Spring 2021

The Manifold of Neural Responses Informs Physiological Circuits in the Visual System

Luciano Dyballa
Yale University Graduate School of Arts and Sciences, dyballa@gmail.com

Abstract

The Manifold of Neural Responses Informs Physiological Circuits in the Visual System

Luciano Dyballa
2021

The rapid development of multi-electrode and imaging techniques is leading to a data explosion in neuroscience, opening the possibility of truly understanding the organization and functionality of our visual systems. Furthermore, the need for more natural visual stimuli greatly increases the complexity of the data. Together, these create a challenge for machine learning. Our goal in this thesis is to develop one such technique. The central pillar of our contribution is designing a manifold of neurons, and providing an algorithmic approach to inferring it. This manifold is functional, in the sense that nearby neurons on the manifold respond similarly (in time) to similar aspects of the stimulus ensemble. By organizing the neurons, our manifold differs from other, standard manifolds as they are used in visual neuroscience which instead organize the stimuli.

Our contributions to the machine learning component of the thesis are twofold. First, we develop a tensor representation of the data, adopting a multilinear view of potential circuitry. Tensor factorization then provides an intermediate representation between the neural data and the manifold. We found that the rank of the neural factor matrix can be used to select an appropriate number of tensor factors. Second, to apply manifold learning techniques, a similarity kernel on the data must be defined. Like many others, we employ a Gaussian kernel, but refine it based on a proposed graph sparsification technique—this makes the resulting manifolds less sensitive to the choice of bandwidth parameter.

We apply this method to neuroscience data recorded from retina and primary visual cortex in the mouse. For the algorithm to work, however, the underlying circuitry must be exercised to as full an extent as possible. To this end, we develop an ensemble of flow stimuli, which simulate what the mouse would 'see' running through a field. Applying the algorithm to the retina reveals that neurons form clusters corresponding to known retinal ganglion cell types. In the cortex, a continuous manifold is found, indicating that, from a functional circuit point of view, there may be a continuum of cortical function types. Interestingly, both manifolds share similar global coordinates, which hint at what the key ingredients to vision might be.

Lastly, we turn to perhaps the most widely used model for the cortex: deep convolutional networks. Their feedforward architecture leads to manifolds that are even more clustered than the retina, and not at all like that of the cortex. This suggests, perhaps, that they may not suffice as general models for Artificial Intelligence.

The Manifold of Neural Responses Informs Physiological Circuits in the Visual System

A Dissertation
Presented to the Faculty of the Graduate School
of
Yale University
in Candidacy for the Degree of
Doctor of Philosophy

by
Luciano Dyballa

Dissertation Director: Steven W. Zucker

June 2021

Acknowledgments

I would like to express my deepest gratitude to my advisor, Professor Steven Zucker. Throughout my six years at Yale he has provided me with all the knowledge, inspiration, and motivation I could hope for. I truly appreciate his patience, frankness, and friendship.

I am very grateful to Professors Michael Stryker of UCSF and Greg Field of Duke University for the all the insight, support, and encouragement. It has been an honor and a pleasure to participate in our weekly calls for all these years now.

I would like to thank Professors Ronald Coifman and Julie Dorsey for accepting to serve as readers for this thesis.

I wish to thank Mahmood Hoseini and Marija Rudzite for the invaluable partnership in our fruitful collaboration.

I would like to express my appreciation to Susan Hurlburt for the constant warmth and kindness.

On the personal side, I want to thank Bill, Chris, Jake, Lucy, Lydia, and Marianne for having made my time at Yale so enjoyable and fun.

To Luiza, Felix & Cecilia.

Contents

List of Figures

Chapter 1

Introduction

1.1 Black box experiments

The Problem of the Black Box is well-known and has its origins in electrical engineering[8]. One is given a sealed box with terminals for input and output, to which one can feed any desired input and record their corresponding outputs. The challenge is then make deductions about what is in the device, i.e., its inner workings. Of course, the problem is not restricted to literal machines; it can be applied to any *system* that allows for experimentation but is not fully observable, of which examples can be found everywhere (to a child, almost anything is a black box!). Aside from the ultimate goal of understanding their mechanisms and functioning principles, such systems motivate additional questions, in particular:

- Which stimuli are most appropriate for obtaining an informative output?
- Which methods are most useful or efficient for analyzing it?
- What properties of the box are feasibly discoverable and which are not?

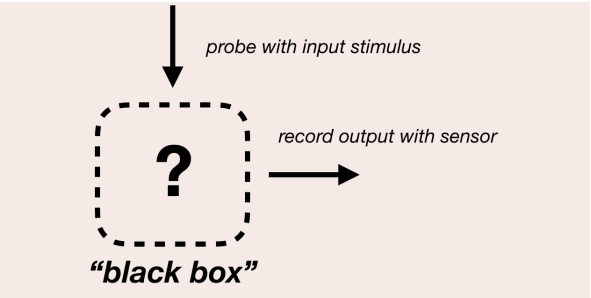

Figure 1.1: Schematic of the black box problem. Which input stimuli are relevant? How to understand its inner workings from its outputs?

The answers to these questions will depend on the nature of the black box, i.e. what its specific inputs and outputs are, as well as on whether prior knowledge is available regarding what is inside the box. In what follows, we will apply our black box analogy to the specific problem of inferring neural circuitry and propose answers the questions above.

1.1.1 The brain as a black box

The brain is a classical example of a black box, with the use of such metaphor for it referring back to the early developments in cybernetics and behaviorism. Given its mesmerizing complexity, it is not surprising that this problem has not been solved for even the simplest of organisms. How can we begin to make sense of it? What does that even mean?

Brains can be studied in different ways. An animal might respond to a stimulus and have its actions annotated by an experimenter. Or, the output of neurons might be recorded directly by the use of electrodes, in which case there might be multiple outputs to record from, simultaneously. The latter is the type we shall focus on. It treats the black box as being the circuits that govern how different neurons interact with each other when responding to a given stimulus (Fig. 1.2.

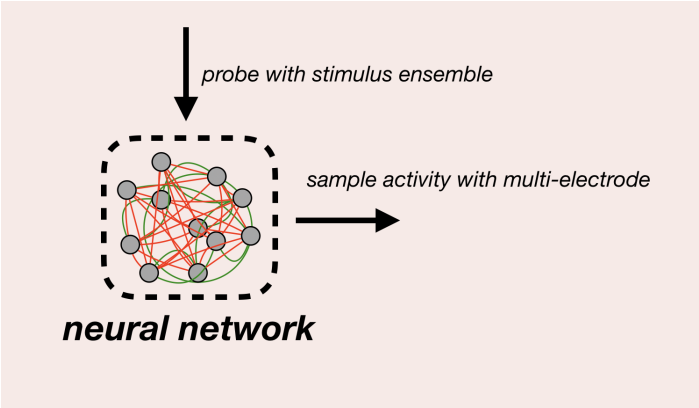

Figure 1.2: The black box problem as applied to networks of neurons. Which stimuli are most relevant? How to infer circuit properties based on the output of a sampling of neurons?

A considerable effort is currently dedicated to the mapping of the anatomic connections in brains, i.e. their microscale connectome [151]. This has been accomplished for a simple organism like the nematode [140, 34]. Although data like these can provide insight into, e.g., wiring principles [33] and hierarchical organization [34], knowledge of anatomical connectivity alone is incomplete without information about the neurons' functional roles. This motivates an accompanying effort to map functional connectivity at the cellular level (e.g., [123]). In particular, one would like to know which neurons interact with each other in response to different sensory stimuli; this means understanding how circuits are

organized, and how they encode sensory information and convert this into actions. Only then can we a chance of "opening" the "black box".

Therefore, given the increasing size of data sets being generated, the development of algorithms suitable to process this type of information becomes crucial. Far from being a solved problem in machine learning, providing an intelligible mapping of how the multiple parts in a general system interact with each other poses a significant challenge, given that it is inherently unsupervised and requires that many sources of information (e.g. multiple neuronal recordings, multiple stimuli) be combined into a coherent high-dimensional data structure.

The extent to which the network can be recovered will of course depend on the complexity of its components and connections. The problem is greatly simplified when these are restricted to linear operations (possibly followed by static nonlinearities). A classic result shows that this type of system, known as the Wiener cascade or linear-nonlinear (LN) model can be identified by its response to white Gaussian noise [144, 24]. This has recently been shown for deep ReLU networks as well [115]. The problem has not been solved for general nonlinear networks.

In this thesis, we present a combination of machine learning techniques designed for revealing organization principles in a general nonlinear neural network. Our specific goal is to infer circuit properties in the mouse visual system by using real-world neuronal spike recordings (with no information about anatomical connectivity) from primary visual cortex (V1) and the retina. The currently available methods and motivation for our own approach are discussed next.

1.2 Organizing neurons in terms of their responses

Inferring circuit structure and dynamics at the cellular level in the visual system is confronted with two major challenges. The first one is the diversity of stimuli needed to exercise the circuits at a natural activity level (i.e. appropriate inputs to the black box). The use of white noise and drifting gratings is widespread but is too impoverished a stimulus set compared to the diversity of sensory inputs experienced by the animal. Many studies demonstrate that conventional artificial stimuli fail to engage mechanisms that profoundly affect responses of neurons throughout the visual system [72, 92, 60]. At the other extreme, unconstrained natural images defy analysis.

Which stimuli are most appropriate for obtaining an informative output? We propose the use of *flows* as a complementary stimulus class—they mimic movement through natural environments and allow for richer geometry, motion, and contrast variations while preserving traditional parameters such as spatial and temporal frequencies, orientation, and directionality. Flows are significant to perceptual organization [15] and have implications to connectivity patterns in visual cortex [14]. Although far from the level of complexity of

natural scenes, we show that flow stimuli are able engage strong responses for a significant fraction of cells in the visual cortex, in particular at high spatial frequencies that would not be predicted based on the use of gratings and linear models.

Once a richer stimulus ensemble is proposed, a more complex set of responses is recorded. Data dimensionality increases fast, and the challenge becomes finding an adequate analysis that takes advantage of the detailed information collected.

Which methods are most useful or efficient for analyzing it? A common approach used to analyze responses of a population of neurons to a stimulus ensemble is to embed stimulus trials in neural coordinates [146, 36]. This is typically done by having each data point be a vector containing the firing rates of all neurons during an individual trial (see example in Fig. 1.3). This embedding is well suited for reading out which stimulus was used on a given trial [39]; that will depend on whether the neural population (as a whole) can tell the stimuli apart (case in which the trials from different stimuli form clusters).

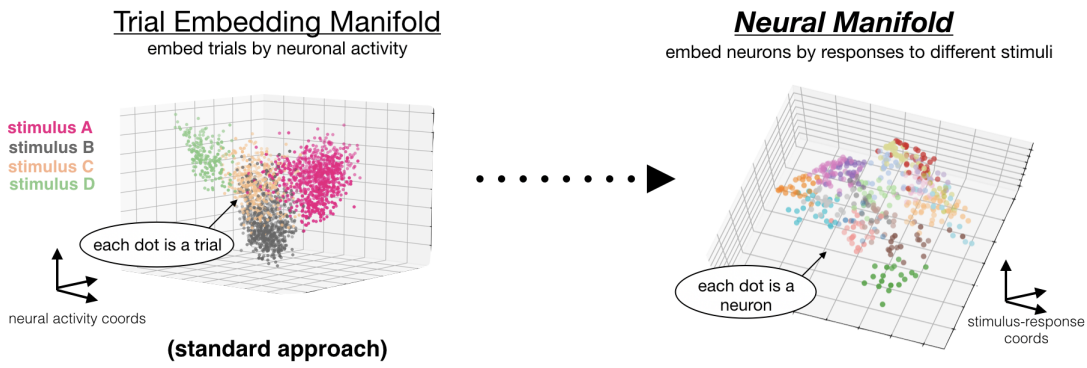

Figure 1.3: Comparison between standard approach in the neuroscience literature *vs.* ours. **Left**: the standard approach of dimensionality reduction, in which individual trials are plotted in 'neural' coordinates. For cortical data, this reveals clouds of trials, with each (differently colored) cluster corresponding to drifting gratings (low vs. high spatial frequency) or flows (positive or negative contrast). **Right**: our proposed approach, in which neurons are plotted in 'stimulus-response' coordinates. Here, different colors represent different neighborhoods sharing similar stimulus preference and response patterns, from which can be inferred that they participate in similar biological circuits.

This approach is also particularly useful for identifying transitory brain 'states' (for example, if in some trials the population activity is particularly depressed, for example, as compared to other trials). However, there are two main drawbacks: first, information about the temporal dynamics of the neurons' responses might be lost due to the use of trial-averaged firing rates; secondly; second, one cannot immediately gain insight on how the different neurons relate to one another, i.e., the axes on this plot are difficult to interpret since they denote correlated combinations of neurons (or factors such as principal components) [104, 47]. Sometimes the interpretation is helped by an associated behavioral task to align the trial data, as with repetitive reaching movements in the motor system

[29, 117], maze position for the hippocampus [59], or attention [126]. However these behavioral (task-dependent) signals are generally lacking for early visual processing [35], so we develop a different approach.

In order to understand what a circuit does, one must first identify its parts. We would like to exploit the fact that sufficiently large, complex systems typically exhibit *localized properties* [8], by utilizing a manifold embedding algorithm that aims to infer global coordinates from information regarding local interactions between data points. Our approach consists in building an embedding of neurons, i.e. the *neural manifold*, in terms of their responses to the stimulus ensemble. Instead of representing trials (and their dynamics) in neural coordinates, we turn the relationship around and seek to represent neurons in stimulus-response coordinates. (Note this is in contrast with 'decoding manifolds' that have been previously proposed [28].) Our goal is to define a manifold of neurons organized in terms of their response properties—its geometry should reflect similarities in how each neuron responds to the input stimuli in comparison to the others. Put differently, nearby neurons on the manifold should respond to similar properties of the stimulus ensemble and with similar response dynamics. We call this abstract notion of network a *physiological network* since it combines both functional and physiological properties (Fig. 1.4). How exactly such inference might be done will be discussed next.

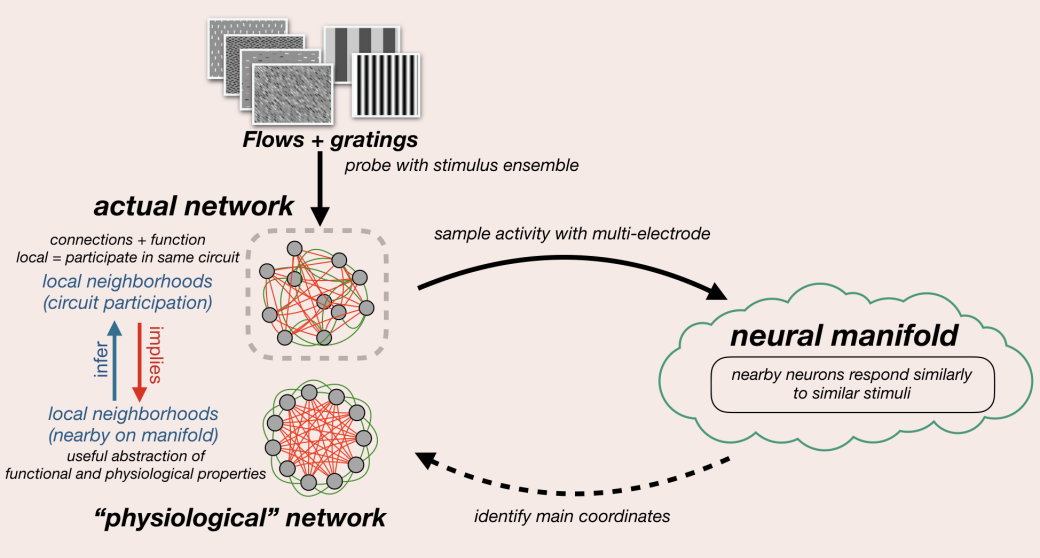

Figure 1.4: Our approach to the black box problem applied to neural circuits. Spike recordings (output) are analyzed and used to inform a manifold learning algorithm, which embeds neurons in stimulus-response coordinates. Identifying the main coordinates allows one to infer properties of an abstract version of the original network that combines functional and physiological properties. If cells are well-organized, then local neighborhoods allow us to infer 'functional' circuit properties, or, in other words, we begin to "open the black box".

1.2.1 Relating manifolds to neural circuits

How can manifolds reveal circuit properties without being given explicit connectivity information? Our method is based on the expectation that the duality that exists between networks and manifolds can allow us to infer useful information about the original network; if the underlying neural circuits were known, the neural manifold could be calculated exactly. Fig. 1.5 builds some intuition: if neurons were sampled from a collection of isolated circuits (first row), then the functional manifold would be discontinuous (clustered); each separate circuit would, in effect, define its own manifold. This is like the 'parallel pathways' case encountered to a large extent in the retina: retinal ganglion cell (RGC) types do not communicate with one another and receive distinct sets of presynaptic bipolar and amacrine cell input: they are distinct circuits. In the case of a simple (e.g. highly symmetrical) connectivity pattern, typical of artificial networks (second row), we expect a low-dimensional manifold. A third possibility, more like visual cortex, is shown in the third row. Neurons are densely interconnected within sub-circuits (e.g., excitatory cliques [152]), which in turn are less tightly coupled with one another. Such a 'partially-decomposable' network will yield a continuous manifold; coordinates across the manifold will elucidate the functional dependencies. A final possibility is a circuit in which all neurons are completely interconnected (fourth row). Its manifold would become 'degenerate', meaning that the network would not exhibit any functional organization/selectivity across groups of neurons, since all responses depend on all neurons. Thus, neural manifolds and circuit architecture are related.

Of course, we do not know the circuits *a priori*—circuits are what we aim eventually to understand. Instead, our approach is to construct the manifolds from neural recordings as an intermediate step toward inferring circuit structure. This is already useful at the most basic, topological level. Indeed, our results for the retina suggest a discontinuous encoding manifold while those for the cortex suggest a continuous manifold (Chapter 6). Our approach is summarized in Fig. 1.6.

It is worth noting that this is in contrast with recent efforts to understand visual processing using deep networks (e.g., [79, 97, 11, 149]). Our approach is data-driven (i.e., unsupervised), and benefits from the fact that our stimuli have configurable parametric/geometric properties, which can be smoothly varied by these parameters (unlike arbitrary natural images). Our analysis approach provides a description of units (neurons) directly in terms of functional properties and their relationships. This complements the type of analysis based on deep networks.

Perhaps the closest two attempts to analyze neuroscience data are as follows. The authors in [99] study data from imaging, rather than electrodes, but also use tensor factorization and diffusion maps. Their goal is a hierarchical representation of the biological data. Second, authors in [91] use manifold techniques to study hippocampal neurons. Most approaches use the standard manifold in neural coordinates (e.g., [23, 49] and references

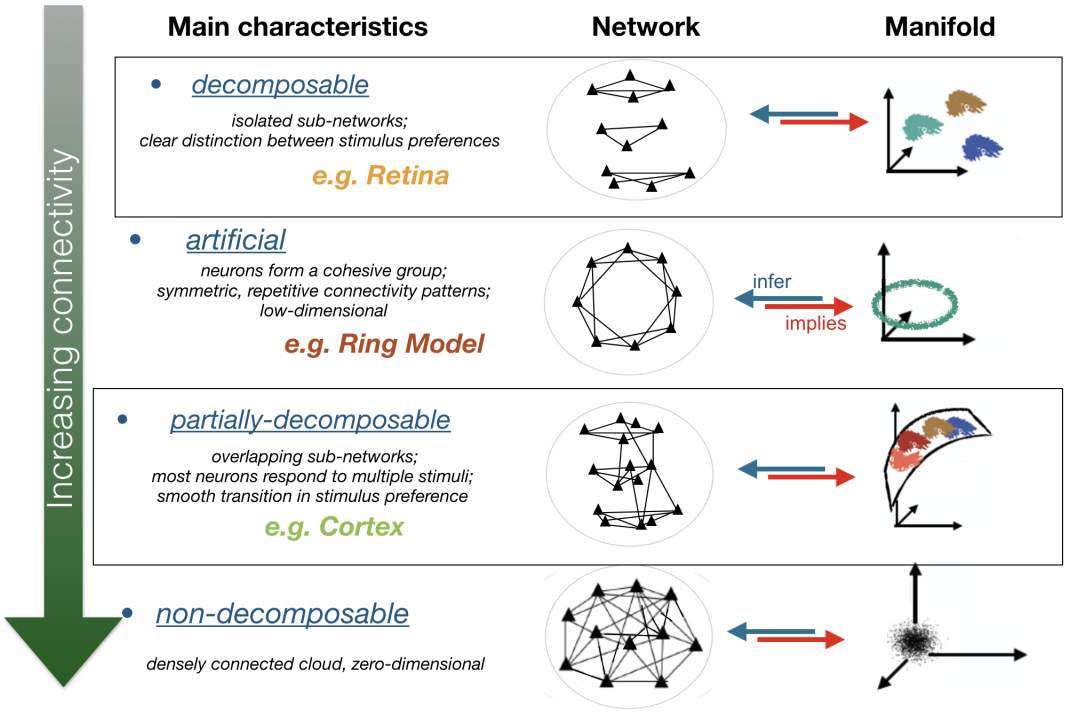

Figure 1.5: Generic network architectures with increasing level of connectivity and their corresponding manifolds. (See text for details.)

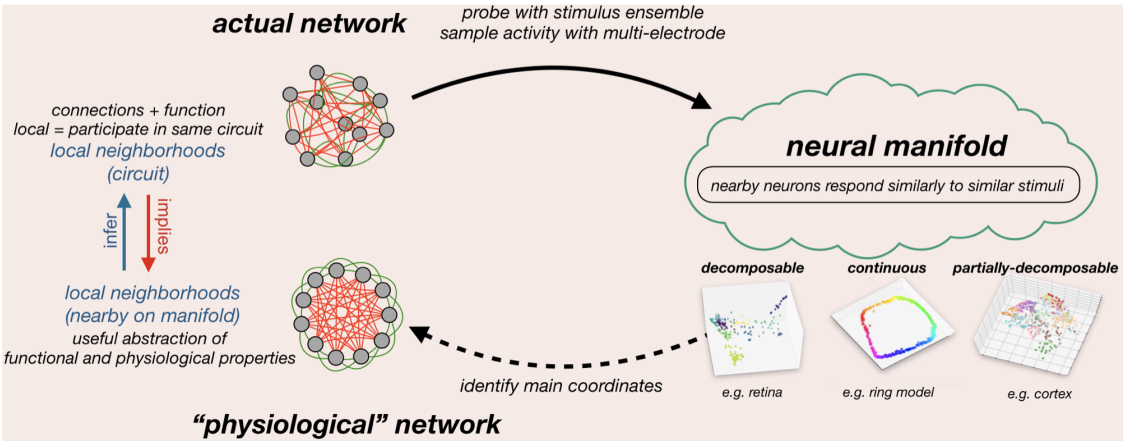

Figure 1.6: The neural manifold approach allows one to infer the general network architecture of the circuits being probed in terms of their responses to the stimulus ensemble.

therein).

1.2.2 Dimensionality reduction steps

Our approach to generate response manifolds is carried out in two main stages. First, we identify groups of neurons that respond similarly over time to different aspects of the

stimulus ensemble. Since these neural groupings still contain a considerable degree of complexity, we reduce their dimensionality even further, obtaining novel coordinates for embeddings that can be interpreted in either neural or stimulus terms. We now describe these stages in more detail (they are schematized in Fig. 1.9).

Step 1: Tensor factorization

Each neuron's average response to each stimuli over time is represented by an array of spiking activity over time (a peristimulus time histogram, or PSTH). Since there is one PSTH for each member of our stimulus ensemble for each neuron, this results in a data tensor (3D array) that has axes (i.e. modes) of neuron $\times$ stimulus $\times$ PSTH (Fig. 1.8). Such tensors are a generalization of data matrices and, analogously to how the latter can be decomposed into their (principal) components, the former can be expanded as a sum of rank-1 components (outer product of *factors*, one for each mode) [77].

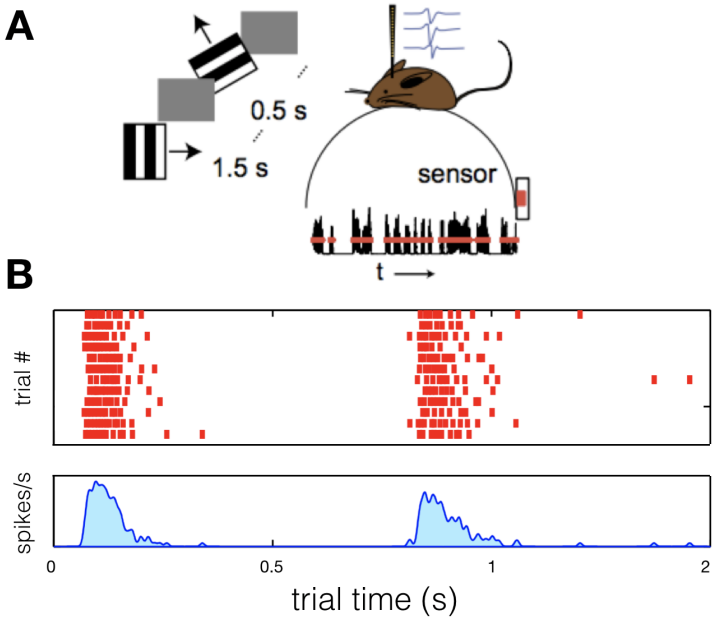

Figure 1.7: Inputs and outputs in our "black box" experiment. **A**: A multi-electrode array records neuronal activity from mouse visual cortex as visual stimuli are presented on a screen. **B**: Top: spikes (in red) are averaged across multiple trials of the same stimulus to produce an average firing rate over the time, or PSTH (in blue).

Each tensor component is an association of factors encoding different aspects of the data. They show which neurons, in proportion, respond to which stimuli (height of bars) and with which PSTH patterns. Different components show different combinations of parts; for example, some could involve one group of neurons and be heavily responsive to low frequency gratings; others could show different groups of neurons that respond to

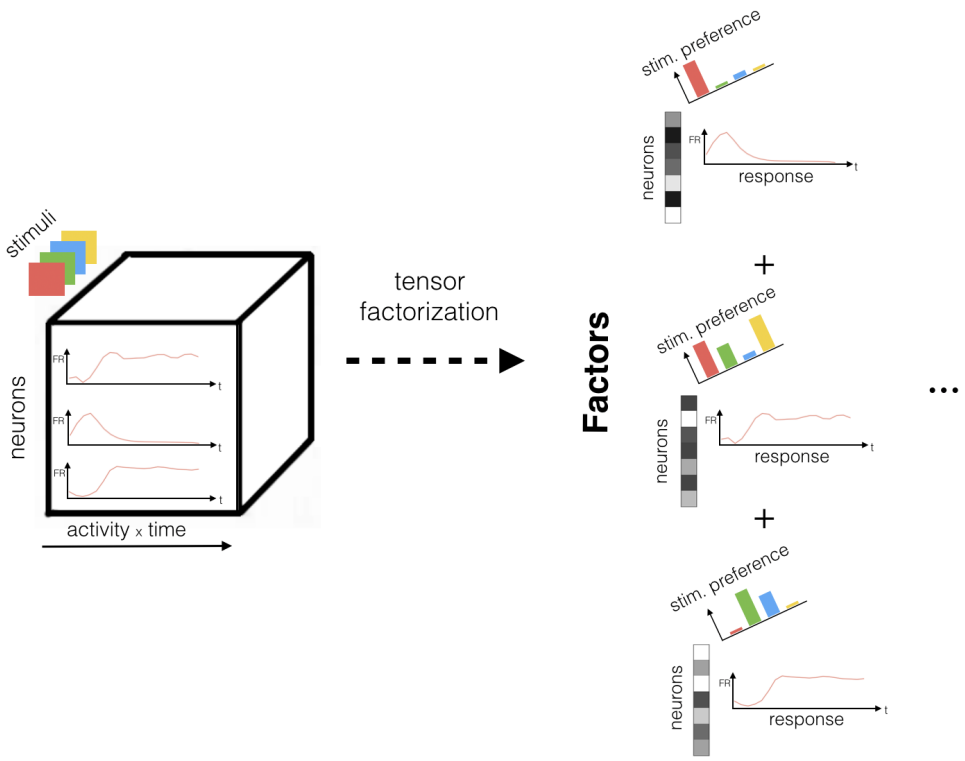

Figure 1.8: Summary of tensor CP decomposition and resulting components as applied to our experiments. Using PSTHs containing spiking activity over the full time course of a trial (as opposed to the average firing rate) naturally suggests the use of a tensor to represent data for multiple stimuli. Each factor is comprised of a neural, a stimulus, and a PSTH component.

flows or, perhaps, a mixture of flows and gratings.

Tensor CP decomposition, or factorization, has gained considerable popularity in the past couple of decades with applications to high-dimensional data ranging from analysis of EEG data [100] to classification of hazardous gases [141], and have been used to analyze neuronal responses in terms of PSTHs of individual trials [145]. The tensor factors obtained seem to provide useful summaries of the multi-linear relationships present in the data.

Naturally, if the system were linear, tensor factorization alone could identify it. Even though neural processing is known to be highly nonlinear [78], we still believe these factors can be of great utility as a preliminary dimensionality reduction step prior to the (nonlinear) embedding algorithm. Because each neural factor is associated to both a stimulus and a PSTH factor in the same component, their neural coefficients essentially tell us how well each cell's response to a certain stimulus can be represented (i.e., reconstructed) by its corresponding PSTH component.

Step 2: Creating the neural manifold

The resulting factors from our real-world experiments (retina and cortex) are neither sparse nor simple. As one might expect from circuit interactions, it is difficult to associate individual neurons with individual stimuli, which leads to our second dimensionality reduction stage: creating the encoding manifold. Although the factors are different, neurons have a distributed role: they can participate in many factors. Stated geometrically, each neural factor can be viewed as a coordinate dimension, and each neuron can be plotted as a 'point' in this neural-factor space. Nearby neurons in this space will have similar loadings (coordinates) in the different dimensions.

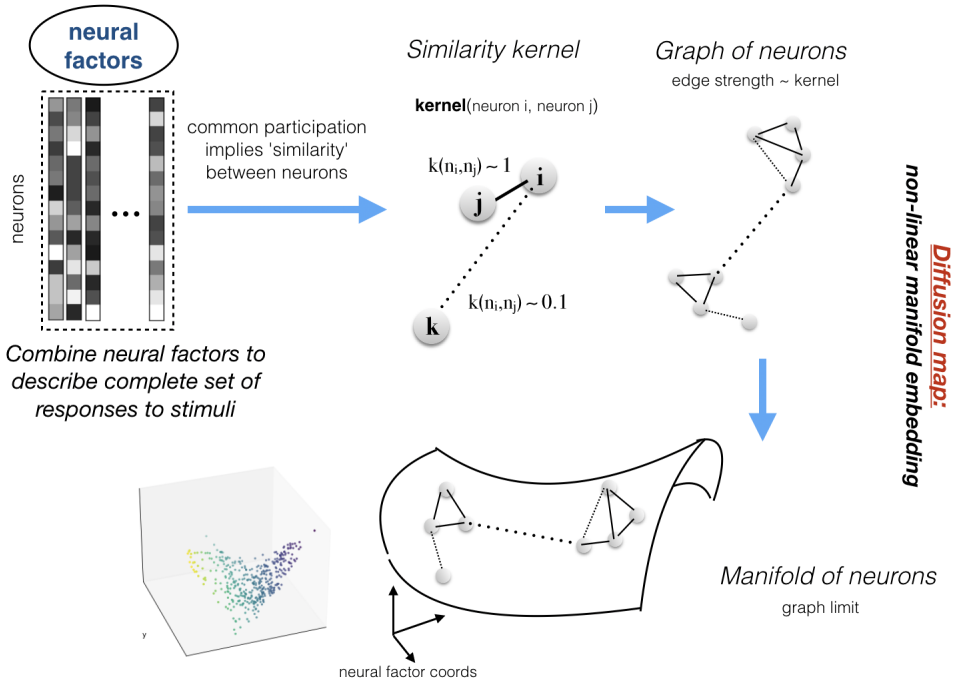

Figure 1.9: Using neural factors as input to a manifold learning algorithm (diffusion maps).

Diffusion maps [31] are used to infer manifolds with network implications. While PCA seeks a linear manifold based on Euclidean distances, the diffusion map algorithm uses a similarity kernel to create a nonlinear manifold organized based on a graph/network where each data point is a node, and local similarity provides the connectivity (Fig. 1.9). Points in the manifold are then organized by *diffusion distance*, which measures how much diffusion can occur from one node to another over the graph: effect, the more paths connecting two nodes, the closer they are; bottlenecks push nodes apart. Interpreting this for a collection of neurons, the more paths between them, the closer they will be on the manifold. The self-excitatory clique motif emerging in connectomics is a perfect example of this [152], and an artificial example where the circuit topology is perfectly recovered is given below (ring model).

1.3 Simulation examples

Now we present two simulation experiments that will be used throughout Part I as examples of application of our algorithm. These are particularly simple models with the objective of, first, confirming that our approach produces the expected results, and, secondly, clarifying how to interpret the results which will help putting our real-world biological results from Chapter 6 into perspective.

1.3.1 Ring model

The ring model from [16] is an artificial model of an orientation hypercolumn, as observed in primary visual cortex of cats and monkeys [69]. Neurons have preferred orientation evenly distributed between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$. Each neuron forms recurrent connections to all others, with excitatory weights to nearby neighbors and inhibitory weights to the remaining ones (see connection weight matrix in Fig. 1.10. Despite its apparent simplicity, this model is sufficient to explain several orientation tuning phenomena observed in primary visual cortex.

After receiving an oriented input (the *stimulus*), each artificial cell responds proportionally to the similarity between the orientation of the stimulus and its own preference, and its ultimate change in output (treated as spikes/s) is obtained after integrating the inputs from all its neighbors. In our simulation, we used 8 different stimuli with orientations evenly distributed between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$. Each stimulus was presented several times with Gaussian noise added to make each response virtually unique, given that the model is deterministic.

1.3.2 Random LN model

Next we simulate neurons following the linear-nonlinear (LN) model ([144, 64, 94]), a heavily used model in neuroscience due to its simplicity (Fig. 1.11). We investigate the particular case of using uniformly random receptive fields (RFs) and white Gaussian noise movies as stimuli. The goal here is to create a system in which each neuron is likely to respond differently to each stimulus, and then to learn whether our algorithm will produce results compatible with such scenario. This represent an extreme case since neurons with different RFs will be essentially "orthogonal" to each other in terms of their responses.

We used two different noise movies were used as stimuli. Spikes were generated by using the result of the dot product between each movie frame and a neuron's RF as the instantaneous firing rate to a Poisson process [93]. This converts the input into a stochastic binary output (spikes) for each neuron. The PSTHs then register the average amount of spikes for each frame in the movie.

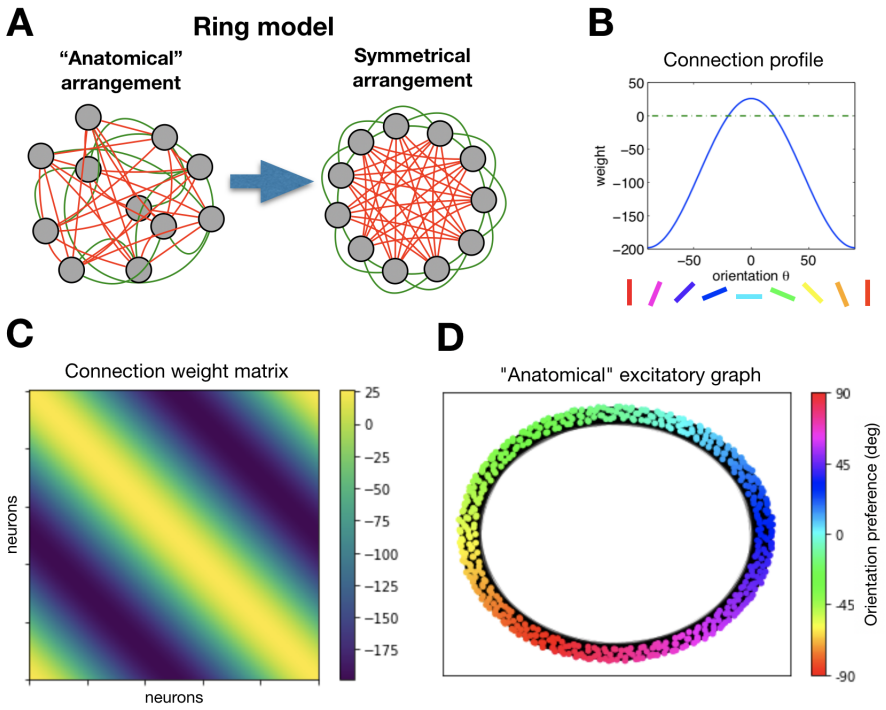

Figure 1.10: Ring model of orientation tuning. **A**: We seek to infer an organization of neurons that relates to their 'functional' role in terms of responses and stimulus selectivity. Although anatomically neurons can be arranged in many ways, intuitively for the ring model this functional relationship between the neurons implies a ring-like symmetry. **B**: Connection profile providing the synaptic weights in the model: neurons with similar orientation tuning excite each other; otherwise inhibition occurs. **C**: Matrix of synaptic connection weights between all pairs of neurons. Positive (negative) values represent excitatory (inhibitory) connections between neurons with similar (dissimilar) stimulus preference. **D**: The matrix in **C** can be used as a graph adjacency matrix by considering its positive entries alone as edge weights and the neurons as vertices. An embedding of such graph can be created using a physical model that positions the vertices by considering each edge as a spring. Stimulus preference varies smoothly around the ring and edges (in black) connect nearby neurons.

1.4 Motivating Conjectures

One of the foundations of visual neuroscience is the notion of the receptive field: the pattern of light in the visual array that stimulates a neuron. Such receptive fields were first identified in the retina and later characterized in the visual cortex (see Fig. 1.12). Thinking of these receptive fields as filters, and placing a nonlinearity in the neural component, provides the foundation and the origin for modern deep convolutional networks [85]. These are now used widely in applications of computer vision, and also as models of visual cortex [148, 113, 74, 80, 89] plus references therein. The key idea is that these receptive

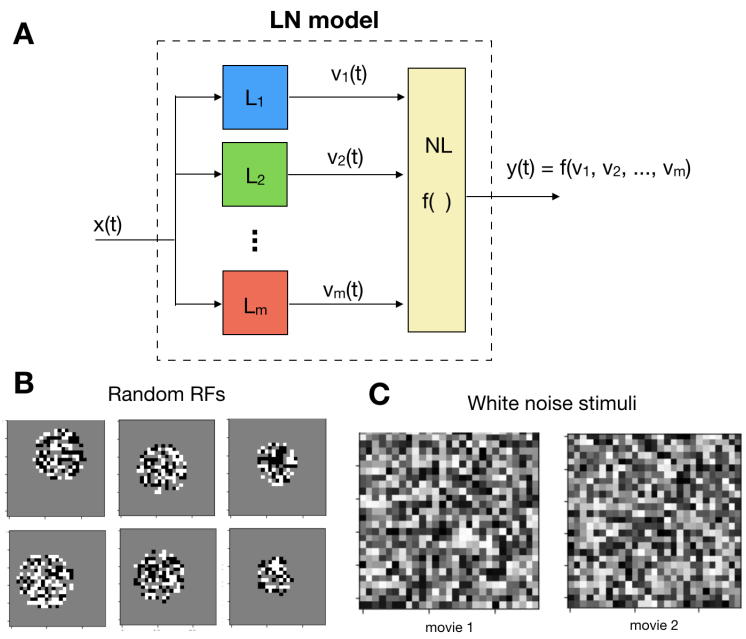

Figure 1.11: Our LN model simulation with random stimuli and receptive fields (RFs). **A**: The classic LN model of computation (adapted from [94]). The input $x(t)$ passes through one or several linear operators L_i and their outputs pass through a common static nonlinearity, being combined into a single output $y(t)$. **B**: Two examples of localized random RFs. Neurons with the same RF belong to the same 'type'. **C**: One example of frame from each of the two white noise movies used as stimuli.

field 'filters' model the functional capabilities of the neuron exhibiting them, and that the networks are built by composing these filters with nonlinearities [133].

Many different questions are raised by these statements. Based on receptive field characterizations, we first ask whether the simple, complex, and hypercomplex (endstopped) cells described classically by Hubel and Wiesel [67] yield a sufficent characterization of their function. Are these three groups of cells distinct and complete; i.e., do they suffice to characterize visual cortex? In the retina, it is now known that there are about 40 different classes of retinal ganglion cells (RGCs) [118, 9]. Given the difference in complexity between the retina and the cortex, one would expect many more different functional types of cortical cells. Does this classical categorization suffice, or should the types of cortical cells be organized differently?

Second is the circuit organization. The Hubel-Wiesel model is feedforward, and deep convolutional networks are feedforward; is there more to the functional interconnection structure of the retina and the cortex? Anatomy would suggest so. While the retina has many feedforward characteristics, the horizontal and amacrine cell types provide for lateral interactions [38]. Do these lateral interactions dominate for any of the RGC types?

And third, are these feedforward circuits in fact a sufficient model for visual cortex?

Furthermore, since deep convolutional networks (DCNs) are now widely used as models of visual cortex, do they suffice? And finally, although such networks are widely used in computer vision, what lessons should applied computer vision take from them [82]?

Such questions are multi-faceted, and we shall provide answers to them in the context of this thesis. First, without a rich ensemble of stimuli, it is impossible to know whether the networks of neurons are exercised to the fullest of their capabilities. It shall turn out that, in fact, using classical laboratory stimuli (e.g. drifting gratings) is insufficient. Second, anatomy clearly indicates that cortical networks are not simply feedforward; our computational techniques will provide richer approximations to them. Finally, we will compare cortical responses to those of DCNs, and will show that they are lacking.

We summarize our results in two conjectures:

CONJECTURE 1 Although the retina can be viewed as an "outcropping" of the brain, the (relatively) distinct retinal ganglion cell types differ fundamentally from the functional cell types in cortex, which are distributed more continuously.

CONJECTURE 2 Although deep convolutional networks have been used as models of cortex, and as models for cognitive vision, they are closer to big retinas than to little brains.

1.5 Overview of this thesis and main contributions

Building manifolds of neurons helps alleviate the sampling issue mentioned above, since an adequate kernel choice can, to some extent, correct for local changes in density. Furthermore, by embedding neurons directly as points we naturally allow for the combination of multiple recordings into a single manifold. However, given the multiple-mode nature of the data (several response arrays for several stimuli), we found that the quality of the diffusion map embeddings can be improved if one preorganizes the neurons' responses in terms of neural factors, a product of non-negative tensor factorization (NTF) (Chapter 2).

Basically, NTF provides a multi-clustering approach to specify which groups of neurons respond similarly (in time) to which components of the stimulus ensemble. As will be shown, when the neural factors are independent, this can suffice for determining which neurons respond to which stimuli.

A substantial difficulty arises, however, when the different neural factors are intertwined with one another, meaning multiple neurons respond to multiple stimuli. This can be interpreted as neurons being coupled in circuits. Clearly, for the visual system, this is the case when any reasonably dense sampling of neurons is obtained. We study this situation using a new construct: the *neural matrix* (Chapter 3). The neural matrix is one of the main contributions of this thesis for machine learning, since it provides an interface between multi-linear methods and nonlinear manifold learning. Its rank, in particular, has important implications for the neuroscience data analysis problem, and, we predict, for many other problems.

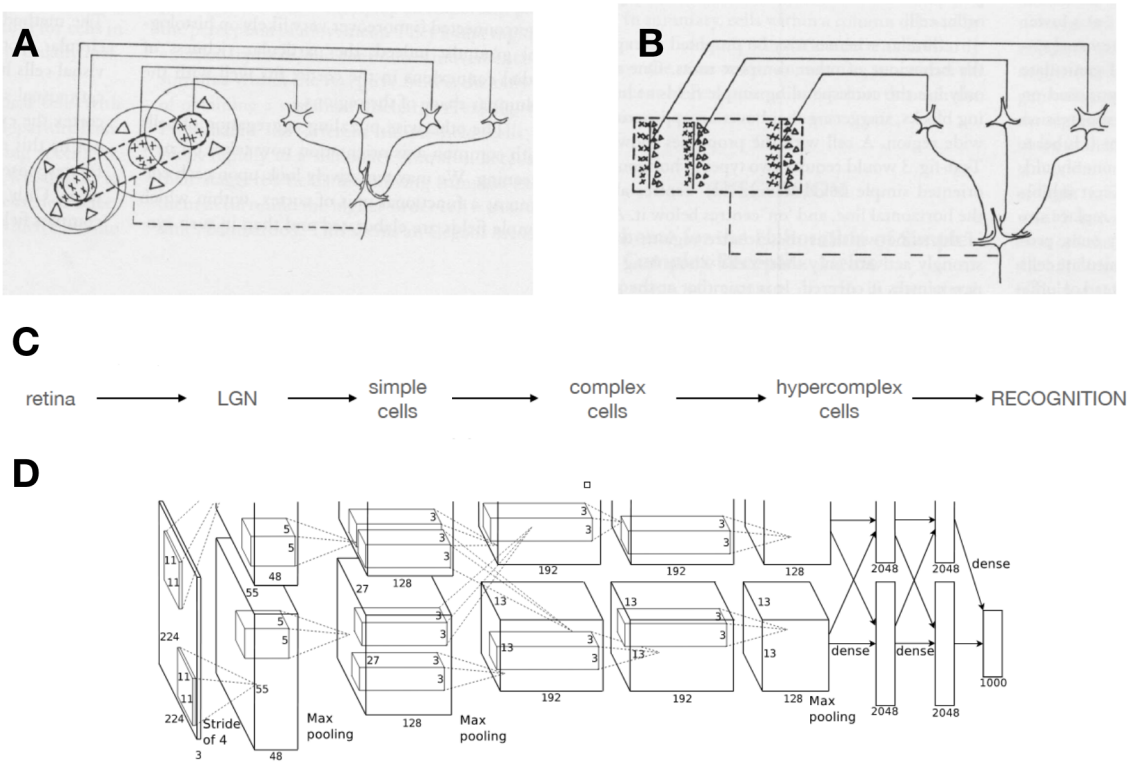

Figure 1.12: From receptive fields to deep neural networks. (a) Visual receptive fields in the lateral geniculate are circular-surround. Collections of aligned cells project to visual cortex, forming a simple cell. Orientation selectivity arises from the alignment of receptive fields. (b) An arrangement of simple cells with 'edge-like' receptive fields combine nonlinearly to provide a complex cell. (c) Sketch of the hierarchy of visual function in Hubel-Wiesel terms. (d) Alexnet [81], a deep network that elaborates this composition of convolutions with nonlinearities, inspired the deep learning period in artificial intelligence. Diagrams in (a) adapted from [68].

Both algorithms require the choice of critical parameters, namely the number of factors to use in NTF and the kernel scale in diffusion maps. We propose a method for determining the optimal number of factors for NTF based on the rank of the neural matrix in Chapter 3, and a more robust similarity kernel for diffusion maps based on graph sparsification in Chapter 4. Formulating and analyzing the neural matrix as an interface between tensor factorization and manifold learning provides the main technical contributions of the thesis.

In order to meet the challenge of using stimuli that adequately exercise the visual system while being amenable to analysis, we have developed and tested an ensemble of flow patterns, a class of naturalistic visual stimuli that span spatial and temporal frequencies, contrast, orientation and directionality. In Chapter 5 we explain in detail how these are generated and how neurons in mouse primary visual cortex respond to them in comparison with traditional drifting gratings.

In Chapter 6, we apply our procedure to biological neurons recorded from mouse retina and primary visual cortex (area V1), as well as to artificial ones in deep convolutional networks. Retinal ganglion cells (RGCs) produce a highly clustered embedding, where each cluster corresponds roughly to a known RGC type. On the other hand, the cortical manifold is qualitatively different, and there is a continuous change in response properties as one navigates through it. Specific regions of the manifold contain molecularly and anatomically distinct collections of neurons; the molecular and anatomical specificity was not used in the creation of the manifold, but its emergence confirms the validity off the diffusion map analysis. Furthermore, when we apply our algorithm to artificial neurons in deep convolutional nets we see highly clustered embeddings, with each cluster associated to a different feature map. Moreover, we introduce a measure of how 'clustered' (as opposed to continuous) an embedding is, which allows for a quantitative comparison between the overall architectures of the functional networks obtained for the different data sets.

We conclude with a discussion, in Chapter 7, of what can be inferred from the contrast between the different types of manifold obtained for different systems, and how can our approach be used to identify the usefulness of novel visual stimuli in biological experiments. In particular, how the rank of the neural matrix may be used to identify the redundancies present in a stimulus ensemble.

Part I

Algorithm development

Chapter 2

Data factorization

- **Goal**: Initial organization of neurons, stimuli, and responses based on a multi-linear model;
- **Method**: Non-negative tensor factorization;
- **Results**: Organization of the tensor; representation of data as PSTHs; algorithm selection; numerical results on artificial problems.

2.1 Matrix factorization *vs.* tensor factorization

The goal of matrix factorization is to factorize a matrix into a product of (usually) two matrices such that their product recovers the original one, or an approximation of it. In many cases, the resulting decomposition can reveal latent features (factors) which provide insight into the main interactions between rows and columns of the original data matrix (the interpretation will depend on whatever physical meaning the rows and columns have). In data analysis it is typically used to find a low rank structure that is sufficient to approximate the data using fewer dimensions. Underlying is the assumption that a big part of the original data matrix is either redundant or unessential for effectively organizing the data points.

One ubiquitous example of low-rank approximation of a matrix is principal component analysis (PCA), used to obtain a low-dimensional (orthogonal) representation of the data that tries to preserve most of its variance. When one is interested in both the column structure *and* the row structure of the matrix, however, then matrix factorization (MF) is generally used to capture the latent factors, and might enforce properties other than orthogonality on the resulting factors, such as sparsity or non-negativity. Non-negative matrix factorization (NMF) is a class of algorithms whose goal is to explain the data by a parts-based additive representation, where zeroes represent absence and positive numbers represent the presence of some property or component [30]. Oftentimes this provides for

more easily-interpretable results when using naturally non-negative data such as images, firing rates, probabilities, etc.

However, there are many circumstances in which data collection allows for a multi-mode analysis. This typically occurs when, in addition to the usual physical dimensions (space, time, etc.), there are multiple sensors being used, or multiple subjects performing either the same or several different tasks. In these cases, the data is more naturally represented as a higher order tensor, and analyzing it using tensor decomposition (as opposed to re-organizing the data to fit standard matrix-oriented algorithms) might help reveal additional structure by preserving all dimensions in the data and allowing for a multi-linear model to hopefully produce physically meaningful components [30, 77, 2].

Tensor factorization (TF) methods for big data analysis have only recently gained considerable popularity, but are quickly becoming a key tool for the extraction of features and multi-linear structure from high-dimensional data. Although different types of decomposition exist (see, e.g., [77] for a review) we will focus on the Canonical Polyadic / CANDECOMP / PARAFAC (CP) tensor decomposition, which was developed in the 1970s [21, 57] but first invented almost a century ago [62, 63].

2.2 CP decomposition

Recall that in matrix factorization a matrix M is approximated by a sum of R rank-one matrices:

$$M \approx AB^T = \sum_{r=1}^{R} a_r b_r^T. \tag{2.1}$$

Typically, the goal is to find a low-rank approximation of M, which means R is smaller than the true rank of M. This means each entry in M can be expressed using fewer dimensions, which creates a good "summary" of the data. This can make the data easier to embed, or to cluster, by making the distance relations between data points more 'meaningful'. Other applications include compression, denoising, matrix completion. Looking at the factors that allow for this summarization might provide helpful insight into what underlying features are most important within the data set.

Analogously,[1] we can approximate an n-way tensor $T \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_n}$ is approximated by a sum of rank-one tensors (illustrated in Fig. 2.1):

$$T \approx \tilde{T} \equiv \sum_{r=1}^{R} v_r^{(1)} \circ v_r^{(2)} \circ \ldots \circ v_r^{(n)}, \tag{2.2}$$

where R is the number of components chosen, and "$\circ$" stands for the vector outer product. We say each rank-one tensor is formed by the outer product between each factor in the

[1] See [145] for a detailed comparison between CP and PCA.

same component (we follow the convention that a 'component' refers to each set of associated factors, one from each tensor mode). This notation can be made more analogous to eq. 2.1 by explicitly denoting the factor component matrices (following [76]), as shown below for the specific case of a 3-way tensor:

$$\tilde{T} = [\![A, B, C]\!] \equiv \sum_{r=1}^{R} a_r \circ b_r \circ c_r, \tag{2.3}$$

where A is called a *factor matrix* containing the factors a_r as its columns, and similarly for B and C (in our specific examples from part 1.3 these would be the neural, stimulus, and PSTH matrices). It is also convenient to assume that all columns are normalized to unit length, with the weights absorbed into a vector $\lambda \in \mathbb{R}^R$. This can be denoted as:

$$\tilde{T} = [\![\lambda\ ;\ A, B, C]\!] \equiv \sum_{r=1}^{R} \lambda_r a_r \circ b_r \circ c_r. \tag{2.4}$$

Algorithms for CP decomposition are usually based on the alternating least squares (ALS) method ([22, 58]), and several variations exist [30]. Most of them use squared reconstruction error as objective function:

$$\min_{A,B,C} \|T - \tilde{T}\|_{\mathrm{F}}^2. \tag{2.5}$$

where $\|\cdot\|_{\mathrm{F}}$ is the Frobenius matrix-norm.

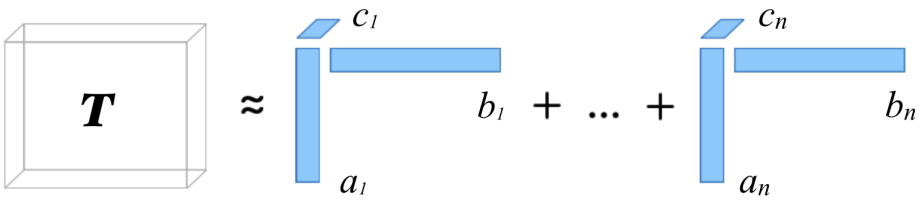

Figure 2.1: In CP decomposition, an n-way tensor T is approximated by a sum of rank-1 tensors. Each rank-1 tensor is called a 'component' formed by the outer product of n vectors called 'factors'.

Since our data comes from neuronal firing rates (averages over spike counts), we choose to use non-negative tensor decomposition (NTF), which is formulated as in eq. 2.5 with an additional non-negativity constraint. While it is true that the inclusion of this constraint may decrease some of the explained variance [30], on the other hand one gains in ease of interpretation of the components. Also, results from [145] indicate that it produces more stable factors, is unlikely to overfit, and has comparable parameter efficiency (in terms of number of learned parameters required for achieving similar error as negative

TF). We experimented with different algorithms, but the preferred one is the gradient-based direct optimization approach (OPT) from [3], which was shown to give higher accuracy if "overfactoring". This prevents degeneracy of the solution, which can be helpful when making a choice on the number of factors to select.

A few peculiarities: the factor matrices are not orthogonal and may in fact have linearly dependent columns. Importantly, the best rank-r factorization may not be part of the best rank-$(r-1)$ factorization [77]. Because NTF requires an initial guess for the factors, each run may yield different results, even when the same number of components is chosen.

2.2.1 Tensor normalization

Because the objective of the TF algorithm is to minimize reconstruction error, as defined in eq. 2.5 (although see [65] for the use of generic cost functions), PSTH patterns with longer periods of activity (or with larger area) will be more advantageous in terms of cost minimization compared to more sparse patterns. This creates a bias towards producing factors for reconstructing all of the 'sustained' response patterns first, which might well represent an entire stimulus. The solution to make all PSTHs 'equal' under the "eyes" of the TF algorithm is to prenormalize them to have unit norm.

There is one further complication, though: when each neuron responds to multiple stimuli, it is desirable to preserve the relative levels of activity between each stimulus. So we normalize each PSTH from the same neuron to have norm equal to the their average firing rate (FR) divided by the maximum FR among all stimuli (i.e., a real number between 0 and 1). In this way, the original magnitude of the FRs is discarded, but the relative FR between stimuli is preserved (something akin to the relative "importance" of each stimulus for that cell). Each neuron is thus given equal total weight.

We maintain this relative scale, however, because in our experiments most neurons have a clear preference for some subset of the stimulus ensemble, while their response to other stimuli is nearly zero—but almost always not exactly zero, since neurons typically have a resting or background activity level (which can oscillate over time, so even if that level is subtracted from the PSTH—a common procedure in the literature, but see [131]—some sparse spikes are likely to remain). Therefore, normalizing all PSTHs individually would cause these spurious spikes to be greatly amplified and possibly affect the resulting factors.

Of course, normalizing each PSTH individually is also a valid (and simpler) approach, suitable when one is not dealing with neuronal spikes, or knows that all neural responses are significant and should be treated with equal importance by the factorization algorithm, but is inappropriate here.

2.3 Interpreting the resulting factors

Tensor CP decomposition encounters diverse applications such as analysis of EEG data [100], student interaction in a social network[65], weather patterns [65], and classification of hazardous gases [141], to cite only a few. In [145], examples are given of applications of NTF to neuronal activity for different experiments, including spatial navigation in mouse prefrontal cortex (see also [65]). In particular, those authors in [145] in analyze spike trains of neurons in response to different trial conditions, construct a tensor whose modes are neurons, trials, and PSTHs. As in most uses of CP decomposition, the factors are treated as the end-result themselves, with conclusions being drawn from the patterns present in each factor. This might be sufficient when the data is highly clustered or low-dimensional, or when the different types of trial (e.g. stimuli used) have little interaction, since in these cases participation of subsets of the data points across the different factors is likely to be mostly disjoint. This will be the case for the examples that were introduced in 1.3, as shown below.

Our tensor uses neurons, stimuli, and PSTHs as modes. We choose not to use individual trials since there is no expectation of change in the responses over different repetitions of the stimulus, as in e.g. a learning task. Furthermore, by using stimuli as a mode, we obtain more interpretable factors that directly relate to parts of the stimulus ensemble.

In Fig. 2.2, we can see that, for the ring model, the neurons present in each factor overlap slightly as we move from one stimulus to the next (sorted by orientation). This happens due to their response function decreasing slowly as a function of the difference between their preferred orientation and the orientation of the input stimulus. Now, since all neurons respond with the same temporal dynamics (minus noise), a single PSTH pattern is required to accurately reconstruct the responses to any of the 8 stimuli in the original tensor. Therefore, 8 components is all we need, regardless of the number of neurons used.

In the LN model with random stimuli/RF (Fig. 2.3), the data is highly clustered since each different RF responds quite differently to each different stimulus. This gives rise to separate factors for each combination of RF and stimulus. Since there are 10 different RFs and 2 stimuli (and they are all nearly orthogonal due to their random nature), we need 20 factors in total, regardless of the number of neurons having the same RF, since these will be all grouped together in the same factors encoding for their RF-stimulus combination.

2.3.1 When factors become entangled

In the general scenario where the same cells (or whatever entity is being encoded by the tensor) participate in multiple factors, or when the same stimulus can elicit multiple temporal response patterns, the complexity of the results becomes a challenge, and visual inspection alone is not sufficient (as will be shown in Chapter 6).

However complex the neural factors might be, their information is still useful, and in

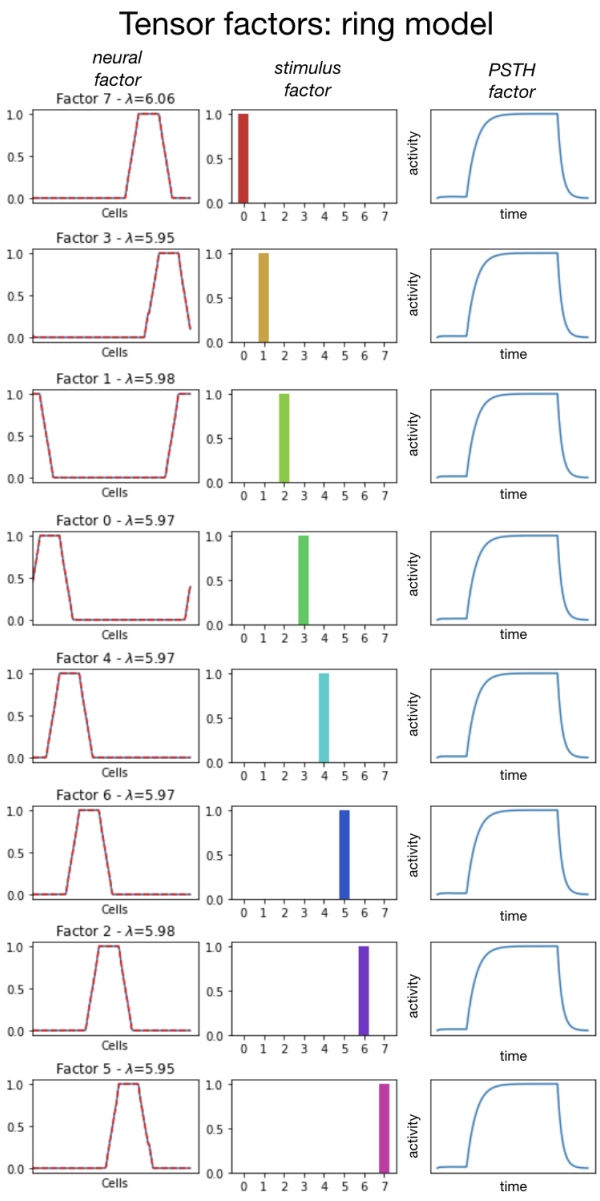

Figure 2.2: The 8 components (each row) obtained with tensor factorization for the ring model simulation. Each stimulus (i.e. orientation) used is assigned to an individual factor, and each corresponding group of cells overlaps with the ones from neighboring orientations. The PSTH factors are all identical, although each neuron's PSTH is really a noisy version of that pattern.

fact represented in fewer dimensions, which makes the set of factors a good candidate for being fed to other machine learning algorithms (in the same way that PCA can be used before a clustering algorithm, for example). On the other hand, the tensor structure of each component makes this non-trivial, since the vast majority of algorithms accept only matrices as input.

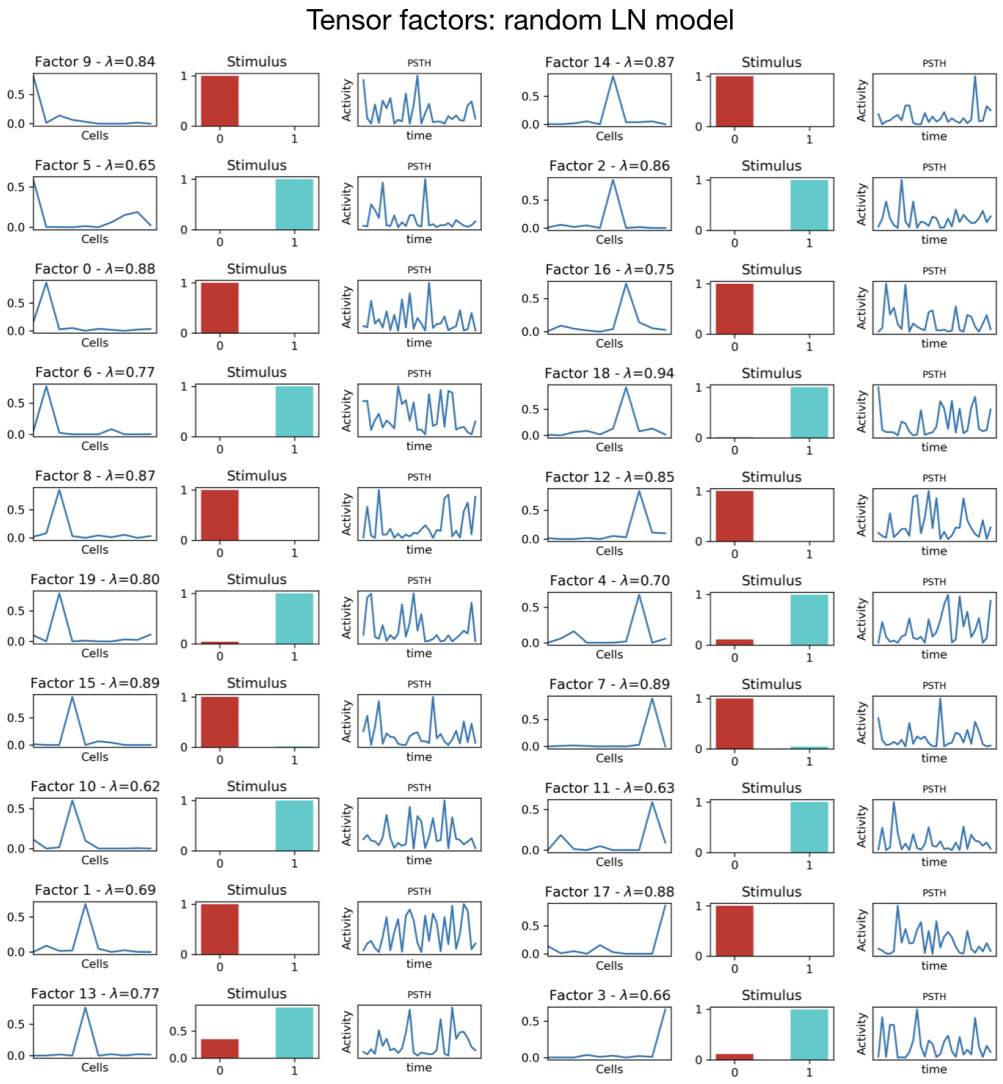

Figure 2.3: Examples of the 20 tensor factors obtained when using one neuron per type (N=10): each component 'explains' a single neuron for a single stimulus.

However, if one is interested in organizing only one of the tensor modes (in our case, neurons), its corresponding factor matrix $\mathcal{N}$ can be seen as providing an encoding of the full information in the components: since each factor is associated with other factors in different modes, one can "read" each row in $\mathcal{N}$ as expressing the amount of each component needed to represent that entity (in the case of a neuron, which PSTH patterns are present in its response to each stimulus).

Thinking now in terms of columns of $\mathcal{N}$ (i.e., the neural factors), what do they actually represent? In the literature, non-negative factors are usually thought of as providing a parts-based representation of the data. How can we extend this notion of 'parts' to neural factors?

Our data represents the responses of multiple neurons over time for trials of different stimuli. If we treat the recorded neurons as our sampled 'population', then at every instant there is a certain population activity profile, or *neural configuration*, that changes throughout the time course of each stimulus presentation. This means our neural factors could be deemed as parts of such configurations: each one shows the average population activity for a certain stimulus (or stimuli), but restricted to the cells that respond with a similar PSTH pattern.

We will develop in the next few sections an approach to using neural factors as lower-dimensional representation of the tensor data, which provides a principled way in which the information contained in the components can be used for posterior analysis using additional machine learning techniques.

Chapter 3

The neural matrix

- **Goal**: Use the tensor neural loadings as a basis for constructing kernels for manifold learning;
- **Method**: Use the rank of the neural matrix rather than reconstruction norm;
- **Results**: Rank of neural matrix indicates individual contributions of different stimuli; supports kernel development when factors are entangled; when orthogonal, reveals functional types.

3.1 Factor magnitudes

In the output of CP decomposition, each factor vector might have its own magnitude; as shown above (eq. 2.4), it is common practice to make them all equal to 1 and collect their original magnitudes into a single scalar, $\lambda_f = \prod_{k=1}^{M} \|\boldsymbol{f}_f^{(k)}\|$ where f indexes the component and k the factor mode, for an M-way tensor. Hence, reconstruction of an individual data point can be expressed as:

$$\boldsymbol{x}_i = \sum_{f=1}^{F} \lambda_f \boldsymbol{f}_f^{(1)} \circ \boldsymbol{f}_f^{(2)} \circ \ldots \circ \boldsymbol{f}_f^{(M)}, \tag{3.1}$$

and for the entire tensor as:

$$\tilde{\boldsymbol{T}} = \boldsymbol{X}^{(1)} \boldsymbol{\Lambda} \left(\boldsymbol{X}^{(2)} \circ \boldsymbol{X}^{(3)} \circ \ldots \circ \boldsymbol{X}^{(M)} \right), \tag{3.2}$$

where Λ is the diagonal matrix with entries $\Lambda_{ff} = \lambda_f$.

Now, assuming our neural factors represent the first mode ($k=1$), we can denote $\boldsymbol{X}^{(1)}$ by $\mathcal{N}$. It is helpful to think of the outer product between the remaining modes of each component as representing a basis vector (i.e. if one vectorizes the resulting $(M-1)$-way tensor, resulting in a unit-norm vector). (More precisely, this collection of vectors

represent a *frame* since in general it will be redundant, i.e. contain more vectors than the true dimensionality of the space that they span.) We can then define a matricized version of $\boldsymbol{T}$, call it $\boldsymbol{X}_{(\mathcal{N})}$, as:

$$\boldsymbol{X}_{(\mathcal{N})} \approx \mathcal{N}\boldsymbol{\Lambda}\boldsymbol{B} \tag{3.3}$$

with each column in $\boldsymbol{B}$ being

$$\boldsymbol{B}_{:,f} \equiv \text{vec}\left(\boldsymbol{f}_f^{(2)} \circ \boldsymbol{f}_f^{(3)} \circ \ldots \circ \boldsymbol{f}_f^{(M)}\right) \tag{3.4}$$

for $f = 1, \ldots, F$. In this case, $\lambda_f \boldsymbol{f}_f^{(1)}$ can be thought of as coordinate coefficients for such frame. Therefore, in order to make the entries in $\mathcal{N}$ have a straightforward meaning, we can multiply each of its columns, i.e. $\boldsymbol{f}_f$, by their corresponding λ_f:

$$\mathcal{N}_\lambda = \mathcal{N}\boldsymbol{\Lambda}. \tag{3.5}$$

The matrix $\mathcal{N}_\lambda$ now contains all information needed to reconstruct each neuron in terms of its 'stimulus-response' frame. Interestingly, rearranging eq. 3.3 as

$$\tilde{\boldsymbol{X}}_{(\mathcal{N})}\boldsymbol{B}^{-1} = \mathcal{N}\boldsymbol{\Lambda} \tag{3.6}$$

makes it remarkably analogous to the formula for projection of the data points onto its principal components in PCA: $\boldsymbol{X}\tilde{\boldsymbol{V}} \approx \tilde{\boldsymbol{U}}\tilde{\boldsymbol{S}}$ (from diagonalization of $\boldsymbol{X}$ by SVD as $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$, and with the tilde meaning only the top PCs are used in the approximation). In other words, we approximate each $\boldsymbol{x}_i$ as a linear combination of the 'stimulus-response' frame vectors $\boldsymbol{b}_f$ in $\boldsymbol{B}$:

$$\boldsymbol{x}_i \approx \sum_{f=1}^{F} (\boldsymbol{y}_i \cdot \boldsymbol{b}_f)\, \boldsymbol{b}_f. \tag{3.7}$$

where $\boldsymbol{y}_i$ stands for the i-th row in $\mathcal{N}_\lambda$. So $\boldsymbol{y}_i$ expresses new coordinates for $\boldsymbol{x}_i$.

Additionally, it might happen that a group of cells show the same response pattern, $\mathcal{P}$ to a single stimulus but that pattern ends up split into two (or more) factors. This can happen if other cells' responses share part of that pattern; then, an "intersection" factor becomes part of the result, with the remainder of pattern $\mathcal{P}$ being allocated to a second factor. If all responses have been prenormalized, the coefficients in the λ-scaled neural factors will automatically account for this splitting, with the sum of coefficients for the same cell and stimulus adding to approximately 1, regardless of the number of factors into which they were split).

3.2 Metric correction

In machine learning, when no prior information is known about the relationship between the coordinates describing features from the raw data, the assumption of an orthogonal basis seems reasonable and it means a dot product (inner product) between two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ in $\mathbb{R}^l$ can be computed simply by taking their elementwise product followed by summation:

$$\boldsymbol{a} \cdot \boldsymbol{b} = \boldsymbol{a}^T \boldsymbol{b} = \sum_{i=1}^{l} a_i b_i \tag{3.8}$$

assuming column vectors. However, when using the neural matrix $\mathcal{N} \in \mathbb{R}^{N \times F}$, we know that each entry $\mathcal{N}_{ij}$ is a coefficient specifying how much of the j-th factor is used to reconstruct the i-th data point. In other words, it can be interpreted as the projection of $\boldsymbol{x}_i$ onto the (non-Cartesian, i.e. non-orthogonal or oblique) coordinates given by the tensor factors. This means no assumption about orthogonality is necessary: we can compute the appropriate metric $g \in \mathbb{R}^{F \times F}$ for taking dot products in this frame, where g is the matrix composed of the pairwise dot products between the basis elements $g_{ij} = \boldsymbol{f}_i \cdot \boldsymbol{f}_j$. Then, the general expression for the inner product between vectors $\boldsymbol{a}, \boldsymbol{b}$ in the space spanned by the neural matrix columns becomes:

$$\begin{aligned} \boldsymbol{a} \cdot_{\mathcal{N}} \boldsymbol{b} &= \boldsymbol{a}^T g \boldsymbol{b} \\ &= a_1 b_1 (\boldsymbol{f}_1 \cdot \boldsymbol{f}_1) + \ \ldots \ + a_1 b_F (\boldsymbol{f}_1 \cdot \boldsymbol{f}_F) + \\ &\ldots \\ &+ \ a_F b_1 (\boldsymbol{f}_F \cdot \boldsymbol{f}_1) + \ldots + a_F b_F (\boldsymbol{f}_F \cdot \boldsymbol{f}_F). \end{aligned} \tag{3.9}$$

Notice this reduces to the familiar formula in eq. 3.8 when g is the identity (as in an orthonormal basis). The vector norm in this space is thus defined in terms of this modified inner product as:

$$\|\boldsymbol{v}\| = \sqrt{\boldsymbol{v} \cdot \boldsymbol{v}} = \sqrt{\boldsymbol{v}^T g \boldsymbol{v}}. \tag{3.10}$$

(Since g is symmetric positive semi-definite, this new norm still obeys symmetry, positivity and the triangle inequality.) Therefore, the squared distance formula analogous to squared Euclidean distance in this space becomes:

$$\|\boldsymbol{a} - \boldsymbol{b}\|^2 = \sqrt{(\boldsymbol{a} - \boldsymbol{b})^T g (\boldsymbol{a} - \boldsymbol{b})} = \|\boldsymbol{a}\| + \|\boldsymbol{b}\| - 2\boldsymbol{a}^T g \boldsymbol{b}. \tag{3.11}$$

This will be used in Chapter 4 when computing distances for diffusion maps.

3.3 Choosing the number of tensor factors

When using neural factor matrix $\mathcal{N}$ as an intermediate low-dimensional representation of the data, it becomes important to make an educated guess for the number of factors to choose when running NTF.

When reconstruction error is the main desideratum, one can simply pick the minimum number of factors that is sufficient to achieve an error below some prespecified tolerance. Our main goal, however, is to find a "good" kernel, or features, that maximally organize the data, regardless of how accurately is it reconstructed (although both requirements are certainly related to same extent). In general, the selection the number of factors F, i.e., the effective dimensionality of the latent space, is a tuning parameter whose selection is quite challenging and computational costly [155].

A different approach defines a notion of similarity between the components resulting from different initializations for the same F; ideally one looks for a robust solution, so an F should be chosen for which this similarity is high [145]. In [155], a an automated inference scheme under a variational Bayesian framework is proposed, and applied to tensor reconstruction/completion problems. However, for our particular problem of maximally organizing neurons (or entities), we do not necessarily need to satisfy the full tensor rank; we can restrict our rank requirements to that of the matrix $\mathcal{N}$, which we expect can make our problem more easily tractable.

In section 2.3.1 we searched for a conceptual meaning of neural factors. How could we extend those ideas to explain what the rank of $\mathcal{N}_\lambda$ means? First, note that the PCs are an economical (orthogonal) way of summarizing the information present in the columns of the original matrix, or a linear combination of parts of states s.t. the cells are maximally organized using the fewest columns. Thus, while neural factors are parts of neural configurations, PCs do not necessarily carry that same clear meaning. Nevertheless, the notion of rank that emerges from them still says something about the "diversity" of these configurations in terms of which neurons fire together and which don't, regardless of their specific response patterns (or parts of patterns). The space spanned by the PCs is the "neural space", and the rank of $\mathcal{N}_\lambda$ quantifies the dimensionality of this space.

Under this view, it makes sense for us to turn the problem of determining the optimal F into that of determining the rank of $\mathcal{N}_\lambda$. This non-trivial task will be elaborated through the remainder of this chapter.

3.4 The approximate rank of a data matrix

Most real-world matrices contain some degree of noise, redundancy, or uncertainty in its entries, so almost surely will have full rank (in the linear algebra sense), but rarely does that represent their "true" rank, that is, the actual number of coordinates (or basis vectors,

or latent variables, ...) required to express the idealized (i.e. noiseless) version of the data in it. It is also possible that the 'relevant' information lies in a lower dimensional space. In fact, quite often data matrices in the sciences will have a low-rank structure, a fact that is exploited by a multitude of machine learning algorithms and is precisely what allows any 'learning' to occur in the first place (see [137] for a theoretic discussion on why real-world data matrices are typically low-rank).

Perhaps the most commonly used method to provide a qualitative estimate of rank for a data matrix $\boldsymbol{A}$ is PCA. PCA eliminates that redundancy by re-expressing it in terms of an orthonormal basis formed by the eigenvectors of the sample covariance matrix. Its eigenvalues express the variance of the data when projected onto each eigenvector. The principal components (PCs) are then chosen as those eigenvectors with largest variance.[1] The variance plot (or Scree plot) can be inspected to provide a subjective estimate of the rank as the number of PCs that account for (or "explain") the majority of the data variance, and relies on there being a significant gap (a.k.a. an "elbow") between the variance of the first few "true PCs" *vs.* the remaining ones. Similarly to the situation with the reconstruction error plot, sometimes (e.g. as in Fig. 3.5) that can be easily done, but not always (Fig. 3.1. This approach can be made quantitative when the problem at hand allows one to specify the exact percentage of variance required.

The *numerical* ϵ*-rank*, r_ϵ, of a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times l}$ is defined as

$$r_\epsilon = \min\left(rank(\boldsymbol{B}) : \boldsymbol{B} \in \mathbb{R}^{n\times l}, \|\boldsymbol{A} - \boldsymbol{B}\|_2 \leq \epsilon\right),$$

where $\|\cdot\|_2$ stands for the spectral norm, and $rank(\boldsymbol{B})$ is the true rank of some matrix $\boldsymbol{B}$. In other words, it corresponds to the number of columns of $\boldsymbol{A}$ that are linearly independent for any perturbation of $\boldsymbol{A}$ with norm at most ϵ [136]. The singular values $\sigma_i, i \in [1, \min(n, l)]$ of $\boldsymbol{A}$ with ϵ-rank r_ϵ must, therefore, satisfy $\sigma_{r_\epsilon} > \epsilon \geq \sigma_{r_{\epsilon+1}}$. This rank should be robust to small perturbations on the choice of ϵ and the principal values σ_i. It becomes clear, then, that in practice this will only hold when there is a sufficient gap between the set of $\{\sigma_i\}, i \leq r_\epsilon$ and the remaining $\{\sigma_j\}, j > r_\epsilon$ associated with noise [52, 55, 136].

This really implies that the main requisite for determining approximate rank is determining which gap between two consecutive singular values is the "correct" one. Many possible spectra are possible, as depicted in Fig. 3.1; for some of them this is an easy problem, for others not at all. One approach for determining the optimal gap is by inspecting the 'density-of-states' (DOS) curve, which essentially computes a probability distribution over the range of singular values, i.e. the spectral densities [135, 136]. Starting from zero, one looks for the first sharp drop in density followed by a valley, which signifies a range of values that no singular value assumes, i.e. a prominent gap in the spectrum. It basically

[1] This is equivalent to computing SVD on the centered data and using the top right singular vectors as PCs (when data are row vectors), and computing the sample variances as the squared singular values scaled by $1/(N-1)$.

attempts to cluster the singular values into relevant *vs.* irrelevant groups. For cases like that of Fig. 3.1-D, there might be no such gap, so an empirical threshold parameter needs to be introduced for this choice to be made automatically [136].

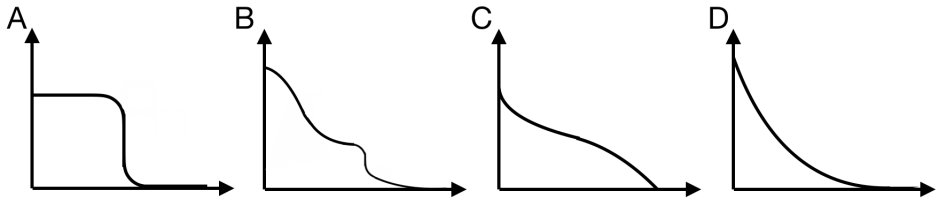

Figure 3.1: Examples of possible spectra of singular values of a data matrix. While the case in **A** is ideal for determining the approximate rank due to its well-defined gap between the first few singular values and the rest (approximately zero). In **B**, a choice must be made between two gaps: the first is taller but the second is sharper. In **C** and **D** it is difficult to identify any clear gaps.

Another class of methods take on a more rigorous statistical point of view. For example, classical results on the asymptotics of the PCA spectrum exist but for specific scenarios, such as when the noise variances in all PCs are known and are all the same [7]. Approaches based on Stein's Unbiased Risk Estimator (SURE) [17] adopt an unbiased estimator (assuming additive Gaussian noise) for the mean square error between the true signal and the reconstructed one, and use number of PCs that minimizes SURE [147, 18, 138]; this approach is combined with a cross-validation scheme in [139]. In [25], an exact distribution-based method is proposed for hypothesis testing and construction of confidence intervals for signals in a noisy matrix (again under Gaussian noise assumption).

Information-theoretic methods form yet another class of methods (e.g., [5, 120]) but typically require that the number of data points be large compared to their dimensionality [147]. Numerous other approaches have been proposed for rank estimation in specific applications (e.g, music transcription [87]), with matrix completion being a particularly popular one (e.g, [122, 71]).

It becomes clear after considering these examples that the choice of a rank estimation method is application specific and that different approaches may apply depending on the available information, such as the statistics of the noise present in the data. Also, note that the notion of estimating the rank of a general matrix $\boldsymbol{A}$ is usually associated with that of selecting the number of principal components in PCA, given that in general $\boldsymbol{A}$ is rectangular; furthermore, in data analysis centering the data points is also a welcome preprocessing operation. Although they can be made unit-norm, tensor factors are not orthogonal, which means an inherent degree of redundancy is expected. SVD naturally eliminates that redundancy by re-expressing it in terms of an orthonormal basis. Furthermore, determining the rank based on the variance explained by each component is attractive since it directly relates to our goal of "maximally organizing" the data points.

3.5 Determining the rank of a factor matrix

Because our matrix is a product of tensor decomposition, we find ourselves in the particularly advantageous position where we can produce an arbitrary number of matrices (by making different choices for the number of tensor factors, F, and random initializations for NTF) that should all express the same underlying structure. When F is small, we can imagine that there are not sufficient tensor factors to capture all the relevant structure present in the data. When F is too large, we may observe degeneracy of the resulting components (e.g. some factors capturing mostly noise, other meaningful factors being split up), which would tend to make for a poorer organization of the data. So, intuitively, we would expect to find an intermediary range for which the PCs remain approximately stable and optimally express the inner data structure. Yet another particularity is that, even if the original tensor noise could be reasonably approximated as Gaussian (which it is probably not, due to Poisson-like nature of spiking processes [93]), the "noise" introduced by under- or overfactoring is of a different nature entirely.

We propose an approach that aims to exploit this rather unique scenario. Instead of focusing on finding the largest gap in the spectrum, we will attempt to determine the "meaningful" principal values (PVs), i.e., the variances explained by each of the PCs, based on their evolution as F increases.[2] We will define a method for deciding on the approximate rank R of our neural factor matrix $\mathcal{N}_\lambda$ by using the illustrative example of the ring model simulation.

3.5.1 Ring model

Our idea arises from the empirical observation that some principal values (PVs) increase rapidly in value and then decrease as we choose a larger number F of factors, while others increase slowly and never reach a distinctive peak. The former behavior is usually observed for the top PVs; the latter is typical of high-numbered PVs (those toward the "end" of the spectrum). This is precisely what happens in Fig. 3.2-A: notice how all the first λ's seem to achieve a maximum before decreasing in value, while others appear to monotonically increase until they reach a plateau. Furthermore, for large F, while the gaps between all of the first 7 λ's decrease, the distinctive gap between λ_7 and the remaining ones remains stable.

How to explain what is happening? Starting with small F, the very first few PCs that emerge usually remain stable for some range of values of F; their variance is high and eventually reaches a peak or a plateau. This implies that the neural factors in that range have a considerable reconstruction power (since the PCs returned by PCA/SVD are

[2] We prefer to use PVs (i.e. the eigenvalues of the sample covariance matrix), instead of singular values of the data matrix since the former assign an intuitive meaning to the spectrum that is directly related to data organization in space, which is really our ultimate goal (as exposed in Chapter 1).

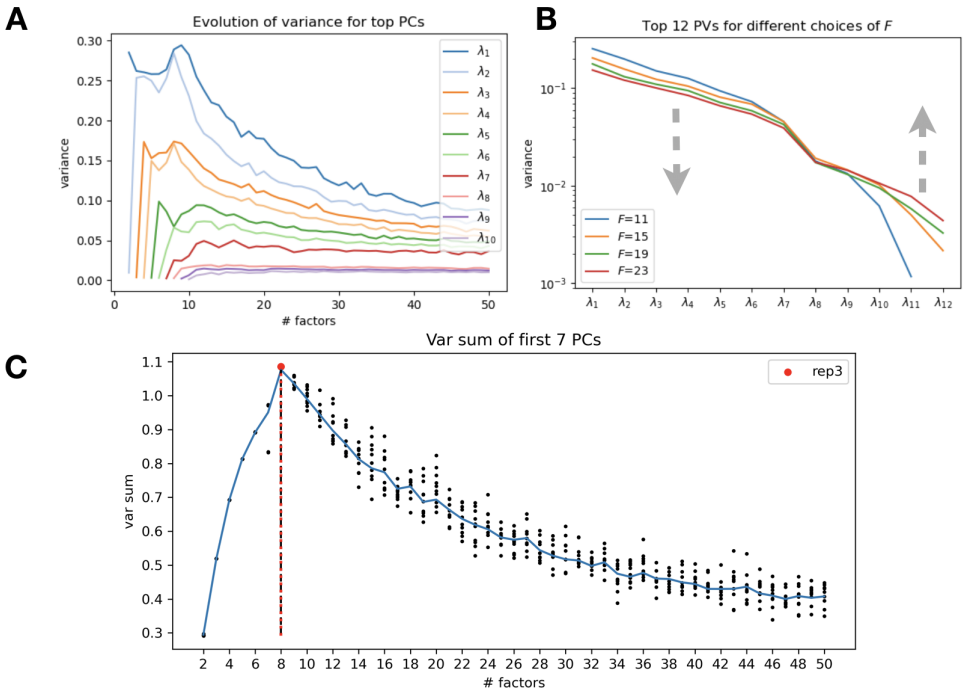

Figure 3.2: Determining neural matrix rank and optimal number of tensor factors for the ring model data set. Variances were computed as the mean across 25 repetitions of the NTF algorithm using a random initial guess. **A**: The evolution of each top PV (labeled as λ_i, i.e., variance explained by the i-th PC, for $i = 1, \ldots, 10$) is plotted against F. (Refer to text for details.) **B**: for large enough F, there is a tendency for the meaningful PCs (λ_1–λ_7) to lose variance explained, and for noisy PCs to slowly accumulate it (the arrows represent this trend). **C**: Sum of variances explained by the first $R = 7$ PCs. Each dot represents a single run of NTF, and the blue curve is their mean variance sum. The optimal F is defined as that which gives the maximum sum of the R first variances.

linear combinations of the neural factors in $\mathcal{N}_\lambda$).[3] Then, as we ask for more and more factors, eventually the "new" ones will "rob" variance from the top PCs (given that the factors together cannot reconstruct "more" than what's in the original data). As mentioned earlier, the best factors for some choice of $F = k$ may not be part of the best factorization with $F = k - 1$, so a more accurately description is that larger Fs cause some factors that remained stable over some interval of smaller Fs to no longer be a part of the solution, being "split" into two or more novel factors that, together, will approximately reconstruct the same parts of the data.

This phenomenon is illustrated in Fig. 3.3. In particular, observe the degeneracy that begins to occur subtly after $F > 8$ and becomes stronger as $F \gg 8$. All factor plots

[3]Note that factors with high reconstruction power are not sufficient for obtaining high variance PCs but the latter is conditional on the former, since factors that explain few neurons are unlikely to affect the overall variance of the data points. This is why using reconstruction error is a useful objective function in our approach despite not being our main concern.

use the same color scale, so notice how, for $F < 8$, some of the factors exhibit a non-uniform distribution of coefficients for the neurons in their "stimulus-preference cluster". At $F = 8$, rearrangement of their order gives an approximately block-diagonal identity structure, with near-perfect uniformity between the clusters. For $F > 8$, some clusters begin to overlap (as there are now more factors than response patterns), which in turn causes them to have smaller coefficients. For $F \gg 8$ some of them become "noisy", exhibiting discontinuities in the interval of neurons that they cover.

This is also reflected in the left singular vectors (SVs) (i.e. the linear embedding one obtains by projecting $\mathcal{N}_\lambda$ onto its PCs): note how the first $F - 1$ SVs organize the points into coordinates with large variance (these also look like spatial frequencies, as expected due to the nature of PCA); also note that, because of the block-diagonal structure of $\mathcal{N}_\lambda$, one of the factors can be expressed as a linear combination of the others, so that makes the last PV be approximately zero. Observe that, for large F, the same splitting phenomenon occurs, and the first SVs begin to cover fewer neurons, which brings their variance down and increases the relative importance of higher-numbered PCs. This latter fact becomes more clear by inspection of their corresponding PVs for the same range of F values (Fig. 3.4).

Note how, as F becomes large, the low-numbered PVs are decreasing. Meanwhile, the high-numbered PVs go from initially zero to some small non-zero values. This means the top PCs explain less and less variance, while the "bottom" PCs slowly increase their explained variance.

Fig. 3.2-B summarizes what is happening by plotting the initial spectrum (top 12 PVs) for different values of F beyond the ideal 8. Observe how, as F increases, the variances of the first 7 PCs are decreasing, while that of the PCs beyond the eighth are increasing (the arrows represent this trend). (This latter fact can only be noticed once a log scale is used, since their actual values are very close to 0.) The eighth PV, λ_8, can be seen pictorially as a "fulcrum" splitting the two groups.

We use the plot in Fig. 3.2-A to determine the rank R of our neural factor matrix $\mathcal{N}_\lambda$. Notice how all the first 7 λ's seem to achieve a maximum before decreasing in value, while others appear to monotonically increase until they reach a plateau. Furthermore, for large F, while the gaps between all of the first 7 λ's decrease, the distinctive gap between λ_7 and the remaining ones remains stable. This is used to set the rank R of the neural matrix as 7.

Finally, in Fig. 3.2-C we build the curve of sum of variances for the first $R = 7$ PCs. The optimal F is defined as that which gives the maximum sum of the R first variances; or, one can choose the specific repetition of NTF that yielded the highest variance sum (note that it is possible for an individual repetition for a different number of factors F' to have higher variance sum; in these cases, one can choose to use that particular result as the optimal one instead). Interestingly, the variability across different repetitions for the same F increases markedly for $F > 8$, which alludes to the method for selecting the number of

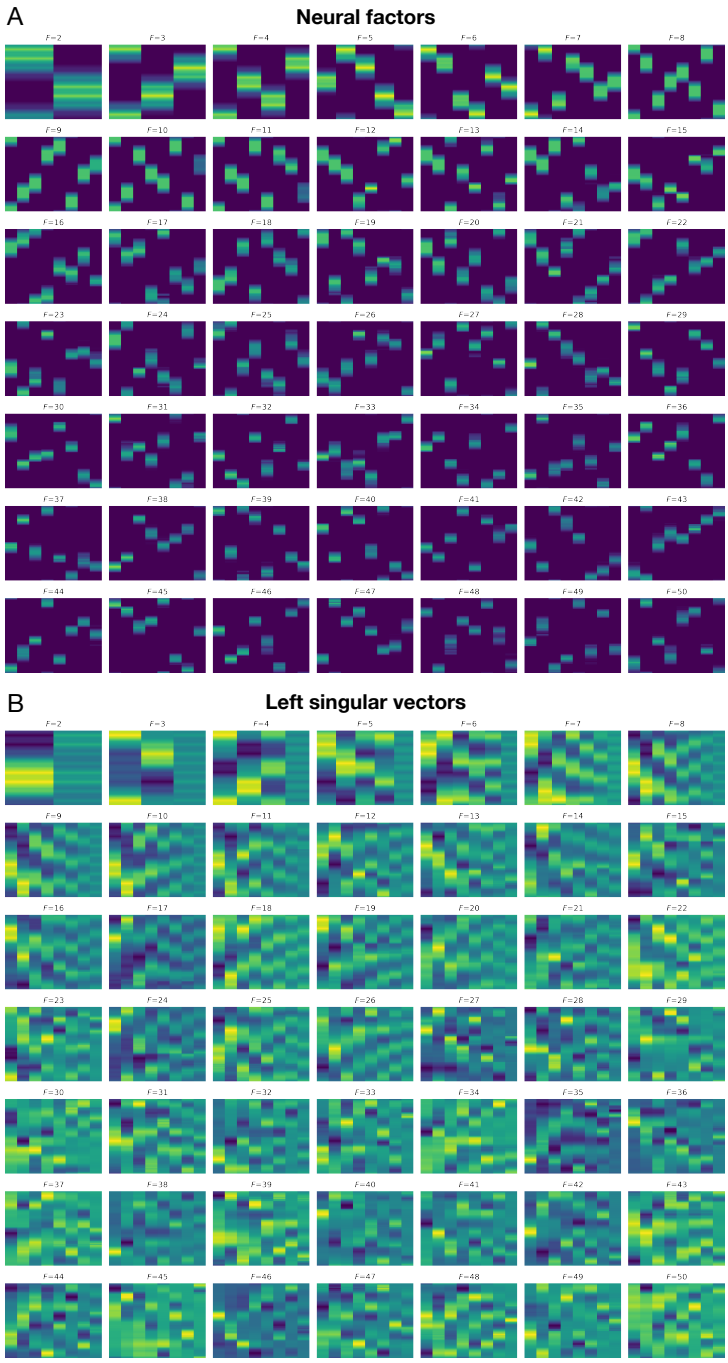

Figure 3.3: Visualization of the neural matrix $\mathcal{N}_\lambda$ and associated left singular vectors as heat maps for the ring model simulation described in Chapter 1 using various choice for the number of tensor factors F. Each plot in **A** depicts the first 8 columns of $\mathcal{N}_\lambda$ (i.e. neural factors sorted by their associated magnitudes λ in decreasing order), while the ones in **B** show the top 8 singular vectors—scaled by their corresponding singular values—also as columns (rows represent neurons).

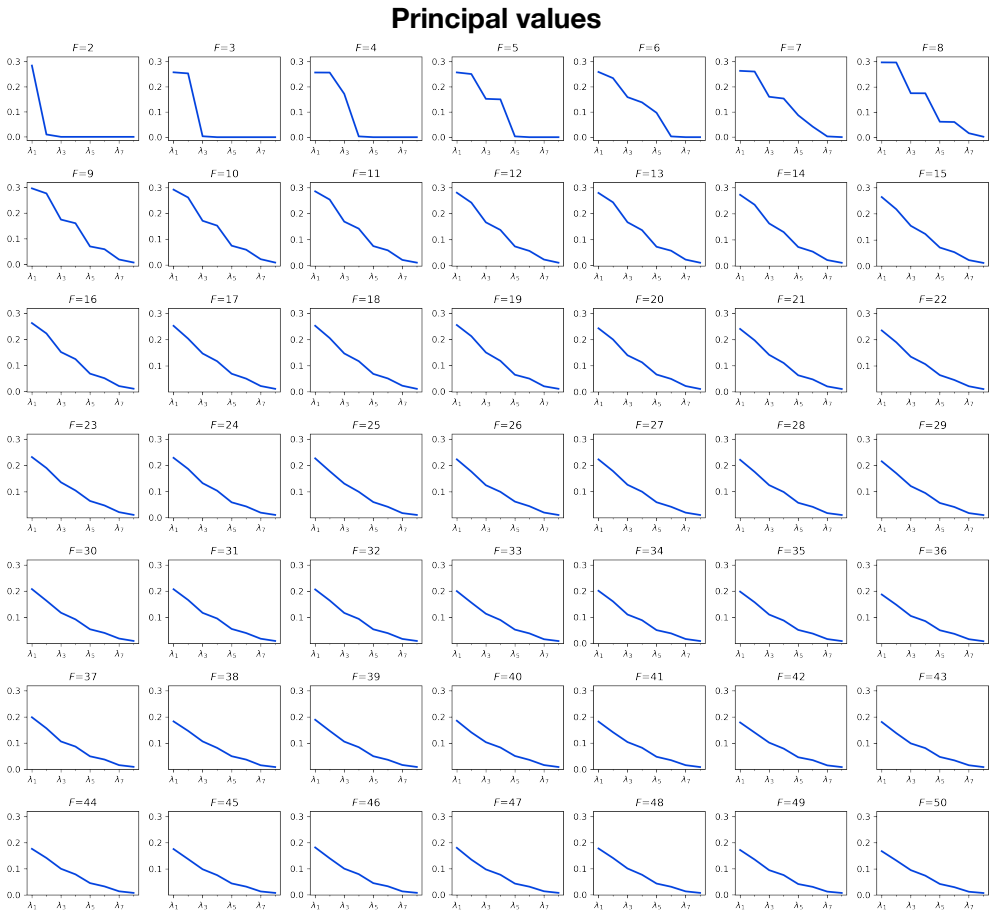

Figure 3.4: Evolution of the neural matrix's covariance spectrum for the ring model data set as the number of factors, F, increases. The top 8 principal values are shown.

factors based on similarity between different repetitions mentioned in section 3.3.

Referring back to the factors shown in Fig. 2.2, the choice of 8 tensor factors summarizes, in a balanced manner, which groups of cells respond to which stimulus orientation and with which intensity, and the neural manifold exactly captures the circular symmetry of the relationship between the neurons (Fig. 1.10). Observe that no information about connectivity was used as input to the algorithms. This means that, in fact, we are inferring 'functional connections' between the neurons, which in turn implies they participate in the same, or related, circuits.

3.5.2 Random LN model

Next, we analyze the analogous results for the random-RF LN model (from section 1.3). We shall see if the same logic applies to the decomposition of a different data set, and whether we can gain further insight into the interplay between tensor factors and the rank

of the factor matrix.

In the specific case when each neuron has its own RF (i.e., there is a single neuron of each "type"), the most reasonable decomposition is for each neuron to have its own set of factors, since there is almost no overlap between their responses. The neural matrix should thus resemble the identity when there is a single stimulus; for more stimuli, we have multiple exclusive columns for each neuron (see Fig. 3.5), but in both cases the principal value distribution is nearly uniform, with $N - 1$ non-zero PVs, since the stimuli are combined into the same PC for each RF type. When there is overlap between responses, however, individual neural factors begin to explain several neurons, with the amount of organization each factor imposes on the neuronal population ultimately determining the amount of variance with which that factor contributes.[4]

Therefore, it appears that in theory there is a general limiting case in which all PVs explain the same variance, which happens when F hits a ceiling (at most $F = N \times S$, where N is the number of neurons and S the number of stimuli), and explains the observed evolution of PV spectra as F increases (Fig. 3.2-B and Fig. 3.5-C). This indeed happens in our random-RF LN model; in practice, with real neural data several neurons share enough similarities in their responses that they will always share some of the factors.

The two plots in Fig. 3.5-A compare the neural matrices obtained when 20 and 25 factors are used. In the first one it is clear that each data point uses (or 'is explained by') two factors very strongly, and almost none of the others; concurrently, each factor describes the response essentially by a single data point (i.e. by a single RF). This is in agreement with what is known from the experiment: two stimuli were used, and each one produces a distinct response for each RF. When using 25 factors, however, note how some points now require more than 2 factors to be reasonably reconstructed, and not all factors have a clear selectivity for a single RF. This is in agreement with our observations for the ring model (Fig. 3.3-A). Examples of the resulting factors were given in Fig. 2.3.

3.5.3 WNIST

In this section we introduce a new data set called WNIST (short for Writers' NIST, in analogy with the famous MNIST data set from [86]). It contains a collection of handwritten digits organized by their writer's identity, i.e.: each data point contains one sample of each digit from 0 to 9 written by the same individual. Since in the original NIST database [53] each individual wrote the same digit multiple times, it is thus possible to have multiple data points corresponding to the same writer (Fig. 3.6). Each data point can be thought of as a collection of 'responses' (handwritten digit images) to different 'stimuli' (numbers

[4]By organization we mean either a clustered structure, where a group of cells are strongly explained by that factor, or a continuous one, where the factor contribution varies gradually across the population (resembling a continuous function). The use of a manifold algorithm in Chapter 4 is particularly suitable for inferring the underlying structure regardless of its kind.

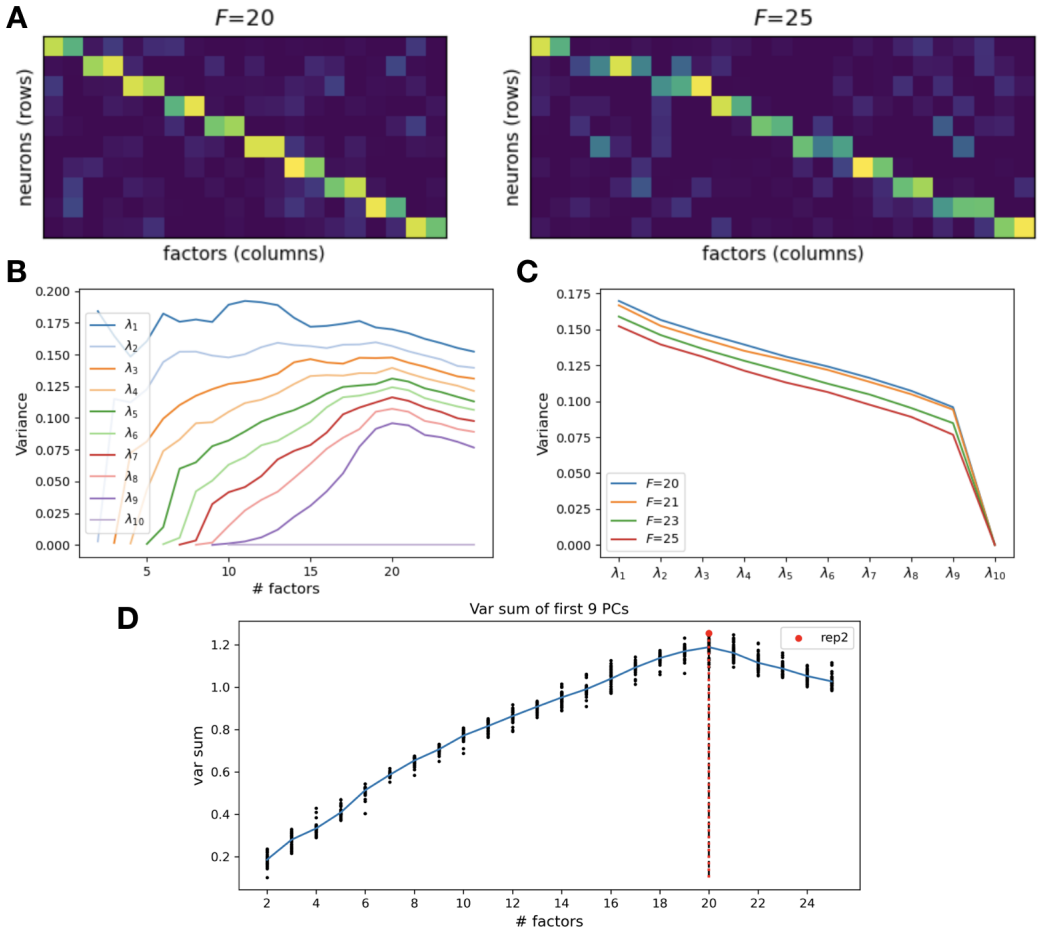

Figure 3.5: Determining neural matrix rank and optimal number of tensor factors for the random-RF LN model. Variances were computed as the mean across 25 repetitions of the NTF algorithm using a random initial guess. **A**: neural matrices with different number of factors chosen. (Refer to text for details.) **B**: The evolution of variances for increasing F can be analyzed to infer the rank R of $\mathcal{N}_\lambda$. Since we have 10 data points (rows of $\mathcal{N}_\lambda$), this limits the rank to be at most 10, so there are no new non-zero eigenvalues beyond λ_{10}. Regardless, the total variance still decreases due to disintegration of the factors. (Compare with Fig. 3.2-A, for which some of the variance lost by the top PCs goes to the bottom PCs.) Here it is clear that $R = 9$. **C**: Spectra for different values of $F \geq 20$. Note how the top 9 PVs decrease with increasing F. This is analogous to Fig. 3.2-B except for the absence of the right-hand side of the "fulcrum". **D**: As in Fig. 3.2-C, the sum for the first $R = 9$ variances is used to determine the optimal F for our LN model data set, which is 20 (confirming our expectation since there are 10 nearly orthogonal RFs and 2 nearly orthogonal stimuli).

0–9) by the same subject (writer), in direct analogy with our neuronal data sets, in which points represent a collection of responses (PSTHs) to different visual stimuli by the same neuron. Each individual digit image is vectorized so the data tensor still has 3 modes.

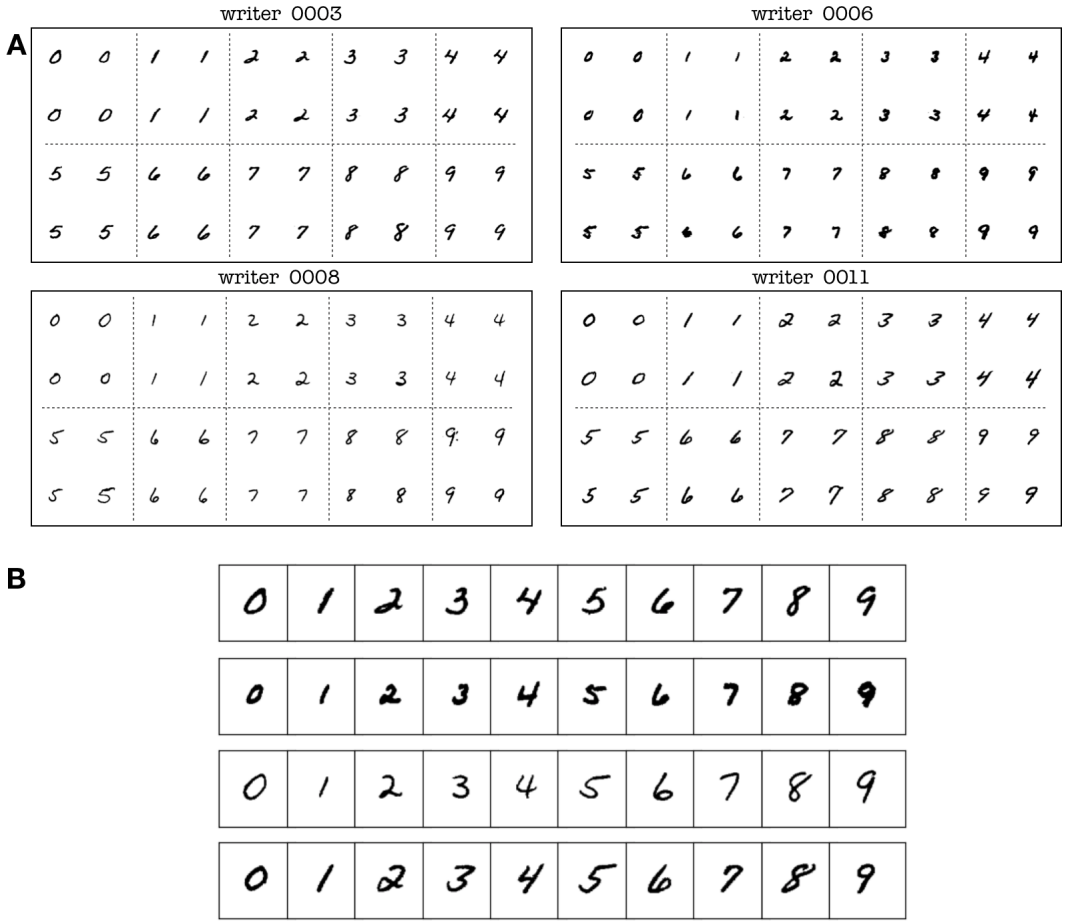

Figure 3.6: The WNIST data set. **A**: Multiple examples of all 10 digits by four different writers from the NIST's Special Database 19, partition *hsf_0*. Some writers have more similar style than others (e.g., 0003 and 0011 *vs.* 0006 and 0008). **B**: Data points (depicted here as rows) can be formed by selecting one example of each digit from the same writer; each digit for the same point can be thought of as a 'response' to a different 'stimulus' by the same subject, in analogy with our neuronal data sets.

We applied our algorithm to a subset of WNIST including 3 writers only, to ensure clarity of the results. Examples of factors obtained in this experiment are given in Fig. 3.7. In Fig. 3.8, we show the plots used for selection of the optimal number of factors and initialization. In the following chapter, we shall see whether the manifold obtained by using this choice of factors can "identify" the writers from their samples.

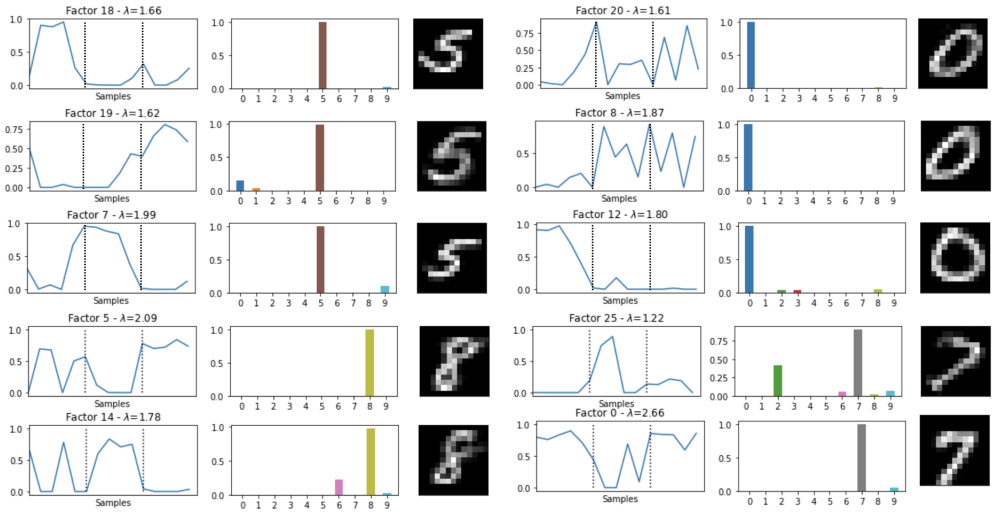

Figure 3.7: Examples of factors obtained for a few digits using WNIST with three writers and 5 writing samples per writer. The three factors for the digit 5 clearly separate the samples into three almost disjoint groups (samples are sorted by writer, separated by dotted lines). For the digit 0 there is a more variation in the style by the same writers. Digits 7 and 8 require fewer factors to be reconstructed. Note that one of the factors for 7 also reconstructs the digit 2 to a lesser extent; it also represents a style that is exclusive of the second writer.

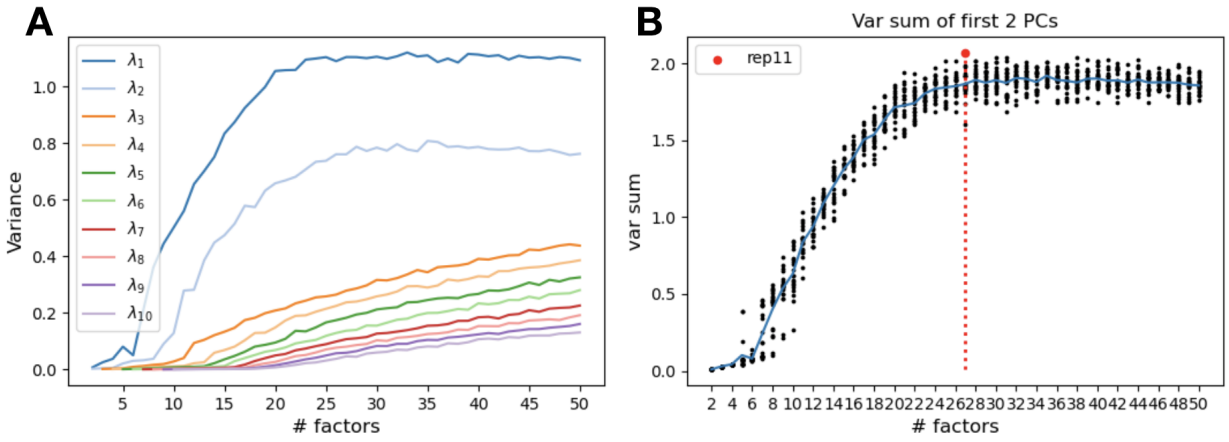

Figure 3.8: Determining the rank for the WNIST data set. **A** and **B** illustrate the procedure used for determining the rank R of the neural matrix, optimal number of tensor factors, F, and corresponding best initialization (as in previous examples). Here, although $R = 2$ is low, $F = 27$ is much higher, since each digit appears with possibly many different styles.

Chapter 4

Inferring manifolds from data points

- **Goal**: Representation of neural functionality as a manifold; nearby neurons on manifold respond similarly;
- **Method**: Diffusion geometry with kernel constructed from neural factor matrix; works even when there are abrupt structural changes in density;
- **Results**: Topology of manifold reveals functional circuitry for artificial examples; sparsification of the data graph add robustness to the diffusion kernel; mean flow ratio is proposed as a measure of the global connectivity of the manifold.

4.1 Diffusion maps

Manifold learning algorithms are based on the assumption that the data has an underlying locally low-dimensional geometry embedded in high-dimensional ambient space. In other words, the data approximately describes a low-dimensional manifold, i.e. the neighborhood around every data point resembles Euclidean space. Their goal is to find this underlying manifold and a parametrization for it in terms of intrinsic coordinates over the manifold. (We will discuss the validity of this assumption by means of examples, as well as propose adaptations to deal with non-ideal cases.)

In contrast with tensor factorization, which is a linear method (factors are linearly combined to approximate the original tensor), most manifold learning algorithms are nonlinear. In the case of diffusion maps [83, 31], which we will use in this work, the nonlinearity is introduced by the similarity kernel, introduced to describe local data affinities.

4.1.1 The algorithm

In short, the diffusion maps algorithm consists of the following steps [31]:

1. Given N data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^l$, choose a symmetric, semi-positive similarity kernel $k : \mathbb{R}^l \times \mathbb{R}^l \to \mathbb{R}$ and compute its value for each pair of points;

2. Using kernel values as edge weights, define an undirected weighted graph $G = (V, E)$ with each point as vertex and with weighted adjacency matrix $W \in \mathbb{R}^{N \times N}$ such that $W_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

3. Build a row-stochastic matrix, P, by dividing each i-th row of W by the degree of $\boldsymbol{x}_i$ (row sums of W):

$$P = D^{-1}W, \tag{4.1}$$

   where is the diagonal matrix $D_{ii} = \sum_j^N W_{ij}$. The leading eigenvectors of P may be used as a basis for a low dimensional representation of the data [12].

4. Compute the eigendecomposition of P, which has a complete set of eigenvectors ψ_i and eigenvalues $1 = \lambda_0 \geq |\lambda_1| \geq \cdots \geq |\lambda_{N-1}| \geq 0$. [1]; The first eigenvector, ψ_0, is the all-ones vector, $\mathbf{1}$ (since the row sums of P are all equal to 1; so $P\mathbf{1} = \mathbf{1}$). The remaining dominant eigenvectors $\psi_1, \ldots, \psi_k$ for some $k \ll N$ are then used to define diffusion coordinates that will give a low-dimensional embedding of the data.

5. By scaling each eigenvector by its corresponding eigenvalue raised to a power $t \geq 0$ representing diffusion time, we have the pleasant result that the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$) in these new coordinates is equivalent to *diffusion distance* in the data graph[2]. The latter is defined as the length of the vector computed as the difference between the probability distributions after one step in two random walks: one starting from $\boldsymbol{x}_i$ and another starting from $\boldsymbol{x}_j$, weighted by their empirical densities (or degrees, as define above)[31]. This is a notion of distance that reflects the local connectivity in the data graph, i.e., it will be small when there is a large number of short paths connecting $\boldsymbol{x}_i$ to $\boldsymbol{x}_j$, and is key to the motivation of using diffusion maps in the first place. The data is therefore mapped as:

$$\boldsymbol{x}_i \mapsto \left(\lambda_1^t \psi_1(i), \lambda_2^t \psi_2(i), \cdots, \lambda_k^t \psi_k(i)\right). \tag{4.2}$$

   One can either fix t as 1 or observe the evolution of the random walk/diffusion process through time (since $|\lambda_i| \leq 1 \forall i$ they will either stay at 1 or approach 0 as time increases, i.e. reach equilibrium).

[1] Although P is not symmetric, one can conjugate it by $D^{-\frac{1}{2}}$ to produce a symmetric matrix $P_s = D^{\frac{1}{2}} P D^{-\frac{1}{2}} = D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$ which is similar to P (in the linear algebra sense) (it must be symmetric since both $D^{-\frac{1}{2}}$ and W are symmetric); therefore it has real eigenvectors and eigenvalues. If it is diagonalized as $P_s = \Omega \Lambda \Omega^T$, P can now be expressed as $P = D^{-\frac{1}{2}} \Omega \Lambda \Omega^T D^{\frac{1}{2}}$. Then, by letting $\Psi = D^{-\frac{1}{2}} \Omega$ and $\Phi = D^{\frac{1}{2}} \Omega$, we have $P = \Psi \Lambda \Phi^T$, where $\Phi^T \Psi = I$. Thus, $P = \Psi \Lambda \Psi^{-1} = \Psi \Lambda \Phi^T$ is the eigendecomposition for P.

[2] Up to an accuracy δ which is a function of the number k of eigenvectors chosen [31].

Note that the eigendecomposition of P is inherently associated with that of the normalized graph Laplacian, which can be negatively defined (following [32]) as

$$L = D^{-1}W - I = P - I, \tag{4.3}$$

where I is the identity. Since $(P - I)x = Px - x = \lambda x - x = (\lambda - 1)x$, L has the same eigenvectors and its eigenvalues are one unit less than those of P.

The use of diffusion maps is justified based on the following main advantages over other manifold methods:

- Diffusion distances take into account bottlenecks in the graph: points that are connected by few disjoint paths will end up very distant from each other (even when they have a short path between them, i.e. small geodesic distance);

- The geometric features of the cloud of points can be exaggerated, enhancing the dimensionality reduction (e.g., see dumbbell example from [83]). This happens especially when the data is clustered or when there is a change in the local dimensionality of the manifold (see examples in section 4.4 below).

- Diffusion distance is naturally more robust to sampling and noise if compared to geodesic distance (which is used in other manifold methods, e.g. [134]). Additionally, one can choose an anisotropic modification of the kernel to deal with nonuniform densities (see section 4.2.1 below).

- Theoretical convergence guarantees: it can be shown [83] that if the data points are independently and uniformly distributed over a manifold $\mathcal{M}$, we have convergence of the discrete graph Laplacian L from eq. 4.3 to the continuous Laplace-Beltrami operator[3], in the limit of sample size $N \to \infty$ and $\varepsilon \to 0$. Put differently, the graph Laplacian, under ideal conditions, numerically approximates a continuous diffusion operator on the underlying manifold, by using only a finite subset of its points [32].

As possible drawbacks of this method, we can mention: the manifold assumption might not be valid, and although that is not a problem, sometimes this can give non-ideal results, as in the case when the data is a single Gaussian cluster (see below). Also, although an appropriate kernel bandwidth can be many times easy to pick, it is still a free parameter and in some cases the resulting embedding is highly sensitive to its choice.

[3]The Laplace-Beltrami is a generalization of the Laplacian when applied to a smooth function over a Riemannian manifold [105] (for n-dimensional Euclidean space, it coincides with the standard Laplacian); it appears in the solution to diverse physical problems, in particular those involving diffusion processes, hence the name diffusion maps. By computing the eigenvectors of L, we approximate eigenfunctions of $\mathcal{L}$ that represent states of diffusion stability.

4.2 Similarity kernel

A symmetric, semi-positive similarity kernel is used to essentially decide the neighborhood of each data point based on the original Euclidean distances between them. For each pair of data points $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathbb{R}^l$, it returns a number between 0 and 1 which determines how close, or strongly connected, they are. Typically a (rotation-invariant) Gaussian kernel is used:

$$k_\varepsilon(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/\varepsilon), \tag{4.4}$$

where $\|\cdot\|$ is the Euclidean norm of $\mathbb{R}^l$. This gives a continuous similarity scale from 1 (when $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are exactly the same point) down to some predetermined cutoff below which the kernel is considered to be zero (usually due to numerical precision). Since this kernel is symmetric, an undirected weighted graph $G = (V, E)$ can be immediately constructed by identifying each data point with a vertex and using similarities as edge weights. Notice how the Gaussian's scale, or bandwidth, parameter, ε, effectively determines each point's neighbors and the strength of their connections in the graph.

Note that with this type of kernel, the actual input to the diffusion maps algorithm is a distance matrix. But Euclidean distance computed in our raw multi-dimensional data is not adequate. So, as seen in chapter 2, NTF creates an intermediary representation of the neural data as a matrix for which Euclidean distance (with metric tensor) is actually meaningful. NTF acts, thus, as a distance kernel prior to the diffusion kernel.

4.2.1 Correcting for nonuniform density with anisotropic diffusion

Local variations in the density of the data points may appear due to several reason: nonuniform data collection, sampling noise or systematic bias towards parts of the data, or, even if perfectly homogeneous sampling is achieved, the inherent characteristics of the phenomenon being sampled may produce nonuniform accumulation of data (e.g. a uniform sampling over time of a speed-varying parametric spatial curve).

Gaussian-based diffusion propagation will capture not only the data geometry, but also its distribution, or density [83]. This may be desired in some cases, but in many applications one would like the algorithm to be able to uncover the same underlying structure (i.e. 'manifold') regardless of the distribution of the data points. That is indeed our case when studying neuronal activity sampled with multi-electrode arrays.

The degree (eq. 4.1) of each point, $d(\boldsymbol{x}_i)$, can provide a good approximation of the true data density [31]. When this density is nonuniform, L from eq. 4.3 actually approximates the Fokker-Planck operator[31]. In order to separate the geometry from the distribution of points, one must use a modified kernel:

$$\tilde{k}_\varepsilon(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{k_\varepsilon(\boldsymbol{x}_i, \boldsymbol{x}_j)}{d(\boldsymbol{x}_i)d(\boldsymbol{x}_j)}. \tag{4.5}$$

The algorithm follows in much the same way as in section 4.1.1 by making $\tilde{W}_{ij} = \tilde{k}_\varepsilon(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $\tilde{D}_i i = \sum_j^N \tilde{W}_i j$. Once can then proceed by computing the transition matrix $\tilde{P}$, and its corresponding Laplacian, $\tilde{L}$, which in the limit of large sample and small scales will approximate the Laplace-Beltrami operator (solely defined through geometry) [83].

Examples of density normalization using the anisotropic kernel in comparison with the original Gaussian kernel are given in Fig. 4.1. Non-uniformity in density can be present in different ways, as the two cases illustrate.

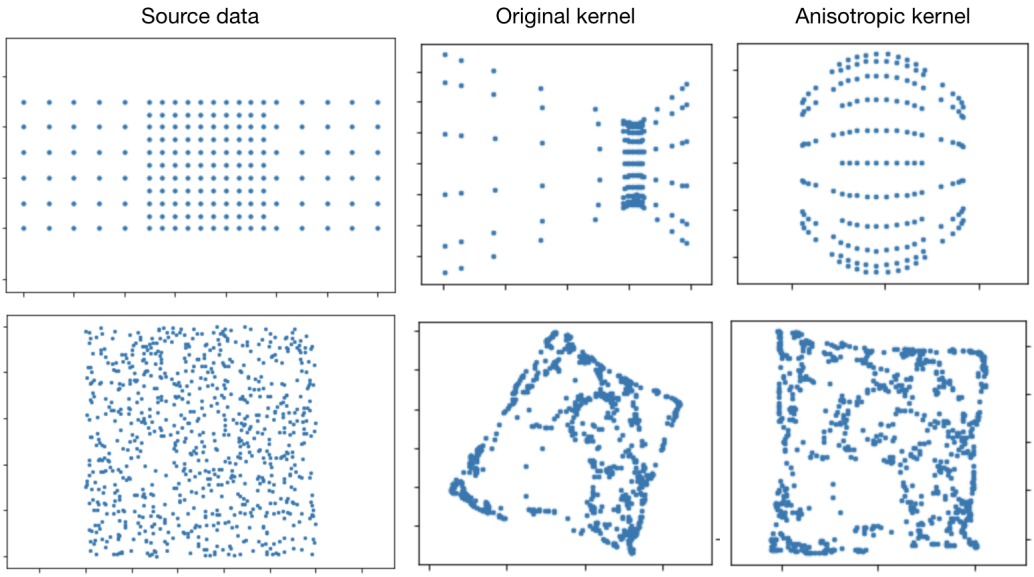

Figure 4.1: Comparison between using diffusion maps with the original kernel *vs.* the anisotropic version (normalized for density correction) on two toy data sets: one with discrete changes in density (top) and another with random changes in density (bottom). **Top**: whereas the embedding using the non-normalized kernel is majorly influenced by the small asymmetry between the two density transitions in the original data, the normalized one correctly "inflates"the middle denser section so that those points have comparable density to those in the lateral sections. **Bottom**: the square geometry is better preserved by the anisotropic kernel. Due to the "exaggerating" characteristics of diffusion maps (pushing apart gaps and bottlenecks), the wide holes in the source data become wider in the embedding. The anisotropic kernel, on other hand, makes that behavior more constrained by compensating for imbalances in the degrees of nearby points, giving a more faithful representation of the original data.

4.2.2 Choosing an appropriate kernel bandwidth

Ideally, we want ε to be just large enough to be able to capture local manifold patches. There are common heuristics for this value: using the median of all distances (or another percentile), the mean distance to the k-th nearest neighbor, or the maximal distance from

a point to its nearest neighbor in the data.

In [83], ε is the smallest non-zero distance to any neighbor, averaged over all data points:

$$\varepsilon = \frac{1}{N} \sum_{i=1}^{N} \min_{j:\boldsymbol{x}_i \neq \boldsymbol{x}_j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2. \tag{4.6}$$

In [84], an ε is chosen so that each data point is numerically connected to at least one other point. In [73], the authors define the max-min measure as a multiple of the maximal minimal distance per sample:

$$\varepsilon = C \max_i \min_{j:\boldsymbol{x}_i \neq \boldsymbol{x}_j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 \tag{4.7}$$

for some number $C \in [2, 3]$.

Other methods for computing an adequate bandwidth [32, 54] are based on inspection of the curve given by the sum of all weights $W_{ij}(\varepsilon)$ against ε in log-log scale (call it Z), whose slope is proportional to the intrinsic dimension of the data. One looks for a linear region of Z and chooses an ε within within that segment. In [54], this is made more precise by choosing the point of maximum of the gradient of Z, which is many cases should occur near the center of the linear region. One complication of this approach is that there might be more than one linear section in Z, or more than one local maximum in $\frac{\mathrm{d}Z}{\mathrm{d}\log\varepsilon}$, which imposes additional criteria for making this a truly automated choice.

There are also multi-scale approaches [154, 98] which assign a different bandwidth to each point. However, one must still use an heuristic when choosing an appropriate ε for each point. Usually this involves applying one of the above-mentioned heuristics over a prespecified number of nearest neighbors; but this means having to decide how many neighbors to consider for each point, which in turn also requires some form of heuristic. Multi-scale approaches also suffer from the fact that the kernel may no longer be symmetric, so an additional symmetrization step is necessary before computing the random walk matrix.

A review of these methods is presented in [88] along with guidelines for kernel bandwidth selection for different tasks. Still another multi-scale approach is proposed by BGH

In our experiments, we will use eq. 4.6 for computing ε. Its means advantages are simplicity and the fact that the same global . Intuitively, this provides for a more "fair" comparison between points throughout, although it is possible that some denser parts might end up more "compacted" than others. We rely on the anisotropic kernel to correct for that when the density variation is small.

It is worth mentionig

4.3 The data graph

In this section we will take a closer look at how the data graph is actually constructed depending on the kernel's characteristics.

4.3.1 Which nodes are connected to which

When a Gaussian kernel is used, either all points are connected to all others (albeit possibly through edges with very small weights), or there is a threshold below which the kernel value is zero (meaning no connection). Such threshold can be chosen:

- due to numerical precision (i.e. there is a limit to the smallest possible weight that can be represented by the machine), which is intuitively reasonable since extremely small weights shouldn't influence the results, but has the possible drawback that the resulting connections are not sufficiently selective to capture the true underlying geometry, especially in cases when the manifold has folds or is highly curved; or
- by adopting some heuristic that directly specifies the presence or absence of a weighted edge between two data points (e.g. using so-called ϵ-neighborhoods or n-nearest neighbors, as in [13]); these can be reasonable or not depending on what is known in advance about the data set.

Since the latter requires an additional tuning parameter, we will limit our analysis to the first option, and compare it with our own proposal for deciding on which connections to keep, based on applying edge sparsification to the original graph.

4.3.2 Graph sparsification preserving shortest paths

The analogy between graphs and electric circuits goes back to Kirchhoff [75]. A graph can be modeled as a resistor network by treating every edge as a resistor, with a voltage source connected between a pair of vertices. For an edge (i, j) with weight $w_{ij} \in \mathbb{R}_{>0}$, its resistance can be given by $1/w_{ij}$ [26, 10, 127, 129]. Our procedure begins by first computing resistances from the similarity weights. Due to the Gaussian kernel, these will grow exponentially as the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ (in the ambient space, $\mathbb{R}^l$) grows. This attributes, to any path p defined in G, a total length given by the sum of the resistances in the edges induced by p.

Form a new graph $G_R = (V, E)$ using resistances as weights (corresponding to edge "lengths"). The next step is to compute all shortest paths between all nodes (e.g., using Dijkstra's algorithm [42]); then, for every edge $(u, v) \in E$, delete that edge if the shortest path between nodes u and v contains more than one edge (i.e., is not the direct edge (u, v) itself). In graph-theoretic terms, the shortest weighted path length between u and v defines the *distance* between them, here denoted by $\mathrm{dist}(u, v)$.

Of course, for this to work we must guarantee that the shortest path we find from u to v will be preserved after all the pruning is finished. This can be easily proved: an edge (u, v) is deleted only if the shortest weighted path p between u and v contains more than 1 edge. If there is no shorter path, that means p contains a single edge u-v. Then u-v is not deleted, therefore the shortest path is preserved. On the other hand, suppose p contains more than one edge. Then, for any edge (i, j) in p, if it is deleted that means there is a shorter path from i to j. But, then, this implies that there was an even shorter path between u and v. That cannot be, otherwise p would not be the shortest path between u and v in the first place. So no edges in shortest paths are ever deleted, or, equivalently, all shortest paths are preserved, QED.

Since the (graph-theoretic) distances are preserved, the original graph G shares a formal notion of similarity with the sparsified one, $\tilde{G}$: they are said to be *1-distance similar* [10].[4] Additionally, $\tilde{G}$ is formally a *sparse 1-spanner* of G, as defined in [106, 4], and it has much in common with greedy geometric spanners [6].

Furthermore, in the spectral sparsification scheme from [128], a probability π_e of each edge remaining in the graph is computed in terms of its effective resistance. It can be easily shown that any edge e that is pruned by our own method will have π_e strictly less than 0.5, which indicates that our pruned edges are likely 'unessential' in a spectral sense as well.

Intuitively, the goal of sparsifying is to remove the redundancy provided by the weak connections while leaving the "essential" structure of the graph unchanged (Fig. 4.4). Naturally, the resulting sparsified graph is computationally more tractable due to its sparser adjacency matrix. But, most importantly, its resulting manifold is likely to be more robust to variations in the choice of the bandwidth parameter, ε (Fig. 4.2). This is highly desirable since it reduces the importance of the exact value set for ε, and indicates that the sparsified graph better captures the minimal structure, or 'skeleton', of the data (while at the same time having typically many more connections than the bare minimum to ensure connectedness, which would be a spanning tree).

Once the graph has been sparsified, its edges can reassume the original similarity weights. Optionally, one may do away with the weights altogether and use discrete edges, i.e. treat all weights as unity. This is possible because typically the remaining edges are either the ones with the strongest weights, or the weak ones responsible for making long-range connections, i.e., traversing "gaps" in the data distribution. Weak edges are responsible for "pushing" data points apart in the embedding, but this is also achieved by means of bottlenecks (due to the random-walk-related property of diffusion distances), which makes the actual weight values unimportant provided there are sufficient data points to make the bottlenecks sufficiently salient. Therefore, using unit weights may not be possible in all situations, but when applicable they result in an even stronger robustness to the

[4] Two graphs G and $\tilde{G}$ with the same vertices are σ-distance similar if $\mathrm{dist}_G(i,j)/\sigma \leq \mathrm{dist}_{\tilde{G}}(i,j) \leq \sigma \mathrm{dist}_G(i,j)$.

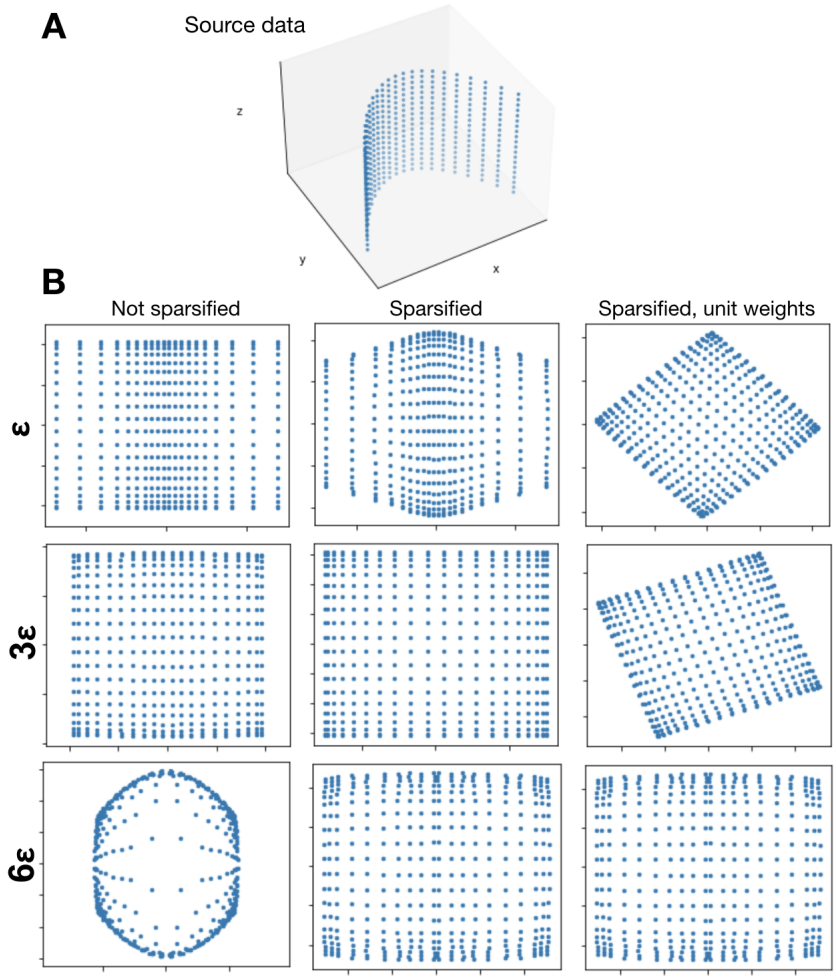

Figure 4.2: Comparison between computing (anisotropic) diffusion maps using the original data graph (G), the sparsified graph ($\tilde{G}$), and the sparsified graph using unit weights ($\tilde{G}^1$) for the curved square plane data set. Different rows represent using different scales of the global bandwidth parameter ε obtained using eq. 4.6 for a Gaussian kernel. Sparse data graphs remain stable over a larger range of ε. E.g, in the first row, note how the manifold from $\tilde{G}^1$ is not affected by the curved center, whereas those from both G and $\tilde{G}$, despite the anisotropic correction. When using 6ε as bandwidth parameter, the standard graph G fails to yield a square, while both $\tilde{G}$) and $\tilde{G}^1$ remain largely unaffected.

choice of ε. The final step is to apply the anisotropic normalization for density correction. In effect, the entire procedure can be treated as defining a new sparse, density-invariant kernel. We compare the three options (no sparsification, sparsification preserving original weight, and sparsification using unit weights) in Figs. 4.2,4.4.

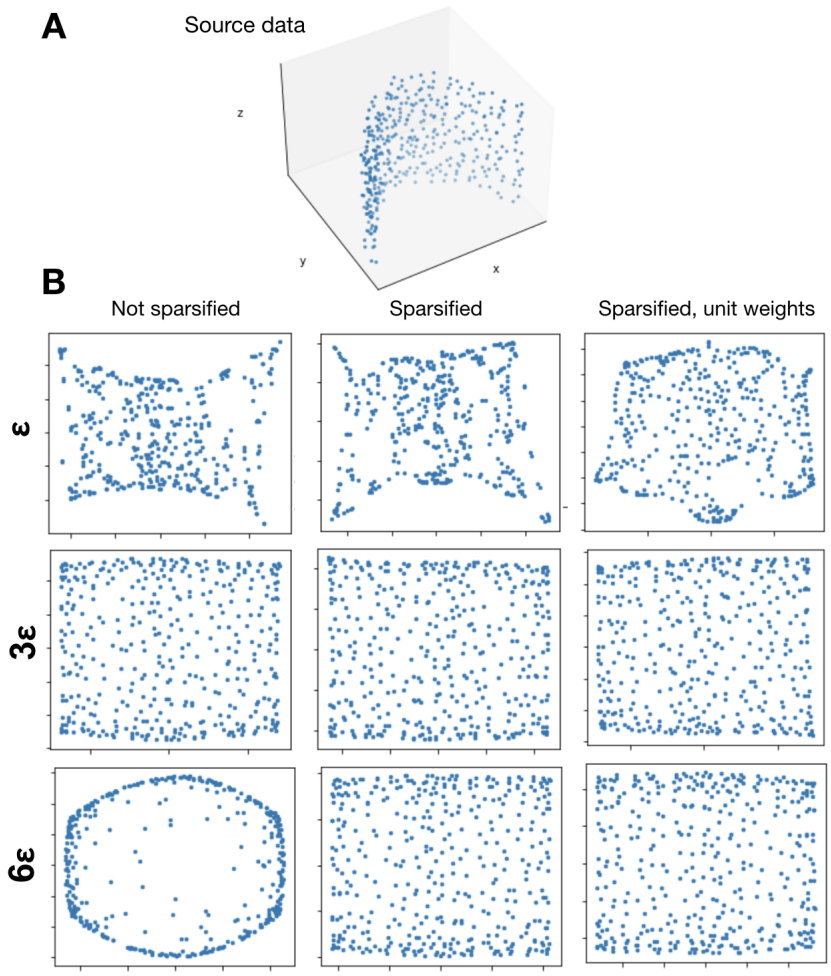

Figure 4.3: Comparison between computing (anisotropic) diffusion maps using the original data graph (G), the sparsified graph ($\tilde{G}$), and the sparsified graph using unit weights ($\tilde{G}^1$) for a noisy version of the curved plane from Fig. 4.2. Roughly the same observations apply, showing that the higher stability of sparse also applies under presence of noise.

4.4 Examples

We can now complete our analysis of the simulation examples from section 1.3 with the computation of their corresponding manifolds. In the previous chapter, we established the construction of their corresponding data matrices. Using the metric distance formula from eq. 3.11 and the global ε from eq. 4.6, the Gaussian kernel values can be computed and their data graphs constructed to be used with diffusion maps. Fig. 4.5 shows the resulting manifold for the ring model, and Fig. 4.6 that for the random LN model.

In Fig. 4.7, we show the resulting embedding for the WNÏST data set introduced in Chapter 3. In particular, note how each writer's samples are organized into well-separated clusters. In order to quantify the quality of the embedding, we computed the Normalized

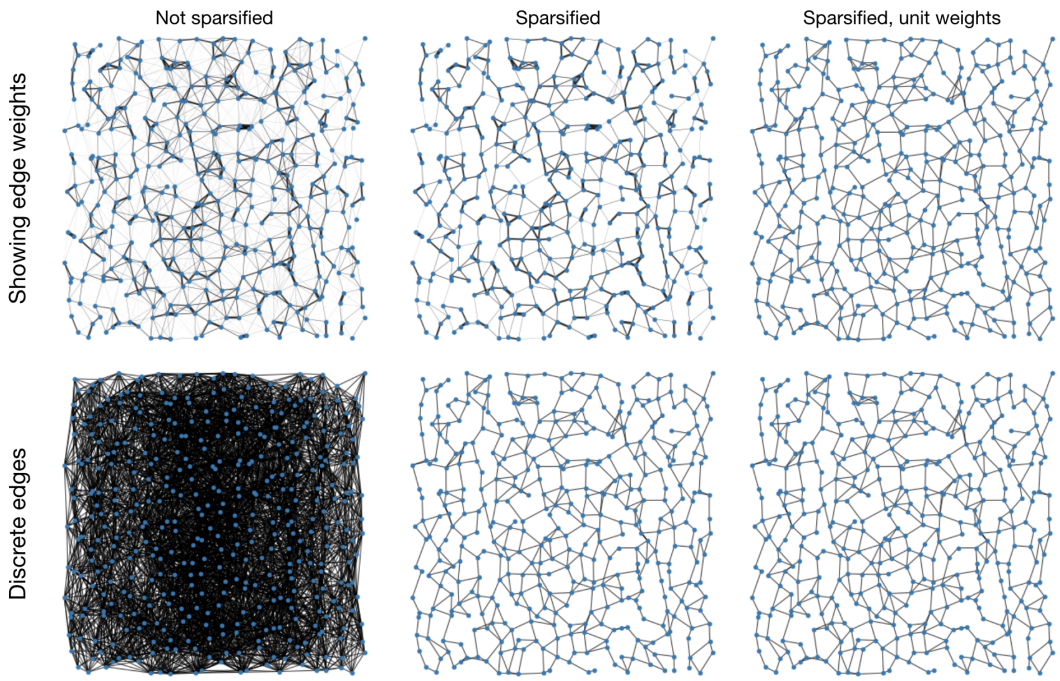

Figure 4.4: Graphs for the noisy curved plane data set from Fig. 4.3. **First row**: edge thicknesses are proportional to their weights. Note how the curved mid-section results in a higher connectivity in that region; this is alleviated by sparsification, which identifies most of those edges as redundant. When using unit weights, the stronger weights in the center section are treated equally to the weaker ones near the left and right borders. **Bottom row**: the same graphs as above, expect drawing edges with equal thickness, illustrating the dramatic reduction in the number of edges caused by sparsification (most edges in the top left plot have weights that are too small to be depicted proportionally).

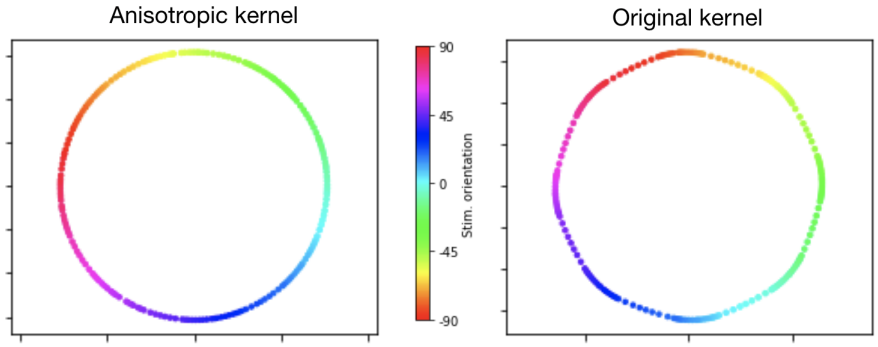

Figure 4.5: The neural manifold recovered for the ring model data set. In accordance with our expectations, neurons are sorted by orientation preference and continuously arranged around a ring. Notice that because there are 8 factors, this is reflected in 8 small concentrations of density near their centers. The use of the anisotropic kernel corrects for that non-uniformity and yields a perfect circular ring.

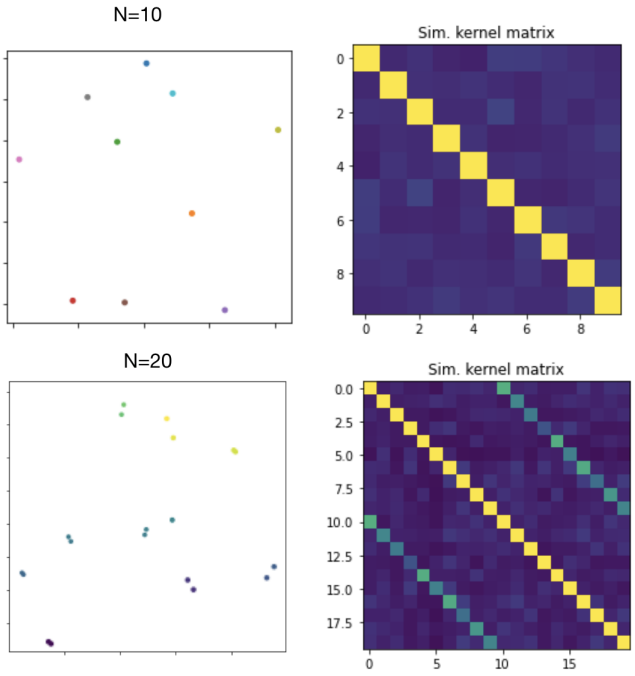

Figure 4.6: Embeddings for our random LN model. **Top**: Embedding obtained when using one neuron per type (N=10): the "manifold" is completely disconnected since all neurons are equally dissimilar, as shown by its similarity matrix (right). Each color represents a different neuron/RF. **Bottom**: If the experiment is repeated using two neurons for each RF type, we now have pairs of points clustered together (one pair for each type). The similarity matrix shows that every point is similar to another point shifted by 10 positions.

Mutual Information (NMI) score [143] between the true writer labels and the results of performing hierarchical clustering in diffusion coordinates for different choices of F and initializations (Fig. 4.7-B). This is compared with running diffusion maps directly on the matricized tensor, i.e. with the 10 digits in each sample concatenated into a single vector: the tensor approach gives a perfect match, while the matrix approach gives 21% only.

This result is empirical confirmation that applying diffusion maps directly on the concatenated data from multiple sources (or modes) may not be sufficient to provide a good organization. Firstly, such approach increases the original dimensionality of the data. Furthermore, computing distances using the combined vector will give the same weighting to all vector entries regardless of the mode they actually represent. This is undesirable since different sources might produce vectors with quite distinct properties, such as dimensionality or sparseness of its entries.

In [73], multiple manifolds are generated, one for each source of input (or stimulus), and their corresponding coordinates for each data point concatenated. The assumption is that the multiple input sources are related to the same phenomenon, so all manifolds should exhibit related coordinates, which is not the case in our targeted application (each

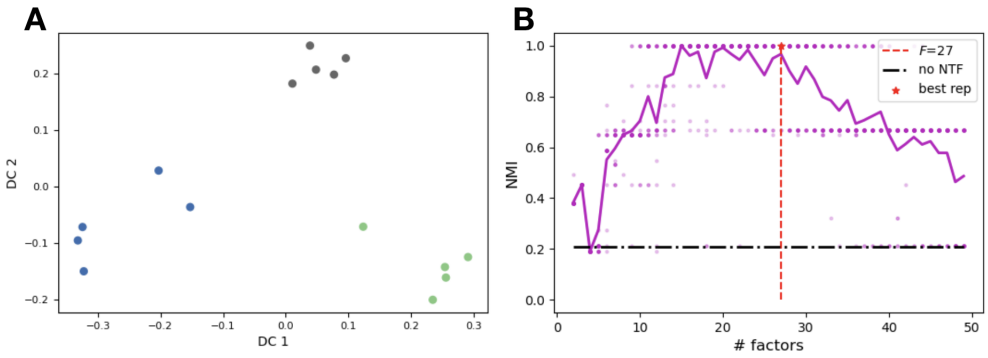

Figure 4.7: Embedding results for WNIST. **A**: Diffusion map embedding of the writing samples, colored by writer identity. The 3 writers are well separated. **B**: Normalized Mutual Information (NMI) scores (representing similarity, see text) between the true writer labels and a hierarchical clustering assignment in diffusion coordinates for different choices of F and initializations (mean value given by continuous curve). Note how the optimal F coincides with the plateau of high accuracy. For comparison we also plot the NMI values obtained using the optimal F and initialization found in **B** (1.0, represented by a star) and by running diffusion maps directly on the matricized tensor, i.e. all 10 digits concatenated into a single vector (0.21).

stimulus can drive a specific activity configuration over the neuronal population, resulting in highly different individual manifolds). We would like to produce a unified manifold with its diffusion coordinates describing the effect of all stimuli combined, which justifies the use of our approach.

4.5 Quantifying manifold "continuity"

It is highly desirable to have a means of comparing different manifold embeddings in some objective way. As will become clear after we present our biological results in Chapter 6, of particular interest is a measure to quantify how "clustered" a embedding is, or, from the opposite perspective, how "uniformly continuous" its manifold is.

One way to characterize this is the following: If one starts from a particular node in the data graph, how "easy" will it be to visit other nodes, considering that strong connections are easily traversed but weak ones required more work to be crossed? Intuitively, even if two nodes i and j are not directly connected by a strong edge, but have common neighbors with strong connections, then it is still easy to move from i to j. This is basically the same notion that motivated diffusion distances, and can be well expressed by the *effective conductance* between i and j, i.e. the inverse of the traditional concept of *effective resistance* that is applied to electrical networks. Again, we will exploit the duality between manifolds and their corresponding networks to relate graph-theoretic properties back to the manifold topology.

4.5.1 Effective resistances

The effective resistance R_{ab} between two vertices a and b in an electrical network is the resistance of the entire network when we treat it as one complex resistor (i.e., if we reduce the rest of the network to a single edge) [127]. Consider a graph $G = (V, E)$. For any two vertices, u and v, that are connected by an edge, the effective resistance can be calculated in several ways [130]: (1) Using transformations of the corresponding electrical circuit (including the so-called star-mesh transformations); (2) Injecting unit current into the vertex a and taking the same current out from the vertex b. The effective resistance is then equal to the difference of potentials in a and b; and (3) Through the spectral decomposition of the corresponding graph Laplacian, L. In particular, it can be shown that the effective resistances R_e^{eff} for all edges $e \in E$ can be computed as the diagonal entries in the matrix

$$R = BL^{+}B^{T}, \tag{4.8}$$

where L^{+} is the pseudoinverse of L and $B_{m\times n}$ is the *signed edge-vertex incidence matrix* [128]. This means there is a straightforward way of computing effective resistances in G, and by consequence, effective conductances.

4.5.2 Mean flow ratio

The maximum flow problem is a classic problem in graph theory and was originally formulated as a means to optimize railway traffic between cities [119]. It applies to multitude of scenarios that can be represented as a weighted graph where each edge $u - v$ has a 'capacity' (its weight) representing how much flow can be transported by that edge. Several algorithms exist for solving it with varying time complexities. Of course, it directly relates to our problem posed above, since we can treat edge conductance as a capacity, i.e. a measure of the ease with which an edge can be traversed. Since we use effective conductances, though, nodes in the original kernel graph that were connected by a weak edge but that had several disjoint paths connecting them will now be connected by a strong edge.

Therefore, by creating a new graph G_C with effective conductances as edge weights, the max flow between two any two nodes u and v in G_C can tell us whether there is a path where all edges are strong (high effective conductance, meaning all points along this path were highly connected in the original graph G) or whether there are any big gaps along the way (case in which the max flow is small).

For an individual node i, the ratio ρ_i between its average max flow to all nodes and the average max flow to the nodes adjacent to i is a measure of "connected" i is to all nodes

compared to his proximity to its immediate neighbors:

$$\rho_i = \frac{\sum_{j=1}^{N} \text{maxflow}(i,j)}{\sum_{k,(i,k)\in E} \text{maxflow}(i,k)} \tag{4.9}$$

where E is the set of edges in G_C and $j \neq i$. We call this quantity the *flow ratio* of i. Gaps in any part of the manifold will cause the numerator to be small, so ρ_i is also small.

Finally, order to have a global conductance measure of how well-connected, or "continuous" the manifold of G is, we propose the *mean flow ratio* quantity, ϕ_G, defined as the average flow ratio over all nodes in G_C:

$$\phi_G = \sum_{i=1}^{N} \rho_i. \tag{4.10}$$

The entire procedure is summarized as follows:

1. Compute the data graph G using the anisotropic kernel (choosing ε with, e.g., eq. 4.6);
2. Compute edge resistances $R_e = 1/W_e$ (and, optionally, sparsify G as in section 4.3.2);
3. Compute effective conductances as $1/R_e^{\text{eff}}$, with R_e^{eff} obtained from eq. 4.8;
4. Define new weighted graph G_C where the weight for edge $e = (u, v)$ is the effective conductance between u and v.
5. Compute the max flow between all pairs of nodes.
6. For every node i, compute its flow ratio ρ_i as in eq. 4.9.
7. The graph mean flow ratio ϕ_G is the mean flow ratio over all nodes (eq. 4.10).

Of course, sparsifying the graph makes the max flow computations considerably more efficient, while giving very similar results.

This quantity is high when, on average, the flow from a node to other nodes in the network is low (e.g., due to bottlenecks or big gaps, or even disconnected components) compared to the flow to its adjacent neighbors. When the flow is nearly the same between any two nodes (e.g. in a clique or on a grid), the quantity is close to 1. Also note that, because the max flow to neighbors constrains the flow to other nodes, $\rho_i \leq 1$.

Since we compute a ratio, the actual weight values are not important: only their relative strengths. E.g. a path graph with constant edge weights will have $\phi_G = 1$ regardless of their actual values; but a graph with two disconnected (or nearly disconnected) paths will have a smaller ϕ_G, since the numerator of ρ_i will decrease (in this example, if both paths have a similar number of nodes, we have the pleasant result that $\phi_G \approx 0.5$, since the flow to half the nodes is nearly zero for every node i).

Another pleasant property is that ϕ_G is little affected by 'outliers', i.e. isolated data points that are far from all others. Assuming each node is connected to at least one other point (otherwise it is easy to identify the outliers), their ρ_i should be close to 1 since they will be weakly connected to all nodes, which will not tend to affect the global mean ϕ_G give that outliers by definition are very few. Importantly, this is in stark contrast with the graph-theoretic notions of conductance and algebraic connectivity, which are highly affected by the presence of outliers.

Below, we show examples of the mean flow ratio applied to clustered *vs.* non-clustered toy data sets.

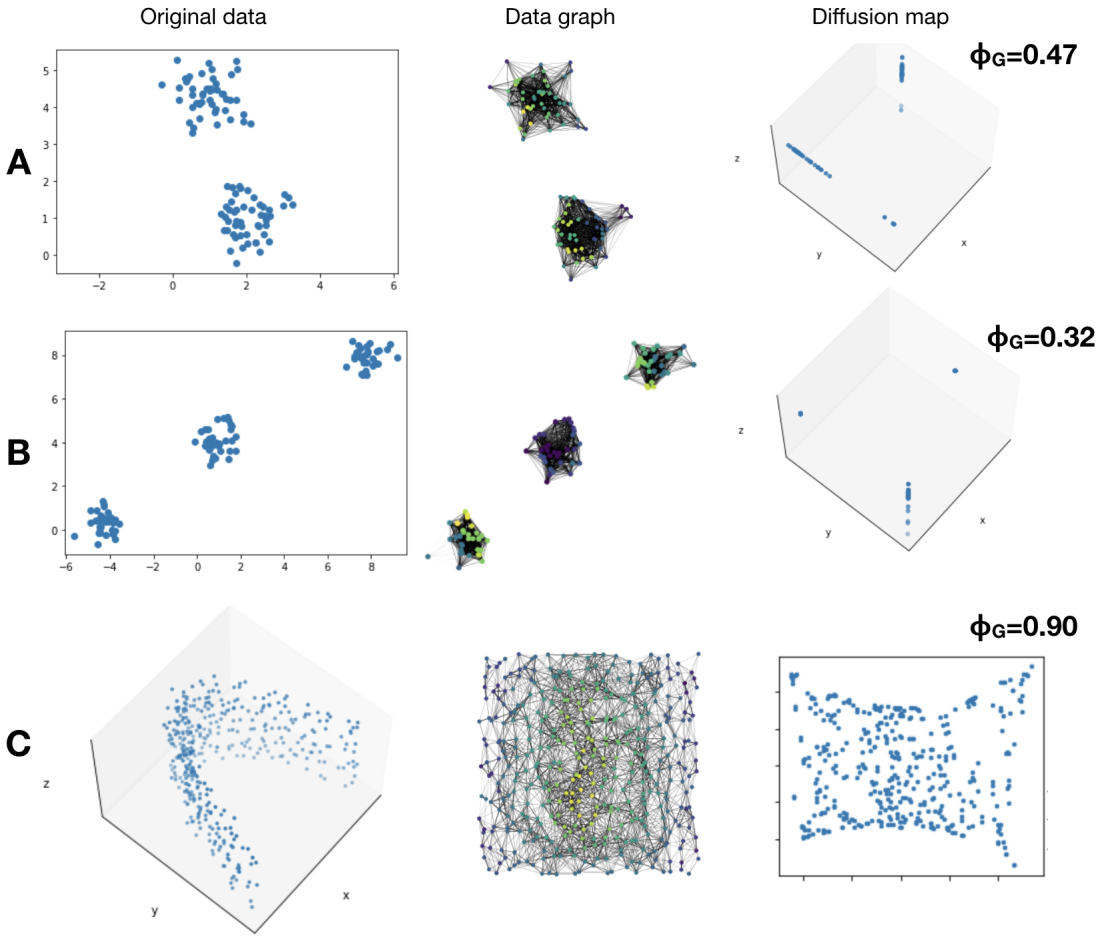

Figure 4.8: Mean flow ratio (ϕ_G) values for toy data sets. Middle column shows data graph with unweighted edges and nodes colored by their individual flow ratios (ρ_i). No sparsification was used. **A**: two isolated clusters have $\phi_G = 0.47 \approx 1/2$. **B**: three isolated clusters have $\phi_G = 0.32 \approx 1/3$. **C**: a curved, noisy plane gives $\phi_G = 0.90 \approx 1$.

Part II

Results in neuroscience

Chapter 5

Flow stimuli: generating an appropriate stimulus ensemble

- **Goal**: develop stimulus ensemble rich enough to engage much of the mouse's visual system through V1.
- **Method**: combining novel flow stimuli with standard laboratory stimuli
- **Results**: developed flow stimuli with guaranteed spatial and temporal frequency support; developed stimulus generator and deployed it in the laboratories of our collaborators in UCSF and Duke; published results confirming that the awake, behaving mouse responds in novel ways to the flow stimuli.[1]

5.1 Why use flows?

The mouse has become a major model for studying vision because of the genetic, imaging, and molecular tools available [45]. Studies have revealed relationships between macroscopic states of the brain and activity in visual cortex (running *vs.* stationary [102, 50], pupil size and activity [109, 96], and visual interest (e.g., [61, 96, 142]). However, a basic conundrum has arisen: behaviorally, mice are capable of sharp, visually-mediated behaviors [107, 56, 19], such as accurate prey capture [66], but when assessed using standard assays, such as spatial frequency gratings (Fig. 5.1), the mouse appears to have very poor vision. Although orientation-selectivity has been found [101], receptive fields are large (typically $\sim$25 degrees^2) when estimated by spike triggered averaging, and spatial frequency tuning is concentrated below 0.08 cycles/degree (cpd). While this motivates the use of gratings at 0.04 cpd in experiments, it raises the question: How does the visual system perform so exquisitely in natural tasks?

We show here that ecologically-relevant stimuli can exercise mouse visual cortex in novel and manifold ways. While plaids [70, 103] and random-dot kinematograms [43,

[1] Most of the content of this chapter has been previously published in [44].

132] are a step beyond gratings, the leap to natural images (e.g., [20]) is more common (e.g., [48, 114]). However, natural images are difficult to obtain [116], difficult to control parametrically, and difficult to analyze beyond second-order [124].

For a mouse running through a field, the visual projection is like a 'waterfall' flowing past, with oriented segments moving into and out of occlusion relationships (Fig. 5.1A)[157]. This visual metaphor motivates our stimuli. We approximate such patterns with a class of visual flows comprised of dots, so they are more natural than drifting gratings but can be parametrically controlled in their orientation (content and angle), spatial frequency, and direction of motion. We call them flows because, intuitively, they consist of a field of particles (either dots or dotted line-segments) dropped into a 'flowing river'. More formally, each dot is displaced along a vector field in space and time and follows a dynamical system [1]. When the orientation structure is removed, the flows reduce to random-dot kinematograms; when the temporal structure is removed, the flows reduce to static Glass patterns [51]. Thus they are rich in geometry and, for humans, the perception of such flows differs from strictly aligned patterns [90, 156]. Parametric variations in orientation, direction, etc., define an ensemble of stimuli.

We here explore activity in mouse V1 in response to the flow ensemble. In many cases flow stimuli elicit more vigorous responses than drifting gratings, particularly at high spatial frequencies 3–5 octaves above 0.04 cpd. Some V1 neurons are classical, resembling feature detectors, while others exhibit a mixed selectivity rarely reported in early visual cortex. The rich ensemble of selectivities in V1 may equip the mouse to behave in the natural world.

5.2 Design of flow stimuli

Flow stimuli were designed as variations of Glass patterns [51] in which patterns formed by n collinear dots move with direction determined by an underlying vector field, or flow field (see examples of possible flow patterns in Fig. 5.2). If $n > 1$, the flow element maintains its orientation orthogonal to its direction of motion. At each presentation, the initial position of each flow element is a normally-distributed random variable with a given standard deviation and whose mean is one of the evenly-spaced positions on the screen (determined by the chosen stimulus spacing).

The flow field is created by partitioning the screen into a grid of square tiles, each being assigned a unit vector in $\mathbb{R}^2$ whose direction is given by a normally-distributed random variable. Since the dots move like a flock, we exploit algorithms from computer graphics to make their motion more naturalistic [110, 111] as well as to avoid overlapping [112]. As a result, the flow elements can make successive changes in direction as they drift across the flow field by following a smooth and continuous trajectory, without abrupt changes in direction. They will also wrap around the screen boundaries to preserve a constant number

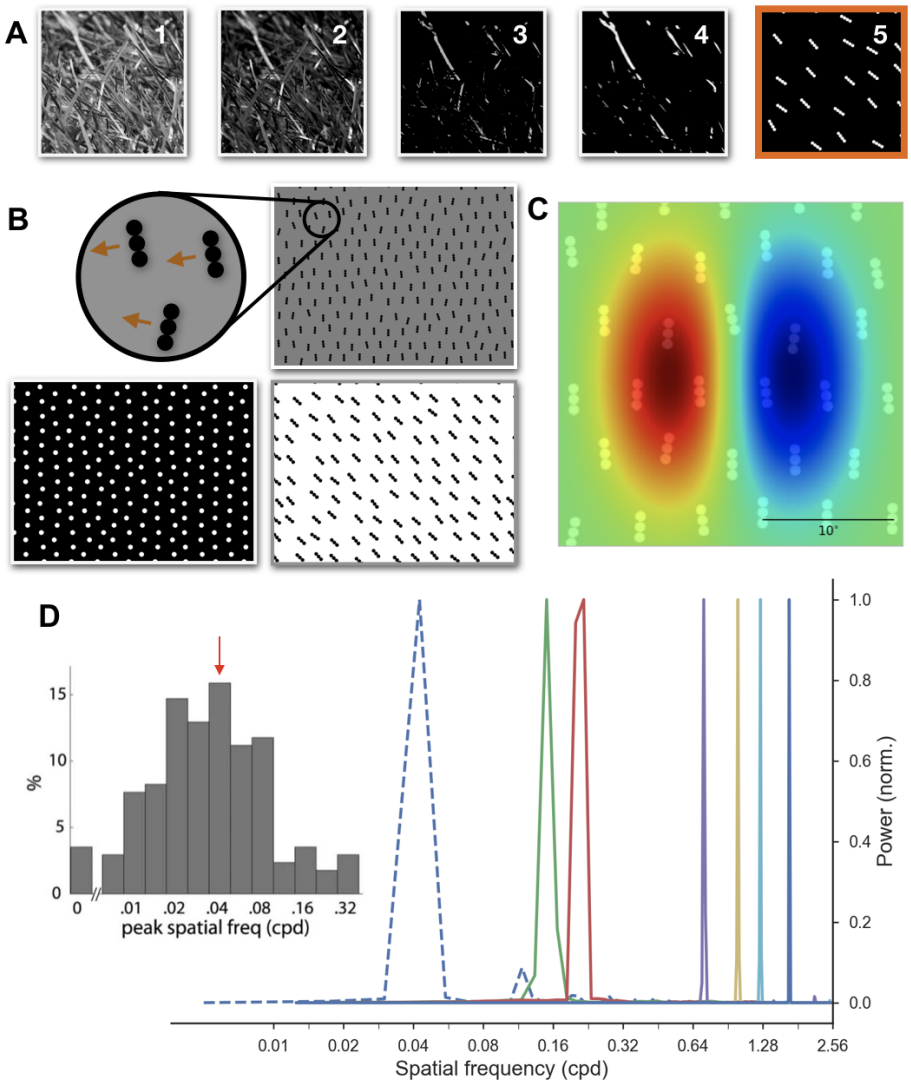

Figure 5.1: Introducing flow stimuli. ***A***, Ecological motivation for using flow stimuli: modifying an image of a grassy patch to progressively emphasize higher contrasts converges to a binary pattern of random, oriented line segments. ***B***, We abstract this to flow fields consisting of dotted segments of different lengths, emphasizing two geometries (oriented [3 or 4 aligned dots] or non-oriented [single dots]), two contrast polarities (positive or negative), and various sizes. ***C***, Flows are inconsistent with classical filtering views of V1. A Gabor receptive field at 0.04 cpd superimposed onto the 3-dot flow whose energy peaks at 0.24 cpd (top right example in ***B***), for comparison. ***D***, The 1-D discrete Fourier transforms (single-sided) of the flows utilized in our experiments (peaks at 0.15, 0.24, 0.7, 1.0, 1.25, and 1.6 cpd) have power well beyond 0.04 cpd (dashed curve), which is the spatial frequency previously reported as optimal for cells in mouse V1 (cf. inset, from [101]). To compare stimuli, each spectrum is normalized by the power at the peak frequency.

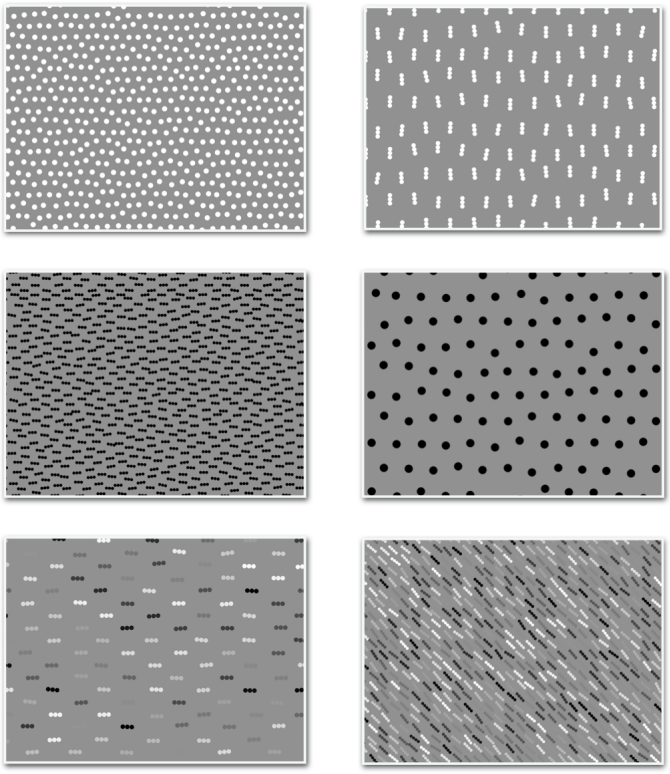

Figure 5.2: Examples of possible flow patterns. **Top**: positive flows with single- (left) and 3-dot (right) elements. **Middle**: two examples of negative flows; dot sizes determine the stimulus spatial frequency. **Bottom**: another possibility is to have flow patterns with non-uniform distribution of contrast, e.g., following natural scene statistics (not used in these experiments).

of patterns being shown at all times.

5.2.1 Materials and methods

(For a complete description of the experimental methods used, please refer to [44].

Extracellular recordings in awake mice

Alert mice were studied while on a spherical treadmill modified from the design described by [102] that permits free locomotion. Extracellular microelectrode recordings were performed as modified from [102]. The electrode is placed at an angle of 30°–45°to the cortical surface and inserted to a depth of 500–1000 μm below the cortical surface. For each animal, the electrode is inserted no more than twice.

Visual stimuli

Visual stimuli were presented with gamma-correction correction on a monitor placed 25 cm from the mouse, subtending 60°–75°of visual space. For current source density (CSD) analysis, we present a contrast-reversing square checkerboard (0.04 cpd, square-wave reversing at 1 Hz). For localization of receptive fields by spike-triggered averaging we presented spatiotemporal band-limited noise patterns, as in [101].

To characterize neural responses with single-unit recordings, we presented interleaved drifting square-wave grating stimuli and flow stimuli moving in 8 directions at a temporal frequency of 4 cycle/s over two sets of spatial frequencies: the first included 0.04, 0.15, and 0.24 cpd (100% contrast, trial duration 1.5 s); the second included 0.04, 0.7, 1.0, 1.25, and 1.6 cpd (50% contrast, trial duration 1.0 s). All stimuli variations were repeated 20–25 times according to a randomized sequence. Contrast and trial duration were maintained the same for all stimuli used in the same experiment.

We used flow stimuli with two different geometries. The first are non-oriented single-dot flows, and the other are oriented flow elements with either 3 or 4 dots. Both oriented and non-oriented variations had one version with positive contrast (white dots against a black or gray background), and another with negative contrast (black dots against a white or gray background). Dominant spatial frequency contents of 0.15 cpd, 0.24 cpd, 0.7 cpd, 1.0 cpd, 1.25 cpd, and 1.6 cpd were used, corresponding to the following dot diameters, in degrees of visual angle (dot spacings, in multiples of diameter): 2.1 (2), 1.5 (2), 1.4 (1), 1.0 (1), 0.75 (1), 0.5 (1) for single dots; for n dots, diameter is divided by $\sqrt{n}$ so as to preserve total area. For all flow stimuli, both parallel and jittered versions (0 and 0.1 rad std. dev. of flow field direction distribution, respectively) were used, with no detectable difference in the results; flow field tile side: 5 times dot diameter; initial positions on the lattice had std. dev. equal to 10% of dot spacing.

In the first cohort, the luminance of the screen displayed during the interstimulus period was set to the global average luminance of the stimulus of the upcoming trial—this allows one to control for responses due to the global change in the screen luminance only, and not to the actual moving stimuli. Such a strategy, however, does not completely eliminate the effect locally, given that there will still be a change in the luminance of the background as soon as the trial starts. Thus, in order to control for possible responses due to the change in background luminance, we in the second cohort a constant gray background was used both in the flow trials and in the interstimuli intervals. For a similar reason, the diameter of each dot in a multi-dotted flow element was chosen such that the total area occupied by the element was the same as that of the single-dotted version of the stimulus with same spatial frequency. In any event, the use of statistical tools to assess significant responses, the presence of a large proportion of cells being well-tuned to direction or orientation, and visual inspection of the actual individual PSTHs obtained are strong confirmation that our results cannot be trivially explained by screen luminance effects.

Single-neuron analysis

Single units are identified using MountainSort [27], which runs in approximately $2\times$ real time for fully automated spike sorting of data from our 128-site electrodes. Manual curation after a run on one hour of data takes an additional half hour, typically yielding 150 isolated single units.

Cortical layer

The cortical layer containing each isolated unit is determined using current source-density (CSD) analysis on data collected during presentations of contrast-reversing square checkerboard. Extracellular voltages sampled at 25 kHz are bandpass filtered between 1 and 300 Hz to yield local field potentials and then averaged across all 1 s positive-phase presentations of the checkerboard. CSD for each channel is computed as the second spatial derivative of the average LFP traces along the length of the silicon probe. The borders between layers 2/3–4, 4–5, and 5–6 are identified by spatiotemporal patterns of sinks and sources in the CSD plot ([**?**]; for example see Fig. 1C of [37]).

Data analysis

Isolated single units that stopped firing altogether after a certain time of the recording or that only started firing after some time were assumed to have moved away or toward the multielectrode array, respectively; therefore the trials with zero spikes at the beginning and end of recordings were discarded.

An orientation selectivity index (OSI) was defined as: $(R_{\mathrm{pref}}-R_{\mathrm{ortho}})/(R_{\mathrm{pref}}+R_{\mathrm{ortho}})$, where $R_{\mathrm{pref}} = (R_{\mathrm{peak\_dir}} + R_{\mathrm{peak\_dir}+\pi})$ is the response (average firing rate) for the preferred orientation and $R_{\mathrm{ortho}} = (R_{\mathrm{ortho\_dir}} + R_{\mathrm{ortho\_dir}+\pi})$ is the response for the orientation orthogonal to the preferred one. A direction selectivity index (DSI) was defined as: $(R_{\mathrm{pref}} - R_{\mathrm{null}})/(R_{\mathrm{pref}} + R_{\mathrm{null}})$, where R_{pref} is the response (average firing rate) for the preferred orientation and R_{null} is the response for the null orientation (π rad apart from the preferred one). A stimulus selectivity index (SSI) was defined for a pair of stimuli as $(R_{\mathrm{max}} - R_{\mathrm{min}})/(R_{\mathrm{max}})$, where R_{max} (R_{min}) is the average peak firing rate of the stimulus with higher (lower) firing rate in the pair.

PSTHs were plotted using a Gaussian interpolation kernel. Bin sizes were chosen following [**?**].

Power spectra of the stimuli were computed with a fast Fourier transform (FFT) algorithm. Due to the stochasticity of the flow stimuli, the spectra are averages over 40 different trials. Data were not treated with any window.

5.3 Results

5.3.1 Analysis of stimulus selectivity in V1

Cells in V1 have diverse preferred stimuli

We developed an ensemble of stimuli including drifting gratings, single dot flows (random dot kinematograms), and oriented flows where each element consists of 3 or 4 dots (see Methods). The stimuli had either positive contrast (bright dots) or negative contrast (dark dots). Activity is plotted as an array of peristimulus time histograms (PSTHs) and tuning curves for each unit, to facilitate a quick assessment of the different "dimensions" to a cell's response. Experiments were conducted in two cohorts, the first with grating stimuli at 0.04 cpd and both grating and flow stimuli at 0.15 and 0.24 cpd, and the second cohort with grating stimuli at 0.04 cpd and both grating and flow stimuli at 0.7, 1.0, 1.25, and 1.6 cpd. All stimuli in both cohorts had a fixed temporal frequency of 4 Hz.

We begin with example cells from the cohort 1. The first one (Fig. 5.3A) has the response profile one would expect for a *simple cell* in V1. It responds almost exclusively to low-frequency gratings; the PSTHs for high-frequency gratings and for flows (both one dot and three-dot elements) remain virtually at baseline. Its spike triggered average depicts a classical receptive field, consistent with the frequency response, and it is well tuned for orientation. But such cells were relatively rare in our experiments (discussed below). Another example (Fig. 5.3B) exhibits a weak response to gratings and a stronger response to flows. The STA, which would predict a strong response to low-frequency gratings, completely fails to predict this response profile. Finally, many cells are multi-dimensional (Fig. 5.3C): they respond well to several stimuli from the ensemble, including gratings and flows at multiple spatial frequencies. Note the diversity in the temporal response profile: a periodic (often interpreted as linear) response to gratings at low spatial frequency; a sustained (interpreted as nonlinear) response to gratings at higher frequencies; and a transient burst of activity to positive, oriented flows. It would be inappropriate to label this cell a classical feature detector. The STA again does not predict the response profile, and the PSTHs reveal different tuning widths, different first-spike latencies, as well as linear *vs.* nonlinear and transient *vs.* sustained responses.

5.3.2 Responses to optimal flows span a wide range of spatial frequencies

To quantify this diversity at the population level, we relaxed the notion of a unique preferred stimulus for a cell to allow for multiple possible preferences, according to the following definitions. While this leads to a crude classification of cell types, we stress that it is merely a set of labels for discussion; the underlying complexity remains in the PSTHs.

An individual stimulus is *significant* for a particular cell if the average firing rate for that stimulus is significantly higher than that for its preceding interstimulus interval (Mann-Whitney test). A cell prefers a *stimulus class* (e.g., flows or gratings) if at least one variation of that class (spatial frequency, geometry, or contrast polarity) is significant and has average peak firing rate significantly higher than the peak firing rates of all significant variations of the other class (Kruskal-Wallis rank-sum test, Conover-Iman post-hoc, Bonferroni correction, $p < 0.05$). When there is no preferred stimulus class but there are significant stimuli in both classes, we classify the cell as multi-class, or simply MULTI. Thus the preferred stimulus class, or *type* of a cell, is one of GRATING, FLOW, MULTI, NON-SELECTIVE. By this classification, the cell in Fig. 5.3A would be classified as a GRATING cell; Fig. 5.3B would be a FLOW cell; and Fig. 5.3C would be a MULTI cell.

Once each cell's type, or preferred stimulus class, has been determined, its *preferred spatial frequency* can be defined as the one with highest average firing rate among all significant variations of the preferred class (or classes, when cells are labeled MULTI).

We plot the proportion of preferred types at each preferred frequency in Fig. 5.3D; the two separate plots denote units from experimental cohort 1 (0.04–0.24 cpd, n = 357 cells, 3 animals) and cohort 2 (0.04–1.6 cpd, n = 256 cells, 3 animals), respectively. Note the predominance of GRATINGS among cells at the lowest frequency, replicating the inset in Fig. 5.1D, and the predominance of FLOW and MULTI types at the higher frequencies.

We now examine the distribution of preferred *types* in two different ways, either including or not including the responses to low-frequency gratings. This is necessary, since the performance measure is a simple spike statistic that is easily dominated by the gratings. First, when low-frequency gratings are included among the stimuli, by the above definitions 45% of the cells respond equally well; i.e. are in the BOTH type; 28% are FLOW cells; 26% prefer GRATINGS; and 29% of the cells were not significantly responsive to any of the stimuli displayed (see Fig. 5.3E, blue). When low-frequency gratings are not included, so that the comparison is among flows and gratings at the same spatial frequencies, responses favoring FLOW (50%) and BOTH (43%) predominate over those to GRATINGS (7%) (see Fig. 5.3E, red). The difference between these two plots comes from a more detailed analysis: the cells responding strongly to 0.04 cpd can be divided into roughly two subgroups: one that has no significant response other than to low-frequency gratings and another that also responds well to flows (or, in fewer cases, to both flows and high-frequency gratings). These plots include all cells. A similar distinction obtains when only cells well-tuned to orientation (OSI > 0.5) or direction (DSI > 0.5) are considered (Fig. 5.3F).

In summary of these first data, among cells with significant preference for flows or both flows and gratings, responses were distributed across all spatial frequencies explored. For "classical" cells (those that significantly preferred gratings to flows) there is a clear preference for 0.04 cpd with a distribution in accordance with [101] (see Fig. 5.1D). Curiously, some cells that are well tuned to low-frequency gratings are also well tuned to flows with higher spatial frequency, albeit usually with lower firing rates. Nevertheless, many

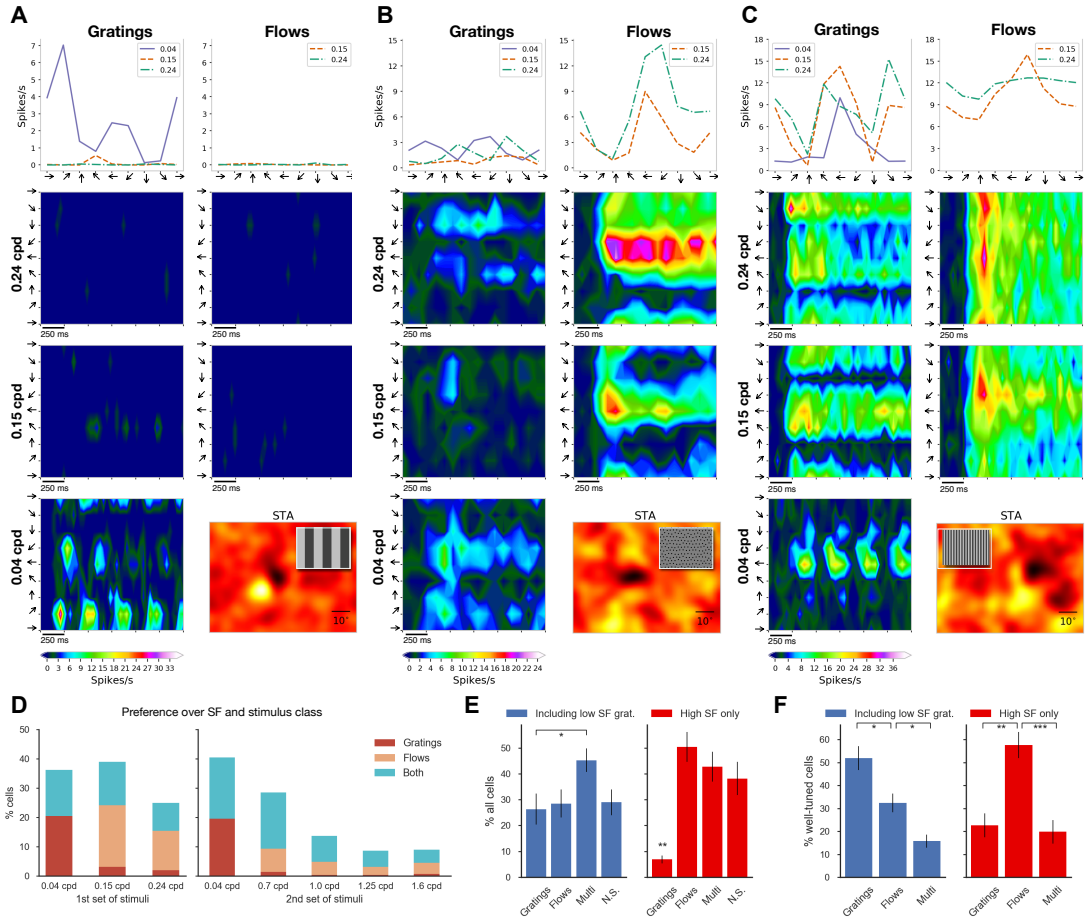

Figure 5.3: Variety of responses in V1. ***A–C***, Tuning curves and PSTHs of three example cells in response to drifting gratings and flows at 0.04, 0.15, and 0.24 cpd in 8 equally-spaced directions of motion. Time axis in histograms encompasses an entire period of stimulus presentation (1.5 s). Insets in STAs show, at the same scale, stimuli that produced the most significant responses. ***A***, Cell responding to low-frequency gratings only. Bin size 34 ms. ***B***, Cell responding preferably to single-dot flows with negative contrast. Bin size 83 ms. ***C***, Cell responding strongly to both oriented (3 dots), positive flows and gratings (at both high and low spatial frequencies). Bin size 46 ms. ***D***, Distribution of optimal spatial frequency in terms of proportion of cells significantly responding to at least one of the stimuli. In the group of experiments using the first set of stimuli (left panel, 0.04–0.24 cpd, n = 357 cells, 3 animals), the majority of cells fired more strongly for stimuli at 0.15 cpd, followed closely by 0.04 cpd. For the second set of stimuli (right panel, 0.04–1.6 cpd, n = 256 cells, 3 animals) there was an overwhelming preference for 0.04 cpd, although more than half the cells had optimal spatial frequency in the range 0.7–1.6 cpd. (Continues.)

Figure 5.3: *(Continued.)* ***E***, Distribution of preferred stimulus among all cells. When low-frequency gratings (0.04 cpd) are included among the stimuli (left panel), the majority of cells respond equally well to both classes ("Multi"), followed by only flows and only gratings; 29% of the cells were not significantly responsive ("N.S.") to any of the stimuli displayed (n=1026 cells; 10 experiments, 6 animals). When we do not include low-frequency gratings, thereby limiting the comparison to flows and gratings with similar spatial frequencies only, there is a significant preference for flows only and for both over gratings only. Comparison of the left and right panels reveals that approximately 20% of cells preferred low-frequency gratings. When we recompute stimulus preference considering only stimuli with comparable spatial frequencies, most cells that preferred low-frequency gratings now either prefer none of the high-frequency stimuli, or significantly prefer flows over high-frequency gratings, given that the fraction that prefers both remains essentially constant in the two scenarios. Error bars represent s.e.m. ***F***, Distribution of preferred stimulus among well-tuned cells (i.e., those with OSI ¿ 0.5 or DSI ¿ 0.5), n=295 cells (left), 241 cells (right); 8 experiments, 4 animals. Here, notice that most of the cells responding to orientation and/or direction will fire more strongly to low-frequency gratings; the right panel reveals, however, that the fraction of cells well-tuned to flows is just as large. And, similarly to ***E***, many of the well-tuned cells preferring 0.04 cpd gratings prefer flows to gratings of comparable spatial frequency. Error bars represent s.e.m.

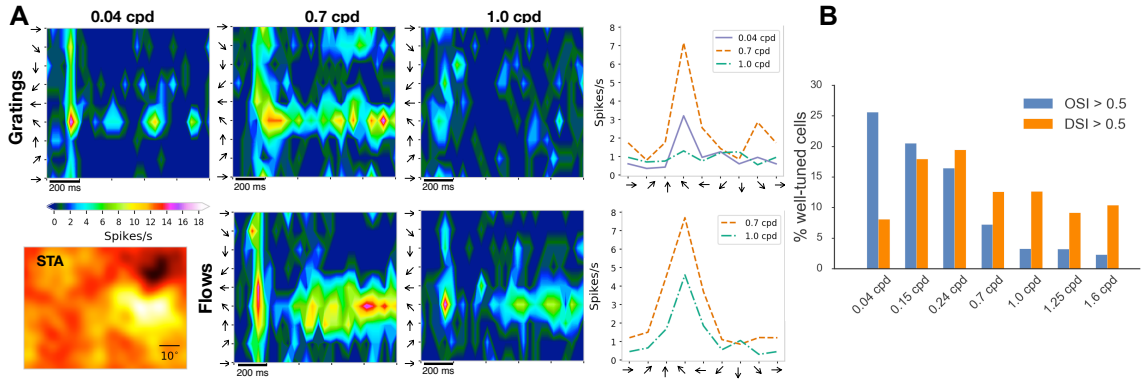

Figure 5.4: Cells remain highly selective at higher spatial frequencies. ***A***, Example of cell exhibiting a stronger response to oriented, negative flows at 0.7 and 1.0 cpd when compared to gratings at various spatial frequencies. Bin size 47 ms. ***B***, Overall proportion of well-tuned cells among cells significantly responsive to each spatial frequency (Mann-Whitney test, $p < 0.05$), irrespective of stimulus class. Sample sizes: 0.04 cpd (n = 508), 10 experiments, 6 animals; 0.15 cpd (n = 385), 0.24 cpd (n = 365): 5 experiments, 3 animals; 0.7 cpd (n = 214), 1.0 cpd (n = 214), 1.25 cpd (n = 186), 1.6 cpd (n = 173): 5 experiments, 3 animals.

of these cells have higher firing rates for flows compared to gratings of similar spatial frequency, showing that there is some aspect of the flow stimulus that strongly excites these cells despite the fact that the flow elements would excite the filter predicted by these cells'

STAs. Supplemental Materials show plots of responses to the entire stimulus ensemble for these and other cells.

Cells remain well-tuned at high spatial frequencies

Since higher firing rates do not necessarily imply high orientation- or direction-selectivity, and since a cell might retain its selectivity at several spatial frequencies (SFs), we investigated the fraction of well-tuned cells (OSI ¿ 0.5 or DSI ¿ 0.5) across spatial frequencies regardless of preferred stimulus (Fig. 5.4B). This is an estimate of the probability of a cell significantly responsive to a certain SF being well-tuned.

There are many cells well-tuned to direction and/or orientation at all SFs. Cells with high orientation selectivity tend to prefer stimuli in the 0.04–0.24 cpd range. The direction-selective cells seem to be more uniformly distributed across SFs, with a preference for intermediate SFs (0.15–0.24 cpd).

Higher stimulus selectivity in superficial and deep layers

To further characterize how the response profile of MULTI cells is distributed across stimulus variations, we extend the concept of selectivity indices such as OSI and DSI (e.g., [101]) to compare pairs of stimulus classes. A stimulus selectivity index (SSI) is thus defined for a pair of classes (e.g., flows *vs.* gratings, or 1-dot flows *vs.* 3-dot flows) as $(R_{\max} - R_{\min})/(R_{\max})$, where $R_{\max}$ (resp., $R_{\min}$) is the average peak firing rate (FR) of the stimulus with higher (resp., lower) FR in the pair. Essentially, it measures the difference in FR between two stimuli, relative to the one with highest FR. E.g., an SSI of 0.2 means the FR for the least preferred stimulus is 20% lower than that for the preferred one. When comparing stimulus classes for which there are possibly several stimulus variations in each class, we take the variation that elicited the highest response in each one. Note that the SSI for a cell population assesses how well those cells' responses can be used to differentiate between two stimuli, regardless of which one is the preferred one.

Cells responsive to both flows and high-frequency gratings were found in all cortical layers. Cells in layer 2/3 had significantly higher values of SSI than all other layers for differentiating flows from gratings ($p < 10^{-3}$, $p = 10^{-6}$, and $p = 10^{-4}$ for layers 4, 5, and 6, respectively), while cells in layer 5 had significantly lower SSI than layers 2/3 ($p < 10^{-4}$) and 6 ($p < 0.05$) when differentiating between flows with opposite contrast polarities, and lower than layer 2/3 ($p < 0.005$) when differentiating oriented from non-oriented flows (Fig. 5.5). The same trends were found when only broad-spiking cells (putative excitatory, see [101]) were considered. Thus, speculatively, cells in the superficial layers could have higher selectivity, while cells in layer 5 could be more invariant to geometry, length, and contrast. This may be related to [101], in which it was reported that layer 5 cells were significantly less linear than cells in other layers.

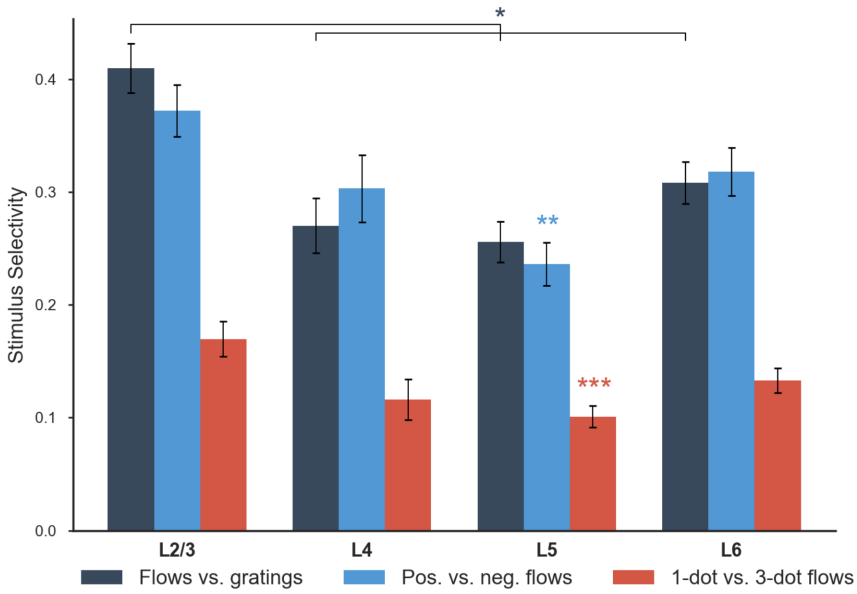

Figure 5.5: Cells in different layers have distinct selectivity toward different stimulus classes, as measured by a stimulus selectivity index (SSI), which indicates the relative preference for either of two stimulus classes, in terms of firing rate. E.g., an SSI of 0.41 in layer 2/3 for gratings *vs.* flows means that cells in that layer have an average 41% difference in FR between their peak responses to flows and to gratings (regardless of which one is higher). Cells in layer 2/3 had significantly higher SSI than all other layers when it came to differentiating between flows and gratings (*, $p < 10^{-3}$, $p = 10^{-6}$, and $p = 10^{-4}$ for layers 4, 5, and 6, respectively). On the other hand, cells in layer 5 had significantly lower SSI than layers 2/3 (**, $p < 10^{-4}$) and 6 ($p < 0.05$) when differentiating between flows with opposite contrast polarities, and lower than layer 2/3 (***, $p < 0.005$) when differentiating between oriented and non-oriented flows (Fig. 5.5). Error bars represent s.e.m.

Preference between different variations of flow stimuli goes beyond differences in spatial frequency

Among cells that responded significantly to flows, we also compared the average proportion of cells that significantly preferred oriented (3 dots) *vs.* non-oriented flow patterns (single dots) (Fig. 5.6A). Analysis of the entire population across different experiments does not reveal any particular preference, with the vast majority responding to both geometries. However, if analysis is restricted to those cells well-tuned to direction and/or orientation, the preference for a specific flow geometry—be it oriented or non-oriented—increases markedly. In particular, there is an overall preference for the oriented patterns.

Fig. 5.6B shows that only a minority of the cells responding to flows prefer negative contrast (15%, on average). The vast majority prefer either positive contrast or respond significantly to both contrast polarities. This difference in preference disappears among cells that are well-tuned to direction and/or orientation.

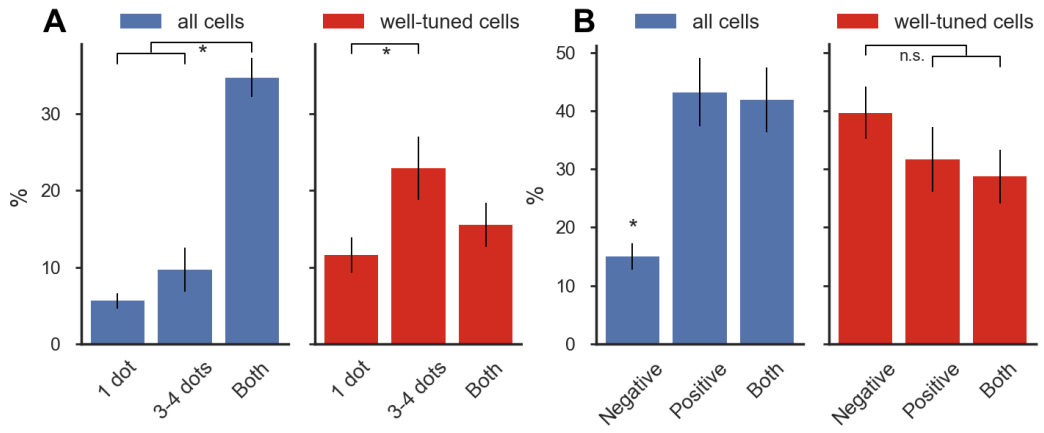

Figure 5.6: Preference over flow stimuli variations. Percentages refer to the population of cells that had significant response to at least one flow variation. All cells: n = 667, 10 experiments, 6 animals; well-tuned cells: n = 187, 8 experiments, 4 animals (applies to panels ***A*** and ***B***). Error bars represent s.e.m. (*, $p < 0.001$). ***A***, Flow geometry preference. Among all cells responding significantly to flows, most showed no significant preference for either type. Among well-tuned cells, there is a significant preference for oriented flows over non-oriented flows. ***B***, Flow contrast polarity preference. Among all cells significantly responding to flows, positive polarity was preferred. The population of well-tuned cells showed no overall preference for contrast polarity.

Chapter 6

Network architectures for the visual system

- **Goal**: Learn manifolds for the mouse retina and primary visual cortex using stimulus ensemble from Chapter 5; compare results with manifolds obtained for deep convolutions networks (DCNs) using "stimuli" from Imagenet [40].
- **Method**: apply algorithm from Part I. Apply conductance measure to assess degree of 'clustering.'
- **Results**:
  1. **Retina**: organization of retinal ganglion cell types confirms techniques; reveals a clustered, or 'discontinuous' manifold;
  2. **Cortex**: organization of cortical cell types confirms techniques; reveals a more continuous manifold, indicating feedback circuitry;
  3. **DCNs**: although 'deep nets' are widely used as models of visual cortex and as cognitive theories, our manifolds suggest they are more like the retinas than like the cortex.

6.1 Real-world examples

Now we apply the algorithm developed in the first few chapters to biological data recorded from mouse retina and primary visual cortex.

Representing neural responses as PSTHs

Neural responses in which (at least 10) individual trials are combined into a peristimulus time histogram (PSTH) to form, a 2-D array of direction/orientation *vs.* time (cf. examples in Chapter 5). Six stimulus classes were considered: low-frequency gratings, high-frequency gratings, 1- and 3-dot negative flows, and 1- and 3-dot positive flows, meaning

each data point is represented by 6 PSTHs. Because the stimuli in each class were presented with at least two different spatial frequencies (except for low-freq. gratings, which was always 0.04 cpd), we selected the PSTH with highest mean FR at the optimal direction.

Furthermore, even though our PSTHs are 2-D, we choose not to make stimulus direction into a fourth tensor mode—instead, we concatenating the PSTHs for different directions intro a single PSTH vector. This may seem counterintuitive since, in principle, a tensor is used precisely because of its summarization power (i.e. reconstructing higher dimensions using fewer parameters). Our reason for keeping a 3-mode tensor is based on technical aspects of tensor factorization.

Splitting the PSTH into multiple modes forces the total PSTH reconstructed by each component to be decomposable as an outer product. This restricts each component to simple and symmetrical patterns, e.g. like circles or horizontal/vertical rectangles in 2-D. (A diagonal line cannot be represented by a simple component, i.e., as the the outer product of two 1-D vectors: it needs to be split into several components, each one representing a small circle akin to a "pixel", as shown in [121]).

By vectorizing the multi-dimensional PSTH into a single mode, factors are only split when their parts reconstruct responses for distinct groups of neurons (not necessarily disjunct). When there are complex non-separable patterns in the PSTHs, this has three advantages: (i) it can reduce the number of components required—this gives fewer factors with greater variance, instead of many factors that fit most cells, giving low variance; (ii) factors are more interpretable; (iii) this prevents correlations/dependencies between different neural factors.

To see this, consider a 2-D PSTH pattern that cannot be reasonably approximated as the outer product two vectors alone. If we use a 4-way tensor, this will require several separate components to reconstruct the same group of neurons. Since the factor matrix will be used for computing distances between neurons, this would artificially make neurons that respond in that way to be more distant to the remaining neurons than others which have a simpler PSTH pattern. (Of course, responses should be considered equal regardless of whether they are horizontal or diagonal.)

Choosing the tensor modes

Note that, in contrast with the tensor used in [145], we use a "stimulus" mode instead of one containing all individual trials. Using trials is a requirement when one expects to find evidence of long-range temporal adaptation, e.g. learning a task, or wishes to find evidence for temporary neuromodulatory states. Such approach uses a smoothed spike train for each trial; in our case, averaging across multiple trials helps reduce the variability typical of spike recordings, especially those from behaving animals.

Direction/orientation preferences

Since many cells in visual cortex have some direction/orientation preference, but the distribution of such preference is unlikely to be sufficiently uniform due to the sparseness of our neuronal sampling, we adopted the following scheme: each neuron's tuning curve was fit to a double Gaussian (cf. [95]), and those that had a root-mean-squared error $< 15\%$ were shift-rotated to have their highest peak at a common orientation of $\pi/2$. This allows for cells that show similar direction/orientation selectivity to be all similar to each other, regardless of their specific preferred direction/orientation. Later in this chapter, an example will demonstrate what happens when such rotation is undone.

6.2 Results for the retina

Extracellular microelectrode recordings were performed on the mouse retina in the Field lab at Duke University. Spikes were sorted using custom software, resulting in a total of $N = 258$ inferred neurons. The present results correspond to a single experiment using the same stimulus ensemble of gratings and flows as described in Chapter 5. Receptive fields were estimated using spike-triggered average in response of a checkerboard stimulus, and the RGC types for a subset of the cells were inferred using the methodology from [153, 150, 108].

Although retinal ganglion cells are not directly connected (at least not with synapses), they do share common inputs from bipolar and amacrine cells, justifying the notion that we can use the manifold abstraction to infer not only the functional connectivity between RGCs [46], but also properties of some of the upstream circuity.

6.2.1 Factorization

Following the procedure laid out in Chapter 3, we obtain an estimate of the rank of the neural matrix (Fig. 6.1), which we then use to compute an appropriate number of factors (Fig. 6.2). Details are provided in the captions for these figures.

6.2.2 Neural manifold coordinates

Inspection of the retinal embedding in Fig. 6.3 reveals that what we have in this case is more a collection of manifolds rather than a single manifold. Although technically its corresponding data graph has a single connected component, weights connecting the different clusters are very small. This highly clustered organization is reminiscent of our embedding for the LN model in which neurons of the same RF type ended up clustered together. RGCs are classically organized into separate types, and recent efforts have attempted to

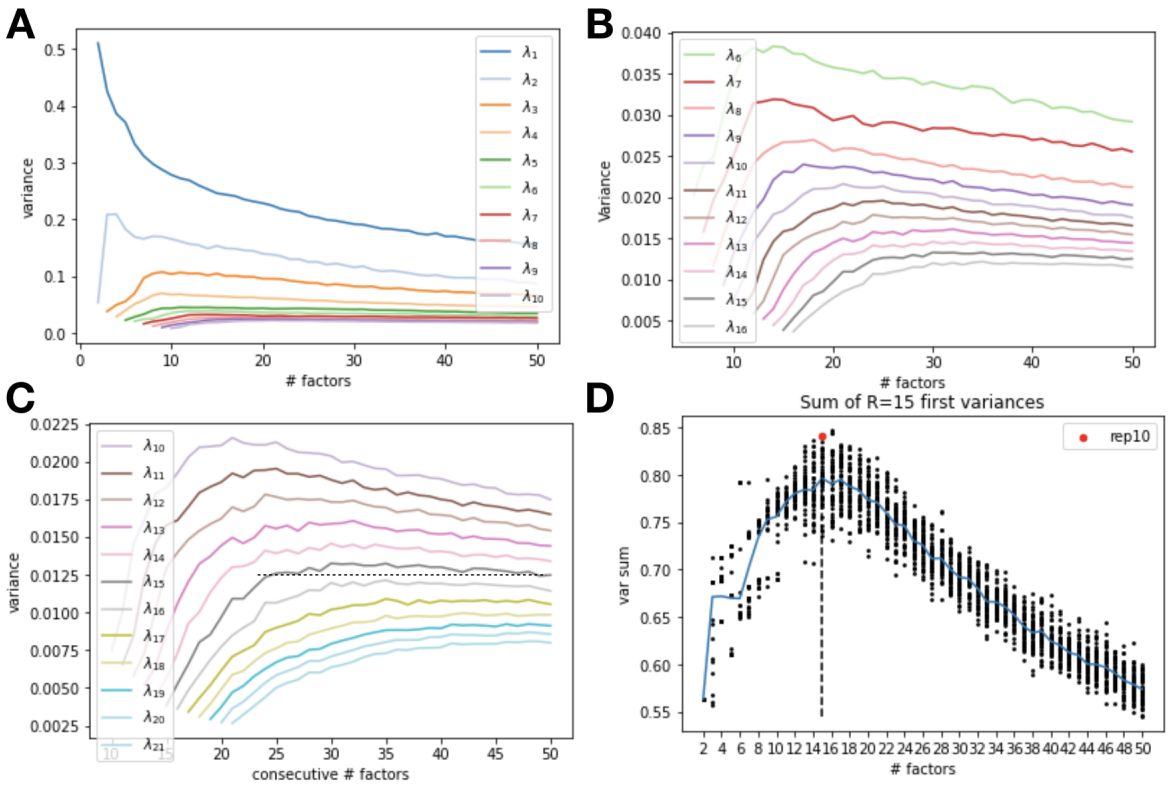

Figure 6.1: Determining the rank of the neural matrix for the retina experiment. **A**: Evolution of variance curves for the first few PCs. **B**: Focusing on higher-numbered eigenvalues for better resolution. It seems that the curve λ_{16} never quite decreases, indicating that its corresponding PC is not significant. **C**: zooming in even further around λ_{16}, we check that on the other hand λ_{15} is clearly not monotonic (the dashed line helps visualize this). Finally, in **D** the sum-of-variances curve for the first 15 PCs has a max for $F = 15$, which tells us the number of factors needed.

give a general hierarchy of these cells using other machine learning approaches; a total of around 40 types are suggested.

A useful way to try to validate our results is by checking whether known RGC types are organized together in the manifold. In fact, this confirms our expectations that by looking to organize cells in terms of their responses we may be able infer their physiological properties.

A subset of the cells in the data set were previously labeled by the experimenters using complementary information, and these end up neatly positioned together in the manifold (see Fig. 6.4. Further confirmation is given by the mosaic organization of cells in each of these clusters, i.e. completely tiling the visual field, which is a hallmark of cell types in the retina [41, 46].

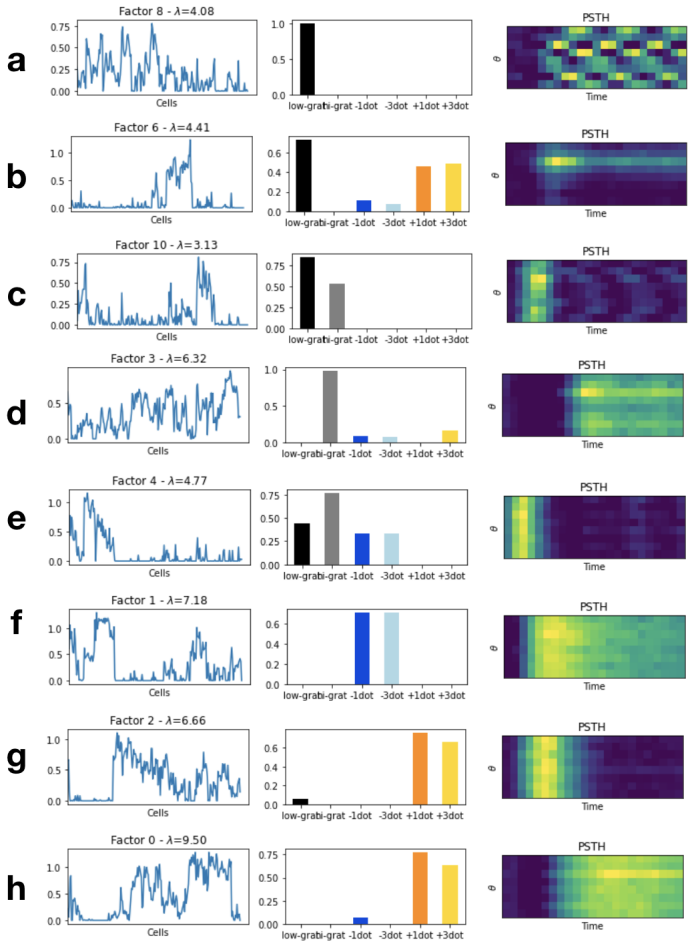

Figure 6.2: Example of factors returned by NTF for the retina data set.

6.2.3 Local neighborhoods

6.3 Results for the cortex

Here, we apply the algorithm to the same data set used in Chapter 5 (first cohort of experiments). These experiments were performed in the Stryker lab at the University of California–San Francisco (UCSF). Data from four different recordings are combined, resulting in a total of $N = 357$ sorted neurons. Laminar information and putative cell types were determined cf. [44].

6.3.1 Factorization

After estimating the rank of the neural matrix for V1 (Fig. 6.1), we can compute an appropriate number of components (examples in Fig. 6.6). A glance at the neural factors

reveals they are neither sparse nor simple. As one might expect from circuit interactions, it is difficult to associate individual neurons with individual stimuli. Details are provided in the captions for these figures.

6.3.2 Neural manifold

In Fig. 6.7 the resulting manifold for the cortical data set is shown. Heat maps are superimposed to demonstrate how the neurons are positioned according to main organization "coordinates". Preference for a given stimulus is computed as its relative FR within the ensemble (between 0 and 1). Examples of local neighborhoods across the manifold are given in Figs. 6.8 , 6.10.

6.4 Deep convolutional networks

Next, we apply the algorithm to deep convolutional networks (DCNs). We used two popular pretrained models, Alexnet [81] and VGG16 [125], both pretrained on the imagenet data set [40] of natural images. As our 'stimuli', we use a subset of imagenet's validation set; each trial is a random shift of the input image across position. The average value across trials converts each unit's activity into a 'firing rate'. In order to make the trials correspond more exactly to those in our biological experiments, in which the stimuli drift across the screen, we consider a trial to be is a random shift of the input image across position. With this, a neuron is likely to fire if there is a good match between its RF and some portion of the input image, irrespective of its precise location. Different trials are generated with different shifts (analogously to the different randomized flow instances used in different trials of our biological experiments). The average value across trials converts each unit's activity into a 'firing rate'. We sample neurons from a subset of the feature maps in a given layer. Results are given in Figs. 6.11 , 6.12.

The embeddings from DCNs, as one could expect, are organized mainly by feature map, i.e., neurons sharing the same weights end up tightly clustered together. Still, the organization of the feature maps within an embedding is a product of their preference for certain images over others. This means the embedding could be a useful source of information when training and designing deep nets, e.g. for identifying redundant filters (candidates for dropout) or those that are more selective towards a specific image category.

6.5 Quantitative comparison

We now apply the flow ratio method from section 4.5 as a way to compare the different manifolds in terms of whether they are more "clustered" or "continuous" (Fig. 6.13). The low value of ϕ_G for deep nets means they are (globally) poorly connected, i.e. most

points cannot reach all points via paths of high conductance. The cortical manifold, on the other hand, is richly connected, and its value of ϕ_G is similar to that of the noisy plane shown in Fig. 4.8. The conductance for the retinal embedding lies somewhat in between. Sparsifying the graphs or not doing so gave very similar results.

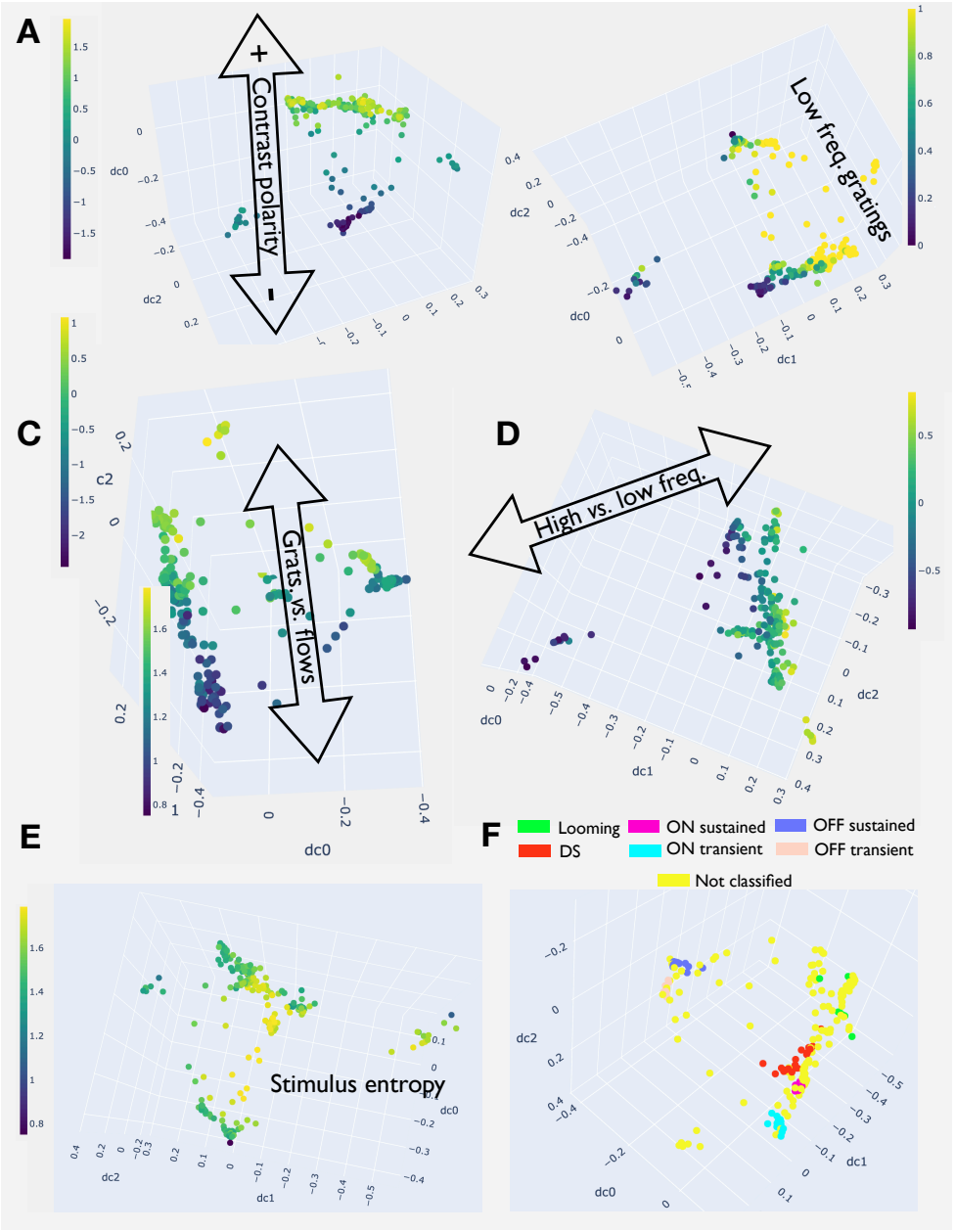

Figure 6.3: Main coordinates organizing the retinal "manifold". **A**: Heat map represents preference in terms of relative firing rate (FR) magnitudes for positive flows minus preference for negative flows. A positive (negative) value indicates higher activity for positive (negative) flows. **B**: cells responding to low-frequency gratings are concentrated on one side of the embedding. **C**: as in **A**, but now comparing all gratings *vs.* all flows. **D**: Preference for high- *vs.* low-frequency gratings is also well organized. **E**: Stimulus entropy is computed as the entropy of the vector containing the relative FR for all stimulus classes: low (high) entropy means a cell responds to few (most) stimuli. **F**: Labeled RGC types reveals that the manifold correctly organizes them into clusters. Even though cells with opposite contrast preferences are on opposite sides (ON *vs.* OFF), their responses' temporal dynamics can be roughly aligned with each other (transient *vs.* sustained).

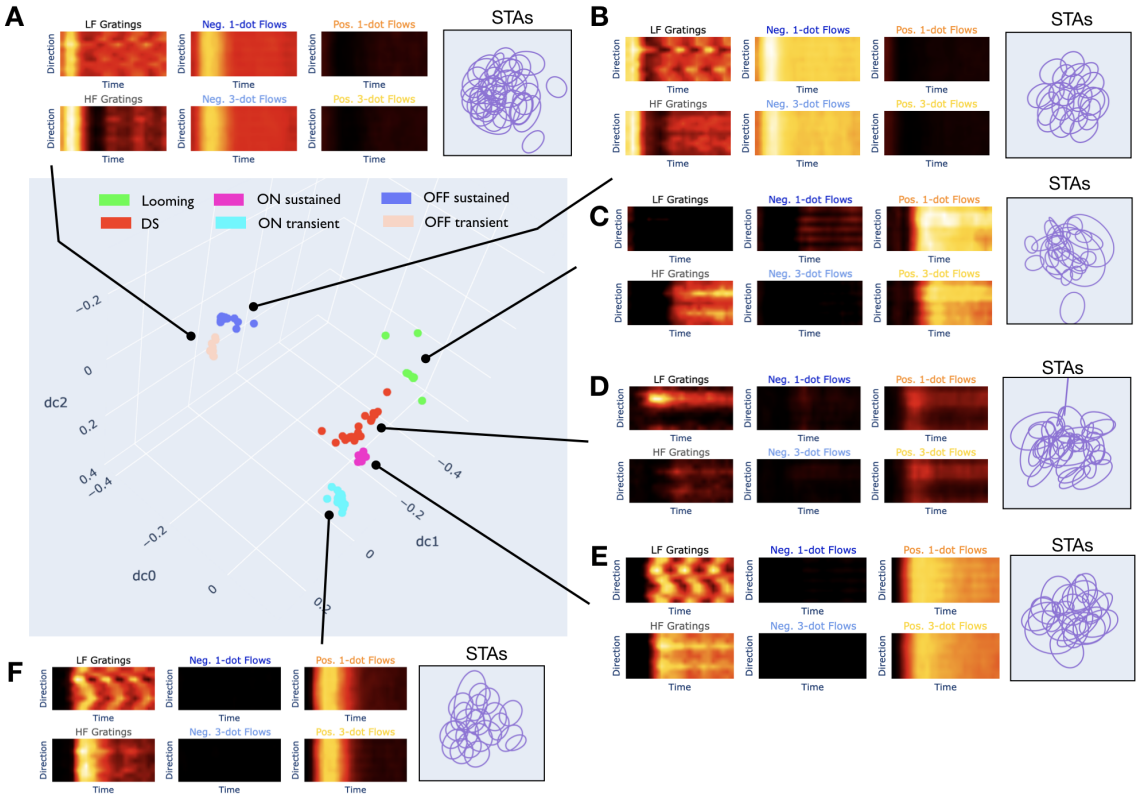

Figure 6.4: Cluster centroids of labeled RGC types in the retina. **A**–**F** represent centroids of RGC clusters centered at the indicated positions in the central plot (the same as in Fig. 6.3-F with the unlabeled cells not shown, for clarity), with their average PSTH for each stimulus class (left) as well as position and shape of their RFs, as estimated by spike-triggered averages (STAs). PSTH centroids correspond to the expected responses given the cell types present. In particular, note the mosaic organization of cells' RFs in each of these clusters, i.e. completely tiling the visual field. Note how the DS cells in **D** seem to prefer positive flows, as predicted by factor **b** in Fig. 6.2. Also interesting is the marked oriented response to high-freq. gratings for the looming cells in **C**.

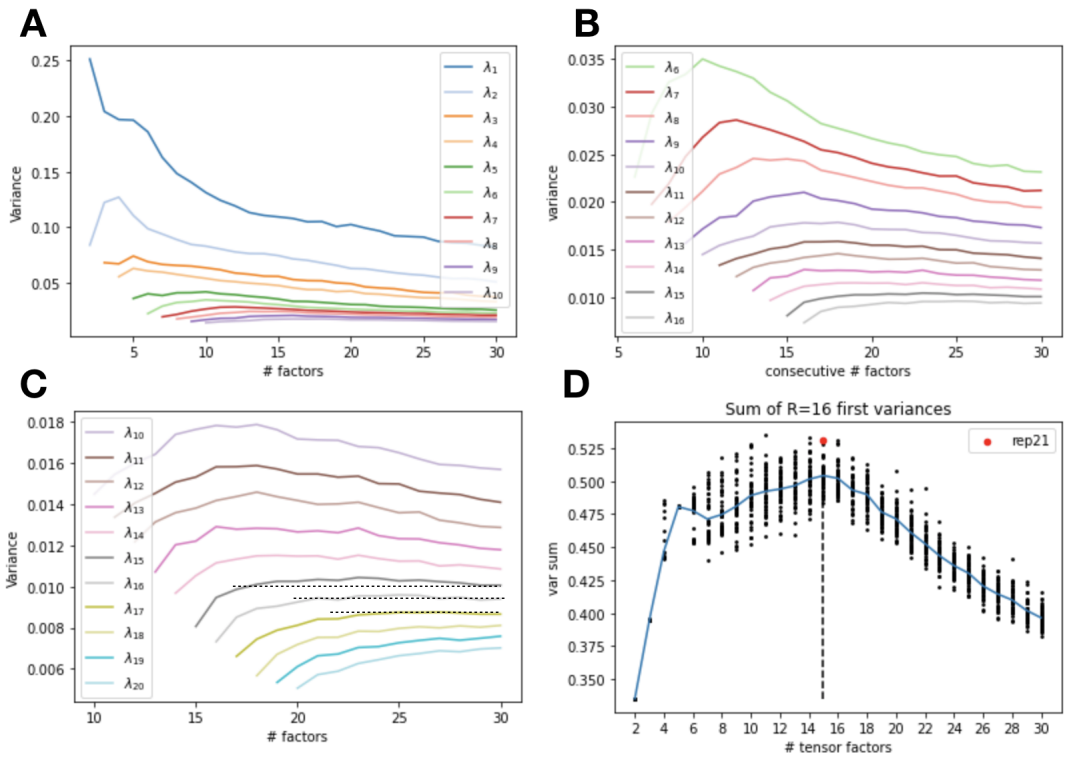

Figure 6.5: Determining the rank of the neural matrix for the cortical experiment. **A**, **B**: Evolution of variance curves for the first 16 PCs. **C**: The dashed curves help in deciding which curves have a peak and which don't. Here, λ_{15} clearly does, and λ_{17} does not, but it is hard to tell for λ_{16}. **D**: The variance sum curve can help with the choice of R. Choosing $R = 16$ here would actually result in a peak at F=15, indicating that the choice was inadequate (since when $F = 15$ the neural matrix has 15 columns, so $\mathbb{R} \leq 15$).

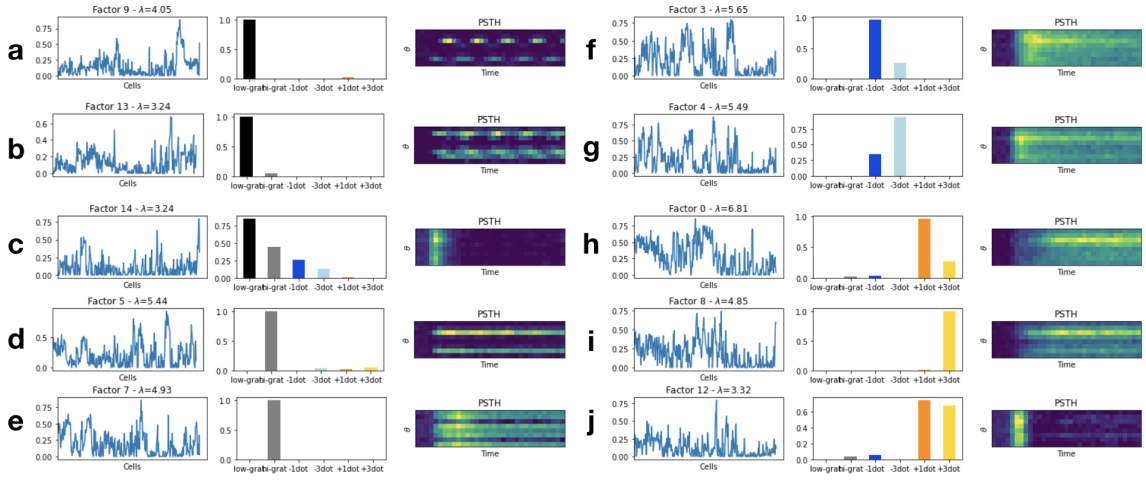

Figure 6.6: Examples of factors returned by NTF applied to our cortical data set. Neurons here clearly have a distributed role: they can participate in many factors, i.e. respond in different ways to different stimuli. Factor **a** describes linear-like (phase-selective) responses to low-freq. gratings. The PSTH pattern in **b** resembles a sustained (complex) response to low-freq gratings. Factor **c** represents a transient response for both gratings and negative flows. In **d**, a highly orientation-selective response to high-freq. gratings. In **e**, a good example of the additive nature of the non-negative parts-based representation: although this exact PSTH pattern is never present by itself, in combination with **d** it reconstructs responses with larger tuning width. Both pos. and neg. flows seem to differentiate between 1- and 3-dot flows (compare **f** with **g** and **h** with **i**). Notice how, for both polarities, the factor with the higher 3-dot contribution is more orientation selective, while the 1-dot factor is directional selective. Factor **i** shows that positive flows have a transient response part as well.

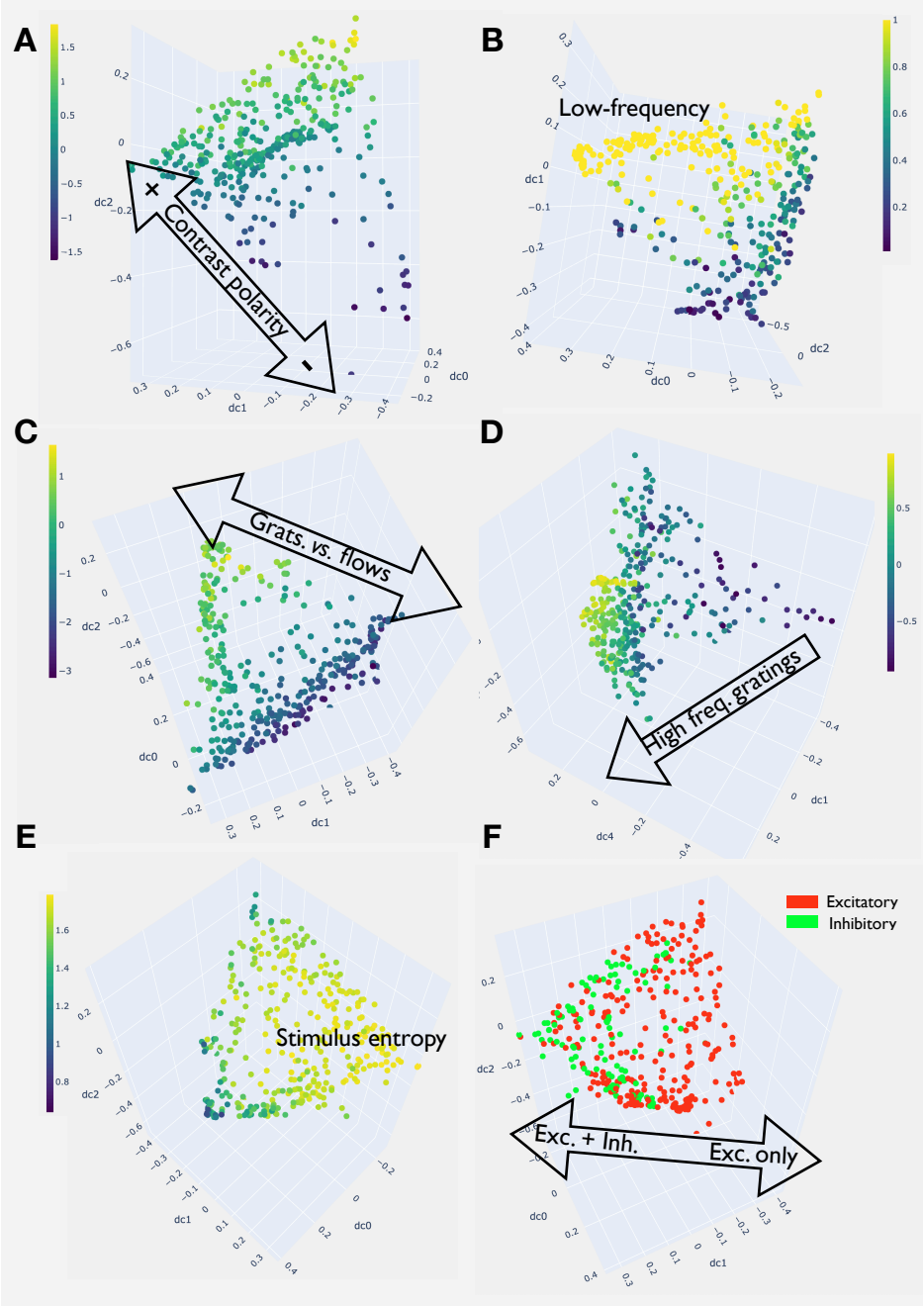

Figure 6.7: Main coordinates organizing the cortical manifold (similarly as in Fig. 6.3). **A**: Flow contrast polarity is one of the main coordinates. It is also clear that a preference for positive flows is more common. **B**: Preference for low-frequency gratings in terms of relative FR. **C**: Preference for gratings (low + high frequency) minus preference for flows (positive + negative). **D**: Looking at higher diffusion coordinates permits one to organize the neurons by their preference toward high frequency gratings as well. **E**: Cells are also organized in terms of "stimulus entropy", i.e. how evenly distributed is their preference over the ensemble. Preference for fewer stimuli are "pushed" to the periphery. **F**: Interestingly, cell types (inhibitory *vs.* excitatory) are also somewhat organized, with a portion of the manifold being exclusively excitatory, even though this information was not given to the algorithm.

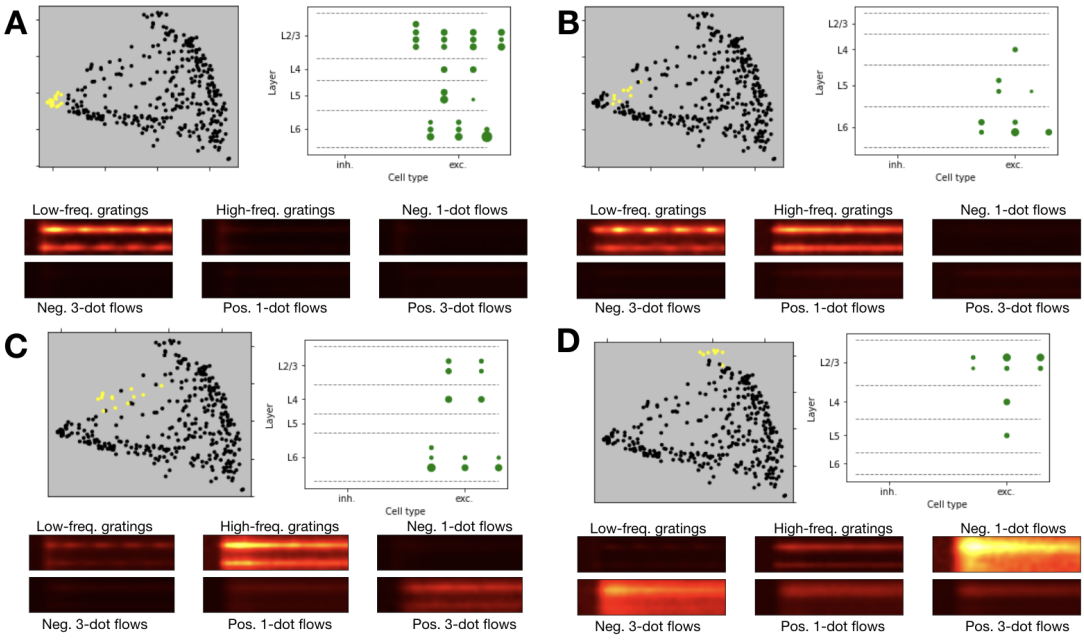

Figure 6.8: Local neighborhoods have distinct cell types and laminar profiles. Top left plot indicates position of neighborhood on manifold; the top right diagram shows the layer and cell-type distribution of individual neurons (circle areas are proportional to average firing rate). PSTHs at the bottom are the centroids of the points selected (average PSTH for each stimulus class). Starting at the position in **A**, we see exclusive response to low-freq gratings in a linear-like fashion (phase-oriented). These are all excitatory and mostly occupy layers 2/3 and 6. Moving upwards, in (**B**) we begin to see cells that respond to high-frequencies as well (layer 2/3 contribution disappears). In **C**, the response to low-freq. gratings is weaker, and there are oriented responses to both high-freq. gratings and positive 3-dot flows (but not to 1-dot flows). **D**: a group of cells mostly in layer 2/3 with a strong preference for negative flows, in particular 1-dots.

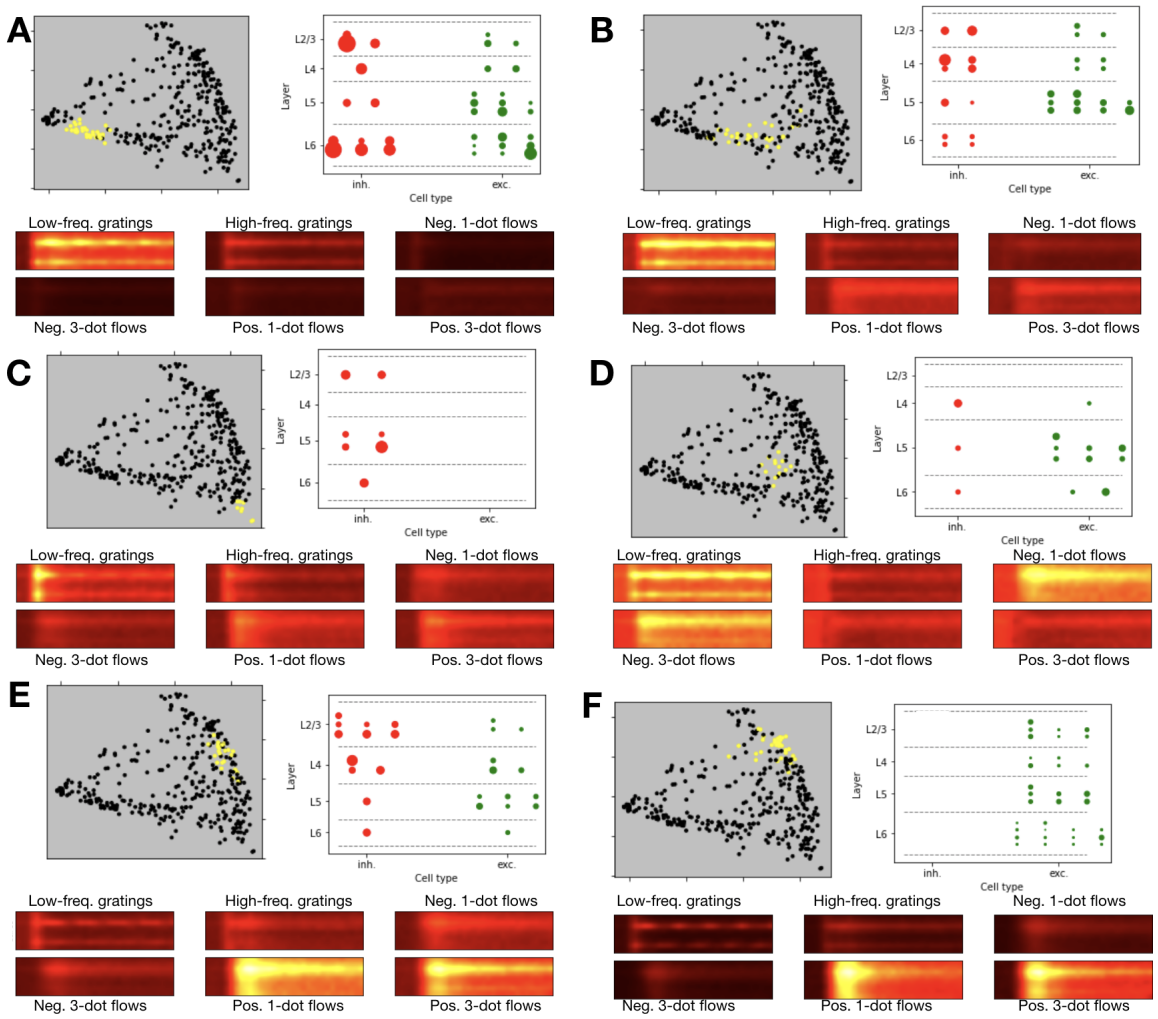

Figure 6.9: Similarly as in Fig. 6.8, but tracing a different path along the manifold. If we now move to the right, in**A** we see that low-freq. grating responses are more complex-like; inhibitory cells are present now, and in all layers; in **B**, responses to positive flows begin to appear. Moving further to the right **C** we see a neighborhood with marked transient responses to low-freq. gratings, all inhibitory; moving upwards (**D**) we see strong responses to negative flows as well, and inhibition is reduced. In **E** and **F**, two groups responding strongly to positive flows, the first containing both inhibitory and excitatory neurons and the other completely excitatory. Although both have a clear preference for positive flows, the former responds to both flow polarities, while the latter is more selective.

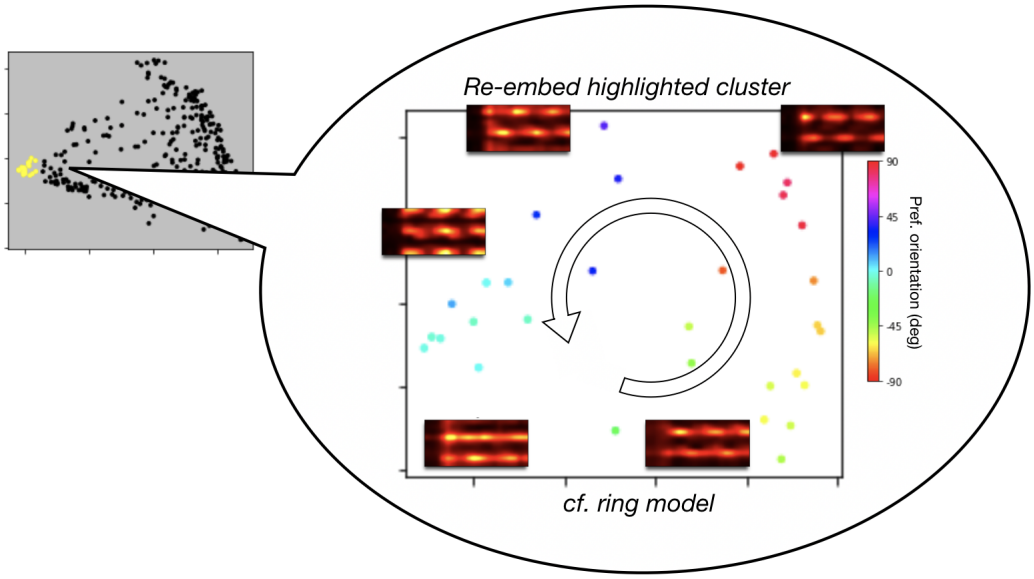

Figure 6.10: Re-embedding a specific neighborhood from the full cortical manifold. Here, we run the algorithm on cells that respond exclusively to low-frequency gratings (Fig. 6.8-A). By letting their original direction/orientation preferences intact, we obtain an embedding that is reminiscent of the ring model, as orientation preference is organized around a circle.

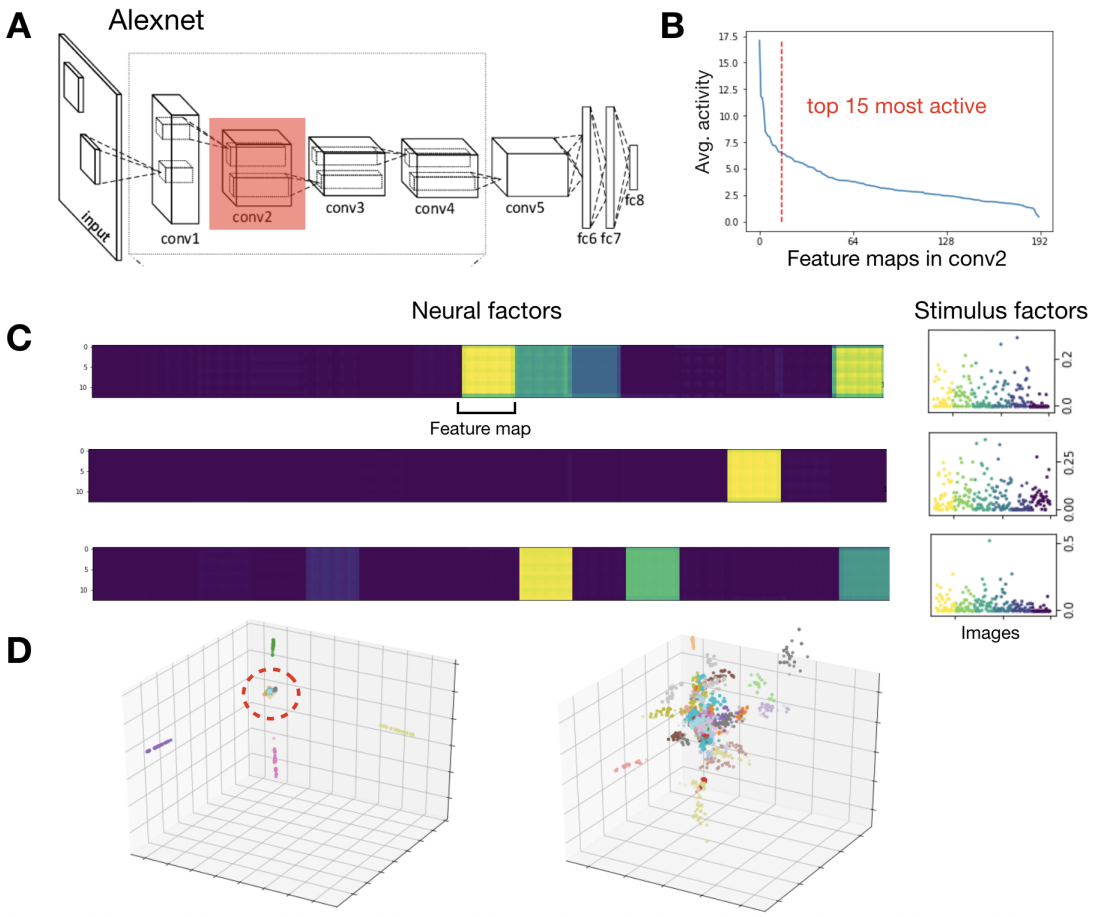

Figure 6.11: Building the neural manifold of a deep convolutional network. **A**: Diagram of Alexnet (adapted from [81]). **B**: After using images from Imagenet as stimuli, 169 neurons were sampled from each of 15 most activate feature maps from the second layer ($N = 2535$). **C**: examples of the factors obtained. Neural factors: visualized as heatmap, each neuron is represented as a pixel, and individual feature maps are arranged as 13×13 square tiles. The factors tell us that neurons in the same feature map fire together, given that our stimuli are randomly shifted over different trials. Stimulus factors: each dot is an image from one of the 5 categories used. Different factors (i.e., different feature maps) respond more strongly to specific images or categories. **D**: We obtained a highly clustered embedding where neurons are color-labeled by feature map. Neurons from the same feature map are clustered together, which is not entirely surprising since they share the same weights, so should give similar outputs for the same input images.

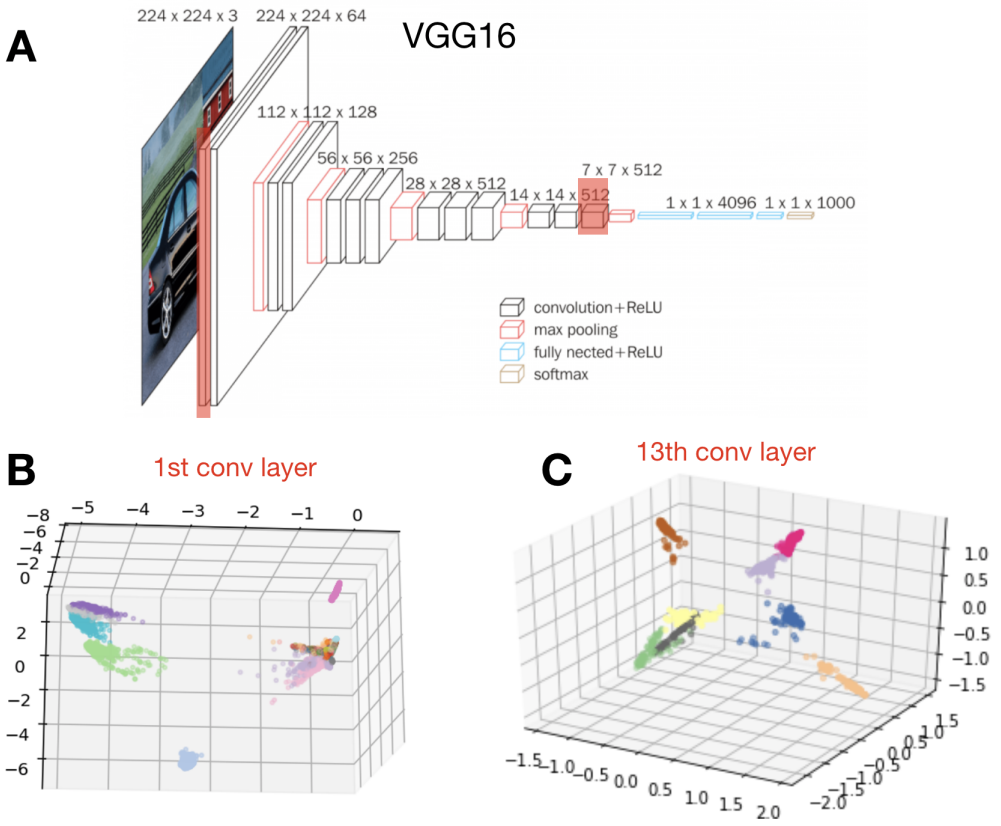

Figure 6.12: Embeddings of superficial and deep layers from the VGG16 network [125]. A: Diagram of the network; as in Fig. 6.11 using Imagenet as input stimuli. B: Results for both the first (B) and 13th (C) convolutional layers look qualitatively the same: neurons in the same feature map are tightly clustered, with some feature maps more similar than others (in terms of their preference for images). Mixing multiple layers together does not change the results either (not shown).

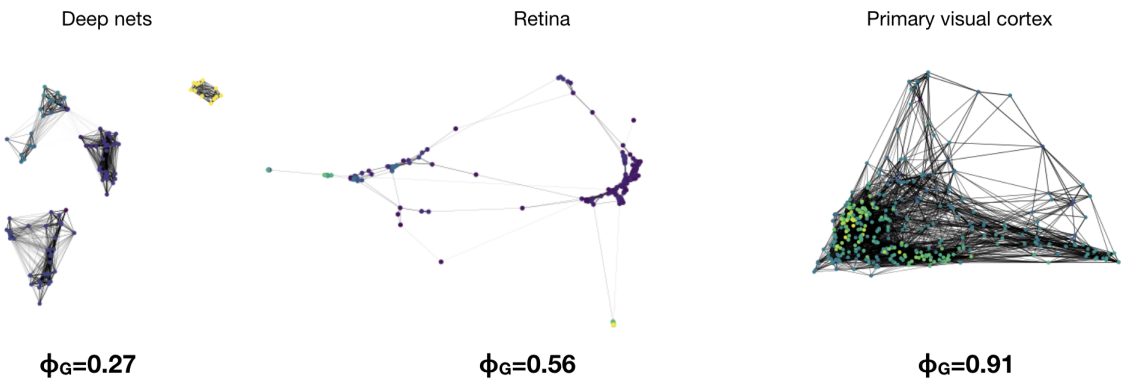

Figure 6.13: Comparison of mean flow ratio (ϕ_G) for the three examples discussed in this chapter. From a functional architecture perspective, deep nets are more similar to the retinathan to primary visual cortex. Deep nets: highly clustered; local neighborhoods correspond to feature maps. Retina: clustered; local neighborhoods correspond to physiological types. Cortex: properties vary smoothly between nearby neighborhoods.

Chapter 7

Discussion

7.1 Summary and conclusions

The rapid development of multi-electrode and imaging techniques is leading to a data explosion in neuroscience, opening the possibility of truly understanding the organization and functionality of our visual systems. However, at the operational level the need for more natural stimuli greatly increases the complexity of the data. Together, these create a challenge for machine learning. Techniques to infer organization and function from an operating visual system are required. Our goal in this thesis was to develop one such technique. The central pillar of our contribution is designing a manifold of neurons, and providing an algorithmic approach to inferring it. This manifold is functional, in the sense that nearby neurons on the manifold are those that are likely to participate in common circuits, since neurons that are near one another respond similarly (in time) to similar aspects of the stimulus ensemble. By organizing the neurons, our manifold differs from other, standard manifolds as they are used in visual neuroscience which instead organize the stimuli. In effect, our manifold is a type of 'inverse' to the standard approach.

Our contributions are twofold. First, we developed the machine learning techniques to infer the manifold of neurons. We set this as a multistage process, beginning with the most basic organization of all the data. Adopting a multi-linear view of potential circuitry, we developed a tensor representation of the data. Three modes are used: (i) the neurons, (ii) the stimulus ensemble, and (iii) their responses represented as a PSTH (peristimulus time histogram).

Non-negative tensor factorization was used as a means to get an approximate multi-clustering of the data. That is, each factor suggests, in a linear approximation, which neurons are responding to certain stimuli in a similar fashion. When the neural loadings are orthogonal and the interactions linear, this suffices as a functional model, as shown with an artificial example. In general, however, the factors are far from orthogonal. We interpret this as indicating that the same neurons could participate in the processing of many

different stimuli. That is, neurons in general participate in multiple functional circuits, which leads to our second stage, the use of diffusion geometry to articulate the neural manifold. Key to this choice is the network/manifold duality: at finite scale the data are discrete, thereby suggesting a network among neurons.

Complicating matters is determining the tensor rank, or how many factors to use. Normally a reconstruction norm is employed, but that is not always appropriate in the neuroscience context. Noise is inherent in the process and our goal is not to model it. Instead, we develop an intermediate representation between the tensor factors and the diffusion kernel, what we call the neural matrix. The columns of this matrix are specified by the neural (loadings of the) factors, and can be interpreted as a basis for a new space in which neurons can be plotted.

The neural matrix is informative. Since this space is Euclidean, other basis vectors could be obtained through linear combinations of existing ones. This would not add new structure to the space, and leads to our condition on tensor rank: to add new factors until the neural matrix saturates in expressivity. This latter expressivity condition is formally modeled in terms of 'variance explained.'

The overall topology of the manifold provides information about qualitative circuit characteristics. If the data were sampled from functionally distinct circuits, then the manifold would be discontinuous, with a separate 'cluster' of neurons for each distinct circuit component. If the data were sampled from a richly interconnected circuit, then the manifold would be continuous. Artificial examples of random receptive fields and the ring model for orientation selectivity in visual cortex confirm these observations. Applying these ideas to actual neurophysiological data is a major goal for this project.

Before studying real data, a key problem remains. To apply manifold learning techniques, a kernel on the data must be defined. This kernel specifies how 'similiar' two data points—-neurons, in our case—are to each other. Like many others, we employ a Gaussian kernel, but this requires specifying its bandwidth. Normally one assumes that the data are roughly uniform but, for our neuroscience problem, this is clearly not the case. Instead, we introduce a kernel in the neural matrix space, then refine it based on graph sparsification techniques. This is a second contribution to the machine learning component of the thesis, and we expect that the use of graph-theoretic conditioners for the kernel in manifold learning will become a direction for further research.

We are now ready to turn to the neuroscience problem. The first issue is that, for the algorithm to work, the underlying circuitry must be exercised to as full an extent as possible. Earlier work had indicated that standard stimuli were too limiting for the mouse's visual system, but unconstrained natural stimuli are virtually impossible to analyze. (In terms above, the stimulus mode of the tensor would become large, so it would be impossible to gather enough data.) We discovered that a class of flow stimuli, which simulate what the mouse would 'see' running through a field, actually exercise much more of its visual system than the artificial stimuli. These, together with the artificial stimuli, provide

our stimulus ensemble.

We have begun two collaborations to show our stimuli to the mouse. Beginning with the retina, the Field Laboratory at Duke University was able to record responses in retinal ganglion cells to the stimulus ensemble. At first glance, one might guess that the retina would consist of discrete circuits, since much of the processing is feed-forward. Applying our algorithms to the Duke data showed that, in fact, this is only partly the case, at least for preliminary data. Some retinal ganglion cell types do cluster, while others are more extended in the manifold. In retrospect this is completely plausible, since horizontal and amacrine cells provide the substrate for interaction.

The situation in cortex is drastically different. Now, working with data from area V1 in mouse obtained in the Stryker Laboratory at the University of California–San Francisco, preliminary data were obtained from an awake, behaving mouse viewing the full stimulus ensemble. Coordinates on the manifold of neurons are similar to those on the retinal embeddings.

The primary difference between retina and cortex was the extent of recurrence and interconnection. A measure of conductance was developed to formalize this, and indeed the cortex exhibits an extremely high degree of connectivity. Modern work suggests there are about 40 different types of retinal ganglion cells. Our manifolds suggest that, for cortex, the situation will be even more extreme. From a functional circuit point of view, in fact, there may be a continuum of cortical function types.

Finally, we turn to perhaps the most widely used model for cortex, deep convolutional networks. Their feedforward architecture leads to manifolds that are even more clustered than the retina, and not at all like those of cortex. This suggests, perhaps, that they may not suffice as general models for Artificial Intelligence.

We close by revisiting the two conjectures at the beginning of this thesis.

CONJECTURE 1 Although the retina can be viewed as an 'outcropping' of the brain, the (relatively) distinct retinal ganglion cell types differ fundamentally from the functional cell types in cortex, which are distributed much more continuously.

Our analysis of preliminary data is consistent with this conjecture.

CONJECTURE 2 Although deep convolutional networks have been used as models of cortex, and as models for cognitive vision, they are closer to big retinas than to little brains.

Our analysis of standard networks, when compared to biological networks, is consistent with this conjecture.

7.2 Final remarks and future work

7.2.1 Exploring different stimulus ensembles

A key feature of our approach is that it allows for explicitly comparing different stimulus ensembles in terms of the ranks of the neural matrices they produce. Relevant stimuli are those that when added to a given ensemble will result in an accompanying increase in rank. The possibility of combining multiple experiments into a single manifold is highly attractive since it is a way of remedying the sampling restrictions in real neurons.

7.2.2 Robust kernels for manifold learning

Sparse data graphs remain stable over a larger range of ε, which reduces the importance of a specific choice of that parameter. Importantly, our measure of how continuous/clustered a manifold is also robust regardless of sparsification. The examples shown seem to indicate that perhaps the best approach is to choose a larger ε than the average minimal distance (eq. 4.6) and then sparsify the resulting graph. Not only does this seem to give stable results, it is computationally efficient, especially for large data sets, providing a promising avenue for further investigation. Additionally, sparsification allows for the identification of each point's 'discrete neighbors', which can be used in multiscale approaches to bandwidth selection.

7.2.3 Applications to artificial neural networks

Our results show that deep convolutional networks yield manifolds that are disconnected like the retina, not continuous like V1. For neuroscience, this could explain the apparent ceiling in modeling cortical data; the limitations of categorical tasks; and suggests future modeling directions. For AI, our manifold can identify co-activations of feature maps across layers, revealing higher-order features, and suggesting different approaches to performing 'dropout.' More generally, it illuminates limitations on the categorization problem, and underlines the importance of recurrence in networks for more complex tasks. A natural next step will be to extend this analysis to artificial recurrent networks and compare the results.

Bibliography

[1] Ralph Abraham, Jerrold E Marsden, and Jerrold E Marsden. *Foundations of mechanics*, volume 36. Benjamin/Cummings Publishing Company Reading, Massachusetts, 1978. 59

[2] E. Acar and B. Yener. Unsupervised multiway data analysis: A literature survey. *IEEE Transactions on Knowledge and Data Engineering*, 21(1):6–20, 2009. 19

[3] Evrim Acar, Daniel M. Dunlavy, and Tamara G. Kolda. A scalable optimization approach for fitting canonical tensor decompositions. *Journal of Chemometrics*, 25(2):67–86, February 2011. 21

[4] Reyan Ahmed, Greg Bodwin, Faryad Darabi Sahneh, Keaton Hamm, Mohammad Javad Latifi Jebelli, Stephen Kobourov, and Richard Spence. Graph spanners: A tutorial review. *Computer Science Review*, 37:100253, 2020. 48

[5] Hirotugu Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974. 31

[6] Ingo Althöfer, Gautam Das, David Dobkin, Deborah Joseph, and José Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9(1):81–100, 1993. 48

[7] Theodore Wilbur Anderson et al. Asymptotic theory for principal component analysis. *Annals of Mathematical Statistics*, 34(1):122–148, 1963. 31

[8] W Ross Ashby. *An introduction to cybernetics*. Chapman & Hall Ltd, 1961. 1, 5

[9] Tom Baden, Philipp Berens, Katrin Franke, Miroslav Román Rosón, Matthias Bethge, and Thomas Euler. The functional diversity of retinal ganglion cells in the mouse. *Nature*, 529(7586):345–350, 2016. 13

[10] Joshua Batson, Daniel A Spielman, Nikhil Srivastava, and Shang-Hua Teng. Spectral sparsification of graphs: theory and algorithms. *Communications of the ACM*, 56(8):87–94, 2013. 47, 48

[11] Submitter Eleanor Batty, Nikhil Parthasarathy, William Falcon, Thomas Rutten, Mohit Rajpal, EJ Chichilnisky, and Liam Paninski. Deep networks for decoding natural images from retinal signals. 6

[12] Mikhail Belkin. Problems of learning on manifolds. 2003. 42

[13] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003. 47

[14] Ohad Ben-Shahar and Steven Zucker. Geometrical computations explain projection patterns of long-range horizontal connections in visual cortex. *Neural computation*, 16(3):445–476, 2004. 3

[15] Ohad Ben-Shahar and Steven W. Zucker. The perceptual organization of texture flow: A contextual inference approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(4):401–417, 2003. 3

[16] Rani Ben-Yishai, R Lev Bar-Or, and Haim Sompolinsky. Theory of orientation tuning in visual cortex. *Proceedings of the National Academy of Sciences*, 92(9):3844–3848, 1995. 11

[17] Emmanuel J Candes, Carlos A Sing-Long, and Joshua D Trzasko. Unbiased risk estimates for singular value thresholding and spectral estimators. *IEEE transactions on signal processing*, 61(19):4643–4657, 2013. 31

[18] Emmanuel J Candes, Carlos A Sing-Long, and Joshua D Trzasko. Unbiased risk estimates for singular value thresholding and spectral estimators. *IEEE transactions on signal processing*, 61(19):4643–4657, 2013. 31

[19] Matteo Carandini and Anne K Churchland. Probing perceptual decisions in rodents. *Nat Neurosci*, 16(7):824, 2013. 58

[20] Matteo Carandini, Jonathan B Demb, Valerio Mante, David J Tolhurst, Yang Dan, Bruno A Olshausen, Jack L Gallant, and Nicole C Rust. Do we know what the early visual system does? *J Neurosci*, 25(46):10577–10597, 2005. 59

[21] J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319, 1970. 19

[22] J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319, 1970. 20

[23] Rishidev Chaudhuri, Berk Gerçek, Biraj Pandey, Adrien Peyrache, and Ila Fiete. The intrinsic attractor manifold and population dynamics of a canonical cognitive circuit across waking and sleep. *Nature neuroscience*, 22(9):1512–1520, 2019. 6

[24] EJ Chichilnisky. A simple white noise analysis of neuronal light responses. *Network: Computation in Neural Systems*, 12(2):199–213, 2001. 3

[25] Yunjin Choi, Jonathan Taylor, and Robert Tibshirani. Selecting the number of principal components: Estimation of the true rank of a noisy matrix. *The Annals of Statistics*, pages 2590–2617, 2017. 31

[26] Timothy Chu, Yu Gao, Richard Peng, Sushant Sachdeva, Saurabh Sawlani, and Junxing Wang. Graph sparsification, spectral sketches, and faster resistance computation via short cycle decompositions. *SIAM Journal on Computing*, (0):FOCS18–85, 2020. 47

[27] Jason E Chung, Jeremy F Magland, Alex H Barnett, Vanessa M Tolosa, Angela C Tooker, Kye Y Lee, Kedar G Shah, Sarah H Felix, Loren M Frank, and Leslie F Greengard. A fully automated approach to spike sorting. *Neuron*, 95(6):1381–1394, 2017. 63

[28] SueYeon Chung, Daniel D Lee, and Haim Sompolinsky. Classification and geometry of general perceptual manifolds. *Physical Review X*, 8(3):031003, 2018. 5

[29] Mark M Churchland, John P Cunningham, Matthew T Kaufman, Justin D Foster, Paul Nuyujukian, Stephen I Ryu, and Krishna V Shenoy. Neural population dynamics during reaching. *Nature*, 487(7405):51–56, 2012. 5

[30] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-ichi Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009. 18, 19, 20

[31] Ronald R Coifman and Stéphane Lafon. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006. 10, 41, 42, 44

[32] Ronald R Coifman, Yoel Shkolnisky, Fred J Sigworth, and Amit Singer. Graph laplacian tomography from unknown random projections. *IEEE Transactions on Image Processing*, 17(10):1891–1899, 2008. 43, 46

[33] Ludovico Coletta, Marco Pagani, Jennifer D Whitesell, Julie A Harris, Boris Bernhardt, and Alessandro Gozzi. Network structure of the mouse brain connectome with voxel resolution. *Science Advances*, 6(51):eabb7187, 2020. 2

[34] Steven J Cook, Travis A Jarrell, Christopher A Brittin, Yi Wang, Adam E Bloniarz, Maksim A Yakovlev, Ken CQ Nguyen, Leo T-H Tang, Emily A Bayer, Janet S Duerr, et al. Whole-animal connectomes of both caenorhabditis elegans sexes. *Nature*, 571(7763):63–71, 2019. 2

[35] Benjamin R Cowley, Matthew A Smith, Adam Kohn, and Byron M Yu. Stimulus-driven population activity patterns in macaque primary visual cortex. *PLoS computational biology*, 12(12):e1005185, 2016. 5

[36] John P Cunningham and M Yu Byron. Dimensionality reduction for large-scale neural recordings. *Nature neuroscience*, 17(11):1500–1509, 2014. 4

[37] Maria C Dadarlat and Michael P Stryker. Locomotion enhances neural encoding of visual stimuli in mouse V1. *J Neurosci*, 37(14):3764–3775, 2017. 63

[38] Jonathan B Demb and Joshua H Singer. Functional circuitry of the retina. *Annual review of vision science*, 1:263–289, 2015. 13

[39] Sophie Deneve, Peter E Latham, and Alexandre Pouget. Reading population codes: a neural implementation of ideal observers. *Nature neuroscience*, 2(8):740–745, 1999. 4

[40] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 71, 76

[41] Steven H Devries and Denis A Baylor. Mosaic arrangement of ganglion cell receptive fields in rabbit retina. *Journal of neurophysiology*, 78(4):2048–2060, 1997. 74

[42] Edsger W Dijkstra et al. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959. 47

[43] RM Douglas, A Neve, JP Quittenbaum, NM Alam, and GT Prusky. Perception of visual motion coherence by rats and mice. *Vision Research*, 46(18):2842–2847, 2006. 58

[44] Luciano Dyballa, Mahmood S Hoseini, Maria C Dadarlat, Steven W Zucker, and Michael P Stryker. Flow stimuli reveal ecologically appropriate responses in mouse visual cortex. *Proceedings of the National Academy of Sciences*, 115(44):11304–11309, 2018. 58, 61, 75

[45] Lief Fenno, Ofer Yizhar, and Karl Deisseroth. The development and application of optogenetics. *Annu Rev Neurosci*, 34, 2011. 58

[46] Greg D Field, Jeffrey L Gauthier, Alexander Sher, Martin Greschner, Timothy A Machado, Lauren H Jepson, Jonathon Shlens, Deborah E Gunning, Keith Mathieson, Wladyslaw Dabrowski, et al. Functional connectivity in the retina at the resolution of photoreceptors. *Nature*, 467(7316):673–677, 2010. 73, 74

[47] Emmanouil Froudarakis, Philipp Berens, Alexander S Ecker, R James Cotton, Fabian H Sinz, Dimitri Yatsenko, Peter Saggau, Matthias Bethge, and Andreas S Tolias. Population code in mouse V1 facilitates readout of natural scenes through increased sparseness. *Nature neuroscience*, 17(6):851–857, 2014. 4

[48] Emmanouil Froudarakis, Philipp Berens, Alexander S Ecker, R James Cotton, Fabian H Sinz, Dimitri Yatsenko, Peter Saggau, Matthias Bethge, and Andreas S Tolias. Population code in mouse V1 facilitates readout of natural scenes through increased sparseness. *Nat Neurosci*, 17(6):851, 2014. 59

[49] Emmanouil Froudarakis, Uri Cohen, Maria Diamantaki, Edgar Y Walker, Jacob Reimer, Philipp Berens, Haim Sompolinsky, and Andreas S Tolias. Object manifold geometry across the mouse cortical visual hierarchy. *bioRxiv*, 2020. 6

[50] Yu Fu, Jason M Tucciarone, J Sebastian Espinosa, Nengyin Sheng, Daniel P Darcy, Roger A Nicoll, Z Josh Huang, and Michael P Stryker. A cortical circuit for gain control by behavioral state. *Cell*, 156(6):1139–1152, 2014. 58

[51] Leon Glass. Moire effect from random dots. *Nature*, 223(5206):578, 1969. 59

[52] Gene Golub, Virginia Klema, and Gilbert W Stewart. Rank degeneracy and least squares problems. Technical report, Stanford University, CA, Dept. of Computer Science, 1976. 30

[53] Patrick J Grother and Kayee Hanaoka. Nist special database 19. *Handprinted forms and characters database, National Institute of Standards and Technology*, 2016. 37

[54] Laleh Haghverdi, Florian Buettner, and Fabian J Theis. Diffusion maps for high-dimensional single-cell analysis of differentiation data. *Bioinformatics*, 31(18):2989–2998, 2015. 46

[55] Per Christian Hansen. *Rank-deficient and discrete ill-posed problems: numerical aspects of linear inversion*. SIAM, 1998. 30

[56] Kenneth D Harris and Alexander Thiele. Cortical state and attention. *Nat Rev Neurosci*, 12(9):509, 2011. 58

[57] Richard A Harshman et al. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970. 19

[58] Richard A Harshman et al. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970. 20

[59] Christopher D Harvey, Philip Coen, and David W Tank. Choice-specific sequences in parietal cortex during a virtual-navigation decision task. *Nature*, 484(7392):62–68, 2012. 5

[60] Alexander Heitman, Nora Brackbill, Martin Greschner, Alexander Sher, Alan M Litke, and EJ Chichilnisky. Testing pseudo-linear models of responses to natural scenes in primate retina. *bioRxiv*, page 045336, 2016. 3

[61] Eckhard H Hess and James M Polt. Pupil size as related to interest value of visual stimuli. *Science*, 132(3423):349–350, 1960. 58

[62] Frank L Hitchcock. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics*, 6(1-4):164–189, 1927. 19

[63] Frank L Hitchcock. Multiple invariants and generalized rank of a p-way matrix or tensor. *Journal of Mathematics and Physics*, 7(1-4):39–79, 1927. 19

[64] S Hochstein and RM Shapley. Linear and nonlinear spatial subunits in y cat retinal ganglion cells. *The Journal of physiology*, 262(2):265–284, 1976. 11

[65] David Hong, Tamara G Kolda, and Jed A Duersch. Generalized canonical polyadic tensor decomposition. *SIAM Review*, 62(1):133–163, 2020. 21, 22

[66] Jennifer L Hoy, Iryna Yavorska, Michael Wehr, and Cristopher M Niell. Vision drives accurate approach behavior during prey capture in laboratory mice. *Curr Biol*, 26(22):3046–3052, 2016. 58

[67] David H Hubel and Torsten N Wiesel. Receptive fields of single neurones in the cat's striate cortex. *J Physiol*, 148(3):574–591, 1959. 13

[68] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154, 1962. 15

[69] David H Hubel and Torsten N Wiesel. Uniformity of monkey striate cortex: a parallel relationship between field size, scatter, and magnification factor. *Journal of Comparative Neurology*, 158(3):295–305, 1974. 11

[70] Ashley L Juavinett and Edward M Callaway. Pattern and component motion responses in mouse visual cortical areas. *Curr Biol*, 25(13):1759–1764, 2015. 58

[71] Carme Julià, Angel D Sappa, Felipe Lumbreras, Joan Serrat, and Antonio López. Rank estimation in missing data matrix problems. *Journal of Mathematical Imaging and Vision*, 39(2):140–160, 2011. 31

[72] Christoph Kayser, Rodrigo F Salazar, and Peter Konig. Responses to natural scenes in cat v1. *Journal of neurophysiology*, 90(3):1910–1920, 2003. 3

[73] Yosi Keller, Ronald R Coifman, Stéphane Lafon, and Steven W Zucker. Audio-visual group recognition using diffusion maps. *IEEE Transactions on Signal Processing*, 58(1):403–413, 2009. 46, 52

[74] William F Kindel, Elijah D Christensen, and Joel Zylberberg. Using deep learning to probe the neural code for images in primary visual cortex. *Journal of vision*, 19(4):29–29, 2019. 12

[75] Gustav Kirchhoff. Ueber die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Vertheilung galvanischer Ströme geführt wird. *Annalen der Physik*, 148(12):497–508, 1847. 47

[76] Tamara G. Kolda. Multilinear operators for higher-order decompositions. Technical Report SAND2006-2081, Sandia National Laboratories, April 2006. 20

[77] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009. 8, 19, 21

[78] Zbigniew J Kowalik, Andrzej Wróbel, and Andrzej Rydz. Why does the human brain need to be a nonlinear system? *Behavioral and Brain Sciences*, 19(2):302–302, 1996. 9

[79] Nikolaus Kriegeskorte. Deep neural networks: a new framework for modeling biological vision and brain information processing. *Annual review of vision science*, 1:417–446, 2015. 6

[80] Nikolaus Kriegeskorte and Tal Golan. Neural network models and deep learning. *Current Biology*, 29(7):R231–R236, 2019. 12

[81] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012. 15, 76, 86

[82] Norbert Kruger, Peter Janssen, Sinan Kalkan, Markus Lappe, Ales Leonardis, Justus Piater, Antonio J Rodriguez-Sanchez, and Laurenz Wiskott. Deep hierarchies in the primate visual cortex: What can we learn for computer vision? *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1847–1871, 2012. 14

[83] Stéphane Lafon. *Diffusion maps and geometric harmonics*. PhD thesis, Yale University, 2004. 41, 43, 44, 45, 46

[84] Stephane Lafon, Yosi Keller, and Ronald R Coifman. Data fusion and multicue data matching by diffusion maps. *IEEE Transactions on pattern analysis and machine intelligence*, 28(11):1784–1797, 2006. 46

[85] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015. 12

[86] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 37

[87] Seokjin Lee. Estimating the rank of a nonnegative matrix factorization model for automatic music transcription based on stein's unbiased risk estimator. *Applied Sciences*, 10(8):2911, 2020. 31

[88] Ofir Lindenbaum, Moshe Salhov, Arie Yeredor, and Amir Averbuch. Gaussian bandwidth selection for manifold learning and classification. *Data mining and knowledge discovery*, 34(6):1676–1712, 2020. 46

[89] Grace W Lindsay. Convolutional neural networks as a model of the visual system: past, present, and future. *Journal of cognitive neuroscience*, pages 1–15, 2020. 12

[90] Norah K Link and Steven W Zucker. Sensitivity to corners in flow patterns. *Spat Vis*, 2(3):233–244, 1987. 59

[91] Ryan J Low, Sam Lewallen, Dmitriy Aronov, Rhino Nevers, and David W Tank. Probing variability in a cognitive map using manifold inference from neural dynamics. *BioRxiv*, page 418939, 2018. 6

[92] Sean P MacEvoy, Timothy D Hanks, and Michael A Paradiso. Macaque v1 activity during natural vision: effects of natural scenes and saccades. *Journal of neurophysiology*, 99(2):460–472, 2008. 3

[93] Zachary F Mainen and Terrence J Sejnowski. Reliability of spike timing in neocortical neurons. *Science*, 268(5216):1503–1506, 1995. 11, 32

[94] V. Z. Marmarelis. Signal transformation and coding in neural systems. *IEEE Transactions on Biomedical Engineering*, 36(1):15–24, 1989. 11, 13

[95] Mark Mazurek, Marisa Kager, and Stephen D Van Hooser. Robust quantification of orientation selectivity and direction selectivity. *Frontiers in Neural Circuits*, 8:92, 2014. 73

[96] Matthew J McGinley, Martin Vinck, Jacob Reimer, Renata Batista-Brito, Edward Zagha, Cathryn R Cadwell, Andreas S Tolias, Jessica A Cardin, and David A McCormick. Waking state: rapid variations modulate neural and behavioral responses. *Neuron*, 87(6):1143–1161, 2015. 58

[97] Lane T McIntosh, Niru Maheswaranathan, Aran Nayebi, Surya Ganguli, and Stephen A Baccus. Deep learning models of the retinal response to natural scenes. *Advances in neural information processing systems*, 29:1369, 2016. 6

[98] Gal Mishne and Israel Cohen. Multiscale anomaly detection using diffusion maps. *IEEE Journal of selected topics in signal processing*, 7(1):111–123, 2012. 46

[99] Gal Mishne, Ronen Talmon, Ron Meir, Jackie Schiller, Maria Lavzin, Uri Dubin, and Ronald R Coifman. Hierarchical coupled-geometry analysis for neuronal structure and activity pattern discovery. *IEEE Journal of Selected Topics in Signal Processing*, 10(7):1238–1253, 2016. 6

[100] Morten Mørup, Lars Kai Hansen, Josef Parnas, and Sidse M Arnfred. Decomposing the time-frequency representation of eeg using non-negative matrix and multi-way factorization. *Technical University of Denmark Technical Report*, pages 1–28, 2006. 9, 22

[101] Cristopher M Niell and Michael P Stryker. Highly selective receptive fields in mouse visual cortex. *J Neurosci*, 28(30):7520–7536, 2008. 58, 60, 62, 65, 68

[102] Cristopher M Niell and Michael P Stryker. Modulation of visual responses by behavioral state in mouse visual cortex. *Neuron*, 65(4):472–479, 2010. 58, 61

[103] Ganna Palagina, Jochen F Meyer, and Stelios M Smirnakis. Complex visual motion representation in mouse area V1. *J Neurosci*, 37(1):164–183, 2017. 58

[104] Chethan Pandarinath, Daniel J O'Shea, Jasmine Collins, Rafal Jozefowicz, Sergey D Stavisky, Jonathan C Kao, Eric M Trautmann, Matthew T Kaufman, Stephen I Ryu, Leigh R Hochberg, et al. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature methods*, 15(10):805–815, 2018. 4

[105] S. Paycha. Introductory article: Differential geometry. In Jean-Pierre Françoise, Gregory L. Naber, and Tsou Sheung Tsun, editors, *Encyclopedia of Mathematical Physics*, pages 33–40. Academic Press, Oxford, 2006. 43

[106] David Peleg and Alejandro A Schäffer. Graph spanners. *Journal of graph theory*, 13(1):99–116, 1989. 48

[107] Glen T Prusky, Paul WR West, and Robert M Douglas. Behavioral assessment of visual acuity in mice and rats. *Vision Research*, 40(16):2201–2209, 2000. 58

[108] Sneha Ravi, Daniel Ahn, Martin Greschner, EJ Chichilnisky, and Greg D Field. Pathway-specific asymmetries between on and off visual signals. *Journal of Neuroscience*, 38(45):9728–9740, 2018. 73

[109] Jacob Reimer, Emmanouil Froudarakis, Cathryn R Cadwell, Dimitri Yatsenko, George H Denfield, and Andreas S Tolias. Pupil fluctuations track fast switching of cortical states during quiet wakefulness. *Neuron*, 84(2):355–362, 2014. 58

[110] Craig W Reynolds. Flocks, herds and schools: A distributed behavioral model. *Computer Graphics (SIGGRAPH 87 Conference Proceedings)*, 21(4):25–34, 1987. 59

[111] Craig W Reynolds. Steering behaviors for autonomous characters. In *Proceedings of the 1999 Game Developers Conference*, pages 763–782, 1999. 59

[112] Craig W Reynolds. Interaction with groups of autonomous characters. In *Proceedings of the 2000 Game Developers Conference*, volume 21, 07 2000. 59

[113] Blake A Richards, Timothy P Lillicrap, Philippe Beaudoin, Yoshua Bengio, Rafal Bogacz, Amelia Christensen, Claudia Clopath, Rui Ponte Costa, Archy de Berker, Surya Ganguli, et al. A deep learning framework for neuroscience. *Nature neuroscience*, 22(11):1761–1770, 2019. 12

[114] Rajeev V Rikhye and Mriganka Sur. Spatial correlations in natural scenes modulate response reliability in mouse visual cortex. *J Neurosci*, 35(43):14661–14680, 2015. 59

[115] David Rolnick and Konrad Kording. Reverse-engineering deep relu networks. In *International Conference on Machine Learning*, pages 8178–8187. PMLR, 2020. 3

[116] Daniel L Ruderman and William Bialek. Statistics of natural images: Scaling in the woods. In *Adv Neural Inf Process Syst*, pages 551–558, 1994. 59

[117] Abigail A Russo, Sean R Bittner, Sean M Perkins, Jeffrey S Seely, Brian M London, Antonio H Lara, Andrew Miri, Najja J Marshall, Adam Kohn, Thomas M Jessell, et al. Motor cortex embeds muscle-like commands in an untangled population response. *Neuron*, 97(4):953–966, 2018. 5

[118] Joshua R Sanes and Richard H Masland. The types of retinal ganglion cells: current status and implications for neuronal classification. *Annual review of neuroscience*, 38:221–246, 2015. 13

[119] Alexander Schrijver. On the history of the transportation and maximum flow problems. *Mathematical programming*, 91(3):437–445, 2002. 54

[120] Gideon Schwarz et al. Estimating the dimension of a model. *Annals of statistics*, 6(2):461–464, 1978. 31

[121] Amnon Shashua and Tamir Hazan. Non-negative tensor factorization with applications to statistics and computer vision. In *Proceedings of the 22nd international conference on Machine learning*, pages 792–799, 2005. 72

[122] Qiquan Shi, Haiping Lu, and Yiu-Ming Cheung. Rank-one matrix completion with automatic rank estimation via l1-norm regularization. *IEEE transactions on neural networks and learning systems*, 29(10):4744–4757, 2017. 31

[123] Joshua H Siegle, Xiaoxuan Jia, Séverine Durand, Sam Gale, Corbett Bennett, Nile Graddis, Greggory Heller, Tamina K Ramirez, Hannah Choi, Jennifer A Luviano, et al. Survey of spiking in the mouse visual system reveals functional hierarchy. *Nature*, pages 1–7, 2021. 2

[124] Eero P Simoncelli and Bruno A Olshausen. Natural image statistics and neural representation. *Annu Rev Neurosci*, 24(1):1193–1216, 2001. 59

[125] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 76, 87

[126] Adam C Snyder, M Yu Byron, and Matthew A Smith. Distinct population codes for attention in the absence and presence of visual stimulation. *Nature communications*, 9(1):1–14, 2018. 5

[127] Daniel Spielman. Spectral and algebraic graph theory. *Yale lecture notes, draft of December 4*, 2019. 47, 54

[128] Daniel A Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6):1913–1926, 2011. 48, 54

[129] Ljubiša Stanković, Danilo Mandic, Miloš Daković, Miloš Brajović, Bruno Scalzo, Shengxi Li, Anthony G Constantinides, et al. Data analytics on graphs part i: Graphs and spectra on graphs. *Foundations and Trends® in Machine Learning*, 13(1), 2020. 47

[130] Ljubiša Stanković, Danilo Mandic, Miloš Daković, Miloš Brajović, Bruno Scalzo, Shengxi Li, Anthony G Constantinides, et al. Data analytics on graphs part i: Graphs and spectra on graphs. *Foundations and Trends® in Machine Learning*, 13(1), 2020. 54

[131] Richard B Stein, E Roderich Gossen, and Kelvin E Jones. Neuronal variability: noise or part of the signal? *Nature Reviews Neuroscience*, 6(5):389–397, 2005. 21

[132] Jeffrey N Stirman, Leah B Townsend, and Spencer L Smith. A touchscreen based global motion perception task for mice. *Vision Research*, 127:74–83, 2016. 59

[133] Gilbert Strang. The functions of deep learning. *SIAM news*, 51(10):1–4, 2018. 13

[134] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000. 43

[135] Shashanka Ubaru and Yousef Saad. Fast methods for estimating the numerical rank of large matrices. In *International Conference on Machine Learning*, pages 468–477. PMLR, 2016. 30

[136] Shashanka Ubaru, Yousef Saad, and Abd-Krim Seghouane. Fast estimation of approximate matrix ranks using spectral densities. *Neural computation*, 29(5):1317–1351, 2017. 30, 31

[137] Madeleine Udell and Alex Townsend. Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science*, 1(1):144–160, 2019. 30

[138] Magnus O Ulfarsson and Victor Solo. Dimension estimation in noisy pca with sure and random matrix theory. *IEEE transactions on signal processing*, 56(12):5804–5816, 2008. 31

[139] Magnus O Ulfarsson and Victor Solo. Selecting the number of principal components with sure. *IEEE Signal Processing Letters*, 22(2):239–243, 2014. 31

[140] Lav R Varshney, Beth L Chen, Eric Paniagua, David H Hall, and Dmitri B Chklovskii. Structural properties of the caenorhabditis elegans neuronal network. *PLoS Comput Biol*, 7(2):e1001066, 2011. 2

[141] N. Vervliet and L. De Lathauwer. A randomized block sampling approach to canonical polyadic decomposition of large-scale tensors. *IEEE Journal of Selected Topics in Signal Processing*, 10(2):284–295, 2016. 9, 22

[142] Martin Vinck, Renata Batista-Brito, Ulf Knoblich, and Jessica A Cardin. Arousal and locomotion make distinct contributions to cortical activity patterns and visual encoding. *Neuron*, 86(3):740–754, 2015. 58

[143] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *The Journal of Machine Learning Research*, 11:2837–2854, 2010. 52

[144] Norbert Wiener. *Nonlinear problems in random theory*. MIT Press, 1958. 3, 11

[145] Alex H Williams, Tony Hyun Kim, Forea Wang, Saurabh Vyas, Stephen I Ryu, Krishna V Shenoy, Mark Schnitzer, Tamara G Kolda, and Surya Ganguli. Unsupervised discovery of demixed, low-dimensional neural dynamics across multiple timescales through tensor component analysis. *Neuron*, 98(6):1099–1115, 2018. 9, 19, 20, 22, 29, 72

[146] Ryan C Williamson, Brent Doiron, Matthew A Smith, and M Yu Byron. Bridging large-scale neuronal recordings and large-scale network models using dimensionality reduction. *Current opinion in neurobiology*, 55:40–47, 2019. 4

[147] Santosh Kumar Yadav, Rohit Sinha, and Prabin Kumar Bora. An efficient svd shrinkage for rank estimation. *IEEE Signal Processing Letters*, 22(12):2406–2410, 2015. 31

[148] Daniel LK Yamins and James J DiCarlo. Using goal-driven deep learning models to understand sensory cortex. *Nature neuroscience*, 19(3):356–365, 2016. 12

[149] Daniel LK Yamins, Ha Hong, Charles F Cadieu, Ethan A Solomon, Darren Seibert, and James J DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the national academy of sciences*, 111(23):8619–8624, 2014. 6

[150] Xiaoyang Yao, Jon Cafaro, Amanda J McLaughlin, Friso R Postma, David L Paul, Gautam Awatramani, and Greg D Field. Gap junctions contribute to differential light adaptation across direction-selective retinal ganglion cells. *Neuron*, 100(1):216–228, 2018. 73

[151] Wenjing Yin, Derrick Brittain, Jay Borseth, Marie E Scott, Derric Williams, Jedediah Perkins, Christopher S Own, Matthew Murfitt, Russel M Torres, Daniel Kapner, et al. A petascale automated imaging pipeline for mapping neuronal circuits with high-throughput transmission electron microscopy. *Nature communications*, 11(1):1–12, 2020. 2

[152] Yumiko Yoshimura, Jami LM Dantzker, and Edward M Callaway. Excitatory cortical neurons form fine-scale functional networks. *Nature*, 433(7028):868–873, 2005. 6, 10

[153] Wan-Qing Yu, Norberto M Grzywacz, Eun-Jin Lee, and Greg D Field. Cell type-specific changes in retinal ganglion cell function induced by rod death and cone reorganization in rats. *Journal of neurophysiology*, 118(1):434–454, 2017. 73

[154] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Proceedings of the 17th International Conference on Neural Information Processing Systems*, NIPS'04, page 1601–1608, Cambridge, MA, USA, 2004. MIT Press. 46

[155] Qibin Zhao, Liqing Zhang, and Andrzej Cichocki. Bayesian CP factorization of incomplete tensors with automatic rank determination. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1751–1763, 2015. 29

[156] Steven W Zucker. Early orientation selection: tangent fields and the dimensionality of their support. *Computer Vision, Graphics, and Image Processing*, 32(1):74–103, 1985. 59

[157] Steven W Zucker. The fox and the forest: toward a type I/type II constraint for early optical flow. In *Proc. of the ACM SIGGRAPH/SIGART Interdisciplinary Workshop on Motion: Representation and Perception*, pages 29–62. Elsevier North-Holland, Inc., 1986. 59