# Low-rank network models of neural computations

Adrian Valente

PSL
UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à l'École Normale Supérieure

# Low-rank network models of neural computations

Soutenue par
**Adrian VALENTE**
Le 19 octobre 2022

École doctorale n°158
**Cerveau, Cognition, Comportement**

Spécialité
**Neurosciences**

Composition du jury :

| | |
|---|---|
| Devika NARAIN<br>Erasmus Medical Center | *Rapporteur (absente)* |
| Henning SPREKELER<br>Technische Universität Berlin | *Rapporteur* |
| Tatiana ENGEL<br>Cold Spring Harbor Laboratory | *Examinatrice* |
| Jonathan PILLOW<br>Princeton University | *Membre invité* |
| Peter NERI<br>École Normale Supérieure | *Président du jury* |
| Srdjan OSTOJIC<br>École Normale Supérieure | *Directeur de thèse* |

1794
ENS | PSL

## Preface

The work that follows contains the outcome of what was both the most demanding and the most gratifying journey I have underwent. It began in late 2019 when Srdjan Ostojic invited me to spend a fascinating afternoon at his lab. Although, to be honest, it probably began a few years earlier, when I started reading popular science books about brain function, hoping to find answers and finding only question after question, pushing me to deepen the subject by signing up on a master's course, a master's thesis in Paris, and eventually a whole PhD, immersing me entirely in the field before I could realise what had happened. And if I am to be fully honest, this journey really began in 1995, when my parents, additionally to life, managed to instil in me a curiosity for the world around us and a passion for science which stuck with me very strongly (much more strongly than the passion for River Plate, although things can still change).

*Pour cela, ainsi que pour votre soutien sans faille, je vous suis immensément reconnaissant. Cette passion m'apporte une satisfaction continue, celle d'apprendre tous les jours, de partager et d'échanger des connaissances avec d'autres passionnés, et surtout, chose que j'aurais cru impossible il y a encore peu, de contribuer à mon échelle à la construction de ce vaste édifice du savoir commun. Je sais que cette joie de découvrir et d'apprendre me suivra facilement où que j'aille, car elle ne requiert rien d'autre qu'un état d'esprit. Merci donc d'avoir su partager cet état d'esprit avec moi.*

But let us fast-forward a few years to 2019, when Srdjan Ostojic offered me a master's thesis subject, and ultimately a PhD subject, to explore jointly with Alexis Dubreuil, on a research track opened by Francesca Mastrogiuseppe. I had little experience in neuroscience, and could not immediately understand the subject, but I knew right away that if I was to do research, this was what I wanted to study. It consisted in exploring questions I had been wondering about for many years without ever being able to formulate them until then. I hope the enjoyment I had in studying this subject will be reflected in the following pages and will be shared with the reader.

As all those who have been in research know, the path to a PhD is an extremely strenuous one. And like all theses prepared between the years 2020 and 2022, this one has been done in spite of particularly unfavorable circumstances. Environmentally-induced anxiety had to be managed, and many of the most pleasing aspects of research, that is conferences and social exchanges, suddenly disappeared. Yet I have been extremely lucky in this path, in terms of the supervisor, the colleagues and the friends I was gifted with, which all made the good memories of the journey far outweigh the difficulties.

*À mes amis donc, à ceux qui continuent de partager avec moi les passions les plus ésotériques et les conversations les plus passionnantes et saugrenues, à ceux avec qui on s'est évadés dans les endroits les plus inaccessibles qu'on ait trouvé, à ceux avec qui on a fait travailler aussi dur les jambes que l'estomac, à ceux qui m'ont vu m'arracher les cheveux sur du code ou des figures, à ceux qui sont toujours là pour prendre un verre, puis deux et encore quelques autres, bref à tous ceux qui m'ont apporté de la joie pendant ces années: vous êtes les meilleurs, merci beaucoup.*

As this work focuses on the importance of collective activity in neural processes, I have to underline how much it owes to the collective activity within our lab and to continuous exchanges of ideas, and in particular to the following people: to my director Srdjan of course who trusted in me from the beginning, who has always been a supportive and reassuring presence, and managed to strike what I found was the perfect balance between precise advice and scientific liberty ; to Alexis Dubreuil, who immediately took me under his wing and taught me a sea of knowledge about neuroscience and the process of doing research, and with whom a large part of this work was co-authored ; to Manuel Beiran with whom important aspects of the theory were developed, and with whom conversations were always as varied as stimulating ; to Francesca Mastrogiuseppe for opening such a captivating research direction that I explored with constant delight, and for her continued support and advices on the

most detailed aspects of my work ; and to Jonathan Pillow, a scientist whose rigorous and creative approach to statistics and neuroscience I admire, and with whom I have been lucky to pursue exciting collaborations whose first fruits are present in this thesis.

And as these are the figures who had the most direct impact on my work, it owes also a lot to my dear friends and colleagues with whom I have been lucky to coincide in the team, Giulio Bondanelli, Rupesh Kumar, Josef Ladenbauer, Friedrich Schuessler, Ljubica Cimesa, João Barbosa, Yuxiu Shao, Othman Lahrach, Betsy Herbert, and of course Arianna di Bernardo with whom I am happy to share the joys of her first discoveries. Conversations with all of you have helped me tremendously in making my first steps in the wonderful worlds of research and neuroscience, and I hope we will have numerous occasions to meet again, and not only in scientific contexts.

I also have to thank all the other members of the Group for Neural Theory, and of the Laboratoire de Neurosciences Cognitives et Computationnelles, too numerous to be all mentioned, but who all have a contributed to making this lab such a pleasant place to work. Thank you for maintaining such a reliably fun, friendly, light-hearted and good-humoured atmosphere, for these are the perfect conditions under which a PhD should mature. I wish these last years could have given us the opportunity to spend more time together, and I hope that our paths will get to cross often. To be complete, it is indispensable for me to credit all those I met in the broader community of systems neuroscience, a colorful, surprising and welcoming family, and much particularly the students, faculty members, and organizers of the exceptional Eresfjord summer school. Although I have no point of comparison, I am pretty confident in that the neuroscientific community is rather unique in the world of research, probably due to its interdisciplinarity, which makes the field so open to novelty and original.

I wish to finish by encouraging the current and future members of the lab to keep this spirit of constant exchange and good ambient. Science is indeed like a good meal, it is always enjoyable but it tastes much better when it is shared. And as this was the most French sentence with which I could end this appetizer, I shall give the floor to the *plat de résistance*, not without thanking of course the jury members who kindly accepted to review and evaluate this work.

Adrian Valente, Paris, August 2022.

# Contents

# Introduction I

*La science remplace du visible compliqué par de l'invisible simple.*[1]

Jean PERRIN

## I.1 Prologue

The human brain contains 10 billion neurons, connected in an intricate network through 10,000 times more synapses, conspiring altogether to generate percepts, deliberations, memories, and all other aspects of cognition. Even the brains of simpler organisms like fruit flies (*Drosophila melanogaster*), whose whole connectome is currently being mapped (176), contain on the order of a hundred thousand neurons. This outstanding complexity leads to a wide range of challenges, both experimental and theoretical.

Advances in experimental techniques now allow neuroscientists to probe the brain in very extensive manners, for example by collecting simultaneous recordings of tens of thousands of neurons in awake animals (199) or by reconstructing large-scale connectomes as mentioned above. Such large datasets reflect the intrinsic complexity of neural circuits, and they sometimes even challenge earlier beliefs about the functional organization of the brain (196). They hence require the development of new analysis tools, as well as new theories able to channel this complexity into simpler ideas.

A particularly recurrent observation in neural recordings is that cells exhibit certain collective forms of correlated activity, known as *low-dimensional* activity patterns. These collective patterns can be linked to many aspects of behavior, and seem to reflect fundamental constraints on the structure of neural networks in the brain. Interestingly, similar patterns are found in artificial neural networks, a broad class of models designed partly to loosely imitate biological networks and partly as powerful artificial intelligence algorithms. These low-dimensional patterns could thus form a basic organizational principle of computations in complex networks.

In this work, we will explore several aspects of this idea, and will be particularly interested in developing tools to discover low-dimensional activity patterns, and relate them to the connectivity structure of neural networks. We will rely on several fundamental concepts which are introduced hereafter.

[1] "Science replaces visible complexity by invisible simplicity". Jean Perrin was a French physicist, former student at ENS and professor at Paris university, recipient of the 1926 Nobel prize for his experiments showing the reality of atoms.

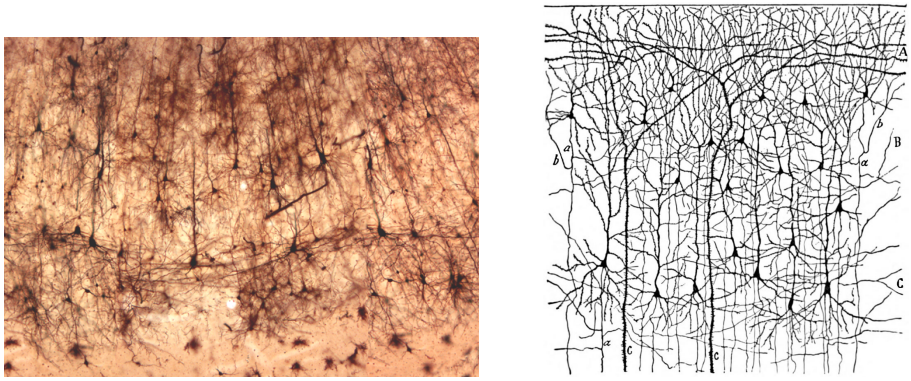

Figure I.1: Left: Golgi stained neurons from a monkey neocortex (reproduced from ref. (185), credited to Kathy Rowland). Right: one of Ramón y Cajal's numerous drawings of intricate brain connectivity, here representing cells of the visual cortex of an infant (Fig. 4 of *Comparative Study of the Sensory Areas of the Human Cortex* (1899))

## I.2 From neurons to networks

At the dawn of the twentieth century, the field of neuroscience was born in the midst of a heated controversy between histologists Camillo Golgi and Santiago Ramón y Cajal (70). The first one, observing brain slices with a stain of his invention, revealed a beautifully intricate web-like structure (Fig. I.1a), which we now know to be composed of the axons and dendrites connecting our neurons. Limited by the capacities of optical microscopes, he could not observe the very narrow gaps separating neurons at synapses, and, like many of his peers, believed this web was a single giant entity, called the "reticulum". At about the same time, a Spanish researcher interested in studying brain anatomy started using Golgi's stain, carefully recording his observations in drawings that were to become famous (Fig. I.1b). His extensive work led him to maintain that the brain was not composed of a giant reticular web, but of disjoint cells, termed *neurons*. This proposal relied on the hypothesis that neurons were separated by very tight interstices, a remarkably bold hypothesis since it could not be verified until the advent of electron microscopy, half a century later. Despite their disagreements, the two scientists shared the 1906 physiology Nobel prize for their contributions, and Ramón y Cajal's *neuron doctrine* quickly started to prevail in the nascent community of neuroscientists.

The logical next step within this paradigm was to study the physiology of individual neurons. A general understanding was achieved - through the works notably of Louis Lapicque (107; 22), Charles Sherrington (188) and Edgar Adrian (2) among others - and can be outlined as follows:

- each neuron communicates to its downstream neighbors by sending *action potentials*, also known as *spikes*, small electrical impulses that can swiftly travel along its axon.
- Each neuron's activity can be represented by the sequence of spikes it generates, a sequence called the *spike train*. More specifically, and although this remains a debated point (213), we will consider in this work that the information encoded by a neuron is contained in the frequency at which it generates spikes at a particular moment, which we term its *firing rate*.
- This firing rate is transmitted through synapses to downstream neurons, being multiplied in the process by a synaptic weight, which will be positive at excitatory synapses, negative at inhibitory ones. The resulting number will be a contribution to the *input current* of the downstream neuron.

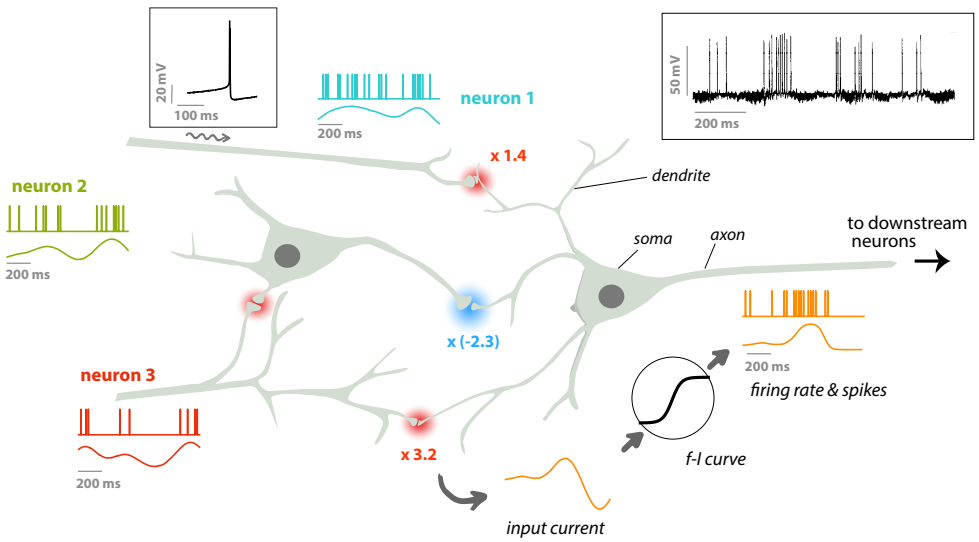


FIGURE I.2: Fundamentals of neural signalling (see text for explanations). For each neuron represented here, a simulated spike train and the associated smoothed firing rate are represented. Red and blue shadings represent respectively excitatory and inhibitory synapses. Top left inset represents a single action potential recorded intracellularly (from a cultured rat cell) and top right inset a real-life example of a spike train (recorded from an electric fish Eigenmannia, both insets reproduced from ref. (39))

- the input current will finally be transformed into the firing rate by a non-linear mapping (39) (known as the f-I, or frequency-input, curve). Spikes are thus generated randomly at the resulting rate.

It is this simplified model of the neuron stripped to its most fundamental principles that we will consider in this work, and we will see that despite the absence of numerous other biological processes, it remains capable of generating many interesting and complex behaviors.

Once the fundamental properties of individual neurons had been revealed, neuroscientists aimed at understanding how they contributed to the different aspects of behavior and cognition, and how they exchanged information in the circuits they formed. In this endeavor, the neuron doctrine also shaped many of the initial studies, leading to a number of foundational discoveries. One of the main approaches was that of the *tuning curve*, whereby researchers recorded the electrical activity of an individual cell while submitting an animal to diverse environmental stimulations, reporting the average firing rate obtained for each stimulus (Fig. I.3a). Following this methodology, David Hubel and Torsten Wiesel famously discovered cells in the visual cortex of cats that were responsive to specific orientations of visual bars, and could form the first steps of a computational pathway aimed at recognizing all sorts of shapes (84) (Fig. I.3b). Indeed, neurons that were receptive to particular aspects of visual stimuli have been found throughout the hierarchy of visual cortices, leading up to the discovery of neurons that were selective to particular faces in the IT cortex (209). In a similar feat, O'Keefe and Dostrovsky discovered place cells, neurons in the hippocampus of a rat that were active only when the animal was in a particular location (133) (Fig. I.3c). This discovery started to unveil the abilities of brains to represent space, leading to the later description of grid cells in the entorhinal cortex[2] (62; 128) (Fig. I.3c). The individual cell

[2]Hubel and Wiesel obtained the 1981 physiology and medicine Nobel prize for their discoveries about

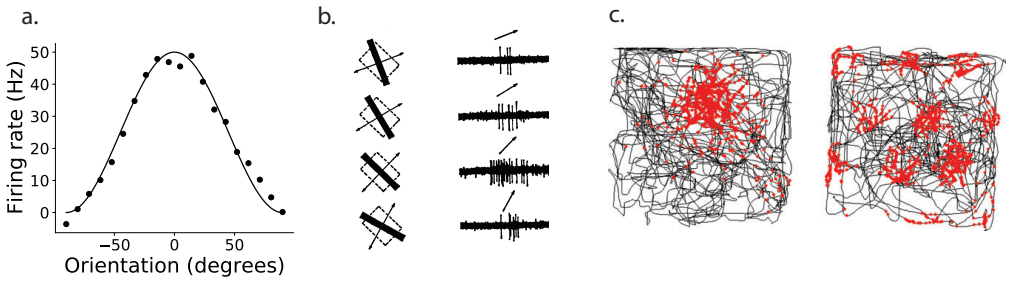


Figure I.3: a. Example of a tuning curve: firing rate of a cell in response to visual stimuli of different orientations (simulated data, inspired on ref. (39)). b. Spike train responses of a monkey striate cortex cell in response to visual stimuli of different orientations (reproduced from ref. (85)). c. Firing events (red dots) for a hippocampal place cell (left) and a medial enthorinal cortex grid cell (right), overlaid on the trajectory (black line) of an animal in a box (reproduced from ref. (128)).

approach led to advances not only in the study of perception, but also in that of action, as emphasized by the discovery of neurons that drove particular behaviors when stimulated (20).

Between perception and action lie a series of complex executive functions, sometimes termed *higher-order functions*, which control our behavior in a more general sense, rendering us able to integrate several sources of information, to recall the past, and to plan ahead. These include for example deliberation, decision-making, working memory, context-dependent computations, abstraction, and many more. Several brain areas have been linked to these functions like the prefrontal cortex for mammals, and single-neuron studies in these areas have been able to relate neural firing rates to cognitive variables like an upcoming decision (94; 186; 71) or a memorized information (61; 100; 72). Let us consider an example in the field of decision-making: in 1999, Jong-Nam Kim and Michael Shadlen submitted macaques to random dots kinetograms (94), a class of visual stimuli composed of points moving randomly, albeit with an average motion leaning towards the left or towards the right (the coherence, Fig. I.4a). The monkey had to decide which way this average was pointing to and perform an eye saccade towards a target in this direction. The researchers recorded single neurons in the prefrontal cortex (PFC) of the monkeys while they were performing the task, finding cells that encoded the perceived direction during stimulation and delay, and that could hence be the support of this decision-making process (Fig I.4b). A broad corpus of experimental and theoretical studies supported these observations, and proposed models of competing cell populations that could integrate the available evidence and encode a decision during a delay in order to perform such a task (186; 71; 220; 221).

This line of investigation quickly reached its limits however, and as larger recordings were performed on more complex tasks, it became apparent that a vast majority of neurons in these areas were *mixed selective*, meaning they encoded several variables at the same time (113; 155; 60). This is apparent notably in multimodal tasks, in which the brain has to mix different streams of information. A compelling example is provided by the context-dependent decision-making task tested by Valerio Mante and colleagues in 2013 (118), in which macaques were submitted to random dots kinetograms similar to those used by Kim and Shadlen, but that also varied along a color dimension, with a certain percentage of dots being red and the others green (Fig. I.4c). A context cue at the beginning of each

cell tuning in the visual system. John O'Keefe shared the 2014 Nobel prize with May-Britt and Edvard Moser for their respective discoveries of place cells and grid cells.

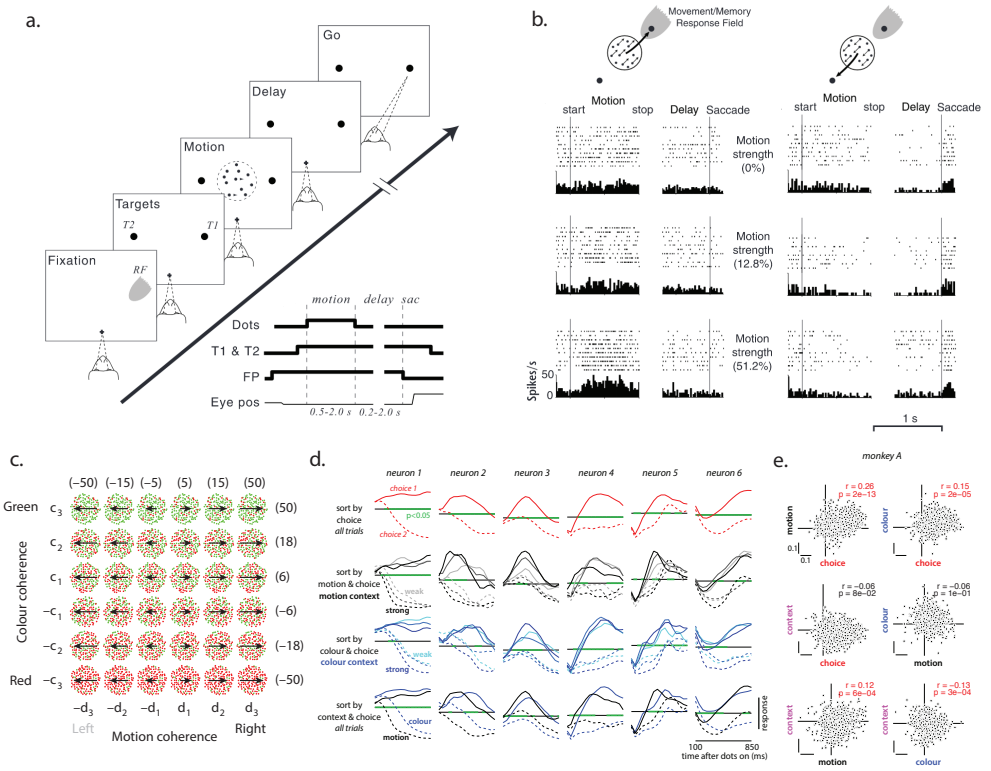


FIGURE I.4: a. Task design of reference (94): along different epochs of a trial, monkeys had to fixate a central point, wait for the two decision targets to appear, observe the moving dots stimulus, keep fixating the center during a delay and finally communicate its decision by making an eye saccade to the target towards which most dots were moving. b. Activity of an example neuron recorded in the prefrontal cortex of a macaque during the task. A decision towards the receptive field of this neuron elicited higher firing rates than towards the other direction, and this effect was stronger for higher motion coherences. c. Set of stimulations used in reference (118), varying in average dot motion (x-axis) and average color (y-axis). d. Time-dependent responses of 6 neurons recorded in this study, across different task conditions. Green overlay on the x-axis represents significant selectivity towards the represented conditions. e. Selectivity of individual cells recorded in this study with respect to the four task variables (motion, color, choice and context), visualized in all 6 possible pairwise point clouds. Correlation and the corresponding p-value are indicated for each point cloud.

trial indicated the animal whether to pay attention to the motion or the color of the dots, and it had to integrate only the cued information to form its decision while ignoring the other. Electrophysiological recordings in the prefrontal cortex of tested monkeys showed that the cells responses were complex and variable, certain cells seemingly encoding combinations of variables or changing their selectivity depending on context (Fig. I.4d). Even more surprisingly, visualizing the selectivities of individual cells to different variables showed that all possible combinations of selectivities were represented in the recorded neurons (Fig. I.4e), ruling out the hypothesis of specialized cells that each represent a single cognitive variable.

Understanding these complex and mixed representations required a fundamental change of paradigm, and the last twenty years have seen neuroscientists gradually move from the

single-neuron doctrine towards the *population approach*, where representations are assumed to be represented across large populations of neurons (233; 66; 64; 52). In this view, the fundamental computational and representational unit of the brain is not the individual neural cell but a set of anatomically and functionally close cells that we can term a *circuit* or *network*, and it would be meaningless to study separately each neuron of such a circuit. Representations and computations would *emerge* spontaneously from the complex and large-scale interactions that occur in such a circuit, in a similar fashion as the construction of a hive emerges from the interactions between a large population of bees[3]. It is particularly challenging to define precisely at what scale such circuits should be defined, as they could range from the cortical column to the whole brain, which is, as we mentioned, a fully interconnected organ. Fortunately for us, the brain can be decomposed in clearly separable and seemingly specialized areas[4], and we will generally mean by "circuit" such a small area, or even a small patch of one or a few millimeters in side on the surface of the cortex.

These theoretical transformations have been paralleled by the development of new experimental techniques that enabled researchers to track and record the activity of large numbers of neurons simultaneously in behaving animals. Among those we can mention the development of multi-electrode arrays which can record on the order of hundreds of cells in a localized patch, and of calcium imaging which can record thousands of cells simultaneously, or even the activity of the entire nervous system for small organisms like the zebrafish. These new techniques change the nature of data that is obtained from experiments: in the population view, the response to a environmental stimulation is not characterized by a single time-varying neural firing rate $r(t)$, but by the joint firing rates of the set of recorded neurons, which can be assembled into a vector $\boldsymbol{r}(t) = (r_1(t), \dots, r_N(t))$ if there are $N$ neurons. This vector can then be seen as an element of an $N$-dimensional abstract space, which we will call the *neural state-space* or *activity space*, in which each axis represents a single neuron. For a given stimulus the time-dependent activity of the network draws a continuous trajectory (curved line) in this space (Fig. I.5b). Of course as soon as more than 3 neurons are studied, the state-space cannot be visualized or apprehended in any intuitive way, but neuroscientists can now use the powerful tools of high-dimensional geometry and statistics to reach conclusions at the population level. In particular, 2 families of statistical methods are widely used in the analysis of high-dimensional data and will prove to be relevant for neuroscience:

- **dimensionality reduction algorithms**: these methods aim at finding low-dimensional subsets of a high-dimensional space that explain most of the variability of the studied data. These methods can find linear subsets (mathematically called *subspaces*) of the original space, or non-linear, curved ones (called *manifolds*) (36; 87). These can then be used for visualization purposes or to better understand the structure of data. The application of these techniques to neural recordings has been a very fruitful area of research, as it has been found that numerous recordings of hundreds or even thousands of cells could be well summarized by only a few dimensions (66) (yet, see (198) or (106) for more nuanced accounts) . We will develop further these ideas, crucial in this work, in the section I.4 of the introduction. For the moment, let us illustrate this idea with the example of the context-dependent decision-making task mentioned earlier (118). As we explained, individual neurons seem to encode mixtures of the diverse variables associated to the task (the average motion, average color, context, and choice). However, a targeted dimensionality reduction algorithm can identify one axis of the neural state-space corresponding to each of these variables, and projecting the recorded population vectors on these axes reveals how information is encoded

[3] The concept of emergence, one of the main ideas in the study of complex systems, is well captured by the aphorism: "The whole is more than the sum of its parts.", attributed to Aristotle.

[4] Although recent brain-wide recordings have started to shake these established views (196; 199)

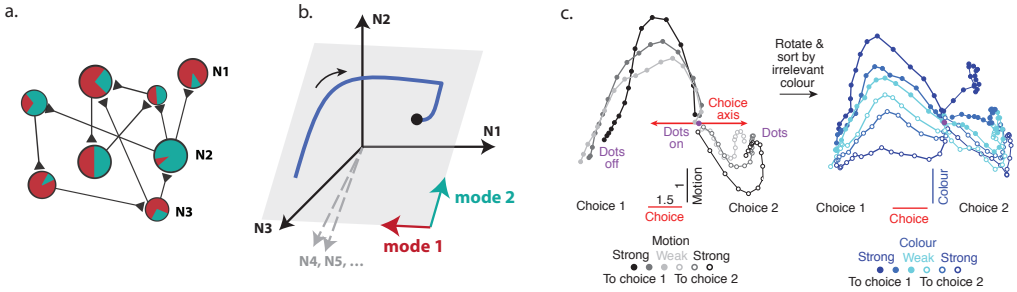


FIGURE I.5: a. Schematic of a network of a few neurons, all exhibiting mixed selectivity to two cognitive variables, a red and a turquoise one, to different degrees. b. The joint time-dependent activity of those neurons can be conceptualized as a trajectory in a high-dimensional *state-space*, whose axes correspond to the activity of each neuron of the network. Here, only the projection on the first three axes can be visualized, other axes being left to the reader's imagination. A dimensionality reduction algorithm extracts a low-dimensional *subspace* (grey plane) of this state-space on which the trajectory (blue line) lies. The axes of this subspace correspond to the *latent variables* or *neural modes* encoded by the network, and each neuron contributes to diverse degrees to each mode. c. Dimensionality reduction applied to the context-dependent decision-making task of Mante and colleagues (118). Left: activity in the motion context projected on the choice and motion axes identified by the targeted dimensionality reduction algorithm. Each curve corresponds to the average over trials that presented each level of motion coherence. Trajectories start at the center, at stimulus onset. One can notice how network activity is first driven along the motion-encoding axis before turning onto the choice axis. Right: Same trials, this time averaged over shared color coherence and choice and projected onto the choice and color axes. This projection shows that although it is irrelevant to the choice, the sensory evidence for color remains encoded in another axis of the state-space.

and transformed in this network: for example, projecting the activity on the motion and choice axes, during motion-context trials reveals that the population starts by encoding the perceived motion as sensory information arrives to the prefrontal cortex, and then gradually starts forming a choice, as manifested by the bend of collective activity along the choice axis. Projecting population activity on the same axes during color-context trials reveals how the sensory information about the motion of dots still arrives to the prefrontal cortex but does not contribute to the choice as it is correctly ignored in this context. In summary, in this view, variables are encoded in a distributed manner across the neural populations, and materialized by *axes*, also called *modes* of the neural state-space. The role of individual neurons becomes diluted in the collective behavior, and neural computations are instead supported by these shared, *latent* variables, which are entirely abstracted away from their cellular substrate.

- **clustering methods**: it is also natural when studying the activity of a large population of neurons to ask if the studied cells are organized in specialized groups (subpopulations). Neurons can be grouped by their response profiles to stimuli, for example by applying an automatic clustering procedure to neural recordings, but they can also be grouped by their physiological or genetic properties. Indeed, there is a wide heterogeneity of cells in the brain, which vary in terms of the neurotransmitters they emit[5], the connexions they can form[6],

[5]In particular, Dale's law, which implies that a neuron can form only excitatory or only inhibitory synapses with its downstream neurons, but not both, remains a fundamental organizational principle of the brain (39).

[6]Here, we can notably distinguish between interneurons, which form connexions only at a very local

their cortical layer, their transcriptome and other aspects.

These are, in a very general sense, two of the main tools available to extract insights from high-dimensional data. They also correspond to two deep organizational views of brain circuits (6), a purely geometrical view on the one hand, in which information is multiplexed randomly across neurons and encoded at an abstract collective level, and a more cell-driven view, in which circuits are divided in groups of cells with specific, well-defined, roles. These two views appear particularly difficult to reconcile, and a major focus of this work will be to show how both are essential to explain different aspects of neural computations.

Hence, more than one century after the neuron doctrine was finally recognized by the neuroscientific community, the functional relevance of single neurons is more questioned than ever, and the importance of considering neurons as part of an indivisible network that bears the burden of computation is becoming clearer. The development of powerful brain-inspired computer algorithms that we will expose in the following section further strengthens this case.

## I.3 Computations in brains and machines

The physiological processes of neurons described in the previous chapter can be summarized in a rather simple mathematical model, and implemented in algorithms that could then imitate the computational abilities of biological brains. Since the foundational work of Warren McCulloch and Walter Pitts in 1943 (124), this insight has proved to be particularly fruitful, leading to the development of powerful artificial intelligence algorithms but also reshaping our understanding of neural computations.

*Artificial neural networks* (ANNs) exist in a variety of forms, but to fix ideas we will formalize them as follows: they consist of *units* akin to the biological neurons, that receive several input signals $u_1(t), \dots, u_P(t)$, either external or from other neurons, each multiplied by a synaptic weight $w_i$. The sum of those weighted signals forms the *input current* of the unit, noted $x(t)$, and is then passed through a non-linear *activation function* $\phi(\cdot)$ to produce a firing rate $r(t) = \phi(x(t))$. This model thus assumes that neurons only use firing rates to compute, and not the specific timing of spikes, and that the numerous other biological details of neural tissue are not important to neural computations. Although the question of which aspects of neural biology are relevant for computations is far from settled, and although it is very likely that other processes play a crucial role there, this very simple model is at least *theoretically sufficient* to implement any neural computation imaginable, in a way that we will detail below.

An important aspect of these artificial networks is that their units are all identical, and hence they can differ only in the way neurons are wired, that is in their architecture (which units are connected to which), and in the strengths of their connexions (materialized by the synaptic weights). In general, the architecture of a network is specified by the modeller in order to achieve particular functionality, and the development of new, more or less brain-like, network architectures has been an important factor of progress in this field. We can mention for example the single-layer perceptron, developed by Frank Rosenblatt in 1958 (158), which was the first trainable ANN ; the feedforward, or deep neural network (DNN) in which several layers of neurons are stacked, each neuron receiving its inputs only from neurons in the previous layer and transmitting its output only to the next layer ; the convolutional neural network (CNN), a particular form of feedforward architecture in which the neurons

level, and are usually inhibitory, and pyramidal cells, whose axons can travel very long distances, and are usually excitatory.

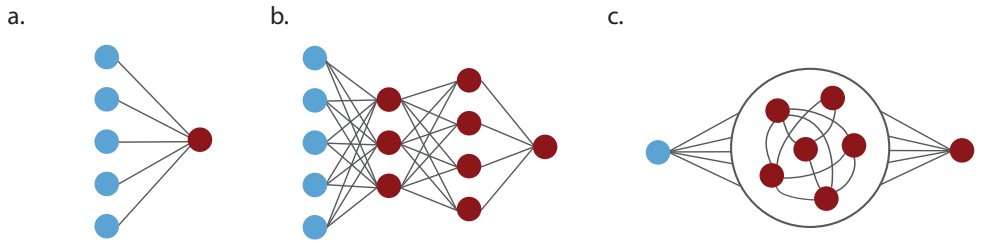


FIGURE I.6: Different flavours of artificial neural networks. Red circles represent internal and output units (characterized by a summation and a non-linear activation function), blue circles represent input signal units. a. A single-layer perceptron (158), formally equivalent to a single unit. b. A multi-layer feedforward network, here with two hidden layers and an output unit. c. A recurrent neural network (RNN), with a single input signal and an output unit.

are connected in order to perform convolutions on their inputs, which proves particularly relevant for applications on images (58) ; the recurrent neural network (RNN), in which all neurons of the network can be connected to each other, potentially forming cycles such that computations unwind through time (Fig. I.6) ; or the more recent Transformer architecture, based on artificial attention mechanisms and particularly suited for language processing (214).

Once a network is wired following a particular architecture, its synaptic weights have to be set to precise values for it to perform a desired computation. The process of modifying the weights in order to obtain a specific behavior is called *training* and can be performed using diverse techniques. The first training algorithm proposed for ANNs was the perceptron algorithm, also included in Rosenblatt's 1958 work (158), and was based on a simple converging recurrence. Although mathematically tractable, it could only be applied to the single-layer perceptron architecture, which was limited to solving linearly separable decision problems[7]. Ideas for algorithms that could train networks of any architecture have been proposed throughout the following fifty years, revolving around classical non-convex optimization methods like gradient descent or second-order optimization, and adapted to feedforward networks with the backpropagation algorithm (162), and to recurrent networks with the backpropagation-through-time extension (223). Scale and implementation challenges have made their use unpractical for most applications until very recent years, but they are now a cornerstone of artificial intelligence's development and of its numerous feats. Finally, a family of algorithms based on the recursive least-squares method has been proposed to train recurrent neural networks, notoriously ill-behaved with respect to gradient-based methods[8] (16; 38), leading to the development of reservoir computing (112; 86) and the FORCE (204) and full-FORCE algorithms (40). ANNs can now be trained to perform tasks that humans could never dream of programming by hand, whether it is solving the tridimensional structure of a protein (208; 210), planning and choosing strategies in simulated game-like environments (190; 215; 179), interacting with humans through natural language (21) and many more.

[7]This observation famously led Marvin Minsky and Seymour Papert to question the relevance of artificial neural networks in their 1969 book *Perceptrons* (125), and in general led the artificial intelligence research community to delve most often into symbolic, non-neurally-inspired approaches for the remainder of the twentieth century

[8]Although recent improvements of gradient-based methods for recurrent networks have made those very practical, and we will use them throughout this thesis.

Yet, although the process of wiring and training networks is now mastered for a wide range of applications, a theoretical understanding of how and under which conditions sophisticated behavior emerges from a network, and what its capabilities and limits are, remains elusive. To computer scientists and theoretical neuroscientists alike, ANNs remain mysterious *black boxes*, to which large "bags of tricks" found by trial-and-error can be applied in order to enhance their function, but whose internal processes are unknown, and which can thus exhibit unpredictable behaviors. The number of cases in which ANNs behave in ways unexpected by our current theory is growing, ranging from adversarial attacks (207) to double descent (13; 3) and lottery tickets (56) phenomenons. As ANNs are taking a leading role in many critical aspects of our societies, the question of their interpretability is becoming unavoidable (134). We hope that this work, at the crossroads of neuroscience and computer science research, will also help in the efforts to lift the veil on the mysteries of artificial neural networks.

Indeed, if we mention this subject so extensively, it is not only because ANNs are inspired from early neuroscientific research, but because their relation to neuroscience has considerably deepened in recent years (9; 154; 229; 171; 228). As we mentioned, ANNs differ in numerous ways from biological neural circuits, notably because the algorithms used to train them are not inspired on biological processes, and are unlikely to be implemented in real brains[9]. Instead, biological circuits likely rely on variants of Hebbian learning[10], which, at the moment of writing, and despite continued advances, are only capable of learning the simplest tasks in an artificial setting, far from the performances of the other algorithms[11] (14). This observation alone justifies that ANNs can not today claim to be accurate models of brain circuitry. Despite that fact, they can illuminate many of their mysteries.

Indeed, a remarkable property of artificial neural networks is their *universality*: it has been known since the 1980s that feedforward networks with at least two layers of neurons could approximate with arbitrary precision any continuous function with finite support, simply by adding enough neurons and setting the right synaptic weights (37). It has later been shown that, just as feedforward networks were universal approximators of functions, recurrent networks were universal approximators of dynamical systems (41). Since any input-output mapping can be formalized as a function, and any computation (in the Turing-Church sense) corresponds to a dynamical system, this means that ANNs are universal computing devices, and can capture any functionality that the brain is implementing. Hence, the first insight offered by artificial neural networks to neuroscience is the following: even if they capture only a few physiological processes, these are *theoretically sufficient* to explain how the brain can compute any function. Moreover, due to their structure, they justify the connectionist tradition, that postulates that intelligent behavior can arise from the interactions of many simple elements.

Inspired by these observations, neuroscientists have started to compare patterns of activity recorded in animal brains performing certain tasks and in ANNs performing similar functions finding startling parallelisms. For example, ANNs trained to perform image recognition have been shown to exhibit similar neural responses as those found throughout the visual processing pathways of primates (226) (Fig. I.7b). Similar observations have been raised about auditory processing (91), or showed the emergence of grid cells in networks trained

[9] Although some proposals for biologically-plausible implementations of backpropagation have been put forward (108; 164; 109).

[10] Summarized by what is probably the most famous punchline in the neuroscientific litterature: "Neurons that fire together, wire together" (78).

[11] A noteworthy reason for these difficulties is that ANNs are usually trained to optimize very well specified objective functions, whether they consist of reproducing a pre-defined input-output mapping contained in a curated dataset, or of enhancing their performance at tasks in limited environments where rewards are easy to identify. What general objectives a biological being could optimize throughout its lifetime, or whether such objectives exist, remain vast open questions.

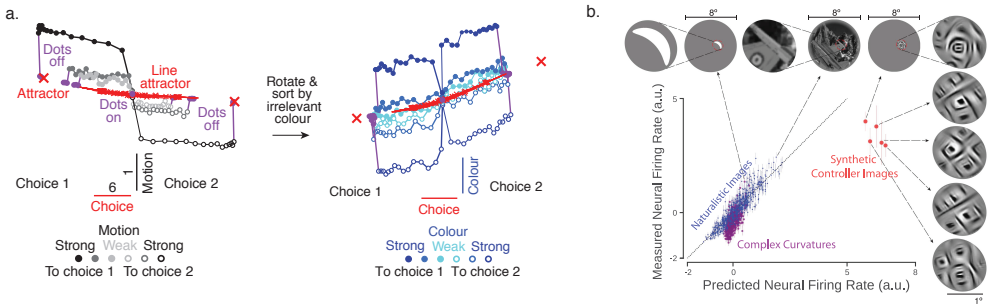


FIGURE I.7: Comparisons of activity in brains and artificial neural networks. a. Dimensionality reduction algorithm applied to the neural state space of an artificial RNN trained on a context-dependent decision making task, similar to that performed by macaques and illustrated in figs. I.4, I.5, from the work (118). The same trials as in Fig. I.5c are illustrated, on the same axes. Note that the similarity emerged spontaneously when requiring the network to perform the task, since it was not required to approximate neural data. b. Comparison of neural rates measured in a primate visual cortex (V4) neuron in response to different types of images (naturalistic, synthetic), and of the firing rates predicted from the activity of a deep feedforward network trained to do image recognition. Here, the activity of the ANN was even used to generate "synthetic controller images" that were predicted to drive even more strongly the recorded neuron, and indeed appeared to do so in the experiment (red dots). Note that the ANN was again trained only to perform image classification and was not required to reproduce recorded neural activity. Reproduced from ref. (10).

to perform spatial navigation tasks (35). Recurrent networks trained to generate motor primitives exhibit activity patterns that are found in motor cortices of animals (206; 163) and those trained on higher-order cognitive tasks exhibit similarities with prefrontal cortex recordings (118; 27; 227; 153; 219). In their foundational study, Sussillo and Mante trained an RNN to perform a context-dependent decision-making task similar in principle to that performed by macaques (118). They discovered that neurons in the artificial network also exhibited mixed selectivity to task variables, and after applying the same dimensionality reduction algorithm they used on neural recordings to the activity of their artificial neurons, observed strikingly similar low-dimensional trajectories (Fig. I.7a).

The interpretation of these results remains a delicate question, but they mostly provide a remarkable opportunity for neuroscience by providing *in silico* models of observations similar to biological ones. An outstanding question in neuroscience is that of relating network structures to function, and although experimental techniques allow large-scale recordings and tracing the connexions between neurons in the simplest organisms, access to the specific synaptic weights connecting neurons on a large scale in an animal remains out of reach. Artificial network models of neural computations are hence a practical simplified model for tackling this question, by providing unlimited access to the activity and connectivity and allowing us to test numerous hypotheses on the links between structure and function, which can afterwards be verified in biological organisms through more specific experimental procedures like optogenetic perturbations. Moreover, one could argue that understanding ANNs constitutes an intermediate challenge for neural theorists in the tortuous path to a theory of complex brains, helping them discover which tools and frameworks are most likely to be useful in this endeavor.

## I.4 Neural dynamics and the state-space approach

As we mentioned in section I.2 of this introduction, a dominant approach in modern neuroscience considers the simultaneous activity of the multiple neurons in a circuit as the basis for neural computations. The instantaneous activity in a neural circuit - or an artificial RNN - is formalized as a vector $\boldsymbol{r}(t)$ of its $N$-dimensional state-space, and the trajectories shaped through time by this vector inform us on the computations performed by the circuit. This raises the question of how these task-related trajectories arise from network circuitry.

A first part of the answer lies in the observation that a neural circuit or an RNN is formally a *dynamical system*, that is a mathematical object whose evolution in time depends on its current state and inputs (217; 23; 203; 216). In general, a dynamical system can be described by a differential equation[12], of the form:

$$\frac{d\boldsymbol{r}(t)}{dt} = F(\boldsymbol{r}(t), \boldsymbol{u}(t)) \tag{I.1}$$

where $\boldsymbol{u}(t)$ is a vector of the external input signals fed to the circuit, and $F$ a (potentially complex) function that describes time evolutions within the circuit. Such a description falls into the deterministic tradition of classical physical sciences, in which knowing the state of a system at a certain instant and all external influences on it allows to predict the future states of this system. Although this equation is deterministic, it is fairly straightforward to add sources of randomness to it, like noisy fluctuations to the input currents of neurons, rendering it able to reproduce trial-to-trial variability observed in neural recordings. We will be interested in the characteristics of this noise in the fourth chapter of this thesis, but for the most part will consider that, for any given condition, random fluctuations in our circuit models will merely add some variability to trajectories around a reliable *trial-averaged* trajectory that supports the underlying computations and will be the focus of our efforts.

Within this framework, the state-space can be viewed as a dynamical landscape where for a given input, each point is associated to an arrow pointing in the direction in which the state would evolve from there (a visualization which we formally call a *phase portrait*, see Fig. I.8). This landscape is marked by landmark features which can be tied to the network's function (200): for example certain points of the state-space might be associated to a null arrow, meaning once the network state reaches this point, it does not move anymore (unless an external input perturbs it). Such a point is called a *fixed point* of the dynamics (Fig. I.8a), and can serve, for example, as a substrate for memory, as proposed in John Hopfield's foundational work (83). Sometimes, one can find continuous families of such fixed points, then termed *manifold attractors*, that can take the shape of lines (184; 118), rings, or more complex structures. One such structure can be observed as the lines of red crosses illustrated in Fig. I.7a, where Mante and Sussillo show how *line attractors* can implement an analog memory of the integrated evidence in a decision-making task. Another example can be seen from *ring attractors* (Fig. I.8c), circular collections of fixed points that have been found within circuits of cells that track the orientation of an animal's body (head-direction cells) (96; 29). Even the grid cells, remarkable discovery of the single-cell approach, can be seen dynamically as circuits that collectively encode torus-shaped attractors (68) (Fig. I.8d). Other illuminating features of those dynamical landscapes include *limit cycles* (Fig. I.8b), which can form a substrate for oscillatory activity (206), or the more elusive chaotic attractors, which can explain the seemingly random and erratic patterns of spontaneous brain activity (194).

[12] or its discrete-time equivalent, the recurrence equation

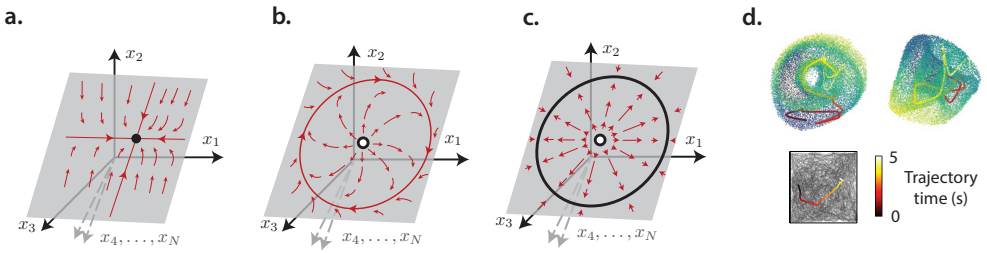


FIGURE I.8: Some examples of neural dynamical systems. a-c. Some phase portraits on a low-dimensional subspace of the neural state-space. In general, all points of the phase space would be associated to a high-dimensional arrow in the phase portrait, but in many cases low-dimensional dynamics will dominate (for example for low-rank networks, see chapter II). a. A phase portrait exhibiting a single attractive fixed point (in black). b. A limit cycle, notice how the center of the cycle is a fixed point, albeit unstable. c. A ring attractor. All the points on the black circle are stable fixed points of the dynamics. d. A torus-shaped collection of dynamical fixed points found in the state-space of a rat's medial enthorinal cortex, viewed from two angles (after a non-linear dimensionality reduction), with a spatial trajectory of the rat overlaid as the colored line on the torus, and on a map of the experiment box (bottom). Every point of the torus is dynamically stable if the rat doesn't move, but as it wanders through space, inputs induce a change of position on the torus, which can hence encode the coordinates of the rat's position. (Adapted from ref. (68))

The dynamical systems approach used in conjuction with dimensionality reduction provides a rich set of tools to dissect neural computations, in particular in artificial RNNs for which the arrows at every point of the state-space can be mathematically computed, which gives access to the complete set of dynamical fixed points, cycles, *etc.* In a pioneering work in 2013, Omri Barak and David Sussillo have for example studied the dynamics in RNNs trained to perform working memory or rhythmic tasks, showing the deep links between the computations and the underlying dynamical landscapes (205). Moreover, numerous research projects have exploited the similarities between trajectories in neural recordings and RNNs by training RNNs on tasks similar to those performed by animals, and inferring dynamical mechanisms from the more manipulable RNNs (118; 206; 27; 219; 153; 193; 127; 228). Pushing forward this paradigm, some studies have even proposed to train RNNs to directly reproduce the recorded neural trajectories, hence being sure to obtain networks whose activity matches biological one (150; 33; 54; 143), a strategy we should review in more detail in chapter V. And without necessarily resorting to artificial RNNs, a broad effort is being undertaken by neuroscientists to develop new algorithms that discover low-dimensional dynamics in neural data and are able to infer dynamical systems that could reproduce them (231; 114; 144; 135; 47; 132; 110; 136; 44; 69; 63; 168; 169; 177; 105; 46), some of which, described at an abstract statistical level, we will attempt to link back to neural connectivity structure in the chapter IV of this thesis.

Yet this powerful toolset does not provide a complete understanding of the emergence of neural computations. In particular, an essential remaining question is how the network structure determines the activity dynamics. It is notably apparent that biological brains are organized following very numerous architectural constraints, for example the separation between excitatory and inhibitory cells (Dale's law), separation in cortical layers, specific patterns of inter-area projections and many more principles that have been shaped by millions of years of evolutionary history. In stark contrast, artificial RNNs are often trained

from a purely random and homogeneous blank state. Are then some of these organizational principles needed from a purely computational point of view, to achieve certain functions? Are they reflected in neural state-space dynamics? And does a similar kind of structure emerge in RNNs during their training?

A field that is accustomed to predict dynamics from the structure of a system of many interacting elements is that of statistical physics, whose tools have been profitably applied to neuroscience. Thanks to the use of mean-field theory and energy-based methods, physicists working at the intersection with neuroscience have characterized network structures that lead to chaotic dynamics (194), to multistable attractor landscapes for encoding memories (Hopfield networks) (83), or to attractors with ring-shaped (15; 236) or toroidal topology (24; 73). It is notable that the initial state from which artificial networks are trained leads to either trivial or chaotic dynamics, and yet that these chaotic dynamics can be channelled into useful computations through the addition of inputs (149) and feedback loops (86), a fact exploited by reservoir computing and the FORCE learning methods. The effect of these feedback loops suggest that low-rank modifications to the connectivity of a recurrent network could be sufficient to induce particular behaviors in that network. This insight was more directly evidenced by the work of Francesca Mastrogiuseppe and Srdjan Ostojic (122; 123; 202), which notably showed that networks whose connectivity consisted of a random and a low-rank part exhibited low-dimensional patterns of activity that could be directly predicted by examining the low-rank term of the connectivity alone. Low-rank matrices are simpler mathematical objects than general, full-rank matrices, being defined by less parameters and prone to useful decompositions. The fact that they capture essential aspects of neural dynamics holds the promise to untangle some of the intricate links between network structure and function, and this is the direction that this thesis will explore.

Finally, let us note that through the use of the neural state space, the dynamical systems approach treats all neuron as identical and interchangeable elements of a biological substrate for computations defined as an abstract space. This paradigm hence does not as of today contain the tools necessary to answer questions about network organization. These questions are however well addressed by more cell-driven models, which posit for example competing populations to support decision-making tasks (220), or circular patterns of connectivity that can give rise to ring attractor dynamics (15; 236). The fundamental difference between those models and artificial RNNs is that the former do not give rise to the *mixed selectivity* phenomenon that we described above, characteristic of the state-space approach. Viewed in the state-space, the dynamics of cell-driven models would be aligned to particular axes corresponding to the specific group of neurons encoding each relevant variable. This stands in contrast with the agnostic view of the state-space in which low-dimensional dynamics traverse random subspaces of the state-space, which naturally leads every neuron to encode random combinations of task variables. Some recent studies have aimed at closing the gap between those apparently irreconciliable paradigms, by measuring quantitatively the degree of randomness or organization of biological networks. In particular, in 2014, David Raposo, Matthew Kaufman and Anne Churchland have found that the functional organization of a posterior parietal cortex network was as random as expected by chance (152): in other words, the low-dimensional axes that encoded task variables in the neural state-space crossed it perfectly randomly. This could have closed the debate, but a more recent study with the same analyses by Junya Hirokawa, Alexander Vaughan and colleagues at Adam Kepecs' lab reached the opposite conclusion on neurons recorded in the rat orbitofrontal cortex during a reinforcement learning task (81). This contradiction brings to light the fact that the two approaches, the cell-agnostic, random state-space one, and the cell-driven one, may cover different aspects of neural computations, and may thus become apparent during different tasks or in different areas of the encephalon. This discrepancy will be the main focus of the third chapter of this thesis.

## I.5 Outline of the work

In this thesis, we will mostly be concerned in the two following questions: how does the connectivity structure of a recurrent network determine its dynamical properties, and hence its computational abilities? And can we determine the structural organization and predict the dynamical behavior of a neural circuit from samples of its responses? We will adress those questions by building upon the theory of low-rank recurrent neural networks outlined in Mastrogiuseppe & Ostojic 2018 (122) and Beiran et al. 2021 (11) (to which the author of this thesis has contributed). Most specifically, these references have characterized the dynamical behaviors of low-rank networks whose connectivity was defined by the modeller, following certain distributions. This led to a rather exhaustive characterization of the behaviors of these networks, and in particular to the demonstration that they were universal approximators of low-dimensional dynamical systems. We will now in this work explore the capacities of low-rank networks to answer the two questions raised above.

- In chapter II, we will first review the mathematical properties of low-rank RNNs, most of which are described in references (122) and (11), and give some intuitions to the reader on the characteristics of these networks, and how they can be used to reverse-engineer network solutions found by numerical optimization to cognitive tasks.
- In chapter III, we will use this framework to effectively reverse-engineer networks of minimal rank trained to perform a range of cognitive tasks. We will be interested in relating the connectivity found by the training algorithms to the low-dimensional dynamics that appear to solve the tasks. This will reveal the fact that an organization of the network in subpopulations is necessary for some, but not all, tasks, and will allow us to build some bridges between the neural state-space and the cell-driven paradigms.
- In chapter IV, we will start to approach the question of dynamical inference from neural data by relating an important class of statistical inference models used in this context, the linear latent dynamical systems and the Kalman filtering algorithm, to our class of low-rank recurrent neural networks. This will allow us to characterize how this purely statistical approach can be related to properties of the connectivity of the neural circuits studied, and to reveal the blind spots of each model class.
- Finally, in chapter V, we will put into practice the use of low-rank RNNs to infer dynamics as well as structural properties of neural circuits from their activity. We will notably be interested in two use-cases of our method: reverse-engineering recurrent neural networks that are full-rank (*i.e.* trained without constraints on their connectivity), hence making them more interpretable, and inferring low-dimensional dynamics from a dataset of prefrontal cortex recordings in nonhuman primates.

## Summary of Chapter 2

Low-dimensional dynamics appear to be a fundamental principle of neural computations, arising in numerous contexts both in biological nervous systems and in artificial recurrent neural networks. Investigating the dynamics in such systems reveals how dynamical features control their function, but the sheer complexity of the structure of these networks hinders efforts to interpret those dynamics as manifestations of the underlying connectivity. Here, we draw on the recently developed framework of low-rank recurrent neural networks to build a theory expliciting the dynamics expressed by some networks from statistical properties of their connectivity. In particular, we derive a mean-field theory of dynamics in low-rank networks whose connectivity parameters follow a mixture-of-Gaussians distribution, and show through diverse examples how this theory can be geometrically interpreted and give rise to dynamics that illuminate underlying computations. Finally, we explain how such constrained networks can be trained and analyzed.

This chapter is based on parts of the methods of the paper *The role of population structure in computations through neural dynamics*, Alexis Dubreuil*, Adrian Valente*, Manuel Beiran, Francesca Mastrogiuseppe, Srdjan Ostojic, *Nature Neuroscience*, 25, 783-794 (2022) (43) for its sections 2 and 5, and summarizes some results presented in the paper *Shaping dynamics with multiple populations in low-rank recurrent networks*, Manuel Beiran*, Alexis Dubreuil, Adrian Valente, Francesca Mastrogiuseppe, Srdjan Ostojic, *Neural Computation*, 6 (33), 1572-1615 (2021) (11) for its sections 3 and 4.

# Low-rank recurrent networks as a window on neural dynamics II

## II.1 Introduction

Neural computations, particularly those involving time-dependent or complex multimodal processes are often thought to arise from recurrent interactions between vast amounts of neurons (233; 23; 217). As a consequence, the recurrent neural network (RNN) model has become one of the cornerstones of computational neuroscience, proving able to tackle numerous cognitive tasks, from memory retrieval (83) to decision-making (220), through short-term memory (184; 15) and spatial navigation (24) or generation of motor instructions (206). Even sensory processes, traditionally thought to unfold in a bottom-up, feedforward fashion, appear to benefit from recurrent interactions (90; 93).

A traditional approach to model neural computations with recurrent networks involves carefully tuned recurrent connections, for example between well-defined groups of neurons (220; 221; 50) or with particular symmetries (83; 15; 184; 236; 24; 73), or even drawing on specific analytic solutions (151; 49). This strategy presents the advantage of leading to deep insights into the relationship between connectivity structure and computational outcomes, but requires new solutions to be worked out for every problem, and can face difficulties in modelling mixed selective patterns of activity in higher-order areas, in which neurons often exhibit overlapping or context-dependent roles.

On the other hand, recent developments in optimization algorithms and deep learning made it feasible to train general-purpose RNNs without *a priori* structure on any behavioral task (223; 112; 86; 204), a strategy which has been applied extensively throughout systems neuroscience (118; 206; 150; 7; 35; 227; 219; 193; 174; 54; 143; 33) and provided an invaluable tool to probe how goal-directed computations can arise from a web of complex connections between elementary neurons.

These learning algorithms are applied on an initial random connectivity substrate (sometimes termed a random *reservoir*), which by itself leads to well-known behaviors involving chaotic trajectories or non-normal transients (194; 65; 19). This initial random substrate interacts with learned inputs and feedback loops in order to channel unstructured dynamics towards a computational goal (149; 86; 204). Recent theoretical modelling suggests that connectivity structures expressed as low-rank matrices are sufficient to induce desired behaviors from a random reservoir (86; 204; 156; 122; 123; 180; 202), and even that these low-rank structures can emerge on top of random connectivity during typical learning processes (172; 181). These results open vast possibilities to understand how trained connectivity relates to neural dynamics and ultimately to behavior, which we will explore throughout

this thesis.

More specifically, we build upon a recent study by Mastrogiuseppe & Ostojic (122), in which authors showed that adding a low-rank connectivity structure above a random initial connectivity could induce task-oriented low-dimensional behavior in a recurrent network. Follow-up studies by Schuessler and colleagues (180; 181) characterized more precisely the interactions between the random and structured parts of the connectivity during and after learning, while Beiran and colleagues (11) sought to comprehensively describe the dynamical landscapes that could arise from low-rank connectivity alone, and showed how a statistical description of connectivity in terms of mixture-of-Gaussians distributions was sufficient to relate connectivity and dynamics. In this work, we will focus on this latter line of work, and apply the theoretical understanding of low-rank networks to mechanistically understand how goal-directed artificial RNNs can perform cognitive tasks (chapter III), and how they can be related to neural-data analysis methods (chapter IV) and applied to neural recordings (chapter V).

In this chapter, we review the theory developed in references (122) and (11), while adapting it to our use-cases of training RNNs constrained to be low-rank. This will allow us to develop two insights which will be crucial to the rest of the thesis: the relationship between a statistical description of the low-rank connectivity and the geometrical organization of neural dynamics; and a gain-modulation mechanism based on the non-linear transfer function at the individual neuron level, but able to flexibly adapt dynamics at the population level.

## II.2 Formalism

### II.2.1 Introducing low-rank recurrent neural networks

In this work, we consider recurrent neural networks (RNNs) of rate units as studied by Mastrogiuseppe & Ostojic (122). In this framework, a network is composed of $N$ units, each unit $i$ being characterized by an *activation* $x_i(t)$ which obeys to the ordinary differential equation:

$$\tau \frac{dx_i}{dt} = -x_i + \sum_{j=1}^{N} J_{ij}\phi\left(x_j\right) + \sum_{s=1}^{N_{in}} I_i^{(s)} u_s(t) + \eta_i(t) \tag{II.1}$$

where $\tau$ corresponds to a membrane time constant, $J_{ij}$ to the connection from unit $j$ to unit $i$, $\phi$ is a non-linear transfer function (that we will take to be $\phi = \tanh$, the hyperbolic tangent, unless stated otherwise), $\eta_i(t)$ is a stochastic process introducing noise into unit $i$, and the unit receives $N_{in}$ input signals noted $u_s(t)$ for $s \in \{1, \ldots, N_{in}\}$ through the weights $I_i^{(s)}$. The non-linearly transformed activation $\phi(x_i(t))$ will be referred to as the *firing rate* of each unit.

An output $z(t)$ can also be produced by a linear readout of firing rates through a set of weights $w_i$:

$$z(t) = \sum_{i=1}^{N} w_i \phi(x_i(t)) \tag{II.2}$$

Equations (II.1) and (II.2) can also be written in a vector form:

$$\tau \frac{d\boldsymbol{x}}{dt} = -\boldsymbol{x} + \mathbf{J}\phi(\boldsymbol{x}) + \boldsymbol{W}_{in}\mathbf{u}(t) + \boldsymbol{\eta}(t) \tag{II.3}$$

$$z(t) = \mathbf{w}^T \phi(\boldsymbol{x}) \tag{II.4}$$

where $\boldsymbol{x}(t) = \{x_i(t)\}_{i\in\{1,\ldots,N\}}$ is called the *state-space vector* of the network, $\mathbf{J} \in \mathbb{R}^{N\times N}$ is the connectivity matrix, $\mathbf{u}(t) \in \mathbf{R}^{N_{in}}$ is the input vector, $\boldsymbol{W}_{in} \in \mathbb{R}^{N\times N_{in}}$ are the input weights and $\phi$ is here applied as an element-wise function to all vector coordinates.

In (122), the connectivity is taken to be the sum of a random matrix and a low-rank structure:

$$J_{ij} = g\chi_{ij} + P_{ij} \tag{II.5}$$

where $\chi_{ij} \sim \mathcal{N}(0,1)$ and $\mathbf{P}$ is a matrix of rank $R \ll N$, and can thus be written as a sum of $R$ outer products:

$$P_{ij} = \frac{1}{N}\sum_{k=1}^{R} m_i^{(k)} n_j^{(k)} \tag{II.6}$$

It has been shown in (122) that if the quenched noise, scaled by $g$, is not too large, the network dynamics are essentially equivalent to those of the network induced by the low-rank component only. This enables us to focus on networks defined only by a low-rank connectivity, and such that:

$$J_{ij} = \frac{1}{N}\sum_{k=1}^{R} m_i^{(k)} n_j^{(k)} \tag{II.7}$$

or in vector form:

$$\mathbf{J} = \frac{1}{N}\left(\boldsymbol{n}^{(1)}{\boldsymbol{m}^{(1)}}^{\top} + \cdots + \boldsymbol{m}^{(R)}{\boldsymbol{n}^{(R)}}^{\top}\right) \tag{II.8}$$

where the the vectors $\boldsymbol{n}^{(r)}$ and $\boldsymbol{m}^{(r)}$ are of dimension $N$. Note that this decomposition is not unique, and we will consider a particular one by writing the singular value decomposition of the $\mathbf{J}$ matrix:

$$\mathbf{J} = \mathbf{U}\mathbf{S}\mathbf{V}^{\top} \tag{II.9}$$

where $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{N\times R}$ are matrices with orthonormal columns and $\mathbf{S} \in \mathbb{R}^{R\times R}$ is a diagonal matrix containing the singular values of $\mathbf{J}$ in descending order, $s_1, dots, s_R$. We will take $\boldsymbol{m}^{(r)} = \sqrt{s_r}\mathbf{U}_{:,r}$ and $\boldsymbol{n}^{(r)} = \sqrt{s_r}\mathbf{V}_{:,r}$ in order to obtain a low-rank connectivity decomposition that is unique up to a change of sign in the $\boldsymbol{m}^{(r)}$ and $\boldsymbol{n}^{(r)}$ vectors. A particular consequence of this decomposition is that all $\boldsymbol{m}^{(r)}$ vectors are orthogonal to each other, and similarly for the $\boldsymbol{n}^{(r)}$ vectors.

Finally, in the case when $\mathbf{J}$ is not constrained to be of a certain rank $R$ we will call the network "full-rank", since under most conceivable probability measures on matrices, the probability of obtaining a non-full-rank matrix is nil.

### II.2.2 Biological interpretation of the models

The networks described by Eq. (II.1) are classically called in neuroscience *rate networks*, and are typically interpreted by mapping each unit to a single neuron and by associating $x_i(t)$ to its input electrical current (as can be measured by the patch clamp technique). In that case, $\phi(x_i(t))$ can be understood as the firing rate of unit $i$, $\phi$ representing the frequency-input curve (or f-I curve, see Fig. I.2) that can be inferred from *in vitro* or *in vivo* experiments (giving a wide range of possible transfer functions depending on the neuron class, brain area, and measurement context).

The $-x_i$ term on the right-hand side of Eq. (II.1) can thus be interpreted as the current leak through the neuron's membrane, the connections $J_{ij}$ as the synaptic strengths between the neurons of the network, the input weights $I_i^{(s)}$ as the synaptic strengths of incoming connections from other brain areas or from afferent nerves, while the noise $\eta_i(t)$ models the intrinsic randomness of physico-chemical processes inside neurons and synapses.

In this dissertation, we will however significantly depart from this interpretation since we will often consider parameters that would not respect certain biological constraints: in particular, our networks will exhibit negative firing rates, are not required to obey Dale's law (39), and are not required to have sparse connectivity. We shall thus prefer a rather

more abstract interpretation of equation (II.1), where $\phi(x_i(t))$ can represent some centered measure of a neuron's activity (like z-scored smoothed firing rates, or calcium imaging fluorescence traces), or even where each so-called *unit* doesn't need to map to a single neuron. We will come back to the biological underpinnings of our model in the discussion of this thesis, and for the time being will simply consider them as general abstract models of recurrently connected non-linear simple units.

### II.2.3 Low-dimensional dynamics

Let us consider a network driven by the dynamics (II.1) with a low-rank connectivity matrix given by (II.7) and no noise. In that case, the network dynamics can be written as:

$$\tau \frac{dx_i}{dt} = -x_i + \frac{1}{N} \sum_{r=1}^{R} m_i^{(r)} \sum_{j=1}^{N} n_j^{(r)} \phi(x_j) + \sum_{s=1}^{N_{in}} I_i^{(s)} u_s(t). \tag{II.10}$$

At any time $t$, the right-hand-side is confined to the linear subspace spanned by the vectors $\boldsymbol{m}^{(r)}$ and $\boldsymbol{I}^{(s)}$. Indeed, due to the leak term in the dynamics, any component of the vector $\boldsymbol{x}(t)$ that is outside of this subspace will decay exponentially fast, and after this transient the dynamics of $\boldsymbol{x}(t)$ remain in the said subspace for all $t$. The activation vector $\boldsymbol{x}(t)$ can therefore be decomposed on a basis formed by the vectors $\boldsymbol{m}^{(r)}$ and $\boldsymbol{I}^{(s)}$, defining $R$ internal collective variables $\kappa_r(t)$, and $N_{in}$ external collective variables $v_s(t)$:

$$\boldsymbol{x}(t) = \sum_{r=1}^{R} \kappa_r(t) \boldsymbol{m}^{(r)} + \sum_{s=1}^{N_{in}} v_s(t) \boldsymbol{I}_{\perp}^{(s)}. \tag{II.11}$$

where the $\boldsymbol{I}_{\perp}^{(s)}$ are obtained by a Gram-Schmidt orthogonalization process of the $\boldsymbol{m}^{(r)}$ and $\boldsymbol{I}^{(s)}$ vectors (and the $\boldsymbol{m}^{(r)}$ are already orthogonal to each other given the chosen decomposition).

The first term on the right-hand side in Eq. (II.11) represents the component of the activity on the *recurrent space* (219; 153) defined as the subspace spanned by the output connectivity vectors $\boldsymbol{m}^{(r)}$. The corresponding internal collective variables $\kappa_r$ are defined as projections of the activation vector $\boldsymbol{x}(t)$ on the $\boldsymbol{m}^{(r)}$:

$$\begin{aligned} \kappa_r(t) &= \frac{1}{\|\boldsymbol{m}^{(r)}\|^2} \sum_{j=1}^{N} m_j^{(r)} x_j(t) \\ &= \frac{{\boldsymbol{m}^{(r)}}^{\top} \boldsymbol{x}(t)}{\|\boldsymbol{m}^{(r)}\|^2}. \end{aligned} \tag{II.12}$$

The second term on the right-hand side in Eq. (II.11) represents the component of the activity on the *input space* defined as the subspace spanned by the $\boldsymbol{I}_{\perp}^{(s)}$ vectors, the set of input vectors orthogonalized with respect to the recurrent subspace. The corresponding external collective variables $v_s$ are defined as projections of the activation vector $\boldsymbol{x}(t)$ on $\boldsymbol{I}_{\perp}^{(s)}$:

$$\begin{aligned} v_s(t) &= \frac{1}{\|\boldsymbol{I}_{\perp}^{(s)}\|^2} \sum_{j=1}^{N} I_{\perp,j}^{(s)} x_j(t) \\ &= \frac{{\boldsymbol{I}_{\perp}^{(s)}}^{\top} \boldsymbol{x}(t)}{\|\boldsymbol{I}_{\perp}^{(s)}\|^2}. \end{aligned} \tag{II.13}$$

The embedding dimensionality of the dynamics in state-space is thus given by the sum of the dimension $R$ of the recurrent subspace, *i.e.* the rank of the connectivity, and the dimensionality $N_{in}$ of the input space.

The dynamics of the internal variables $\kappa_r$ are obtained by projecting Eq. (II.1) onto the output connectivity vectors $\boldsymbol{m}^{(r)}$:

$$\tau \frac{d\kappa_r}{dt} = -\kappa_r(t) + \kappa_r^{rec}(t) + \frac{1}{\|\boldsymbol{m}^{(r)}\|^2} \sum_{j=1}^{N} m_j^{(r)} \sum_{s=1}^{N_{in}} I_j^s u_s(t) \tag{II.14}$$

where $\kappa_r^{rec}$ represents the recurrent input to the $r$-th collective variable, defined as the projection of the firing rate vector $\phi(\boldsymbol{x})$ onto the input-selection vector $\boldsymbol{n}^{(r)}$:

$$\kappa_r^{rec}(t) = \frac{1}{N} \sum_{j=1}^{N} n_j^{(r)} \phi(x_j(t)). \tag{II.15}$$

Inserting Eq. (II.11) into $\kappa_r^{rec}$ leads to a closed set of equations for the $\kappa_r$:

$$\kappa_r^{rec}(t) = \frac{1}{N} \sum_{j=1}^{N} n_j^{(r)} \phi \left( \sum_{r'=1}^{R} \kappa_r'(t) m_j^{(r')} + \sum_{s=1}^{N_{in}} I_{\perp,j}^s v_s(t) \right). \tag{II.16}$$

The dynamics of the external variables $v_s$ are obtained by projecting Eq. (II.1) onto the orthogonalized input vectors $\boldsymbol{I}_\perp^{(s)}$. They are given by external inputs $u_s(t)$ filtered by the single neurons time constant $\tau$:

$$\tau \frac{dv_s}{dt} = -v_s + u_s. \tag{II.17}$$

Unless stated otherwise, in this work we assume for simplicity that the stimuli $u_s$ vary on a timescale slower than $\tau$, and replace $v_s$ with $u_s$. We also assume throughout that input vectors are orthogonal to the output connectivity vectors, ie. $\boldsymbol{I}^{(s)} = \boldsymbol{I}_\perp^{(s)}$ for all $s$. Hence the third term on the r.h.s. of equation (II.14) equals zero.
Finally, the readout value can be obtained by inserting Eq. (II.11) into Eq. (II.1), which gives:

$$z(t) = \frac{1}{N} \sum_{j=1}^{N} w_j \phi \left( \sum_{r=1}^{R} \kappa_r(t) m_j^{(r)} + \sum_{s=1}^{N_{in}} I_{\perp,j}^s v_s(t) \right). \tag{II.18}$$

### II.2.4 Collective dynamics in the mean-field limit

Equation (II.14) shows that the state-space dynamics can be summarized by an $R$-dynamical system formed by the set of internal variables $\boldsymbol{\kappa} = \{\kappa_r\}_{r=1...R}$, formalized as:

$$\frac{d}{dt}\boldsymbol{\kappa}(t) = F(\boldsymbol{\kappa}(t), \boldsymbol{u}(t)) \tag{II.19}$$

where $F$ is a non-linear multivariate function that depends on the set of connectivity parameters given by the entries of vectors $\boldsymbol{m}^{(r)}$, $\boldsymbol{n}^{(r)}$ and $\boldsymbol{I}^{(s)}$. More precisely, each neuron $i$ is characterized by a set of $2R + N_{in} + 1$ parameters:

$$\{I_i^{(1)}, \ldots, I_i^{(N_{in})}, n_i^{(1)}, \ldots, n_i^{(R)}, m_i^{(1)}, \ldots, m_i^{(R)}\}_{i=1...N} \tag{II.20}$$

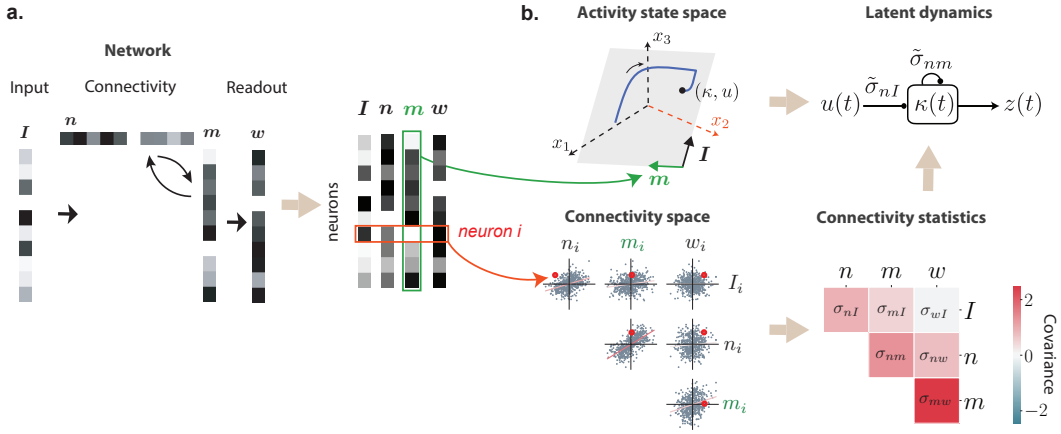


Figure II.1: Relationships between connectivity and activity in low-rank RNNs. (a) The connectivity parameters in a low-rank RNN (left) can be grouped in a matrix where each row contains the input, recurrent and output parameters (values illustrated in grayscale) of a given neuron (right). (b) The connectivity can therefore be represented in two complementary manners that together determine low-dimensional dynamics. Top-left: Columns of the matrix in (a) define specific directions (illustrated as arrows) in the activity state space, where each axis is the activity $x_i$ of neuron $i$. The connectivity constrains the trajectories of activity to lie in a low-dimensional subspace spanned by input vectors $\boldsymbol{I}^{(s)}$ and recurrent vectors $\boldsymbol{m}^{(r)}$. The activity trajectory (illustrated in blue) is parametrized along those directions by input variables $u_s$ and internal variables $\kappa_r$. Bottom left: Each row of the matrix in (a) defines a point in the connectivity space (specific example in red), where each axis corresponds to entries along each connectivity vector. The full network is described by the distribution of the cloud of points. Here we illustrate a four-dimensional distribution by its pairwise two-dimensional projections. Bottom right: a Gaussian distribution in connectivity space is specified by its covariance matrix that describes the shape of the point cloud (regression lines shown in bottom left). Top right: The latent dynamics can be reduced to an effective circuit (Eq. (III.3)), in which each internal variable is represented as a unit that receives external inputs, and interacts with itself (and other internal variables) through a set of effective couplings determined by the connectivity covariances illustrated in the bottom-left panel.

Each neuron can thus be represented as a point in a *connectivity space* of dimension $2R + N_{in} + 1$, and the connectivity of the full network can therefore be described as a set of $N$ points in this space (Fig. II.1b, bottom left). Note that the right-hand-side of Eq. (II.16) consists of a sum of $N$ terms, where the term $j$ contains only the connectivity parameters of neuron $j$. The connectivity parameters of different neurons therefore do not interact in $\kappa_r^{rec}$, so that the r.h.s of Eq. (II.16) can be interpreted as an average over the set of points corresponding to all neurons in the connectivity space.

Our main assumption will be that in the limit of large networks ($N \to \infty$), the set of points in the connectivity space is described by a probability distribution $P(n^{(1)}, \dots, n^{(R)}, m^{(1)}, \dots, m^{(R)}, I^{(1)}, \dots, I^{(N_{in})}, w) := P(\underline{n}, \underline{m}, \underline{I}, w)$. In this mean-field limit, the r.h.s. of Eq. (II.16) becomes:

$$\kappa_r^{rec}(t) = \int \mathrm{d}\underline{m}\, \mathrm{d}\underline{n}\, \mathrm{d}\underline{I}\, \mathrm{d}w\, P(\underline{n}, \underline{m}, \underline{I}, w)\, n^{(r)} \phi\left( \sum_{r'=1}^{R} \kappa_{r'}(t) m^{(r')} + \sum_{s'=1}^{N_{in}} I_{\perp}^{(s')} v_{s'}(t) \right), \quad \text{(II.21)}$$

where we have used the shorthand $\mathrm{d}\underline{m}\, \mathrm{d}\underline{n}\, \mathrm{d}\underline{I} = \prod_{r'=1}^{R} \prod_{s'=1}^{N_{in}} \left( \mathrm{d}m^{(r')} \mathrm{d}n^{(r')} \mathrm{d}I^{(s')} \right)$. The col-

lective dynamics are therefore fully specified by the single-neuron distribution of connectivity parameters. Once this distribution is specified, any network generated by sampling from it will have identical collective dynamics in the limit of a large number of neurons. In this view, the vectors defining network connectivity, *i.e.* the $\boldsymbol{I}^{(s)}$, $\boldsymbol{n}^{(r)}$, $\boldsymbol{m}^{(r)}$ and $\boldsymbol{w}$ vectors will be assimilated to probabilistic *connectivity parameters*, which formally are random variables in a shared probabilistic space, noted $n^{(r)}$, $m^{(r)}$, $I^{(s)}$, and $w$.
The joint distribution of connectivity parameters $P(\underline{n}, \underline{m}, \underline{I}, w)$ also determines the values of the readout:

$$z(t) = \int \mathrm{d}\underline{m}\, \mathrm{d}\underline{n}\, \mathrm{d}\underline{I}\mathrm{d}w\; P(\underline{n}, \underline{m}, \underline{I}, w)\; w\phi\left(\sum_{r'=1}^{R} \kappa_{r'}(t) m^{(r')} + \sum_{s'=1}^{N_{in}} I_{\perp}^{(s')} v_{s'}(t)\right). \tag{II.22}$$

### II.2.5 Dynamics in Gaussian-mixture low-rank networks

To approximate any arbitrary joint distribution of connectivity parameters $P(\underline{n}, \underline{m}, \underline{I}, \underline{w})$, we use multivariate Gaussian mixture models (GMMs). This choice was based on the following considerations: (i) GMMs are able to approximate an arbitrary multivariate distribution (99); (ii) model parameters can be easily inferred from data using GMM clustering; (iii) GMMs afford a natural interpretation in terms of sub-populations; (iv) GMMs allow for a mathematically tractable and transparent analysis of the dynamics as shown below (11).

In a multivariate Gaussian mixture model, every neuron belongs to one of $P$ sub-populations. For a neuron in subpopulation $p$, the set of parameters $\{\{n_i^{(r)}\}_r, \{m_i^{(r)}\}_r, \{I_i^{(s)}\}_s, w_i\}$ is generated from a multivariate Gaussian distribution with mean $\boldsymbol{\mu}_p$ and covariance $\boldsymbol{\Sigma}_p$, where $\boldsymbol{\mu}_p$ is a vector of size $2R+N_{in}+1$, and $\boldsymbol{\Sigma}_p$ is a covariance matrix of size $(2R+N_{in}+1)^2$. The full distribution of connectivity parameters is therefore given by

$$P(\underline{n}, \underline{m}, \underline{I}, w) = \sum_{p=1}^{P} \alpha_p \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p) \tag{II.23}$$

$$:= \sum_{p=1}^{P} \alpha_p P_p(\underline{n}, \underline{m}, \underline{I}, w) \tag{II.24}$$

where the coefficients $\alpha_p$ define the fraction of neurons belonging to each subpopulation.

Each subpopulation directly corresponds to a Gaussian cluster of points in the connectivity space. The vector $\boldsymbol{\mu}_p$ determines the center of the $p$-th cluster, while the covariance matrix $\boldsymbol{\Sigma}_p$ determines its shape and orientation.

To characterize the covariance matrices, we write for any two connectivity parameters $a$ and $b$ (being one of the $n^{(r)}$, $m^{(r)}$, $I^{(s)}$, or $w$ parameters) their *overlap* or aggregated covariance $\sigma_{ab}$ given by:

$$\sigma_{ab} = \frac{1}{N} \sum_{i=1}^{N} a_i b_i. \tag{II.25}$$

This overlap admits a probabilistic interpretation as the covariance between the parameters, implying that it can be visualized as the slope of a linear regression on the normalized point cloud representing the entries of each neuron on those two parameters (Fig. II.1b, bottom left). Considering that in a network, these parameters are materialized as vectors in the neural state-space $\boldsymbol{a}$ and $\boldsymbol{b}$ (Fig. II.1b, top left), the overlap $\sigma_{ab}$ can then be interpreted as a rescaled scalar product, equal to $\cos(\widehat{\boldsymbol{ab}})\|\boldsymbol{a}\|\|\boldsymbol{b}\|$, and can be visualized from the angle between those two vectors. In particular, an overlap of 0 corresponds to probabilistic independence between the parameters $a$ and $b$, and is materialized by approximate orthogonality of the sampled

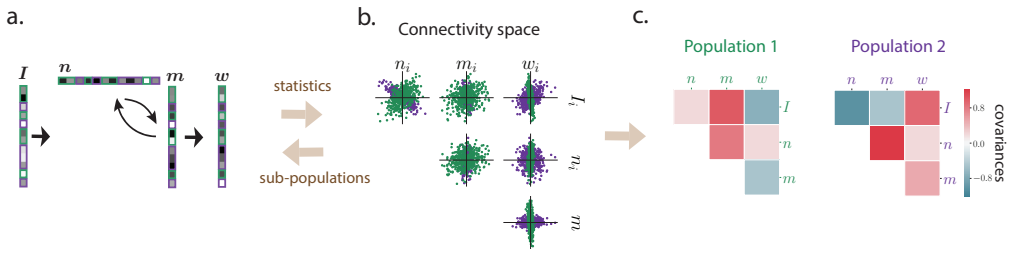


Figure II.2: Method for representing a low-rank connectivity structure in terms of multiple subpopulations. (a) The connectivity vectors are represented as a set of points in connectivity space, each point corresponding to connectivity parameters of one neuron. (b) Illustration of two-dimensional projections of the full distribution in connectivity space, which in this example is four dimensional. Every neuron belongs to a subpopulation that corresponds to a component of the mixture-of-Gaussian distribution of parameters in this connectivity space. The green and purple colors denote the two subpopulations, which in this illustration have identical centers but different shapes. (c) Each sub-population is therefore defined by a different set of covariances, that correspond to overlaps between vectors shown in green and purple colors in the left panel.

vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ (if $N$ is large enough). Higher overlaps correspond to strong correlation between the variables, materialized by an alignment of the state-space vectors.

Note that because the output vectors $\boldsymbol{m}^{(r)}$ (resp. input-selection vectors $\boldsymbol{n}^{(r)}$) are mutually orthogonal, the covariances between the parameters $\{m^{(r))}\}_{r=1...R}$ (respectively $\{n^{(r)}\}_{r=1...R}$) vanish.

For the set of neurons $i$ belonging to population $p$, we will write as $\sigma_{ab}^{(p)}$ the covariance, also called *overlap* between two connectivity parameters $a$ and $b$ over population $p$:

$$\sigma_{ab}^{(p)} = \frac{1}{\alpha_p N} \sum_{i \in \text{pop. } p} a_i b_i, \tag{II.26}$$

which are the elements of the covariance matrix $\boldsymbol{\Sigma}_p$, defining the $p$-th component of the Gaussian mixture (Fig. II.2c). Note that the previous global overlap is simply the weighted average of population-wise covariances: $\sigma_{ab} = \sum_p \alpha_p \sigma_{ab}^{(p)}$.

Since every neuron belongs to a single population, the r.h.s of Eq. (II.16) can be split into P terms, each corresponding to an average over one population. As within each population the distribution is a joint Gaussian, Eq. (II.21) becomes a sum of $P$ Gaussian integrals

$$\kappa_r^{rec}(t) = \sum_{p=1}^{P} \alpha_p \int \mathrm{d}\underline{m}\mathrm{d}\underline{n}\mathrm{d}\underline{I}\mathrm{d}w \; P_p(\underline{n}, \underline{m}, \underline{I}, w) \; n^{(r)} \; \phi\left( \sum_{r'=1}^{R} \kappa_{r'}(t) m^{(r')} + \sum_{s'=1}^{N_{in}} I_{\perp}^{(s')} v_{s'}(t) \right). \tag{II.27}$$

### II.2.6 Effective circuit description of latent dynamics

In the following, we focus on zero-mean multivariate Gaussian mixture distributions for the connectivity parameters, and input vectors orthogonal to $\{\boldsymbol{m}^{(r)}\}_{r=1...R}$, as distributions with these assumptions will be sufficient for the present work. The more general case of Gaussian mixtures with non-zero means is treated in (11). Using Stein's lemma for Gaussian distributions, the dynamics of the internal collective variables can be expressed as a dynamical system (see Appendix A.1)

$$\frac{d\kappa_r}{dt} = -\kappa_r + \sum_{r'=1}^{R} \tilde{\sigma}_{n^{(r)}m^{(r')}} \kappa_{r'} + \sum_{s=1}^{N_{in}} \tilde{\sigma}_{n^{(r)}I^{(s)}} v_s. \tag{II.28}$$

In the next chapters, $v_s$ will be replaced by $u_s$ which amounts to assume that inputs vary slowly with respect to the single neuron time constant $\tau$.
In Eq. (II.28), $\tilde{\sigma}_{n^{(r)}m^{(r)}}$ represents the effective self-feedback of the collective variable $\kappa_r$, $\tilde{\sigma}_{n^{(r)}m^{(r')}}$ sets the interaction between the collective variables $\kappa_r$ and $\kappa_{r'}$, and $\tilde{\sigma}_{n^{(r)}I^{(s)}}$ is the effective coupling between the input $u_s$ and $\kappa_r$. These effective interactions between the internal variables are given by weighted averages over populations:

$$\tilde{\sigma}_{ab} = \sum_{p=1}^{P} \alpha_p \sigma_{ab}^{(p)} \langle \Phi' \rangle_p, \tag{II.29}$$

where $\sigma_{ab}^{(p)}$ is the covariance between connectivity parameters $a$ and $b$ for population $p$ (Eq. (II.26)), and $\langle \Phi' \rangle_p$ is the average gain of population $p$, defined as:

$$\langle \Phi' \rangle_p = \langle \Phi' \rangle(\Delta^{(p)}), \tag{II.30}$$

with

$$\langle \Phi' \rangle(\Delta) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} \mathrm{d}z \, e^{-z^2/2} \phi'(\Delta z) \tag{II.31}$$

and

$$\Delta^{(p)} = \sqrt{\sum_{r'=1}^{R} (\sigma_{m^{(r')}}^{(p)})^2 \kappa_{r'}^2 + \sum_{s=1}^{N_{in}} (\sigma_{I^{(s)}}^{(p)})^2 v_s^2} \tag{II.32}$$

the standard deviation of activation variables in population $p$, where $\sigma_a^{(p)}$ is the variance of a vector $\boldsymbol{a}$ on population $p$.
In Eq. (II.28), the covariances $\sigma_{ab}^{(p)}$ are set by the statistics of the connectivity and input vectors, but the gain factors $\langle \Phi' \rangle_p$ in general depend both on internal and external collective variables $\kappa_k$ and $v_j$. As a consequence, the dynamics in Eq. (II.28) is non-linear, and in fact it can be shown that given a sufficient number of sub-populations, the right-hand side in Eq. (II.28) can approximate any arbitrary dynamical system (11).
In the special case of linear networks (i.e. $\Phi(x) = x$), the gain is constant so that the effective couplings $\tilde{\sigma}_{ab}$ in Eq. (II.29) are equal to the overlaps $\sigma_{ab}$ of vectors $a$ and $b$ over the full population, as defined in Eq. (II.25). The population structure therefore only plays a role for non-linear networks.
The value of the readout (Eq. (II.22)) can also be expressed in terms of effective interactions as

$$z = \sum_{r'=1}^{R} \tilde{\sigma}_{m^{(r')}w} \kappa_{r'} + \sum_{s=1}^{N_{in}} \tilde{\sigma}_{I^{(s)}w^{(k)}} v_s. \tag{II.33}$$

### II.2.7 Drivers and modulators of latent dynamics

Eq. (II.28) shows that feed-forward inputs to the network can have two distinct effects on the collective dynamics of internal variables $\kappa_r$. If the input vector $I^{(s)}$ overlaps with the $r$-th input-selection vector $n^{(r)}$, i.e. the corresponding covariance $\sigma_{n^{(r)}I^{(s)}}^{(p)}$ is non-zero for population $p$, the input directly drives the latent dynamics, in the sense that $v_s$ acts as an effective external input to the dynamics of $\kappa_r$ in Eq. (II.28).

In contrast, when all covariances between the input vectors and the input selection vectors are zero (i.e. $\sigma^{(p)}_{n^{(r)}I^{(s)}} = 0$ for all $r, p$), the corresponding input does not drive the latent dynamics, but can still modulate them by modifying the gain through Eq. (II.32) if the variance $\sigma^{(p)}_{I^{(s)}}$ of the input on some population $p$ is non-zero.

The inputs can therefore play roles of drivers and modulators of latent dynamics, depending on whether the corresponding input vectors overlap or not with the input selection vectors $n^{(r)}$.

### II.2.8 Geometric interpretation

Equations (II.11)-(II.17) provide a direct link between the connectivity of a low-rank recurrent network and its dynamics without assuming any further constraints on the nonlinear transfer function $\phi$ or on the connectivity weights. As such, a first interpretation of the different vectors defining the connectivity can be outlined as follows:

- the $\boldsymbol{I}^{(s)}$ vectors represent directions of the state-space on which the input signals are directly encoded at the population level;
- the $\boldsymbol{n}^{(r)}$ vectors represent *selection axes* of the state-space, which select which collective variables signals encoded at the population level are incorporated into the recurrent dynamics of the network, through the calculation of the corresponding latent recurrently-generated variables $\kappa_r$;
- the $\boldsymbol{m}^{(r)}$ vectors represent directions of the state-space on which the recurrently-generated variables $\kappa_r$ are encoded at the population level.

Hence, as explained earlier the autonomous dynamics remain confined to the subspace spanned by the $\boldsymbol{m}^{(r)}$ vectors, and input-driven dynamics to the subspace spanned by the $\boldsymbol{I}^{(s)}$ vectors. What the mean-field theory outlines, through equation (II.28), is that the key factors in determining the dynamical system expressed by the network are the overlaps between the $\boldsymbol{n}^{(r)}$ vectors and respectively the $\boldsymbol{m}^{(r)}$ vectors for recurrent dynamics, and the $\boldsymbol{I}^{(s)}$ vectors for input-driven dynamics. This overlaps admit a geometrical interpretation as the alignment between those pairs of vectors. Hence, when interpreting low-rank RNNs, we will pay a particular attention to the way in which the $\boldsymbol{n}^{(r)}$ geometrically align with the other vectors defining the connectivity.

For example, in a rank-one network, if a vectors $\boldsymbol{n}$ has a significant overlap with the $\boldsymbol{I}$ vector, this means that external input can be moved within the internal recurrent variable $\kappa(t)$. In order for this variable to be integrated by the input however, one would need $\kappa(t)$ to exhibit self-sustaining dynamics, which can happen (following Eq. (II.28)) if the overlap between $\boldsymbol{n}$ and $\boldsymbol{n}$ approaches one. Hence the simple description of single-variable integrator in this framework: it requires the $\boldsymbol{n}$ vector to be aligned with both $\boldsymbol{I}$ and $\boldsymbol{m}$, as detailed in the next chapter and illustrated in Fig. III.2c. In this case, the input can be considered as a *driver* of the dynamics following the definition of the previous paragraph.

In the special case where a constant tonic input is fed to the network, one consequence of equations (II.11)-(II.17) is that the dynamics will quickly converge to an affine subspace of the states-space, parallel to the subspace spanned by the $\boldsymbol{m}^{(r)}$ vectors but shifted along an axis corresponding to the input being fed. This phenomenon effectively drives a translation of the dynamics to another parallel sector of the state-space, where the dynamical landscape can be different than in the absence of inputs, explaining how inputs can play the role of *modulators* of the dynamics.

## II.3 Gaussian low-rank recurrent networks

Let us now explicit the kind of network dynamics that can be obtained when the connectivity parameters are sampled from a Gaussian distribution (we will fix, in all that follows, that $\phi = \tanh$). This corresponds to the situation presented in paragraphs II.2.5 and II.2.6, except that we will assume there is a single population ($P = 1$). Consequently, the distribution in equation (II.23) becomes:

$$P(\underline{n}, \underline{m}, \underline{I}, w) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{II.34}$$

and the effective overlap between two connectivity parameters $a$ and $b$ given by equation (II.29) becomes:

$$\tilde{\sigma}_{ab} = \sigma_{ab}\langle\Phi'\rangle, \tag{II.35}$$

with the overall gain factor

$$\langle\Phi'\rangle = \langle\Phi'\rangle(\Delta) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} \mathrm{d}z\, e^{-z^2/2}\phi'(\Delta z), \tag{II.36}$$

and

$$\Delta = \sqrt{\sum_{r'=1}^{R} \sigma^2_{m^{(r')}}\kappa^2_{r'} + \sum_{s=1}^{N_{in}} \sigma^2_{I^{(s)}} v_s^2}. \tag{II.37}$$

It follows from these simplified equations that the effective overlaps that define the network dynamics following equation (II.28) are in a Gaussian network all scaled together by a single factor. This fact strongly limits the dynamics that can be expressed by such a network, although an interesting diversity of solutions can still be obtained.

**Rank-one Gaussian networks**. Let us first express the range of dynamics that can be obtained in rank-one Gaussian RNNs. Such a network is characterized by two connectivity vectors $\boldsymbol{n}$ and $\boldsymbol{m}$ and eventually input vectors (we will look at the effect of a single one). Equation (II.28) becomes:

$$\frac{d\kappa}{dt} = -\kappa(t) + \sigma_{nm}\langle\Phi'\rangle\kappa(t) + \sigma_{nI}\langle\Phi'\rangle u(t). \tag{II.38}$$

In the absence of inputs, dynamics remain confined on the line spanned by the $\boldsymbol{m}$ vector, and can thus be visualized as a one-dimensional potential $V(\kappa)$ such that $\frac{d\kappa}{dt} = -\frac{dV}{d\kappa}$. The only parameter that can influence the autonomous dynamics is the overlap $\sigma_{nm}$. The overall gain factor $\langle\Phi'\rangle(\Delta)$ is a function of $\kappa$ when there is no input and can easily be visualized, showing that it simply decreases as $\kappa$ moves away from zero, which formalizes the fact that neurons' activity starts to saturate due to the non-linear transfer function and cannot diverge indefinitely. Hence, the only two dynamics that can be obtained from equation (II.38) are: (i) if $\sigma_{nm} < 1$, the presence of a single stable attractive fixed point at zero, materialized by a single-well potential (Fig. II.3a), or (ii) if $\sigma_{nm} > 1$, the presence of an unstable fixed point at zero and two stable fixed point, symmetrically arranged around it, materialized by a double-well potential (Fig. II.3b). Hence, a strong enough overlap between vectors $\boldsymbol{n}$ and $\boldsymbol{m}$ induces a self-sustaining recurrent loop of activity in the network, which due to the symmetric non-linearity tanh, drives bistable dynamics in the network. Finally, note that when $\sigma_{nm}$ is very close to one, the fixed point at zero becomes marginally stable, and the network thus approximates a line attractor around this point, as could be observed in tasks requiring evidence integration (184; 118) (Fig. I.7a).

If an input $\boldsymbol{I}u(t)$ is fed to the network, its effect depends on the overlap $\sigma_{nI}$. If that overlap is non-zero, the input signal $u(t)$ will enter the dynamics of $\kappa(t)$, following Eq. (II.38).

For instance, for a positive overlap, and with a bistability induced by $\sigma_{mn} > 1$, the dynamics are biased towards one of the stable fixed points under presence of an input (Fig. II.3c). This allows to retrieve a decision-making behavior, in a way that will be detailed in the next chapter. On the other hand, if $\sigma_{nI} = 0$, an input will simply contribute to Eq. (II.38) by reducing the gain term, and hence the effective overlap $\tilde{\sigma}_{mn}(\kappa = 0)$. Note that this effective overlap at zero really sets the broad dynamical landscape in a Gaussian network, and when it decreases below zero the bistability disappears. Hence, for inputs orthogonal to $\boldsymbol{n}$ but strong enough, the two stable fixed points collapse into a single one at zero (Fig. II.3d). Note that here, the dynamics in presence of an input can still be considered as one-dimensional and along $\boldsymbol{m}$ if the input is constant, and in the steady-state regime: indeed, after a short transient along the $\boldsymbol{I}$ vector, the activity $\boldsymbol{x}(t)$ will then again be constrained to a one-dimensional affine line, parallel to $\boldsymbol{m}$ and shifted from the origin along the $\boldsymbol{I}$ axis. This enables us to interpret the effect of inputs as modifications on the autonomous dynamic field.

**Rank-two Gaussian networks**. Let us now look at some interesting cases of rank-two Gaussian networks, without proving the technical details (which can be found in ref. (11)). In this case, four overlaps will fully determine the autonomous dynamics, and it is informative to organize them in the following "effective Jacobian" matrix:

$$\mathbf{J}^{ov} = \begin{pmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{pmatrix}, \tag{II.39}$$

where we used the shortcut notation $\sigma_{kl} = \sigma_{m^k n^l}$. Interestingly, the eigenvalues of $\mathbf{J}$ happen to also be the eigenvalues of $\mathbf{J}^{ov}$, and those directly set the local (linearized) dynamics at zero as can be readily seen from Eq. (II.28). But crucially, for Gaussian networks the local behavior induce the broader dynamical landscape of the network. Here are the autonomous dynamics that can be observed by manipulating those parameters (letting $\lambda_1 \geq \lambda_2$ be the two eigenvalues of $\mathbf{J}^{ov}$ if they are real, $\lambda_1$ and $\lambda_1$ if they are complex conjugate):

(i) if $\lambda_1 < 1$, a single stable fixed point exists at 0;

(ii) if $\lambda_2 < 1 < \lambda_1$, the fixed point at 0 becomes unstable (effectively a two-dimensional saddle point) and two stable fixed points appear symmetrically around it, along the eigenvector associated with $\lambda_1$, the networks is thus bistable (Fig. II.3e);

(iii) if $1 < \lambda_2 < \lambda_1$, several phenomena can occur: if the network is "sufficiently symmetric" (that is, if the eigenvalues of $\mathbf{J}^{ov}$ are real), the two stable fixed points along the eigenvector associated to $\lambda_1$ are complemented by two saddle points along the eigenvector associated to $\lambda_2$, while zero becomes unstable in both directions (Fig. II.3f); if the eigenvalues of $\mathbf{J}^{ov}$ become imaginary however, all the non-zero fixed points disappear (in saddle-node bifurcations) to leave a limit cycle around the 0 unstable fixed point (Fig. II.3h);

(iv) finally, if $1 < \lambda_1 = \lambda_2$, a ring attractor appears around 0 (Fig. II.3g).

The above enumeration exhibits a range of interesting and useful behaviors, but also lacks a wide range of imaginable dynamics: for example the cohabitation of two pairs of stable fixed points on the two directions spanned by the main eigenvectors is impossible with a Gaussian low-rank network, unless it is when the whole ring is stable. Similarly, for all networks with a rank higher than 2, an analysis based on the eigenvalues of $\mathbf{J}^{ov}$ can be performed, and it can be observed that the eigenvector associated to the largest eigenvalue "wins" and becomes associated with the two only stable fixed points, unless several eigenvalues are equal in which case stable hyperspheres can be obtained.

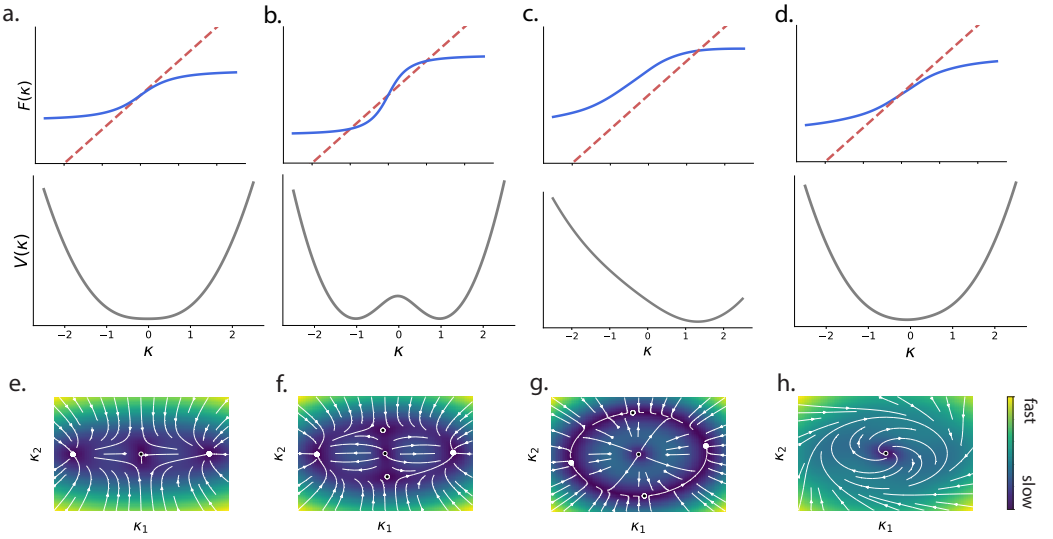


FIGURE II.3: a-d. Example dynamics for Gaussian rank-one networks. Top: nullclines of the r.h.s. of Eq. (II.38), including the $y = \kappa$ line (red dashed) and $y = \tilde{\sigma}_{nm}\kappa + \tilde{nI}u(t)$ curve (blue). The intersection points are thus stable points of the dynamics. Bottom: corresponding potential function of the one-dimensional dynamics (mathematical primitive of the 1D field). All plots illustrate a realization with $N = 500$ neurons. a. Case with $\sigma_{mn} = 0.8$, b. with $\sigma_{mn} = 2$, c. with $\sigma_{mn} = 2$, $\sigma_{nI} = 1$, $\sigma_{I^2} = 1$ and in presence of a constant input of strength $u(t) = 2$, d. same with $\sigma_{nI} = 0$. e-f. Example dynamics for Gaussian rank-two networks (simulations with $N = 2048$). e. Example of case $\lambda_1 > 1 > \lambda_2$. f. Case $\lambda_1 > \lambda_2 > 1$. g. Case $\lambda_1 = \lambda_2 > 1$, implementing a ring attractor. Notice that we can still observe only two authentic stable fixed points on this ring, and a slow drift ohterwise along it, due to finite size effects. h. Case where $\lambda_1$ and $\lambda_2$ are complex conjugate, implementing a limit cycle.

The effect of inputs on rank-two Gaussian networks is similar to that explained for rank-one networks, with inputs that overlap with $\boldsymbol{n}^{(1)}$ or $\boldsymbol{n}^{(2)}$ inducing a bias in a certain direction, and orthogonal inputs tending to shrink the dynamical landscape towards zero.

## II.4 Gaussian-mixture low-rank recurrent networks

All dynamical landscapes cannot be obtained with a simple Gaussian-parametrized low-rank network, and in order to obtained more varied behaviors we have to allow more flexibility on parameter sampling. One possibility is to parametrize the connectivity space with for example a mixture-of-Gaussians distribution, following Eq. (II.23). It turns out that, as demonstrated in reference (11), such a parametrization is sufficient to approximate any dynamical landscape (with particular conditions at infinity), including the effect of inputs on dynamics, provided a sufficiently large number of components $P$ in the mixture. This is particularly fortunate for us since the dynamics remain tractable in the mean-field limit for these networks, which hence stay interpretable while becoming extremely flexible.

A typical example of the more varied dynamics that can be obtained with Gaussian-mixture low-rank networks are phase portraits that would exhibit more than two stable fixed points without implementing an entire stable hypersphere (as those were the only two cases given by Gaussian networks). For instance, cohabitation of two pairs of fixed points in two orthogonal directions, as illustrated in Fig. II.4 becomes possible. This example would actually require components with a non-zero mean, which we did not include in our

formalism in this thesis, and for which further details can be found in reference (11).

Another interesting feature of Gaussian-mixture low-rank networks are that they enable a finer control of internal dynamics *via* "modulator" inputs, as explained in section II.2.7. Indeed, Eqs. (II.28) and (II.29) show that the dynamical system implemented by the network can be modified if each effective overlap can be tuned separately. As we have seen, for Gaussian networks, gains and effective overlaps are all modulated together, and an input orthogonal to the $\boldsymbol{m}^{(r)}$ and $\boldsymbol{n}^{(r)}$ vectors can only "shrink" the dynamical landscape as illustrated for example in Fig. II.3d. In the mixture-of-Gaussians case, the gain of each population is an independent variable, and each can thus be modified by different inputs. Hence, a modulator input could for instance increase a certain effective overlap by cancelling the contribution of one single population to it, hence enriching considerably the range of input-induced dynamics in a single network. A practical rank-one example is given in the next chapter, let us for the moment illustrate this phenomenon with a rank-two situation that will prove useful later.

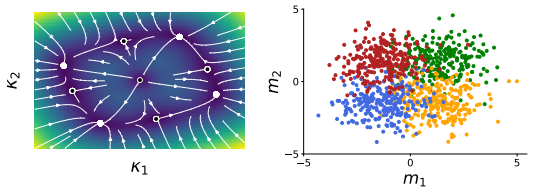


Figure II.4: Example of complex dynamical landscape obtained with rank-2 mixture-of-Gaussians connectivity. Here the connectivity is sampled from 4 populations (colored in the bottom, slice $m_1 - m_2$ of the connectivity space), leading to 4 stable fixed points, separated by 4 saddle points, and with an unstable source at 0 (top). This setup is reminiscent of a Hopfield network (see ref. (11) for more details)

Let us consider the network with 2 populations $p \in \{1, 2\}$, and the following overlaps: $\sigma_{11}^{(p)} = 3$, $\sigma_{22}^{(p)} = 2.5$, $\sigma_{12}^{(p)} = 1$ for $p = 1, 2$, and $\sigma_{21}^{(1)} = 1$, $\sigma_{21}^{(2)} = -1$. Hence, the two populations differ only in terms of the correlation betwen their parameters $m^{(2)}$ and $n^{(1)}$, forming two clouds of points in connectivity space that can be visualized on the plane combining these two parameters. An "effective Jacobian" matrix at the origin can be calculated as in Eq. (II.39), and will here be:

$$\mathbf{J}^{ov} = \begin{pmatrix} 3 & 1 \\ 0 & 2.5 \end{pmatrix} \tag{II.40}$$

The autonomous dynamics of this network can be visualized, and show a bistable behavior, slight variant of that exhibited in Fig. II.3f, only with fixed points that are not aligned with the $\boldsymbol{m}^{(1)}$ and $\boldsymbol{m}^{(2)}$ vectors because of the extra-diagonal term in this effective Jacobian. Now, let us add a modulator input, *i.e.* with no overlap with any of the $\boldsymbol{m}^{(r)}$ and $\boldsymbol{n}^{(r)}$ vector, but targeting only cells of the first population: more precisely, $I_i$ is sampled with a standard deviation of 10 for cells of the first population, and 0 for cells of the second population. When a tonic input signal $u(t) = 1$ is fed to the network through this input vector, it differentially modulates the gains of each population, strongly decreasing that of population 1 (setting $\langle \Phi' \rangle_1 = 0$) without affecting that of population 2 ($\langle \Phi' \rangle_2 = 1$). This in turn affects the effective overlap $\tilde{\sigma}_{21}$ by cancelling the contribution of population 1 to it:

$$\tilde{\sigma}_{21} = 0.5\sigma_{21}^{(1)} \langle \Phi' \rangle_1 + 0.5\sigma_{21}^{(2)} \langle \Phi' \rangle_2 \approx 0.5\sigma_{21}^{(2)} = -0.5, \tag{II.41}$$

while other effective overlaps are simply reduced by half since contributions of both populations were equal. Overall, in the steady tonic input regime and with no recurrent contribution (for now $\kappa_1 = \kappa_2 = 0$), we obtain the following effective Jacobian:

$$\tilde{\mathbf{J}}^{ov} \approx \begin{pmatrix} 1.5 & 0.5 \\ -0.5 & 1.25 \end{pmatrix} \tag{II.42}$$

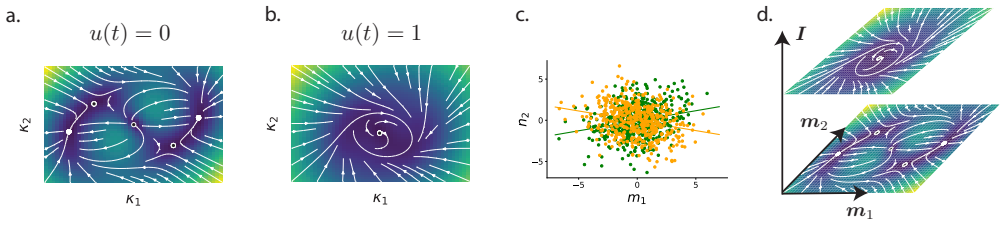


FIGURE II.5: Example of input-mediated modulation of dynamics in a Gaussian-mixture rank-2 network (see text for its characteristics). a. In the absence of external input, the dynamical landscape exhibits two stable fixed points (filled white dots), separated by two saddle points and a central unstable source. b. In the presence of a constant external input, neural activity evolves along an affine plane, parallel to vectors $\boldsymbol{m}^{(1)}$ and $\boldsymbol{m}^{(2)}$. On this affine plane, the dynamical landscape is now changed and exhibits no stable fixed points, and a limit cycle instead. c. Connectivity parameters of each neuron of the network on the $\boldsymbol{m}^{(1)}$ and $\boldsymbol{n}^{(2)}$ vectors. For each population (in green and orange), a linear regression line is superimposed. d. Illustration of the linear subspace on which autonomous dynamics unfold (along vectors $\boldsymbol{m}^{(1)}$ and $\boldsymbol{m}^{(2)}$ and of the affine subspace along which they unfold when a steady input is fed to the network (shifted along the $\boldsymbol{I}$ axis).

which can be verified with simulations. This asymmetric effective Jacobian now resembles the one which was used to induce a limit cycle in a Gaussian network (Fig. II.3h), and appears to indeed induce a cycle in the affine plane of the input-driven dynamics of our network. This example hence illustrates how mixture-of-Gaussians low-rank networks allow finely tuned inputs to completely reshape their internal dynamics, allowing them to shift between dynamical primitives depending on circumstances. We will in particular see in the next chapter how the rank-two network that we presented provides a solution to the delayed match-to-sample task.

## II.5 Training low-rank recurrent networks

One of the main aims of this work is to train low-rank RNNs, either on cognitive tasks (chapter III) or directly on neural data (chapter V) in order to reverse-engineer the obtained solutions. For this, we define trainable low-rank RNNs as follows: first, we discretize Eq. (II.1) with the Euler method and a timestep $\Delta t = 20$ ms, implementing the equation:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \alpha\left(-\boldsymbol{x}_t + \mathbf{J}\phi\left(\boldsymbol{x}_t\right) + \mathbf{I}u_t\right) + \sigma\boldsymbol{\eta}_t, \tag{II.43}$$

where $\alpha = \frac{\Delta t}{\tau} = 0.2$ and $\boldsymbol{\eta}_t$ is i.i.d. Gaussian white noise applied to each neuron. To force the connectivity matrix $\mathbf{J}$ to be low-rank, we parametrize it as a product of low-rank matrices:

$$\mathbf{J} = \frac{1}{N}\mathbf{M}\mathbf{N}^\top, \tag{II.44}$$

where $\mathbf{M}, \mathbf{N} \in \mathbb{R}^{N\times R}$ with $R$ chosen by the modeller[1]. During training, backpropagation-through-time can be applied in order to train the weights of each of the matrices $\mathbf{M}$, $\mathbf{N}$, $\mathbf{I}$, not necessarily at the same time (we will in general train only $\mathbf{M}$ and $\mathbf{N}$ and leave $\mathbf{I}$ as random weights, although it is sometimes insightful to train the input weights as well). Finally a random linear readout is generated from the network activity:

[1]Other solutions could have been chosen: for example certain regularization schemes are known to induce low-rank biases to connectivity matrices by minimizing the nuclear norm. However, we chose the above solution in order to precisely control the obtained rank.

$$z_t = \boldsymbol{w}^\top \boldsymbol{x}_t, \tag{II.45}$$

to compute the loss:

$$\mathcal{L} = \sum_{k,t} M_t (z_{k,t} - z^*_{k,t})^2 \tag{II.46}$$

where $z^*_{k,t}$ represents the desired output for trial $k$ and timestep $t$, and the $M_t$ weights are $\{0, 1\}$ masks marking which at which timesteps the readout is constrained to a target value. During training, the rank of $\mathbf{J}$ is constrained to be at most $R$ by the parametrization (II.44), but this parametrization presents the disadvantage of inducing a degeneracy: if an invertible linear transformation is applied to the columns of $\mathbf{N}$ and its inverse to the columns of $\mathbf{M}$ (that is, if an isomorphic transformation is applied to the $R$-dimensional latent space), the same matrix $\mathbf{J}$ is obtained. This can complicate analyses and comparisons of networks. As stated in equation (II.9), a particular definition of $\mathbf{M}$ and $\mathbf{N}$ can be obtained from $\mathbf{J}$ by its singular value decomposition, for example by posing $\mathbf{M} = \mathbf{U}\sqrt{\mathbf{S}}$ and $\mathbf{N} = \mathbf{V}\sqrt{\mathbf{S}}$ (where $\sqrt{\mathbf{S}}$ has to be understood as the diagonal matrix containing the square roots of the diagonal entries of $\mathbf{S}$). To remove the degeneracy, we thus redefine $\mathbf{M}$ and $\mathbf{N}$ at the end of training, from the SVD of the matrix $\hat{\mathbf{M}}\hat{\mathbf{N}}^\top$, where $\hat{\mathbf{M}}$ and $\hat{\mathbf{N}}$ are the results of training. Hence, the vectors $\{\boldsymbol{m}^{(r)}\}$ and $\{\boldsymbol{n}^{(r)}\}$ used in the theoretical analyses are defined as the columns of these final matrices, and are both sets of orthogonal vectors[2].

Within this framework, methods to train low-rank networks do not pose further difficulties and we will in the following chapter focus on reverse-engineering trained networks. The reader interested in training procedures and tricks can however refer to appendix A.4.

[2] A degeneracy remains in two situations: first the vectors $\boldsymbol{m}^{(r)}$ and $\boldsymbol{n}^{(r)}$ are defined up to a multiplication by $-1$ of both vectors at the same time. Second, if several singular values are identical, the associated singular vectors can lie anywhere along a certain subspace (this situation appears for example in the ring attractor networks, Fig. II.3g). This however does not impact the results of the analyses.

## Summary of Chapter 3

Neural computations are currently investigated using two separate approaches: sorting neurons into functional sub-populations, or examining the low-dimensional dynamics of collective activity. Whether and how these two aspects interact to shape computations is currently unclear. Using a novel approach to extract computational mechanisms from networks trained on neuroscience tasks, here we show that the dimensionality of the dynamics and sub-population structure play fundamentally complementary roles. While various tasks can be implemented by increasing the dimensionality in networks with fully random population structure, flexible input-output mappings instead require a non-random population structure that can be described in terms of multiple sub-populations. Our analyses revealed that such a sub-population structure enables flexible computations through a mechanism based on gain-controlled modulations that flexibly shape the collective dynamics. Our results lead to task-specific predictions for the structure of neural selectivity, inactivation experiments, and for the implication of different neurons in multi-tasking.

This chapter is based on the manuscript *The role of population structure in computations through neural dynamics*, Alexis Dubreuil*, Adrian Valente*, Manuel Beiran, Francesca Mastrogiuseppe, Srdjan Ostojic, *Nature Neuroscience*, 25, 783-794 (2022). Accompanying code is also available at the address `https://github.com/adrian-valente/populations_paper_code`.

# The role of population structure in computations through neural dynamics III

## III.1 Introduction

The quest to understand the neural bases of cognition currently relies on two disjoint paradigms (6). Classical works have sought to determine the computational role of individual cells by sorting them into functional populations based on their responses to sensory and behavioral variables (84; 129; 76). Fast developing tools for dissecting neural circuits have opened the possibility of mapping such functional populations onto genetic and anatomic cell types, and given a new momentum to this cell-category approach (1; 230; 101; 74; 145; 81; 82). This viewpoint has however been challenged by observations that individual neurons often represent seemingly random mixtures of sensory and behavioral variables, especially in higher cortical areas (32; 113; 155; 118; 139), where sharply defined functional cell populations are often not directly apparent (118; 152; 76). A newly emerging paradigm has therefore proposed that neural computations need instead to be interpreted in terms of collective dynamics in the state space of joint activity of all neurons (23; 155; 118; 64; 153; 173). This computation-through-dynamics framework (217) hence posits that neural computations are revealed by studying the geometry of low-dimensional trajectories of activity in state space (118; 150; 27; 153; 219; 193), while remaining agnostic to the role of any underlying population structure.

In view of the apparent antagonism between these two approaches, two works have sought to precisely assess the presence of functional cell populations in the posterior parietal cortex (PPC) (152) and prefrontal cortex (81). Rather than define cell populations by classical methods such as thresholding the activity or selectivity of individual neurons, these studies developed new statistical techniques to determine whether the distribution of selectivity across neurons displayed a non-random population structure (76). Using analogous analyses, but different behavioral tasks, the two studies reached opposite conclusions. Raposo et al found no evidence for non-random population structure in selectivity, and argued that PPC neurons fully multiplex information. Hirokawa et al also observed that individual neurons responded to mixtures of task features, but in contrast to Raposo et al, they detected important deviations from a fully random distribution of selectivity, a situation they termed *non-random mixed selectivity*. By clustering neurons according to their response properties, they defined separate, though mixed-selective populations that appeared to represent distinct task variables and to reflect underlying connectivity. To resolve the apparent discrepancy, Hirokawa et al conjectured that revealing non-random population structure in higher cortical areas may require sufficiently complex behavioral tasks.

These conflicting findings therefore raise a fundamental theoretical question: do specific computational tasks require a non-random population structure, or alternatively can any task in principle be implemented with a fully random population structure as in (152)? To address this question, we trained recurrent neural networks on a range of systems neuroscience tasks (203; 7; 227) and examined the population structure that emerged in both selectivity and connectivity using identical methods as (152; 81). Starting from the premise that computations are necessarily determined by the underlying connectivity (122), we then developed a new approach for assessing the computational role of population structure in connectivity for each task. Together, these analyses revealed that, while a fully random population structure was sufficient to implement a range of tasks, specific tasks required a non-random population structure *in connectivity* that could be described in terms of a small number of statistically-defined subpopulations. This was in particular the case when a flexible reconfiguration of input-output associations was needed, a common component of many cognitive tasks (165) and more generally of multi-tasking (227; 45; 120). To extract the mechanistic role of this population structure for computations-through-dynamics, we focused on the class of low-rank models (122; 180; 181) that can be reduced to interpretable latent dynamics characterized by a minimal intrinsic dimension and number of subpopulations (11). We found that the subpopulation structure of the connectivity enables networks to implement flexible computations through a mechanism based on gain modulations(187; 53) of effective interactions between latent variables. Our results lead to task-specific predictions for the statistical structure of single-neuron selectivity, for inactivations of specific subpopulations, as well as for the implication of different neurons in multi-tasking.

## III.2 Identifying non-random population structure in trained networks

We trained recurrent neural networks (RNNs) on five systems neuroscience tasks (227; 229) spanning a range of cognitive components: perceptual decision-making (DM) (71), parametric working-memory (WM) (157), multi-sensory decision-making (MDM) (152), contextual decision-making (CDM) (118) and delay-match-to-sample (DMS) (126). We then searched for evidence of non-random population structure by comparing the selectivity, connectivity and performance of the trained networks with randomized shuffles.

We first asked if training on each task led to the emergence of non-random structure in selectivity. Following (Raposo et al. 2014) (152) and (Hirokawa et al 2019) (81), we represented each neuron as a point in a *selectivity space*, where each axis was given by the linear regression coefficient of neural firing rate with respect to a task variable such as stimulus, decision or context (Fig. III.1a). The dimension of the selectivity space ranged from 2 to 4 depending on the task (see Methods), and each trained network led to a distribution of points in that space (Fig. III.1b). For each network, we used the ePAIRS (152; 81) test to compare the obtained distribution with a randomized shuffle corresponding to a multivariate Gaussian (Fig. III.1b,c). A non-significant outcome suggests an isotropic distribution of single-neuron selectivity, a situation that has been denoted as non-categorical mixed selectivity (152) and we refer to it as*fully-random population structure*. A statistically significant outcome instead indicates that neurons tend to be clustered along multiple axes of the selectivity space. Following (152; 81), we refer to this situation as non-random mixed selectivity, or *non-random population structure*. The ePAIRS analysis revealed the presence of non-random population structure for two out of the five tasks, the contextual decision-making and delay-match-to-sample tasks (Fig. III.1d) (proportion of statistically significant networks under the ePAIRS test, $p < 0.05$, Bonferroni corrected : DM 1/100, WM 6/100, MDM 10/100, CDM 87/100, DMS 100/100, A.1). In particular, we found a clear difference between the multi-sensory (152) and context-dependent (118) decision making tasks, which had an identical input structure and therefore selectivity spaces of identical

dimensions, but required different mappings from inputs to outputs.

The selectivity in trained RNNs necessarily reflects the underlying connectivity (122). We therefore next sought to determine the presence of non-random population structure directly in the connectivity of trained networks by applying an analogous analysis in a *connectivity space*. To define a connectivity space with a minimal number of parameters, we focused on RNNs constrained to have recurrent connectivity matrices $J_{ij}$ of a fixed rank $R$, a type of connectivity structure that typically emerges when training RNNs on simple tasks (181). A matrix of rank $R$ can be written as

$$J_{ij} = m_i^{(1)} n_j^{(1)} + \cdots + m_i^{(R)} n_j^{(R)}, \tag{III.1}$$

so that neuron $i$ is characterized by $2R$ recurrent connectivity parameters $\{m_i^{(r)}, n_i^{(r)}\}_{r=1\ldots R}$, as well as $N_{in}$ input weights $I_i^{(s)}$ and a readout weight $w_i$ (see Methods). For each task, we determined the minimal required rank $R$ (A.3). We then represented the connectivity of each neuron as a point in a $(2R + N_{in} + 1)$-dimensional *connectivity space*, and described the connectivity of a full network as the corresponding distribution of points (Fig. III.1e,f). Similarly to the selectivity analysis, we assessed the presence of non-random population structure by comparing connectivity distributions of trained networks with randomized shuffles corresponding to multivariate Gaussians with matching empirical means and covariances. The results were consistent with the analysis of selectivity (Fig. III.1g,h), and showed a gap between the same two groups of tasks (Fig. III.1h, number of networks with statistically significant clustering for each task: DM 3/100; WM 5/100; MDM 1/100; CDM 100/100; DMS 100/100; $p < 0.05$ with Bonferroni correction). In particular the MDM and CDM tasks again led to opposite results although their connectivity spaces were identical.

The analyses of selectivity and connectivity are purely correlational, and do not allow us to infer a causal role of the observed structure. To determine when non-random population structure is computationally necessary, or conversely when random population structure is computationally sufficient, we therefore developed a new *resampling* analysis. For each task, we first generated new networks by sampling the connectivity parameters of each neuron from the randomized distribution used to assess structure in Fig. III.1e-h, *i.e.* a multivariate Gaussian distribution with mean and covariance matching the trained low-rank RNNs. This procedure preserved the rank of the connectivity (Fig. III.1e), and the overall correlation structure of connectivity parameters, but scrambled any non-random population structure (Fig. III.1j,k). We then quantified the performance of each randomly resampled network on the original task. This key analysis revealed that the randomly resampled networks led to a near perfect accuracy for the DM, WM and MDM tasks, but not for the CDM and DMS tasks (Fig. III.1l). This demonstrates that, on one hand, random population structure is sufficient to implement the DM, WM and MDM tasks, while on the other hand non-random population structure is necessary for CDM and DMS tasks. These results held independently of the constraints on the rank of the connectivity, and in particular for unconstrained, full-rank networks in which only the learned part of the connectivity was resampled (A.4).

It is important to stress that the performance of resampled networks is a much more direct assessment of the computational role of the non-random population structure than the analyses of selectivity and connectivity through the ePAIRS test. Indeed, the ePAIRS analyses can lead to false positives in which statistically significant non-random structure is found in both selectivity and connectivity although resampled networks with a single Gaussian still match the performance of the trained network (Sup Fig. A.2). As an illustration, networks trained on the DM task sometimes exhibited two diametrically opposed clusters in the connectivity space, suggesting two concurrent pools of self-excitatory populations, reminiscent of solutions previously found for this task (220; 224; 175). Generating resampled networks scrambled that structure, but still led to functioning networks, which showed that in the DM task the population structure does not bear an essential computational role, and

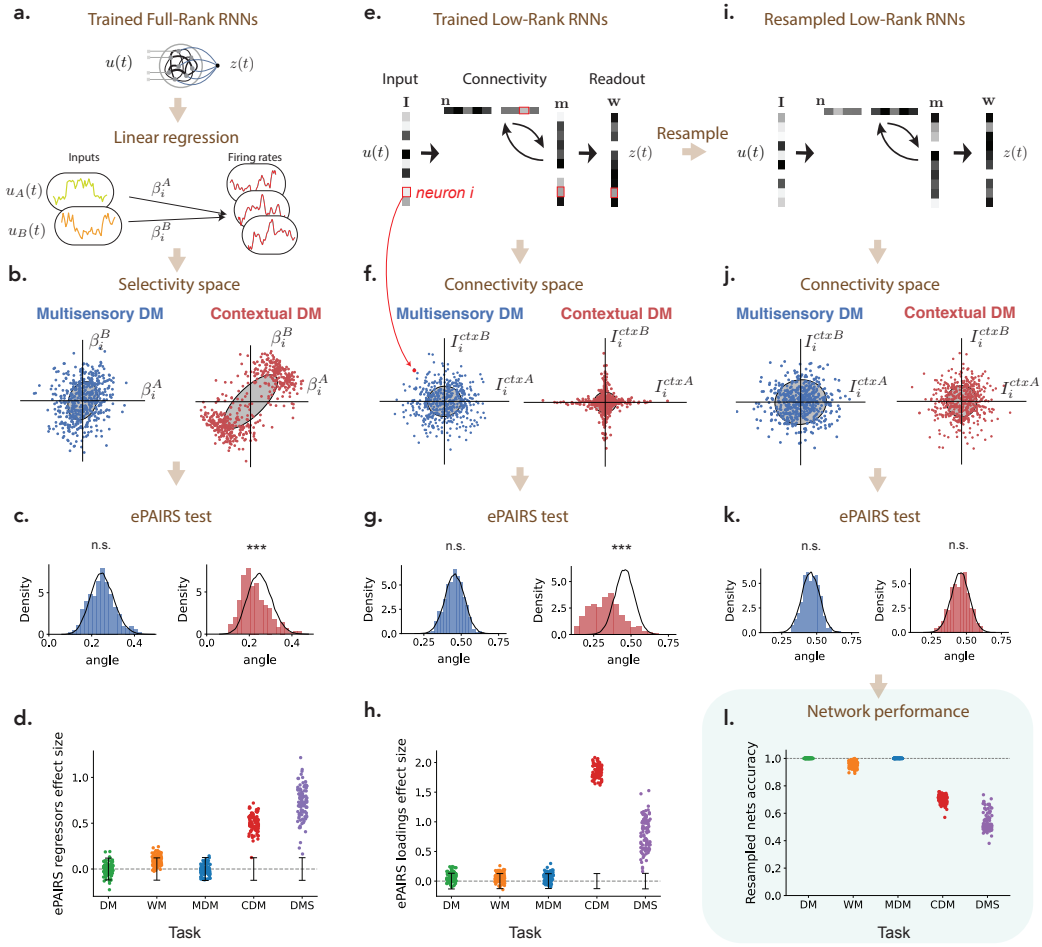


Figure III.1: Identifying non-random population structure in selectivity, connectivity and computations. (a) Recurrent neural networks (RNNs) were trained separately on five tasks. For each task, and each trained RNN, selectivity was first quantified by computing linear regression coefficients $\beta_i^{var}$ for each neuron $i$ with respect to task-defined variables such as stimulus features or decision (see Methods A.3.4). Each neuron was then represented as a point in a selectivity space where each axis corresponds to the regression coefficient with respect to one variable. For each network, we then compared the resulting distribution of points with a random shuffle corresponding to a multivariate Gaussian with matching empirical covariance. (b) Illustration of the distribution of regression coefficients in selectivity space for two networks trained on respectively the multi-sensory (MDM) and context-dependent decision-making (CDM) tasks which received identical inputs (two stimuli $A$ and $B$ and two contextual cues) but required different outputs. The full selectivity space was four dimensional. The plots show two-dimensional projections of the selectivity distribution onto the plane defined by regression coefficients with respect to stimuli $A$ and $B$. Gray ellipses correspond to the 1 s.d. ellipse of a Gaussian distribution with matching mean and covariance. (c) Distribution of angles between each point and its nearest neighbor in the selectivity space illustrated in panel b (colored histograms), compared with that of a matching multivariate Gaussian (null distribution, black line). The mismatch between the two distributions was quantified using the ePAIRS test (152; 81) (two-sided, see Methods A.3.5). The mismatch was significant for the CDM task ($p = 3 \times 10^{-25}$, effect size $c = 0.58$ $n = 512$ neurons; ***: $p < 0.001$), but not for the MDM task ($p = 0.61$, $c = 0.01$, $n = 512$).

Figure III.1 *(previous page)*: (d) Population structure in the selectivity space across networks and tasks: effect size of the ePAIRS test on the selectivity space for 100 networks trained on each of the tasks (see A.1 for p-values). Black bars represent 95% confidence intervals for null distributions, centered around mean null effect size. (e) To assess for population structure in connectivity, we focused on low-rank networks, where connectivity is fully specified by vectors over neurons (122). Each neuron is characterized by one parameter on each vector (illustrated in grayscale, entries for a specific neuron are outlined in red), and can be represented as a point in a connectivity space where each axis corresponds to the parameters on one vector. We assessed the presence of non-random population structure in that space using a procedure identical to the analysis of selectivity (c-d). (f) Illustration of the distribution of parameters in connectivity space for the two networks trained on respectively the MDM and CDM tasks. For these tasks, minimal trained networks were of rank $R = 1$ (A.3), so that the connectivity space was of dimension 7 (four inputs, two recurrent vectors and one readout). The plots show two-dimensional projections of the full connectivity distribution onto the plane defined by input parameters of contextual cues $A$ and $B$. (g) Comparison of nearest-neighbor angle distributions in connectivity space for trained networks and the randomized shuffles as in c. The difference is significant for the CDM task ($p = 2 \times 10^{-142}$, $c = 1.89$, $n = 512$), but not for the MDM task ($p = 0.72$, $c = 0.005$, $n = 512$). (h) Population structure in the connectivity space across networks and tasks: effect size of the ePAIRS test on the connectivity space for 100 networks trained on each of the five studied tasks (see A.1 for p-values). (i) To identify the causal role of population structure on computations, we randomly generated new networks by resampling from the null distribution in connectivity space that preserved the mean and covariance structure but scrambled any non-random population structure. (j-k) In randomly resampled networks, the statistics of connectivity are by design identical to shuffles used for the ePAIRS test (MDM: $p = 0.08$, $c = 0.05$, $n = 512$ ; CDM: $p = 0.68$, $c = -0.01$, $n = 512$). (l) Performance of each randomly resampled network on its corresponding task as measured by accuracy.

might be an artifact of specific training parameters. Spurious structure can also appear in selectivity when the non-linearity is strongly engaged (Sup Fig. A.2).

In summary, our analyses of trained recurrent neural networks revealed that certain tasks can be implemented with a fully-random population structure in both connectivity and selectivity, while others appeared to require additional organization in the connectivity that led to non-random structure in selectivity. We next sought to understand the mechanisms by which the population structure of connectivity determines the dynamics and the resulting computations. In a first step, we examined the situation in which the population structure is fully random. In a second step, we asked whether non-random population structure in the connectivity space could be represented in terms of separate clusters or subpopulations, and how this additional organization expands the computational capabilities of the network.

## III.3 Interpreting computations in terms of latent dynamics

To unravel the mechanisms by which population structure impacts computations, we developed a method for interpreting low-rank networks in terms of underlying low-dimensional dynamics (217; 11). Here we first outline this general model reduction approach (Fig. II.1), and next apply it to trained recurrent networks.

In line with methods for analyzing large-scale neural activity (23; 36; 64; 173), we represented the dynamics as trajectories $\boldsymbol{x}(t) = \{x_i(t)\}_{i=1...N}$ in the *activity state space*, where each dimension corresponds to the activation of one neuron (Fig. II.1b). As in dimensionality reduction analyses, we then parametrized these trajectories by a small

number of latent variables (36; 64). Crucially, for low-rank networks this dimensionality reduction is exact, because the connectivity structure directly restricts the dynamics to lie in a low-dimensional subspace (11). Specifically, $\boldsymbol{x}(t)$ can be decomposed into a set of internal variables $\kappa_r(t)$ and inputs $u_s(t)$ that respectively quantify activity along recurrent and input-driven directions $\boldsymbol{m}^{(r)}$ and $\boldsymbol{I}^{(s)}$ in state-space (219), where $\boldsymbol{m}^{(r)}$ and $\boldsymbol{I}^{(s)}$ are *connectivity and input vectors* obtained by grouping connectivity parameters across neurons (see Fig. II.1b and Eq. II.11 ). A mathematical analysis of network dynamics then shows that the set of internal variables $\boldsymbol{\kappa} = \{\kappa_r\}_{r=1...R}$ forms a dynamical system driven by inputs $\boldsymbol{u} = \{u_s\}_{s=1...N_{in}}$, with a temporal evolution given by

$$\frac{d}{dt}\boldsymbol{\kappa}(t) = F(\boldsymbol{\kappa}(t), \boldsymbol{u}(t)) \tag{III.2}$$

where $F$ is a non-linear function that determines the amount of change of $\boldsymbol{\kappa}$ at every time step. In the limit of large networks, the precise shape of $F$ is set by the statistics of the connectivity parameters across neurons (Eq. II.21), *i.e.* precisely the distribution of points in the connectivity space that we previously examined in Fig. III.1f. The connectivity can therefore be interpreted in two complementary ways, either in terms of directions in the activity state-space (Fig. II.1b top left) or in terms of distributions in the connectivity space (Fig. II.1b bottom left) and these two representations together determine the low-dimensional latent dynamics.

In summary, in line with the computation-through-dynamics framework (153; 217), low-rank networks can be exactly reduced to low-dimensional, non-linear latent dynamical systems which determine the performed computations. We next examined how the population structure in trained recurrent networks impacts the resulting latent dynamical system.

## III.4 Latent dynamics for fully random population structure

Our resampling analyses of trained RNNs revealed that a range of tasks could be performed by networks in which the population structure was fully random in connectivity space (Fig. III.1l). We therefore first examined the latent dynamics underlying computations in that situation. Crucially, a fully random population structure limits the available parameter space, and strongly constrains the set of achievable latent dynamics independently of their dimensionality (11) (see Chapter II). We start by specifying these constraints on the dynamics, and show they nevertheless allow networks with random population structure to implement a range of tasks of increasing complexity by increasing the rank of the connectivity and therefore the dimensionality of the dynamics.

Networks with fully random population structure were defined in Fig. III.1i-l as having distributions of connectivity parameters computationally equivalent to a Gaussian distribution. In such networks, the statistics of connectivity are fully characterized by a set of covariances between connectivity parameters, each of which can be directly interpreted as the alignment, or overlap between two connectivity vectors (Fig. II.1b bottom left, see Eq. II.25). For this type of connectivity, a mean-field analysis shows that the latent low-dimensional dynamics can be directly reduced to an effective latent circuit, where internal variables $\kappa_r$ integrate external inputs $u_s$, and interact with each other through *effective couplings* set by the overlaps between connectivity vectors multiplied by a common, activity-dependent gain factor (11). In such reduced models, the role of individual parameters can then be analyzed in detail.

As a concrete example, a unit-rank network ($R = 1$) with connectivity vectors $\boldsymbol{m}$ and $\boldsymbol{n}$ and a single feed-forward input vector $\boldsymbol{I}$ ($N_{in} = 1$) leads to two-dimensional activity, fully described by a single internal variable $\kappa(t)$ and a single external variable $u(t)$ (Fig. II.1b). The latent dynamics of $\kappa(t)$ are given by

$$\tau \frac{d\kappa}{dt} = -\kappa + \tilde{\sigma}_{nm}\kappa + \tilde{\sigma}_{nI}u(t), \tag{III.3}$$

where $\tilde{\sigma}_{nm}$ and $\tilde{\sigma}_{nI}$ are effective couplings defined as $\tilde{\sigma}_{nm} = \langle\Phi'\rangle\sigma_{nm}$ and $\tilde{\sigma}_{nI} = \langle\Phi'\rangle\sigma_{nI}$, where $\sigma_{nm}$ (resp. $\sigma_{nI}$) is the fixed overlap between the vector $\boldsymbol{n}$ and the vector $\boldsymbol{m}$ (resp. $\boldsymbol{I}$). The connectivity vector $\boldsymbol{n}$ therefore selects inputs to the latent dynamics (122): the overlap between $\boldsymbol{n}$ and $\boldsymbol{I}$ controls how strongly the latent dynamics integrate feed-forward inputs, while the overlap between $\boldsymbol{n}$ and $\boldsymbol{m}$ controls the strength of positive feedback in the latent dynamics. Crucially, all the effective couplings are scaled by the same factor $\langle\Phi'\rangle$ that represents the average gain of all neurons in the network. This gain depends on the activity in the network (see Eq. II.30), which makes the dynamics in Eq. III.3 non-linear. The fact that all the effective couplings are scaled by the same factor however implies that, in networks with a fully random population structure, the overall form of the effective circuit is fixed by the connectivity overlaps, and this strongly limits the range of possible dynamics for the internal variables (11). Tasks for which a fully random population structure is sufficient are therefore those that can be implemented by a fixed effective circuit at the level of latent dynamics.

We first applied our model reduction framework (Fig.II.1b) to the perceptual decision making task, where a network received a noisy scalar stimulus $u(t)$ along a random input vector, and was trained to report the sign of its temporal average along a random readout vector (Fig. III.2a). Minimizing the rank of the trained recurrent connectivity matrix, we found that a unit-rank network was sufficient to solve the task (Sup Fig. A.3). The network connectivity was fully characterized by four connectivity vectors: the input vector $\boldsymbol{I}$, recurrent connectivity vectors $\boldsymbol{n}$ and $\boldsymbol{m}$, and the readout vector $\boldsymbol{w}$ (Fig. III.2a). As a result, the activity $\boldsymbol{x}(t)$ evolved in a two-dimensional plane spanned by $\boldsymbol{I}$ and $\boldsymbol{m}$, and was fully described by two corresponding collective variables $u(t)$ and $\kappa(t)$ (Fig. III.2c). The resampling analysis in Fig. III.1l showed that trained networks were fully specified by the overlaps, or covariances between connectivity vectors, as generating new networks by sampling connectivity from a Gaussian distribution with identical covariances led to identical performance. The latent dynamics of $\kappa(t)$ could then be reduced to a simple effective circuit (Fig. III.2d, Eq. III.3). Inspecting the values of covariances in the trained networks (Fig. III.2b, Sup Fig. A.7) and analyzing the effective circuit revealed that the latent dynamics relied on a strong overlap $\sigma_{nI}$ to integrate inputs, and an overlap $\sigma_{nm} \approx 1$ to generate a long integration timescale via positive feedback. The internal variable $\kappa(t)$ therefore represented integrated evidence along a direction in state space determined by the connectivity vector $\boldsymbol{m}$ (Fig. III.2c,d). The readout vector $\boldsymbol{w}$ was aligned with $\boldsymbol{m}$, so that the output directly corresponded to integrated evidence $\kappa(t)$. Controlling only three parameters in the latent dynamics was sufficient to reproduce the psychometric input-output curve of the full trained network (Fig. III.2e). Note that this network implementation is very similar to the implementation that has been proposed in previous work without making use of a learning algorithm (122). The findings from the perceptual decision task directly extended to the multi-sensory decision-making task (152), in which the latent dynamics were identical, but integrated two inputs corresponding to two different stimulus features.

We next turned to the parametric working memory task (157), where two scalar stimuli $f_1$ and $f_2$ were successively presented along an identical input vector $\boldsymbol{I}$, and the network was trained to report the difference $f_1 - f_2$ between the values of the two stimuli (Fig. III.2f). We found that this task required rank $R = 2$ recurrent connectivity (Sup Fig. A.3), so that the activity was constrained to the three-dimensional space spanned by $\boldsymbol{I}$ and the connectivity vectors $\boldsymbol{m}^{(1)}$ and $\boldsymbol{m}^{(2)}$. The low-dimensional dynamics could therefore be described by two internal variables $\kappa_1(t)$ and $\kappa_2(t)$ that represented activity along $\boldsymbol{m}^{(1)}$ and $\boldsymbol{m}^{(2)}$, and formed a two-dimensional dynamical system that integrated the input $u(t)$ received along $\boldsymbol{I}$. The resampling analysis indicated that in this case also the trained connectivity was fully specified by covariances between connectivity vectors (Fig. III.1i-l). Inspecting the

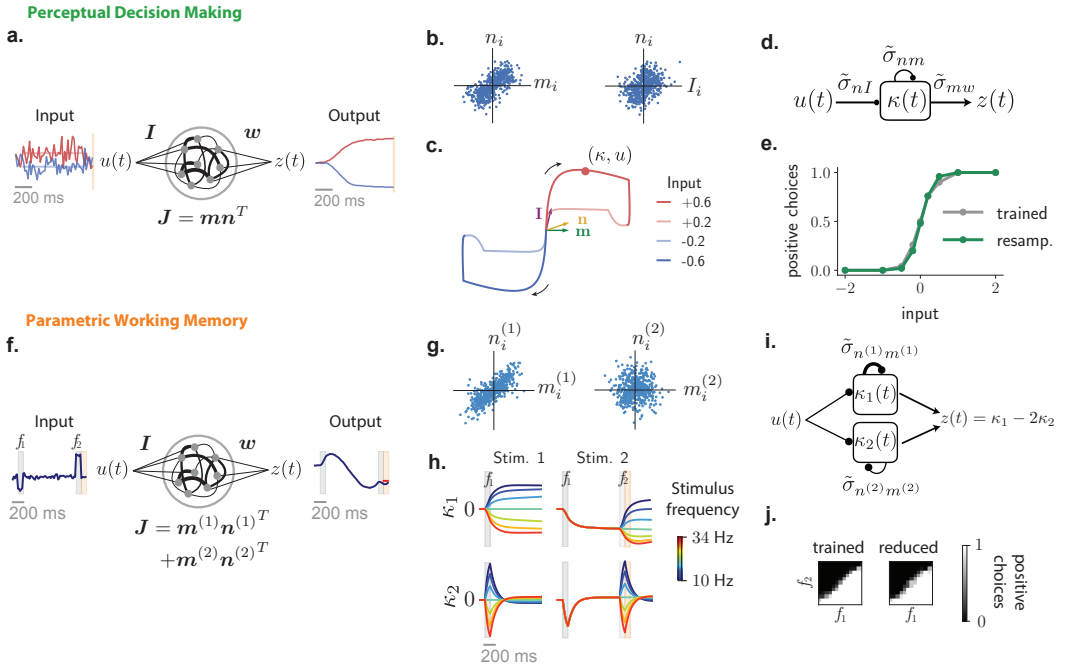


Figure III.2: Low-dimensional latent dynamics in networks with a random population structure. (a)-(e) Perceptual decision making task. (a) A rank-one network was trained to output the sign of the mean of a noisy input signal. Two example trials for a positive (red) and a negative (blue) input mean. (b) Two two-dimensional projections of the obtained four-dimensional connectivity space. Each point represents the connectivity parameters of one neuron. (c) Low-dimensional trajectories in the two-dimensional subspace spanned by vectors $\boldsymbol{m}$ and $\boldsymbol{I}$ for four trials. (d) The latent dynamics are equivalent to an effective circuit governed by 2 effective couplings (Eq. III.3), which are determined by the overlaps $\sigma_{nI}$ and $\sigma_{nm}$ of the vector $\boldsymbol{n}$ with $\boldsymbol{I}$ and $\boldsymbol{m}$ (see vectors in panel c). The readout from the network is set by the overlap $\sigma_{mw}$ between the vectors $\boldsymbol{m}$ and $\boldsymbol{w}$. (e) Psychometric function showing the fraction of positive outputs for the trained network, and for a reduced network generated by controlling only three parameters corresponding to the effective couplings in f (see Supplementary Note 2.1). (f)-(j) Parametric working memory task. (f) A rank-two network was trained to compute the difference between two stimuli $f_1$ and $f_2$ separated by a variable delay. (g) Two projections of the obtained six-dimensional connectivity space. (h) Since the network is rank-two, the recurrent activity is parametrized by two internal variables, $\kappa_1$ and $\kappa_2$ that correspond to activity along connectivity vectors $\boldsymbol{m}^{(1)}$ and $\boldsymbol{m}^{(2)}$. The variable $\kappa_1$ acts as an integrator that encodes the stimuli persistently: it encodes $f_1$ following the first stimulus, and $f_1 + f_2$ following the second one. The variable $\kappa_2$ responds transiently to each stimulus, and therefore encodes $f_2$ at the decision time. (i) The latent dynamics are described by an effective circuit where the two internal variables evolve independently, with different amounts of positive feedback. (j) Psychometric response matrix for the trained network, and a reduced network generated by controlling only six parameters corresponding to the effective couplings in i (see Supplementary Note 2.2). Each matrix displays the fraction of positive responses for each combination of stimuli $f_1$ and $f_2$.

connectivity distribution (Fig. III.2g, Sup Fig. A.7) revealed that the two internal variables $\kappa_1$ and $\kappa_2$ did not directly interact, but instead independently integrated stimuli through dynamics given by Eq. III.3 (Fig. III.2i). For $\kappa_1$, a strong overlap $\sigma_{n^{(1)}m^{(1)}}$ led to strong positive feedback that generated a persistent representation of the intensity $f_1$ of the first stimulus along the direction of state space set by the connectivity vector $\boldsymbol{m}^{(1)}$ (Fig. III.2h top). For $\kappa_2$, the overlap $\sigma_{n^{(2)}m^{(2)}}$, and therefore the positive feedback, was weaker, leading to a transient response that encoded the most recent stimulus along the direction $\boldsymbol{m}^{(2)}$ in the state space (Fig. III.2h bottom). The readout vector $\boldsymbol{w}$ was aligned with both $\boldsymbol{m}^{(1)}$ and $\boldsymbol{m}^{(2)}$, but with overlaps of opposite signs, so that the output of the network in the decision period corresponded to the difference between $\kappa_1$ and $2\kappa_2$, and therefore effectively $f_2 - f_1$ (Fig. III.2i). Controlling only five parameters in the latent dynamics was therefore sufficient to reproduce the psychometric matrix describing the input-output mapping of the full trained network (Fig. III.2j).

In summary, networks with random population structure can perform tasks of increasing complexity by relying on the dimensionality of recurrent dynamics to represent an increasing number of task-relevant latent variables. The random population structure however limits ways in which these latent variables can be combined by fixing the shape of the equivalent circuit. As a consequence, for more complex tasks a fully random population structure was not sufficient. We next sought to further elucidate this aspect.

## III.5 Representing non-random structure with multiple populations

The resampling analysis in Fig. III.1l indicated that tasks such as context-dependent decision-making and delayed-match-to-sample relied on a population structure in connectivity that was not fully random. To better understand the underlying structure and its computational role, we further examined RNNs trained on these two tasks, and asked whether their connectivity could be represented in terms of multiple populations. We first examined whether a multi-population connectivity structure is sufficient to implement the two tasks, and in a second step examined how such a structure modifies latent dynamics and expands their computational capacity.

To identify computationally-relevant populations, we took inspiration from (81), and first performed clustering analyses in the connectivity space where non-random population structure was found (Fig. II.2a). Applying a Gaussian mixture clustering algorithm on the cloud of points formed by each trained network, we partitioned the neurons into separate subpopulations. In the trained networks, all clusters were centered close to the origin, but each had a different shape and orientation that corresponded to multiple peaks in the distribution detected by the ePAIRS analysis (Fig. III.1f-g). Each population was therefore characterized by a different set of covariances, or overlaps, between input, recurrent, and output connectivity vectors. We then extended our resampling approach from Fig. III.1i-l, and generated new networks by first randomly assigning each neuron to a population, and then sampling its connectivity parameters from a Gaussian distribution with the fitted covariance structure. Finally, we inspected the performance of these randomly generated networks, and compared them with fully trained ones. By progressively increasing the number of fitted clusters, we determined the minimal number of populations needed to implement the task (see Methods). Within this approach, networks with a fully random population structure such as those described in Fig. III.2 correspond to a single overall population in connectivity space.

We first considered context-dependent decision making, where stimuli consisted of a combination of two scalar features that fluctuated in time (118). Depending on a contextual cue, only one of the two features needed to be integrated (Fig. III.3a), so that the same stimulus could require opposite responses, a hallmark of flexible input-output transformations

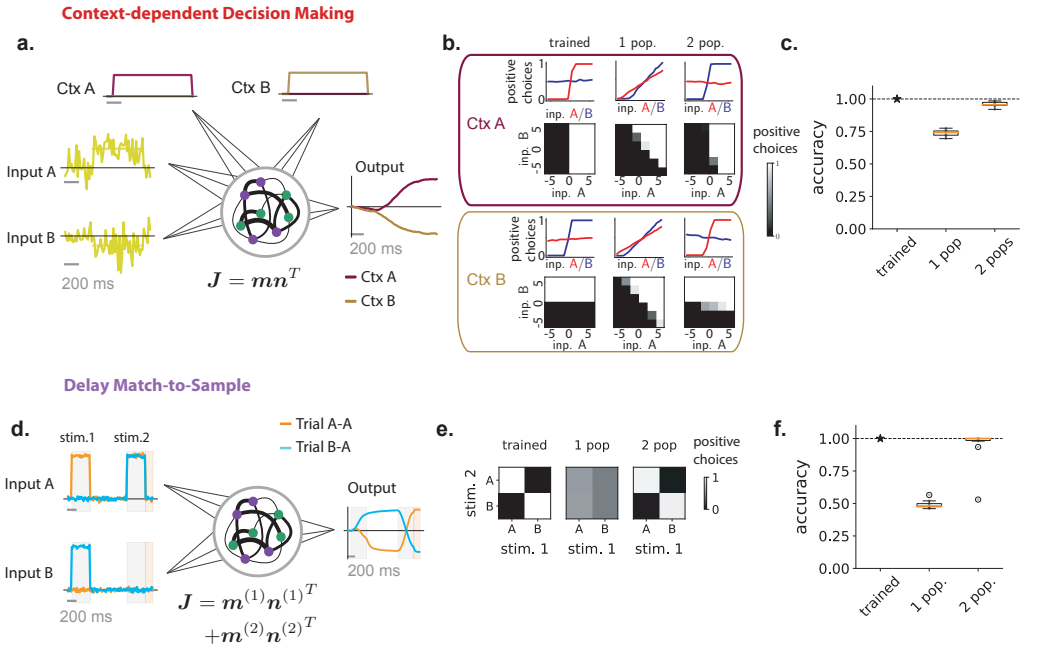


Figure III.3: Multi-population connectivity structure captures the computational requirements for context-dependent tasks. (a)-(c) Application to the context-dependent decision making task. (a) Networks received stimulus inputs consisting of two noisy features along two different input vectors, together with one of two contextual cues in each trial. Unit-rank networks were trained to output the sign of the mean of the cued feature. Here we illustrate two example trials sharing the same stimulus inputs and opposite contextual cues (context A activated in dark red, context B in pale brown), leading to opposite outputs. (b) Psychometric functions and response matrices. Each psychometric matrix displays the fraction of positive responses for each combination of stimulus features. Each psychometric function represents the fraction of positive responses for the value of one feature, averaging over the other. The two rows show psychometric functions and matrices in different contexts, for a trained network (left column), and for networks generated by resampling connectivity from a single population (middle column) or two subpopulations (right column). (c) Average accuracy of a trained network and for 10 draws of resampled single-population and two-population networks (line: median, box: quartiles, whiskers: range, in the limit of median $\pm$ 1.5 interquartile range, points: outliers). (d)-(f) Application to networks performing the delayed match-to-sample (DMS) task. (d) Networks received a sequence of two stimuli during two stimulation periods (in light gray) separated by a delay. Each stimulus belonged to one out two categories (A or B), each represented by a different input vector. Rank-two networks were trained to produce an output during a response period (in light orange) with a positive value if the two stimuli were identical, and a negative value otherwise. Here we illustrate two trials with stimuli A-A and B-A respectively. (e) Psychometric response matrices. Fraction of positive responses for each combination of first and second stimuli, for a trained network (left) and for networks generated by resampling connectivity from a single population (middle) or two populations (right). (f) Average accuracy of a trained network and for 10 draws of resampled single-population and two-population networks.

(60). We found that unit-rank connectivity was sufficient (Sup Fig. A.3), and focused on such networks. The analysis in Fig. III.1l showed that generating networks by resampling connectivity from a single, fully-random population led to a strong degradation of the performance, although it remained above chance. A closer inspection of psychometric matrices representing input-output transforms in different contexts revealed that the resampled single-population networks in fact generated correct responses for stimuli requiring identical outputs in the two contexts, but failed for incongruent stimuli, for which responses needed to be flipped according to context (Fig. III.3b). This observation was not specific to unit-rank networks, as randomizing population structure in higher-rank (Sup Fig. A.5) and full-rank networks (Sup Fig. A.4) led to a similar reduction in performance. We therefore performed a clustering analysis in the connectivity space. The number of clusters varied across networks (Sup Fig. A.6 and appendix A.2), but the minimal required number was two. For such minimal networks, we found that randomly resampling from the corresponding Gaussian mixture distribution led to an accuracy close to the original trained connectivity (Fig. III.3c). In particular, the randomly generated networks correctly switched their response to incongruent stimuli across contexts, in contrast to networks with a single population (Fig. III.3b). This indicated that connectivity based on a structure in two populations was sufficient to implement the context-dependent decision-making task.

We next turned to the delayed-match-to-sample task (126; 50; 27), where two stimuli were interleaved by a variable delay period, and the network was trained to indicate in each trial whether the two stimuli were identical or different (Fig. III.3d). This task involved flexible stimulus processing analogous to the context-dependent decision-making task because an identical stimulus presented in the second position required opposite responses depending on the stimulus presented in the first position. We found that this task required a rank two connectivity (Sup Fig. A.3), but, similarly to the context-dependent decision making task, a fully random population structure was not sufficient to perform the task, as networks generated by randomizing connectivity parameters reduced the output to chance level (Fig. III.1l, III.3e,f). Fitting instead two clusters in the connectivity space showed that two subpopulations were sufficient, as sampling connectivity based on a two-population structure led to networks with a performance close to that of the fully trained network (Fig. III.3f).

Altogether, our analyses based on clustering connectivity parameters, and randomly generating networks from the obtained multi-population distributions, indicated that connectivity distributions described by a small number of populations were sufficient to implement tasks requiring flexible input-output mappings. To identify the mechanistic role of this multi-population structure, we next examined how it impacted the latent dynamics implemented by trained networks.

## III.6 Gain modulation of latent dynamics

To unveil the mechanisms underlying flexible input-output mappings in networks with connectivity based on multiple populations, we examined how such a structure impacts the latent dynamics of internal variables. Here we first describe how, in contrast to a single-population, a multi-population structure allows external inputs to flexibly modulate the overall form of the circuit describing latent dynamics. We then show how this general principle applies specifically to the two flexible tasks (Fig. III.3). We focus here on networks with minimal rank and minimal number of populations, and show in the next section that the inferred predictions hold more generally.

In Fig. III.3 we defined subpopulations as subsets of neurons characterized by different overlaps between input, recurrent and output connectivity vectors in a network of fixed rank. In a network with a multi-population structure, the number of internal variables describing low-dimensional dynamics is determined by the rank of the recurrent connectivity, as in networks without population structure (Fig. II.1a). Remarkably, a mean-field analysis

(11) (see previous chapter) shows that the latent low-dimensional dynamics can still be represented in terms of an effective circuit where internal variables $\kappa_r$ integrate inputs and interact with each other through effective couplings (Fig. III.4a). The key effect of the multi-population structure is however to modify the form of the effective couplings and endow them with much greater flexibility than in the case of a single, fully random population. Indeed, in a network with a single population, the effective couplings were given by connectivity overlaps multiplied by a single, global gain factor, and modulating the gain therefore scaled all effective couplings together. In contrast, in networks with multiple populations, each population is described by its own set of overlaps between connectivity sub-vectors (Fig. III.3a), and, importantly, by its own gain, which corresponds to the average slope $\phi'(x_i)$ on the input-output nonlinearity of neurons in the population. The effective couplings between inputs and internal variables are then given by a sum over populations of connectivity overlaps each weighted by the gain of the corresponding population (Eq. (II.29)). As an illustration, in the case of two populations, the effective coupling between the input and the internal variable becomes

$$\tilde{\sigma}_{nI} = \sigma^{(1)}_{nI}\langle\Phi'\rangle_1 + \sigma^{(2)}_{nI}\langle\Phi'\rangle_2 \tag{III.4}$$

where $\sigma^{(1)}_{nI}$ and $\sigma^{(2)}_{nI}$ are the overlaps for each population between the input vector $\boldsymbol{I}$ and the input-selection vector $\boldsymbol{n}$, while $\langle\Phi'\rangle_1$ and $\langle\Phi'\rangle_2$ are the gains of the two populations, that depend implicitly both on inputs and the values of internal variables. Crucially, additional inputs restricted to a given population can modulate its gain independently of other populations by shifting the position of neurons on the non-linear input-output function. Depending on the geometry between input vectors and input-selection vectors, different sets of inputs can play distinct roles of drivers and modulators (187), allowing the network to flexibly remodel the effective circuit formed by collective variables in different trials or epochs according to the demands of the task.

We applied this model-reduction analysis to the context-dependent decision-making task, for which the minimal trained networks were of unit rank and consisted of two subpopulations (Fig. III.3a). Analyzing the statistics of input and connectivity vectors for each population, we found that the input vectors $\boldsymbol{I}^A$ and $\boldsymbol{I}^B$ corresponding to the two stimulus features $u_A$ and $u_B$ had different overlaps with the input-selection vector $\boldsymbol{n}$ in the two populations (Fig. III.4b right) so that the two stimulus features $u_A$ and $u_B$ acted as drivers of latent dynamics. The contextual input vectors $\boldsymbol{I}^{ctxA}$ and $\boldsymbol{I}^{ctxB}$ in contrast had weak overlaps with the input-selection vector $\boldsymbol{n}$ (Sup Fig. A.7), but strongly different amplitudes on the two populations (Fig. III.4b left). They therefore modified the gains of the two populations in an opposite manner (Fig. III.4c bottom), and played the role of modulators that changed the form of the effective circuit describing latent dynamics in each context (Fig. III.4c top). More specifically, the latent dynamics of the internal variable $\kappa$ could be approximated by (Methods and Sup Fig. S4):

$$\tau\frac{d\kappa}{dt} = -\kappa + \tilde{\sigma}_{mn}\kappa + \sigma^{(1)}_{nI^A}\langle\Phi'\rangle_1 u_A(t) + \sigma^{(2)}_{nI^B}\langle\Phi'\rangle_2 u_B(t) \tag{III.5}$$

where $\langle\Phi'\rangle_1$ and $\langle\Phi'\rangle_2$ are the average gains of the two populations, $\sigma^{(1)}_{nI^A}$ the overlap for the first population between the input vector for stimulus feature $A$ and the input-selection vector $\boldsymbol{n}$, and $\sigma^{(2)}_{nI^B}$ the overlap for the second population between $\boldsymbol{n}$ and the input vector for stimulus feature $B$. By modulating the gains of the two populations in a differential manner between the two contexts (Fig. III.4c bottom), the contextual cues controlled the effective couplings between stimulus inputs and the internal variable $\kappa$, and determined which feature was integrated by the internal variable in each context (Fig. III.4d). This mechanism implemented an effective input gating, but only at the level of the latent dynamics of the internal variable $\kappa$ that integrated relevant evidence. Importantly, as observed in experimental data (118), on the level of the full network, the two stimulus features were

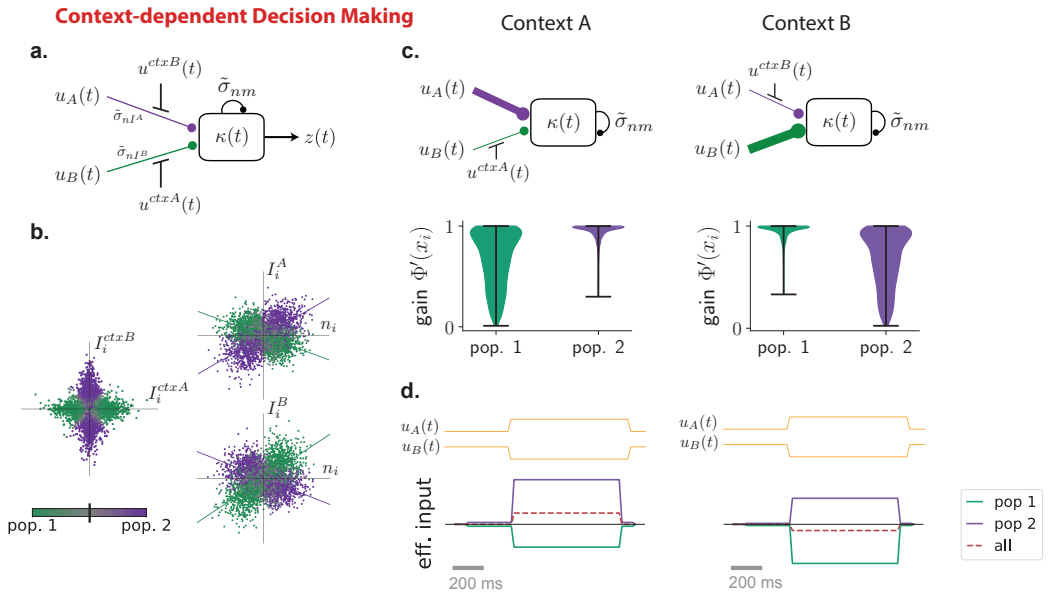


FIGURE III.4: Mechanisms of computations based on a multi-population connectivity, for the context-dependent decision-making task. (a) Circuit diagram representing latent dynamics in the reduced model of context-dependent decision-making task (Eq. III.5). The internal variable $\kappa$ is represented as a unit that integrates the two stimulus features $u_A$ and $u_B$ through effective couplings $\tilde{\sigma}_{nI^A}$ and $\tilde{\sigma}_{nI^B}$. Contextual inputs $u^{ctxA}$ and $u^{ctxB}$ modulate the gains of the two populations and therefore the effective couplings that govern which stimulus feature is integrated. Lines with round ends represent effective couplings, lines with straight ends represent gain modulation. (b) Projections of the six-dimensional connectivity space for a network trained on the task. Each point represents the parameters of one neuron, with the color shade indicating the probability that it belongs to each subpopulation, as found by the clustering procedure (Methods). For the remaining analysis, the two subpopulations are defined by a hard threshold at 0.5 on this probability. Left: plane defined by components of the contextual-cue vectors $\boldsymbol{I}^{ctxA}$ and $\boldsymbol{I}^{ctxB}$; right: two planes defined by components on the input-selection vector $\boldsymbol{n}$ and the two stimulus feature vectors $\boldsymbol{I}^A$ and $\boldsymbol{I}^B$ (lines show linear regressions for each population). (c) Effective circuits in each context (top) and corresponding gains of neurons in each population (bottom). For each neuron $i$, the gain is defined as the slope of $\phi(x_i)$ during stimulation period. Violin plots show the distribution of gains for all neurons in each population (pop. 1: $n = 2028$, pop. 2: $n = 2068$) in context A (left) and B (right). In context A, the average gain of neurons in population 1 (green) is lower than population 2 (purple), which decreases the effective connectivity between input feature B and the latent variable (top left circuit). The opposite happens in context B (top right circuit). (d) Effective inputs to the latent variable $\kappa$ in the two contexts (bottom) in response to the same stimulus input (top). Solid lines show inputs mediated by each population (Eq. (II.27)), the dashed line shows the total input, which changes signs between the two contexts, leading to opposite responses.

instead equally represented in both contexts, but along directions in state space orthogonal to the direction that encoded internal collective variable (Sup Fig. A.8).

For the delayed-match-to-sample task, we found that the multi-population structure also led to a modulation of latent dynamics, but across task epochs rather than across trials. Fig. III.5e-j describes an example minimal network implementing this task, where one of the stimuli played the role of a modulatory input, and transiently modified the latent dynamics when presented (Fig. III.5e,h,j). More specifically, the network was of rank two, so that the latent dynamics were described by effective interactions between two internal variables $\kappa_1$ and $\kappa_2$ (Fig. III.5e), and could be visualised in terms of a flow in a dynamical landscape in the $\kappa_1 - \kappa_2$ plane (Fig. III.5f). The minimal connectivity moreover consisted of two populations (Fig. III.5g). Stimulus A modulated the gain of the first population (Fig. III.5i), and therefore, when presented, modified the effective couplings in the latent dynamics and the dynamical landscape (Fig. III.5j and Sup Fig. A.9)). The main effect of the inputs was therefore to shape the trajectories of internal variables by modulating the dynamical landscape at different trial epochs (Fig. III.5j and Sup Fig. A.9). In particular, stimulus A strongly enhanced negative feedback (Fig. III.5h), which led to a limit-cycle in the dynamics that opened a fast transient channel that could flip neural activity in the $\kappa_1 - \kappa_2$ plane (27). The four trials in the task therefore corresponded to different sequences of dynamical landscapes (Fig. III.5j) leading to different neural trajectories and final states determining the correct behavioral outputs (Sup Fig. A.9).

## III.7 Predictions for neural selectivity and inactivations

Analyzing networks of minimal rank and minimal number of subpopulations allowed us to identify the mechanisms underlying computations based on a multi-population structure in connectivity. We next sought to generate predictions of the identified mechanisms that are experimentally testable without access to details of the connectivity. We then tested these predictions on networks with a higher number of subpopulations or higher rank, obtained by varying the constraints used during training. We focus here specifically on the context-dependent decision-making (CDM) task, and contrast it with the multi-sensory decision-making (MDM) task, for which networks received an identical input structure, but were required to produce an output independent of context.

For the CDM task, reducing the trained networks to effective circuits revealed that the key computations relied on a differential gain modulation of separate subpopulations by contextual inputs. For each neuron, contextual cues set its functioning point on its non-linearity, and thus the gain of its response to incoming stimuli. A direct implication is that neurons more strongly modulated by contextual cues change more strongly their gain across contexts, and thereby the amplitude of their responses to stimulus features (Fig. III.6a). An ensuing prediction at the level of selectivity of individual neurons is therefore that the pre-stimulus selectivity to context should be correlated with the change across contexts of regression coefficients to stimulus features (Fig. III.6b). Our analyses therefore predict a specific form of multiplicative interactions, or non-linear mixed selectivity to stimulus features and context cues (155), but also imply that the two subpopulations can be identified based on their selectivity to context (Fig. III.6b).

The multiplicative interaction between context and stimulus selectivity is a necessary, but not a sufficient condition for implementing context-dependent responding. A second, necessary component of the computational mechanism is that each subpopulation integrates dominantly one of the two features into the latent dynamics, as seen from the overlaps between the input vectors and the input-selection vectors (Fig. III.4b right). This leads to a specific prediction for inactivation experiments: inactivating separately subpopulations defined by their selectivity to context disrupts performance in one context, while leaving the other intact (Fig. III.6c-d). In contrast, inactivating a random subset of neurons leads only

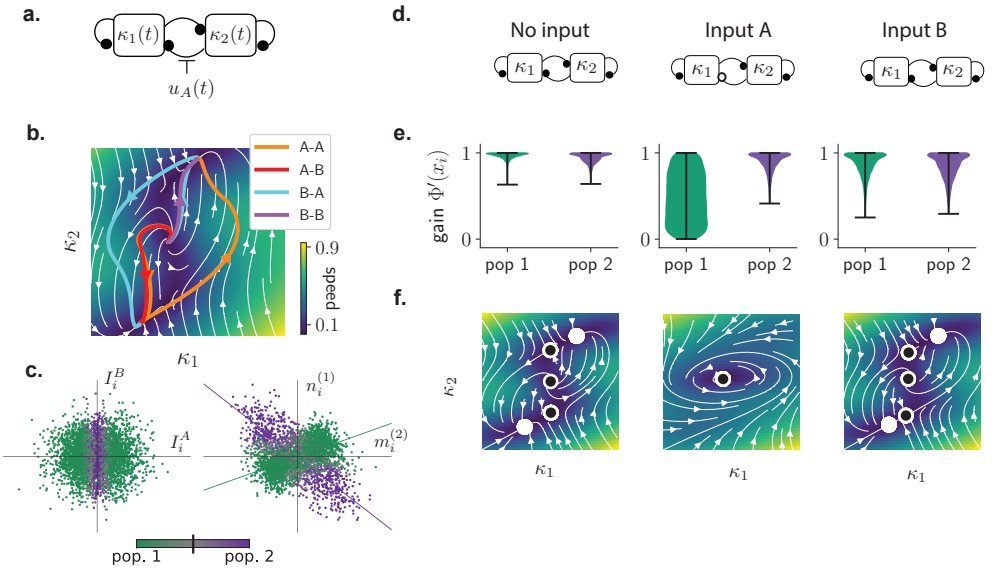


FIGURE III.5: Low-dimensional latent dynamics in networks performing the delayed match-to-sample (DMS) task. (a) Circuit diagram representing latent dynamics for a minimal network trained on the DMS task. The network was of rank two, so that the latent dynamics were described by two internal variables $\kappa_1$ and $\kappa_2$. Input A acts as a modulator on the recurrent interactions between the two internal variables. (b) Dynamical landscape for the autonomous latent dynamics in the $\kappa_1 - \kappa_2$ plane (*ie.* the $\boldsymbol{m}^{(1)}$-$\boldsymbol{m}^{(2)}$ plane). Colored lines depict trajectories corresponding to the 4 types of trials in the task (see Sup Fig. A.9 for details of trajectories). Background color and white lines encode the speed and direction of the dynamics in absence of inputs. (c) Two 2d projections of the seven-dimensional connectivity space, with colors indicating the two subpopulations and lines corresponding to linear regressions for each of them on the right panel. (d) Effective circuit diagrams in absence of inputs (left), and when input A (middle) or input B (right) are present. Filled circles denote positive coupling, open circles negative coupling. Input A in particular induces a negative feedback from $\kappa_2$ to $\kappa_1$. (e) Distribution of neural gains for each populations (pop. 1: $n = 3050$, pop. 2: $n = 1046$), in the three situations described above. The gain of population 1 (green) is specifically modulated by input A. (f) Dynamical landscapes in the 3 situations described above (see Methods). Filled and empty circles indicate respectively stable and unstable fixed points. The negative feedback induced by input A causes a limit cycle to appear in the latent dynamics.

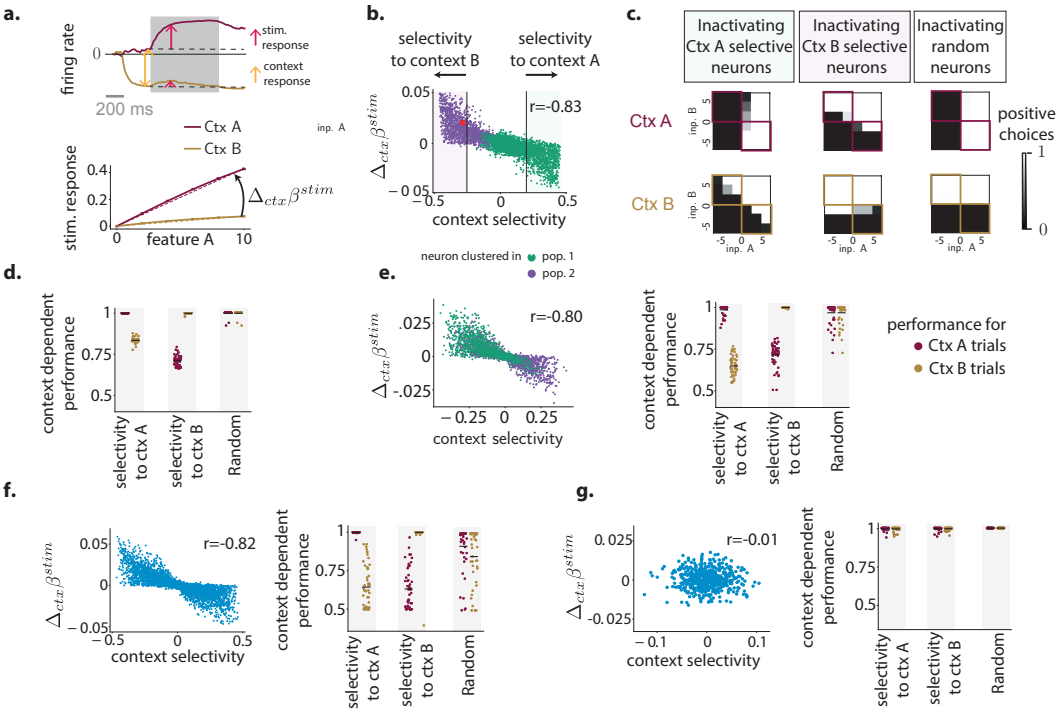


Figure III.6: **Figure 6. Predictions for neural selectivity and inactivations.** (a-d) Predictions for the context-dependent decision-making task based on the minimal unit-rank, two-populations network (Fig. III.4a). (a) Context-dependent stimulus response for an example neuron. Top: response to an identical stimulus in two contexts (gray box: stimulus-presentation period). The context response was defined as the change of pre-stimulus baseline across contexts (orange arrow). The stimulus response was defined in each context as the deviation from the pre-stimulus baseline (red arrows). Bottom: context-dependent responses of the same neuron to stimuli with increasing strength of feature $A$. In each context, we computed the regression coefficient with respect to feature strength (dashed lines), and the corresponding change in stimulus selectivity $\Delta_{ctx}\beta^{stim}$ (Methods Eq. (A.23)). (b) Interaction between context selectivity and the change in stimulus selectivity across neurons. Each point shows the change in stimulus selectivity versus selectivity to context for one neuron (see Methods Eq. (A.22)). Dot color corresponds to population determined from clustering procedure (Fig. III.4). Red dot: example neuron in (a). (c) Inactivations based on context selectivity lead to specific performance deficits. Psychometric response matrices when inactivating the 256 out of 1024 neurons with highest positive context selectivity (left), highest negative context selectivity (middle) or randomly chosen across the whole network (right). (d) Summary of the effects of inactivations: average performance over incongruent stimuli corresponding to colored squares of the psychometric matrix in (c). Each dot represents an inactivation of a random subset of 256 out of 1024 neurons. Inactivated neurons are chosen randomly among the neurons with either positive context selectivity (left column), negative context selectivity (middle column) or without constraint (right column). (e-g) Tests of the predictions for selectivity (left panels) and inactivations (right panels) on: (e) a unit-rank network consisting of three populations (A.6); (f) a network trained without a rank constraint; (g) a network trained on the multi-sensory decision-making (MDM) task.

to an overall decrease in performance independently of the context (Fig. III.6c-d).

We first tested the two predictions on networks constrained to be of minimal, unit rank, but in which clustering analyses in connectivity space revealed more than two subpopulations (Sup Fig. A.6), as in Yang et al. (227). The two predictions for selectivity and inactivations were therefore directly borne out for such networks (Fig. III.6e). We next turned to networks trained without rank constraint, and tested the two predictions without analyzing connectivity, as would be the case in experimental studies. The two predictions were again borne out (Fig. III.6f), confirming that key aspects of the computational mechanisms extend to networks in which the connectivity was of higher rank, and the dynamics higher dimensional.

Finally, we examined unit-rank networks trained on the MDM task. Such networks received an input structure identical to the CDM task, consisting of two stimulus features and two context cues. In contrast to the CDM task, the networks were trained to average the two stimulus features, and contextual cues were irrelevant, so that a fully random population structure was sufficient to perform the task (Fig. III.1l). We therefore expected that the two predictions made for the CDM task do not necessarily hold in this case. We indeed found that training networks on the MDM task led to weaker selectivity to context, and weaker correlation between context selectivity and the change in stimulus selectivity (Fig. III.6g). Specific neurons still exhibited selectivity to contextual cues, but inactivating them led to changes in performance similar to inactivating a random subset of neurons (Fig. III.6g). Importantly, this finding was unchanged when we matched the strength of context selectivity between MDM and CDM task by increasing the amplitude of contextual inputs (Sup Fig. A.10).

Altogether, identical context selectivity therefore led to opposite effects of inactivations across tasks, as predicted by our minimal-rank models.

## III.8 Implications for multi-tasking

A recent study reported that multiple populations emerge in networks trained simultaneously on multiple tasks, and can be repurposed across tasks (227). Our results more specifically suggest that a multi-population structure in connectivity is needed only when an identical stimulus requires different outputs depending on the context set by the performed task. While this is the case in many multi-tasking situations, concurrent tasks are alternatively often based on different sets of stimuli (34; 57; 48). Here we show that the reduced models developed by analyzing networks trained on individual tasks can be used to build networks that perform multiple tasks in parallel (Fig. III.7). More specifically, multiple tasks on an identical set of stimuli can be performed by combining and repurposing multiple subpopulations, while in contrast multiple tasks on separate sets of stimuli can be performed with a single population by relying on dynamics in orthogonal subspaces (45; 235). As a result, when identical stimuli are processed, some individual neurons exhibit task-specialisation, while for separate sets of stimuli all neurons are multi-taskers, and contribute to multiple tasks in parallel. These findings are in direct agreement with the activity of neurons in the prefrontal cortex during flexible categorisation, which show specialisation when identical stimuli are processed (160), and multi-tasking when separate stimuli sets are used (34).

To illustrate task-specialization, we first consider a network that receives stimuli composed of two sensory features, and depending on a rule cue performs one out of three different tasks on them : perceptual decision-making on the first stimulus feature, perceptual decision-making on the second stimulus feature, or integration of the two features as in the multi-sensory decision making task (Fig. III.7a). This multi-tasking setup is in fact a direct extension of context-dependent decision-making, and we implemented it using a simplified network based on the CDM task, consisting of unit-rank connectivity with three separate subpopulations (Sup Fig. A.6). In that network, each subpopulation has a well defined computational role. One of them plays the role of an evidence integrator, by endowing the

latent dynamics with a long timescale through strong positive feedback. That population is repurposed across all tasks (Fig. III.7c orange neuron), and inactivating it leads to performance degradation on all three tasks (Fig. III.7b). The other two populations relay separately the two sensory features into the latent dynamics, as in the CDM task (Fig. III.4b-d). Each of them participates in only two of the three tasks, as corroborated by changes in task performance after selective inactivations (Fig. III.7b). Neurons belonging to these two populations are therefore specialised for specific tasks, as seen in their task-specific responses to stimuli (Fig. III.7c green and purple neurons).

We next illustrate multi-tasking in a network that performs two tasks on distinct sets of stimuli, the perceptual decision-making (DM) and the parametric working-memory (WM) tasks (Fig. III.7d). Such a network can be obtained by directly superposing the connectivity matrices $\mathbf{J}_{DM}$ and $\mathbf{J}_{WM}$ of two minimal networks of rank-one and two that perform the individual tasks with random population structure (Fig. III.2). The resulting connectivity $\mathbf{J} = \mathbf{J}_{DM} + \mathbf{J}_{WM}$ is of rank three, and has a random population structure. The corresponding latent dynamics are based on a recurrent sub-space of dimension three, and the two tasks rely on two orthogonal subspaces with one dimension implementing the DM task, and the other two implementing the WM task (Fig. III.7e). Because of the random population structure, each neuron is a random combination of collective variables corresponding to different tasks, so that all neurons display multi-tasking activity (Fig. III.7f).

## III.9 Discussion

The goal of this study was to determine whether and when a non-random population structure is necessary for networks to perform a specific computation based on recurrent dynamics. To address this question, we first trained recurrent neural networks on a range of standard systems neuroscience tasks, and examined the emerging population structure in the selectivity and connectivity, and its relationship with the computations. We then identified underlying mechanisms by extracting the latent low-dimensional dynamics. Although a number of tasks could be implemented with random population structure in connectivity, we found that tasks based on flexible input-output mappings instead appeared to require an additional structure that could be accurately approximated in terms of a small number of subpopulations which played functionally distinct roles.

The starting motivation of this work was the apparent discrepancy between the experimental results of Ref. (152) and Ref. (81) (see also (82)). Analyzing neural activity in the rat posterior parietal cortex during a multi-sensory decision-making task, Ref. (152) found no evidence for non-random population structure in selectivity. Applying identical analyses to the prefrontal cortex, Ref. (81) instead identified population structure in activity during a more complex task that combined perceptual and value-guided decisions. Our results suggest that the difference between tasks provides a possible explanation for these diverging conclusions. Examining networks trained on an abstracted version of the multi-sensory integration task of Ref. (152), we found that a non-random population structure was not needed. Implementing a full version of the task used in Ref. (81) would have required reinforcement learning that falls beyond the scope of the supervised methods for training networks used here. The core component of that task was however a flexible weighing of two sensory features depending on the context set by reward history. That requirement of context-dependent weighing of input streams is in fact identical to the context-dependent decision-making task, in which all-or-none weights were assigned to the two stimulus features depending on the contextual cues. The gain-modulation mechanism underlying networks that performed the CDM task can more generally assign graded weights to each feature as required for the task of Ref. (81). This mechanism requires multiple populations, so that our analyses predict that a non-random population structure is needed for the task used in Ref. (81).

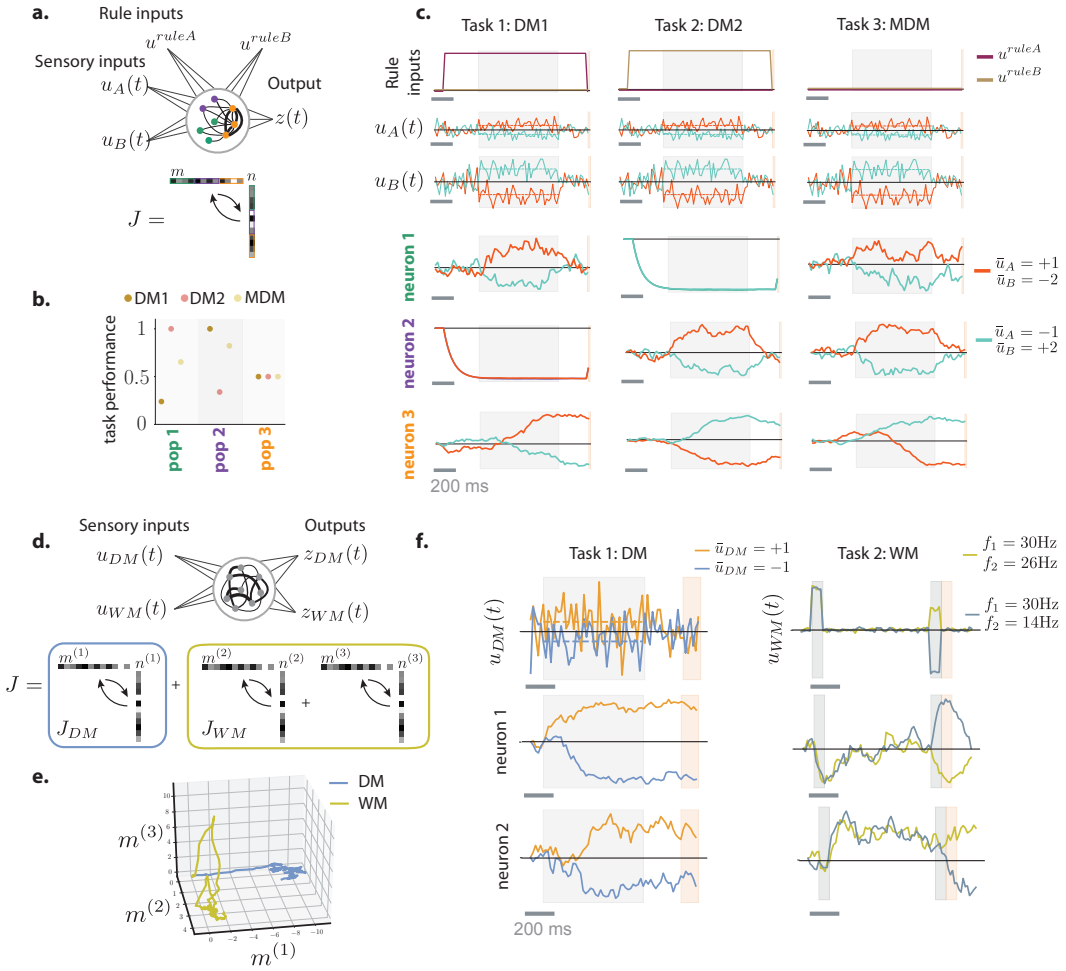


FIGURE III.7: **Figure 7. Implications of multi-population structure for multi-tasking.** (a) A network performing three different tasks on the same set of stimuli consisting of two features $u_A$ and $u_B$: decision-making based on $u_A$ (DM1), decision-making based on $u_B$ (DM2), decision-making based on integrating $u_A$ and $u_B$ (MDM). The model is obtained from the unit-rank network performing the CDM task based on three populations indicated in color. (b) Effects on the performance of individual tasks when specific populations are inactivated. In each case one third of the neurons in the network is inactivated, corresponding to one of the three populations. (c) Illustration of task specialization of different populations. The orange population plays the role of an integrator, and participates to all tasks. Green and purple populations respectively relay $u_A$ and $u_B$. Different columns correspond to different tasks. Top three rows display stimulus and rule inputs. Bottom three rows display single unit activities of three selected neurons (one in each population) in two trials of each task. (d) A network performing two different tasks on distinct sets of stimuli, the decision-making (DM) task on $u_{DM}$, and the working-memory task on $u_{WM}$. This network is obtained by superposing the low-rank recurrent connectivity matrices corresponding to the two tasks (illustrated at the bottom). (e) The two tasks rely on neural activity in orthogonal subspaces of the state space. Each subspace is determined by the input connectivity vectors of the corresponding task. (f) Illustration of multi-tasking of two example neurons.

We found that in trained networks relying on a non-random population structure, connectivity could be accurately described by a small number of subpopulations. Mechanistically, the role of such a subpopulation structure can be understood from two perspectives. From the neural state-space perspective, the collective dynamics explore a low-dimensional recurrent subspace, and the subpopulation structure shapes the non-linear dynamical landscape of the activity in that subspace (205). Specifically, different inputs differentially activate different subpopulations, and shift the recurrent subspace into different regions of the state-space with different non-linear dynamical landscapes. A complementary picture emerges from the perspective of the effective circuits which describe the low-dimensional latent dynamics in terms of interactions between collective variables through effective couplings (Fig. III.4). In that picture, the subpopulation structure allows inputs to control the effective couplings by modulating the average gain of different subpopulations. The computations then rely on two functionally distinct types of additive inputs: drivers that directly entrain the collective variables, and modulators that shape the gains of the different subpopulations, and thereby the interactions between collective variables. Interestingly, gain modulation has long been posited as a mechanism underlying selective attention (148), a type of processing closely related to flexible input-output tasks considered here. While patterns of gain modulation (167; 53; 201), and the distinction between drivers and modulators (187) are fundamentally physiological concepts, here we found that an analogous mechanism emerges in abstract trained networks at the collective level of latent dynamics.

To focus on the functional role of population structure, before training we initialized our networks with fully unstructured connectivity, and in particular did not include any explicit anatomical constraints such as Dale's law. Our analyses nevertheless show that the non-random population structure that emerges through training can be accurately described in terms of abstract sub-populations, defined as clusters in the connectivity space. What could be the physiological counter-parts of the different functional sub-populations that we identified? There are at least two distinct possibilities. In the network trained on the context-dependent decision-making task, we found that the two sub-populations differed only in the relationship of their connectivity with respect to feed-forward and contextual inputs. Such sub-populations therefore bear an analogy with input- and output-defined cortical populations such as for instance defined by inputs from the thalamus (77; 178) or outputs to the striatum (237). In the network trained on the delayed-match-to sample task, the two sub-populations instead differed at the level of recurrent connectivity: one population implemented positive, and the other negative feedback, the two being in general balanced (Sup Fig. A.7). This situation is reminiscent of excitatory and inhibitory sub-populations, which effectively implement positive and negative feedback in biological networks. Note that we do not mean to suggest that such population structure emerges biologically over the course of learning a task. Here we used artificial network training protocols to identify computational constraints, and we did not assume that they correspond to biological task-learning mediated by synaptic plasticity. The population wiring structure that emerges through network training could for instance be interpreted as the result of evolutionary selection leading to anatomic structure encoded at the genetic or developmental level (234).

Previous studies have reported that when training networks on a given task, some aspects of the solutions are invariant (117) while others depend on the details of the implementation (227; 45; 55). Our analyses confirmed these observations. Our main result for the computational requirement of non-random population structure in connectivity (Fig.III.1l) held independently of the details of the training, and in particular in absence of constraints on the rank of the network (Sup Fig. A.4). For tasks requiring a non-random population structure, the number of subpopulations needed to approximate connectivity however varied across networks (Sup Fig. A.6). For those tasks, our results show that a single global population is insufficient and that fundamental computational mechanisms are conserved across a range of different networks (Fig. III.6). Our analyses however do not predict the specific dimensionality or number of populations to be expected. More

systematic model selection could for instance be performed by further constraining recurrent neural networks based on recorded neural activity (150; 4).

The fact that neurons are selective to mixtures of task variables rather than individual features has emerged as one of the defining properties of representations in higher order areas of the mammalian cortex (60). Moving beyond a simple dichotomy between pure and mixed selectivity, recent studies argued that mixed selectivity does not necessarily preclude the presence of a population structure, and introduced the notion of non-random mixed selectivity (152; 81). Our results predict that the expected type of structure and mixed selectivity depends on the complexity of the performed task. In particular, for tasks requiring flexible input-output associations, we predict the presence of non-random population structure. The resulting non-random mixed-selectivity however becomes apparent only in response to specific combinations of variables, while selectivity to other variables can remain fully random (Fig. III.6). Ultimately, as the task complexity is increased, identifying the signatures of computational mechanisms in the neural activity requires a careful comparison with computational models on a task-by-task basis.

## Summary of Chapter 4

A large body of work has suggested that neural populations exhibit low-dimensional dynamics during behavior. However, there are a variety of different approaches for modeling low-dimensional neural population activity. One approach involves latent linear dynamical system (LDS) models, in which population activity is described by a projection of low-dimensional latent variables with linear dynamics. A second approach involves low-rank recurrent neural networks (RNNs), in which population activity arises directly from a low-dimensional projection of past activity. Although these two modeling approaches have strong similarities, they arise in different contexts and tend to have different domains of application. Here we examine the precise relationship between latent LDS models and linear low-rank RNNs. When can one model class be converted to the other, and vice versa? We show that latent LDS models can only be converted to RNNs in specific limit cases, due to the non-Markovian property of latent LDS models. Conversely, we show that linear RNNs can be mapped onto LDS models, with latent dimensionality at most twice the rank of the RNN. A surprising consequence of our results is that a partially-observed RNN is better represented by an LDS model than by an RNN consisting of only observed units.

This chapter is based on the manuscript *Probing the relationship between latent linear dynamical systems and low-rank recurrent neural network models*, Adrian Valente, Srdjan Ostojic, Jonathan W. Pillow *Neural Computation*, 34(9), p. 1871–1892 (2022).

# Relationship between linear low-rank networks and linear latent dynamical systems IV

## IV.1 Introduction

Recent work on large-scale neural population recordings has suggested that neural activity is often confined to a low-dimensional space, with fewer dimensions than the number of neurons in a population (31; 66; 64; 173; 87). To describe this activity, modellers have at their disposal a wide array of tools that give rise to different forms of low-dimensional activity (36). Two classes of modeling approaches that have generated a large following in the literature are: (i) descriptive statistical models; and (ii) mechanistic models. Broadly speaking, descriptive statistical models aim to identify a probability distribution that captures the statistical properties of an observed neural dataset, while remaining agnostic about the mechanisms that gave rise to it. Mechanistic models, by contrast, aim to reproduce certain characteristics of observed data using biologically-inspired mechanisms, but often with less attention to a full statistical description. Although these two classes of models often have similar mathematical underpinnings, there remain a variety of important gaps between them. Here we focus on reconciling the gaps between two simple but powerful models of low-dimensional neural activity: latent linear dynamical systems (LDS) and linear low-rank recurrent neural networks (RNNs).

The latent LDS model with Gaussian noise is a popular statistical model for low-dimensional neural activity in both systems neuroscience (191; 182) and brain-machine interface settings (95). This model has a long history in electrical engineering, where the problem of inferring latents from past observations has an analytical solution known as the Kalman filter (88). In neuroscience settings, this model has been used to describe high-dimensional neural population activity in terms of linear projections of low-dimensional latent variables. Although the basic form of the model includes only linear dynamics, recent extensions have produced state-of-the-art models for high-dimensional spike train data (231; 95; 144; 114; 135; 5; 44; 238; 69; 192).

Recurrent neural networks, by contrast, have emerged as a powerful framework for building mechanistic models of neural computations underlying cognitive tasks (203; 7; 118), and have more recently been used to reproduce recorded neural data (150; 33; 54; 143). While randomly-connected RNN models typically have high-dimensional activity (194; 102), recent work has shown that RNNs with low-rank connectivity provide a rich theoretical framework for modeling low-dimensional neural dynamics and the resulting computations (122; 103; 142; 180; 11; 43; 18; 104). In these low-rank RNNs, the structure of low-dimensional dynamics bears direct commonalities with latent LDS models, yet the precise relationship

between the two classes of models remains to be clarified. Understanding this relationship would open the door to applying to low-rank RNNs probabilistic inference techniques developed for LDS models, and conversely could provide mechanistic interpretations of latent LDS models fitted to data.

In this chapter, we examine the mathematical relationship between latent LDS and low-rank RNN models. We focus on linear RNNs, which are less expressive but simpler to analyze than their non-linear counterparts, while still leading to rich dynamics (79; 89; 18). We show that even if both LDS models and linear low-rank RNNs produce Gaussian distributed activity patterns with low-dimensional linear dynamics, the two model classes have different statistical structure and are therefore not in general equivalent. More specifically, in latent LDS models, the output sequence has non-Markovian statistics, meaning that the activity in a single timestep is not independent of its history given the activity on the previous timestep. This stands in contrast to linear RNNs, which are Markovian regardless of the rank of their connectivity. A linear low-rank RNN can nevertheless provide a first-order approximation to the distribution over neural activity generated by a latent LDS model, and we show that this approximation becomes exact in several cases of interest, and in particular in the limit where the number of neurons is large compared to the latent dimensionality. Conversely, we show that any linear low-rank RNN can be converted to a latent LDS, although the dimensionality of the latent space depends on the overlap between the subspaces spanned by left and right singular vectors of the RNN connectivity matrix, and may be as high as twice the rank of this matrix. The two model classes are thus closely related, with linear low-rank RNNs comprising a subset of the broader class of latent LDS models. An interesting implication of our analyses is that the activity of an RNN in which only a subset of neurons are observed is better fit by a latent LDS model than by an RNN consisting only of observed units.

## IV.2 Modeling frameworks

We start with a formal description of the two model classes in question, both of which describe the time-varying activity of a population of $n$ neurons.

### IV.2.1 Latent LDS model

The latent linear dynamical system (LDS) model, also known as a linear-Gaussian state-space model, describes neural population activity as a noisy linear projection of a low-dimensional latent variable governed by linear dynamics with Gaussian noise (88; 159) (See schematic, Fig. IV.1A). The model is characterized by the equations:

$$\boldsymbol{x}_{t+1} = \mathbf{A}\boldsymbol{x}_t + \boldsymbol{w}_t, \qquad \boldsymbol{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}) \tag{IV.1}$$

$$\boldsymbol{y}_t = \mathbf{C}\boldsymbol{x}_t + \boldsymbol{v}_t, \qquad \boldsymbol{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R}). \tag{IV.2}$$

Here, $\boldsymbol{x}_t$ is a $d$-dimensional latent (or "unobserved") vector that follows discrete-time linear dynamics specified by a $d \times d$ matrix $\mathbf{A}$, and is corrupted on each timestep by a zero-mean Gaussian noise vector $\boldsymbol{w}_t \in \mathbb{R}^d$ with covariance $\mathbf{Q}$. The vector of neural activity $\boldsymbol{y}_t$ arises from a linear transformation of $\boldsymbol{x}_t$ via the $n \times d$ observation (or "emissions") matrix $\mathbf{C}$, corrupted by zero-mean Gaussian noise vector $\boldsymbol{v}_t \in \mathbb{R}^n$ with covariance $\mathbf{R}$. Generally we assume $d < n$, so that the high-dimensional observations $\boldsymbol{y}_t$ are explained by the lower-dimensional dynamics of the latent vector $\boldsymbol{x}_t$. For clarity, in the main text we focus on LDS models without external inputs, and study their effect in the Appendix B.4.

The complete model also contains a specification of the distribution of the initial latent vector $\boldsymbol{x}_0$, which is commmonly assumed to have a zero-mean Gaussian distribution with covariance $\mathbf{\Sigma}_0$:

$$\boldsymbol{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_0). \tag{IV.3}$$

The complete parameters of the model are thus $\theta_{LDS} = \{\mathbf{A}, \mathbf{C}, \mathbf{Q}, \mathbf{R}, \mathbf{\Sigma}_0\}$. Note that this parametrization of an LDS is not unique: any invertible linear transformation of the latent space leads to an equivalent model if the appropriate transformations are applied to matrices $\mathbf{A}$, $\mathbf{C}$, $\mathbf{Q}$, and $\mathbf{\Sigma}_0$.

### IV.2.2 Low-Rank Linear RNN

A linear RNN, also known as an auto-regressive (AR) model, represents observed neural activity as a noisy linear projection of the activity at the previous timestep. We can write the model as (Fig. IV.1B):

$$\boldsymbol{y}_{t+1} = \mathbf{J}\boldsymbol{y}_t + \boldsymbol{\epsilon}_t, \qquad \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{P}), \tag{IV.4}$$

where $\mathbf{J}$ is an $n \times n$ recurrent weight matrix, and $\boldsymbol{\epsilon}_t \in \mathbb{R}^n$ is a Gaussian noise vector with mean zero and covariance $\mathbf{P}$. We moreover assume that the initial condition is drawn from a zero-mean distribution with covariance $\mathbf{V}_0^y$:

$$\boldsymbol{y}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{V}_0^y). \tag{IV.5}$$

A low-rank RNN model is obtained by constraining the rank of the recurrent weight matrix $\mathbf{J}$ to be $r \ll n$. In this case the recurrence matrix can be factorized as

$$\mathbf{J} = \mathbf{M}\mathbf{N}^\top, \tag{IV.6}$$

where $\mathbf{M}$ and $\mathbf{N}$ are both $n \times r$ matrices of rank $r$.

Note that this factorization is not unique, but a particular factorization can be obtained from a low-rank $\mathbf{J}$ matrix using the truncated singular value decomposition: $\mathbf{J} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$, where $\mathbf{U}$ and $\mathbf{V}$ are semi-orthogonal $n \times r$ matrices of left and right singular vectors, respectively, and $\mathbf{S}$ is an $r \times r$ diagonal matrix containing the largest singular values. We can then set $\mathbf{M} = \mathbf{U}$ and $\mathbf{N} = \mathbf{S}\mathbf{V}^\top$.

The model parameters of the low-rank linear RNN are therefore given by $\theta_{RNN} = \{\mathbf{M}, \mathbf{N}, \mathbf{P}, \mathbf{V}_0^y\}$.

### IV.2.3 Comparing the two models

Both models described above exhibit low-dimensional dynamics embedded in a high-dimensional observation space. In the following, we examine the probability distributions $P(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T)$ over time series $(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T)$ generated by the two models. We show that in general, the two models give rise to different distributions, such that the family of probability distributions generated by the LDS model cannot all be captured with low-rank linear RNNs. Specifically, RNN models are constrained to purely Markovian distributions, which is not the case for LDS models. However, the two model classes can be shown to be equivalent when the observations $\boldsymbol{y}_t$ contain exact information about the latent state $\boldsymbol{x}_t$, which is in particular the case if the observation noise is orthogonal to the latent subspace, or in the limit of a large number of neurons $n \gg d$. Conversely, a low-rank linear RNN can in general be mapped to a latent LDS with a dimensionality of the latent state at most twice the rank of the RNN.

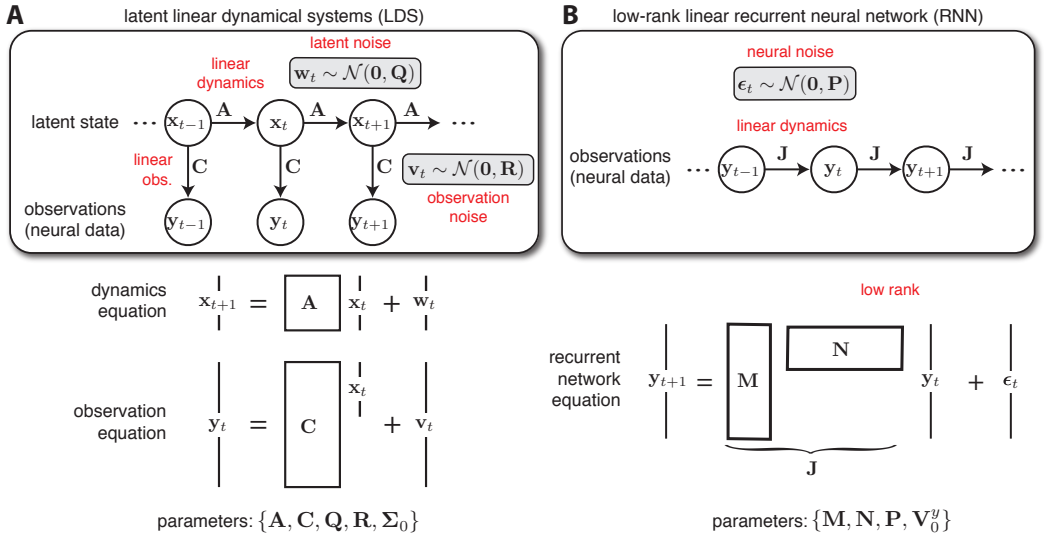


Figure IV.1: **A** Schematic representation of the latent linear dynamical system model, as defined by (eqs. IV.1-IV.3). **B** Schematic representation of the low-rank linear RNN, as defined by (eqs. IV.4-IV.5).

## IV.3 Mapping from LDS models to linear low-rank RNNs

### IV.3.1 Non-equivalence in the general case

Let us consider a latent LDS described by (eqs. IV.1-IV.3) and a low-rank linear RNN defined by (eqs. IV.4-IV.5). We start by comparing the properties of the joint distribution $P(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_T)$ for any value of $T$ for the two models. For both models, the joint distribution can be factored under the form:

$$P(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_T) = P(\boldsymbol{y}_0) \prod_{t=1}^{T} P(\boldsymbol{y}_t \mid \boldsymbol{y}_{t-1}, \ldots, \boldsymbol{y}_0), \tag{IV.7}$$

where each term in the product is the distribution of neural population activity at a single time point given all previous activity (see Appendix B.1 for details). More specifically, each of the conditional distributions in (IV.7) is Gaussian, and for the LDS we can parametrize these distributions as:

$$P(\boldsymbol{x}_t | \boldsymbol{y}_{t-1}, \ldots, \boldsymbol{y}_0) := \mathcal{N}(\hat{\boldsymbol{x}}_t, \mathbf{V}_t) \tag{IV.8}$$

$$P(\boldsymbol{y}_t | \boldsymbol{y}_{t-1}, \ldots, \boldsymbol{y}_0) = \mathcal{N}(\mathbf{C}\hat{\boldsymbol{x}}_t, \mathbf{C}\mathbf{V}_t\mathbf{C}^\top + \mathbf{R}), \tag{IV.9}$$

where $\hat{\boldsymbol{x}}_t$ is the mean of the conditional distribution over the latent at timestep $t$, given observations until timestep $t-1$. It obeys the recurrence equation:

$$\hat{\boldsymbol{x}}_{t+1} = \mathbf{A}(\hat{\boldsymbol{x}}_t + \mathbf{K}_t(\boldsymbol{y}_t - \mathbf{C}\hat{\boldsymbol{x}}_t)), \tag{IV.10}$$

where $\mathbf{K}_t$ is the Kalman gain given by

$$\mathbf{K}_t = \mathbf{V}_t\mathbf{C}^\top(\mathbf{C}\mathbf{V}_t\mathbf{C}^\top + \mathbf{R})^{-1}. \tag{IV.11}$$

and $\mathbf{V}_t$ represents a covariance matrix, which is independent of the observations and follows a recurrence equation detailed in Appendix B.1.

Iterating equation (Eq. IV.10) over multiple timesteps, one can see that $\hat{\boldsymbol{x}}_{t+1}$ depends not only on the last observation $\boldsymbol{y}_t$, but on the full history of observations $(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_t)$, which

therefore affects the distribution at any given timestep. The process $(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_t)$ generated by the LDS model is hence non-Markovian.

Conversely, for the linear RNN, the observations $(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_t)$ instead *do* form a Markov process, meaning that observations are conditionally independent of their history given the activity from the previous timestep:

$$P(\boldsymbol{y}_t \mid \boldsymbol{y}_{t-1}, \ldots, \boldsymbol{y}_0) = P(\boldsymbol{y}_t \mid \boldsymbol{y}_{t-1}). \tag{IV.12}$$

The fact that this property does not in general hold for the latent LDS shows that the two model classes are not equivalent. Due to this fundamental constraint, the RNN can only approximate the complex distribution (Eq. IV.7) parametrized by an LDS, as detailed in the following section and illustrated in figure IV.2.

### IV.3.2 Matching the first-order marginals of an LDS model

We can obtain a Markovian approximation of the LDS-generated sequence of observations $(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_t)$ by deriving the conditional distribution $P(\boldsymbol{y}_{t+1} \mid \boldsymbol{y}_t)$ under the LDS model, and matching it with a low-rank RNN (135). This type of first-order approximation will preserve exactly the one-timestep-difference marginal distributions $P(\boldsymbol{y}_{t+1}, \boldsymbol{y}_t)$ although structure across longer timescales might not be captured correctly.

First, let us note that we can express both $\boldsymbol{y}_t$ and $\boldsymbol{y}_{t+1}$ as noisy linear projections of $\boldsymbol{x}_t$:

$$\boldsymbol{y}_t = \mathbf{C}\boldsymbol{x}_t + \boldsymbol{v}_t, \tag{IV.13}$$

$$\boldsymbol{y}_{t+1} = \mathbf{C}(\mathbf{A}\boldsymbol{x}_t + \boldsymbol{w}_t) + \boldsymbol{v}_{t+1}, \tag{IV.14}$$

which follows from (Eq. IV.1).

Let $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_t)$ denote the Gaussian marginal distribution over the latent vector $\boldsymbol{x}_t$ at time $t$. Then we can use standard identities for linear transformations of Gaussian variables to derive the joint distribution over $\boldsymbol{y}_t$ and $\boldsymbol{y}_{t+1}$:

$$\begin{bmatrix} \boldsymbol{y}_t \\ \boldsymbol{y}_{t+1} \end{bmatrix} \sim \mathcal{N}\left( \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{C}\mathbf{\Sigma}_t\mathbf{C}^\top + \mathbf{R} & \mathbf{C}\mathbf{\Sigma}_t\mathbf{A}^\top\mathbf{C}^\top \\ \mathbf{C}\mathbf{A}\mathbf{\Sigma}_t\mathbf{C}^\top & \mathbf{C}(\mathbf{A}\mathbf{\Sigma}_t\mathbf{A}^\top + \mathbf{Q})\mathbf{C}^\top + \mathbf{R} \end{bmatrix} \right). \tag{IV.15}$$

We can then apply the formula for conditioning of multivariate Gaussians (see (17) equations (2.81) - (2.82)) to obtain:

$$\boldsymbol{y}_{t+1} \mid \boldsymbol{y}_t \sim \mathcal{N}\left(\mathbf{J}_t\boldsymbol{y}_t, \mathbf{P}_t\right) \tag{IV.16}$$

where

$$\mathbf{J}_t = \mathbf{C}\mathbf{A}\mathbf{\Sigma}_t\mathbf{C}^\top(\mathbf{C}\mathbf{\Sigma}_t\mathbf{C}^\top + \mathbf{R})^{-1} \tag{IV.17}$$

$$\mathbf{P}_t = \mathbf{C}(\mathbf{A}\mathbf{\Sigma}_t\mathbf{A}^\top + \mathbf{Q})\mathbf{C}^\top + \mathbf{R} - \mathbf{C}\mathbf{A}\mathbf{\Sigma}_t\mathbf{C}^\top(\mathbf{C}\mathbf{\Sigma}_t\mathbf{C}^\top + \mathbf{R})^{-1}\mathbf{C}\mathbf{\Sigma}_t\mathbf{A}^\top\mathbf{C}^\top. \tag{IV.18}$$

In contrast, from (Eq. IV.4), for a low-rank RNN the first-order marginal is given by

$$\boldsymbol{y}_{t+1} \mid \boldsymbol{y}_t \sim \mathcal{N}\left(\mathbf{J}\boldsymbol{y}_t, \mathbf{P}\right). \tag{IV.19}$$

Comparing equations (IV.16) and (IV.19), we see for the LDS model, the effective weights $\mathbf{J}_t$ and the covariance $\mathbf{P}_t$ depend on time through $\mathbf{\Sigma}_t$, the marginal covariance of the latent at time $t$, while for the RNN they do not. Note however that $\mathbf{\Sigma}_t$ follows the recurrence relation

$$\mathbf{\Sigma}_{t+1} = \mathbf{A}\mathbf{\Sigma}_t\mathbf{A}^\top + \mathbf{Q} \tag{IV.20}$$

which converges towards a fixed point $\mathbf{\Sigma}_\infty$ that obeys the discrete Lyapunov equation

$$\mathbf{\Sigma}_\infty = \mathbf{A}\mathbf{\Sigma}_\infty\mathbf{A}^\top + \mathbf{Q}, \tag{IV.21}$$

provided all eigenvalues of $\mathbf{A}$ have absolute value less than 1.

The LDS can therefore be approximated by an RNN with constant weights when the initial covariance $\mathbf{\Sigma}_0$ is equal to the asymptotic covariance $\mathbf{\Sigma}_\infty$, as noted previously (135). Note that even if this condition does not hold at time 0, $\mathbf{\Sigma}_\infty$ will in general be a good approximation of the latent covariance after an initial transient. In this case we obtain the fixed recurrence weights:

$$\mathbf{J} = \mathbf{C}\mathbf{A}\mathbf{\Sigma}_\infty\mathbf{C}^\top(\mathbf{C}\mathbf{\Sigma}_\infty\mathbf{C}^\top + \mathbf{R})^{-1} := \mathbf{M}\mathbf{N}^\top \tag{IV.22}$$

where we define $\mathbf{M} = \mathbf{C}$ which has shape $n \times d$ and $\mathbf{N}^\top = \mathbf{A}\mathbf{\Sigma}_\infty\mathbf{C}^\top(\mathbf{C}\mathbf{\Sigma}_\infty\mathbf{C}^\top + \mathbf{R})^{-1}$ which has shape $d \times n$, so that $\mathbf{J}$ is a rank $r$ matrix with $r = d$.

### IV.3.3 Cases of equivalence between LDS and RNN models

Although latent LDS and low-rank linear RNN models are not equivalent in general, we can show that the first-order Markovian approximation introduced above becomes exact in two limit cases of interest: (i) for observation noise orthogonal to the latent subspace, and (ii) in the limit $n \gg d$, with coefficients of the observation matrix generated randomly and independently.

Our key observation is that if $\mathbf{K}_t\mathbf{C} = \mathbf{I}$ in (IV.10) with $\mathbf{I}$ the identity matrix, we have $\hat{\boldsymbol{x}}_{t+1} = \mathbf{A}\mathbf{K}_t\boldsymbol{y}_t$, so that the dependence on the observations before timestep $t$ disappears, and the LDS therefore becomes Markovian. Interestingly, this condition $\mathbf{K}_t\mathbf{C} = \mathbf{I}$ also implies that the latent state can be inferred from the current observation $\boldsymbol{y}_t$ alone (see (B.7) in Appendix B.1) and that this inference is exact, since the variance of the distribution $P(\boldsymbol{x}_t|\boldsymbol{y}_t)$ is then equal to 0 as seen from (B.8). We next examine two cases where this condition is satisfied.

We first consider the situation where the observation noise vanishes, ie. $\mathbf{R} = \mathbf{0}$. Then, as shown in Appendix B.1, the Kalman gain is $\mathbf{K}_t = (\mathbf{C}^\top\mathbf{C})^{-1}\mathbf{C}^\top$, so that $\mathbf{K}_t\mathbf{C} = \mathbf{I}$. In that case, the approximation of the LDS by the RNN defined in the section 3.2 is exact, with equations (IV.17) and (IV.18) becoming:

$$\mathbf{J} = \mathbf{C}\mathbf{A}(\mathbf{C}^\top\mathbf{C})^{-1}\mathbf{C}^\top \tag{IV.23}$$

$$\mathbf{P} = \mathbf{C}\mathbf{Q}\mathbf{C}^\top. \tag{IV.24}$$

More generally, this result remains valid when the observation noise is orthogonal to the latent subspace spanned by the columns of the observation matrix $\mathbf{C}$ (in which case the recurrence noise given by (IV.24) becomes $\mathbf{P} = \mathbf{C}\mathbf{Q}\mathbf{C}^\top + \mathbf{R}$).

A second case in which we can obtain $\mathbf{K}_t\mathbf{C} \approx \mathbf{I}$ is in the limit of many neurons, $n \gg d$, assuming that coefficients of the observation matrix are generated randomly and independently. Indeed, under these hypotheses the Kalman gain given by (IV.11) is dominated by the term $\mathbf{C}\mathbf{V}_t\mathbf{C}^\top$, so that the observation covariance $\mathbf{R}$ becomes negligible, as shown formally in Appendix B.2. Intuitively this means that the information about the latent state $\hat{\boldsymbol{x}}_t$ is distributed over a large enough population of neurons for the Kalman filter to average out the observation noise and estimate it optimally without making use of previous observations. Ultimately, this makes the LDS asymptotically Markovian in the case where we have an arbitrarily large neural population relative to the number of latent dimensions.

To illustrate the convergence of the low-rank RNN approximation to the target latent LDS in the large $n$ limit, in figure IV.2 we consider a simple example with a one-dimensional latent space and observation spaces of increasing dimensionality. To visualize the difference between the LDS and its low-rank RNN approximation, we plot the trace of the autocorrelation matrix of observations $\boldsymbol{y}_t$ in the stationary regime, $\rho(\delta) = \mathrm{Tr}(\mathbb{E}[\boldsymbol{y}_t\boldsymbol{y}_{t+\delta}^T])$. Since the RNNs are constructed to capture the marginal distributions of observations separated by at most one timestep, the two curves match exactly for a lag $\delta \in \{-1, 0, 1\}$, but dependencies at

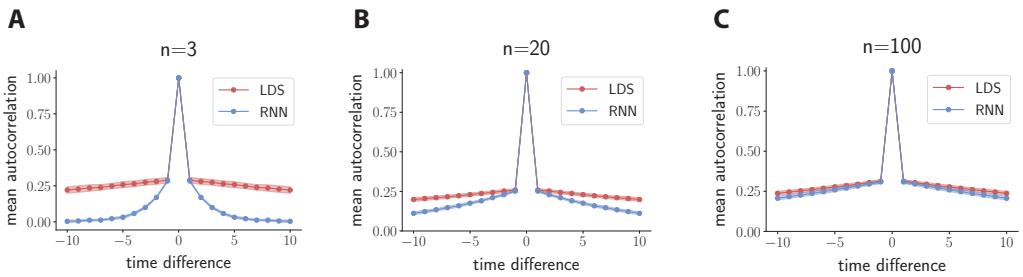


FIGURE IV.2: Mean autocorrelation of observations $\mathbf{y}_t$ from latent LDS processes compared with their first-order RNN approximations. The latent space is one-dimensional ($d = 1$), and the dimension $n$ of the observation space is increased from left to right: a. $n = 3$, b. $n = 20$, c. $n = 100$. The parameters of the latent state processes are fixed scalars ($\mathbf{A} = (0.97)$, $\mathbf{Q} = (0.1)$), while the elements of the observation matrices $\mathbf{C}$ are drawn randomly and independently from a centered Gaussian distribution of variance 1. The observation noise has covariance $\mathbf{R} = \sigma_v^2 \mathbf{I}_n$ with $\sigma_v^2 = 2$. Note that we have chosen observation noise to largely dominate over latent state noise in order to obtain a large difference between models at low $n$. Dots and shaded areas indicate respectively mean and standard deviation of different estimations of the mean autocorrelation done on 10 independent folds of 100 trials each (where $\mathbf{C}$ was identical across trials).

longer timescales cannot be accurately captured by an RNN due to its Markov property (Fig. IV.2a). However, these differences vanish as the dimensionality of the observation space becomes much larger than that of the latent space (Fig. IV.2b-c), which illustrates that the latent LDS converges to a process equivalent to a low-rank RNN.

## IV.4 Mapping low-rank linear RNNs onto latent LDS models

We now turn to the reverse question: under what conditions can a low-rank linear RNN be expressed as a latent LDS model? We start with an intuitive mapping for the deterministic case (*i.e.*, when noise covariance $\mathbf{P} = \mathbf{0}$), and then extend it to a more general mapping valid in the presence of noise.

We first consider a deterministic linear low-rank RNN obeying:

$$\boldsymbol{y}_{t+1} = \mathbf{M}\mathbf{N}^\top \boldsymbol{y}_t. \tag{IV.25}$$

Since $\mathbf{M}$ is an $n \times r$ matrix, it is immediately apparent that for all $t$, $\boldsymbol{y}_t$ is confined to a linear subspace of dimension $r$, spanned by the columns of $\mathbf{M}$. Hence, we can define the $r$-dimensional latent state as

$$\boldsymbol{x}_t = \mathbf{M}^+ \boldsymbol{y}_t \tag{IV.26}$$

where $\mathbf{M}^+$ is the pseudoinverse of $\mathbf{M}$ defined as $\mathbf{M}^+ = (\mathbf{M}^\top \mathbf{M})^{-1} \mathbf{M}^\top$ (well defined since $\mathbf{M}$ is of rank $r$), so that we retrieve $\boldsymbol{y}_t$ as:

$$\boldsymbol{y}_t = \mathbf{M}\boldsymbol{x}_t. \tag{IV.27}$$

We then obtain a recurrence equation for the latent state :

$$\begin{aligned} \boldsymbol{x}_{t+1} &= \mathbf{M}^+ \boldsymbol{y}_{t+1} \\ &= \mathbf{M}^+ \mathbf{M}\mathbf{N}^\top \boldsymbol{y}_t \\ &= \mathbf{N}^\top \mathbf{M} \boldsymbol{x}_t \\ &:= \mathbf{A}\boldsymbol{x}_t \end{aligned} \tag{IV.28}$$

which with $\mathbf{A} = \mathbf{N}^\top \mathbf{M}$ describes the dynamics of a latent LDS with $d = r$. A key insight from (IV.28) is that the overlap between the columns of $\mathbf{N}$ and $\mathbf{M}$ determine the part of the activity that is integrated by the recurrent dynamics (122; 180; 11; 43).

In presence of noise $\boldsymbol{\epsilon}_t$, $\boldsymbol{y}_t$ is no longer confined to the column space of $\mathbf{M}$. Part of this noise is integrated into the recurrent dynamics and can contribute to the activity accross many time steps. This integration of noise can occur in an LDS at the level of latent dynamics through $\boldsymbol{w}_t$, but not at the level of observation noise $\boldsymbol{v}_t$, which is independent accross timesteps. As noted above, recurrent dynamics only integrate the activity present in the column space of $\mathbf{N}$. In the presence of noise, this part of state space therefore needs to be included into the latent variables. More importantly, a similar observation can be made about external inputs when they are added to the RNN dynamics (see Appendix B.4).

A full mapping from a noisy low-rank RNN to an LDS model can therefore be built by extending the latent space to the linear subspace $\mathcal{F}$ of $\mathbb{R}^n$ spanned by the columns of $\mathbf{M}$ and $\mathbf{N}$ (see Appendix B.3), which has dimension $d$ with $r \leq d \leq 2r$. Let $\mathbf{C}$ be a matrix whose columns form an orthogonal basis for this subspace (which can be obtained via the Gram-Schmidt algorithm). In that case we can define the latent vector as :

$$\boldsymbol{x}_t = \mathbf{C}^\top \boldsymbol{y}_t, \tag{IV.29}$$

and the latent dynamics are given by

$$\boldsymbol{x}_{t+1} = \mathbf{A}\boldsymbol{x}_t + \mathbf{w}_t, \tag{IV.30}$$

where the recurrence matrix is $\mathbf{A} = \mathbf{C}^\top \mathbf{J} \mathbf{C}$, and the latent dynamics noise is $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ with $\mathbf{Q} = \mathbf{C}^\top \mathbf{P} \mathbf{C}$. Introducing $\boldsymbol{v}_t = \boldsymbol{y}_t - \mathbf{C}\boldsymbol{x}_t$, under a specific condition on the noise covariance $\mathbf{P}$ we obtain a normal random variable independent of the other sources of noise in the process (Appendix B.3), so that $\boldsymbol{y}_t$ can be described as a noisy observation of the latent state $\boldsymbol{x}_t$ as in the LDS model:

$$\boldsymbol{y}_t = \mathbf{C}\boldsymbol{x}_t \ + \ \boldsymbol{v}_t. \tag{IV.31}$$

### IV.4.1 Subsampled RNNs

Experimental recordings typically access only the activity of a small fraction of neurons in the local network. An important question for interpreting neural data concerns the statistics of activity when only a random subset of $k$ neurons in an RNN are observed. This situation can be formalized by introducing the set of observed activities $\mathbf{o}_t$:

$$\begin{aligned} \boldsymbol{y}_{t+1} &= \mathbf{J}\boldsymbol{y}_t + \boldsymbol{\epsilon}_t \\ \mathbf{o}_t &= \boldsymbol{y}_t[:k] = \mathbf{D}\boldsymbol{y}_t. \end{aligned} \tag{IV.32}$$

Here $[:k]$ symbolizes the selection of the first $k$ values of a vector and $\mathbf{D}$ is the corresponding projection matrix on the subspace spanned by the first $k$ neurons. The system described by (IV.32) is exactly an LDS, but with latent state $\boldsymbol{y}_t$ and observations $\mathbf{o}_t$. In contrast to the regime considered in the previous sections, the latents have a higher dimensionality than observations. However, assuming as before that $\mathbf{J}$ is low-rank, this model can be mapped onto an equivalent LDS following the steps in appendix B.3:

$$\begin{aligned} \boldsymbol{x}_{t+1} &= \mathbf{A}\boldsymbol{x}_t + \boldsymbol{w}_t \\ \mathbf{o}_t &= \mathbf{D}\mathbf{C}\boldsymbol{x}_t + \mathbf{D}\boldsymbol{v}_t. \end{aligned} \tag{IV.33}$$

This LDS is equivalent to (IV.32), but with latent dynamics $\boldsymbol{x}_t$ of dimension $r < d < 2r$ where $r$ is the rank of $\mathbf{J}$. The dynamics of the latent state $\boldsymbol{x}_t$ are identical to those of the fully-observed low-rank RNN (IV.30), but the observations are generated from a subsampled observation matrix $\mathbf{DC}$.

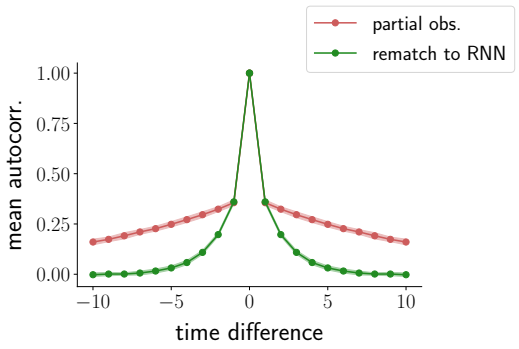


FIGURE IV.3: Mean autocorrelation of $k$ neurons subsampled from an $n$-dimesional rank-one RNN, compared with a $k$-dimensional RNN built to match the first-order marginals of partial observations. Formally, we first built an LDS equivalent to the partial observations as in (IV.33), and then the corresponding RNN as in section IV.3.2. The rank-one RNN contains $n = 20$ neurons, of which only $k = 3$ are observed. The mismatch occurs because the long-term correlations present in the partial observations are caused by the larger size of the original RNN with 20 neurons, and can not be reproduced by an RNN with only 3 neurons.

Interestingly, this mapping highlights the fact that the activity statistics of the $k$ subsampled neurons are in general not Markovian, in contrast to the full activity $\boldsymbol{y}_t$ of the $n$ neurons in the underlying RNN. In particular, for that reason the statistics of $\mathbf{o}_t$ cannot be exactly reproduced by a smaller RNN consisting of $k$ units (Fig. 3). Remarkably, when considering the subsampled activity of an RNN, a latent LDS is therefore a more accurate model than a smaller RNN containing only the observed units.

## IV.5 Discussion

In this chapter we have examined the relationship between two simple yet powerful classes of models of low-dimensional activity: latent linear dynamical systems (LDS) and low-rank linear recurrent neural networks (RNN). We have focused on these tractable linear models with additive Gaussian noise to highlight their mathematical similarities and differences. Although both models induce a jointly Gaussian distribution over neural population activity, generic latent LDS models can exhibit long-range, non-Markovian temporal dependencies that cannot be captured by low-rank linear RNNs, which describe neural population activity with a first-order Markov process. Conversely, we showed that generic low-rank linear RNNs can be captured by an equivalent latent LDS model. However, we have shown that the two classes of models are effectively equivalent in limit cases of practical interest for neuroscience, in particular when the number of sampled neurons is much higher than the latent dimensionality.

Although these two model classes can generate similar sets of neural trajectories, different approaches are typically used for fitting them to neural data: parameters of LDS models are in general inferred by variants of the expectation-maximization algorithm (231; 135; 132; 47), which include the Kalman smoothing equations (159), while RNNs are often fitted with variants of linear regression (150; 49; 147; 18) or backpropagation-through-time (43). The relationship uncovered here therefore opens the door to comparing different fitting approaches more directly, and in particular to developing probabilistic methods for inferring RNN parameters from data.

We have considered here only linear RNN and latent LDS models. Non-linear low-rank RNNs without noise can be directly reduced to non-linear latent dynamics with linear observations following the same mapping as in Section 4 (122; 180; 11; 43), and

therefore define a natural class of non-linear LDS models. A variety of other non-linear generalizations of LDS models have been considered in the literature. One line of work has examined linear latent dynamics with a non-linear observation model (231) or non-linear latent dynamics (231; 47; 44; 136; 95). Another line of work has focused on switching LDS models (110; 69) for which the system undergoes different linear dynamics depending on a hidden discrete state, thus combining elements of latent LDS and hidden Markov models. Both non-linear low-rank RNNs and switching LDS models are universal approximators of low-dimensional dynamical systems (59; 30; 11). Relating switching LDS models to local linear approximations of non-linear low-rank RNNs (11; 43) is therefore an interesting avenue for future investigations.

## Summary of Chapter 5

An influential framework within systems neuroscience posits that neural computations can be understood in terms of low-dimensional dynamics in recurrent circuits. A number of methods have thus been developed to extract latent dynamical systems from neural recordings, but inferring models that are both predictive and interpretable remains a difficult challenge. Here we propose a new method called Low-rank Inference from Neural Trajectories (LINT), based on a class of low-rank recurrent neural networks (lrRNNs) for which a link between connectivity and dynamics has been previously demonstrated. By fitting such networks to trajectories of neural activity, LINT yields a mechanistic model of latent dynamics, as well as a set of axes for dimensionality reduction and verifiable predictions for inactivations of specific populations of neurons. Here, we first demonstrate the consistency of our method and apply it to two use cases: (i) we reverse-engineer "black-box" vanilla RNNs trained to perform cognitive tasks, and (ii) we infer latent dynamics and neural contributions from electrophysiological recordings of nonhuman primates performing a similar task.

This chapter is based on the manuscript *Extracting computational mechanisms from neural data using low-rank RNNs*, Adrian Valente, Jonathan W. Pillow, Srdjan Ostojic, submitted at NeurIPS 2022.

# Extracting computational mechanisms from neural data using low-rank RNNs V

## V.1 Introduction

As large-scale neural recordings in behaving animals are becoming commonplace, a pressing question is how computational principles can be extracted from the electrical activity of thousands of cells. An influential framework posits that neural computations rely on latent low-dimensional dynamics (203; 217) distributed across populations of neurons (233; 173). In line with this proposal, a number of data-analysis methods have been developed to infer latent dynamics from neural recordings (231; 114; 144; 135; 47; 132; 110; 136; 44; 69; 63; 168; 169; 177; 105) (reviewed in (46)). While these statistical approaches often provide compelling descriptions of the recorded data, they generally lack a mechanistic interpretation that would, for instance, allow them to make predictions for responses to novel interventions on the underlying neural circuits. How to identify mechanistic, predictive models from neural data currently remains an important challenge.

Recurrent neural networks (RNNs) have emerged as key models for studying neural computations in this framework (229). Indeed, RNNs can be trained to solve numerous cognitive tasks, exhibiting surprisingly similar dynamical mechanisms as those observed in neural recordings (118; 206; 27; 219; 153), and have also successfully been trained to reproduce the activity of neurons recorded *in vivo* (150; 33; 54; 143). While potentially predictive, the obtained networks are however typically challenging to understand mechanistically (205; 117; 116; 115; 227; 211), and ongoing research directions aim at reducing them to simplified, interpretable models, for example through the use of linearized dynamical systems (47; 69; 192) or through "network distillation" methods (175; 105).

Here, we exploit a particular class of interpretable RNNs, namely *low-rank RNNs* (lrRNNs) (122; 11; 43; 180; 181; 212), to develop a new method that extracts mechanistic and predictive low-dimensional models from observed neural activity, which can be applied to both artificial and biological neural networks. Our method, Low-rank Inference from Neural Trajectories (LINT) infers a minimal rank lrRNN from a dataset and exploits the theory of low-rank networks to relate the obtained connectivity with low-dimensional dynamics and computations. It produces three outputs: a set of axes for dimensionality reduction of the trajectories, an effective connectivity that implements a latent dynamical system, and predictions for interventions on specific subsets of recorded neurons (Fig. V.1a).

Our contributions can be summarized in three steps: first, we verify the consistency of LINT by applying it to data simulated from lrRNNs, and show that it recovers the effective part of the connectivity that reproduces the dynamics and computations. Second, we apply

LINT to data generated from full-rank RNNs trained on cognitive tasks, and demonstrate that the resulting lrRNNs can capture their dynamics and offer mechanistic insights into how they function. Notably, we identify a novel population-based mechanism enabling context-dependent switching in a vanilla RNN, and verify it by targeted inactivation experiments. Finally, we apply LINT to electrophysiological recordings in nonhuman primates performing a context-dependent decision making task (118); this reveals that a rank-one network can capture most aspects of the dynamics present in the data, and that computations in this neural circuit appear to be supported by a small proportion of all recorded neurons.

## V.2 Approach

Here we describe our method for extracting an interpretable low-dimensional projection and dynamical model from neural trajectories by fitting a low-rank recurrent network to data.

**Low-rank RNNs (lrRNNs)**. We start from rate-based recurrent neural networks of $N$ units, each characterized by an activation variable $x_i$ which follows the dynamics:

$$\tau \frac{dx_i}{dt} = -x_i + \sum_{j=1}^{N} J_{ij}\phi\left(x_j\right) + \sum_{s=1}^{N_{in}} I_i^{(s)} u_s(t) + \eta_i(t). \tag{V.1}$$

Here $\mathbf{J}$ represents the network connectivity matrix and $\phi$ is a nonlinear transfer function defining the neural firing rate $\phi(x_i)$, which we take here to be tanh. Each network also receives $N_{in}$ input signals $u_s(t)$ via a set of weights $\boldsymbol{I}^{(s)}$ that we refer to as *input vectors*.

Low-rank RNNs represent a subclass of models in which the connectivity matrix is constrained to be of finite rank $R \ll N$ (122; 11; 43). In this case, $\mathbf{J}$ can be written as a sum of outer products of *connectivity vectors* $\boldsymbol{n}^{(r)}$ and $\boldsymbol{m}^{(r)}$:

$$J_{ij} = \frac{1}{N} \sum_{r=1}^{R} m_i^{(r)} n_j^{(r)}. \tag{V.2}$$

To define a unique representation, we take as $\boldsymbol{m}^{(r)}$ the left singular vectors of the connectivity matrix, and as $\boldsymbol{n}^{(r)}$ the right singular vectors multiplied by the corresponding singular value.

**Network inference.** We infer lrRNNs from data by training the connectivity parameters to reproduce recorded neural trajectories (either single-trial or condition-averaged trajectories). Specifically, we use back-propagation on connectivity vector parameters $n_i^{(r)}$ and $m_i^{(r)}$ and input parameters $I_i^{(s)}$ to minimize the squared difference between target and reproduced trajectories:

$$\mathcal{L} = \sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{t=1}^{T} (\phi(x_i^{(c)}(t)) - \phi(\tilde{x}_i^{(c)}(t)))^2 \tag{V.3}$$

where $\tilde{x}_{i,t,c}$ represents the target trajectory in condition $c$, for neuron $i$ and timestep $t$ and $x_{i,t,c}$ the corresponding trajectory produced by the model.

**Dimensionality reduction**. One property of lrRNNs is that they constrain the activity vector $\boldsymbol{x}(t)$ to evolve in a low-dimensional subspace spanned by the $R$ connectivity vectors $\boldsymbol{m}^{(r)}$ and the $N_{in}$ input vectors $\boldsymbol{I}^{(s)}$ (11). The trajectories therefore explore at most $R + N_{in}$ dimensions and can therefore be parametrized as:

$$\boldsymbol{x}(t) = \sum_{r=1}^{R} \kappa_r(t) \boldsymbol{m}^{(r)} + \sum_{s=1}^{N_{in}} v_s(t) \boldsymbol{I}^{(s)}, \tag{V.4}$$

where $v_s(t)$ are the low-pass filtered input signals $u_s(t)$ and $\kappa_r$ are a set of latent variables generated by recurrent activity. Thus, lrRNNs provide by design a reduction of $N$-dimensional

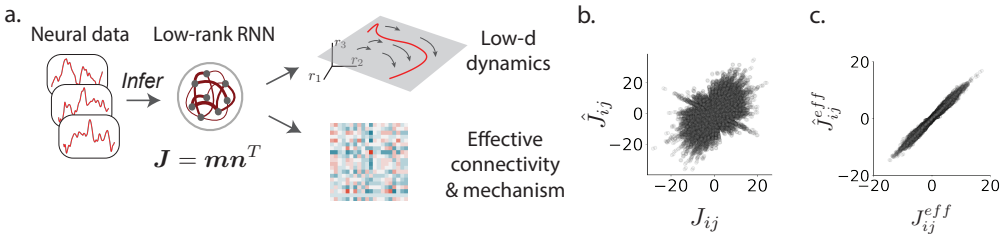


FIGURE V.1: a. LINT pipeline: Low-rank RNNs are fitted to either simulated or recorded neural trajectories. The obtained lrRNNs provide interpretable low-dimensional dynamics as well as a computational mechanism based on an effective connectivity structure. b-c. LINT tested on trajectories simulated from a rank-one lrRNN trained to perform the CDM task (see Table 1 for results across tasks). b. For each pair of neurons, relationship between the original ($J_{ij}$) and inferred ($\hat{J}_{ij}$) synaptic connectivity ($512^2$ pairs, $r = 0.57$). c. For the same networks, relationship between the original effective connectivity ($J^{eff}_{ij}$, see text) and the inferred one ($\hat{J}^{eff}_{ij}$, $r = 0.99$)

neural activity to at most $R + N_{in}$ dimensions that can be directly interpreted in terms of components on a *recurrent subspace* spanned by the connectivity vectors $\boldsymbol{m}^{(r)}$ and on an *input subspace* spanned by inputs vectors $\boldsymbol{I}^{(s)}$ (219).

**Latent dynamics**. In lrRNNs, the latent variables $\kappa_r(t)$ form a non-linear low-dimensional dynamical system, described by:

$$\frac{d}{dt}\boldsymbol{\kappa}(t) = F(\boldsymbol{\kappa}(t), \boldsymbol{u}(t)) \tag{V.5}$$

where $\boldsymbol{\kappa}(t) = \{\kappa_r(t)\}_{r=1..R}$ is an $R$-dimensional vector representing the activity on the recurrent subspace, $\boldsymbol{u}(t)$ represents the input signals, and $F$ is a non-linear function that can be directly determined from network connectivity parameters (11; 43). Moreover, it can be shown that rank-$R$ networks are universal approximators of $R$-dimensional dynamical systems (11), and thus that the function $F$ can approximate any non-linear mapping in $R$ dimensions.

**Task-optimized RNNs**. We applied our method first to trajectories produced by task-optimized RNNs, which produce an output signal $z(t)$ from the recurrent dynamics (V.1) via a linear readout:

$$z(t) = \sum_{i=1}^{N} w_i \phi(x_i(t)) \tag{V.6}$$

In task-optimized networks, the parameters are trained with backpropagation through time to minimize the squared error between the output $z(t)$ and a target $z^*(t)$. In this work, we consider four cognitive tasks: Decision Making (DM), Working Memory (WM), Context-Dependent Decision Making (CDM, Fig. V.2a) and Delayed Match-to-Sample (DMS) (see Appendix C.1 for task definitions). We then generate neural trajectories from both low-rank (in section 3.1) or full-rank (see section 3.2) task-optimized RNNs. All networks are implemented in pytorch (140), and training details can be found in Appendix C.2.

## V.3 Validation with synthetic data and effective aspects of connectivity

We first validated the capacity of the LINT method to recover low-rank connectivity features and reproduce neural trajectories on data simulated from lrRNNs trained on cognitive

Table V.1: Synthetic data validation results for LINT
(CC: connectivity correlation, ECC: effective connectivity correlation)

| Task | Rank | Trajectory $R^2$ | CC | ECC |
|---|---|---|---|---|
| Decision Making (DM) | 1 | 0.97 | 0.50 | 0.99 |
| Working Memory | 2 | 0.96 | 0.43 | 0.93 |
| Context-dependent DM | 1 | 0.91 | 0.57 | 0.99 |
| Delayed Match-to-Sample | 2 | 0.98 | 0.39 | 0.63 |

tasks. For this, we first train lrRNNs with 512 neurons to produce a correct behavioral output on four systems neuroscience tasks, each time retaining the minimal rank solution (43) (see Table 1). For each task-optimized lrRNN, we then generate simulated trajectories corresponding to trials in the trained task, and apply our method to fit these trajectories using equivalent models with the same rank and number of neurons. We found that the inferred networks were able to reliably reproduce the original trajectories when fed with the same inputs, as quantified by the $R^2$ scores between original and fitted trajectories. Although the inferred networks were not explicitly trained on behavioral outputs, they were able to perform their task accurately when plugged to the original readout vector, implying that they also captured behavioral aspects of the original networks.

LINT provides an inferred low-rank connectivity that we could compare with the original one. We noted that the inferred connectivity weights are not identical to the original ones, although a certain degree of correlation is present (Fig. V.1b and Table 1, column CC for connectivity correlation). However, from a theoretical point of view, a range of different connectivity matrices can lead to identical dynamics (147; 49; 11). The low-rank framework allows for a precise characterization of the relevant part of the connectivity, and in particular allows us to define an effective connectivity matrix $\mathbf{J}^{eff}$ that captures the minimal features required to obtain certain dynamics (keeping only the components of the $\boldsymbol{n}^{(r)}$ vectors that overlap with the $\boldsymbol{m}^{(r)}$ and $\boldsymbol{I}^{(s)}$ vectors, see chapter II). Our results on the synthetic data show that indeed there is a very high degree of similarity between original and inferred effective connectivities, demonstrating that LINT retrieves the relevant aspects of the neural connectivity (Table 1, ECC for Effective Connectivity Correlation, and Fig. V.1c).

In conclusion, our method is able to accurately fit neural trajectories generated by low-rank RNNs, and infers networks that give rise to similar trajectories, behavioral performance, and retrieve the essential connectivity features of the original ones.

## V.4 Application to reverse-engineering full-rank RNNs

We next ask to which extent our method can infer computational mechanisms from activity generated by unconstrained, full-rank networks. Indeed, RNNs trained on simple tasks without a rank constraint often exhibit low-dimensional dynamics that can be related to the way computations unfold in the network (203; 117; 116; 115; 211). Any such low-dimensional dynamics can in principle be reproduced with low-rank networks due to their universal approximation properties (11). Yet the best approach for inferring the corresponding low-rank connectivity remains to be determined. Here we show that our method vastly outperforms a direct low-rank approximation of the connectivity matrix based on truncating the SVD (181). We then demonstrate how the inferred low-rank models can be used to interpret the computational mechanisms in the original full-rank networks.

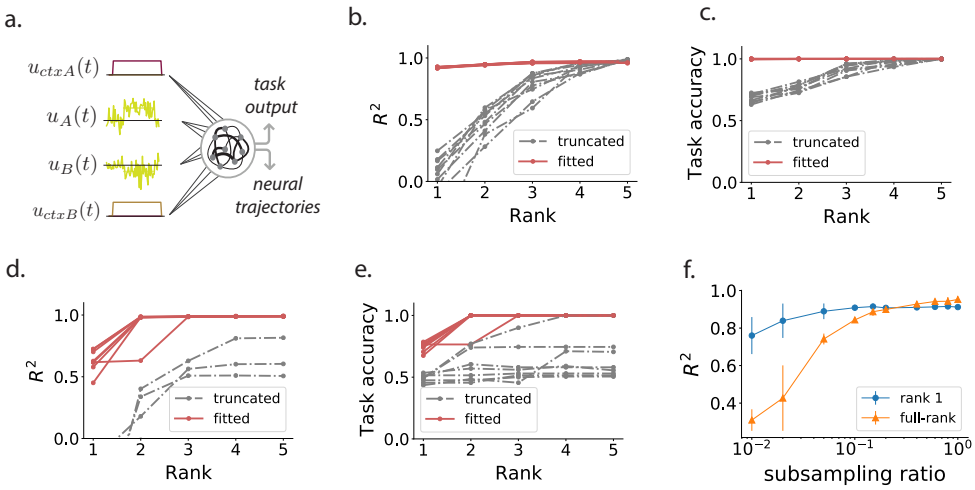


FIGURE V.2: LINT applied to full-rank task-optimized RNNs. a. Description of the CDM task inputs and outputs. Neural trajectories correspond to the neural firing rates, choice output from a linear readout as in eq. V.6. b. For the CDM task, similarity between trajectories produced by the original full-rank network and a series of fitted networks of increasing ranks (red), or low-rank networks obtained by truncating the original connectivity matrix (grey). c. Task accuracy of the same networks when associated with the original task readout. d-e. Same as b-c. for the DMS task. f. For the CDM task, similarities between trajectories of the original and fitted networks of rank one (blue) or full-rank (orange), when networks are fitted only to a random subsample of neurons of the original network. Error bars: mean ± std over 10 random subsamples for each ratio value.

### V.4.1 Extracting low-dimensional dynamics through low-rank connectivity

We consider full-rank, vanilla RNNs trained on two complex systems neuroscience tasks, respectively context-dependent decision-making(118) (CDM) and delayed match-to-sample(27) (DMS). The full-rank RNNs reach a 100% accuracy on each task, and as expected exhibit low-dimensional dynamics. From these RNNs, we generate trajectories corresponding to trials in the trained tasks, and then use LINT to infer lrRNNs of increasing rank.

In the CDM task, a network receives two signal inputs which vary randomly around a positive or negative mean, as well as two binary context cues, one of which is set to 1 in each trial. The network has to output the average sign of the signal indicated by the active context, while ignoring the other signal (Fig. V.2a, see Appendix C.1 for task details). Applying LINT, we found that rank-one networks are sufficient to reproduce the original trajectories with a high goodness of fit. Increasing the rank improved only marginally the fit (Fig. V.2b). In contrast, low-rank networks obtained by truncating the SVD of the trained full-rank connectivity matrix required a rank of at least 9 to reach a comparable fit (Fig. V.2b). This shows that LINT captures a simplified connectivity structure that could not be trivially extracted from the original connectivity. Moreover, even though not explicitly trained on the behavioral outputs, the inferred rank-one networks performed the task correctly when plugged onto the original readout, implying that they captured well the task-related dynamics (Fig. V.2c).

For the DMS task, we similarly found that rank-two networks were able both to capture the dynamics of the original unconstrained network and to accurately perform the task, with no notable improvement when the rank is increased (Fig. V.2d-e). As for the CDM task,

LINT vastly outperformed direct truncation of the original connectivity matrix. Finally, reproducing the experiment over a larger number of unconstrained RNNs, trained with diverse hyperparameters and random seeds leads to similar results (Sup. Fig. C.1).

**Subsampling**. We then investigated the robustness of LINT to the subsampling of neural data. Indeed, in a biological setting, one cannot expect to have access to all units of a network. It is thus important to assess whether we can still recover the relevant low-dimensional dynamics and information from only a handful of neurons of the original network. We considered a full-rank network trained on the CDM task, and fitted networks to its trajectories, either without constraining their rank or by fixing it to one. Without subsampling, the inferred full-rank networks have a very slight advantage over rank-one networks in reproducing the original trajectories. Yet when considering subsamples of neurons, rank-one networks appear to be more robust than unconstrained ones, keeping a good performance until subsampling ratios as low as 1% of original neurons (Fig. V.2f).

### V.4.2 Extracting computational mechanisms from inferred low-rank connectivity

Low-rank models of behavioral tasks open the door to mechanistic interpretations of the underlying dynamics (122; 43). Here we show how LINT allows us to extract computational mechanisms from full-rank networks performing the CDM task.

A first output of LINT is a set of interpretable axes defining a task-related subspace for dimensionality reduction. The inferred rank-one model of the CDM task specifically yields five axes that correspond to the two stimulus inputs, the two context inputs and an internal latent variable, generated by recurrent connectivity, which represents integrated evidence and therefore choice. Projecting the activity of the full-rank network along these axes shows how low-dimensional dynamics transform inputs into the choice output (Fig. V.3). In this case, the axes determined by LINT are closely related to those obtained by standard targeted dimensionality reduction (TDR, Sup. Fig. C.2) (118). However, in contrast with TDR, the axes inferred by LINT correspond to connectivity features causing the low-dimensional dynamics, and provide a method for an unsupervised discovery of recurrently-generated latent variables. In particular, we did not *a priori* specify that the recurrent loop output $\boldsymbol{m}$ should encode choice, but rather observed that choice was generated through this recurrent mechanism.

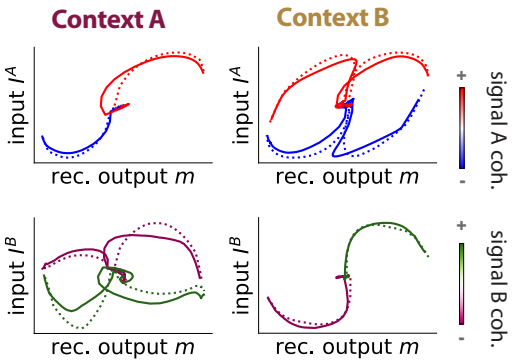


Figure V.3: Low-dimensional projections produced by LINT. Trial-averaged trajectories for different task conditions, in the original full-rank network (full lines) and the inferred low-rank network (dashed lines), projected on axes obtained from the fitted lrRNN connectivity: input A and B axes, and the output axis of the rank-one recurrent connectivity $\boldsymbol{m}$, which encodes choice. All trajectories start at center.

Going beyond dimensionality reduction, LINT extracts an effective low-dimensional model of the task. For the CDM task, the inferred model is a one-dimensional, non-linear latent dynamical system, where the latent variable integrates the relevant evidence. To understand how context-dependent selection of the evidence was performed, we analyzed the connectivity parameters of the inferred rank-one RNN. Indeed, recent work has introduced a clustering method for analyzing low-rank connectivity, and has showed that in trained lrRNNs, context-dependent integration relies on a gain-modulation of latent dynamics mediated

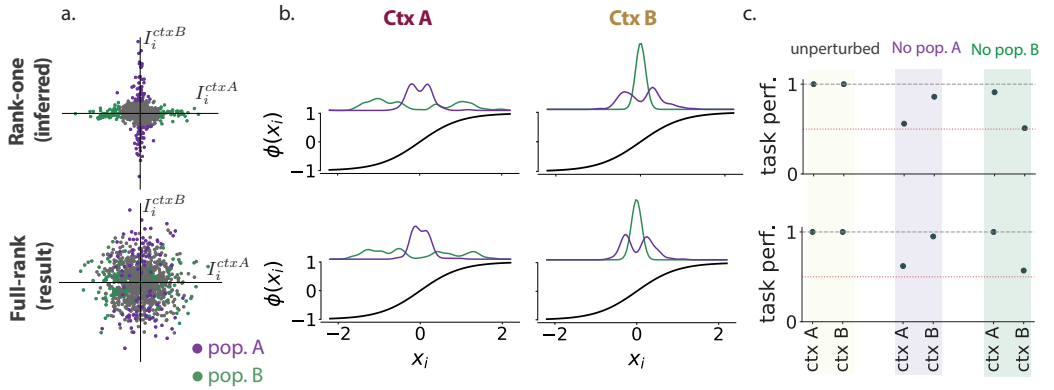


FIGURE V.4: Extracting mechanisms and testing predictions from a low-rank network (top) inferred from a full-rank network trained on the CDM task (bottom). a. Three populations of neurons (grey, green and purple) are identified by clustering connectivity parameters in the fitted rank-one network (top). Here, we plot the same populations in joint distributions of contextual input vectors in the rank-one (top) vs full-rank (bottom) networks. The populations are not directly identifiable from input parameters in the original network (bottom). b. Neural transfer function (tanh) and smoothed distributions of mean neural activity for populations A and B in the inferred rank-one network (top) and the full-rank network (bottom). c. Inactivation experiments: task performance on each context for unperturbed networks and after inactivating populations A and B in the fitted rank-one network (top) and inactivating the same neurons in the original unconstrained network (bottom). Red dots: chance level.

by two distinct populations of neurons (43). Applying a similar clustering approach to rank-one networks inferred by LINT leads to a comparable mechanism. Specifically, this method identifies two populations distinguished by strong values of the fitted weights for the context-cue inputs (Fig. V.4a top and Sup. Fig. C.2). In each context, these context-cue inputs place the neurons belonging to the two populations at different positions on the non-linear transform function $\phi(x)$ (Fig. V.4b top), thereby modulating in opposite ways their gains (defined for each neuron as the local slope $\phi'(x)$). Combined with different statistics of the connectivity parameters on each of the populations, this modulation is sufficient to implement the desired context-switching behavior of the network (see details in Chapter III). The same pattern of gain modulation was found in the original full-rank network, indicating that it is using a similar mechanism (Fig. V.4b bottom).

The computational mechanism extracted using LINT produces predictions for inactivations that can be directly tested in the original full-rank model. The analysis described above assigns neurons to different populations that have specific computational roles. Specifically, in the inferred rank-one network, inactivating separately each population leads to a specific loss of performance in a single context, but not the other one (Fig. V.4c top and Sup. Fig. C.2 for detailed error patterns). Since individual neurons in the original networks directly correspond to neurons in the rank-one network on a one-to-one basis, one can map those populations back onto the original network and reproduce the inactivation experiment. The predicted effects of inactivations on context-dependent performance are directly reproduced in the full-rank network, demonstrating that it relies on the identified computational mechanism (Fig. V.4c bottom and Sup. Fig. C.2). One can note that although the two identified populations clearly stand out in the inferred rank-one connectivity, they are not directly apparent in the original full-rank connectivity (Fig. V.4a bottom), showing that although the mechanism extracted by LINT applies to the full-rank network, using an lrRNN was a

necessary step in this process.

## V.5 Application to neural recordings

We next apply our method to *in vivo* recordings. We consider here electrophysiological recordings from non-human primates performing a context-dependent decision-making task similar to that studied in previous sections (118). More specifically, two macaques (designed by A and F) were presented with random dots stimuli that varied along two dimensions: the overall motion direction of the dots, and their overall color, ranging from two extremes with a set of intermediary coherences in between. At the beginning of each trial, a cue indicated a context for the trial, ordering the subject to report either the average motion or color with an eye saccade, while ignoring the irrelevant evidence. Non-simultaneous recordings were performed in the frontal eye field (FEF) an area of the macaque prefrontal cortex, with respectively 727 neurons for monkey A and 574 neurons for monkey F recorded in all 72 conditions the task presented (ignoring error trials).

For the analysis, we start by binning (5 ms) and smoothing (50 ms std Gaussian window) spike train data, and then compute the trial-averaged response of every neuron in each of the task conditions, forming a pseudo-population tensor of size $N \times T \times 72$ with $N$ the number of neurons in each monkey and $T = 150$ the number of discrete time steps. We denoise this tensor by projecting its first axis on the subspace spanned by the top 12 principal components of the pseudo-population activity, and then project back to the high-dimensional space spanned by all neurons (118). Finally, in order to consider only task-related neural activity, we subtract from each neuron's trajectory its condition-averaged mean. We denote the entries of the final obtained tensor by $\tilde{x}_{i,t,c}$.

We apply LINT to the obtained trajectories, training thestrict recorded activity to the output range of our activation function, here taken to be tanh. Importantly, the inferred networks receive inputs following the same structure than in the CDM task of previous sections, that is two noisy signal inputs and two contextual cue inputs (Fig. V.2a), and are left unconstrained for the first 350 ms of each trial while receiving only contextual cues, to account for the original task procedure. In particular, we make the implicit hypothesis that choice signals are generated by the recurrent activity of the network from received inputs. The quality of fits is quantified by leaving out a random subset of 8 conditions during network inference, and evaluating the $R^2$ of fitted networks on these left-out conditions.

We find that for both monkeys the neural activity is well reproduced by a rank-one network, with minimal improvements when the rank is increased (Fig. V.5a and Sup. Fig. C.5, $R^2 = 0.66$ for monkey A, and $R^2 = 0.57$ for monkey F). Moreover, when simulated independently, the inferred rank-one networks are able to perform the task by adding a linear readout (accuracy curves in Fig. V.5a, Sup. Fig. C.5). This is the case even though the recurrent and input connections were not trained on task performance, demonstrating that the reproduced trajectories contain information about the choice made by the monkey. The activity of both original and reproduced trajectories can moreover be projected on the axes identified by LINT, providing a geometrical picture of task execution (Fig. V.5c). These axes are closely related to those found by targeted dimensionality reduction: notably the context, motion, and color axes identified by TDR closely match the corresponding input axes in the inferred network, while the choice TDR axis can be identified to the $\boldsymbol{m}$ axis of the rank-one recurrent connectivity (Sup. figs. C.4,C.5). The projections of the original activity on the connectivity axes (full lines on Fig. V.5b) therefore shows how inferred connectivity explains the geometry of recorded data.

A closer look at the distribution of inferred connectivity weights provides more information on the neural mechanisms at play (Sup. figs. C.6, C.7). In contrast to the fits of task-optimized networks, we observed for both monkeys a clearly non-normal, heavy-tailed distribution of inferred connectivity parameters (Fisher's kurtosis between 4.9 and 62.7 for different

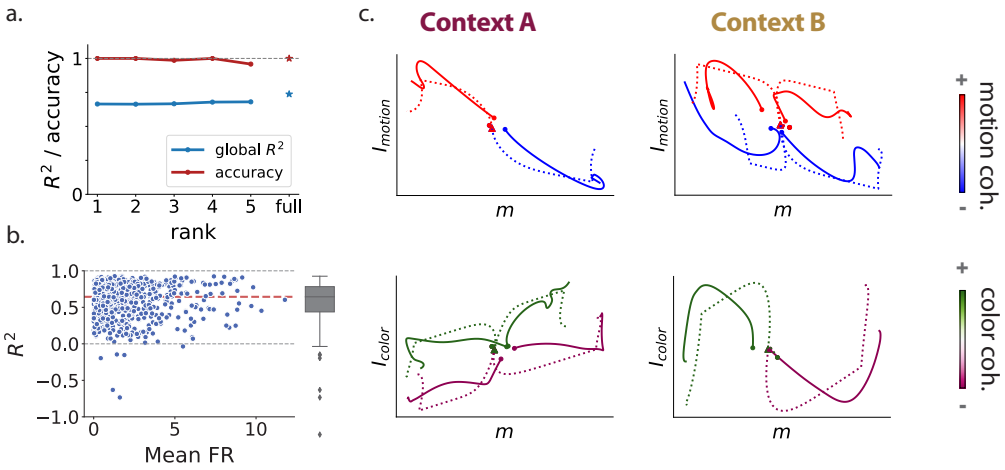


FIGURE V.5: LINT applied to neural recordings from a non-human primate performing a context-dependent task (monkey A) (118). a. Global $R^2$ of model trajectories with respect to original ones, and task accuracy of inferred networks of increasing ranks. b. For the rank-one inferred network, $R^2$ of each recorded neuron plotted against its average data firing rate, median $R^2$ in red, with marginal box-plot of $R^2$ values. 14 neurons with an $R^2 < -1$ are not shown, all with a mean firing rate $< 2.2$ (see Sup. Fig. C.4). c. Trial-averaged trajectories in the recorded data (full lines) and for the rank-one model (dashed lines), for different task conditions (only one coherence value for each coherence sign is plotted here), projected on axes identified by LINT (motion and color input axes and the output axis $\boldsymbol{m}$ of the rank-one recurrent matrix). Trajectories start at the center.

connectivity parameters), echoing some past observations on biological synaptic weights (195; 25). Separating with a clustering algorithm (GMM) large and small-weight neurons shows that the latter can be inactivated without affecting the task performance of the network in a significant way (Sup. Figs. C.6, C.7), even though they represent a majority of neurons in the circuit (570/727 neurons in monkey A, 389/574 in monkey F). This suggests that a minority of neurons with large connectivity supports the computation performed by the network. Among these essential neurons, we identified some neurons that exhibited strong firing rates in the original data, but also a few neurons with small firing rates, which were poorly fit by LINT (with $R^2 < 0$, there are 20/727 such neurons in monkey A, 18/574 in monkey F). Further experiments will be needed to distinguish whether the recorded FEF circuit relies on a few very active neurons to tackle the computational requirements of the task, or whether it relies on an external choice input fed by non-recorded neurons, which LINT would generate by assigning some neurons to a role of "hidden units".

## V.6 Discussion

In this paper, we introduced LINT, a new method for inferring a latent dynamical system from neural recordings based on the theory of low-rank RNNs. This method yields a mechanistic, interpretable and predictive model from neural trajectories, and bridges different levels of analysis, from state-space geometry to neural connectivity.

After verifying the consistency of our method, we demonstrated its potential to extract the mechanisms used by "black-box" vanilla RNNs trained to perform cognitive tasks. In particular, we found that low-rank RNNs could reproduce the dynamics of vanilla RNNs trained to perform diverse tasks, across a range of hyperparameters. Moreover, the obtained lrRNNs provided interpretable low-dimensional projections of the fitted activity, as well as

predictions for the effects of inactivations of specific populations of neurons. These predictions were verified in the original network, validating computational mechanisms derived from low-rank connectivity weights. These results pave the way for a better understanding of how RNNs generate their outputs, which we believe is a very relevant endeavour given the increasing importance of RNNs and related classes of networks across sciences and engineering.

Finally, we applied LINT to electrophysiological recordings in the prefrontal cortex of nonhuman primates performing a context-dependent decision-making task. LINT was able to infer rank-one networks that reproduced both neural activity and task performance, from which low-dimensional projections and interpretable connectivity could be extracted.

It is worth noting that other statistical methods as mTDR (4) can provide a better fit to this data although it does not provide a dynamical system generating the trajectories. Our method is unique in that it bridges that standard state-space geometrical approach of methods like mTDR with a predictive model of the dynamics that methods like LFADS (136) can offer, as well as understanding at the neural connectivity level of how the exhibited dynamics emerge from recurrent interactions.

Methods for inferring latent dynamics from biological data often rely on non-obvious hypotheses about the inputs received by a circuit, and here we hypothesized that choice was recurrently generated within the recorded area. This raises the question of disentangling whether a latent variable is recurrently generated or externally fed to a circuit, which is a particularly important challenge in systems neuroscience, although methods have recently been proposed to address this issue (63; 177; 170). LINT potentially provides a new approach for tackling this question, by comparing inferences made with different possible input structures, by exploiting the potential of single-trial simultaneous recordings which we have not used here, or by building multi-area models from wide-scale recordings that are increasingly available nowadays (189; 183; 197; 143).

# Discussion VI

In this work, we have investigated the properties of recurrent networks of rate neurons with low-rank connectivity. We have demonstrated how these particular networks are amenable to mathematical analysis and geometrical interpretation, how they lead to dynamical patterns similar to those found in recorded neural activity, and how they illuminate neural computations by relating them to the network connectivity structure. Numerous questions naturally remain open, and here we will expose some thoughts and ideas that complete this work or are in our opinion relevant for future exploration.

**Relation between low-rank networks and biology.** The low-rank rate networks that have been studied throughout this work may seem to the biologist a rather implausible abstraction, due to the absence of a large number of physiological constraints. Let us first recall that the main reason why we departed from biology in this thesis was that we were aiming at finding the simplest models able to capture certain properties of neural activity, in particular its computational abilities through distributed, low-dimensional dynamics. As such, it can be understood in its current form more as a study on the structure of non-linear high-dimensional dynamical systems than in biological nervous nets. However, as pointed to by the last chapter, we hope that our theory can lead to insights on the function of biological systems, and this will be a major focus of future work.

A first reason why we believe this is that even if the mathematical functions capturing interactions between neurons are different between models and living beings, both are at their core dynamical systems with computational abilities, and can thus be studied and compared at that more abstract level. Some of the facts outlined in chapter III, such as input-driven reshaping of dynamics, the presence of specialized modules that are activated or de-activated through gain modulation mechanisms, or the presence of feedback loops that integrate evidence remain relevant independently of the implementation details, and could be found in other types of artificial architectures or in biological systems.

A second reason why low-rank networks could prove useful in biological studies is that many of the known physiological constraints can be added to them while maintaining the core effects of low-rank connectivity. For instance, sparsity of recurrent connections is known to immediately render a connectivity matrix full-rank, but can nevertheless be applied on top of a low-rank connectivity while maintaining its low-dimensional induced dynamics (80). Similarly, preliminary results suggest that low-rank networks functioning with spikes instead of rates would exhibit similar properties (for details concerning realistic transfer functions, see the last paragraph of this discussion). Other preliminary results suggest that Dale's law can also be superimposed to a network with low-rank connectivity without affecting its core dynamics. On this subject, the biological existence of specialized excitatory

or inhibitory neurons, and even of multiple inhibitory classes with specific genetic and functional properties might suggest a parallel with the presence of cell populations we found in artificially trained networks in chapter III. Although tempting, this is a parallel we are not willing to make, for our results suggest more the presence of modules with different connectivity statistics but similar units, rather than modules with different types of units. Moreover, the separation between excitatory and inhibitory neurons could be, in our opinion, more fruitfully mapped to the role of gain modulation at the canonical circuit level (see the last paragraph of this discussion).

Our results could be however mapped to the presence of functionally specialized modules that is assessed in biological experiments, and seems to manifest itself notably in conditions were flexible input-output mappings are required (81; 146; 82) (although see (51)). The relationship between those computationally-required modules, and the role of gain modulation in selecting them aligns well with current research into multitasking networks (227; 146; 119; 42), and could be easily expanded into broader ranges of tasks with more complex modular architectures. Moreover, recent results suggest that networks whose design combines interpretable low-rank connectivity with multiple areas can be fruitfully analyzed and mapped to neural data (8), opening the door for more varied applications of those ideas.

**Potential for the study of more complex networks and tasks.** Results from the last chapter suggest that low-rank networks can provide insights into the mechanisms implemented by full-rank networks, generalizing the theory of low-rank RNNs to a more practical use-case. This method opens a vast range of applications, as it can be directly applied to full-rank, vanilla RNNs trained on other tasks, but also to RNNs with gated architectures such as LSTMs and GRUs, which are more often used in the field of artificial intelligence, and which seem to share numerous functional similarities with vanilla RNNs (117).

The study of more intricate, higher-dimensional tasks remains a key challenge for the application of low-rank networks. The results exposed in chapter III concerning the multitasking capabilities of low-rank networks (see Fig. III.7) show that they can be readily applied to tasks that can be decomposed in multiple primitives. Research in multitasking networks often leverages or observes an organization of networks in modules which are activated by contextual cues (227; 119; 120; 146; 42), or rather focuses on task-specialization within orthogonal subspaces activated by the presence of their corresponding inputs (45; 28; 130; 131). These two strategies occur in those distinct situations (contextual switching between tasks, or non-competing processing of distinct inputs), aligning with our findings. They are also incidentally two of the main strategies explored to tackle catastrophic forgetting in ANNs, suggesting very practical applications of these questions. Training networks to perform hierarchically complex tasks, or designing them using insights from low-rank networks would be an interesting next step.

Numerous other tasks are fertile terrains to which the theory of low-rank networks can be applied: the study of timing tasks has in this manner been prolifically studied, leading to new hypotheses about the relationship between neural geometry and generalization (12). In another context, relationships between manifold attractor networks and low-rank networks is established (121), and opens the way to studies of network structures implementing spatial navigation and related hippocampal functionality such as episodic memory. Numerous other tasks rely on low-dimensional activity manifolds, and could thus be prone to similar analyses, for instance Bayesian inference (193) or sequence learning tasks (137; 225). However, it is to be noted that these tasks, like a vast number of those studied in systems neuroscience are intrinsically low-dimensional, and could thus not reflect the full breadth of possibilities of a neural circuit (67). Tackling richer tasks like the interpretation of language seems a more distant goal, although possibly attainable through the study of lower-dimensional primitives.

**Gain modulation and the neural transfer function.** Here, we used the tanh

non-linear transfer function for individual neurons, which presents interesting mathematical properties but leads to unrealistic negative firing rates. Experimental evidence generally suggests that neurons in *in vivo* circuits are better characterized by a rectified linear (ReLU) transfer function. When using ReLU, symmetry around 0 is lost, although preliminary results indicate that variability above or below a baseline firing rate could form a basis for geometric computations that still revolve around a central resting state, as is observed in numerous neural recordings. More concerning is the fact that with non-bounded transfer functions like ReLU, the stability of the network is not guaranteed anymore mathematically, and activity could theoretically diverge towards non-physiological regimes. This does not seem to happen, neither in biological circuits nor in artificial RNNs. A simple mechanism that could explain this control of divergent activity patterns in biological circuits is the presence of inhibition, and what is more, of detailed excitatory-inhibitory balance. One could thus interpret the saturation at high firing rates in our networks as a way to encompass the presence of local inhibition directly at the level of single units, each of them being then comparable to a pyramidal cell with its inhibitory entourage. Here, we wish to emphasize the parallel between the gain modulation mechanism found in our work, that arises from the saturating non-linearity in our models, and proposed mechanisms of gain modulation through inhibitory activity in biological networks. In our case, we were pleasantly surprised to find that multiplicative gain modulation could arise simply from standard additive synaptic interactions, and without any built-in mechanism to multiply firing rates, a possibility that has rarely been described computationally (see (166) for an example). Knowing whether this phenomenon is also found in EI networks with non-bounded transfer functions would thus be of major interest. More generally, understanding the role of inhibitory cells within abstractions such as canonical circuits has often been a prolific idea (26; 75) and we expect further links to non-linear response profiles at the circuit level and gain modulation to be promising future avenues.

On a more technical note, it is straightforward that within ReLU networks, our proposed mechanism of context-switching through gain modulation remains possible, albeit only at the low end of the transfer function. The presence of similar populations with specific computational roles has nevertheless been reported in networks of ReLU units for several tasks including context-dependent decision-making (such mechanisms have been evidenced in, for example, (227; 55)), and seems to suggest a role of the transfer function that would directly extend our mechanism. Understanding how artificial ReLU networks build internal mechanisms to avoid activity divergence remains an interesting direction for mathematical research.

Finally, the mechanism we have outlined empirically demonstrates the essential computational role of gain modulation in neural computations, a primitive that has long been hypothesized to be at the core of many cognitive functions including attention and executive control (53; 148). Here, we showed that gain modulation naturally emerges as the main factor driving performance in our simplified low-rank networks, but also in unconstrained full-rank networks, and although it is considerably more challenging to confirm it in a biological setting recent results show that it could explain in a compelling way some puzzling properties of multi-area neural recordings in a context-dependent task (8). It could in a similar manner explain other intriguing properties of attention, such as the fact that it can increase noise or sometimes decrease firing rates in pyramidal cells (161). Hence, the understanding of contextual control mechanisms in ANNs that we have developed in our work could be extended into the following working hypotheses for similar mechanisms in biological networks: a possibility is that contextual inputs to a circuit, from areas such as the thalamus which have been shown to modify dynamics and response properties of cortical networks (178; 138; 111), could be a mechanism for the brain to select a course of action among several possible ones (at the executive and motor level) and to select the focus of attention (at the sensory level); we expect these contextual inputs to increase the gain of certain clusters of neurons that implement computational primitives relevant to

the task at hand, to the detriment of other clusters of neurons which would see their gain decrease; neurons which see their gain decrease can however see their activity increase, a seemingly paradoxical property; these clusters of neurons could be distinguished from their selectivity properties, although sometimes complex non-linear mixing of several response properties would be needed to differentiate them (see Fig. III.6); finally, we expect this gain modulation mechanism to arise through purely additive interactions, possibly via the mediation of local inhibition. Investigating whether these hypotheses hold, in biological or in more biologically-realistic networks, remains a vast program that we hope to lead to interesting results.

To paraphrase Santiago Ramón y Cajal, there are no problems exhausted by humans, only humans exhausted by problems. With this, and although the number of remaining questions to solve could keep growing, it is time for the author and readers to call this exploration to a rest. We hope this work has convinced them at least of how much potential rests in the recurrent connections and continued dialogue between disciplines, as was the case here between computer science, algebra, statistics, physics and biology, a potential that the author would be more than happy to discuss over a good drink with the reader who has made it to this final point.

# Bibliography VI


[1] H. Adesnik, W. Bruns, H. Taniguchi, Z. J. Huang, and M. Scanziani. A neural circuit for spatial summation in visual cortex. *Nature*, 490(7419):226–231, 2012.

[2] E. D. Adrian. The basis of sensation. 1928.

[3] M. S. Advani, A. M. Saxe, and H. Sompolinsky. High-dimensional dynamics of generalization error in neural networks. *Neural Networks*, 132:428–446, 2020.

[4] M. C. Aoi, V. Mante, and J. W. Pillow. Prefrontal cortex exhibits multidimensional dynamic encoding during decision-making. *Nature neuroscience*, 23(11):1410–1420, 2020.

[5] E. W. Archer, U. Koster, J. W. Pillow, and J. H. Macke. Low-dimensional models of neural population activity in sensory cortical circuits. *Advances in neural information processing systems*, 27, 2014.

[6] D. L. Barack and J. W. Krakauer. Two views on the cognitive brain. *Nature Reviews Neuroscience*, pages 1–13, 2021.

[7] O. Barak. Recurrent neural networks as versatile tools of neuroscience research. *Current opinion in neurobiology*, 46:1–6, 2017.

[8] J. Barbosa, R. Proville, C. C. Rodgers, S. Ostojic, and Y. Boubenec. Flexible selection of task-relevant features through across-area population gating. *bioRxiv*, 2022.

[9] D. G. Barrett, A. S. Morcos, and J. H. Macke. Analyzing biological and artificial neural networks: challenges with opportunities for synergy? *Current opinion in neurobiology*, 55:55–64, 2019.

[10] P. Bashivan, K. Kar, and J. J. DiCarlo. Neural population control via deep image synthesis. *Science*, 364(6439):eaav9436, 2019.

[11] M. Beiran, A. Dubreuil, A. Valente, F. Mastrogiuseppe, and S. Ostojic. Shaping dynamics with multiple populations in low-rank recurrent networks. *Neural Computation*, 33(6):1572–1615, 2021.

[12] M. Beiran, N. Meirhaeghe, H. Sohn, M. Jazayeri, and S. Ostojic. Parametric control of flexible timing through low-dimensional neural manifolds. *bioRxiv*, 2021.

[13] M. Belkin, D. Hsu, S. Ma, and S. Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

[14] G. Bellec, F. Scherr, A. Subramoney, E. Hajek, D. Salaj, R. Legenstein, and W. Maass. A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature communications*, 11(1):1–15, 2020.

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

[16] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[17] C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[18] G. Bondanelli, T. Deneux, B. Bathellier, and S. Ostojic. Network dynamics underlying off responses in the auditory cortex. *Elife*, 10:e53151, 2021.

[19] G. Bondanelli and S. Ostojic. Coding with transient trajectories in recurrent neural networks. *PLoS computational biology*, 16(2):e1007655, 2020.

[20] M. Brecht, M. Schneider, B. Sakmann, and T. W. Margrie. Whisker movements evoked by stimulation of single pyramidal cells in rat motor cortex. *Nature*, 427(6976):704–710, 2004.

[21] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[22] N. Brunel and M. C. Van Rossum. Lapicque's 1907 paper: from frogs to integrate-and-fire. *Biological cybernetics*, 97(5):337–339, 2007.

[23] D. V. Buonomano and W. Maass. State-dependent computations: spatiotemporal processing in cortical networks. *Nature Reviews Neuroscience*, 10(2):113–125, 2009.

[24] Y. Burak and I. R. Fiete. Accurate path integration in continuous attractor network models of grid cells. *PLoS computational biology*, 5(2):e1000291, 2009.

[25] G. Buzsáki and K. Mizuseki. The log-dynamic brain: how skewed distributions affect network operations. *Nature Reviews Neuroscience*, 15(4):264–278, 2014.

[26] M. Carandini and D. J. Heeger. Normalization as a canonical neural computation. *Nature Reviews Neuroscience*, 13(1):51–62, 2012.

[27] W. Chaisangmongkon, S. K. Swaminathan, D. J. Freedman, and X.-J. Wang. Computing by robust transience: how the fronto-parietal network performs sequential, category-based decisions. *Neuron*, 93(6):1504–1517, 2017.

[28] A. Chaudhry, N. Khan, P. Dokania, and P. Torr. Continual learning in low-rank orthogonal subspaces. *Advances in Neural Information Processing Systems*, 33:9900–9911, 2020.

[29] R. Chaudhuri, B. Gerçek, B. Pandey, A. Peyrache, and I. Fiete. The intrinsic attractor manifold and population dynamics of a canonical cognitive circuit across waking and sleep. *Nature neuroscience*, 22(9):1512–1520, 2019.

[30] T. W. Chow and X.-D. Li. Modeling of continuous time dynamical systems with input by recurrent neural networks. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 47(4):575–578, 2000.

[31] M. M. Churchland, M. Y. Byron, M. Sahani, and K. V. Shenoy. Techniques for extracting single-trial activity patterns from large-scale neural recordings. *Current opinion in neurobiology*, 17(5):609–618, 2007.

[32] M. M. Churchland and K. V. Shenoy. Temporal complexity and heterogeneity of single-neuron activity in premotor and motor cortex. *Journal of neurophysiology*, 97(6):4235–4257, 2007.

[33] Z. Cohen, B. DePasquale, M. C. Aoi, and J. W. Pillow. Recurrent dynamics of prefrontal cortex during context-dependent decision-making. *bioRxiv*, 2020.

[34] J. A. Cromer, J. E. Roy, and E. K. Miller. Representation of multiple, independent categories in the primate prefrontal cortex. *Neuron*, 66(5):796–807, 2010.

[35] C. J. Cueva and X.-X. Wei. Emergence of grid-like representations by training recurrent neural networks to perform spatial localization. *arXiv preprint arXiv:1803.07770*, 2018.

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

[37] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[38] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in neural information processing systems*, 27, 2014.

[39] P. Dayan and L. F. Abbott. *Theoretical neuroscience: computational and mathematical modeling of neural systems*. MIT press, 2005.

[40] B. DePasquale, C. J. Cueva, K. Rajan, G. S. Escola, and L. Abbott. full-force: A target-based method for training recurrent networks. *PloS one*, 13(2):e0191527, 2018.

[41] K. Doya. Universality of fully connected recurrent neural networks. *Dept. of Biology, UCSD, Tech. Rep*, 1993.

[42] L. Driscoll, K. Shenoy, and D. Sussillo. Flexible multitask computation in recurrent networks utilizes shared dynamical motifs. *bioRxiv*, 2022.

[43] A. Dubreuil, A. Valente, M. Beiran, F. Mastrogiuseppe, and S. Ostojic. The role of population structure in computations through neural dynamics. *Nature Neuroscience*, 25(6):783–794, 2022.

[44] L. Duncker, G. Bohner, J. Boussard, and M. Sahani. Learning interpretable continuous-time models of latent stochastic dynamical systems. In *International Conference on Machine Learning*, pages 1726–1734. PMLR, 2019.

[45] L. Duncker, L. Driscoll, K. V. Shenoy, M. Sahani, and D. Sussillo. Organizing recurrent network dynamics by task-computation to enable continual learning. *Advances in Neural Information Processing Systems*, 33, 2020.

[46] L. Duncker and M. Sahani. Dynamics on the manifold: Identifying computational dynamical activity from neural population recordings. *Current opinion in neurobiology*, 70:163–170, 2021.

[47] D. Durstewitz. A state space approach for piecewise-linear recurrent neural networks for identifying computational dynamics from neural measurements. *PLoS computational biology*, 13(6):e1005542, 2017.

[48] D. Elgueda, D. Duque, S. Radtke-Schuller, P. Yin, S. V. David, S. A. Shamma, and J. B. Fritz. State-dependent encoding of sound and behavioral meaning in a tertiary region of the ferret auditory cortex. *Nature neuroscience*, 22(3):447–459, 2019.

[49] C. Eliasmith and C. H. Anderson. *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT press, 2003.

[50] T. A. Engel and X.-J. Wang. Same or different? a neural circuit mechanism of similarity-based pattern match decision making. *Journal of Neuroscience*, 31(19):6982–6996, 2011.

[51] B. Engelhard, J. Finkelstein, J. Cox, W. Fleming, H. J. Jang, S. Ornelas, S. A. Koay, S. Y. Thiberge, N. D. Daw, D. W. Tank, et al. Specialized coding of sensory, motor and cognitive variables in vta dopamine neurons. *Nature*, 570(7762):509–513, 2019.

[52] A. Fairhall. The receptive field is dead. long live the receptive field? *Current opinion in neurobiology*, 25:ix–xii, 2014.

[53] K. A. Ferguson and J. A. Cardin. Mechanisms underlying gain modulation in the cortex. *Nature Reviews Neuroscience*, 21(2):80–92, 2020.

[54] A. Finkelstein, L. Fontolan, M. N. Economo, N. Li, S. Romani, and K. Svoboda. Attractor dynamics gate cortical information flow during decision-making. *Nature Neuroscience*, 24(6):843–850, 2021.

[55] T. Flesch, K. Juechems, T. Dumbalska, A. Saxe, and C. Summerfield. Orthogonal representations for robust context-dependent task performance in brains and neural networks. *Neuron*, 110(7):1258–1270, 2022.

[56] J. Frankle and M. Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[57] J. B. Fritz, S. V. David, S. Radtke-Schuller, P. Yin, and S. A. Shamma. Adaptive, behaviorally gated, persistent encoding of task-relevant auditory information in ferret frontal cortex. *Nature neuroscience*, 13(8):1011, 2010.

[58] K. Fukushima and N. Wake. Handwritten alphanumeric character recognition by the neocognitron. *IEEE transactions on Neural Networks*, 2(3):355–365, 1991.

[59] K.-i. Funahashi and Y. Nakamura. Approximation of dynamical systems by continuous time recurrent neural networks. 6(6):801–806, 1993.

[60] S. Fusi, E. K. Miller, and M. Rigotti. Why neurons mix: high dimensionality for higher cognition. *Current opinion in neurobiology*, 37:66–74, 2016.

[61] J. M. Fuster and G. E. Alexander. Neuron activity related to short-term memory. *Science*, 173(3997):652–654, 1971.

[62] M. Fyhn, S. Molden, M. P. Witter, E. I. Moser, and M.-B. Moser. Spatial representation in the entorhinal cortex. *Science*, 2004.

[63] A. R. Galgali, M. Sahani, and V. Mante. Residual dynamics resolves recurrent contributions to neural computation. *bioRxiv*, 2021.

[64] J. A. Gallego, M. G. Perich, L. E. Miller, and S. A. Solla. Neural manifolds for the control of movement. *Neuron*, 94(5):978–984, 2017.

[65] S. Ganguli, D. Huh, and H. Sompolinsky. Memory traces in dynamical systems. *Proceedings of the National Academy of Sciences*, 105(48):18970–18975, 2008.

[66] P. Gao and S. Ganguli. On simplicity and complexity in the brave new world of large-scale neuroscience. *Current opinion in neurobiology*, 32:148–155, 2015.

[67] P. Gao, E. Trautmann, B. Yu, G. Santhanam, S. Ryu, K. Shenoy, and S. Ganguli. A theory of multineuronal dimensionality, dynamics and measurement. *BioRxiv*, page 214262, 2017.

[68] R. J. Gardner, E. Hermansen, M. Pachitariu, Y. Burak, N. A. Baas, B. A. Dunn, M.-B. Moser, and E. I. Moser. Toroidal topology of population activity in grid cells. *Nature*, 602(7895):123–128, 2022.

[69] J. Glaser, M. Whiteway, J. P. Cunningham, L. Paninski, and S. Linderman. Recurrent switching dynamical systems models for multiple interacting neural populations. *Advances in neural information processing systems*, 33:14867–14878, 2020.

[70] M. Glickstein. Golgi and cajal: The neuron doctrine and the 100th anniversary of the 1906 nobel prize. *Current Biology*, 16(5):R147–R151, 2006.

[71] J. I. Gold, M. N. Shadlen, et al. The neural basis of decision making. *Annual review of neuroscience*, 30(1):535–574, 2007.

[72] P. S. Goldman-Rakic. Cellular basis of working memory. *Neuron*, 14(3):477–485, 1995.

[73] A. Guanella, D. Kiper, and P. Verschure. A model of grid cells based on a twisted torus topology. *International journal of neural systems*, 17(04):231–240, 2007.

[74] B. Hangya, H.-J. Pi, D. Kvitsiani, S. P. Ranade, and A. Kepecs. From circuit motifs to computations: mapping the behavioral repertoire of cortical interneurons. *Current opinion in neurobiology*, 26:117–124, 2014.

[75] B. Hangya, H.-J. Pi, D. Kvitsiani, S. P. Ranade, and A. Kepecs. From circuit motifs to computations: mapping the behavioral repertoire of cortical interneurons. *Current opinion in neurobiology*, 26:117–124, 2014.

[76] K. Hardcastle, N. Maheswaranathan, S. Ganguli, and L. M. Giocomo. A multiplexed, heterogeneous, and adaptive code for navigation in medial entorhinal cortex. *Neuron*, 94(2):375–387, 2017.

[77] K. D. Harris and T. D. Mrsic-Flogel. Cortical connectivity and sensory coding. *Nature*, 503(7474):51–58, 2013.

[78] D. O. Hebb. *The organization of behavior*. Wiley, New York, 1949.

[79] G. Hennequin, T. P. Vogels, and W. Gerstner. Optimal control of transient dynamics in balanced networks supports generation of complex movements. *Neuron*, 82(6):1394–1406, 2014.

[80] E. Herbert and S. Ostojic. The impact of sparsity in low-rank recurrent neural networks. *bioRxiv*, 2022.

[81] J. Hirokawa, A. Vaughan, P. Masset, T. Ott, and A. Kepecs. Frontal cortex neuron types categorically encode single decision variables. *Nature*, 576(7787):446–451, 2019.

[82] D. L. Hocker, C. D. Brody, C. Savin, and C. M. Constantinople. Subpopulations of neurons in lofc encode previous and current rewards at time of choice. *eLife*, 10:e70129, 2021.

[83] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[84] D. H. Hubel and T. N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3):574, 1959.

[85] D. H. Hubel and T. N. Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, 195(1):215–243, 1968.

[86] H. Jaeger and H. Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *science*, 304(5667):78–80, 2004.

[87] M. Jazayeri and S. Ostojic. Interpreting neural computations by examining intrinsic and embedding dimensionality of neural activity. *Current opinion in neurobiology*, 70:113–120, 2021.

[88] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82:35–45, 1960.

[89] T.-C. Kao, M. S. Sadabadi, and G. Hennequin. Optimal anticipatory control as a theory of motor preparation: A thalamo-cortical circuit model. *Neuron*, 109(9):1567–1581, 2021.

[90] K. Kar, J. Kubilius, K. Schmidt, E. B. Issa, and J. J. DiCarlo. Evidence that recurrent circuits are critical to the ventral stream's execution of core object recognition behavior. *Nature neuroscience*, 22(6):974–983, 2019.

[91] A. J. Kell, D. L. Yamins, E. N. Shook, S. V. Norman-Haignere, and J. H. McDermott. A task-optimized neural network replicates human auditory behavior, predicts brain responses, and reveals a cortical processing hierarchy. *Neuron*, 98(3):630–644, 2018.

[92] D. R. Kepple, R. Engelken, and K. Rajan. Curriculum learning as a tool to uncover learning principles in the brain. In *International Conference on Learning Representations*, 2021.

[93] T. C. Kietzmann, C. J. Spoerer, L. K. Sörensen, R. M. Cichy, O. Hauk, and N. Kriegeskorte. Recurrence is required to capture the representational dynamics of the human visual system. *Proceedings of the National Academy of Sciences*, 116(43):21854–21863, 2019.

[94] J.-N. Kim and M. N. Shadlen. Neural correlates of a decision in the dorsolateral prefrontal cortex of the macaque. *Nature neuroscience*, 2(2):176–185, 1999.

[95] S.-P. Kim, J. D. Simeral, L. R. Hochberg, J. P. Donoghue, and M. J. Black. Neural control of computer cursor velocity by decoding motor cortical spiking activity in humans with tetraplegia. *Journal of neural engineering*, 5(4):455, 2008.

[96] S. S. Kim, H. Rouault, S. Druckmann, and V. Jayaraman. Ring attractor dynamics in the drosophila central brain. *Science*, 356(6340):849–853, 2017.

[97] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[98] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[99] N. Kostantinos. Gaussian mixtures and their applications to signal processing. *Advanced signal processing handbook: theory and implementation for radar, sonar, and medical imaging real time systems*, pages 3–1, 2000.

[100] K. Kubota and H. Niki. Prefrontal cortical unit activity and delayed alternation performance in monkeys. *Journal of neurophysiology*, 34(3):337–347, 1971.

[101] D. Kvitsiani, S. Ranade, B. Hangya, H. Taniguchi, J. Huang, and A. Kepecs. Distinct behavioural and network correlates of two interneuron types in prefrontal cortex. *Nature*, 498(7454):363–366, 2013.

[102] R. Laje and D. V. Buonomano. Robust timing and motor patterns by taming chaos in recurrent neural networks. *Nature neuroscience*, 16(7):925–933, 2013.

[103] I. D. Landau and H. Sompolinsky. Coherent chaos in a recurrent neural network with structured connectivity. *PLoS computational biology*, 14(12):e1006309, 2018.

[104] I. D. Landau and H. Sompolinsky. Macroscopic fluctuations emerge in balanced networks with incomplete recurrent alignment. *Physical Review Research*, 3(2):023171, 2021.

[105] C. Langdon and T. A. Engel. Latent circuit inference from heterogeneous neural responses during cognitive tasks. *bioRxiv*, 2022.

[106] F. Lanore, N. A. Cayco-Gajic, H. Gurnani, D. Coyle, and R. A. Silver. Cerebellar granule cell axons support high-dimensional representations. *Nature Neuroscience*, 24(8):1142–1150, 2021.

[107] L. Lapicque. Recherches quantitatives sur l'excitation électrique des nerfs traitée comme une polarisation. *Journal of Physiology and Pathololgy*, 9:620–635, 1907.

[108] T. P. Lillicrap, D. Cownden, D. B. Tweed, and C. J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature communications*, 7(1):1–10, 2016.

[109] T. P. Lillicrap, A. Santoro, L. Marris, C. J. Akerman, and G. Hinton. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21(6):335–346, 2020.

[110] S. Linderman, M. Johnson, A. Miller, R. Adams, D. Blei, and L. Paninski. Bayesian learning and inference in recurrent switching linear dynamical systems. In *Artificial Intelligence and Statistics*, pages 914–922. PMLR, 2017.

[111] L. Logiaco, L. Abbott, and S. Escola. Thalamic control of cortical dynamics in a model of flexible motor sequencing. *Cell reports*, 35(9):109090, 2021.

[112] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560, 2002.

[113] C. K. Machens, R. Romo, and C. D. Brody. Functional, but not anatomical, separation of "what" and "when" in prefrontal cortex. *Journal of Neuroscience*, 30(1):350–360, 2010.

[114] J. H. Macke, L. Buesing, J. P. Cunningham, B. M. Yu, K. V. Shenoy, and M. Sahani. Empirical models of spiking in neural populations. *Advances in neural information processing systems*, 24, 2011.

[115] N. Maheswaranathan and D. Sussillo. How recurrent networks implement contextual processing in sentiment analysis. *arXiv preprint arXiv:2004.08013*, 2020.

[116] N. Maheswaranathan, A. Williams, M. Golub, S. Ganguli, and D. Sussillo. Reverse engineering recurrent networks for sentiment classification reveals line attractor dynamics. *Advances in neural information processing systems*, 32, 2019.

[117] N. Maheswaranathan, A. H. Williams, M. D. Golub, S. Ganguli, and D. Sussillo. Universality and individuality in neural dynamics across large populations of recurrent networks. *Advances in neural information processing systems*, 2019:15629, 2019.

[118] V. Mante, D. Sussillo, K. V. Shenoy, and W. T. Newsome. Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature*, 503(7474):78–84, 2013.

[119] C. D. Marton, G. Lajoie, and K. Rajan. Efficient and robust multi-task learning in the brain with modular task primitives. *arXiv preprint arXiv:2105.14108*, 2021.

[120] N. Y. Masse, G. D. Grant, and D. J. Freedman. Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *Proceedings of the National Academy of Sciences*, 115(44):E10467–E10475, 2018.

[121] F. Mastrogiuseppe. Shaping slow activity manifolds with low-rank recurrent networks. In *Cosyne 2019 Workshops*, 2019.

[122] F. Mastrogiuseppe and S. Ostojic. Linking connectivity, dynamics, and computations in low-rank recurrent neural networks. *Neuron*, 99(3):609–623, 2018.

[123] F. Mastrogiuseppe and S. Ostojic. A geometrical analysis of global stability in trained feedback networks. *Neural computation*, 31(6):1139–1182, 2019.

[124] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[125] M. Minsky and S. Papert. Perceptrons: an introduction to computational geometry. *Cambridge tiass., HIT*, 479:480, 1969.

[126] Y. Miyashita. Neuronal correlate of visual associative long-term memory in the primate temporal cortex. *Nature*, 335(6193):817–820, 1988.

[127] M. Molano-Mazon, D. Duque, G. R. Yang, and J. de la Rocha. Pre-training rnns on ecologically relevant tasks explains sub-optimal behavioral reset. *bioRxiv*, 2021.

[128] E. I. Moser, E. Kropff, M.-B. Moser, et al. Place cells, grid cells, and the brain's spatial representation system. *Annual review of neuroscience*, 31(1):69–89, 2008.

[129] E. I. Moser, M.-B. Moser, and B. L. McNaughton. Spatial representation in the hippocampal formation: a history. *Nature neuroscience*, 20(11):1448, 2017.

[130] L. B. Naumann, J. Keijser, and H. Sprekeler. Invariant neural subspaces maintained by feedback modulation. *Elife*, 11:e76096, 2022.

[131] R. Nogueira, C. C. Rodgers, R. M. Bruno, and S. Fusi. The geometry of cortical representations of touch in rodents. *bioRxiv*, pages 2021–02, 2021.

[132] M. Nonnenmacher, S. C. Turaga, and J. H. Macke. Extracting low-dimensional dynamics from multiple large-scale neural population recordings by learning to predict correlations. *Advances in Neural Information Processing Systems*, 30, 2017.

[133] J. O'Keefe and J. Dostrovsky. The hippocampus as a spatial map: Preliminary evidence from unit activity in the freely-moving rat. *Brain research*, 1971.

[134] C. O'Neil. *Weapons of math destruction: How big data increases inequality and threatens democracy*. Broadway books, 2016.

[135] M. Pachitariu, B. Petreska, and M. Sahani. Recurrent linear models of simultaneously-recorded neural populations. *Advances in neural information processing systems*, 26, 2013.

[136] C. Pandarinath, D. J. O'Shea, J. Collins, R. Jozefowicz, S. D. Stavisky, J. C. Kao, E. M. Trautmann, M. T. Kaufman, S. I. Ryu, L. R. Hochberg, et al. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature methods*, 15(10):805–815, 2018.

[137] M. F. Panichello and T. J. Buschman. Shared mechanisms underlie the control of working memory and attention. *Nature*, 592(7855):601–605, 2021.

[138] M. B. Pardi, J. Vogenstahl, T. Dalmay, T. Spanò, D.-L. Pu, L. B. Naumann, F. Kretschmer, H. Sprekeler, and J. J. Letzkus. A thalamocortical top-down circuit for associative memory. *Science*, 370(6518):844–848, 2020.

[139] I. M. Park, M. L. Meister, A. C. Huk, and J. W. Pillow. Encoding and decoding in parietal cortex during sensorimotor decision-making. *Nature neuroscience*, 17(10):1395, 2014.

[140] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

[141] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[142] U. Pereira and N. Brunel. Attractor dynamics in networks with learning rules inferred from in vivo data. *Neuron*, 99(1):227–238, 2018.

[143] M. G. Perich, C. Arlt, S. Soares, M. E. Young, C. P. Mosher, J. Minxha, E. Carter, U. Rutishauser, P. H. Rudebeck, C. D. Harvey, et al. Inferring brain-wide interactions using data-constrained recurrent neural network models. *BioRxiv*, pages 2020–12, 2021.

[144] B. Petreska, B. M. Yu, J. P. Cunningham, G. Santhanam, S. Ryu, K. V. Shenoy, and M. Sahani. Dynamical segmentation of single trials from population neural data. *Advances in neural information processing systems*, 24, 2011.

[145] L. Pinto and Y. Dan. Cell-type-specific activity in prefrontal cortex during goal-directed behavior. *Neuron*, 87(2):437–450, 2015.

[146] L. Pinto, K. Rajan, B. DePasquale, S. Y. Thiberge, D. W. Tank, and C. D. Brody. Task-dependent changes in the large-scale dynamics and necessity of cortical regions. *Neuron*, 104(4):810–824, 2019.

[147] E. Pollock and M. Jazayeri. Engineering recurrent neural networks from task-relevant manifolds and dynamics. *PLoS computational biology*, 16(8):e1008128, 2020.

[148] N. C. Rabinowitz, R. L. Goris, M. Cohen, and E. P. Simoncelli. Attention stabilizes the shared gain of v4 populations. *Elife*, 4:e08998, 2015.

[149] K. Rajan, L. Abbott, and H. Sompolinsky. Stimulus-dependent suppression of chaos in recurrent neural networks. *Physical review e*, 82(1):011903, 2010.

[150] K. Rajan, C. D. Harvey, and D. W. Tank. Recurrent network models of sequence generation and memory. *Neuron*, 90(1):128–142, 2016.

[151] R. P. Rao. Bayesian computation in recurrent neural circuits. *Neural computation*, 16(1):1–38, 2004.

[152] D. Raposo, M. T. Kaufman, and A. K. Churchland. A category-free neural population supports evolving demands during decision-making. *Nature neuroscience*, 17(12):1784, 2014.

[153] E. D. Remington, D. Narain, E. A. Hosseini, and M. Jazayeri. Flexible sensorimotor computations through rapid reconfiguration of cortical dynamics. *Neuron*, 98(5):1005–1019, 2018.

[154] B. A. Richards, T. P. Lillicrap, P. Beaudoin, Y. Bengio, R. Bogacz, A. Christensen, C. Clopath, R. P. Costa, A. de Berker, S. Ganguli, et al. A deep learning framework for neuroscience. *Nature neuroscience*, 22(11):1761–1770, 2019.

[155] M. Rigotti, O. Barak, M. R. Warden, X.-J. Wang, N. D. Daw, E. K. Miller, and S. Fusi. The importance of mixed selectivity in complex cognitive tasks. *Nature*, 497(7451):585–590, 2013.

[156] A. Rivkind and O. Barak. Local dynamics in trained recurrent neural networks. *Physical review letters*, 118(25):258101, 2017.

[157] R. Romo, C. D. Brody, A. Hernández, and L. Lemus. Neuronal correlates of parametric working memory in the prefrontal cortex. *Nature*, 399(6735):470–473, 1999.

[158] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[159] S. Roweis and Z. Ghahramani. A unifying review of linear gaussian models. *Neural computation*, 11(2):305–345, 1999.

[160] J. E. Roy, M. Riesenhuber, T. Poggio, and E. K. Miller. Prefrontal cortex activity during flexible categorization. *Journal of Neuroscience*, 30(25):8519–8528, 2010.

[161] D. A. Ruff and M. R. Cohen. Attention can either increase or decrease spike count correlations in visual cortex. *Nature neuroscience*, 17(11):1591–1597, 2014.

[162] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[163] A. A. Russo, S. R. Bittner, S. M. Perkins, J. S. Seely, B. M. London, A. H. Lara, A. Miri, N. J. Marshall, A. Kohn, T. M. Jessell, et al. Motor cortex embeds muscle-like commands in an untangled population response. *Neuron*, 97(4):953–966, 2018.

[164] J. Sacramento, R. Ponte Costa, Y. Bengio, and W. Senn. Dendritic cortical microcircuits approximate the backpropagation algorithm. *Advances in neural information processing systems*, 31, 2018.

[165] K. Sakai. Task set and prefrontal cortex. *Annu. Rev. Neurosci.*, 31:219–245, 2008.

[166] E. Salinas. Context-dependent selection of visuomotor maps. *BMC neuroscience*, 5(1):1–22, 2004.

[167] E. Salinas and P. Thier. Gain modulation: a major computational principle of the central nervous system. *Neuron*, 27(1):15–21, 2000.

[168] O. G. Sani, H. Abbaspourazad, Y. T. Wong, B. Pesaran, and M. M. Shanechi. Modeling behaviorally relevant neural dynamics enabled by preferential subspace identification. *Nature Neuroscience*, 24(1):140–149, 2021.

[169] O. G. Sani, B. Pesaran, and M. M. Shanechi. Where is all the nonlinearity: flexible nonlinear modeling of behaviorally relevant neural dynamics using recurrent neural networks. *bioRxiv*, 2021.

[170] B. A. Sauerbrei, J.-Z. Guo, J. D. Cohen, M. Mischiati, W. Guo, M. Kabra, N. Verma, B. Mensh, K. Branson, and A. W. Hantman. Cortical pattern generation during dexterous movement is input-driven. *Nature*, 577(7790):386–391, 2020.

[171] A. Saxe, S. Nelli, and C. Summerfield. If deep learning is the answer, what is the question? *Nature Reviews Neuroscience*, 22(1):55–67, 2021.

[172] A. M. Saxe, J. L. McClelland, and S. Ganguli. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116(23):11537–11546, 2019.

[173] S. Saxena and J. P. Cunningham. Towards the neural population doctrine. *Current opinion in neurobiology*, 55:103–111, 2019.

[174] S. Saxena, A. A. Russo, J. Cunningham, and M. M. Churchland. Motor cortex activity across movement speeds is predicted by network-level strategies for generating muscle activity. *Elife*, 11:e67620, 2022.

[175] R. Schaeffer, M. Khona, L. Meshulam, I. Fiete, et al. Reverse-engineering recurrent neural network solutions to a hierarchical inference task for mice. *Advances in Neural Information Processing Systems*, 33, 2020.

[176] L. K. Scheffer, C. S. Xu, M. Januszewski, Z. Lu, S.-y. Takemura, K. J. Hayworth, G. B. Huang, K. Shinomiya, J. Maitlin-Shepard, S. Berg, et al. A connectome and analysis of the adult drosophila central brain. *Elife*, 9:e57443, 2020.

[177] M. Schimel, T.-C. Kao, K. T. Jensen, and G. Hennequin. ilqr-vae: control-based learning of input-driven dynamics with applications to neural data. *bioRxiv*, 2021.

[178] L. I. Schmitt, R. D. Wimmer, M. Nakajima, M. Happ, S. Mofakham, and M. M. Halassa. Thalamic amplification of cortical connectivity sustains attentional control. *Nature*, 545(7653):219–223, 2017.

[179] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[180] F. Schuessler, A. Dubreuil, F. Mastrogiuseppe, S. Ostojic, and O. Barak. Dynamics of random recurrent networks with correlated low-rank structure. *Physical Review Research*, 2(1):013111, 2020.

[181] F. Schuessler, F. Mastrogiuseppe, A. Dubreuil, S. Ostojic, and O. Barak. The interplay between randomness and structure during learning in rnns, 2020.

[182] J. Semedo, A. Zandvakili, A. Kohn, C. K. Machens, and B. M. Yu. Extracting latent structure from multiple interacting neural populations. *Advances in neural information processing systems*, 27, 2014.

[183] J. D. Semedo, A. Zandvakili, C. K. Machens, M. Y. Byron, and A. Kohn. Cortical areas interact through a communication subspace. *Neuron*, 102(1):249–259, 2019.

[184] H. S. Seung. How the brain keeps the eyes still. *Proceedings of the National Academy of Sciences*, 93(23):13339–13344, 1996.

[185] S. Seung. *Connectome: How the brain's wiring makes us who we are*. HMH, 2012.

[186] M. N. Shadlen and W. T. Newsome. Neural basis of a perceptual decision in the parietal cortex (area lip) of the rhesus monkey. *Journal of neurophysiology*, 86(4):1916–1936, 2001.

[187] S. M. Sherman and R. Guillery. On the actions that one nerve cell can have on another: distinguishing "drivers" from "modulators". *Proceedings of the National Academy of Sciences*, 95(12):7121–7126, 1998.

[188] C. Sherrington. *The integrative action of the nervous system*. CUP Archive, 1906.

[189] M. Siegel, T. J. Buschman, and E. K. Miller. Cortical information flow during flexible sensorimotor decisions. *Science*, 348(6241):1352–1355, 2015.

[190] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[191] A. C. Smith and E. N. Brown. Estimating a state-space model from point process observations. *Neural computation*, 15(5):965–991, 2003.

[192] J. Smith, S. Linderman, and D. Sussillo. Reverse engineering recurrent neural networks with jacobian switching linear dynamical systems. *Advances in Neural Information Processing Systems*, 34:16700–16713, 2021.

[193] H. Sohn, D. Narain, N. Meirhaeghe, and M. Jazayeri. Bayesian computation through cortical latent dynamics. *Neuron*, 103(5):934–947, 2019.

[194] H. Sompolinsky, A. Crisanti, and H.-J. Sommers. Chaos in random neural networks. *Physical review letters*, 61(3):259, 1988.

[195] S. Song, P. J. Sjöström, M. Reigl, S. Nelson, and D. B. Chklovskii. Highly nonrandom features of synaptic connectivity in local cortical circuits. *PLoS biology*, 3(3):e68, 2005.

[196] N. A. Steinmetz, P. Zatka-Haas, M. Carandini, and K. D. Harris. Distributed coding of choice, action and engagement across the mouse brain. *Nature*, 576(7786):266–273, 2019.

[197] N. A. Steinmetz, P. Zatka-Haas, M. Carandini, and K. D. Harris. Distributed coding of choice, action and engagement across the mouse brain. *Nature*, 576(7786):266–273, 2019.

[198] C. Stringer, M. Pachitariu, N. Steinmetz, M. Carandini, and K. D. Harris. High-dimensional geometry of population responses in visual cortex. *Nature*, 571(7765):361–365, 2019.

[199] C. Stringer, M. Pachitariu, N. Steinmetz, C. B. Reddy, M. Carandini, and K. D. Harris. Spontaneous behaviors drive multidimensional, brainwide activity. *Science*, 364(6437):eaav7893, 2019.

[200] S. H. Strogatz. *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. CRC press, 2018.

[201] J. P. Stroud, M. A. Porter, G. Hennequin, and T. P. Vogels. Motor primitives in space and time via targeted gain modulation in cortical networks. *Nature neuroscience*, 21(12):1774–1783, 2018.

[202] L. Susman, F. Mastrogiuseppe, N. Brenner, and O. Barak. Quality of internal representation shapes learning performance in feedback neural networks. *Physical Review Research*, 3(1):013176, 2021.

[203] D. Sussillo. Neural circuits as computational dynamical systems. *Current opinion in neurobiology*, 25:156–163, 2014.

[204] D. Sussillo and L. F. Abbott. Generating coherent patterns of activity from chaotic neural networks. *Neuron*, 63(4):544–557, 2009.

[205] D. Sussillo and O. Barak. Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural computation*, 25(3):626–649, 2013.

[206] D. Sussillo, M. M. Churchland, M. T. Kaufman, and K. V. Shenoy. A neural network that finds a naturalistic solution for the production of muscle activity. *Nature neuroscience*, 18(7):1025–1033, 2015.

[207] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[208] R. J. Townshend, S. Eismann, A. M. Watkins, R. Rangan, M. Karelina, R. Das, and R. O. Dror. Geometric deep learning of rna structure. *Science*, 373(6558):1047–1051, 2021.

[209] D. Y. Tsao and M. S. Livingstone. Mechanisms of face perception. *Annual review of neuroscience*, 31:411, 2008.

[210] K. Tunyasuvunakool, J. Adler, Z. Wu, T. Green, M. Zielinski, A. Žídek, A. Bridgland, A. Cowie, C. Meyer, A. Laydon, et al. Highly accurate protein structure prediction for the human proteome. *Nature*, 596(7873):590–596, 2021.

[211] E. Turner, K. V. Dabholkar, and O. Barak. Charting and navigating the space of solutions for recurrent neural networks. *Advances in Neural Information Processing Systems*, 34:25320–25333, 2021.

[212] A. Valente, S. Ostojic, and J. Pillow. Probing the relationship between linear dynamical systems and low-rank recurrent neural network models. *Neural Computation*, 2022.

[213] J. L. Van Hemmen and T. J. Sejnowski. *23 problems in systems neuroscience*. Oxford University Press, 2005.

[214] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[215] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[216] T. P. Vogels, K. Rajan, L. F. Abbott, et al. Neural network dynamics. *Annual review of neuroscience*, 28:357, 2005.

[217] S. Vyas, M. D. Golub, D. Sussillo, and K. V. Shenoy. Computation through neural population dynamics. *Annual Review of Neuroscience*, 43:249–275, 2020.

[218] M. J. Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*. Cambridge University Press, 2019.

[219] J. Wang, D. Narain, E. A. Hosseini, and M. Jazayeri. Flexible timing by temporal scaling of cortical responses. *Nature neuroscience*, 21(1):102–110, 2018.

[220] X.-J. Wang. Probabilistic decision making by slow reverberation in cortical circuits. *Neuron*, 36(5):955–968, 2002.

[221] X.-J. Wang. Decision making in recurrent neuronal circuits. *Neuron*, 60(2):215–234, 2008.

[222] M. Welling. The kalman filter. *Lecture Note*, pages 92–117, 2010.

[223] P. J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[224] A. H. Williams, T. H. Kim, F. Wang, S. Vyas, S. I. Ryu, K. V. Shenoy, M. Schnitzer, T. G. Kolda, and S. Ganguli. Unsupervised discovery of demixed, low-dimensional neural dynamics across multiple timescales through tensor component analysis. *Neuron*, 98(6):1099–1115, 2018.

[225] Y. Xie, P. Hu, J. Li, J. Chen, W. Song, X.-J. Wang, T. Yang, S. Dehaene, S. Tang, B. Min, et al. Geometry of sequence working memory in macaque prefrontal cortex. *Science*, 375(6581):632–639, 2022.

[226] D. L. Yamins, H. Hong, C. F. Cadieu, E. A. Solomon, D. Seibert, and J. 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.

[227] G. R. Yang, M. R. Joglekar, H. F. Song, W. T. Newsome, and X.-J. Wang. Task representations in neural networks trained to perform many cognitive tasks. *Nature neuroscience*, 22(2):297–306, 2019.

[228] G. R. Yang and M. Molano-Mazón. Towards the next generation of recurrent network models for cognitive neuroscience. *Current Opinion in Neurobiology*, 70:182–192, 2021.

[229] G. R. Yang and X.-J. Wang. Artificial neural networks for neuroscientists: A primer. *Neuron*, 107(6):1048–1070, 2020.

[230] L. Ye, W. E. Allen, K. R. Thompson, Q. Tian, B. Hsueh, C. Ramakrishnan, A.-C. Wang, J. H. Jennings, A. Adhikari, C. H. Halpern, et al. Wiring and molecular features of prefrontal ensembles representing distinct experiences. *Cell*, 165(7):1776–1788, 2016.

[231] B. M. Yu, A. Afshar, G. Santhanam, S. Ryu, K. V. Shenoy, and M. Sahani. Extracting dynamical structure embedded in neural activity. *Advances in neural information processing systems*, 18, 2005.

[232] B. M. Yu, K. V. Shenoy, and M. Sahani. Derivation of kalman filtering and smoothing equations. In *Technical report*. Stanford University, 2004.

[233] R. Yuste. From the neuron doctrine to neural networks. *Nature reviews neuroscience*, 16(8):487–497, 2015.

[234] A. M. Zador. A critique of pure learning and what artificial neural networks can learn from animal brains. *Nature communications*, 10(1):1–7, 2019.

[235] F. Zenke, B. Poole, and S. Ganguli. Continual learning through synaptic intelligence. *International Conference on Machine Learning*, pages 3987–3995, 2017.

[236] K. Zhang. Representation of spatial orientation by the intrinsic dynamics of the head-direction cell ensemble: a theory. *Journal of Neuroscience*, 16(6):2112–2126, 1996.

[237] P. Znamenskiy and A. M. Zador. Corticostriatal neurons in auditory cortex drive decisions during auditory discrimination. *Nature*, 497(7450):482–485, 2013.

[238] D. Zoltowski, J. Pillow, and S. Linderman. A general recurrent state space framework for modeling neural dynamics during decision-making. In *International Conference on Machine Learning*, pages 11680–11691. PMLR, 2020.

# Supplements for chapter III A

## A.1 Parametrization and collective dynamics for mixture of Gaussians connectivity vectors

In this section we show how connectivity vectors with entries drawn from mixtures of multivariate Gaussians can be constructed from independent Gaussians, as mentioned in Eq. (A.25). We then derive the dynamics of the internal collective variables (Eq. (II.28)) in this setting.

We considered distributions of connectivity parameters characterized by $P$ covariance matrices $\mathbf{\Sigma}_p$, and zero means $\boldsymbol{\mu}_p = \mathbf{0}$, $p = 1, \dots, P$. For a neuron $i$ belonging to population $p$, each vector entry $a_i \in \{n_i^{(r)}, m_i^{(r)}, I_i^{(s)}, w_i\}$ is constructed as a linear transformation of the same set of values $\{X_i^{(d)}\}_{d=1\dots N_{in}+2R+1}$

$$a_i = \sum_{d=1}^{N_{in}+2R+1} b_{a,d}^{(p)} X_i^{(d)}. \tag{A.1}$$

Here the $X_i^{(d)}$ are drawn from $\mathcal{N}(0,1)$, independently for each $i$ and $d$. The linear coefficients $\{b_{a,d}\}_{d=1\dots N_{in}+2R+1}$ are different for each connectivity vector $a \in \{n^{(r)}, m^{(r)}, I^{(s)}, w\}$, but identical across neurons within a given population. These sets of coefficients therefore determine the covariance $\sigma_{ab}^{(p)}$ between entries of connectivity vectors within a given population $p$:

$$\sigma_{ab}^{(p)} = \sum_{d=1}^{D} b_{a,d}^{(p)} b_{b,d}^{(p)} = (\boldsymbol{b}_a^{(p)})^T \boldsymbol{b}_b^{(p)} \tag{A.2}$$

The row-vectors ${\boldsymbol{b}_a^{(p)}}^T$ in fact correspond to the rows of the Cholesky factorization of the covariance matrix.
We next turn to the derivation of Eq. (II.28). With the parametrization for the entries of connectivity vectors defined in Eq. (A.1), the recurrent inputs to the $r$-th internal collective variable Eq. (II.21) can be written as

$$\kappa_r^{rec} = \sum_{p=1}^{P} \alpha_p \int \left( \prod_{d=1}^{D} DX^{(d)} \right) \sum_{d=1}^{D} b_{n^{(r)},d}^{(p)} X^{(d)} \phi \left( \sum_{r'=1}^{R} \kappa_{r'} \sum_{d=1}^{D} b_{m^{(r')},d}^{(p)} X^{(d)} + \sum_{s=1}^{N_{in}} v_s \sum_{d=1}^{D} b_{I^{(s)},d}^{(p)} X^{(d)} \right) \tag{A.3}$$

with $DX^{(d)} = \frac{dX^{(d)}}{\sqrt{2\pi}} e^{-(X^{(d)})^2/2}$. For a given $p$, we then compute each of the $D$ integrals $\int \left(\prod_{d=1}^{D} DX^{(d)}\right) b^{(p)}_{n^{(r)},d} X^{(d)} \phi\left(\ldots\right)$ applying successively Stein's lemma

$$\int Dz\, z f(z) = \int Dz f'(z), \tag{A.4}$$

and using the fact that a sum of independent Gaussians is a Gaussian with variance given by the sum of variances, so that

$$\int DxDy \ldots f(\alpha x + \beta y + \ldots) = \int Dz f(\sqrt{\alpha^2 + \beta^2 + ...} z). \tag{A.5}$$

This leads to

$$\begin{aligned} \kappa_r^{rec} &= \sum_{p=1}^{P} \alpha_p \sum_{d=1}^{D} b^{(p)}_{n^{(r)},d} \left( \sum_{r'=1}^{R} b^{(p)}_{m^{(r')},d} \kappa_{r'} + \sum_{s=1}^{N_{in}} b^{(p)}_{I^{(s)},d} v_s \right) \int Dz \phi'\left(\Delta^{(p)} z\right) \\ &= \sum_{p=1}^{P} \alpha_p \left( \sum_{r'=1}^{R} \sigma^{(p)}_{n^{(r)} m^{(r')}} \kappa_{r'} + \sum_{s=1}^{N_{in}} \sigma^{(p)}_{n^{(r)} I^{(s)}} v_s \right) \int Dz \phi'\left(\Delta^{(p)} z\right) \end{aligned} \tag{A.6}$$

with

$$\Delta^{(p)} = \sqrt{\sum_{r'=1}^{R} (\sigma^{(p)}_{m^{(r')} m^{(r')}})^2 \kappa_{r'}^2 + \sum_{s=1}^{N_{in}} (\sigma^{(p)}_{I^{(s)} I^{(s)}})^2 u_s^2}. \tag{A.7}$$

Inverting the sums on $p$ and $r', s$ indices and assuming that input vectors $\boldsymbol{I}^{(s)}$ are orthogonal to the output vectors $\{\boldsymbol{m}^{(r)}\}_{r=1,\ldots,R}$ (as in all the reduced models described in the section below), we get the compact description in terms of effective couplings for the dynamics of internal collective variables Eq. (II.28)

$$\frac{d\kappa_r}{dt} = -\kappa_r + \sum_{r'=1}^{R} \tilde{\sigma}_{n^{(r)} m^{(r')}} \kappa_{r'} + \sum_{s=1}^{N_{in}} \tilde{\sigma}_{n^{(r)} I^{(s)}} v_s \tag{A.8}$$

with, for any two vectors $\boldsymbol{a}, \boldsymbol{b}$, the effective couplings

$$\tilde{\sigma}_{ab} = \sum_{p=1}^{P} \sigma^{(p)}_{ab} \langle \Phi' \rangle_p \tag{A.9}$$

and averaged gains

$$\langle \Phi' \rangle_p = \int Dz \phi'(\Delta^{(p)} z). \tag{A.10}$$

## A.2 Alternative implementation of the CDM task

The solution found by training a unit rank network illustrated in figures III.3 and III.4 is not necessarily unique, and indeed when training rank one networks on the same task with different hyperparameters or training procedures we found alternative implementations, all relying on non-random population structure. Here, we detail an implementation based on 3 functional sub-populations, illustrated in A.6, reminiscent of implementations already described in the literature (227), and used in the main text for testing predictions (Fig. III.6) and describing a multi-tasking mechanism (Fig. III.7).

More specifically, for this network trained with fixed input weights and modifiable recurrent connectivity, 4 sub-populations were needed to explain the implemented computational mechanism, in the sense that resampling networks from a Gaussian mixture model fitted to the trained connectivity gave functioning networks with four sub-populations and not with a lower number (Sup. Fig. A.6a). These sub-populations all had a zero mean in the connectivity space but diverse covariance structures, as can be seen from projections of the 7-dimensional connectivity space (Sup. Fig. A.6b) and the fitted covariance matrices (Sup. Fig. A.6c). We observed that among the four sub-populations found by the clustering procedure, the largest one (shown in grey) did not bear any computational role in this task, and in particular inactivating it did not result in any performance loss (Sup. Fig. A.6e). Sub-population 4 (illustrated in green) was characterized by strong entries on the $\boldsymbol{m}$ vector and a positive covariance between its $\boldsymbol{m}$ and $\boldsymbol{n}$ entries, showing that it performed the effective evidence integration. Sub-population 2 (in purple) presented strong entries on the $\boldsymbol{I}^{ctxB}$ vector along with a positive covariance between its $\boldsymbol{n}$ and $\boldsymbol{I}^{A}$ entries, showing it transmitted the entry signal $u_A$ to the integratory feedback loop driven by the $\boldsymbol{n}$-$\boldsymbol{m}$ loop and supported by sub-population 4, unless it was driven to a low-gain regime by the strong entries on $\boldsymbol{I}^{ctxB}$. Note that the effective couplings between input and recurrence vectors that drive the computation have to be computed at the level of the whole network (following Eq. (III.5)), even though these couplings might be supported by covariance structures in only one sub-population. In a complementary manner, sub-population 3 (in orange) had strong entries on the $\boldsymbol{I}^{ctxA}$ vector and a positive overlap between its $\boldsymbol{I}^{B}$ and $\boldsymbol{n}$ entries.

The mechanism can thus be understood as follows (Sup. Fig. A.6d): in context A, strong weights on the $\boldsymbol{I}^{ctxA}$ drive sub-population 3 to a low-gain regime, preventing it from relaying input signal B to the integratory feedback loop supported by sub-population 4. The opposite phenomenon occurs in context B with sub-population 2 being driven to a low-gain regime (Sup. Fig. A.6f). This interpretation of the implemented mechanism can be verified by inactivating each sub-population (Sup. Fig. A.6e): without sub-population 2, the network is unable to perform correctly in context A, but its performance in context B is unaffected, while the converse is true without sub-population 3. Finally, without sub-population 4 the network is unable to integrate any evidence and thus to perform the task in any context.

## A.3 Methods

### A.3.1 Recurrent Neural Networks

We considered networks of $N$ rate units that evolve over time according to

$$\tau \frac{dx_i}{dt} = -x_i + \sum_{j=1}^{N} J_{ij}\phi(x_j) + I_i^{FF}(t) + \eta_i(t). \tag{A.11}$$

Here $x_i$ represents the *activation* or total current received by the $i$-th unit, and $\phi(x_i) = \tanh(x_i)$ is its firing rate. Moreover, each neuron received a feed-forward input $I_i^{FF}$ and an independent white-noise input $\eta_i(t)$ specified below.
The recurrent connectivity is set by the connectivity matrix $\mathbf{J} = \{J_{ij}\}_{i,j=1\ldots N}$. For full-rank networks, the coefficients $J_{ij}$ were treated as independent parameters. For low-rank networks $\mathbf{J}$ was constrained to be of rank $R$, and parametrized as

$$J_{ij} = \frac{1}{N}\sum_{r=1}^{R} m_i^{(r)} n_j^{(r)} \tag{A.12}$$

i.e. $\mathbf{J}$ was a sum of $R$ outer-products of vectors $\boldsymbol{m}^{(r)} = \{m_i^{(r)}\}_{i=1\ldots N}$ and $\boldsymbol{n}^{(r)} = \{n_i^{(r)}\}_{i=1\ldots N}$. Throughout the text, we refer to the vectors $\boldsymbol{m}^{(r)}$ and $\boldsymbol{n}^{(r)}$ as the *connectivity vectors*, with $\boldsymbol{m}^{(r)}$ the $r$-th output vector, and $\boldsymbol{n}^{(r)}$ the $r$-th input-selection vector. Without loss of

generality, we will assume that all the output vectors (and respectively all the input-selection vectors) are mutually orthogonal. Such a representation is uniquely defined by the singular-value decomposition of $\mathbf{J}$ by taking $\boldsymbol{m}^{(r)}$ to be the left singular vectors, and $\boldsymbol{n}^{(r)}$ the right singular vectors multiplied by the corresponding singular values.
The feedforward inputs $I_i^{FF}(t)$ were generated by $N_{in}$ temporally-varying scalar stimuli $u_s(t)$, each fed into the unit $i$ through a set of weights $I_i^{(s)}$:

$$I_i^{FF}(t) = \sum_{s=1}^{N_{in}} I_i^{(s)} u_s(t). \tag{A.13}$$

We refer to $\boldsymbol{I}^{(s)} = \{I_i^{(s)}\}_{i=1...N}$ as the $s$-th *input vector*.
The output of the network was defined by a readout value

$$z = \frac{1}{N} \sum_{j=1}^{N} w_j \phi(x_j), \tag{A.14}$$

where $\boldsymbol{w} = \{w_i\}_{i=1...N}$ is the *readout vector*.
The time constant of neurons was $\tau = 100$ms. For simulation and training, equation (A.11) was discretized using Euler's method with a time step $\Delta t = 20$ms. The white noise $\eta_i$ was simulated by drawing at each time step a random number from a centered Gaussian distribution of standard deviation 0.05.

### A.3.2 Network training procedure

We used backpropagation through time (223) to train networks to minimize loss functions corresponding to specific tasks. For each task (see details below), we specified the temporal structure of trials and the desired mapping from combinations of stimulus inputs to target readouts $\hat{z}$, and then stochastically generated trials. We minimized the mean squared error loss function

$$\mathcal{L} = \sum_{k,t} M_t (z_{k,t} - \hat{z}_{k,t})^2 \tag{A.15}$$

where $z_{k,t}$ and $\hat{z}_{k,t}$ are respectively the actual, and the target readout values and the indices $k, t$ respectively run over trials and time steps. The terms $M_t$ are $\{0, 1\}$ masks that were non-zero only during a decision period at the end of each trial, when the readouts were required to match their target values. For each task we also define a performance measure called accuracy, defined as the fraction of test trials for which the network output has the same sign as the expected output (i.e. $\text{sign}(\sum_t M_t \hat{z}_{k,t}) = \text{sign}(\sum_t M_t z_{k,t})$)
For full-rank networks (Figs. 1,6) the gradients were computed with respect to individual entries $J_{ij}$ of the connectivity matrix. For results on full-rank networks in Fig. 1 (left column) and A.4, matrices $\mathbf{J}$ were initialized with random independent Gaussian weights of mean 0 and variance $\rho = 1/N$. For the A.4, we also trained networks whose weights were initialized with a variance $\rho = 0.1/N$, since these tend to be approximated more easily by low-rank networks (181).
For low-rank networks, we specifically looked for solutions in the subspace of connectivity matrices with rank $R$. The loss functions were therefore minimized by computing gradients with respect to the elements of connectivity vectors $\{\boldsymbol{m}^{(r)}\}_{r=1...R}, \{\boldsymbol{n}^{(r)}\}_{r=1...R}$. Unless specified otherwise in the description of individual tasks, we did not train the entries of input vectors $\{\boldsymbol{I}^{(s)}\}_{s=1...N_{in}}$ and the readout vectors $\{\boldsymbol{w}\}$ but only an overall amplitude factor for each input and readout vector. All vectors were initialized with their entries drawn from Gaussian distributions with zero mean and unit standard deviation, except for the readout vector, for which the standard deviation was 4. The initial network state at the beginning of

each trial was always set to $\mathbf{0}$. We used the ADAM optimizer (97) in pytorch (140) with the decay rates of the first and second moments of 0.9 and 0.999, and learning rates between $10^{-3}$ and $10^{-2}$.
To identify networks of minimal rank that performed each task, the number of pairs of connectivity vectors $R$ was treated as a hyper-parameter. We first trained full rank networks ($R = N$) and determined the accuracy with which they solved the task. We then started training rank $R = 5$ networks, and progressively decreased the rank until there was a sharp decrease in accuracy (A.3). The minimal rank $R^*$ was defined for each task such that the accuracy at $R^*$ was at least of 95%.
To ease the clustering and resampling procedure, and approach mean-field solutions, we trained large networks (of sizes 512 neurons for the networks of figures III.1 and III.2, 4096 neurons for the context-dependent DM and DMS task networks of Figures III.4 and **??**, and 1024 neurons in figure III.7).

### A.3.3 Definition of individual tasks

#### A.3.3.1 Perceptual decision making (DM) task

**Trial structure.** A fixation epoch of duration $T_{fix} = 100$ms was followed by a stimulation epoch of duration $T_{stim} = 800$ms, a delay epoch of duration $T_{delay} = 100$ms and a decision epoch of duration $T_{decision} = 20$ms.
**Inputs and outputs.** The feed-forward input to neuron $i$ on trial $k$ was

$$I_i^{FF}(t) = I_i u^{(k)}(t) \tag{A.16}$$

where, during the stimulation period, $u^{(k)}(t) = \overline{u}^{(k)} + \xi^{(k)}(t)$, with $\xi^{(k)}(t)$ a zero-mean Gaussian white noise with standard deviation $\sigma_u = 0.1$. The mean stimulus $\overline{u}^{(k)}$ was drawn uniformly from $\pm 0.1 \times \{1, 2, 4\}$ on each trial. The elements $I_i$ of the input vector were generated from a Gaussian distribution with zero mean and unit standard deviation, and fixed during training.
During the decision epoch, the output $z$ was evaluated through a readout vector $\boldsymbol{w} = \{w_i\}_{i=1...N}$, the elements $w_i$ of which were generated from a Gaussian distribution with zero mean and standard deviation of 4, and fixed during the training. On trial $k$, the target output value $\hat{z}_k$ in the loss function (Eq. (A.15)) was defined as the sign of the mean input $\overline{u}^{(k)}$.

#### A.3.3.2 Parametric working memory (WM) task

**Trial structure.** A fixation epoch of duration $T_{fix} = 100$ms was followed by a first stimulation epoch of duration $T_{stim1} = 100$ms, a delay epoch of duration $T_{delay}$ drawn from a uniform distribution between 500 and 2000ms, a second stimulation epoch of duration $T_{stim2} = 100$ms and a decision epoch of duration $T_{decision} = 100$ms.
**Inputs and outputs.** The feed-forward input to neuron $i$ on trial $k$ was

$$I_i^{FF}(t) = I_i \left( u_1^{(k)}(t) + u_2^{(k)}(t) \right) \tag{A.17}$$

where $u_1^{(k)}(t)$ and $u_2^{(k)}(t)$ were non-zero during the first and second stimulation epochs respectively. On trial $k$ and during the corresponding stimulation epoch, the values of these inputs were $u_{1,2}^{(k)} = \frac{1}{f_{max} - f_{min}} (f_{1,2}^{(k)} - \frac{f_{max} + f_{min}}{2})$, with $f_1^{(k)}$ and $f_2^{(k)}$ drawn uniformly from $\{10, 11, \ldots, 34\}$, and $f_{min} = 10$ and $f_{max} = 34$. The elements $I_i$ of the input vector were generated from a Gaussian distribution with zero mean and unit standard deviation, and fixed during the training.
During the decision epoch, the output $z$ was evaluated through a readout vector $\boldsymbol{w} = \{w_i\}_{i=1...N}$, the elements $w_i$ of which were generated from a Gaussian distribution with zero

mean and standard deviation of 4, and fixed during the training. On trial $k$, the target output value $\hat{z}_{(k)}$ in the loss function (Eq. (A.15)) was defined as $\hat{z}_{(k)} = \frac{f_1^{(k)} - f_2^{(k)}}{f_{max} - f_{min}}$.

#### A.3.3.3 Context-dependent decision making (CDM) task

**Trial structure.** A fixation epoch of duration $T_{fix} = 100$ms was followed by a first context-only epoch of duration $T_{ctxt1} = 0$ms for figure 1 and $350ms$ for Figs. III.3-III.6 plots, followed by a stimulation epoch of duration $T_{stim} = 800$ms, a second context-only epoch of $T_{ctxt2} = 500$ms, and a decision epoch of $T_{decision} = 20$ms.
**Stimuli and outputs.** The feed-forward input to neuron $i$ on trial $k$ was

$$I_i^{FF}(t) = u_A^{(k)}(t) I_i^A + u_B^{(k)}(t) I_i^B + u_{ctxA}^{(k)}(t) I_i^{ctxA} + u_{ctxB}^{(k)}(t) I_i^{ctxB}. \tag{A.18}$$

Here $u_{ctxA}^{(k)}$ and $u_{ctxB}^{(k)}$ correspond to contextual cues. On each trial, during the context-only and the stimulation epochs, one of the two cues took a value $+0.1$ (or $+0.5$ for Figs. III.3-III.6), while the other was 0. The inputs $u_A^{(k)}(t)$ and $u_B^{(k)}(t)$ represent two sensory features of the stimulus. They were non-zero only during the stimulation epoch, and took the same form as in the perceptual decision-making task, with means $\overline{u}_A^{(k)}$ and $\overline{u}_B^{(k)}$, and fluctuating parts $\xi_A^{(k)}(t)$ and $\xi_B^{(k)}(t)$ drawn independently for each feature, on each trial. The elements of the input vectors were generated from a Gaussian distribution with zero mean and unit standard deviation on both populations. For the networks presented in the main text, input vectors were trained, while for the networks reported in Supplementary Note 2.3 all the input vectors were fixed throughout training.
During the decision epoch, on trial $k$ the target $\hat{z}_{(k)}$ in the loss function (Eq. (A.15)) was defined as the sign of the mean $\overline{u}_X^{(k)}$ of feature $X = A$ or $B$ for which the contextual cue was activated, i. e. $u_{ctx}^{(k)} = 1$. The readout vector was fixed throughout training.

#### A.3.3.4 Multi-sensory decision making (MDM) task

**Trial structure.** A fixation epoch of duration $T_{fix} = 100$ms was followed by a context-only period of duration $T_{ctx} = 350$ms, a stimulation epoch of duration $T_{stim} = 800$ms, a delay epoch of duration $T_{delay} = 300$ms and a decision epoch of duration $T_{decision} = 20$ms.

**Inputs and outputs.** The feed-forward input to neuron $i$ on trial $k$ had the same structure as for the context-dependent decision-making task, and was given by:

$$I_i^{FF}(t) = u_A^{(k)}(t) I_i^A + u_B^{(k)}(t) I_i^B + u_{ctxA}^{(k)}(t) I_i^{ctxA} + u_{ctxB}^{(k)}(t) I_i^{ctxB}. \tag{A.19}$$

where the two stimulus inputs $u_A^{(k)}(t)$ and $u_B^{(k)}(t)$ represent two sensory modalities, and $u_{ctxA}^{(k)}$ and $u_{ctxB}^{(k)}$ are contextual inputs. In this task, the contextual inputs were irrelevant for the output, and we included them as a control. The inputs $u_A^{(k)}(t)$ and $u_B^{(k)}(t)$ were generated as for the CDM task, with the difference that on each trial the two inputs provided congruent evidence for the output, i.e. their means were of the same sign.

Specifically in each trial a sign $s_k \in \{-1, 1\}$ is generated randomly, as well as a modality that can be A, B, or AB. Then if the modality is A or AB, a mean $\overline{u}_A^{(k)}$ is chosen from $0.1 \times s_k \times \{1, 2, 4\}$ and the signal $u_A^{(k)}(t)$ during the stimulation period is set to that mean plus a gaussian white noise as in the perceptual decision making task. A contextual input signal $u_{ctxA}^{(k)}(t)$ is set to 0.1 from the beginning of the contextual period to the end of the trial. If the modality is B, then the signal $u_A^{(k)}(t)$ is only equal to the zero-centered gaussian

white noise. The signals $u_B^{(k)}(t)$ and $u_{ctxB}^{(k)}(t)$ are set in a similar manner. During the decision epoch, the target $\hat{z}^{(k)}$ is the underlying common sign $s_k$.

The networks received input signals through input vectors $\boldsymbol{I}^A$, $\boldsymbol{I}^B$, $\boldsymbol{I}^{ctxA}$ and $\boldsymbol{I}^{ctxB}$ which were trained, and output was read through a readout vector $\boldsymbol{w}$ which was fixed throughout training.

#### A.3.3.5 Delayed-match-to-sample task

**Trial structure.** A fixation epoch of duration $T_{fix} = 100$ms was followed by a first stimulus epoch of duration $T_{stim1} = 500$ms, a delay epoch of a duration drawn uniformly between 500ms and 3000ms, a second stimulus epoch of duration $T_{stim2} = 500$ms, and a decision epoch of duration $T_{decision} = 1000$ms.

**Stimuli and outputs.** During each stimulus epoch, the network received one of two stimuli $A$ or $B$, which were randomly and independently chosen on each trial and stimulus epoch. These two stimuli were represented by two input vectors $\boldsymbol{I}^A$ and $\boldsymbol{I}^B$, so that the feed-forward input to neuron $i$ on trial $k$ was:

$$I_i^{FF}(t) = u_A^{(k)}(t) I_i^A + u_B^{(k)}(t) I_i^B \tag{A.20}$$

where the inputs $u_A^{(k)}(t)$ and $u_B^{(k)}(t)$ were non-zero only when stimuli $A$ or $B$ are respectively received, in which case they were equal to one.
During the decision epoch, the target output value $\hat{z}$ in the loss function (Eq. (A.15)) was equal to +1 if the same stimulus was received in both stimulation epochs and -1 otherwise.

### A.3.4 Regression analyses and selectivity space

We used multivariate linear regression to predict time-averaged neural firing rates $r_i = \phi(x_i)$ from task variables, using a linear model :

$$\boldsymbol{r}_i = \boldsymbol{X}\boldsymbol{\beta}_i + \boldsymbol{\epsilon}_i. \tag{A.21}$$

Here $\boldsymbol{r}_i = \{r_{i,1}, \ldots, r_{i,K}\}$ is a vector containing the time-averaged firing rates of neuron $i$ in trials 1 to $K$, $\boldsymbol{X}$ is the design matrix where rows correspond to different trials and columns correspond to $D$ task variables such as stimulus, context and decision in each condition (defined below for each task), $\boldsymbol{\beta}_i$ is a $D$-by-1 vector of regression coefficients, and $\boldsymbol{\epsilon}_i$ is a $K$-by-1 vector of residuals.
The regression coefficients defined the *selectivity space* (Fig. 1a-d) of dimension $D$ where each axis corresponded to the regression coefficient with respect to one task variable, and each neuron was represented as point $\boldsymbol{\beta}_i$.
The choice of task variables and window of time-averaging of firing rates depended on the task:

- For the DM task, two regressions were performed on different time windows, leading to $D = 2$ two coefficients per neuron: a regression of average firing rate during the first 100ms of stimulation period against mean stimulus which defined the coefficient $\beta_i^{stim}$ and a regression of average firing rate during the decision period against network choice which defined the coefficient $\beta_i^{choice}$. This was done to avoid ill-conditioning due to correlations between choice and stimulus.

- For the WM task, the mean firing rate during the decision period was regressed against both $f_1$ and $f_2$, leading to $D = 2$ two coefficients per neuron.

- For the MDM task and the CDM task, the average firing rate during the stimulation period was regressed against both mean stimulus features $\overline{u}_A^{(k)}$ and $\overline{u}_B^{(k)}$ and both contextual input signals $u_{ctxA}^{(k)}$ and $u_{ctxB}^{(k)}$, leading to $D = 4$ coefficients per neuron, $\beta_i^A$, $\beta_i^B$, $\beta_i^{ctxA}$ and $\beta_i^{ctxB}$. In Fig. III.6, the selectivity to context was characterized by a single regression coefficient $\beta_i^{ctx}$ obtained by regressing the absolute value of the firing rate $|r_i|$, averaged over the pre-stimulus period where only the contextual cues are non-zeros, against a regressor $X$ that takes the value $+1$ in context A and $-1$ in context B. The context selectivity is extracted through the linear model for $K$ trials

$$|\boldsymbol{r}_i| = \boldsymbol{X}\beta_i^{ctx} + \boldsymbol{\epsilon} \tag{A.22}$$

  In order to characterize the changes in selectivity with context, we substracted the pre-stimulus firing rate to the firing rate averaged over the first 100ms of stimulus presentation, and regressed this quantity against $\overline{u}_A^{(k)}$ and $\overline{u}_B^{(k)}$ separately in each context to obtain the regression coefficients $\beta_{ctxA,i}^A, \beta_{ctxA,i}^B, \beta_{ctxB,i}^A, \beta_{ctxB,i}^B$. The change in selectivity is then given by

$$\Delta_{ctx}\beta_i^{A/B} = |\beta_{ctxA,i}^{A/B}| - |\beta_{ctxB,i}^{A/B}| \tag{A.23}$$

  In Fig. III.6 the analysis is presented for feature A, similar results are obtained for feature B (not shown).

- For the DMS task, the average firing rate during the decision period was regressed against both first and second stimulus identity (with $X_{k,s} = 1$ if stimulus $s$ is A in trial $k$, 0 otherwise, $s \in \{0, 1\}$), leading to $D = 2$ regression coefficients per neuron.

### A.3.5 ePAIRS analysis

To statistically assess the presence of non-random population structure in the selectivity and connectivity spaces of trained networks, we implemented a version of the ePAIRS statistical test (81), which is itself derived from the PAIRS test developed in (152). We consider a point cloud $\mathbf{X} = (X_{ij})_{1\leq i\leq N, 1\leq j\leq d}$, where the rows $\boldsymbol{x}_i$ corresponds to different points (here neurons) and columns correspond to different axes of the considered space (regression coefficients to different variables in the selectivity space, entries of different input, connectivity and readout vectors in the connectivity space), which is centered by removing the mean ( so that for each $j$, $\sum_i X_{ij} = 0$). The ePAIRS test examines the directional distribution of points, i.e. the empirical distribution of $\boldsymbol{x}_i/\|\boldsymbol{x}_i\|$, and determines whether it is statistically distinguishable from the null distribution generated by a multivariate Gaussian with a covariance matrix identical to the covariance of $\mathbf{X}$. A significant outcome indicates of the ePAIRS test that the empirical distribution presents multiple "preferred" directions incompatible with a Gaussian.
More specifically, the analysis proceeds as follows:

1. For each point $\boldsymbol{x}_i$, we determine its $l$ nearest neighbors in terms of the cosine metric (*ie.* the $l$ points for which $\cos(\widehat{\boldsymbol{x}_i\boldsymbol{x}_j}) = \boldsymbol{x}_i^T\boldsymbol{x}_j/(\|\boldsymbol{x}_i\|\|\boldsymbol{x}_j\|)$ are the highest, $l$ being a hyperparameter set to 3 in our case).

2. For each neuron, we compute the mean angle $\alpha_i$ with its $l$ nearest neighbors, defining an empirical distribution $\hat{p}_{data}(\alpha)$.

3. To generate the corresponding null distribution, a multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ is fit to the cloud of points $\mathbf{X}$, with $\mathbf{\Sigma}$ the empirical covariance of $\mathbf{X}$, computed as $\mathbf{\Sigma} = \frac{1}{N}\mathbf{X}^T\mathbf{X}$. Then the steps 1-2 are applied on 500 samples of the multivariate Gaussian with the same number $N$ of data points to compute a Monte-Carlo null distribution $\hat{p}_{null}(\alpha)$.

4. Finally, the difference between the data and the null distributions is assessed using a two-sided Wilcoxon's rank-sum test, giving a p-value, and the effect size $c$ is computed as

$$c = \frac{\mu_{null} - \mu_{data}}{\sigma_{null}}, \tag{A.24}$$

where $\mu$ and $\sigma$ represent the means and standard deviations of $\hat{p}_{null}(\alpha)$ and $\hat{p}_{data}(\alpha)$. An effect size $c > 0$ indicates that angles between neighbors are smaller in the data than in the resampled point clouds, meaning that points are more highly clustered than expected. On the contrary, $c < 0$ indicates that points are more regularly dispersed than expected from random.

### A.3.6 Resampling and clustering trained networks

For a given trained network, we first fitted a single multivariate Gaussian to its connectivity distribution by computing the empirical covariance matrix (matrix of size $(N_{in} + 2R + 1)^2$). We then generated networks by resampling connectivity parameters from this distribution, and examined their performance (Fig. 1i and A.4). In all trained networks we examined, the empirical means were close to zero, and we neglected them.

For the CDM and DMS tasks, we performed a clustering analysis in the connectivity space by fitting multivariate mixtures of Gaussians with an increasing number of clusters, and by resampling from the obtained distributions until we found networks that were able to optimally perform the task, as defined by an accuracy higher than 95% for at least 95% of the sampled networks. We used variational inference with a gaussian prior for the mean with a precision equal to $10^5$ to enforce a zero-mean constraint for all components of the mixtures, and a Dirichlet process prior for the weights with concentration 1 divided by number of components, using the model `BayesianGaussianMixture` of the package scikit-learn (141).

Since the inference and resampling processes are susceptible to finite-size fluctuations, for the DMS task in **??** we complemented the clustering with some retraining of the covariance matrices found for each component. For this we developed a class of Gaussian mixture, low-rank RNNs, in which the covariance structure of each population is trainable. Directly training the covariance matrices is difficult given that they need to be symmetric definite positive; we therefore used a trick akin to the reparametrization trick used in variational auto-encoders (98): the set of input, connectivity and readout vectors were defined as a linear transformation of a basis of i.i.d. normal vectors, such that for any connectivity vector $\boldsymbol{a}$:

$$a_i = (\boldsymbol{b}_a^{(p)})^T \boldsymbol{X}_i, \tag{A.25}$$

where $p$ is the population index of neuron $i$ (sampled from a categorical distribution with weights $\{\alpha_p\}_{p=1...P}$ derived by the variational inference), $\boldsymbol{X}_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\boldsymbol{0}, \mathbb{1})$ are random normal vectors of dimension $N_{in}+2R+1$, and the vectors $\boldsymbol{b}_a^{(p)}$ correspond to the rows of the Cholesky factorization of the covariance matrix (such that $\sigma_{ab}^{(p)} = (\boldsymbol{b}_a^{(p)})^T \boldsymbol{b}_b^{(p)}$ see Supplementary Note 1 for more details). We then trained the vectors $\boldsymbol{b}_v^{(p)}$, with the population indices being sampled only once, and the $\boldsymbol{X}_i$ being resampled at each training epoch.

## A.4 Training low-rank RNNs: some tricks of the trade

In this section, we aim to informally share some tips and tricks that in our experience increase the chances of successfully training a low-rank network on a task, albeit we do not necessarily have a justification for them.

- Propagation of gradients in vanilla RNNs is notoriously difficult, and strategies involving careful tuning of initial weights as well as gradient clipping (38) can considerably ease training. In particular, we often initialized the weights of $\mathbf{M}$ and $\mathbf{N}$ by i.i.d. samples of the standard normal distribution, which with the $1/N$ factor in the definition of recurrent connectivity allows for approximately stable dynamics to emerge and remain over many timesteps (122). Another possibility is to parametrize $\mathbf{J}$, not following Eq. (II.44) but as $\mathbf{J} = \mathbf{M}\mathbf{N}^\top$ instead (that is, without the $1/N$ factor), and initialize the entries of $\mathbf{M}$ and $\mathbf{N}$ from a normal distribution with standard deviation $1/\sqrt{N}$. Depending on situations, one or the other definitions of connectivity might be easier to use, although we do not have a clear explanation why this is the case (note in the second definition the imbalance between the magnitudes of recurrent weights, which are on the order of $1/\sqrt{N}$, and input weights on the order of 1, if those are trained).

- A popular strategy to train animals as well as artificial networks is by *shaping*, also known as curriculum learning. In our cases, it involves for example progressively increasing the duration of tasks, and in particular the range of delay periods for all tasks, or adding smaller coherences during the training process. An interesting subject for discussion, which we did not explore is whether the shaping strategy modifies the computational implemetation used for task, as suggested by several studies (92; 127).

- We generally found that it was easier to train full-rank networks on a given task than low-rank networks (but for low-rank networks, increasing the rank does not in general make the training faster or easier). An interesting strategy to obtain low-rank network solutions of tasks is thus to first train a full-rank RNN on it, then initialize a low-rank network from the truncated SVD of the trained full-rank RNN, and train it on the task from this state. As illustrated in chapter V and (181), a full-rank RNN with truncated SVD in general looses its ability to perform a task, even though it has only learned low-rank structure (181). Nevertheless, the truncated SVD seems to contain a certain amount of signal useful to retrieve the desired low-rank structure faster than from an initial random connectivity. Note also that such a truncated SVD might embed non-zero overlaps between the $\boldsymbol{m}^{(r)}$ and $\boldsymbol{n}^{(r)}$ vectors, while for random initial connectivity these overlaps will be very small. It is possible that these are taken advantage of by the low-rank RNN to propagate gradients and generate useful dynamical features. As such, another strategy that proved to be efficient was simply to initialize $\mathbf{M}$ and $\mathbf{N}$ so that the columns of $\mathbf{N}$ have a certain non-zero overlap with those of $\mathbf{M}$ (that is $m_i^{(r)}$ and $n_i^{(r)}$ are correlated and not independent).

To conclude, these are strategies that appear to have worked for the authors in some situations, sometimes when used jointly, sometimes independently. A precise characterization of when and why they work remains lacking, as the training procedure was in this work a means and not a goal of our research. Nevertheless, due to the considerable memory and computation efficiency of low-rank networks compared to full-rank ones (at no point in the training or usage of such networks is it required to compute a $N^2$-parameter matrix), it is possible that understanding how to train them better proves useful for practical applications.

## A.5 Supplementary figures

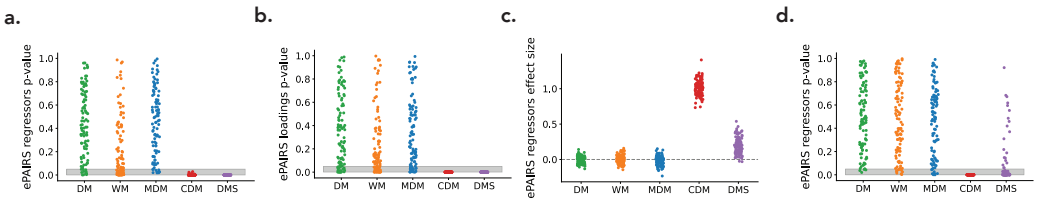


FIGURE A.1: Additional ePAIRS results. (a) p-values given by the ePAIRS test on selectivity spaces for the full-rank networks displayed in Fig. 1d (two-sided ePAIRS test, 100 networks per task, $n = 512$ neurons for each network). (b) p-values given by the ePAIRS test on connectivity spaces for the low-rank networks displayed in Fig 1h (two-sided ePAIRS test, 100 networks per task, $n = 512$ neurons for each network). (c) ePAIRS effect sizes on the selectivity space for the same low-rank networks (two-sided ePAIRS test, 100 networks per task, $n = 512$ neurons for each network). (d) Corresponding ePAIRS p-values.

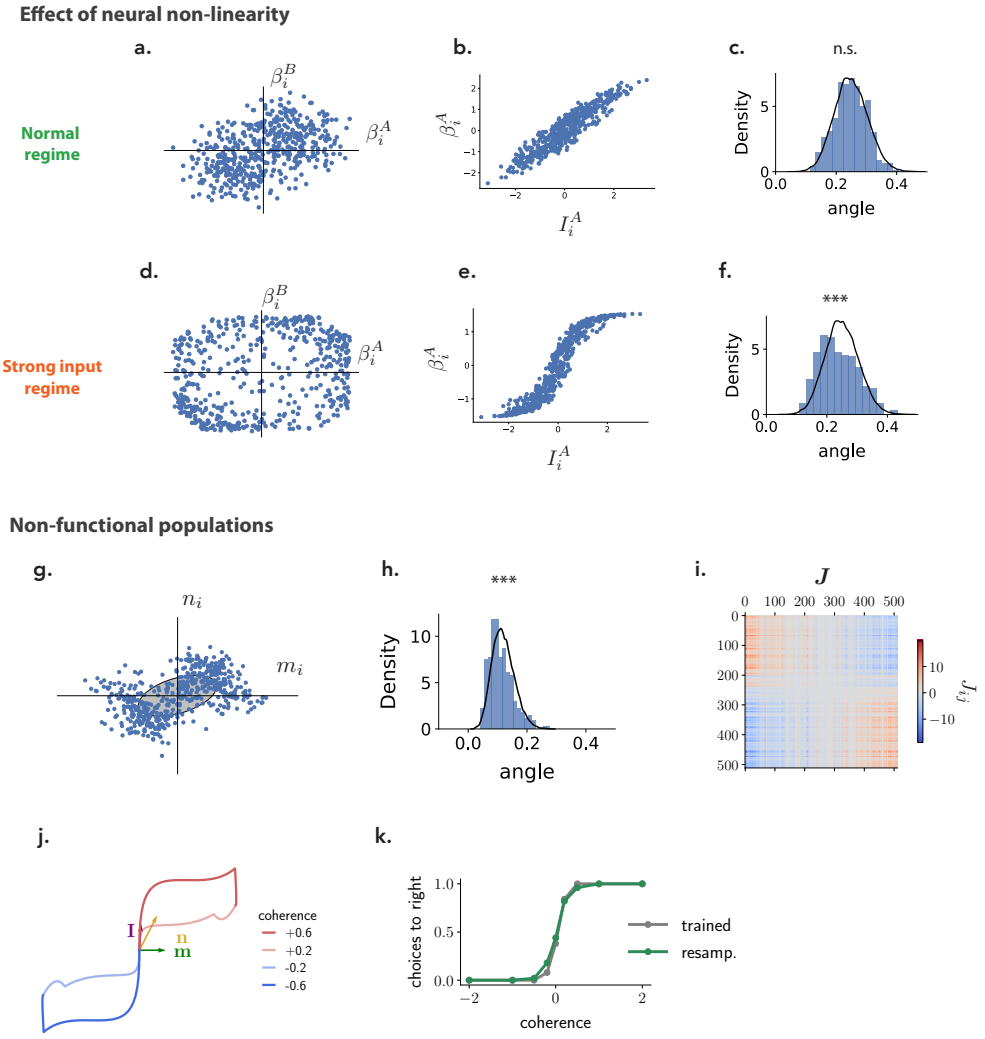


Figure A.2: This figure illustrates two situations in which the ePAIRS test leads to false positives, and identified non-random population structure that is not computationally relevant. (a)-(f) The activity in a single network trained on the MDM task and used in Fig. 1 was compared in two conditions: (a-c) in response to inputs of the scale used for training; (d-f) in response to inputs scaled by a factor 10. The corresponding regressor spaces were then tested for non-random population structure via the ePAIRS test. No evidence for population structure in selectivity or connectivity space is found for inputs in the range used for training (a-c). Using stronger inputs however leads to positive ePairs in the selectivity space, although the underlying connectivity is identical (d-f). (a) Slice of the selectivity space for this network representing regression coefficients for each neuron with respect to inputs A and B. (b) In this input regime, regression coefficients with respect to inputs are linear functions of the components along the corresponding input vectors (each point represents a neuron in the network). (c) An ePAIRS test on the selectivity space in that case leads to a non-significant outcome ($p = 0.48$, $c = 0.03$, $n = 512$ neurons). (d) As in (a), for the same network, but driven with inputs 10 times larger than those used for training. The individual units are in this case driven to saturation so that the points in the selectivity space are concentrated along the borders of a square.

Figure A.2 *(previous page)*: (e) Same as (b) in the strong input regime. The relation between the original input vector and the obtained regression coefficients reflects the underlying non-linearity as neurons are driven to saturation. (f) ePairs on the square-like distribution in selectivity space shown in (d) rejects the null hypothesis for random population structure ($p = 2 \times 10^{-6}$, $c = 0.30$, $n = 512$). (g)-(k) An example network trained on the Perceptual Decision Making task exhibiting spurious, computationally irrelevant population structure detected by the ePAIRS test. This network was obtained by using a different scaling of recurrent weights during training than in the rest of the work. For all networks in the main text, the recurrent connectivity was defined as $\mathbf{J} = \frac{1}{N}\sum_r \boldsymbol{m}^{(r)}\boldsymbol{n}^{(r)^T}$ with entries of vectors $\boldsymbol{m}^{(r)}$ and $\boldsymbol{n}^{(r)}$ being of order 1 and the $\mathcal{O}(1/N)$ scaling of the connectivity matrix being explicitly added in the network dynamics. For this example the recurrent connectivity was instead defined as $\mathbf{J} = \sum_r \boldsymbol{m}^{(r)}\boldsymbol{n}^{(r)^T}$ with entries of the connectivity vectors being of order $\mathcal{O}(1/\sqrt{N})$, making the scaling of the connectivity matrix being implicit, which led to different gradient descent dynamics and to significantly different solutions. Here a rank-one network of 512 neurons is shown. (g) Scatter plot of the entries of each neuron on the recurrent connectivity vectors $\boldsymbol{m}$ and $\boldsymbol{n}$, showing two clusters symmetrical with respect to the mean. Note that this cluster structure is very different from those seen in the rest of the paper, which corresponded to zero-mean clusters with different covariance matrices, while here two non-zero-mean clusters are visible. (h) The ePAIRS test detected evidence for non-random population structure on the connectivity space (which is here 4-dimensional, composed of vectors $\boldsymbol{I}$, $\boldsymbol{n}$, $\boldsymbol{m}$ and $\boldsymbol{w}$. Here, $p = 5 \times 10^{-9}$, $c = 0.35$, $n = 512$). (i) The two clusters seen in the scatter plot can also be made apparent in the connectivity matrix $\mathbf{J}$ if its entries are properly ordered, here by ascending values of $m_i + n_i$. (j) State-space response trajectories to different stimuli projected on the $\boldsymbol{m}$-$\boldsymbol{I}$ plane are similar to those found for the network shown in Fig. 2. (k) As for the network in Fig. 2, networks resampled from a Gaussian distribution fit to the connectivity space of the trained network (black ellipse in panel a) performed equally well as the trained network, showing that the population structure found by the ePAIRS procedure was not computationally relevant, and might be an artifact of learning.

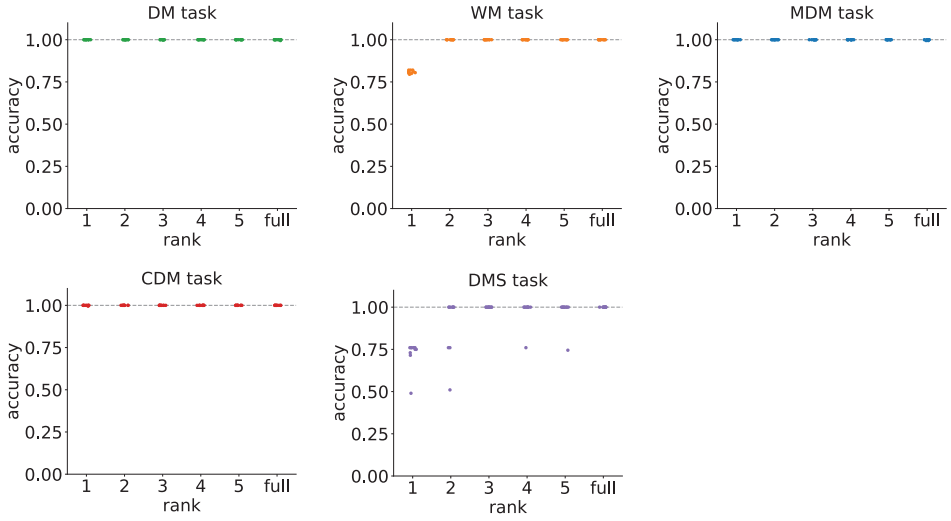


Figure A.3: **Determination of the minimal rank for each task.** For each task and each rank $R$ between 1 and 5, ten rank-$R$ networks were trained with different random initial connectivity. For each task, a panel displays the performance of trained networks as function of their rank.

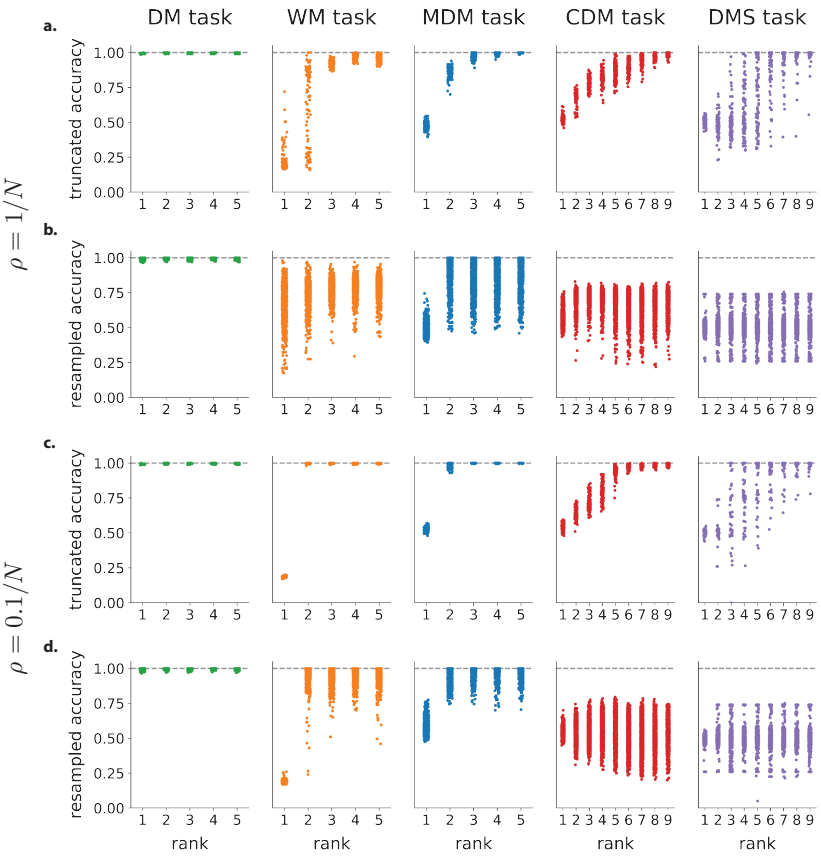


FIGURE A.4: **Analysis of trained full-rank networks.** (a)-(b) Analysis of full-rank networks trained with initial connectivity weights of variance $1/N$ (100 networks for each task). (a) Performance of truncated-rank networks. Following (181), we extract from full-rank networks the learned part of the connectivity $\Delta\mathbf{J} = \mathbf{J} - \mathbf{J}_0$ defined as the difference between the final connectivity $\mathbf{J}$ and the initial connectivity $\mathbf{J}_0$. We then truncate $\Delta\mathbf{J}$ to a given rank via singular value decomposition, and add it back to $\mathbf{J}_0$. For each task, a panel displays the performance of the obtained networks as function of the rank used for the truncation. (b) Resampling analysis of truncated networks. Starting from the truncated networks in (a) we fit multivariate Gaussians to the distribution of their $\Delta\mathbf{J}$ in the corresponding connectivity spaces. We then generate new networks by resampling from this distribution, as done on the trained low-rank networks for Fig. 1i-l. For each task, a panel displays the performance of the obtained resampled network as function of the rank used for the truncation. (c)-(d) Same analyses as (a)-(b) for sets of networks trained with initial connectivity weights of variance $0.1/N$ (100 networks for each task, for DMS 49/100 networks that had an accuracy $< 95\%$ after training and were ignored). Networks with weaker initial connectivity are better approximated by their resampled low-rank connectivity. This is due to the fact that larger initial connectivities induce correlations between $\Delta\mathbf{J}$ and $\mathbf{J}_0$ (181). The resampling destroys both this correlation and the population structure, leading to performance impairments even when the population structure is potentially irrelevant.

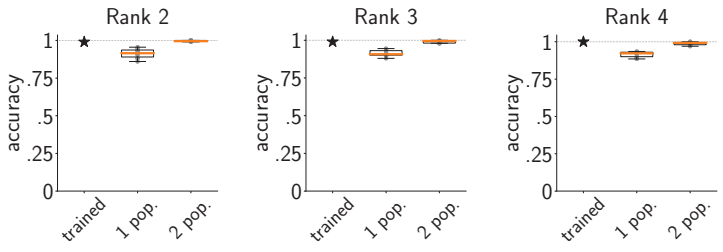


Figure A.5: **Increasing the rank maintains the requirement for population structure.** For this figure we have trained low-rank networks with a rank higher than 1 on the CDM task, fitted a single Gaussian or a mixture of 2 Gaussians to the obtained connectivity space, and retrained the obtained distribution (Methods A.3.6) to obtain resampled networks with a performance as high as possible. Even with this additional layer of retraining of the fitted distributions (which is only present in the main text for the DMS task) the obtained single-population networks fell short of performing the CDM task with a good accuracy. Here, 10 draws of a single network for each combination of rank and number of populations are shown (line: median, box: quartiles, whiskers: range, in the limit of median $\pm$ 1.5 interquartile range, points: outliers).

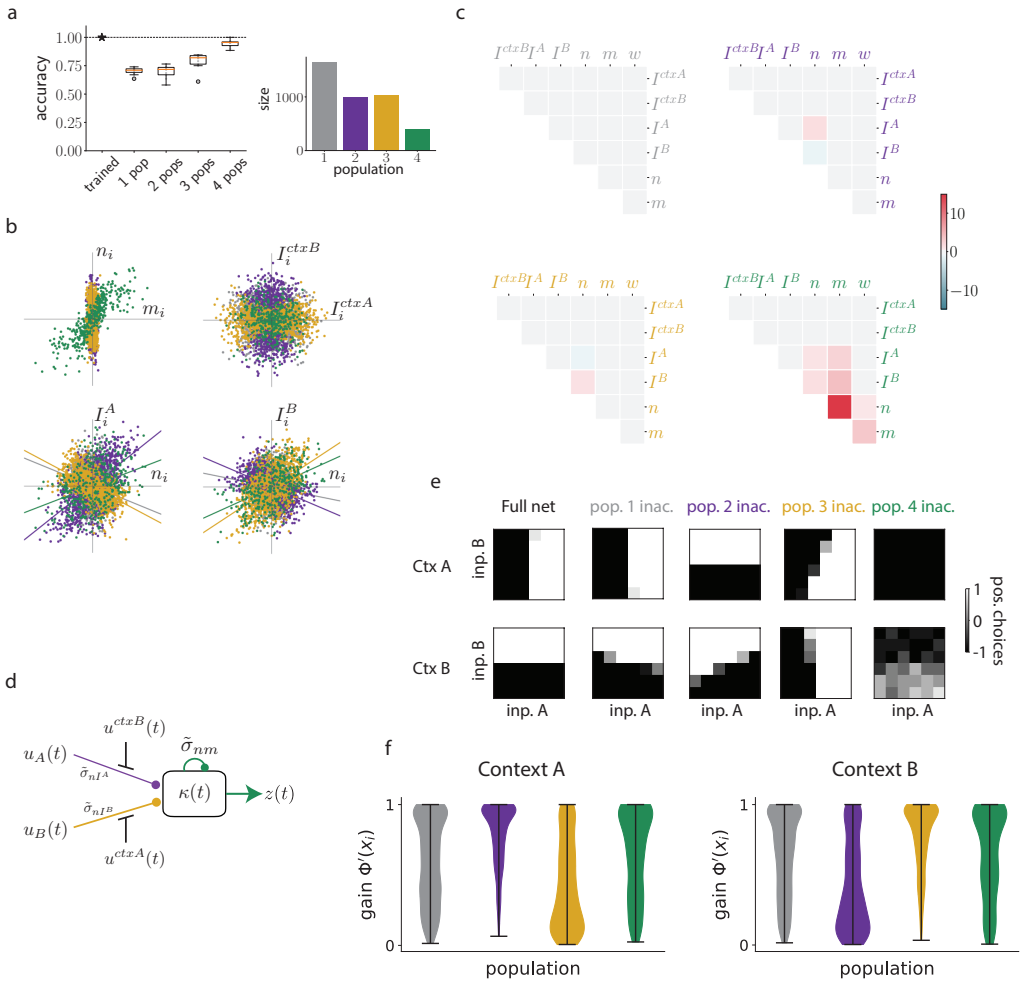


FIGURE A.6: **Alternative implementation of the CDM task**. A network trained with different hyperparameters offers an example of an alternative solution for the CDM task, using 3 effective population and a fourth one accounting for neurons that are not involved in the task (see Supplementary Text 3)). (a) Left: for each number of sub-populations, a boxplot shows the performance of 10 networks with connectivity resampled from a Gaussian Mixture Model (GMM) fitted to the trained network (line: median, box: quartiles, whiskers: range, in the limit of median $\pm$ 1.5 interquartile range, points: outliers). Right: for the GMM with four sub-populations, size of each component found by the clustering procedure. (b) Four 2d projections of the 7-dimensional connectivity space. (c) Upper-right triangle of the empirical covariance matrices for each of the four populations. (d) Illustration of the mechanism used by the network at the level of latent dynamics. Populations $2-4$ control one effective coupling each, indicated by the matching color. (e) Psychometric matrices similar to those shown in Fig. III.3 after inactivation of each sub-population. (f) Violin plots showing the gain distributions of neurons in each of the four sub-populations in each context.

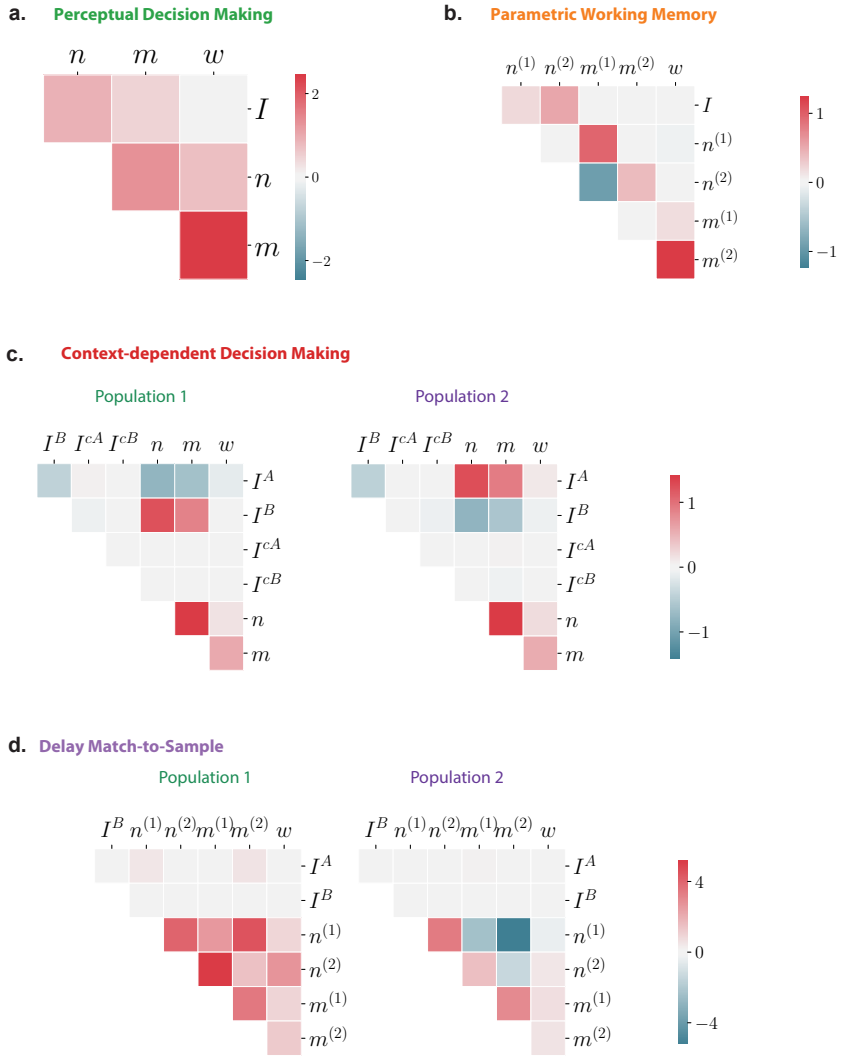


Figure A.7: **Statistics of connectivity in trained networks.** Upper left corner of the empirical covariance matrix between connectivity vectors for networks trained on each task, after clustering neurons in two populations for tasks CDM and DMS. These covariance matrices are then used for resampling single-population and two-population networks that successfully perform each task.

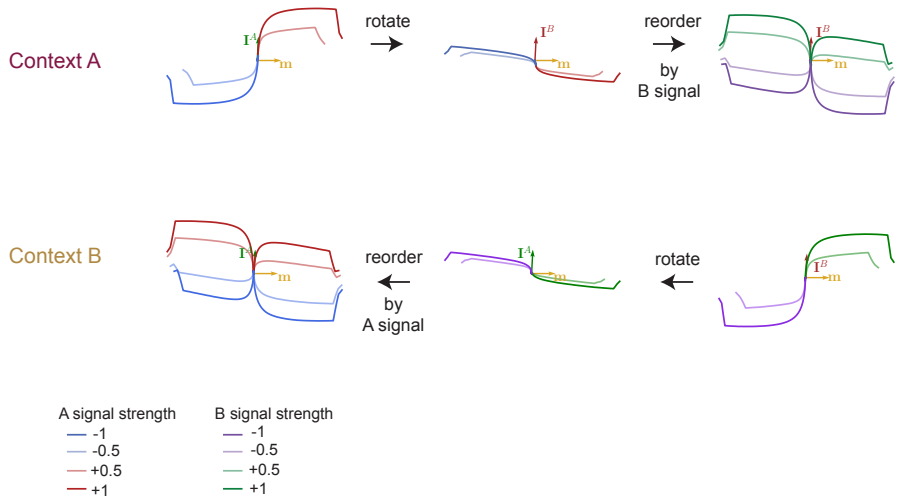


FIGURE A.8: **Context-dependent decision-making state-space dynamics**. Here we reproduce figures akin to those presented in (118) for the trained low-rank network used in figures 4 and 5. We generate 32 conditions corresponding to different combinations of context, signal A coherence and signal B coherence and then project condition-averaged trajectories either on the plane spanned by the recurrent connectivity vector $\boldsymbol{m}$ (which corresponds to the choice axis) and the input vector $\boldsymbol{I}^A$, or on the $\boldsymbol{m} - \boldsymbol{I}^B$ plane. Similarly to what was observed in (118), signal A strength is encoded along the $\boldsymbol{I}^A$ axis, even when it is irrelevant (lower left corner), and signal B strength is encoded along the $\boldsymbol{I}^B$ axis, even when it is irrelevant (top right corner).

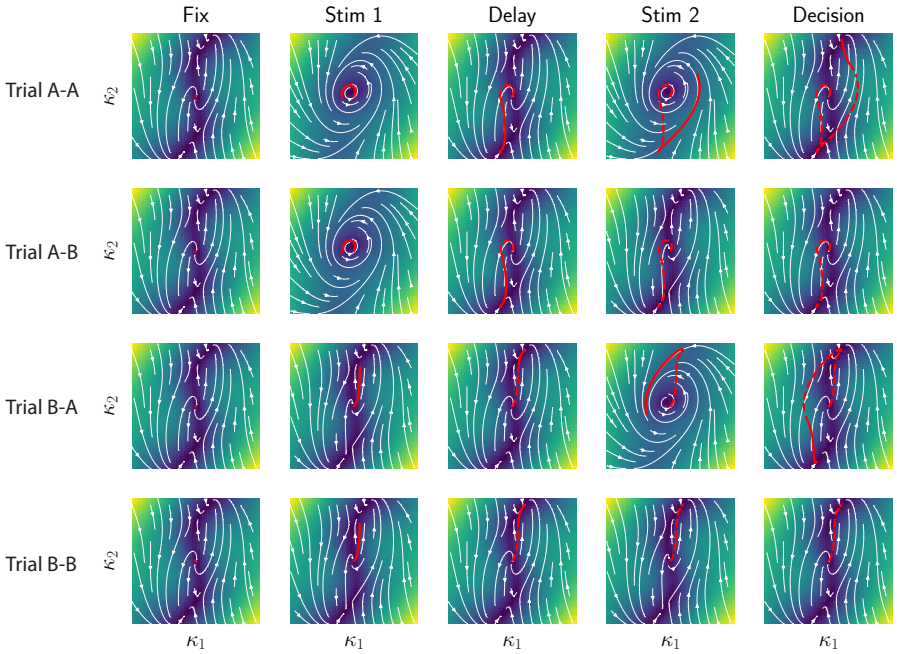


Figure A.9: Dynamics of the variables $\kappa_1$ and $\kappa_2$ for a rank 2 network trained on the delay-match-to-sample task on 4 different trial types and for each epoch of the task. The trajectories are superposed on the flow field of the network as visible on an affine plane of the state space (see main figure 4i), dependent on the input present at each epoch. Dotted lines indicate the parts of the trajectory from previous epochs.

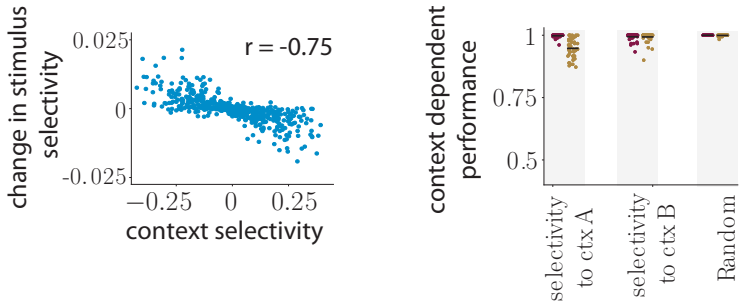


Figure A.10: **Control for the strength of context cues in the MDM task.** Here the context input vectors have been multiplied by a factor five compared to the network analyzed in Fig. III.6g. (a) Context cues are thus able to set the functioning point of some neurons closer to the saturating part of the transfer function, leading to the observation of non-linear mixed-selectivity between context and changes in sensory representation with context. (b) As opposed to the CDM task, this particular feature of selectivity is not functional as revealed by specifically inactivating neurons with a high selectivity to context A or B, showing a similar decrease in behavioral performance as for randomly selected neurons.

# Supplements for chapter IV B

## B.1 Kalman filtering equations

We reproduce in this appendix the recurrence equations followed by the conditional distributions in equation (IV.7) for both the latent LDS and the linear RNN models.

For the latent LDS model, the conditional distributions are Gaussians and their form is given by the Kalman filter equations (88; 232; 222). Following (232), we observe that for any two timesteps $\tau \leq t$ the conditional distributions $P(\boldsymbol{y}_{t+1}|\boldsymbol{y}_\tau, \dots, \boldsymbol{y}_0)$ and $P(\boldsymbol{x}_{t+1}|\boldsymbol{y}_\tau, \dots, \boldsymbol{y}_0)$ are Gaussian, and we introduce the notations :

$$P(\boldsymbol{y}_t|\boldsymbol{y}_\tau, \dots, \boldsymbol{y}_0) := \mathcal{N}(\hat{\boldsymbol{y}}_t^\tau, \mathbf{W}_t^\tau) \tag{B.1}$$
$$P(\boldsymbol{x}_t|\boldsymbol{y}_\tau, \dots, \boldsymbol{y}_0) := \mathcal{N}(\hat{\boldsymbol{x}}_t^\tau, \mathbf{V}_t^\tau) \tag{B.2}$$

In particular, we are interested in expressing $\hat{\boldsymbol{y}}_{t+1}^t$ and $\hat{\boldsymbol{x}}_{t+1}^t$, which are the predicted future observation and latent state, but also in $\hat{\boldsymbol{x}}_t^t$ which represents the latent state inferred from the history of observations until timestep $t$ included. To lighten notations, in the main text we remove the exponent when it has one timestep difference with the index, by writing $\hat{\boldsymbol{x}}_{t+1}$, $\hat{\boldsymbol{y}}_{t+1}$, $\mathbf{W}_{t+1}$ and $\mathbf{V}_{t+1}$ instead of respectively $\hat{\boldsymbol{x}}_{t+1}^t$, $\hat{\boldsymbol{y}}_{t+1}^t$, $\mathbf{W}_{t+1}^t$ and $\mathbf{V}_{t+1}^t$.

First, note that we have the natural relationships:

$$\hat{\boldsymbol{x}}_{t+1}^t = \mathbf{A}\hat{\boldsymbol{x}}_t^t \tag{B.3}$$
$$\hat{\boldsymbol{y}}_{t+1}^t = \mathbf{C}\hat{\boldsymbol{x}}_{t+1}^t \tag{B.4}$$
$$\mathbf{V}_{t+1}^t = \mathbf{A}\mathbf{V}_t^t\mathbf{A}^\top + \mathbf{Q} \tag{B.5}$$
$$\mathbf{W}_{t+1}^t = \mathbf{C}\mathbf{V}_{t+1}^t\mathbf{C}^\top + \mathbf{R} \tag{B.6}$$

so that it is sufficient to find expressions for $\hat{\boldsymbol{x}}_t^t$ and $\mathbf{V}_t^t$. After calculations detailed in (232) or (222), we obtain:

$$\hat{\boldsymbol{x}}_t^t = \hat{\boldsymbol{x}}_t^{t-1} + \mathbf{K}_t(\boldsymbol{y}_t - \mathbf{C}\hat{\boldsymbol{x}}_t^{t-1}) \tag{B.7}$$
$$\mathbf{V}_t^t = (\mathbf{I} - \mathbf{K}_t\mathbf{C})\mathbf{V}_t^{t-1} \tag{B.8}$$

where $\mathbf{K}_t$ is the Kalman gain given by:

$$\mathbf{K}_t = \mathbf{V}_t^{t-1}\mathbf{C}^\top(\mathbf{C}\mathbf{V}_t^{t-1}\mathbf{C}^\top + \mathbf{R})^{-1}. \tag{B.9}$$

These equations form a closed recurrent system, as can be seen by combining (B.3) and (B.7), and (B.5) and (B.8) to obtain a self-consistent set of recurrence equations for the predicted latent state and its variance:

$$\hat{\boldsymbol{x}}_{t+1}^t = \mathbf{A}(\hat{\boldsymbol{x}}_t^{t-1} + \mathbf{K}_t(\boldsymbol{y}_t - \mathbf{C}\hat{\boldsymbol{x}}_t^{t-1})) \tag{B.10}$$

$$\begin{aligned}\mathbf{V}_{t+1}^t &= \mathbf{A}(\mathbf{I} - \mathbf{K}_t\mathbf{C})\mathbf{V}_t^{t-1}\mathbf{A}^\top + \mathbf{Q} \\ &= \mathbf{A}(\mathbf{I} - \mathbf{V}_t^{t-1}\mathbf{C}^\top(\mathbf{C}\mathbf{V}_t^{t-1}\mathbf{C}^\top + \mathbf{R})^{-1}\mathbf{C})\mathbf{V}_t^{t-1}\mathbf{A}^\top + \mathbf{Q}\end{aligned} \tag{B.11}$$

From (B.10) we see that the predicted state at time $t+1$, and thus the predicted observation, depends on observations at timesteps $\tau \leq t-1$ through the term $\hat{\boldsymbol{x}}_t$, making the system non-Markovian. Also note that equations for the variances don't involve any of the observations $\boldsymbol{y}_t$, showing these are exact values and not estimations.

This derivation however is not valid in the limit case $\mathbf{R} = \mathbf{0}$, since $\mathbf{K}_t$ is then undefined. In that case however, we can observe that $\boldsymbol{y}_t$ lies in the linear subspace spanned by the columns of $\mathbf{C}$, so that one can simply replace (B.7) by:

$$\hat{\boldsymbol{x}}_t^t = \mathbf{C}^+\boldsymbol{y}_t = \boldsymbol{x}_t, \tag{B.12}$$

where $\mathbf{C}^+ = (\mathbf{C}^\top\mathbf{C})^{-1}\mathbf{C}^\top$ is the pseudoinverse of $\mathbf{C}$. Since this equation is deterministic, the variance of the estimated latent state is equal to $\mathbf{0}$, so that equation (B.8) becomes $\mathbf{V}_t^t = \mathbf{0}$. This case can be encompassed by equations (B.3) - (B.8) if we rewrite the Kalman gain as:

$$\mathbf{K}_t = \mathbf{C}^+ = (\mathbf{C}^\top\mathbf{C})^{-1}\mathbf{C}^\top. \tag{B.13}$$

Finally, for the linear RNN, the conditional distribution of equation (B.1) is directly given by :

$$P(\boldsymbol{y}_{t+1}|\boldsymbol{y}_t, \ldots, \boldsymbol{y}_0) = \mathcal{N}(\mathbf{J}\boldsymbol{y}_t, \mathbf{P}) \tag{B.14}$$

which shows that the predicted observation only depends on the last one, making the system Markovian.

## B.2 Equivalence in the large network limit

Here we make the assumption that the coefficients of the observation matrix are generated randomly and independently. We show that in the limit of large $n$ with $d$ fixed, one obtains $\mathbf{K}_t\mathbf{C} \to \mathbf{I}$ so that the LDS is asymptotically Markovian and can therefore be exactly mapped to an RNN.

We start by considering a latent LDS whose conditional distributions obey equations (B.1) - (B.11), with the Kalman gain obeying (B.9). To simplify (B.9), we focus on the steady state where variance $\mathbf{V}_t$ has reached its stationary limit $\mathbf{V}$ in (B.11).

Without loss of generality, we reparametrize the LDS by applying a change of basis to the latent states such that $\mathbf{V} = \mathbf{I}$. We also apply a change of basis to the observation space such that $\mathbf{R} = \mathbf{I}$ in the new basis (this transformation does not impact the conditional dependencies between the $\boldsymbol{y}_t$ at different timesteps, and it can also be shown that it cancels out in the expression $\mathbf{K}_t\mathbf{C}$). The equation (B.9) then becomes:

$$\mathbf{K}_t\mathbf{C} = \mathbf{C}^\top(\mathbf{I} + \mathbf{C}\mathbf{C}^\top)^{-1}\mathbf{C}. \tag{B.15}$$

Applying the matrix inversion lemma gives $(\mathbf{I} + \mathbf{C}\mathbf{C}^\top)^{-1} = \mathbf{I} - \mathbf{C}(\mathbf{I} + \mathbf{C}^\top\mathbf{C})^{-1}\mathbf{C}^\top$, from which we get:

$$\mathbf{K}_t\mathbf{C} = \mathbf{C}^\top\mathbf{C} - \mathbf{C}^\top\mathbf{C}(\mathbf{I} + \mathbf{C}^\top\mathbf{C})^{-1}\mathbf{C}^\top\mathbf{C}.$$

Using a Taylor expansion we then write:

$$\begin{aligned}(\mathbf{I}+\mathbf{C}^\top\mathbf{C})^{-1} &= (\mathbf{I}+(\mathbf{C}^\top\mathbf{C})^{-1})^{-1}(\mathbf{C}^\top\mathbf{C})^{-1} \\ &= (\sum_{k=0}^{\infty}(-(\mathbf{C}^\top\mathbf{C})^{-1})^k)(\mathbf{C}^\top\mathbf{C})^{-1} \\ &\approx (\mathbf{C}^\top\mathbf{C})^{-1} - ((\mathbf{C}^\top\mathbf{C})^{-1})^2 + ((\mathbf{C}^\top\mathbf{C})^{-1})^3,\end{aligned}$$

which gives:

$$\begin{aligned}\mathbf{K}_t\mathbf{C} &\approx \mathbf{C}^\top\mathbf{C} - \mathbf{C}^\top\mathbf{C}(\mathbf{C}^\top\mathbf{C})^{-1}\mathbf{C}^\top\mathbf{C} + \mathbf{C}^\top\mathbf{C}((\mathbf{C}^\top\mathbf{C})^{-1})^2\mathbf{C}^\top\mathbf{C} - \mathbf{C}^\top\mathbf{C}((\mathbf{C}^\top\mathbf{C})^{-1})^3\mathbf{C}^\top\mathbf{C} \\ &\approx \mathbf{C}^\top\mathbf{C} - \mathbf{C}^\top\mathbf{C} + \mathbf{I} - (\mathbf{C}^\top\mathbf{C})^{-1}.\end{aligned}$$

Assuming the coefficients of the observation matrix are iid. with zero mean and unit variance, for $n$ large we obtain $\mathbf{C}^\top\mathbf{C} = n\mathbf{I} + \mathcal{O}(\sqrt{n})$ from the central limit theorem, so that $(\mathbf{C}^\top\mathbf{C})^{-1} = \mathcal{O}(1/n)$ (which can again be proven with a Taylor expansion). This finally leads to $\mathbf{K}_t\mathbf{C} = \mathbf{I} + \mathcal{O}(1/n)$.

An alternative proof takes advantage of the spectral theorem applied to $\mathbf{C}^\top\mathbf{C}$. Indeed, since it is a symmetric matrix, it can be decomposed as $\mathbf{C}^\top\mathbf{C} = \mathbf{U}\mathbf{D}\mathbf{U}^\top$ where $\mathbf{U}$ is an orthonormal matrix and $\mathbf{D}$ the diagonal matrix of eigenvalues. Starting from (B.15) we derive:

$$\begin{aligned}\mathbf{K}_t\mathbf{C} &= \mathbf{C}^\top\mathbf{C} - \mathbf{C}^\top\mathbf{C}(\mathbf{I}+\mathbf{C}^\top\mathbf{C})^{-1}\mathbf{C}^\top\mathbf{C} \\ &= \mathbf{U}\mathbf{D}\mathbf{U}^\top - \mathbf{U}\mathbf{D}\mathbf{U}^\top(\mathbf{I}+\mathbf{U}\mathbf{D}\mathbf{U}^\top)^{-1}\mathbf{U}\mathbf{D}\mathbf{U}^\top \\ &= \mathbf{U}\mathbf{D}\mathbf{U}^\top - \mathbf{U}\mathbf{D}\mathbf{U}^\top(\mathbf{U}(\mathbf{D}+\mathbf{I})\mathbf{U}^\top)^{-1}\mathbf{U}\mathbf{D}\mathbf{U}^\top \\ &= \mathbf{U}\mathbf{D}\mathbf{U}^\top - \mathbf{U}\mathbf{D}\mathbf{U}^\top\mathbf{U}(\mathbf{D}+\mathbf{I})^{-1}\mathbf{U}^\top\mathbf{U}\mathbf{D}\mathbf{U}^\top \\ &= \mathbf{U}\mathbf{D}\mathbf{U}^\top - \mathbf{U}\mathbf{D}^2(\mathbf{D}+\mathbf{I})^{-1}\mathbf{U}^\top \\ &= \mathbf{U}(\mathbf{D} - \mathbf{I}/(\mathbf{D}+\mathbf{I}))\mathbf{U}^\top \\ &= \mathbf{U}(\mathbf{D}/(\mathbf{D}+\mathbf{I}))\mathbf{U}^\top \\ &= \mathbf{U}(\mathbf{I} - \mathbf{I}/(\mathbf{D}+\mathbf{I}))\mathbf{U}^\top \\ &= \mathbf{I} - \mathbf{U}(\mathbf{I}/(\mathbf{D}+\mathbf{I}))\mathbf{U}^\top\end{aligned}$$

Assuming as before that the coefficients of $\mathbf{C}$ are iid. Gaussian with zero mean and unit variance, $\mathbf{C}^\top\mathbf{C}$ is then the empirical covariance of iid. samples of a Gaussian ensemble with identity matrix covariance. The matrix $\mathbf{C}^\top\mathbf{C} = \mathbf{U}\mathbf{D}\mathbf{U}^\top$ then follows the $(\mathbf{I}, n)$-Wishart distribution, and for $n$ large its eigenvalues are all greater than $\sqrt{n}$ (using for example the tail bounds of (218), Theorem 6.1). This shows that $(\mathbf{I}/(\mathbf{D}+\mathbf{I})) = \mathcal{O}(1/\sqrt{n})\mathbf{I}$, completing the proof.

## B.3 Derivation of the RNN to LDS mapping

As mentioned in section IV.4, we consider an RNN defined by (IV.4), with $\mathbf{J} = \mathbf{M}\mathbf{N}^\top$ and note $\mathbf{C}$ an orthonormal matrix whose columns form a basis of $\mathcal{F}$, the linear subspace spanned by the columns of $\mathbf{M}$ and $\mathbf{N}$. Note that $\mathbf{C}\mathbf{C}^\top$ is an orthogonal projector onto the subspace $\mathcal{F}$, and that since all columns of $\mathbf{M}$ and $\mathbf{N}$ belong to this subspace we have $\mathbf{C}\mathbf{C}^\top\mathbf{M} = \mathbf{M}$ and $\mathbf{C}\mathbf{C}^\top\mathbf{N} = \mathbf{N}$. Hence, we have:

$$\mathbf{C}\mathbf{C}^\top\mathbf{J}\mathbf{C}\mathbf{C}^\top = \mathbf{J}. \tag{B.16}$$

We thus define the latent vector as $\boldsymbol{x}_t = \mathbf{C}^\top \boldsymbol{y}_t$, and we can then write:

$$\begin{aligned}
\boldsymbol{x}_{t+1} &= \mathbf{C}^\top \boldsymbol{y}_{t+1} \\
&= \mathbf{C}^\top \mathbf{J} \boldsymbol{y}_t + \mathbf{C}^\top \boldsymbol{\epsilon}_t \\
&= \mathbf{C}^\top \mathbf{C}\mathbf{C}^\top \mathbf{J}\mathbf{C}\mathbf{C}^\top \boldsymbol{y}_t + \mathbf{C}^\top \boldsymbol{\epsilon}_t \quad \text{(by (B.16))} \\
&= \mathbf{C}^\top \mathbf{J}\mathbf{C}\mathbf{C}^\top \boldsymbol{y}_t + \mathbf{C}^\top \boldsymbol{\epsilon}_t \quad \text{(because } \mathbf{C}^\top\mathbf{C} = \mathbf{I}\text{)} \\
&= \mathbf{A}\boldsymbol{x}_t + \boldsymbol{w}_t,
\end{aligned}$$

where we have defined the recurrence matrix $\mathbf{A} = \mathbf{C}^\top \mathbf{J}\mathbf{C}$ and the latent dynamics noise $\boldsymbol{w}_t = \mathbf{C}^\top \boldsymbol{\epsilon}_t$ which follows $\boldsymbol{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ with $\mathbf{Q} = \mathbf{C}^\top \mathbf{P}\mathbf{C}$.

Let us define $\boldsymbol{v}_t = \boldsymbol{y}_t - \mathbf{C}\boldsymbol{x}_t = (\mathbf{I} - \mathbf{C}\mathbf{C}^\top)\boldsymbol{y}_t$. We need to determine the conditions under which $\boldsymbol{v}_t$ is (i) normally distributed, and (ii) independent of $\boldsymbol{y}_{t-1}$ and $\boldsymbol{x}_t$. For this, we write:

$$\begin{aligned}
\mathbf{C}\boldsymbol{x}_t &= \mathbf{C}\mathbf{A}\boldsymbol{x}_{t-1} + \mathbf{C}\boldsymbol{w}_{t-1} \\
&= \mathbf{C}\mathbf{C}^\top \mathbf{J}\mathbf{C}\boldsymbol{x}_{t-1} + \mathbf{C}\boldsymbol{w}_{t-1} \\
&= \mathbf{C}\mathbf{C}^\top \mathbf{J}\mathbf{C}\mathbf{C}^\top \boldsymbol{y}_{t-1} + \mathbf{C}\boldsymbol{w}_{t-1} \\
&= \mathbf{J}\boldsymbol{y}_{t-1} + \mathbf{C}\boldsymbol{w}_{t-1},
\end{aligned}$$

and hence:

$$\begin{aligned}
\boldsymbol{v}_t &= \boldsymbol{\epsilon}_{t-1} - \mathbf{C}\boldsymbol{w}_{t-1} \\
&= (\mathbf{I} - \mathbf{C}\mathbf{C}^\top)\boldsymbol{\epsilon}_{t-1},
\end{aligned}$$

which is independent of $\boldsymbol{y}_{t-1}$ and has a marginal distribution $\boldsymbol{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ with $\mathbf{R} = \mathbf{P} - \mathbf{C}\mathbf{C}^\top \mathbf{P}\mathbf{C}\mathbf{C}^\top$, but is not in general independent of $\boldsymbol{w}_{t-1}$. A sufficient and necessary condition for the independence of $\boldsymbol{w}_{t-1}$ and $\boldsymbol{v}_t$ is that the RNN noise covariance $\mathbf{P}$ has all its eigenvectors either aligned with or orthogonal to the subspace $\mathcal{F}$ (in this case, the covariance $\mathbf{R}$ is degenerate and has $\mathcal{F}$ as a null space, which implies that observation noise is completely orthogonal to $\mathcal{F}$). If that is not the case, the reparametrization stays valid up to the fact that the observation noise $\boldsymbol{v}_t$ and the latent dynamics noise $\boldsymbol{w}_t$ can be correlated.

## B.4 Addition of input terms

Let us consider an extension of both the latent LDS and the linear RNN models to take into account inputs. More specifically we will consider adding to both model classes an input under the form of a time-varying signal $u_t$ fed to the network through a constant set of input weights. In the latent LDS model, the input is fed directly to the latent variable and equations (IV.1)-(IV.2) become :

$$\boldsymbol{x}_t = \mathbf{A}\boldsymbol{x}_{t-1} + \mathbf{B}u_t + \boldsymbol{w}_t, \qquad \boldsymbol{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}) \tag{B.17}$$

$$\boldsymbol{y}_t = \mathbf{C}\boldsymbol{x}_t + \boldsymbol{v}_t, \qquad \boldsymbol{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R}), \tag{B.18}$$

The linear RNN equation (IV.4) becomes :

$$\boldsymbol{y}_t = \mathbf{J}\boldsymbol{y}_{t-1} + \mathbf{W}_{in}u_t + \boldsymbol{\epsilon}_t, \qquad \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{P}), \tag{B.19}$$

so that we will represent by $\mathbf{B}$ a low-dimensional input projection, and $\mathbf{W}_{in}$ a high-dimensional one.

For the LDS to RNN mapping, we can directly adapt the derivations of section IV.3.2, which lead to :

$$\boldsymbol{y}_{t+1} \mid \boldsymbol{y}_t \sim \mathcal{N}\left(\mathbf{C}\mathbf{B}u_t + \mathbf{J}_t \boldsymbol{y}_t, \mathbf{P}_t\right) \tag{B.20}$$

with the same expressions for $\mathbf{J}_t$ and $\mathbf{P}_t$, given in equations (IV.17)-(IV.18).

For the RNN to LDS mapping, assuming again that $\mathbf{J}$ is low-rank and written as $\mathbf{J} = \mathbf{M}\mathbf{N}^\top$, we can define:

$$\boldsymbol{x}_t = \mathbf{C}^\top \boldsymbol{y}_t$$

where $\mathbf{C}$ is a matrix whose columns form an orthonormal basis for the subspace $\mathcal{F}$ spanned by the columns of $\mathbf{M}$, $\mathbf{N}$ and $\mathbf{W}_{in}$. This latent vector then follows the dynamics:

$$\boldsymbol{x}_{t+1} = \mathbf{C}\mathbf{J}\mathbf{C}^\top \boldsymbol{x}_t + \mathbf{C}^\top \mathbf{W}_{in} u_t + \mathbf{C}^\top \boldsymbol{\epsilon}_t \tag{B.21}$$

which corresponds to equation (B.17), and it is straightforward to show that it leads to equation (B.18), with the technical condition that the covariance of $\boldsymbol{\epsilon}_t$ should have its eigenvectors aligned with the subspace $\mathcal{F}$ to avoid correlations between observation and recurrent noises.

# Supplements for chapter V C

## C.1 Cognitive tasks

Here we describe the input and output structure for the four cognitive tasks used in this study. In all tasks we use the notations from section 2, considering $N_{in}$ input signals $u_s(t)$ and one target output $z^*(t)$, that can be defined only at specific timepoints of a trial. Durations are given in an abstract time mapping, but tasks are implemented in a discretized time with timestep $dt = 20ms$

**Decision-making task (DM)**. The network receives one input signal $u_s(t)$, equal to Gaussian white noise with standard deviation 0.1 (as for subsequent tasks), added to a mean coherence $\overline{u}$ drawn uniformly from $\pm 0.1 \times \{1, 2, 4\}$. The target output $z^*$ is defined only at the final timestep of the trial, and is equal to the sign of the trial coherence. Each trial starts with a 100ms fixation period with no input, followed by an 800ms stimulus epoch, a 300ms delay epoch and a decision timestep.

**Working memory task (WM)**. This task is inspired on the traditional parametric working memory and comparison experimental paradigm. The network receives one input signal, equal to:

$$u(t) = f_1\delta_1(t) + f_2\delta_2(t) + \xi(t)$$

where $f_1$ is randomly sampled in $[10, .., 34]$, $f_2 - f_1$ is randomly sampled in $\{-24, -16, -8, 8, 16, 24\}$ with the constraint that $10 \leq f_2 \leq 34$ and $\xi(t)$ is white Gaussian noise. The target output $z^*(t)$ is defined only at the final timestep and equal to the exact value $f_2 - f_1$. Each trial starts with a 100ms fixation period with no input, followed by a 100ms stimulus 1 epoch (where $\delta_1(t) = 1$), followed by a 500ms delay epoch, followed by a 100ms stimulus 2 epoch (where $\delta_2(t) = 1$) and a decision timestep.

**Context-dependent decision-making (CDM)**. This task aims at modelling the experimental work of (Mante, Sussillo et al., 2013) (118). The network receives four inputs, two noisy input signals $u_A(t)$ and $u_B(t)$ defined as for the DM task with independently drawn coherences and noise, and two contextual inputs $u_{ctxA}(t)$ and $u_{ctxB}(t)$ defined as a one-hot encoding of the trial context. The target output during the final task epoch is set to the sign of the coherence of the input indicated by the active contextual cue.

For the application of LINT to a full-rank network, the task was defined in five epochs: a 100ms fixation epoch with no inputs, a 350ms epoch with only contextual inputs, an 800ms stimulus epoch, with both noisy stimuli and contextual inputs, a 100 ms delay epoch and a 20ms decision epoch which is the only one where the target output $z^*(t)$ was defined.

For the application of LINT to electrophysiological recordings, four epochs were used: a 350ms epoch where only contextual inputs were active, a 650ms stimulation epoch, an

80ms delay epoch and a 20ms decision epoch (used for computing task accuracy on the fitted networks). The trained networks were constrained to reproduce neural activity only during the last three epochs. The coherences used in this part of the work were sampled from $\pm\{0.047, 0.15, 0.5\}$ for monkey A, $\pm\{0.07, 0.19, 0.54\}$ for monkey F.

**Delay Match-to-Sample (DMS)**. This task reproduces a paradigm where two consecutive stimuli each either of type A or B are presented, and the subject distinguishes between matches (both stimuli of the same type) and non-matches (stimuli of different types). In our models, the stimuli are given through two input signals $u_A(t)$ and $u_B(t)$, with Gaussian white noise centered around 0 or 1 while the corresponding stimulus is active. The task comprises five epochs, a fixation epoch of duration 100ms, a first stimulation epoch of 500ms, a delay epoch of variable duration between 500 and 3000ms, a second stimulation epoch of 500ms and a decision epoch of 1000ms. During each stimulation epoch a single type between A and B is sampled and the corresponding input signal is set to a mean of 1. The target output $z^*(t)$ during the decision epoch is equal to 1 for a match, -1 for a non-match.

## C.2 Training details and hyperparameters.

**Task-optimized lrRNNs**. The full-rank RNNs were defined following a discretized version of equation (V.1):

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \frac{\Delta t}{\tau}\left(-\boldsymbol{x}_t + \mathbf{J}\phi(\boldsymbol{x}_t) + \sum_{s=1}^{N_{in}} \boldsymbol{I}^{(s)} u_{s,t} + \boldsymbol{\eta}_t\right), \tag{C.1}$$

with $\boldsymbol{\eta}_t$ a random normal vector with independent entries and standard deviation 0.05 on each entry, and $\Delta t = 20ms$, $\tau = 100ms$. All networks were defined in pytorch (140) and trained using the ADAM optimizer. We trained networks on 800 random trials, for a certain number of epochs being divided in 25 batches of 32 trials. We considered networks with $N = 512$ units for this part of the work.

For the low-rank RNNs (section V.3), we trained the $\boldsymbol{m}^{(r)}$ and $\boldsymbol{n}^{(r)}$ vectors. For each task, we found the minimal rank by training networks of increasing rank until they performed the task with more than 95% accuracy. For the DM and CDM tasks, the weights were all initialized following a random Gaussian distribution of standard deviation 1, and 4 for the readout weights $w_i$. For the WM task, the rank-two networks were initialized from the SVD of the connectivity matrix of a full-rank RNN previously trained on the task. For the DMS task, this initialization trick was also used, as well as the following shaping procedure: rank-two networks were first trained on trials with a maximal delay of 700ms, then 1000ms, and finally 4000ms. The learning rates used were of 0.01 for all low-rank networks, and on the order of $10^{-4}$ for full-rank networks.

For the full-rank RNNs (section V.4), we trained the connectivity matrix $\mathbf{J}$. The initial weights were sampled from centered Gaussian distributions with a standard deviation of 1 for input and readout weights, and $\rho/\sqrt{N}$ for connectivity coefficients $J_{ij}$. The values of $\rho$ used in the main text were 0.1 for the CDM task, and 0.8 for the DMS task, with results for other values displayed in Sup. Fig. **??**.

**LINT on data from task-optimized lrRNNs**. For the application of LINT to synthetic data generated from the task-optimized lrRNNs, we generated 800 trials for each task, simulated the trajectories displayed by the task-optimized networks on these trials, and trained identical RNNs, all initialized with random weights, to reproduce these trajectories. The networks were then tested on 800 newly sampled trajectories to obtain the reported values of $R^2$. Given trials $k \in \{1, \dots, K\}$, timesteps $t \in \{1, \dots, T\}$ and neurons $i \in \{1, \dots, N\}$, the global $R^2$ is computed as:

$$R^2 = 1 - \frac{\sum_k \sum_t \sum_i \left( x_i^{(k)}(t) - \tilde{x}_i^{(k)}(t) \right)^2}{\sum_k \sum_t \sum_i \left( \tilde{x}_i^{(k)}(t) - \langle \tilde{x}_i^{(k)}(t) \rangle_{k,t,i} \right)^2} \tag{C.2}$$

following the notations of section V.2.

**LINT on data from task-optimized vanilla RNNs**. For this application of LINT, synthetic data was generated from task-optimized full-rank RNNs by simulating their responses to 800 trials for each of the two tasks. Inferred low-rank networks were trained for 500 epochs on those trials, and then tested on 800 newly generated trials. Reported $R^2$ values were computed with equation (C.2).

**LINT on electrophysiological data**. Low-rank networks trained to reproduce trajectories were trained on a set of 64 random conditions out of 72, and tested on the remaining 8 conditions to compute reported $R^2$. To compute the task accuracy, a linear decoder $\boldsymbol{w}$ was trained on the decision epoch of the task to report the correct choice on all 72 conditions, and the obtained accuracy was reported.

## C.3 Delay Match-to-Sample full-rank network reverse-engineering

To display another example of how LINT can be exploited to reverse-engineer mechanisms used by a "black-box" full-rank RNN, we apply it to the networks trained on another task. Here we consider the DMS task, where the network receives two consecutive stimuli chosen among two possible classes A and B, separated by a delay period, and has to output during its response period a positive value if the stimuli were of the same class ("match"), a negative value otherwise ("non-match", see full details in appendix C.1).

When training full-rank RNNs on this task, it appeared that they could be well approximated by lrRNNs of a rank usually equal to two (Fig. V.2), irrespective of their training hyperparameters (Sup. Fig. C.1). Here, we focus on a full-rank RNN trained with an initial connectivity of standard deviation equal to $0.8/\sqrt{N}$ and $N = 512$ units, which was well approximated by a rank-two lrRNN ($R^2$ fit qualities for individual neurons shown in Sup. Fig. C.3a). Indeed, this network, although trained without any constraint on its connectivity, exhibited low-dimensional trajectories as can be seen by performing a PCA (Sup. Fig. C.3b). An idea of how neural geometry enables the networks to perform the task can be obtained by observing projections of the trajectories on spaces spanned by the first principal components. Typically, observing projections on the top 3 components (Sup. Fig. C.3c), it becomes apparent that the network relies on four fixed points, one for remembering that the first stimulus is A during the delay period (middle right), one for remembering that the first stimulus is B (at the left), also used to indicate a match, one to indicate a non-match (at the right) and finally one to indicate only a match A-A (at the right). Stimulus inputs seem to drive activity through transient tunnels from one fixed point to another. This picture provides a certain grasp of phenomena happening in the network, but does not illuminate how dynamics enable activity to correctly jump between fixed points, nor how connectivity enables this dynamical picture to emerge.

The rank-two network inferred by LINT has its activity constrained by design to a four-dimensional subspace of the neural state-space, spanned by the two input vectors of the network $\boldsymbol{I}^A$ and $\boldsymbol{I}^B$ as well as the two output vectors of the recurrent connectivity $\boldsymbol{m}^{(1)}$ and $\boldsymbol{m}^{(2)}$. These four vectors indeed have a significant overlap with the top four principal components (Sup. Fig. C.3d), with the difference that they disentangle input from recurrent contributions to neural activity. The rank-two connectivity is also characterized by two input-selection vectors of the recurrent connectivity $\boldsymbol{n}^{(1)}$ and $\boldsymbol{n}^{(2)}$, which will be a key driver of the network dynamics, as well as a task readout $\boldsymbol{w}$.

Projecting the neural trajectories on the subspace spanned by the two output recurrent vectors shows indeed that the activity of the rank-two network reproduces well that of the

original network, and that both networks rely on four fixed points to perform the task in the manner outlined above (Sup. Fig. C.3e). In the absence of inputs, the neural dynamics of the rank-two network stay confined to the two-dimensional $\boldsymbol{m}^{(1)}$-$\boldsymbol{m}^{(2)}$ space, so that the full autonomous dynamics can be visualized as a vector field, illuminating the dynamical inner workings of such networks (Sup. Fig. C.3f). This reveals the existence of the four stable fixed points in the dynamical landscape, as well as an unstable fixed point at the origin and four saddle points. Moreover, while tonic inputs are received, which is the case during each stimulation period, the dynamics shift to a two-dimensional affine subspace of the neural state space, parallel to the $\boldsymbol{m}^{(1)}$-$\boldsymbol{m}^{(2)}$ plane but shifted along the received input vector. The dynamics while a tonic input is received can thus also be visualized as a two-dimensional vector field (Sup. Fig. C.3g), explaining how inputs drive transitions from one fixed point to another. More specifically, only the two stable fixed point on the lower part of the plane are kept during a stimulation by input A, while dynamics have a slight clockwise rotational component. Hence, if input A is received during a short stimulation period, they are driven towards the lower right part of the plane, where they can stay in a fixed point during the delay period (red and orange trajectories), and if a second input A is received, they will be driven towards the lower left fixed point. Conversely, under input B stimulation, only the two fixed points on the top of the plane are kept, with a very slight counterclockwise rotational component on this plane (at least under the vicinity of the origin, although not very visible in the plotted field). Hence, if stimulus B is received first, the network will be driven to the top left fixed point. If a second stimulus B is received, the network will stay in that state, whereas if a second input A is received it will be driven towards the top right fixed point (non-match).

These full dynamical landscapes provide us with a step-by-step decomposition of the task trials, and can also lead to predictions for trials taht do not appear in the task (for example we could predict the network behavior if three stimulations are received, or with longer or shorter stimulations). Moreover, the rank-two connectivity can illuminate how this dynamical behavior emerges from the learned synaptic weights (**? ?** ). A detailed account would go beyond the scope of this text, but some insights can be extracted easily: first, looking at the distribution of the weights of all neurons on vectors $\boldsymbol{m}^{(1)}$ and $\boldsymbol{m}^{(2)}$ (Sup. Fig. C.3h), it appears that they separate in four clusters, each scattered around a different mean forming a quadrilateral on this plane. This type of population structure has been shown to enable the apparition of polygons of stable fixed points in the neural state space (see notably (**?** )). The way in which inputs modify dynamics can also be explained by connectivity features: more specifically, inputs A and B seem to modify the gains of different groups of neurons, as was the case for the contextual cues in the CDM task. Importantly, the neurons that are driven to a low-gain regime by input A (defined as the set of neurons $i$ such that $|I_i^A| > 1$) are characterized by a negative correlation between vectors $\boldsymbol{n}^{(1)}$ and $\boldsymbol{m}^{(2)}$, and neurons driven to a high-gain regime by the same input exhibit a different correlation between these two vectors. Due to this differential distribution, while input A is received the effective coupling between the two latent variables $\kappa_1$ and $\kappa_2$ is modified. A converse situation happens with input B, explaining how input A generates this slight clockwise rotational component in the dynamics, and input B generates an opposite rotation. These local rotations, coupled with correlations between inputs and the recurrent vectors lead to the apparition of the plotted dynamical landscapes. These analyses could be verified by examining gains of neurons and performing ablation studies in the full-rank network, as has been done for the CDM task.

## C.4 Supplementary figures

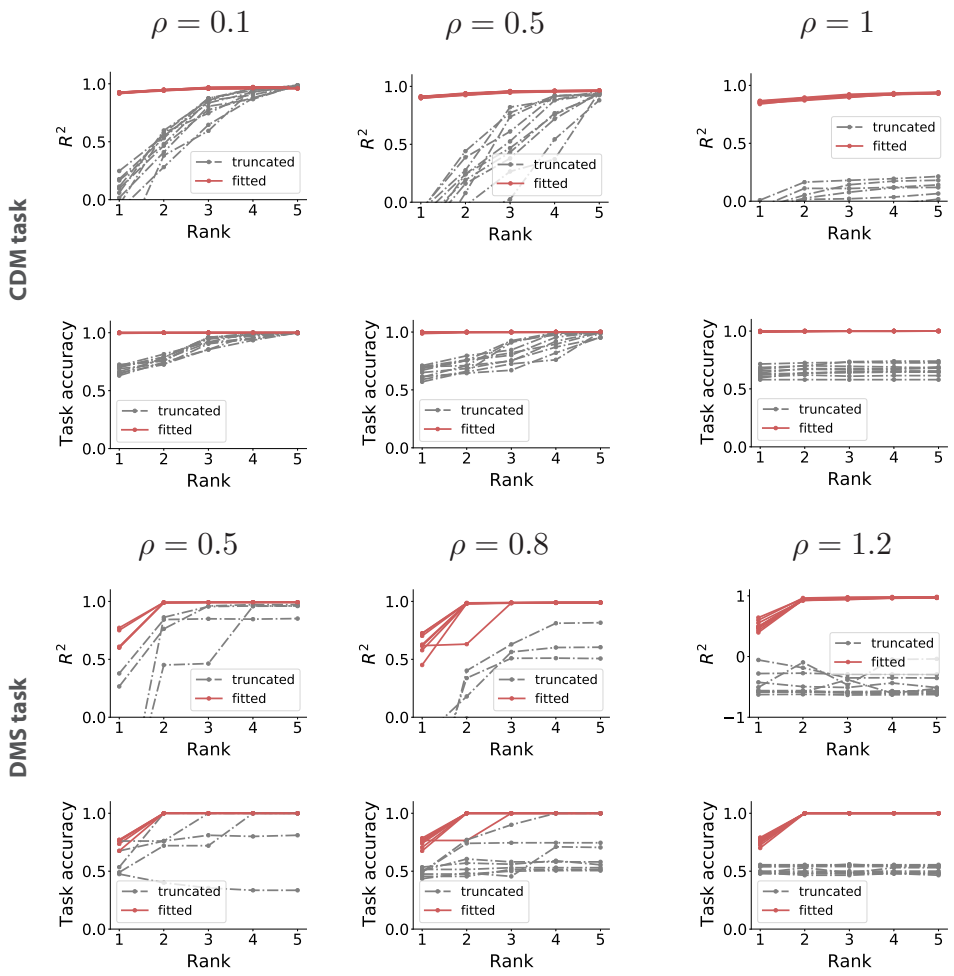


FIGURE C.1: For different values of $\rho$, where the standard deviation of initial recurrent weights is $\rho/\sqrt{N}$, we train 10 unconstrained networks, and either fit low-rank networks to their trajectories with increasing ranks or truncate their connectivity matrices to the same rank. The obtained $R^2$ similarity between original and fitted trajectories and task accuracy when plugging low-rank networks to the original readout are illustrated with a different line for each original unconstrained network. Top 2 rows: experiments on Context-dependent decision making task. Bottow 2 rows: experiments on the Delay Match-to-Sample task. Note that with higher initial random recurrent weights, unconstrained networks tend to go to what has been termed as the "lazy" training regime, with potentially higher dimensional trajectories with respect to the the "rich" training regimes for smaller initial weights (55). This is visible through the poor performance of truncated connectivity matrices, but does not harm the effectiveness of our method.

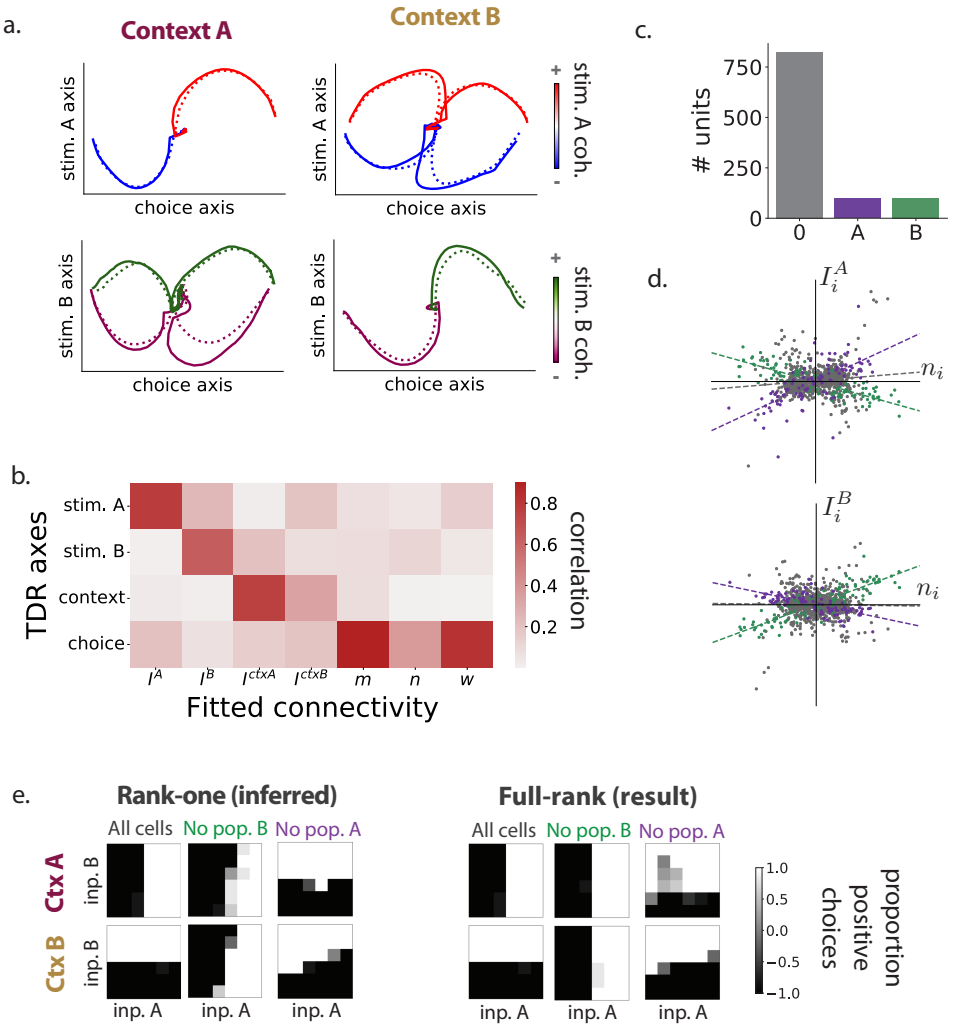


Figure C.2: Additional figures on LINT applied to a full-rank network performing the CDM task (exhibited in figs. V.3 and V.4). a. Low-dimensional projections of trial-averaged population trajectories for several combinations of context, stimuli A and B and choice, as in Fig. V.3 in the original full-rank network (full lines) and the inferred rank-one network (dashed lines), projected on axes found by targeted dimensionality reduction (TDR) (118) applied to the full-rank network. b. Correlation between the four axes found by TDR on the full-rank network and the connectivity axes inferred by LINT. c. Number of units assigned to each of the three populations used for the reverse-engineering. Here, we manually defined population A as the 100 units with the strongest absolute context A input weight in the rank-one network (see Fig. V.3a top), and population B as the 100 units with the strongest context B input weight not in pop. A. Applying Bayesian GMM clustering with 3 components and a strong mean precision prior gives very similar results. d. For the inferred rank-one network, joint distributions of connectivity weights on the input vector $\boldsymbol{I}^A$ and $\boldsymbol{n}$, as well as on $\boldsymbol{I}^B$ and $\boldsymbol{n}$. For each population, linear regression lines are plotted. e. Psychometric response matrices in each context for all combination of stimulus coherences, for the inferred and original network when they are left unperturbed or after lesioning populations A and B. Unperturbed matrices indicate expected behavior. One can observe that inactivation of population B leads the networks to always behave as if in context A (losing its capacity to perform in context B), whereas the opposite phenomenon happens when population A is inactivated.

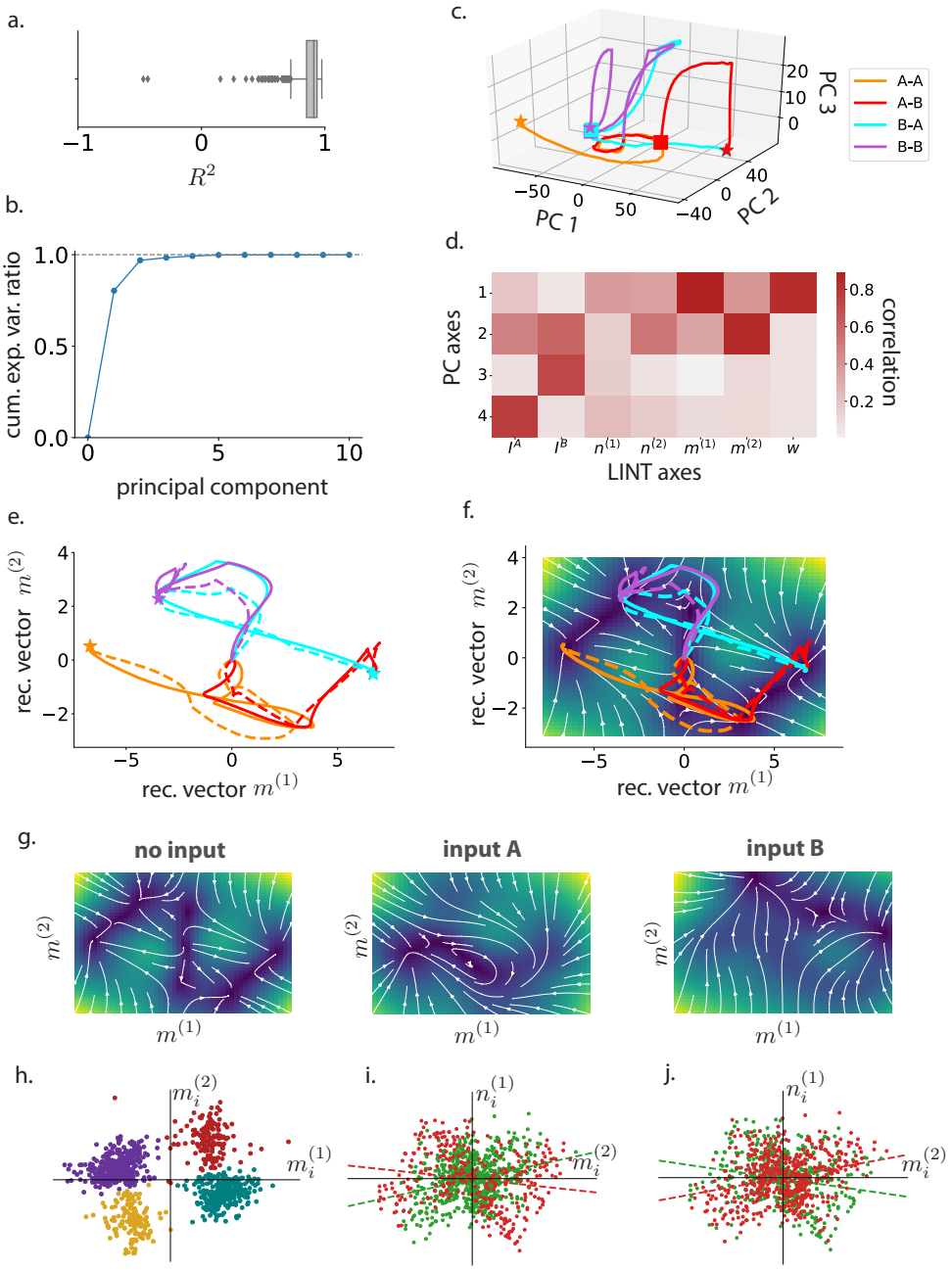


FIGURE C.3: LINT applied to a full-rank RNN trained on the DMS task. A rank-two network was inferred and is analyzed in this figure (see Appendix C.3 for details). a. Boxplot representing the distribution of $R^2$ fitting values for individual neurons. b. Cumulative explained variance ratio for top 10 principal components of a PCA applied to trajectories of the full-rank network. c. Trajectories in the four possible task conditions in the original full-rank network, projected on the top 3 principal components (squares: delay period, stars: end of trial). d. Correlation between axes inferred by a PCA on the full-rank trajectories and connectivity axes of the inferred rank-two network. e. Trajectories of the full-rank network (full lines) and the rank-two model (dashed lines) in the four task conditions, projected on the two recurrent connectivity output vectors $\boldsymbol{m}^{(1)}$ and $\boldsymbol{m}^{(2)}$ (same colors as in c). f. Same trajectories, superposed on the vector field representing autonomous dynamics in the rank-two RNN. Colors indicate speed of the dynamics (blue: slow, yellow:fast). g. Vector fields representing the dynamics on the $\boldsymbol{m}^{(1)}$-$\boldsymbol{m}^{(2)}$ plane. h-j. Connectivity parameter distributions on the rank-two models... h. on the two recurrent output vectors $m^{(1)}$ and $m^{(2)}$ - four populations can be identified by GMM clustering. i. on the connectivity vectors $m^{(2)}$ and $n^{(1)}$ - low-gain neurons while input A is received in red, others in green with overlaid linear regressions for both groups. j. on the same vectors, with low-gain neurons while input B is received in red, others in green.

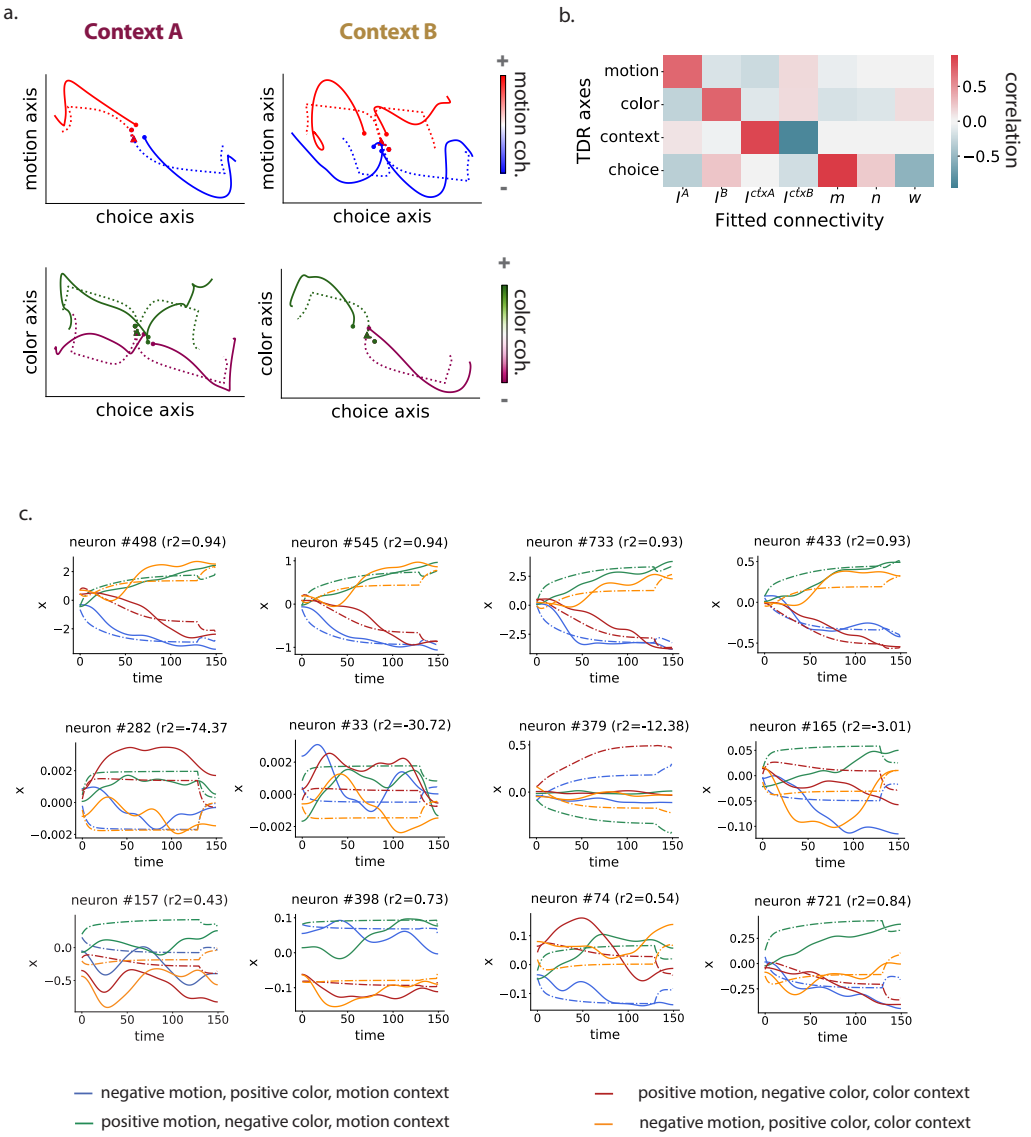


Figure C.4: Additional illustrations of LINT applied to electrophysiological recordings in monkey A. a. Two-dimensional projections of trial-averaged population trajectories for several combinations of context, choice, and motion or color coherence (indicated by the color code), as in Fig. V.5c, in the recorded data (full lines) and the rank-one model (dashed lines), projected on axes inferred by TDR. b. Correlation coefficients between axes identified by TDR and LINT connectivity axes. c. Pre-processed data responses and rank-one model responses for individual neurons to 4 different task conditions (uniquely identified by a context, a color coherence, and a motion coherence. Strongest coherences displayed here). Top row: four best fitted neurons. Middle row: four worse fitted neurons. Bottom row: four randomly selected neurons.

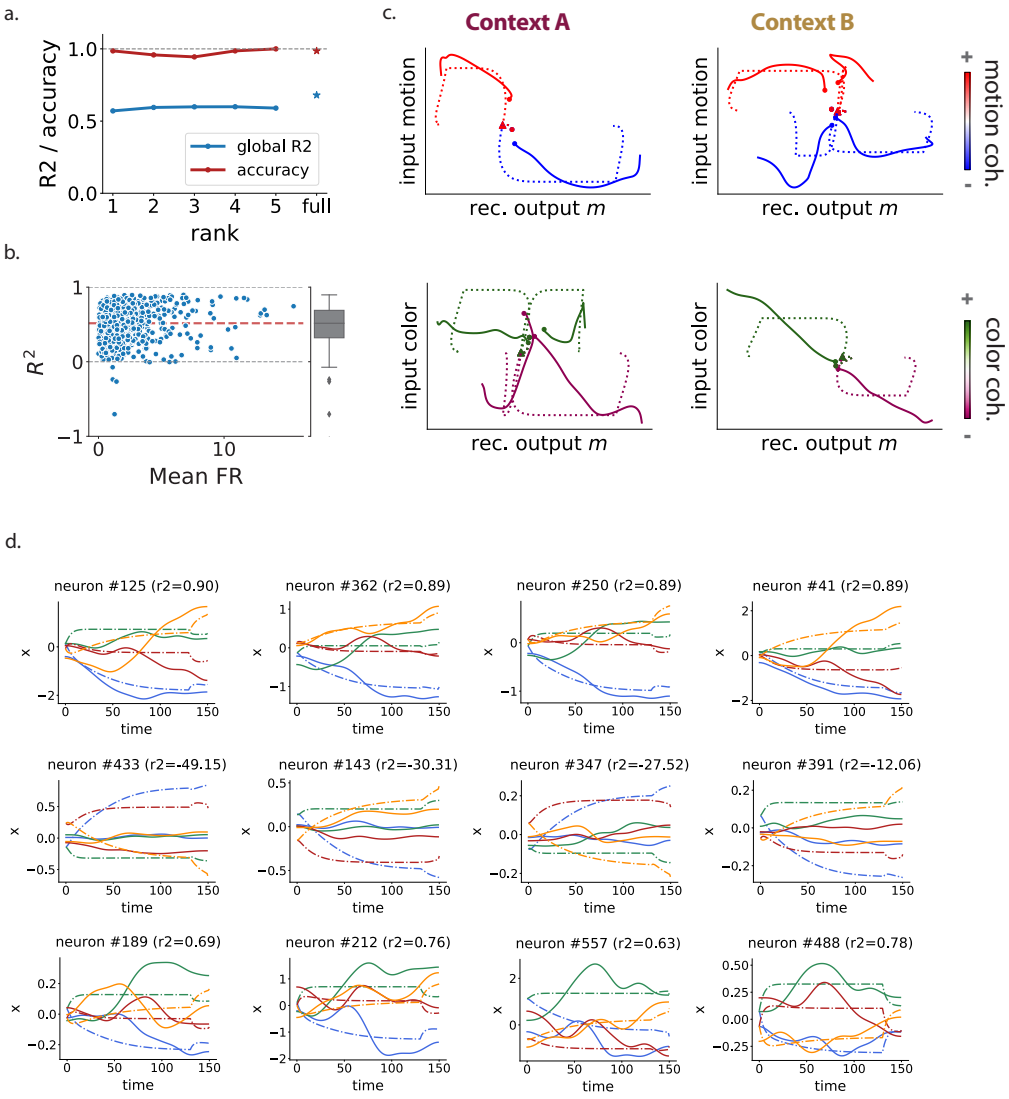


FIGURE C.5: LINT applied to electrophysiological recordings of a second monkey (monkey F) performing the same task. a-c. Same as Fig. V.5 for monkey F data. For panel b, 13 neurons for which $R^2 < -1$ do not appear, all having a mean firing rate of less than 3.8 Hz. d. Same as Sup. Fig. C.4 for monkey F, with the 4 best fitted neurons, the 4 worse fitted neurons, and 4 randomly selected neurons.

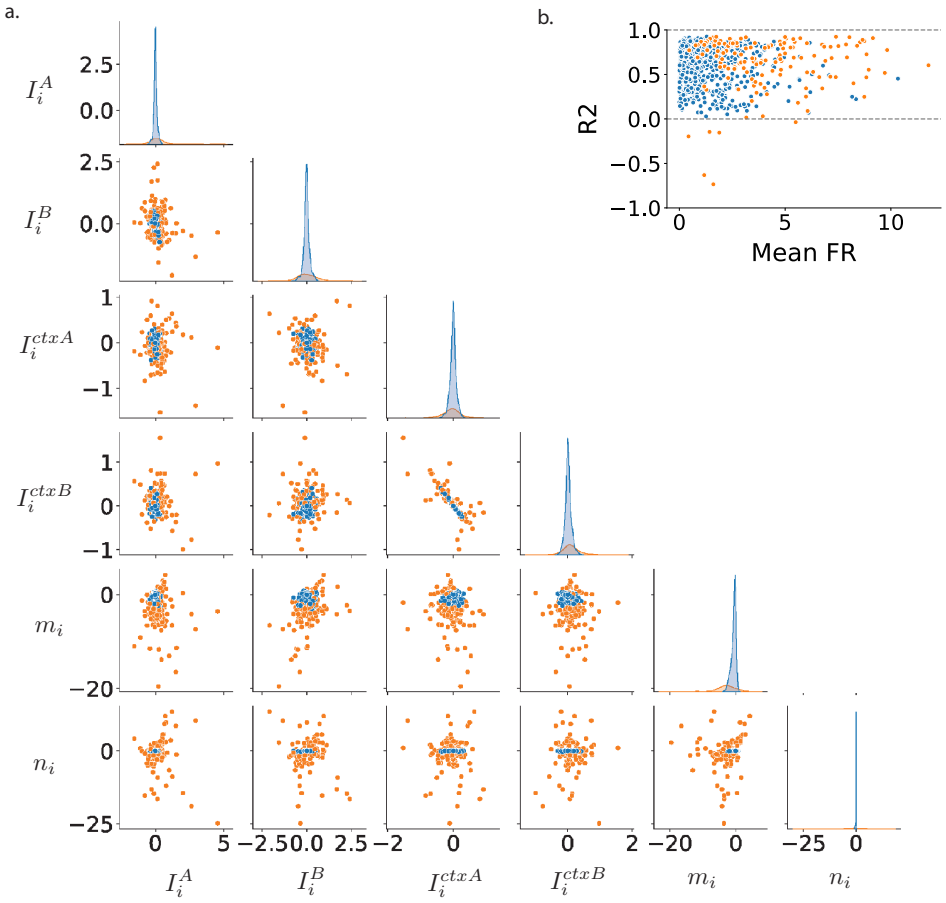


Figure C.6: Distribution of learned connectivity parameters for monkey A. a. Full six-dimensional distribution of the inferred weights on the input and recurrent connectivity vectors of the rank-one model, plotted through two-dimensional and one-dimensional marginals. GMM clustering can identify two groups of neurons: large-weight neurons (orange) and small-weight neurons (blue). Inactivating the blue population does not affect task performance of the network. b. Same clusters visualized in the mean firing-rate - $R^2$ point cloud (Fig. V.5a).

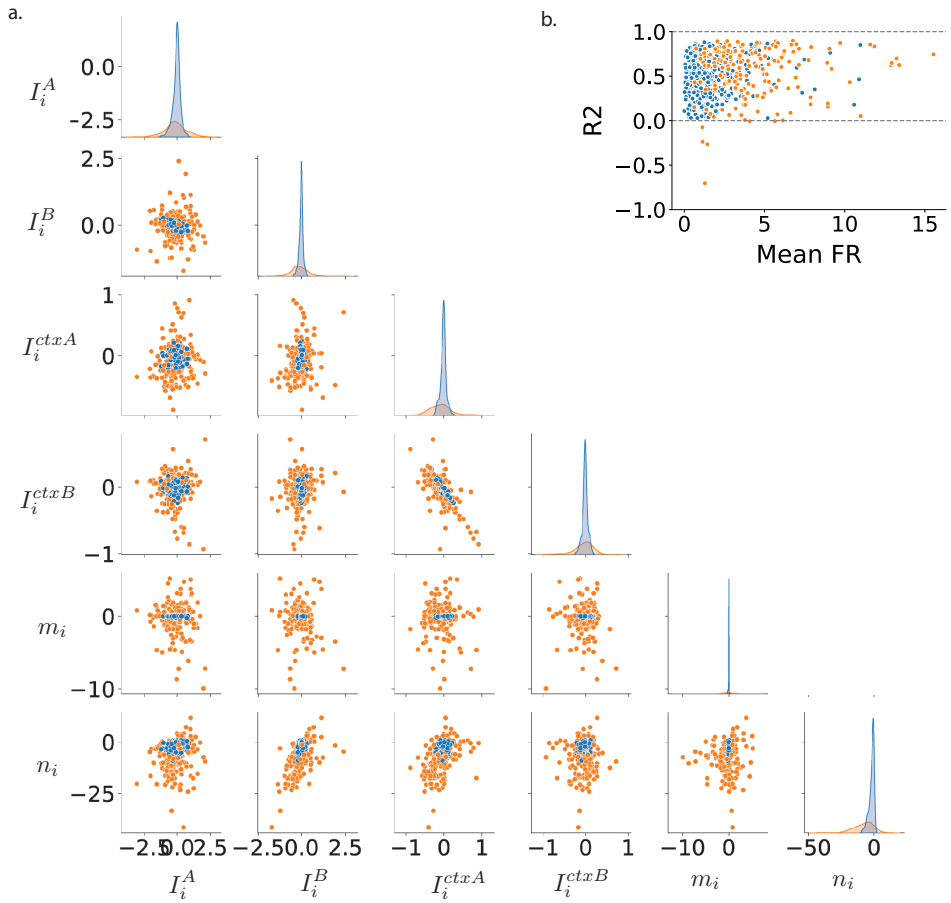


FIGURE C.7: Same as Sup. Fig. C.6

## RÉSUMÉ

À tout instant, des myriades de neurones coopèrent au sein d'un système nerveux, produisant des motifs d'activité collectifs qui forment un substrat biologique pour la perception, la cognition, et le comportement. Les enregistrements in vivo de centaines voire milliers de neurones chez l'animal suggèrent que ces motifs d'activité s'organisent souvent selon des géométries particulières, dites à basse dimensionnalité, à partir desquelles les représentations mentales peuvent être extraites. Ainsi, un paradigme actuel influent en neurosciences postule que les fonctions cognitives émergent à partir des dynamiques de réseaux de neurones dont l'activité forme des motifs de basse dimensionnalité.
Une question essentielle qui demeure est de comprendre comment la structure d'un réseau neuronal génère ces dynamiques particulières de l'activité collective liées à sa fonction. Une voie prometteuse pour éclairer cette question consiste à étudier des réseaux récurrents dont les connexions sont organisées dans une matrice de bas-rang. Il a été montré précédemment que cette propriété mathématique du réseau y induit naturellement des dynamiques de basse dimensionnalité qui encodent des représentations utiles, telles qu'observées dans les expériences sur réseaux biologiques et artificiels. De tels réseaux de bas-rang peuvent-ils alors nous permettre d'ouvrir la « boîte noire » des réseaux récurrents, et de comprendre comment leur fonctionnalité émerge de leur structure ? Et peuvent-ils s'appliquer à des enregistrements de neurones biologiques, afin d'en extraire des hypothèses sur la structure sous-jacente des réseaux observés ? Cette thèse vise à répondre à ces questions, à travers de nouvelles méthodes pour entraîner les réseaux de bas-rang, ainsi qu'à travers une nouvelle théorie qui relie une description statistique de ces réseaux à leurs dynamiques et leur fonction.
Nous allons dans un premier temps exposer notre théorie et nos méthodes, expliquant comment les matrices de bas-rang nous permettent de relier structure et fonction dans un réseau récurrent. Nous allons ensuite étudier des réseaux entraînés à réaliser de nombreuses tâches cognitives, construisant une compréhension systémique de leur fonctionnement. Ces réseaux de bas-rang seront ensuite reliés à une classe de méthodes statistiques communes dans l'interprétation d'enregistrements neuronaux, les systèmes dynamiques latents. Pour terminer, nous démontrerons les capacités des réseaux de bas-rang pour disséquer le fonctionnement de réseaux artificiels récurrents non-contraints, ainsi que pour interpréter des enregistrements corticaux in vivo.

## MOTS CLÉS

réseaux de neurones récurrents, dynamique de réseaux, neurosciences computationnelles, réduction de dimensionalité, codage neuronal

## ABSTRACT

At every instant, myriads of neurons collaborate in a nervous system, generating collective patterns of activity that form a biological substrate for perception, cognition, and behavior. Recent in vivo recordings of hundreds or thousands of neurons in awake animals suggest that this activity patterns form specific geometries, known as low-dimensional, from which mental representations can be extracted. Hence, an influential contemporary paradigm in systems neuroscience posits that neural computations emerge from the collective dynamics of networks of neurons which generate low-dimensional activity patterns visible at the population level.
Understanding how the structure of a network impacts its dynamics and ultimately its function remains an important open question, both for artificial and biological neural systems. A promising direction to illuminate this question is to study recurrent networks whose connectivity is constrained to be low-rank. This particular mathematical property has been previously shown to directly induce low-dimensional dynamics in a network, as those observed in artificial and biological systems. Can thus low-rank networks help us "open the black-box" of recurrent computations? Can they reveal the links between network structure and function? Can they generate hypotheses and insights from neural recordings? This thesis aims at answering those questions, by developing new methods to train low-rank RNNs, and a new theory that links a statistical description of network structure to its dynamics and function.
We will start by exposing our theory and methods, showing how low-rank matrices link structure and computations in a recurrent network. We will then dissect low-rank networks trained to perform a range of cognitive tasks, obtaining mechanistic insights from their trained connectivity. We will then relate the low-rank RNN model class to a set of methods widely used to interpret neural recordings, the latent dynamical system models. Finally, we will demonstrate the capabilities of low-rank RNNs to probe the computational mechanisms in unconstrained, full-rank RNNs, and to interpret in vivo cortical recordings.

## KEYWORDS

recurrent neural networks, network dynamics, computational neuroscience, dimensionality reduction, population coding