WASHINGTON UNIVERSITY IN ST. LOUIS

Division of Biology and Biomedical Sciences
Neurosciences

Dissertation Examination Committee:
Timothy E. Holy, Chair
Carlos R. Ponce, Co-Chair
Dennis Barbour
Gabriel Kreiman
Gaia Tavoni

Charting the Landscape of Ventral Stream Neural Code on Generative Image Manifolds

by
Binxu Wang

A dissertation presented to
Washington University in St. Louis
in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy

Aug 2023
St. Louis, Missouri

Table of Contents

List of Figures

List of Tables

Acknowledgments

I am privileged to have spent my 5-year PhD journey in two amazing places: first three years in Washington University in St Louis, and the last two years at Harvard University in Boston. It should not be surprising that a larger than normal number of individuals have contributed to this journey and helped me get where I am.

My greatest appreciation goes to my PhD advisor Carlos R. Ponce, without whom this journey wouldn't have been possible. He is a truly incredible advisor, collaborator, and friend.

Carlos has a near infectious passion for science, and he is an unbelievably patient and understanding mentor. As a pure theorist joining the lab, I didn't know anything about doing any biological experiments, and I was not sure if I wanted to. However, seeing him building rigs, engineering parts, and recording data (while singing musicals) had this magical power on me that "wow this is cool, maybe I should try it". He tirelessly demonstrated how to perform each experiment himself until I felt comfortable doing so. After learning how to handle monkeys and do recordings, I immediately see the immense power of combining computation with experimental designs from the start. For most theorists, it's beyond their wildest dreams to be able to think about a new idea, develop experiments, record data and pilot test the idea by analysis in one or two days. But Ponce lab made this possible for me. His sheer joy and enthusiasm when seeing a new experimental result (as he said, "if I had a tail, you would see it wiggling"), are truly contagious, and these moments would influence me and remind me why we are doing science.

We were not without conflicts. Coming from different backgrounds, theory vs experiment, physics vs biology, we often have different opinions. However heated the discussion might be, we usually find the alternative view informative after discussion. Most importantly, he always

respected my intellectual freedom and let me develop analysis and experiments that drove my curiosity the most. This turned out to be fruitful and resulted in the near solo works of ICLR and GECCO. Even as a PI, he has been eager to pick up new skills, and he has encouraged his experimental students to be more computational and vice versa, the motivation to reduce the gap between experiment and theory has inspired and encouraged me a lot.

Regarding this relationship, one less conventional aspect is that I am one of his first students, rotating from virtually day one of the lab. As a student, observing the struggle and growth of a young PI, as well as witnessing the creation and flourish of a laboratory from scratch, has been an invaluable treasure for my academic career. Through his guidance and exemplary, I have learnt many skills crucial for an independent researcher: I learned how to write research papers properly, how to deliver a talk like a virtuoso, and how to manage tricky social situations inside and outside labs. Carlos is not a native speaker of English and he admits to be a bit socially introverted, yet he makes sure his presence is impeccable through remarkable dedication and meticulous preparation. Seeing Carlos thrive has been truly inspiring to me as an introvert and foreign speaker. Beyond science, he is such a caring, considerate, and lovable mentor and friend, I cannot express how much I love and admire his character as a person.

Inside the Ponce lab, I'm grateful for the incredibly kind and capable colleagues and lab members, Olivia Rose, Mary Carter, and Elizabeth Cleaveland, they have contributed greatly to our primate research, including daily caring for monkeys, enrichments, performing surgeries, and collecting a part of the electrophysiology data presented in the thesis. They have made the life of non-human primates (macaques) and the human primate (me) in the lab much more enjoyable. Without their dedicated experimental support and emotional care, I might have quit primate research earlier and could never be as productive as I was with their help. Aside from them, I

really enjoyed discussion and friendship from many current and previous labmates, Victoria Zhang, Katherine Mueller, Giordano, Alireza Dehaqani, Jeevun Kansupada, Antonio Montanaro, Chandana Kuntala. Their presence had made the life inside and outside much more vibrant and fun. Scientific or philosophical discussions with many of them have triggered amazing ideas.

Timothy Holy is another advisor who deeply influenced my science and career. I spent the 3rd rotation in his lab doing light-sheet imaging on larval zebrafish. His scientific taste of what the important questions in neuroscience are and how to tackle them via technical and computational development have influenced me so much. His arduous focus on computation and quality of code has deeply rooted in my heart and benefited me till this day. His personal advice to continue honing my skills in both math and neuroscience 4 years ago continues to resonate with me and it still motivates me to climb these mountains. One of my favorite projects (ICLR 2021) was inspired by Tim's comments --- "*Why don't you compute the Hessian and use it in optimization*". That single line has inspired me for more than a year and it led to develop numerical algorithms for computing Hessian, and the experimental paradigm of measuring the "Hessian" of tuning landscape. Even after I moved to Harvard, I still benefited from chatting with Tim about career and research whenever I flew back to St Louis and randomly knocked at his office door.

The advisor of my 2nd rotation, Dr. Joshua Morgan, has also deeply influenced my trajectory in science. During my rotation in his connectomics lab, I trained large-scale neural network models to help segment the 3D Electron Microscopy volume. After continuous training on V100 GPU for a month, the model gradually learned to segment longer and longer branches of neurite like magic and did it at scale --- which has taken human years to do so. For the first time, I was struck by the profound influence that deep learning can have on neuroscience. From then on, I've always favored problems where large-scale computation could help. During that rotation, I have

learned many great practices of software debugging, neural network training and parallel computing on cluster which still benefited me till this day.

Additionally, I'm thankful to numerous faculty members, such as Dennis Barbour, Gaia Tavoni, Gabriel Kreiman, Demba Ba, Cengiz Pehlevan, Margaret Livingstone, John Assad, and Hang Zhang, who have either invited me to speak or attended my talks and offered invaluable feedback. Their comments have greatly broadened my knowledge and contributed to the scientific stories.

I am deeply grateful for my *alma mater*, Yuanpei college in Peking University, which built a broad and solid knowledge background for me in Physics, Math, Computation, and Philosophy. Even more crucially, the focus on intellectual independence and personal choice prepared my character for self-directed research in PhD. Having pursued coursework from diverse disciplines in college, I gained the priceless courage of marching into any unknown domain of knowledge and grasping whatever I need for my quest. Without this foundational knowledge and the daring spirit of an explorer, I wouldn’t be where I am today. I'm also profoundly thankful for my college advisor, Louis Tao, who was the individual cultivating my interest in theory and my intellectual autonomy. His appreciation for the elegance of research continues to shape my preferences to this day.

I am fortunate to receive a substantial level of academic inspiration from friends who are distant. The most important of them is the “Hot Spring Harbor” (inspired by cold spring harbor), comprised of my college friends Yunyi Shen (MIT EECS), Hao Sun (Cambridge Applied Math), Jialong Jiang (Caltech Comp Bio). We had all studied and researched a diverse range of topics with different expertise. Many of them have helped me and guided me through mathematical and computational problems in research and emotional challenges in life. Further, my friends

Zhengdao Chen, Yao Fu, Yuxiu Shao, Lingwei Kong, Tianxing Zheng, Weilong Fu have also provided kind and crucial advice and feedbacks to my research through this journey. These friends have been like stars in the night sky, giving me guidance even during the most isolated and lonely days in this journey.

I'm grateful for my friends and co-authors John Vastola, Colin Conwell (CoCo), Arturo Deza. The hours I spent writing equations and pitching new ideas to John and drawing random figures on the white board in Jan's lab are still some of my most enjoyable and thought-provoking experience of scientific discussion. During this productive intellectual collaboration, we delivered a few popular machine learning tutorials, and even wrote a paper together, which was the key component to my job talk in Kempner. With CoCo and Arturo, the collaboration starting from the Brain Minds and Machines summer school resulted in a paper and even initialized a startup. These individual collaborations taught me the intricacies of independent research and working harmoniously with peers, all while lacking direct guidance from professors.

Friends in WashU have made to my PhD journey much more enjoyable, Keran Yang, Kaining Zhang, Pingchuan Ma, Yifan Xu, Zhangying Cai, Aelita Zhu, Wilbur Shi, Zhikai Liu, Ruiyang Liu, Xiaodan Wang, Yilin Wang, Emily Hsiang, Rene Gao and many others. I vividly remember our hiking excursions and road trips through Midwest, Yellowstone, Seattle, and Wisconsin, as well as the nights we spent in one another's living rooms playing boardgames or talking about science, history, and life. I'm still nostalgic about the St Louis lifestyle we had.

After moving to Boston, I've enjoyed the companion of many friends and colleagues, including Jingxuan Fan, Siyan Zhou, Ningjing Xia, Shane Shang, Lily Zhang, Saloni Sharma, Kasper Vinken, Will Xiao, Albert Chen, Shin Kira, Julia Nguyen, Shih-Yi Tseng, etc. Taking

courses, attending journal clubs together and joining them at parties have been a joy in life. I want to extend special appreciation to the Chinese Dance community in Boston. This incredible group, including Qi Yang, Aileen, Ningjing, Xiaohang, Lena Tao, Mia Tsai, and others are among the first group who warmly embraced me after I moved to the city. Rehearsing, performing, and hanging out with them have made my life outside lab much more beautiful.

Financially, I'm grateful for the CCSN pathway fellowship at Washington University and Victoria Quan fellowship from Harvard for funding my PhD journey.

Finally, I would like to express my gratitude to my mother. Her unconditional emotional support and intellectual guidance (she has some brilliant ideas for experiments!) throughout the journey have led me to the achievements today.

Binxu Wang

Washington University in St. Louis

Aug 2023

Dedicated to beauty.

"*Wherefore one ought to distinguish two kinds of causes, the necessary and the divine, and in all things to seek after the divine for the sake of gaining a life of blessedness, so far as our nature admits thereof, and to seek the necessary for the sake of the divine, reckoning that without the former it is impossible to discern by themselves alone the divine objects after which we strive, or to apprehend them or in any way partake thereof."*

— *Timaeus* 68e, Plato

ABSTRACT OF THE DISSERTATION

Charting the Landscape of Ventral Stream Neural Code

on Generative Image Manifolds

by

Binxu Wang

Doctor of Philosophy in Biology and Biomedical Sciences
Neurosciences

Washington University in St. Louis, 2023

Professor Timothy Holy, Chair

Professor Carlos R. Ponce, Co-Chair

The natural world features high-dimensional retinal inputs to the visual system. In contrast, to study vision in lab, only sparsely sampled image sets are used. Given this immense image manifold, what is a principled way to sample and understand the neural representations on it? In this thesis, we frame the neural coding question in a geometric way. Neural tuning can be conceptualized as a landscape on the natural image manifold, and generative models such as Generative Adversarial Networks (GANs) provide a concrete instantiation of the manifolds. We hypothesized that the maximally activating images or peaks on these landscapes are critical for understanding visual neurons. Thus, we used the neuron-guided image synthesis paradigm to find these peaks, where an evolutionary algorithm iteratively optimized the images to increase the firing rate of a target neuron, gradually reaching a peak on the tuning landscape.

We first characterized the Riemannian geometry of the GAN image manifolds and leveraged the geometry to develop a better evolutionary optimizer to control the neurons. We applied the closed-loop paradigm to neurons recorded in V1, V4, and posterior inferotemporal cortex (pIT)

in two monkeys. Along the ventral stream, we found a few consistent trends. Going up the hierarchy, the tuning peaks became sharper, and they took more optimization iterations to find. By constraining the optimization in linear subspaces, we found the tuning peaks became higher dimensional. Further, we compared the image optimization in multiple generative spaces: a pattern-based generator, and an object-based generator. We found that going up the hierarchy, it became increasingly easy to guide optimization on the object manifold and increasingly hard on the pattern manifold. Further, on both manifolds, the optimized images for IT neurons have higher objectness scores than V4. Thus, the tuning peaks of higher visual neurons (pIT) were located closer to the object manifold, and their tuning functions were more aligned with object-based parametrization.

In the future, we'd like to consider how these landscapes are combined to construct population representations, and how these maximally activating stimuli are critical to driving downstream behaviors.

Chapter I : Introduction

I.1 A Whirlwind Tour Through Primate Ventral Stream

For us primates, vision serves as the main channel of information. Within the brain, a substantial portion of resources is allocated to handle and interpret this abundant source of data. For example, in macaque monkey, it was estimated that 55% of surface area in neocortex is related to visual processing ((Felleman and Van Essen, 1991) Table 2). If we compare the neocortex in the brain to the silicon chips in computers, then the visual cortex as "graphic processing units" (GPU) takes up such a major part of this chip. Thus, to understand the brain and the intelligence emerging from it, it may be critical to understand the visual frontend of it.

Vision study has a long history. Since the early modern age (1600-1700s), western philosophers have been fascinated by the faculty of vision. For example, informed by anatomical study, René Descartes (1596-1650) already had a mechanistic view of vision, where light particles strike the eye, and the pulses pass through the optic nerves to the brain (*Principles of Philosophy,* 1644 (Descartes, Miller and Miller, 1988)). This picture of early vision pathway is relatively accurate even in today's regard. In that age, philosophers (e.g. John Locke) had been puzzled by the problem of objective and subjective quality, namely, which quality belongs to the object and which belongs to the observer or our mind. Their puzzle highlights the vast difference between the quality of information input into the brain and the perceptual output we get. The input to our visual system is the spatiotemporal pattern of photons with varying wavelength falling onto the two-dimensional retina, in contrast, what we perceive daily is objects with color and shapes convincingly living in three-dimensional space. Thus, it's natural to ask which aspects we see are a faithful reflection of the world and which aspects are a construction of the mind. This drives

philosophers into debates of whether color is a property of the object or that of us observers. In the synthesis view of Immanuel Kant (1724-1804), the philosopher proposed that our knowledge of the world doesn't come solely from sensory data or from pure reason (*a priori*), but rather from a synthesis of these two sources (Critique of pure reason, 1781, (Kant, 1908)). Further, he thought our minds contained inbuilt structures to organize sensory data into perceptual experiences. This perspective aligns closely with our contemporary understanding of vision. In the context of the 21st century, the 'inbuilt structures' that Kant mentioned can be understood as the neural system responsible for processing sensory data, along with the accompanying inductive biases. Two centuries ago, the absence of neurophysiological methods and a computational understanding of vision left philosophers unable to grasp how this information was processed and represented within the brain. Fast forward 200 years, the same sense of wonder regarding how the incoming information in the visual system translates into our perceptions, and what 'inbuilt structures' enable this process, continue to fuel the exploration of visual neuroscience.

Organization of Visual Cortices. Inside the primate brain, the visual information originated from retina projects onto the cortices and other subcortical regions. Here, we focus on cortical processing. Within the cortices, the visual processing cortices are organized as a hierarchical and interconnected network (Felleman and Van Essen, 1991; Van Essen, Anderson and Felleman, 1992; Markov *et al.*, 2014). With tracing techniques, researchers could study the long-range projections patterns between visual cortical areas and determined which projection is “feed-forward” and which projection is “feedback”. By analyzing the relative fraction of feed-forward and feedback projections, the researchers define the hierarchical level to the cortices.

Aside from hierarchical organization, an influential view, two stream hypothesis (Goodale and Milner, 1992; Goodale and Westwood, 2004; Gallivan and Goodale, 2018) proposed that there are

separate visual pathways for perception and action: the ventral stream is oriented towards identifying and recognizing objects, while the dorsal stream oriented towards driving visually guided actions (pick-up the object). In this thesis, the focus of our investigation is the ventral pathway in the primate visual system, which is more related to visual object perception.

Ventral Hierarchy. The ventral pathway started from the primary visual cortex (V1), and went through V2, V4, and posterior, central, anterior subregions of inferotemporal cortex (IT), organized in a hierarchical fashion. Through anatomy, we knew that the visual information went through a series of feed-forward processing stages with feedback and recurrent in each stage. So how does the visual information get represented and transformed at these stages? This is a driving question for this thesis.

Basic properties of visual neurons. Temporally, going up the visual hierarchy, the neurons respond to stimuli with longer latencies which is in line with the feedforward information transmission. Spatially, visual neurons have receptive fields (RF), i.e. they respond to stimulation of a localized region in the retina. This receptive field location changed relatively smoothly along the visual cortical surface forming the retinotopic map. The receptive field size increases with eccentricity (i.e. angular distance to the center of gaze). Namely the receptive fields located at the center of gaze are smaller than those located at peripheral space. Going up the visual hierarchy, at the same eccentricity, the receptive field size increases, i.e. the neurons are integrating more information corresponding to a larger size of space (Gattass, Gross and Sandell, 1981; Gattass, Sousa and Gross, 1988; Freeman and Simoncelli, 2011). Within the RF, neurons selectively respond to certain visual patterns with high firing rates, and others to low firing rates. These varying firing rates i.e. neural activation depending on the input, form the neural representation of the visual world. Thus, to find how the neural activation and the visual stimuli correspond to each other is the key question for studying visual neural code.

Much has been discussed regarding the visual code within the ventral stream. In order to position this thesis appropriately, we'd like to briefly review how our predecessors have studied the visual system. We will see that the stimuli space we used to study the visual system largely defined the progress we made in understanding it.

I.2 Stimuli Sampling Strategies for Studying Visual Neural Code

To study any sensory system, the basic paradigm is to apply stimuli to the system and measure neuronal responses (Adrian, 1919; Barlow, 1953; Kuffler, 1953). However, the visual system is faced with an immense input space with an infinite kind of stimuli. To make it concrete, let's assume a tiny RGB image of 64 pixels wide. Regarding the image as a vector space, then its dimensionality is 12288. Even if we discretize the value of each pixel, and allow 256 levels for each color pixel, then the number of possible images is $256^{12288} \approx 10^{29592}$. Comparing to which, the number of atoms in the universe ($\sim 10^{82}$) appears negligible. Due to the continuous and high-dimensional nature of images, there is little hope to exhaust the space of images. So, given a limited budget of stimuli that we can test, an important question every vision researcher faced is, what kind of images should we use as stimuli?

Researchers have tackled this problem from two fronts. On the one front, to reduce the dimensionality of the image space by better understanding the structure of natural images. It's obvious that random combination of pixels does not constitute meaningful images: pixels need to suffice certain statistical relationships to appear natural (Simoncelli and Olshausen, 2001; Freeman and Simoncelli, 2011). Thus, the space of natural images has a much smaller dimensionality than the number of all pixels. This effort is instantiated by the study of natural image distribution and generative model of images, which we'll briefly review in Section I.3.

On the other front, researchers also need to sample their stimuli "wisely" based on their understanding about the visual system. We categorize the approach for sampling stimuli into two general families 1) static sampling based on heuristics, 2) adaptive sampling based on neural feedbacks. In the static sampling approaches, the researchers pre-select a fixed set of stimuli, according to some heuristics or based on some hypothesis they wanted to test (Pasupathy and Connor, 1999; Hegdé and Van Essen, 2004, 2006, 2007; Yau *et al.*, 2012). In the adaptive sampling approach, the researchers sample the stimuli iteratively and choose new stimuli based on the recorded neuronal responses in a feedback loop. In the following subsections, we will review a few classic works belonging to these two camps and analyze their underlying designs and analysis.

I.2.1 Heuristic-based Static Stimuli Sampling

For the study of visual systems, what kind of stimuli spaces have people developed? In our understanding, there are two types of principles guiding their heuristic sampling, one is parametric sampling; the other is ethological relevance. Many classic works tend to use simple stimuli parametrized by a set of parameters θ, and then they vary the parameter systematically to test neuronal code of the parameter θ. The other type of work is to choose natural images that may be ethologically relevant, for example, monkey and human faces, hands, animate and inanimate objects, natural scenes. Here, we will review some classic image spaces and the logic behind their development (Figure I-1).

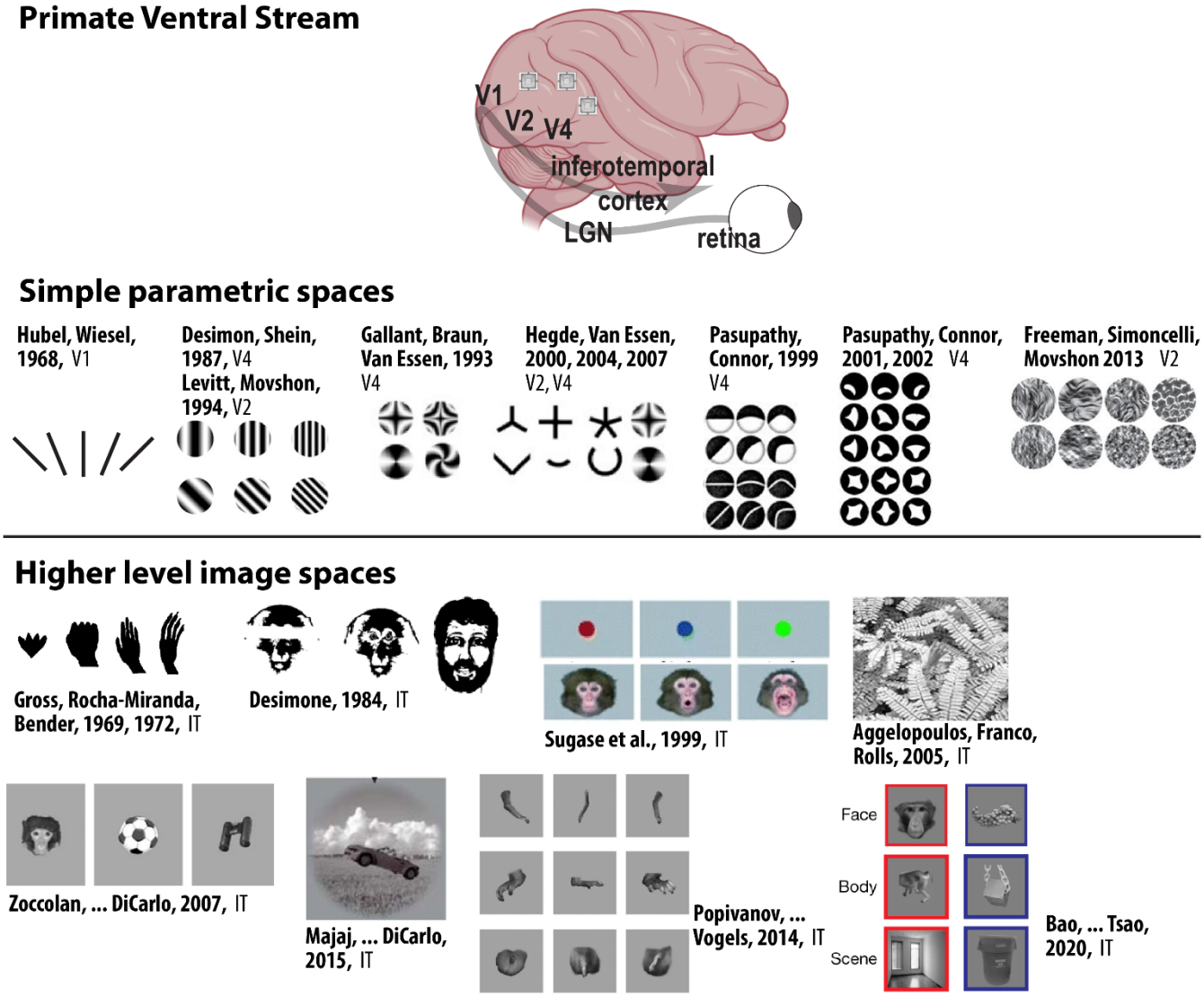


Figure I-1 Schematics of primate ventral stream, and samples from classic image space. Simple parametric shape spaces, and Higher level ethological relevant image spaces.

Since the foundational work of Hubel and Wiesel (Hubel and Wiesel, 1959, 1968), oriented bar and grating has been a prominent choice for visual stimuli. These stimuli have a few tunable parameters and have proved to be effective for many visual cortical neurons across visual area (V1, V2, V4) and across species (cats, macaques).

One direct step forward is to modify the gratings or construct more complex stimuli from them. In 1993, Gallant, Braun and van Essen used other variants of gratings, e.g. polar or hyperbolic grating for

V4 neurons (Gallant, Braun and Van Essen, 1993). Later, Hegdé and van Essen used these non-cartesian gratings and shapes constructed from line segments and curves (e.g. tri-stars, cross, star, angle, arc, circle), to compare the visual neural code in V2 and V4 (Hegdé and Van Essen, 2004, 2006, 2007).

Similarly, another step forward is to study simple geometric motifs, e.g. curves and contours. For example, Pasupathy and Connor conducted a systematic study of V4 neural code with curve and angles elements parametrized by curvature and angle size (Pasupathy and Connor, 1999); further, they connected these angular curve elements into a closed 2d contour with various number of projections (Pasupathy and Connor, 2001, 2002) and studied the single neuron and population coding of 2d shape. As a side note, when they extended this work to study neural code for 3d shapes, the immense stimuli space inspired them to use adaptive stimuli sampling method, which we will review in the next section (Sec. I.2.2) (Yamane *et al.*, 2008a).

In Movshon and Simoncelli lab, instead of constructing stimuli from simple motifs, they reasoned deeper about image statistics. If we applied orientation filter to images, the first order statistics (mean) of filter response defined the overall distribution of oriented edges. They found that, the second order statistics (correlation) of filter response defined the naturalistic textures (Freeman and Simoncelli, 2011). By matching the second order statistics to different texture photos, they can synthesize different texture families with varying degrees of naturalness. By contrasting these synthetic texture with those with distorted 2nd order statistics, they revealed the difference of neural code between V1 and V2 cortex (Freeman *et al.*, 2013; Ziemba *et al.*, 2016, 2019).

For higher order vision, in many cases, a different kind of heuristic has been applied (Figure I-1 lower). Since the very early study of inferotemporal cortex in primates (Gross, Bender and Rocha-Miranda, 1969), researchers have found that it's hard to drive their responses with simple geometric

stimuli. In the 1969 paper to determine the receptive fields of IT neurons, the authors commented "*One major difference between the properties of striate and prestriate neurons and those of inferotemporal neurons was that the responses of the latter tended to be less clear and thus their receptive fields more laborious to determine. There are two possibilities that may account for this difference. The first is that by largely confining the stimuli to bars, edges, rectangles, and circles we may never have found the "best" stimulus for each unit…*" This initial observation of response to complex but ethologically relevant natural images has inspired many follow-up works studying inferotemporal cortex. For example, researchers have experimented with photographs of faces of primate and human (Wallis and Rolls, 1997; Sugase *et al.*, 1999; Tsao *et al.*, 2003, 2006), animate and inanimate objects (Hung *et al.*, 2005; Kreiman *et al.*, 2006), natural scene videos (Baddeley *et al.*, 1997), and objects embedded in natural scene backgrounds (Rolls, Aggelopoulos and Zheng, 2003; Aggelopoulos, Franco and Rolls, 2005; Majaj *et al.*, 2015) etc. Using these high-level stimuli, this approach has facilitated the discovery of face patches in primates, and the finding that neural representation in the inferotemporal cortex supports rapid classification of objects. However, as the objects reside in discrete categories, they can only be morphed locally by changing shape, location, orientation etc., but they are not parametrized in a global space. Thus, in contrast to the parametric spaces mentioned above, we usually don't have a continuous tuning curve for objects and the results are regarded as "categorical selectivity" for objects (but see (Freedman *et al.*, 2003)). The different paradigms of stimuli sampling for lower order cortices and higher order cortices create this gap of knowledge, which motivated our study of neural code on continuous image manifolds throughout ventral stream in this thesis (Chapter IV, V and VI).

To summarize, the classic approach of visual neuroscience uses stimuli spaces such as shapes combining simpler features, textures that represent higher order image statistics, and ethologically relevant images.

Table I-1 Classic stimuli spaces used for studying neural codes of visual cortical areas.

Paper	Stimuli space	Target Visual area
(Hubel and Wiesel, 1968)	Moving black bar	V1, macaque
(Desimone and Schein, 1987)	Bars (varying length, width, orientation, and polarity of contrast); Sinusoidal gratings (varying spatial frequency, phase, orientation, and overall size).	V4, macaque
(Gallant, Braun and Van Essen, 1993)	Polar, Hyperbolic, and Cartesian sinusoidal gratings	V4, macaque
(Levitt, Kiper and Movshon, 1994)	Drifting sinusoidal grating, with color	V2, macaque
(Pasupathy and Connor, 1999)	Angular and curve element	V4, macaque
(Pasupathy and Connor, 2001, 2002)	Contour of closed 2d shapes (constructed by angular elements)	V4, macaque
(Hegdé and Van Essen, 2004, 2006, 2007)	Cartesian, polar or hyperbolic grating; Geometric shapes formed by line segments and curves (e.g. tri-stars, cross, star, angle, arc, circle)	V2, V4, macaque
(Freeman *et al.*, 2013; Ziemba *et al.*, 2016)	Synthetic texture with matched 2^{nd} order statistics	V2, macaque
(Gross, Bender and Rocha-Miranda, 1969;	Moving bars; Random cut out shapes (including hand shapes)	Inferotemporal cortex, macaque

Gross, Rocha-Miranda and Bender, 1972)		
(Desimone *et al.*, 1984)	Patterns with irregular edges; images of human hand or hand shape; macaque and human faces	Inferotemporal cortex, macaque
(Sugase *et al.*, 1999)	Human and monkey faces (varying identity, facial expression); Geometric shapes (varying shape and color)	Inferotemporal cortex, macaque
(Baddeley *et al.*, 1997)	Natural scenes videos	V1 and Inferotemporal cortex, macaque
(Rolls, Aggelopoulos and Zheng, 2003; Aggelopoulos, Franco and Rolls, 2005)	Objects placed on gray or complex natural scene backgrounds	Inferotemporal cortex, macaque
(Zoccolan *et al.*, 2007)	Isolated objects on gray backgrounds; local morphs of objects; identity preserving transformation of objects.	Inferotemporal cortex, macaque
(Majaj *et al.*, 2015)	64 3D objects rendered on 10 naturalistic backgrounds (640 images in total)	Inferotemporal cortex, macaque
(Tsao *et al.*, 2003)	Faces, bodies, hands, fruits, technological objects and grid-scrambled technological objects	V4, TE, macaque (fMRI)
(Kornblith *et al.*, 2013)	Rendered 3d indoor scene. (With other face, objects, textures)	lateral place patch (LPP), medial parahippocampal gyrus (MPP), macaque
(Popivanov *et al.*, 2014; Popivanov, Schyns and Vogels, 2016)	Body parts stimuli (with other face and body of monkey and human, object, animal etc.)	midSTS (fMRI-defined) body patch, macaque

I.2.2 Closed-loop Visual Neuroscience: Model-free

In contrast to sampling stimuli based on the tradition of field or heuristic, an alternative way is to let neurons help us choose what stimuli should we use. Using the heuristic methods, the whole field of visual neuroscience has a loop of improvement: one research paper reporting successful stimuli, and then other labs adopting and modifying them in their next experiments to better stimulate the neurons. Using feedback from neurons directly, this loop of improvement is closed in a single experiment, thus we call these approaches closed-loop visual neuroscience. We will review the development of this paradigm in this subsection (I.2.2) and the next one (I.2.3).

The idea of using neuronal or perceptual response to guide stimuli selection has a long history in sensory neuroscience and psychophysics, and it has become increasingly popular in the study of visual neural code in the last five years. In sensory neuroscience, this approach has been originally known under

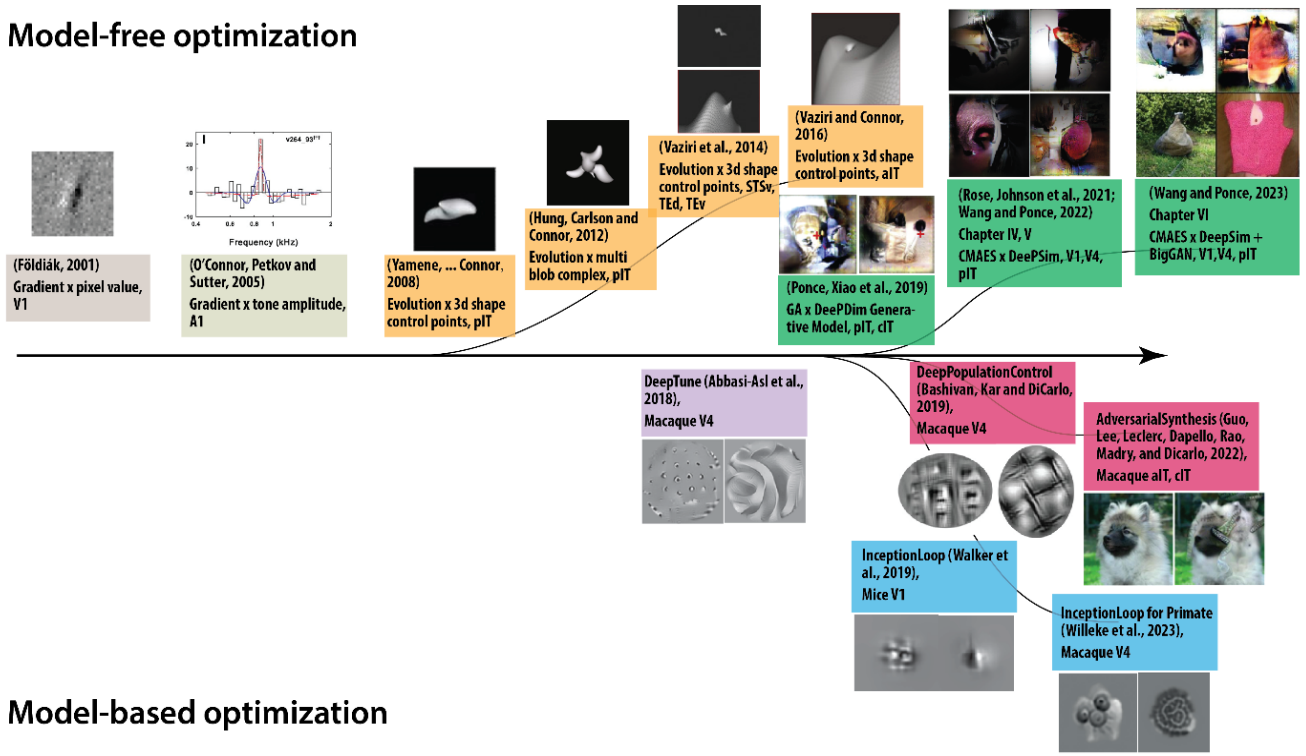


Figure I-2 **Historical timeline of closed-loop visual neuroscience,** and the position of the thesis research.

the name of *adaptive stimulus optimization* or *adaptive sampling* (Benda *et al.*, 2007; DiMattina and Zhang, 2013). The problem of sampling stimuli can be formulated as a problem of active learning or optimal experimental design, etc., thus its theoretical foundation can be found in the field of statistics and machine learning. In our work, we have also borrowed tools and ideas from the field of evolutionary computing and gradient-free optimization. Here I'd like to review the historical development of this approach (Figure I-2).

In an early effort (Földiák, 2001), Földiák utilized gradient-ascent algorithm to directly optimize the pixels of an image stimulus for V1 neurons in macaques (with experiments conducted in Movshon lab at NYU). Obviously, the analytical gradient from neuron to the pixels was not available. So, in their method, the gradient was estimated by a stochastic approximation. They presented multiple stimuli with randomly perturbed pixels $\mathbf{x}_i + \delta \boldsymbol{x}_{ij}$ around a reference stimulus $\mathbf{x}_i$ and then correlated the neuronal response r_{ij} to the perturbations $\delta \boldsymbol{x}_{ij}$. This weighted averaged direction formed an estimation of the gradient direction. Then the reference stimulus is updated along this direction, as following.

$$\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \alpha_i \sum_j \left(r_{ij} - \overline{r_i}\right) \delta \mathbf{x}_{ij} \qquad \text{(I-1)}$$

After proving this concept, they found multiple runs of optimization would end up with grating-like patterns with different phases, which is consistent with the classic result of phase invariants in complex cells. Note that, the stochastic method to approximate gradient (Eq. 1) has influenced many works down the line, and it shared the same spirits with the evolutionary strategy algorithms we developed in Chapter III.

This idea was soon adapted into the study of auditory system (O'Connor, Petkov and Sutter, 2005). The authors optimized auditory stimuli for neurons recorded in primary auditory cortex (A1) in

macaques. They used multi-tone composite stimuli, which were parametrized by the amplitudes in 24-36 tones, and they used a similar algorithm as above to optimize the stimuli (Földiák, 2001). They perturbed the stimuli along some random directions $\delta\mathbf{x}_{ij}$; and then estimated the gradient direction by weighted averaging these perturbation directions using the normalized neuronal firing rate $\frac{(r_{ij}-\bar{r}_i)}{\Delta r_{max}}$, and then modified the stimuli along this estimated gradient direction.

$$\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \alpha_i \sum_j \frac{(r_{ij}-\bar{r}_i)}{\Delta r_{max}} \delta\mathbf{x}_{ij} \qquad \text{(I-2)}$$

The major modification from Eq. I-1 is the additional normalization terms to accommodate different dynamic range of neurons. This modification is similar in spirit to evolutionary algorithms, e.g. CMA-ES which was also invariant to scale of response.

For a parametric stimuli space, researchers usually systematically test all combinations of parameters, or first choose good values along some parameter directions (e.g. orientation) and then test the neuronal tuning to other parameters of interest. However, the stimuli space grew exponentially with the number of dimensions, so exhaustive sampling becomes virtually impossible for spaces with tens to hundreds of dimensions. In this case, the adaptive sampling approach becomes necessary for studying the neuronal encoding in these vast stimuli spaces.

To tackle this challenge, the Connor lab resorted to the adaptive sampling strategy to study the neuronal coding of a series of geometric stimuli spaces in three-dimensional space (Yamane *et al.*, 2008a; Hung, Carlson and Connor, 2012; Vaziri *et al.*, 2014; Vaziri and Connor, 2016). Here we reviewed their seminal work of 2008 (Yamane *et al.*, 2008a), to show how the adaptive sampling method could be used answer visual neuroscientific questions. In this paper, they studied the encoding of closed three-

dimensional shapes in higher visual cortex (namely central and anterior IT), by applying the evolutionary algorithm to parametric 3d shapes. These shapes were parametrized by a grid of control points for NURBS (Non-uniform rational B-spline). Then the shapes were rendered with shading and disparity using a computer graphics engine. For the evolution algorithm, its objective was not simply to maximize the neuronal firing rate, but to keep samples from different ranges of neuronal response including the high response and lower responses. In the end, this method yielded a portfolio of stimuli spanning a larger dynamic range of neuronal responses, which facilitated the follow-up modelling and analysis of neuronal tuning and selectivity. Using this specific image parametrization and adaptive algorithm, they successfully navigated the high-dimensional 3d shape space. As they have noted, this approach allowed them to spend more "stimuli budget" in the active domain of the neuron instead of in the null-response domain; further it can reveal some initially unknown tuning parameters. By changing ways of rendering the shape (e.g. with or without shading, texture, with or without stereo display), they found that the depth cues such as disparity or shading were necessary for the three-dim tuning of the neuron. By testing tuning to other parameters e.g. lighting direction, rotation, translation and depth, they found the neural code for 3d shape was separable from the code of other dimensions. Finally, they used a graph-convolution like subunit model to fit the neuronal response and attribute the key features of the stimuli. Via population analysis of these fit models, they found the tuning peaks were more biased towards high curvature parts (ridge and spiky peaks) of three-dim shapes.

This work demonstrated an exemplary workflow that tackled the challenges for adaptive stimuli sampling paradigm. Since the stimuli sampling is largely driven by neurons online, 1) the stimuli parameters are not sampled regularly during evolution, 2) the stimuli set differs for different neurons in the population, 3) for the same neuron, different runs of evolution results in different sets of stimuli.

Thus, to tackle **challenge 1**, it's usually necessary to do *post hoc* tuning experiments to test the neural tuning for other parameters (e.g. position, orientation). As individual neurons are tested with different sets of stimuli (**challenge 2**), to understand the key factor that drives the neuronal firing, computational modelling is usually necessary to dissect and attribute the key part of the stimuli. Further, since the neurons from a population are tested with different stimuli (**challenge 2**), in order to draw conclusions about population, usually its necessary to compare and pool fitted parameters of the model. Finally, as evolution process has intrinsic stochasticity, it's good to perform multiple runs of Evolution to confirm the neural tuning (**challenge 3**). As this thesis used evolutionary approach to study neural code, we shared the same challenges, and tackled them with our approches (Chapter IV, Chapter V, Chapter VI).

Since then, they have used similar evolutionary algorithm and more complex shape parametrization to study the neural code for complex shape constructed from blobs (Hung, Carlson and Connor, 2012), and the neural code for smaller-scale object shape versus larger-scale environmental geometry (Vaziri *et al.*, 2014; Vaziri and Connor, 2016).

Even though this line of work in Conner lab developed more and more elaborate ways of parametrizing shapes, rendered three-d shapes on black background are still a tiny fraction of all possible images. The abstract art of Picasso cannot be expressed by the projection of a 3d shape. So, what's a more general way of parametrizing all 2d images? This parametrization needs to be both expressive and compact --- or the search will not be feasible. Since 2014, researchers have leveraged deep neural networks to learn from natural image statistics and to synthesize images. A prominent way is Generative adversarial network or GAN. These models can map a vector to images, which can be regarded as an advanced way of parametrizing images. (For a more extensive review, see Sec. I.3)

Thus, using GAN as the image space, one can probe the neuronal tuning on an even broader space than that of 3d shapes. Leveraging this advancement, (Ponce, Xiao *et al.*, 2019) combined the genetic search algorithm with pre-trained deep image generator (Dosovitskiy and Brox, 2016a), which was showed to be highly expressive. The image generator maps a 4096d vector to an image, and the genetic algorithm performed optimization in the 4096d latent space. Using this method, the authors can synthesize highly activating stimuli for neurons in posterior IT and central IT. These highly activating images (prototypes) looked intriguingly like primate faces or fragments of objects, and the highest activating natural image also looked like the prototype, and the distance to the prototype can be used to predict the response to natural images. This work formed the foundation for my thesis.

Through this history, we can see this paradigm develops along a few directions. The image parametrization evolved from pixel level parametrization to more compact 3d geometric parametrization and deep neural parametrization, and the optimization procedure evolved from approximated gradient ascent to genetic algorithms. We catalogue this line of research along the axes of optimizers, parametrization, target cortical region in the Table 2. In this framework, my thesis work contributed to this line of research in multiple fronts, (1) understanding the geometric structure of the image space in the deep image generator (Chapter II), (2) benchmarking and developing evolutionary optimizers leveraging the geometry of image manifold (Chapter III), (3) using the optimization tool to characterize the landscape of neural tuning (Chapter IV), (4) extending the tool to multiple image parametrization (object based vs pattern based), and compare their differential alignment to neural tuning (Chapter VI).

Table I-2 Synopsis of the history of adaptive stimuli optimization in sensory neuroscience.

Paper	Optimizer	Stimuli Parametrization	Target cortical area

(Földiák, 2001)	Gradient-ascent; gradient was estimated by correlating response to perturbations	Pixels	V1, macaque
(O'Connor, Petkov and Sutter, 2005)	Gradient-ascent; gradient was estimated by correlating response to perturbations	Amplitude of tones in a multi-tone complex sound	A1, macaque
(Yamane *et al.*, 2008a)	Evolutionary, with local and global morph, maintain activation diversity	Control points (NURBS) of a 3d ellipsoidal shape	Anterior and central IT, macaque
(Hung, Carlson and Connor, 2012)	Evolutionary morphing, maintain activation diversity	Medial axis stimuli constructed from connecting axial components (blobs)	Anterior and central IT, macaque
(Vaziri *et al.*, 2014)	E(Wang and Carlos R. Ponce, 2022a)volutionary, maintain activation diversity	Control points (NURBS) of a smaller scale shape or a larger scale landscape	ventral superior temporal sulcus (STSv) and dorsal / ventral inferotemporal gyrus (TEd, TEv), macaque
(Vaziri and Connor, 2016)	Evolutionary, maintain activation diversity	Control points (NURBS) of a smaller scale shape or a larger scale landscape	anterior IT, macaque
(Ponce *et al.*, 2019a)	Genetic Algorithm (classic GA)	Latent vector of DeePSim generator	central IT, posterior IT, macaque
(Wang and Carlos R. Ponce, 2022b) Chapter III	GA; Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES)	Latent vector of DeePSim generator	V1/V2, V4 and posterior IT, macaque
(Rose, Johnson, Wang and Carlos R. Ponce,	CMA-ES	Latent vector of DeePSim generator	V1/V2, V4 and posterior IT, macaque

2021; Wang and Carlos R. Ponce, 2022d) Chapter IV			
(Wang and Ponce, 2023, in prep) Chapter V	CMA-ES and Hessian-CMA	Latent vectors of DeePSim generator and BigGAN generator	V1/V2, V4 and posterior IT, macaque

I.2.3 Closed-loop Visual Neuroscience: Model-based

The works we catalogued above are model-free, in the sense that during the adaptive stimuli sampling process, no model of the neuronal activity is explicitly built, and the optimization is performed directly with respect to the recorded neuronal firing rate. With the advent of powerful deep learning methods to model visual neurons in ventral stream (Yamins et al., 2014; Schrimpf et al., 2018), another thread emerged in closed-loop visual neuroscience, which is model-based stimuli optimization (Figure I-2 lower). In these works, the researchers first performed system identification which fits an image-computable model that predicts the neuronal firing rate – a digital twin using the fancy words of 2023. Then the optimal stimuli were synthesized guided by the *in silico* model of neuron instead of the biological neuron itself. This line of works is catalogued in Table 3. The key technique of this line of work is the neuronal model and the method used to synthesize the image (feature visualization). So we emphasize the difference between the methods along these two aspects.

In a foundational yet somewhat underappreciated work, DeepTune (Abbasi-Asl *et al.*, 2018), the authors trained predictive models of 71 neurons recorded from V4, and synthesized the optimal stimuli for them using gradient ascent on the predictive model. These predictive models were built by linearly regressing the neural activation onto the activations of a hidden layer of a pretrained convolutional neural

network (CNN) (layer 2, 3, 4 of AlexNet, VGG, GoogleNet), with L1 or L2 penalty (LASSO or Ridge). Note that, to counter the bias of individual model, they ensembled the 18 models to predict the neuron. Further, they used gradient from the ensemble to synthesize best stimuli for the neuron. Note that, they added total variation and l_p norm of the image as regularization terms during the image synthesis, to reduce the high-frequency artifacts. This will be a consistent theme threading the works doing model-based optimization, and we will see more techniques for regularizing the high frequency noise during feature visualization (Table I-3).

In 2019, two studies came out back-to-back, both applying similar methods to the visual cortices of macaques and mice.

The Inception Loop from Tolias lab (Walker *et al.*, 2019) used this paradigm to study the tuning of neurons in mice V1 at scale. They used 5000 samples from the standard machine learning dataset ImageNet (Deng *et al.*, 2009), as the stimuli to the mice and the training image for the predictive model. Note that, their neuron predictive model is specially built to account for several known physiological effect of mice vision. They used a relatively shallow architecture, a three-layer convolutional neural network as the **Backbone** shared by all neurons. On top of this CNN, a **Readout** head extracts features from a specific spatial location, i.e. the hypothetical spatial receptive field of the neuron. Since mice cannot fixate, their center of gaze could shift due to head or body movement. The authors creatively trained a **Shifter** that predicts overall gaze-shifts which are used to shift the readout location for all neuron readouts. Further, since motion and arousal are known to modulate the firing rate of visual neurons in mice, the authors trained a small neural network (**Modulator**) based on running speed and pupil size to predict this gain modulation factor. They trained this neuron prediction system on a large population of neurons recorded via 2-photon imaging. Using this model, they synthesized the most

exciting images (MEI) for each neuron which can activate those neurons with high specificity. Most recently, the Tolias lab adapted this model-based closed-loop paradigm to study neuron population in macaque V4, recorded with silicon probes (Willeke *et al.*, 2023). Compared to the prior work, this paper used a much simpler neuronal model, which is just a CNN backbone and a readout head. The backbone was layer3 from ResNet50 adversarial trained on ImageNet, and the readout head spatially average feature from a location. Here, the backbone was pretrained and then fixed, and the neural data was only used to fit the readout head. This simpler model is suitable for the more stable tuning functions of primate visual neurons.

In another thread, the Deep Population Control from DiCarlo lab applied this method to control neuronal population in macaque V4. In contrast to (Walker *et al.*, 2019), the authors used much fewer object images (640 object images) to train the neuron model, which is layer3 from AlexNet with a readout head. One innovation in their work is their population based optimization objective, i.e. to synthesize images that drive the population in a specific way. For example, one of their objective was to activating one neuron while suppressing all others neurons in the population. A follow-up work from DiCarlo lab (Guo *et al.*, 2022), used similar approach to synthesize stimuli for IT neurons. This time their objective was to synthesize "adversarial" images for *in vivo* neurons and compare their robustness with CNN units. For this objective, they used projected gradient descent to find smallest perturbations on natural images that can maximally increase or decrease neuronal firing rate. As a result, they found the IT neurons may already be as robust as current adversarial robust models.

We catalogued this family of approaches along the dimension of training data, model architecture, and regularization techniques for feature visualization in Table I-3. We can find a few trends from this history. In terms of the readout head, it progresses from a linear readout in DeepTune (Abbasi-Asl *et al.*,

2018) to some spatial localized readout in all the later works. DeepTune regressed the neuronal activations onto the whole activation tensors in a hidden layer, while all the following works average features from local part of the tensor and then regress neuron activation on it (e.g. where-what factorized convolution (Klindt *et al.*, 2017), Gaussian spatial readout head (Lurz *et al.*, 2021)). It seems that spatially localized readout saves parameters and it's better suited for visual neurons with receptive fields. Another trend is that, we can see more and more regularization techniques are employed during feature visualization. Further, the more recent works started to use Adversarial Robust CNN as backbones. Both of these are related to how we can remove adversarial artifacts from the visualization (Olah, Mordvintsev and Schubert, 2017), which we will explain in detail below.

Table I-3 Synopsis of Model-based stimuli optimization approaches.

Paper	Training Data for Neuron Model	Model Architecture	Feature Visualization Regularization Technique	Target Neuron
DeepTune (Abbasi-Asl *et al.*, 2018)	4000-12000 static natural images, grayscale, 2 repeats	**Backbone**, layer 2,3,4 of pretrained AlexNet, VGG, GoogleNet; **Readout** by linear regression on all features with L1 or L2 regularization (Lasso or Ridge).	Gradient ascent Ensemble of 18 models; Total variation and l_p -norm regularization ($p = 6$).	V4, macaque
DeepPopulationControl (Bashivan, Kar and DiCarlo, 2019)	640 object images (Majaj *et al.*, 2015)	**Backbone** AlexNet custom trained on ImageNet, Layer 3; Retinae transform front end; **Readout** Linear readout by where-what factorized weight.	Gradient ascent (700 steps) Total variation regularization; Random spatial jittering at each step; Normalize gradient; Pixel value clipping.	V4, macaque

(Guo *et al.*, 2022)	1000 imagenet images	**Backbone**, ResNet50 Adversarially trained (L2 $\epsilon = 2$) on ImageNet, Layer 4; **Readout** Linear readout by where-what factorized weight	Projected gradient descent (250 steps); simulated annealing with restart; 100 independent runs	Anterior & central IT, macaque
InceptionLoop (Walker *et al.*, 2019)	5100 ImageNet (ILSVRC2012) images, grayscale. (100 images were repeated 10 times)	**Backbone**, 3-layer CNN backbone shared across population; **Readout** layer from a specific location and linearly recombine the features; a **Shifter** to shift readout location accounting for eye movement; **Modulator** to account for gain modulation from brain state.	Gradient ascent Ensemble of four identical models trained with different initialization; Gaussian blur the image after each step; Low pass filter the gradient. $G(\omega_x, \omega_y) = \frac{1}{(2\pi)^2}\left(\omega_x^2+\omega_y^2\right)^{-\alpha}$	V1, mice
InceptionLoop for primates **(Willeke *et al.*, 2023)**	9000-12075 ImageNet (ILSVRC2012) images, grayscale	**Backbone**, ResNet50 Adversarial trained ($\epsilon = 0.1$) on ImageNet, Layer 3; **Readout** Gaussian spatial readout (Lurz *et al.*, 2021), Linear nonlinear model with ELU +1 nonlinearity	Gradient ascent (1000 steps) Ensemble of five models trained with different initialization (best 5 out of 10); Gaussian white noise initialization; Gaussian blur at each step; bound maximum energy (L2 norm) of image; pixel value clipping. Optimize a transparency mask to keep the most informative part only.	V4, macaque

I.2.4 Comparison of Model-based and Model-free Adaptive Stimuli Selection

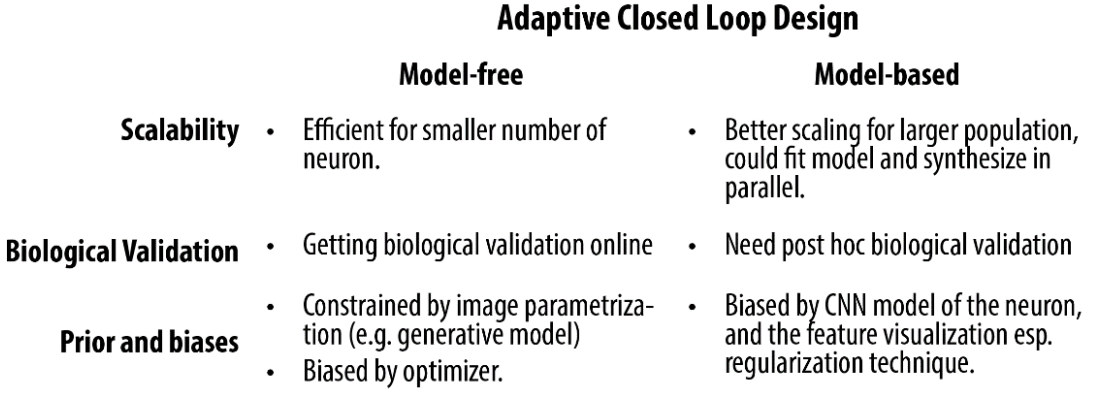


Figure I-3 Comparison of Model-free and Model-based stimuli selection approach.

In the last two subsections, we reviewed the two families of adaptive stimuli sampling approaches, here we will compare their pros and cons in detail. Interestingly, the difference between these two families of approaches is parallel to the two camps of methods in reinforcement learning (RL) or control: the model-free reinforcement learning (e.g. policy gradient, Q learning) versus model-based reinforcement learning (e.g. control and planning). So, their advantages and disadvantages are also analogous to those of the model-based versus model-free RL.

Online vs *post hoc* validation In model-free stimuli sampling, the optimizer directly interacts with the response of neural system (i.e. the real environment in the RL sense). Thus, during the online experiments, any increase in the neuronal firing rate is instantaneously validated to be a real biological effect. However, there is some fear that the recording itself is not stable. To control the recording condition, usually some fixed sets of stimuli are presented repeatedly as reference. In contrast, for model-based method, the optimizer directly optimizes the activation of the computational model (i.e. the simulated environment in RL sense), thus the proposed images need to be validated in the next session

post hoc. For this reason, many works using model-based approach need two recording sessions in two days to complete the cycle (Bashivan, Kar and DiCarlo, 2019; Walker *et al.*, 2019).

Scalability The two camps of methods have interesting tradeoffs in their efficiency and scalability. Online optimization (model-free) paradigm has the limitation that it is not parallelizable – where optimal stimulus can only be synthesized one at a time; in contrast, for model-based sampling, after collecting population responses to the training set and fitting the models for whole population, the optimal stimuli can be synthesized offline for the whole population in parallel processing. Thus, for higher channel count recordings, model-based stimuli sampling *scales better* (Willeke *et al.*, 2023). In contrast, for fewer neurons, model-free stimuli sampling methods can yield results faster online.

Prior and Biases For any system that leverages AI algorithms, there are priors and inductive biases involved, so where are they? For model-free methods, the generative image model and the optimizer are the major source of bias. In model-free approaches, usually some generative models of images are used: works from Connor lab (Yamane *et al.*, 2008a) used the geometric parametrization of 3d shapes; works from Ponce lab (Ponce *et al.*, 2019a) used image parametrization by pretrained deep image generator. These image spaces and parametrizations constitute a form of prior and biases, e.g. some images are easier to generate from the generative model and some other images are impossible to generate. In some sense, the generator represents the hypotheses of researchers about the tuning and function of neurons. The parametrization that suits the neuronal tuning will be easier to optimize, and vice versa. We explicitly tested the alignment of neuronal tuning and different generative image manifolds in Chapter V.

Further, since the model-free method needs to perform optimization online, the hyperparameters of optimizer and the initialization also introduce some prior and biases to the system. However, since the experimenters usually have limited time with the subject (recorded neuron), it's harder to benchmark

different optimizers and their biases online. We have tried to characterize these *in vivo* and *in silico* in Chapter III.

For model-based stimuli sampling paradigms, the prior and biases come from the model and the feature visualization algorithm. Similar to model-based control, any limitation or defect of the model itself will affect the synthesized images and the further biological interpretation. In this case, since a deep convolutional neural network (CNN) is used, the distribution of features in the CNN (instantiated as the weights) affects the visualized features; more importantly, its *adversarial vulnerability* will create adversarial artifacts that affect and deteriorate the feature visualization. To counter the model bias problem, usually multiple CNN models are ensembled to create a less biased model of the neuron, averaging out individual biases (Abbasi-Asl *et al.*, 2018; Walker *et al.*, 2019). To tackle the high-frequency artifacts created by adversarial vulnerability, the feature visualization process is usually regularized with multiple techniques. These techniques include, adding regularization loss in optimization total variation loss, l_p norm loss; random spatial jittering the image during optimization; Gaussian blurring the image; Gaussian smoothing the gradient (see Table I-3). Further, in response to this problem, many recent works used adversarial trained neural networks (e.g. ResNet50-robust) as the backbone (Guo *et al.*, 2022; Willeke *et al.*, 2023), which were trained to be less vulnerable to adversarial attack and could also create better feature visualization (Madry *et al.*, 2017; Santurkar *et al.*, 2019).

I.3 Natural Image Priors and Generative Models

Beyond the research focused on sampling natural images, another branch of study that has impacted this thesis is concerned with understanding the structures inherent in natural images.

The task of visual system is to represent and analyze the visual input in nature, and it has been widely conceived that the tuning properties of visual neurons are deeply rooted in the statistics of the natural

world (Barlow H., 1961). As nicely stated by (Field, 1987), "*an efficient sensory system should match its analyzers to the nature of the signals it processes.*" So, ever since the discovery of Gabor-like V1 receptive field by David and Hubel, understanding the statistical structure of natural images and using it to study neural tuning has been a long tradition in visual neuroscience. As you would see, this tradition influenced the computer vision community and the deep generative models nowadays. For the field of computer science and signal processing, understanding the statistics of natural images is the key to data compression (e.g. JPEG algorithm), image denoising, super resolution and building computer vision systems. To reproduce the known statistical structure of images is the key to generative modelling of images. Thus, this topic is of joint interest to both neuroscientists and computer scientists.

Not all combinations of pixels look like natural images**,** and natural images can be regarded as a much narrower distribution among all possible pixel arrays $\mathbb{R}^{3\times H\times W}$. The support of this distribution can be called "natural image manifold" which is used throughout the thesis. This term is not mathematically exact, but it majorly captures two intuitions about the distribution of natural images, 1) the sense of low dimensionality, as the valid natural images tend to be lower dimensional than all the pixels. 2) the fact that locally not all directions of changing the image are equal. Some directions transform the image in a valid way ("on manifold" directions), while some other directions distort the image non-sensibly ("off manifold" directions). As this thesis focuses on understanding neuronal tuning on this natural image manifold, it's helpful to review some fundamental works for understanding this manifold.

I.3.1 Early Descriptive Image Statistics

What are some characteristics of this distribution? In (Field, 1987), they investigated the spectral statistics of photographs. In their study, they found that, the spectra of natural photos are distinct from noise: the amplitudes fall off as an inverse function of spatial frequency $1/f$, in contrast, the white noise

images have a flat spectrum. In other words, more power and variance are concentrated in lower frequency than higher frequency for natural images. Since then, multiple works have reported more detailed structure of this $1/f$ spectrum. (van der Schaaf and van Hateren, 1996) visualized the power on the frequency plane and found natural images have more power along horizonal and vertical orientations. In the seminal work of (Simoncelli and Olshausen, 2001), they analyzed the color distribution, correlation length, spatiotemporal power spectra of natural images, etc. These early works to characterize natural image statistics are summarized by an excellent review (Srivastava *et al.*, 2003). These general statistical characteristics form the constraints of what we regarded as natural image.

How are image statistics related to higher level information? (Torralba and Oliva, 2003) analyzed the spectra of natural images from different categories, they found many high-level information e.g. semantic categories (face, scene, object), natural versus man-made environment, the distance of viewing objects, all have their distinct spectral signatures. For example, the natural scenes usually have more isotropic power distribution across orientation; while the man-made scenes usually have higher power concentrated on horizontal and vertical directions. It's conceivable that this statistical information can be leveraged by brain and machine to distinguish between high level image category.

I.3.2 Early Generative Image Models

If we really understand the distribution of natural images, we should be able to sample from the distribution and synthesize them. One intuitive idea is to generate images such that they conform to the statistical signatures above. For example, the $1/f$ spectrum can be used to synthesize so-called "pink-noise" images. But these images lack spatial structure. Fourier transform of the whole image obliterates the spatial structure, in contrast, the wavelet transform retains spatial structures in addition to the spectral structure. Using wavelet transforms, (Simoncelli and Portilla, 1998; Portilla and Simoncelli, 2000)

characterized the joint distribution of wavelet coefficients for texture images. Further, they found that by optimizing images such that their wavelet coefficients follow these distributions, it is enough to generate naturalistic textures. In this regard, textures are defined by the 2nd order information of the wavelet coefficient. This line of research laid the foundation for their later characterization of V2 (Freeman and Simoncelli, 2011; Freeman *et al.*, 2013; Ziemba *et al.*, 2016), which showed that the V2 neurons are sensitive to these 2nd order statistics while V1 neurons are not.

In a classic line of work termed "dead leaves model" (Lee, Mumford and Huang, 2001), they focused on the occlusion aspect of image forming. Namely, the natural scenes are usually formed by objects of varying sizes occluding each other. By simulating this occlusion process, they could build images from overlaying shapes. Other statistical models have also been used. Given the spatial nature of images, (Zhu, Wu and Mumford, 1998) used Markov Random Field (MRF) to learn spatial statistics of natural textures. (Zoran and Weiss, 2012) used Gaussian Mixture Model (GMM) to learn the image statistics of image patches and reported state-of-the-art likelihood, denoising and generation results (in 2012). Further, they proposed a link between GMM and "dead-leaf model".

To summarize the theme, the early generative image models usually characterize the distribution of features in certain feature space (e.g. Fourier, wavelet), with simpler parametric distributions e.g. Gaussian, Laplace, or Gaussian Mixture. The intuition is, with a proper feature transformation, the distribution of features will become simpler (more independent) and amenable to parametric fitting. The concept of transforming images to a space where it could be modelled parametrically remains relevant even for generative models today.

I.3.3 Deep Generative Models

Before deep learning era, the image features are usually defined by human intuition (e.g. wavelet features). Since the advent of deep learning revolution around 2012, deep neural networks were found to be an automatic way of learning useful features from data. Thus, given this universal function approximator and feature learning machine, how can we leverage them to model images? There are several families of generative models, to name a few, Variational Autoencoder (VAE), Generative adversarial networks (GAN), Energy-based model and Normalizing Flow, and the latest candidate of Diffusion model. We will introduce the principles of some models and their relevance to this thesis.

Variational autoencoder was proposed in 2013 (Kingma and Welling, 2022), which was a simple adaptation of autoencoder. It learns an encoder that maps images to lower dimensional latent vectors and a decoder that maps latents to images. It has the simple objective of image reconstruction and a regularization loss to make the distribution in latent space more Gaussian-like. These models are stable to train, efficient to sample, and they have an invertible map from image space to latent space. However, in practice, they tend to generate blurrier images.

Generative Adversarial Network Around 2014, the idea of GAN was proposed by Ian Goodfellow (Goodfellow *et al.*, 2020). The idea is to let a neural network (discriminator) be the judge of another (generator). The discriminator learns to tell the generated images and real images apart, and the generator learns to generate fake images against the discriminator. The generator maps a latent vector following a simple distribution (Gaussian) to an image. In geometric terms, the generator learns to warp a simple low dimensional distribution to the image space to match the target distribution, while the discriminator provides adaptive guidance that pushes the warped distribution to match the target one as close as possible. Throughout the years, numerous technical developments were gradually built into GAN to make

its training stable, sample efficient and better scaling to large images (Radford, Metz and Chintala, 2015; Karras *et al.*, 2017, 2020; Brock, Donahue and Simonyan, 2018; Donahue and Simonyan, 2019; Karras, Laine and Aila, 2019). For a long time (2014 ~ 2021), GAN has been the univocal king in the realm of generative image models in terms of image quality and speed of sampling.

Further, as the GAN has a compact latent space, it is suitable as a model of natural image manifold. As a mapping from lower dimensional space to image space, the generator of the GAN can be regarded as a parametrization of manifold. For example, (Zhu *et al.*, 2016) used pretrained GAN of certain image dataset (e.g. shoes) as constraints for image editing. Using this constraint, edits on images can be projected back onto the image manifold, enforcing it to be within the naturalistic distribution (e.g. still being a proper shoes). In our thesis, we majorly used pre-trained GAN to parameterize image manifold and analyze the neuronal tuning on this manifold. One caveat is that the GAN may not be as expressive as we want, i.e. there are certain types of images that may be hard for the generator to create. Because of this, by choosing certain GAN, we incorporate certain biases about the images.

Flow-based model The idea of normalizing flow, is to learn an invertible map between a simple distribution (Gaussian) with the target distribution in the same dimension (Dinh, Krueger and Bengio, 2015; Dinh, Sohl-Dickstein and Bengio, 2017; Kingma and Dhariwal, 2018), where the map is parametrized by a special designed neural network. With this map, we can sample from the simple distribution and then map it to the target. Using the change of variable formula, we can evaluate the probability density of generated samples exactly, which also allowed us to train the model with maximum likelihood estimation (MLE). However, the flow-based model didn't learn compact latent space, so it's less efficient for parametrizing images for our purpose, but on the other hand, it can be more expressive than GANs.

Diffusion / score-based model Finally, diffusion model can be regarded as a modern reincarnation of normalizing flow and variational autoencoder. It connects a Gaussian distribution with the target data distribution with a diffusion process. By using a neural network to learn the gradient of the diffused data distribution (score function), we can reverse the diffusion process and iteratively denoise a noise pattern into image. Recently, these models have demonstrated that they can achieve higher image quality and higher diversity than GANs (Dhariwal and Nichol, 2021). Further, the image generation process can be easily controlled by additional conditional signal e.g. natural language prompt (Rombach *et al.*, 2021). Due to these desirable properties, these models have gained immense popularity and have taken the community by storm since 2021. We will speculate on how our study could be generalized to these newer image generative models in the final chapter (Sec. VII.3.1).

In summary, we reviewed seminal works investigating the natural image distribution and generative model of images. In the thesis, we selected generative adversarial networks as the major generative image model, owing to its efficiency in sampling and its compact latent space.

I.4 Dynamics Encoding of Visual Neuron

Vision is dynamic. Even when the input images are static, the visual perception and the neuronal responses evoked by the input are dynamic. In Section I.1, I.2, we mostly treated the visual system as a static function, mapping image input to latent representations. Here, we'd like to refine this picture by inspecting the dynamic response to images. What kind of information is encoded in different time windows of the responses? This line of work has been summarized by the excellent review of (Hegdé, 2008). Here, we'll review a few seminal results that influenced our understanding.

The classic result of (Sugase *et al.*, 1999), proposed coarse to fine encoding in temporal response. By computing the mutual information of stimuli category and individual neuronal responses, they found that

the earlier responses of visual neurons in IT cortices transmit global information (broader categories, monkey vs human), while the later sustained response gradually encoded finer-grained information (different facial expressions or identities). Subsequently, their computational modelling work (Matsumoto *et al.*, 2005), proposed an attractor network to explain the observed information dynamics.

In a series of works focusing on shape encoding dynamics V1, V2 and V4, (Hegdé and Van Essen, 2004, 2006) found that, using 2d shape stimuli, the tuning sharpness of individual neurons in V1, V2 and V4 was low during the transient response (peak of firing rate), and the sharpness increased in the later response ((Hegdé and Van Essen, 2006) Fig. 4). On the population level, the initial responses of V2 and V4 neurons were more correlated, and the later response become more decorrelated or fine-grained. In geometric terms, the initial responses are lower dimensional (i.e. could be explained by fewer PCs), and the later responses are higher dimensional (i.e. needs more PCs to explain).

In the study of shape processing in V4, (Brincat and Connor, 2006) used their shape tuning model to analyze the dynamics of tuning in early and late response. They found that V4 neurons could be tuned to multiple parts of the 2d contour, and the tuning to multiple parts could be linear additive; or these multipart tuning could be nonlinear, i.e. neurons respond only when multiple parts present. There was heterogeneity among V4 neurons: some are purely linear, some are purely nonlinear, and many mixing these two types of tuning. But for all these neurons, usually the nonlinear tuning responses emerged and peaked later (peaked at 184ms post onset) than the linear tuning responses (peaked at 122ms post onset). This finding is consistent with the previous coarse-to-fine tuning result, as the nonlinear response suggests more specific selectivity or finer tuning. They further suggested a more specific computational mechanism for these dynamics, i.e. recurrent computation turning linear tuning to nonlinear tuning. Later, using a similar design, (Yau *et al.*, 2012) studied the dynamics of curvature tuning for V4 neurons. They

found that the initial responses of V4 neurons could be better explained by linear addition of tuning to individual line orientations (parts), while the later tuning is better explained by the nonlinear tuning to both orientations and the relative angle between them (whole).

In higher visual cortices (e.g. IT), the later neural responses were found to be relevant to challenging cases of object recognition (Kar *et al.*, 2019). Throughout the response dynamics, the images hard to be classified during perception could only be decoded better based on later responses than the earlier responses. Further, the responses in later time windows were harder to predict through deep feedforward neural networks models (AlexNet, VGG), and they could be better predicted by deeper CNN (ResNet, Inception) or recurrent CNN model (CorNet-S (Kubilius *et al.*, 2018)). This highlights the importance of recurrent processing for understanding the later neuronal responses.

Taking it together, this line of research suggests that through recurrence the later response to static images has more specific tuning to conjunction of features, and this more nonlinear and fine-grained tuning could be used to solve difficult object classification problems. These works form the basis for our results about the dynamics of neuronal tuning on the generative image manifold in Chapter IV and Chapter VI. In our results, we also discovered that the neuronal tuning on generative image manifold was broad at first, and then narrowed down to a sharper tuning peak. Further, we found that the tuning of later neuronal responses in IT cortex was more aligned with object-based image manifold than pattern manifold. These were consistent with the prior works we introduced.

I.5 Structure and Contribution of the Thesis

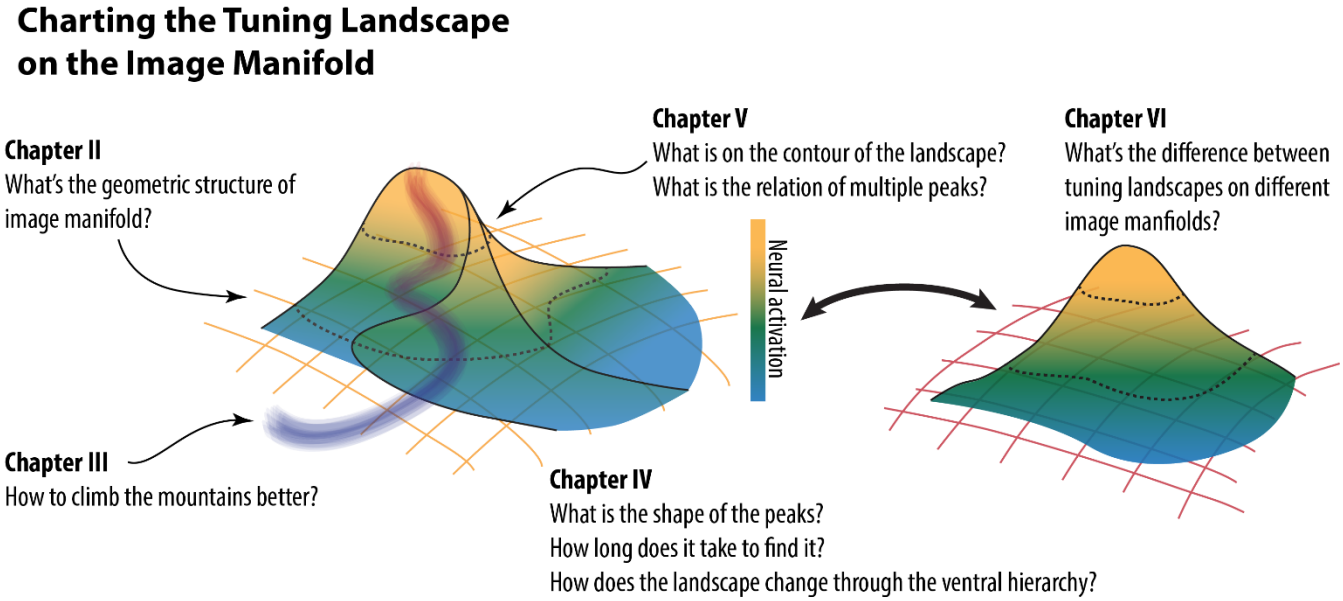


Figure I-4 Structure of the thesis and the driving questions for each chapter.

In this final part, let me illustrate the structure of the thesis research as a roadmap (**Figure** I-4). We conceptualize the natural image as a manifold and the thesis aims at characterizing the visual neural code as a landscape over this manifold. In Chapter II and III, we first develop the computational techniques to study visual neurons. To understand neural tuning on a given space, we first need to understand the structure of the image space well. In Chapter II, we developed the computational tool and mathematical framework to characterize the geometry of any generative image manifolds --- which is the stage on which neuronal tuning will be charted (published in ICLR 2021). Next, we need efficient ways to navigate this image manifold. In Chapter III, we benchmarked a series of evolutionary optimizers and developed a better gradient-free optimizer for controlling neuronal firing rate and navigating the image manifold leveraging the known geometry of the space (published in GECCO 2022). Starting Chapter IV, V and VI, we conducted a set of neurobiological experiments, to chart the tuning landscape of neurons across the ventral visual hierarchy. In Chapter IV, we used evolutionary algorithms to search for peaks of tuning functions and characterized these tuning peaks for neurons throughout the ventral stream. Using

this data, we analyzed how local and global geometric characteristics of tuning landscapes changed along the ventral stream (published in Cell Reports 2022). In Chapter V, we inspected the geometry of tuning landscapes through the lens of level sets and analyzed other high-dimensional properties of tuning peaks including the relationship between multiple peaks on the landscape (published in NeurReps 2022). In Chapter VI, we studied the alignment of ventral stream neuronal tuning and multiple image manifolds by comparing neuron-guided image evolution on a pattern-based and an object-based manifold (in preparation). In Chapter VII, we discussed the open question and next steps in the intersection of generative models and visual neuroscience.

I.6 My Work Outside the Thesis

In addition to the chapters in this thesis, I have also researched some other topics during my time in graduate school.

One less mentioned aspect in the thesis is the characteristic of the *images* themselves locating at the peaks of tuning function (*prototypes*). In (Rose, Johnson, Wang and Carlos R. Ponce, 2021), we analyzed the image statistics of the prototypes for neurons from V1 to pIT and showed the progression of complexity along the ventral hierarchy. Further, the prototypes from higher level neurons attract more than average saccade for primates, highlighting the behavioral relevance of prototype images.

Another challenge for the neuron-guided image synthesis paradigm is how to leverage the large set of irregularly sampled images and neural responses into a model of neuron to dissect its feature selectivity. We answered this question in (Wang and Carlos R. Ponce, 2022a). We built a factorized convolution model to predict the neuronal response within each *Evolution* experiment, and then perturb the model to understand the key feature that drives the neurons.

How visual representations are learned by biological systems is a grand open question that intrigues me. It's widely believed that self-supervised learning could be a potential learning scheme to train the visual system. The key idea for contrastive learning is to train the representations of different views of the same image to be similar and vice versa. So, what's a plausible way to generate multiple views? In (Wang *et al.*, 2021), we proposed a biological plausible image transformation, namely warping the image with the nonlinear mapping from retina to cortex i.e. cortical magnification. We found that this can be used to augment images to learn visual representation in self-supervised learning algorithm (SimCLR). This is like an existance proof for self-supervised learning in biological system: as the visual system makes 2-3 saccades per second, training the cortical representation of natural scene to be similar across saccades will be a potential way to train the visual system.

Further, I'm also interested in understanding generative models and the source of their creative power. In Chapter II, we dissected the latent space of generative adversarial network (GAN) with differential geometry. Recently, diffusion models have gained high popularity. In (Wang and Vastola, 2023), we derived an analytical theory for the diffusion models, that could qualitatively explain many aspect of diffusion (e.g. outlines are specified first and then the details) and quantitatively predict the early phase of actual diffusion trajectory with analytical equations.

For a full list of my work during grad school please see my webpage at scholar.harvard.edu/binxuw/publications.

Chapter II : Geometry of Natural Image Manifold in Generative Adversarial Network

Abstract

Generative adversarial networks (GANs) 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, limiting 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 key previous results related to GAN interpretability. 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 illustrate that defining the geometry of the GAN image manifold can serve as a general framework for understanding GANs.

II.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, 2020; Brock, Donahue and Simonyan, 2018; Donahue and Simonyan, 2019; Karras, Laine and Aila, 2019). 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; Pividori, Grinblat and Uzal, 2019; Ponce *et al.*, 2019a; Shen and Zhou, 2020; Wang and Carlos R. Ponce, 2022d). 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 from 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 II.2. Using this geometric framework, we introduce an efficient way to compute the metric tensor H on the image manifold in Section II.3, and empirically investigate the properties of H in various GANs in Section II.4. We explain the properties of this metric in terms of network architecture and training in Section II.5. We show that this understanding provides a unifying principle behind previous methods 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 II.6.

II.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, Kumar and Thomas Fletcher, 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, Hansen and Hauberg, 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 (28×28 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, our geometric analysis is architecture-agnostic and it's applied to modern large-scale GANs (e.g. BigGAN, StyleGAN2). Further, we extend the pixel L2 distance assumed in previous works to any differentiable distance metric.

II.3 Methods

Formulation A generative network, denoted by G, is a mapping from latent code z to image I, $G: \mathbb{R}^n \to \mathcal{I} = \mathbb{R}^{H\times W\times 3}, z \mapsto I$. Borrowing the language of Riemannian geometry, $G(z)$ parameterizes a submanifold in the image space with $z \in \mathbb{R}^n$. 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 latent codes as the distance between the images they generate, i.e. pullback the distance function to latent space through G.

$$d: \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+, \qquad d(z_1, z_2) := D\big(G(z_1), G(z_2)\big)$$

The Hessian matrix (second order partial derivative) of the 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 z_0 as a function of z, $f_{z_0}(z) = d^2(z_0, z)$. Obviously, $z = z_0$ is a local minimum of $f_{z_0}(z)$, thus $f_{z_0}(z)$ can be locally approximated by a positive semi-definite quadratic form $H(z_0)$ as in Eq. (II-2). This matrix induces an inner product and defines a vector norm, $\|v\|^2 = v^T H(z_0) v$. This squared vector norm approximates the squared image distance, $d^2(z_0, z_0 + \delta z) \approx \|\delta z\|^2 = \delta z^T H(z_0) \delta z$. 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 $\alpha_H(v) = \frac{v^T H v}{v^T v}$ the *approximate speed of image change* along v as measured by metric H.

$$d^2(z_0, z) \approx \delta z^T \frac{\partial^2 d^2(z_0,z)}{\partial z^2}|_{z_0} \delta z \qquad \text{(II-1)}$$

$$H(z_0) \coloneqq \frac{\partial^2 d^2(z_0,z)}{\partial z^2}|_{z_0} \qquad \text{(II-2)}$$

Numerical Method As defined above, the metric tensor H can be computed by doubly differentiating the squared distance function d^2. Here we use a convolutional neural network (CNN)-based distance metric, LPIPS (Zhang *et al.*, 2018), as it has been demonstrated to approximate human perceptual similarity judgements. The direct method to compute Hessian is by building a computational graph towards the gradient $g(z) = \partial_z d^2|_{z=z_0}$ and then computing the gradient towards each element in $g(z)$. This method computes H column by column, therefore its time complexity is proportional to the latent-space dimension n and the backpropagation time through this graph.

For situations when direct backpropagation is too slow (e.g. FC6GAN, StyleGAN2), we developed an approximation method to compute the major eigen-dimensions of the Hessian more efficiently. These top eigen-pairs are useful in applications like optimization and exploration; moreover, they form the best low-rank approximation to the Hessian. As we will later discover, the spectra of these Hessians have a fast decay, thus far less than n eigenvectors are required to approximate them, cf. Sec II.4.2. As a matrix, the Hessian is a linear operator, which could be defined as long as one can compute the Hessian vector product (HVP). Since the gradient to z commutes with inner product with v, HVP can be rewritten as the gradient to $v^T g$, or the directional derivative along v to the gradient $g(z)$ (Eq.2). The first form $\partial_z\left(v^T g(z)\right)$ is easy to compute in reverse-mode auto-differentiation, and the directional derivative is

easy to compute in forward-mode auto-differentiation (or finite differencing). Then, Lanczos iteration is applied to the HVP operator defined in these two ways to solve the largest eigen pairs, which can reconstruct an approximate Hessian matrix. The iterative algorithm using the two HVP definitions are termed Backward Iteration and Forward Iteration respectively. For details and efficiency comparison, see Appendix II.8.2.

$$HVP: v \mapsto Hv = \partial_z\left(v^T g(z)\right) = v^T \partial_z g(z)$$
$$\approx \frac{g(z + rv) - g(z - rv)}{2\|rv\|}$$

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

Connection to Jacobian This formulation and computation of the Riemannian metric is generic to any mapping into a metric space. Consider a mapping $\varphi(z): \mathbb{R}^n \to \mathbb{R}^M$, which could be the feature map of a layer in the GAN, or a CNN processing the generated image. We can pull back the squared L2 distance and metric from $\mathbb{R}^M$, $d^2(z_1, z_2) = \|\varphi(z_1) - \varphi(z_2)\|^2$, and define a manifold.

The metric tensor H_φ of this manifold can be derived as Hessian of d^2. Note, there is a simple relationship between the Hessian of d^2, H_φ and the Jacobian of φ, J_φ (Eq. II-3). Through this we know the eigenvalues and eigenvectors of the Hessian matrix H_φ correspond to the squared singular values and right singular vectors of the Jacobian J_φ. This allows us to examine the geometry of any representation throughout the GAN, and analyze how the geometry in the image space builds up.

$$H_\phi(z_0) = \frac{\partial^2}{\partial z^2}\frac{1}{2}\|\phi(z_0) - \phi(z)\|_2^2|_{\boldsymbol{z}_0} = J_\phi(z_0)^T J_\phi(z_0) \qquad \text{(II-3)}$$

$$v^T H_\phi(z_0) v = \left\|J_\phi(z_0)v\right\|^2, \qquad J_\phi(z_0) = \partial_z \phi(z)|_{z_0}$$

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 $\phi(I): \mathcal{I} \to \mathbb{R}^F$, then the metric tensor of the image manifold can be derived from the Jacobian of the composite of the generator and the representation map ϕ,

$$H(z) = J_{\phi \circ G}^T \cdot J_{\phi \circ G}, \qquad J_{\phi \circ G} = \partial_z \phi(G(z))$$

This connection is crucial for understanding how geometry depends on the network architecture.

II.4 Empirical Observations

Using the above method, we analyzed the geometry of the latent space of the following GANs: DCGAN (Radford, Metz and Chintala, 2015), DeePSiM / FC6GAN (Dosovitskiy and Brox, 2016b), BigGAN (Brock, Donahue and Simonyan, 2018), BigBiGAN (Donahue and Simonyan, 2019), Progressive Growing of GANs (PGGAN) (Karras *et al.*, 2017), StyleGAN 1 and 2 (Karras, Laine and Aila, 2019; Karras *et al.*, 2020) - model specifications reviewed in Sec. II.8.3. These GANs are progressively deeper and more complex, and some employ a style-based architecture instead of the conventional DCGAN architecture (e.g. StyleGAN 1,2). 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.

II.4.1 Top Eigenvectors Capture Significant Image Changes.

In differential geometry, a metric tensor H captures an infinitesimal notion of distance. To determine whether this quantity represents evident image changes, we randomly picked a latent code z_0, then computed the metric tensor $H(z_0)$ and its eigen-decomposition $H(z_0) = \sum_i \lambda_i v_i v_i^T$. Then we explored linearly [1] in the latent space along the eigenvectors $G(z_0 + \mu_i v_i)$. We found that images changed much faster when moving along top than along bottom eigenvectors, both per visual inspection and LPIPS (**Figure II-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 much more subtle pixel-wise changes. Moreover, PGGAN and StyleGANs trained on the face dataset (CelebA, FFHQ) have top eigenvectors that represent similar interpretable transforms of faces, such as head direction, sex or age (**Figure II-10**). These observations raised the possibility that top eigenvectors also captured perceptually relevant changes: we tested this directly with positive results in Sec. II.6.1.

[1] For some spaces, we used spherical linear exploration (i.e. SLERP), where we restrict the vector to a sphere of certain norm. We project v_i onto tangent space of z_0 and travel on the big circle from z_0 along v_i.

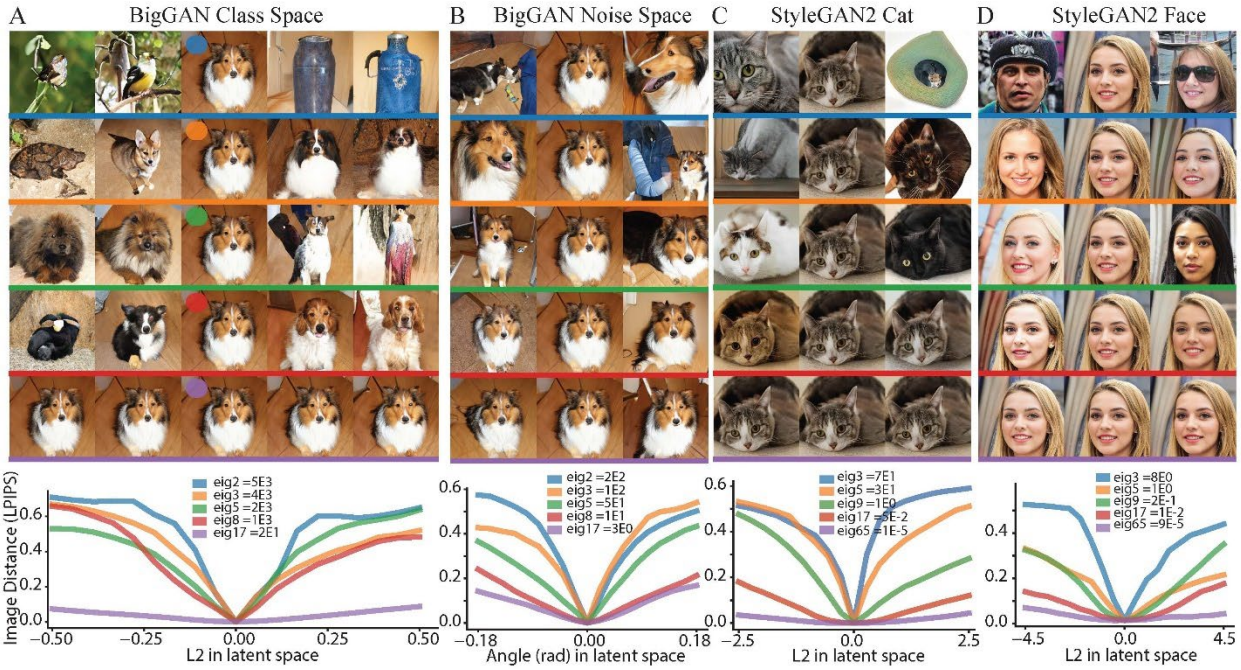


Figure II-1 Images change at different rates along top vs bottom eigenvectors. Each panel (A-D) shows perturbations around a randomly chosen reference image (center column); each row shows perturbations introduced by moving along each of five eigenvectors; each contiguous column is separated by the same 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). Eigenvector rank and associated eigenvalue labeled in the legend.

II.4.2 Spectrum Structure of GANs

To explore how eigenvalues were distributed, for each GAN, we randomly sampled 100-1000 z vectors in the latent space, used backpropagation to compute $H(z)$ and then performed the eigen-decomposition. In **Figure II-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 was responsible

for major image changes (**Table II-2**), while most dimensions induced nuanced changes (e.g. shading, background) — thus GAN latent spaces were highly anisotropic.

We found this anisotropy in every GAN we tested, which raises the question of why it has not been discussed more frequently. One possibility is that the statistical properties of high dimensionality create an illusion of isotropy. When traveling along a random direction v in latent space, the approximate rate of image change $\alpha_H(v) = \frac{v^T H v}{v^T v}$ is a weighted average of all eigenvalues as in Eq. 9. In Sec II.8.6, we show analytically that the variance of $\alpha_H(v)$ across random directions will be $2/(n + 2)$ times smaller than the variance among eigenvalues $\{\lambda_i\}$. For example, in BigGAN latent space (256 dimensions), the eigenvalues span over six orders of magnitude, while the $\alpha(v)$ for random directions has a standard deviation less than one order of magnitude (**Figure II-2**, **Figure II-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 eigen-dimensions of extremely large and small eigenvalues.

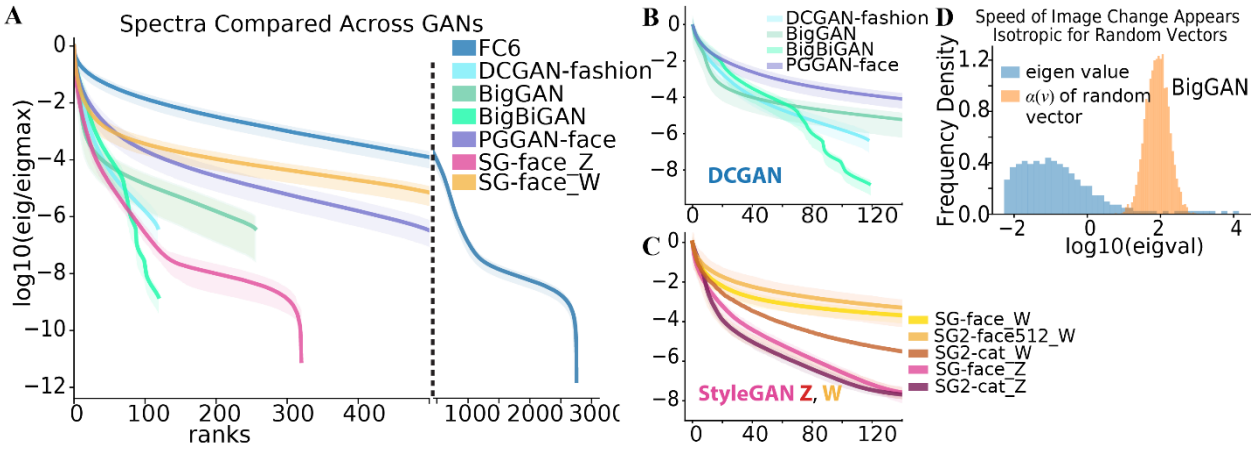


Figure II-2 Spectra of GANs, shown as a function of individual model (**A**) and types of architecture (**B, C**). DCGAN-type (green-blue), Z and W space for StyleGAN (SG1, 2) (red, orange). Lines and shaded areas show the averaged spectra and the 5-95% percentile of each eigenvalue among samples (for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

II.4.3 Global Metric Structure

Because the metric $H(z)$ describes local geometry, the next question is how it varies at different positions in the latent space. We computed the metric $H(z)$ at randomly selected z and examined their similarity using a statistic adopted from (Kornblith *et al.*, 2019). In this statistic, we applied the eigenvectors $U_i = [u_1, \ldots u_n]$ from a metric tensor H_i at position z_i to the metric tensor H_j at z_j, as $u_i^T H_j u_i$. These values formed a vector Λ_{ij}, representing the effects of metric H_j on eigenvectors of H_i. Then we computed the Pearson correlation coefficient between Λ_{ij} and the target eigenvalues, Λ_j, as $\mathrm{corr}(\Lambda_j, \Lambda_{ij})$. 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.

$$\Lambda_{ij} = \mathrm{diag}\left(U_i^T H\left(z_j\right) U_i\right)$$

$$C_{ij}^{H} = \mathrm{corr}\left(\Lambda_{ij}, \Lambda_{j}\right), \qquad C_{ij}^{Hlog} = \mathrm{corr}\left(\log\left(\Lambda_{ij}\right), \log\left(\Lambda_{j}\right)\right)$$

Using this correlation statistic, we computed the consistency of the metric tensor across hundreds of positions within each GAN latent space. As shown in **Figure II-3**C, the average correlation between eigenvalues and vHv values of two points C_{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 (**Figure II-3**D, Table II-4). 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.

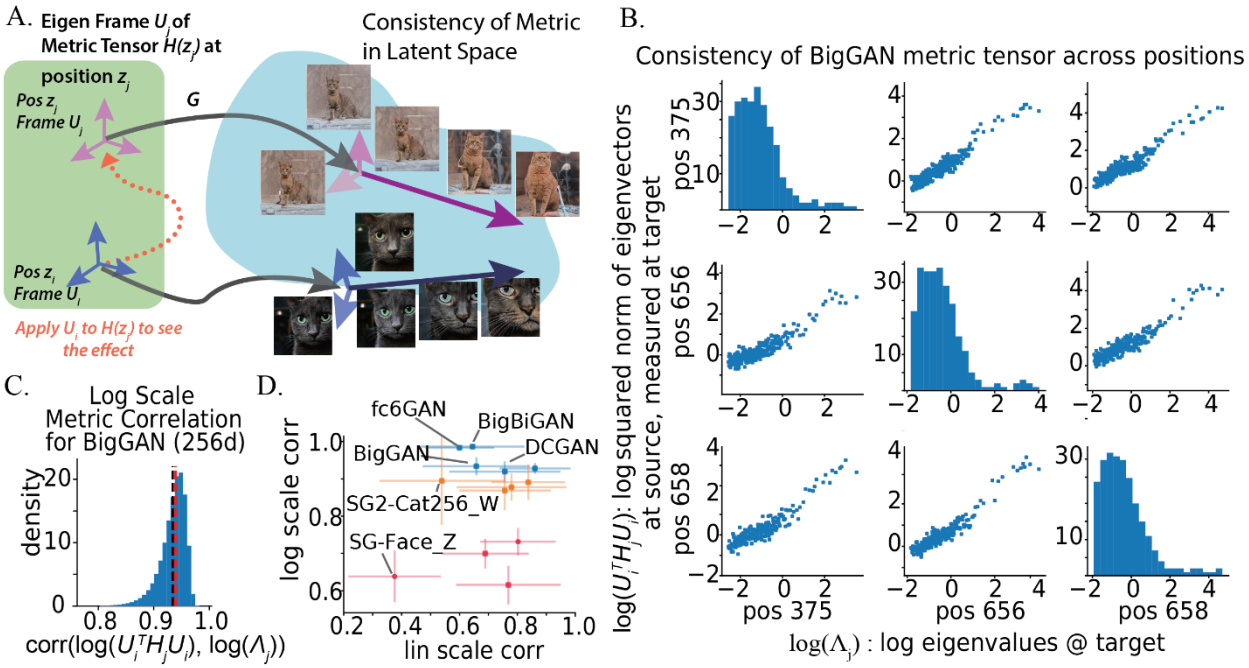


Figure II-3 Globalization of metric structure: A. Schematic of the geometric picture. In the latent space (green area), the metric eigen frames at each point (blue and violet) are mapped to transformations in image space (blue area); the length and saturation of image-space vectors represent the eigenvalue (i.e. amplification factor) of G. We show that the top eigen space 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 CHlog 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)

II.4.4 Implication of the Null Space

As the spectra have a large portion of small eigenvalues and the metric tensors are correlated in space, the bottom eigenvectors should create a global subspace, in which latent traversal will result in small or even imperceptible changes in the image. This is supported by our perceptual study, as over half of the subjects cannot see any change in image when latent vector moves in bottom eigenspace. (Sec. II.6.1). This perceptually "null" space has implications about exploration in the GAN space and interpretable axes discovery. As $G(z+v) \approx G(z)$, if one axis u encodes an interpretable transform $G(z) \to G(z+u)$, then shifting this vector by a vector in the null space v will still result in an interpretable axis $G(z) \to G(z+v+u) \approx G(z+u)$. Thus, each interpretable axis has a family of "equivalent" axes which encode

similar transforms, differing from each other by a vector in "null" space. However, adding component v in the null space will decrease the rate of image change along that axis. In this sense, the vectors using the smallest step size to achieve that transform should be the "purest" axes of the family. Further, the cosine angle between two interpretable axes may not represent the similarity of the transforms they encode. A large angle can be found between two axes of the same family but at different image traversal speed. We compared the axes from previous works in Appendix II.8.9, II.8.10 and observed that projecting out a large part of their axes did not affect the semantics it encoded (**Figure II-8**, **Figure II-9**).

II.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 $H(z) = J_{\phi \circ G}^{T} J_{\phi \circ G}$ (Eq. II-3). As the 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 (**Figure II-4** A). 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 ψ_i by applying a set of vectors onto the metric tensors of these layers H_{ψ_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 for a weight-shuffled control BigGAN: in that case, the top eigen-dimension of the first few layers was not particularly amplified on the image manifold, and vice versa (**Figure II-4** B). This shows that the amplification effect of layers becomes 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 (**Figure II-7**).

II.6 Applications

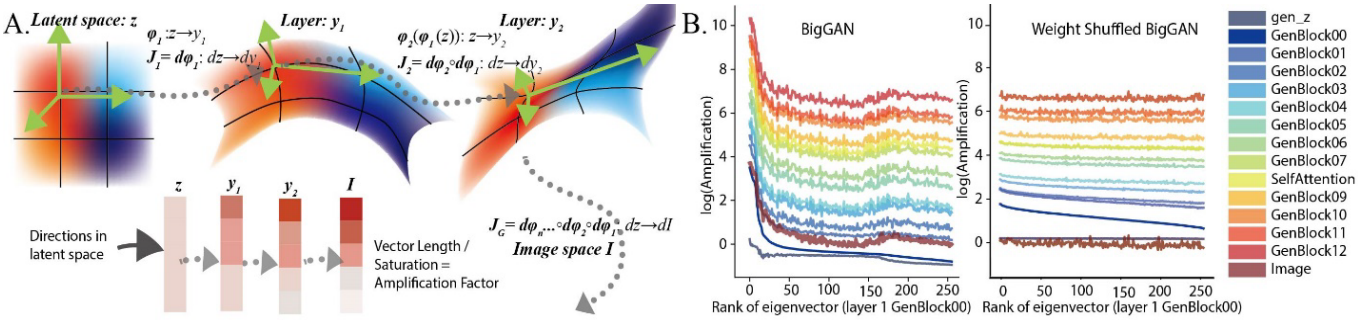


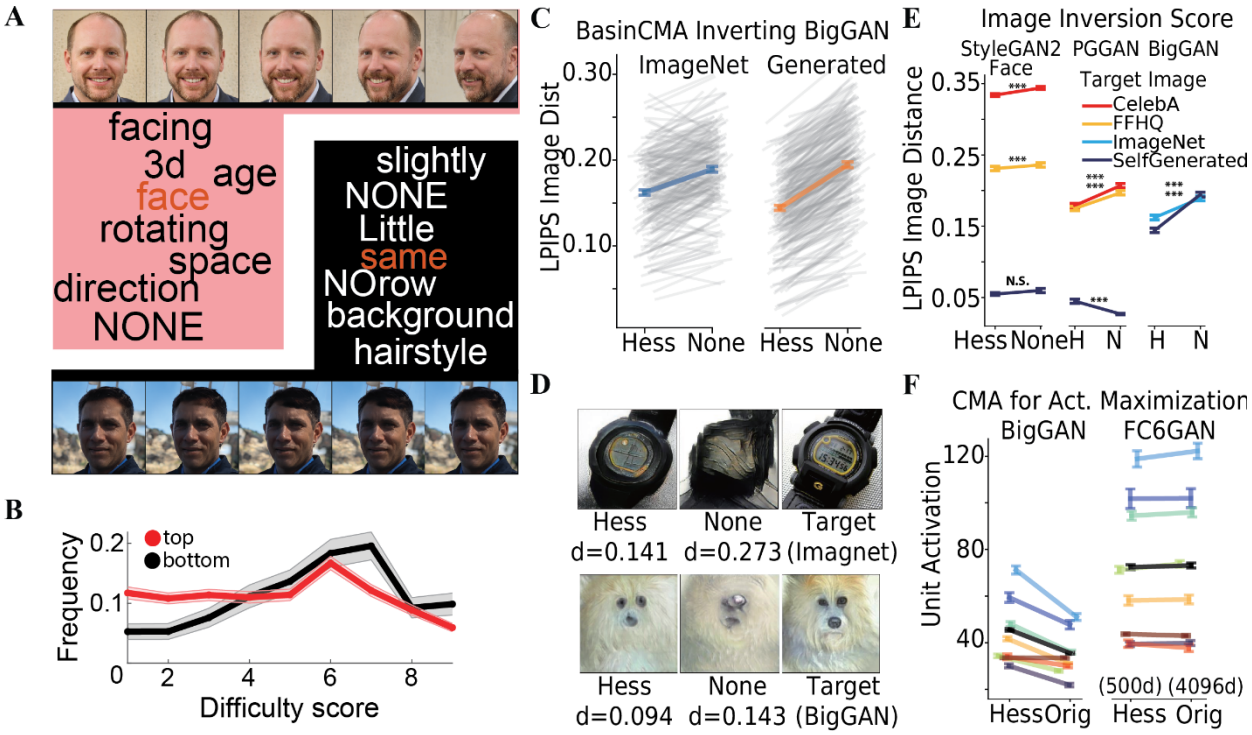
Figure II-4 Anisotropy is induced and maintained throughout the GAN architecture. A. As latent space gets warped and mapped into image space, directions in latent spaces are scaled differently by the Jacobian of the map. **B.** Amplification of eigenvectors of the metric tensor of the first conv layer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

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) finding human-interpretable axes in the latent space, 2) improving gradient- based optimizers, 3) accelerating gradient-free search.

II.6.1 Interpretable Axes Discovery

When users wish to manipulate generated images via their latent code, it is useful to reduce the number of variables needed to effectuate that 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 visualized the image changes corresponding to the top eigenvectors in BigGAN, BigBiGAN, PGGAN,

StyleGAN1,2 (**Figure II-1**). We found many of these eigenvectors appeared to capture interpretable transformations like zooming, head direction and object position, consistent across reference image.



II-5 Applications of the Riemannian metric tensor A. Perceptual properties of eigenvectors. Word cloud shows subjects' descriptions ($N = 24$) of the image transforms induced by the top- (red) vs. the bottom eigenvector (black) in StyleGAN2-Face. **B.** Distribution of difficulty scores associated with top- vs. bottom eigenvectors (red, black) across all four GANs for $N = 185$ subjects. Lines show mean frequency ± standard error (per bootstrap). **C-E.** Eigenbasis pre-conditioning improves GAN inversion. **C.** BasinCMA with eigenbasis pre-conditioning (Hess,H) outperformed a method using normal basis (None,N) in inverting ImageNet and BigGAN generated images; **D.** Examples of fitted ImageNet and BigGAN images with our Hessian BasinCMA and original method (LPIPS distance below). **E.** Results for PGGAN and StyleGAN2 inverting samples from CelebA and FFHQ. **F.** Hessian-CMAES (Hess) outperforms CMAES (Orig) in maximizing CNN activation in BigGAN, and increases sampling efficiency in FC6GAN latent space. Each line represents a different layer of optimized units in AlexNet. *** denotes paired t-test $p < 1 \times 10^{-6}$.

To test if this was apparent to people other than the authors, we conducted a study using Amazon's Mechanical Turk. We tested the perceptual properties of the axes identified by the metric tensor,

including the top 10 eigenvectors, random vectors orthogonal to the top 15d eigenspace, and bottom 10 eigenvectors. Images were generated using four different GANs (PGGAN, BigGAN noise space, StyleGAN2-Cat and -Face), and were presented to 185 participants. In each trial, five randomly sampled reference images were perturbed along a given axis, and participants were asked if they could a) perceive a change, b) indicate an estimate of its magnitude [0%-100%] c) describe a common change in their own words and how many of the five images shared this change, c) indicate how similar were the 5 image changes (consistency, score of 1-9, 9 most similar) and finally, d) state how difficult it was to describe this change (difficulty score, scale of 1-9, 9 most difficult).

Only 48.5% of the subjects reported seeing any change happen for bottom eigenvectors, while the fraction was 93.5% and 89.8% for top and orthogonal directions respectively. Further, when subjects observed some change, they reported that the image transformations induced by top eigenvectors were larger ($70.3\% \pm 0.6\%$) than those of orthogonal directions ($66.8\% \pm 0.9\%$, $p = 7.0 \times 10^{-4}$, two-sample T-test) and than those of bottom eigenvectors ($61.5\% \pm 1.6\%$, $p = 2.1 \times 10^{-10}$). This was true even though we picked a step size in the top eigenspace that was 5-10 times smaller than in the orthogonal and bottom eigenspaces. Further, subjects reported the top 10 eigenvectors had a higher mean perceptual consistency score ($6.72 \pm 0.06, n = 929$ responses) than the orthogonal ($6.42 \pm 0.09, P = 5.8 \times 10^{-3}, n = 448$ responses) and bottom eigenvectors ($6.12 \pm 0.14, P = 1.3 \times 10^{-5}, n = 242$ responses). Participants reported that the top eigenvectors were easier to interpret (4.91 ± 0.08) than the bottom eigenvectors ($5.69 \pm 0.14, p = 3.8 \times 10^{-6}$, albeit comparably to the orthogonal eigenvectors 4.82 ± 0.11, P = 0.5). Thus, overall, we conclude that the Hessian eigenvectors not only capture informative axes of image transformations, but that these were also perceptually relevant, corresponding

to similar semantic changes when applied to different reference vectors (**Figure II-11**) — axes interpretable not just in local sense, but in a global sense.

II.6.2 Improving Gradient-Based GAN Inversion.

For applications like GAN-assisted drawing and photo editing (Zhu *et al.*, 2016; Shen and Zhou, 2020), one crucial step is to find a latent code corresponding to a 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, since we find that the local Hessians are highly correlated across the latent space, we can pre-compute it once for each latent space and use the global average Hessian 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 and Ba, 2015). 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 global 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 (**Figure**

II-5). Similarly, eigenbasis reconditioning consistently improved inversion of PGGAN and StyleGAN2-Face for real image sampled from both FFHQ and CelebA using ADAM method. In short, the understanding of homogeneity and anisotropy of the latent space can improve gradient-based optimization.

II.6.3 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, when searching for activity-maximizing stimuli for neurons in primate visual cortex, or when optimizing perceptual evaluation in the user (Ponce *et al.*, 2019b; Chiu *et al.*, 2020; Xiao and Kreiman, 2020; Wang and Carlos R. Ponce, 2022b). 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 (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 to avoid overshoot. We built in this heuristic to improve CMAES, termed CMAES-Hessian. With our method, the search can be limited to the most informative directions, which should increase sampling efficiency; further, our method tunes the exploration step size in a way that is inversely proportional to the rate of image change. To test this approach, we applied our 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 FC6GAN and

BigGAN. We found that the dimension of the search space could be reduced from 4096 to 500 for FC6GAN 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 (**Figure II-5**F).

II.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).

There have been multiple efforts directed at identifying interpretable axes in latent space using unsupervised methods, including (Ramesh, Choi and LeCun, 2018; Härkönen *et al.*, 2020; Peebles *et al.*, 2020; Shen and Zhou, 2020; Voynov and Babenko, 2020). Our description of the connection between the metric tensor of the image manifold and the Jacobian matrices of intermediate layers unifies these previous results. As we have showed, the top right singular vectors of the weights (i.e. Jacobian) of the first few layers (as used in (Shen and Zhou, 2020)), correspond to the top eigenvectors of the metric tensor of the image manifold, and these usually relate to interpretable transforms. Similarly, the top principal components (PCs) 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. Likewise, (Ramesh, Choi and LeCun, 2018) also observed that the top right singular vectors of the Jacobian of G are locally disentangled. Regarding (Voynov and Babenko, 2020) and (Peebles *et al.*, 2020), we empirically compared their interpretable axes and our eigenvectors, and

found that in some of the GANs, the discovered axes have a significantly larger alignment with our top eigenspace and they are highly concentrated on individual top axes than expected from random mixing. We refer readers to Appendix II.8.9, II.8.10 and **Figure II-8**, **Figure II-9** for further comparison.

Although we have answered how this 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 and 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 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. 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; **Figure II-10**). 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.

Acknowledgments

We appreciate the conceptual and technical inspirations from Dr. Timothy Holy (WUSTL). We are grateful for the constructive discussion with Zhengdao Chen (NYU), whose pointers to the relevant literature helped launch this work. We thank Hao Sun (CUHK) in providing experience for the submission and rebuttal process. We thank friends and colleagues Yunyi Shen (UW–Madison), Lingwei Kong (ASU) and Yuxiu Shao (ENS) who read and commented on our early manuscript. This work was supported by Washington University in St. Louis, the David and Lucille Packard Foundation, the McDonnell Center for Systems Neuroscience (pre- doctoral fellowship to B.W.), and a seed grant from the Center for Brains, Minds and Machines (CBMM; funded by NSF STC award CCF-1231216).

Author contributions

B.W. conceptualized the project, designed the algorithm, developed the code base, performed the numerical experiment and analyzed the data. B.W. and C.R.P. interpreted the results. B.W. and C.R.P. designed the human MTurk task and analyzed the data. B.W. and C.R.P. prepared and revised the manuscript.

II.8 Appendix

II.8.1 Connection To Information Geometry

It is useful to compare our work to the "information geometry" (Amari, 2016) on the space of distributions. In that 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 difference function D^2 is analogous to this KL-divergence; the image $G(z)$ as parameterized by latent code z is analogous to the distribution p_θ parameterized by θ. 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.

II.8.2 Methods for Computing the Hessian

One direct way to compute the Hessian of a given scalar function (e.g. squared distance d^2 in our case), is to compute $g_{z_0}(z) = \partial_z d^2|_{z=z_0}$, create a computational graph from code z to the gradient $g_{z_0}(z)$, and back propagate from gradient vector $g_{z_0}(z)$ element by element. In this way the computational time is linear to time of a single backward pass times the latent space dimension n.

Given a large latent space or a deep network (e.g. 4096 dimensions in FC6GAN, or 512 in StyleGAN2), this method can be very slow. An efficient way is to use the Hessian vector product (HVP) operator and iterative eigenvalue algorithms like power iteration or Lancsoz iterations to solve the eigenvectors corresponding to eigenvalues of largest amplitudes. These largest eigen pairs create the best low rank approximation to the real Hessian matrix. Note that, to find the smallest amplitude eigen pairs,

inverse Hessian vector product operator is required, which is much more expensive to compute. However, as the eigenspace with the smallest eigenvalues represent directions that do not change images much, the exact eigenvector does not matter. We can just define an arbitrary basis in the "null" space orthogonal to the eigenspace with large amplitude eigenvalues.

There are two ways to construct a HVP operator: one way uses the 2nd order computational graph from z to the gradient $g(z)$ to compute HVP by back-prop, i.e. $HVP_{backward}$; the other way uses finite difference on the first-order gradient to compute HVP i.e. $HVP_{forward}$. As it does not require backpropagation, a single operation of $HVP_{forward}$ is faster than $HVP_{backward}$ but it is less accurate and takes more iterations to converge. We use the ARPACK (Lehoucq, Sorensen and Yang, 1998) implementation of the Lanczos algorithm as our iterative eigenvalue solver.

$$HVP_{backward}: v \mapsto \partial_z\left(v^T g(z)\right)$$

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

We termed the direct method Full BackProp (BP), the iterative method using $HVP_{backward}$ and HV Pforward Backward Iteration and Forward Iteration respectively. We computed the Hessian at the same z_0 using these three methods in different GANs and compared their temporal cost empirically in **Table II-1**.

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, i.e. the change in representation or image space caused by the direction, we need to push forward

the right singular vectors through the Jacobian, which is feasible through forward-mode autodiff or finite difference.

Table II-1 Computational Cost for Three Methods: Computation time is measured on a GTX 1060 GPU. The iterative method has a variable runtime which depends on the number of eigenpairs required. In this table, we all use one half of the full dimension as the eigenpaired required, which is the largest we can require using ARPACK implementation of Lanczos. Thus these numbers should be seen as an upper limit of time for Backward and Forward Iteration method. 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

II.8.3 Specification Of GAN Latent Space

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

- **DCGAN** model was 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)** was 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 was re-written and translated into Pytorch, with weights ob- tained from official page https://lmb.informatik.uni-freiburg.de/people/dosovits/code.html of (Dosovitskiy and Brox, 2016a). 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 4096d latent space. This model is highly expressive in fitting arbitrary pattern, but not particularly photorealistic.

- **BigGAN model** was obtained through Hugging Face'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** was 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 **Figure II-2**).
- **StyleGAN model** was obtained via a translation of NVIDIA's Tensorflow implementation https://github.com/rosinality/style-based-gan-pytorch. We used 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 just modulates the feature maps in the conv layers, instead of serving as a spatial input as in DCGAN, FC6GAN, PGGAN.
- **StyleGAN2 models** are obtained via a translation of NVIDIA's Tensorflow implementation https://github.com/rosinality/stylegan2-pytorch. This is an improved version of StyleGAN: it also has a network mapping the 512d Z to the 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 trained on LSUN cat dataset (Yu *et al.*, 2015) at 512 resolution.

- **WaveGAN** model is obtained from the repository https://github.com/mostafaelaraby/wavegan-pytorch/. Its architecture resembles that of DCGAN, but applied to the one dimensional wave form generation problem. We customly trained it on the wave forms of piano performance clips. It has a 100d latent space, using Gaussian as latent space distribution.

II.8.4 Quantification of Power Distribution in Spectra

Table II-2 Quantification of Spectra anisotropy (Models marked with † are audio wave form generating GANs using different distance metric function.)

	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
WaveGAN MSE†	100	17	38	74	94
WaveGAN STFT†	100	2	9	19	42

We quantified the anisotropy of the space, i.e. the low rankness of the metric tensor in Table II-2. To do this, we computed the number of eigenvalues needed to account for the 0.99, 0.999, 0.9999, 0.99999 fraction of the sum of all eigenvalues. This can be thought of as the minimal number of dimensions

needed 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.

There are a few interesting patterns we noticed in this table. For BigGAN, we noted that the class subspace is more low-ranked than the noise subspace, i.e. fewer directions could account for most of the changes across object classes than within classes. For StyleGAN 1 and 2, we analyzed the geometry of Z space and W space separately, and found that in all the models the metric in W space is significantly more isotropic i.e. less low rank than Z space. Thus, in this regard, the $z \rightarrow w$ mapping warped the spherical distribution in Z space to an elongated one in W space, but the mapping from W space to image is still more isotropic.

II.8.5 Geometric structure is robust to the image distance metric

Our work used the LPIPS distance metric to compute the Riemannian metric tensor. To determine how much of the results depended on this choice of metric, we computed the metric tensor at the same hidden vector using different image distance functions, specifically a) structural similarity index measure (SSIM) and b) Mean Squared Error (MSE) in pixel space, which do not depend on CNN. We computed the Hessian at 100 random sampled vectors in BigGAN, Progressive Growing GAN (Face), StyleGAN2 (Face 256), using MSE, SSIM and LPIPS, and then compared their Hessian spectra and eigenvectors. We found that the entry-wise correlation across the Hessian matrices (d^2 elements) ranged from $[0.94 - 0.99]$. The correlation of eigenvalue spectra ranged from $[0.987 - 0.995]$. Measuring Hessian similarity using the statistics we developed C^{Hlog} and C^{Hlin} resulted in correlations concentrated at 0.99. Thus, we found that the Hessian matrices and their spectra were highly correlated across image distance metrics, and that the Hessian matrices had a similar effect on the eigenvectors of each other.

Table II-3 **Comparison of Hessian Computed with Different Sample Dissimilarity Metric d** We experimented with BigGAN, PGGAN and StyleGAN2 (FFHQ 256 resolution), we compared the Hessian computed by LPIPS and that by MSE or SSIM at 100 latent vectors. The statistics we showed are: element-wise Hessian correlation (H corr), eigen spectra correlation (eigval corr), the Hessian consistency measure CHlin and CHlog. The linear regression between the log spectra of LPIPS the and that of the alternative (SSIM or MSE) yields the slope (slope) and intercept (intercept). The mean and standard deviation (in parenthesis) of the the 100 statistics are shown. In the last row, WaveGAN† is an audio generating GAN. We measured the similarity of the Hessian us- ing MSE of wave forms (MSE) and MSE of spectrogram (STFT) as dissimilarity metric. Hessian computed using these two measures are less similar to each other.

		H corr	Eigval corr	C^{Hlin}	C^{Hlog}	slope	intercept
BigGAN	MSE	0.973(0.033)	0.995(0.008)	0.996(0.014)	0.999(0.001)	1.01(0.03)	1.47(0.22)
	SSIM	0.988(0.012)	0.997(0.003)	0.999(0.002)	0.999(0.001)	1.06(0.02)	-0.40(0.07)
PGGAN	MSE	0.938(0.047)	0.987(0.016)	0.987(0.038)	0.999(0.000)	1.02(0.02)	1.75(0.19)
	SSIM	0.970(0.020)	0.993(0.009)	0.997(0.007)	0.999(0.000)	1.06(0.01)	-0.23(0.11)
StyleGAN2	MSE	0.945(0.035)	0.989(0.011)	0.990(0.020)	0.987(0.011)	1.13(0.16)	1.32(0.34)
	SSIM	0.945(0.047)	0.987(0.015)	0.988(0.027)	0.991(0.008)	1.08(0.18)	-0.17(0.26)
WaveGAN†	STFT	0.529(0.229)	0.892(0.088)	0.661(0.313)	0.938(0.062)	1.09(0.05)	4.67(0.30)

One major difference across Hessians from different image distance metric was evident in the scale of the eigenvalues. We regressed the log Hessian spectra induced by SSIM or MSE onto the log Hessian spectrum induced by LPIPS, and found the intercepts of the regression were usually not zeros (Table II-3). This result showed different image distance metrics exhibit different "unit" or intrinsic scale, although they all factored out the same structure in the GAN.

This result is contextualized by Section II.5. As equation (II-3) showed, the Riemannian metric or Hessian of the generator manifold is the inner product matrix of the Jacobian of the representational map. The effect of image distance metric on the Riemannian metric is to add a few more terms on top of the chain of Jacobians. The Jacobian terms from the layers of generator seem to have a larger effect than the final terms coming from the image distance metric.

Note that this does not mean that the choice of sample space distance function is irrelevant. Going beyond image generation, when applying our method to an audio generating GAN, the WaveGAN, we

found that the choice of distance function in the space of sound waves will substantially affect the Hessian obtained. We used the MSE of wave forms and MSE of spectrograms (denoted by STFT) to compute metric tensor of that sound wave manifold. We found the element-wise Hessian correlation between these is around 0.53, while the other Hessian similarity metric are also lower than the counterparts for BigGAN, PGGAN and StyleGAN2 (Table II-3). We think the MSE of spectrograms is a more perceptually relevant distance metric of sound waves than MSE of wave forms, and this difference is reflected in the geometry they induced i.e. anisotropy and homogeneity (Table II-2, Table II-4). Thus, when and how the sample space distance metric will affect the geometry of generative model still requires more development to be answered.

II.8.6 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 v, $\alpha(v) = \frac{v^T H v}{v^T v}$, and $v \sim \mathcal{N}(0, I)$. This random variable represents the effect of the symmetric matrix on random directions. Note that a change of variable using the orthogonal matrix $w = U^T v$ will not change the distribution $w \sim \mathcal{N}(0, I)$. Through this the random variable α could be rewritten as

$$\alpha(v) = \frac{v^T H v}{v^T v} = \sum_i \frac{\lambda_i \left\| u_i^T v \right\|_2^2}{\|v\|_2^2}, \qquad v \sim \mathcal{N}(0, I)$$

$$= \frac{\sum_i \lambda_i w_i^2}{\sum_i w_i^2}, \qquad w_i \sim \mathcal{N}(0, I), i = 1, \dots d$$

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

$$c_i \coloneqq w_i^2 / \sum_i w_i^2$$

As each element in w is distributed as i.i.d. unit normal, w2 is distributed as i.i.d. chi-square distribution of parameter 1. $w_i^2 \sim \chi^2(1) \sim \Gamma\left(\frac{1}{2}, \frac{1}{2}\right)$. Thus, the normalized weights $c_i = w_i^2 / \sum_i w_i^2$ conform to a Dirichlet distribution $c \sim Dirichlet(1/2\,, 1/2\,, \ldots 1/2)$. Through moment formula of Dirichlet distribution, it is straightforward to compute the mean and variance of $\alpha(w) = c^T \lambda$

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

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

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

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 (PPL) used in (Karras *et al.*, 2020) doesn't really result in a flat spectrum. In our notation, the PPL regularization minimizes $\mathbb{E}_z \mathbb{E}_v \|\alpha_z(v) - b\|^2$, which is to minimize the variance of the distribution of $\alpha(v)$ with v sampled from normal distribution and z sampled from latent distribution. The global minimizer for this regularization is indeed a mapping G with flat spectrum, i.e. an isometry up to some scaling. However, we can see through

our derivation and **Figure II-6** that even for highly anisotropic spectrum, this variance will not be very large. Thus, we should expect a limited effect of this regularization.

II.8.7 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, since eigenvectors with close-by eigenvalues are likely to rotate into each other when computing eigen-decomposition (Van der Sluis and Van der Vorst, 1987). However, the statistics should be invariant to this eigenvector mixing, and also take the eigenvalues into account. In our statistics, we applied the eigenvectors $U_1 = [u_{1,i}, \dots]$ of one matrix H_1 to the other H_2, i.e. examined the length of these vectors as measured by the other matrix as the metric tensor, $\alpha_{H_2}(u_{1,i}) = u_{1,i}^T H_2 u_{1,i}$. Recall that $\alpha_{H_1}(u_{1,i}) = u_{1,i}^T H_1 u_{1,i} = \lambda_{1,i}$ and $\alpha_{H_2}(u_{2,i}) = u_{2,i}^T H_2 u_{2,i} = \lambda_{2,i}$,i. If the eigenvectors fall in the eigenspace of the same eigenvalue in H_2, then $\alpha_{H_2}(u_{1,i})$ will equal the eigenvalue, and thus our statistic is invariant to rotation within the eigenspace. If the eigenvectors are totally uncorrelated, the resulting $\alpha_{H_2}(u_{1,i})$ will distribute like that of random vectors as in **Figure II-6**. As we compute the correlation between the eigenvalue $\lambda_{2,i} = \alpha_{H_2}(u_{2,i})$ and the $\alpha_{H_2}(u_{1,i})$, we summarize the similarity of action of H_2 on eigenvectors U_2 and U_1.

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 yield NaN, as there is no variation in the spectra to be correlated. Similarly, if the metric tensor has a more isotropic spectrum, then it will generally have a smaller correlation with others. In that sense, spectral anisotropy also plays a role in our

statistics for metric similarity or homogeneity of the manifold. In all the GAN spaces 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 there are different ways to average symmetric positive definite matrices (SPSD), induced by different measures of distance in the space of SPSD (Yuan *et al.*, 2020). Here we

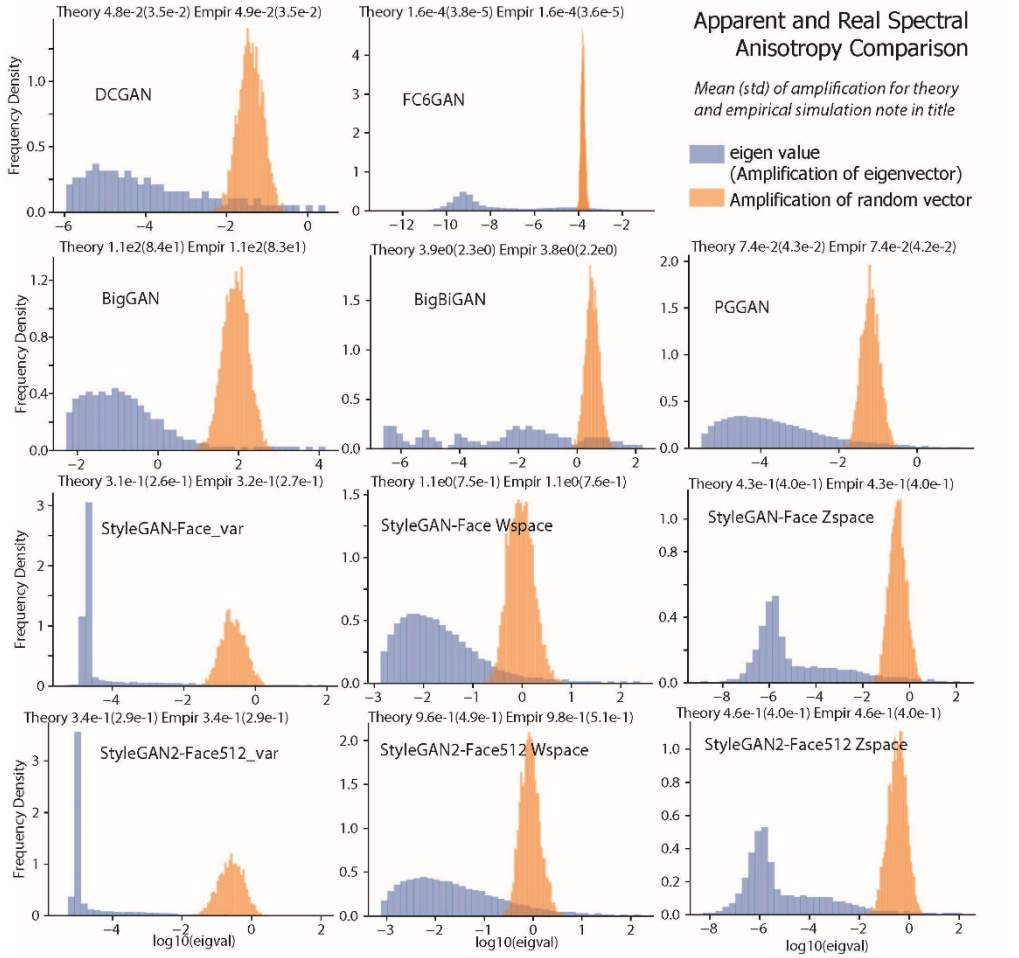


Figure II-6 Spectral Histogram compared to Apparent Anistropy in Different GAN models (FC6GAN, DCGAN, BigGAN, BigBiGAN, PGGAN, StyleGAN, StyleGAN2) The apparent speed of image change $\boldsymbol{\alpha}(\boldsymbol{v})$ has much smaller variability than the variability in the whole spectra. Eq. 13 can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

picked the simplest one to estimate the global Hessian in the latent space: averaging the metric tensors element by element.

II.8.8 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 the weight tensor from each layer 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].

Table II-4 Quantification of Manifold Homogeneity by metric consistency $\boldsymbol{C^H}$ on log scale and linear scale. Same data generating **Figure II-3** D. Models marked with † are audio wave form generating GANs.

	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
WaveGAN MSE†	0.906	0.022	0.776	0.139
WaveGAN STFT†	0.809	0.096	0.467	0.285

Table II-5 Hessian preconditioning improves GAN inversion The mean and standard error of fitting score (minimum LPIPS distance to target using 4 random initial vectors) are presented in the table, which is the

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

same data generating **Figure II-5**. For each GAN, target dataset pair, 200-300 different target images are used.

		Hessian		None	
GAN	Target Image	Mean	SEM	Mean	SEM
StyleGAN1024	CelebA	0.334	0.002	0.344	0.002
StyleGAN1024	FFHQ	0.231	0.003	0.236	0.003
StyleGAN1024	GANgen	0.055	0.002	0.060	0.003
PGGAN	CelebA	0.179	0.003	0.207	0.003
PGGAN	FFHQ	0.175	0.003	0.197	0.003
PGGAN	GANgen	0.045	0.003	0.027	0.001
BigGAN	ImageNet	0.162	0.003	0.189	0.003
BigGAN	GANgen	0.144	0.003	0.195	0.003

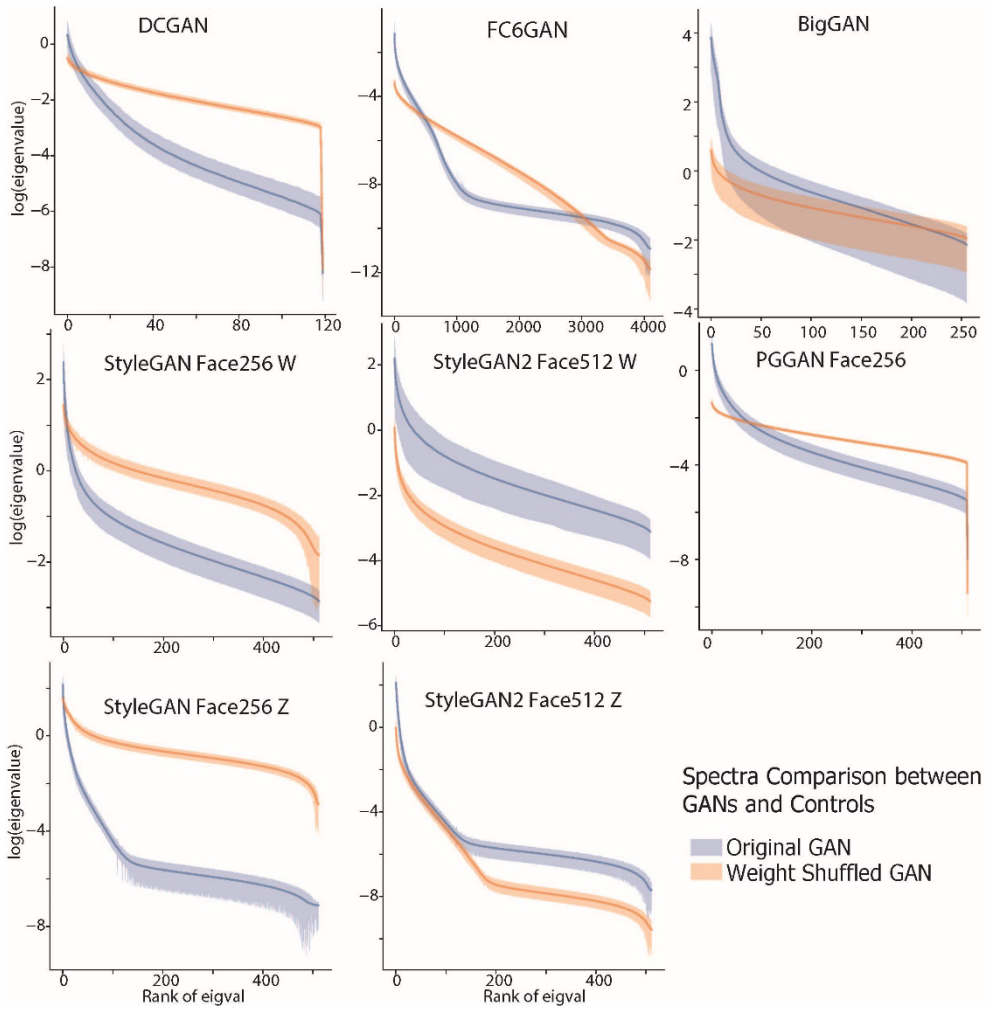


Figure II-7 **Comparing Spectra of Original and Weight Shuffled GANs**: Most Shuffled GAN shows a slower spectral decay and a smaller maximum eigenvalue.

In Figure II-7, we showed that the shuffled controls exhibited flatter spectra and smaller top eigenvalues. 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}) 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 or nan, 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.

II.8.9 Detailed Comparison to Previous Unsupervised Ways to Discover Interpretable Axes

Here we compare the axes discovered by our method with those from a previous approach. Specifically, we applied our method to the same pre-trained GANs used in (Voynov and Babenko, 2020), comparing the axes they discovered versus our Hessian structure. Although this method follows a quite different approach compared to ours and those of (Härkönen *et al.*, 2020; Shen and Zhou, 2020), we thought it would be interesting to determine if the interpretable axes discovered in their approach had a relationship with the Hessian structure defined above. If so, this could serve as independent confirmation of the effectiveness of both types of approaches.

In their work, for each generative network G, two additional models were simultaneously trained to discover interpretable axes: a "deformator" M and a "latent shift predictor" P. The "deformator" M learned to propose vectors $\{v_i\}$ to alter the image, which were used to create images pairs $\big(G(z), G(z+v_i)\big)$ using random reference vectors z; the "latent shift predictor" P took in the images

pairs and learned to classify the direction in which the latent code shifted $\hat{v}_i = P(G(z), G(z + v_i))$. The axes learned by the deformator M were subsequently annotated and a subset was selected by humans.

Using their code, we compared these annotated axes with the Hessian structure we computed on their GANs (PGGAN512, BigGAN noise and StyleGAN Face). In PGGAN512, we found that their discovered axes had a significantly larger $v^T Hv$ (i.e. approximate rate of image change) than random vectors in that space; in other words, their axes were significantly more aligned with the top eigen space ($P < 0.05$ for all axes). Further, we wanted to investigate whether their axes aligned with individual eigenvectors identified by our Hessian or whether their axes randomly mixed with our top eigen space. To achieve this goal, we search for the power coefficients that are significantly higher than expected from projection of unit random vector. In fact, for each and every of the discovered axes, we found 1-3 eigenvectors that they are significantly ($p < 5 \times 10^{-4}$) aligned to. Moreover, these strongly aligned eigenvectors are all in our top 60 eigen dimensions, in fact, 3 of their axes aligned with eigenvector 11 and 2 of their axes aligned with our eigenvector 6. (**Figure II-8** A) Moreover, we ”purify” their axes by a) retaining projection coefficients only in top 60 eigenvectors, or b) retaining the coefficients only in the 1-3 strongly aligned eigen vectors and set all the other 500+ coefficients to zero, and compared their effect on a same set of reference vectors, using the same step size. We found that by project out coefficients in the lower space, the image transformation is perceptually very similar (**Figure II-8**B,C). If we only retains the eigenvectors that it highly aligns to, the image transform will be more different, but the annotated semantics in the transform seems to keep (**Figure II-8**D,E). Thus, their method also discovered that the top eigenspace of PGGAN contained informative transformations, and further confirmed that optimizing interpretability may improve alignment with individual eigen vectors rather than mixing all the eigen dimensions.

Note, as we project out coefficients, the resulting vector has a smaller than unit norm, thus we are moving a smaller distance in latent space using the same step size (**Figure II-8**B-E title). If we renormalize the vector to unit norm we will need to take a smaller step size to achieve the same transform. This is confirming our predictions in Sec. II.4.4: Each interpretable axis u has a family of equivalent axes $u + v$, which add a direction v in the lower eigenspace or null space of the GAN. These axes encode the same transforms but the speeds of image change on them are different. In this sense, the top eigenspace could be used to provide a "purer" version of the interpretable axes discovered elsewhere.

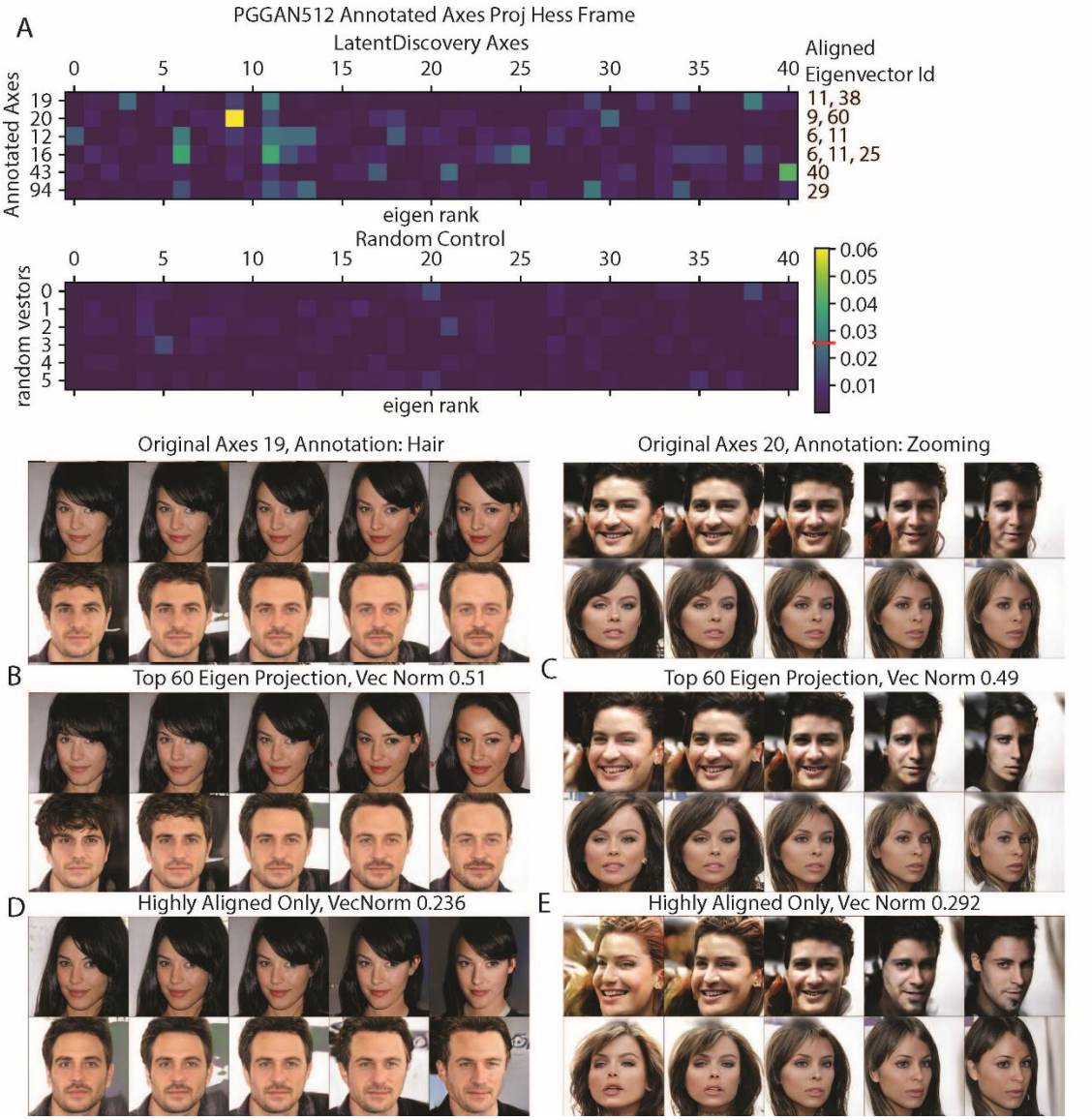


Figure II-8 Analyzing interpretable axes from (Voynov & Babenko, 2020) under the Hessian framework. A. Projection power of their annotated interpretable axes and 6 unit norm random vectors on the top 40 eigen vectors. The color code is matched. The red line on colorbar denotes $\boldsymbol{p < 1 \times 10^{-4}}$ threshold for the power value, and the significantly aligned axes $\boldsymbol{p < 5 \times 10^{-4}}$ are indicated. B,C, Image transforms encoded by projection of 2 of their vectors (19, 20) into the top 60d eigenspace. D,E Similar to B,C, but their vectors are projected onto the aligned eigenvectors only. The norms of the projected vectors are noted on title. Panel B,D and C,E share the same reference image and the same step size across each mini column, though the distance travelled along BC and DE is smaller as the vector is shortened.

Although both types of approaches are promising, by removing the need to train additional networks, our method can be viewed as a more efficient way to identify informative axes. Further work comparing

axes discovered by different methods will elucidate the connection between interpretable axes and the Hessian structure more.

II.8.10 Detailed Comparison to Hessian Penalty

In this section, we compare our work with that of (Peebles *et al.*, 2020), which also focused on the use of a Hessian matrix to "disentangle" directions in generator latent space. This Hessian penalty approach devised a stochastic finite difference estimator of the non-diagonal elements of the Hessian, using only a forward pass, resulting in an efficient regularization of Hessian diagonality during GAN training. This clever approach led to several benefits, including increases in smoothness in latent interpolation, and improvements in interpretability. However, there are clear contrasts between the Hessian penalty approach and ours.

At the highest conceptual level, both works relied on the analysis of the Hessian matrix. Our framework is motivated by a geometric picture of the image manifold and latent representations, while their work is motivated by the idea of increasing the independence of latent factors. The Hessian matrices involved are also different: we computed the Hessian of the squared distance metric d^2 in image space, while they computed the Hessian of every pixel in the image.

At the implementation level, the methods are also different. We computed the exact Hessian matrices or their low-rank approximations based on backpropagation and the Lanczos iteration. They devised a stochastic finite-difference estimator of the total non-diagonal elements of the Hessian, using only a forward pass, which they termed the Hessian penalty. This remarkable achievement enabled them to efficiently regularize Hessian diagonality during GAN training. As another point of contrast, the Hessian penalty does not provide an explicit geometric picture as the exact Hessian matrices do in our work. Specifically, their penalty did not reveal the spectral structure of the Hessian matrices, which is encoded

in the diagonal elements of the matrix. Because of this, the anisotropy can be explicitly demonstrated in our work.

It is clear that the Hessian penalty provides a very important and complementary approach to ours. This work showed that regularizing Hessian diagonality during training could promote disentangled latent representation using PGGAN (this provides a more detailed evaluations of disentanglement than we achieved).

Interestingly, many of the phenomena they observed in Hessian-penalized generative networks were reminiscent to phenomena we observed in normally trained generative networks with a Hessian eigen-frame rotation of the latent space. As a first example, they found that after applying the Hessian penalty, many factors stopped affecting the image (termed "latent space shrinkage" in their work). We also found this phenomenon in normal pre-trained GANs, i.e. they showed a large bottom eigenspace with close to 0 eigenvalues, in which the eigenvectors generated small to no changes in the image (termed "null space" in our work; Sec. II.4.4). Thus if we performed a Hessian eigen-frame rotation, non-Hessian-penalized generators will exhibit similar behavior as theirs do.

As a second example, they also found that enforcing the Hessian penalty on middle layers is helpful in regularizing the Hessian diagonality in the image space, which is reminiscent of our finding that the Hessian eigen-frames are usually well aligned across the layers of generator (Section II.5), though the spectra get shaped throughout the layers.

The Hessian penalty raises an interesting question: what makes a diagonal Hessian matrix special? Because the Hessian is a real symmetric matrix, at each point z, it can be diagonalized by a rotation of the latent space. However, to achieve a diagonal matrix across the latent space, Peebles et al. (2020) had to

(implicitly) enforce each point to share the same rotation (i.e. Hessian eigen-frame). Given the training involved, this encouraged the homogeneity or flatness of latent space as identified in our framework. However, as shown in our work, most GANs exhibit homogeneity or flatness in their latent space even without the Hessian penalty. So it would be interesting to compare generators trained with Hessian penalty against those trained without the penalty but with a post hoc rotation of the latent space using the global eigen-frame. We expect that their training will exhibit flatter geometry than the post hoc rotated latent space. However, even if this is not true, it would still be interesting if this flat geometry can emerge from modern GAN training i.e. fitting the natural image distribution.

Finally, aside from regularizing GAN training, (Peebles *et al.*, 2020) also explored the use of the Hessian penalty in finding interpretable directions in pre-trained BigGAN (BigGAN-128). Similar to (Voynov and Babenko, 2020), the axes they discovered showed a striking correspondence to those identified in our approach (**Figure II-9**). We showed that when we computed the Hessian at a few points in the noise space of the generator, the interpretable axes they found aligned well with our top eigenvectors, with a one-to-one or one-to-two matching. For example, their reported interpretable axes 5, 6 and 8 (for the golden retriever class) had 0.998, 0.964, 0.990 of their power concentrated in our top 10-dimensional eigenspace. Moreover, they aligned with single eigenvectors 0, 5 and 2 with power 0.988, 0.499, 0.852 respectively. Due to the presence of close by eigenvalue, eigen vectors can mix into each other, resulting in the phenomenon that the axes identified in (Peebles *et al.*, 2020) can correspond to a few adjacent eigenvectors (e.g. their axes 5 correspond to our eigenvectors 4 and 5, with corresponding eigenvalues 8.3 and 7.4; as reference, eigenvalue 3 = 17.6, eigenvalue 6 = 5.6, a much larger gap.). We found that power concentration within the top eigenspace is a great indicator of the interpretability of their reported axes: all but one of the reported axes showed over 0.95 power concentrated in top 10d

eigenspace, while the axes they found not to be interpretable showed power 0.103 ± 0.141 (mean ± std) in the top 10d eigenspace. One advantage of our approach is computational efficiency. As the alternative method required learning (i.e. iterative optimization of a mixing matrix), it took us 40 minutes × 50 epochs to finish the training on a 6GB GTX 1060 GPU. In contrast, the present method directly computes the Hessian matrices at one- or a few points, taking 12 seconds to compute a full Hessian matrix (5-50 points usually suffice). The reason for this difference is that their method optimizes for a basis that diagonalizes the Hessian matrix based on a noisy estimate of diagonality – the Hessian penalty; in contrast, our approach directly computes and diagonalizes the matrix at given points. Finally, our method orders the axes by the eigenvalues, facilitating focus on the top eigenspace, and thus alleviating the need to go through all the axes to find interpretable ones.

In summary, we believe that as a stochastic regularizer of the Hessian matrix, the Hessian penalty is a valuable and complementary approach to ours. Our methods provide additional value by accurately estimating the top eigenvectors and eigenspectrum, suitable for analyzing geometry post hoc. However, unlike the Hessian penalty, direct use of our method to regularize training may be inefficient. It would be interesting to explore whether there is a middle ground that incorporates the advantages of both their estimator and our more precise calculation.

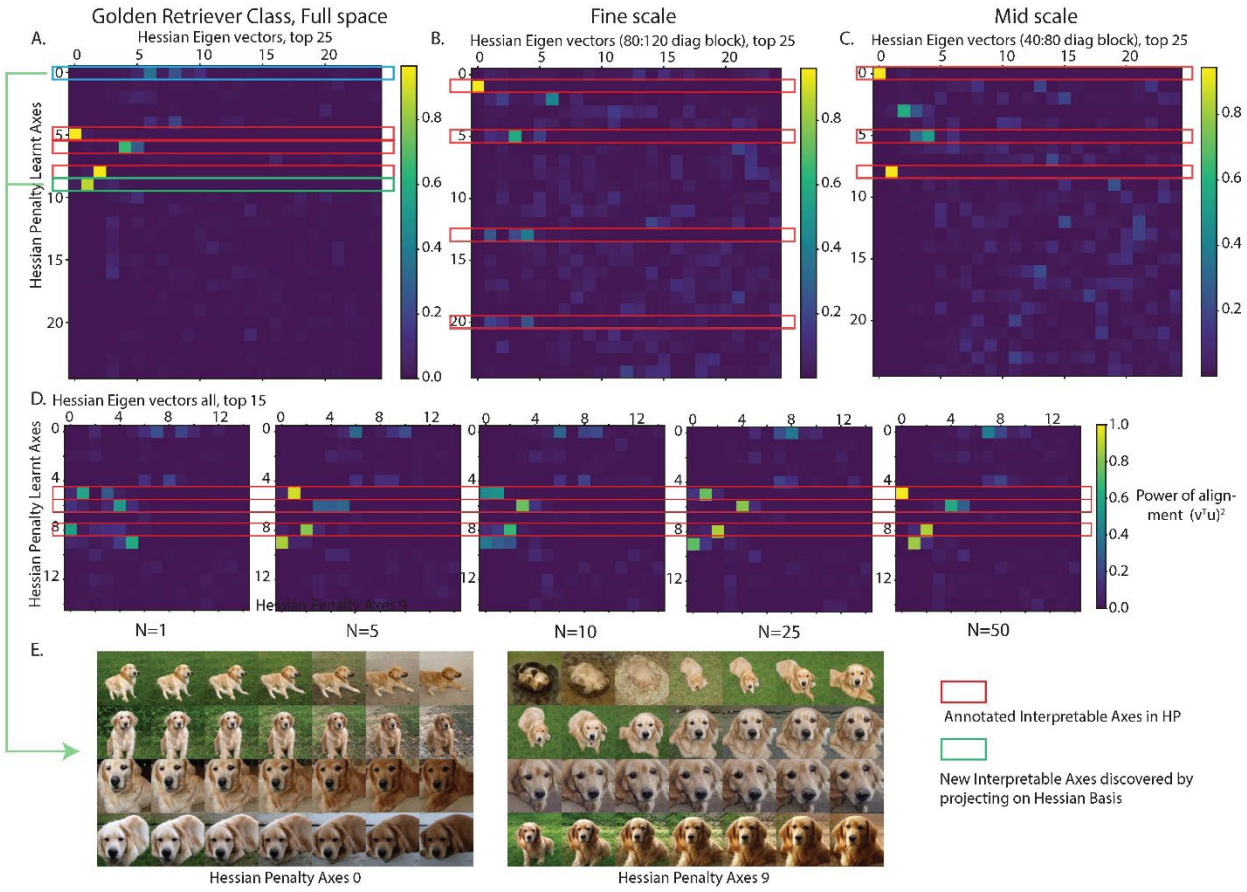


Figure II-9 Analyzing interpretable axes from (Peebles et al., 2020) under the Hessian framework. A. Projection power of Hessian-penalty-identified axes on the top 25 eigen vectors. B,C. For the axes corresponding to fine or mid-scale changes, we performed eigen decomposition to the corresponding 40d diagonal block of the averaged full Hessian. We then computed the projection power with their axes. D. Comparison of the alignment with the Hessian eigen frame averaged over different number of points ($\boldsymbol{N = 1, 5, 10, 25, 50, 100}$) in the full space. Although the specific matched eigen id varies between $\boldsymbol{N}$, those interpretable axes all remain in our top eigenspace, with power over 0.95. E. Visualize the axes in Peebles et al. (2020) with a good alignment with our top 10d eigenspace (axes 0 and 9, boxes marked in green), other than those annotated in their paper (boxes marked in red). They are arguably interpretable as well.

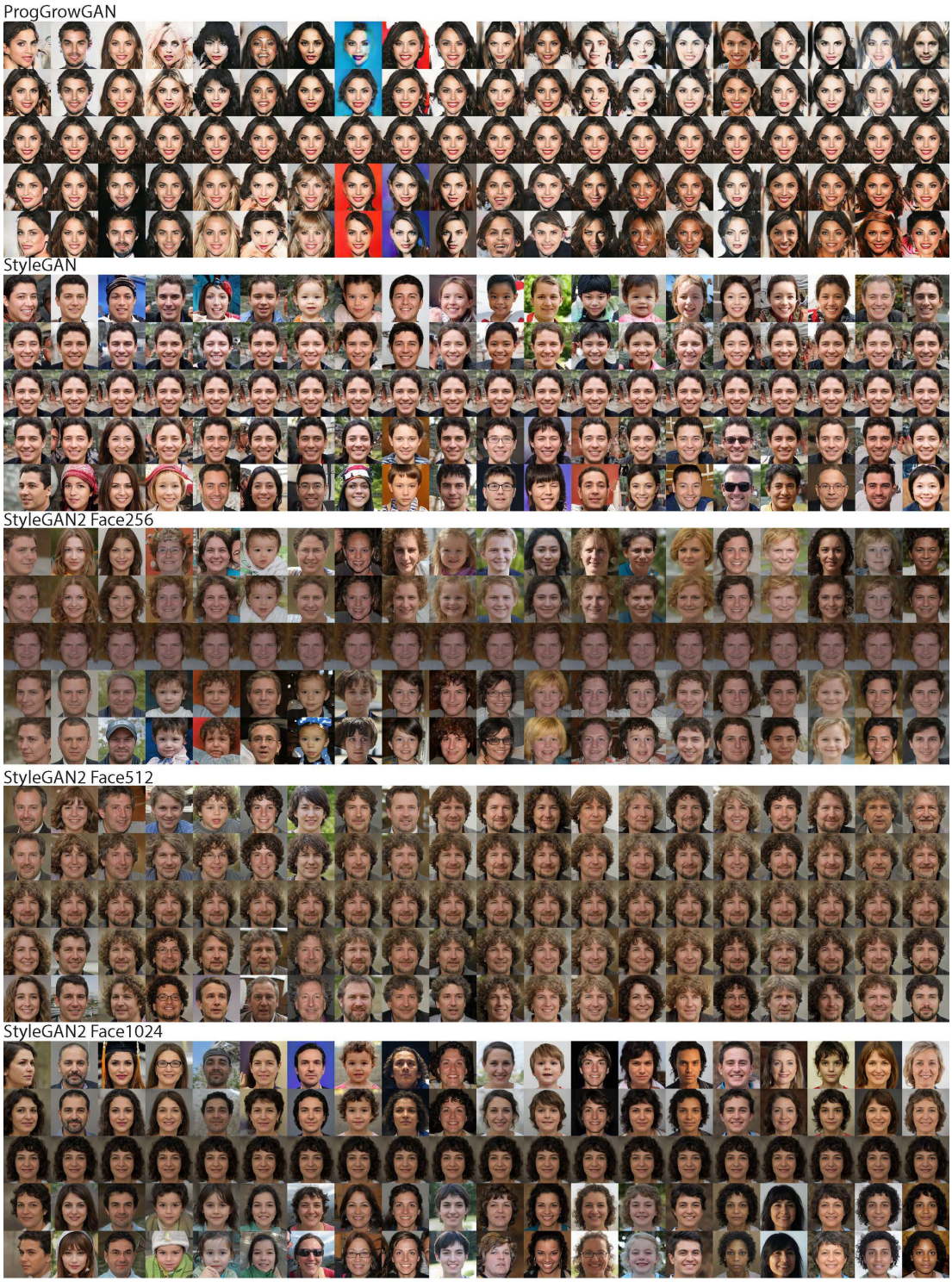


Figure II-10 Similar transforms encoded in the top eigendimension of GANs trained on face dataset. Linear exploration along top 20 eigenvectors from origin in latent space are showed for each GAN. Linear equidistance sampling on each eigenvector occupies a column and their eigenvalues are sorted in descending order from left to right. Step size along each vector is adjusted according to its eigenvalue for best continuity.

Figure II-11 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.

Chapter III : High Performance Evolutionary Algorithm for Neuronal Control on Image Manifolds

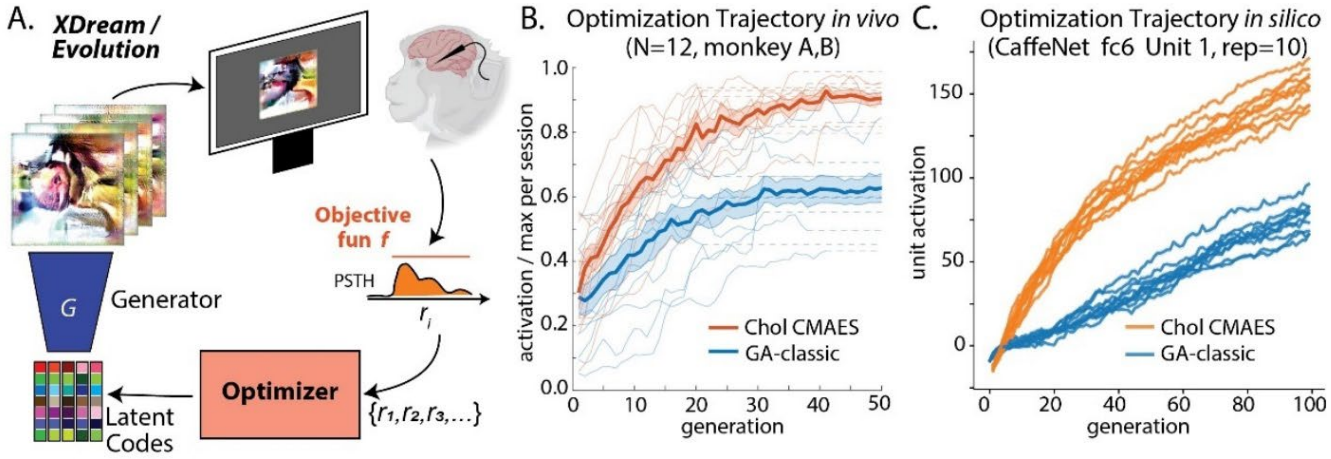


Figure III-1 Cholsky-CMAES Excelled in Activation Maximization both *in silico* and *in vivo*. A. Schematics of the XDream Evolution experiment. B. *in vivo* optimization trajectory from 12 paired Evolution experiments in two monkeys. Thin curves show the trajectories for individual experiments; shaded thick curves show the mean and standard error (SEM) of trajectories across experiments. Experiments that terminated earlier were extrapolated by constant (dashed line) to match the generation number for mean and SEM calculation. C. *in silico* optimization trajectory comparison for unit 1 in fc6 layer of CaffeNet, mean activation per generation is plotted.

Abstract

Recently, optimization has become an emerging tool for neuroscientists to study neural code. In the visual system, neurons respond to images with graded and noisy responses. Image patterns eliciting highest responses are diagnostic of the coding content of the neuron. To find these patterns, we have used black-box optimizers to search a 4096d image space, leading to the evolution of images that maximize neuronal responses. Although genetic algorithm (GA) has been commonly used, there haven't been any systematic investigations to reveal the best performing optimizer or the underlying principles necessary to improve them.

Here[3], we conducted a large scale *in silico* benchmark of optimizers for activation maximization and found that Covariance Matrix Adaptation (CMA) excelled in its achieved activation. We compared CMA against GA and found that CMA surpassed the maximal activation of GA by 66% *in silico* and 44% *in vivo*. We analyzed the structure of Evolution trajectories and found that the key to success was not covariance matrix adaptation, but local search towards informative dimensions and an effective step size decay. Guided by these principles and the geometry of the image manifold, we developed SphereCMA optimizer which competed well against CMA, proving the validity of the identified principles.

III.1 Introduction

An essential goal in sensory neuroscience is to define how the neurons respond to natural stimuli and extract useful information to guide behavior. To a first approximation, visual neurons emit high rates of electrical signals to stimuli with certain visual attributes, so their outputs can be interpreted as conveying the presence of such features (e.g. "face neurons" (D H Hubel and Wiesel, 1962; Quiroga *et al.*, 2005)). So, to study the visual selectivity of neurons, it is crucial to choose highly activating stimuli.

Traditionally, researchers have used intuition (Gross, 1994) or limited theoretical frameworks to choose a fixed set of stimuli, i.e., simple images, such as circles and rings for studying lateral geniculate nucleus cells, oriented bars for V1 neurons, hyperbolic gratings for V2 neurons (Hegdé and Van Essen, 2004, 2006), curved shapes for V4 neurons (Pasupathy and Connor, 2001, 2002), or select categories such as faces for inferotemporal cortex neurons (Desimone *et al.*, 1984). The desired property of these stimuli is their ability to drive neuronal activity. But as visual neurons become more selective along the posterior-anterior anatomical axis — responding to more complex visual attribute combinations — it becomes more difficult to choose highly activating stimuli.

[3] Code available at https://github.com/Animadversio/ActMax-Optimizer-Dev

To tackle this problem, an alternative approach is to use adaptive search methods to find highly activating stimuli. The idea is to treat neuronal responses as a function of visual stimuli, and to iteratively search for variants that maximize this function. As images evolve under this search, they acquire visual attributes informative of the neuron's intrinsic tuning, independent of bias in human intuition.

Problem Formulation. Formally, neurons can be conceptualized as noisy scalar functions f over the image space I. One common objective is to maximize the function $f(I)$ —also known as activation maximization, commonly used for interpreting the coding content of neurons or CNN hidden units (Nguyen *et al.*, 2016):

$$r = f\ (I) + \epsilon\ , f: \mathcal{I} \to \mathbb{R} \qquad \text{(III-1)}$$

$$I_* = \operatorname{argmax}_I f\,(I) \qquad \text{(III-2)}$$

While this approach can be easily generalized to other objectives, we focus on activation maximization since making neurons fire strongly has been the most common goal in sensory neuroscience studies over the decades (David H Hubel and Wiesel, 1962).

In the context of *in vivo* experiments, the process amounts to recording a set of neuronal activations $\{r_i\}$ in response to a set of randomly sampled images $\{I_i\}$ (each image displayed for a short duration, e.g. 100 ms, Figure III-1A). In the next step, the optimizers update their state and propose a new set of images. By repeating this process for dozens of iterations, the images begin to acquire visual attributes that drive the neuron's activity highly. Usually, in one of these so-called *Evolution* experiments, the number of image presentations is limited to 1000-3000, taking 20-40 min. This mandates a high sample efficiency of the optimizer.

For most optimization algorithms, stimuli need to be represented in and generated from a "vector space". We consider a smooth parametrization of images $G : \mathbb{R}^d \to \mathcal{I}, z \mapsto I$ using a lowerdimensional vector space. In this implementation, the mapping from vector space to images is instantiated by an image generative neural network (DeePSim Generative Adversarial Network (GAN) (Dosovitskiy and Brox, 2016b)), which maps 4096d latent vectors to images. Thus, the problem is about searching for images in the latent space such that it maximizes the response of the neuron.

$$z^* = \text{argmax}_z \mathbb{E}\big[f\left(G(z)\right)\big] \qquad \text{(III-3)}$$

This optimization approach has been applied to neurons in visual areas V4 and inferotemporal cortex (IT) (Yamane *et al.*, 2008a; Carlson *et al.*, 2011; Hung, Carlson and Connor, 2012; Vaziri and Connor, 2016; Ponce *et al.*, 2019b; Xiao and Kreiman, 2020, 2020; Rose, Johnson, Wang and Carlos R. Ponce, 2021). Image search was effectuated by classic genetic algorithms (GAs), acting in the space of parametrized 3D shapes or GANs. Though the use of GAs was successful in this domain (Xiao and Kreiman, 2020), it has not been tested comprehensively against modern optimizers, which motivated us to determine if we could improve performance on this front.

This problem features a unique set of challenges, for example, search dimensionality is very high (e.g. $d = 4096$ in [24, 28, 32]), and the function $f(.)$ must be evaluated with considerable noise in neuronal responses. Further, the total number of function evaluation N_f is highly limited ($N_f < d$), thus the dimensions could not be exhaustively explored.

In this project, we worked to improve the performance of optimizer in this domain and to extract the underlying principles for designing such optimizer. The main contributions are as follows:

- We conducted two *in silico* benchmarks, establishing the better performance of CMA-type optimizers over other optimizers including commonly used genetic algorithms (GA).

- We validated the performance increase of the Cholesky-CMA algorithm, with a focused contrast to classic GA *in vivo*.
- We found that the CMA search trajectories exhibited the spectral signature of high-dimensional guided random walks, preferentially traveling along the top eigen-dimensions of the underlying image manifold.
- We found one reason that CMA succeeded was the decrease of angular step size, thanks to the spherical geometry of image space and the increased vector norms in Evolution.
- Guided by image space geometry, we built in these mechanisms to develop a *SphereCMA* algorithm, which outperformed the original CMA *in silico*.

III.2 Screening of Black Box Optimizers

Because *in vivo* testing of optimizer performance can be costly and time-consuming, we began by screening a large set of algorithms *in silico* using convolutional neural network (CNN) units as proxies for visual neurons, then tested the top performing algorithms with actual neurons in a focused comparison.

III.2.1 Large Scale in silico Survey

To simulate neuronal tuning function f that an optimizer might encounter in a biological setting, we used units from pre-trained CNNs as models of visual neurons (Lindsay, 2020; Xiao and Kreiman, 2020). For the *in vivo* Evolution experiments, we aimed for optimizers that performed well with neurons across visual areas (including V1, V4, IT) and across different levels of signal-to-noise and single-neuron isolation. Thus, we designed the benchmark "problem set" to include units from multiple *CNN architectures*, *layers*, and *noise levels*, testing the overall performance for each optimizer.

We chose AlexNet and a deeper, adversarially trained ResNet (ResNet50-robust) as models of ventral visual cortex. The latter was chosen because it exhibits visual representations similar to the brain (high rank on Brain-Score [29]). For each network, we picked 5 layers of different depths. It has been noted that units from shallow-to-deep layers prefer features of increasing complexity [22], similar to that in the ventral stream cortical hierarchy [28].

For detailed information about these networks and their layers, see Sec. III.6.1. In the context of *in vivo* recordings, single-presentation neuronal responses are highly noisy (Czanner *et al.*, 2015). To simulate the Poisson-like noisy response $\tilde{r}$, we added Gaussian noise with zero mean and standard deviation proportional to the raw response αr (ratio α represented the noise level). We tested three noise levels for each objective function: no noise ($\alpha = 0$), low noise ($\alpha = 0.2$), and high noise ($\alpha = 0.5$).

$$\tilde{r} = \max(0, (1 + \alpha\epsilon)r), \qquad \epsilon \sim N(0, 1)$$

In the first round, we chose 12 gradient-free optimizers as implemented or interfaced by nevergrad (Rapin and Teytaud, 2018): NGOpt, DE, TwoPointsDE, ES, CMA (Hansen and Ostermeier, 2001), DiagonalCMA (Akimoto and Hansen, 2020), RescaledCMA, SQPCMA, PSO, OnePlusOne, TBPSA, and RandomSearch. Here we compared these algorithms by their default hyper-parameter settings. Note that RandomSearch just sampled random vectors from an isotropic Gaussian distribution, finding the vector with the highest score, which formed the naive baseline (for a short introduction to these algorithms, see (Rapin and Teytaud, 2018)). Each algorithm ran with a budget of 3000 function evaluations and three repetitions per condition.

Among the 12 optimizers, we found that Covariance Matrix Adaptation Evolution Strategy (CMA) and DiagonalCMA were the top two algorithms in achieving maximal activation (Figure III-2). Since the upper bound of the activation of a unit in CNN was arbitrary, we divided the raw

activations by the empirical maximal activation achieved by that given unit, across all algorithms and repetitions. By this measure, when pooling all conditions, CMAES achieved 0.699 ± 0.004, DiagonalCMA achieved 0.677 ± 0.004, as a reference Random Search baseline achieved 0.139 ± 0.002 (mean ± sem, $N = 1500$, Table III-3, Figure III-2. This difference of CMA-driven performance vs. any other optimizer was significant per a two-sample t-test: $t > 60$ for all comparisons, except for CMA vs DiagonalCMA, where $t = 3.80, p = 1.4 \times 10^{-4}$). We found the same result held consistently for units across CNN models, layers and noise levels (see comparison in **Table III-3**). The optimization score traces and the most activating images found for an example ResNet unit are shown in **Figure III-8** and **Figure III-10**.

As for time efficiency, we measured the total run time taken by optimizing the objective function with a budget of 3000 evaluations[4]. With units from AlexNet as objective, CMA had a longer runtime of 104.5 ± 30.0 s (mean ± std); DiagonalCMA accelerated the runtime by roughly five-fold (23.0 ± 4.5 sec), although at a slightly compromised score (6.1%). As a reference, the baseline algorithm RandomSearch had an average runtime of 17.6 ± 2.2 sec. Same trends held for ResNet50 units, though the runtime values were generally longer because ResNet50 is deeper (**Table III-3**). In conclusion, we found CMA and DiagonalCMA algorithms were both well-suited for this domain, while DiagonalCMA achieved a good trade-off between speed and performance.

III.2.2 Comparison of CMA-type Algorithm with GA in silico

Given the general success of CMA and DiagonalCMA algorithms, we were motivated to test other types of CMA algorithms in the second round, comparing runtime values and achieved activations.

[4] These optimizers were tested on single core machine with V100 GPU without batch processing of images or activations.

As described in (Hansen, 2016), the CMA algorithm maintains and updates a Gaussian distribution $\mathcal{N}(m, \sigma C)$ in the $\mathbb{R}^d$ space, with the mean vector m and step size σ initialized by the user; the covariance matrix C is initialized as identity matrix I. In each step, the algorithm samples a population of codes from

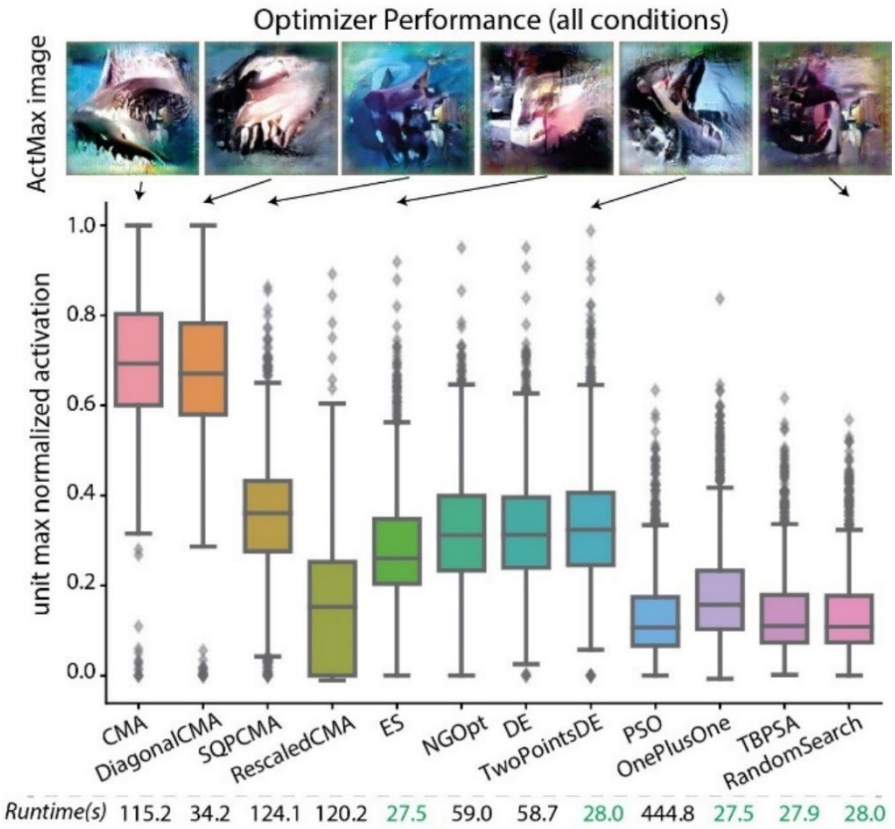


Figure III-2 CMAES Excelled in Large Scale Benchmark of Activation Maximization. Normalized activation were pooled across all conditions. Mean runtime across all condition were annotated below each optimizer. The activation maximizing image found by the optimizers for an CNN unit (AlexNet-fc8-unit003 tiger shark unit) were showed above the scores.

this exploration distribution, $z_t(i) \sim \mathcal{N}(m_t, \sigma_t C_t)$. After evaluating these codes, it updates the mean vector m_t by a weighted combination of the highest-ranking codes. Note that σ_t controls both the spread of samples in a generation and the average step size of the mean vector update $\|m_{t+1} - m_t\|$. This step size σ_t is adapted based on the accumulated path length in the last few steps. Moreover, CMA updates the covariance matrix C_t by a few low-rank matrices to adapt the shape of the exploration distribution.

In the original CMAES algorithm, after each covariance update, the covariance matrix C needed to be eigen-decomposed, in order to get C^{-1}. For a high dimensional space $d = 4096$, it is costly to compute this decomposition at each update. Using the diagonal covariance matrix approximation, the inversion step could be simplified from a $O(n^3)$ operation to $O(n)$ operation, which makes the DiagonalCMA (Akimoto and Hansen, 2020) much faster (Figure III-2). This inspired us to use modified covariance update rules to accelerate the optimizer.

We found Cholesky-CMA-ES (Loshchilov, 2017) which was proposed as a large-scale variant of CMAES. By storing the Cholesky factor A and its inverse of the covariance matrix, it could update these factors directly, without factorizing the covariance matrix at each update. An additional parameter `A_update_freq` could be adjusted to tune the frequency of this update.

We implemented the $(\mu/\mu_w, \lambda)$-Cholesky-CMA-ES algorithm (CholeskyCMA) (Loshchilov, 2017) and compared it against the CMA and DiagonalCMA implemented in **pycma** library (Hansen and Auger, 2014; Akimoto and Hansen, 2020) and the Genetic Algorithm (GA), classically used in this domain (Yamane *et al.*, 2008b; Ponce *et al.*, 2019b; Srinath *et al.*, 2021). For CholeskyCMA, the hyper-parameters σ_0 and update frequency of Cholesky factor A were tuned and fixed $\sigma_0 = 3.0$, $\text{A\_update\_freq} = 10$. For GA, we used the code and hyperparameters from (Ponce *et al.*, 2019b). We allowed for 3000 function evaluations per optimizer, which was 75 generations with a population size of 40.

We slightly modified the *in silico* benchmark: we chose 7 layers from AlexNet (conv2 to fc8), 10 units from each layer, with 3 noise levels ($\alpha = 0.0, 0.2, 0.5$). Each optimizer were run with 5 repetitions in each condition, totaling 1050 runs. We evaluated the clean score, i.e. the highest noise-free activation of

the given unit and the runtime for each optimizer. Here we also used the empirical maximal clean score achieved for each unit to get the normalized clean score and to calculate statistics.

The results were summarized in **Table III-1** and **Figure III-11**, **Figure III-12**. We found that all CMA algorithms outperformed Genetic Algorithm by a large margin: if we pooled all conditions, the mean normalized score of CMA was 166.7% of that of GA optimizer. Noise deteriorated the score for all optimizers, and there the performance gap between GA and CMA type algorithm was narrowed but persisted: the mean normalized score for CMA was 196.2% of GA in noise free scenario; 151.9% for $\alpha = 0.2$; 146.4% for $\alpha = 0.5$ (**Figure III-11**). The overall performance values of the three CMA-type optimizers were not statistically different, and all surpassed GA. (1-way ANOVA, $F_{2,3147} = 1.70, p = 0.18$).

When we examined the performance per layer, we found the relative performance of optimizers had a significant interaction with the source layer of the unit. DiagonalCMA was more effective than Cholesky and original CMA in the earlier layers, but performed less well in deeper layers (**Table III-1**, **Figure III-12**). We tested this interaction with ANOVA on a linear model:

$$\text{activation} \sim \text{optimizer} + \text{noiselevel} + \text{layer} + \text{layer} * \text{optimizer} + \text{noiselevel} * \text{optimizer}$$

in which, "optimizer" was modelled as a categorical variable (CholeskyCMA, CMA, DiagonalCMA), "noiselevel" and "layer" were continuous variable. All the factors except optimizer had statistically significant main effect; and interaction term layer * optimizer had $F_{2,3141} = 14.98, P = 3.34 \times 10^{-7}$ (see Table III-2). This curious interaction would be interpreted below in the context of different covariance matrix adaptation mechanisms (Sec. III.3.1).

As for runtime[5], DiagonalCMA was still the fastest, with average runtime 6.6 ± 0.1 sec (mean ± sem, $N = 1050$, same below), while the second fastest optimizer was GA with runtime 16.6 ± 0.2 sec. In contrast, the classic CMA algorithm taking 97.0 ± 0.8 was the slowest among the four, while the CholeskyCMA took 42.4 ± 0.4 sec per run. Indeed, updating Cholesky factors made it run faster without reducing performance.

III.2.3 CMA outperformed GA *in vivo*

After the second round of *in silico* benchmarks, we were ready to let the top candidates compete with the previous state-of-the-art Genetic Algorithm *in vivo*. We chose CholeskyCMA algorithm and compared it against the classic GA, since its speed and performance were both good across noise level and visual hierarchy. For detailed methods of *in vivo* experiments, we refer readers to Sec. III.6.2, but briefly, we recorded electrophysiological activity in two animals using chronically implanted arrays, placed within V1, V4 and IT.

We conducted 14 experiments. Two out of fourteen (2/14) experiments did not result in a significant increase in firing rate of the neuron for either optimizer (per criterion $p < 0.001$, for t-test between firing rates in first two and last two generation), which we excluded from the further analysis. From the 12 experiments where at least one optimizer increased the firing rate, we normalized the firing rate of the neuron to the highest generation averaged firing rate for the CholeskyCMA optimizer (Figure III-1B). The normalized final generation activation for CholeskyCMA was 0.908 ± 0.018 compared to 0.628 ± 0.045 for GA (paired t-test, $t = 6.69, p = 3.4 \times 10^{-5}, df = 11$. Raw firing rates for each experiment

[5] Here, latent codes were processed in batch of 40 by generator and CNN, thus the same optimizer ran faster than the previous experiment.

are shown in **Figure III-13**). Thus, the maximal activation of CholeskyCMA surpassed GA by 44%, which was comparable to the activation gain in the high-noise condition (46.2%, $\alpha = 0.5$) *in silico*.

From this, we concluded that CholeskyCMA outperformed classic GA algorithm both *in vivo* and *in silico*, becoming the preferred algorithm for conducting activation maximization.

Table III-1 Layer-wise Performance Comparison of CMA-Style Algorithms. This table presents mean normalized clean score for each optimizer, averaged across all 10 units and 5 repetitions in given layer under given noise level α. For each unit, the clean score was normalized by the highest noise-free activation achieved across optimizers and noise. The last row showed the mean run time ($\boldsymbol{N = 1050}$). Abbreviations: Genetic Algorithms (*GA*), CholeskyCMA (*Chol*), CMA-ES (*CMA*) and DiagonalCMA (*Diag*) as implemented in pycma library, SphereCMA with Exponential decay (*Sp exp*) or inverse function decay (*Sp inv*) of step size (see Sec. III.4). Red text showed the best performing optimizer in each condition, the multiple red scores are not statistically different (per $p > 0.001$). See **Figure III-11** for the distribution of score from all 7 layers.

layer	α	GA	Chol	CMA	Diag	Spexp	Spinv
conv2	0.0	0.525	0.838	0.770	0.823	0.901	0.822
	0.2	0.473	0.654	0.626	0.685	0.659	0.637
	0.5	0.406	0.565	0.545	0.633	0.548	0.546
conv3	0.0	0.443	0.791	0.793	0.812	0.886	0.819
	0.2	0.398	0.618	0.607	0.663	0.603	0.598
	0.5	0.345	0.524	0.492	0.526	0.495	0.486
conv4	0.0	0.378	0.758	0.723	0.728	0.859	0.754
	0.2	0.358	0.554	0.546	0.618	0.570	0.532
	0.5	0.299	0.446	0.435	0.468	0.440	0.420
conv5	0.0	0.372	0.756	0.764	0.760	0.913	0.792
	0.2	0.354	0.547	0.539	0.598	0.542	0.523
	0.5	0.298	0.456	0.429	0.464	0.433	0.409
fc6	0.0	0.244	0.571	0.577	0.519	0.766	0.613
	0.2	0.228	0.361	0.370	0.429	0.404	0.399
	0.5	0.201	0.309	0.276	0.276	0.259	0.260
fc7	0.0	0.320	0.663	0.706	0.659	0.830	0.662
	0.2	0.295	0.444	0.435	0.457	0.438	0.404
	0.5	0.249	0.338	0.344	0.345	0.310	0.318

	0.0	0.308	0.703	0.733	0.600	0.865	0.712
fc8	0.2	0.281	0.448	0.458	0.439	0.473	0.446
	0.5	0.228	0.328	0.319	0.324	0.316	0.305
Average		0.333	0.556	0.547	0.563	0.596	0.546
runtime (s)		16.6	42.4	97.0	6.6	25.4	25.9

III.3 The Analysis of CMA Evolution

Why did CMA-type algorithms perform so well? Was it the covariance updates, adaptation of step size, or a fortuitous match between the geometric structure of the latent space and the algorithm? We were motivated to find which component contributed to its success. We reasoned that the optimizers should work best when they conform to the geometry of the generative image manifold G, and the geometry of neuronal tuning function f on the manifold. So, we focused on analyzing the geometry of the search trajectory with respect to the geometry of image space (Wang and Ponce, 2021). When available, we analyzed the *in silico* and *in vivo* evolutions back-to-back to validate the effect.

III.3.1 The "Dysfunction" of Covariance Matrix Adaptation

First, we noticed that in a high-dimensional space, the covariance matrix updates were impaired in the original $(\mu/\mu_w, \lambda)$ CMA or CholeskyCMA algorithm. In the default setting[6], the covariance learning rates $c_1, c_\mu \propto 1/d^2$, which were exceedingly small at $d = 4096$. Thus, the covariance matrix was updated negligibly and could be well approximated by an identity matrix. Recently, this was also pointed out in (Akimoto and Hansen, 2020) (Sec.4,5) and the authors proposed to increase the learning rate in DiagonalCMA. We tested the effectiveness of this modification.

[6] See the default setting of c_1, c_μ in Tab. 2 of (Hansen, 2016).

Empirically, we validated this for the original, Cholesky, and Diagonal CMAES algorithms. We quantified this by measuring the condition number of the covariance matrix $\kappa(C)$ and its relative distance to identity matrix $\Delta(C)$.

$$\kappa(C) = \frac{\lambda_{\max}(C)}{\lambda_{\min}(C)}, \qquad \Delta(C) = \frac{\|C - I\|_F^2}{\|C\|_F^2}$$

We found that the final generation condition number $\kappa(C)$ of the CholeskyCMA was 1.000175 ± 0.000004 (mean ± std, $N = 175$, same below), while the relative distance to identity $\Delta(C)$ was $8.05 \pm 0.31 \times 10^{-12}$. In comparison, condition number $\kappa(C)$ for the original CMAES was 1.002739 ± 0.000046, and for the DiagonalCMA, 1.124728 ± 0.034774. The relative distances $\Delta(C)$ to the identity matrix were $7.13 \pm 0.01 \times 10^{-8}$ and $1.63 \pm 0.21 \times 10^{-4}$ for original CMA and DiagonalCMA algorithms. Though as designed, DiagonalCMA updated the covariance matrix more effectively than the other two CMA algorithms, its final covariance was still quite isotropic. On the other hand, for the original and CholeskyCMA algorithm, we could safely approximate the exploration distribution $p(z_{t+1}^{(i)}|m_t)$ by an isotropic Gaussian scaled by the step size σ_t

$$p\left(z_{t+1}^{(i)}\middle|m_t\right) \sim \mathcal{N}(m_t, C) \approx \mathcal{N}(m_t, \sigma_t I)$$

This isotropic exploration throughout the Evolution experiments simplified subsequent analyses of the algorithm.

How is this related to the performance of the algorithm? This relates to the interaction between the unit layer and optimizer, noted above (Sec. III.2.2, Table III-1, Table III-2): the DiagonalCMA outperformed original CMA in earlier layers of CNN but not in deeper layers. It seems the faster update of diagonal covariance matrix was only beneficial for units in earlier layers. The diagonal covariance was designed for a separable, ill-conditioned functional landscape. We noticed that although all units had highly ill-

conditioned landscapes, the units in shallower layers had tuning for fewer dimensions than units in deeper layers (Sec. III.6.4). In other words, the units in earlier layers had a larger invariant space, and a diagonal covariance might suit this landscape better than the more complex ones for deeper units.

In short, we conclude that the effectiveness of CMA-type algorithm was not in its adaptation of the exploratory distribution shape, and we postulate that it could work with fixed covariance.

III.3.2 Evolution Trajectories Showed a Sinusoidal Structure, Characteristic of Random Walks

Next, we investigated the geometric structure of the Evolution trajectories, through the lens of Principal Component Analysis (PCA), a linear dimension-reduction algorithm. For each experiment, we computed the mean latent vector for each generation $\{\overline{z}_t\}, t = 1 \ldots T$, and applied PCA to this T-by-d matrix of mean vectors ($T = 75$).

We found a pervasive sinusoidal structure to the typical trajectory. When a given trajectory was projected onto the top principal components (PC), the projection resembled cosine waves (Figure III-3 A). On the kth component, the projected trajectories were well-fit by cosine functions of frequency $k\pi$: $A\cos(k\pi t/T)$ (**Figure III-19**A). If we allowed the phase and frequency to be fit, $A\cos\big(2\pi\omega(t/T + \phi)\big)$, the fit R^2 was above 0.80 for the top 16 PCs for 95% of trajectories (**Figure III-19**B). As a result, projecting the mean trajectory onto the top PCs will result in Lissajous curves (**Figure III-21**). Further, we analyzed the PCs of the *in vivo* Evolution experiments ($N = 264$), and found the same sinusoidal structure (Figure III-3 C). Even for control evolutions driven by noise, this sinusoidal structure persisted (**Figure III-20**).

This intriguing structure was reminiscent of the PCA of another type of high-dimensional optimization trajectory: that of neural network parameters during training (Lorch, 2016). This structure was later

observed and analytically described for high-dimensional random walks or discrete Ornstein-Uhlenbeck (OU) processes guided by potential (Antognini and Sohl-Dickstein, 2018). To test this connection, we examined the explained variance of each principal components (Figure III-3 B). We found that the explained variance $\rho(k)$ of the kth PC was well approximated by Eq. III-4, which was derived as the theoretical limit for PCA of random walk with T steps in high-dimensional space[7] (Antognini and Sohl-Dickstein, 2018). If we take the step number $T \to \infty$ to infinity in Eq. III-4, the explained variance of kth PC scales with k^{-2} (Eq. III-4), showing a fast decay.

$$\rho(k) = \frac{\frac{1}{2}\left[1-\cos\left(\frac{\pi k}{T}\right)\right]^{-1}}{\frac{1}{6}(T^2-1)}, \lim_{T\to\infty} \rho(k) = \frac{6}{\pi^2 k^2} \qquad \text{(III-4)}$$

Indeed, both in theory and in our data, the first principal component explained more than 60% variance of mean evolution trajectory, and the top five PC explained close to 90% of the variance (Figure III-3 B). In this view, the evolution trajectory of mean latent code $\bar{z}$ of CMA-ES could be regarded as a (guided) random walk in high dimensional space, with its main variance residing in a low-dimensional space.

[7] Our Eq. 8 was adapted from Eq.12 in (Antognini and Sohl-Dickstein, 2018). We found the original equation had the incorrect normalization factor, so we corrected the denominator as it is now.

How are evolutionary trajectories related to random walks? In the extreme, when the response is pure noise, the randomly weighted average of the candidates $\{z_t^{(i)}\}$ will be isotropic; thus each step $p(m_{t+1}|m_t)$ is taken isotropically. Intuitively, this is close to a random walk. In our evolutions, the selection of candidates was biased by the visual-attribute tuning of units or neurons, thus their associated trajectories were random walks guided by given potentials. As derived in (Antognini and Sohl-Dickstein, 2018), for random walks guided by quadratic potentials, the structure of trajectory should be dominated by the dimensions with the smallest driving force (curvature). As we will see below (Sec. III.3.3, Sec. III.6.3, III.6.4), the latent space had a large portion of dimensions to which the units were not tuned. This may explain why our evolution trajectories looked like random walks via PCA. In summary, we interpreted this as an intriguing and robust geometric property of high-dimensional curves (Antognini and Sohl-Dickstein, 2018), but its full significance remains to be defined.

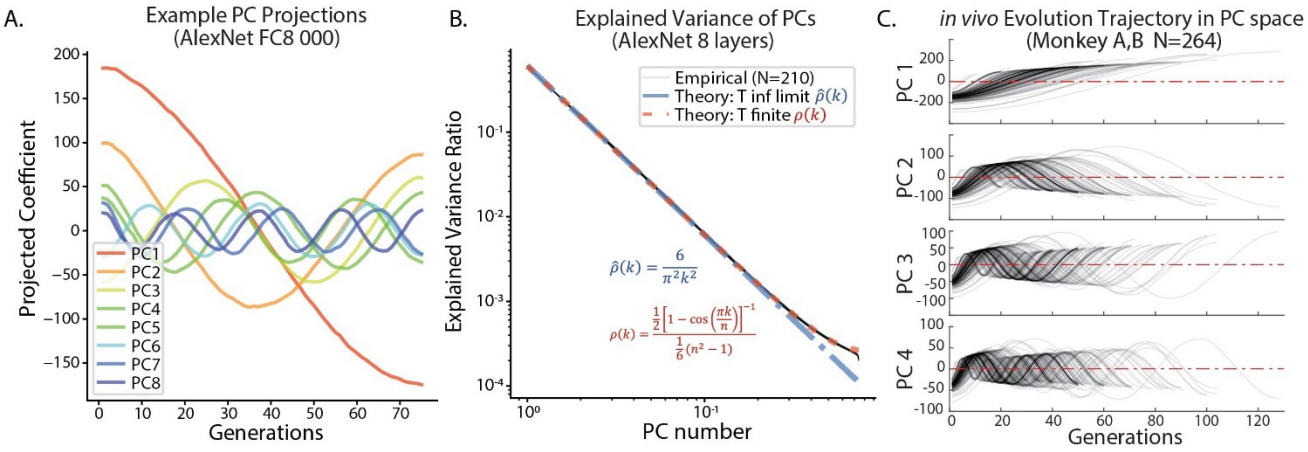


Figure III-3 Sinusoidal Geometry of Evolution Trajectory as a Signature of Random Walk. **A**. Mean search trajectory projected onto the first 8 PCs of an example Evolution. **B**. Explained variance of each PC coincided well with the theoretical value (Eq. III.4) $\boldsymbol{N = 210}$ runs plotted, precisely overlapping. **C**. Mean evolution trajectory projected onto top 4 PCs for $\boldsymbol{N = 264}$ Evolution experiments *in vivo*.

III.3.3 Evolution Trajectories Preferentially Travel in the Top Hessian Eigenspace

Given that individual trajectories had a low-dimensional structure, we asked if there was a common subspace that these Evolution trajectories *collectively* shared. We reasoned that each optimization trajectory should lead to a local optimum of the tuning function of each given CNNs unit or cortical neuron, encoding relevant visual features. Though different neurons or units prefer diverse visual attributes of different complexity (Rose, Johnson, Wang and Carlos R. Ponce, 2021), it was possible that the preferred visual features populated certain subspaces in the latent space of our generator.

We first collected the *in silico Evolution* trajectories of $N = 1050$ runs, across all conditions, and represented each trajectory by the mean code from the final generation, i.e. the evolution direction ζ_j . We shuffled the entries of each vector $\{\zeta_j\}, j = 1 \ldots N$ to form a control collection of evolution directions $\{\zeta_{shfl}\}$, preserving their vector norms. We found that the collection of observed *Evolution* directions was lower-dimensional than its shuffled counterpart: by PCA, the explained variance of top 7 PCs was larger than the shuffled counterpart ($p < 0.005$ comparing to 500 shuffles, **Figure III-14**).

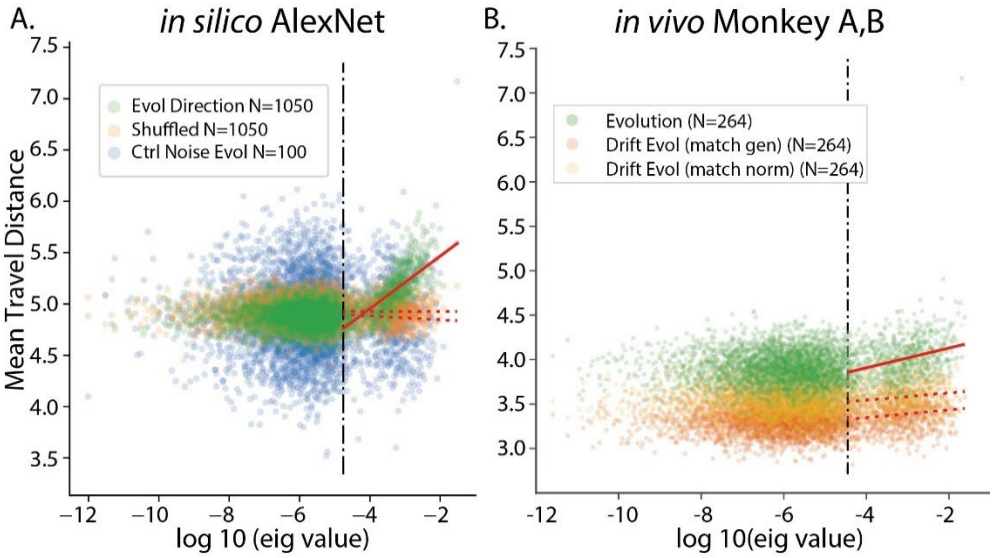


Figure III-4 Evolution Directions Preferably Aligned with the Top Hessian Eigenspace. Mean distance traveled along each eigenvector, plotted against the log eigenvalue. A. In silico evolutions, shuffled code and evolution driven by noise. The cutoff line (at 800th eigenvalue) plotted in dashed line. Regression lines plotted for the top 800 points for each condition. B. Same as A but for in vivo evolutions, and in silico noise-driven evolutions matching the generation number or norm of each in vivo evolution.

Next, we examined the relationship between the collection of trajectories and the global geometry of the space (Sec. III.6.3). Previously, we reported that the latent space of the generator exhibits a highly anisotropic geometry, as quantified by the Riemannian metric tensor H of this image manifold (Wang and Ponce, 2021). The bilinear form $v^T Hv$ defined by this Riemannian metric tensor represented the degree of image change when moving in direction v. Consequently, moving in top eigen-directions changes the generated image much more than other directions, while moving in the bottom eigenspace scarcely affects the image (**Figure III-15** B,C). Moreover, this structure is homogeneous in the space, thus similar directions will cause rapid image change at different positions in the latent space. Thus, there exist *global dimensions* that affect the image a lot, and dimensions that do not. We hypothesized that the optimization trajectories might preferentially travel in some part of this eigenspace.

We used the averaged metric tensor H from Wang and Ponce (2021, (Wang and Ponce, 2021)). Its eigenvectors $U = [u_1, u_2, \dots u_d]$ with eigenvalues $\lambda_k, k = 1 \dots d$ became our reference frame for analyzing trajectories. We projected the collection of evolution directions $\{\zeta_j\}$ onto this basis, and examined the mean projection amplitude on each eigenvector $\frac{1}{P}\sum_i^P |u_k^T \zeta_i|$ as a function of eigenvalue λ_k (Figure III-4 A). Since for *in silico* experiments, the initial generation was the origin, this quantity was the average distance travelled along the eigenvector across runs. We expected the trajectories to travel further for eigen-dimensions where more CNN units exhibit "gradient".

We applied the Kolmogorov–Smirnov test to determine if the distribution of projection coefficient in the top and bottom eigendimensions were different. We found that the projection coefficients onto the top 800 eigen-dimensions as a distribution were significantly different from those onto bottom dimensions (KS statistics 0.223, $p = 9.3 \times 10^{-18}$, Figure III-4 A). Further, within the top 800 PC dimensions, the mean traveled distance strongly correlated with the log eigenvalues (Pearson correlation 0.738, $p = 2.8 \times 10^{-138}$, Figure III-4 A), i.e. the trajectories tended to travel further for dimensions with larger eigenvalues.

As for *in vivo* evolutions, we examined the collection of evolution trajectories ($N = 264$ from two monkeys, (Rose, Johnson, Wang and Carlos R. Ponce, 2021)) and projected them onto the Hessian eigenframe as above. We found that they preferentially aligned with the top eigenspace than the lower eigenspace (Figure III-4 B): the distance traveled in the top 800 eigenspace correlated with the eigenvalues (Pearson correlation 0.319, $p = 1.2 \times 10^{-15}$).

Since *in vivo* evolutions used a set of initial codes $\{z_0^{(i)}\}$ instead of the origin, we used noise-driven evolutions starting from these initial codes as control. As a baseline, the noise-driven evolutions had a

lower correlation between the distance traveled and the eigenvalues ($r = 0.198$ for the noise evolution with matching generations; $r = 0.223$ for that with matching code norm). Our results were robust to the choice of cutoff dimension (600-1000).

In conclusion, this result showed us that the evolution trajectories traveled further in the top eigenspace, and the average distance traveled was positively correlated with the eigenvalue. How do we interpret this effect? Since the top eigen-dimensions change the images more perceptibly, the tuning functions of neurons and CNN units were more likely to exhibit a "gradient" in such subspace. As the lower eigenspace did not induce perceptible changes, those dimensions barely affected the activations of units or neurons. Thus, no signal could guide the search in lower eigenspace, inducing dynamics similar to pure diffusion. As the visually tuned units could exhibit gradient in the top eigenspace, the search would be a diffusion with a driving force, allowing the optimizers to travel farther in the top eigenspace.

III.3.4 The Spherical Geometry of Latent Space Facilitated Convergence

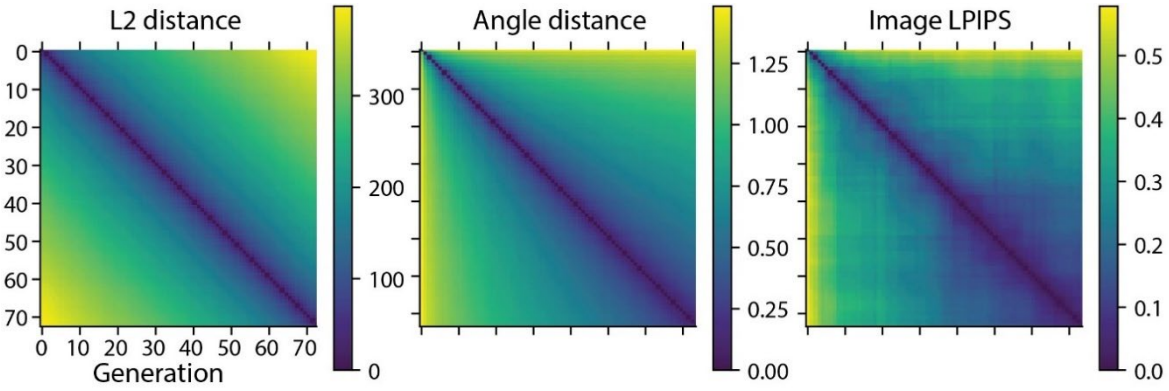


Figure III-5 **Angular Distance as a Better Proxy for Image Distance**. We show L2 and angular distance between pairs of mean codes $\bar{\boldsymbol{z}}_{\boldsymbol{t}}$ across 75 generations, and perceptual distance (LPIPS) between pairs of images generated from mean codes.

Finally, we noticed one geometric structure that facilitated the convergence of the CMA algorithm. For the DeePSim generator (Dosovitskiy and Brox, 2016b) with DCGAN architecture (Radford, Metz and Chintala, 2015), the mapping G was relatively linear. Namely, changing the scale of the input z mainly

affected the contrast of the generated image $G(z)$ (**Figure III-15**). Thus, when the base vector z_0 has a larger norm, travelling the same *euclidean distance* Δz will induce a smaller perceptual change; in contrast, traveling the same angular deviation $\Delta\theta$ will result in a similar pattern change regardless of the norm of the base vector z_0 (**Figure III-16**). We reasoned that the *angular distance* in latent space would be a better proxy to the *perceptual distance* across generated images.

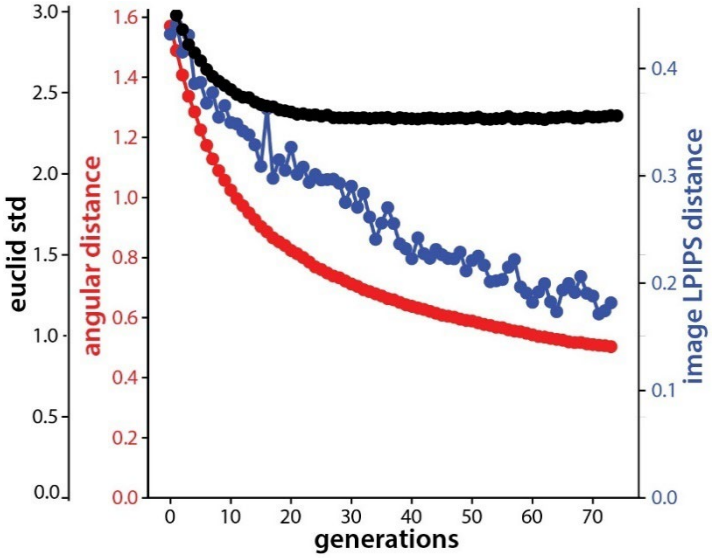


Figure III-6 Perceptual Variability Better Modelled by Angular Distance. We plot the average angular distance of latent codes within each generation; Euclidean standard deviation $\tilde{\boldsymbol{\sigma}}_{\boldsymbol{t}}$; and the mean LPIPS image distance within each generation.

We validated this using the perceptual dissimilarity metric *Learned Perceptual Image Patch Similarity* (LPIPS, (Zhang *et al.*, 2018)). We measured image variability in each generation using the mean LPIPS distance between all pairs. We measured the code variability by the standard deviation (std) of latent codes $\tilde{\sigma}_t$ averaged across d dimensions, which was an estimate of the step size σ_t and was strongly correlated with the mean L2 distance among codes. We also measured the mean angular distance between all pairs of codes in each generation. We found that the std of latent codes $\tilde{\sigma}_t$ decreased from $\tilde{\sigma}_0 = 3.0$ (first generation) to $\tilde{\sigma}_T = 2.31 \pm 0.04$ ($N = 1050$), and hit the floor at around 25 generations

(Figure III-6). In contrast, the perceptual variability kept decreasing till the end, which was better approximated by the trend of angular variability within each generation (from 1.57 rad to 0.50 rad, Figure III-6). This discrepancy of Euclidean and angular distance was mediated by the increasing vector norm during evolution: we found that the squared norm of the mean latent codes $\bar{z}_t$ scaled linearly with generation $\|\bar{z}_t\|^2 \propto t$ ($R^2 = 0.9985 \pm 0.008$, $N = 1050$, **Figure III-20**C), which is a classic property of a random walk. As a combined effect of the increasing vector norm and the decreasing step size σ_t, the angular variability within a generation decreased towards the end. Similarly, we found that across each given trajectory, angular distance was also a better proxy for the perceptual distance between images than L2 distance (**Figure III-5**). The mean-code images from late generations were more similar to each other than the ones from initial generations, as predicted by the angular distance between the mean vectors — but not L2 distance.

For a visual neuron or a CNN unit, the perceptual distances across images matter more. Thus, when designing the new optimizer, we reasoned it should control the angular step size as a proxy to perceptual distance, instead of the Euclidean distance (σ) (Sec. III.4).

III.4 Proposed Improvement: SphereCMA

Finally, we proposed an optimizer incorporating the findings above. We would test whether this new optimizer could perform as well as the original CMA algorithms.

This optimizer is designed to operate on a hypersphere, thus it was called SphereCMA (Algorithm 1). This design was guided by the training of the image generator G. Many generative models (GAN) were trained to map an isotropic distribution of latent codes $p(z)$ (e.g. Gaussian (Karras, Laine and Aila, 2019; Karras *et al.*, 2020) or truncated Gaussian (Brock, Donahue and Simonyan, 2018)) to a distribution of natural images. Thus, the generator G has only learned to map latent codes z sampled from this

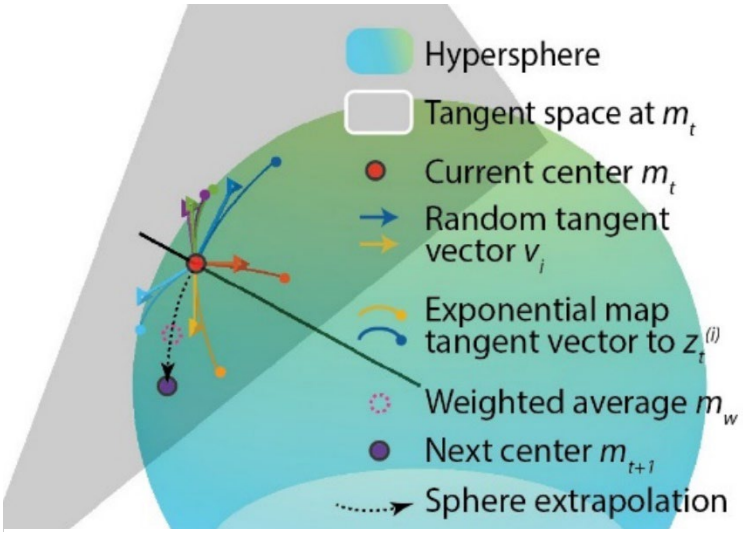


Figure III-7 Schematics of SphereCMA Update Procedure

distribution. In a high-dimensional d space, the density of standard normal distribution concentrates around a thin spherical shell with radius $R \approx \sqrt{d}$. Due to this, latent codes should be sampled from this hypersphere to obtain natural images. Thus, we enforced the optimizer to search on this sphere.

To design it, we leveraged the principles we learned from CMA-ES and the geometry of the latent space. First, since higher contrast images usually stimulate neurons or CNN units better, we set the spherical optimizer to operate at a norm comparable to that achieved by the end of a successful evolution experiment ($R = 300$), ensuring a high contrast from the start. Second, since the covariance matrix update did not contribute to the success of the CMA-ES algorithm in this application (Sec. III.3.1), we kept the algorithm exploring in an isotropic manner, without covariance update. To sample codes on the sphere, we used a trick from differential geometry, sampling vectors v_i isotropically in the tangent space of the mean vector m_t, and then mapping these tangent vectors onto the sphere as new samples $z_{t+1}^{(i)}$ (Figure III-7). Third, since the angular distance better approximated the perceptual change in the image (Sec. III.3.4), we controlled the perceptual variability in each generation by sampling codes with a fixed angular distance (the step size μ) to the mean vector. Fourth and final, as the latent vector norm was

fixed, the SphereCMA lacked the step size shrinking mechanism endowed by increasing code norm (Sec. III.3.4). We built in an angular step size decay function $\mu_{\mathrm{dec}}(t)$ modelling the dynamics in Figure III-6, to help convergence.

We tested this algorithm against other CMA-style algorithms in the same setup as in Sec. III.2.2. Specifically, we tested two versions with different step-size-decay function μ_{dec}, exponential (*Sp Exp*) and inverse function (*Sp Inv*). The change of angular step size through generations is shown in **Figure III-18**, the parameters for the decay function were tuned by Bayesian optimization and fixed.

We found that, SphereCMA with exponential decay (*Sp Exp*) outperformed all other CMA algorithms when pooled across all noise levels and layers (Table III-1, two-sample t-tests, $t_{2098} = 4.03, p = 5.7 \times 10^{-5}$ for CholeskyCMA, $t_{2098} = 4.92, p = 9.1 \times 10^{-7}$ for original CMA, $t_{2098} = 3.30, p = 1.0 \times 10^{-3}$ for DiagonalCMA, $N = 1050$). Compared to CholeskyCMA, SphereCMA surpassed its achieved activation by around 7%. Interestingly, when we compared the performance per noise level, we found that SphereCMA-Exp was the best performing algorithm under the noise-free condition ($\alpha = 0$) across the seven layers (two-sample t-test, $t_{698} > 9, p < 1 \times 10^{-20}$ for all of CholeskyCMA, original CMA and DiagonalCMA; runs were pooled across 7 layers, $N = 350$, same below). But in noisier conditions, it performed comparably or sometimes worse than other CMA optimizers: at $\alpha = 0.2, 0.5$ all

comparisons with other optimizers were not significant except for SphereCMA-Exp vs DiagonalCMA at $\alpha = 0.5$ ($t_{698} = -3.08, p = 2.2 \times 10^{-3}$). For optimization trajectory comparison, see **Figure III-9**.

Algorithm 1 SphereCMA

Require: Objective $f(.)$, space dimension d, radius R, population size B, step size decay function $\mu_{\mathrm{dec}}(.)$, learning ratio lr = 1.5

Subroutines ExpMap (Alg.2), SphereExtrapolate (Alg.3) RankWeight (Alg.4)

Isotropically sample B vectors $z_0^{(i)}$ with length R in $\mathbb{R}^d$
for $t = 0 : \mathrm{maxstep}$ **do**
  Evaluate objective function f for samples $r_i = f\left(z_t^{(i)}\right)$
  Compute weights w based on the rank of scores r.
  Weighted average the samples $m_w \leftarrow \sum_i w_i z_t^{(i)} / \sum_i w_i$
  Normalize m_w to norm R, $m_w \leftarrow R m_w / |m_w|$
  Fetch the current center vector $\mathrm{m_t} \leftarrow \mathrm{z_t^{(0)}}$
  Calculate the new center vector m_{t+1} by spherical extrapolation along the arc from m_t to m_w by lr.
  Sample B isotropic random vectors, e.g. $\mathrm{u_i} \sim \mathcal{N}(0, \mathrm{I^d})$.
  Project u_i to the tangent space of new center m_{t+1}. $v_i \leftarrow u_i - m_{t+1} m_{t+1}^T u_i / |m_{t+1}|^2$
  Get current angular step size $\mu \leftarrow \mu_{dec}(t)$
  Get new samples $z_{t+1}^{(i)}$ by exponential map from the center m_{t+1} along the tangent vectors v_i with angle μ. $z_{t+1}^{(i)} \leftarrow ExpMap(m_{t+1}, \mu V_{tang})$
  Add the new center vector to the population $z_{t+1}^{(0)} \leftarrow m_{t+1}$
end for

Overall, this result confirmed that the principles extracted from analyzing CMA-style algorithms were correct and crucial for their performance. By building these essential structures into our spherical optimizer, we could replicate and even improve the performance of CMA-ES algorithms. Finally, the interaction between the performance of SphereCMA-Exp and the noise level, provided us a future direction to find the source of noise resilience in the original CMA algorithm and improve on the SphereCMA algorithm.

III.5 Discussion

In this study, we addressed a problem common to both machine learning and visual neuroscience – the problem of defining the visual attributes learned by hidden units/neurons across the processing hierarchy. Neurons and hidden units are highly activated when the incoming visual signal matches their encoded attributes, so one guiding principle for defining their encoded information is to search for stimuli that lead to high activations. Since the brain does not lend itself to gradient descent, it is necessary to use evolutionary algorithms to maximize activations of visual neurons *in vivo*. Here, we identified a class of optimizers (CMA) that work well for this application and analyzed why they perform so well, and finally developed a faster and better optimizer (SphereCMA) based on these analyses. Here are some lessons we learned in this exploration. First, screening with comprehensive *in silico* benchmarks accelerates the algorithm development *in vivo*. Secondly, geometry of latent space matters. As a general message, when developing optimizers searching in the latent space of some generative models, we should pay attention to the distance structure i.e. geometry of the generated samples (e.g. images) instead of the latent space. Optimizers leveraging space geometry shall perform better. We hope our workflow can help the optimizer design in other domains, e.g. search for optimal stimuli in other sensory modalities and drug discovery in molecular space.

Acknowledgments

We appreciate Mary Burkemper in her assistance in monkey managing and data collection. We thank Yunyi Shen, Kaining Zhang for reading the manuscript and providing valuable feedbacks. We are grateful to the RIS cluster and staff in Washington University in St Louis for their GPU resources and technical support.

B.W. is funded by the McDonnell Center for Systems Neuroscience in Washington University (pre-doctoral fellowship to B.W.). Additional research funding by the David and Lucile Packard Foundation and the E. Matilda Ziegler Foundation for the Blind (C.R.P.).

III.6 Appendix

III.6.1 Details of Pre-trained CNN Models

In this work, we used units from two pretrained CNN models, AlexNet and ResNet50-robust. Both were pre-trained for object recognition on ImageNet (Deng *et al.*, 2009).

AlexNet: The architecture and weights were fetched from torchvision.modelzoo. As for layers, we used the activations from conv2-fc8 layer, the activation were non-negative (post ReLU rectification) except for fc8, which we used the logits.

ResNet50-robust: The architecture were defined in torchvision.modelzoo, with the weights fetched from https://github.com/MadryLab/robustness. We used the version with adversarial training setting: L_∞ norm with strength $8/255$. It has been noticed that the adversarially trained network exhibited more perceptually aligned gradients, the direct gradient descent from the units could generate impressive feature visualizations. We used this model in our benchmark since the achieved activations were harder to be affected by "adversarial" patterns generated by the generator. For layers, we used the output of last Bottleneck block from layer1-4, namely layer1.Bottleneck2, layer2.Bottleneck3, layer3.Bottleneck5, layer4.Bottleneck2, and the linear layer before softmax.

Table III-2 Statistical Test of Factors Affecting of CMA-type Algorithm Performance Model formula in R style: $\mathrm{score} \sim \mathrm{C(optimizer)} + \mathrm{noiselevel} + \mathrm{layernum} + \mathrm{C(optimizer)}{:}\,\mathrm{layernum} + \mathrm{C(optimizer)}{:}\,\mathrm{noiselevel}$. The factor optimizer had three categorical levels: CholeskyCMA, CMA, DiagonalCMA, noiselevel had three continuous values: 0.0, 0.2, 0.5, layernum had seven continuous values, 2 to 8 encoding conv2 to fc8. Data were subset of the benchmark dataset 2, Tab.1.

	sum sq	df	F	$Pr(>F)$

optimizer	0.14	2	3.40	3.36×10^{-02}
layernum	19.14	1	953.56	4.39×10^{-183}
noiselevel	43.56	1	2170.60	0.0
optimizer:layernum	0.60	2	14.98	3.33×10^{-07}
optimizer:noiselevel	0.19	2	4.66	9.58×10^{-03}
Residual	63.03	3141		

III.6.2 Detailed Methods for *in vivo* Evolution Experiments

Here we detailed the method by which we conducted *in vivo* Evolution experiments, same as that in (Ponce *et al.*, 2019b; Rose, Johnson, Wang and Carlos R. Ponce, 2021).

Two macaque monkeys (A,B) were used as subjects with Floating Multielectrode Array (FMA) implanted in their visual cortices, in V1/V2, V4 and posterior IT. In an electrophysiology experiments, the electric signal recorded in each electrode (i.e. channel) will be processed and the spikes will be detected from it using an online spike sorting algorithm from Plexon. These spikes represented the output from a few neurons or a local population in the visual cortex. In an *in vivo* session, we first performed a receptive field mapping experiment. An image was rapidly (100ms duration) showed in a grid of positions in the visual field. The spike times following the stimuli onset will be binned into a histogram, i.e. post-stimulus time histogram (PSTH). We measured the spatial extant where the image evoked neuronal responses above the baseline level. Based on the PSTH of the recorded units, we selected a responsive unit, with a well-formed receptive field as our target unit. This was the *in vivo* counterpart of the unit we selected in CNN *in silico*.

Finally, we performed the Evolution experiment. During the experiment, the images would be presented to the animal subjects on the screen in front of them, centered at the receptive field found previously; each image were presented 100ms followed by a 150ms blank screen. The spike count in $[50, 200]$ ms time window after the stimulus onset was used as the score for each image r_i. A set of 40 texture images from (Freeman and Simoncelli, 2011) were inverted and then generated from the generator G; they were used as the initial generation $\{z_i^0\}$. After the neuronal responses to all images in a generation were recorded, the latent codes $\{z_i^t\}$ and recorded responses $\{r_i\}$ were sent to the optimizer, which proposed the next set of latent codes $\{z_i^{t+1}\}$. These codes were mapped to new image samples $\{G(z_i^{t+1})\}$ which were showed to animal subjects again. This loop continued for 20-80 rounds until the activation saturated or the activation didn't increase from the start. When we compared two optimizers (CMA vs GA in Sec. III.2.3), we ran them in parallel, i.e. interleaving the images that were proposed by the two optimizers. This ensured that the performance difference of the two optimizers were not due to change in signal quality or in electrode position.

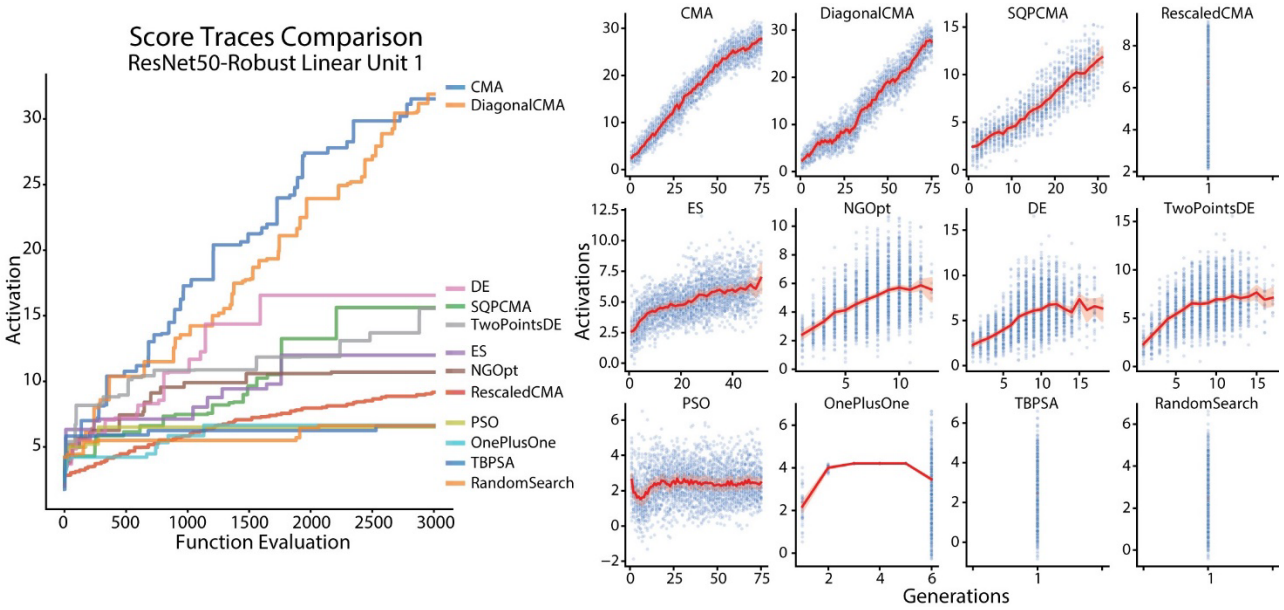


Figure III-8 Figure A.1: Score Traces Comparison of Nevergrad Optimizers with an Example Unit. Left: Cumulative max score as as a function of number evaluation for each optimizer. Right: Scores of images plotted as a function of generations for each optimizer. The red solid trace and shading represented the mean and sem of scores each generation. Note the vast difference in the total generation number among optimizers. As the highest performing optimizers, CMA and DiagonalCMA also used the largest number of generations. All runs were performed with the unit 1 (0-indexing) in the final Linear layer of ResNet50-robust, without noise.

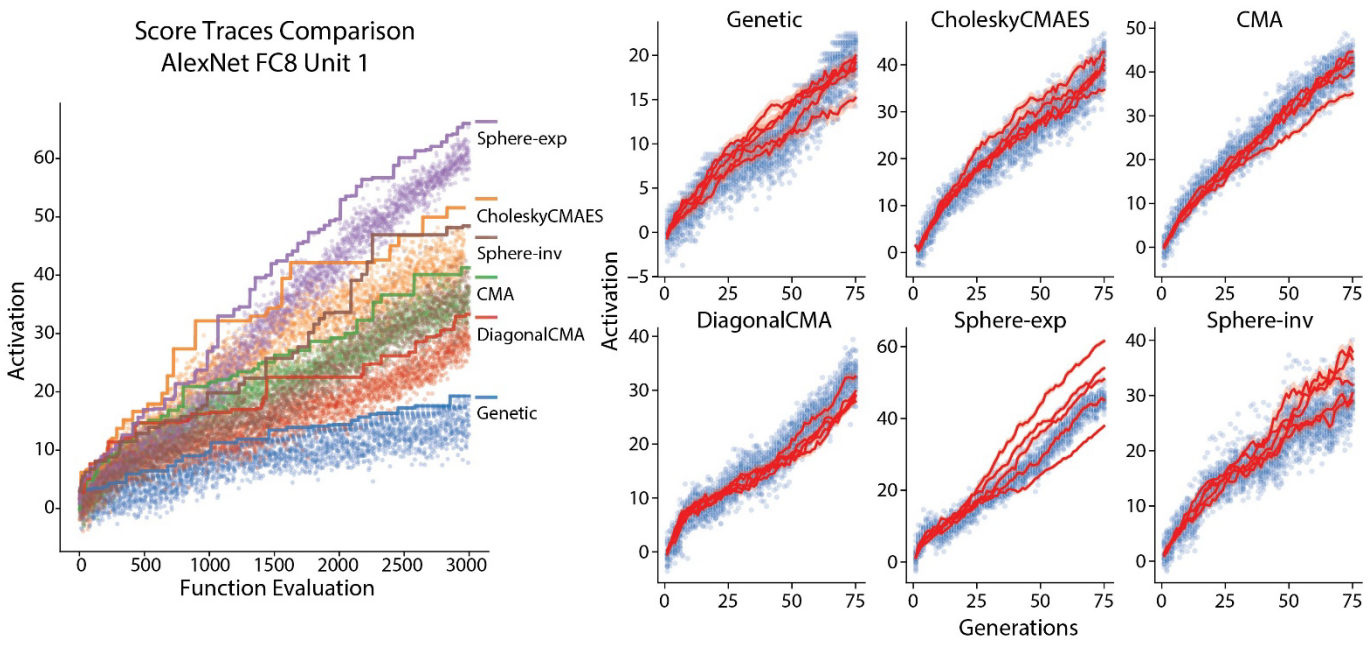


Figure III-9 Figure A.2: Score Traces Comparison of GA, CMA and Spherical Optimizers with an Example Unit. Left: Cumulative max score as a function of number of evaluation (image presentation), with scores of individual images showed as the cloud with same color. Right: Scores as a function of generations. The red solid trace and shading represented the mean and sem of scores each generation, 5 runs with different random seeds were showed. All runs were performed with the unit 1 (0-indexing) in the final fc8 layer of AlexNet, without noise.

Figure III-10 Figure A.3: Comparison of Generated Images from the Code with Highest Score for each Optimizers The objective function was the noise-free unit 3 in fc layer of ResNet50-Robust which corresponds to ImageNet label tiger shark. Arguably, the images with highest scores (noted in title) found by CMA and DiagonalCMA contained features we associated with "tiger shark", which was consistent with the function of the target unit – to classify images as tiger shark.

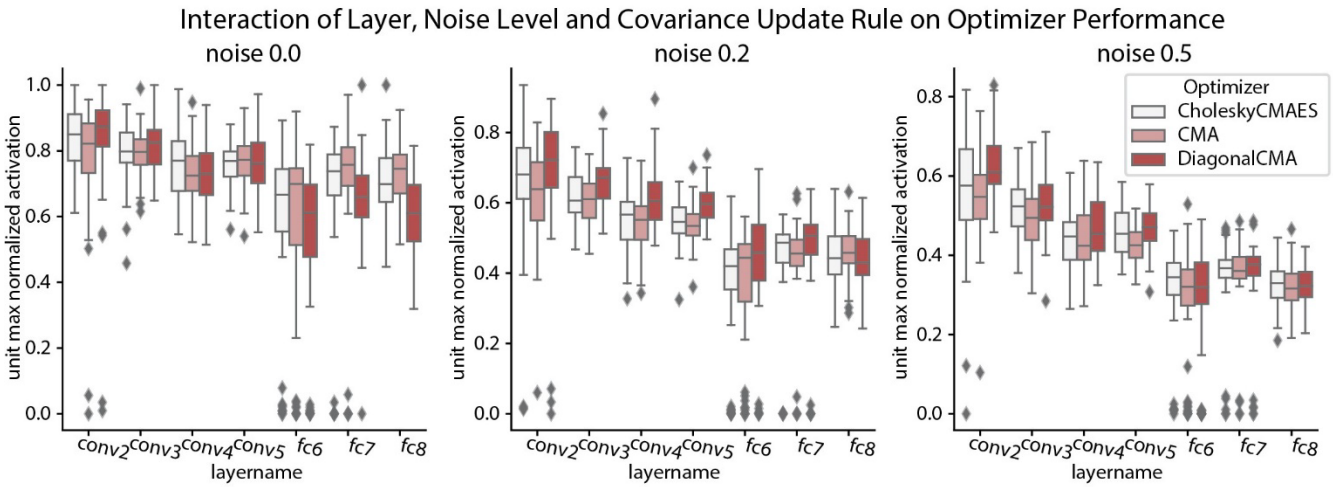


Figure III-12 Figure A.5: Interaction between unit depth, noise level and covariance update rules on optimizer performance in CNN. Plotted with the same experimental data as Tab. 1. Same plotting convention as Fig. A.4. The statistical test for interactions showed in Tab. A.1.

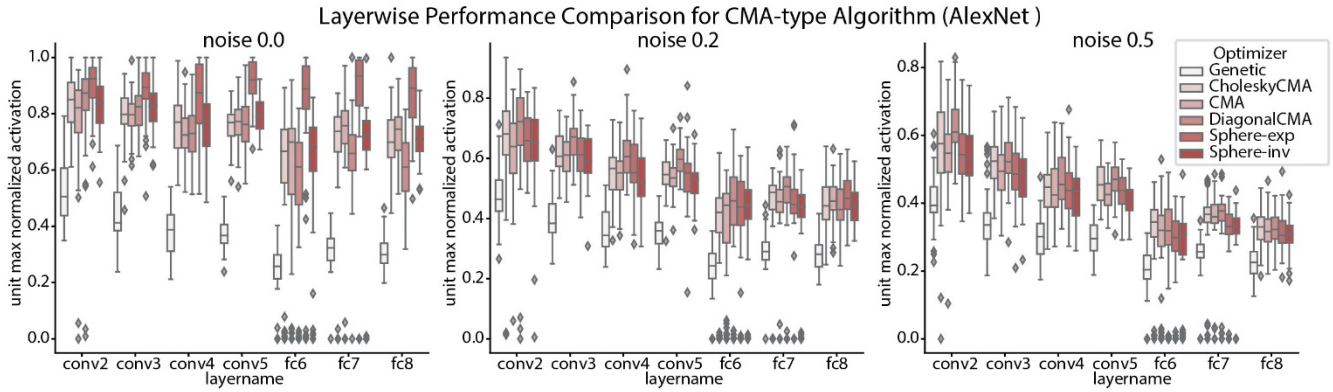


Figure III-11 Figure A.4: Layer-wise performance comparison of CMA-style algorithms. Plotted with the same data as Tab. 1. Box and inset represent quartiles of the normalized scores for units in each layer.

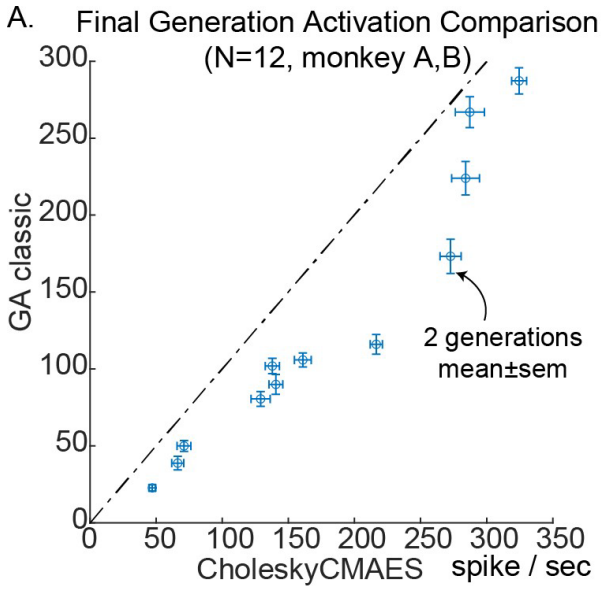


Figure III-14 Figure A.6: Raw Activation Achieved by CMA and GA in *in vivo* Comparison A. Comparing the response amplitude (firing rate in 50-200ms with baseline rate subtracted) in the last 2 generations for the two optimizers, mean ± sem were showed.

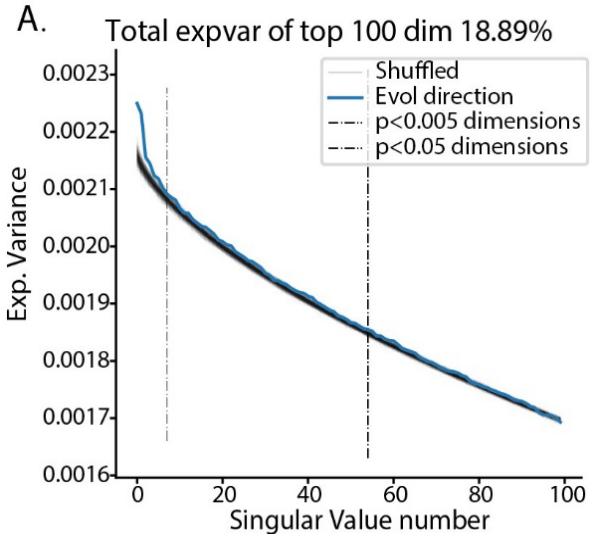


Figure III-13 Figure A.7: Collection of Evolution Trajectories were Lower Dimensional. A. Comparing the SVD of the collection of in silico evolution directions and the collection of shuffled directions.

Table III-3 Table A.2: Performance Comparison of Nevergrad Optimizers per Layer and Noise-level. CMA and DiagonalCMA excelled at all noise level and layers than other algorithms. The performance gaps were larger for deeper layers and higher noise level.
Abbreviations: RN50: ResNet50-robust; L1-Btn2: layer1-bottleneck2. Name of optimizers: Diag: DiagonalCMA; SQP: SQPCMA; Rescaled: RescaledCMA; NG: NGopt; 2pDE: TwoPointsDE; 1+1: OnePlusOne; Rand: RandomSearch.
These full names could be used to fetch optimizers from the registry of nevergrad package by *ng.optimizers.registry[optimname]*. Due to the limitation of interface of nevergrad, we didn't record the clean score before adding noise. This should make the score distribution noisier but will not change the rank of the algorithms.

net-layer	α	CMA	Diag	SQP	Rescaled	ES	NG	DE	2pDE	PSO	1+1	TBPSA	Rand
RN50-L1-Btn2	0.0	0.478	0.473	0.345	0.218	0.325	0.359	0.347	0.358	0.222	0.237	0.209	0.213
	0.2	0.599	0.582	0.426	0.240	0.416	0.425	0.417	0.428	0.274	0.322	0.255	0.252
	0.5	0.809	0.801	0.558	0.326	0.572	0.555	0.575	0.607	0.376	0.443	0.367	0.361
RN50-L2-Btn3	0.0	0.545	0.534	0.346	0.318	0.331	0.351	0.360	0.370	0.197	0.203	0.208	0.208
	0.2	0.637	0.644	0.401	0.326	0.410	0.411	0.392	0.449	0.220	0.281	0.249	0.250
	0.5	0.831	0.825	0.529	0.442	0.565	0.574	0.520	0.606	0.303	0.409	0.335	0.338
RN50-L3-Btn5	0.0	0.578	0.567	0.344	0.174	0.230	0.283	0.279	0.301	0.111	0.121	0.131	0.128
	0.2	0.619	0.600	0.357	0.173	0.270	0.311	0.308	0.325	0.123	0.176	0.142	0.138
	0.5	0.846	0.787	0.458	0.203	0.370	0.422	0.411	0.411	0.164	0.234	0.206	0.205
RN50-L4-Btn2	0.0	0.624	0.631	0.259	0.139	0.188	0.236	0.227	0.243	0.096	0.104	0.090	0.095
	0.2	0.667	0.656	0.272	0.124	0.217	0.238	0.243	0.258	0.105	0.134	0.106	0.106
	0.5	0.724	0.740	0.267	0.130	0.279	0.294	0.316	0.335	0.137	0.192	0.149	0.141
RN50-fc	0.0	0.721	0.651	0.311	0.225	0.190	0.214	0.238	0.256	0.090	0.096	0.093	0.095
	0.2	0.664	0.678	0.336	0.151	0.218	0.234	0.242	0.252	0.098	0.132	0.108	0.107
	0.5	0.760	0.753	0.351	0.163	0.292	0.286	0.304	0.324	0.135	0.171	0.139	0.144

Alex-conv2	0.0	0.592	0.585	0.254	0.100	0.236	0.317	0.324	0.338	0.087	0.135	0.075	0.081
	0.2	0.643	0.657	0.278	0.080	0.284	0.387	0.356	0.371	0.096	0.181	0.086	0.087
	0.5	0.792	0.751	0.331	0.093	0.374	0.421	0.438	0.423	0.106	0.242	0.107	0.112
Alex-conv3	0.0	0.658	0.657	0.327	0.125	0.214	0.306	0.310	0.298	0.083	0.092	0.078	0.075
	0.2	0.689	0.697	0.296	0.100	0.245	0.298	0.316	0.307	0.083	0.140	0.084	0.080
	0.5	0.822	0.762	0.314	0.112	0.336	0.376	0.370	0.367	0.110	0.184	0.106	0.106
Alex-conv5	0.0	0.710	0.661	0.342	0.014	0.186	0.259	0.271	0.267	0.059	0.073	0.067	0.064
	0.2	0.676	0.656	0.362	0.012	0.227	0.286	0.262	0.267	0.068	0.113	0.072	0.072
	0.5	0.798	0.764	0.438	0.018	0.295	0.346	0.343	0.335	0.090	0.163	0.089	0.094
Alex-fc7	0.0	0.824	0.722	0.318	0.117	0.202	0.259	0.277	0.276	0.086	0.121	0.084	0.082
	0.2	0.688	0.617	0.324	0.080	0.242	0.273	0.281	0.277	0.097	0.159	0.093	0.091
	0.5	0.749	0.739	0.354	0.098	0.327	0.381	0.351	0.377	0.136	0.219	0.126	0.122
Alex-fc8	0.0	0.814	0.768	0.334	0.199	0.173	0.224	0.223	0.228	0.069	0.089	0.089	0.087
	0.2	0.680	0.672	0.351	0.137	0.205	0.221	0.237	0.243	0.076	0.117	0.098	0.099
	0.5	0.744	0.695	0.382	0.127	0.262	0.304	0.295	0.308	0.100	0.168	0.140	0.136
Average		0.700	0.677	0.352	0.159	0.289	0.328	0.327	0.340	0.133	0.181	0.139	0.139
Runtime(s) RN50	126.1	45.5	129.6	133.5	38.9	69.2	69.5	39.3	454.3	38.3	38.5	38.5	
Std	38.2	5.9	34.2	39.6	2.9	5.4	5.5	2.9	31.0	2.7	2.9	2.8	
Runtime(s) Alex	104.5	23.0	118.6	107.1	16.2	48.8	47.9	16.8	435.4	16.8	17.5	17.6	
Std	30.0	4.5	67.7	31.1	1.5	4.8	4.8	1.6	36.1	1.6	2.2	2.2	

III.6.3 Geometry of the Generative Image Manifold

As showed in (Wang and Ponce, 2021) (Chapter II), the deep neural network image generators usually have a highly anisotropic latent space. We visualized it in **Figure III-18** B,C. When travelling the same distance along an eigenvector, it induced a much larger perceptual change in the image when the eigenvalue was larger; and eigenvectors like the rank 1000 one (**Figure III-18** B) merely changed the image in the range tested. Thus, these lower eigenvectors formed an effective "null space" where the input was not meaningful to the generated images or the activation of the units or neurons. In later development of optimization algorithms (Wang and Ponce, 2021), we projected out the lower eigen-dimensions and only optimized the top eigen-dimensions of the generator. This could reduce the search dimensionality to around 500d.

As for the spherical geometry of the generative image mapping G, it can be seen from **Figure III-18**, scaling the latent vectors will majorly change the contrast instead of content of the generated image. This made angular distance relevance to our generator. However, unlike anisotropy, this linearity of G is not a general property of generators. For other image generators such as BigGAN, StyleGAN (Brock, Donahue and Simonyan, 2018; Karras, Laine and Aila, 2019; Karras *et al.*, 2020), scaling the latent vector will not result in a scaling in pixel space: the image content will also change. We suggest that for these generator spaces, the code norm should not be fixed but should be varied and explored during optimization. So, when developing optimizers in those latent spaces, as code norm increases automatically for CMA type algorithm due to diffusion, unconstrained evolution will bias the generated image towards those images with larger code norms.

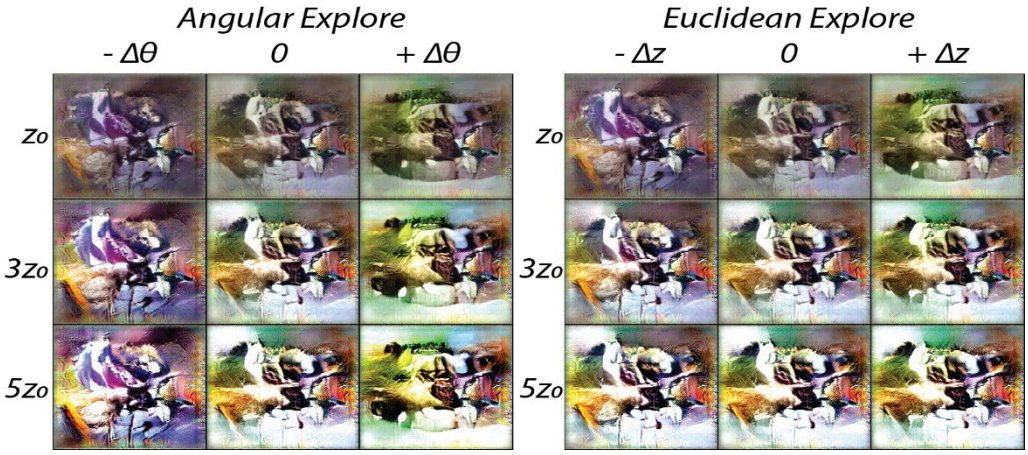


Figure III-15 Figure A.8: Example of Exploration with fixed distance versus fixed angle. Images generated from a random vector $\boldsymbol{z_0}$ and its scaled versions $\boldsymbol{3z_0}, \boldsymbol{5z_0}$ were showed in the center columns. Left panel visualized the perturbed vectors by the same angle $\boldsymbol{\Delta\theta}$; Right panel showed the perturbed vectors by a fixed vector $\boldsymbol{\Delta z}$. The first rows in two columns the same. In left panel, the columns are more similar; while in right panel, the images in the 2nd and 3rd rows were more similar. We argued that with perturbation of the the same Euclidean length, the effect on image will decrease for a base vector of larger norms. But perturbation of the same angle will result in similar perceptual change regardless of the norm of the base vector. Also see illustration in Fig. B.1A

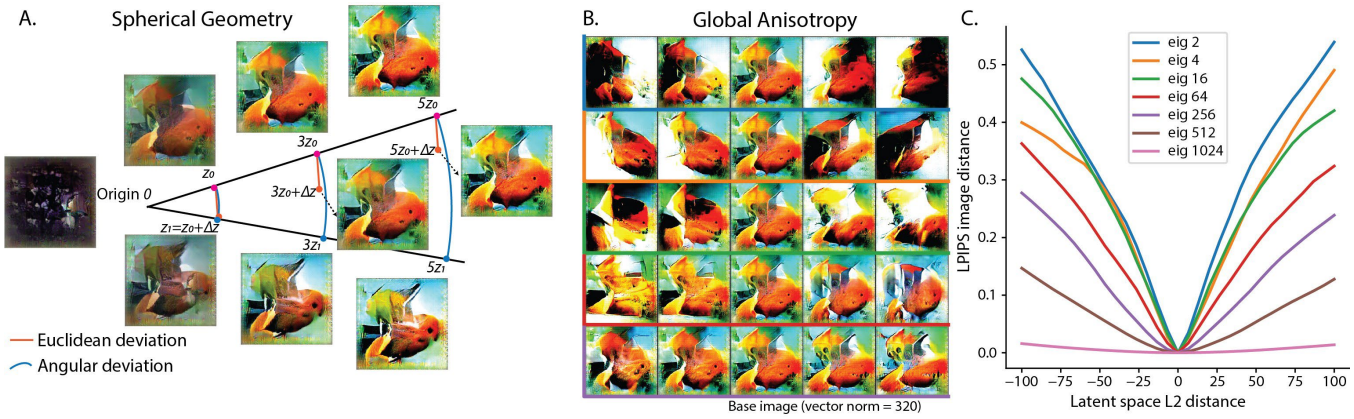


Figure III-16 Figure B.1: Spherical Geometry and Anisotropy of the Generative Image Space. A. illustrates the spherical geometry of generative space. B. illustrates the anisotorpic structure of the generative space. Each row visualizes the effect on generated image by perturbing the reference code $z0$ along an eigenvector by L2 distances $\boldsymbol{d = -100, -50, 0, 50, 100}$, i.e. $\boldsymbol{z_0 + du_k}$. The rows correspond to five eigenvectors (number $\boldsymbol{k = 2, 4, 16, 64, 256}$), showed in descending order of their eigenvalues. C. The LPIPS image distance as a function of the Euclidean distance along each eigen dimensions. The images in B visualized the perceptual meaning of the LPIPS distances in C. Structure of B.C. was inspired by Fig.1 in [31].

III.6.4 Geometry of the Tuning Function Landscape

The geometry of the latent space is important for the effectiveness of optimization algorithms. More important is the geometry of the objective function in this space. As a foundation, we characterized the geometry esp. curvature of the tuning functions of *in silico* units. Recall the tuning function of unit in the image space $f: \mathcal{I} \to \mathbb{R}$, and the image generative map $G: \mathbb{R}^d \to \mathcal{I}$. The effective function to be optimized in our problem was their composition $f \circ G: \mathbb{R}^d \to \mathbb{R}$, we measured its curvature by computing the Hessian spectra of $f \circ G$. Intuitively, this eigenvalue represented the sharpness of tuning along the eigenvector in the latent space: a larger eigenvalue corresponds to a sharper tuning and a smaller eigenvalue corresponds to a broader tuning.

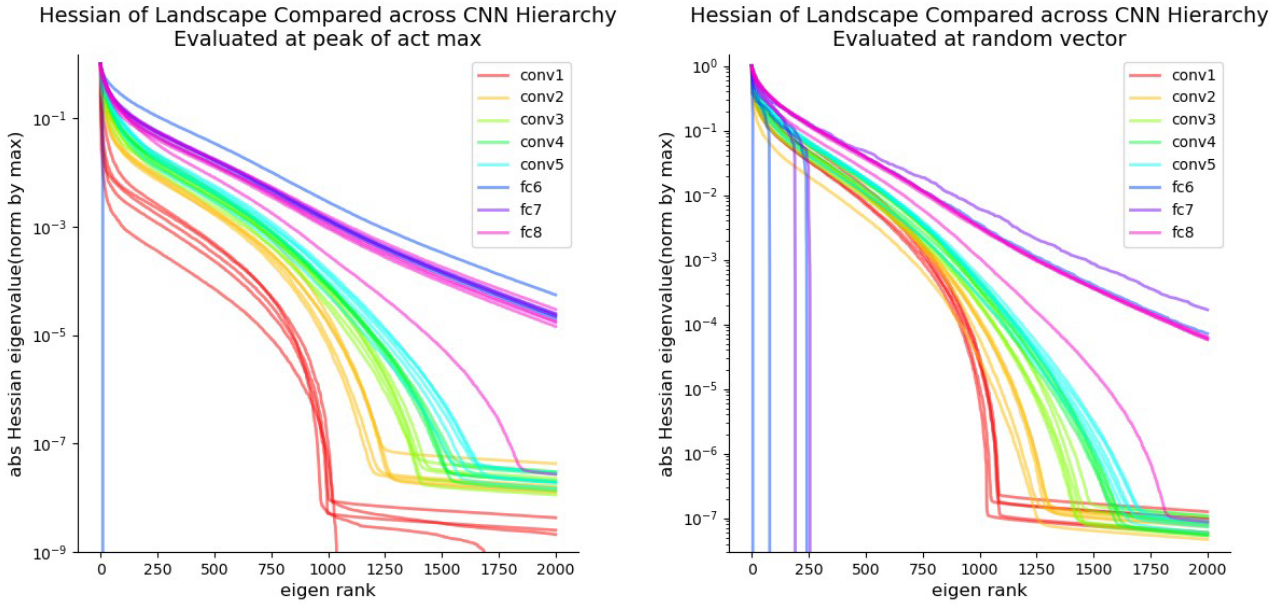


Figure III-17 **Figure C.1: Hessian Spectrum of Units Across AlexNet Hierarchy. Left.** Evaluated at the peak i.e. the final generation latent code of a CMAES search. **Right**. Evaluated at random initialized vectors. It could be appreciated that conv1 had a sharp decay in its spectra while the spectra of conv2, conv3, conv4 units got gradually higher dimensional. Fully connected layers fc6-fc8 all exhibited higher dimensional and slower decay spectra.

We selected 5 units from each of the 8 layers of AlexNet, and then searched for a local maximum z^* for each unit using CholeskyCMAES optimizer. Next, we calculated the eigen-decomposition of the Hessian $H = \frac{\partial^2 (f \circ G)}{\partial z \partial z}\Big|_{z^*}$ evaluated at the local optima. We borrowed the technique from (Wang and Ponce, 2021): we represented the Hessian matrix implicitly by constructing a Hessian vector product (HVP) operator, and then applied Lanczos algorithm on this linear operator to extract the top 2000 eigen-pairs with the largest absolute eigenvalues. We examined the absolute eigen spectra, normalized by the largest absolute eigenvalue. This was reasonable since the overall scale of eigenvalues was related to the scale of activation of the unit, which differed from layer to layer.

We found that all the spectra were highly "ill-conditioned" spanning eigenvalues of 4 to 9 orders of magnitude. This was consistent with the findings in (Wang and Ponce, 2021). The image generative network G is already a highly ill-conditioned mapping, such that some latent dimensions affect the image much more than other dimensions. Since the tuning function $f \circ G$ in the latent space is a composite of f and G, the Jacobians of the two mappings multiply, so it inherited this ill-condition of the generative map.

More interestingly, as we compared the spectra of tuning landscapes across layers, we found that the spectra decayed rapidly for units in conv1 and decayed slower going up the hierarchy from conv2 to conv5, and became much slower for units in fully connected layers fc6-fc8 (Figure III-17). We found similar spectral result when we evaluated the Hessian at random vectors z_{rnd} instead of the peaks found by optimizers z^*, confirming that this was a general phenomenon of the landscape. Though we noted that, the spectra of fc layer units precipitated sometimes, presumably because the unit was not active enough.

We interpreted this as units in early convolutional layers were tuned for very few dimensions in the latent space, and they were relatively invariant for a large subspace. In contrast, the units deeper in the hierarchy were tuned for a larger dimensional space, and were selective for more specific combination of features.

This observation was relevant to the performance of optimizers with different covariance update rules (**Table III-1**). For units in earlier conv layers (e.g. conv1-5), there was a large invariant subspace, so the peaks of optimization were easy to find, so optimizers don't need to find or model the exact tuning axis. In contrast, for units that tuned for hundreds-thousands of dimensions (fc6-8) all these axes need to be be jointly optimized to achieve the highest activation. Because of this, we reasoned that the diagonal scaling of covariance matrix may not work well in the deeper layers like fc6-8. This could be one explanation for why DiagonalCMA performed better than CMA in earlier layers, but not as well in deeper layers (**Figure III-12**, **Table III-1**).

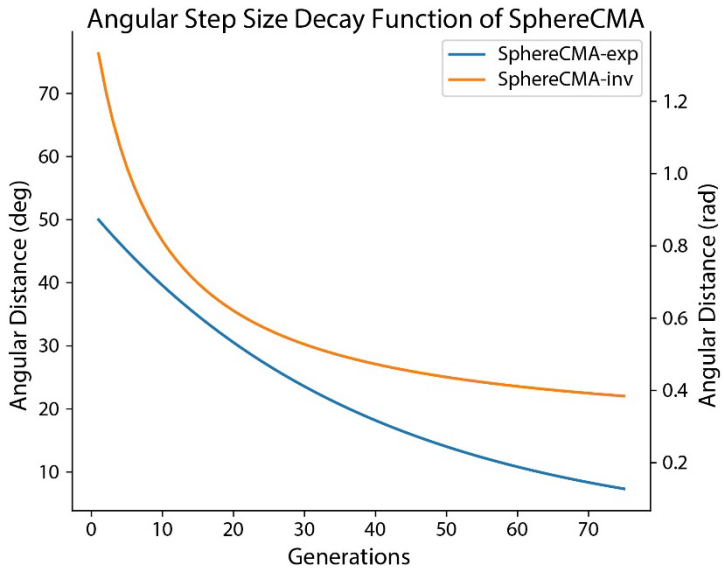


Figure III-18 Figure A.9: Angular step size decay curves for SphereCMAExp and SphereCMA-Inv.

III.6.5 Additional Analysis: Dimensionality of the Collection of Evolution Trajectories

We tested if the collection of evolution trajectories exhibited a lower dimensionality than shuffled controls. We collected the evolution trajectories of $N = 1050$ runs, across all noise level and layers and their PC structure. Since we observed that the mean trajectories could be well represented by their top PCs, we represented each search trajectory by a single vector, evolution direction ζ. We computed this direction ζ by projecting the mean latent code of final generation $\overline{z}_T$ onto the top 5 PCs space $\zeta = \mathcal{P}_{PC1:5}(\overline{z}_T)$ (which explained over 88% of the variance of the mean trajectory). We shuffled the entries of these vectors to form a control collection of trajectories $z_{shuffle}$, preserving their vector norms. We shuffled 500 times to establish statistical significance.

Generally, the collection of evolution directions were *not low dimensional*, with top 100 dimensions accounting for only 18.9% of variance (**Figure III-14**A). But comparing to the

shuffled counterparts, the evolution trajectories resided in a (slightly) lower dimensional space: The first 7 singular values were larger than the shuffled counterpart ($p < 0.005$, **Figure III-14**A). This signature suggested that the search trajectories exhibited some common lower dimensional structure.

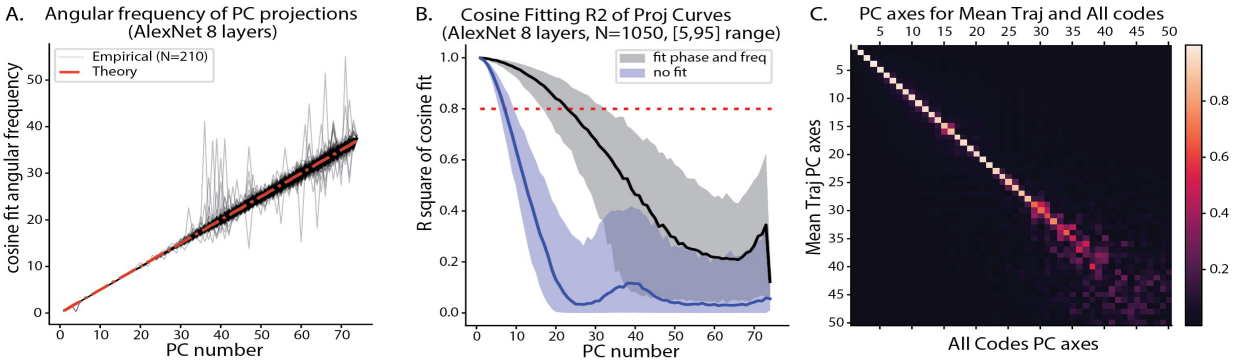


Figure III-19 Figure D.1: Sinusoidal Geometry of Evolution Trajectory as Signature of Random Walk (continued). A. Angular frequency $\boldsymbol{\omega}$ of the cosine fitting as a function of PC number k, with the theory value $\boldsymbol{\omega = k/2}$. $\boldsymbol{N = 210}$ evolutions were plotted. B. $\boldsymbol{R^2}$ of cosine function for the $\boldsymbol{k}$th PC projection. Central line showed the median, shaded area showed the $\boldsymbol{[5, 95]}$% confidence interval. C. Top PCs for the mean trajectory and top PCs for all latent codes were well aligned.

III.6.6 Additional Analysis: PCA of Evolution Trajectory Formed by All Codes also Exhibited Sinusoidal Structure

In addition to the analysis, we did to the mean search trajectory as in Sec. III.3.2, we considered how individual latent codes sampled during Evolution fit into the global geometry of the trajectory. We stacked all the latent codes $\left\{z_t^{(i)}\right\}, t = 1\ldots, T, i = 1, \ldots n_P$ sampled by the CholeskyCMA during Evolution into a $T \cdot n_P$ -by-d matrix $\tilde{X} = \left[z_1^{(1)}, z_1^{(2)}, \ldots, z_T^{(n_P)}\right]$ ($T \cdot n_P = 3000$) and applied PCA to decompose it (n_P is population size, T is number of generations, d is latent space dimension). We found that the main structure of the search trajectory remained intact: the codes projected onto top principal axes were still sinusoidal curves but with dispersion. We compared the PC vectors computed from all latent codes, and those from the

mean trajectory by calculating their the inner product or cosine angles. We found that the top 30 PC vectors of two basis were matched to a high degree: 28 PC vectors of the mean trajectory had cosine angle > 0.8 with the PC vector of the same rank for all latent codes (**Figure III-19**C). This showed that the global structure of all sampled codes coincided with the mean search trajectory, while the individual codes contributed to local variability around the mean search trajectory.

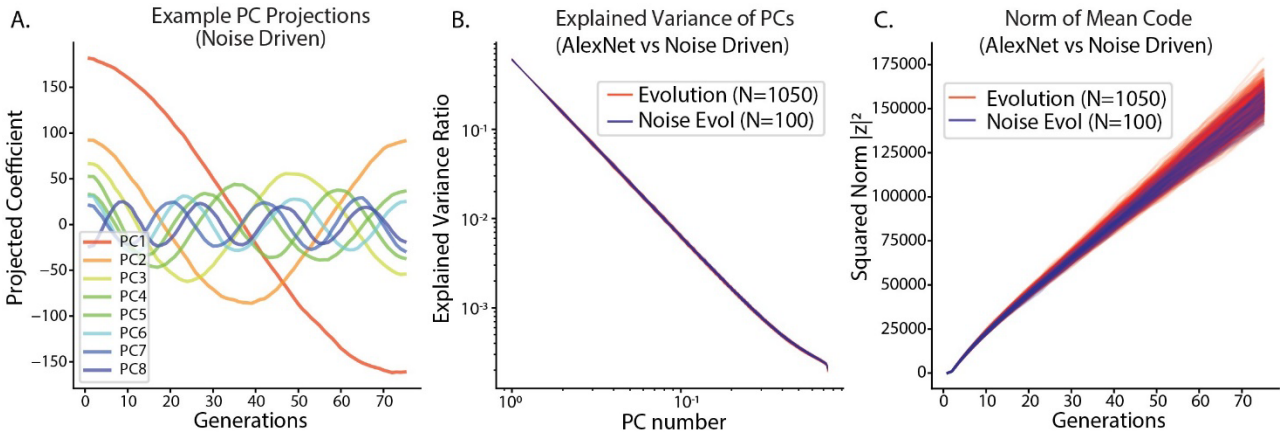


Figure III-20 Figure D.2: PCA of Noise-Driven Evolution also Exhibits Sinusoidal Structure A. Mean search trajectory projected onto the first 8 PCs of a noise-driven Evolution. B. Explained variance as a function of PC number k for noise driven evolution and real evolution coincided well. The existence of driving force seems not to have a large effect on the spectrum of Evolution trajectories. C. Code norm during evolution for noise driven and AlexNet units driven evolution. For both, the squared norm scaled linearly with the Generation. At the final generation, the norm of latent codes was higher for AlexNet driven evolution: Mean norm in evolution experiments was $\mathbf{394.80 \pm 7.05}$ (mean ± std) in contrast to the noise-driven control 390.96 ± 7.09 ($t = 5.257, p = 1.7 \times 10^{-7}, df = 1151$). The difference in norm is significant, but the effect size is small.

III.6.7 Additional Analysis: 2D PC Projections of Evolution Trajectory were Lissajous Curves

In our work, we consistently observed that the projection of original CMA, CholeskyCMA and Diagonal CMA evolution trajectory onto top PC axis was sinusoidal curve. One interesting way to visualize these trajectories is to project it onto 2d PC subspaces. As expected, we saw the Lissajous curve formed by the projection (**Figure III-21**). We noted that the Lissajous curves

decreased its amplitude towards the end (red), so the red end didn't overlap perfectly with the blue end. This is in line with a decreasing step size during evolution. As a comparison, we also performed PCA and 2d projection with optimization trajectories driven by SphereCMA algorithm (**Figure III-22**). We can see, it deviated more from the sinusoidal structure predicted by random walk, indicating a different kind of search process on the sphere.

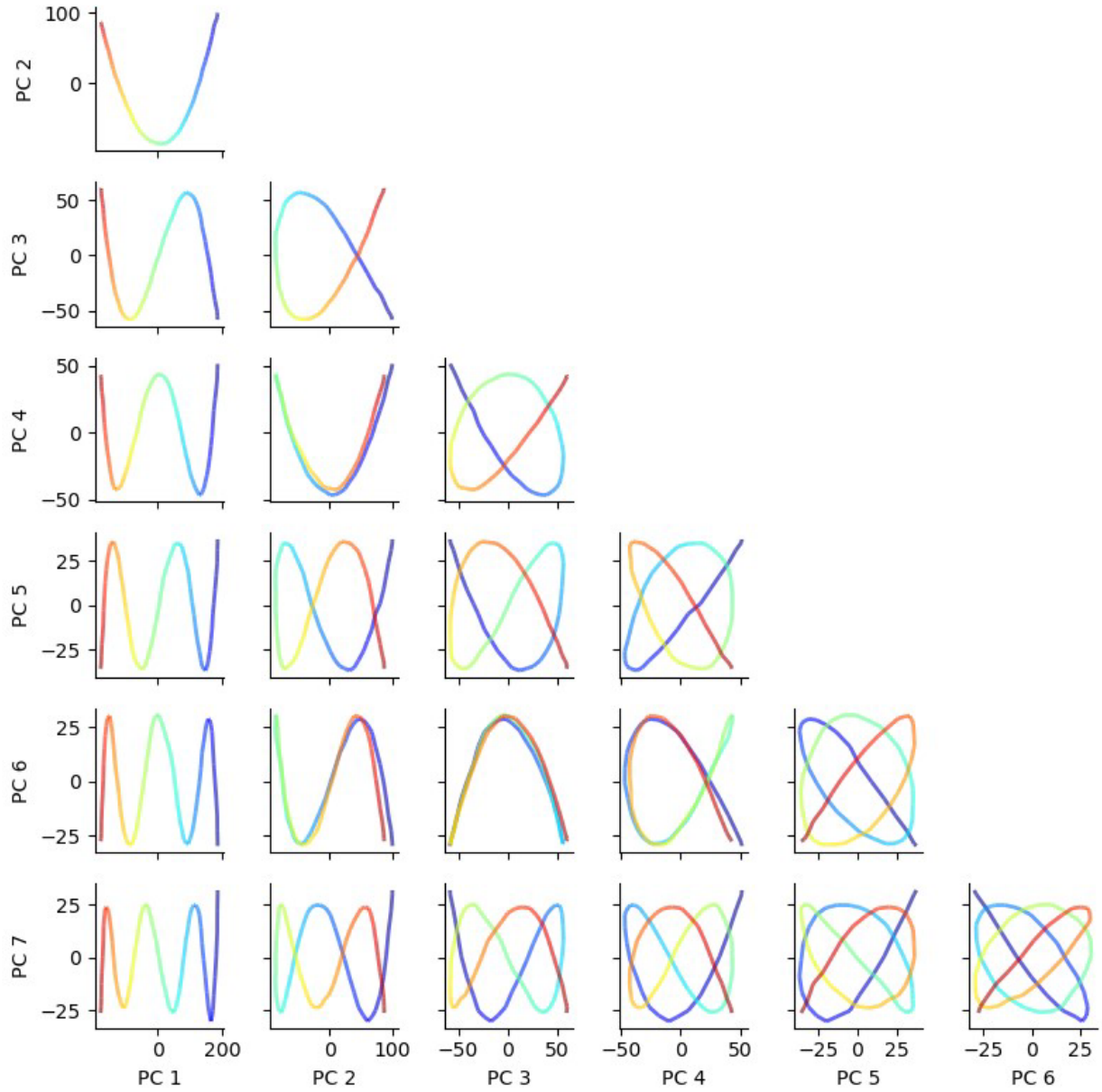


Figure III-21 Figure D.3: Lissajous Curves formed by PC Projection of Evolution Trajectory Blue to red color gradient represents the progression of time, from first to the last step. This is obtained from an example Evolution trajectory with AlexNet FC8 unit 0 as objective and CholeskyCMA as optimizer. (compare Fig. 15 in [2]).

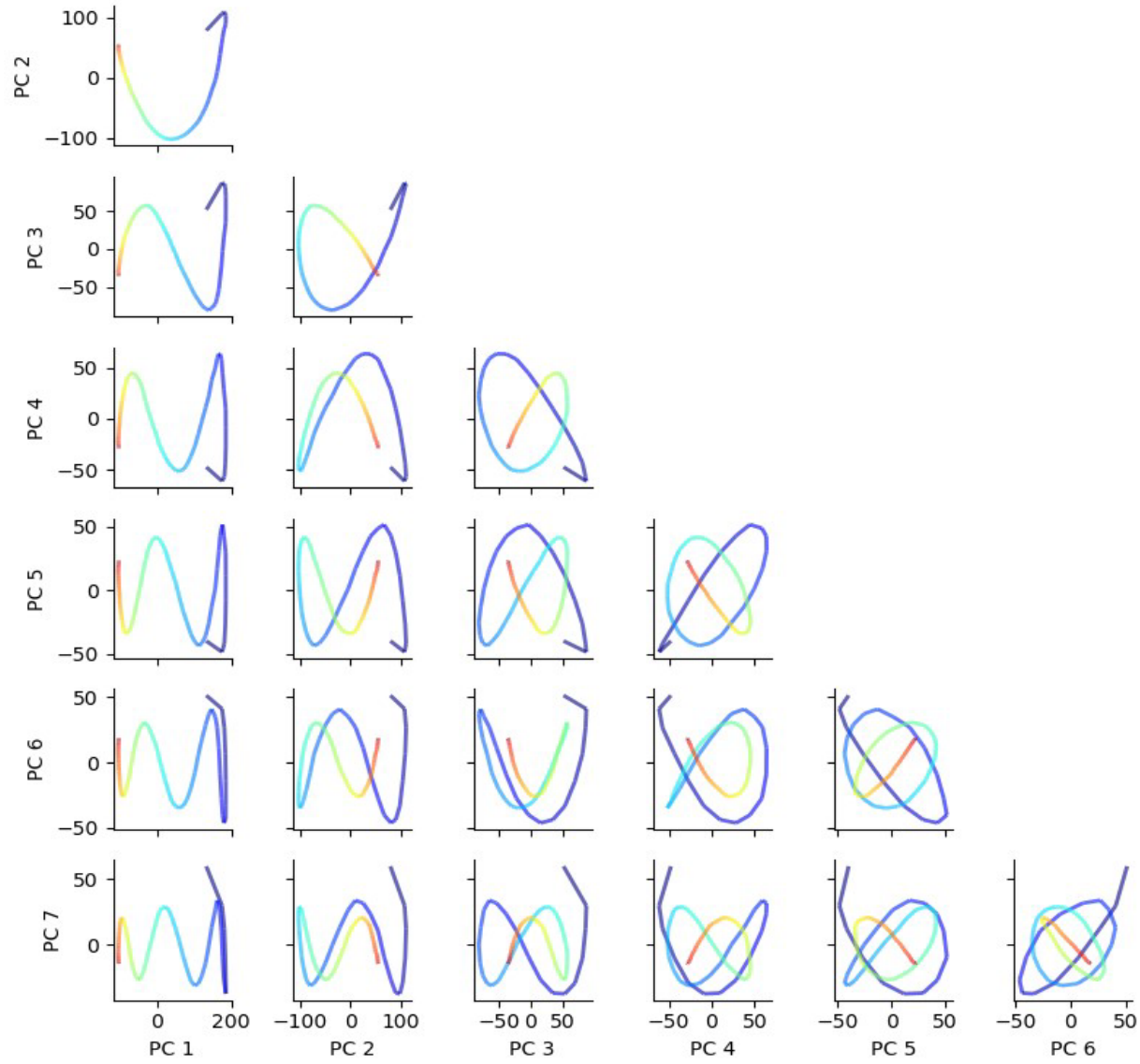


Figure III-22 Figure D.4: Lissajous Curves formed by PC Projection of Evolution Trajectory Driven by SphereCMA Blue to red color gradient represents the progression of time, from first to the last step. This is obtained from an example Evolution trajectory with AlexNet FC8 unit 0 as objective and SphereCMA as optimizer. (compare Figure III-21).

III.6.8 Definition of Subroutines Used in SphereCMA

In this section we define the subroutines we used in the SphereCMA algorithm: ExpMap, RankWeight, SphereExtrapolate.

Algorithm 2 ExpMap

Require: Base vector m, tangent vector v, angle μ to travel from m towards v.

Ensure: $\|m\| > 0, m^T v = 0$

$R \leftarrow \|m\|$

$\hat{m} \leftarrow m/R$

$\hat{v} \leftarrow v/\|v\|$

$u \leftarrow R \cdot (\hat{m}\cos\mu + \hat{v}\sin\mu)$

return u

Algorithm 3 SphereExtrapolation

This is the famous SLERP (Spherical linear interpolation) algorithm.

Require: Base vector m, end vector p, ratio t controlling interpolation or extrapolation.

Ensure: $\|m\| = \|p\|$

$\hat{m} \leftarrow m/\|m\|$

$\hat{p} \leftarrow p/\|p\|$

$\theta = \arccos(\hat{m}^T\hat{p})$

$q \leftarrow \big(\sin\big((1-t)\theta\big)m + \sin(t\theta)\,p\big)/\sin\theta$

return q

Algorithm 4 RankWeight

Require: Array of scores $r = [r_1 \ldots r_B]$, population size B

Require: Parameter: cutoff number $K = B/2$

Initialize raw weights $\widetilde{w}$ as an array of length B.

Top K candidates get positive weights. For $k = 1$ to K, $\widetilde{w}_k \leftarrow \log(K + 1/2) - \log(k)$

Other candidates get no weights. For $k = K + 1$ to B, $\widetilde{w}_k \leftarrow 0$

Normalize all weights to sum to one. $\widetilde{w}_i \leftarrow \widetilde{w}_i / \sum_i \widetilde{w}_i$

Find rank N_i of each score r_i (e.g. largest score has rank 1.)

Fetch the raw weight by the rank of each score. For $i = 1$ to B, $w_i \leftarrow \widetilde{w}[N_i]$

return w

Chapter IV : Geometry of Tuning Landscape of the Ventral Stream

IV.1 Introduction

A central goal in sensory neuroscience is to explain how neurons respond to natural images. Yet to approach that understanding, the field has had to rely on simplified stimuli. Simple visual stimuli are easier to transform smoothly, which means they can be used to generate *tuning curves*, descriptions of the relationship between a neuron's firing rate and a key "variable of interest" (Seung and Sompolinsky, 1993). The best-known example is the orientation tuning curve in primary visual cortex (V1) (**Figure IV-1 A**). Here the variable of interest is orientation, implemented using images of gratings at different slants. If a V1 neuron shows higher activity in response to a particular orientation, with smoothly decreasing activity to gratings with dissimilar orientations, then it is concluded this space is encoded by the cell (Campbell *et al.*, 1968). This type of tuning function has been a reliable tool in studies of early visual cortex (V1, V2, V4) (Hubel and Livingstone, 1987; Anzai, Peng and Van Essen, 2007; Yau *et al.*, 2012) and in dorsal cortical areas such as MT (Maunsell and Van Essen, 1983). Decades of work have resulted in the accumulation of an open set of variables describing neuronal function. However, there is no rationale about the uniqueness of any of these variables, nor an estimated number of all other potential variables that could be encoded by visual cortex. How many other explicit variables could there be? How many more to explain a neuron's response to natural images?

Here, we take a broader perspective on this problem, by considering the concept of a *tuning landscape*, defined as the neuronal response function over the entire image manifold. Since the natural image manifold is a bounded space (*topologically compact*), if neuronal responses

represent a continuous function, they have to exhibit a maximum in that space (per the extreme value theorem) (James Munkres, 2000); when input deviates from the maximum, the function will smoothly decay or stay constant. Given this maximum, we can revisit a framework where such a peak represents a special combination of visual attributes stored by the neuron — a *prototype* (Rosch, 1973; Edelman, 1999) (**Figure IV-1B**). In this framework, prototypes serve as landmarks in this representational space, and the neuron's tuning function signals the similarity between any given visual input and its prototype(s) (Poggio and Girosi, 1990a). If this mechanism is accurate, then it follows that classic tuning axes (*e.g.,* orientation) are accurate descriptions of neuronal function, but not unique. A more general distance-based function could explain why neurons respond to nearly all kinds of image types, from artificial computer-generated stimuli to randomly selected natural images (**Figure IV-1 C**). The theoretical foundations for this concept have been developed extensively, partially as solutions to view invariance (*e.g.*, radial basis function networks) (Poggio and Girosi, 1990a, 1990a; Maruyama, Girosi and Poggio, 1992), and as kernel machines (Anselmi *et al.*, 2015) for visual recognition. While this idea has been tested in some electrophysiological (Logothetis, Pauls and Poggio, 1995) and imaging studies (Kay and Yeatman, 2017), its full explanatory potential has not been outlined, partially because there has been (a) no clear way to identify the specific landmarks (prototypes) represented by neurons and (b) no easy way to smoothly manipulate complex naturalistic images. In this study, we solve both problems using image generative models. We used a deep *generator* (Dosovitskiy and Brox, 2016b) that maps 4096-dimensional latent vectors to naturalistic images. This model provided a parametric proxy for natural image space, allowing us to smoothly manipulate natural images and to use search algorithms (*optimizers*) to find tuning peaks in this space. We characterized the tuning landscape of neurons in V1/V2, V4 and

inferotemporal cortex (IT) (**Figure IV-1 D,E**), by mapping neuronal responses on a 2-D manifold sampled around the peak, then identifying neuronal tuning shape, extent, and smoothness. We also explored the global distribution of peaks by conducting image searches in reduced subspaces. We found three systematic changes in both *in vivo* and *in silico* visual hierarchies: the tuning width decreased along the hierarchy, the search convergence time increased, and there was an increasing gap in the responses evoked by optimal images in reduced subspaces. We explain these trends using a simple model of radial tuning varying in tuning width and intrinsic tuned dimensionality. Overall, our results suggest that ventral stream neurons can be viewed as operating in a multidimensional distance space, in analogy to hippocampal place cells, where responses signal the proximity between two points in physical space — the neuron's spatial field center at (x_0, y_0) and the input location at (x_1, y_1). In this analogy, the radial distance is the primary functional feature, while the direction of approach to the spatial field center is secondary.

IV.2 Results

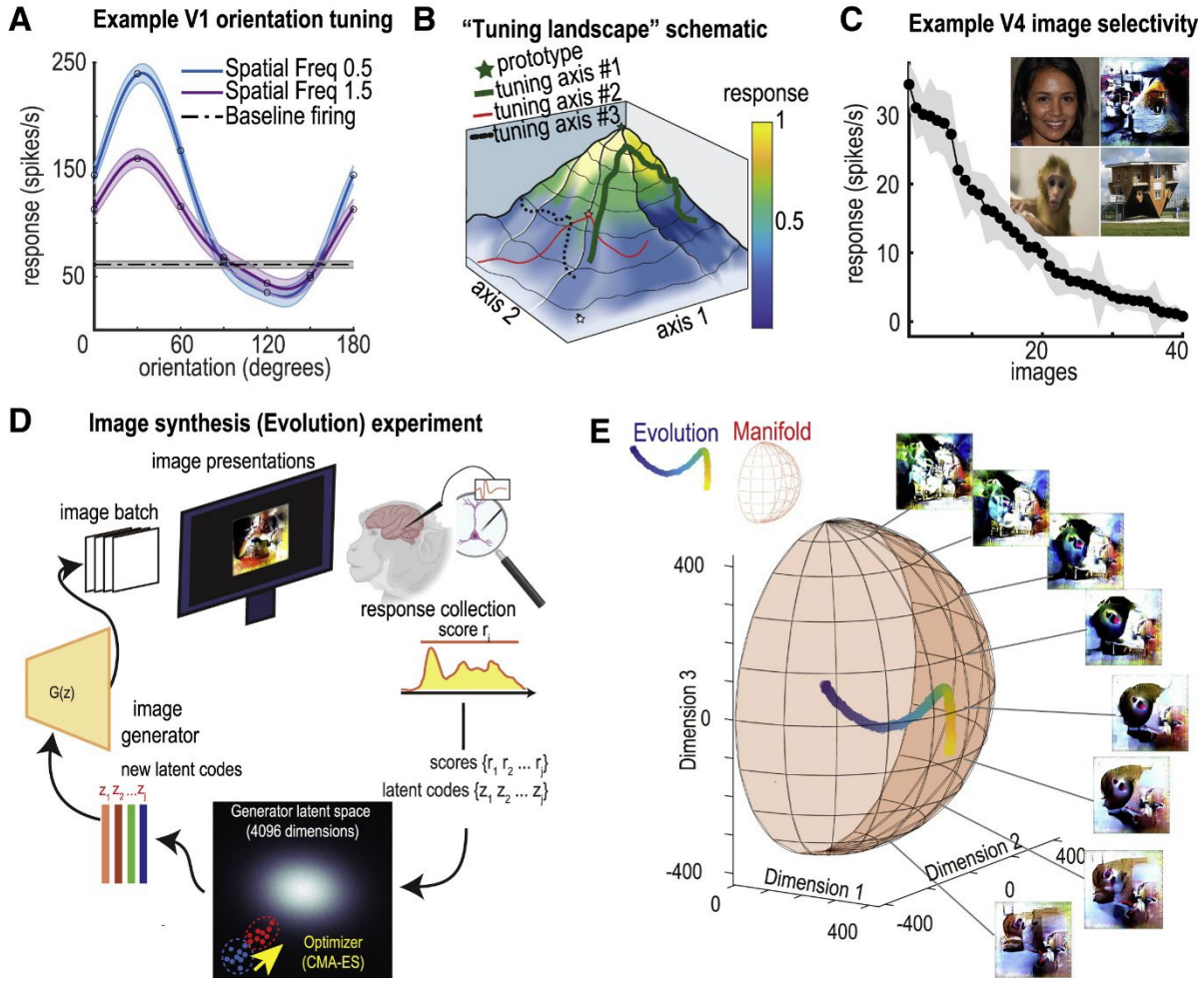


Figure IV-1 **Figure 1. Conceptual approach. A.** Orientation tuning curve (mean ± SEM) for a V1 multiunit site as evoked by Gabor patches. **B.** Schematic of a conceptual "tuning landscape," the neuron's activation function over the space of all visual inputs (2-D images); schematic shows two input axes, green star shows location of peak; lines over the landscape illustrate different experimentally defined tuning axes (e.g., *orientation*, *curvature*) as represented by images. **C.** Responses of a single neuron in anterior V4 to randomly selected stimuli; inset shows top four images. **D.** Workflow for the *Evolution* of a preferred image (*prototype*), combining the image generator (***G***, latent codes as inputs and images as outputs), neuronal responses to each generator image, and an optimizer algorithm (CMA-ES) that samples the generator's latent space to maximize neuronal responses. **E.** Main experimental strategy: a two-fold approach to identify a peak in the image generator space (*Evolution*) followed by an experiment sampling neuronal responses as sampling moves away from preferred location (*Manifold*) (**Figure IV-7**).

IV.2.1 Neurons show bell-shaped tuning around peaks in the generator space

To measure the shape and extent of neuronal tuning in the generator space, our experimental approach comprised two parts, an *Evolution*- and a *Manifold* experiment (**Figure IV-1 D, E**). The *Evolution* experiment identified images that maximized neuronal responses, and the *Manifold* experiment sampled the neighborhood around such images. This allowed us to measure a local tuning landscape and to characterize the width and smoothness of tuning around the observed peak.

Evolution experiments. This methodology was used previously (Rose, Johnson, Wang and Carlos R. Ponce, 2021). Briefly, after selection of a neuronal site (single- or multiunit) in V1/V2, V4 or IT (receptive fields in **Figure IV-7 A**), we presented shapeless, texture generator images and recorded the firing rates evoked by each image. These single-trial responses were used as scores for the input vectors behind each generated image; then the optimizer (CMA-ES algorithm) (Loshchilov, 2017) took the scores and vectors and proposed new input vectors likely to maximize firing rates in the next generation. These input vectors were fed back into generator to create stimuli for the next round, closing the loop. Each cycle lasted for 0.5–1 min; each experiment ran for ~30 min, until the neuronal site's mean response stabilized into a local maximum — having "evolved" a preferred stimulus (Figure IV-2 **B**). We refer to these GAN-derived preferred stimuli as *prototypes*, as inspired by prototype- and similarity theory, which suggests that complicated concepts may be summarized by templates (Edelman, 1999; Leopold *et al.*, 2001). Prototypes are pictorial illustrations of the specific combinations of colors, shapes and textures encoded at each given site; they are often complex (Tanaka, 2003), and while consistent in shape and color over time, upon repeated recordings, they can vary in size, position, and other nuisance variables. Each experiment also included the presentation of reference images

used to track neuronal isolation stability; these included photographs, Gabor patches or other stimuli, depending on the site's preferences, as established in separate experiments (Rose, Johnson, Wang and Carlos R. Ponce, 2021).

Manifold experiments. The location of the prototype in the generator space was an anchor point to sample images along a manifold. To define this sampling manifold, we performed a principal component analysis (PCA) on the trajectory of latent codes during evolution. The first principal component corresponded roughly to the direction taken during prototype synthesis (**Figure IV-8 A-C**); we then created a 2-D sphere centered at the origin in the subspace spanned by the first three components. The radius of this sphere was defined by the norm of the latent code in the last generation of the *Evolution* experiment (**Figure IV-2 A, Methods**). The choice of exploring on the sphere was motivated by the geometry of the generator latent space (Wang and Ponce, 2020; Wang and Carlos R. Ponce, 2022b) and by literature on sampling GANs (White, 2016; Kilcher, Lucchi and Hofmann, 2017; Wang and Ponce, 2021). Images sampled from a grid around the prototype (termed the *manifold image space*) were then displayed in a rapid-serial-visual presentation design, along with images from other stimulus sets used to define neuronal tuning (Enroth-Cugell *et al.*, 1983; Pasupathy and Connor, 1999; Lin *et al.*, 2014; Russakovsky *et al.*, 2015), such as Gabor patches and curved 2-D contours. We refer to the responses over the manifold image space as the *tuning map*, defined as a function over the 2-D hemisphere. In the next three sections, we report results pooling single- and multiunits (SUs and MUs, collectively named *sites*); differences between SUs and MUs are examined in **Figure IV-4**, **Figure** IV-13.

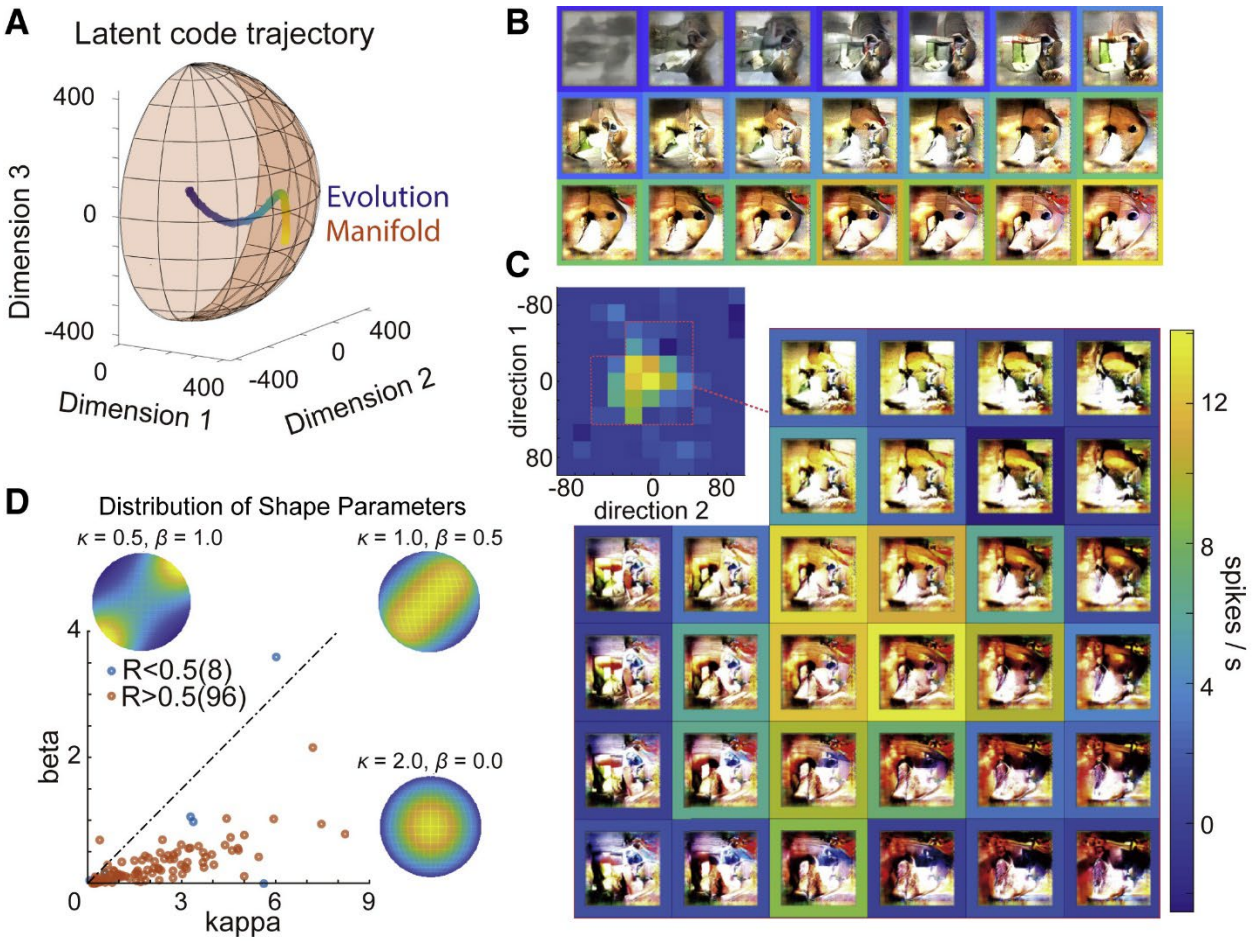


Figure IV-2 **Figure 2. Characterizing tuning landscapes. A.** Schematic illustrating the two-part design for defining the tuning landscape: *Evolution* and *Manifold* experiments. The blue-yellow curve illustrates the trajectory of latent vectors during an *Evolution* experiment, as projected onto its top-three principal-component space, with color representing generation number (blue, first; yellow, last). This trajectory was measured in an *Evolution* experiment driven by a single neuron in posterior IT. The orange hemisphere illustrates the sampling space for *Manifold* experiments, based on the hemisphere in the top-three principal-component space with the longitude and latitude grid (18° intervals). This hemisphere explored the manifold around the approximate endpoint of the *Evolution* trajectory. **B.** Change in generated images throughout the *Evolution* experiment for the same single unit. Images illustrate mean latent code (every fourth generation, from left to right, top to bottom). These latent codes correspond to the curved trajectory in **A.** Color around each image represents the mean activation for that generation. **C.** Example tuning map for the same single unit. Inset shows the image corresponding to each position of the map. **D.** Distribution of the shape parameters κ, β of the Kent-function fits; maps well-fit by Kent are colored red; the dashed line $\kappa = 2\beta$ is the boundary in parameter space between unimodal and bimodal Kent functions (**Figure IV-7,8,9,10, Figure** IV-13,14**)**.

Most *Evolution* experiments (79.1% of 91) resulted in a significant increase in the firing rate of the site under study (per two-sample t-tests of firing rate in initial two generations versus last two generations, with criterion $P < 0.001$). Having identified tuning peaks in the generator space, we then examined the shape of tuning function in the *Manifold* experiments. We found that neuronal sites decreased their responses smoothly when moving away from the prototype in any direction (**Figure IV-2 C**, Figure IV-7 **B).** This kind of tuning was reminiscent of tuning curves of V1 simple cells responding in the space of oriented gratings. Sites were significantly modulated by the manifold image set ($104/110$ of experiments, $P < 0.001$, one-way ANOVA, 101 images per test, F-stat range $[1.56, 68.64]$). For more relationships between the success of the *Evolution* experiments and other properties of tuning maps, see **Figure IV-8**.

Above, we noted that a common aspect of tuning curves is their *smoothness*. Theoretical work suggests that tuning smoothness can serve as a key inductive bias that enables downstream readout neurons to learn from few examples (Bordelon, Paulson and Pehlevan, 2021). We hypothesized that smoothness is important because the neuronal code is noisy, and so a neuronal population comprising smooth tuning maps would allow information decoding with smaller errors. To characterize the smoothness of these manifold tuning maps without committing to a specific function type, we measured the Dirichlet energy (DE) of the observed maps and compared this value to the energy of maps where the image-response relationships were shuffled (see **Methods** and Figure IV-7 **E**). This energy metric integrates the squared norm of the gradient vectors over the hemisphere, thus quantifying the relative smoothness of the tuning function: a constant function will have zero DE, while a *rugged* tuning function will result in a high DE. We found that the observed tuning maps were remarkably smooth, showing a significantly smaller DE than the upper ceiling defined by shuffled-data (observed vs. image-identity shuffled maps,

two-sample t-test, $P < 0.001$ in $96/110$ experiments, T-stat ranged from $[-186.6, -8.3]$, Cohen's d' ranged from $[-8.32, -0.37]$ for the smooth channels).

Next, we set out to quantify the tuning shape around the prototype. By visual inspection, we noticed that most tuning functions appeared bell-shaped. To quantify this, we used the Kent function to fit the 2-D tuning maps. The Kent function is defined on a sphere and is analogous to a 2-D Gaussian in Euclidean space. This function comprises parameters for a) activity baseline, b) tuning amplitude, c) two parameters for peak location, and d) three shape parameters. We found Kent functions generally fit the tuning maps well, with a median R^2 of 0.72 ($N = 110$) (Figure IV-2 **D**; for examples of raw and fitted tuning maps, see **Figure** IV-9). For reference, the R^2 noise ceiling obtained by bootstrapping single-trial responses had a median 0.876 ($N = 110$) (**Methods**). To contextualize this result, we also repeated the *Evolution-Manifold* experiments using convolutional neural networks, which are noise-free, and measured how well the Kent function could fit these maps. The tuning map fits from VGG-16 network had an R^2 value of 0.918 ± 0.050 (mean ± SD, median 0.922, $N = 427$ units sampled across nine layers). The tuning maps of V1 driving sites were less well-fit by the Kent function than V4 and IT driving sites: R^2 for V1 were 0.665 ± 0.027 ($N = 31$), for V4 0.787 ± 0.029 ($N = 21$) and IT 0.787 ± 0.020 ($N = 50$).

Having established that the Kent function could serve as a good model, we then used its shape parameters β, κ to quantify tuning, as the ratio of these parameters indicates the isotropy vs. anisotropy of the peak on the manifold sphere. The parameter $\beta = 0$ indicates a perfectly isotropic peak, $\kappa/\beta > 2$ indicates a unimodal function, and $\kappa/\beta < 2$ indicates a bi-modal function (see **Figure IV-2 D** and **Figure IV-7 D**). In experiments with good Kent fits ($R^2 > 0.5$,

$n = 96$), the 95% confidence interval of κ/β was [7.02,9.51], significantly larger than 2 (one-sample t-test, $t = 18.37, P = 5.0 \times 10^{-33}$). This showed that most tuning functions were indeed unimodal, bell-shaped, and relatively isotropic within the measured domain.

In summary, these analyses showed that when neurons were presented with smooth and complex deformations from their preferred stimulus, they also showed a smooth reduction in firing rate; this reduction was similar no matter in what direction the transformation occurred. This is the signature of a radial basis function, where the response change depends largely on the distance from the preferred stimulus, and less so in the direction taken by the deformation. This type of response change is likely to be characteristic of neurons in natural conditions, where changes in the retinal image can be smooth (as in viewpoint rotations) while varying across multiple visual features.

IV.2.2 Relating neuronal tuning in generator space to other image spaces

Neurons showed radial tuning around their preferred stimulus, and we interpreted this tuning as a likely default behavior of neurons when presented with generic but smooth image changes. However, our overarching motivation was to define how neurons respond across even larger image domains, involving images with different low-level statistics, such as photographs and computer-generated artificial stimuli. While the generator produces an astronomical number of variations, it has limits. It produces images with a certain textural style and is less photorealistic than newer generators (e.g. BigGAN (Brock, Donahue and Simonyan, 2018)). Here, we advanced our goal of defining tuning function shapes by including images created both within and outside the generator space. Specifically, we collected responses of given sites to *Manifold* images, to stimuli from classic image spaces such as Gabor patches (parametrized by orientation and spatial frequency), to bounded 2-D contours (parameterized by orientation and

curvature) (Pasupathy and Connor, 1999), and to ImageNet photographs. Together with our generator images, this image set spanned a range from simple/artificial to complex/natural images. While we reasoned it would be easy to measure neuronal responses to these images, the key challenge was to find a common axis that would fit images with such diverse statistics. To solve this problem, we developed a *radial tuning curve analysis* (Figure IV-3**A, left**), aimed to construct *one-dimensional* tuning curves using image-response pairs with stimuli created from different spaces, and to define the shape and extent of this tuning function. We first considered each space separately and fit the neuronal responses as a function of *image distance* to the most activating image in that space (responses were smoothed using Gaussian process regression). Across these spaces, two features of each tuning curve were comparable: (1) peak amplitude and (2) radial tuning width (Figure IV-3**A, right**). The peak amplitude was defined as the maximum response among images in each space, signaling the effectiveness of the image space. For the radial tuning width, we smoothed and integrated the area under this radial tuning curve. This estimated the tuning width of the peak, *i.e.*, how far away images could deviate from an optimum while still allowing the neuron to stay responsive. Our image distance was a *perceptual similarity* measure (LPIPS) based on convolutional neural networks (CNNs), networks trained to match human perceptual judgments (Zhang *et al.*, 2018). We performed this analysis for each driving site and compared the peak amplitude values and radial tuning widths across each space.

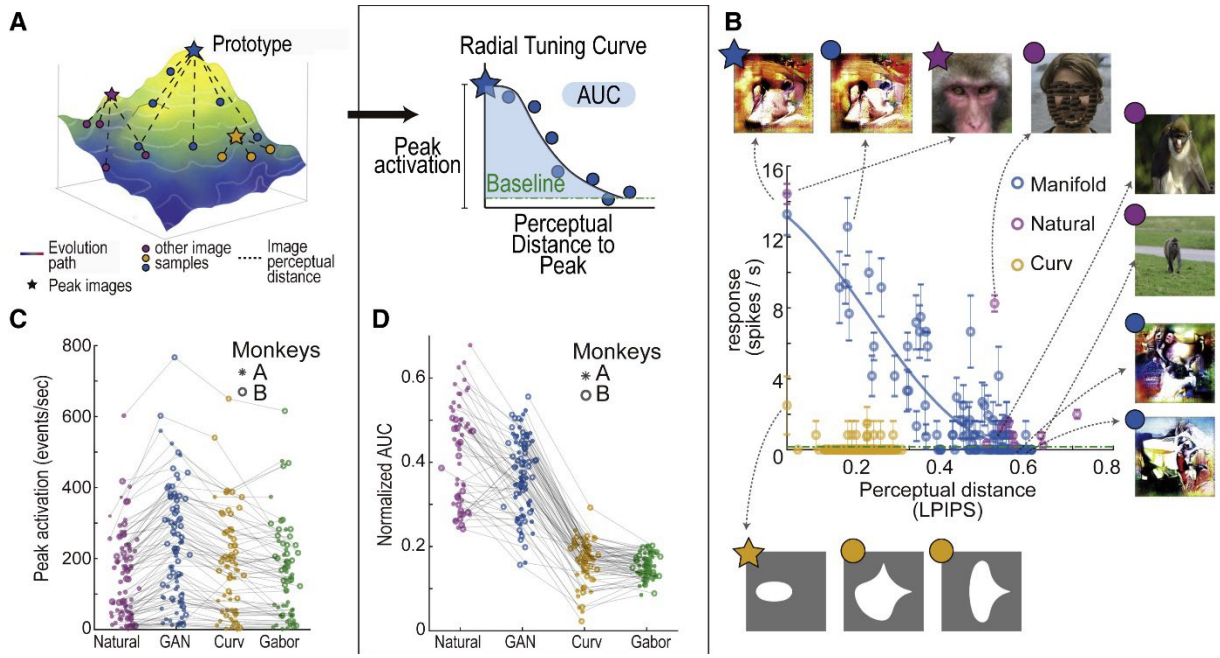


Figure IV-3 **Figure 3. Characterizing tuning landscapes with radial tuning curves. A.** Schematic of *radial tuning curve* analysis: in each image space, we identified the most activating image and measured the perceptual distance of all other images to it. After plotting the activation as a function of image distance, we computed two statistics for each space: peak activation (**C**) and normalized area-under-the-curve (normAUC, **D**). **B.** Example of a radial tuning curve (same IT neuron as in **Figure IV-2**), quantifying responses to images across different spaces (generator images, blue; photographs, purple; curvature images, yellow). Solid blue line shows a fit through the *Manifold*-image responses, which provided the best continuously sampled space. **C.** Comparison of peak activations across image spaces, for the same driving sites recorded in both animals. Each site is denoted by a point; gray lines connect the values across image spaces for the same site. **D.** Tuning width estimated by the normalized AUC of the radial tuning curve, also compared across image spaces (**Figure IV-11)**.

We found the neuron-optimized manifold space had the highest peak amplitude (Figure IV-3**B,C)**: the peak firing rate was 228.5 $\pm$ 16.8 spikes/s (mean z-score activation 2.532 $\pm$ 0.094, combining single- and multiunits) compared to 144.6±12.7 spikes/s (z-score 0.630 $\pm$ 0.074) for the best natural images and 181.2±16.2 spikes/s (z-score 1.074 $\pm$ 0.112) and 160.9±15.7 spikes/s (z-score 0.592 $\pm$ 0.086) for the best curved-object images and Gabor patches (Figure IV-3 **C**, paired t-test, z-scored amplitude, vs natural reference $t_{90} = 17.4, P =$

1.5×10^{-30}; vs curvature images $t_{71} = 10.2, P = 1.5 \times 10^{-15}$; vs Gabor patches $t_{69} = 16.3, P = 2.8 \times 10^{-25}$, z-score specifics, **Methods**). This confirms that optimized prototypes were more activating for neurons than manually selected ones (Ponce et al., 2019), even in the context of great V1/V4 artificial stimuli such as curvature patches (**Figure IV-11** for comparisons focusing on V1 and V4 — for the V1 population, the peaks in the manifold space were higher than those for Gabor patches; for V4 population, the peaks in manifold space were more similar to those in curvature space).

Next, we computed the area under the interpolated radial tuning curves (AUC), above the baseline firing rate. We normalized the AUC by dividing the peak activation of each tuning curve, to compare the tuning width while controlling for peak height (Figure IV-3 **B; Methods**). We found that the median normalized AUC for the manifold space was 0.370 ± 0.009 (Figure IV-3 **D**), compared to 0.164 ± 0.005 for the curvature space (paired t-test, $P = 3.9 \times 10^{-42}$, Student $t_{80} = 27.16$), 0.153 ± 0.003 for Gabor shapes ($P = 1.4 \times 10^{-40}, t_{78} = 26.34$), 0.424 ± 0.016 for natural images ($P = 2.0 \times 10^{-8}, t_{57} = -6.52$); when including sessions where V1 sites drove the *Evolution* and Gabors were used as evolution references, the average was 0.334 ± 0.016 ($P = 8.4 \times 10^{-2}, t_{90} = 1.75$, $N.S.$). We should emphasize that even when we focused on neurons sampled from V1 and V4 and their classic stimuli spaces, Gabor patches and curvature images, our results held: neurons had a larger absolute and normalized area under the curve (AUC) in the manifold space than in classic stimuli spaces (normalized AUC, for the V4 population, Manifold vs curvature, $t_{19} = 10.22, P = 3.7 \times 10^{-9}$; for the V1 population, Manifold vs Gabor, $t_{34} = 8.24, P = 1.3 \times 10^{-9}$, Figure IV-11). This shows that tuning curves measured in simpler image spaces can underestimate both the dynamic response range of the

neuron and the extent (width) of its tuning. This can be viewed as evidence that classic tuning curves are local sections of a higher dimensional tuning landscape.

What does the radial tuning width mean? To contextualize these results, we characterized the geometry of these image spaces using LPIPS. First, we measured the diversity of each space as the average distance between two random samples. We calculated the distance between 1000 random samples from the 50,000 ImageNet validation set and found that the mean pairwise distance was 0.56 ± 0.07 (mean ± SD, [5, 95] %ile $[0.46, 0.67]$). In comparison, when we randomly sampled 500 *Manifold* images, they spanned a mean difference of 0.47 ± 0.04 ([5, 95]%ile $[0.41, 0.53]$) — curvature- and Gabor-patch spaces spanned a mean distance of 0.17 ± 0.05 ([5, 95]%ile $[0.08, 0.24]$) and 0.16 ± 0.04 ([5, 95]%ile $[0.08, 0.22]$). Thus, generator images covered a range of shape diversity closer to that in the real-world images. Next, we examined the sample density in these image space using nearest-neighbor distances. For natural images, we computed the distance to a nearest neighbor (within the 50K image set) for 1000 images from the ImageNet validation set: the mean distance was 0.364 ± 0.051 (mean±SD), ([5,95]%ile $[0.271,0.440]$). In contrast, in the *Manifold* experiments, the mean distance between nearest samples was 0.087 ± 0.035 ([5,95]%ile $[0.029, 0.136]$); for curvature and Gabor-patch spaces this distance was 0.031 ± 0.011 ([5, 95]%ile $[0.022, 0.045]$) and 0.081 ± 0.006 ([5, 95]%ile $[0.068, 0.087]$). For the radial tuning analysis, this difference in sampling density made the tuning width estimate in natural image space less accurate than the more densely sampled spaces such as the generator-, curvature- and Gabor spaces.

We have shown that the tuning width estimates in the generator space were smaller compared to those measured in the space of real-world images. Why is this? First, our image generator is

an imperfect approximation of the natural image space, namely, it is not diverse and as "realistic" as the natural world itself; a more photorealistic GAN might reveal tuning widths closer to natural images. However, tuning width values estimated from natural images were less reliable, due to the larger sampling gap in photographs. This sampling gap was difficult to overcome by increasing the sample size of natural images — even in a 50,000-image set, only 121 out of 1000 images had any neighbors within 0.3 perceptual distance, already a coarse step. Finally, the gaps between natural image samples were sometimes larger than the tuning width in generator space (**Figure IV-3B**). It is possible that natural image samples marked different tuning peaks on the tuning landscape, further inflating the estimation of radial tuning width. This large sampling gap is likely one reason smooth tuning curves are seldom reported for IT cortex neurons.

In summary, we defined the shape and extent of neuronal tuning across image spaces, including that of the generator, photographs, and classic parametric stimuli such as Gabor functions. We used a perceptual similarity metric to compare the neuronal responses to this diverse image set. We found that neurons showed a smooth decay consistent with that found in the generator space. The tuning functions measured around the neuronal prototype were larger in amplitude and wider in extent than those measured over classic non-optimized stimuli. This confirmed that manually selected image sets may not always overlap with the preferred domain of the neuron and could underestimate its full dynamic range and input domain. We interpreted this tuning width as a feature that makes these neurons well-suited for processing naturalistic images, *i.e.,* allowing them to remain informative over large swaths in image space. Further, we hypothesize this expansive tuning function is the reason we could use search algorithms (e.g., CMA-ES) to find peaks on the tuning landscapes in the first place (see Sec. IV.2.4 *Inferring the*

geometry of the tuning landscape by modeling): as we quantified previously (Wang and Carlos R. Ponce, 2022c), the mean distance among each generation of images usually started around 0.4 and gradually decayed to ~0.2. This step size of image change seems to be well suited to "climb" the slope of neuronal tuning peaks.

IV.2.3 Areal differences of tuning landscapes

We have shown that neurons in the ventral stream have smooth radial tuning functions spanning a large space of image transformations. Next, we investigated how these functions differed across the visual hierarchy, focusing on neuronal sites in V1/V2, V4, and IT. Neurons in these cortical areas are usually studied using different stimuli, limiting comparisons. Here the generator space served as a common space for all areas. We examined the local and global properties of the tuning landscapes using the *Evolution-Manifold* design — finding a site's local maximum and then exploring the landscape by inducing deformations to the favored image(s). Further, we indirectly compared the global characteristics of the tuning landscape by analyzing the *Evolution* trajectories and by using a new set of **reduced-dimension *Evolution*** experiments.

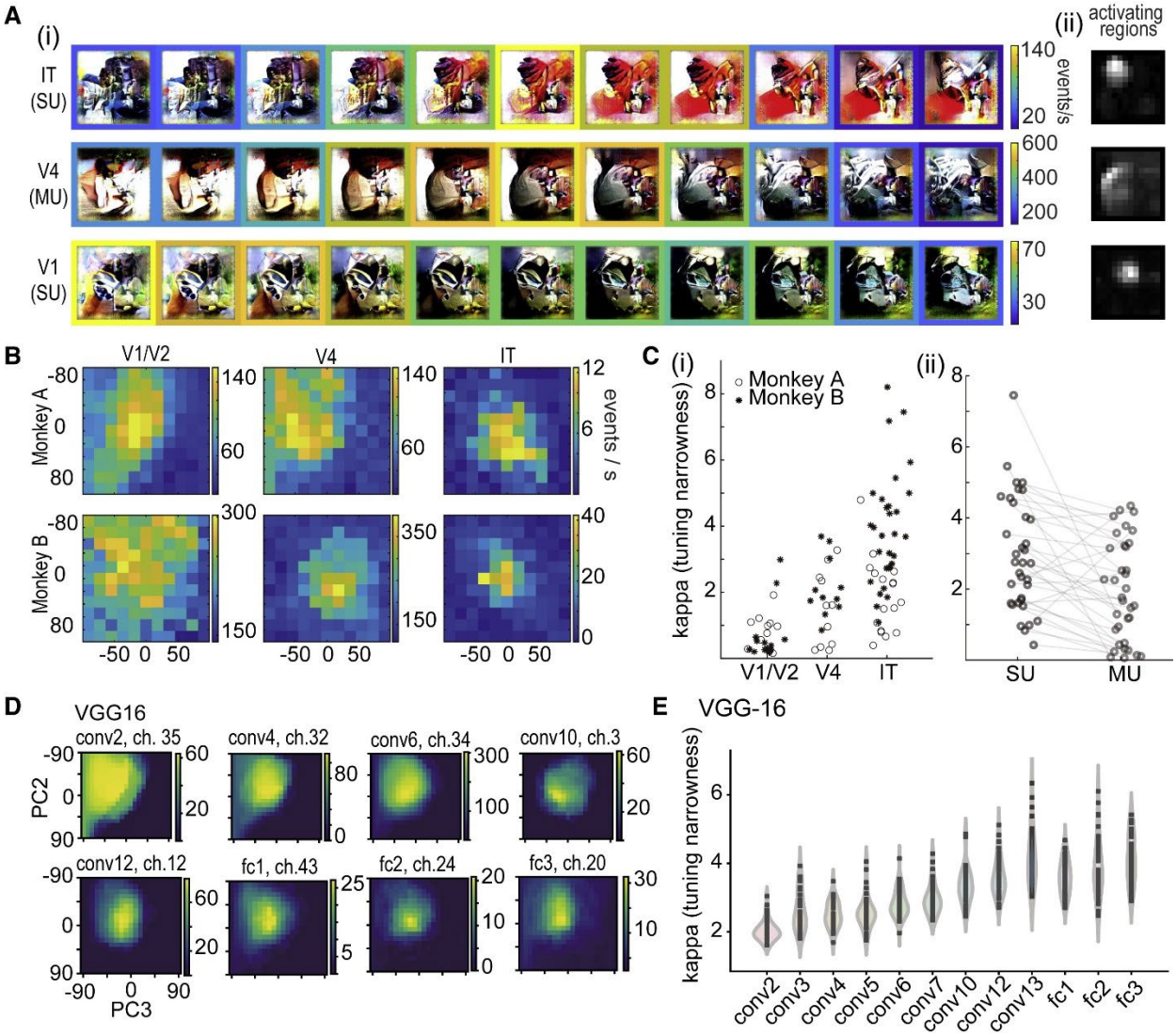


Figure IV-4 **Figure 4. Comparison of tuning landscapes along the hierarchy. A. (i)** Examples of tuning axes sampled from manifolds in IT, V4 and V1 (rows); images show deformations from each site's prototype (frames = neuronal response). **(ii)** Activating regions within the synthetic image, identified via correlation-based feature localization (see **Methods**). **B.** Example 2-D tuning maps of sites at the V1/V2 border, in V4 and in PIT (*columns*), for both animals (*rows*). **C. (i)** Population tuning width values as quantified by the κ coefficient (higher values → narrower tuning) across visual areas and animals. **(ii)** Tuning sharpness values κ for single- and multi-units. **D.** Example tuning maps of units in VGG-16. Maps ordered by layer depth. **E.** Tuning sharpness value κ as a function of layer in VGG-16. Each point represents one hidden-unit tuning map (**Figure** IV-9**,** Figure IV-10**,** Figure IV-12**)**.

Local properties. Because neurons become increasingly sensitive to specific visual patterns over the cortical hierarchy (Kobatake and Tanaka, 1994; Rust and Dicarlo, 2012), one prediction

is that tuning width in the generator space might vary across areas (Figure IV-4 **A**). To quantify tuning width across areas, we used the κ parameter of the Kent function, which characterizes tuning sharpness: the higher the κ value, the narrower the tuning width. In tuning maps with reasonable fits ($R^2 > 0.5, N = 96$), we found that sites from higher visual areas showed a larger κ value than those from lower areas — V1 sites showed a mean (±SD) κ of 0.72 ± 0.13; V4 sites, 1.82 ± 0.23, and IT sites, 3.19 ± 0.26. (Figure IV-4 **B, C(i),** one-way ANOVA, $F_{2,93} = 26.7, P = 7 \times 10^{-10}$; Spearman correlation between κ and area level was $0.661 (df = 96, P = 2.4 \times 10^{-13})$). We confirmed this progression using two non-parametric statistics: we compared (a) the normalized area under the curve (AUC) for the radial tuning curves in each area (smaller values indicate narrower tuning) and (b) the normalized volume under the surface (VUS) for each tuning map (smaller means narrower, **Methods**). Both of these values showed a similar trend: more anterior visual neurons showed sharper tuning peak values than posterior neurons (Spearman correlation of AUC values with ordinate hierarchical position [V1/V2, V4 and IT], $\rho = -0.56$ $(df = 103, P = 5 \times 10^{-10})$; for VUS $\rho = -0.57$ $(df = 103, P = 3 \times 10^{-10})$. We also compared the tuning width values for single- vs. multi-unit signals in the same channels. Single units had narrower tuning, consistent with the view that multi-unit represents the local aggregated signal from single units (**Figure IV-4C(ii),** for more examples see **Figure IV-13**). As noted above, the tuning maps of V1 sites were less well-fit by the Kent function than those of V4 and IT sites: mean R^2 for V1 was 0.67 ± 0.03 $(N = 31)$, for V4, 0.79 ± 0.03 $(N = 21)$ and for IT, 0.79 ± 0.02 $(N = 50)$. This trend was in line with the noise-ceiling R^2 of the three microelectrode array populations, measured by randomly splitting trials into two sets and computing the R^2: for V1, the noise ceiling for R^2 was 0.74 ± 0.03 $(N = 31)$, for V4, 0.89 ± 0.02 $(N = 23)$, and for IT, 0.88 ± 0.02 $(N = 50)$. Overall, this suggests that the tuning maps in

V1 were more affected by fluctuation, less bell-shaped, and thus harder to fit by the Kent function.

As a control, we asked if there were any significant differences in the perceptual similarity of images in the V1, V4, and IT manifolds. We were surprised to find a small but significant difference in image diversity (quantified by the mean LPIPS image distance between two images on the hemisphere) between manifolds created by IT and V1 (IT, $0.421 \pm 0.003, n = 33$; V1, $0.410 \pm 0.003, n = 36$, two-sample t-test $t_{67} = 3.083, P = 0.003$). This overall modulation of image diversity per visual area was only marginally significant (one-way ANOVA across the three areas $F = 4.669, p = 0.012$). This effect size was small: for example, neuronal tuning in perceptual metric space ranged from 0–0.60 (**Figure IV-3**), and the mean difference in perceptual similarity between V1 and IT was ~0.01. We hypothesize that this difference in image diversity was a byproduct of the *Evolution* and *Manifold* design: searching for features of different complexity (i.e., for V1 versus for IT) may lead to trajectories in different subspaces; since images were manipulated along the 2nd and 3rd principal-component subspaces of the search trajectory, the manifold exploration in different subspaces could result in small but consistent differences in terms of image diversity, reproducible per *in-silico Evolution* and *Manifold* experiments. As another control, we also conducted *Evolution-Manifold* experiments in V1 sites using textures that were 1°-wide (vs. 3°-wide as in the experiments above) and found that the results above continued to hold (**Figure IV-10**). We also investigated the shape of tuning landscapes far away from the peak, by examining the concurrent responses of sites that were not driving the evolution: those tuning maps were more ramp- and slope-shaped (**Figure IV-14**).

We also asked if this pattern of increasing tuning sharpness was specific to visual cortex or if it could emerge in artificial visual hierarchies. We performed the same *Evolution-Manifold* experiments *in silico,* driven by the activation of randomly selected units from pre-trained CNNs. We found the same pattern of results: tuning peaks over the manifold became sharper as a function of layer depth. For example, in the VGG16 network (Simonyan & Zisserman, 2014), κ values correlated positively with layer depth (linear regression slope, 0.189 ± 0.008, $P = 1.9 \times 10^{-91}$, $N = 568$ units in 12 layers, **Fig. 4E**). We found the same result using AlexNet, ResNet-101, ResNet50, ResNet50-Robust, DenseNet and CorNet-S (*P*-values ranging from 3×10^{-22} to 2×10^{-124}, N ranging from 600–1200, **Methods**, Figure IV-12 **A-F**).

So far, we have explored a local property of the tuning landscape, specifically that visual hierarchies comprise units with progressively sharper tuning peaks in this naturalistic space. One straightforward interpretation of this result relates to sparse coding (Rolls and Tovee, 1995), as higher-order cortical units respond to more specific combinations of visual attributes, such as motifs present in faces (Desimone *et al.*, 1984; Tsao *et al.*, 2006). If these attribute combinations were less common within the generator space, compared to simpler features such as colorful curved edges, then it would be easier to reduce the responses of higher-order neurons by "moving away" from the peak by the same distance.

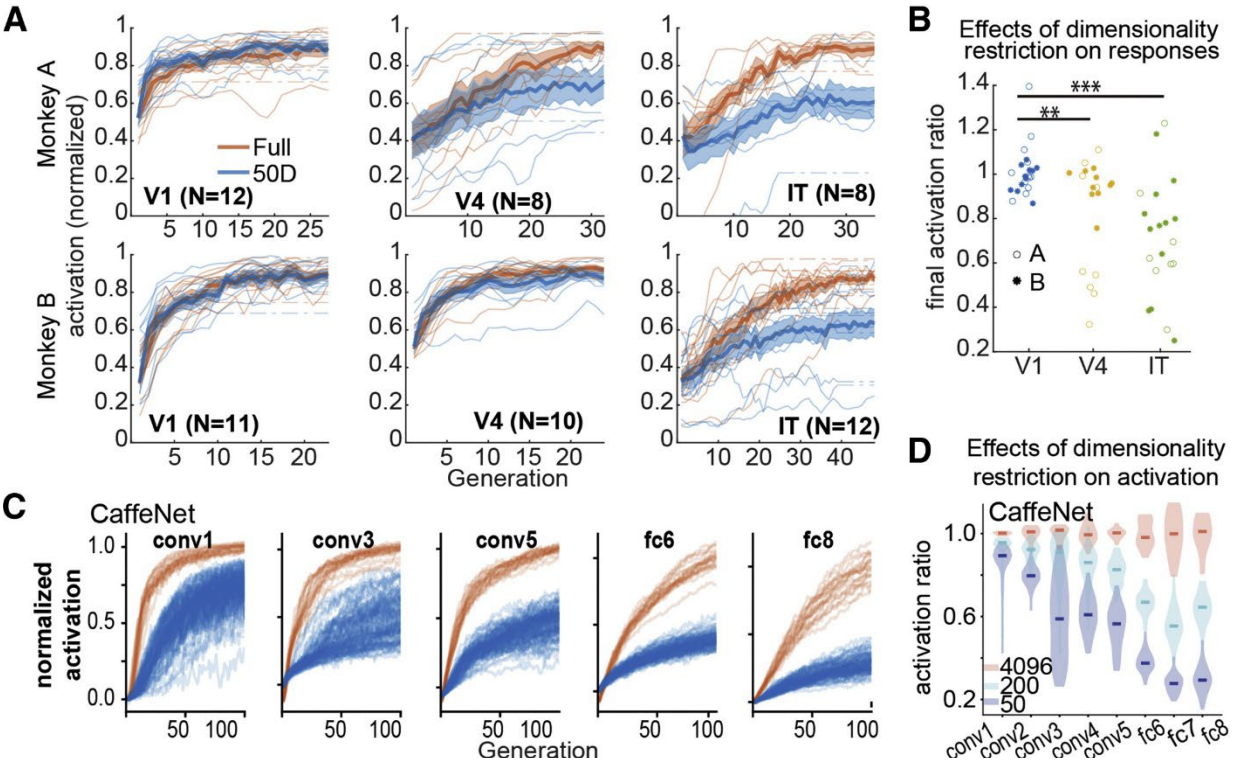


Figure IV-5 **Figure 5. Comparison of required search dimensionality along the hierarchy.** **A.** Activity as a function of generation for neurons in V1/V2, V4 and IT using the full input space (4096 dimensions, blue) or a reduced space (50D, orange), for each animal (top, bottom). Activity normalized by maximal response within each session, with mean ± SEM computed across sessions, and smoothed via moving average (N = 3 generations). Mean activation values at later generations show broader error because of fewer experiments with data in those generations. **B.** Effects of dimensionality restriction on activation maximization for sites in V1, V4 or IT. Y-axis shows the ratio of the final-generation activation values measured after evolving in a 50D space and in a 4096D space $\frac{\bar{r}_{50d}-r_{bsl}}{\bar{r}_{full}-r_{bsl}}$ (see **Methods**). Each dot shows one neuronal site, for both animals. Asterisks show statistical significance. **C.** Activity as a function of generation for hidden units in CaffeNet layers using the full space (orange) or reduced 50D space (blue). **D.** Effects of dimensionality restriction on activation maximization for hidden units in different layers. Y-axis shows the ratio of the final-generation activation values measured after evolving in a 50D space (deep blue), a 200D space (light blue) and a 4096D space (red) (Figure IV-12**)**.

Next, we examined a global property of the tuning landscape: the distribution of prototypes. *Evolution* experiments showed the existence of highly activating images in the search space and a path following the gradient of the tuning function, from the initial images (textures) to the final images (prototypes). By analyzing these experiments, we could probe the global geometry of the tuning landscape indirectly. The speed for finding a prototype related to the frequency of these

highly activating images in the latent space. For example, specific combinations of visual features (*e.g.,* shapes, colors, and textures) should be statistically rarer compared to simpler features (such as oriented edges). If higher order neurons select for more complex features, it should take longer to find them. This prediction aligns with previous studies of sparse object coding in the ventral stream, for example, those finding that V4 neurons are tuned mostly for acute curvatures (Carlson *et al.*, 2011), in contrast to V1 neurons, which as a population tile a relatively larger portion of orientation space (although V1 does overrepresent some orientations (Dragoi, Sharma and Sur, 2000)). To test this idea in the landscape context, we examined the convergence time of *Evolution* experiments. We found that the mean firing rate for sites in posterior regions converged to a steady state over fewer generations than the firing rate for sites in more anterior regions: the convergence timescale for V1/V2 neurons was 12 ± 1.2 generations ($N = 34$ experiments), compared to V4, at 21.4 ± 3.3 ($N = 20$), and IT at 28.7 ± 2.5 ($N = 30$) (Figure IV-12 **G-J**). The convergence timescale was quantified by the number of generations needed to reach 0.632 ($1 - 1/e$) of the maximum mean firing rate per generation. This trend confirms a previous report (Rose, Johnson, Wang and Carlos R. Ponce, 2021), also observed *in silico* in AlexNet. This would be expected if higher visual neurons prefer features which were sparsely distributed in the space, and harder to find.

If a neuron's preferred visual features were common in the generative space, the algorithm might still find a path to a peak even if the search was limited to a fraction of the full latent space. To test this, we performed a set of **reduced-dimension *Evolution*** experiments, comprising two concurrent *Evolutions* for each target site: in one track, the search occurred in the full 4096-dimensional latent space; in the other one, the search algorithm was constrained to a randomly selected 50-dimensional subspace. Trials from both tracks were interleaved. Then, we

measured the ratio R between the activation increase in the 50D evolution and that in the 4096D evolution. We found that the mean ratio R for V1/V2 sites was 1.009 ± 0.023 ($n = 23$), for V4 sites, 0.839 ± 0.056 ($n = 19$), and for IT sites 0.708 ± 0.060 ($n = 20$) (**Figure IV-5 A,B**); Spearman correlation between hierarchy level and the activation fraction was -0.537 ($P = 6.7 \times 10^{-6}$). We also calculated $\overline{d'}$, the average activation gap (per d-prime d') across all generations and found a growing gap along the hierarchy (Spearman correlation between hierarchy level and mean $\overline{d'}$ was $0.547, p = 1.4 \times 10^{-6}$). Thus, early ventral-stream neurons could still guide the evolution of their preferred image in constrained 50D spaces, reaching similar (or sometimes higher) responses compared to the 4096D space. In contrast, late ventral-stream neurons could not guide the evolution of their preferred image as successfully. This result was also replicated on the artificial visual hierarchy CaffeNet, where we found that limiting search space dimension had a larger impact on units in deeper layers than in shallower layers (Spearman correlation between ordinal layer number and ratio R is -0.893, $p = 2.4 \times 10^{-279}$; $n = 800$, 100 experiments per layer for eight layers, **Figure IV-5 C,D**, see **Methods**).

Together, these results suggest that the features preferred by early cortical neurons were more prevalent in the generator image space than the features preferred by late cortical neurons, likely due to the latter's selectivity for complex features (Kobatake and Tanaka, 1994); this may also explain why early ventral stream neurons showed broader tuning widths than those of higher-order neurons. We conclude that this trend in tuning specificity is likely to be a general feature of hierarchical networks that learn from natural data.

IV.2.4 Inferring the geometry of the tuning landscape by modeling

Above, we reported three systematic changes along the hierarchy of the ventral stream and CNNs: deeper in the hierarchy, (1) the time to convergence in *Evolution* increased, (2) the tuning width of neurons decreased, and (3) reducing the search dimensionality had a stronger negative effect on activation. Is there a mechanism that accounts for all these effects? We postulated that these trends were manifestations of a systematic difference of tuning landscapes along the ventral stream. To provide more intuition into this, we designed a simple synthetic tuning function to simulate these changes. We modeled the neuron's tuning function in the 4096-dimensional latent space of the generator, with a simple multivariate Gaussian. We then conducted the *Evolution, Manifold* and *reduced-dimension Evolution* experiments on this synthetic tuning function while varying two major parameters – the number of tuning dimensions D and the tuning width σ. We quantified these experiments using the above statistics (normalized Volume under Surface, convergence timescale, and activation ratio between 50D and full space evolution) and tested which variations in these two parameters could reproduce the three systematic changes in the ventral pathway.

More formally, the neuronal Gaussian tuning function was parametrized by a center z_0, the bandwidth σ^2, and the Hessian matrix H. Viewed from the peak of that Gaussian z_0, this is a radial basis function, with neural activation decreasing along any direction leading away from the peak with a speed depending on σ and H. The bandwidth σ^2 controlled the general tuning width, while the Hessian H controlled the curvature (or tuning width) along different directions.

$$r(z) = \exp\left[-\frac{1}{2\sigma^2}(z - z_0)^T H (z - z_0)\right]$$

$$\frac{\partial^2 r}{\partial z^2}|_{z_0} = -\frac{1}{\sigma^2}H$$

With no loss of generality, we assumed H is a diagonal matrix, since we can always rotate the coordinates of z to an eigen basis of H, diagonalizing it. In previous work, the number of tuned feature dimensions by an IT neuron was hypothesized to be one variable underlying the trade-off between selectivity and tolerance in IT (Zoccolan *et al.*, 2007). In our model, we changed the number of tuned features by controlling the eigen-spectrum (*i.e.,* diagonal values) of H. We made the simplifying assumption that the Hessian matrix had only two different eigenvalues: 1 and $\epsilon = 10^{-6} \approx 0$. In the eigen space of $\lambda = 1$, the model neuron had a Gaussian bell-shaped tuning curve with tuning width σ along each dimension. In the eigenspace of $\lambda = \epsilon$, the neuron had effectively flat tuning — it was agnostic to the feature changes along those dimensions. We let the neuron have D tuning dimensions (with eigenvalue 1) and $4096 - D$ non-tuning dimensions (with eigenvalue ϵ), then the Hessian matrix is $H = \mathrm{diag}([1,1, \dots 1, \epsilon, \epsilon, \dots, \epsilon])$. In this experiment, we manipulated the number of *tuned dimensions* D, and the general tuning width σ. We illustrated the effect of D and σ in 2-dimension in Figure IV-6 **A.** The center z_0 of the tuning function was chosen isotropically on the sphere of radius R, by sampling a 4096-dimensional random vector from normal distribution and normalizing its norm $\|z_0\| = R$.

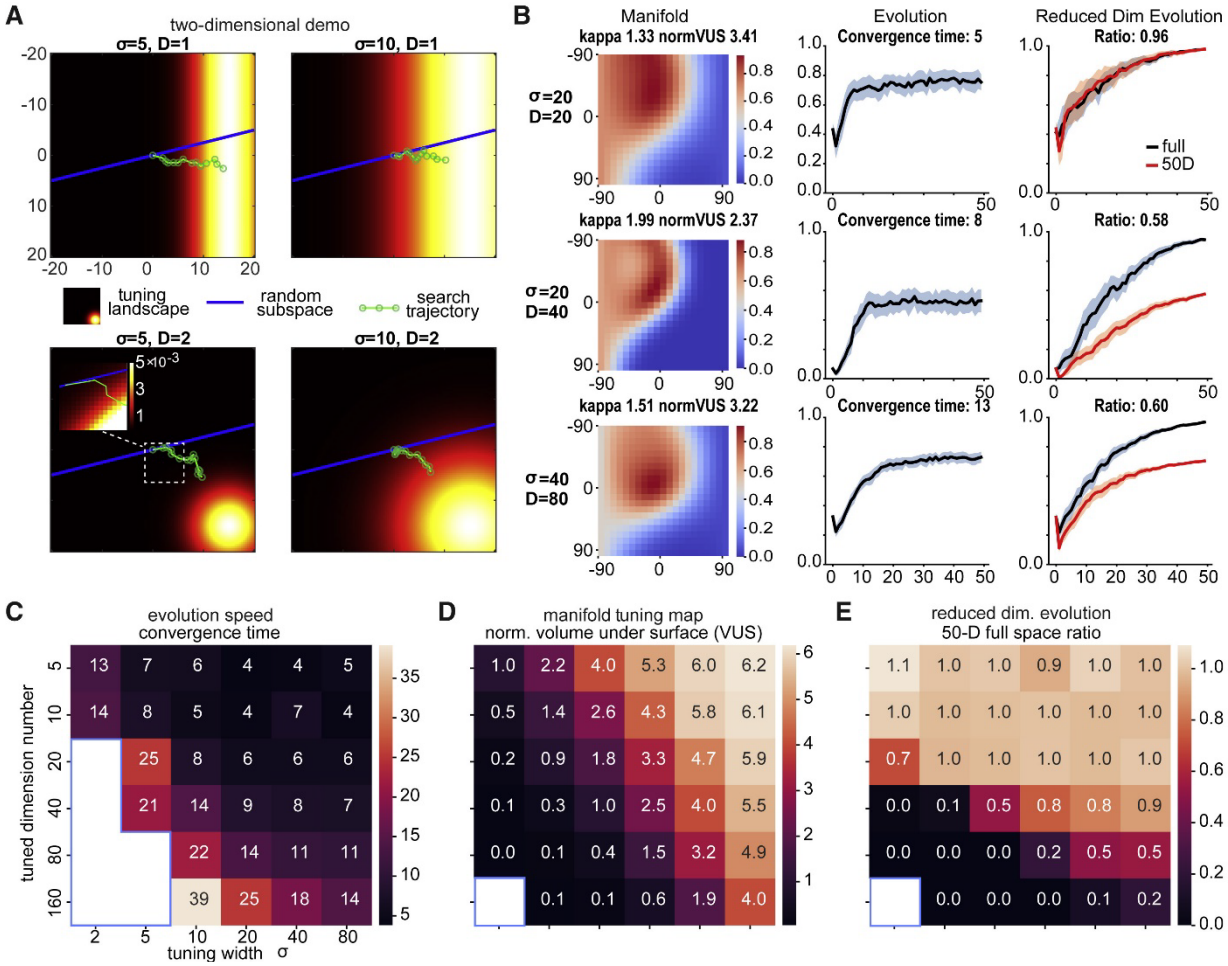


Figure IV-6 **Figure 6**. **Inferring tuning landscape geometry. A.** Two-dimensional demonstration of the main model, showing evolution paths (green) over image space when the activation function has different values for tuning width or dimensionality. **B.** *Evolution*, *Manifold* and *Reduced-Dimensionality* experiments in three example conditions ($\boldsymbol{\sigma = 20, D = 20}$; $\boldsymbol{\sigma = 20, D = 40}$; $\boldsymbol{\sigma = 40, D = 80}$). Relevant parameters annotated on top. **C.D.E.** depict the phase space spanned by the tuned dimension $\boldsymbol{D}$ and the tuning width $\boldsymbol{\sigma}$. Each panel plots a statistic of the *Evolution-Manifold* experiment as a function of $\boldsymbol{D}$ and $\boldsymbol{\sigma}$, averaged over 5 repetitions. **C.** Evolution convergence time, in the unit of generation. On this heatmap, the white region defined by blue boundary denotes the conditions where *Evolutions* failed. **D.** The tuning width of *Manifold* experiment quantified by normalized Volume under Surface. White region outlines conditions where the Manifold tuning map was flat. **E.** Ratio between the final activation of 50D and full space *Evolution* search. White region outlines conditions where both 50D and full space *Evolution* failed.

Evolution speed. We hypothesized that the tuning width σ would affect the success rate and convergence time of the *Evolution* experiments. Here, we found that given the same number of

tuned features D, the larger the tuning width σ, the faster the search converged. At the other extreme, an overly small tuning width led to a failed *Evolution* for a larger tuned dimension ($D = 20{\sim}160$). When the tuning width σ was fixed, the time required to converge also increased as a function of tuned dimension D. These trends made sense: if the tuning function covers a tiny region in the whole image space and has close to zero value everywhere else, then it will be nearly impossible for the evolutionary algorithm to find a slope and to climb the mountain; and if there are more tuned axes D, it should take longer to optimize each. Thus, the first systematic change of increasing search convergence time along the hierarchy could be captured by moderately decreasing the tuning width σ and increasing the number of tuned features D (note that this model did not incorporate noise as in neuronal recordings, so convergence time comparisons were limited).

Manifold tuning width. We hypothesized that the width of Gaussian tuning function σ would affect the tuning width (normalized VUS) measured in the Manifold experiment. These quantities were not the same: the manifold on which we measured the tuning sharpness was not guaranteed to reside in the tuned subspace of the neuron. Second, the peak found by the *Evolution* experiment was not guaranteed to be the global maximum of the tuning function. Despite that, we found that the manifold result could be reproduced by the difference in σ and surprisingly, also by the tuned dimension D (Figure IV-6 **D**). Intuitively, the larger the tuning width σ, the broader tuned it was at the peak found by the *Evolution* experiments. Moreover, given the same σ, the smaller the number of tuned dimensions D, the broader tuned it was in a *Manifold* experiment. Intuitively, along those non-tuned axes, the "neuron" showed an infinitely wide tuning curve. Thus, if the subspace covered by the PC2,3 axes mixed up some tuned- and some untuned dimensions, it would lead to a broader tuning curve. Notably, the shape of tuning

map in this simplified example (**Figure IV-6 B up, middle**) resembled the shape of tuning maps for some V1 neurons observed *in vivo*.

Effect of reduced dimension. Finally, we hypothesized that the larger the tuned dimensionality D, the larger the detrimental effect of constraining searches to a random 50D subspace. We found we could reproduce this observed effect by changing the number of tuned dimensions D. With a small number of tuned dimensions $(e.g.\ D = 10)$, the two evolutions (4096D vs 50D) resulted in indistinguishable evolution trajectories, regardless of the tuning width σ (**Figure IV-6 B, E**). This scenario is comparable what we observed for V1 neurons. However, with a larger D, the difference between 4096D and 50D evolution emerged. Specifically, with the same tuning width σ, the performance gap increased with the number of tuned dimensions D; given the same D, the performance gap decreased with the tuning width σ (**Figure IV-6 E**). Overall, this model provided a simple explanation of the systematic changes we observed *in vivo* and *in silico.* Under the assumptions of this model, there is a likely increase of tuned dimensionality D across stages of the ventral stream, potentially accompanied by a change in tuning width σ. This inference generalizes previous results on the tuned dimensions of IT neurons (Zoccolan *et al.*, 2007).

IV.3 Discussion

To generalize our understanding of neuronal responses in the natural world, we must characterize tuning using stimuli of appropriate complexity, comprising enough visual attributes to evoke the full response range of given neurons. Driven by this motivation, we have assumed a neuro-centric (vs. strictly human-interpretable) perspective, asking how neurons respond when the visual world changes smoothly around their preferred stimulus, regardless of the specific transformation. We believe this brings us one step closer to natural conditions, illustrating how

neurons respond to generic smooth transformations, before they learn statistical associations between different images — those helpful for invariance, for example (Li & DiCarlo, 2010). We paired complex-image generators with the classical *tuning function*, which maps neuronal responses to values of a given parameter space. Tuning functions comprise at least one neuronal response peak and a measure of response decay as stimuli deviate from this preferred value. This decay is important because when viewed in the framework of Fisher information, neuronal responses are highly informative about the stimulus regions causing the largest changes in firing rate (Dayan and Abbott, 2005). However, this is complicated in scenarios when neuronal responses reflect information about multiple variables (Benichoux *et al.*, 2017), and in situations of high neuronal variability, when it is the peak that can be most informative (Butts and Goldman, 2006). Consequently, we prioritized identification of response peaks corresponding to the neuron's "preferred" combination of visual attributes, not because we believe that neurons can only be informative when signaling at their maximal response, but because to a downstream decoder, any change in a neuron's response must signify a change from *something*. From views in classical single-neuron studies to those distributed coding frameworks, that *something* is thought to be the visual attributes or parameters that best activate the neuron.

Once we found a location in image space evoking a maximal response, we had to choose how to "move" away from that location. In our data, there seemed to be no special axes for deviating from the peak – all were relatively isotropic, at least within the expected range of neuronal response variability. This is consistent with work applying basis function interpolation as a solution to the problem of categorizing novel views or objects — as described by Edelman, "*the shape of the basis function reflects the prior knowledge concerning the change in the output as one moves away from the data point…In the absence of evidence to the contrary, all*

directions of movement are considered equivalent, making it reasonable to assume that the basis function is radial" (Edelman, 1999). But since we only measured 2-D sections of larger spaces, this may not be true if more directions were probed. Imaginably, there could be special locations linking these peaks, some inducing invariance responses, some inducing sharper activity drop-off. So ultimately, how does the view of neuronal *tuning landscapes* relate to invariance? To answer this question, more electrophysiology experiments are required (and underway). Preliminary findings suggest that invariance is not a constant feature of the neuron but depends on the activity evoked by the choice of probe stimuli. While we leave this for future work, these findings highlight the importance of testing the full response dynamics of neurons.

Overall, we conclude that it serves to think of neuronal tuning in multi-attribute feature space the same way as we think about hippocampal place cells acting in a two-dimensional arena, where a neuron's primary responses depend on the proximity to a physical location, less on the direction in which the animal moves to or from it (but see (McNaughton, Barnes and O'Keefe, 1983) for an effect of entering direction on place field). To push this analogy further, recent works support the view that place cells form a map not just for physical spaces but also for cognitive spaces. Namely, they could encode "locations" in the multi-dimensional space spanned by task related variables, such as pitch of auditory cues, accumulated evidence, or even social status of a virtual character (Tavares *et al.*, 2015; Aronov, Nevers and Tank, 2017; O'keefe *et al.*, 2017). So, the tuning landscape view of visual encoding in the ventral stream connects to the modern view of hippocampus in the cognitive space.

Author Contributions.

BW: Conceptualization, Methodology, Software, Formal analysis, Investigation, Data Curation, Writing – Original Draft, Visualization. **CRP:** Conceptualization, Methodology, Validation, Investigation, Resources, Writing – Review and Editing, Visualization, Supervision, Funding Acquisition.

Acknowledgments.

We thank C. Harvey and C. Pack for comments on the manuscript, J. Freeman and E. Simoncelli for inspiring the title, T. Holy for the initial motivation and discussion throughout the project. This work was supported by the David and Lucile Packard Foundation (#2020-71377), the E. Matilda Ziegler Foundation for the Blind and the NSF CCF-1231216 Center for Brains, Minds and Machines.

IV.4 Method

IV.4.1 Neuronal recording

Two male adult rhesus macaques (A and B) were implanted with chronic floating microelectrode arrays (Microprobes for Life Sciences, MD) in the right hemisphere: one array was located at the posterior lip of the lunate sulcus (corresponding to the V1/V2 transition), one on the prelunate gyrus (V4) and another anterior to the inferior occipital sulcus (PIT). We refer to these sites as V1/V2, V4 and IT (Gattass, Gross and Sandell, 1981; Gattass, Sousa and Gross, 1988; Boussaoud, Desimone and Ungerleider, 1991). The locations of the arrays were chosen based on sulcal landmarks, around local vasculature.

Neurophysiology data was acquired using OmniPlex Neural Recording Data Acquisition Systems (Plexon Inc.), with the PlexControl client to sort electrical events online based on waveform and interspike intervals. Because all *Evolution* experiments were based on a closed loop between neuronal activity and image synthesis, spike sorting was done at the beginning of each experiment. Within each channel, events were estimated as arising from single-units, multi-units or "hash" using a 1-5 scale, where 1 indicated strong confidence on the presence of a single-unit (based on waveform shape, inter-spike interval and separation from the main hash signal) and 5 indicated hash/multiunit activity. We use the term *site* to refer to all signal types; across experiments, sites comprised mostly multiunits/hash and a fraction of single units.

After data collection, spike/event times were discretized into 1-ms bins and convolved with a symmetric Gaussian probability density function with a 2-ms standard deviation.

IV.4.2 General animal experiment setup

Experimental sessions were run using MonkeyLogic2 (Hwang, no date), which directed the presentation of visual stimuli on ViewPixx EEG monitors (ViewPixx Technologies). Refresh rate was 120 Hz at a resolution of 1920x1080 pixels. The monkeys were placed 58 cm from the screen. Gaze position was tracked via ISCAN cameras (ISCAN Inc.). Animals fixated 0.25°-diameter circles, with fixation window that permitted eye movements up to 1.0–1.3° from the fixation point during stimulus presentation; they obtained reward if they held their gaze on the target for 2-3 s. Rewards delivered via DARIS Control Module System (Crist Instruments).

IV.4.3 Receptive fields mapping

At the start of each experimental session, receptive field locations were estimated as follows. While the animal performed a fixation task, at each moment, a single square test image was presented at a single position for 100-ms. The image could be a photograph or a previously

collected generator image and on any given presentation, it could be sized either at 1°-, 2°- or 4°-width. Positions were randomly sampled from grids ranging from [-2° to 2°], [-4° to 4°] or [-8° to 8°] of the central visual field, in steps of 1°, 2°or 4° for the three sizes. After data collection, neuronal responses (events/second) were quantified as a function of stimuli location. A 2-D Gaussian function was fit to this 2-D response grid to estimate the center of the receptive field. Because the test image width was so large and because we did not sample a dense enough position grid, we did not obtain a strict estimate of RF size (particularly for V1/V2 and V4 sites) and our data over-estimates it. The receptive-field center distribution of all neuronal sites from V1/V2, V4 and IT visual areas are shown in Figure IV-7 **A**. After estimating RF center location, *Evolution* and *Manifold* experiments were carried out with stimuli at the estimated center location, sized to cover the region with most activity. Usually, stimuli were made larger than the estimated RF size. This helped to prevent the site from responding to the high-contrast, salient image edge. Moreover, larger image sizes provided a canvas to engage both the classical RF and its surround during optimization of the neuronal response.

IV.4.4 Evolution experiments

After finding the optimal location the evolving textures, the experiment started by presenting 30 synthetic images and 10 reference images, only one presentation per image. Reference images were selected if they were known to evoke high activity from the array site under study, per independent experiments. The initial 30 synthetic images were approximations of Portilla and Simoncelli textures (Portilla and Simoncelli, 2000) recreated in the generator. After all images were presented, their input vectors and the site's spike rate responses (averaged over 50-200 ms after image onset) were provided to the update function of Cholesky CMA-ES algorithm, which then gave 40 new vectors as outputs. These vectors were provided to the generator to create 40

new images, and the cycle began again, i.e., the 40 synthetic images and the same 10 reference images were presented to the subject. Each experiment comprised tens of cycles (*generations*) and were stopped 10-20 generations after firing rate convergence was observed.

IV.4.5 Manifold experiments

After evolving a preferred stimulus (a *prototype*), the goal was to measure the tuning landscape around it, by sampling images in a smooth, continuous fashion. By examining the distribution of latent vectors during the *Evolution* experiments and applying principal component analysis, we found a properly scaled PC1 recreated the measured prototype — PC1 vector re-created an image like the evolved image in the final generation, once scaled to the norm and sign of the observed prototype latent vector. Image contrast could be manipulated via positive scaling of the latent vector (illustrated in **Figure III-16 A** in **Chapter III**, (Wang and Carlos R. Ponce, 2022b)). Settling on PC1 $\boldsymbol{v_1}$ of the trajectory to represent the prototype, the next two vectors orthogonal to PC1 were used to form a basis of three vectors. A two-dimensional sphere was defined in the subspace spanned by these three vectors, and images were sampled along the azimuth and elevation angle θ, ϕ uniformly. Samples were along θ, ϕ in $[-\frac{\pi}{2}, \frac{\pi}{2}]$ with $\frac{\pi}{10}$ interval, chosen to make the transition between neighboring images perceptually smooth. As a result, for each manifold experiment, there were $121 = 11 \times 11$ points sampled on the hemisphere around the prototype.

$$\boldsymbol{z}(\theta, \phi) = [\boldsymbol{v_1}, \boldsymbol{v_2}, \boldsymbol{v_3}] \cdot [\cos\theta \cos\phi\,, \sin\theta \cos\phi\,, \sin\phi]^T$$

$$\theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right], \phi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$$

Note these points did not form a uniform grid on the sphere. The distortion by mapping a 2d grid onto a spherical surface was greater near the "north pole" and "south pole" of the sphere; for example, the 11 images at either $\phi = \frac{\pi}{2}$ or $-\frac{\pi}{2}$ were exactly the same, resulting in 101 unique images in each hemisphere. Thus, we correct for this distortion effect in the subsequent computation of tuning functions and statistics of tuning maps.

We performed 46 experiment sessions on monkey A, and 45 experiment sessions on monkey B. As we mentioned above, in 10 of the monkey B experiments, we performed *Manifold* experiments in three subspaces following the *Evolution* experiment, using PC2-3, PC49-50, and two random vectors as the perturbation vectors $\boldsymbol{v_2}, \boldsymbol{v_3}$ in the *Manifold* experiments. Thus, we had $N = 91$ standard experimental pairs with an *Evolution* experiment and a *Manifold* experiment in PC2-3 space. Initial analysis found that the tuning maps in the PC49-50 and random subspaces were comparable to those in PC2-3 space. So, when characterizing the tuning maps, we pooled the additional maps (PC49-50, Random) with the PC2-3 maps, resulting in $N = 111$ tuning maps of neuronal sites driving the *Evolution*.

IV.4.6 Reduced-dimension Evolution experiments

To probe the global geometry of the tuning landscape, and to investigate how it changed from posterior to anterior visual cortex, we performed *reduced-dimension Evolution* experiments. In each session, after receptive field mapping, we performed paired *Evolution* experiments, using the same neuronal unit as the driving unit in two parallel *Evolution* experiments, using independent optimizers. We called each *Evolution* a thread. The stimuli created by the two *Evolution* threads were presented in an interleaved fashion. One thread was the *reduced-dimension Evolution,* with the optimizer constrained to search in a randomly selected 50-dimensional subspace; the other thread was the full-space *Evolution*, searching in the

full 4096-dimensional space. The 50-dimensional subspace was selected independently for each session. We developed a new optimizer that operated on the hypersphere of arbitrary dimension *SphereCMA* (Wang and Carlos R. Ponce, 2022c) for these experiments. This ensured that the search codes would have the same norm throughout the *Evolution*. Without this constraint, using the original Cholesky CMA-ES, the vector norm of the codes differed greatly in the two threads, which made it difficult to compare the two threads. The same optimizer was applied to each *Evolution* thread, with the same population size ($N = 31$) but different dimensionalities (4096D vs 50D). The same 10 reference images were used for both threads. We performed 34 sessions on each monkey (A and B), within which 23 sessions were driven by V1 neurons, 20 by V4 neurons, and 25 by IT neurons.

IV.4.7 CNN models of the ventral stream

In the past few years, many CNN models have been developed to solve the object recognition problem. Though generally, these models incorporate many working principles of the ventral stream, some of them are better models than others. To find the CNN models that resemble the population representations in the primate ventral stream the most, we consulted the BrainScore (Schrimpf *et al.*, 2018) (http://www.brain-score.org/) leaderboard and selected a few top and classic networks: AlexNet (Krizhevsky, Sutskever and Hinton, 2012) (No. 57), VGG16 (Simonyan and Zisserman, 2015) (No.5), ResNet50 (He *et al.*, 2016) (No.11), ResNet50-Robust (Madry *et al.*, 2017) (No.3), ResNet101 (He *et al.*, 2016) (No.4), DenseNet-169 (Huang *et al.*, 2016) (No.9), CorNet-S (Kubilius *et al.*, 2018) (No. 1). Source and specification of the models are as follows, all implemented in PyTorch (Paszke *et al.*, 2019). ResNet50-Robust weights were obtained from https://github.com/MadryLab/robustness , ε = 8 / 255 version. CorNet-s model definition and weights from https://github.com/dicarlolab/CORnet. All other models'

definitions and weights are obtained from the model zoo in *torchvision* https://pytorch.org/vision/stable/models.html. All seven models were used to analyze tuning sharpness progression along the visual hierarchy (**Figure IV-4 D-E, Figure IV-12 A-F**). AlexNet, VGG16, ResNet50).

IV.4.8 *In silico* Evolution-, Manifold and Reduced-Dimension experiments.

To corroborate our findings in the ventral stream hierarchy *in vivo*, we performed parallel experiments for CNN. For each network, we picked the major convolutional and fully connected layers ($n = 8 - 16$). For convolutional layers, we selected the units in the center of the feature map of each channel, mirroring the process of us presenting the image at the receptive field (RF) of recorded neurons.

For each unit, we first measured its receptive field by backpropagating its activation back to the image, where we defined a square box around the pixels that contributed to its activation. This "receptive field" was in turn used to resize the images. Next, we used the activity of this unit to drive the *Evolution* experiment, using the same CMA-ES algorithm and generator. Finally, we used the same method to analyze the trajectory and to sample images on 2d hemispheres around the prototype as we did in the *Manifold* experiments. We measured the activations of the recorded units responding to the manifold image set, thus obtaining the tuning maps of these CNN units.

For the reduced-dimension *Evolution*, we replicated these experiments using units from the eight layers of CaffeNet and AlexNet, with 100 units selected from the feature map center of the

first 100 channels of each layer[8]. For each unit, we performed a reduced-dimension *Evolution* in a 50, 100, 200, and 400-dimensional random subspace and the 4096d full space. For each dimensionality for each unit, the *Evolution* was repeated 10 times with independently sampled random subspace. We used the same optimization algorithm (SphereCMA) *in vivo* and *in silico.* The score trajectories were analyzed in the same fashion *in vivo* and *in silico*.

IV.5 Quantification and Statistical Analysis

IV.5.1 Fitting tuning maps with Kent function.

By visual inspection, neuronal tuning functions were usually unimodal and likely to be well-fitted with a Gaussian function, but since the domain of the sampled tuning function was not a flat Euclidean space, it was not strictly correct to fit these responses using a Gaussian function. Thus, we used the Kent function, which is a natural analogue to the Gaussian function on the 2-D sphere S^2.

$$f(\boldsymbol{x}) = A\exp\{\kappa\,\boldsymbol{\gamma_1}^T\cdot\boldsymbol{x} + \beta[(\boldsymbol{\gamma_2}^T\cdot\boldsymbol{x})^2 - (\boldsymbol{\gamma_3}^T\cdot\boldsymbol{x})^2]\} + b$$

where $\boldsymbol{x}$ is a 3D vector, which, in our case, is the 3D coordinate of the latent vector on the PC basis of the *Manifold* experiments. $\boldsymbol{\gamma_1}, \boldsymbol{\gamma_2}, \boldsymbol{\gamma_3}$ form an orthonormal basis set in 3D space, in which $\boldsymbol{\gamma_1}$ is the direction of the center of the distribution; and $\boldsymbol{\gamma_2}, \boldsymbol{\gamma_3}$ are analogues to the maximal and minimal covariance axes, if the peak is anisotropic. We parametrized the 3D orthonormal basis set using the following angle convention: θ, ϕ represent the azimuth and elevation angles corresponding to $\boldsymbol{\gamma_1}$ vector, while ψ is the angle of $\boldsymbol{\gamma_2}$ to the equator of the sphere. Thus, when the function is unimodal (single peak), θ, ϕ parametrize the location of the peak on the sphere, and ψ encodes the direction of elongation around the peak. Beyond these,

[8] Except for the first layer, conv1, which has fewer channels.

A, b control the scaling and baseline, κ controls the concentration (peaked-ness) of the function, β controls the degree of anisotropy around the peak. Overall, this function comprises 7 parameters $A, b, \kappa, \beta, \theta, \phi, \psi$. We fit the Kent function to the tuning maps from *in silico* experiments or *in vivo* experiments using `curve_fit` from `SciPy` in `Python` or `fit` function in Matlab.

IV.5.2 Noise ceiling of explained variance.

To control trial-to-trial variability in tuning map fitting, we computed the noise ceiling of explained variance R^2 as follows. For each *Manifold* session, we resampled the single trial neuronal responses to each image to get a bootstrapped mean tuning map $r'[i]$. Then we computed the explained variance of the original mean tuning map $r[i]$ by the bootstrapped one $r'[i]$.

$$R_{bstrp} = 1 - \frac{\sum_i (r[i] - r'[i])^2}{\sum_i (r[i] - \bar{r})^2}$$

This procedure was replicated 500 times to estimate the average $\bar{R}_{bstrp}$, which we regarded as the ceiling of explainable variance when fitting the noisy tuning map.

IV.5.3 Quantifying tuning width using Volume Under the Surface.

We generalized the idea of *area under the curve* (AUC) to *the volume under the surface* (VUS) for the 2-D tuning map in our scenario. The tuning map was defined on a hemisphere $A[\theta, \phi]$, with $\theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right], \phi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$; we integrated the evoked firing rate over the hemisphere and normalized its peak firing rate to 1. This statistic has a theoretical maximum of $2\pi \approx 6.28$ which is obtained when the neuron responds equally and above baseline over the whole hemisphere; it also has a theoretical minimum of 0, where no place on the hemisphere

(i.e., no image) evoked activity above baseline. The value of VUS can be interpreted as the equivalent area on the hemisphere that evoked the maximal firing rate in this space:

$$VUS = \frac{1}{\bar{A} - b} \int d\theta d\phi \max(A[\theta, \phi] - b, 0)\,, \bar{A} = \max A[\theta, \phi]$$

The baseline b is the mean firing rate in the [0,50]-ms period after stimulus onset, across all trials in the experiment. $A[\theta, \phi]$ is computed as the mean firing rate in [51,200] milliseconds. Numerical integration on the sphere was performed using the method in Rafaely (2019) (Rafaely, 2019).

IV.5.4 Quantifying tuning map smoothness by Dirichlet Energy.

To quantify the smoothness of a tuning map, we adapted a functional from mathematics, the Dirichlet energy of a function. This functional integrates the squared norm of gradient in the domain of the function, which, in our case, is a spherical manifold. Here we used finite difference estimation of gradient and discrete quadrature over sphere for integration. The distortion effect of the non-uniform sampling on sphere is accounted for in this calculation.

$$DE = \int_M \|\nabla f(x)\|^2 dx$$

To estimate how trial-to-trial variability affected DE, we performed 1000 trials resampling for each image, thus we obtained a distribution of average tuning maps and a distribution of Dirichlet energy. To estimate the null distribution, we shuffled the correspondence of the response and the position on the image manifold and computed the Dirichlet energy again (**Figure IV-7 B**). We used the t statistic and Cohen's d' of these two distributions to quantify the smoothness of a tuning map. Recall that Cohen's d' is the difference between the mean of two distributions measured in the unit of combined standard deviation. So, a wider null distribution

or a wider trial resampled distribution will make d' smaller, while a larger separation between the Dirichlet energy of the shuffled map and the resampled map will create a larger smoothness measure (**Figure IV-7E**).

Similarly, total variation energy is defined as follows, i.e., integrating the norm of the gradient (without squaring) on the manifold.

$$TVE = \int_M \|\nabla f(x)\| dx$$

We replicated our analysis of smoothness using this TVE measure, with equivalent results: 94/110 experiment with smaller total variation energy than shuffled control ($P < 0.001$), range of $T \in [-277.6, -5.8]$, Cohen's $d' \in [-12.41, -0.26]$.

IV.5.5 Quantifying activation increase in *Evolution* experiments.

We wanted to design a statistic to quantify the activation increase during *Evolution* experiment. Inspired by delta fluorescence over fluorescence (DFOF) in two-photon imaging, we used the difference of activation over initial activation (DAOA) to quantify the relative increase.

$$DAOA = \frac{\bar{r}_{end,50:200} - \bar{r}_{init,50:200}}{\bar{r}_{init,50:200}}$$

This statistic was key to connect the *Evolution* experiment to Manifold tuning maps (**Figure IV-8 E**). We used the κ in Kent fitting to quantify sharpness of tuning maps. In all experiments where the tuning maps were well-fit by a Kent function ($R^2 > 0.5$), the Spearman correlation between DAOA and κ was 0.609 ($P = 1.0 \times 10^{-8}, N = 76$). One possibility was that this correlation was mediated by the areal difference of tuning width: we found that neuronal sites in higher visual cortex had sharper tuning maps and larger activation increase in *Evolution*

experiments. We tested this by computing the correlation for neuronal sites in three cortical areas separately, and we found these correlation values were also statistically significant: V1 ($\rho = 0.564, P = 2.6 \times 10^{-3},\ N = 27$), V4 ($\rho = 0.492\ , P = 0.029,\ N = 20$), IT ($\rho = 0.586, P = 1.0 \times 10^{-3},\ N = 29$). Pictorially, the neuron-guided *Evolution* that "climbed" to a taller peak ($\bar{r}_{end,50:200}$) on the tuning landscape or started at a lower level ($\bar{r}_{init,50:200}$), tended to reach a sharper peak measured by the *Manifold* experiment. This relationship was consistent with the picture of the *Evolution* experiment allowing neuronal searches for a peak on the tuning landscape, and the *Manifold* experiment characterizing the tuning around the peak.

IV.5.6 Comparing tuning across image space via radial tuning curve analysis.

In this experiment, we measured site responses to images to different image sets: Manifold, gratings, curved objects, and photographs, resulting in image-response pairs $\{I_i, \bar{r}_i\}$ for each image set. We first computed the image distance matrices between all pairs of images using the LPIPS distance D (Zhang *et al.*, 2018)

$$d[i,j] = D(I_i, I_j)$$

Then, we found the image evoking the highest response in the neuron, i.e., the tuning peak in that image space.

$$k = \arg\max_i \bar{r}_i\,, \qquad r_{MAX} = \max_i \bar{r}_i$$

Finally, we fit the neuronal response as a function $\hat{f}$ of the image distance to the highest activating image I_k, this is the *radial tuning function*. Specifically, we used a non-parametric fitting method, Gaussian process regression (Matlab function *fitrgp.m*).

$$\hat{f} = \text{fitrgp}(d[k,:], r[:])$$

To compute the area under the curve, we integrated the area under the estimated function; the definite integral is performed from 0 to the maximal distance from the peak image D_{MAX} (Figure IV-3**A** right). Normalized AUC was computed by dividing the area under the curve by the peak activation.

$$AUC = \int_0^{D_{MAX}} \hat{f}(x)dx\,, D_{MAX} = \max_i d[k,i]$$

$$normAUC = AUC/r_{MAX}$$

As the distance matrix was computed between all image pairs, the *normAUC* statistic quantified how fast the response decreased from a peak on the tuning landscape, averaging the different directions of deviating from the peak.

IV.5.7 Convergence speeds of *Evolution* experiments

For each *Evolution* experiment, we computed the response firing rate in the evoked time window [50,200] ms for every image. Then we used Gaussian process regression (Rasmussen & Williams, 2006) (*fitrgp.m* in Matlab) to obtain the smoothed and averaged optimization trajectory $\hat{f}(t)$. Next, we computed the generation number C_{63} when 63.2% of maximal activation r_{max}, (i.e., $0.632(r_{max} - r_{init}) + r_{init}$) was reached, quantifying the time scale of convergence per *Evolution* experiment.

IV.5.8 Effect of dimensionality restriction on *Evolutions*

For the reduced-dimension *Evolution* experiments, we measured the effect of constraining search dimensionality on the activation increase for units in all three cortical areas (V1/V2, V4, and IT). We defined a ratio R between the activation increases in 4096D and 50D as follows:

$$R = \frac{\overline{r_{50d,end}} - \overline{r_{50d,init}}}{\overline{r_{4096d,end}} - \overline{r_{4096d,init}}}$$

$\overline{r_{50d,init}}$ and $\overline{r_{50d,end}}$ are the average firing rate for all images in the first and the last generation in the 50D reduced-dimension *Evolution*; $\overline{r_{4096d,init}}$ and $\overline{r_{4096d,end}}$ are those average firing rates for the full space *Evolution*.

To reduce the noise in the single-trial neural responses, we also used the integrated d' to quantify the difference: we calculated d' between the set of single trial firing rates in each generation of 4096D vs 50D *Evolution* (i in the following equation), and then averaged the d' over all generations.

$$\overline{d'} = \frac{1}{N}\sum_{i}^{N} d'\left(r_{50d,i}[:], r_{4096d,i}[:]\right)$$

IV.5.9 Correlated feature attribution

The goal of this model was to localize the visual attributes — shapes, colors, and textures — selected by neurons during the *Evolution* experiments, within the whole evolved image. While this can be approximated by superimposing the independently measured receptive fields of the neurons, this analysis presents an alternative that works using only the *Evolution* data itself. We encourage the readers to read our paper dedicated to this method (Wang and Carlos R Ponce, 2022) and our code (*https://github.com/Animadversio/Neuronal_Feature_Attribution_Model*). Briefly, our tactic was to rely on CNN feature (hidden) units sharing similar response properties as the recorded neuron. First, we picked a convolutional layer in a pretrained CNN (e.g., AlexNet, VGG-16, ResNet-50, ResNet-50-robust) and computed the correlation and covariance of each unit with the observed (V1/V2, V4 or IT) neuronal responses across all *Evolution* images

$\{I_i^e\}$. The correlation and covariance were both tensors with the same shape as the activation tensor of a layer to a single image $F[c, x, y]$. Using Python convention of indexing, the tensors were

$$C[c, x, y] = corr(r[:], F[:, c, x, y])$$

$$Q[c, x, y] = cov(r[:], F[:, c, x, y])$$

Note that thousands of images were displayed in the *Evolution* experiments. Given the hundreds of thousands of features in the convolutional layer, the subsequent memory cost required us to compute correlation values via online updates. We built a custom pipeline to compute the correlation and covariance value for each feature unit when propagating images through the neural network, in a batch-update fashion. Next, we selected the most correlated feature units by thresholding the t statistics associated with correlation value $t = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}}$. We set the threshold as $t \geq 3$ for the presented results, although we varied this threshold to test the robustness with no changes in overall conclusions. The units with correlation values less than this threshold were excluded by setting their weight to zero. Because CNNs usually exhibit boundary artifacts, we also excluded the feature units at the border of feature maps. The original covariance tensor $Q[c, x, y]$ became the sparse tensor $\tilde{Q}[c, x, y]$ after unit exclusion. Note, we chose to include only the positively correlated units in the model, because we found the feature visualization from it more intuitive.

Next, we factorized the sparse $\tilde{Q}$ tensor through non-negative matrix factorization (Lee and Seung, 1999, 2001) (decomposing a spatial-channel tensor into the product of a few non-negative spatial masks and channel vectors).

$$\arg\min_{V_r,S_r} \left\| \tilde{Q}[c,x,y] - \sum_r V_r[c]S_r[x,y] \right\|_2^2, S_r[x,y] \geq 0, V_r[c] \geq 0$$

We used the coordinate-descent solver with Frobenius norm minimization as objective and *nndsvda* initialization as implemented in `scikit-learn` package (Pedregosa *et al.*, 2011). Here we kept the factor number to three. The order of the factors was arbitrary, but the magnitude of each factor signified how much it contributed to the whole tensor, potentially representing the feature's importance to the *Evolution* process. As an example, factorizing the $\tilde{Q}$ tensor in layer 3 of ResNet50-robust model to 3 components usually accounted for 0.132 ± 0.003 variance of the original tensor ($N = 90$).

Finally, we used penalized regression (Ridge) to determine the weights w_r of the three factors, such that these weights multiplying the CNN activations predict the neuronal activity the best.

$$\min_{\mathrm{w_r}} \sum_i \left\| r[i] - \sum_{c,x,y,r} F[i,c,x,y]V_r[c]S_r[x,y]w_r \right\|_2^2$$

This model could predict the neuronal response to *Evolution* and *Manifold* experiment well, Pearson correlation between predicted and actual neuronal response to Manifold images was 0.70 ± 0.02 across $N = 90$ sessions; correlation was 0.67 ± 0.03 when including all reference images. We visualized the spatial masks combined by weights $|\sum_{r=1}^{3} w_r S_r[x,y]|$ as the feature attribution mask (**Figure IV-4A (ii)**), with the brighter region representing the image area correlated with and probably contributing to higher neuronal activation.

Because the goal of this analysis was to find an image-level attribution instead of just predicting neural responses, we used correlation rather than regression to identify image regions related to changes in neural activity. Directly using penalized regression on the full feature tensor or dimension reduced feature tensor (per sparse random projection or principal component analysis) often resulted in an overly sparse weight tensor with no coherent spatial structure, inadequate for interpretation (Wang and Carlos R. Ponce, 2022a). Moreover, we chose to use matrix factorization instead of just mean or max compression of the weight tensor across the channel dimension, since this disambiguated unique features and highlighted the spatially coherent features more clearly. Visually, it broke down a complex feature into spatial composition of several simpler ones.

IV.5.10 Inclusion criteria for non-driving units

In addition to the driving units, we also recorded the activities of other neuronal sites in our three electrode arrays. We used one-way ANOVA tests to select neuronal sites that were modulated in the Manifold image space, using image identity as the main factor with the criterion of $p < 0.001$. These well-modulated sites were included in other analyses, comprising $N = 3427$ non-driving- and 104 driving sites; these constituted 43.4% and 1.3% of all recorded units.

IV.5.11 Measuring tuning map similarity

We measured the similarity of tuning maps based on *functional correlation* (Wikipedia Editors, n.d.) on the manifold, which generalized the correlation of functions in Euclidean space. This could be interpreted as the angular similarity between two mean-subtracted functions defined on the manifold. Our rationale for using this method is as follows. As the tuning maps we measured were defined over a hemisphere, some points were sampled closer to each other

than to the others (e.g., around the hemisphere "poles"), thus the similarity of the response at those points in any tuning maps was not surprising. Functional correlation tackles this problem elegantly by considering the underlying metric structure over which we are sampling the map.

The equations for this correlation follow below; the integrals were evaluated using discrete quadrature on the hemisphere (Rafaely, 2019). J is the Jacobian of the map from $[\theta, \phi]$ parameter to the Euclidean coordinate of the hemisphere.

$$\bar{f} = \int_M dx \, |\det J| f(x), \qquad Var[f] = \int_M dx \, |\det J| \big(f(x) - \bar{f}\big)^2$$

$$\langle f, g \rangle = \int_M dx |\det J| \, f(x) g(x)$$

$$corr(f, g) = \frac{\langle f - \bar{f}, g - \bar{g} \rangle}{\sqrt{Var[f] \cdot Var[g]}}$$

Note that correlation in Euclidean space yields an even higher correlation value, so our results were not artifacts created by the correlation calculated on a spherical domain.

IV.5.12 Naïve Bayes decoding for population neural activity

We used Naïve Bayes decoding (Rish and Rish, 2001) method to assess how variability affected decoding. We assumed the response of each neuronal site was an independent normal variable with a mean $\mu_i(I)$ and variance $\sigma_i^2(I)$ depending on the image identity I. Thus, we could write the likelihood of an image identity given a vector of neural responses as

$$\log \mathcal{L}(I|\boldsymbol{r}) = \sum_i -\frac{1}{2}\left(\frac{r_i - \mu_i(I)}{\sigma_i(I)}\right)^2 - \log \sigma_i$$

Then assuming a uniform prior on the image identity, we could get the distribution of image identity based on a response vector using Bayes' rule.

$$p(I_j|\boldsymbol{r}) = softmax(\log \mathcal{L}(I|\boldsymbol{r}))_j$$

With this conditional distribution, we estimated the expected decoding error in terms of the L2 or angular distance between the latent vectors corresponding to the image

$$\mathbb{E}_{L2} = \sum_j \|z_j - z_{real}\| p(I_j|\boldsymbol{r})$$

$$\mathbb{E}_{ang} = \sum_j \arccos\langle z_j, z_{real}\rangle \, p(I_j|\boldsymbol{r})$$

IV.6 Supplementary Figures

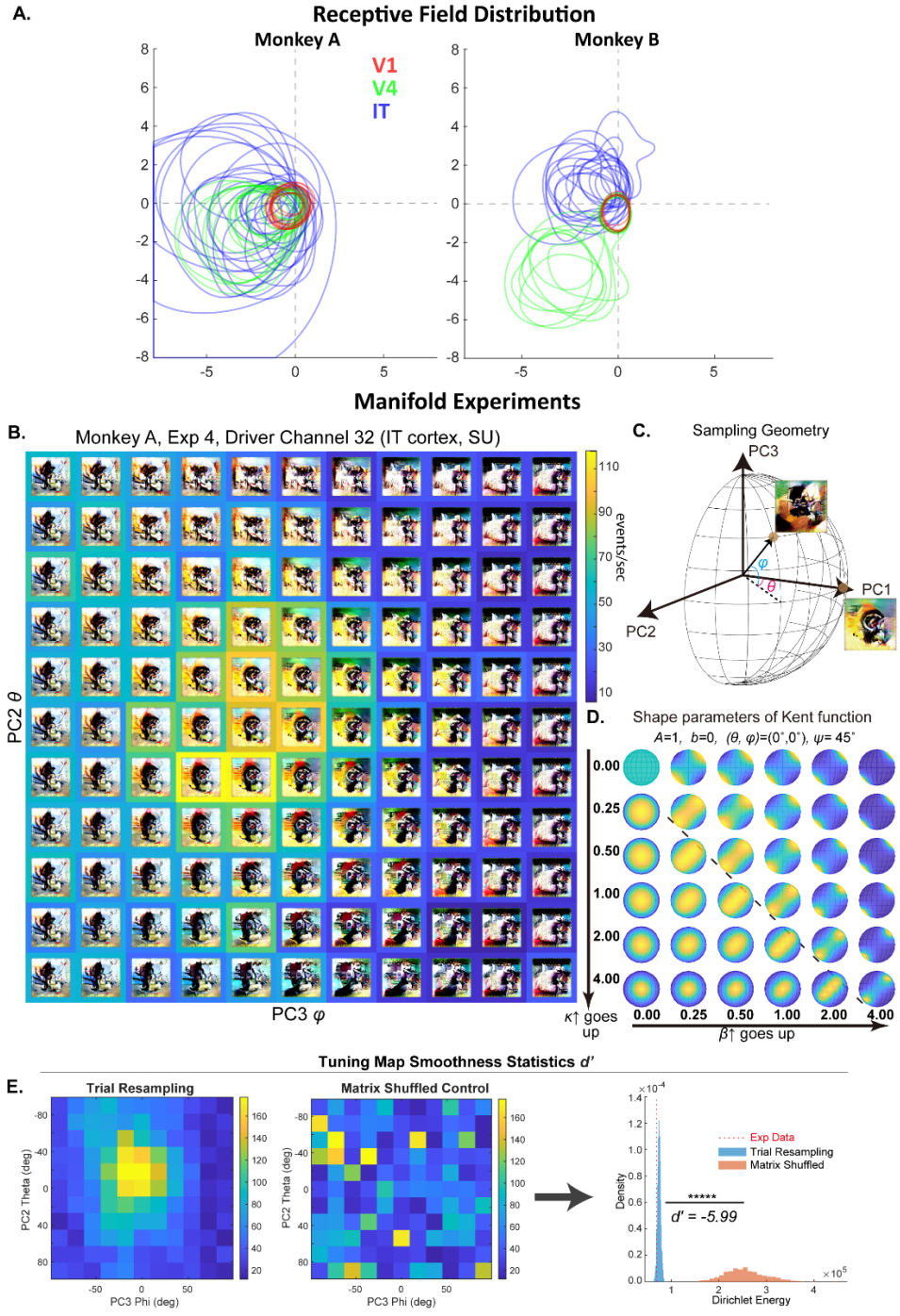


Figure IV-7 **S1 Experimental method details.** **A.** Receptive fields of recorded sites from V1/V2, V4 and IT in both monkeys (A, left; B, right). Each contour shows the estimated retinal area evoking more than 50% of the maximal neuronal response for each site (response defined as mean firing rate in 50-200ms subtracting baseline firing rate). Contour color indicates the visual area of the neuronal sites: red for V1/V2, green for V4, blue for IT. X and Y axes are retinal coordinates, showing degrees of visual angle, where (0,0) as the fixation point. **B.** An example

tuning map of an IT single-unit in monkey A; the images are framed by color representing the neuronal response rate (color bar, events / s), see **Figure IV-2 C**. **C.** Demonstration of the sampling geometry in the principal components 1,2,3 space. In the Kent function, $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ correspond to the azimuth (longitude) and elevation (latitude) angle in a 3-D sphere defined by PC1-3. **D.** Effect of shape parameters $\boldsymbol{\kappa}, \boldsymbol{\beta}$ on the Kent function, with other parameters fixed $\boldsymbol{A = 1, b = 0, \theta = 0^{\circ}, \phi = 0^{\circ}, \psi = 45^{\circ}}$. A larger $\boldsymbol{\kappa}$ creates a sharper peak, while a larger $\boldsymbol{\beta}$ makes the peak less isotropic around it. The tilted elongation direction displays the effect of $\boldsymbol{\psi = 45^{\circ}}$. Note the dashed line $\boldsymbol{\kappa = 2\beta}$ in the parameter space, which separates unimodal and bi-modal Kent functions. **E.** The procedure of computing the smoothness index $\boldsymbol{d'}$. For each tuning map, we resampled the single trial responses to obtain a distribution of maps (*trial resampling*); we also shuffled the correspondence of mean response and the coordinates to get another set of maps (*matrix shuffled control*). We computed the Dirichlet Energy (DE) of each map and computed the $\boldsymbol{d'}$ distance between the two distributions to quantify the smoothness of the map. Related to Sec. IV.4 **STAR Method** and **Figure IV-2** the ***Results*** section **IV.2.1** ***Neurons show smooth, bell-shaped tuning around peaks in the generator space***.

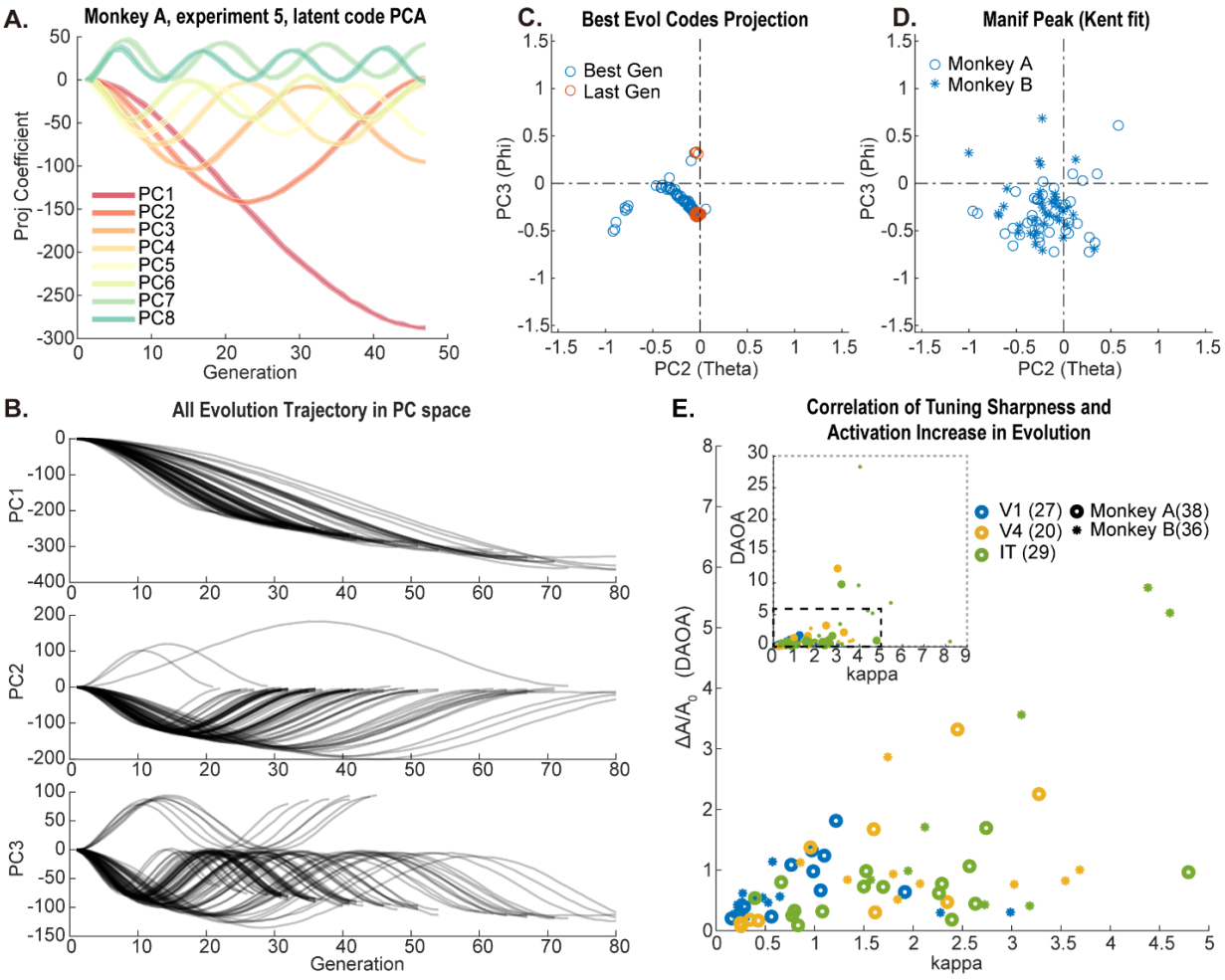


Figure IV-8 **S2 Geometry of evolution trajectories and their relationships to tuning maps.** A-B show the structure of the evolution trajectory through the lens of PCA, see also **Figure IV-2** A. A. Example *Evolution* trajectory projected the top (1–8) principal components. Data from monkey A, driven by an IT neuronal site. B. Collection of all *Evolution* trajectories ($\boldsymbol{N = 91}$) projected onto their top three principal components. Projection coefficients are plotted on the y axes, with arbitrary units representing length in the latent space. This highlights the sinusoidal PC structure of the *Evolution* trajectory, which is characterized more thoroughly in (Wang and Ponce, 2022). We note that the end point of *Evolution* trajectory did not fall exactly along the PC1 vector, thus we predicted that the peak location in Manifold tuning maps should also deviate from the center in a systematic way. C-D show the location of peak on Manifold tuning maps correlates with the location of peak during *Evolution*, when projected on to the top-3 PC subspace. C. The spherical coordinates ($\boldsymbol{\theta, \phi}$) of latent codes during *Evolution*, for the highest activation (best) generation (blue) and the last generation (orange), for all well-fit *Manifold* experiments ($\boldsymbol{N = 76}$). We found that $\boldsymbol{\theta, \phi}$ of the best generation were both negative: mean ± SEM of $\boldsymbol{\theta}$ was $\boldsymbol{-0.220 \pm 0.024}$ rad and $\boldsymbol{\phi}$, $\boldsymbol{-0.188 \pm 0.015}$ **rad**. D. Peak location of tuning maps estimated by Kent function in all well-fit *Manifold* experiments ($\boldsymbol{N = 76}$). Consistent with C., the $\boldsymbol{\theta, \phi}$ coordinates of the tuning peak were also negative: mean ± SEM of $\boldsymbol{\theta}$ was $\boldsymbol{-0.217 \pm 0.041}$ **rad**, and $\boldsymbol{\phi}$ $\boldsymbol{-0.304 \pm 0.035}$ **rad**. Further, the spherical coordinates of the highest activating *Evolution* codes positively correlated with the coordinates of the peak location in the corresponding *Manifold* experiment (Pearson correlation for $\boldsymbol{\theta}$ is $\boldsymbol{0.378, P = 7.7 \times 10^{-4}}$; for $\boldsymbol{\phi}$ is $\boldsymbol{0.430, P = 1.1 \times 10^{-4}, N = 76}$). E. The relation between tuning sharpness and *Evolution* success. We found sharpness of *Manifold* tuning maps $\boldsymbol{\kappa}$ positively correlated with the relative activation increase ($\boldsymbol{DAOA}$, see STAR methods) in the *Evolution* experiment (Spearman correlation $\boldsymbol{0.609, P = 1.0 \times 10^{-8}, N = 76}$). The correlation holds for individual visual

areas: V1 ($\boldsymbol{\rho = 0.564, P = 2.6 \times 10^{-3},\ N = 27}$), V4 ($\boldsymbol{\rho = 0.492\ , P = 0.029,\ N = 20}$), IT ($\boldsymbol{\rho = 0.586, P = 1.0 \times 10^{-3},\ N = 29}$). The main plot magnifies the dashed rectangle region of the inset plot, showing that the positive correlation is not a result of outliers. Simply put, *Evolution* experiments with larger activation increases led to sharper peaks in the *Manifold* experiments. Related to **Figure IV-1** and **2** in the ***Results*** section IV.2.1 ***Neurons show smooth, bell-shaped tuning around peaks in the generator space.***

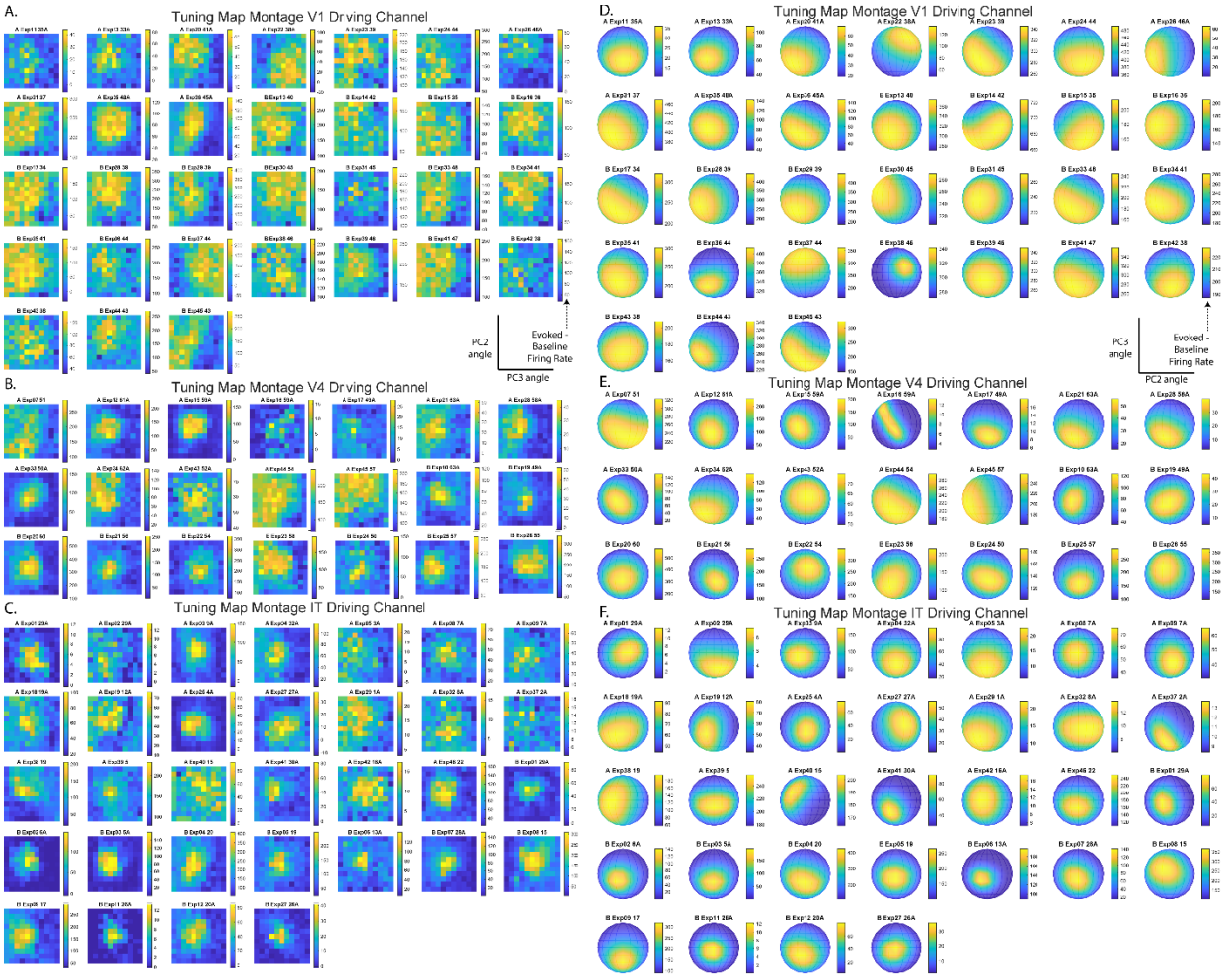


Figure IV-9 **S3 All significantly modulated tuning maps of *Evolution-Manifold* driving units, per visual area** (**A-C** and **D-F** correspond to V1/V2, V4, IT, in both monkeys A, B), sorted by collection order. **A-C**, show the raw tuning map, the mean activation values on the $(\boldsymbol{\theta, \phi})$ parameter grid. **D-F**, show the Kent function fits of tuning maps on the hemisphere. Note the gradual change of tuning width and shape from V1/V2, V4 to posterior IT. Related to Figure IV-2 in the *Results* section IV.2.1 *Neurons show smooth, bell-shaped tuning around peaks in the generator space*, and Figure IV-4 in the section IV.2.3 *Areal differences of tuning landscapes*.

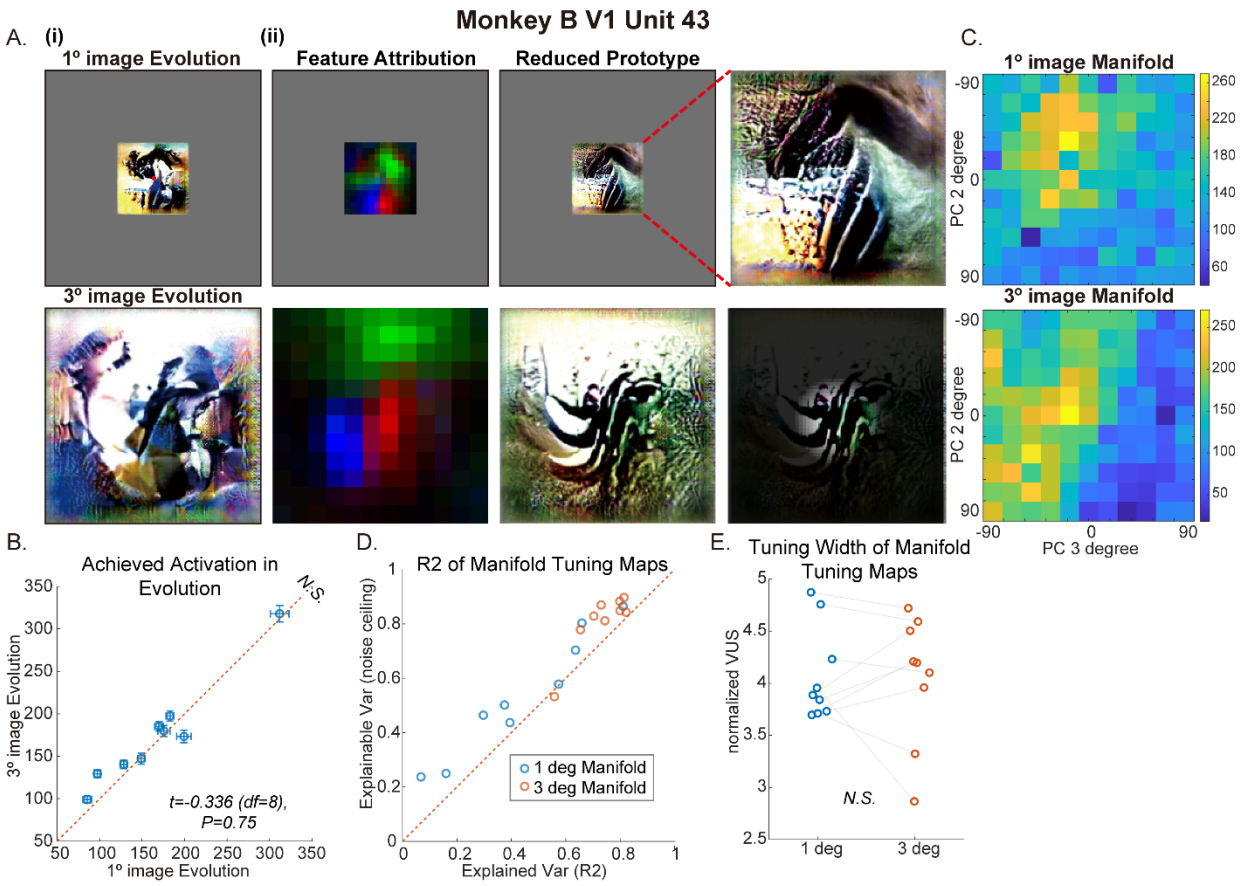


Figure IV-10 S4 ***Evolution* and *Manifold* Experiments using different image sizes yielded comparable results**. In this figure, we examined how image size used in *Evolution* and *Manifold* experiments could have affected results. We conducted nine experiment pairs (18 sessions, each had an *Evolution* and a *Manifold*) in monkey B, focusing on V1 sites. In each pair of sessions, one used a 1°x1° image size for the *Evolution* and *Manifold* experiments, while the other used a 3°x3° image size. We matched all other factors: the same V1 site was used as the driving unit; images were centered at the same estimated receptive field center location; each pair of experiments were carried out on the same day to control for recording quality. **A.** Example of a pair of *Evolution* experiments using 1°-wide (upper) and 3°-wide-image (lower) driven by the same V1 site in monkey B in the same session, the images resized to maintain the 1:3 ratio. The columns (from left to right) show the prototypes, the feature attribution masks and the model-reduced prototype by feature visualizing; The last column shows the zoom-in of the 1° reduced prototype and the 3° reduced prototype with the receptive field mask: 100% transparency for area with maximum activation in receptive field mapping experiment, 10% transparency for area with lower than half of maximal activation. One can appreciate the similarity of the oblique grating-like features selected by the neuron, given different image sizes. **B.** Paired comparison of the achieved activations (mean firing rate in the last two generations) in the 1°- and 3°- *Evolutions.* All *Evolutions* succeeded (18/18) and in 6/9 pairs of *Evolution*, the activation achieved in both threads were not statistically different. This showed that choosing an image size larger than classic receptive field may not affect the success of *Evolution*. **C.** Tuning map of *Manifold* experiments for the 1°- and 3°- *Evolutions* showed in **A**. **D.** $\boldsymbol{R^2}$ of Kent fitting and

noise ceiling of these $\boldsymbol{R^2}$ plotted against each other. 1° image Manifold tuning maps had a lower noise ceiling and a lower explained variance: $\boldsymbol{R^2}$ for 1° tuning maps was $\boldsymbol{0.442 \pm 0.087}$ ($\boldsymbol{n = 9}$, mean ± SEM same below) versus $\boldsymbol{0.736 \pm 0.031 (n = 9)}$ for 3° ones; noise ceiling of $\boldsymbol{R^2}$ for 1° tuning maps was $\boldsymbol{0.538 \pm 0.079}$ comparing to $\boldsymbol{0.810 \pm 0.039}$ for 3° ones. **E.** Paired comparison of tuning width of 1°- and 3°- Manifold tuning maps, using the normalized Volume under the Surface statistics. The difference was not statistically significant. Related to **Figure IV-2** and **Figure IV-4** and the ***Results*** section IV.2.3 ***Areal differences of tuning landscapes***.

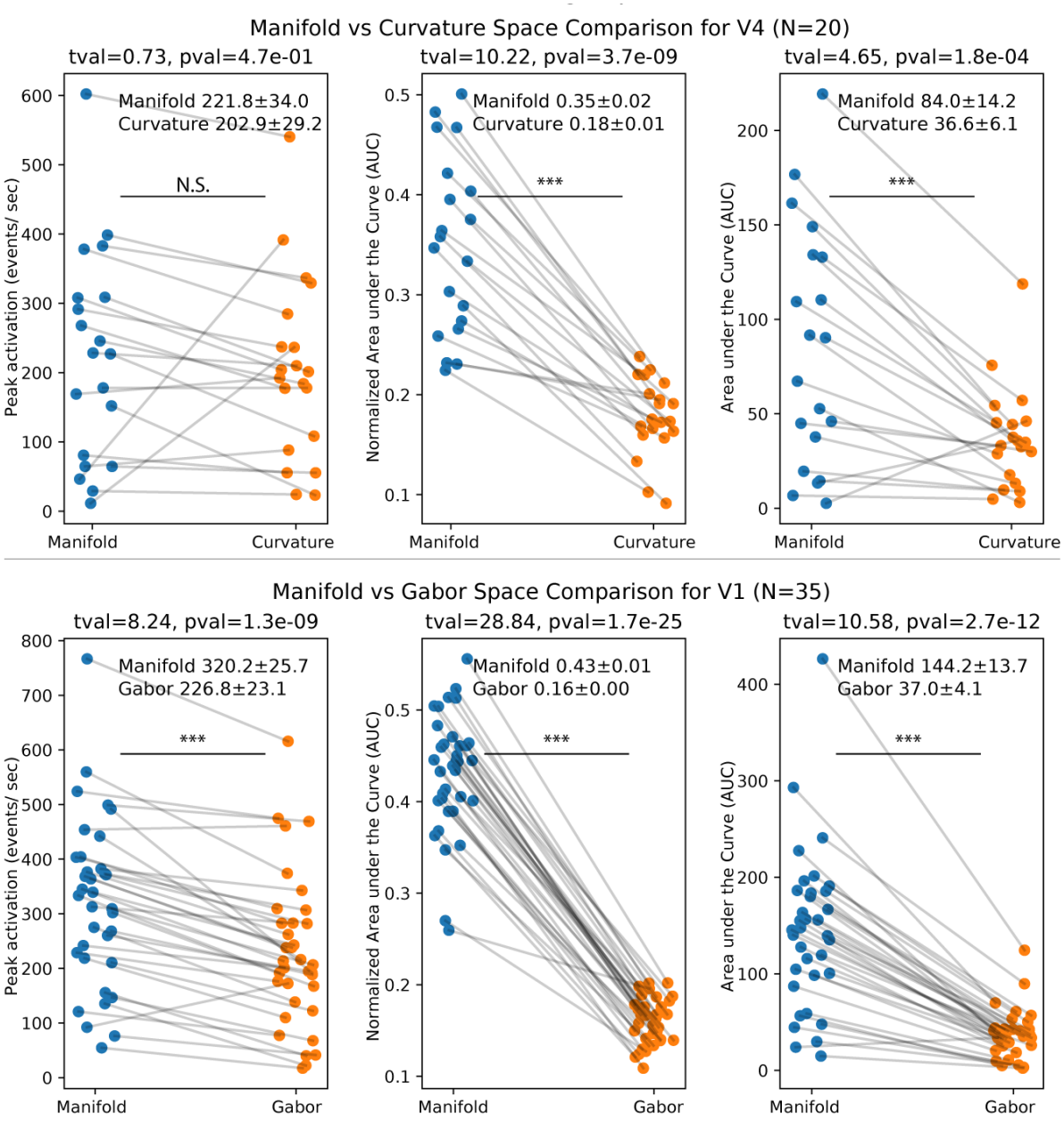


Figure IV-11 S5 Comparison of radial tuning in manifold space with classic image spaces for V1 and V4. Comparing to Figure IV-3, here we conducted a focused analysis comparing the neural tuning in Manifold image space with the tuning in image spaces that are classically used to stimulate these areas. Bottom row compares Manifold vs Gabor patches for V1 sites, top row compares Manifold space vs Curvature images for V4 sites. Left column: peak activation; middle column: normalized Area under Curve (normAUC); right column: area under curve (the product of peak activation and normAUC). For V1, the peak activation in Manifold image space was higher than that in Gabor patches space; the radial tuning was also broader in manifold image space. For V4, the peak activation in Manifold image space was comparable to that of the Curvature image space; while the radial tuning is still broader. This showed that the optimized generator images were as activating or more activating than those in the classic image spaces for

V1 and V4. Further, we found a larger tuning width or slower decay of firing rate in terms of image distance in the generator space. We interpreted that though more features were manipulated in the Manifold space, the neuron is only tuned for a subset of them (see Figure IV-6). Thus, compared to classic image space where only few features were manipulated, the tuning width on Manifold image space was larger. Related to Figure IV-3 and the *Results* section IV.2.2 *Relating neuronal tuning in generator space to other image spaces*.

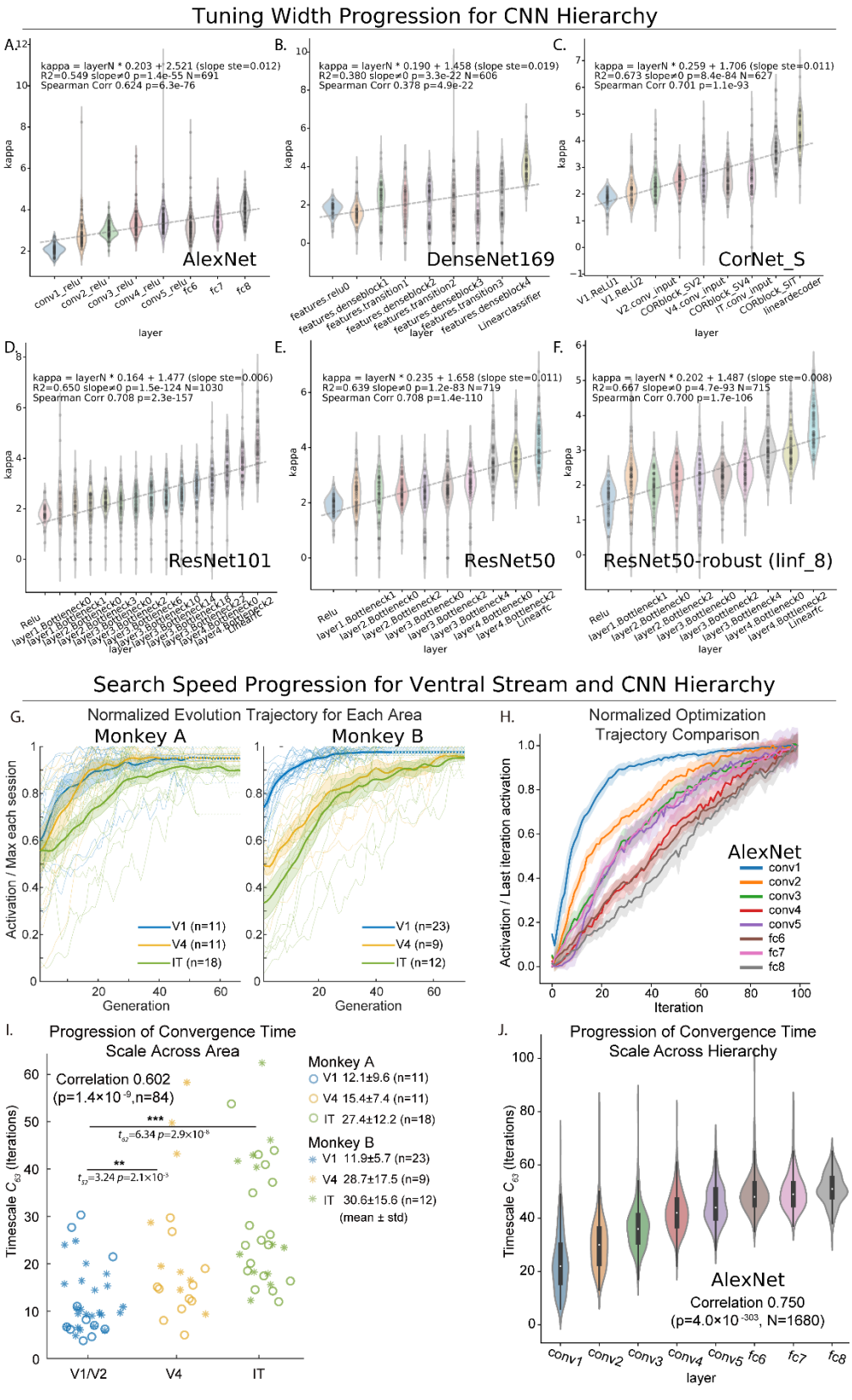

Tuning Width Progression for CNN Hierarchy
A.
kappa = layerN * 0.203 + 2.521 (slope ste=0.012)
R2=0.549 slope≠0 p=1.4e-55 N=691
Spearman Corr 0.624 p=6.3e-76
AlexNet
B.
kappa = layerN * 0.190 + 1.458 (slope ste=0.019)
R2=0.380 slope≠0 p=3.3e-22 N=606
Spearman Corr 0.378 p=4.9e-22
DenseNet169
C.
kappa = layerN * 0.259 + 1.706 (slope ste=0.011)
R2=0.673 slope≠0 p=8.4e-84 N=627
Spearman Corr 0.701 p=1.1e-93
CorNet_S
D.
kappa = layerN * 0.164 + 1.477 (slope ste=0.006)
R2=0.650 slope≠0 p=1.5e-124 N=1030
Spearman Corr 0.708 p=2.3e-157
ResNet101
E.
kappa = layerN * 0.235 + 1.658 (slope ste=0.011)
R2=0.639 slope≠0 p=1.2e-83 N=719
Spearman Corr 0.708 p=1.4e-110
ResNet50
F.
kappa = layerN * 0.202 + 1.487 (slope ste=0.008)
R2=0.667 slope≠0 p=4.7e-93 N=715
Spearman Corr 0.700 p=1.7e-106
ResNet50-robust (linf_8)
kappa
layer
Search Speed Progression for Ventral Stream and CNN Hierarchy
G.
Normalized Evolution Trajectory for Each Area
Monkey A
Monkey B
Activation / Max each session
Generation
V1 (n=11)
V4 (n=11)
IT (n=18)
V1 (n=23)
V4 (n=9)
IT (n=12)
H.
Normalized Optimization Trajectory Comparison
Activation / Last iteration activation
Iteration
AlexNet
conv1
conv2
conv3
conv4
conv5
fc6
fc7
fc8
I.
Progression of Convergence Time Scale Across Area
Timescale C_{63} (Iterations)
Correlation 0.602
(p=1.4×10 -9,n=84)
***
t_{82}=6.34 p=2.9×10-8
**
t_{82}=3.24 p=2.1×10-3
V1/V2
V4
IT
Monkey A
V1 12.1±9.6 (n=11)
V4 15.4±7.4 (n=11)
IT 27.4±12.2 (n=18)
Monkey B
V1 11.9±5.7 (n=23)
V4 28.7±17.5 (n=9)
IT 30.6±15.6 (n=12)
(mean ± std)
J.
Progression of Convergence Time Scale Across Hierarchy
Timescale C_{63} (Iterations)
AlexNet
Correlation 0.750
(p=4.0×10 -303, N=1680)
layer

Figure IV-12 S6 **Tuning width progression and convergence speed progression in visual hierarchies *in silico* and *in vivo*. A-F.** Tuning width decreases as a function of depth for *in silico* visual hierarchy. For the *Manifold* experiments in artificial neural network, kappa value increased as a function of layer depth; tested models were AlexNet, DenseNet-169, CorNet-S, ResNet-101, ResNet-50, ResNet50-Robust. For more information on model specification and source, see **STAR Methods**. **G-J.** Search convergence time increases along *in vivo* and *in silico* visual hierarchy. **G.** Average normalized optimization trajectories in V1, V4, IT for both monkeys. For each *Evolution* experiment, we averaged neuronal responses within each generation, and smoothed responses across generations with a moving average ($\boldsymbol{N = 3}$ generations), then normalized the curve to the maximal average activation in that experiment to 1.0 (individual experiment showed in thin lines). Since *Evolution* experiments had different numbers of generations, we *extrapolated* the early-ending curves with a constant value, putting forth the assumption that they would keep the same activation of the last two blocks — extrapolated parts shown in dashed lines. Then we averaged the normalized optimization trajectories across experiments in each brain area (thick lines). **H.** Convergence time scale $\boldsymbol{C_{63}}$ comparison across areas for both monkeys. The mean$\pm$std of $\boldsymbol{C_{63}}$ for each area and animal was annotated on the side. **I.** Example normalized optimization trajectories for *in silico* evolution for units in AlexNet, one unit per layer. Mean and standard deviation of activation per generation are shown. **J.** Convergence time scale $\boldsymbol{C_{63}}$ of *in silico evolution* for units in the eight layers in AlexNet. Related to **Figure IV-4 and 5** in the ***Results*** section **IV.2.3** ***Areal differences of tuning landscapes***.

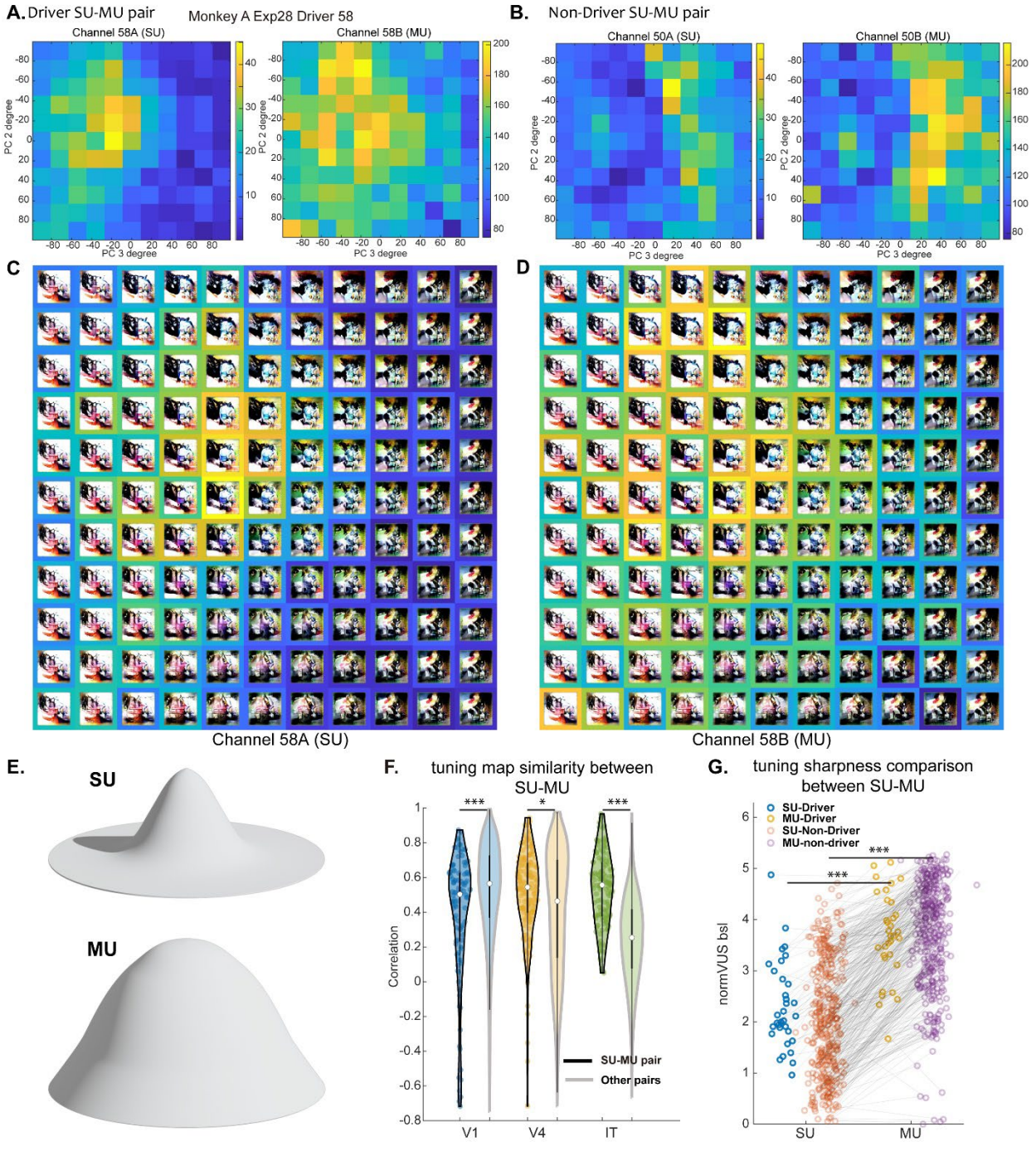


Figure IV-13 **S7 Tuning maps of paired single- and multi-units.** In this figure, we compared properties of tuning landscapes of single-units (SUs) or multi-units (MUs) in each *Manifold* experiment. Classically, multi-units are understood as the aggregated signal of local neuronal populations. So, we reasoned that the MU landscapes should be an additive version of SU landscapes — taller and broader in their peaks, but still similarly placed relative to the SU landscapes. **A-D.** Tuning maps from experiment session 28 in monkey A, with channel 58 in the V4 array as the driver. In all plots, color frames show the activation score for the image,

computed as $\bar{r}_{50:200} - \bar{r}_{bsl}$, with the baseline firing rate $\bar{r}_{bsl}$ in [0-50] ms averaged across the whole session. **A.** The SU-MU pair in the driving channel 58; only the spike rate of the SU was sent to the *Evolution* optimizer (CMA-ES) as score. **B.** Another SU-MU pair, channel 50 in V4, not driving the *Evolution*. **C.D.** Tuning maps of the same SU-MU pair as **A,** showing the actual image corresponding to the response. **E.** The schematics representing our observation: the SU-MU pair exhibited correlated tuning maps, with the MU's tuning peak being broader and higher than the paired SU. **F.** Same-channel SU-MU pair exhibited similar tuning maps. In V4/IT cortex, SU-MU pairs were more similar comparing to average level in the array. In IT, the mean correlation of SU-MU pairs was $\mathbf{0.574}$ ($\boldsymbol{n = 135}$), for any well-modulated unit pairs it was 0.256 ($\boldsymbol{n = 9540}$), significantly larger (two-sample t-test with Fisher r-to-z transform, SU-MU > other pairs, $\boldsymbol{t_{9673} = 16.173, P = 4.5 \times 10^{-58}}$). **G.** SU maps exhibited sharper tuning maps compared to the paired (same channel) MU, as quantified by normalized volume under surface (normVUS) statistics. The mean±sem normVUS for SU was $\mathbf{2.062 \pm 0.054}$ ($\boldsymbol{N = 408}$), for paired the MU it was $\mathbf{3.702 \pm 0.051}$, significantly smaller (SU - MU, paired t-test, $\boldsymbol{t_{407} = -35.917, P = 2.9 \times 10^{-128}}$). This relationship holds both for driver and non-driver units. Related to **Figure IV-4** in the ***Results*** section IV.2.3 ***Areal differences of tuning landscapes***.

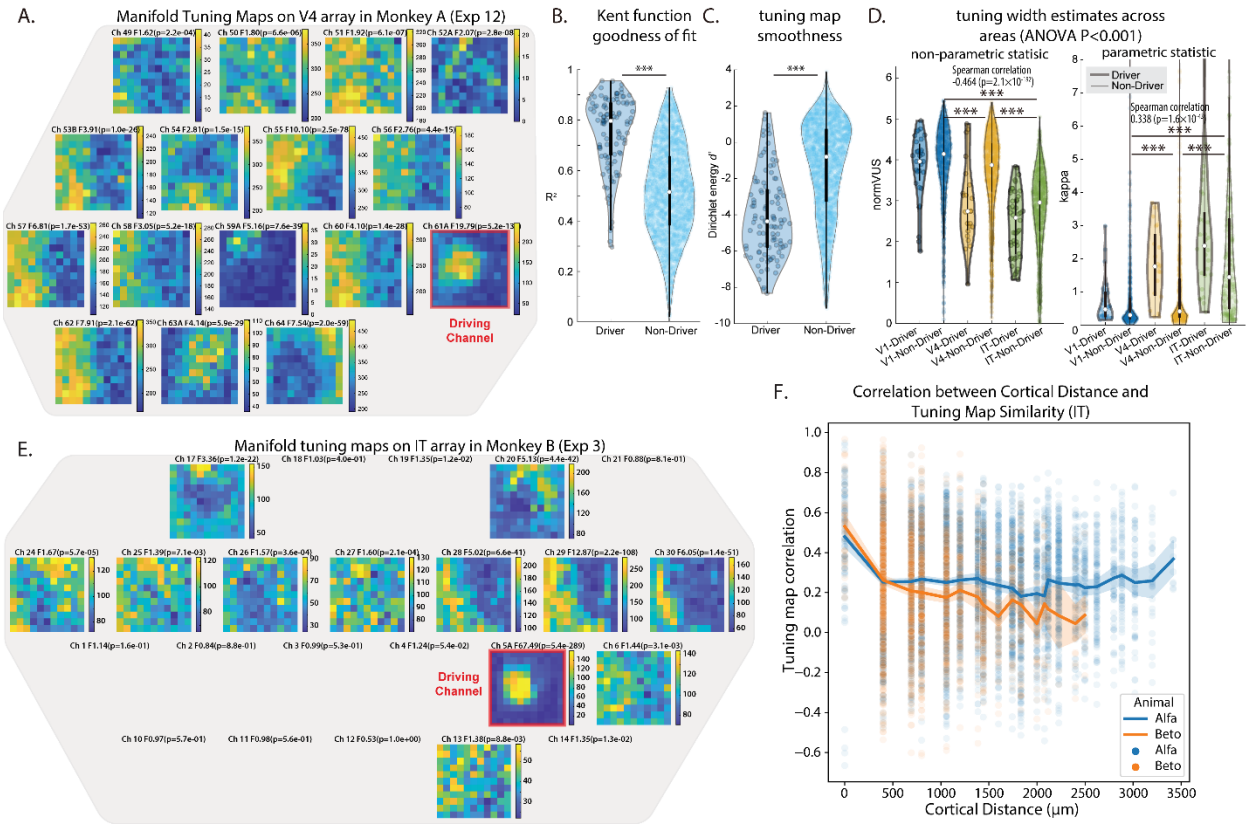


Figure IV-14 **S8 Tuning landscape away from the peak and similarity of tuning maps along the cortical surface**. For driving units, by our experimental design, the 2d *Manifold* image space is sampled around the activation maximizing images found through *Evolution*. This allowed us to characterize the general shape of tuning around a peak on the tuning landscape (Figure IV-2, Figure IV-4). To characterize properties of tuning landscape not centered on the peak, we measured the tuning maps for all the non-driving units simultaneously recorded in the electrode arrays during *Manifold* experiment. In **A-D**, we quantified individual maps using the same statistics as in Figure IV-2 and Figure IV-4, showing they followed similar trend as the driving units. **A.** Tuning maps in the *Manifold* image space of a population of neurons in V4 area in monkey A, in an example session (12) with driving site 61 marked by a red frame. The tuning maps of all channels in V4 array are arranged in the same layout as the electrodes on the visual cortex. Channel number, F statistics and p-value of ANOVA annotated on top of each tuning map. **B.** Tuning maps of driver units are better fit by the Kent function, as quantified by fitting $\boldsymbol{R^2}$. **C.** Tuning maps of driver units are smoother than that of the non-driver units as quantified by the $\boldsymbol{d'}$ statistics of the Dirichlet Energy. **D**. The tuning width of non-driver units get sharper along the visual hierarchy similar to the driver units, as measured by non-parametric statistic *normVUS* and parametric statistic *kappa* in Kent function. In **E-F**, we quantified the change of tuning maps along the cortical surface to investigate the possibility of a cortical feature map. **E.** Tuning maps are plotted in the same layout as the electrodes in IT-implanted array on the visual cortex in monkey B, for an example session with driving site 5 marked by a red frame. Units not significantly modulated by this image set ($\boldsymbol{p > 0.01}$) were not shown. One can appreciate a subtle, gradual change in tuning map shape, esp. in peak location from channel 24 to channel 30. **F.** Tuning map similarity as a function of cortical distance between IT electrodes for each

monkey, pooling all electrode pairs that are both well-modulated (ANOVA $\boldsymbol{p < 0.01}$). Cortical distance of $\boldsymbol{0\mu m}$ refers to the case where both units were spiked sorted from one electrode, e.g. the case of a single and a multi-unit pair. Though both monkeys have a significant negative trend, we found that monkey B had a much stronger negative correlation (Spearman correlation $\boldsymbol{-0.180}$, $\boldsymbol{P = 2.4 \times 10^{-18}}$, $\boldsymbol{N = 2322}$) than the other monkey (A) (Spearman correlation $\boldsymbol{-0.037}$, $\boldsymbol{P = 0.002}$, $\boldsymbol{N = 7066}$). This analysis showed the possibility that nearby neuronal sites could have more similar tuning maps, but the strength of the correlation depended on visual area and the placement of the array on the cortex. Related to Figure IV-2 and Figure IV-4 in *Results* section IV.2.3 *Areal differences of tuning landscapes*.

Chapter V : The Level Set and Invariances of Tuning Landscape

Abstract

Visual representations can be defined as the activations of neuronal populations in response to images. The activation of a neuron as a function over all image space has been described as a "tuning landscape". As a function over a high-dimensional space, what is the structure of this landscape? In this study, we characterize tuning landscapes through the lens of level sets and Morse theory. A recent study measured the in vivo two-dimensional tuning maps of neurons in different brain regions. Here, we developed a statistically reliable signature for these maps based on the change of topology in level sets. We found this topological signature changed progressively throughout the cortical hierarchy, with similar trends found for units in convolutional neural networks (CNNs). Further, we analyzed the geometry of level sets on the tuning landscapes of CNN units. We advanced the hypothesis that higherorder units can be locally regarded as isotropic radial basis functions, but not globally. This shows the power of level sets as a conceptual tool to understand neuronal activations over image space.

Keywords: level sets, invariance, visual neuroscience, natural image manifold, neural tuning, radial basis function, generative adversarial network, iso-response curve

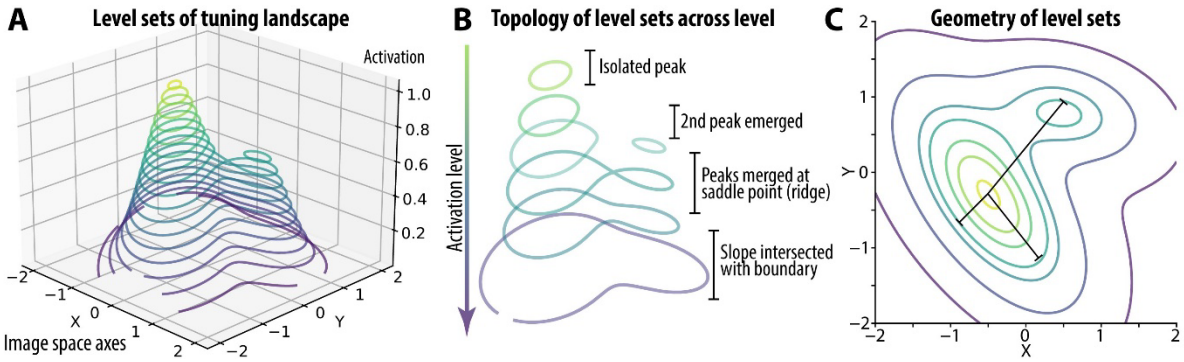


Figure V-1 **Conceptual schematics. A.** The level sets of a schematic landscape. **B.** One method for characterizing these sets: tracking topology through levels. **C.** A second method: analyzing the geometry of each level set within and across the connected components.

V.1 Introduction

The visual capabilities of humans and machines depend on the representations of their constituent neuronal units. To a first approximation, a visual representation is a set of feature functions over image space. For primate visual systems, these functions are instantiated by the firing rates of individual cortical neurons. For automated visual systems such as deep artificial neural networks (ANNs), these functions are instantiated by the activations of hidden units. The feature function is usually a continuous non-negative function of the image: for neurons, the mean firing rate changes continuously with the image; for *in silico* units, activation is a continuous function by design. Thus, we can picture the activation of a neuron or unit over the entire image space as a **tuning landscape** (Wang and Carlos R. Ponce, 2022d) **(Chapter IV**). If we do so, one can then ask: what are the image transformations that leave a unit's activations unchanged? In the landscape picture, this translates to finding the isoresponse *contours* or **level sets** of the activation function (**Figure** V-1 A). Transforming images within each level set will leave activations unchanged. This is a generalized definition of "invariance", the ability of a neuron to respond equally despite nuisance transformations (Ito *et al.*, 1995). At the population level, the intersection of individual neural level sets defines the set of metamers for a population

representation (Freeman and Simoncelli, 2011; Feather *et al.*, 2022). Thus, understanding the structure and visual content of level sets can shed light on important neuronal functional properties — a collection of level sets is similar to a computer tomography scan of the tuning landscape. Using tools from Morse Theory (Milnor, 2016), one can track the shape and topology of level sets as a function of the activation, revealing information such as the number of peaks, and the shape of each peak (**Figure** V-1B).

Here, we analyzed the tuning landscape of neurons and units through the lens of level sets. Using data from a novel experimental design (Wang and Carlos R. Ponce, 2022d), we obtained a collection of level sets on 2D tuning maps from neurons in visual cortex. By tracking how the level-set topology changed as a function of activation, we developed a topological signature for each tuning map. This signature was found to change systematically across the visual hierarchy, advancing previous results. We also examined images within the level sets to capture the feature attributes to which the unit was invariant. Finally, using level sets, we studied the high-dimensional tuning landscape of units in an ANN trained for object recognition. We found that higher-order units could be well-approximated by isotropic radial basis functions locally, while globally exhibiting multiple separated peaks. In contrast, the tuning-landscape peaks of mid-level units were much more anisotropic, and better connected by "mountain ridges".

V.2 Formalism and Mathematical Background

Let us consider the tuning function of a neuron or unit, mapping images to non-negative activation values[9] $f: I \to \mathbb{R}_+$. We assume f is a continuous function of the image. Since image

[9] For neurons, we will consider their mean firing rates in a given time window and disregard dynamics, *e.g.,* fluctuation and adaptation effects.

space I is compact [10], this continuous function has a global maximum and minimum value, according to the extreme value theorem (Rudin and others, 1976). The level set of a tuning function on general image space $\mathcal{I}$ is defined as the set of images evoking the activation level *c*, $\Omega_c = \{I \in \mathcal{I} \mid f(I) = c\}$. In this study, the image space is parametrized by a generative image model $G : \mathbb{R}^d \rightarrow I$ (Dosovitskiy and Brox, 2016b), so the level set lies on this *d* dimensional image manifold.

$$\Omega_c = \{I = G(z), z \in \mathbb{R}^d \mid f(G(z)) = c\} \quad (1)$$

When *c* is a *regular value* of the function *f*, the level set on a *d*-dim manifold is a $(d-1)$ dim hypersurface (per the implicit function theorem (Hatcher, 2000)). When *c* is a *critical value* (*e.g.,* the value of a peak or a saddle point), the level set may contain discrete points. From Morse theory (Audin, Damian and Erné, 2014), if an interval $[a, b]$ contains only regular values, then the level set Ω_a and Ω_b has the same topology (it is *homeomorphic*). Using this fact, when we find that the topology of a level set changes in an interval $[a, b]$, we can conclude there is a critical value (*e.g.,* a second tuning peak) lying in this interval (for examples, see **Figure** V-1 B). Thus, tracking the level set as a function of activation level can characterize the landscape.

For a high-dimensional image space, fully representing and characterizing a $dm-1$ dimensional hypersurface is intractable. So, we used two strategies to study these: in Sec. V.3, we consider a tuning function on a 2D sub-manifold in the image space; then, the level sets

[10] . When regarding an image as a red-green-blue (RGB) pixel array, the space of images is bounded.

become 1D curves—easier to study. In Sec. V.4, we sample discrete points from the high-dimensional level sets and estimate their properties (**Figure** V-1 C).

V.3 Level Sets of Neuronal Tuning *in vivo*

In this section, we study the tuning landscapes of visual cortex neurons using level sets. Because most datasets comprise arbitrary, irregularly sampled natural images, they do not facilitate neuronal response interpolation needed to calculate a level set. In contrast, generative image models allow for the dense sampling of visually similar stimuli and perfectly suited level set extraction — thus, we used these generative models for our experiments.

Experimental Design. The activity of visual neurons in V1, V4, and posterior inferior temporal (pIT) cortex were recorded using chronic microelectrode arrays (Microprobes for Life Sciences). A generative image model *G* with 4096 latent dimensions (Dosovitskiy and Brox, 2016b) parameterized a naturalistic image manifold. We used this manifold to characterize neuronal tuning functions and to find each site's tuning peaks. In each experimental session, one cortical site was selected as an experimental target (which could be a single neuron or multi-unit cluster), and then the neuron-guided image synthesis or *Evolution* approach (Ponce *et al.*, 2019a) was used to search for images that maximized firing rate (**Figure** V-2 A). After responses converged (Loshchilov, 2014; Wang and Carlos R. Ponce, 2022b), the evolutionary trajectory in the latent space was analyzed by principal component analysis (PCA). Then, a 2D hemisphere was created in the subspace spanned by the top three PC vectors of the trajectory (**Figure** V-2B). We denoted the coordinate map of hemisphere as $\phi: [-\pi/2, \pi/2] \times [-\pi/2, \pi/2] \rightarrow \mathbb{R}^{4096}, (\theta, \varphi) \mapsto z$. Latent codes were sampled regularly on the longitude-latitude (θ, φ) grid, and then mapped to images by *G*. This 2D hemisphere was designed to section through the peak and to visualize the

shape of tuning around optimal stimuli[11]. Finally, the sampled images were shown to the subject and neuronal responses r recorded. Each session led to a 2D tuning map, which could be understood as the composite function $f \circ G \circ \phi: (\theta, \varphi) \mapsto z \mapsto I \mapsto r$, mapping spherical coordinates (θ, φ) to neuronal activations r (firing rate averaged over [50,200]ms after stimulus onset).

Level Set Extraction. As shown in Wang and Ponce (2022c) , neurons from V1 to pIT showed smooth tuning maps on the image manifold (**Figure** V-2 C). Spherical spline interpolation was used to obtain a smooth function over the hemisphere (see appendix V.7.1 for details). We sampled $K = 21$ levels uniformly between the *min* and *max* values of the tuning map and extracted these level sets (**Figure** V-2 D,E).

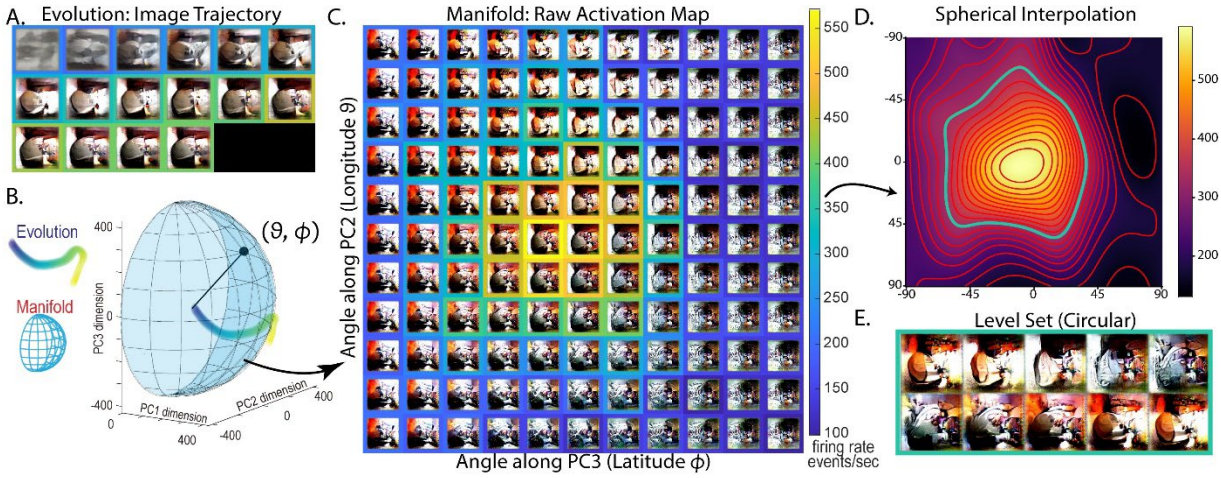


Figure V-2 **Example *Manifold* experiment (*in vivo*)** (Wang and Ponce, 2022c). **A.** Process of image optimization as driven by a V4 multiunit in monkey B (*Evolution* experiment). Image frame color denotes mean firing rate evoked per generation (same color scale as **C**). **B.** Latent space geometry of *Evolution* trajectory and *Manifold* sampling grid, plotted in the 3D subspace spanned by the PC1,2,3 vectors of the *Evolution* trajectory, adapted from Wang and Ponce (2022c, Figure 2). **C.** Raw neuronal response as a tuning map over the *Manifold* images. **D.** Spherical interpolation of tuning map and all level sets. **E.** One circular level set with corresponding image loop.

[11] This specific way of sampling was informed by the spherical geometry of the latent space of the generator G (Wang and Ponce, 2022a, Figure B.1, here Figure III-16). In this latent space, angular distance was a better correlate of perceptual distance than linear distance; changing the vector norm largely changes image contrast. So, we fixed the vector norm and sample vectors on a sphere.

V.3.1 Topological Signature of Level Sets on Tuning Maps

First, we examined the topology of each level set. Since each level set Ω_c is a 1D manifold, it has to be the union of segments that are topologically homeomorphic to either a circle S^1 or a line $(0,1)$ (Hatcher, 2000). Based on this fact, for each level set, we computed the number of segments (connected components) that are homeomorphic to circles N_S and to lines N_L. We tracked these numbers N_S, N_L as a function of the activation level c. We call this the *topological signature* of the level sets.

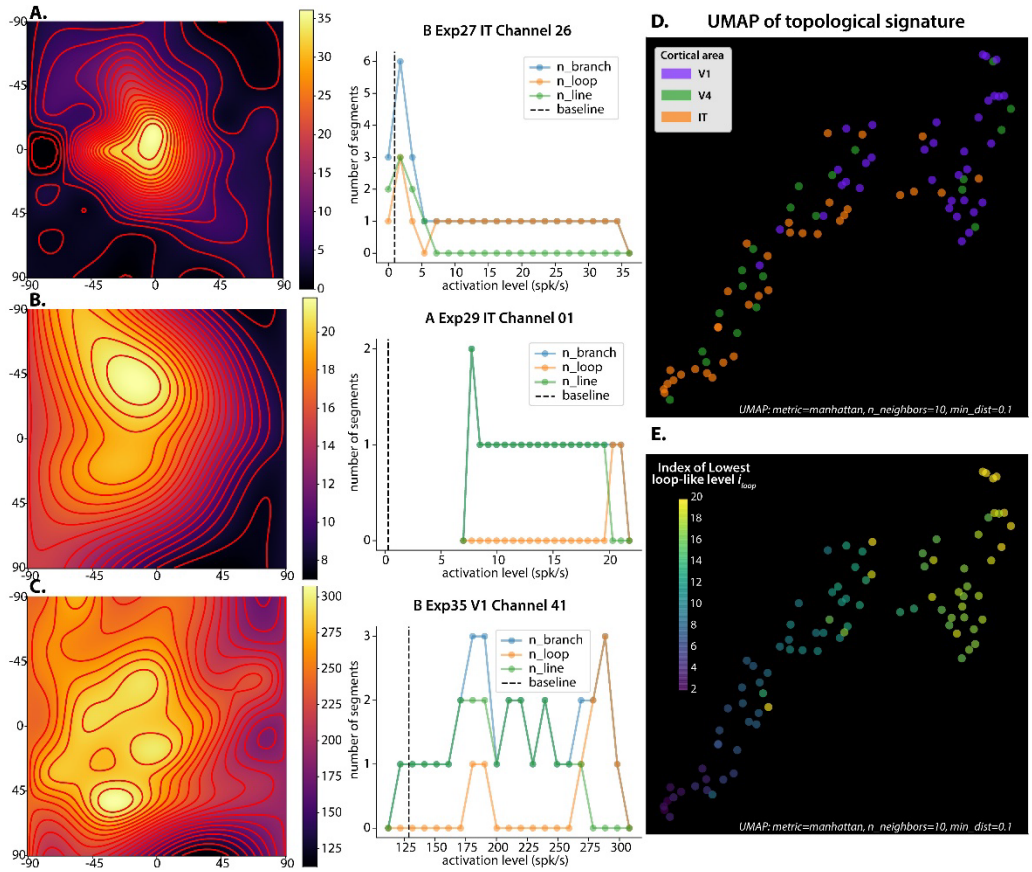


Figure V-3 Topological signatures of level sets in vivo. A-C. Example tuning maps and their topological signatures. (Left panel) K level set contours on the tuning manifold. (Right panel) Solid lines and dots denote the number of branches, loops and lines on each level as a function of neuronal activation level. A. A sharp single peak surrounded by many circle segments. B. A broader single peak, with more line segments. C. An irregular, multi-peak landscape, with multiple circular and line segments. D. UMAP plot of individual cortical sites (color indicates area of origin). E. Same plot with color coding of index iloop.

We found that generally, at a high activation level, the topology of the level set was a circle S^1. At lower activations, the level set became more elongated and fractured. We tracked the level set topology down from the peak and defined an index i_{loop} as the *lowest level* where the level set was still a single circle. This index quantifies the level where the level set changed its topology, either by intersecting the boundary or going through a second peak. Since all levels between i_{loop} and K are homeomorphic to a loop, we can infer that above the level i_{loop}, the landscape looks exactly like a bell (homeomorphic to a disk D^2).

To see how this topological signature translated to the landscape's geometry, we investigated a few examples in Figure V-3. A sharp isolated peak on the tuning map (*A*) was reflected by many level sets with a single circular component at a high activation level. A broader peak (*B*) was encoded by multiple line-like level sets, since the mountain slope intersected with the boundary of the hemisphere. A more fractured and multi-peak landscape (*C*) led to multiple connected components, even at high activation levels.

We then performed a population analysis using dimensionality reduction (UMAP) (McInnes *et al.*, 2018) on the topological signatures of all tuning maps ($n = 90$). We used the number of loops N_S and lines N_L at K levels as the feature and Manhattan distance as the metric. The UMAP analysis revealed a spectrum of topological features across all neurons (Figure V-3D). We found that the index i_{loop} strongly correlated with this spectrum (Spearman correlations with UMAP coordinate 1,2 were 0.923, $p = 3\times10^{-38}$ and 0.82,$p = 8\times10^{-23}$), providing one interpretation of this map. When we labeled the UMAP points by their visual area, we found that the progression of visual hierarchy (V1 to V4 to pIT) mapped onto this spectrum (Figure V-3E). The index i_{loop} decreased across the ventral hierarchy (Spearman correlation of i_{loop} with hierarchical level was -0.565, $p = 6.5 \times 10^{-9}$; one-way ANOVA of i_{loop} and cortical area, $F = 23.01$, $p = 9.1 \times 10^{-9}$).

Simply put, more IT neurons have tuning maps as in example *A*, while more V1 neurons have maps as in example *C*. In topological terms, a narrow isolated tuning peak resulted in more loop-like level sets and lower i_{loop}, while multi-peaks on a highland led to non-connected level sets at a high level, leading to a higher i_{loop}.

As a comparison, we performed the same Manifold experiments on units from a deep neural network (for details see V.7.4). The same level set analysis revealed a trend of topological signature change across the layers, similar to that across the ventral hierarchy (for *in silico* experiments, Spearman correlation of i_{loop} with the layer depth was -0.589, $p = 0, df = 9502$. For qualitative comparison of the mean topological signature, see **Figure** V-7 and **Figure** V-8).

V.3.2 Emergent invariance in level sets

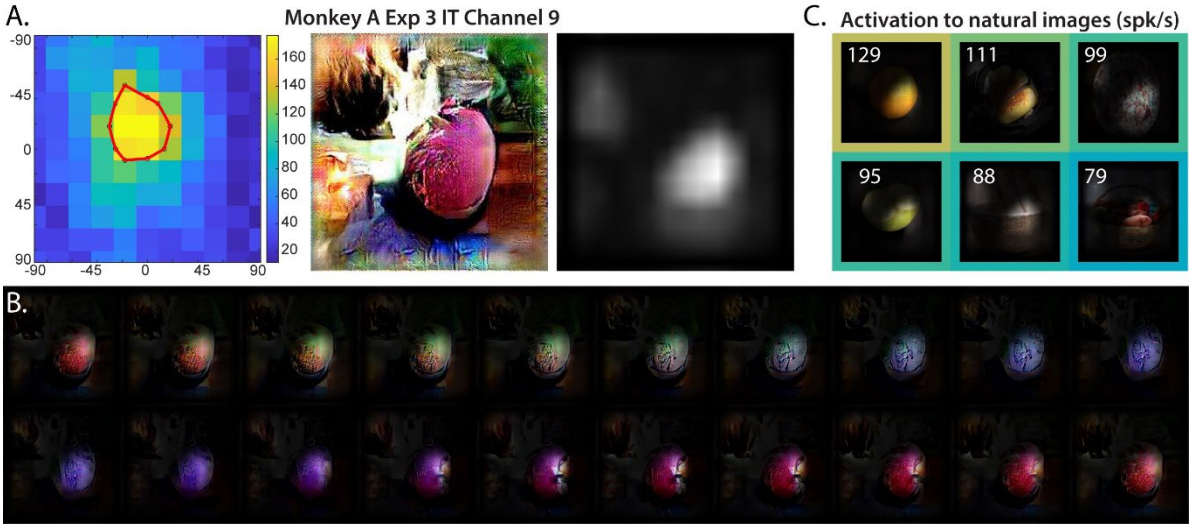


Figure V-4 **Example of an interpretable image transformation of one level set *in vivo*.** A circular level set that mapped to a color/hue circle in the image space. **A.** (**Left**) The circular level set on the tuning map of an IT neuron in monkey A. (**Middle**) The peak image on the manifold. (**Right**) The feature attribution mask highlighting the image region that correlated the most with the neural activation. For details see Sec.B.2. **B.** The image content of the level set, with the feature attribution mask emphasizing the region important to the neuron. **C.** Natural images that were also strongly activating. Firing rates are labeled at the corner and encoded in the color frame.

Next, we examined the visual content of each level set to study its associated feature invariance. We found that the image transformations of a subset of level sets were highly interpretable. For example, in one case, a pIT neuron guided the *Evolution* search towards a red-pink round object (**Figure** V-4 A). We extracted and visualized the level set at a high level (80% max). Intriguingly, the circular level set looping around the peak corresponded nearly perfectly to a *hue circle* with approximately the same spherical shape as the "object". Thus we could describe the image transform along this level set as "changing the color of the object by rotating along the hue circle". More formally, we can define a group action of a S^1 circle group transforming the image by traveling on the circle. We also examined the same neuron's responses to natural images in the same session (**Figure** V-4 C). Though their activation level was not as high as the peak image, they had different colors: an orange, a green apple, brown toast, etc. This suggests that this IT neuron was relatively invariant to the object's color. More generally, since circular level sets were common on the landscape, especially around tuning peaks, we conclude that it is relatively simple to create image transformations to which a neuron will be invariant. Crucially, we also noted that many level sets corresponded to image transformations that were hard to summarize in simple words (see **Figure V-9**). We will discuss the implications of this observation in the end.

V.4 Level Sets of Deep Neural Network Units

Next, we characterized the tuning landscape of deep neural network units on the same generative image manifold. Without *in vivo* experimental constraints, we were able to explore the full image manifold and characterize level sets more comprehensively. This could generate testable hypotheses for neural representations *in vivo*.

Methods. Here we used an optimization method to sample from level sets and characterize them using these samples. Since each level set Ω_c can have multiple connected components, we were interested in both local and global properties (i.e. structure of one component or across components). We analyzed *local* properties by first finding a local maximum, and then searching for level set images starting from that maximum. *Global* properties were analyzed by searching for level set images starting from random initializations. Pictorially, a local search ends at points near the connected component around a given peak; a global search ends with points lying on different "peaks" (**Figure V-5** A). By analyzing the distance structure among images within a component (*local*) or across components (*global*), we could characterize the geometry of the tuning landscape.

More concretely, we first performed an initial *Evolution* to obtain one activation maximizing image $I^* = G(z^*)$ and its code z^* as the base. Then, we searched for level set images using the objective Eq.2. The first term penalized the deviation of activation from the level c; the second term minimized or maximized the perceptual distance from a target I^*, depending on the sign of β. For a local characterization, the search was initialized from the peak z^*; for a global characterization, from a random vector. We used the perceptual image distance LPIPS (Zhang *et al.*, 2018) as D, and masked the images by their receptive field masks (methods detailed in appendix V.7.3). For each unit, we performed the search under three main conditions: local search, minimizing image distance ($\beta = 5$); local search, maximizing image distance ($\beta = -5$) ; global search, maximizing image distance ($\beta = -5$). We also performed the local and global search with $\beta = 0$ as baseline conditions.

$$\arg\min_z |f(G(z)) - c| + \beta D(G(z), I^*) \qquad (2)$$

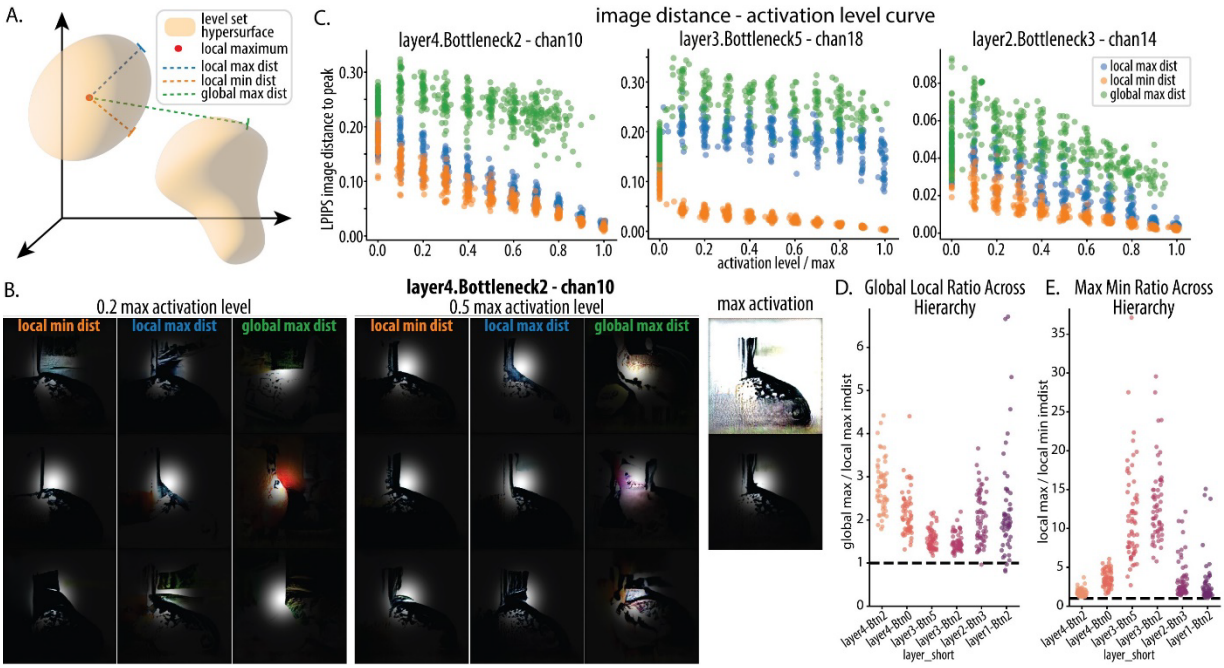


Figure V-5 **Characterizing level sets for *in silico* units. A.** Conceptual schematics showing level sets as hyper-surfaces and our strategy for characterizing them. **B.** Image samples from two level sets (0.2,0.5) for a unit in layer 4 (images masked by its receptive field). Each column shows three samples obtained under the same search condition (*local min, local max, global max image distance*). The base image (local maximum) is shown in the rightmost column. **C.** Examples of image distance - activation level curve, in three different layers. **D.** Ratio of global versus local max image distance in a level set, as a function of layer. **E.** Ratio of local max versus min image distance in a level set, as a function of layer.

Model network We chose the adversarially trained ResNet50-robust (He *et al.*, 2016; Engstrom *et al.*, 2019), as the *in silico* visual hierarchy to be compared with the ventral stream. This choice was made based on its high ranking on the Brain-Score leaderboard (Schrimpf *et al.*, 2018) at the time of this study. We sampled six layers and 60 units per layer; for each unit, we searched for 11 activation levels, with 5 different search conditions, and 40 repetitions, totaling 792,000 optimization runs.

V.4.1 Global Topology of Level Sets

First, we focused on the basic topological property of path-connectedness of the level set. We made the assumption that if a level set was path-connected, a gradient-based search within the set

could explore every point in it. So, we compared the level-set samples found by local versus global searches, both maximizing distances to the base image I^*. Specifically, we compared their image distances to the base image. The ratio of image distance between global and local search estimates the ratio of the distance between connected components and the "radius" of one component. Indeed, we found that the global search resulted in a larger diversity of level-set images, which were further apart from the initial peak I^* (**Figure** V-5 B,C). Throughout the hierarchy, the ratio of the mean image distance between the global and the local search was always greater than one (**Figure** V-5 D). Thus, we conclude that these level sets were generally *not path-connected*, and the tuning landscapes exhibited peaks at multiple locations which are not connected by a ridge. We also note that the global/local ratio for hierarchically higher units was larger, and the ratio for mid-hierarchical units was closer to one (`layer3.B2`, mean±std 1.53 ± 0.24, `layer3.B5`, 1.62 + 0.30*, N* = 60). This suggests that the landscape of mid-level units was "more connected" — meaning that when traveling along one level set component, the farthest distance one can reach was close to the farthest one can get globally. Pictorially, this means that *the peaks of mid-level units were connected* by "mountain ridges". In contrast, *the peaks of high-level units were more isolated*, and each local component was farther apart.

V.4.2 Local Geometry of the Level Sets and Tuning Peak Isotropy

Next, we examined the local geometry of the level sets. We computed the ratio between the *max* and *min* image distances D to the peak from one local component $\overline{\Omega}_c$.

$$\lambda_{maxmin}(c) = \frac{\min\limits_{I \in \overline{\Omega}_c} D(I,\ I^*)}{\max\limits_{I \in \overline{\Omega}_c} D(I,\ I^*)}$$

The max-min ratio λ_{maxmin} is closely related to the condition number of the Hessian matrix at the tuning peak, given a local quadratic approximation to the neural tuning function. The numerator points to the flattest direction to the level set, while the denominator points to the steepest direction from the peak. So, using this index we can quantify the perceptual anisotropy of the tuning peak.

We found that for mid-layer units, the ratio was large, *i.e.,* there was a large gap between the minimum and maximum image changes that induced a certain level of activation decrease (**Figure** V-5 CE). In contrast, for the higher order visual representations (especially the penultimate layer in ResNet50-robust), the ratio was surprisingly close to 1 (mean±std, 1.77 ± 0.46*,* $N = 60$) — so even the "fastest" descent and "slowest" descent path from the peak had a similar slope. This highlights the isotropic nature of the peak.

In summary, we can interpret these results in the context of the radial basis function (Poggio and Girosi, 1990b). It has been proposed that in a multidimensional space, the tuning of neurons could be viewed locally as a radial basis function (Wang and Carlos R. Ponce, 2022d). Here, we validated that the tuning landscapes of higher-order visual units can be locally approximated well by isotropic radial basis functions: any direction deviating from the peak has a similar slope. However, this approximation does not hold globally, as images sampled from other components of the level set did not fall on the same trend **Figure** V-5. This suggests we should not limit global characterization of tuning to just one radial basis function, and certainly not to readily interpretable tuning variables (Wang and Carlos R. Ponce, 2022d). In contrast, for mid-level units, there exist some very flat and also very steep paths deviating from the peak, illustrated by an elongated level set with large aspect ratio. Thus even locally, their tuning landscapes are anisotropic and exhibit certain "untuned" directions. This result is consistent with the model

proposed in Wang and Ponce (2022c, Figure IV-6 in Chapter IV), where higher visual neurons have more tuned axes than the middle ones and appear to be more isotropic.

V.5 Discussion

In machine learning, the concept of level sets has been a classic tool for representing and analyzing shapes (Osher, Fedkiw and Piechor, 2004). In neuroscience, it has been used under the name of "iso-response method" to understand the computation that integrates two or three features in *early* sensory neurons (e.g. locust auditory receptor, retina ganglion cells, V1 neurons, (Gollisch *et al.*, 2002; Rust *et al.*, 2005; Gollisch and Herz, 2012; Horwitz and Hass, 2012)). In this work, we extended this tradition and applied this concept to analyze the geometry of tuning landscapes of neurons throughout the visual hierarchy. Thanks to the advance of image generative models and closed-loop experiment techniques, we are now able to obtain level sets both for higher visual neurons and in the high-dimensional space of complex natural images. As we showed, the geometry of level sets exhibited trends across the visual hierarchy that were consistent *in vivo* and *in silico* (**Figure** V-7 and **Figure** V-8). This adds another quantitative tool for comparing representations in brains and machines, in addition to representation similarity analyses (Kriegeskorte and Wei, 2021).

The next stage for visual neuroscience is to define how neurons respond across naturalistic image transformations, specifically characterizing their selectivity and invariance. When considering neuronal responses across a large image manifold (via the *tuning landscape* perspective), it appears trivial to define a group action (i.e. image transformation) of a S_1 circle group on a local image domain, to which the neuron activation is "invariant." The group action is to transform the image by rotating on the circular level set. More generally, the level set of a function on a d dimensional manifold is a $d - 1$ dim hypersurface. We hypothesize that if a

neuron is tuned to d dimensions, one should find a $d - 1$ dimensional hypersphere-like level set around its local maximum, where the neuron will be invariant to any transformation on the hypersphere. This may have implications for the development of equivariant neural networks.

There are compelling examples where the level sets on the tuning landscape contained interpretable image transformations (*e.g.* color of the object), but more often than not, there are no easy labels for the transformations characterized by the set (Figure V-9). This is likely because neurons/units just learn useful features, and signal distances to these features, regardless of interpretability. By being tuned to multiple features, the diminishing presence of one feature can be compensated by the persistence of another. Thus, nameable image transformations (*e.g.,* rotation) may be a small fraction of all the image transformations to which a unit or neuron is invariant.

Acknowledgments

We thank Mary Carter for help in monkey care and data collection, Kaining Zhang for helpful discussion on analysis and algebraic topology, and Matthew Farrell for his talk that inspired this work.

V.6 Appendix A. Related Work

Iso-response Set of Tuning Functions There is a rich history of analyzing the level sets of tuning functions for biological neurons and artificial units. Much of this literature can be found under the terms of *iso-response sets*. For *in silico* units, (Paiton *et al.*, 2020) analyzed the iso-response surfaces of units in a sparse coding network, and found that one direction of adversarial perturbations was orthogonal to the surface. This makes sense since this was the "steepest" tuning direction. For biological neurons, (Rust *et al.*, 2005, pp. 2--) (Figure 5) analyzed the iso-

response contours of primate V1 neurons on the 2D plane spanned by the activities of two linear filters (subunits). By inspecting the shape of the ellipsoidal contours and their alignment with the axes, the authors inferred the form of nonlinear operations combining the linear filters. (Horwitz and Hass, 2012) used a closed-loop experiment to find the iso-response surface for V1 neurons in the 3d color space spanned by cone-photoreceptor activity space. By analyzing the geometry of these 2D surfaces (*e.g.*, some being planar, others ellipsoidal or cup-shaped), they identified different linear or nonlinear computations used by these V1 neurons used to integrate cone input signals. (Gollisch and Herz, 2012) reviewed experimental results on the iso-response curve: the locust auditory receptors, with respect to the intensity of two pure tones in superposition (Gollisch *et al.*, 2002); retina ganglion cells, with respect to the contrast levels of two patches; and V1 neurons with respect to colors (Horwitz and Hass, 2012). They proposed that by analyzing the geometry of the "iso-response" curve on the plane spanned by two stimulus features, one could disambiguate different feature integration mechanisms. They also suggested that closed-loop optimization was crucial to estimate these level sets. These previous works focused on early sensory neurons or receptors and analyzed contours in simple stimulus spaces. Here, we extended this approach to neurons in higher visual cortices (V4 and pIT) within complex naturalistic image manifolds. This advance was made possible by our closed-loop experimental approach that could find neurons' "preferred" stimuli and by the use of an image generator that could manipulate images smoothly.

Shifting to studies of perception, (Zetzsche, Krieger and Wegmann, 1999) found the set of stimuli that had a *just noticeable distance* (JND) from a given Gabor patch on a 2D image space. By comparing the shape of the contour formed by these stimuli under different parametrizations ((Zetzsche, Krieger and Wegmann, 1999), Fig.5,6), the authors found that certain

parametrization schemes (such as polar coordinates) were more fitting to perception because they led to more isotropic, circular contours. This argument can also apply to the more isotropic tuning peaks for higher visual units in our study. It suggests the latent space parametrization of images is more "natural" for higher visual neurons like IT.

Level Sets in Machine Learning A level set is a classic concept in multivariate calculus and geometry. In machine learning, it is a popular tool for representing $d - 1$ dimensional hyper-surface data with a scalar function on *d* dimensional spaces (Osher, Fedkiw and Piechor, 2004). The level set method is also a classic approach for image segmentation, which represents the boundary of an object by the level sets of a function on the 2D plane. More recently, the neural implicit function method represents 2D shapes and surfaces by the level sets of a 3D volumetric function parametrized by a neural network (Chen and Zhang, 2019). In many cases, the function learned by the network is the *signed distance function* (SDF) to the object surface (Park *et al.*, 2019). In these works, the level set is a tool to learn and represent a target signal. In contrast, in our work, we used level sets to analyze representations already learned in biological and computational visual systems.

Object Manifold in Neural Representation One similar line of work is the one studying the geometry of object manifolds (Chung, Lee and Sompolinsky, 2018; Cohen *et al.*, 2020). Our conceptual pictures complement each other: they consider the manifold in the neural activation space, traced by the neural activations to the different views of the same object. We considered the level set manifolds in the image space, traced by the different images that evoked comparable neuronal responses.

V.7 Appendix B. Method details

V.7.1 B.1. Details for Spherical Interpolation

In Wang and Ponce (2022c), images and associated responses were organized on a regular grid of spherical coordinates θ,φ. Thus, the spherical geometry needed to be taken into account when interpolating the responses. We used the RectSphereBivariateSpline method in SciPy to interpolate and smooth neuronal response on the hemisphere. We realized that neuronal response noise would affect all downstream analyses, so we set the smoothing parameter s as the sum of the squared standard error of the mean (sem) of response over the tuning map. This balanced the reconstruction error and the smoothness of the map.

For contour extraction, we used the marching squares algorithm (Lorensen and Cline, 1987) implemented in skimage.measure.find_contours function for each contour line. Finally, we re-parametrized each contour line with arc length parametrization and sampled the curve uniformly. Namely, the latent codes were sampled with equal angular distance along the curve. The resulting level sets were shown in **Figure** V-2 D,E.

V.7.2 B.2. Feature Attribution Mask

For the *in vivo* level set analysis, we masked the image with a "feature attribution mask", which can be estimated using a recently developed method Wang and Ponce (2022b). Conceptually, this is a data-driven way to find the image region and pattern that correlate the most with the neuronal responses. Briefly, this method took in all the image and response pairs in the Evolution trajectory and correlated every unit activation in a convolutional neural network with the neuronal response. Then the covariance tensor was factorized into the combination of a few rank-1 factors, each one being the product of a spatial mask and a feature vector. Finally, we fit a simple linear model to determine the relative weights of the

factors. We used these weights to combine the spatial masks and obtained the alpha mask on the image as in **Figure V-4 and 9**.

V.7.3 B.3. Receptive Field Estimation

For the *in silico* level set analysis, the image was masked by a "receptive field mask". Simply, we estimated the receptive field by computing the gradient from the activation to the image pixels $\nabla_I f(I)$, then we repeat this by sampling white noise images 200 times and averaging these gradient maps

$$\mathbb{E}_{I\sim unif[0,1]} \nabla_I f(I)$$

This average gradient map is normalized and smoothed to be the receptive field mask, as in Figure IV-5, 10, 11, 12 and 13

V.7.4 B.4. Manifold Experiment *in silico*

As in Wang and Ponce (2022c), we conducted Evolution and Manifold experiments on units sampled along the ResNet50-robust. We sampled units from the center of feature maps of every channel in the CNN, totaling 38012 units from 10 layers (relu, layer1.B1, layer2.B0, layer2.B2, layer3.B0, layer3.B2, layer3.B4, layer4.B0, layer4.B2, fc , B is the abbreviation for Bottleneck layer). In each experiment, the receptive field of the unit was measured; the image was resized to fit the receptive field of the unit. Then the same *Evolution* and *Manifold* experiments were applied to the unit to obtain its activation maximizing image and the 2D tuning maps around the image. Then the tuning maps were analyzed by the same level set topology analysis as in Sec. V.3.

V.8 Appendix C. Extended Results

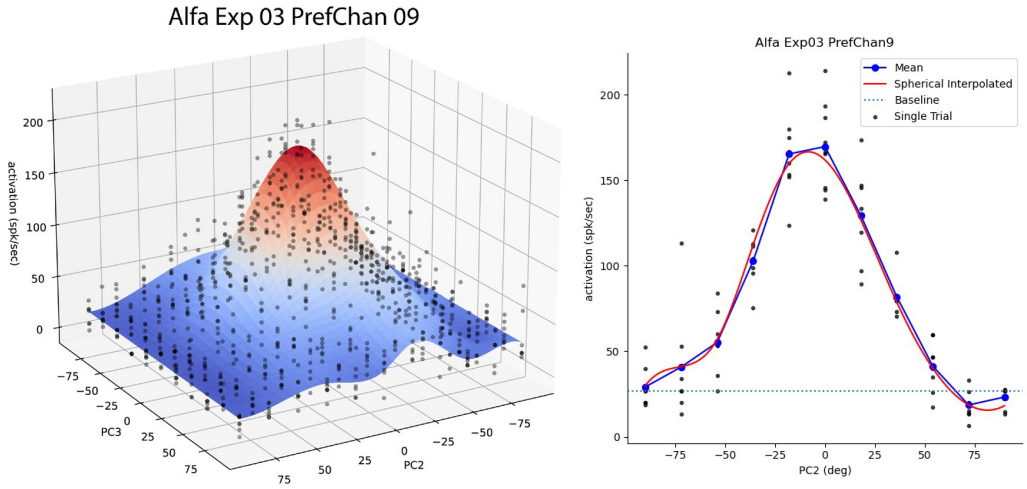


Figure V-6 **Neuronal fluctuation *in vivo***. **Left**. The surface plot shows the spherical interpolation of a tuning map *in vivo* (a neuron in IT of monkey A). 3d scatter shows the single trial neuronal responses to each image. **Right**. 1d section through the 2D tuning map, showing the single trial firing rates (scatter), mean response (blue), and the spherical interpolated activation curve (red). This motivates us to use the smoother interpolated curve when computing the level sets.

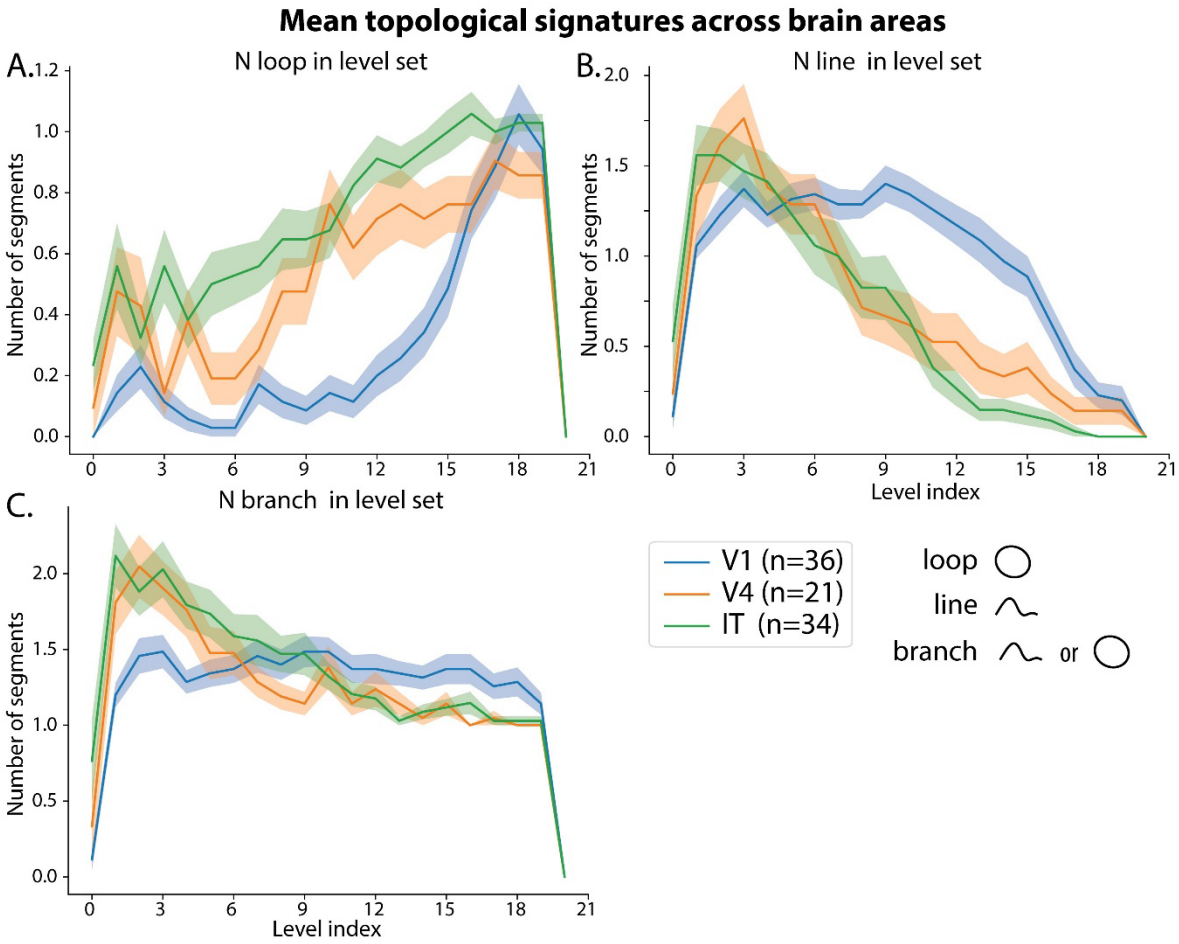


Figure V-7 **Mean topological signature per visual area *in vivo*.** Each curve shows the mean±sem of number of loops (**A**), lines (**B**), and branches i.e. connected components (**C**) as a function of level index; each color corresponds to a given visual area (V1, V4, or IT).

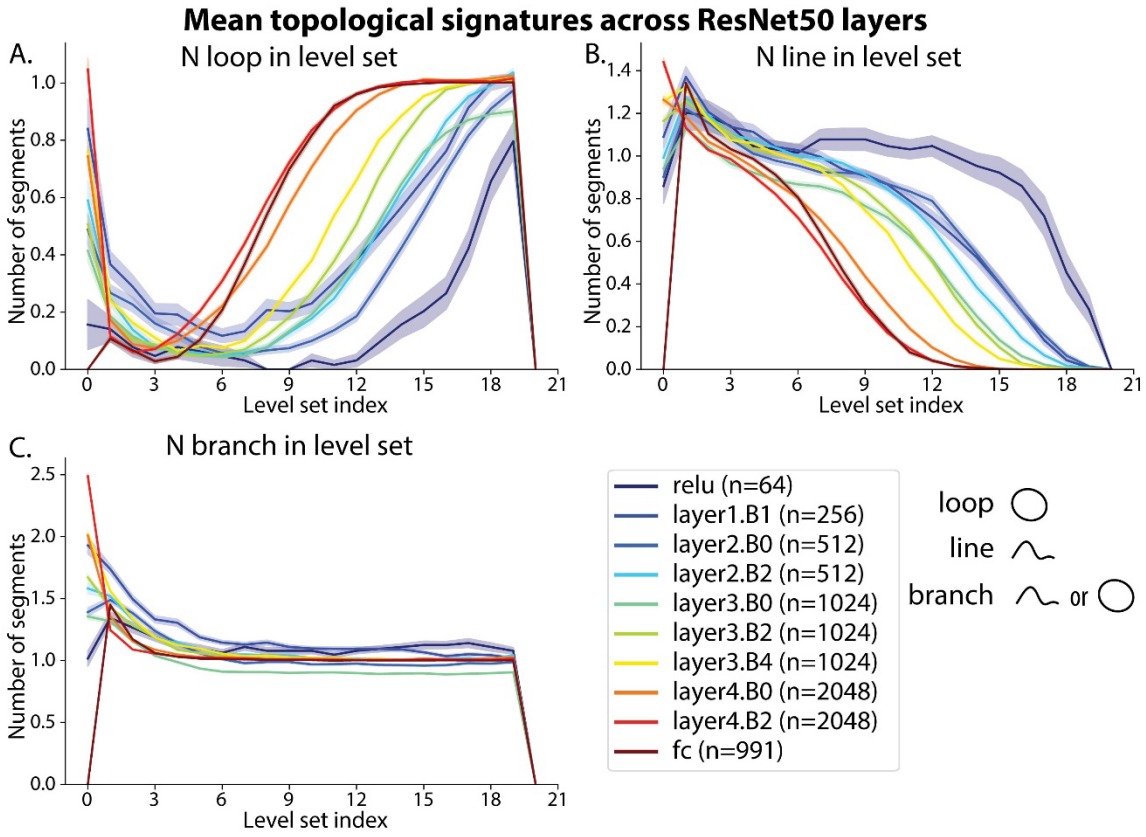


Figure V-8 **Mean topological signature per layer in network ResNet50-robust.** Each curve shows the mean±sem number of loops (**A**), lines (**B**), and branches i.e. connected components (**C**) as a function of level index; each color corresponds to a given layer.

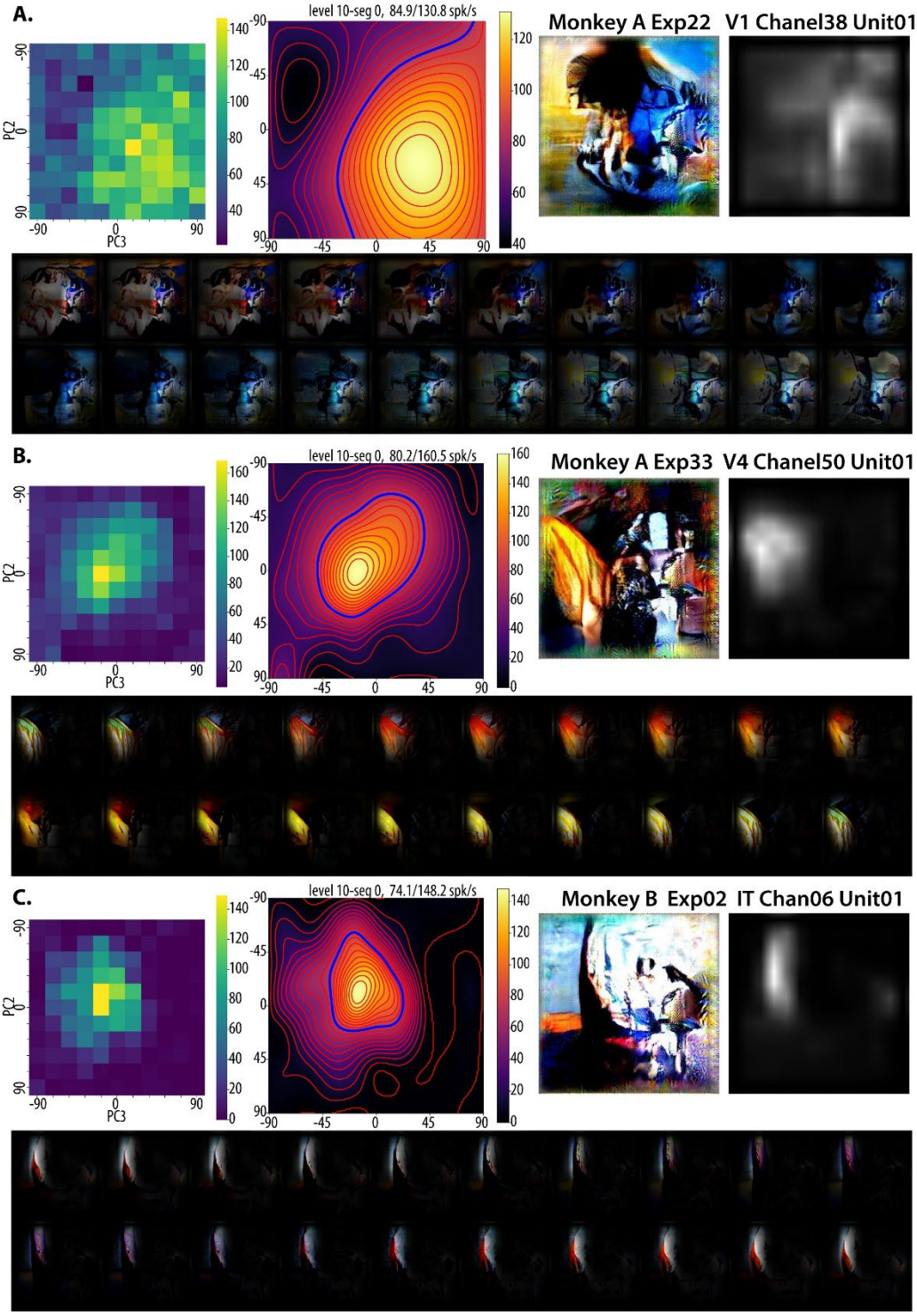


Figure V-9 Examples of level sets lacking readily interpretable image transformations. Examples from **A.** V1, **B.** V4, and **C.** IT are showed in the three panels. Figure layout in each panel is similar to Figure 4.

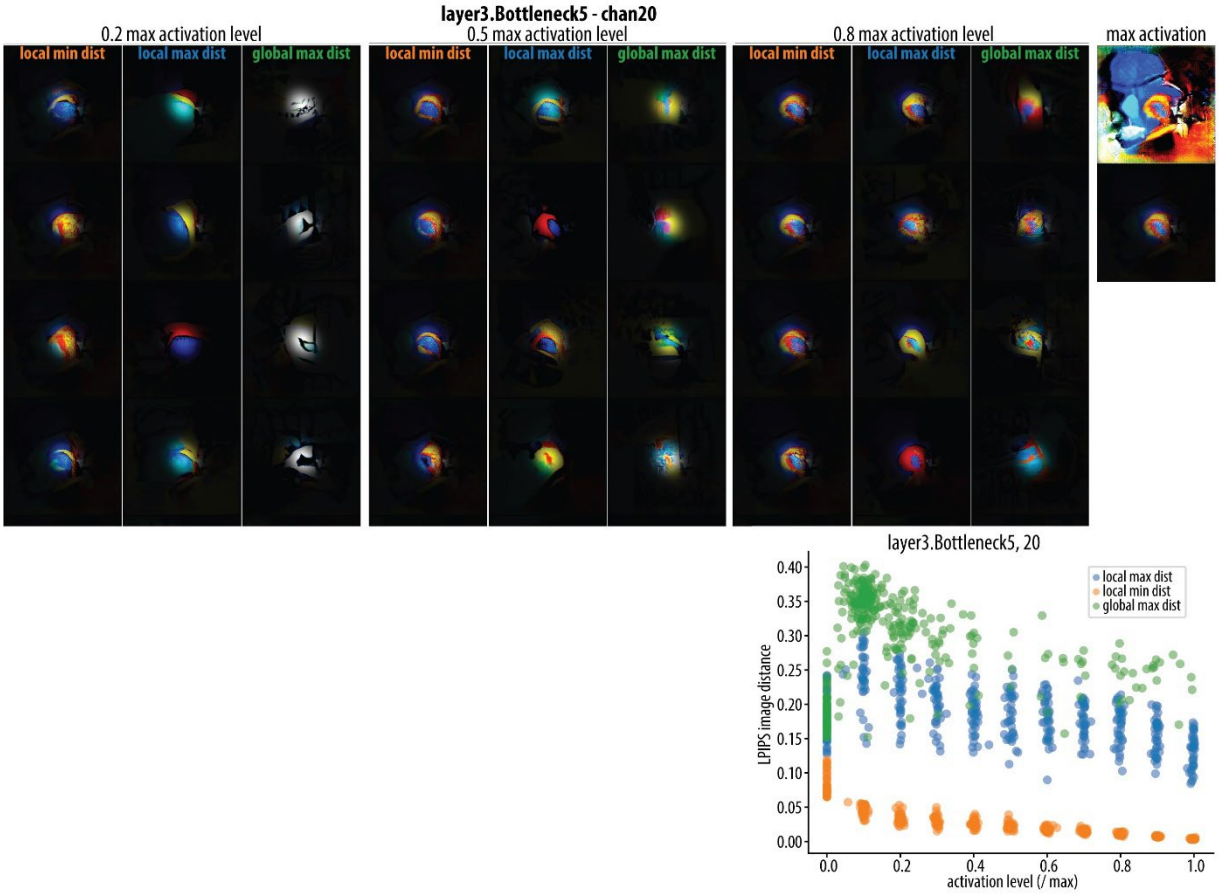


Figure V-10 ***in silico* Level set image samples for an mid-level unit in layer 3**. **Upper** Level set images for three levels are shown (0.2,0.5,0.8 max activation level). Samples obtained with the same objective are shown in a column. Three objectives were *local min*, *local max* and *global max* image distance search. **Lower** Summary of all level set images for this unit. The large gap between the local min and local max distance highlights the anisotropy of the peak. The smaller gap between the local max and global max signifies the different connected components of the level set are not too far away. For examples with another organization, see Figure 11, Figure 12, Figure 13.

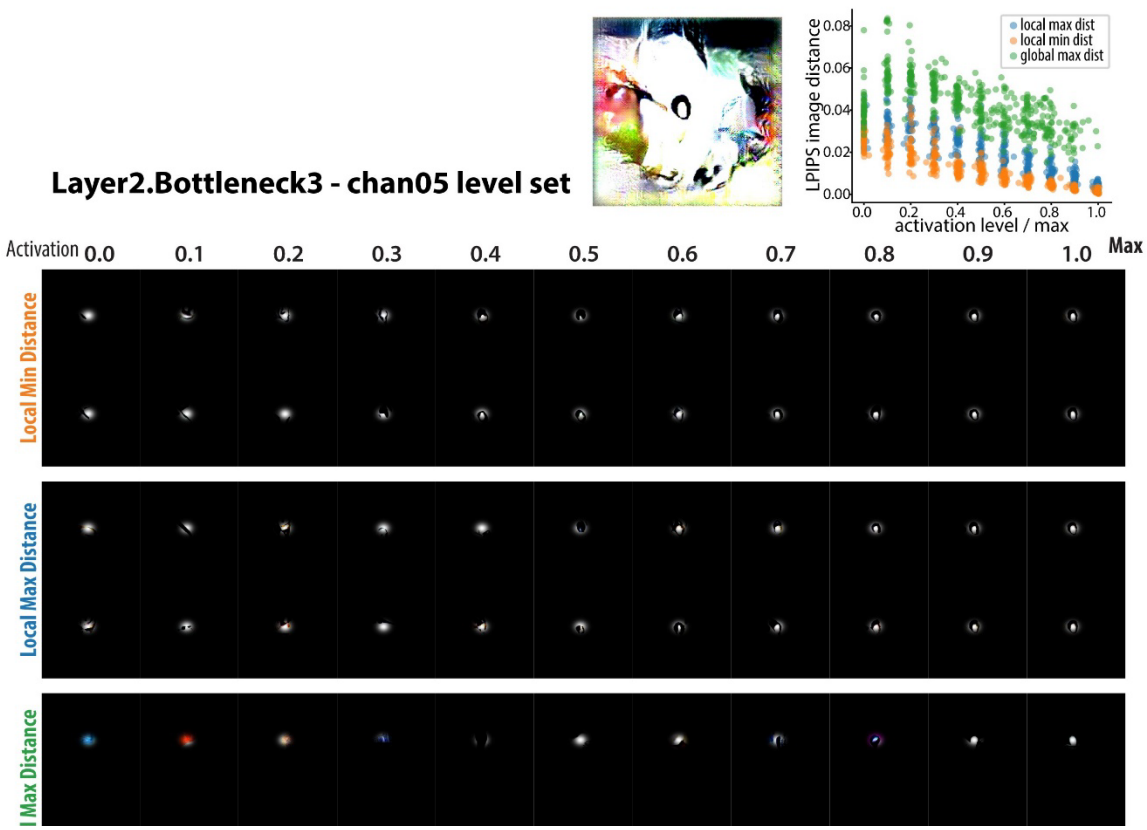


Figure V-11 ***in silico*** **Level set image samples across activation levels of a unit in Layer2-B3**. **Upper left** Initial peak, base image I^*. **Upper right** Summary of all level set images for this unit. **Lower**, Each block collect the level set images obtained in one optimization condition: from top to bottom, *local min distance, local max distance, global max distance*. Column from left to right shows level set images for 0.0 to 1.0 max activation level, two images per level per condition.

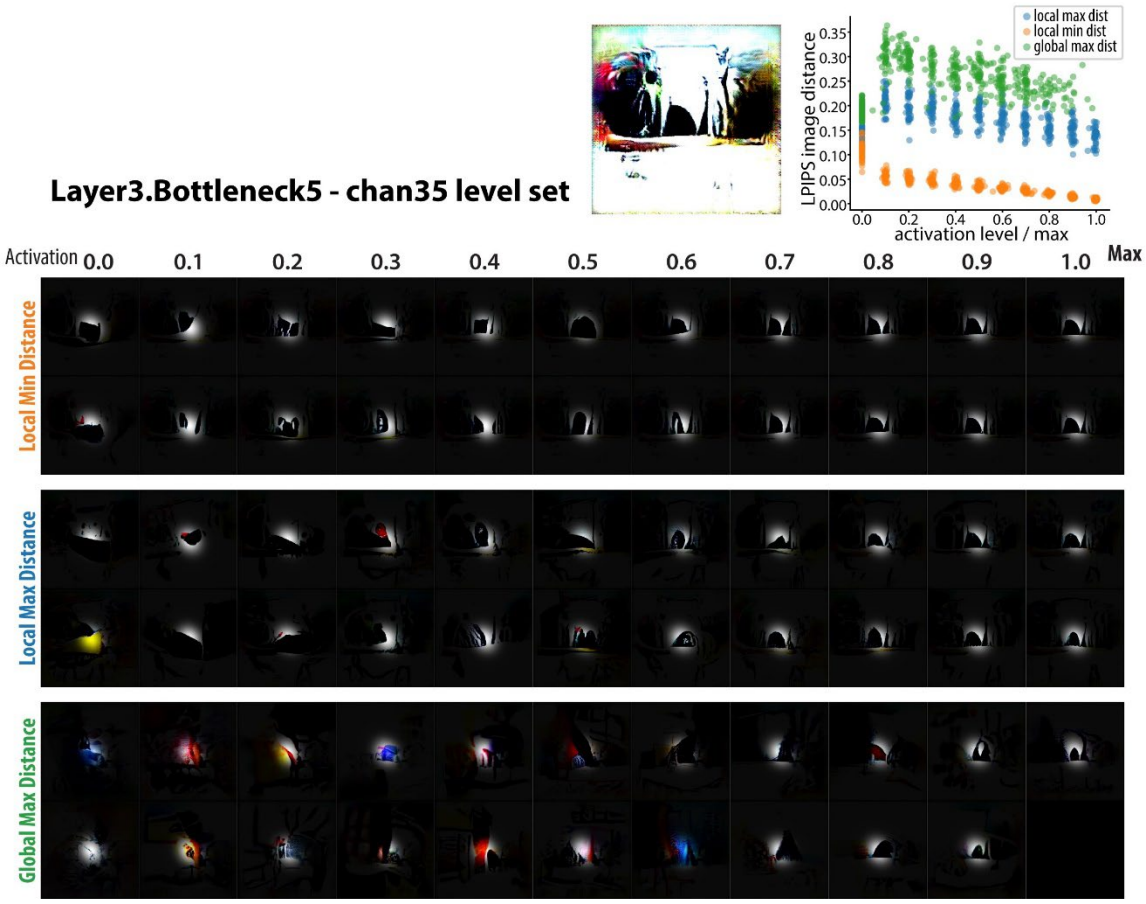


Figure V-12 ***in silico*** **level-set image samples across activation levels of a unit in Layer3-B5.** Same organization as in previous figure.

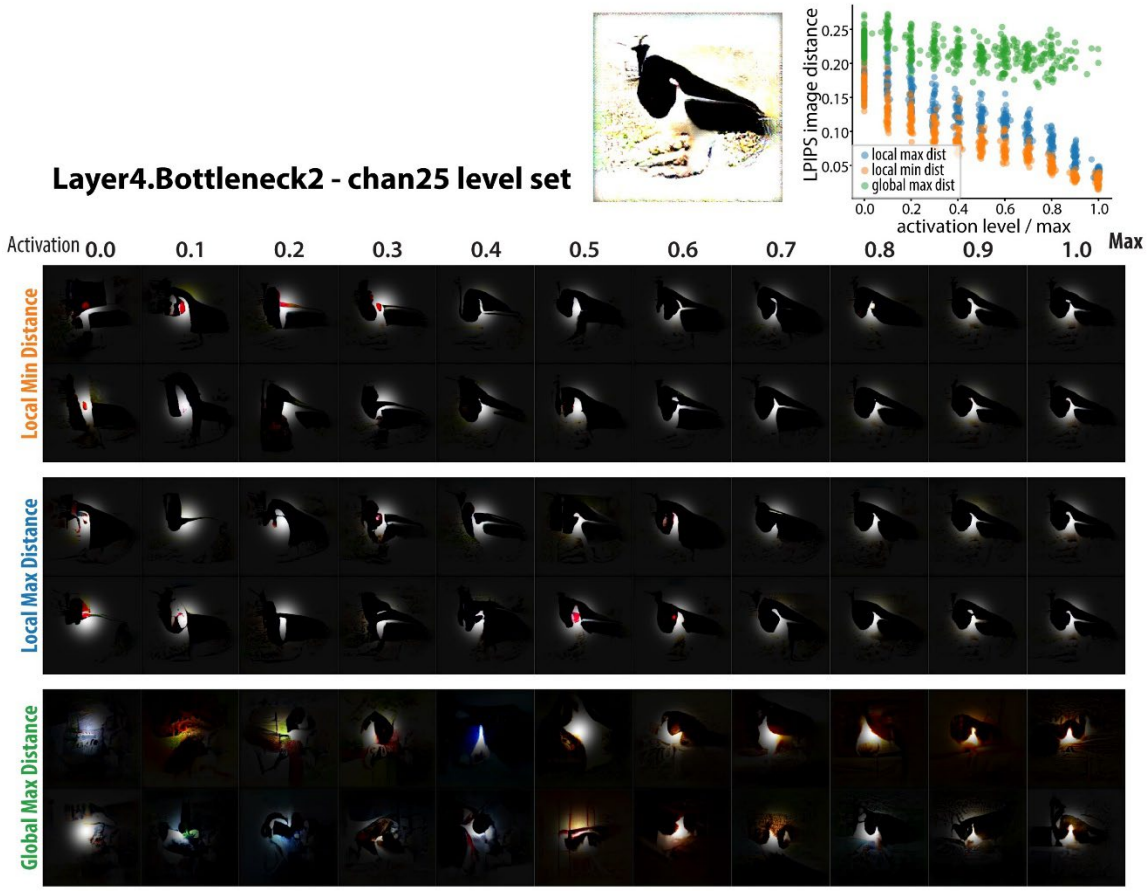


Figure V-13 ***n silico*** **level-set image samples across activation levels of a unit in Layer4-B2**. Same organization as in previous figure.

Chapter VI The Alignment of Ventral Stream Tuning with Multiple Image Manifolds

Abstract

The neural basis of our object sense is an important question for both biological and artificial visual systems. To study it in the visual cortex (including V1, V4 and pIT), we set out to optimize images for neurons in all these regions, applying a closed-loop evolutionary algorithm. Crucially, we used two generative image models: (1) DeePSim, which parametrizes irregular image patterns, and (2) BigGAN which parametrizes object identity and nuisance variables. We found that neurons could guide image optimization on both pattern- and object-based image manifolds, in two monkeys. Image optimization in DeePSim successfully increased the neuronal firing rate in nearly all experiments for both V1 and V4. However, when it came to BigGAN, it failed consistently for V1; while for V4, it succeeded in about 50% of experiments. Interestingly, in IT, the success rate was around 60% for both image spaces. This suggests that neuronal tuning along the ventral stream became more aligned with object-based and less aligned with pattern-based parametrization. In V1 and V4, optimized DeePSim images evoked higher firing rates than BigGAN; in contrast, in IT, the optimized images from both spaces evoked comparable rates. Intriguingly, in convolutional neural networks (CNNs), optimized DeePSim images were more activating than BigGAN across the hierarchy. This suggests that CNN units were less object-aligned than IT neurons, corroborating the finding that CNN models are more texture-biased. Next, by analyzing neuronal dynamics, we found that IT response preference for BigGAN images emerged later in time, suggesting that "object

preference" in IT required recurrent processing. To evaluate the images, we found that the V4 neurons guided the BigGAN images to become less object-like relative to baseline, while guidance from IT neurons led the BigGAN images to have higher objectness score than baseline. In contrast, guidance from both V4 and IT made DeePSim images more object-like. Visually, we could identify similar features synthesized in both spaces, and we found the feature similarity of the optimized images correlated more with the similarities of the response dynamic in two spaces rather than mean response. This suggests that the dynamics of the neuronal response encode critical sensory information. All in all, our results point to a dynamic preference for objects that gradually evolves over time in the higher visual cortex, e.g., IT, while still retaining robust responses across pattern-generating space. These results also highlight a gap of object alignment between the current computational models and biological visual system, which may be addressable by recurrent processing.

VI.1 Introduction

From the dawn of visual neuroscience, highly activating images have been used to highlight the feature preference of visual neurons. Many classic works manually design simplistic and parameterized image spaces and characterize the neuronal response patterns in those spaces (Pasupathy and Connor, 1999). The recent advances in generative image models enabled us to compactly parameterize naturalistic images, opening up possibilities to study neuronal selectivity in a more automated fashion. Using evolutionary algorithms and generative image models, researchers have successfully synthesized highly activating images for neurons across the ventral stream in primate (Ponce *et al.*, 2019b), and found intriguing trends across the visual hierarchy (Rose, Johnson, Wang and Carlos R Ponce, 2021; Wang and Carlos R. Ponce, 2022d). However, limited by the search algorithm (Xiao and Kreiman, 2020), previous works focused on one

specific generative image model, a generative adversarial network (GAN) called DeepSim (Dosovitskiy and Brox, 2016b). As generative image models have progressed rapidly in the last seven years, this has raised questions about the generality of the method and whether the results tell us more about the generative network or about the visual neurons.

Recently, with a more detailed understanding of the geometry of the generator latent space (Wang and Ponce, 2020) and an improved evolutionary algorithm (Wang and Carlos R. Ponce, 2022c), we have unlocked the ability to apply the neural-guided image synthesis paradigm to many new image generators. This technical advance enabled us to compare different generative image manifolds as a factor that influences the *Evolution* process. This holds the potential to reveal new insights into classic questions in neuroscience: i.e. the nature of representation on the visual cortex. More specifically, this question asks about the alignment between the neural representation and image manifolds.

Let's imagine, a wealthy patron hired two artists of different styles to create their best painting for them. The patron and the artists work in an interactive fashion, each time the artists produce some potentially interesting paintings, and the patron will give feedback to the artists by telling them which ones they like more. Both artists will try their best to adjust according to the feedback to make the *patron* happy. But those artists with an intrinsic similar "artistic taste" to the patron will succeed easier, while others will have a harder time figuring out what the patron likes. Specifically, the artist with an aligned taste can start with a better sketch, and not many changes are needed to achieve a pleasing outcome. On the flip side, the misalignment between the taste of the patron and the generative power of the artists makes it hard to adjust the artwork according to their feedback.

In this work, we let the visual neurons from the primate ventral stream be our patron and two different generative image models be our artists, this enables us to test their differential alignment with stages of ventral stream processing. Specifically, we compared neuron-guided image synthesis in two GANs back-to-back: the classic DeePSim GAN and the modern BigGAN (Brock, Donahue and Simonyan, 2018). DeePSim GAN has served as a general natural image prior to vision research (Ponce *et al.*, 2019a; Rose, Johnson, Wang and Carlos R. Ponce, 2021; Wang and Carlos R. Ponce, 2022d), but it does not generate whole objects in the image by default, and random samples from it look like compositions of shape and textures similar to abstract painting. In contrast, BigGAN is trained to conditionally generate object images in ImageNet (Deng *et al.*, 2009) with nuisance variation, and random samples from BigGAN will be an image centered on objects with a photorealistic style (**Figure** VI-1). We compared the success rate of Evolution, the optimized activation, the optimization dynamics, the neural dynamics and the optimized images in the two spaces to shed light on how the alignment of neuronal tuning and image manifold change across visual cortices and change through time.

VI.2 Results

VI.2.1 Develop optimizers for BigGAN object image manifold

Developed by DeepMind in 2018, BigGAN enjoys a much more complex network structure, which includes self-attention Zhang et al. (2019) and modulated batch normalization Chen et al. (2018). Different from DeepSim, BigGAN is a class-conditional GAN (**Figure** VI-1B), which receives a class conditioning vector $c \in \mathrm{C}$ and a noise vector $z \in \mathrm{Z}$. Each of the 1000 object classes in ImageNet is represented by a different class vector c; while the variations within each class are controlled by the noise vector z which is sampled from a truncated Gaussian distribution during training. Thus, smoothly traveling in the class embedding space C will interpolate between

the categories of objects while traveling in the noise space Z will change nuisance variables like aspect ratios, orientation, viewing angles, etc Wang and Ponce (2020) (**Figure** VI-1C). Given its different latent space structure, how to efficiently search in this latent space via *Evolution* paradigm became our first challenge.

Developing closed-loop optimizers for the stochastic visual neurons *in vivo* is timeconsuming and costly. However, in the field of evolutionary computing, it's a common strategy to configure optimizers on synthetic problems with similar landscape characteristics, and then deploy them into the real world (Long *et al.*, 2022; Wang and Carlos R. Ponce, 2022c). In visual neuroscience, convolutional neural networks (CNN) have been widely used as models of primate visual hierarchy (Yamins *et al.*, 2014). Further, the tuning landscapes of the hidden units have been shown to share many similar properties of the ventral stream neurons (Wang and Carlos R. Ponce, 2022d). Thus, we first optimized the configurations of evolutionary algorithms for BigGAN on the hidden units of CNN *in silico*, and then tested them on real neurons *in vivo*.

We found a successful configuration of Evolutionary Strategy algorithms that work for BigGAN *in silico* and showed that it also works *in vivo*, (Figure VI-2 A). We noted that the key parameter for success is the *standard deviation of the sampling distribution* or the *step size of exploration*. This parameter controls the distance of exploration in the latent space, which translates to the distances between samples in a batch of images. If the images within a batch were too similar to each other, the difference in neuronal response to images will be overwhelmed by single-trial response variability, obscuring the "gradient" direction of the tuning landscape. On the other extreme, if the images within a batch are too far apart from each other, pictorially, they may locate on different peaks on the tuning landscape, then traveling in the

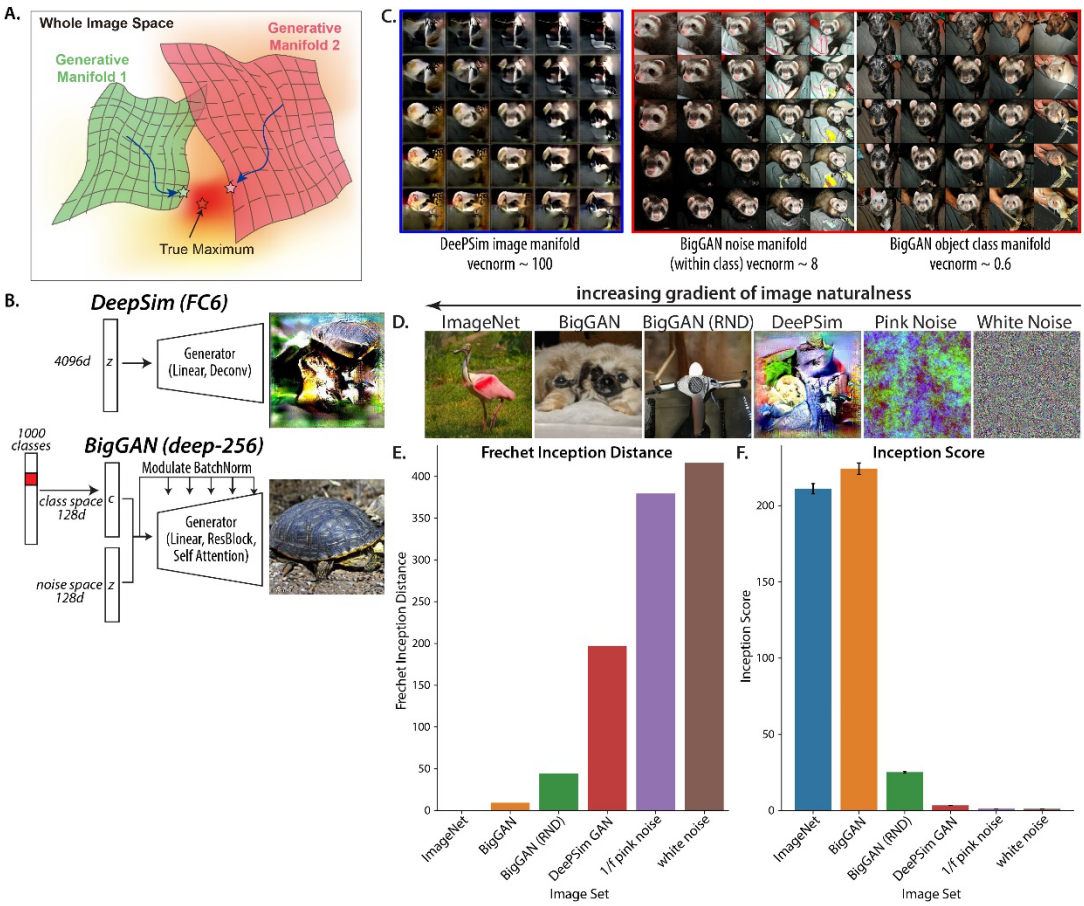


Figure VI-1 **Different image manifolds and their parametrization by GANs (BigGAN and DeePSim GAN). A.** A schematic of different GANs parametrizing different manifolds in the image space. **B**. Different architectures of BigGAN and DeepSim GAN, note the side modulation structure in BigGAN Chen et al. (2018). **C**. Samples from different image distributions, from left to right: ImageNet, samples from ImageNet validation set; BigGAN, images sampled from BigGAN using the trained class vectors c and noise distributions z; BigGAN (RND), samples from BigGAN using normally distributed class vector c and noise distribution z; DeePSim, images generated by DeePSim GAN from normal distributed latents $z \sim \mathrm{N}(0, 4I)$; pink noise following $1/f$ spectrum; white noise by i.i.d. uniform distributed pixel values. **D**. Quantifying the naturalness of image statistics by *Frechet Inception Distance* between the image distribution (50000 samples) and ImageNet validation images. **E**. Quantifying the diversity and objectness of image distribution by Inception Score.

averaged direction of these latent codes may not be helpful. Fundamentally, this parameter should be tuned to suit the average "slope" of a tuning landscape. Heuristically, we configured the parameter for each generator such that the samples in each batch are perceptually different

but not totally unrelated. We tested these optimizers on *in silico* units from convolutional neural networks, and confirmed that they can successfully optimize activation for these units.

Note that this is potentially a general strategy to develop evolutionary optimizers for different generative image models including other GANs or Diffusion models. Using this strategy, we also showed proof-of-principle results that closed-loop image optimization can work for other generative spaces such as StyleGAN2 (Karras *et al.*, 2020), while the thorough investigations of those spaces are left for future investigations.

VI.2.2 Comparison of Neuron guided image synthesis in two generative spaces

With this technical success, we were able to perform the comparison of generative spaces *in vivo*. We conducted a series of parallel *Evolution* experiments on two monkeys (A, B). Neurons were recorded through Floating Microelectrode Array (FMA) implanted in three visual cortical areas (V1, V4, posterior IT) in each monkey. Spiking activities of single and multi-units were sorted out online. Our experimental procedure was the following: At the beginning of each session, we mapped the receptive fields of all the recorded neurons and chose a responsive unit as our target, which we called the *driver unit*. Using the firing rate of the driver unit as the optimization target, two *Evolution* experiments, which we called *threads*, were conducted in parallel, one searched in BigGAN space, and the other searched in DeePSim space, with the optimizers configured for each space (**Figure VI-2**A). In each *block*, the images proposed by the optimizer for BigGAN and DeePSim were randomly interleaved in a sequence; then these images were shown to the monkey in a rapid serial visual presentation (RSVP) fashion. Instead of separating images from DeePSim and BigGAN into blocks, this design maximally matched the recording quality and state of the animal between the two threads of optimization.

Specifically, each image was presented on screen for 100ms duration and 150ms inter-stimuli interval; 3-5 images were shown in a trial where the monkey needed to maintain fixation in a circle of 1 deg radius. Failure to maintain fixation will abolish that trial. After all the images were presented, the mean firing rate of the driver unit in 50-200ms to each image was summarized as a scalar score and fed back to the two optimizers (Details see Methods). Then the optimizers proposed two sets of new latent vectors, which were rendered into images, initiating the next block of the optimization loop.

In total, we collected 170 sessions of paired Evolution. We excluded experiments where the recording quality (e.g. baseline firing) was not stable or the experiment has fewer than 15 blocks. After that, we retained 154 *valid* sessions (90 in A, 64 in B) for downstream analysis.

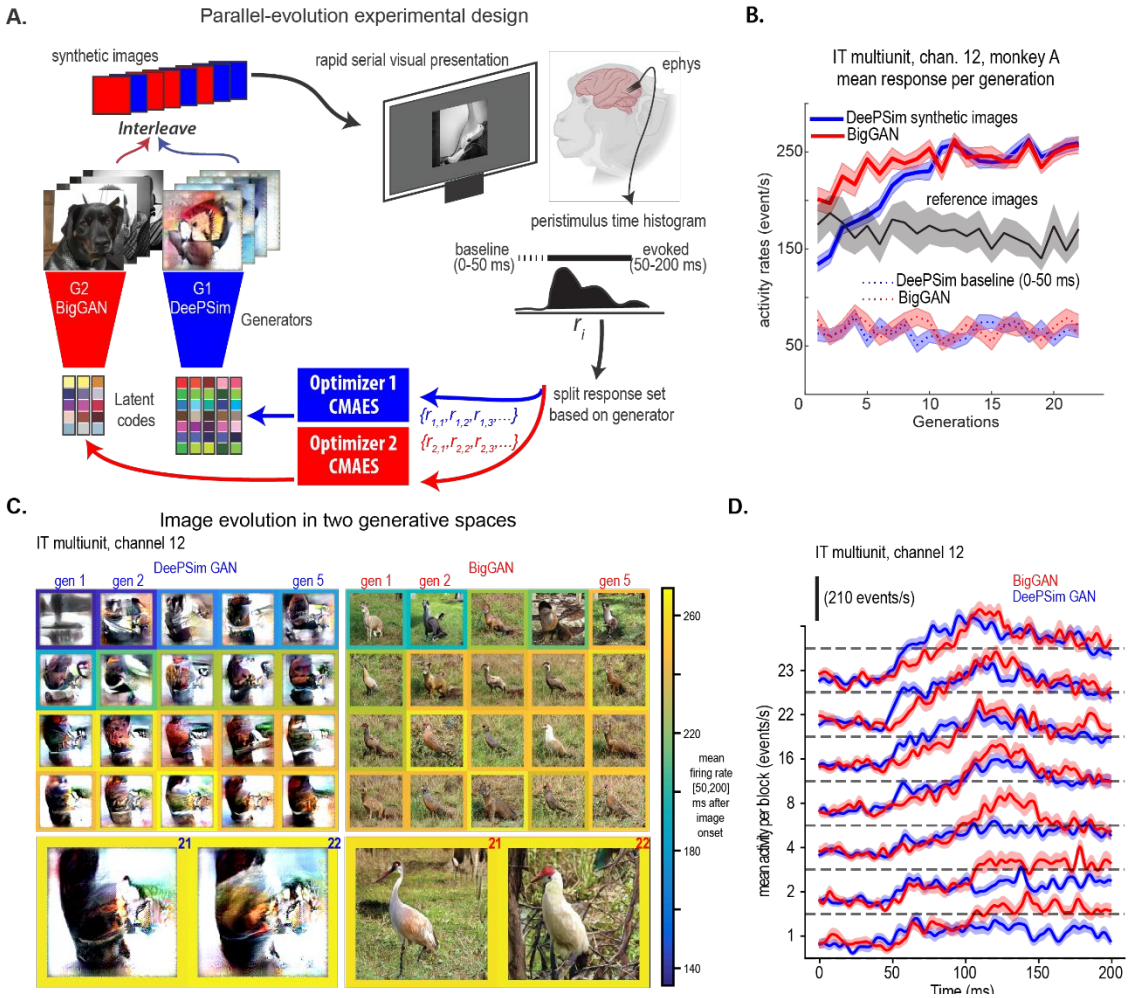


Figure VI-2 **Example of a successful paired Evolution *in vivo*. A**. Schematics showing the workflow of parallel Evolution experiments *in vivo*. **B**. Neural activation during paired Evolution in an example session where both threads successfully increased the firing rate of the target unit. Line and shading showed the mean and standard error of mean (sem) firing rate. Colored solid line, the evoked firing rate (50-200ms) for the images proposed by the optimizers in BigGAN and DeePSim space; Gray solid line, the evoked firing rate for a fixed set of natural images as reference, same for both Optimizers; Colored dotted line, the baseline firing rate (0-50ms) for each thread. **C.** Evolution of images in the DeePSim and BigGAN space in the same experiment. The image evoking the highest firing rate in each block is shown as an exemplar; The color framing the images encodes the average evoked firing rate of the images in that block. The exemplars from the last two blocks were shown in bigger frames. We can see some similar features emerged from the two Evolution threads, but embedded in images with different styles. **D.** Evolution of response dynamics (peri-stimuli time histogram) across generations.

As an example, for an IT neuron in monkey A, the paired Evolution in both DeePSim and BigGAN space successfully increased the neuronal firing rate. (**Figure** VI-2 B,C) When examining the images throughout Evolution, we can see the different ways images evolved in DeePSim and BigGAN space. For DeePSim, the image optimization started from a colorless texture, and the preferred feature was "synthesized" from scratch. In contrast, for BigGAN, the image optimization started with a well-formed alpaca-like creature on a green background, and then the object identity and nuisance variables were gradually updated to maximize the neuronal firing rate. After Evolution, in DeePSim space, the neuron guided the generation of a brownish curvy surface on a white background; while in BigGAN space, the neuron guided the generation of a bird-like creature on a green grassland background (**Figure** VI-2 D.). Potentially, the neuron was responding to the common curvy shape in the bird's neck, as in the optimized DeePSim images. Though these optimized images are apparently different from our eyes, they evoked comparable firing rates in the target neuron (comparing the mean firing rate in $[50,200]$ms window for images of the last two blocks; two-sample t-test $t_{128} = 0.97, p = 0.33$, not significant). This suggests that the neuron could respond similarly to some key features embedded in very different images.

Nevertheless, inspecting the peri-stimuli time histograms (PSTH) across Evolution, we can see that the neural dynamic evoked by the optimized BigGAN and the DeePSim images have subtle differences, i.e. the peaks of neural response for BigGAN images had higher latency than the peaks for DeePSim images (**Figure** VI-2 C). This difference between the response dynamics will be more systematically analyzed below.

VI.2.3 Success rate of Evolution highlights the alignment of the visual area and the generative spaces

Table VI-1 Success rate of *Evolution* experiments. Success criterion, t-test between the activations in the first two generations and two generations around the maximum activation, $p < 0.01$. Similar tables with the alternative success criterions are shown in the Supplementary Information, Table VI-2, VI-3.

	BigGAN	DeePSim	Both
V1	0/10 (00.0%)	10/10 (100.0%)	0/10 (00.0%)
V4	20/38 (52.6%)	37/38 (97.4%)	20/38 (52.6%)
IT	65/106 (61.3%)	67/106 (63.2%)	52/106 (49.1%)
Total	85/154 (55.2%)	114/154 (74.0%)	72/154 (46.8%)

First, we examined the "easiness" for neurons in each visual area to guide Evolution on each image manifold. Taking the analogy of artist and patron further, if the taste of the patron and the style (generative ability) of the artist are matched, it's easy for the patron to nudge the artist to update their painting and produce a pleasing outcome, and vice versa.

We used the success rate of Evolution to quantify this notion. We quantified the success of an Evolution thread by comparing the neural activations (mean firing rate in 50-200ms) in the first two blocks and the two neighboring blocks with the maximum activation, using a t-test (**Table VI-1**). Using this criterion ($p < 0.01$), the overall success rate of BigGAN Evolutions (55.2%, 5-95% confidence interval [48.6%,61.6%], same below) was lower than that of DeePSim Evolution (74.0%, CI [67.8%,79.3]). Although, with a more liberal success criterion ($p < 0.05$), the overall success rates in the two spaces were comparable: DeePSim, 75.3%, CI [69.1%,80.5%] vs BigGAN, 67.5%, CI [61.0%,73.3%]. (Table VI-2)

When we broke down the experiments by the cortical area of the driver unit, we saw BigGAN and DeePSim display strikingly different patterns of success along the ventral stream (**Table VI-1**). We found that, for BigGAN, the success rate was increasing from nonexistent in V1 ($0/10$), to about half in V4 ($20/38$), to around 61% in IT ($65/106$). In contrast, the success rate was near 100% for DeePSim in V1 and V4 cortices ($10/10$ and $37/38$), while it was much lower in the IT cortex ($67/106$). So, for the DeePSim generator, the success rate is generally decreasing along the ventral hierarchy (**Figure VI-3** A). Comparing the two GANs, there was a clear gap between the success rates of DeePSim and BigGAN in both V1 and V4, while their success rates were comparable in pIT. Further, for pIT experiments, the successes of Evolution in the two spaces were not independent (per chi-square contingency test, $\chi^2_{dof=1} = 18.55, p = 1.7\times10^{-5}$). Namely, if Evolution in one space (e.g. DeePSim) succeeded, the Evolution in the other space (BigGAN) was also more likely to succeed, suggesting some common properties of the IT neurons affected the optimization success in both spaces.

We noted that, due to fluctuation in neuronal response, adaptation, or instability of optimization, the firing rate of the neuron could reach a maximal value and then drop before the end of the session. So we also quantified the success of *Evolution* with a more stringent criterion, where the t-test was done between the evoked activations of the first two blocks and those of the last two complete blocks. This results in an overall lower success rate, but the trend across areas and generative spaces are qualitatively similar (Table VI-3)).

These results showed that neurons (esp. in V4 and IT) could guide image generators to synthesize highly activating stimuli both in the DeePSim and the BigGAN space. The success of the Evolution experiments implied that these neurons exhibit gradual tuning on both image

manifolds, such that the evolutionary optimizer could climb their slopes. Moreover, going up the hierarchy, it becomes gradually easier to guide image synthesis in BigGAN space, but gradually harder to guide image synthesis in DeePSim space, consistent with prior results (Rose, Johnson, Wang and Carlos R. Ponce, 2021). The difference in success rate across visual hierarchy suggested that there is a sense of "alignment" between the neuronal tuning ("patron") and the generative network ("artist"): the pattern-based parametrization of DeePSim is more aligned with the neuronal tuning in V1 and V4, while the object-based image parametrization of BigGAN is becoming more and more aligned with the tuning from V1 to IT.

VI.2.4 Activation gap of optimized images differs across areas

A natural following question is, *which artist wins the favor* of neurons? To answer it we analyzed the evoked firing rate of BigGAN and DeePSim images before and after optimization. To take more experiments into account and get more statistical power, we used a more liberal success criterion (max > init $p < 0.05$) in the following analysis. Though, the results didn't change with more stringent criteria.

We found that before optimization, in the first generation, the BigGAN images generally evoked a higher firing rate across all three visual areas (BigGAN > DeePSim, paired t-test, $t_9 = -3.491, p = 6.8 \times 10^{-3}$ for V1, $t_{37} = -4.399, p = 8.9 \times 10^{-5}$ for V4, $t_{105} = -9.286, p = 2.4 \times 10^{-15}$ for IT, all 154 valid experiments included). Potentially, this is due to the initial block of BigGAN images already contained color and objects in contrast to the colorless textures for initial DeePSim images (**Figure VI-3** C). However, after optimization, the results were different. For experiments where at least one thread succeeded, DeePSim images evoked higher activations for V1 and V4 neurons ($t_9 = 4.651, p = 1.2 \times 10^{-3}$ for V1, and $t_{36} = 5.985, p = 7.3 \times 10^{-7}$ for V4); but for IT neurons, BigGAN images generally evoked a similar level of activation ($t_{85} = -1.41, p =$

$0.16, N.S.$) (We noted that, in monkey A, the optimized BigGAN images evoked a slightly higher activation for IT neurons ($t_{47} = -2.32, p = 0.025$), and in monkey B, they are compatible $t_{37} = 0.70, p = 0.49$) (**Figure VI-3** D). One confounding factor for the results is that the success rate differed for BigGAN and DeePSim. To control for this, we considered the Evolution experiments where both threads succeeded (N=87). In this case, for V4 neurons, the optimized activation for DeePSim space was still higher than BigGAN (DeePSim mean±sem 0.898 ± 0.028, BigGAN 0.784 ± 0.020, $t_{23} = 3.707 p = 1.2 \times 10^{-3}$); for IT neurons, the optimized activation of two spaces was comparable (DeePSim, 0.821 ± 0.018, BigGAN 0.820 ± 0.021, $t_{60} = 0.06, p = 0.95$). This trend along the visual hierarchy is worth noting. Similar to the previous results on success rate, the optimization gap between DeePSim and BigGAN is closed in the IT cortex.

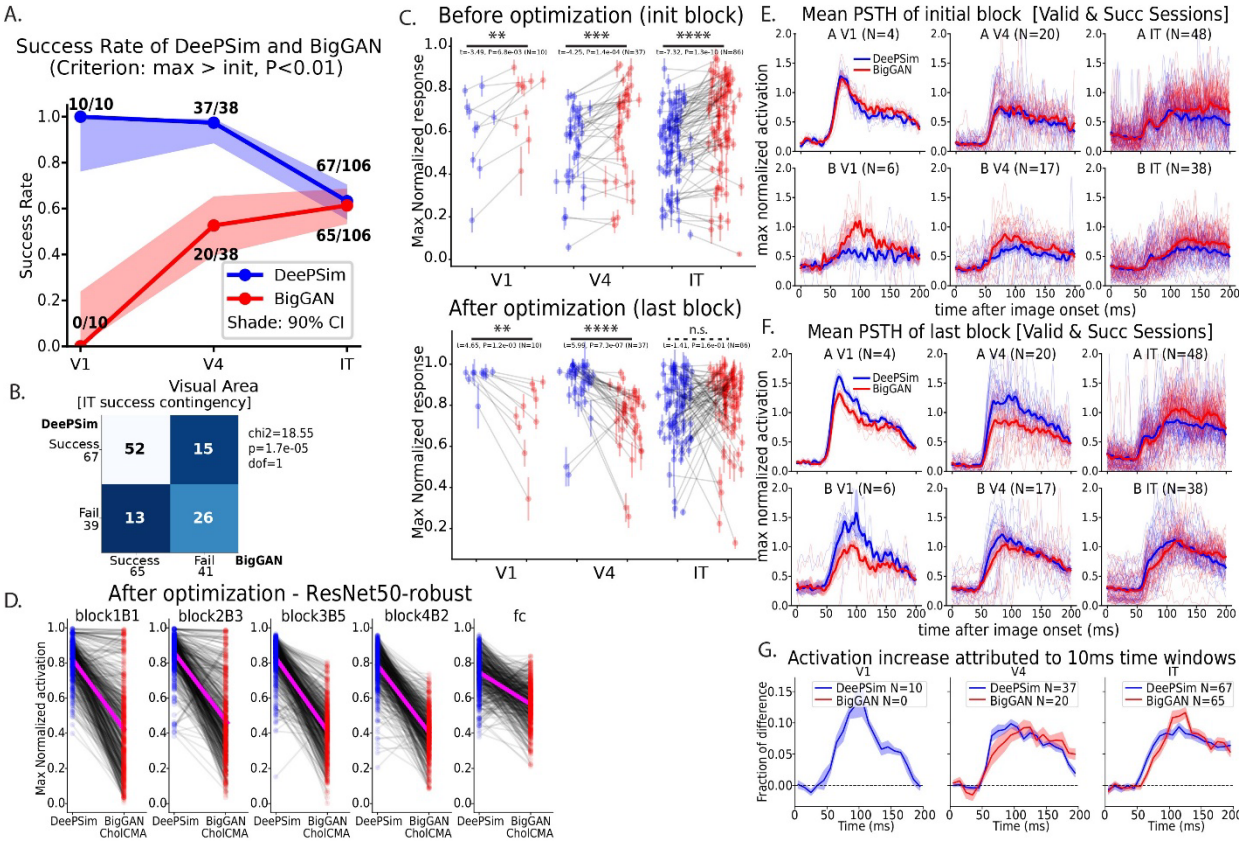


Figure VI-3 **Success Rate and Activation of Evolution in DeePSim and BigGAN *in vivo* and *in silico*. A.** Success rate of Evolution in two spaces across the visual areas, V1, V4, and IT, with criterion max > initial activation $p < 0.01$. **B.** Contingency table of Evolution success in DeePSim and BigGAN, the successes in both spaces were not independent per chi-square test. **C.** Mean activation before (top) and after optimization (bottom), the activations are normalized by the maximum block mean activation across blocks and threads. **D.** The max normalized activation of the last block in DeePSim and BigGAN space for *in silico* experiments on resnet50-robust. **E.** Averaged peri-stimuli time histogram of units in V1, V4 and IT before (top) and after (bottom) optimization. **F.** Activation increase attributed to different 10ms time bins throughout the time course.

To examine this trend in the computational models, we conducted similar parallel evolution experiments on units from *in silico* models of visual hierarchy, Convolutional Neural Networks (CNN). Specifically, for each CNN model, we selected the unit on the center of the feature map as our model of the driver neuron and applied the paired Evolution experiment to it, with 10

repetitions in each generative model. (See Supplementary Information for details of *in silico* paired Evolution.) Surprisingly, we found that, after *Evolution*, the optimized images in DeePSim space almost always achieved a higher activation than those in the BigGAN for units throughout the hierarchy (**Figure VI-3** D, DeePSim > BigGAN, paired t-test, $t_{499} > 24.7, p < 4.7 \times 10^{-89}$ for all layers). Similar to the ventral stream, we also observed that this gap of activation between DeePSim and BigGAN depended on the depth of the unit in the hierarchy, where deeper units had a smaller gap. In ResNet50-robust, for units in the final object classification layer (fc) and the penultimate layer (block4B2), the gap between the two spaces was smaller than the gap for earlier layers (**Figure VI-3** D, two sample t-test, DeePSim - BigGAN gap in block1,2,3,4 > gap in fc, $t_{998} > 16, p < 5 \times 10^{-55}$, for all earlier blocks).

As a side note, the gap of activation between the two spaces partially depended on the Evolutionary or gradient-based optimizer we used, especially for the BigGAN generative model. We tested more variants of optimizers for *in silico* experiments. We found that the optimizers leveraging the anisotropic geometry of the GAN image manifold (HessCMA, HessAdam) will achieve a higher activation, consistent with the result in (Wang and Ponce, 2020). But still, even when equipped with a better optimizer, BigGAN images were hard to be optimized to a level on par or exceeding the activation of the DeePSim images. (See Supplementary Information for results on gradient-based and Hessian-informed optimization.)

In summary, in V4, even when Evolution succeeded in both spaces, optimization in DeePSim space still obtained a higher activation than BigGAN, despite the fact that it started from a lower activation. We interpreted this as additional evidence that the V4 tuning is more aligned with the pattern-based image parametrization and that the tuning peaks of the V4 neuron locate closer to

the DeePSim manifold than the object manifold. However, in IT, the optimization in the two spaces converged to comparable values, showing an increasing alignment to BigGAN manifold than V4. Comparing the biological ventral stream to current computational visual hierarchies, IT neurons also demonstrate a stronger alignment to the BigGAN image manifold. This highlights a knowledge gap in our current model of ventral stream processing, why the units in CNNs are more aligned with the pattern-based generator? To understand this, we dived into one feature that biological neurons have but CNN units lack, which is temporal dynamics.

VI.2.5 BigGAN Evolution evoked and recruited later neuronal response

Visual neurons exhibit dynamic responses to static visual input, and in classic literature, it has been shown that the firing rate responses at different delays encode different visual features (Celebrini *et al.*, 1993; Ringach, Hawken and Shapley, 1997; Lamme, Rodriguez-Rodriguez and Spekreijse, 1999; Sugase *et al.*, 1999; Lamme and Roelfsema, 2000; Brincat and Connor, 2006). These observations have been analyzed and understood under the framework of recurrent neural processing. As the recurrent processing kicks in, the response changes from broader to sharper tuning, from spatially local to contextual, from coarse to fine-grained information about the stimuli, and from linear additive tuning to multiple parts to nonlinear conjunctive tuning to multiple parts.

In order to understand how the "object preference" of neurons changes through time, we analyzed the dynamical response i.e. the PSTH throughout the Evolution (example in **Figure** VI-2 D). Note that, in this data format, we have two senses of dynamics, one is the dynamical neuronal response to images (with the unit of time (ms)), and the other one is the change of neuronal response through the course of image optimization (with the unit of blocks). So how does the dynamical neuronal response interact with the optimization dynamics?

By design, the optimization objective of the experiment is to maximize the overall firing rate in the [50,200]ms time window, but which part within the 50-200ms time window is recruited to increase the overall firing rate is subject to the "preference" of the neuron and the generative space. Given an elevated firing rate, which part of the dynamic response contributed to the increased firing rate? For all successful evolution $p < 0.01$, we computed the firing rate increase in each 10 ms time bin as a fraction of the total increase of mean firing rate (0-200ms) (**Figure** VI-3 G). We found that in IT, the activation increase in BigGAN can be attributed more to later periods, [100,110] ms (DeePSim > BigGAN, two-sample t-test, same below, $t_{130} = -2.129, p = 0.035$) and [120,130] ms ($t_{130} = -2.967, p = 0.0036$) and less to initial periods, [50,60] ms ($t_{130} = 2.144, p = 0.034$) and [60,70] ms ($t_{130} = 2.574, p = 0.011$). This showed that successful Evolution in BigGAN recruited a later portion of neuronal response.

From another perspective, we computed the firing rate in different 25ms time windows during the neuronal responses, and tracked that throughout the blocks, which we called a time-specific optimization trajectory. We found that, for V4 neurons, the response to the DeePSim image surpassed that of the BigGAN images in all the 25ms time windows of the PSTH (**Figure** VI-4 E, upper). However, for IT neurons, in the initial time windows ([50,75]ms, [75,100]ms), the optimization in DeePSim space increased the firing rate more than that in the BigGAN; but in later time windows (starting 100ms) BigGAN images started higher and stayed over DeePSim images, and didn't get surpassed during optimization (**Figure** VI-4E, lower). This showed that, for V4 neurons, throughout the course of responses, DeePSim were generally easier to optimize than BigGAN. However, for IT neurons, the earlier more transient responses were more attuned to DeePSim and optimized it better, while the later responses were more attuned to BigGAN and preferred BigGAN than DeePSim throughout optimization process.

This observation is consistent with the prior research on the dynamics of visual tuning. As the earlier response have linear additive tuning to various features (e.g. geometric parts for V4 in (Brincat and Connor, 2006; Yau *et al.*, 2012)), DeePSim generator can easily synthesize these features to activate the early part of tuning. However, if the configuration of these features are not optimal, they may not activate the later nonlinear tuning. In contrast, the BigGAN synthesize more coherent combinations of features embedded in objects, which may not be as easy to optimize for earlier responses, but it could suits the preference of later nonlinear tuning better.

VI.2.6 Optimization dynamics of BigGAN and DeePSim evolutions suggesting their differential alignment with visual cortices

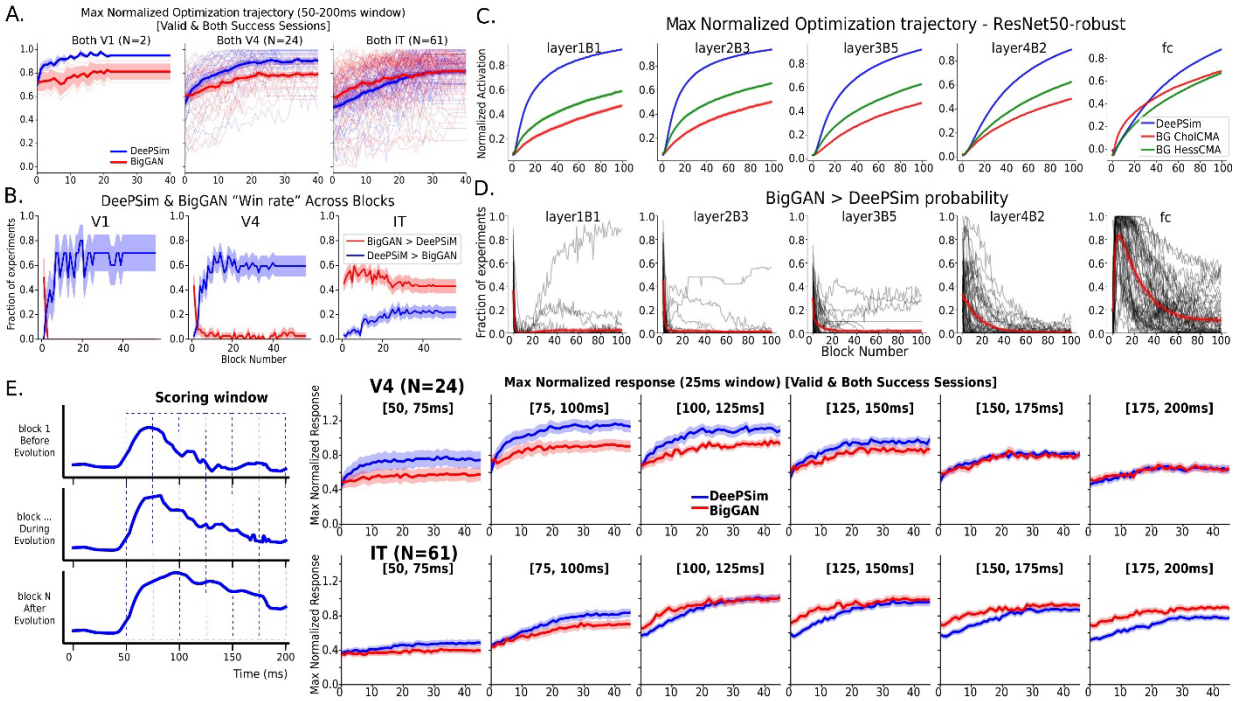


Figure VI-4 **Compare Activation During Evolution in DeePSim and BigGAN *in vivo* and *in silico*. A.** Average score trajectories during Evolution for the two GANs in three areas. **B.** Win rate of DeePSim and BigGAN through blocks. Win-rate of BigGAN is quantified by the fraction of sessions where responses to BigGAN images were significantly higher than those to DeePSim images. This fraction is plotted as a function of block for the three areas. **C.** Averaged optimization trajectory for units in ResNet50-robust, throughout the layers, the activation of each unit is normalized by the max activation ever achieved for that unit. **D.** Win rate of BigGAN for ResNet50robust units. The thin black trace is the averaged win rate for each unit across 10 repetitions, the red curve is the average win rate curve for each layer averaged across all the units. The relation of D to C is similar to the relation of B to A. **E.** The evolution trajectory of V4 and IT neurons with firing rates in different 25ms time windows. For IT neurons, in the earlier time windows, the DeePSim evolutions prevailed and surpassed BigGAN throughout the blocks. However, in the later time windows, BigGAN surpassed DeePSim throughout the Evolution of IT neurons.

The Evolution experiment is an optimization process of neuronal activation. So how does neuronal activation change through Evolution? We call this optimization dynamics of neural activation. By climbing the peaks, the optimization dynamics carry information about the underlying tuning landscapes. Intuitively, if the preferred motif is prevalent on the generative

image manifold, the Evolution will converge faster to it; if not, it will take longer to synthesize it from scratch.

We visualized the averaged max normalized optimization trajectory across the three areas (**Figure** VI-4 A, details see methods). For the IT cortex, the activation evoked by BigGAN images stayed on top of the DeePSim for a longer number of blocks. But for V1 and V4, it's surpassed by V1 and V4 after a few blocks. We can see the "win rate" of BigGAN stays around 40%.

We also quantified the time constant of the successful Evolution trajectory by the number of generations that reaches 0.632 fraction of activation increase (see Methods). The generation required to search for optimal images in DeePSim space increased along visual hierarchy (IT > V4 22.134 ± 7.827 $(N = 67)$ versus 14.324 ± 8.759 $(N = 37)$, $t_{102} = -4.668, p = 9.3\times10^{-6}$, IT > V1 $10.100\pm9.712(N = 10)$, $t_{75} = -4.395, p = 3.6\times 10^{-5}$), consistent with the previous findings (Rose, Johnson, Wang and Carlos R. Ponce, 2021; Wang and Carlos R. Ponce, 2022d). Further, comparing the two generative image spaces, for IT neurons, optimization in the BigGAN space converged faster than in the DeePSim space (DeePSim > BigGAN 22.250 ± 8.169 vs 17.673 ± 8.222 $(N = 52)$, $t_{51} = 3.358, p = 1.5 \times 10^{-3}$). (**Figure** VI-4 A)

We repeated the same analysis on the *in silico* experiments. We found the same result that for deeper units in the CNN, the activation of BigGAN thread would stay higher than DeePSim for a longer period (**Figure** VI-4C,D).

This difference in dynamics also suggests an increasing alignment to the object image manifolds (BigGAN) in the IT cortex than the V4 cortex.

VI.2.7 Effect of Evolution on the "objectness" of generated images

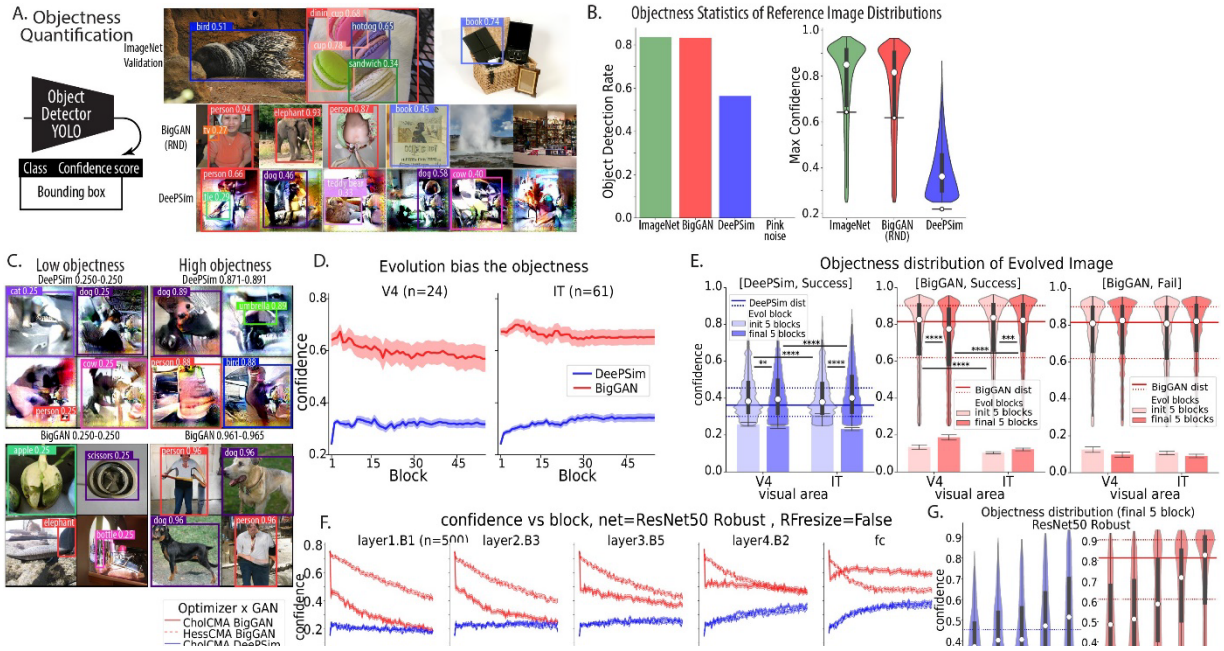


Figure VI-5 **Effect of Evolution on the "objectness" of image. A.** Examples for objectness quantification pipeline. Object detection and their bounding boxes from YOLO were shown on sample images from ImageNet validation set, BigGAN and DeePSim. Multiple or no bounding box might be proposed, and the max confidence is used for quantification. **B.** The distribution of objectness score for image spaces. Left, bar plot, the detection rate; Right, violin plot showing the distribution of (non-zero) objectness score among 50000 images. **C.** Image samples from concurrent Evolution experiments with the highest and lowest (non-zero) objectness score and their detection(s). The samples with the highest objectness score indeed create visual impressions of the detected objects. **D.** Evolution of mean objectness score of images sampled from each block, experiments where Evolution succeeded in both spaces were included $p < 0.05$. **E.** Objectness distribution of generated images from the first 5 blocks vs the last 5 blocks (violin plot), median showed in white dot, quartile range showed in thick black lines. The baseline objectness distributions for the corresponding GAN are showed in horizontal lines; the median showed in solid line, and the quartiles showed in dashed lines. **F.** Evolution of objectness score for *in silico* experiments on ResNet50-robust, separated by the layer of the unit (similar format to D). **G.** Distribution of objectness score as a function of the layer of guiding unit, computed for image samples in the last five blocks of Evolution (similar formats to E).

Just like the patron's taste will influence the trend of art creation, the tuning and preference of neurons will bias the image generation process. So how do the generated images change across the Evolution process? Specifically, does neuron-guided image sampling make the images more object-like or otherwise?

The sense of object is deeply rooted in visual perception , thus by construction, it's defined by the perceptual system, i.e. human observer. To efficiently quantify the objectness of a large set of images, we resorted to the computational surrogates, *object detection* systems in computer vision as the observer and let them be our judge. The object detection networks (e.g. You only look once, YOLO (Redmon *et al.*, 2016; Wang, Bochkovskiy and Liao, 2023)) take each image as input and output zero or any number of objects along with their bounding box, class label, and confidence score (**Figure** VI-5 A). Casually speaking, the confidence score signals the estimated likelihood of the predicted class of the object being in the bounding box. During supervised training, it's trained to predict the Intersection over Union (IoU) of the predicted bounding box and the true bounding box. We took the maximal confidence value across all the detection boxes in an image as our "objectness" score of the whole image; when no object was detected, we filled the objectness score as 0. To give a sense of what kind of objectness these models were measuring, we visualized the detection boxes for some natural and generated images per the YOLOv5 model (**Figure** VI-5 A). These models can detect persons, animals or objects from ImageNet or generated images, although the class labels are not always correct, given that the 80 classes used for training YOLO are not identical to ImageNet 1000 classes. Further, for outdoor naturalistic scenes or highly cluttered scenes (e.g. the geyser and grocery store generated from BigGAN), no objects were detected, consistent with the contrast between scenes and objects. (For details about this objectness measure and alternative methods, please refer to Method.)

As a baseline, we quantified the objectness of several reference image distributions (**Figure** VI-5 B). We sampled 50,000 images from each distribution and sent them through the YOLO model. As expected, for random pink noise patterns, no object was detected, resulting in an objectness score of 0. As for DeePSim, when we sampled latent vectors from a Gaussian

distribution $z \sim \mathrm{N}(0,4I)$, the generated images have an objectness score of 0.220 ± 0.001, with object detected in 56.3% of the images. Images generated by Gaussian random latent vectors in BigGAN have an objectness score of 0.618 ± 0.001 with a detection rate of 83.3%. As for the validation set of ImageNet 2012, objects were detected in 83.6% of the images, with an average objectness score of 0.643 ± 0.001. This validated our perception that random BigGAN images usually have objects on the image, similar to that of ImageNet, and the YOLO system can detect it with high confidence. This also showed that the images sampled from the DeePSim generative model had a primitive sense of "objectness", though not as high as the BigGAN images.

Generative image models such as GAN are mappings between a latent space and the image space. During the training of GAN, the latent vectors were sampled from a simple distribution $p(\mathbf{z})$, which mapped to a baseline image distribution $p(G(\mathbf{z}))$. During Evolution, the latent vectors were guided by neurons towards an updated distribution $q(\mathbf{z})$, which should concentrate more on the *high activation domains* of the neuron on the image manifold if the optimization succeeded. Technically, this could be understood in the framework of energy-based models, where the Evolution process approximates the Langevin dynamics guided by the energy function. So how does the Evolution process shape the image distribution? Or equivalently, how does the neuron allocate their high-activation domains on the two image manifolds (**Figure** VI-1)?

The averaged trends of objectness evolution through the optimization process were shown in **Figure** VI-5 D. Here, we included the experiments where both threads succeeded (max > init $p < 0.05$), and excluded V1 experiments due to the limited success of BigGAN. For evolution trajectories that terminated too early, we extrapolated them by the same value as the last blocks.

Surprisingly, the guidance from both V4 and IT neurons made the objectness of BigGAN images decrease gradually, while V4 neurons induced a more substantial decrease. In contrast, the guidance from both areas made the objectness of DeePSim images increase while IT neurons induced a larger increase. Quantitatively, we compared the objectness score between images from the first five blocks and the last five blocks in each Evolution experiment, using the original distribution of the images generated from GAN as a baseline (**Figure** VI-5 E). We found that neuron-guided sampling generally increased the objectness of images in DeePSim space: from 0.304 ± 0.003 to 0.315 ± 0.003 for V4 neurons and from 0.279 ± 0.002 to 0.331 ± 0.002 for IT neurons (two-sample t-test last five > initial five blocks, same below, $t_{11542} = -2.72, p = 6.6\times10^{-3}$ for V4, $t_{20722} = -17.24, p = 3.7\times10^{-66}$ for IT). Comparing the two areas, Evolution guided by IT neurons generally resulted in higher objectness of DeePSim images in the last generations (IT > V4 $t_{17145} = 4.59, p = 4.5 \times 10^{-6}$).

In contrast, the effect of Evolution on BigGAN images depended on the cortical area of the driver neuron and the success of the Evolution. Intriguingly, if a neuron-guided Evolution succeeded, the BigGAN images become less object-like. When it's successfully guided by V4 neurons, the final objectness score dropped from 0.653 ± 0.006 to 0.581 ± 0.006 which was lower than the reference BigGAN distribution ($t_{5233} = 8.11, p = 6.3\times10^{-16}$). When the Evolution was successfully guided by IT neurons, the objectness of BigGAN images slightly decreased from 0.684 ± 0.003 to 0.667 ± 0.003 ($t_{17055} = 3.66, p = 2.5\times10^{-4}$), but it still remained higher than the reference BigGAN image distribution ($t_{58405} = 17.17, p = 6.8 \times 10^{-66}$). Thus, successful guidance from IT neurons resulted in a higher objectness of BigGAN images (IT > V4, $t_{11304} = 12.50, p = 1.3 \times 10^{-35}$) As a control, when the Evolution in BigGAN space didn't succeed, the objectness of BigGAN samples didn't decrease but increased slightly instead: from $0.654 \pm$

0.008 to 0.683 ± 0.007 for V4 ($t_{3109} = -2.80, p = 5.2 \times 10^{-3}$), and 0.659 ± 0.005 to 0.693 ± 0.005 for IT ($t_{6179} = -4.79, p = 1.7 \times 10^{-6}$).

In our landscape view, this suggests that the neurons in posterior IT have their tuning peaks close to the high "objectness" manifold, and sampling guided by these neurons will result in higher objectness images on both DeePSim and BigGAN image manifold; while the tuning peaks of V4 neurons are more "abstract" and more distant from this high "objectness" manifold, and guidance from it will lead the images further away from the object manifold of BigGAN and nearer the pattern manifold of DeePSim.

We replicated this analysis on the *in silico* Evolution guided by units from convolutional neural networks (CNNs) and we found a similar trend (**Figure** VI-5 F): in DeePSim space, the Evolution generally increased the objectness of the samples; while in BigGAN space, it decreased the objectness, with the effect size depending on the layer of the unit. Further, for deeper units in the network, DeePSim images increased their objectness more and BigGAN images decreases their objectness less (**Figure** VI-5 G). In summary, deeper units converged to images with higher objectness in both DeePSim and BigGAN space. This showed that the higher-level units in a computational visual hierarchy also have their tuning peaks closer to the high-objectness image distribution, qualitatively similar to what we found for *in vivo* neurons in the ventral hierarchy. For *in silico* experiments, we also examined the effect of Evolutionary optimizer (esp. HessianCMA) on this result and found similar trends across the hierarchy.

VI.2.8 Similarity of the Evolved images was predicted by the similarity of optimized neuronal dynamics (PSTH)

Finally, we asked what kind of features did each artist create for the neuron? Do they agree with each other? Directly, visualizing the images synthesized by the two image generators, we

can see that similar motifs are embedded in them: in an example paired *Evolutions* driven by an IT unit in monkey A, the curved vertical contour between the brown shape and the white background in the DeePSim image (left) is visually similar to the contour between the curly neck of the egret and the yellow-greenish background (right, **Figure** VI-2 D). More pairs of examples can be found in **Figure** VI-6 A.

Since the stimuli sampled during Evolution experiments are highly variable, we used a previously developed pipeline to find the image features that most correlated with the response of the neurons and amplified them (Wang and Carlos R. Ponce, 2022d, 2022a). For each thread of Evolution, we used all images and responses in it to fit a predictive model of neuron, and then optimized image *in silico* to drive this model unit. We called the resulting image from *in silico* optimization the *feature exemplar*. These re-evolved images emphasized the key features better than the single max-response images, and the similarity between the two threads can be better seen (**Figure** VI-6 B).

Unlike neuronal firing rate, the degree of similarity between images depends on a perceptual system. So, the challenge is to quantify the *sense and degree* of similarity between images. For example, image similarity in terms of local texture and color is very different from the similarity in terms of object identity. Here, we used ResNet50-robust as our perceptual system, and used its penultimate layer as the embedding space. We compared the prototype image similarity by the cosine similarity of the image embedding c_{embed}.

We first asked, are the prototype images from the two threads driven by the same neuron similar to each other? We found that for IT cortex, the answer is yes. Using the layer4 as embedding module, the image embedding similarity between images from paired Evolution was

higher than the embedding similarity of shuffled pairs from different driver unit when both or one thread succeeded (paired vs shuffled 0.438 ± 0.010 vs 0.407 ± 0.007 paired t-test, $t_{60} = 3.64, p = 5.8 \times 10^{-04}$); the image similarity between the *feature exemplars* were even higher, but the same results held (paired vs shuffled 0.546 ± 0.010 vs 0.516 ± 0.007 $t_{60} = 3.74, p = 4.2 \times 10^{-04}$). However, we didn't find a significantly higher similarity for paired successful evolutions driven by V4 units (paired vs shuffled 0.435 ± 0.016 vs 0.438 ± 0.015 $t_{23} = -0.28, p = 0.78$, N.S.). This showed that, using the relatively high-level image similarity metric, successful paired Evolution driven by IT neurons were guided towards to similar patterns on a high level. But for V4 neurons, the two threads were not guided to be similar on such high level.

Further, we observed that some pairs of *Evolutions* ended up with more similar prototypes in the two spaces and some with less similar prototypes. How should we understand this heterogeneity? We computed the mean PSTH corresponding to the max activation block in each thread and compared these PSTHs (**Figure** VI-6 C). We found that the integrated difference between the two normalized PSTH d_{PSTH} was a good predictor of the similarity of image embeddings of the two space c_{embed}. For paired Evolution experiments where both threads succeeded, the Pearson correlation between them d_{PSTH} and c_{embed} is $-0.447, (N = 87, p = 1.4 \times 10^{-5})$ (**Figure** VI-6 F). We noticed that the distance between PSTH is a better predictor of image dissimilarity than distance between scalar activation d_{act}, which has Pearson correlation with c_{embed} $-0.215, (N = 87, p = 0.046)$. This showed that the specific time-course differences between PSTH encode important information about the difference between the two prototypes.

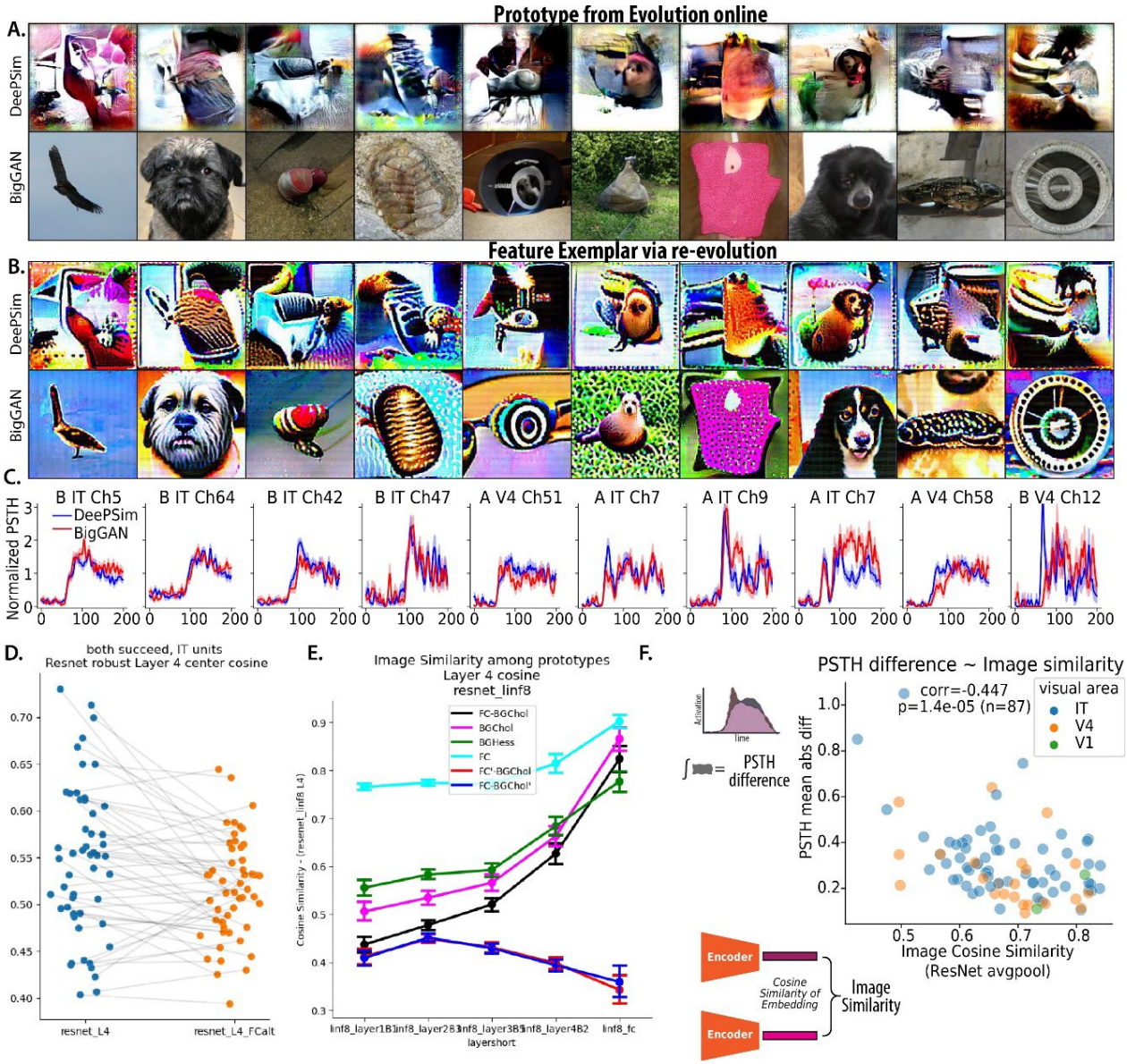


Figure VI-6 **Similarity of Evolved features in the two generative spaces conditioned on the successfulness of Evolution. A.** Selected evolved images from DeePSim and BigGAN, the image empirically evoking the highest activation is selected. **B**. The feature exemplar by re-optimizing the *in silico* model of the online *Evolution*. **C**. Comparison of mean PSTH for the image block evoking the highest firing rate in DeePSim and BigGAN evolution. **D**. Image similarity of paired DeePSim and BigGAN prototype is higher than that similarity of the shuffled pairs. **E**. Plot similar to **D** applied to Evolution of *in silico* units from ResNet50-robust. **F.** The similarity of image embedding is anticorrelated by the PSTH distance.

VI.3 Discussion

In this paper, we presented evidence for an increasing alignment of neural tuning with the BigGAN object image manifold than the DeePSim image manifold along the ventral stream. We showed that the success rate of Evolution in DeePSim space decreased along the hierarchy, while that of BigGAN increased along the hierarchy. When both threads succeeded, there was a gap of activation between optimized images for V4 neurons (DeePSim surpassed BigGAN), while the gap was closed for pIT neurons. Through the response time course, the initial response of pIT neurons were more aligned with the pattern image manifold and during Evolution DeePSim recruited more initial responses; in contrast, the later responses were more aligned with the object image manifold, i.e. during Evolution, BigGAN recruited more later responses. Finally, we found that while neuron guidance generally increased the objectness of DeePSim images and decreased the objectness of BigGAN images; successful guidance from IT neurons led to higher objectness score than V4 in both spaces. Further, guidance from IT neurons led to convergence of features in a higher-level space.

Overall, our results paint a picture that different areas from the ventral hierarchy are aligned differentially with different generative latent space. When the parameters that control the image are properly aligned with the variables the neurons tuned to, then the tuning landscape will be smoother, and easier for Evolutionary optimizer to operate. In V4, the easier hill climbing in the DeePSim latent space versus the BigGAN latent space argues that the neurons' tuning aligns more with the abstract axes in the DeePSim space than the BigGAN space; while in pIT, the alignment to two spaces seems comparable at least and may be more aligned to the BigGAN space. Although we have just tried two different generative spaces, there are infinite numbers of them. Maybe each visual cortex or each visual neuron is optimally aligned with a different image generative model or image manifold, on which it would be easiest to optimize and control. It's

worth testing whether more anterior regions in the ventral stream will align even more with the object-based image parametrization, and even less with the pattern-based image parametrization. For these purposes having a continuous gradient of generative models parametrizing different levels of image manifolds may be helpful, where we can smoothly traverse this axis. The recently popular conditional diffusion generative models may be a good fit for this. As they are promptable by natural language, changing the natural language prompt could condition it into parametrizing different image manifolds.

VI.4 Methods

VI.4.1 General Experimental Setup

Experimental sessions were controlled using MonkeyLogic2, which directed the presentation of visual stimuli on ViewPixx EEG monitors (ViewPixx Technologies). The refresh rate was 120 Hz at a resolution of 1920*x*1080 pixels. The monkeys were placed 58 cm from the screen. Gaze position was tracked via ISCAN cameras (ISCAN Inc.). Animals fixated 0.25°-diameter circles within a 2.0°-wide window during stimulus presentation; they obtained a reward if they held their gaze on the target for 2-3 s. Rewards are delivered via DARIS Control Module System (Crist Instruments).

VI.4.2 Animal Subject

Two male adult rhesus macaques (A and B, Macaca mulatta, ages 8-9, 10-11 kg) were implanted with chronic floating microelectrode arrays (Microprobes for Life Sciences, MD) in the right hemisphere: one array was located at the posterior lip of the lunate sulcus (corresponding to the V1/V2 transition), one on the prelunate gyrus (V4) and another anterior to

the inferior occipital sulcus (PIT). We refer to these sites as V1/V2, V4 and IT. The locations of the arrays were chosen based on sulcal landmarks, around local vasculature.

VI.4.3 Neuronal Recording

Neurophysiology data was acquired using OmniPlex Neural Recording Data Acquisition Systems (Plexon Inc.), with the PlexControl client to sort electrical events online based on waveform and inter-spike-intervals. Because all Evolution experiments were based on a closed loop between neuronal activity and image synthesis, spike sorting was done at the beginning of each experiment. Within each channel, events were estimated as arising from single-units, multi-units, or "hash" using a 1-5 scale, where 1 indicated strong confidence on the presence of a single-unit (based on waveform shape, inter-spike interval and separation from the main hash signal) and 5 indicated hash/multiunit activity. We use the term site to refer to all signal types; across experiments, sites comprised mostly multiunits/hash and a fraction of single units. After data collection, spike/event times were discretized into 1-ms bins and convolved with a symmetric Gaussian probability density function with a 2-ms standard deviation.

VI.4.4 Pretrained Image Generative Models

For BigGAN, we used the BigGAN-deep-256 version, implemented in pytorch-pretrained-BigGAN. For DeePSim, we used the FC6 version, with a custom PyTorch implementation translated from the original Caffe implementation.

VI.4.5 Developing Evolutionary algorithm

We used and adapted the evolutionary algorithms developed in (Wang and Ponce, 2020; Wang and Carlos R. Ponce, 2022d). For the DeePSim GAN evolution, we used the Cholesky CMAES algorithm and the HessianCMA algorithm. The Cholesky CMAES has the same parameters as in the previous works (Rose, Johnson, Wang and Carlos R. Ponce, 2021; Wang

and Carlos R. Ponce, 2022d): standard deviation was initialized as 3.0. For the HessianCMA Optimizer, we pre-computed the 500 most informative dimensions (top eigenvectors) in the latent space of the DeePSim GAN (Wang and Ponce, 2020), and only optimize in that linear subspace, which is more sample efficient and yields comparable activation with the optimizer operating in the full 4096D space.

For the BigGAN evolution, we used the Cholesky CMAES and HessianCMA algorithms (Wang and Ponce, 2020) with the optimized parameters found on *in silico* experiments: the initial standard deviation was 0.06.

VI.4.6 Neuron-guided image synthesis

We used the same experimental protocol for neuron-guided image synthesis as in (Wang and Carlos R. Ponce, 2022d) (Section IV.4.4, IV.4.6).

In each Paired Evolution session, we selected a neuronal site from the array, as our target site (*driver*). We only selected driver units with well-defined receptive field and visual evoked responses. We first determined the optimal location (image center and size) for the driver using RF mapping experiments. Then we placed the stimuli at the optimal location and started the *Paired Evolution* experiments. We have two image generators; each has a set of 30 initial images as the starting point. For DeePSim generator these initial images are the Portilla and Simoncelli textures inverted into the DeePSim space. For BigGAN generator, these are some images generated from random vectors with low vector norm. The experiment started by presenting these initial generated images for both generators and 10 reference images. Reference images were selected if they were known to evoke high activity from the array site under study, per prior experiments. After all images were presented once, the driver's spike rate responses during 50-

200 ms post image onset was averaged as the score of the corresponding image. The latent vectors from each generator and their corresponding scores were provided to the Evolutionary optimizer (e.g. CMA-ES, HessianCMA) tuned for this generator. Then, the optimizer proposed the next batch of latent vectors for the corresponding generator. These vectors were rendered by the generator to create new images. Finally, the new batch of images from both generators were randomly interleaved into a sequence and presented to the monkey, starting the new block or iteration. Each experiment comprised 10 to 60 blocks, and was stopped after the neuronal firing rate converged or stopped increasing for both threads.

VI.4.7 Image Similarity Measure

Generally, image similarity is computed in a certain feature space, here we choose ResNet50-robust as our image embedding model and computed the cosine similarity of embeddings as the image similarity score.

VI.4.8 PSTH Similarity Measure

To quantify the similarity or distance between PSTH, we used two measures, one integrated the absolute area between the two PSTH curves (Eq. VI-1), we call it $\boldsymbol{d_{psth}}$, quantifying the dynamical difference between PSTHs. The other distance $\boldsymbol{d_{act}}$ computed the difference between the averaged level of PSTH curve, effectively integrating the signed area between the PSTH curves (Eq. VI-2). This distance disregards the temporal difference by averaging them out.

$$d_{psth} = \int_t \left|\bar{r}_{DeePSim}(t) - \bar{r}_{BigGAN}(t)\right| dt \qquad \text{(VI-1)}$$

$$d_{act} = \int_t \left(\bar{r}_{DeePSim}(t) - \bar{r}_{BigGAN}(t)\right) dt \qquad \text{(VI-2)}$$

VI.4.9 Feature Attribution of Evolution

To find which features are important during the *Evolution* process, we used our methodology developed previously (Wang and Carlos R. Ponce, 2022d, 2022a). Simply put, we distill the

image and response pairs into a model, by finding all the correlated units in the model with the given neuronal firing rate. Then optimize the model to find its max firing rate stimulus, this is what we termed feature exemplar. For more details see Section IV.5.9 Correlated feature attribution.

VI.5 Appendix

Table VI-2 Success rate of *Evolution* experiments with alternative Success criterion: **t-test between the activations in the first two generations and two generations around the maximum activation, *init* 2 < *max* 2, $p < 0.05$.**

	BigGAN	DeePSim	Both
V1	2/10 (20.0%)	10/10 (100.0%)	2/10 (20.0%)
V4	24/38 (63.2%)	37/38 (97.4%)	24/38 (63.2%)
IT	78/106 (73.6%)	69/106 (65.1%)	61/106 (57.5%)
Total	104/154 (67.5%)	116/154 (75.3%)	87/154 (56.5%)

Table VI-3 Success rate of *Evolution* experiments with alternative Success criterion: **t-test between the activations in the first two generations and last two generations *init* 2 < *last* 2, $p < 0.01$.**

	BigGAN	DeePSim	Both
V1	0/10 (00.0%)	10/10 (100.0%)	0/10 (00.0%)
V4	15/38 (39.5%)	35/38 (92.1%)	14/38 (36.8%)
IT	54/106 (50.9%)	59/106 (55.7%)	41/106 (38.7%)
Total	69/154 (44.8%)	104/154 (67.5%)	55/154 (35.7%)

Chapter VII : Discussion and Outlook

VII.1 Summary of the Thesis

In this thesis, we analyzed the structure of visual neural code of ventral stream neurons as a function on the natural image manifold. We chose deep generative adversarial network as an approximation of natural image manifold. First, we characterized the geometric structure of this image manifold. By computing the Riemannian metric tensor of the manifold, we found that the latent space of the generator is highly anisotropic, some directions induced large changes on the images while some directions induced negligible changes. This anisotropy is aligned across the space, thus there is a global anisotropy in the latent space. We further found that the directions that induce large changes (top eigen dimensions) usually correspond to interpretable transformations of images. Leveraging this anisotropic structure of the latent space we were able to develop more efficient evolutionary algorithms to control neuronal responses. Further, leveraging the spherical geometry, we benchmarked and developed a variant of CMA-ES algorithm as the best optimizer to control neuronal responses.

Using these tools, we characterized the neuronal tuning landscapes on the highly expressive 4096d image manifold (DeePSim). We first searched for peaks on the tuning landscapes and then characterized 2d tuning map sections around the peak. We found that neurons exhibit smooth bell-shaped tuning around the peaks found by evolutionary optimizers. These peaks spanned a larger dynamic range and a wider image domain than those tuning curves in classic image spaces. We further found that along the visual hierarchy, the width of the tuning peak got increasingly sharp; the number of optimization steps needed to find a peak also increased. Further, earlier visual neurons can reach their tuning peaks in a random 50d subspace, while the

higher visual neurons found it increasingly hard to reach high activation in a restricted subspace. We inferred that there is an increase in intrinsic tuning dimensionality and decrease to tuning width throughout the ventral stream.

We also compared how the tuning landscape differed on two image manifolds: the pattern-based image generator (DeePSim) and the object-parametrizing image generator (BigGAN). We found that throughout the ventral hierarchy it became increasingly easy to drive image optimization on BigGAN image manifold but increasingly hard to do so on DeePSim image manifold. Further, when optimization succeeded in both spaces, the optimized DeePSim images reached higher activations than optimized BigGAN in V4; however both spaces reached comparable activation in pIT. This highlights an increasing alignment of neural tuning with the object-based manifold along the ventral stream. Further, we observed the dynamics of this preference in IT neurons: optimization in BigGAN space tended to recruit later neuronal responses. For the initial response, the DeePSim optimization can often win over BigGAN; while the later response, it's the opposite. This also showed that for IT neurons, through recurrent computation, the neuronal tuning is becoming more aligned with the object-based image manifold. In terms of image, neuron-guided sampling decreased the "objectness" of BigGAN images while increasing "objectness" of DeePSim images during the evolutionary process. Across the cortical regions, IT driven sampling resulted in a higher objectiveness of both BigGAN and DeePSim images than those of V4. This highlighted that the tuning peaks of both IT and V4 neurons were off the object image manifold, while those of IT neurons were closer to it.

VII.2 Generative Model and Vision, What's Next?

Connecting our work to the literature in Chapter I, what are some open questions revealed by the picture of tuning landscape? Here are some future directions that reader may consider as follow ups.

VII.2.1 Relating the Tuning Landscape to Natural Image Statistics

In this thesis, we have experimentally located many peaks of tuning functions for neurons in V1, V4, pIT and units in Deep neural networks. So we can answer, where the tuning landscapes allocate their peaks. But, why are the peaks (prototypes) where they are now? We can explain V1 receptive field (prototype) with efficient coding and natural images (Olshausen and Field, 1996). A natural question for these units deeper in the visual system is, can we explain their peaks positioning using natural image statistics and some other principle?

In two works from Connor lab (Yamane *et al.*, 2008a; Carlson *et al.*, 2011), they have explicitly investigated the population statistics of the location of the tuning peaks (center of their Gaussian subunit) in a feature space (e.g. orientation, curvature), with respect to the distribution of these features in natural images. They found that for both 2d contours and three-dimensional shapes, the peaks of their visual neurons were more biased towards the high curvature parts. They found that this biased to high curvature features could be explained by a sparse coding objective for the neural representation, similar to the idea of (Olshausen and Field, 1996).

It will be interesting for a computational and theoretical study to show how our experimentally observed prototype could be related to the distribution of natural images, and what are the underlying optimization principles governing their position.

VII.2.2 Dynamics and Stability of Tuning Landscape

Vision is dynamics, and this thesis has not done enough to analyze the dynamical part of vision. We think it will be interesting to analyze the dynamics of tuning landscape both in short timescales (100ms) and long timescales (days or months).

We provided compelling evidence that the peak of tuning landscapes (prototypes) can be stable across 2-4 weeks: the prototype images from independent Evolution experiments contain highly similar visual motifs. Even though the dynamic range of firing rate changed across this period, the maximizer of firing rate stayed relatively stable. David Leopold lab has also provided evidence for representation stability in primate IT cortex across days and weeks (Bondar *et al.*, 2009). Recently, they also showed that the neuronal responses also subject to gradual plasticity that decrease the late responses to familiar objects (Koyano *et al.*, 2022). Comparing to the mice posterior cortex where representation drift is a major theme (Rule, O'Leary and Harvey, 2019; Driscoll, Duncker and Harvey, 2022; Masset, Qin and Zavatone-Veth, 2022; Qin *et al.*, 2023), the primate visual cortex seems relatively stable. It will be interesting to investigate the long-term stability of the prototype and the tuning landscape more systematically.

Beyond natural condition, if features of certain prototype are relevant to tasks of the subject, it will also be interesting to see how the landscape will change related to the task learning process. It's plausible that the landscape will change the slope around the critical feature, such that the task could be easier solved with more discriminable features.

In a short timescale, the tuning landscape is also dynamic: starting with a broader tuning lighting up the whole image manifold and then narrowing down to a more specific peak on the manifold. It will be interesting to know what principles govern the dynamic of the peak.

VII.2.3 Tuning Landscapes and Cortical Maps

Visual neurons are packed in visual cortex, and many maps have been discovered in primary visual cortex (V1). In the language of tuning landscape, cortical map can be regarded as the fact that the peak of tuning landscape changed smoothly in a certain feature space when we traverse the cortical surface (for example, orientation, spatial frequency space for V1).

So, for higher visual cortices, we have been able to extract these peaks from a population of neurons across the cortical surface. It's natural to ask if these peaks are organized as a cortical map like V1. We have showed evidence in the IT cortex of one monkey, that these tuning maps were more correlated for nearby electrodes ($400\mu m$ distance between channels) than farther away electrodes (Figure IV-14). Recently, (Willeke *et al.*, 2023) also provided evidence that in primate V4, the maximal exciting images (i.e. prototype) for units recorded by the same of penetration linear probe (40 μm distance between channel) were more similar to each other. This consistent evidence suggests that the neuronal tuning landscape might change smoothly along the cortical surface in V4 and IT.

In future works, it will be interesting to visualize maximally activating images along the cortical surface in higher resolution and discover what feature parameters underlie the change of these tuning maps. The result shall reveal the organizing principle for higher visual cortex, equivalent to orientation and spatial frequency for V1.

VII.2.4 From Single Neuron Landscape to Population Representation

In this thesis, the tuning landscape view is developed for a single unit or functional column, and most analyses done in Chapter IV, Chapter V, Chapter VI of the thesis were based on tuning of a single neuron, i.e. scalar response to an image. Since neurons work in population to encode

concepts, it's a necessity to extend this view to a neuronal population and connect it with the population views of the neural code.

In the popular framework to understand visual representati on, the representations are regarded as vectors living in a neural representation space (Chung, Lee and Sompolinsky, 2018; Cohen *et al.*, 2020; Chung and Abbott, 2021). Then we can consider the separability or decodability of categories in these representations. Though this geometric picture is nice and could illuminate many aspects of visual representation, it occludes some other aspects. In this vector space, each axis corresponds to the activation function of a neuron, which is, a tuning landscape in our picture. The visual attributes corresponding to the peaks of the tuning landscape are in some sense the meaning of the axis in the population neural space. Further, in our picture, if the tuning landscapes of two neurons overlap, then there exist images that can activate them both, if not then there will be none. In the population code picture, this translates to whether there are points locating on the plane spanned by the two axes except for those points living on the axes. So, these two pictures are dual to each other in some sense, and exploring their relationship will be a worthy topic. Recently, the deep learning community is also becoming interested in understanding the representation in neural networks as landscapes on image space (Zavatone-Veth *et al.*, 2023). Hopefully this confluence of interest will catalyze research in both fields.

One experiment for this will be to use the evolutionary optimizer to control population patterns similar to (Bashivan, Kar and DiCarlo, 2019). By driving the neural population vector in different directions in the neural space, we may be able to see how much the tuning landscapes of the neurons overlap each other.

VII.2.5 Visual Cortex as Inverse Generative Model

Computer graphics is the process of rendering 3d physical scene into 2d images, then in some sense, visual system can be regarded as the process of inversing graphics engine. The visual system took in the 2d retinal image and analyzed it to figure out the 3d scene that rendered into it. More generally, visual systems may be regarded as an inverse generative model of image, which took in the image and spoke out the latent variables that generate this image.

There are previous works (Yildirim *et al.*, 2020) that have explicitly tested this idea for face patches, showing that neural networks trained to inverse the generative model of faces are better predictive models for the face patch neurons. We think this idea may not have been tested for higher visual cortices more generally. It's conceivable that different visual cortices are estimating the latent variables for different generative models of image, which is consistent with our findings in Chapter VI. For example, it's possible that the mid-level visual areas are more related to the latent space of DeePSim, while the high-level visual areas are more related to the latent space of BigGAN.

VII.2.6 Alignment with Generative Model and Implications for Brain Machine Interface

One step further, if some cortices are perfectly aligned with the latent space of a generative model, then it should be simple to translate the neural representation to the latent, and then visualize the content of the neural representation. Finding aligned representation between brain and machine will boost the application of brain decoding and brain machine interface.

Many recent works have demonstrated the power of using brain representations and advanced generative models to decode visual cortex. In this year, numerous works have combined brain signal fMRI, with (latent) diffusion models to generate or decode the image from

the brain (Chen *et al.*, 2023; Lan *et al.*, 2023; Lu *et al.*, 2023; Ozcelik and VanRullen, 2023; Scotti *et al.*, 2023; Takagi and Nishimoto, 2023). This surge of interest at least showed that this is a promising intersection. We think building and training generative models that are well aligned with the cortical regions will be a promising path to brain decoding.

VII.3 Future Technical Developments

Aside from the open scientific questions, there are some technical problems that may be interesting to solve.

VII.3.1 Generalization to other generative image models

Generative adversarial network is not the only kind of generative model in the world. As we reviewed in Section I.4, Variational Autoencoder, Normalizing Flow and Diffusion are also prominent choices as generative models for image. Recent years, due to the high image quality and steerability, Diffusion generative models have been a popular topic in the generative modelling community (Dhariwal and Nichol, 2021). It's natural to think about the generalization of the *Evolution* paradigm to diffusion models. So, what are the benefits? Similar to BigGAN, the conditional generative model e.g. Stable Diffusion, can be easily prompted into sampling from different conditional distributions ("Landscape drawing in the style of Van Gogh" vs "Cute cat on a sunny day, Leica photograph"). But with natural language prompt, these models can be controlled in a much more fine-grained fashion than BigGAN. This nice feature provides an efficient way to test the alignment of neural tuning with different kinds of generative image manifolds. Further, as tuning functions of neurons are differentially aligned with different image manifolds (Chapter VI), one single generative model may not suit all visual areas. Most ideally, each neuron should define the generative model of its own, where highly activating images are sampled with higher probability. Thus, it may be interesting to let the *neuron guide the discovery*

of proper conditional signal / prompts for a conditional image generative model, and per the generalizability of this model, all the samples from this model are diverse but highly activating images for the neuron.

In this sense, each neuron defines a distribution of its own, assigning high probability to higher activation images and lower probability to lower activation images, similar to an energy-based model (Du and Mordatch, 2019; Grathwohl *et al.*, 2019). Then it will be ideal to fit this distribution with some generative models and discover the trend of this distribution as a function of cortical area.

VII.3.2 Methods to dissect features for Evolution experiments.

Evolution experiments result in a large sequence of image and response pairs. How to best leverage this dataset to understand neuronal selectivity is still an open question. As we have found in **Figure** VI-6, using only one exemplar image to represent the neuronal selectivity is noisy and wasteful. It's better to synthesize all observed responses and images into a model of the neuron and then dissect the model to understand the neuron and population (as in (Yamane *et al.*, 2008a)). In this thesis and beyond (Wang and Carlos R. Ponce, 2022a), we have developed ways of modelling neurons and dissecting features based on the model. It will be interesting and useful to see better methods for building and fitting these models.

Bibliography

Abbasi-Asl, R. *et al.* (2018) 'The DeepTune framework for modeling and characterizing neurons in visual cortex area V4', *bioRxiv*, p. 465534. Available at: https://doi.org/10.1101/465534.

Adrian, E.D. (1919) 'The response of human sensory nerves to currents of short duration', *The Journal of Physiology*, 53(1–2), pp. 70–85.

Aggelopoulos, N.C., Franco, L. and Rolls, E.T. (2005) 'Object perception in natural scenes: encoding by inferior temporal cortex simultaneously recorded neurons', *Journal of Neurophysiology*, 93(3), pp. 1342–1357. Available at: https://doi.org/10.1152/jn.00553.2004.

Akimoto, Y. and Hansen, N. (2020) 'Diagonal Acceleration for Covariance Matrix Adaptation Evolution Strategies', *Evolutionary Computation*, 28(3), pp. 405–435. Available at: https://doi.org/10.1162/EVCO_A_00260.

Amari, S. (2016) *Information geometry and its applications*. Springer.

Amari, S.-I. (1998) 'Natural gradient works efficiently in learning', *Neural computation*, 10(2), pp. 251–276.

Anselmi, F. *et al.* (2015) 'Deep Convolutional Networks are Hierarchical Kernel Machines'. arXiv. Available at: https://doi.org/10.48550/arXiv.1508.01084.

Antognini, J.M. and Sohl-Dickstein, J. (2018) 'PCA of high dimensional random walks with comparison to neural network training', *arXiv preprint arXiv:1806.08805* [Preprint].

Anzai, A., Peng, X. and Van Essen, D.C. (2007) 'Neurons in monkey visual area V2 encode combinations of orientations', *Nature Neuroscience*, 10(10), pp. 1313–1321. Available at: https://doi.org/10.1038/nn1975.

Aronov, D., Nevers, R. and Tank, D.W. (2017) 'Mapping of a non-spatial dimension by the hippocampal–entorhinal circuit', *Nature 2017 543:7647*, 543(7647), pp. 719–722. Available at: https://doi.org/10.1038/nature21692.

Arvanitidis, G., Hansen, L.K. and Hauberg, S. (2017) 'Latent space oddity: on the curvature of deep generative models', *arXiv preprint arXiv:1710.11379* [Preprint].

Audin, M., Damian, M. and Erné, R. (2014) *Morse theory and Floer homology*. Springer.

Baddeley, R. *et al.* (1997) 'Responses of neurons in primary and inferior temporal visual cortices to natural scenes', *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 264(1389), pp. 1775–1783. Available at: https://doi.org/10.1098/rspb.1997.0246.

Barlow H. (1961) 'Possible principles underlying the transformation of sensory messages', *Sensory Communication*, pp. 217–234.

Barlow, H.B. (1953) ‘Summation and inhibition in the frog’s retina’, *The Journal of Physiology*, 119(1), pp. 69–88.

Bashivan, P., Kar, K. and DiCarlo, J.J. (2019) ‘Neural population control via deep image synthesis.’, *Science (New York, N.Y.)*, 364(6439), p. eaav9436. Available at: https://doi.org/10.1126/science.aav9436.

Benda, J. *et al.* (2007) ‘From response to stimulus: adaptive sampling in sensory physiology’, *Current Opinion in Neurobiology*, 17(4), pp. 430–436. Available at: https://doi.org/10.1016/j.conb.2007.07.009.

Benichoux, V. *et al.* (2017) ‘Representation of multidimensional stimuli: Quantifying the most informative stimulus dimension from neural responses’, *Journal of Neuroscience*, 37(31), pp. 7332–7346. Available at: https://doi.org/10.1523/JNEUROSCI.0318-17.2017.

Bondar, I.V. *et al.* (2009) ‘Long-Term Stability of Visual Pattern Selective Responses of Monkey Temporal Lobe Neurons’, *PLOS ONE*, 4(12), p. e8222. Available at: https://doi.org/10.1371/journal.pone.0008222.

Bordelon, B., Paulson, J.A. and Pehlevan, C. (2021) ‘Population Codes Enable Learning from Few Examples By Shaping Inductive Bias’, *bioRxiv*, p. 2021.03.30.437743. Available at: https://doi.org/10.1101/2021.03.30.437743.

Boussaoud, D., Desimone, R. and Ungerleider, L.G. (1991) ‘Visual topography of area TEO in the macaque.’, *The Journal of comparative neurology*, 306(4), pp. 554–75. Available at: https://doi.org/10.1002/cne.903060403.

Brincat, S.L. and Connor, C.E. (2006) ‘Dynamic shape synthesis in posterior inferotemporal cortex’, *Neuron*, 49(1), pp. 17–24.

Brock, A., Donahue, J. and Simonyan, K. (2018) ‘Large Scale GAN Training for High Fidelity Natural Image Synthesis’, *7th International Conference on Learning Representations, ICLR 2019* [Preprint]. Available at: http://arxiv.org/abs/1809.11096 (Accessed: 21 June 2021).

Butts, D.A. and Goldman, M.S. (2006) ‘Tuning curves, neuronal variability, and sensory coding’, *PLoS Biology*, 4(4), pp. 639–646. Available at: https://doi.org/10.1371/journal.pbio.0040092.

Campbell, F.W. *et al.* (1968) ‘The angular selectivity of visual cortical cells to moving gratings’, *The Journal of Physiology*, 198(1), pp. 237–250. Available at: https://doi.org/10.1113/jphysiol.1968.sp008604.

Carlson, E.T. *et al.* (2011) ‘A sparse object coding scheme in area V4’, *Current biology : CB*, 21(4), pp. 288–293. Available at: https://doi.org/10.1016/J.CUB.2011.01.013.

Celebrini, S. *et al.* (1993) ‘Dynamics of orientation coding in area V1 of the awake primate’, *Visual neuroscience*, 10(5), pp. 811–825.

Chen, N. *et al.* (2018) ‘Metrics for deep generative models’, in *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 1540–1550.

Chen, Z. *et al.* (2023) ‘Seeing Beyond the Brain: Conditional Diffusion Model with Sparse Masked Modeling for Vision Decoding’. arXiv. Available at: https://doi.org/10.48550/arXiv.2211.06956.

Chen, Z. and Zhang, H. (2019) ‘Learning implicit fields for generative shape modeling’, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5939–5948.

Chiu, C.-H. *et al.* (2020) ‘Human-in-the-Loop Differential Subspace Search in High-Dimensional Latent Space’, *ACM Trans. Graph.*, 39(4). Available at: https://doi.org/10.1145/3386569.3392409.

Chung, S. and Abbott, L.F. (2021) ‘Neural population geometry: An approach for understanding biological and artificial neural networks’, *Current Opinion in Neurobiology*, 70, pp. 137–144. Available at: https://doi.org/10.1016/j.conb.2021.10.010.

Chung, S., Lee, D.D. and Sompolinsky, H. (2018) ‘Classification and Geometry of General Perceptual Manifolds’, *Physical Review X*, 8(3), p. 031003. Available at: https://doi.org/10.1103/PhysRevX.8.031003.

Cohen, U. *et al.* (2020) ‘Separability and geometry of object manifolds in deep neural networks’, *Nature Communications*, 11(1), p. 746. Available at: https://doi.org/10.1038/s41467-020-14578-5.

Czanner, G. *et al.* (2015) ‘Measuring the signal-to-noise ratio of a neuron’, *Proceedings of the National Academy of Sciences of the United States of America*, 112(23), pp. 7141–7146. Available at: https://doi.org/10.1073/pnas.1505545112.

Dayan, P. and Abbott, L. (2005) *Theoretical Neuroscience*, *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems (Computational Neuroscience)*. Edited by L.F. Abbott. MIT Press.

Deng, J. *et al.* (2009) ‘Imagenet: A large-scale hierarchical image database’, in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255.

Descartes, R., Miller, V.R. and Miller, R.P. (1988) *Principles of Philosophy*. Wilder Publications.

Desimone, R. *et al.* (1984) ‘Stimulus-selective properties of inferior temporal neurons in the macaque.’, *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 4(8), pp. 2051–62.

Desimone, R. and Schein, S.J. (1987) ‘Visual properties of neurons in area V4 of the macaque: sensitivity to stimulus form’, *Journal of neurophysiology*, 57(3), pp. 835–868.

Dhariwal, P. and Nichol, A. (2021) *Diffusion Models Beat GANs on Image Synthesis*, *arXiv.org*. Available at: https://arxiv.org/abs/2105.05233v4 (Accessed: 10 August 2023).

DiMattina, C. and Zhang, K. (2013) 'Adaptive stimulus optimization for sensory systems neuroscience', *Frontiers in Neural Circuits*, 0(JUNE), p. 101. Available at: https://doi.org/10.3389/FNCIR.2013.00101/BIBTEX.

Dinh, L., Krueger, D. and Bengio, Y. (2015) 'NICE: Non-linear Independent Components Estimation'. arXiv. Available at: https://doi.org/10.48550/arXiv.1410.8516.

Dinh, L., Sohl-Dickstein, J. and Bengio, S. (2017) 'Density estimation using Real NVP'. arXiv. Available at: https://doi.org/10.48550/arXiv.1605.08803.

Donahue, J. and Simonyan, K. (2019) 'Large scale adversarial representation learning', in *Advances in Neural Information Processing Systems*, pp. 10542–10552.

Dosovitskiy, A. and Brox, T. (2016a) 'Generating Images with Perceptual Similarity Metrics based on Deep Networks'.

Dosovitskiy, A. and Brox, T. (2016b) 'Generating images with perceptual similarity metrics based on deep networks', in *Advances in neural information processing systems*, pp. 658–666.

Dragoi, V., Sharma, J. and Sur, M. (2000) 'Adaptation-induced plasticity of orientation tuning in adult visual cortex', *Neuron*, 28(1), pp. 287–298. Available at: https://doi.org/10.1016/s0896-6273(00)00103-3.

Driscoll, L.N., Duncker, L. and Harvey, C.D. (2022) 'Representational drift: Emerging theories for continual learning and experimental future directions', *Current Opinion in Neurobiology*, 76, p. 102609. Available at: https://doi.org/10.1016/j.conb.2022.102609.

Du, Y. and Mordatch, I. (2019) 'Implicit Generation and Modeling with Energy Based Models', *Advances in Neural Information Processing Systems*, 32. Available at: https://sites.google.com/view/igebm (Accessed: 30 August 2021).

Edelman, S. (1999) *Representation and Recognition in Vision*. The MIT Press. Available at: https://doi.org/10.7551/mitpress/5890.001.0001.

Engstrom, L. *et al.* (2019) 'Robustness (Python Library)'. Available at: https://github.com/MadryLab/robustness.

Enroth-Cugell, C. *et al.* (1983) 'Spatio-temporal interactions in cat retinal ganglion cells showing linear spatial summation.', *J. Physiol.*, 341, pp. 279–307.

Feather, J. *et al.* (2022) 'Model metamers illuminate divergences between biological and artificial neural networks', *bioRxiv* [Preprint].

Felleman, D.J. and Van Essen, D.C. (1991) 'Distributed hierarchical processing in the primate cerebral cortex.', *Cerebral cortex (New York, N.Y. : 1991)*, 1(1), pp. 1–47. Available at: https://doi.org/10.1093/cercor/1.1.1.

Field, D.J. (1987) 'Relations between the statistics of natural images and the response properties of cortical cells', *Journal of the Optical Society of America A*, 4(12), p. 2379. Available at: https://doi.org/10.1364/josaa.4.002379.

Földiák, P. (2001) 'Stimulus optimisation in primary visual cortex', *Neurocomputing*, 38–40, pp. 1217–1222. Available at: https://doi.org/10.1016/S0925-2312(01)00570-7.

Freedman, D.J. *et al.* (2003) 'A Comparison of Primate Prefrontal and Inferior Temporal Cortices during Visual Categorization', *Journal of Neuroscience*, 23(12), pp. 5235–5246. Available at: https://doi.org/10.1523/JNEUROSCI.23-12-05235.2003.

Freeman, J. *et al.* (2013) 'A functional and perceptual signature of the second visual area in primates', *Nature Neuroscience*, 16(7), pp. 974–981. Available at: https://doi.org/10.1038/nn.3402.

Freeman, J. and Simoncelli, E.P. (2011) 'Metamers of the ventral stream', *Nature Neuroscience*, 14(9), pp. 1195–1204. Available at: https://doi.org/10.1038/nn.2889.

Gallant, J.L., Braun, J. and Van Essen, D.C. (1993) 'Selectivity for Polar, Hyperbolic, and Cartesian Gratings in Macaque Visual Cortex', *Science*, 259(5091), pp. 100–103. Available at: https://doi.org/10.1126/science.8418487.

Gallivan, J.P. and Goodale, M.A. (2018) 'The dorsal "action" pathway', *Handbook of Clinical Neurology*, 151, pp. 449–466. Available at: https://doi.org/10.1016/B978-0-444-63622-5.00023-1.

Gattass, R., Gross, C.G. and Sandell, J.H. (1981) 'Visual topography of V2 in the macaque', *Journal of Comparative Neurology*, 201(4), pp. 519–539. Available at: https://doi.org/10.1002/cne.902010405.

Gattass, R., Sousa, A.P. and Gross, C.G. (1988) 'Visuotopic organization and extent of V3 and V4 of the macaque', *Journal of Neuroscience*, 8(6), pp. 1831–1845. Available at: https://doi.org/10.1523/JNEUROSCI.08-06-01831.1988.

Ghorbani, B., Krishnan, S. and Xiao, Y. (2019) 'An Investigation into Neural Net Optimization via Hessian Eigenvalue Density', in *International Conference on Machine Learning*, pp. 2232–2241.

Gollisch, T. *et al.* (2002) 'Energy integration describes sound-intensity coding in an insect auditory system', *Journal of Neuroscience*, 22(23), pp. 10434–10448.

Gollisch, T. and Herz, A.V. (2012) 'The iso-response method: measuring neuronal stimulus integration with closed-loop experiments', *Frontiers in neural circuits*, 6, p. 104.

Goodale, M.A. and Milner, A.D. (1992) ‘Separate visual pathways for perception and action’, *Trends in Neurosciences*, 15(1), pp. 20–25. Available at: https://doi.org/10.1016/0166-2236(92)90344-8.

Goodale, M.A. and Westwood, D.A. (2004) ‘An evolving view of duplex vision: separate but interacting cortical pathways for perception and action’, *Current Opinion in Neurobiology*, 14(2), pp. 203–211. Available at: https://doi.org/10.1016/j.conb.2004.03.002.

Goodfellow, I. *et al.* (2020) ‘Generative adversarial networks’, *Communications of the ACM*, 63(11), pp. 139–144. Available at: https://doi.org/10.1145/3422622.

Grathwohl, W. *et al.* (2019) ‘Your Classifier is Secretly an Energy Based Model and You Should Treat it Like One’. Available at: https://doi.org/10.48550/arxiv.1912.03263.

Gross, C.G. (1994) ‘How inferior temporal cortex became a visual area’, *Cerebral Cortex*, 4(5), pp. 455–469. Available at: https://doi.org/10.1093/cercor/4.5.455.

Gross, C.G., Bender, D.B. and Rocha-Miranda, C.E. (1969) ‘Visual receptive fields of neurons in inferotemporal cortex of the monkey’, *Science (New York, N.Y.)*, 166(3910), pp. 1303–1306. Available at: https://doi.org/10.1126/science.166.3910.1303.

Gross, C.G., Rocha-Miranda, C.E. and Bender, D.B. (1972) ‘Visual properties of neurons in inferotemporal cortex of the Macaque.’, *Journal of Neurophysiology*, 35(1), pp. 96–111. Available at: https://doi.org/10.1152/jn.1972.35.1.96.

Guo, C. *et al.* (2022) ‘Adversarially trained neural representations may already be as robust as corresponding biological neural representations’. arXiv. Available at: https://doi.org/10.48550/arXiv.2206.11228.

Hansen, N. (2016) ‘The CMA Evolution Strategy: A Tutorial’. Available at: http://arxiv.org/abs/1604.00772 (Accessed: 18 June 2021).

Hansen, N. and Auger, A. (2014) ‘Principled Design of Continuous Stochastic Search: From Theory to Practice’, pp. 145–180. Available at: https://doi.org/10.1007/978-3-642-33206-7_8.

Hansen, N. and Ostermeier, A. (2001) ‘Completely derandomized self-adaptation in evolution strategies.’, *Evolutionary computation*, 9(2), pp. 159–195. Available at: https://doi.org/10.1162/106365601750190398.

Härkönen, E. *et al.* (2020) ‘GANSpace: Discovering Interpretable GAN Controls’, *arXiv preprint arXiv:2004.02546* [Preprint].

Hatcher, A. (2000) *Algebraic topology*. Cambridge: Cambridge Univ. Press. Available at: https://cds.cern.ch/record/478079.

He, K. *et al.* (2016) ‘Deep residual learning for image recognition’, in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, pp. 770–778. Available at: https://doi.org/10.1109/CVPR.2016.90.

Hegdé, J. (2008) ‘Time course of visual perception: Coarse-to-fine processing and beyond’, *Progress in Neurobiology*, 84(4), pp. 405–439. Available at: https://doi.org/10.1016/j.pneurobio.2007.09.001.

Hegdé, J. and Van Essen, D.C. (2004) ‘Temporal dynamics of shape analysis in macaque visual area V2’, *Journal of neurophysiology*, 92(5), pp. 3030–3042.

Hegdé, J. and Van Essen, D.C. (2006) ‘Temporal dynamics of 2D and 3D shape representation in macaque visual area V4’, *Visual neuroscience*, 23(5), pp. 749–763.

Hegdé, J. and Van Essen, D.C. (2007) ‘A comparative study of shape representation in macaque visual areas v2 and v4.’, *Cerebral cortex (New York, N.Y. : 1991)*, 17(5), pp. 1100–16. Available at: https://doi.org/10.1093/cercor/bhl020.

Horwitz, G.D. and Hass, C.A. (2012) ‘Nonlinear analysis of macaque V1 color tuning reveals cardinal directions for cortical color processing’, *Nature neuroscience*, 15(6), pp. 913–919. Available at: https://doi.org/10.1038/NN.3105.

Huang, G. *et al.* (2016) ‘Densely Connected Convolutional Networks’, *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-January, pp. 2261–2269.

Hubel, D. and Livingstone, M.S. (1987) ‘Segregation of form, color, and stereopsis in primate area 18’, *J. Neurosci.*, 7, pp. 3378–3415.

Hubel, D.H. and Wiesel, T.N. (1959) ‘Receptive fields of single neurones in the cat’s striate cortex’, *The Journal of Physiology*, 148(3), pp. 574–591.

Hubel, D H and Wiesel, T.N. (1962) ‘Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex.’, *The Journal of physiology*, 160, pp. 106–54.

Hubel, David H and Wiesel, T.N. (1962) ‘Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex’, *The Journal of physiology*, 160(1), p. 106.

Hubel, D.H. and Wiesel, T.N. (1968) ‘Receptive fields and functional architecture of monkey striate cortex’, *The Journal of Physiology*, 195(1), pp. 215–243.

Huh, M. *et al.* (2020) ‘Transforming and Projecting Images into Class-conditional Generative Networks’, *arXiv preprint arXiv:2005.01703* [Preprint].

Hung, C.C., Carlson, E.T. and Connor, C.E. (2012) ‘Medial Axis Shape Coding in Macaque Inferotemporal Cortex’, *Neuron*, 74(6), pp. 1099–1113. Available at: https://doi.org/10.1016/J.NEURON.2012.04.029.

Hung, C.P. *et al.* (2005) ‘Fast readout of object identity from macaque inferior temporal cortex’, *Science (New York, N.Y.)*, 310(5749), pp. 863–866. Available at: https://doi.org/10.1126/science.1117593.

Hwang, J. (no date) *MonkeyLogic 2*.

Ito, M. *et al.* (1995) 'Size and position invariance of neuronal responses in monkey inferotemporal cortex', *Journal of neurophysiology*, 73(1), pp. 218–226. Available at: https://doi.org/10.1152/JN.1995.73.1.218.

James Munkres (2000) *Topology, 2nd Edition*. Pearson.

Ji, Z. and Telgarsky, M. (2018) 'Gradient descent aligns the layers of deep linear networks', *7th International Conference on Learning Representations, ICLR 2019* [Preprint]. Available at: https://arxiv.org/abs/1810.02032v2 (Accessed: 10 January 2022).

Kant, I. (1908) 'Critique of pure reason. 1781', *Modern Classical Philosophers, Cambridge, MA: Houghton Mifflin*, pp. 370–456.

Kar, K. *et al.* (2019) 'Evidence that recurrent circuits are critical to the ventral stream's execution of core object recognition behavior', *Nature Neuroscience*, 22(6), pp. 974–983. Available at: https://doi.org/10.1038/s41593-019-0392-5.

Karras, T. *et al.* (2017) 'Progressive Growing of GANs for Improved Quality, Stability, and Variation', *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings* [Preprint]. Available at: https://arxiv.org/abs/1710.10196v3 (Accessed: 7 July 2021).

Karras, T. *et al.* (2020) 'Analyzing and improving the image quality of stylegan', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8110–8119.

Karras, T., Laine, S. and Aila, T. (2019) 'A style-based generator architecture for generative adversarial networks', in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, pp. 4396–4405. Available at: https://doi.org/10.1109/CVPR.2019.00453.

Kay, K.N. and Yeatman, J.D. (2017) 'Bottom-up and top-down computations in word- and face-selective cortex', *eLife*, 6. Available at: https://doi.org/10.7554/ELIFE.22341.

Kilcher, Y., Lucchi, A. and Hofmann, T. (2017) 'Semantic Interpolation in Implicit Models', *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings* [Preprint]. Available at: https://doi.org/10.48550/arxiv.1710.11381.

Kingma, D.P. and Ba, J.L. (2015) 'Adam: A method for stochastic optimization', in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR. Available at: https://arxiv.org/abs/1412.6980v9 (Accessed: 22 June 2021).

Kingma, D.P. and Dhariwal, P. (2018) 'Glow: Generative Flow with Invertible 1x1 Convolutions'. arXiv. Available at: https://doi.org/10.48550/arXiv.1807.03039.

Kingma, D.P. and Welling, M. (2022) ‘Auto-Encoding Variational Bayes’. arXiv. Available at: https://doi.org/10.48550/arXiv.1312.6114.

Klindt, D.A. *et al.* (2017) ‘Neural system identification for large populations separating “what” and “where”’, *Advances in Neural Information Processing Systems*, 2017-Decem, pp. 3507–3517. Available at: https://doi.org/10.48550/arxiv.1711.02653.

Kobatake, E. and Tanaka, K. (1994) ‘Neuronal selectivities to complex object features in the ventral visual pathway of the macaque cerebral cortex.’, *Journal of neurophysiology*, 71(3), pp. 856–67.

Kornblith, S. *et al.* (2013) ‘A Network for Scene Processing in the Macaque Temporal Lobe’, *Neuron*, 79(4), pp. 766–781. Available at: https://doi.org/10.1016/j.neuron.2013.06.015.

Kornblith, S. *et al.* (2019) ‘Similarity of neural network representations revisited’, *arXiv preprint arXiv:1905.00414* [Preprint].

Koyano, K.W. *et al.* (2022) ‘Cascades of neuronal plasticity in the macaque visual cortex’, *bioRxiv*, pp. 2022–09.

Kreiman, G. *et al.* (2006) ‘Object Selectivity of Local Field Potentials and Spikes in the Macaque Inferior Temporal Cortex’, *Neuron*, 49(3), pp. 433–445. Available at: https://doi.org/10.1016/j.neuron.2005.12.019.

Kriegeskorte, N. and Wei, X.X. (2021) ‘Neural tuning and representational geometry’, *Nature Reviews Neuroscience 2021 22:11*, 22(11), pp. 703–718. Available at: https://doi.org/10.1038/s41583-021-00502-3.

Krizhevsky, A., Sutskever, I. and Hinton, G.E. (2012) *ImageNet Classification with Deep Convolutional Neural Networks*, *Advances in Neural Information Processing Systems*. Available at: http://code.google.com/p/cuda-convnet/ (Accessed: 22 June 2021).

Kubilius, J. *et al.* (2018) ‘CORnet: Modeling the Neural Mechanisms of Core Object Recognition’, *bioRxiv*, p. 408385. Available at: https://doi.org/10.1101/408385.

Kuffler, S.W. (1953) ‘Discharge patterns and functional organization of mammalian retina’, *Journal of Neurophysiology*, 16(1), pp. 37–68. Available at: https://doi.org/10.1152/jn.1953.16.1.37.

Lamme, V.A., Rodriguez-Rodriguez, V. and Spekreijse, H. (1999) ‘Separate processing dynamics for texture elements, boundaries and surfaces in primary visual cortex of the macaque monkey’, *Cerebral cortex*, 9(4), pp. 406–413.

Lamme, V.A.F. and Roelfsema, P.R. (2000) ‘The distinct modes of vision offered by feedforward and recurrent processing’, *Trends in Neurosciences*, 23(11), pp. 571–579. Available at: https://doi.org/10.1016/S0166-2236(00)01657-X.

Lan, Y.-T. *et al.* (2023) 'Seeing through the Brain: Image Reconstruction of Visual Perception from Human Brain Signals'. arXiv. Available at: https://doi.org/10.48550/arXiv.2308.02510.

Lee, A.B., Mumford, D. and Huang, J. (2001) 'Occlusion Models for Natural Images: A Statistical Study of a Scale-Invariant Dead Leaves Model', *International Journal of Computer Vision*, 41(1), pp. 35–59. Available at: https://doi.org/10.1023/A:1011109015675.

Lee, D.D. and Seung, H.S. (1999) 'Learning the parts of objects by non-negative matrix factorization', *Nature*, 401(6755), pp. 788–791. Available at: https://doi.org/10.1038/44565.

Lee, D.D. and Seung, H.S. (2001) 'Algorithms for Non-negative Matrix Factorization'. Available at: http://papers.nips.cc/paper/1861-algorithms-for-non-negative-matrix-factorization (Accessed: 20 June 2019).

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

Leopold, D.A. *et al.* (2001) 'Prototype-referenced shape encoding revealed by high-level aftereffects', *Nature Neuroscience 2001 4:1*, 4(1), pp. 89–94. Available at: https://doi.org/10.1038/82947.

Levitt, J.B., Kiper, D.C. and Movshon, J.A. (1994) 'Receptive fields and functional architecture of macaque V2', *Journal of Neurophysiology*, 71(6), pp. 2517–2542. Available at: https://doi.org/10.1152/jn.1994.71.6.2517.

Lin, T.Y. *et al.* (2014) 'Microsoft COCO: Common objects in context', in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer Verlag, pp. 740–755. Available at: https://doi.org/10.1007/978-3-319-10602-1_48.

Lindsay, G.W. (2020) 'Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future', *Journal of Cognitive Neuroscience*, pp. 1–15. Available at: https://doi.org/10.1162/jocn_a_01544.

Logothetis, N.K., Pauls, J. and Poggio, T. (1995) 'Shape representation in the inferior temporal cortex of monkeys.', *Current biology : CB*, 5(5), pp. 552–63.

Long, F.X. *et al.* (2022) 'Learning the characteristics of engineering optimization problems with applications in automotive crash', in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1227–1236.

Lorch, E. (2016) 'Visualizing deep network training trajectories with pca', in *ICML Workshop on Visualization for Deep Learning*.

Lorensen, W.E. and Cline, H.E. (1987) 'Marching cubes: A high resolution 3D surface construction algorithm', *ACM siggraph computer graphics*, 21(4), pp. 163–169.

Loshchilov, I. (2014) ‘A computationally efficient limited memory CMA-ES for large scale optimization’, *GECCO 2014 - Proceedings of the 2014 Genetic and Evolutionary Computation Conference*, pp. 397–404. Available at: https://doi.org/10.1145/2576768.2598294.

Loshchilov, I. (2017) ‘LM-CMA: An Alternative to L-BFGS for Large-Scale Black Box Optimization’, *Evolutionary Computation*, 25(1), pp. 143–171. Available at: https://doi.org/10.1162/EVCO_a_00168.

Lu, Y. *et al.* (2023) ‘MindDiffuser: Controlled Image Reconstruction from Human Brain Activity with Semantic and Structural Diffusion’. arXiv. Available at: https://doi.org/10.48550/arXiv.2308.04249.

Lurz, K.-K. *et al.* (2021) ‘Generalization in data-driven models of primary visual cortex’. bioRxiv, p. 2020.10.05.326256. Available at: https://doi.org/10.1101/2020.10.05.326256.

Madry, A. *et al.* (2017) ‘Towards Deep Learning Models Resistant to Adversarial Attacks’, *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings* [Preprint]. Available at: http://arxiv.org/abs/1706.06083 (Accessed: 22 June 2021).

Majaj, N.J. *et al.* (2015) ‘Simple Learned Weighted Sums of Inferior Temporal Neuronal Firing Rates Accurately Predict Human Core Object Recognition Performance’, *Journal of Neuroscience*, 35(39). Available at: https://doi.org/10.1523/JNEUROSCI.5181-14.2015.

Markov, N.T. *et al.* (2014) ‘Anatomy of hierarchy: feedforward and feedback pathways in macaque visual cortex’, *Journal of Comparative Neurology*, 522(1), pp. 225–259. Available at: https://doi.org/10.1002/cne.23458.

Maruyama, M., Girosi, F. and Poggio, T. (1992) ‘A Connection Between GRBF and MLP’. Available at: https://dspace.mit.edu/handle/1721.1/6566 (Accessed: 7 September 2023).

Masset, P., Qin, S. and Zavatone-Veth, J.A. (2022) ‘Drifting neuronal representations: Bug or feature?’, *Biological Cybernetics*, 116(3), pp. 253–266. Available at: https://doi.org/10.1007/s00422-021-00916-3.

Matsumoto, N. *et al.* (2005) ‘Neuronal mechanisms encoding global-to-fine information in inferior-temporal cortex’, *Journal of Computational Neuroscience*, 18(1), pp. 85–103. Available at: https://doi.org/10.1007/s10827-005-5476-4.

Maunsell, J.H.R. and Van Essen, D.C. (1983) ‘Functional properties of neurons in middle temporal visual area of the macaque monkey. I. Selectivity for stimulus direction, speed, and orientation’, *https://doi.org/10.1152/jn.1983.49.5.1127*, 49(5), pp. 1127–1147. Available at: https://doi.org/10.1152/JN.1983.49.5.1127.

McInnes, L. *et al.* (2018) ‘UMAP: Uniform Manifold Approximation and Projection’, *Journal of Open Source Software*, 3(29), p. 861. Available at: https://doi.org/10.21105/joss.00861.

McNaughton, B.L., Barnes, C.A. and O’Keefe, J. (1983) ‘The contributions of position, direction, and velocity to single unit activity in the hippocampus of freely-moving rats’,

Experimental brain research, 52(1), pp. 41–49. Available at: https://doi.org/10.1007/BF00237147.

Milnor, J. (2016) 'Morse Theory.(AM-51), Volume 51', in *Morse Theory.(AM-51), Volume 51*. Princeton university press.

Nguyen, A. *et al.* (2016) 'Synthesizing the preferred inputs for neurons in neural networks via deep generator networks', in *Advances in neural information processing systems*, pp. 3387–3395.

O'Connor, K.N., Petkov, C.I. and Sutter, M.L. (2005) 'Adaptive stimulus optimization for auditory cortical neurons', *Journal of Neurophysiology*, 94(6), pp. 4051–4067. Available at: https://doi.org/10.1152/JN.00046.2005/ASSET/IMAGES/LARGE/Z9K0120551290010.JPEG.

O'keefe, J. *et al.* (2017) 'Auditory landscape on the cognitive map', *Nature 2017 543:7647*, 543(7647), pp. 631–632. Available at: https://doi.org/10.1038/543631a.

Olah, C., Mordvintsev, A. and Schubert, L. (2017) 'Feature Visualization', *Distill*, 2(11), p. e7. Available at: https://doi.org/10.23915/distill.00007.

Olshausen, B.A. and Field, D.J. (1996) 'Emergence of simple-cell receptive field properties by learning a sparse code for natural images', *Nature*, 381(6583), pp. 607–609. Available at: https://doi.org/10.1038/381607a0.

Osher, S., Fedkiw, R. and Piechor, K. (2004) 'Level set methods and dynamic implicit surfaces', *Appl. Mech. Rev.*, 57(3), pp. B15–B15.

Ozcelik, F. and VanRullen, R. (2023) 'Natural scene reconstruction from fMRI signals using generative latent diffusion'. arXiv. Available at: https://doi.org/10.48550/arXiv.2303.05334.

Paiton, D.M. *et al.* (2020) 'Selectivity and robustness of sparse coding networks', *Journal of vision*, 20(12), pp. 10–10.

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

Park, J.J. *et al.* (2019) 'Deepsdf: Learning continuous signed distance functions for shape representation', in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 165–174.

Pasupathy, A. and Connor, C.E. (1999) 'Responses to contour features in macaque area V4.', *Journal of neurophysiology*, 82(5), pp. 2490–502.

Pasupathy, A. and Connor, C.E. (2001) 'Shape Representation in Area V4: Position-Specific Tuning for Boundary Conformation', *Journal of Neurophysiology*, 86(5), pp. 2505–2519. Available at: https://doi.org/10.1152/jn.2001.86.5.2505.

Pasupathy, A. and Connor, C.E. (2002) ‘Population coding of shape in area V4’, *Nature Neuroscience*, 5(12), pp. 1332–1338. Available at: https://doi.org/10.1038/nn972.

Paszke, A. *et al.* (2019) *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, *Advances in Neural Information Processing Systems*.

Pedregosa, F. *et al.* (2011) ‘Scikit-learn: Machine Learning in {P}ython’, *Journal of Machine Learning Research*, 12, pp. 2825–2830.

Peebles, W. *et al.* (2020) ‘The hessian penalty: A weak prior for unsupervised disentanglement’, *arXiv preprint arXiv:2008.10599* [Preprint].

Pividori, M., Grinblat, G.L. and Uzal, L.C. (2019) ‘Exploiting GAN Internal Capacity for High-Quality Reconstruction of Natural Images’, *arXiv preprint arXiv:1911.05630* [Preprint].

Poggio, T. and Girosi, F. (1990a) ‘Networks for Approximation and Learning’, *Proceedings of the IEEE*, 78(9), pp. 1481–1497. Available at: https://doi.org/10.1109/5.58326.

Poggio, T. and Girosi, F. (1990b) ‘Networks for approximation and learning’, *Proceedings of the IEEE*, 78(9), pp. 1481–1497.

Ponce, C.R. *et al.* (2019a) ‘Evolving Images for Visual Neurons Using a Deep Generative Network Reveals Coding Principles and Neuronal Preferences’, *Cell*, 177(4), pp. 999-1009.e10. Available at: https://doi.org/10.1016/J.CELL.2019.04.005.

Ponce, C.R. *et al.* (2019b) ‘Evolving Images for Visual Neurons Using a Deep Generative Network Reveals Coding Principles and Neuronal Preferences’, *Cell*, 177(4), pp. 999-1009.e10. Available at: https://doi.org/10.1016/J.CELL.2019.04.005.

Portilla, J. and Simoncelli, E.P. (2000) ‘A Parametric Texture Model Based on Joint Statistics of Complex Wavelet Coefficients’, *International Journal of Computer Vision*, 40(1), pp. 49–70. Available at: https://doi.org/10.1023/A:1026553619983.

Qin, S. *et al.* (2023) ‘Coordinated drift of receptive fields in Hebbian/anti-Hebbian network models during noisy representation learning’, *Nature Neuroscience*, 26(2), pp. 339–349. Available at: https://doi.org/10.1038/s41593-022-01225-z.

Quiroga, R.Q. *et al.* (2005) ‘Invariant visual representation by single neurons in the human brain’, *Nature*, 435(7045), pp. 1102–1107.

Radford, A., Metz, L. and Chintala, S. (2015) ‘Unsupervised representation learning with deep convolutional generative adversarial networks’, *arXiv preprint arXiv:1511.06434* [Preprint].

Rafaely, B. (2019) ‘Sampling the Sphere’, in. Springer, Cham, pp. 59–80. Available at: https://doi.org/10.1007/978-3-319-99561-8_3.

Ramesh, A., Choi, Y. and LeCun, Y. (2018) ‘A Spectral Regularizer for Unsupervised Disentanglement’, *arXiv preprint arXiv:1812.01161* [Preprint].

Rapin, J. and Teytaud, O. (2018) ‘Nevergrad - A gradient-free optimization platform’, *GitHub repository*. GitHub. Available at: https://GitHub.com/FacebookResearch/Nevergrad.

Redmon, J. *et al.* (2016) ‘You only look once: Unified, real-time object detection’, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788.

Ringach, D.L., Hawken, M.J. and Shapley, R. (1997) ‘Dynamics of orientation tuning in macaque primary visual cortex’, *Nature*, 387(6630), pp. 281–284.

Rish, I. and Rish, I. (2001) ‘An empirical study of the naive bayes classifier’. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.330.2788 (Accessed: 22 June 2021).

Rolls, E.T., Aggelopoulos, N.C. and Zheng, F. (2003) ‘The receptive fields of inferior temporal cortex neurons in natural scenes’, *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*, 23(1), pp. 339–348. Available at: https://doi.org/10.1523/JNEUROSCI.23-01-00339.2003.

Rolls, E.T. and Tovee, M.J. (1995) ‘Sparseness of the neuronal representation of stimuli in the primate temporal visual cortex’, *Journal of neurophysiology*, 73(2), pp. 713–726. Available at: https://doi.org/10.1152/JN.1995.73.2.713.

Rombach, R. *et al.* (2021) ‘High-Resolution Image Synthesis with Latent Diffusion Models’, pp. 10674–10685. Available at: https://doi.org/10.48550/arxiv.2112.10752.

Rosch, E.H. (1973) ‘Natural categories’, *Cognitive Psychology*, 4(3), pp. 328–350. Available at: https://doi.org/10.1016/0010-0285(73)90017-0.

Rose, O., Johnson, J., Wang, B. and Ponce, Carlos R. (2021) ‘Visual prototypes in the ventral stream are attuned to complexity and gaze behavior’, *Nature Communications 2021 12:1*, 12(1), pp. 1–16. Available at: https://doi.org/10.1038/s41467-021-27027-8.

Rose, O., Johnson, J., Wang, B. and Ponce, Carlos R (2021) ‘Visual prototypes in the ventral stream are attuned to complexity and gaze behavior’, *Nature communications*, 12(1), pp. 1–16.

Rudin, W. and others (1976) *Principles of mathematical analysis*. McGraw-hill New York.

Rule, M.E., O’Leary, T. and Harvey, C.D. (2019) ‘Causes and consequences of representational drift’, *Current opinion in neurobiology*, 58, pp. 141–147. Available at: https://doi.org/10.1016/j.conb.2019.08.005.

Russakovsky, O. *et al.* (2015) ‘ImageNet Large Scale Visual Recognition Challenge’, *International Journal of Computer Vision*, 115(3), pp. 211–252. Available at: https://doi.org/10.1007/s11263-015-0816-y.

Rust, N.C. *et al.* (2005) ‘Spatiotemporal elements of macaque v1 receptive fields’, *Neuron*, 46(6), pp. 945–956.

Rust, N.C. and Dicarlo, J.J. (2012) 'Balanced Increases in Selectivity and Tolerance Produce Constant Sparseness along the Ventral Visual Stream', *Journal of Neuroscience*, 32(30), pp. 10170–10182. Available at: https://doi.org/10.1523/JNEUROSCI.6125-11.2012.

Santurkar, S. *et al.* (2019) 'Image Synthesis with a Single (Robust) Classifier', *Advances in Neural Information Processing Systems*, 32.

van der Schaaf, A. and van Hateren, J.H. (1996) 'Modelling the Power Spectra of Natural Images: Statistics and Information', *Vision Research*, 36(17), pp. 2759–2770. Available at: https://doi.org/10.1016/0042-6989(96)00002-8.

Schrimpf, M. *et al.* (2018) 'Brain-Score: Which Artificial Neural Network for Object Recognition is most Brain-Like?', *bioRxiv*, p. 407007. Available at: https://doi.org/10.1101/407007.

Scotti, P.S. *et al.* (2023) 'Reconstructing the Mind's Eye: fMRI-to-Image with Contrastive Learning and Diffusion Priors'. arXiv. Available at: https://doi.org/10.48550/arXiv.2305.18274.

Seung, H.S. and Sompolinsky, H. (1993) 'Simple models for reading neuronal population codes', *Proceedings of the National Academy of Sciences of the United States of America*, 90(22), pp. 10749–10753. Available at: https://doi.org/10.1073/pnas.90.22.10749.

Shao, H., Kumar, A. and Thomas Fletcher, P. (2018) 'The riemannian geometry of deep generative models', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 315–323.

Shen, Y. and Zhou, B. (2020) 'Closed-Form Factorization of Latent Semantics in GANs', *arXiv preprint arXiv:2007.06600* [Preprint].

Simoncelli, E.P. and Olshausen, B.A. (2001) 'Natural Image Statistics and Neural Representation', *Annual Review of Neuroscience*, 24(1), pp. 1193–1216. Available at: https://doi.org/10.1146/annurev.neuro.24.1.1193.

Simoncelli, E.P. and Portilla, J. (1998) 'Texture characterization via joint statistics of wavelet coefficient magnitudes', in *Proceedings 1998 International Conference on Image Processing. ICIP98 (Cat. No.98CB36269)*. *Proceedings 1998 International Conference on Image Processing. ICIP98 (Cat. No.98CB36269)*, pp. 62–66 vol.1. Available at: https://doi.org/10.1109/ICIP.1998.723417.

Simonyan, K. and Zisserman, A. (2015) 'Very deep convolutional networks for large-scale image recognition', in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR. Available at: http://www.robots.ox.ac.uk/ (Accessed: 22 June 2021).

Srinath, R. *et al.* (2021) 'Early Emergence of Solid Shape Coding in Natural and Deep Network Vision', *Current Biology*, 31(1), pp. 51-65.e5. Available at: https://doi.org/10.1016/J.CUB.2020.09.076.

Srivastava, A. *et al.* (2003) ‘On Advances in Statistical Modeling of Natural Images’, *Journal of Mathematical Imaging and Vision 2003 18:1*, 18(1), pp. 17–33. Available at: https://doi.org/10.1023/A:1021889010444.

Sugase, Y. *et al.* (1999) ‘Global and fine information coded by single neurons in the temporal visual cortex’, *Nature*, 400(6747), p. 869.

Takagi, Y. and Nishimoto, S. (2023) ‘Improving visual image reconstruction from human brain activity using latent diffusion models via multiple decoded inputs’. arXiv. Available at: https://doi.org/10.48550/arXiv.2306.11536.

Tanaka, K. (2003) ‘Columns for Complex Visual Object Features in the Inferotemporal Cortex: Clustering of Cells with Similar but Slightly Different Stimulus Selectivities’, *Cerebral Cortex*, 13(1), pp. 90–99. Available at: https://doi.org/10.1093/CERCOR/13.1.90.

Tavares, R.M. *et al.* (2015) ‘A Map for Social Navigation in the Human Brain’, *Neuron*, 87(1), pp. 231–243. Available at: https://doi.org/10.1016/J.NEURON.2015.06.011.

Torralba, A. and Oliva, A. (2003) ‘Statistics of natural image categories’, *Network (Bristol, England)*, 14(3), pp. 391–412.

Tsao, D.Y. *et al.* (2003) ‘Faces and objects in macaque cerebral cortex’, *Nature Neuroscience*, 6(9), pp. 989–995. Available at: https://doi.org/10.1038/nn1111.

Tsao, D.Y. *et al.* (2006) ‘A cortical region consisting entirely of face-selective cells.’, *Science (New York, N.Y.)*, 311(5761), pp. 670–4. Available at: https://doi.org/10.1126/science.1119983.

Van der Sluis, A. and Van der Vorst, H. (1987) ‘The convergence behavior of Ritz values in the presence of close eigenvalues’, *Linear Algebra and its Applications*, 88, pp. 651–694.

Van Essen, D.C., Anderson, C.H. and Felleman, D.J. (1992) ‘Information processing in the primate visual system: an integrated systems perspective’, *Science*, 255(5043), pp. 419–423.

Vaziri, S. *et al.* (2014) ‘A Channel for 3D Environmental Shape in Anterior Inferotemporal Cortex’, *Neuron*, 84(1), pp. 55–62. Available at: https://doi.org/10.1016/j.neuron.2014.08.043.

Vaziri, S. and Connor, C.E. (2016) ‘Representation of Gravity-Aligned Scene Structure in Ventral Pathway Visual Cortex’, *Current Biology*, 26(6), pp. 766–774. Available at: https://doi.org/10.1016/j.cub.2016.01.022.

Voynov, A. and Babenko, A. (2020) ‘Unsupervised Discovery of Interpretable Directions in the GAN Latent Space’, *arXiv preprint arXiv:2002.03754* [Preprint].

Walker, E.Y. *et al.* (2019) ‘Inception loops discover what excites neurons most using deep predictive models’, *Nature Neuroscience*, 22(12), pp. 2060–2065. Available at: https://doi.org/10.1038/s41593-019-0517-x.

Wallis, G. and Rolls, E.T. (1997) ‘Invariant face and object recognition in the visual system’, *Progress in Neurobiology*, 51(2), pp. 167–194. Available at: https://doi.org/10.1016/s0301-0082(96)00054-8.

Wang, B. *et al.* (2021) ‘On the use of Cortical Magnification and Saccades as Biological Proxies for Data Augmentation’. arXiv. Available at: https://doi.org/10.48550/arXiv.2112.07173.

Wang, B. and Ponce, C.R. (2020) ‘A Geometric Analysis of Deep Generative Image Models and Its Applications’, in *International Conference on Learning Representations*. Available at: http://arxiv.org/abs/2101.06006.

Wang, B. and Ponce, C.R. (2021) ‘The Geometry of Deep Generative Image Models and its Applications’. Available at: http://arxiv.org/abs/2101.06006 (Accessed: 22 June 2021).

Wang, B. and Ponce, Carlos R. (2022a) ‘Factorized convolution models for interpreting neuron-guided images synthesis’, in *Conference on Cognitive Computational Neuroscience*.

Wang, B. and Ponce, Carlos R (2022) ‘Factorized convolution models for interpreting neuron-guided images synthesis’, *2022 Conference on Cognitive Computational Neuroscience* [Preprint]. Available at: https://doi.org/10.32470/CCN.2022.1034-0.

Wang, B. and Ponce, Carlos R. (2022b) ‘High-performance Evolutionary Algorithms for Online Neuron Control’. Available at: https://doi.org/10.1145/3512290.3528725.

Wang, B. and Ponce, Carlos R. (2022c) ‘High-Performance Evolutionary Algorithms for Online Neuron Control’, in *Proceedings of the Genetic and Evolutionary Computation Conference*. New York, NY, USA: Association for Computing Machinery (GECCO ’22), pp. 1308–1316. Available at: https://doi.org/10.1145/3512290.3528725.

Wang, B. and Ponce, Carlos R. (2022d) ‘Tuning landscapes of the ventral stream’, *Cell Reports*, 41(6), p. 111595. Available at: https://doi.org/10.1016/J.CELREP.2022.111595.

Wang, B. and Vastola, J.J. (2023) ‘Diffusion Models Generate Images Like Painters: an Analytical Theory of Outline First, Details Later’. arXiv. Available at: https://doi.org/10.48550/arXiv.2303.02490.

Wang, C.-Y., Bochkovskiy, A. and Liao, H.-Y.M. (2023) ‘YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors’, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7464–7475.

White, T. (2016) ‘Sampling generative networks’, *arXiv preprint arXiv:1609.04468* [Preprint].

Willeke, K.F. *et al.* (2023) ‘Deep learning-driven characterization of single cell tuning in primate visual area V4 unveils topological organization’. bioRxiv, p. 2023.05.12.540591. Available at: https://doi.org/10.1101/2023.05.12.540591.

Xiao, W. and Kreiman, G. (2020) ‘XDream: Finding preferred stimuli for visual neurons using generative networks and gradient-free optimization’, *PLOS Computational Biology*, 16(6), p. e1007973. Available at: https://doi.org/10.1371/JOURNAL.PCBI.1007973.

Yamane, Y. *et al.* (2008a) ‘A neural code for three-dimensional object shape in macaque inferotemporal cortex’, *Nature Neuroscience*, 11(11), pp. 1352–1360. Available at: https://doi.org/10.1038/nn.2202.

Yamane, Y. *et al.* (2008b) ‘A neural code for three-dimensional object shape in macaque inferotemporal cortex’, *Nature Neuroscience*, 11(11), pp. 1352–1360. Available at: https://doi.org/10.1038/nn.2202.

Yamins, D.L.K. *et al.* (2014) ‘Performance-optimized hierarchical models predict neural responses in higher visual cortex’, *Proceedings of the National Academy of Sciences of the United States of America*, 111(23), pp. 8619–8624. Available at: https://doi.org/10.1073/pnas.1403112111.

Yau, J.M. *et al.* (2012) ‘Curvature processing dynamics in macaque area V4’, *Cerebral Cortex*, 23(1), pp. 198–209.

Yildirim, I. *et al.* (2020) ‘Efficient inverse graphics in biological face processing’, *Science Advances*, 6(10), p. eaax5979. Available at: https://doi.org/10.1126/sciadv.aax5979.

Yu, F. *et al.* (2015) ‘Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop’, *arXiv preprint arXiv:1506.03365* [Preprint].

Yuan, X. *et al.* (2020) ‘Averaging symmetric positive-definite matrices’, in *Handbook of Variational Methods for Nonlinear Geometric Data*. Springer, pp. 555–575.

Zavatone-Veth, J.A. *et al.* (2023) ‘Neural networks learn to magnify areas near decision boundaries’. arXiv. Available at: https://doi.org/10.48550/arXiv.2301.11375.

Zetzsche, C., Krieger, G. and Wegmann, B. (1999) ‘The atoms of vision: Cartesian or polar?’, *JOSA A*, 16(7), pp. 1554–1565.

Zhang, R. *et al.* (2018) ‘The Unreasonable Effectiveness of Deep Features as a Perceptual Metric’, in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 586–595. Available at: https://doi.org/10.1109/CVPR.2018.00068.

Zhu, J.-Y. *et al.* (2016) ‘Generative visual manipulation on the natural image manifold’, in *European conference on computer vision*. Springer, pp. 597–613.

Zhu, S.C., Wu, Y. and Mumford, D. (1998) ‘Filters, Random Fields and Maximum Entropy (FRAME): Towards a Unified Theory for Texture Modeling’, *International Journal of Computer Vision*, 27(2), pp. 107–126. Available at: https://doi.org/10.1023/A:1007925832420.

Ziemba, C.M. *et al.* (2016) ‘Selectivity and tolerance for visual texture in macaque V2’, *Proceedings of the National Academy of Sciences of the United States of America*, 113(22). Available at: https://doi.org/10.1073/pnas.1510847113.

Ziemba, C.M. *et al.* (2019) ‘Laminar Differences in Responses to Naturalistic Texture in Macaque V1 and V2’, *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*, 39(49), pp. 9748–9756. Available at: https://doi.org/10.1523/JNEUROSCI.1743-19.2019.

Zoccolan, D. *et al.* (2007) ‘Trade-Off between Object Selectivity and Tolerance in Monkey Inferotemporal Cortex’, *Journal of Neuroscience*, 27(45), pp. 12292–12307. Available at: https://doi.org/10.1523/JNEUROSCI.1897-07.2007.

Zoran, D. and Weiss, Y. (2012) ‘Natural Images, Gaussian Mixtures and Dead Leaves’, in *Advances in Neural Information Processing Systems*. Curran Associates, Inc. Available at: https://papers.nips.cc/paper_files/paper/2012/hash/e97ee2054defb209c35fe4dc94599061-Abstract.html (Accessed: 7 August 2023).