near the centre (or close to $\boldsymbol{\mu}$, where the likelihood is highest) as the dimensionality is increased (Talagrand, 1995; Ash, 2012; Nalisnick et al., 2019). The norm statistic we note successfully assigns low likelihoods to some failure cases, where the norm is atypical, but fails to do so in most of the tested cases.

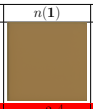
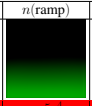
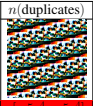
Test case:	$\boldsymbol{x}^{(i)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$	$\mathbf{0}$	$s\mathbf{1}$	$n(\mathbf{1})$	n(ramp)	$0.1\boldsymbol{x}^{(i)}$	$\boldsymbol{x}^{(i)} + 0.1$	n(duplicates)	n(5% rescaled)	n(1% rescaled)
Method/value										
$\log \mathcal{N}(\boldsymbol{x}^{(i)}; \mathbf{0}, \boldsymbol{I})$	$[-5e^4, -5e^4]$	$-3e^4$	$-5e^4$	$-3e^4$	$-5e^4$	$[-3e^3, -3e^3]$	$[-5e^4, -5e^4]$	$[-5e^4, -5e^4]$	$[-5e^4, -5e^4]$	$[-5e^4, -5e^4]$
$\log \chi^2(\|\|\boldsymbol{x}^{(i)}\|\|^2; D)$	$[-1e^1, -7]$	$-\infty$	-7	$-\infty$	-7	$[-7e^4, -7e^4]$	$[-2e^1, -7]$	$[-7, -7]$	$[-7, -7]$	$[-7, -7]$
Shapiro-Wilk	$[2e^{-4}, 1]$	1	1	1	$5e^{-72}$	$[8e^{-3}, 1]$	$[8e^{-3}, 1]$	$[1e^{-50}, 8e^{-19}]$	$[3e^{-105}, 1e^{-99}]$	$[2e^{-85}, 2e^{-69}]$
Chi-square	$[0, 1]$	0	0	0	0	$[0, 0]$	$[6e^{-149}, 9e^{-45}]$	$[0, 1e^{-216}]$	$[0, 0]$	$[0, 0]$
Cramer-von-Mises	$[2e^{-4}, 1]$	$5e^{-7}$	0.0	$5e^{-7}$	$1e^{-8}$	$[4e^{-7}, 4e^{-7}]$	$[2e^{-9}, 2e^{-8}]$	$[4e^{-12}, 7e^{-4}]$	$[2e^{-8}, 6e^{-8}]$	$[2e^{-11}, 5e^{-9}]$
Kolmogorov-Smirnov	$[5e^{-4}, 1]$	0	0	0	$2e^{-105}$	$[0, 0]$	$[9e^{-77}, 9e^{-37}]$	$[1e^{-129}, 7e^{-10}]$	$[8e^{-252}, 2e^{-117}]$	$[1e^{-31}, 2e^{-12}]$
		1	2	3	4	5	6	7	8	9

Table 2: **Normality testing of latents** Reported is the $[0.1\%, 99.9\%]$-confidence interval of the p-value produced by each respective normality testing method on $1e^4$ random samples in the stochastic test cases. We also report the log likelihood of the vector $\boldsymbol{x}^{(i)}$ under the latent distribution and the likelihood of the norm statistic. The diffusion model in Rombach et al. (2022) is used in this example. Green colour for the respective failure test case indicate the method assigning a lower p-value or likelihood to the failure case to at least 99.9% of the failure samples than the lowest 0.1% of real samples, to illustrate that it was successful in distinguishing the failure samples. Chi-square assigns low p-values to many real samples as well, which we indicate in red. The images show the generated image from a random failure sample of the test case. In (failure) Test 1 in column 1 the latent is $\mathbf{0}$, the mode of the latent distribution for this model $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$. In Test 2 it is the constant vector with the maximum likelihood norm of the norm sampling distribution. In Test 3 the constant vector is instead normalized to have its first two moments matching the latent's marginal distribution. $n(\cdot)$ indicate that the vector(s) of the test case is normalized to have zero mean and unit variance. In Test 4 a linspace vector $[-1, \dots, 1]^D$ have been similarly normalized. In Test 5 and 6 the normal samples (from the left most column) have been transformed with scaling and a bias, respectively. In Test 7 random selections of 1% of the dimensions $[1, \cdots, D]$ of the normal samples have been repeated (100 times) and subsequently normalized by $n(\cdot)$. In Test 8 and 9 a proportion of the dimensions of the normal samples have been multiplied by 5 (with indices selected randomly) and the whole vectors are subsequently normalized by $n(\cdot)$, where the proportion is 5% and 1%, respectively.

E INTERPOLATION AND CENTROID DETERMINATION SETUP DETAILS

Baselines For the application of interpolation, we compare to linear interpolation (LERP), spherical linear interpolation (SLERP), and Norm-Aware Optimization (NAO) (Samuel et al., 2023), a recent approach which considers the norm of the noise vectors. In contrast to the other approaches which only involve analytical expressions, NAO involves a numerical optimization scheme based on a discretion of a line integral. For the application of centroid determination we compare to NAO, the Euclidean centroid $\bar{\boldsymbol{x}} = \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{x}_k$, and two transformations of the Euclidean centroid; "standardized Euclidean", where $\bar{\boldsymbol{x}}$ is subsequently standardized to have mean zero and unit variance, and "mode norm Euclidean", where $\bar{\boldsymbol{x}}$ is rescaled to have the norm equal to the (square root of the) mode of $\chi^2(D)$, the chi-squared distribution with D degrees of freedom, which is the maximum likelihood norm given that $\boldsymbol{x}$ has been generated from a unit Gaussian with D dimensions.

Evaluation sets We closely follow the evaluation protocol in Samuel et al. (2023), where we base the experiments on Stable Diffusion 2 (Rombach et al., 2022) and inversions of random images from ImageNet1k (Deng et al., 2009). We (uniformly) randomly select 50 classes, each from which we randomly select 50 unique images, and find their corresponding latents through DDIM inversion (Song et al., 2020a) using the class name as prompt. We note that the DDIM inversion can be sensitive to the number of steps. Therefore, we made sure to use a sufficient number of steps for the inversion (we used 400 steps), which we then matched for the generation; see Figure 15 for an illustration of the importance of using a sufficient number of steps. We used the guidance scale 1.0 (i.e. no guidance) for the inversion, which was then matched during generation. Note that using no guidance is important both for accurate inversion as well as to not introduce a factor (the dynamics of prompt guidance) which would be specific to the Stable Diffusion 2.1 model.

For the interpolation setup we randomly (without replacement) pair the 50 images per class into 25 pairs, forming 1250 image pairs in total. In between the ends of each respective pair, each method then is to produce three interpolation points (and images). For the NAO method, which needs additional

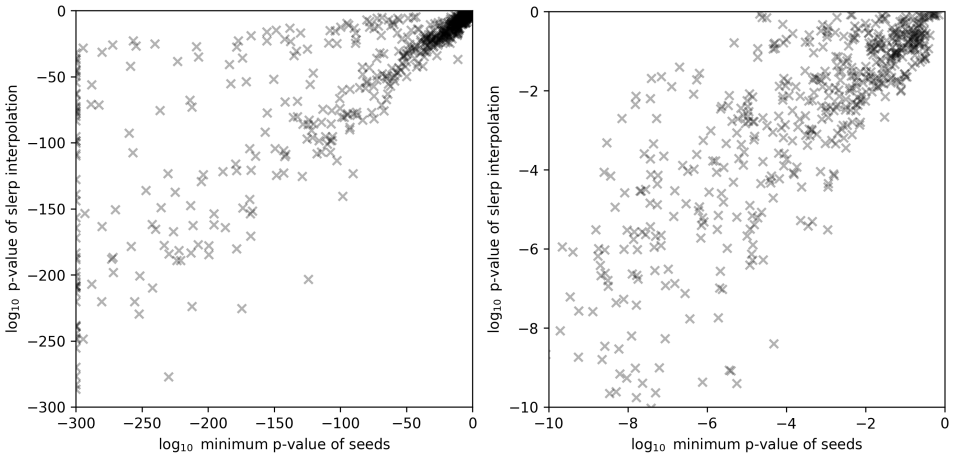


Figure 8: **Low p-values are inherited by interpolants** The x-axis show the minimum Kolmogorov-Smirnov p-value of the two latents for each corresponding spherical interpolation in Figure 5, and the y-axis show the p-value of the resulting interpolant. We note that when any of the two seed latents have a small p-value this largely tend to be inherited by the interpolant. The Pearson correlation coefficient is 0.79 for the indicator of acceptance (defined as 1 if the p-value is greater than $1e^{-3}$ and 0 otherwise) and 0.66 for the $\log_{10}$ p-values.

interpolation points to approximate the line integral, we used 11 interpolation points and selected three points from these at uniform (index) distance, similar to in Samuel et al. (2023).

For the centroid determination setup we for each class form 10 3-groups, 10 5-groups, 4 10-groups and 1 25-group, sampled without replacement per group[1]; i.e. each method is to form 25 centroids per class total, for an overall total of 1250 centroids per method. Similarly to the interpolation setup, for NAO we used 11 interpolations points per path, which for their centroid determination method entails K paths per centroid.

Evaluation We, as in Samuel et al. (2023), assess the methods quantitatively based on visual quality and preservation of semantics using FID scores and class prediction accuracy, respectively.

The FID scores are computed using the pytorch-fid library (Seitzer, 2020), using all evaluation images produced per method for the interpolation and centroid determination respectively, to maximize the FID score estimation accuracy.

For the classification accuracy, we used a pre-trained classifier, the MaxViT image classification model (Tu et al., 2022) as in Samuel et al. (2023), which achieves a top-1 of 88.53% and 98.64% top-5 accuracy on the test-set of ImageNet.

See results in Section G.

F Additional qualitative results

See Figure 9 for a demonstration of subspaces of latents *without* the COG scheme introduced in Equation 3, using a diffusion model (Rombach et al., 2022) and a flow matching model (Esser et al., 2024), respectively. The setup is identical (with the same original latents) as in Figure 12 and Figure 2, respectively, except without applying the proposed (COG) transformation.

See Figure 10 for additional slices of the sports car subspace shown in Figure 2.

See Figure 12 and Figure 13 for Stable Diffusion 2.1 (SD2.1) versions of the Stable Diffusion 3 (SD3) examples in Figure 2 and Figure 6, with an otherwise identical setup including the prompts. Just like

[1]But with replacement across groups, i.e. the groups are sampled independently from the same collection of images.

in the SD3 examples COG defines working subspaces. However, as expected since SD2.1 is an older model than SD3, the visual quality of the generations is better using SD3.

See Figure 11 for an interpolation example.

In the examples above the SD2.1 and SD3 models are provided with a text prompt during the generation. See Figure 14 for an example where the original latents ($\{\boldsymbol{x}_i\}$) were obtained using DDIM inversion (from images) without a prompt (and guidance scale 1.0, i.e. no guidance), allowing generation without a prompt. This allows us to also interpolate without conditioning on a prompt. We note that, as expected without a prompt, the intermediates are not necessarily related to the end points (the original images) but still realistic images are obtained as expected (except for using linear interpolation, see discussion in Section 4) and the interpolations yield smooth gradual changes.

Figure 9: **Without COG transformation**. The setup here is exactly the same as in Figure 12 and Figure 2, respectively, except without the proposed (COG) transformation (see Equation 3). The prompt used is *"A high-quality photo of a parked, colored sports car taken with a DLSR camera with a 45.7MP sensor. The entire sports car is visible in the centre of the image. The background is simple and uncluttered to keep the focus on the sports car, with natural lighting enhancing its features."*. We note that the diffusion model does not produce images of a car without the transformation, and neither model produce anything else than visually the same image for all coordinates.

G Additional quantitative results and analysis

In Table 1 we show that our method outperforms the baselines in terms of FID scores and accuracy, as calculated using a pre-trained classifier following the evaluation methodology of Samuel et al. (2023). Surprisingly, NAO did not perform as well as spherical interpolation and several other baselines, despite that we used their implementation and it was outperforming these methods in Samuel et al. (2023). We note that one discrepancy is that we report FID scores using the standard number of features (2048), while in their paper they are using fewer (64), which in Seitzer (2020) is not recommended since it does not necessarily correlate with visual quality. In Table 3 we report the results all settings of features. We note that, in our setup, the baselines performs substantially better than reported in their setup — including NAO in terms of FID scores using the 64 feature setting (1.30 vs 6.78), and class accuracy during interpolation; NAO got 62% in our and 52% in their setup. However, the accuracy they obtained for NAO in their setup was higher than in our setup; 67% vs 44%, which we study below.

The number of DDIM inversion steps used for the ImageNet images is not reported in the NAO setup (Samuel et al., 2023), but if we use *1 step* instead of 400 (400 is what we use in all our experiments) the class preserving accuracy of NAO for centroid determination, shown in Table 4, resemble theirs more, yielding 58%. As we illustrate in Figure 5 and Figure 15, a sufficient number of inversion steps is critical to acquire latents which not only reconstructs the original image but which can be successfully interpolated. We hypothesise that a much lower setting of DDIM inversions steps were used in their setup than in ours, which would be consistent with their norm-correcting

Figure 10: **Additional slices of a sports car subspace**. The latents $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_5$ (corresponding to images) are converted into basis vectors and used to define a 5-dimensional subspace. The grids show generations from uniform grid points in the subspace coordinate system, where the left and right grids are for the dimensions $\{1, 2\}$ and $\{3, 4\}$, respectively, centered around the coordinate for $\boldsymbol{x}_1$. Each coordinate in the subspace correspond to a linear combination of the basis vectors. The flow matching model Stable Diffusion 3 (Esser et al., 2024) is used in this example. See Figure 2 for two other slices of the space.

Figure 11: **Interpolation**. Shown are generations from equidistant interpolations of latents $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ (in $v \in [0, 1]$) using the respective method. The diffusion model Stable Diffusion 2.1 (Rombach et al., 2022) is used in this example.

optimization method having a large effect, making otherwise blurry/superimposed images intelligible for the classification model.

H Additional ImageNet interpolations and centroids

Figure 12: **Low-dimensional subspaces**. The latents $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_5$ (corresponding to images) are converted into basis vectors and used to define a 5-dimensional subspace. The grids show generations from uniform grid points in the subspace coordinate system, where the left and right grids are for the dimensions $\{1, 3\}$ and $\{2, 4\}$, respectively, centered around the coordinate for $\boldsymbol{x}_1$. Each coordinate in the subspace correspond to a linear combination of the basis vectors. The diffusion model Stable Diffusion 2.1 (Rombach et al., 2022) is used in this example.

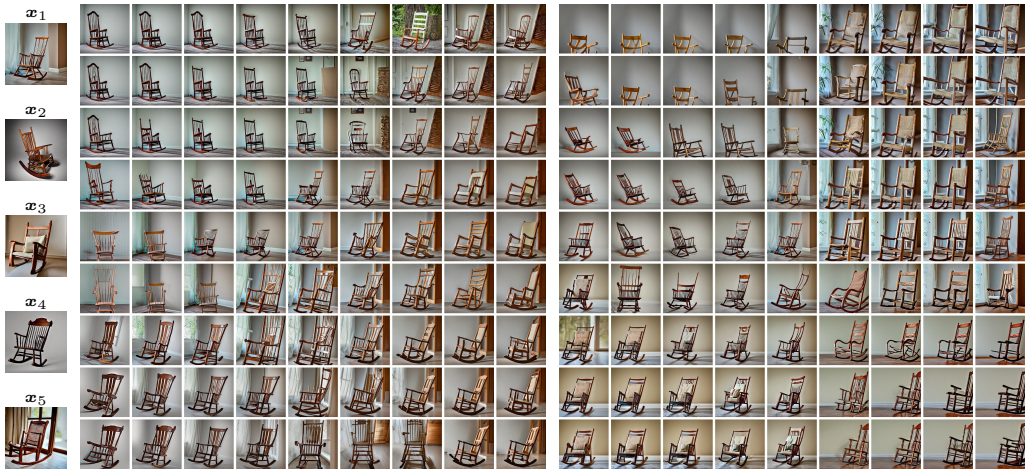


Figure 13: **Low-dimensional subspaces**. The latents $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_5$ (corresponding to images) are converted into basis vectors and used to define a 5-dimensional subspace. The grids show generations from uniform grid points in the subspace coordinate system, where the left and right grids are for the dimensions $\{1, 3\}$ and $\{2, 4\}$, respectively, centered around the coordinate for $\boldsymbol{x}_1$. Each coordinate in the subspace correspond to a linear combination of the basis vectors. The diffusion model Stable Diffusion 2.1 (Rombach et al., 2022) is used in this example.

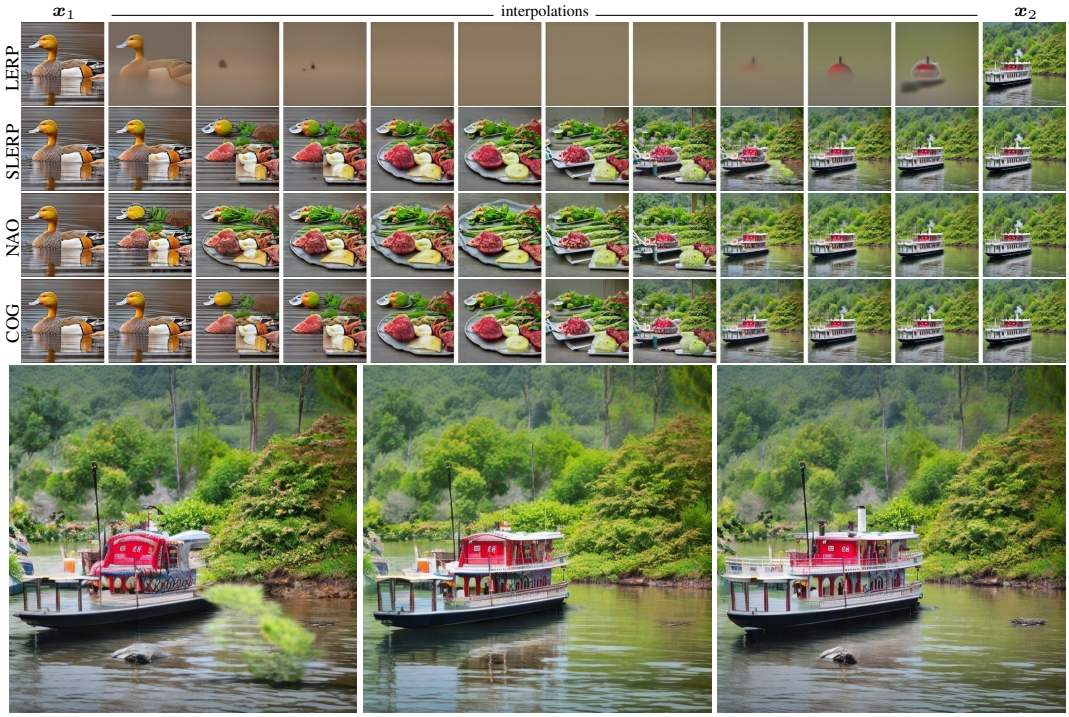


Figure 14: **Interpolation during unconditional generation**. Shown are generations from equidistant interpolations of latents $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ using the respective method. The latents $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ were obtained from DDIM inversion (Song et al., 2020a) with an empty prompt, and all generations in this example are then carried out with an empty prompt. The larger images show interpolation index 8 for SLERP, NAO and COG, respectively. The diffusion model Stable Diffusion 2.1 (Rombach et al., 2022) is used in this example.

Interpolation						
Method	**Accuracy**	**FID 64**	**FID 192**	**FID 768**	**FID 2048**	**Time**
LERP	3.92%	63.4	278	2.18	199	$6e^{-3}$s
SLERP	64.6%	2.28	4.95	0.173	42.6	$9e^{-3}$s
NAO	62.1%	1.30	4.11	0.195	46.0	30s
COG (ours)	67.4%	1.72	3.62	0.156	38.9	$6e^{-3}$s
Centroid determination						
Method	**Accuracy**	**FID 64**	**FID 192**	**FID 768**	**FID 2048**	**Time**
Euclidean	0.286%	67.7	317	3.68	310	$4e^{-4}$s
Standardized Euclidean	44.6%	5.92	21.0	0.423	88.8	$1e^{-3}$s
Mode norm Euclidean	44.6%	6.91	21.6	0.455	88.4	$1e^{-3}$s
NAO	44.0%	4.16	15.6	0.466	93.0	90s
COG (ours)	46.3%	9.38	25.5	0.455	87.7	$6e^{-4}$s

Table 3: Full table with class accuracy and FID scores using each setting in Seitzer (2020). Note that FID 64, 192, and 768 is not recommended by Seitzer (2020), as it does not necessarily correlate with visual quality. FID 2048 is based on the final features of InceptionV3 (as in Heusel et al. (2017)), while FID 64, 192 and 768 are based on earlier layers; FID 64 being the first layer.

Centroid determination of invalid latents						
Method	**Accuracy**	**FID 64**	**FID 192**	**FID 768**	**FID 2048**	**Time**
Euclidean	42.3%	25.7	59.3	1.23	170	$4e^{-4}$s
Standardized Euclidean	46.6%	3.35	16.6	1.23	175	$1e^{-3}$s
Mode norm Euclidean	50.6%	2.08	8.74	1.19	173	$1e^{-3}$s
NAO	57.7%	3.77	13.3	1.05	150	90s
COG (ours)	48.6%	2.96	12.5	1.22	171	$6e^{-4}$s

Table 4: Centroid determination results if we would use a *single step* for the DDIM inversion (not recommended). This results in latents which do *not* follow the correct distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and which, despite reconstructing the original image exactly (if generating also using a single step), cannot be used successfully for interpolation; see Figure 5 and Figure 15 for examples of the result of interpolation on inversions obtained through varying budgets of inversion steps. As illustrated in the figure, interpolations of latents obtained through a single step of DDIM inversion merely results in the two images being superimposed. Note that the class accuracy is much higher for all methods on such superimposed images, compared to using DDIM inversion settings which allow for realistic interpolations (see Table 3). Moreover, the FID scores set to use fewer features are *lower* for these superimposed images, while with recommended setting for FID scores (with 2048 features) they are higher (i.e. worse), as one would expect by visual inspection. The NAO method, initialized by the latent yielding the superimposed image, then optimises the latent yielding a norm close to norms of typical latents, which as indicated in the table leads to better "class preservation accuracy". However, the images produced with this setting are clearly unusable as they do not look like real images.

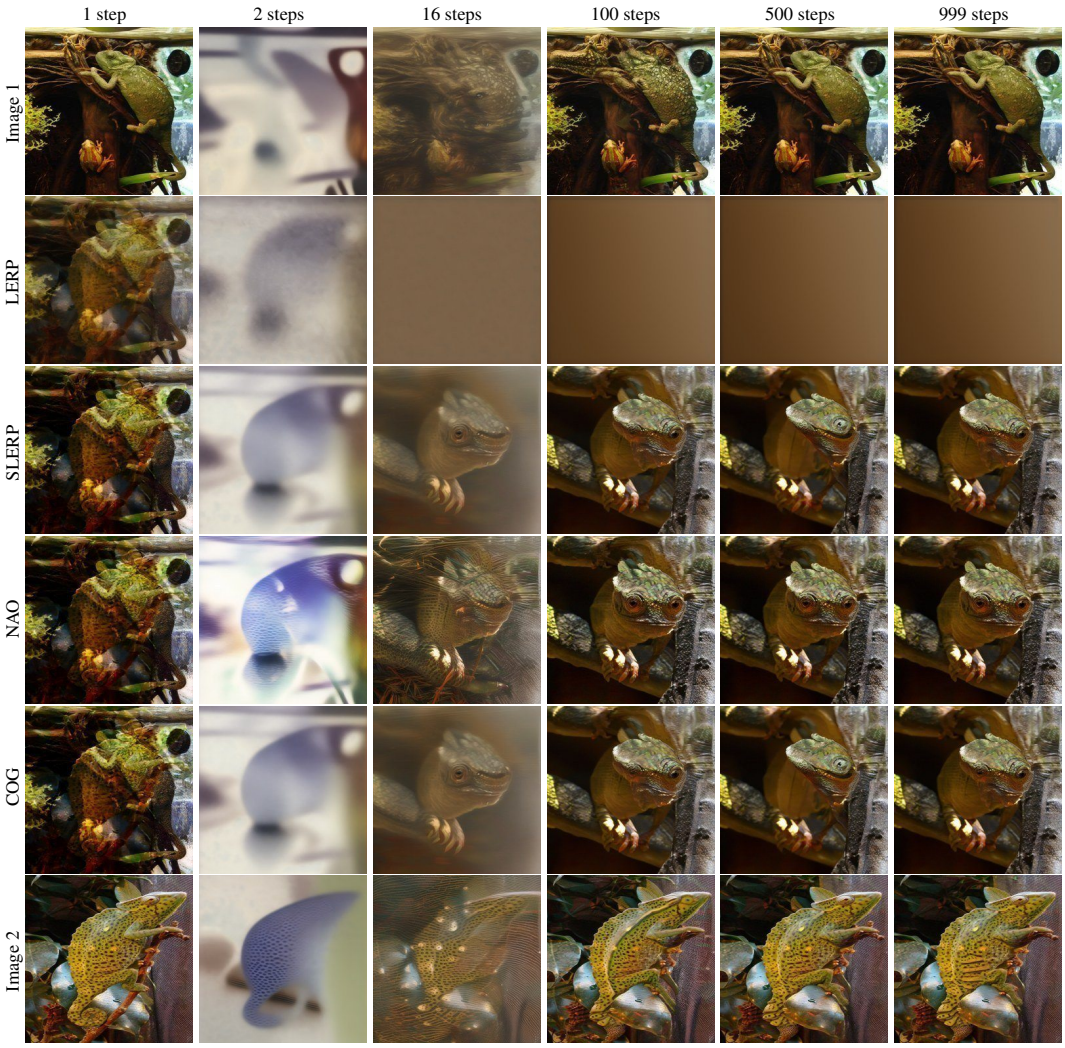


Figure 15: **Accurate DDIM inversions are critical for interpolation**. Shown is the interpolation (center) of Image 1 and Image 2 using each respective method, after a varying number of budgets of DDIM inversion steps (Song et al., 2020a). For each budget setting, the inversion was run from the beginning. We note that although a single step of DDIM inversion yields latents which perfectly reconstructs the original image such latents do not lead to realistic images, but merely visual "superpositions" of the images being interpolated.

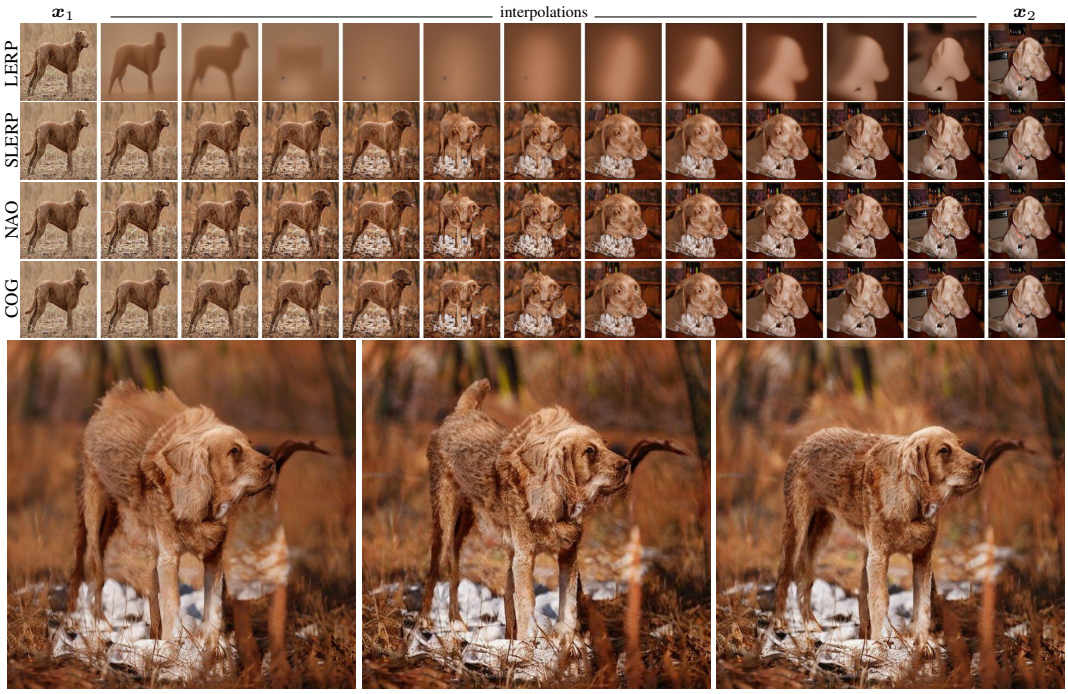


Figure 16: **Interpolation**. Shown are generations from equidistant interpolations of latents $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ using the respective method. The latents $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ were obtained from DDIM inversion (Song et al., 2020a) of two random image examples from one of the randomly selected ImageNet classes ("Chesapeake Bay retriever") described in Section 5. The larger images show interpolation index 6 for SLERP, NAO and COG, respectively. The diffusion model Stable Diffusion 2.1 (Rombach et al., 2022) is used in this example.

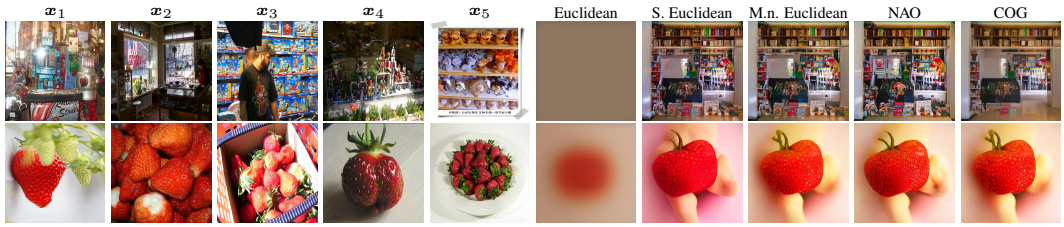


Figure 17: **ImageNet centroid determination**. The centroid of the latents $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, $\boldsymbol{x}_3$ as determined using the different methods, with the result shown in the respective (right-most) plot. The diffusion model Stable Diffusion 2.1 (Rombach et al., 2022) is used in this example.

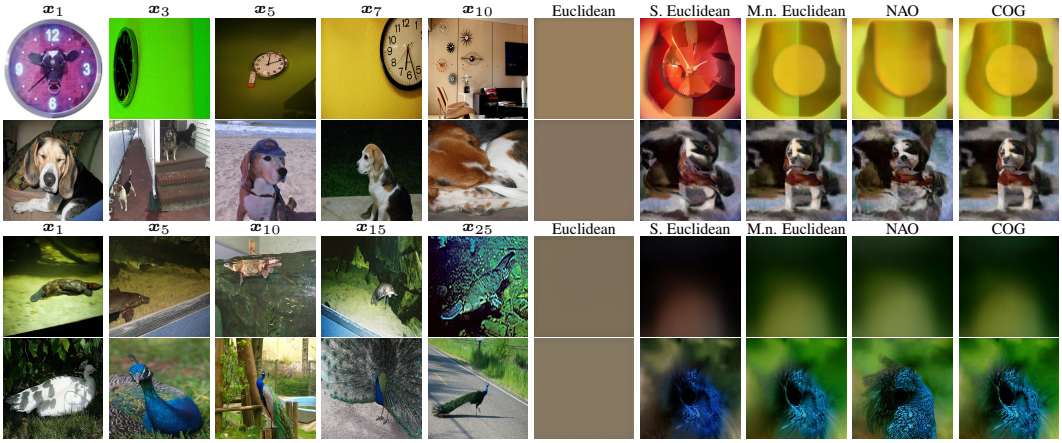


Figure 18: **Centroid determination of many latents**. The centroid of the latents as determined using the different methods, with the result shown in the respective (right-most) plot. Ten and 25 latents are used in the first two and second two examples (rows), respectively, and the plots show a subset of these in the left-most plots. We noted during centroid determination of many latents (such as 10 and 25) that the centroids were often unsatisfactory using all methods; blurry, distorted etc. For applications needing to find meaningful centroids of a large number of latents, future work is needed. The diffusion model Stable Diffusion 2.1 (Rombach et al., 2022) is used in this example.