# THE GEOMETRY OF DEEP GENERATIVE IMAGE MODELS AND ITS APPLICATIONS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Generative adversarial networks have emerged as a powerful unsupervised method to model the statistical patterns of real-world data sets, such as natural images. These networks are trained to map random inputs in their latent space to new samples representative of the learned data. However, the structure of the latent space is hard to intuit due to its high dimensionality and the non-linearity of the generator, which limits the usefulness of the models. Understanding the latent space requires a way to identify input codes for existing real-world images (inversion), and a way to identify directions with known image transformations (interpretability). Here, we use a geometric framework to address both issues simultaneously. We develop an architecture-agnostic method to compute the Riemannian metric of the image manifold created by GANs. The eigen-decomposition of the metric isolates axes that account for different levels of image variability. An empirical analysis of several pretrained GANs shows that image variation around each position is concentrated along surprisingly few major axes (the space is highly anisotropic) and the directions that create this large variation are similar at different positions in the space (the space is homogeneous). We show that many of the top eigenvectors correspond to interpretable transforms in the image space, with a substantial part of eigenspace corresponding to minor transforms which could be compressed out. This geometric understanding unifies previous results of GAN inversion and interpretation. We show that the use of this metric allows for more efficient optimization in the latent space (e.g. GAN inversion) and facilitates unsupervised discovery of interpretable axes. Our results show that defining the geometry of the GAN image manifold can serve as a general framework for understanding GANs.

## 1 BACKGROUND

Generative adversarial networks (GANs) learn patterns that characterize complex datasets, and subsequently generate new samples representative of that set. In recent years, there has been tremendous success in training GANs to generate high-resolution and photorealistic images (Karras et al., 2017; Brock et al., 2018; Donahue & Simonyan, 2019; Karras et al., 2020). Well-trained GANs show smooth transitions between image outputs when interpolating in their latent input space, which makes them useful in applications such as high-level image editing (changing attributes of faces), object segmentation, and image generation for art and neuroscience (Zhu et al., 2016; Shen et al., 2020; Pividori et al., 2019; Elgammal et al., 2017; Ponce et al., 2019). However, there is no systematic approach for understanding the latent space of any given GAN or its relationship to the manifold of natural images.

Because a generator provides a smooth map onto image space, one relevant conceptual model for GAN latent space is a Riemannian manifold. To define the structure of this manifold, we have to ask questions such as: are images homogeneously distributed on a sphere?(White, 2016) What is the structure of its tangent space — do all directions induce the same amount of variance in image transformation? Here we develop a method to compute the metric of this manifold and investigate its geometry directly, and then use this knowledge to navigate the space and improve several applications.

To define a Riemannian geometry, we need to have a smooth map and a notion of distance on it, defined by the metric tensor. For image applications, the relevant notion of distance is in image space rather than code space. Thus, we can pull back the distance function on the image space onto the latent space. Differentiating this distance function on latent space, we will get a differential geometric structure (Riemannian metric) on the image manifold. Further, by computing the Riemannian metric at different points (i.e. around different latent codes), we can estimate the anisotropy and homogeneity of this manifold.

The paper is organized as follows: first, we review the previous work using tools from Riemannian geometry to analyze generative models in section 2. Using this geometric framework, we introduce an efficient way to compute the metric tensor $H$ on the image manifold in section 3, and empirically investigate the properties of $H$ in various GANs in section 4. We explain the properties of this metric in terms of network architecture and training in section 5. We show that this understanding provides a unified principle behind previous methods for GAN inversion and for interpretable axes discovery in the latent space. Finally, we demonstrate other applications that this geometric information could facilitate, e.g. gradient-free searching in the GAN image manifold in section 6.

## 2 Related Work

**Geometry of Deep Generative Model** Concepts in Riemannian geometry have been recently applied to illuminate the structure of latent space of generative models (i.e. GANs and variational autoencoders, VAEs). Shao et al. (2018) designed algorithms to compute the geodesic path, parallel transport of vectors and geodesic shooting in the latent space; they used finite difference together with a pretrained encoder to circumvent the Jacobian computation of the generator. While promising, this method did not provide information of the metric directly and could not be applied to GANs without encoders. Arvanitidis et al. (2017) focused on the geometry of VAEs, deriving a formula for the metric tensor in order to solve the geodesic in the latent space; this worked well with shallow convolutional VAEs and low-resolution images (28 x 28 pixels). Chen et al. (2018) computed the geodesic through minimization, applying their method on shallow VAEs trained on MNIST images and a low-dimensional robotics dataset. In the above, the featured methods could only be applied to neural networks without ReLU activation. Here, we apply the geometric analysis to modern large-scale GANs (e.g. BigGAN, StyleGAN2) and we show it is compatible with any activation function. Finally, all previous works assumed a L2 distance in image space, and here we extend this approach to any differentiable distance metric.

## 3 Methods

**Formulation** A generative network, denoted by $G$, is a mapping from latent code $\boldsymbol{z}$ to image $I$, $G : \mathbb{R}^n \to \mathcal{I} = \mathbb{R}^{H\times W\times 3}, \boldsymbol{z} \mapsto I$. Borrowing the language of Riemannian geometry, $G(\boldsymbol{z})$ parameterizes a submanifold in the image space with $\boldsymbol{z} \in \mathbb{R}^n$. Note for applications in image domain, we care about distance in the image space. Thus, given a distance function in image space $D : \mathcal{I} \times \mathcal{I} \to \mathbb{R}_+, (I_1, I_2) \mapsto L$, we can define the distance between two codes as the distance between the images they generate, i.e. pullback the distance function to latent space through $G$. $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+, d(\boldsymbol{z}_1, \boldsymbol{z}_2) := D(G(\boldsymbol{z}_1), G(\boldsymbol{z}_2))$.

The Hessian matrix (second order partial derivative) of squared distance function $d^2$ can be seen as the metric tensor of the image manifold (Palais, 1957). The intuition behind this is as follows. Consider the squared distance to a fixed reference vector $\boldsymbol{z}_0$ as a function of $\boldsymbol{z}$, $f_{\boldsymbol{z}_0}(\boldsymbol{z}) = d^2(\boldsymbol{z}_0, \boldsymbol{z})$. Obviously, $\boldsymbol{z} = \boldsymbol{z}_0$ is a local minimum of $f_{\boldsymbol{z}_0}(\boldsymbol{z})$, thus $f_{\boldsymbol{z}_0}(\boldsymbol{z})$ can be locally approximated by a positive semi-definite quadratic form $H(\boldsymbol{z}_0)$ as in Eq.1. This matrix induces an inner product and defines a vector norm, $\|\boldsymbol{v}\|_H^2 = \boldsymbol{v}^T H(\boldsymbol{z}_0)\boldsymbol{v}$. This squared vector norm approximates the squared image distance, $d^2(\boldsymbol{z}_0, \boldsymbol{z}_0 + \delta\boldsymbol{z}) \approx \|\delta\boldsymbol{z}\|_H^2 = \delta_{\boldsymbol{z}}^T H(\boldsymbol{z}_0)\delta\boldsymbol{z}$. Thus, this matrix encodes the local distance information on the image manifold up to second order approximation. This is the intuition behind Riemannian metric. In this article, the terms "metric tensor" and "Hessian matrix" are used interchangeably. We will call $\alpha_H(\boldsymbol{v}) = \boldsymbol{v}^T H\boldsymbol{v}/\boldsymbol{v}^T\boldsymbol{v}$ the approximate speed of image change along

$\boldsymbol{v}$ as measured by metric $H$.

$$d^2(\boldsymbol{z}_0, \boldsymbol{z}) \approx d^2(\boldsymbol{z}_0, \boldsymbol{z}_0) + \frac{\partial d^2(\boldsymbol{z}_0, \boldsymbol{z})}{\partial \boldsymbol{z}}|_{\boldsymbol{z}_0}\delta\boldsymbol{z} + \frac{\partial^2 d^2(\boldsymbol{z}_0, \boldsymbol{z})}{\partial \boldsymbol{z}^2}|_{\boldsymbol{z}_0}\delta\boldsymbol{z}^2 = \frac{\partial^2 d^2(\boldsymbol{z}_0, \boldsymbol{z})}{\partial \boldsymbol{z}^2}|_{\boldsymbol{z}_0}\delta\boldsymbol{z}^2 \quad (1)$$

$$H(\boldsymbol{z}_0) := \frac{\partial^2 d^2(\boldsymbol{z}_0, \boldsymbol{z})}{\partial \boldsymbol{z}^2}|_{\boldsymbol{z}_0} \quad (2)$$

**Numerical Method** As defined above, the metric tensor $H$ can be directly computed by doubly differentiating the squared distance function $d^2$. In this paper we use a CNN-based distance metric called Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al., 2018) as our $d^2$, which is easily differentiable. Thus we can compute the Hessian by building a computational graph towards the gradient $g = \partial_{\boldsymbol{z}} d^2$ and then computing the gradient towards each element in $g$. This method computes $H$ column by column, thus its time complexity is proportional to dimension of latent space $n$ and backpropagation time.

For situations when computing the Hessian through brute-force backpropagation is too slow (e.g. FC6GAN, StyleGAN2), we developed an additional method to compute the major eigen-dimensions of the Hessian more efficiently. The rationale is, we are more interested in the top eigen pairs of the Hessian matrix than in the Hessian itself. As we will later discover, the spectra of these matrices have a fast decay, thus these Hessian matrices require far less than $n$ eigenvectors to approximate, cf. Sec 4. Specifically, we use forward or reverse mode differentiation to define Hessian-vector product (HVP) operator as Eq.7. Then we can use iterative algorithms to solve the large eigenvalues and eigenvectors for the HVP operator, and these eigen pairs can in turn approximate the Hessian matrix. For details and efficiency comparison, see Appendix A.2.

Note a similar computational method has been employed to understand the optimization landscape of deep neural networks recently (Ghorbani et al., 2019), although it has not been applied to understand the geometry of latent space of GANs before.

**Connection to Jacobian** This formulation and computation of the Riemannian metric can be generalized to any representational space beyond that of images. Consider a mapping $\phi(\boldsymbol{z}) : \mathbb{R}^n \to \mathbb{R}^M$, which could be the feature map of a middle layer in the GAN, or a CNN processing the generated image. We can pull back the squared L2 distance and metric from $\mathbb{R}^M$, $d^2_\phi(\boldsymbol{z}_1, \boldsymbol{z}_2) = \frac{1}{2}\|\phi(\boldsymbol{z}_1) - \phi(\boldsymbol{z}_2)\|_2^2$, and define a manifold. The metric tensor $H_\phi$ of this manifold can be derived as Hessian of $d^2_\phi$. Note, there is a simple relationship between the Hessian of $d^2_\phi$, $H_\phi$ and the Jacobian of $\phi$, $J_\phi$ (Eq. 3). Through this we know the eigen spectrum of the Hessian matrix $H_\phi$ is the square of singular value spectrum of Jacobian $J_\phi$, and the eigenvectors of $H_\phi$ is the same as the right singular vectors of $J_\phi$. This allows us to examine the geometry of the representation through out the GAN, and analyze how the geometry in the image space builds up.

$$H_\phi(\boldsymbol{z}_0) = \frac{\partial^2}{\partial \boldsymbol{z}^2}\frac{1}{2}\|\phi(\boldsymbol{z}_0) - \phi(\boldsymbol{z})\|_2^2|_{\boldsymbol{z}_0} = J_\phi(\boldsymbol{z}_0)^T J_\phi(\boldsymbol{z}_0) \quad (3)$$

$$\boldsymbol{v}^T H_\phi(\boldsymbol{z}_0)\boldsymbol{v} = \|J_\phi(\boldsymbol{z}_0)\boldsymbol{v}\|^2, \quad J_\phi(\boldsymbol{z}_0) = \partial_{\boldsymbol{z}}\phi(\boldsymbol{z})|_{\boldsymbol{z}_0} \quad (4)$$

In this work, we use LPIPS, which defines image distance based on the squared L2 distance of the first few layers of a pretrained CNN. If we concatenate the activations and denote this representational map by $\varphi(I) : \mathcal{I} \to \mathbb{R}^F$, then the metric tensor of the image manifold can be derived from the Jacobian of generator compositing representation map $\varphi$, $H(\boldsymbol{z}) = J^T_{\varphi\circ G} J_{\varphi\circ G}$, $J_{\varphi\circ G} = \partial_{\boldsymbol{z}}\varphi(G(\boldsymbol{z}))$. This connection is crucial for understanding how geometry depends on the network architecture.

## 4 Empirical Observations

Using the above method, we analyzed the geometry of the latent space of the following GANs: DCGAN (Radford et al., 2015), DeePSiM/FC6GAN (Dosovitskiy & Brox, 2016), BigGAN (Brock et al., 2018), BigBiGAN (Donahue & Simonyan, 2019), Progressive Growing of GANs (PGGAN) (Karras et al., 2017), StyleGAN 1 and 2 (Karras et al., 2019; 2020). (models specificities reviewed in Sec. A.3.) These GANs are progressively deeper and more complex, and some employ a style-based architecture instead of a more conventional DCGAN architecture (e.g. StyleGAN1,2). This diverse set of models allowed us to test the broad applicability of this new approach. In the following sections, "top" and "bottom" eigenvectors refer to the eigenvectors with large and small eigenvalues.

**Top Eigenvectors Capture Perceptually Relevant Changes** In differential geometry, a metric tensor $H$ captures an infinitesimal notion of distance. To determine whether this quantity represents perceptually meaningful image changes, we randomly picked a latent code $\boldsymbol{z}_0$, computed the metric tensor $H(\boldsymbol{z}_0)$ and its eigendecomposition $H(\boldsymbol{z}_0) = \sum_i \lambda_i \boldsymbol{v}_i \boldsymbol{v}_i^T$. Then we explored linearly in the latent space[1] along the eigenvectors $G(\boldsymbol{z}_0 + \mu_i \boldsymbol{v}_i)$. We found that images changed much faster when moving along top than along bottom eigenvectors, both per visual inspection and LPIPS (Fig 1). More intriguingly, eigenvectors at different ranks encoded qualitatively different types of changes; for example, in BigGAN noise space, the top eigenvectors encoded head direction, proximity and size; while lower eigenvectors encoded background changes, shading or more subtle pixel-wise changes. Moreover, PGGAN and StyleGANs trained on the face dataset have top eigenvectors that represent similar interpretable transforms of faces, such as head direction, sex or age (Fig. 8).

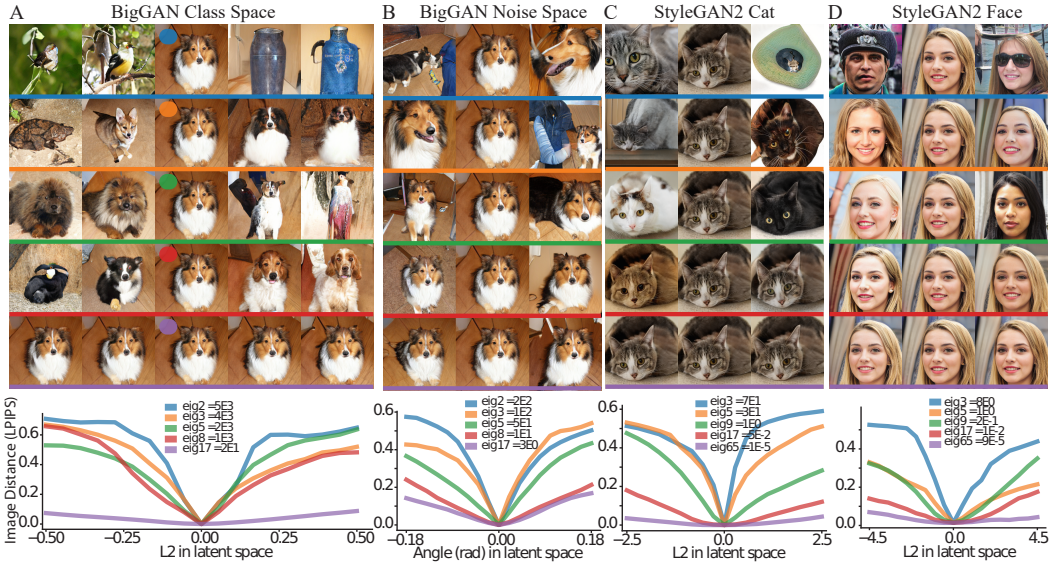


Figure 1: **Images change at different rates along top vs bottom eigenvectors**. Each panel (A-D) shows perturbations around a randomly chosen reference image (center column); each row shows perturbations introduced by moving along each of five eigenvectors; each contiguous column is separated by the same distance in latent space. Eigenvectors are shown in descending order of their eigenvalues. Line plots under each montage show the LPIPS image distance to the reference image as a function of positively and negatively perturbed distance along each eigenvector (x-axis). The rate of image change differed across eigenvectors; top vs bottom eigenvectors encoded changes such as object class, head direction, pose, color, shading or other subtle details (e.g. fur variations in panel A, bottom). A. BigGAN Class subspace; B. BigGAN Noise subspace; C. StyleGAN2 trained on cat images; D. StyleGAN2 trained on faces images. Eigenvector rank and associated eigenvalue labeled in the legend.

**Spectrum Structure of GANs.** To explore how eigenvalues were distributed, for each GAN, we randomly sampled 100-1000 $\boldsymbol{z}$ in the latent space, used backpropagation to compute $H(\boldsymbol{z})$ and then performed the eigendecomposition. In Fig. 2, we plotted the mean and 90% confidence interval of the spectra and found that they spanned 5-10 orders of magnitude, with fast decays; each spectrum was dominated by a few eigenvectors with large eigenvalues. In other words, only a small fraction of dimensions were responsible for major image changes (Table 2), while most dimensions introduced nuanced changes (e.g. shading, background) — GAN latent spaces were highly anisotropic.

We found this anisotropy in every GAN we tested, which raises the question of why it had not been reported previously. One possibility is that the statistical properties of high dimensionality create an illusion of isotropy. When traveling along a random direction $\boldsymbol{v}$ in latent space, the approximate rate of image change $\alpha_H(\boldsymbol{v}) = \boldsymbol{v}^T H \boldsymbol{v} / \boldsymbol{v}^T \boldsymbol{v}$ is a weighted average of all eigenvalues as in Eq. 9. In Sec A.5, we show analytically that the variance of $\alpha(\boldsymbol{v})$ across random directions will be $2/(n+2)$ times smaller than the variance among eigenvalues. For example, in BigGAN latent space (256 dimensions), the eigenvalues span over six orders of magnitude, while the $\alpha(\boldsymbol{v})$ for random

[1]For some spaces, we used spherical linear exploration, where we restrict ourselves to a spherical manifold of certain norm. We project $\boldsymbol{v}_i$ onto tangent space of $\boldsymbol{z}_0$ and then exponential map from $\boldsymbol{z}_0$.

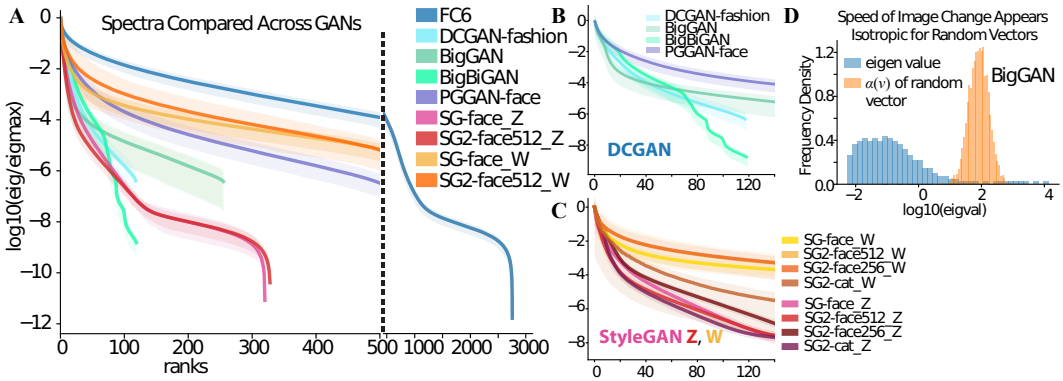


Figure 2: **Spectra of GANs**, shown as a function of individual model (A) and types of architecture (B, C). DCGAN-type (green-blue), Z and W space for StyleGAN (SG1, 2) (red, orange). Lines and shaded areas show the averaged spectra and the 5-95% percentile of each eigenvalue among samples (for quantification, see Table 2). D. Histogram of approximate speed of image change $\alpha(\boldsymbol{v})$ for eigenvectors and random directions, visualizing the "illusion of isotropy".

directions has a standard deviation less than one order of magnitude (Figs. 2, 6). Further, the center of this distribution was closer to the top of the spectrum, and thus provided a reasonable rate of change, while masking the existence of eigendimensions of extremely large and small eigenvalues.

**Global Metric Structure** Because the metric $H(\boldsymbol{z})$ describes local geometry, the next question is how it varies at different positions in the latent space. We computed the metric $H(\boldsymbol{z})$ at randomly selected $\boldsymbol{z}$ and examined their similarity using a statistic adopted from Kornblith et al. (2019). In this statistic, we applied the eigenvectors $U_i = [u_1, ...u_n]$ from a metric tensor $H_i$ at position $\boldsymbol{z}_i$ to the metric tensor $H_j$ at $\boldsymbol{z}_j$, as $u_i^T H_j u_i$. These values formed a vector $\Lambda_{ij}$, representing the effects of metric $H_j$ on eigenvectors of $H_i$. Then we computed the Pearson correlation coefficient between $\Lambda_{ij}$ and the target eigenvalues, $\Lambda_j$, as $corr(\Lambda_j, \Lambda_{ij})$. This correlation measured the similarity of the action of metric tensors on eigenframes around different positions. As the spectrum usually spanned several orders of magnitude, we computed the correlation on the log scale $C_{ij}^{Hlog}$, where the eigenvalues distribute more uniformly.

$$\Lambda_{ij} = diag(U_i^T H(\boldsymbol{z}_j) U_i) \tag{5}$$

$$C_{ij}^{H} = corr(\Lambda_{ij}, \Lambda_j),\ C_{ij}^{Hlog} = corr(\log(\Lambda_{ij}), \log(\Lambda_j)) \tag{6}$$

Using this correlation statistic, we computed the consistency of the metric tensor across hundreds of positions within each GAN latent space. As shown in Fig. 3C, the average correlation between eigenvalues and vHv values of two points $C_{ij}^{Hlog}$ was 0.934 in BigGAN. For DCGAN-type architecture, mean correlations on the log scale ranged from 0.92-0.99; for StyleGAN-1,2, 0.64-0.73 in the Z space, and 0.87-0.89 in the W space (Fig. 3D, Tab.3). Overall, this shows that the local directions that induce image changes of different orders of magnitude are highly consistent at different points in the latent space. Because of this, the notion of a "global" Hessian makes sense, and we estimated it for each latent space by averaging the Hessian matrices at different locations.

## 5 Mechanism

Above, we showed an intriguingly consistent geometric structure across multiple GANs. Next, we sought to understand how this structure emerged through network architecture and training.

To link the metric tensor to the generator architecture, it is helpful to highlight the relationship between the metric tensor and Jacobian matrix $H(\boldsymbol{z}) = J_{\varphi \circ G}^T J_{\varphi \circ G}$ (Eq. 3). As latent space gets warped and mapped onto image space, directions in latent spaces are scaled differently by the Jacobian of the map; specifically, directions that undergo the most amplification will become the top eigenvectors (Fig. 4A). As the Jacobian of the generator is a composition of Jacobians of each layer $d\psi_i$, the scaling effect on the image manifold is a product of the scaling effects of each intermediate layer. We can analyze the scaling effect of different layers $\psi_i$ by applying a set of vectors onto the metric tensors of these layers $H_{\psi_i}$. In BigGAN, when we apply the eigenvectors of the first few layers onto the metric of other layers, the top eigenvectors are still strongly amplified by subsequent layers, thus forming the top eigendimensions of the manifold. Of note, this is not true

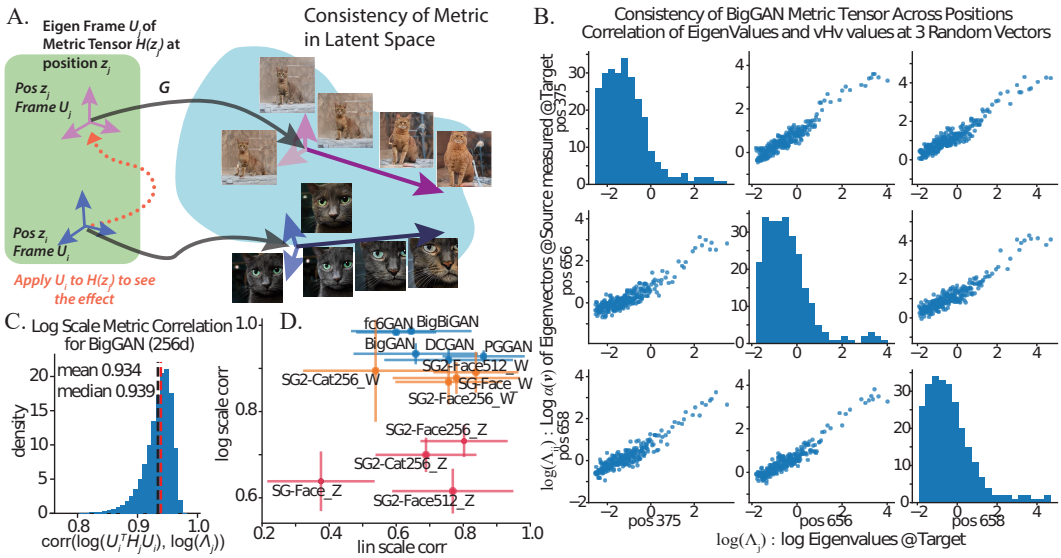


Figure 3: **Globalization of metric structure**: A. Conceptual schematic behind the geometric picture. In the latent space (green area), the metric eigenvectors at each point (blue and violet) form a local frame, which map to transformations in image space (blue area); the length and saturation of image-space vectors represent the eigenvalue (i.e. amplification factor) of $G$. We show the directions with large amplification factors are relatively aligned at different positions. B. Distributions of $\log(\Lambda_{ij})$ and $\log(\Lambda_j)$, showing the action of metric are correlated at 3 different points. C. Histogram of correlation $C_{ij}^{Hlog}$ between all pairs among 1000 points in BigGAN space. D. Comparison of correlation values on linear and log scales for different GAN models. DCGAN-type (blue), Z and W space for StyleGAN1,2 (red and orange).

for a weight-shuffled control BigGAN: in that case, the top eigendimension of the first few layers was not particularly amplified on the image manifold, and vice versa (Fig. 4 B). This shows that, the amplification effect of layers become more aligned through training, with the top eigenspace shared across layers. Further, as the amplification effects are not lined up across layers of weight shuffled networks, these networks should exhibit a more isotropic geometry on their image manifold. Indeed, we find their spectra to be flatter and the largest eigenvalue smaller (Fig. 7).

This finding unifies previous unsupervised methods that discover interpretable axes in the GAN space (Härkönen et al., 2020; Shen & Zhou, 2020; Voynov & Babenko, 2020). As we have showed, the top right singular vectors of the weights (i.e. Jacobian) of the first few layers (as used in Shen & Zhou (2020)), correspond to the top eigenvectors of the metric tensor of image manifold, and these usually relate to interpretable transforms. Similarly, the top principal components (PC) of intermediate layer activations Härkönen et al. (2020) roughly correspond to the top left singular vectors of the Jacobian, thus also to the interpretable top eigenvectors of the metric on the image manifold (although note that these PCs measure the global amplification of the map, not the local amplifications measured by our metric tensor). Finally, (Voynov & Babenko, 2020) discovered interpretable axes by optimizing for axes such that when two latent codes differed by a certain distance along this direction, a trained model could decode this direction out from the generated images. In our framework, it follows that when the axes lie in the top eigenspace, the image difference created will be the largest and thus easier to classify. Thus we show the geometric understanding of GAN image manifold has great explanatory power to all the methods above.

## 6 Application

By defining the geometry of the latent space via the metric tensor, we gain an understanding of which directions in this space are more informative than others. This understanding leads to improvements in three applications: 1) gradient descent-based optimizers, 2) accelerating gradient-free search, 3) finding human-interpretable axes in the latent space.

**Improving Gradient-Based GAN Inversion.** For applications like GAN-assisted drawing and photo editing (Zhu et al., 2016; Shen et al., 2020), one crucial step is to find a latent code corre-

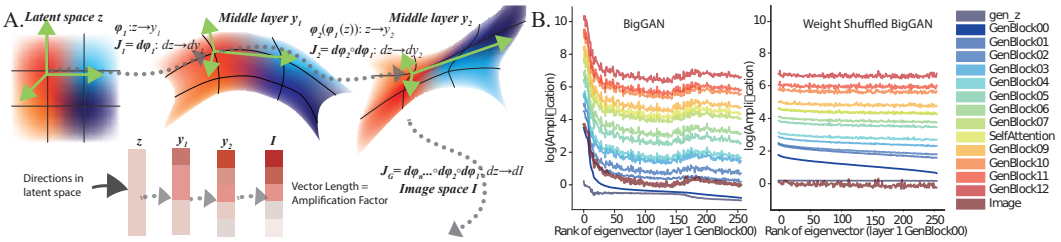


Figure 4: **Anisotropy is induced and maintained throughout the GAN architecture**. A. As latent space gets warped and mapped into image space, directions in latent spaces are scaled differently by the Jacobian of the map. B. Amplification of eigenvectors of the metric tensor of the first conv layer (GenBlock00) in all major layers in BigGAN. C. Same, but for weight-shuffled BigGAN.

sponding to given natural image (termed GAN inversion). For this problem, one basic approach is to minimize the distance between a generated image and the target image $z^* = \arg\min_z D(G(z), I)$. Although second-order information (Hessian) is valuable in optimization, they are seldom used as they are expensive to compute and update. However, because we find that local Hessian information is highly correlated at different points in latent space, we can pre-compute it once for each latent space and use the average Hessian information to boost first-order optimization. As an example, ADAM is a first-order optimization algorithm that adapts the learning rate of each parameter separately according to the moments of gradients on that parameter (Kingma & Ba, 2014). It can be seen as a quasi-second order optimizer that approximates a diagonal Hessian matrix based on first-order information. However, if the true Hessian is far from diagonal, i.e. the space is anisotropic and the valley is not aligned with the coordinates, then this approximation could work poorly.

To test whether the metric can help overcome this problem, we used the eigenvectors of the average Hessian to rotate the latent space; this orthogonal change of variables should make the Hessian more diagonal and thus accelerate ADAM. This method can be seen as a preconditioning step which could be inserted into any pipeline involving ADAM. We tested this modification on the state-of-the-art algorithm for inverting BigGAN, i.e. BasinCMA (Huh et al., 2020), which interleaves ADAM and CMAES steps. We used our Hessian eigenbasis in the ADAM steps, and found that we could consistently lower the fitted distance to the target when inverting ImageNet and BigGAN-generated images (Zhang et al., 2018) . (Fig. 5). Thus this understanding of homogeneity and anisotropy of the space can improve gradient-based optimization.

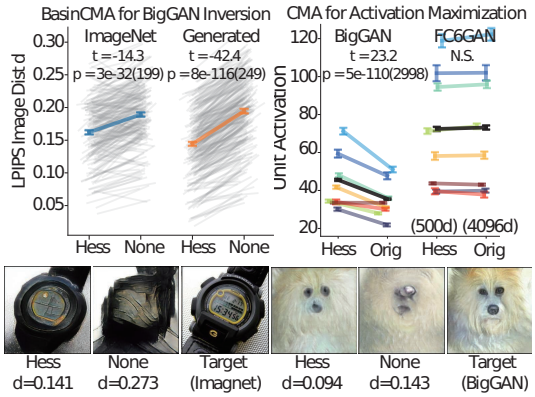


Figure 5: Left, **BasinCMA with Hessian eigenbasis parametrization (Hess) outperform method using normal basis (None)** in minimizing distance to ImageNet and BigGAN generated images; Right, **Hessian-CMAES (Hess) outperforms CMAES (Orig) in maximizing CNN activation** in BigGAN, and increases sampling efficiency in FC6GAN latent space. Color denotes the layer of the optimized units in AlexNet. Bottom, two examples of fitting ImageNet and BigGAN images with our Hessian BasinCMA and original method, with LPIPS distance below.

**Improving Gradient-Free Search in Image Space** In some domains, it is important to optimize objectives in the absence of a gradient, for example, in black-box attacks against image recognition systems via adversarial images, or when searching for activity-maximizing stimuli for neurons in primate visual cortex (Ponce et al., 2019; Xiao & Kreiman, 2020). These applications usually involve a low-dimensional parameter space (such as GANs) and an efficient gradient-free search algorithm, such as covariance matrix adaptation evolution strategy or CMAES. CMAES explores the latent space using a Gaussian distribution and adapts the shape of the Gaussian (covariance matrix) according to the search history and local landscape. However, online learning of a covariance matrix in high-dimensional space is computationally costly, and inaccurate knowledge of it can be detrimental to optimization. Here we applied the prior geometric knowledge of the space to build the covariance matrix instead of learning it from scratch. For example, as illustrated by natural gradient descent (Amari, 1998), one simple heuristic for optimizing on the image manifold is to move

in smaller steps along dimensions that change the image faster and vice versa. We used this heuristic to improve CMAES (a combined approach we refer to as CMAES-Hessian). With our method, the search can be limited to the most informative directions, which should increase sampling efficiency, and as the space is anisotropic, our method further tunes the exploration step size in a way that is inversely proportional to the rate of image change. To test this approach, we applied the novel CMAES-Hessian algorithm to the problem of searching for activation maximizing stimuli for units in AlexNet (Nguyen et al., 2016) in the latent space of two GANs (FC6GAN and BigGAN). We found that the dimensionality of the search space could be reduced from 4096 to 500 for FC6GAN GAN without impairing maximal activation values. Further, we found that CMAES-Hessian consistently led to higher activation values compared to the classic CMAES algorithm in BigGAN space (Fig. 5).

**Interpretable Axes Discovery for Image Manipulation** When users wish to manipulate GAN-based images via their latent code, it would be useful to reduce the number of variables needed to effectuate this given manipulation. Our method provides a systematic way to compute the most informative axes (top eigenspace) in the latent space to use as variables, and the resulting eigenvalues can serve to compute appropriate step sizes along each corresponding axes. We applied this method to FC6GAN, BigGAN, BigBiGAN, StyleGAN1,2, PGGAN, and plotted the changes corresponding to the top eigenvectors (Fig. 1). These eigenvectors corresponded to semantic meaningful transformations like zooming-in and out, head direction and object location. Moreover, eigenvectors usually corresponded to a similar semantic changes when applied to different reference vectors (Fig. 9), so the eigenvectors we discovered were interpretable not just in local sense, but in a global sense. Thus we could compute the average metric tensor once for each latent space and use their top eigendimensions as handles for image manipulation.

## 7 Discussion and Future Directions

In this work, we developed an efficient and architecture-agnostic way to compute the geometry of the manifold learnt by generative networks. This method discovers axes accounting for the largest variation in image transformation, which frequently represent semantically interpretable changes. Subsequently, this geometric method can facilitate image manipulation, increase explainability, and accelerate optimization on the manifold (with or without gradients).

Although we have answered how the anisotropy comes into being mechanistically, there remains the question of why it should exist at all. Anisotropy may result from gradient training: theoretical findings on deep-linear networks for classification show that gradient descent aligns the weights of layers, resulting in a highly anisotropic Jacobian (Ji & Telgarsky, 2018). Whether that analysis transfers to the setting of generative networks remains to be investigated.

Alternatively, assuming that a well-trained GAN faithfully represents the data distribution, this anisotropy may reveal the intrinsic dimensionality of the data manifold. Statistical dependencies of variation in real-world images (e.g. uniform changes in skin color, head direction) imply that the images reside in a statistical manifold of much lower dimension. Further, among transformations that happen on this manifold, there will be some dimensions that transform images a lot and some that do not (e.g. skin color versus facial expression). In that sense, our method may be equivalent to performing a type of nonlinear PCA of the image space through the generator map. In fact, we have found that GANs trained on similar datasets (e.g. PGGAN, StyleGAN1,2 trained on the human face dataset [CelebA,FFHQ]) have top eigenvectors that represent the same set of transforms (e.g. head direction, gender, age; Fig. 8). This supports the "PCA" hypothesis, as these transformations may account for much of the pixelwise variability in face space; the GANs are able to learn to represent these transformations as linear directions, which our method can then identify.

This further raises the intriguing possibility that if the dataset is actually distributed on a lower dimensional space, one could learn generators with smaller latent spaces; or alternatively, it may be easier to learn generators with large latent spaces and reduce them after intensive training. These are questions worth exploring.

AUTHOR CONTRIBUTIONS

ACKNOWLEDGMENTS

## REFERENCES

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

Shun-ichi Amari. *Information geometry and its applications*, volume 194. Springer, 2016.

Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: on the curvature of deep generative models. *arXiv preprint arXiv:1710.11379*, 2017.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

Nutan Chen, Alexej Klushyn, Richard Kurle, Xueyan Jiang, Justin Bayer, and Patrick Smagt. Metrics for deep generative models. In *International Conference on Artificial Intelligence and Statistics*, pp. 1540–1550. PMLR, 2018.

Jeff Donahue and Karen Simonyan. Large scale adversarial representation learning. In *Advances in Neural Information Processing Systems*, pp. 10542–10552, 2019.

Alexey Dosovitskiy and Thomas Brox. Generating images with perceptual similarity metrics based on deep networks. In *Advances in neural information processing systems*, pp. 658–666, 2016.

Ahmed Elgammal, Bingchen Liu, Mohamed Elhoseiny, and Marian Mazzone. Can: Creative adversarial networks, generating" art" by learning about styles and deviating from style norms. *arXiv preprint arXiv:1706.07068*, 2017.

Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. In *International Conference on Machine Learning*, pp. 2232–2241, 2019.

Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *arXiv preprint arXiv:2004.02546*, 2020.

Minyoung Huh, Richard Zhang, Jun-Yan Zhu, Sylvain Paris, and Aaron Hertzmann. Transforming and projecting images into class-conditional generative networks. *arXiv preprint arXiv:2005.01703*, 2020.

Ziwei Ji and Matus Telgarsky. Gradient descent aligns the layers of deep linear networks. *arXiv preprint arXiv:1810.02032*, 2018.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8110–8119, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. *arXiv preprint arXiv:1905.00414*, 2019.

Richard B Lehoucq, Danny C Sorensen, and Chao Yang. *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*. SIAM, 1998.

Anh Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox, and Jeff Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In *Advances in neural information processing systems*, pp. 3387–3395, 2016.

Richard S Palais. On the differentiability of isometries. *Proceedings of the American Mathematical Society*, 8(4):805–807, 1957.

Marcos Pividori, Guillermo L Grinblat, and Lucas C Uzal. Exploiting gan internal capacity for high-quality reconstruction of natural images. *arXiv preprint arXiv:1911.05630*, 2019.

Carlos R Ponce, Will Xiao, Peter F Schade, Till S Hartmann, Gabriel Kreiman, and Margaret S Livingstone. Evolving images for visual neurons using a deep generative network reveals coding principles and neuronal preferences. *Cell*, 177(4):999–1009, 2019.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Hang Shao, Abhishek Kumar, and P Thomas Fletcher. The riemannian geometry of deep generative models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 315–323, 2018.

Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. *arXiv preprint arXiv:2007.06600*, 2020.

Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9243–9252, 2020.

A Van der Sluis and HA Van der Vorst. The convergence behavior of ritz values in the presence of close eigenvalues. *Linear Algebra and its Applications*, 88:651–694, 1987.

Andrey Voynov and Artem Babenko. Unsupervised discovery of interpretable directions in the gan latent space. *arXiv preprint arXiv:2002.03754*, 2020.

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

Will Xiao and Gabriel Kreiman. Xdream: Finding preferred stimuli for visual neurons using generative networks and gradient-free optimization. *PLoS computational biology*, 16(6):e1007973, 2020.

Xinru Yuan, Wen Huang, Pierre-Antoine Absil, and Kyle A Gallivan. Averaging symmetric positive-definite matrices. In *Handbook of Variational Methods for Nonlinear Geometric Data*, pp. 555–575. Springer, 2020.

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.

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *European conference on computer vision*, pp. 597–613. Springer, 2016.

# A Appendix

## A.1 Connection to Information Geometry

It is useful to compare our work to the "information geometry"(Amari, 2016) on the space of distributions. In this formulation, KL divergence is a pseudo-metric function on the space of distributions, and its Hessian matrix towards parameters of distribution is the Fisher information matrix. In information geometry, this Fisher information matrix could be considered as the metric on the manifold of distributions; this metric information can be further used to assist optimization on the manifold of distributions, termed natural gradient descent.(Amari, 1998) In our formulation, the squared image

Table 1: **Computational Cost for Three Methods**: Computation time is measured on a GTX 1060 GPU. Note that the iterative method are variable in runtime due to different speed of convergence. A shallower or narrower network will result in faster computation time. For StyleGAN2 which has configurable depth and width, we use the config-f.

| | Dimension | Full BackProp Time | Backward Iter Time | Forward Iter Time |
|---|---|---|---|---|
| DCGAN | 120 | 12.5 | 13.4 | 6.9 |
| FC6 GAN | 4096 | 282.4 | 101.2 | 90.2 |
| BigGAN | 256 | 69.4 | 70.6 | 67 |
| BigBiGAN | 120 | 15.3 | 15.2 | 13.3 |
| PGGAN | 512 | 95.4 | 95.7 | 61.2 |
| StyleGAN | 512 | 112.8 | 110.5 | 64.5 |
| StyleGAN2* | 512 | 221 | 217 | 149 |

difference function $D^2$ is analogous to this KL-divergence; the image $G(\boldsymbol{z})$ as parameterized by latent code $\boldsymbol{z}$ is analogous to the distribution $p_\theta$ parameterized by $\theta$. The metric tensor we computed is comparable to the Fisher information matrix in their setting. Thus our way of using metric information to assist optimization on manifold is analogous to natural gradient descent.

### A.2 Methods for Computing the Hessian

One direct way to compute the Hessian towards a given objective (e.g. squared distance $d^2$ in our case), is to compute $g_{\boldsymbol{z}_0}(\boldsymbol{z}) = \partial_{\boldsymbol{z}} d^2|_{\boldsymbol{z}=\boldsymbol{z}_0}$, create a computational graph from code $\boldsymbol{z}$ to the gradient $g_{\boldsymbol{z}_0}(\boldsymbol{z})$, and back propagate from gradient vector $g_{\boldsymbol{z}_0}(\boldsymbol{z})$ element by element. In this way the computational time is linear to time of a single backward pass times the hidden space dimension $n$.

Given a large hidden space or a deep network (e.g. 4096d in FC6GAN, or 512d in StyleGAN2), this method can be very slow. However, as the eigenspace of small eigenvalues represent directions that do not change images much, the exact eigenvector does not matter. An efficient way is to use the Hessian vector product (HVP) operator and iterative eigenvalue algorithms to find the eigenvectors corresponding to eigenvalues of high amplitudes. In Power Iteration or Lancsoz Iterations, the Hessian vector product is used to find the largest amplitude eigen pairs. However, to find the smallest amplitude eigen pairs, Inverse Hessian vector product operator is required, it is much more expensive to compute such operator, but practically, there is little practical use for those dimensions. Thus we can define an arbitrary basis in the "null" space complementing to the eigenspace with higher amplitude eigenvalues.

There are two ways of constructing a Hessian vector product operator: one way requires the computational graph from $\boldsymbol{z}$ to the gradient $g(\boldsymbol{z})$, and computes the HVP by back-prop, i.e. $HVP_{backward}$; the other way is to use the first-order gradient and finite difference to compute HVP i.e. $HVP_{forward}$. As it doesn't require backpropagation, a single operation of $HVP_{forward}$ is much faster than $HVP_{backward}$ but it is less accurate and takes more iterations to converge. We use the ARPACK(Lehoucq et al., 1998) implementation of the Lancosz algorithm as the iterative eigenvalue solver,

$$HVP_{backward} : \boldsymbol{v} \mapsto \partial_{\boldsymbol{z}}(\boldsymbol{v}^T g(\boldsymbol{z})) \tag{7}$$

$$HVP_{forward} : \boldsymbol{v} \mapsto \frac{g(\boldsymbol{z} + \epsilon \boldsymbol{v}) - g(\boldsymbol{z} - \epsilon \boldsymbol{v})}{2\epsilon \|\boldsymbol{v}\|} \tag{8}$$

We denote the direct method Full BackProp (BP), the iterative method using $HVP_{backward}$ and $HVP_{forward}$ Backward Iteration and Forward Iteration respectively. Empirically, we computed the Hessian at the same $\boldsymbol{z}_0$ using these three methods in different GANs and compared their temporal cost in Table A.2.

Note, our method can be employed to compute the singular values and right singular vectors of the Jacobian from latent space towards any intermediate layer representation. To obtain the left singular vector, we need to push forward the right singular vectors through the Jacobian, which is feasible through forward-mode autodiff or finite difference.

### A.3 SPECIFICATION OF GAN LATENT SPACE

The pretrained GANs used in the paper are from the following sources.

**DCGAN** model is obtained from torch hub `https://pytorch.org/hub/facebookresearch_pytorch-gan-zoo_dcgan/`. It's trained on 64 by 64 pixel fashion dataset. It has a 120d latent space, using Gaussian as latent space distribution.

**Progressive Growing GAN (PGGAN)** is obtained from torch hub `https://pytorch.org/hub/facebookresearch_pytorch-gan-zoo_pgan/` and we use the 256 pixel version. It's trained on celebrity faces dataset (CelebA). It has a 512d latent space, using Gaussian as latent space distribution.

**DeePSim, FC6GAN** model is customly rewritten and translated into pytorch, with weights obtained from official page `https://lmb.informatik.uni-freiburg.de/people/dosovits/code.html` of Dosovitskiy & Brox (2016). The architecture is designed to mirror that of AlexNet, and the FC6GAN model is trained to invert AlexNet's mapping from image to FC6 layer. Thus it has 4096 d latent space. This model is highly expressive, but not particularly photorealistic.

**BigGAN** model is obtained through huggingface's translation of DeepMind's Tensorflow implementation `https://github.com/huggingface/pytorch-pretrained-BigGAN`, we use biggan-deep-256 version. It's trained on ImageNet dataset in a class conditional way. It has a 128d latent space called noise space, and a 128d embedding space for the 1000 classes called class space. The 2 vectors are concatenated and sent into the network. The distribution used to sample in noise space is truncated normal. Here we analyze the metric tensor computed in the concatenated 256d space (BigGAN) or in the 128d noise space or class space separately (BigGAN-noise, class).

**BigBiGAN** model is obtained via a translation of DeepMind's Tensorflow implementation `https://tfhub.dev/deepmind/bigbigan-resnet50/1`, we use bigbigan-resnet50 version. It's trained on ImageNet dataset in unconditioned fashion. It has a 120d latent space, using Gaussian as latent distribution. Note, the latent vector is split into six 20d trunks and sent into different parts of the model, which explains why the spectrum of BigBiGAN has the staircase form (in Fig. 2).

**StyleGAN** model is obtained via a translation of NVIDIA's Tensorflow implementation `https://github.com/rosinality/style-based-gan-pytorch`. We use the 256 pixel output. It has a 512d latent space called Z space, where the latent distribution is Gaussian distribution. This $Z$ distribution gets warped into another 512d latent space called W space, by a multi-layer perceptron. The latent vector $W$ is sent into a style-based generative network, in which the latent vector modulates the feature maps in the conv layers, instead of just serves as a spatial input as in DCGAN,FC6GAN,PGGAN. We analyzed the geometry of $Z$ space and $W$ space separately, and find that the metric in $W$ space is significantly flatter and more homeogeneous!

**StyleGAN2** models are obtained via a translation of NVIDIA's Tensorflow implementation `https://github.com/rosinality/stylegan2-pytorch`. It's similar to StyleGAN: it has a similar structure of the 512d $Z$ and $W$ space, and the style-based generative network. The various pre-trained models are fetched from `https://pythonawesome.com/a-collection-of-pre-trained-stylegan-2-models-to-download`. More specifically StyleGAN2-Face256 and 512 are both trained on FFHQ dataset, while Face256 generate lower resolution images and use narrower conv layers. StyleGAN2-Cat is trianed on LSUN cat dataset at 512 resolution.

### A.4 QUANTIFICATION OF POWER DISTRIBUTION IN SPECTRA

We quantified the anisotropy of the space, i.e. the low rankness of the metric tensor in Table 2. To do this, we computed the number of eigenvalues needed such that their sum will account for a fraction of 0.99, 0.999, 0.9999, 0.99999 of the sum of all eigenvalues. This can be thought of as the fraction of dimensions one needs to achieve a low rank approximation of the Jacobian with 0.01, 0.001, 0.0001, 0.00001 residue in terms of the Frobenius norm.

Table 2: **Quantification of Spectra anisotropy**

| | dimen | dim.99 | dim.999 | dim.9999 | dim.99999 |
|---|---|---|---|---|---|
| FC6 | 4096 | 297 | 502 | 661 | 848 |
| DCGAN-fashion | 120 | 17 | 35 | 65 | 97 |
| BigGAN | 256 | 10 | 53 | 149 | 224 |
| BigGAN_noise | 128 | 29 | 88 | 120 | 127 |
| BigGAN_class | 128 | 8 | 38 | 98 | 123 |
| BigBiGAN | 120 | 21 | 41 | 62 | 73 |
| PGGAN-face | 512 | 57 | 167 | 325 | 450 |
| StyleGAN-face_Z | 512 | 12 | 27 | 52 | 84 |
| StyleGAN2-face512_Z | 512 | 7 | 17 | 41 | 78 |
| StyleGAN2-face256_Z | 512 | 13 | 28 | 63 | 103 |
| StyleGAN2-cat_Z | 512 | 8 | 14 | 31 | 62 |
| StyleGAN-face_W | 512 | 124 | 355 | 480 | 507 |
| StyleGAN2-face512_W | 512 | 153 | 345 | 471 | 506 |
| StyleGAN2-face256_W | 512 | 157 | 350 | 473 | 506 |
| StyleGAN2-cat_W | 512 | 23 | 57 | 126 | 269 |

### A.5 Random Mixing of Spectra

Here we give a simple derivation of why a highly ill-conditioned Hessian matrix may appear normal, under the probe of random vectors. Given a symmetric matrix $H$, and its eigen decomposition $U\Lambda U^T$, we computed its effect on an isotropic random vector $\boldsymbol{v}$, $\alpha(\boldsymbol{v}) = \boldsymbol{v}^T H \boldsymbol{v} / \boldsymbol{v}^T \boldsymbol{v}$, and $\boldsymbol{v} \sim \mathcal{N}(0, I)$. This random variable represents the effect of the symmetric matrix on random directions.

Note that a change of variable using the orthogonal matrix $\boldsymbol{w} = U^T \boldsymbol{v}$ will not change the distribution $\boldsymbol{w} \sim \mathcal{N}(0, I)$. Through this the random variable $\alpha$ could be rewritten as

$$\alpha(\boldsymbol{v}) = \frac{\boldsymbol{v}^T H \boldsymbol{v}}{\boldsymbol{v}^T \boldsymbol{v}} = \sum_i \lambda_i \frac{\|\boldsymbol{u}_i^T \boldsymbol{v}\|_2^2}{\|\boldsymbol{v}\|_2^2},\ \boldsymbol{v} \sim \mathcal{N}(0, I) \tag{9}$$

$$= \sum_i \lambda_i w_i^2 / \sum_i w_i^2,\ w_i \sim \mathcal{N}(0, I)\ i = 1, 2...d \tag{10}$$

$$= \frac{1}{\sum_i w_i^2} \sum_i \lambda_i w_i^2 = \sum_i c_i \lambda_i \tag{11}$$

$$c_i := w_i^2 / \sum_i w_i^2 \tag{12}$$

As each element in $\boldsymbol{w}$ is distributed as i.i.d. unit normal, $w_i^2$ is distributed as i.i.d. chi-square distribution of parameter 1. $w_i^2 \sim \chi^2(1) \sim \Gamma(\frac{1}{2}, \frac{1}{2})$. Thus the normalized weights $c_i = w_i^2 / \sum_i w_i^2$ conform to a Dirichlet distribution $\boldsymbol{c} \sim Dirichlet(\frac{1}{2}, \frac{1}{2}, ...\frac{1}{2})$. Through moment formula of Dirichlet distribution, it is straightforward to compute the mean and variance of $\alpha(\boldsymbol{w}) = \boldsymbol{c}^T \lambda$

$$\mathbb{E}[\alpha] = \lambda^T E[\boldsymbol{c}] = \frac{1}{n} \sum_i \lambda_i \tag{13}$$

$$Var[\alpha] = \lambda^T Cov[\boldsymbol{c}] \lambda = \frac{2}{n(n+2)} \sum_i \lambda_i^2 - \frac{2}{n^2(n+2)} (\sum_i \lambda_i)^2 \tag{14}$$

$$Var[\lambda] = \frac{1}{n} \sum_i \lambda_i^2 - \frac{1}{n^2} (\sum_i \lambda_i)^2 = \frac{n+2}{2} Var[\alpha] \tag{15}$$

As we can see, the variance of the effect on random directions scales $1/n$ relative to the variance of original eigenvalue distribution. This is why the distribution is much tighter than the whole eigenvalue distribution.

This phenomenon may explain why the perceptual path length regularization as used in Karras et al. (2020) doesn't really result in a flat spectrum. This regularization is to minimize $\mathbb{E}_{\boldsymbol{v}} \|\alpha(\boldsymbol{v}) - b\|^2$, which is to minimize the variance of the distribution of $\alpha(\boldsymbol{v})$ with $\boldsymbol{v}$ sampled from normal

distribution, as in Fig. 6. The global minimizer for this regularization is indeed a mapping with flat spectrum. However, through our derivation, we can see even for highly anisotropic spectrum, this variance will not be very large. Thus we should expect a limited effect of this regularization.

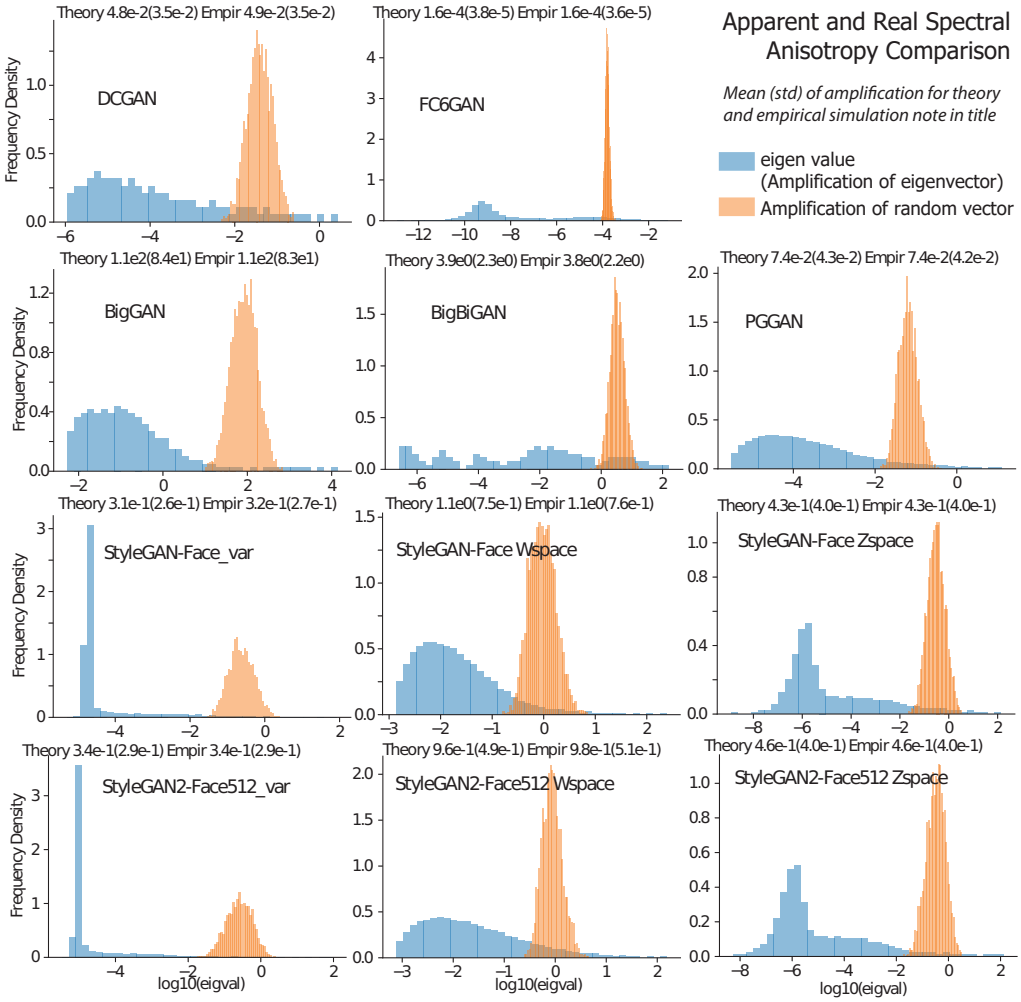


Figure 6: **Spectral Histogram compared to Apparent Anistropy in Different GAN models** (FC6GAN, DCGAN, BigGAN, BigBiGAN, PGGAN, StyleGAN, StyleGAN2) The apparent speed of image change $\alpha(\boldsymbol{v})$ has much smaller variability than the variability in the whole spectra. Eq. 13 can predict the mean and std of the apparent variability.

### A.6 Method to Quantify Metric Similarity

We developed our own statistic to quantify the similarity of metric tensor between different points. Here we discuss the benefits and caveats of it.

Angles between eigenvectors *per se* are not used, because eigenvectors are likely to rotate into each other when computing eigendecomposition in the presence of close-by eigenvalues (Van der Sluis & Van der Vorst, 1987). However, the statistics should be invariant to this eigenvector mixing, and take the eigenvalues into account. We applied the eigenvectors $U_1 = [\boldsymbol{u}_{1,i}, ...]$ of one matrix $H_1$ to the other $H_2$ and examined the length of these vectors as measured by the other matrix as the metric tensor, $\alpha_{H_2}(\boldsymbol{u}_{1,i}) = \boldsymbol{u}_i^T H_2 \boldsymbol{u}_i$. Recall that $\alpha_{H_1}(\boldsymbol{u}_{1,i}) = \boldsymbol{u}_{1,i}^T H_1 \boldsymbol{u}_{1,i} = \lambda_{1,i}$ and $\alpha_{H_2}(\boldsymbol{u}_{2,i}) = \boldsymbol{u}_{2,i}^T H_2 \boldsymbol{u}_{2,i} = \lambda_{2,i}$. If the eigenvectors fall in the eigenspace of the same eigenvalue in $H_2$, then $\alpha(\boldsymbol{u}_{1,i})$ will equal the eigenvalue, and thus our statistic is invariant to rotation within the eigenspace. If the eigenvectors are totally uncorrelated, the resulting $\alpha(\boldsymbol{u}_{1,i})$ will distribute in a

tight distribution. As we compute the correlation between the eigenvalue $\lambda_{2,i} = \alpha_{H_2}(\boldsymbol{u}_{2,i})$ and the $\alpha_{H_2}(\boldsymbol{u}_{1,i})$, we summarize the similarity of action of $H_2$ on eigenvectors $U_1$ and $U_2$.

However, this method assumes an anisotropy of spectra in both metric tensors. For example, if both tensors are identity matrices $H_1 = H_2 = I$, then this correlation will give NaN, as there is no variation in the spectra to be correlated. Similarly,if metric has has a spectrum closer to flat, then it will generally have a smaller correlation with others. In that sense, spectral anisotropy also plays a part in our statistics for metric similarity or homogeneity of the manifold. In all the GAN space we examined, there is a strong anisotropy in the metric spectra, thus this correlation works fine. But there is caveat for comparing this correlation between two GANs when there is also difference in the anisotropy in their spectra, as a smaller anisotropy can also results in a smaller metric similarity.

Finally, we are aware that several measures of distance in the space of symmetric positive definite matrices (SPSD) have been proposed (Yuan et al., 2020). Thus, there are different ways to compute an average metric tensor using these distance function. Here we picked the simplest one: averaging the metric tensors element by element.

Table 3: **Quantification of Manifold Homogeneity** by metric conisistency $C^H_{ij}$ on log scale and linear scale. Same data generating Fig. 3 D

| | Log scale | | Linear Scale | |
|---|---|---|---|---|
| | mean | std | mean | std |
| FC6GAN | 0.984 | 0.002 | 0.600 | 0.119 |
| DCGAN | 0.920 | 0.028 | 0.756 | 0.192 |
| BigGAN | 0.934 | 0.024 | 0.658 | 0.186 |
| BigBiGAN | 0.986 | 0.007 | 0.645 | 0.180 |
| PGGAN | 0.928 | 0.014 | 0.861 | 0.123 |
| StyleGAN-Face_Z | 0.638 | 0.069 | 0.376 | 0.160 |
| StyleGAN2-Face512_Z | 0.616 | 0.052 | 0.769 | 0.181 |
| StyleGAN2-Face256_Z | 0.732 | 0.037 | 0.802 | 0.130 |
| StyleGAN2-Cat256_Z | 0.700 | 0.040 | 0.689 | 0.151 |
| StyleGAN-Face_W | 0.878 | 0.037 | 0.780 | 0.190 |
| StyleGAN2-Face512_W | 0.891 | 0.048 | 0.838 | 0.127 |
| StyleGAN2-Face256_W | 0.869 | 0.052 | 0.756 | 0.159 |
| StyleGAN2-Cat256_W | 0.895 | 0.118 | 0.539 | 0.216 |

### A.7 Geometric Structure of Weight-Shuffled GANs

Here we show the geometric analysis for the shuffled controls for all our GANs. Specifically, we shuffled the elements of each layer weight tensor to keep the overall weight distribution unchanged. To show how learning affected the geometry of the image manifold, we computed the spectra and the associated metric consistency statistic for weight-shuffled GANs[2].

In Fig. 7, we showed that the shuffled controls exhibited a flatter spectrum and a smaller top eigenvalue. There, the correlation of metric tensors in shuffled GANs shows an unclear result. In some GANs, there remains strong correlations in the metrics across locations, while in some, the correlation is close to zero. We think the reason is that our statistic for homogeneity (i.e. a correlation of action of metric tensors $C^{Hlog}_{ij}$) somewhat entangles homogeneity with the anisotropy of the space. That is, when the space has a totally flat spectrum (the map is isometric), then the correlation coefficient of action will be zero, although the metric tensor will be the same everywhere. Thus the change of anisotropy and the change of homogeneity may interfere with each other, thus shuffling can result in a mixed result. We are working to develop new statistics that will measure the similarity of the Hessian, invariant of anisotropy.

[2]We were unable to obtain a sensible spectra for either shuffled or randomly initialized BigBiGAN possibly due to its architecture; but we show the comparison for all other models.

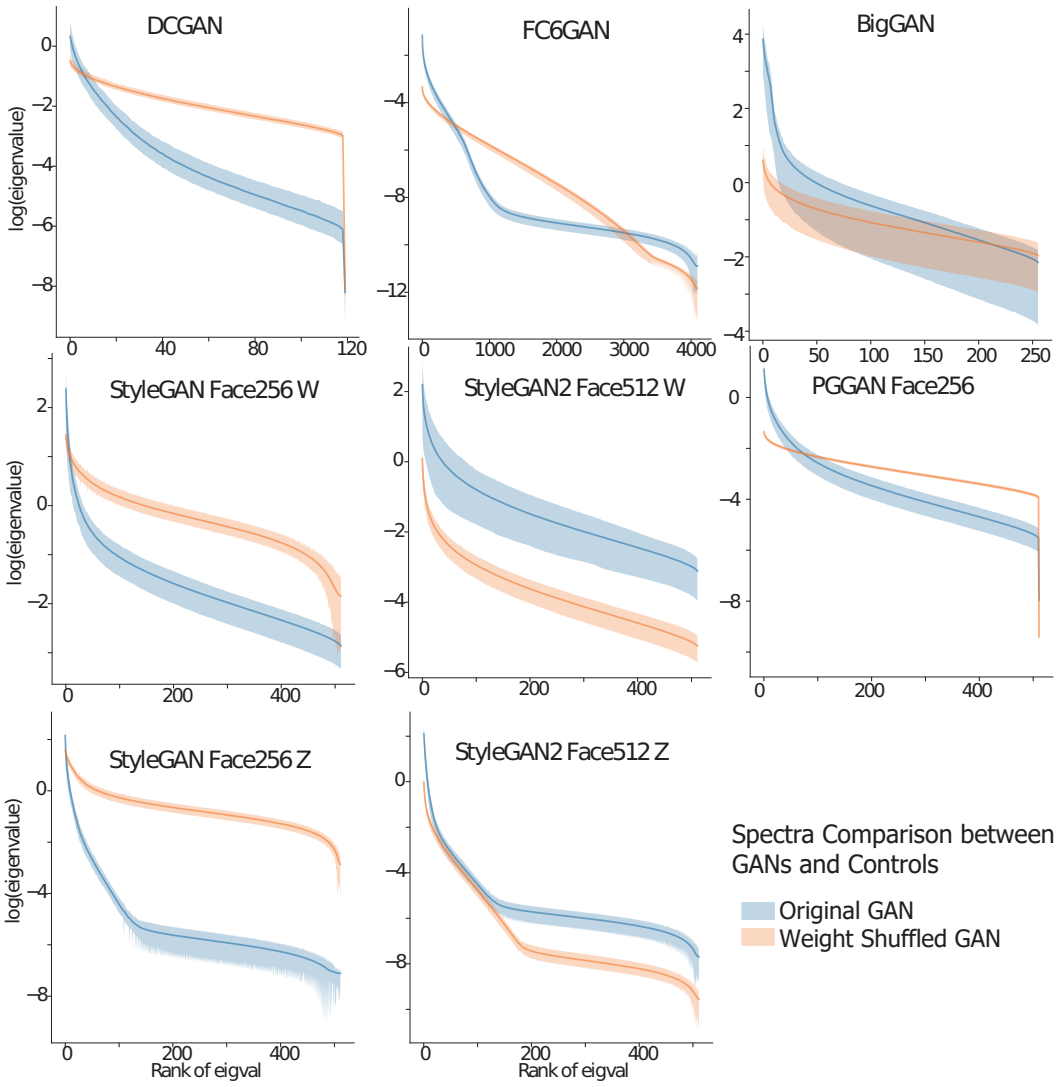


Figure 7: **Comparing Spectra of Original and Weight Shuffled GANs**: Most Shuffled GAN shows a slower spectral decay and a smaller maximum eigenvalue.

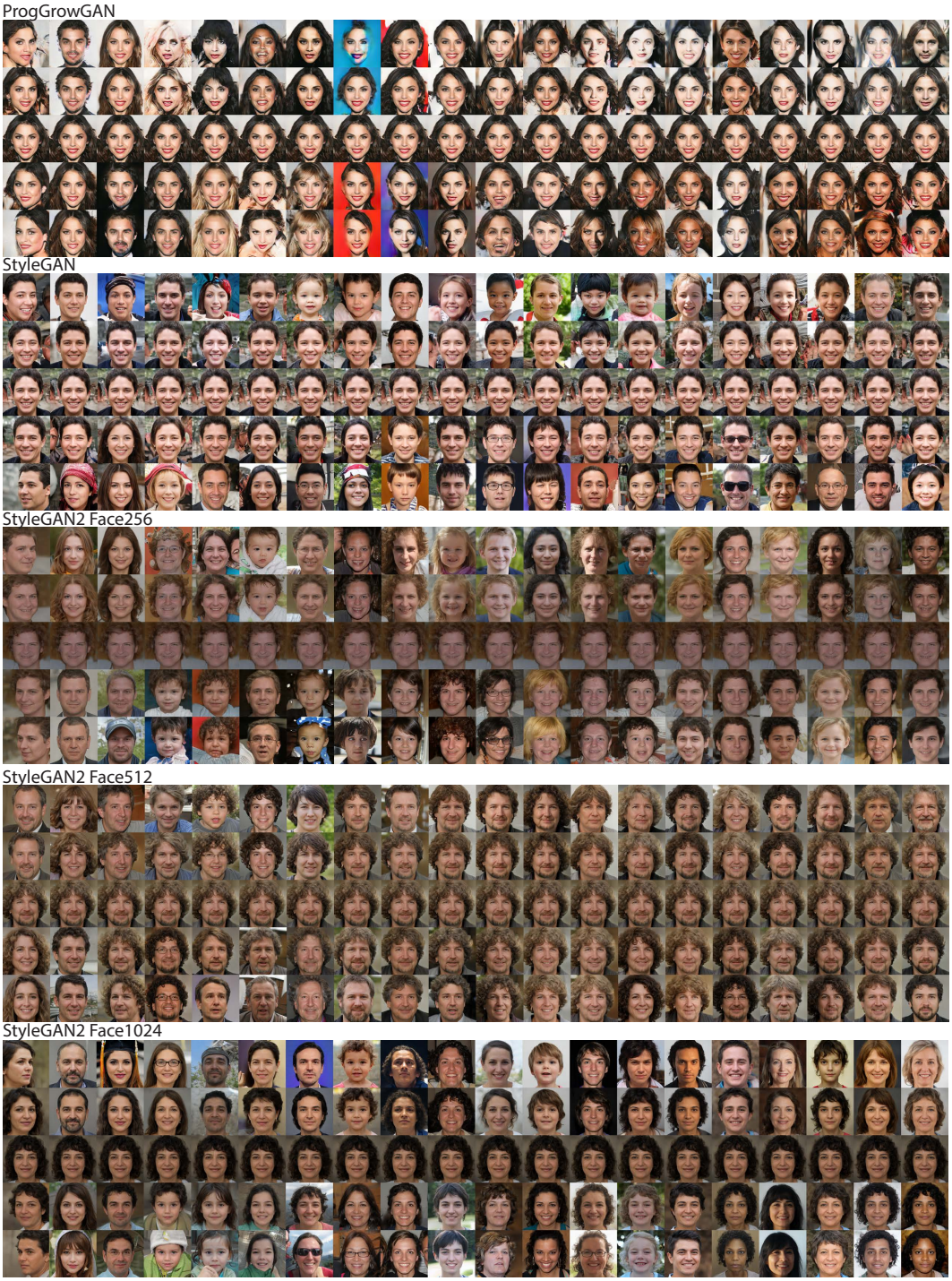


Figure 8: **Similar transforms encoded in the top eigendimension of GANs trained on same face dataset**. Linear exploration along top 20 eigenvectors from origin in latent space are showed for each GAN. Linear equi-distance sampling on each eigenvector occupies a column and their eigenvalues are sorted in descending order from left to right. Step size along each vector is adjusted according to its eigenvalue for best continuity.

Figure 9: **Top eigenvectors encode similar transforms around different reference images**. Linear equidistant explorations from six randomly chosen reference images along the eigenvectors of averaged Hessians. These show qualitatively similar transforms to images — for example, proximity of Cat face (Eig1), proximity and cat number (Eig4), fur color darkening (Eig10) in StyleGAN2 Cat; face direction (Eig1), masculine vs feminine (Eig3), child vs adult (Eig5) in StyleGAN Face.