Politecnico di Torino

Politecnico di Torino

Ingegneria Matematica

A.a. 2024/2025

Graduation Session Nov. 2025

Using Canonical Polyadic Decomposition of Neural Tensors to Explore the Existence of a Feature-Centric, Rather Than Stimulus-Specific, Learning Signal in the Visual Cortex

Supvs.:

Gianluca Mastrantonio

Pau Vilimelis Aceituno

Benjamin Grewe

Candidate:

Lorenzo Martellone

Acknowledgements

Vorrei innanzitutto ringraziare i miei genitori, Domenico e Federica, che hanno sempre creduto in me e non mi hanno mai fatto mancare affetto, cura e opportunità. Mi hanno dato tutto ciò che era in loro potere, senza limiti né restrizioni, anche quando non condividevano le mie scelte, e con il loro esempio mi hanno trasmesso il valore del sacrificio e della dedizione.

Un ringraziamento speciale va a mia nonna, che è stata per me una seconda mamma e mi ha cresciuto come uno dei suoi figli, donandomi affetto e facendomi sentire, da quando sono nato, protetto e importante. Ci tengo anche a ringraziare tutta la mia famiglia, che ha contribuito, e continua a contribuire a ciò che sono più di qualsiasi altra cosa.

Ringrazio la mia ragazza, Ilaria, per essere una presenza sincera e lucida nella mia vita, oltre che un esempio e un punto di riferimento costante.

A tutti gli amici, sia quelli di sempre sia quelli che ho incontrato lungo questo cammino, quelli ormai lontani e quelli ancora vicini, grazie per aver dato significato ai miei momenti e per essermi stati di supporto con la vostra amicizia innumerevoli volte, spesso senza nemmeno rendervene conto.

Finally, I would like to thank my supervisors, Prof. Gianluca Mastrantonio, for providing clear answers, objective criticism and guidance whenever I needed it, Dr. Pau Vilimelis Aceituno for his constant support, availability and patience, and Prof. Benjamin Grewe for the opportunity to join his research group and the Institute of Neuroinformatics of ETH Zurich & University of Zurich, where I felt welcomed, learned a lot, and met people whom I will carry with me for the rest of my life.

Table of Contents

List of Figures

Chapter 1

Introduction

Understanding the mechanisms by which the brain learns remains a central unresolved issue in neuroscience. Significant research focuses on elucidating, at the neuronal level, the algorithms that determine which synapses are modified, the extent of these modifications, and the signals that guide these processes. Recent cortical learning models propose biologically plausible routes to backpropagation: dendritic microcircuit schemes that approximate error backpropagation [1], burst–dependent plasticity that coordinates learning across hierarchical circuits [2], and Deep Feedback Control with locally available target signals [3]. These frameworks are powerful but remain primarily theoretical or simulation–based rather than validated at scale on neural population data.

A converging line of experimental research focuses on reactivations, defined as spontaneous or offline recurrences of stimulus-evoked patterns, which are associated with subsequent changes in cortical responses during learning. In the mouse sensory cortex, both the content and frequency of reactivations predict bidirectional network changes and future sensory responses. These findings suggest that reactivation serves as a learning signal [4] [5].

Building on this idea, recent analyses frame cortical learning in terms of *target signals*: population–level results in mouse neocortex argue that data may align better with Target Learning than with backprop–style mechanisms [6], and complementary work similarly evaluates whether reactivation–derived signals generalize across conditions [7]. These studies motivate the central, neuroscientific question addressed in this work: *What is the nature of the underlying drive that shapes how neurons adapt their responses across stimuli?*

Even as experimental and theoretical work advances, comparatively few candidate learning algorithms have been confronted with rich population datasets. The bottleneck is twofold: it is challenging to acquire stable recordings of neuronal populations at scale, and the resulting datasets are structurally rich, requiring models that match their features instead of forcing the data to conform to the model. This motivates multiway models that treat recordings as higher–order objects and let structure emerge directly from repeated trials and temporal organization.

Tensor component analysis provides exactly this kind of interface. Heuristically, the Canonical Polyadic decomposition (CP/PARAFAC) asks the dataset, "which few archetypal *triplets* recur together?"—a characteristic neuron pattern, a within–trial temporal motif, and a cross–trial profile. Each component is one such triplet; the data are explained as a superposition of a small number of these recurring motifs, and separates them into interpretable building blocks, a perspective rooted in multiway chemometrics and now widely codified in [8] [9]. Classic results clarify when such motifs are essentially unique up to trivial scalings [10]. This "archetype triplet" view extends naturally to statistical modeling and prediction via tensor regression [11].

The range of alternative models beyond standard CP is continually evolving. Recently, sliceTCA was introduced to disentangle distinct covariability classes that may co-occur within the same dataset [12]. When one axis varies in length or alignment across measurements, such as trials with different durations or internal time warps, PARAFAC2 addresses this by allowing the factor associated with that axis to change from slice to slice while preserving a shared latent metric. This approach was originally developed to address chromatographic retention time shifts and remains widely used in chemometrics [13].

In sum, modern tensor methods—CP/TCA, sliceTCA, and PARAFAC2—offer a compact, mode–aware language that naturally amalgamates with neural population data, turning recurring multiway structure into interpretable motifs while accommodating realistic irregularities in experimental recordings [8].

Chapter 2

Background

2.1 Motivation for Tensor-Based Representations of Neural Activity

Neural data usually consists of a collection of time series reporting the activity of neurons in specific regions of the brain; each trace is associated with a region of interest (ROI) if the recorded tissue, where one or more neurons can reside. Calcium imaging is an in vivo recording technique performed with a microscope implanted on the animal's skull and pointed to the brain region under study. A genetically encoded calcium indicator (GCaMP) has been inserted into the neuron, resulting in the production of a fluorescent protein that changes its structure and becomes brightly fluorescent when calcium ions bind to it. When a neuron is active, calcium enters the cell, and this influx causes the GCaMP protein to fluoresce, allowing the microscope to detect neuronal activity. After a post-processing procedure, the video recorded from the microscope is converted to calcium traces, which are proportional to the calcium concentration in the ROI; these turn into a proxy for the activity of the neuron circumscribed in the area. The calcium dynamics remain, however, slower in their rise and decay compared to the rapid spikes fired by excited neurons, resulting in smoother and slower signals relative to the timescale of neuronal activity.

A study usually includes multiple trials, in which the same animal is observed and its cells' activity is recorded; each trial results in a matrix [Neurons $\times$ Samples], and the phenomena of interest such as learning and adaptation are reflected by the evolution of the neural activity over trials, which naturally leads to further storing the data in a tensor [Neurons $\times$ Samples $\times$ Trials]. A this point, the analysis of the data recorded can be conducted in many ways, depending on the research question the neuroscientists are trying to address, but there are some steps that are common to almost any study, such as dimensionality reduction, given

the large amount of neurons Calcium Imaging allows to record (from hundreds to thousands) and flattening of the data, since most algorithms are suited to work with vectorial observations and not the matricial ones eventually associated to trials.

Rotation and loss of physiological interpretability in PCA. Principal Component Analysis (PCA) is one of the most popular dimensionality-reduction techniques used in computational neuroscience, but it comes with a requirement on the data to be 2-dimensional [Features × Observations], therefore forcing the destruction of its original 3-dimensional structure by aggregating observations over the time dimension, resulting in a [Neurons × Trials] matrix. This matrix can then be fed to the algorithm, where neurons are treated as features and trials as repeated observations of the phenomenon under study. This does not happen to be the only disadvantage of PCA.

PCA belongs to a family of techniques known as "low-rank matrix factorizations". One limitation of these methods is that post-multiplying the factor matrix by an invertible transformation does not alter the reconstruction. This leads to rotational indeterminacy, because there are infinitely many equivalent factorizations and possible interpretations of the factors. Principal Component Analysis (PCA) resolves this by enforcing orthogonality on loading vectors. This ensures uncorrelated component scores. Although statistically convenient, this property is not always physiologically meaningful or biologically plausible, since neural responses to distinct stimuli are known to be correlated across neurons.

Because of the orthogonality constraint, PCA often can fail to reflect the reality of neural processes, which typically involve correlated, overlapping activations. For instance, a group of neurons may respond to multiple visual stimuli, indicating that the true underlying activity axes in the neural population space are not orthogonal. When PCA is applied to such correlated sources, it looks for orthogonal axes by rotating the original subspace. This approach aims to avoid rotational indeterminacy while maximizing explained variance, but there is no reason to assume that the physiological explainability of the components actually benefits from these constraints. For instance, a counterexample is shown in Figure 2.1 where the retrieved components PC_1 and PC_2 are linear combinations of neural processes, rather than direct representations of them; PC_1 may increase when either Stimulus 1 or 2 related processes activate, and PC_2 reflects their opposition, but none actually accounts for one of them.

A tool to reduce dimensionality without imposing orthogonality across components is the Canonical Polyadic Tensor Decomposition (CP/PARAFAC), which avoids rotational indeterminacy under mild conditions (e.g., Kruskal's condition 2.3.3).

Moreover, representing data as triplets of factors across neurons, time, and trials, and preserving the 3D tensorial structure of the data, allows the model to capture correlated or overlapping patterns, such as different stimulus-evoked responses, as distinct, physiologically meaningful components without the artificial constraints imposed by PCA.

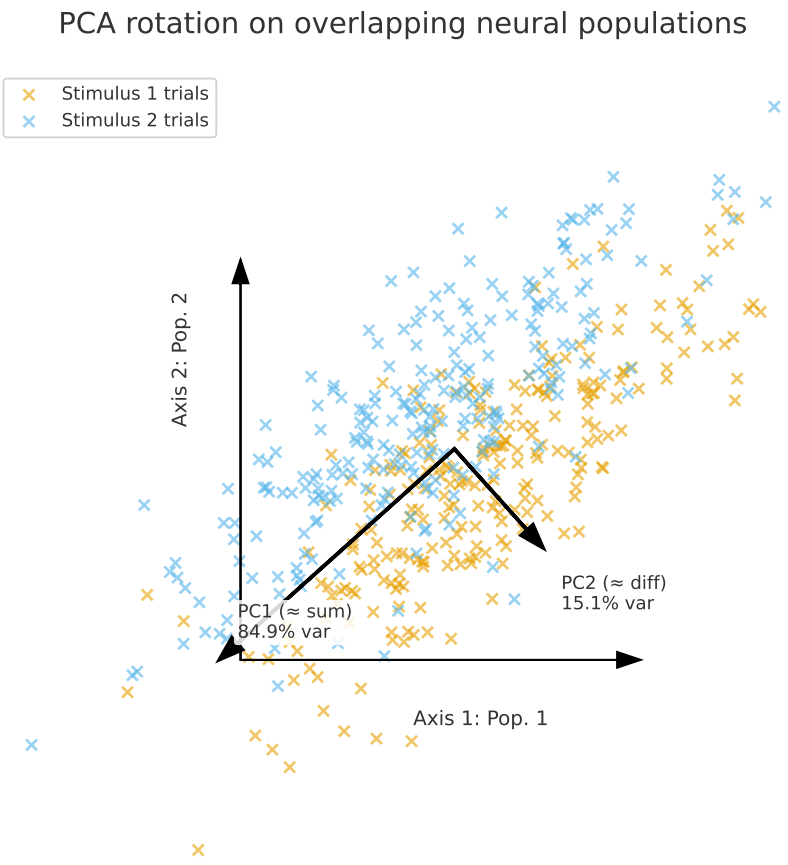


Figure 2.1: Illustration of PCA rotation. Synthetic example showing how PCA enforces orthogonal axes (PC_1, PC_2) on overlapping neuronal subpopulations (Pop. 1, Pop. 2). Because of their overlap, each population responds to both stimuli with different intensities, causing PC_1 to align with the diagonal direction of shared variance while PC_2 captures the orthogonal contrast between them. None of the two components actually correspond to a specific stimulus and thus loses a clear physiological interpretation

2.2 Tensors: preliminaries

A tensor is a multi-dimensional generalization of vectors and matrices that stores data with more than two axes. A vector has one mode (length), a matrix has two modes (rows × columns), and an order-N or N-ways tensor has N modes (e.g., neurons × time × trials).

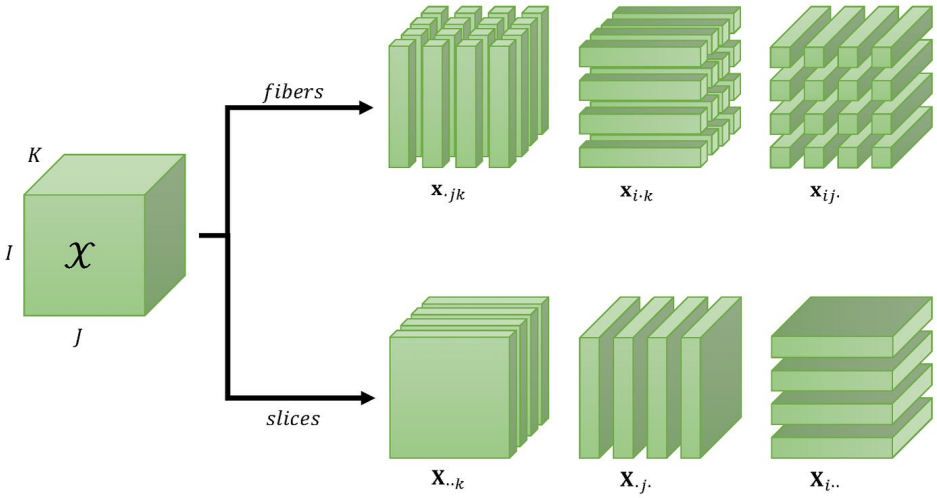


Figure 2.2: Fibers and Slices. Entries of an order-3 tensor $\mathcal{X}$ grouped as fibers (vectors) or slices (matrices).

2.2.1 Notation and basic objects

The following notation is used. Scalars are lowercase letters ($a \in \mathbb{R}$). Vectors are bold lowercase ($\mathbf{a} \in \mathbb{R}^I$). Matrices are bold uppercase ($\mathbf{A} \in \mathbb{R}^{I \times J}$). Higher–order arrays are tensors, written with calligraphic letters ($\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$). The entry at $(i_1, \ldots, i_N)$ is $x_{i_1 \ldots i_N}$.

Fibers and slices. A *mode-n fiber* is a vector obtained fixing all indices except the n-th. A *slice* is the matrix resulting from fixing all but two indices. These objects are useful building blocks for organizing data in tensor form (Figure 2.2).

Unfoldings (matricizations). The *mode-n unfolding* of $\mathcal{X}$, denoted $\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times \prod_{m \neq n} I_m}$, rearranges the entries of $\mathcal{X}$ into a matrix by stacking all mode-n fibers as columns according to a consistent ordering convention that can be kept abstract to remain implementation-agnostic, as long as is respected in algorithms and proofs.

Vectorization. We use vec$(\cdot)$ to stack the columns of a matrix into a vector. For tensors, we first unfold the tensor along a chosen mode and then apply vectorization.

Basic products. The basic products defined for tensors and relevant to this work are:

- **Outer product** of vectors, e.g. $\mathbf{a} \circ \mathbf{b} \circ \mathbf{c}$, makes a rank-1 tensor with entries $(\mathbf{a} \circ \mathbf{b} \circ \mathbf{c})_{ijk} = a_i b_j c_k$.
- **Hadamard product** $\mathbf{A} * \mathbf{B}$ is elementwise multiplication.
- **Kronecker product.** For $\mathbf{A} \in \mathbb{R}^{p \times q}$ and $\mathbf{B} \in \mathbb{R}^{r \times s}$, the Kronecker product $\mathbf{A} \otimes \mathbf{B} \in \mathbb{R}^{(pr) \times (qs)}$ is the block matrix

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & \cdots & a_{1q}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{p1}\mathbf{B} & \cdots & a_{pq}\mathbf{B} \end{bmatrix}.$$

- **Khatri–Rao product (columnwise Kronecker).** If $\mathbf{A} \in \mathbb{R}^{I \times R}$ and $\mathbf{B} \in \mathbb{R}^{J \times R}$ share the same number of columns, their Khatri–Rao product is

$$\mathbf{A} \odot \mathbf{B} \;=\; \begin{bmatrix} \mathbf{a}_1 \otimes \mathbf{b}_1 & \mathbf{a}_2 \otimes \mathbf{b}_2 & \cdots & \mathbf{a}_R \otimes \mathbf{b}_R \end{bmatrix} \in \mathbb{R}^{(IJ) \times R},$$

  i.e., each column is the Kronecker product of the corresponding columns. A useful consequence is:

$$(\mathbf{A} \odot \mathbf{B})^\top (\mathbf{A} \odot \mathbf{B}) = (\mathbf{A}^\top \mathbf{A}) * (\mathbf{B}^\top \mathbf{B}),$$

  where $*$ is the Hadamard (elementwise) product.

The n-mode product. For $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and $\mathbf{A} \in \mathbb{R}^{J \times I_n}$, the *$n$-mode product* $\mathcal{Y} = \mathcal{X} \times_n \mathbf{A}$ is a tensor in $\mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N}$ with

$$y_{i_1 \ldots i_{n-1} j \, i_{n+1} \ldots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 \ldots i_N} \, a_{j i_n}.$$

If applied to unfoldings results in the identities

$$\mathbf{Y}_{(n)} = \mathbf{A}\,\mathbf{X}_{(n)}, \qquad \mathbf{Y}_{(m)} = \mathbf{X}_{(m)} \Big(\mathbf{I} \otimes \cdots \otimes \mathbf{A} \otimes \cdots \otimes \mathbf{I} \Big)^\top \; (m \neq n).$$

2.2.2 Tensor rank and rank-1 tensors

A *rank-1 tensor* in $\mathbb{R}^{I_1 \times \cdots \times I_N}$ is the outer product of N vectors:

$$\mathcal{X} = \mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \circ \cdots \circ \mathbf{a}^{(N)}, \qquad x_{i_1 \ldots i_N} = \prod_{n=1}^{N} a_{i_n}^{(n)}.$$

Notice how this reduces to the definition of *rank-1* matrix in the case of ($\mathcal{X} \in \mathbb{R}^{I_1 \times I_2}$)

The *CP rank* of $\mathcal{X}$ is the smallest integer R such that

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(N)}.$$

An important reformulation consists in collecting the *n-th* mode related vectors $\mathbf{a}_r^{(n)}$ as columns of the matrices $\mathbf{A}^{(n)} = [\mathbf{a}_1^{(n)} \ \cdots \ \mathbf{a}_R^{(n)}] \in \mathbb{R}^{I_n \times R}$, this is implied in the definition of the following form.

Unfolded forms. For each *mode-n unfolding* of $\mathcal{X}$, the CP structure implies

$$\mathbf{X}_{(n)} = \mathbf{A}^{(n)} \Big(\mathbf{A}^{(N)} \odot \cdots \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \cdots \odot \mathbf{A}^{(1)} \Big)^{\top}.$$

where $\odot$ is the Khatri-Rao product.

2.3 Canonical Polyadic (CP/PARAFAC) decomposition

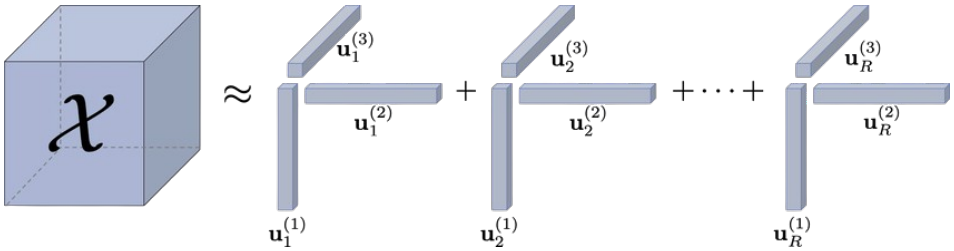


Figure 2.3: CP/PARAFAC schematic. An order-3 tensor $\mathcal{X}$ is approximated as a sum of R rank-1 components: $\mathcal{X} \approx \sum_{r=1}^{R} (\mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \circ \mathbf{u}_r^{(3)})$.

2.3.1 Model definition

The Canonical Polyadic (CP), also called Parallel Factors (PARAFAC), decomposition, writes an order-N tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ as a sum of R rank-1 components:

$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(N)},$$

where $\mathbf{a}_r^{(n)} \in \mathbb{R}^{I_n}$ is the r-th component for mode n. If we stack these vectors as columns $\mathbf{A}^{(n)} = [\mathbf{a}_1^{(n)} \ \cdots \ \mathbf{a}_R^{(n)}]$, the model is fully described by the N factor matrices $\{\mathbf{A}^{(n)}\}_{n=1}^{N}$.

Kolda–Bader double-bracket notation We write $\widehat{\mathcal{X}} = [\![A^{(1)}, \ldots, A^{(N)}]\!]$ to denote the CP model that builds $\widehat{\mathcal{X}}$ from the factor matrices $A^{(n)}$ via a sum of rank-1 outer products.

2.3.2 Reconstruction error

Given a tensor $\mathcal{X}$ and its CP approximation with R components,

$$\widehat{\mathcal{X}} = \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(N)},$$

the *residual* is $\mathcal{E} = \mathcal{X} - \widehat{\mathcal{X}}$. The *reconstruction error* is measured by the Frobenius norm of the residual,

$$\|\mathcal{E}\|_F = \|\mathcal{X} - \widehat{\mathcal{X}}\|_F,$$

i.e., the square root of the sum of squared entries. In standard CP fitting we minimize this quantity (or its square). For comparison across datasets and models is often reported the *relative error*

$$\varepsilon_r = \frac{\|\mathcal{X} - \widehat{\mathcal{X}}\|_F}{\|\mathcal{X}\|_F} \quad \text{and the corresponding fit } 1 - \varepsilon_r,$$

which reaches 100% when the reconstruction is exact (zero residual).

Scaling and permutation. CP has two simple indeterminacies: (i) we can permute the R components in the same way across all modes; (ii) we can rescale columns across modes as long as the product of scalings is 1 (e.g., multiply one column by c in one mode and divide by c in another). In practice, we fix these by *column normalization* (e.g., unit ℓ_2-norm per column) and by sorting components with a clear rule (e.g., by column norm or explained variance).

2.3.3 Kruskal's condition for uniqueness and component identifiability

The property of the Canonical Polyadic (CP/PARAFAC) decomposition to be *unique*, apart from component permutation and column scaling, ensures that the recovered factors correspond to well-defined latent components embedded in data. This property is fundamental for interpreting CP components as actual structures underlying the data. The recovered components are stable and interpretable across runs. The conditions for which uniqueness is verified are mild and discussed in the following.

Kruskal rank and uniqueness. Let $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ be an order-3 tensor s.t.

$$\mathcal{X} = \sum_{r=1}^{R} \lambda_r \, \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r,$$

where $\mathbf{a}_r \in \mathbb{R}^I$, $\mathbf{b}_r \in \mathbb{R}^J$, and $\mathbf{c}_r \in \mathbb{R}^K$ are the factor vectors collected in the matrices $A = [\mathbf{a}_1, \ldots, \mathbf{a}_R]$, $B = [\mathbf{b}_1, \ldots, \mathbf{b}_R]$, and $C = [\mathbf{c}_1, \ldots, \mathbf{c}_R]$.

The *Kruskal rank* (or *k-rank*) of a matrix A, denoted k_A, is defined as the largest integer such that every subset of k_A columns of A is linearly independent.

Kruskal proved the following inequality to be a sufficient condition for the CP decomposition to be *essentially unique* [10]— up to permutation and scaling of the components:

$$k_A + k_B + k_C \;\geq\; 2R + 2.$$

Interpretation in practice. Kruskal's inequality cannot be verified on the true factors, so it represents a theoretical guarantee of identifiability, not a condition to verify on the raw tensor. In applications, one can rely on Kruskal-rank estimates or on stability checks, such as Multi-start reproducibility and other post-fit diagnostics, to support identifiability of the components retrieved.

2.4 From Gaussian Noise modeling to ALS for CP

In this section we derive Alternating Least Squares (ALS) from a statistical model in which the observed tensor is a low-rank CP signal corrupted by i.i.d. Gaussian noise. With this model, maximizing the likelihood is equivalent to minimizing the squared Frobenius residual; this structure yields the alternating least-squares updates for the factor matrices.

2.4.1 Gaussian Noise assumption

Let $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ be the observed tensor and let its CP be $\widehat{\mathcal{X}} = [\![A^{(1)}, \ldots, A^{(N)}]\!]$. Assume additive white Gaussian noise:

$$x_{i_1 \ldots i_N} = \hat{x}_{i_1 \ldots i_N} + \varepsilon_{i_1 \ldots i_N}, \qquad \varepsilon_{i_1 \ldots i_N} \sim \mathcal{N}(0, \sigma^2) \text{ i.i.d.}$$

Equivalently, each entry is Gaussian around its CP mean and independent across indices:

$$x_{i_1,\ldots,i_N} \mid A^{(1)}, \ldots, A^{(N)}, \sigma^2 \;\sim\; \mathcal{N}\Big(\hat{x}_{i_1,\ldots,i_N},\, \sigma^2\Big) \quad \text{i.i.d. over } (i_1, \ldots, i_N).$$

Likelihood. Independence and Gaussianity imply the joint density factorizes entrywise:

$$L(A^{(1)}, \ldots, A^{(N)}, \sigma^2 \mid \mathcal{X}) = \prod_{i_1,\ldots,i_N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_{i_1 \ldots i_N} - \hat{x}_{i_1 \ldots i_N})^2}{2\sigma^2}\right).$$

Taking logs turns the product into a sum:

$$\log L = -\frac{M}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i_1,\ldots,i_N} (x_{i_1 \ldots i_N} - \hat{x}_{i_1 \ldots i_N})^2,$$

where $M = \prod_{n=1}^{N} I_n$ is the number of observed entries. Thus the negative log-likelihood is

$$-\log L = \frac{M}{2} \log(2\pi\sigma^2) + \frac{1}{2\sigma^2} \sum_{i_1,\ldots,i_N} (x_{i_1 \ldots i_N} - \hat{x}_{i_1 \ldots i_N})^2.$$

For fixed σ^2 minimizing $-\log L$ is equivalent to minimizing:

$$\sum_{i_1,\ldots,i_N} (x_{i_1 \ldots i_N} - \hat{x}_{i_1 \ldots i_N})^2.$$

Maximizing $\log L$ over the factors is therefore *equivalent* to minimizing the squared Frobenius residual:

$$\boxed{\min_{A^{(1)},\ldots,A^{(N)}} \left\|\mathcal{X} - [\![A^{(1)},\ldots,A^{(N)}]\!]\right\|_F^2 \;=\; \min_{\widehat{\mathcal{X}}\in\mathcal{M}_R^{\mathrm{CP}}} \left\|\mathcal{X}-\widehat{\mathcal{X}}\right\|_F^2}$$

for a fixed rank R, where $\|T\|_F^2 = \sum t^2$.

2.4.2 Alternating Least Squares

For simplicity, a 3-way tensor $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$ with rank R is considered. The CP model is

$$\hat{x}_{ijk} \;=\; \sum_{\ell=1}^{R} a_{i\ell}\, b_{j\ell}\, c_{k\ell}, \qquad A\in\mathbb{R}^{I\times R},\; B\in\mathbb{R}^{J\times R},\; C\in\mathbb{R}^{K\times R}.$$

We fit the model by minimizing the *squared* Frobenius residual:

$$F(A,B,C) \;=\; \|\mathcal{X}-\widehat{\mathcal{X}}\|_F^2 \;=\; \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\Bigl(x_{ijk}-\sum_{\ell=1}^{R} a_{i\ell}b_{j\ell}c_{k\ell}\Bigr)^2.$$

This problem is *nonconvex* in (A,B,C) jointly, but becomes *convex* in one factor if the other two are fixed. ALS exploits this: fix two factors and update the third by solving normal equations.

Update for A (fix B and C). Setting $\partial F/\partial a_{i\ell} = 0$ gives

$$\frac{\partial F}{\partial a_{i\ell}} = -2\sum_{j,k}\Bigl(x_{ijk}-\sum_{m=1}^{R} a_{im}b_{jm}c_{km}\Bigr)\, b_{j\ell}c_{k\ell} \;=\; 0 \quad \text{for all } i,\ell,$$

which leads to

$$\sum_{m=1}^{R} a_{im}\;\underbrace{\Bigl(\sum_{j} b_{j\ell}b_{jm}\Bigr)\Bigl(\sum_{k} c_{k\ell}c_{km}\Bigr)}_{=:\,G_{\ell m}} \;=\; \underbrace{\sum_{j,k} x_{ijk}\, b_{j\ell}c_{k\ell}}_{=:\,m_{i\ell}},$$

so that $\mathbf{a}_i G = \mathbf{m}_i$, with $G_{\ell m} = (B^\top B)_{\ell m} \odot (C^\top C)_{\ell m}$ elementwise, and $m_{i\ell} = \sum_{j,k} x_{ijk}b_{j\ell}c_{k\ell}$. Each row $\mathbf{a}_i$ is obtained by solving this linear system, and the process repeats symmetrically for B and C.

Matrix form. Using the mode-1 matricization $X_{(1)} \in \mathbb{R}^{I\times(JK)}$ and the Khatri–Rao product $C \odot B \in \mathbb{R}^{(JK)\times R}$, the update for A compactly reads

$$A \leftarrow X_{(1)}\,(C \odot B)\left[(C^\top C) * (B^\top B)\right]^{-1},$$

where $*$ denotes elementwise (Hadamard) product. Analogous updates hold for the other two modes by cyclic permutation:

$$B \leftarrow X_{(2)}\,(C\odot A)\left[(C^\top C)*(A^\top A)\right]^{-1}, \qquad C \leftarrow X_{(3)}\,(B\odot A)\left[(B^\top B)*(A^\top A)\right]^{-1}.$$

Algorithmic loop. Cycle over A, B and C until a stopping rule (small change in $\|\mathcal{X} - \widehat{\mathcal{X}}\|_F^2$ or max iterations). After each update, normalize columns (to fix scaling indeterminacy) and optionally absorb the norms into component weights λ_ℓ.In short, ALS tackles a *nonconvex* problem by solving three coupled but *convex* least-squares subproblems, one per factor, in alternation.

2.4.3 Rank selection (model order)

Computing the CP rank of a tensor $\mathcal{X}$ of order $N \geq 3$ is NP-hard [14], so choosing the model rank R is a practical decision. Common heuristics include:

- **Explained variance / fit (elbow):** increase R until improvements in fit become small.
- **Cross-validation:** select R with the best performance on held-out entries/slices.
- **Stability across runs:** check if components are consistent over multiple random initializations.
- **Diagnostic checks:** watch for overfactoring (high collinearity, noisy or duplicate components).
- **Residual analysis:** if strong structure remains in the residual, a larger R may be justified.
- **Constraints/regularization:** prefer the smallest R that yields stable, interpretable factors under domain constraints.

We usually report sensitivity to nearby ranks (e.g., $R\pm1$) and choose the lowest R that balances fit, stability, and interpretability.

2.5 Non-negative CP

A wide set of constraints, such as sparsity, smoothness, and orthogonality, can be applied to CP decompositions to encode prior knowledge. In this work, we will impose non-negativity on the factors, as it makes the interpretation of the additive components easier.

2.5.1 Nonnegative CP (NNCP): multiplicative updates

When non-negativity is required or desirable for interpretability, an option is to use multiplicative updates, as they preserve non-negativity by construction.
For a third-order nonnegative CP model, the Lee–Seung–type multiplicative update [15] [16] for the mode-1 factor A is

$$A \leftarrow A * \frac{X_{(1)}\, Z + \varepsilon}{A\,(Z^\top Z) + \varepsilon}, \qquad Z = C \odot B,$$

with a small $\varepsilon > 0$ for numerical stability. The same update is applied cyclically to B and C. This is the nonnegative counterpart of the CP-ALS step described in 2.4.2.

Why multiplicative updates preserve non-negativity. Consider the mode-1 NNCP update applied elementwise with $\varepsilon > 0$. Assume we start from nonnegative factors $A, B, C \geq 0$ and a nonnegative data tensor $X \geq 0$. Then:

- $Z = C \odot B \geq 0$ (columnwise Kronecker of nonnegative matrices).
- The numerator $X_{(1)}Z$ is a product of nonnegative matrices, hence $X_{(1)}Z \geq 0$; adding ε keeps it strictly positive.
- The factor $Z^\top Z$ is positive semidefinite with nonnegative entries (since $Z \geq 0$), so $A(Z^\top Z) \geq 0$; adding ε keeps the denominator strictly positive.

Therefore, the elementwise ratio is nonnegative, and multiplying a nonnegative A by a nonnegative ratio yields an updated A that is still nonnegative. By the same argument, the B and C updates preserve nonnegativity. By induction over iterations, NNCP with multiplicative updates maintains $A, B, C \geq 0$.

2.6 From Logistic Regression to CP-Structured Tensor Logistic Regression

In neural data each trial is a matrix $\mathbf{X}_i \in \mathbb{R}^{I \times J}$ (neurons $\times$ time), whereas the circumstances or outcomes associated to the neural response observed during the trial (the specific stimulus presented to the subject, a task successfully completed or not, etc.) can be seen as labels inherent to the specific trial.

The classification task, consisting of predicting those labels from the neural activity, is called decoding and aims to interpret brain activity, associating it with controllable variables or observable outcomes of the experiment. The ultimate goal is to gain insight into how the brain works, understanding if specific populations of neurons are predictive of certain stimuli presented or behaviors assumed from the animal.

Decoding involves the use of decoders-a term in neuroscience for algorithms that perform classification (and, when appropriate, regression) on neural data. Logistic regression is an algorithm often used for this scope, because it's simple, interpretable, and well-suited to label classification, since it outputs probabilities, not just hard labels. One of the goals of this work is to move beyond flat logistic regression by evaluating decoders that operate directly on tensor representations of the data—an approach used extensively in fMRI [17] and, to our knowledge, applied to calcium imaging only by [18] with their LS-STM method.

2.6.1 Multiclass Logistic Regression

Given features $\mathbf{x}_i \in \mathbb{R}^P$ and label $y_i \in \{1, \dots, C\}$, multiclass logistic regression models class probabilities via

$$\boldsymbol{\eta}_i \;=\; \mathbf{x}_i^\top \mathbf{W} \;+\; \mathbf{b} \in \mathbb{R}^C, \qquad p(y_i = c \,|\, \mathbf{x}_i) \;=\; \frac{\exp(\eta_{i,c})}{\sum_{c'=1}^{C} \exp(\eta_{i,c'})}. \tag{2.1}$$

Here $\mathbf{W} \in \mathbb{R}^{P \times C}$ and $\mathbf{b} \in \mathbb{R}^C$ are learned by minimizing the cross-entropy loss (Sec. 2.6.4).

Why move beyond vectors? This model assumes vector inputs; to achieve this it's required to flatten structured inputs (e.g., neuron $\times$ time matrices) shrinking the information contained in the temporal dynamic to a scalar that usually happens to be the mean, the max, or an other statistic. An alternative could be flattening to $\mathbf{x}_i = \mathrm{vec}(\mathbf{X}_i)$, but this approach ignores separability across modes (neurons vs. time) and leads to learn a dense $\mathbf{W} \in \mathbb{R}^{(IJ) \times C}$ vector; this becomes statistically

and computationally inefficient since the number of parameters grows enormously and can lead to an over-parameterized model.

2.6.2 Tensorial Logistic Regression

For matrix inputs $\mathbf{X}_i \in \mathbb{R}^{I\times J}$, we define per-class weight matrices $\{\mathbf{W}_c\}_{c=1}^{C} \subset \mathbb{R}^{I\times J}$ (analogous to the weight vectors in Sec. 2.6.1). Class scores are computed via the Frobenius inner product,

$$\eta_{i,c} \;=\; \langle \mathbf{W}_c, \mathbf{X}_i \rangle + b_c, \qquad \langle \mathbf{A}, \mathbf{B} \rangle \;:=\; \sum_{i=1}^{I}\sum_{j=1}^{J} A_{ij}B_{ij},$$

which extends entrywise to higher-order tensors. Then, for each sample, the label classification still consists in evaluating the class probabilities:

$$\boldsymbol{\eta}_i \;=\; \begin{bmatrix} \langle \mathbf{W}_1, \mathbf{X}_i \rangle \\ \vdots \\ \langle \mathbf{W}_C, \mathbf{X}_i \rangle \end{bmatrix} + \mathbf{b}, \qquad p(y_i = c \,|\, \mathbf{X}_i) = \mathrm{softmax}(\boldsymbol{\eta}_i)_c \tag{2.2}$$

This preserves the two-mode structure (neuron × time matrices) that is lost once data is adapted to the classic logistic regressor, but still uses $IJ \cdot C$ free parameters in $(\mathbf{W}_c)_{c=1}^{C}$. As with the flattened logistic regressor, this often over-parameterizes the model and leads to overfitting, since the parameters outnumber the data available for reliable estimation; in practice, experiments rarely include more than a few hundred trials, a number influenced from task difficulty, session length, and animal engagement or stress.

Most classical statistical models assume regression coefficients are vectors and are not suitable for high-dimensional regression problems. There are two main disadvantages of employing the traditional regression methods. First, it requires vectorization of multiway data, which can ignore the data's inherent high-dimensional structure, resulting in degraded model performance. On the other hand, a large vector-based model will require a large number of parameters, leading to storage and computational burdens, as well as undesired numerical instability.

To overcome these complications, prior knowledge must be incorporated to simplify this regression problem. The same low rank assumption of the CP decomposition can be formulated on the regression coefficients $(\mathbf{W}_c)_{c=1}^{C}$. In this way, not only can the model parameters be reduced, but also the multidirectional relation between the N-dimensional predictor and response can be explored in a structured way, to improve the model performance. Some commonly used low-rank tensor assumptions are based on the CP, Tucker, Tensor Train, and Tensor Ring decomposition methods [17]. Different assumptions over tensor rank determine different subspace exploration strategies and different predictive capabilities of the regression model.

This work focuses on the CP-structure assumption over the regression coefficients of the classifier applied to neural data, since its characteristics most resemble the actual structure we expect neural data to have.

2.6.3 CP-Structured Tensor Logistic Regression

We rewrite the regression coefficients assuming an underlying R-rank CP structure:

$$\mathbf{W} \;=\; \sum_{r=1}^{R} \mathbf{a}_r \mathbf{b}_r^\top \quad \text{with} \quad \mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_R] \in \mathbb{R}^{I \times R}, \quad \mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_R] \in \mathbb{R}^{J \times R}, \tag{2.3}$$

for each trial $\mathbf{X_i}$ then we can compute R bilinear components $z_{i,r}$ each associated with the $r-th$ spatio-temporal motif

$$z_{i,r} \;=\; \mathbf{a}_r^\top \mathbf{X}_i \mathbf{b}_r \quad \Longrightarrow \quad \mathbf{z}_i \;=\; \Big(z_{i,1}, \ldots, z_{i,R}\Big)^\top \in \mathbb{R}^R. \tag{2.4}$$

The following steps are straightforward: the vector $\mathbf{z}_i$ is first multiplied by the Class-by-Component weights $\mathbf{W}_{\text{class}}$, summed to $\mathbf{b}$, and thenitss entries are used in a standard softmax classifier.

$$\boldsymbol{\eta}_i \;=\; \mathbf{z}_i^\top \mathbf{W}_{\text{class}} \;+\; \mathbf{b}, \qquad \mathbf{W}_{\text{class}} \in \mathbb{R}^{R \times C}. \tag{2.5}$$

Compared to dense matrix weights, the parameter count drops from $IJ \cdot C$ to $IR + JR + RC$ (plus biases). At the same time, each component r is interpretable as a spatio-temporal motif $(\mathbf{a}_r, \mathbf{b}_r)$ with class contributions $\mathbf{W}_{\text{class}}[r, :]$. The use of CP/PARAFAC follows the standard tensor literature [8]; the idea of low-rank coefficient tensors for regression was formalized in [11] and is adapted here to the logistic (classification) loss.

2.6.4 Optimization, Loss, and Regularization

An official implementation of the CP-Structured Logistic Classifier is not available, so we implemented it ad hoc in Python using the PyTorch library. The following equations describe the loss function used to estimate the model parameters. The pseudocode of the training algorithm is also reported.
Components are collected in a 3-way array $W_{ijr} = A_{ir} B_{jr}$, while trials in $\mathcal{X} \in \mathbb{R}^{N \times I \times J}$. Then

$$Z_{nr} \;=\; \sum_{i,j} \mathcal{X}_{nij}\, A_{ir} B_{jr}, \qquad \eta_{nc} \;=\; \sum_{r} Z_{nr}\, (\mathbf{W}_{\text{class}})_{rc} + b_c. \tag{2.6}$$

This is implemented efficiently with tensor contractions (e.g. `einsum`).

With one-hot labels $(y_{i,c})$, probabilities $p_{i,c} = \mathrm{softmax}(\boldsymbol{\eta}_i)_c$, and optional quadratic penalties, we minimize the cross-entropy

$$\mathcal{L}(\mathbf{A}, \mathbf{B}, \mathbf{W}_{\text{class}}, \mathbf{b}) \;=\; -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c} \log p_{i,c} + \lambda_A \|\mathbf{A}\|_F^2 + \lambda_B \|\mathbf{B}\|_F^2 + \lambda_W \|\mathbf{W}_{\text{class}}\|_F^2. \tag{2.7}$$

Parameters are learned by first-order optimization (e.g. Adam) with automatic differentiation. Column normalization and component sorting (by $\|\mathbf{W}_{\text{class}}[r,:]\|_2$ or temporal peak in $\mathbf{b}_r$) mitigate CP scaling/permutation indeterminacies [8].

2.6.5 Training pseudocode

Algorithm 1 Training CP–Structured Tensor Logistic Classifier

Require: $\mathcal{X} \in \mathbb{R}^{N \times I \times J}$, labels $y \in \{1, \ldots, C\}^N$, rank R, epochs T, lr η, weight decays $(\lambda_A, \lambda_B, \lambda_W)$

Initialize $\mathbf{A} \in \mathbb{R}^{I \times R}$, $\mathbf{B} \in \mathbb{R}^{J \times R}$, $\mathbf{W}_{\text{class}} \in \mathbb{R}^{R \times C}$, $\mathbf{b} = \mathbf{0}$
for $t = 1$ **to** T **do**
    $W_{ijr} \leftarrow A_{ir} B_{jr}$
    $Z_{nr} \leftarrow \sum_{i,j} \mathcal{X}_{nij} W_{ijr}$ *(bilinear projections)*
    $\eta_{nc} \leftarrow \sum_r Z_{nr} (\mathbf{W}_{\text{class}})_{rc} + b_c, \quad p_{nc} \leftarrow \mathrm{softmax}(\boldsymbol{\eta}_n)_c$
    $\mathcal{L} \leftarrow -\sum_{n,c} \mathbf{1}[y_n = c] \log p_{nc} \;+\; \lambda_A \|\mathbf{A}\|_F^2 + \lambda_B \|\mathbf{B}\|_F^2 + \lambda_W \|\mathbf{W}_{\text{class}}\|_F^2$
    Update $(\mathbf{A}, \mathbf{B}, \mathbf{W}_{\text{class}}, \mathbf{b})$ with one optimizer step on $\nabla \mathcal{L}$ (e.g. Adam)
end for

Chapter 3

Hypothesizing an Unified Learning Signal

3.1 Biological background

Consolidation of learning occurs through the potentiation and depression of synapses within specific regions of the brain. An immediate parallel arises between biological and artificial neural networks: synapses can be modeled as weighted edges between neurons, with the weight encoding the strength of the pre- to postsynaptic interaction. Given a neuronal network, we can define:

$$\begin{aligned} \mathcal{N}: &\quad \text{set of all neurons in the network} \\ \{i,j\}: &\quad \text{oriented synapse from } j \text{ to } i \\ w_{ij}: &\quad \text{weight associated to synapse } \{i,j\} \\ \mathcal{E}: &\quad \text{set of synapses in the network} \end{aligned}$$

For us, neurons will always belong to $\mathcal{N}$ and synapses to $\mathcal{E}$.

The rules that determine which synaptic weights are modified, and how these changes occur during learning, implicitly define the learning algorithm implemented by the brain.

Reactivations—i.e., the spontaneous re-expression of recent population activity during sleep or quiet wakefulness—are believed to be widely implicated in learning: hippocampal ensembles replay prior experience [19] and replay is coordinated between hippocampus and visual cortex [20], consistently as a reinforcement-like teaching signal [21], providing feedback signals that reinforce or weaken specific synaptic connections.

3.2 Framing Empirical observations into Hebbian learning

Hebbian learning is a broadly used theoretical framework for synaptic plasticity. In its simplest form, the synaptic weight from pre-synaptic neuron j to post-synaptic neuron i changes proportionally to the correlation of their activities:

$$\Delta w_{ij} \;=\; \eta\, x_j\, y_i,$$

where x_j and y_i denote pre- and postsynaptic neurons activity and $\eta > 0$ is the constant learning rate. Closely related—but *supervised*—is the delta (Widrow–Hoff) rule,

$$\Delta w_{ij} \;=\; \eta\, x_j \Big(t_i - y_i\Big),$$

where t_i is a teaching target and $(t_i - y_i)$ is an error term. While this update shares the correlational (pre $\times$ post-side) form of Hebbian learning, it replaces post-synaptic activity y_i with an error signal.

Empirical observation motivating our hypothesis. We previously stated that reactivations are a learning signal. Now we can specify what kind of correction they carry and fit their definition into the delta learning rule. Treating the reactivation r_i^t associated with neuron i at time t as the analogue of the term t_i in the delta learning rule, and assuming the activity of the same neuron at time t, x_i^t, plays the role of y_i, we can reformulate the learning rule as:

$$\Delta w_{ij}^t \;=\; \eta\, x_j^t \Big(r_i^t - x_i^t\Big) \;=\; \eta\, x_j^t \Big((x_i^t + \delta_i^t) - x_i^t\Big) \;=\; \eta\, x_j^t\, \delta_i^t$$

where δ_i^t is the feedback received from the post-synaptic neuron i at time t via the apical dendrite. With this model, given stimuli $S_k,\ k \in \{A, B\}$ presented at time t, to which the neuron's response is $x_i^t(S_k)$ and the associated reactivation is $r_i^t(S_k)$, is reasonable to assume the following proportionality relation

$$x_i^t(S_A) - x_i^{t-1}(S_A) \;\propto\; \delta_i^t(S_A) \;=\; r_i^t(S_A) - x_i^t(S_A)$$

Simply stating that for neuron i, the difference among consecutive stimulus-evoked activities is proportional to the weight update that affected the synapses in between

the two. The proportionality relation was demonstrated on the same dataset analyzed in this work, using a classifier trained on $r_i^t(S_A) - x_i^t(S_A)$ to predict the sign of the change in activity $sign(x_i^t - x_i^{t-1})$. Besides confirming the assumption, the tests led to a striking observation [7]: *reactivations* believed to replay S_k-evoked patterns predict not only the subsequent change in the response to the *evoked* stimulus k, as expected, but also the change in the response to the *second* stimulus being learned in parallel,

$$x_i^{t_2}(S_B) - x_i^{t_1}(S_B) \;\propto\; \delta_i^t(S_A) \;=\; r_i^t(S_A) x_i^t(S_A)$$

Here, the chronological sequence of events is not trivial as before; we have $t_2 > t > t_1$, so the reactivation and the response to S_A at time t happen between the first and second stimulus-evoked response to S_B.

Concretely, the reactivation–response contrast $\delta_i^t(S_A)$ computed from S_A carried predictive power for the *sign* of the trial-to-trial change in evoked activity for both S_A and S_B (and symmetrically for $\delta_i^t(S_B)$). This observation supports a unified learning signal from reactivations—i.e., a neuron-level drive δ_i^t that shapes plasticity across all stimulus-specific inputs to neuron i—and motivates our subsequent hypotheses.

Under this view, synaptic updates obey

$$\Delta w_A \propto x_j^A \, \delta_i^t,$$
$$\Delta w_B \propto x_j^B \, \delta_i^t,$$

so the same scalar δ_i^t imposes the same *sign* of change across both pathways.

3.3 Competing Hypotheses

If as observed, reactivations instantiate a *single* neuron-level learning signal, the weight updates for a given neuron should be *coordinated* across its stimulus-specific synapses of the neuron: the same signal should tend to drive either co-potentiation or co-depression of synapses associated with neurons that are principal contributors to the representations of S_A and S_B.

3.3.1 Independence across inputs

It is trivial to notice that the Hebbian framework does not impose constraints on how the learning process should affect the *set of synaptic weights* incoming to a neuron i

$$\mathcal{W}(i) = \{w_{ij} : j \in \mathcal{P}r(i)\}, \qquad \mathcal{P}r(i) = \{j \in \mathcal{N} : \{i,j\} \in \mathcal{E}\}$$

Where $\mathcal{P}r(i)$ is the set of all pre-synaptic neurons in a synapse having i as the post-synaptic neuron. Each synapse weight can be increased or decreased regardless of what happens to the others, so for a fixed post-synaptic neuron i, potentiation and depression can, theoretically, occur on different inputs j independently.

The cross-stimuli significance of the learning signal, can be translated into an important constraint applied to the learning rule: neurons can't freely decide which dendrites associated to which stimuli potentiate, instead they either potentiate or depress all of the synapses in the same way.

3.3.2 Hypotheses formulation

In the following, the hypotheses that frame our research question are formally derived.

Notation. Let $s \in \{A, B\}$ index two stimulus-specific pathways projecting to the same post-synaptic neuron associated with two different stimuli. Denote by $x > 0$ the activity of a neuron; by w_s the synaptic weight of pathway s, and specifically, by x^s_{pre} its pre-synaptic activity, and by x_{post} the post-synaptic activity after a strictly increasing monotonic and possibly nonlinear activation function $x_{\text{post}} = \phi(\cdot)$. We consider a generic correlational update

$$\Delta w_s \;=\; x^s_{\text{pre}}\, \delta^s_{\text{post}}, \tag{3.1}$$

where δ^s_{post} is a postsynaptic learning drive associated with stimulus s.

$\mathbf{H}_1$ (Single-signal, neuron-level)

Assume the two drives, consequent to the same neuron-level learning signal, carry the same sign—implying a same-direction change for both synapses A and B—and are directly proportional through α,

$$\text{sign}(\delta^A_{\text{post}}) = \text{sign}(\delta^B_{\text{post}}), \qquad \delta^A_{\text{post}} = \alpha\, \delta^B_{\text{post}} \quad (\alpha \in \mathbb{R}^+). \tag{3.2}$$

Substituting into the update rule $\Delta w_s = x^s_{\text{pre}}\, \delta^s_{\text{post}}$ gives

$$\Delta w_A = x^A_{\text{pre}}\, \alpha\, \delta^B_{\text{post}} = \alpha\, \frac{x^A_{\text{pre}}}{x^B_{\text{pre}}}\, \Delta w_B.$$

Collecting $\alpha' = \alpha\, \frac{x^A_{\text{pre}}}{x^B_{\text{pre}}} \in \mathbb{R}^+$ yields $\Delta w_A = \alpha'\, \Delta w_B$.

Effect on post-synaptic activation. A weight update Δw_s changes the input to ϕ as

$$\phi\Big(w_A x^A_{\text{pre}}\Big) \xrightarrow{\Delta w_A} \phi\Big((w_A + \Delta w_A)\, x^A_{\text{pre}}\Big),$$

which induces a change in postsynaptic activity Δx^A_{post}. Let $z_A := w_A x^A_{\text{pre}}$ and $\varepsilon_A := \Delta w_A x^A_{\text{pre}}$. By the mean value theorem (MVT),

$$\Delta x^A_{\text{post}} := \phi(z_A + \varepsilon_A) - \phi(z_A) \tag{3.3}$$
$$= \phi'(\xi_A)\, \varepsilon_A. \tag{3.4}$$

For some $\xi_A \in (z_A, z_A + \varepsilon_A)$.
Analogously, with $z_B := w_B x^B_{\text{pre}}$ and $\varepsilon_B := \Delta w_B x^B_{\text{pre}}$,

$$\Delta x^B_{\text{post}} := \phi(z_B + \varepsilon_B) - \phi(z_B) \tag{3.5}$$
$$= \phi'(\xi_B)\, \varepsilon_B. \tag{3.6}$$

For some $\xi_B \in (z_B, z_B + \varepsilon_B)$.

Induced proportionality across pathways. Using $\Delta w_A = \alpha' \Delta w_B$ in the forms obtained with the MVT leads to

$$\frac{\Delta x^A_{\text{post}}}{\Delta x^B_{\text{post}}} \;=\; \alpha' \;\cdot\; \frac{\phi'\Big(\xi_A\Big)}{\phi'\Big(\xi_B\Big)} \;\cdot\; \frac{x^A_{\text{pre}}}{x^B_{\text{pre}}}.$$

Once defined

$$\alpha'' \;:=\; \alpha' \;\cdot\; \frac{\phi'\Big(\xi_A\Big)}{\phi'\Big(\xi_B\Big)} \;\cdot\; \frac{x^A_{\text{pre}}}{x^B_{\text{pre}}},$$

we obtain $\Delta x^A_{\text{post}} = \alpha'' \Delta x^B_{\text{post}}$. The monotonicity of ϕ together with $\alpha' \in \mathbb{R}^+$ then implies $\alpha'' > 0$ and therefore the two changes share the same sign

$$\text{sign}(\Delta x^A_{\text{post}}) \;=\; \text{sign}(\Delta x^B_{\text{post}}). \tag{3.7}$$

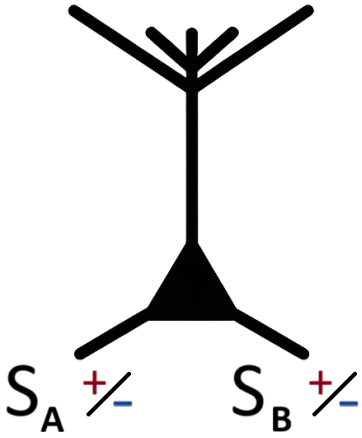


Figure 3.1: Coherent potentiation or depression of synaptic weights on basal dendrites associated with stimulus-specific pathways.

The situation that would result from the just-stated hypothesis being true is graphically summarized for a single post-synaptic neuron in Figure 3.2.

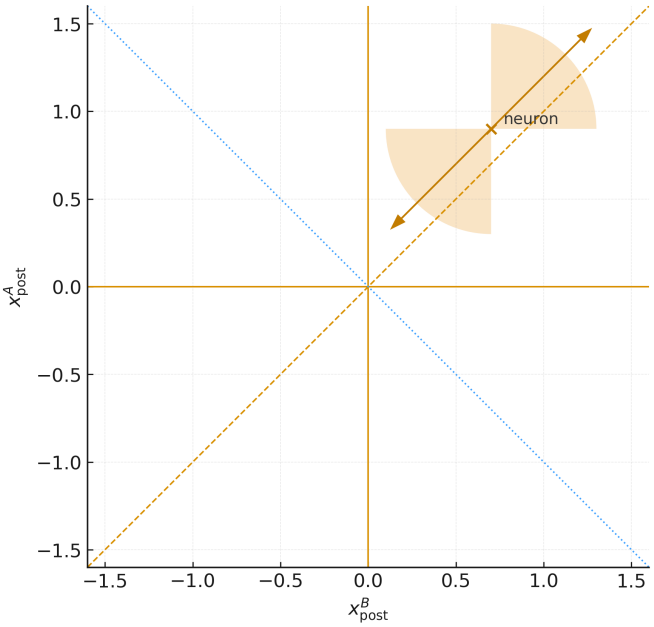


Figure 3.2: How stimulus tuning would possibly result for a neuron under our hypotheses.

Assume we have two stimuli S_k, $k \in \{1, 2\}$; a post-synaptic neuron's activity

x_{post} can be represented as a point in the S_1/S_2 plane, where its position resembles the strength of response to the respective stimulus on the axis, at a specific time instant or trial t_0. During future stimulus presentations, under the constraint

$$\text{sign}(\Delta x^A_{\text{post}}) \;=\; \text{sign}(\Delta x^B_{\text{post}})$$

associated with the hypothesis H_1, the response to the stimuli can either increase or decrease for both, so the response-change vector pointing to the neuron's response location at a successive time instant t_i, $i > 0$ is expected to fall within the orange-shadowed area depicted in Figure 3.2. The two vectors reported in Figure 3.2 represent two possible situations in which the neuron's responsiveness increases or decreases by the same amount over one or more trials, depending on how many occur between t_0 and t_i.

A stimulus response at time t_i resulting in response-change vector aligned to any direction falling in a quadrant bisected by the dashed blue line (II or IV quadrant), would be symptomatic of a tuning process in which the response to a stimulus is potentiated while the other is depressed; a situation inadmissible under H_1.

H_0 (Dual-signal, stimulus-specific)

The most natural and general null hypotheses to pose in contrast to the strong constraint in Eq. 3.7, is that there exist stimulus-specific learning signals $r(S_A)$ and $r(S_B)$ driving the synaptic weights updates Δw_s, implying δ^s_{post} are conditionally independent of each other, and consequently, the variations in the post-synaptic neuron's stimulus-evoked activity across different trials Δx^A_{post} and Δx^B_{post} are not subject to the same-sign constraint expressed in Eq. 3.7. This hypothesis would manifest as the situation illustrated in Figure 3.3: for a given neuron, the change in its response to either S_A or S_B over several trials can take any possible direction, either a stimulus-selective direction (opposite signs across stimuli, NW/SE) or a shared-sign (non-stimulus-specific) direction (same signs across stimuli, NE/SW). The four response-change vectors represent scenarios in which the magnitude of the change is equal across the two stimuli and covers all possible sign combinations.

Weight trajectories under H_0 and H_1. In conclusion, the temporal evolution of the stimulus–specific synaptic weights is summarized in Fig. 3.4. Under the null H_0 (two independent, stimulus–specific drives), the updates to $w_A(t)$ and $w_B(t)$ are independent in sign; a typical outcome is selective potentiation of one pathway with concomitant depotentiation of the other, yielding opposite slopes and even a possible crossing of the two trajectories (panel a). Under the alternative H_1 (a

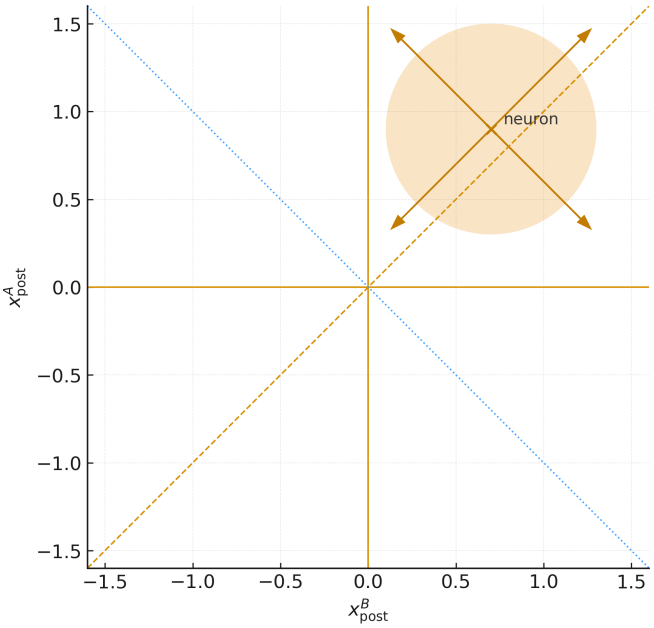
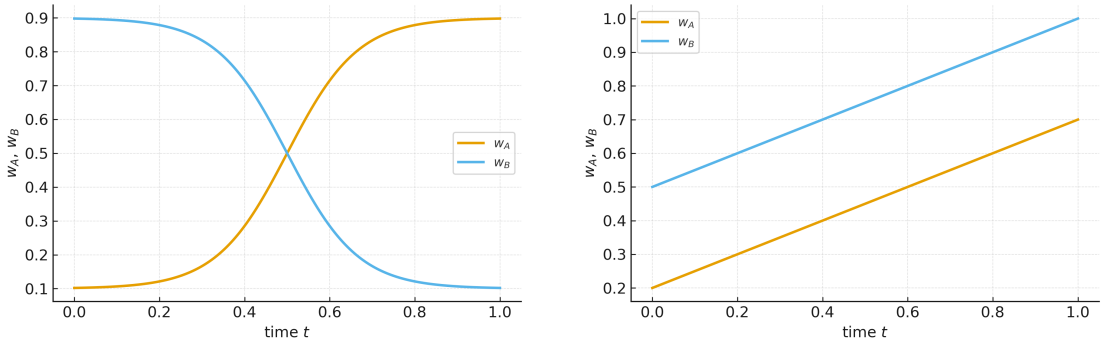


Figure 3.3: Possible stimulus tuning scenarios under the null hypotheses H_0.

single neuron–level learning signal), both weights are driven by the same scalar δ_i^t, so their increments share the sign—both increase or both decrease (panel b).



(a) **(b)**

Figure 3.4: Qualitative plot of the weights trajectories under the two H_0 (a) and H_1 (b).

Chapter 4

Dataset and Methods

4.1 Experimental paradigm and Dataset

We re-analysed the publicly released calcium-imaging dataset from awake, head-fixed mice exposed to repeated visual stimuli, originally collected by Nguyen et al. [5]. Adult Emx1-Cre mice expressed the genetically encoded calcium indicator jGCaMP7s in cortical excitatory neurons. Animals were implanted with a lateral visual-cortex cranial window and imaged with wide-field two-photon microscopy across three planes in layer 2/3, simultaneously sampling ∼6,900 neurons per session across the lateral visual cortical areas.

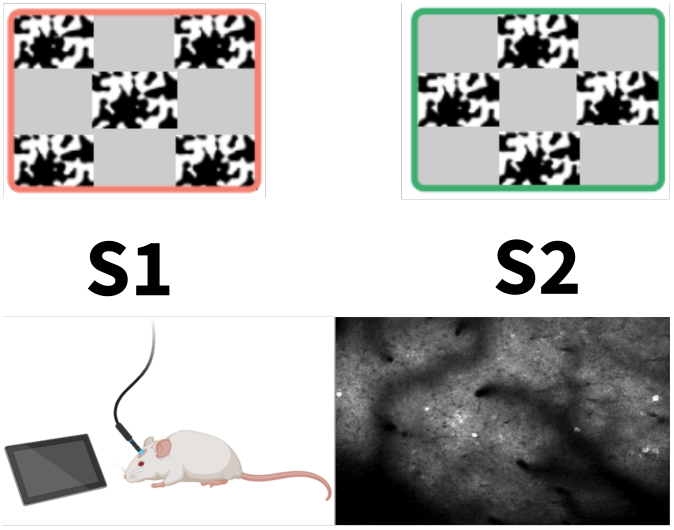
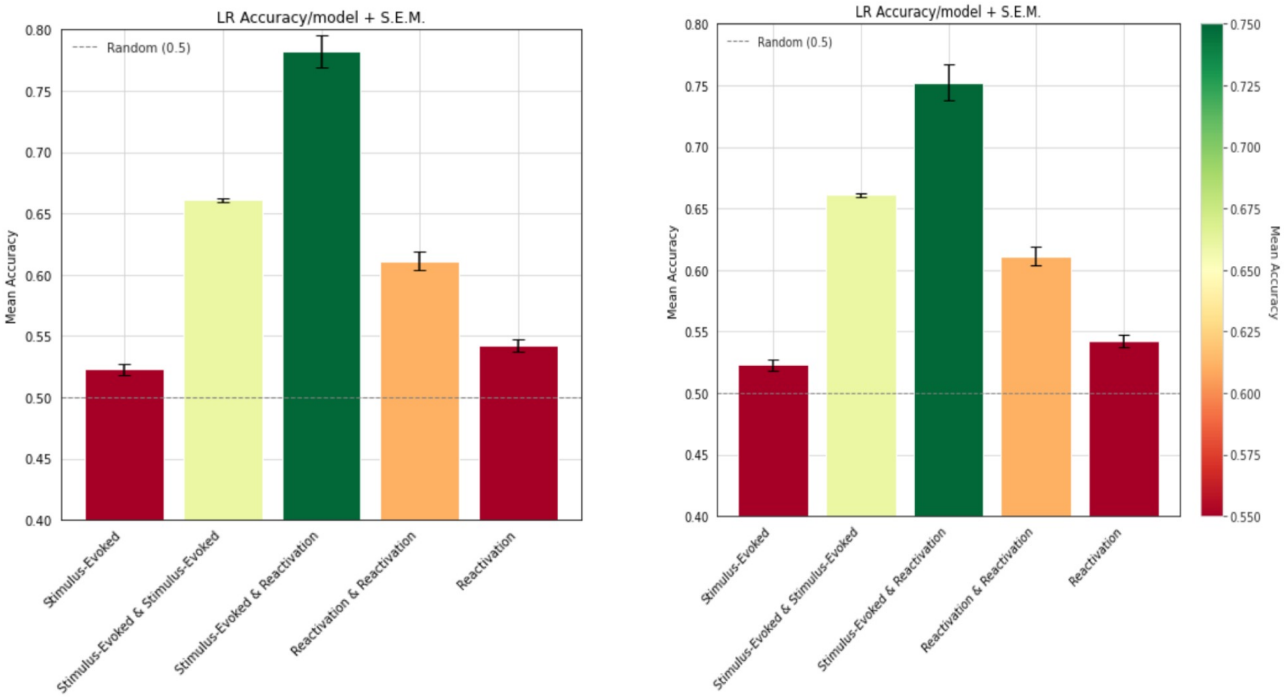


Figure 4.1: Top: The stimuli S1 and S2. Bottom-left: Schematic rendering of the setup. Bottom-right: Example calcium-imaging frame showing some neurons emitting fluorescence.

Each daily session lasted ∼3 h and consisted of a 30 min baseline (grey screen) followed by passive, randomized presentations of two luminance-matched checkerboard patterns (stimulus 1, S1; stimulus 2, S2) Figure 4.1. Individual stimulus epochs lasted 2 s and were separated by a 58 s inter-trial interval (ITI; grey screen), yielding 64 presentations of each stimulus per day (total ∼128 trials). This schedule intentionally inserts long ITIs to capture stimulus-specific *reactivations*—short, synchronous population events in the tens of seconds after each stimulus. Image processing and source extraction were performed using Suite2p for motion correction and region-of-interest (ROI) detection.

In Nguyen et al. [5], analyses use peak-normalized, deconvolved (OASIS algorithm) Ca^{2+} activity for all stimulus-driven neurons. For context, the original study decoded stimulus-specific reactivation content during the ITIs using multinomial logistic regression. In the same deconvolved traces, it has been found [7] that reactivations allowed the prediction of the sign of future changes in responses to the stimulus, independently of whether the stimulus was the same as that for which the reactivations were classified (see Figure 4.2).
In this work, calcium traces rather than deconvolved activity have been used, as they better align with the algorithms used.

Dataset The recordings analyzed in this work covered data from a single mouse over a single day, comprising 128 consecutive trials, imaging at 10.42Hz the activity of ∼5,400 ROIs. After discarding ROIs replicated over the three planes imaged, the number of effective neurons imaged reduced to 5,192 raw calcium traces, composing a matrix [Neurons × Time] with shape [5,192 × 106665] containing the recording performed during ∼30 min of baseline and 128 trials of length 1 min each. With the same shape was also supplied a matrix containing the F^{neuropil}, a collection of calcium traces, one per neuron, recorded in a ring surrounding the neuron itself; the scope this data is to collect the luminescence in the surrounding area of the neuron populated by axons dendrites and possibly other neurons, who's fluorescence that could be overlapping and contaminating the neuron's soma recording, so that it can in a second moment subtracted to the neuron's raw trace using some sort of strategy. A separate behavioral file accompanied the recording, containing frame-by-frame annotations of the experiment. It included the onset and offset frames of each stimulus presentation, the frame marking the end of the baseline period, and other behavioral variables, all synchronized with the calcium-imaging timeline.



(a) **(b)**

Figure 4.2: Accuracy of a logistic classifier decoding the sign (increase/decrease) of future stimulus-evoked activity. (a) Cross-stimulus prediction. Reactivations predict the sign of the next stimulus-evoked response even when the reactivations come from the *opposite* category; the *Stimulus-Evoked & Reactivation* model reaches ~ 0.78 accuracy, exceeding the within-category baseline. **(b) Within-stimulus prediction.** Cross-validated logistic regression decoding the sign of change in the next stimulus-evoked response using different feature sets (bars show mean $\pm$ S.E.M.; dashed line = chance 0.5): (1) *Stimulus-Evoked* — current stimulus-evoked activity; (2) *Stimulus-Evoked & Stimulus-Evoked* — difference between consecutive stimulus-evoked responses; (3) *Stimulus-Evoked & Reactivation* — difference between the current stimulus-evoked response and its intervening reactivation; (4) *Reactivation & Reactivation* — difference between consecutive reactivations; (5) *Reactivation* — reactivation alone. Including reactivations with the current stimulus-evoked response (3) yields the highest accuracy, indicating that reactivations add predictive information about upcoming evoked changes within the same category.

4.2 Preprocessing

A first visual inspection of the data revealed an artifact affecting all the raw calcium traces: five frames, equally spaced and in the same position in every trace, assuming suspiciously low values relative to the rest of the signal. This could be most easily attributed to the microscope recording process, which periodically introduces the artifact (see Figure 4.3). While the first frame was discarded, the remaining four faulty frames were handled in the same way on all of the raw traces, simply with an imputation of the mean between the previous and successive frames.

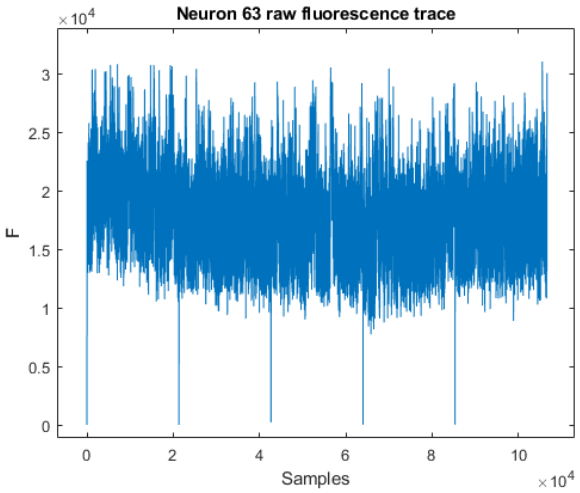


Figure 4.3: Calcium trace of a specific neuron showing the artifact injected from the recording instrumentation

4.2.1 Neuropil correction.

Following standard practice, several pipelines subtract a fraction of the neuropil[1] fluorescence captured in an annular around the ROI,

$$F_i^{\text{corr}}(t) \;=\; F_i(t) - \alpha\, F_i^{\text{neu}}(t).$$

A heuristic originating from earlier two-photon analyses and adopted by default in Suite2p when per-cell estimation is not performed is to set a global value, α, for all

[1]**Neuropil:** the space between neuronal cell bodies that is comprised of dendrites, axons and synapses

neurons ($\alpha = 0.7$ by default), and then compute the neuropil-corrected trace. While simple, a global coefficient can be overly aggressive in cells with little contamination, and too weak where neuropil dominates. Moreover, a fixed–α subtraction often removes part of the signal's continuous component (DC); when contamination is minor this leads to $F_i^{\text{corr}}(t) = F_i(t) - \alpha F_i^{\text{neu}}(t)$ having long negative stretches. This is both biophysically implausible (fluorescence intensities are non-negative) and numerically harmful for downstream $\Delta F/F$, which requires a positive, baseline denominator.

For those reasons, we preferred not to rely on a single global value of α for all cells, but to adopt a per-neuron regression that removes only the neuropil fluctuations while preserving the neuron's DC (offset).

Let $F(t) \equiv F_i(t)$, $N(t) \equiv F_i^{\text{neu}}(t)$ and $\mathcal{B}$ be the $\sim$30 min baseline window at the beginning of the session. We fixed a neuron i and a baseline index set $\mathcal{B} \subset \{1, \ldots, T_{\mathcal{B}}\}$ with size $|\mathcal{B}| = T_{\mathcal{B}}$.

For any constant $\mu \in \mathbb{R}$, we defined the centered regressor $x_\mu(t) = N(t) - \mu$ and the response $y(t) = F(t)$, both restricted to $t \in \mathcal{B}$. We estimate (c, α) by ordinary least squares with an intercept:

$$(c, \alpha) \;=\; \arg\min_{(c,\alpha)} \sum_{t \in \mathcal{B}} \Big(y(t) - c - \alpha\, x_\mu(t)\Big)^2.$$

Write sample means on $\mathcal{B}$ as $\bar{y} = \frac{1}{T_{\mathcal{B}}} \sum_{\mathcal{B}} y(t)$ and $\bar{x}_\mu = \frac{1}{T_{\mathcal{B}}} \sum_{\mathcal{B}} x_\mu(t)$. The normal equations are

$$\sum_{\mathcal{B}} \Big(y - c - \alpha x_\mu\Big) = 0, \qquad \sum_{\mathcal{B}} x_\mu \Big(y - c - \alpha x_\mu\Big) = 0.$$

Solving the first gives $\hat{c} = \bar{y} - \hat{\alpha}\, \bar{x}_\mu$. Substituting into the second yields

$$\hat{\alpha} = \frac{\sum_{\mathcal{B}} (x_\mu - \bar{x}_\mu)\,(y - \bar{y})}{\sum_{\mathcal{B}} (x_\mu - \bar{x}_\mu)^2} = \frac{\mathrm{Cov}_{\mathcal{B}}(x_\mu, y)}{\mathrm{Var}_{\mathcal{B}}(x_\mu)}.$$

Note that $(x_\mu - \bar{x}_\mu) = (N - \bar{N})$ *for any* μ, hence the slope is actually

$$\boxed{\hat{\alpha} = \frac{\sum_{\mathcal{B}} (N - \bar{N})\,(F - \bar{F})}{\sum_{\mathcal{B}} (N - \bar{N})^2} = \frac{\mathrm{Cov}_{\mathcal{B}}(N, F)}{\mathrm{Var}_{\mathcal{B}}(N)}},$$

independent of the chosen centering constant μ (provided $\mathrm{Var}_{\mathcal{B}}(N) > 0$). The intercept then is $\hat{c} = \bar{F} - \hat{\alpha}\,(\bar{N} - \mu)$.

DC preservation. We correct the full trace by removing only the zero-mean neuropil component:

$$F^{\text{corr}}(t) \;=\; F(t) - \hat{\alpha}\,\Big(N(t) - \mu\Big), \qquad t = 1, \ldots, T.$$

Thus, baseline DC is *exactly* preserved iff we set $\mu = \bar{N}$ (the baseline *mean*): $\overline{F^{\mathrm{corr}}}_{\mathcal{B}} = \bar{F}$. We preferred to use the *median* for robustness to outliers

$$\mu_i \;=\; \mathrm{median}\Big\{F_i^{\mathrm{neu}}(t) : t \in \mathcal{B}\Big\},$$

and doing so the identity became approximate; the slope $\hat{\alpha}$ is unchanged, while the intercept shifts by $\hat{\alpha}\,(\bar{N} - \mu)$.

The DC-safe corrected trace is then

$$F_i^{\mathrm{corr}}(t) \;=\; F_i(t) \;-\; \alpha_i' \Big(F_i^{\mathrm{neu}}(t) - \mu_i\Big), \qquad t = 1, \ldots, T.$$

Why this preserves DC. Taking the mean over $\mathcal{B}$,

$$\overline{F_i^{\mathrm{corr}}}_{\mathcal{B}} = \overline{F_i}_{\mathcal{B}} - \alpha_i'\Big(\overline{F_i^{\mathrm{neu}}}_{\mathcal{B}} - \mu_i\Big) = \overline{F_i}_{\mathcal{B}},$$

so the neuron's baseline level is unchanged; only neuropil *fluctuations* around μ_i are removed. In contrast, the naive $F_i - \alpha F_i^{\mathrm{neu}}$ also subtracts neuropil DC and can invert signs or inflate $\Delta F/F$ downstream.

For diagnostics, we monitor residual coupling on the baseline

$$r_i \;=\; \mathrm{Corr}\Big(F_i^{\mathrm{neu}}(\mathcal{B}),\, F_i^{\mathrm{corr}}(\mathcal{B})\Big),$$

and preservation of event power via the demeaned-energy ratio

$$\text{Energy Preserved}_i \;=\; \frac{\sum_t \Big(F_i^{\mathrm{corr}}(t) - \overline{F_i^{\mathrm{corr}}}\Big)^2}{\sum_t \Big(F_i(t) - \overline{F_i}\Big)^2}. \tag{4.1}$$

In Figure 4.4 we report the distribution of $Corr(F_i(t),\, F_i^{\mathrm{neu}}(t))$, over baseline, before any correction.

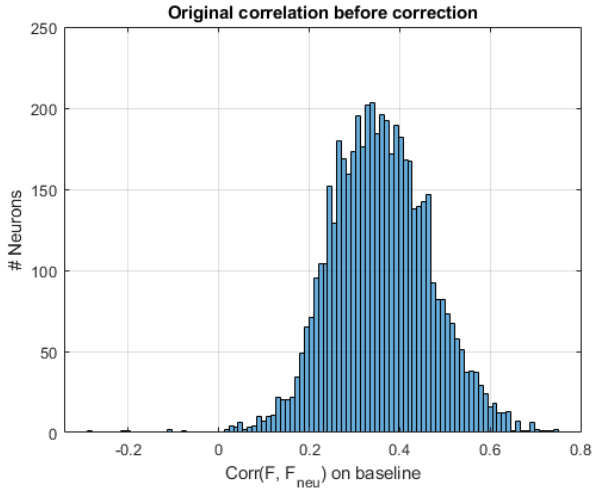


Figure 4.4: Distribution of the correlation between F_i and F_i^{neu} computed over the baseline.

As shown in Figure 4.5, selecting global α often introduces a negative correlation between traces and neuropil fluorescence, an artifact resulting from overly aggressive subtraction.

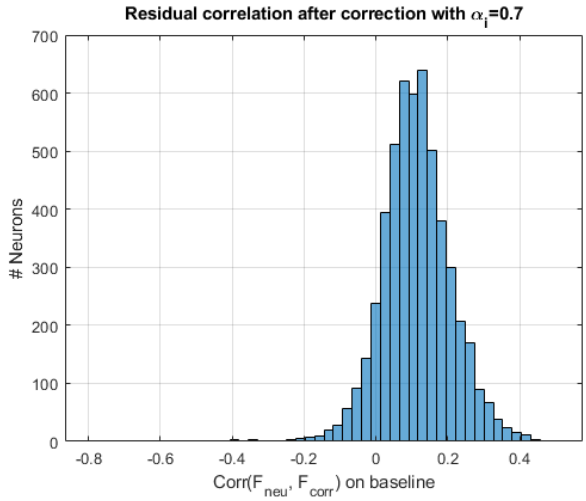
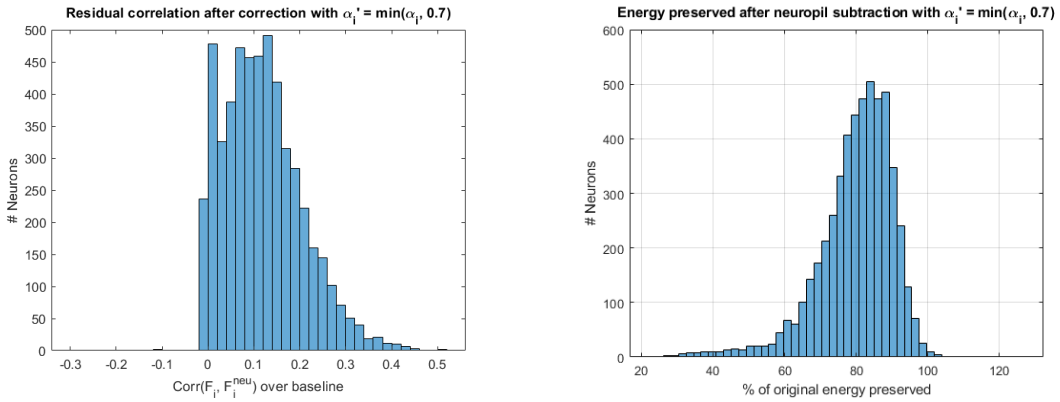


Figure 4.5: Distribution of the correlation between F_i^{corr} and F_i^{neu} computed over the baseline.

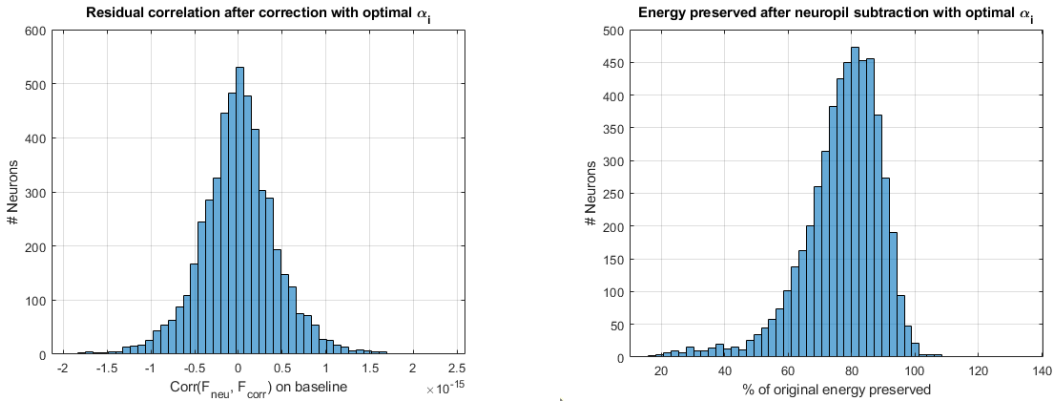


(a) **(b)**

Figure 4.6: Diagnostic plots with $\alpha'_i = \min(\hat{\alpha}_i, 0.7)$.
(a) Distribution of the correlation between F_i^{corr} and F_i^{neu} computed over the baseline.

In Figure 4.6, it is shown that limiting alpha to 0.7 does not introduce negative correlation, yet still leaves most traces correlated with their F^{neu} signal. Figure 4.7 shows that the fully adaptive method successfully makes the signals $F_i^{\text{corr}}(t)$

and $F_i^{\text{neu}}(t)$ orthogonal while avoiding excessive energy loss (4.2.1) compared to the previous solution.

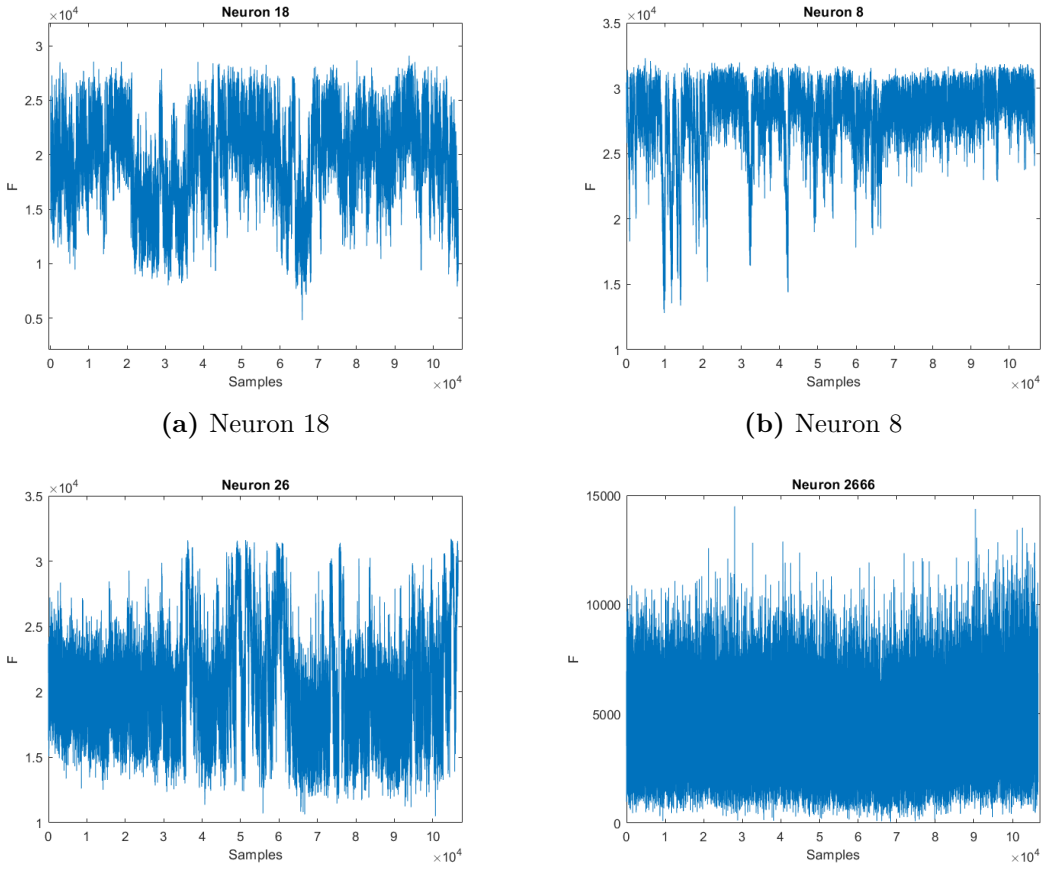


(a) **(b)**

Figure 4.7: Diagnostic plots with $\alpha'_i = \hat{\alpha}_i$.
(a) Distribution of the correlation between F_i^{corr} and F_i^{neu} computed over the baseline. (b) Distribution of the relative energy remaining in the F_i^{corr} traces after the correction

4.2.2 Wavelet-based screening of spurious traces.

After neuropil subtraction, we questioned whether we should have kept all traces or discarded some for quality, and what actually defines the rule for this screening. Given their high number, a visual inspection of the signals was not possible. However, at first glance, we found that many traces were affected by drift and slow-moving artifacts, possibly due to multiple cells captured in the same ROI or to actual ROIs containing only axons and dendrites. Some of those traces are shown in Figure 4.8, along with a healthy trace for comparison. It was evident that the artifacts affecting the traces would mostly manifest as slow drifts, much slower than the actual calcium transients evoked by neuronal activity, leading us to consider screening traces via a multi-resolution wavelet analysis.



(a) Neuron 18 **(b)** Neuron 8

(c) Neuron 26 **(d)** Neuron 2666

Figure 4.8: Calcium traces Panels (a), (b), and (c) display low-quality calcium traces that are either heavily corrupted by noise or do not originate from a single neuron's activity. Panel (d) shows the expected signal shape of an ideal calcium trace, serving as a reference.

As mother wavelet, we used the Daubechies-3 (Db3), a compactly supported wavelet; wavelet denoising is established in two-photon calcium imaging [22] and db3 has been used to denoise neural signals [23].

MODWT and multiresolution analysis. For each trace $x(t)$ we computed the *maximal–overlap discrete wavelet transform* (MODWT) with db3 up to level $J = 16$. The MODWT is a shift-invariant filterbank: at each scale j, the signal is convolved with upsampled, rescaled analysis filters derived from the db3 mother wavelet. Denoting by $\tilde{h}_j$ (high-pass) and $\tilde{g}_j$ (low-pass) the level-j MODWT analysis filters, the detail and scaling coefficient sequences are

$$D_j(t) \;=\; (x * \tilde{h}_j)(t), \qquad A_j(t) \;=\; (x * \tilde{g}_j)(t), \quad j = 1, \ldots, J,$$

where $*$ the discrete convolution.

The associated multiresolution analysis (MRA) provides scale-specific time–domain components by applying the inverse transform, retaining only the coefficients from a given band. We define

$$m_{D_j}(t) \;=\; \mathrm{MODWT}^{-1}\Big(D_j\Big)(t), \qquad m_{A_J}(t) \;=\; \mathrm{MODWT}^{-1}\Big(A_J\Big)(t),$$

so that the components add exactly to the original signal:

$$x(t) \;=\; \sum_{j=1}^{J} m_{D_j}(t) \;+\; m_{A_J}(t).$$

Because the MODWT is a non-orthogonal filter bank, the MRA components are additive in time but not orthogonal in energy: $\sum_j \|m_{D_j}\|_2^2 + \|m_{A_J}\|_2^2$ need not to be equal to $\|x\|_2^2$.

Energy distribution across scales. To quantify how much of a trace lives at each time scale, we computed per–band energies from the MRA components and normalized by the signal energy:

$$E_k \;=\; \sum_t m_k(t)^2, \qquad E_k^{\mathrm{frac}} \;=\; \frac{E_k}{\sum_t x(t)^2}, \qquad k \in \{D_1, \ldots, D_J, A_J\}. \tag{4.2}$$

We found that high–quality calcium traces place most energy in the lower bands, compatible with calcium transients $D_1 \ldots D_4$, while low–quality traces tend to be dominated by very coarse scales, the approximations $D_{\geq 5}$ and A_J. In the trace reported in Figure 4.11, it's evident how the reconstructions from $D_{11} \ldots D_{15}$ mostly account for slow drifts due to noise, and the energy is widely distributed over the coarser detail coefficients; the comparison with the trace reported in Figure

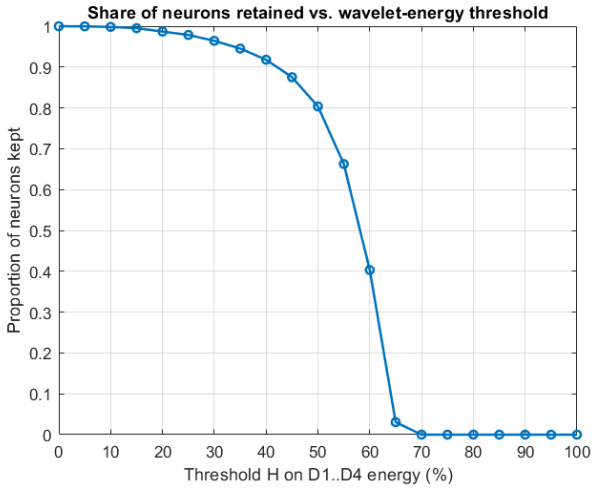


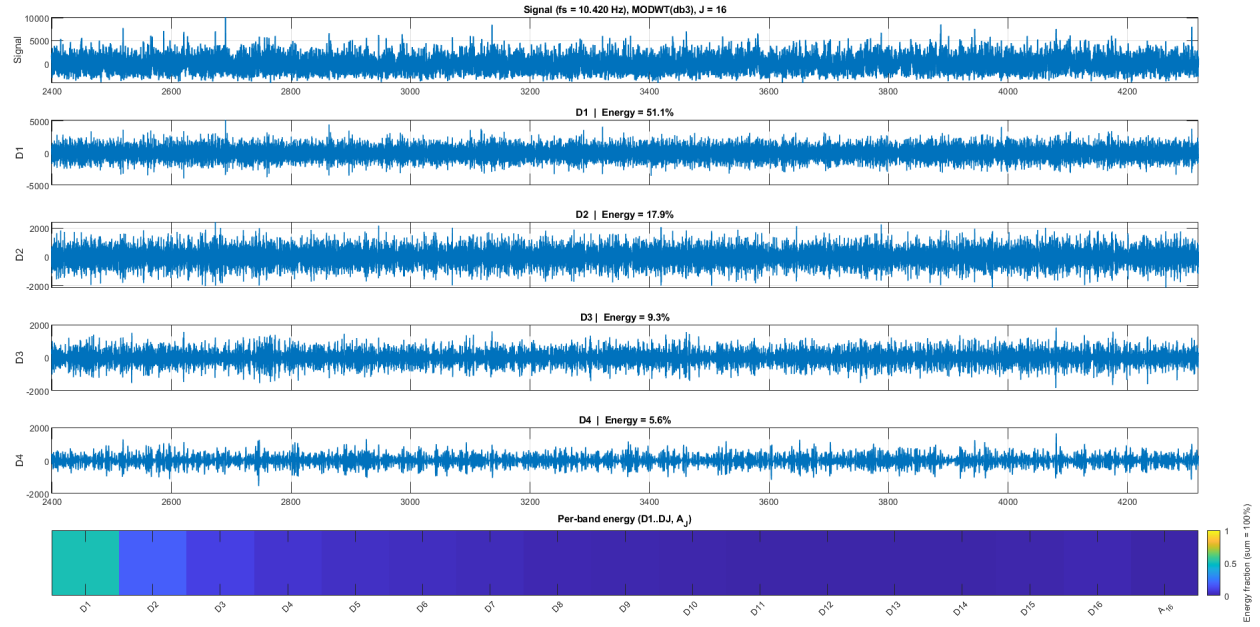
Figure 4.9: Proportion of retained traces vs threshold θ

4.10 underlines how, in a good quality trace, the energy is condensed in the first coefficients $D_1 \dots D_4$ rather than the last ones.

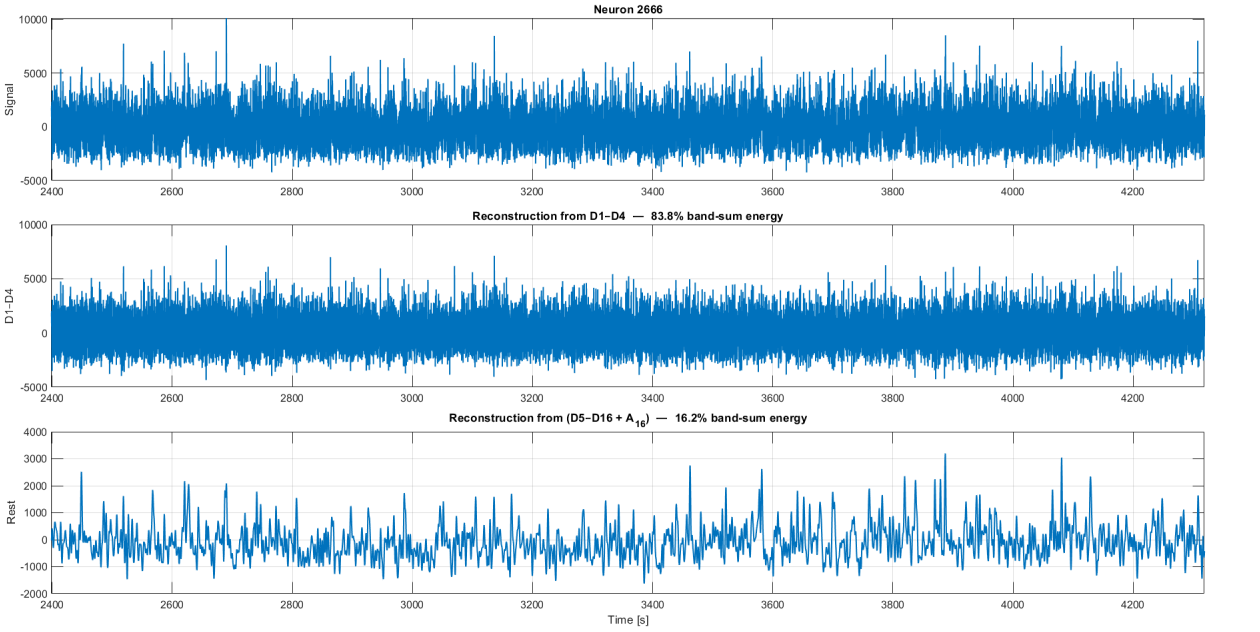
Operationally, we summarized high–frequency dominance with

$$\theta \;=\; \sum_{j=1}^{4} E_{D_j}^{\text{frac}},$$

and used a fixed cutoff (50%) derived from manual inspection together with the curve depicted in Figure 4.9 to reject traces with $\theta \leq 0.5$ as noise–dominated. Traces retained under this criterion exhibit their energy primarily at low-mid levels, consistent with calcium events. This led to discarding $\sim$1000 traces out of the 5192.

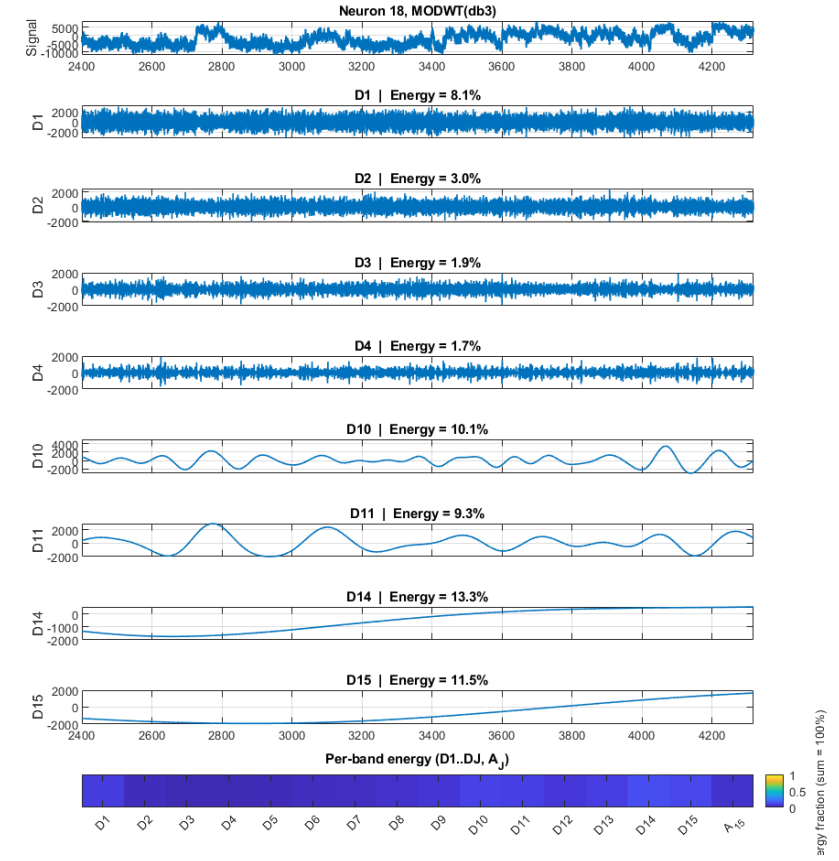


(a)

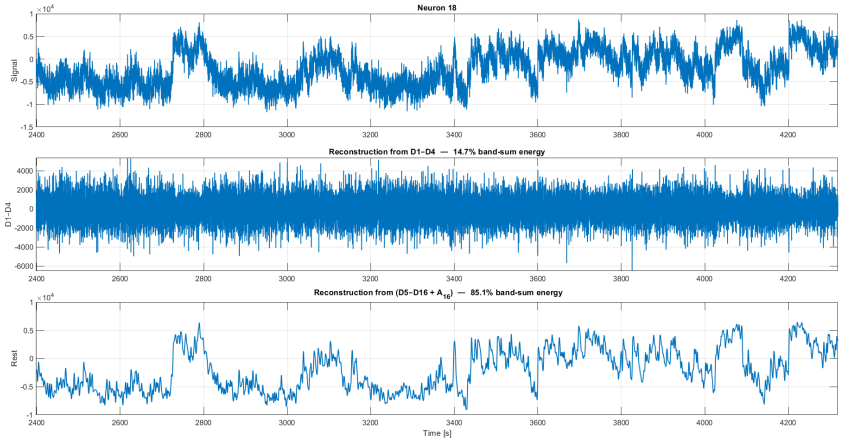


(b)

Figure 4.10: Wavelet decomposition of an ideal calcium trace. (a) The calcium trace is reported with its reconstructions derived from high-detail coefficients; the heatmap at the bottom shows how the energy is distributed over the bands. (b) The same trace is reported with the resulting reconstruction from $D_1 \dots D_4$ and $D_5 \dots A_{16}$.



(a)



(b)

Figure 4.11: Wavelet decomposition of a corrupted trace. (a) A corrupted trace is shown with the reconstructions derived from lower and higher-order coefficients; the heatmap at the bottom shows how the energy is distributed over the bands. (b) The same trace is reported with the resulting reconstruction from $D_1 ... D_4$ and $D_5 ... A_{16}$.

4.2.3 Normalized fluorescence $\frac{\Delta F}{F}$

A standard way to normalize calcium-imaging fluorescence traces is to compute the fractional change relative to a baseline, $\Delta F/F$. Let $F_i \equiv F_i^{corr}$ be the i-th neuron neuropil-corrected trace. For each t, we define

$$\frac{\Delta \mathrm{F}}{\mathrm{F}}_i(t) \;=\; \frac{F(t)_i - F_i^0(t)}{F_i^0(t)}\,, \tag{4.3}$$

where F_0 is a time-varying estimate of the fluorescence baseline of the neuron.

Baseline via rolling lower percentile. To obtain a baseline fluorescence F_0 that is robust to positive transients and can track slow drifts, we first apply a zero-phase Gaussian filter with standard deviation σ (in seconds) to smooth the trace along time,

$$F_s^i \;=\; F_i * g_\sigma, \quad g_\sigma(\tau) \propto \exp\!\Big(-\tfrac{\tau^2}{2\sigma^2}\Big)\,, \tag{4.4}$$

then we use a rolling lower percentile within a sliding window centered at t. Let $W(t)$ denote a temporal window of fixed length w samples around t. For a chosen percentile $q \in [0{,}100]$, we set

$$F_i^0(t) \;=\; \mathrm{perc}_q\Big(\, F_s^i(\tau) \;:\; \tau \in W(t) \,\Big)\,, \tag{4.5}$$

where F_s is smoothed version of F used only for baseline estimation.

Intuitively, (4.5) tracks the baseline by selecting a low quantile within each local window, thereby suppressing upward deflections caused by calcium transients while remains sensitive to slow changes in the baseline possibly due to photo-bleaching or general physiological drift[2] .

Hyperparameters:

- Percentile $q = 20$.
- Window length $w = 60\,\mathrm{s}$.
- Smoothing $sigma = 10\,\mathrm{s}$.

The choice of the window length is motivated by the experimental design; with a trial duration of 1 minute, composed of 2 s of stimulus presentation and 58 s of resting state. A 60-second wide window allows for capturing a large portion of resting activity and for estimating a solid baseline.

[2] **Photo-bleaching** is the gradual, loss of fluorescence as repeated excitation irreversibly deactivates fluorophores, producing a slow downward shift in the baseline even without neural events. **Physiological drift** denotes slow, non–event-related baseline changes (over seconds–minutes) arising from biological and optical factors—e.g., hemodynamics, metabolic state, slight focus creep or motion-induced scattering—that can shift the baseline up or down.

4.3 CP decomposition

We employed CP Tensor decomposition for two primary purposes. The first was to denoise the neural data to recover the underlying neural dynamics embedded in the tensor. The second aim was to retrieve components that could provide insight into the response-consolidation process occurring in the neural population under investigation.

Among the available variants, Non-negative CP decomposition was selected to avoid combinations of positive and negative coefficients arising from distinct components and associated with the same neuron or trial; such results are often counterintuitive and difficult to interpret. The non-negative multiplicative update implementation from [24] was used after each $\Delta F/F$ trace was normalized to the interval [0, 1] using min-max normalization to ensure compatibility with the method. This last step is not harmful, as our analysis is focused on neuron-level changes in stimulus-evoked activity.

The portions of traces corresponding to stimulus presentation, plus an additional half second to account for the extinguishment of calcium transients originating within the stimulus presentation window, were extracted and organized, preserving the chronological order of trials, into a tensor of dimensions [Neurons x Time x Trials] as in Figure 4.12.

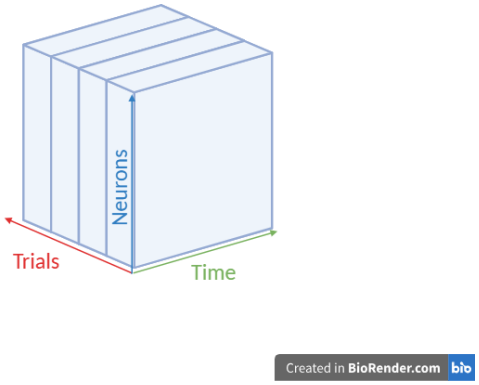


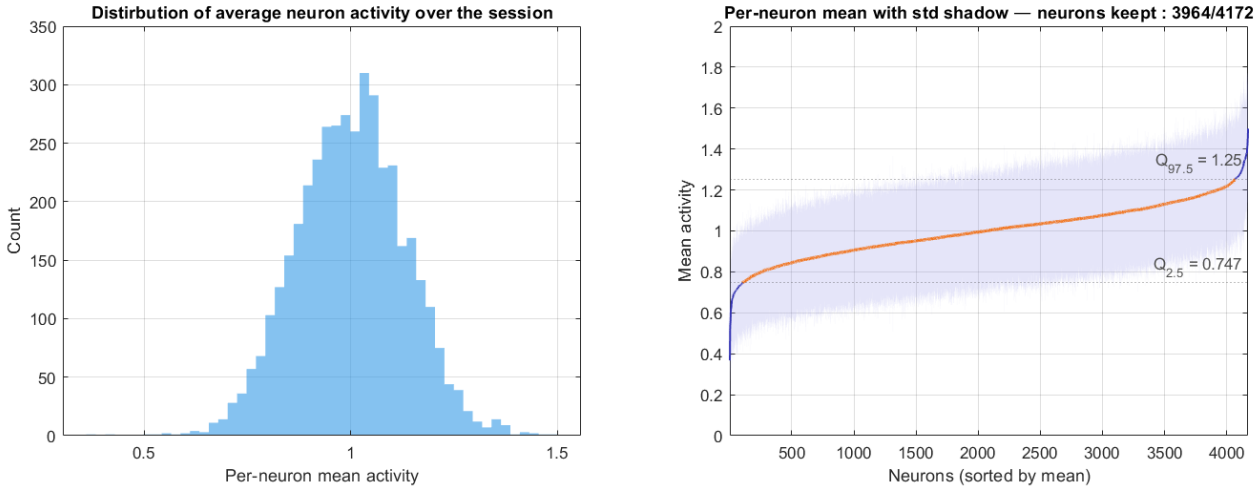
Figure 4.12: Organization of the neural data in a tensor

To mitigate slow trial-wise rundown, we applied a global normalization per trial. For each trial k, all entries were divided by that trial's overall mean across neurons

and time,

$$\tilde{X}_{i,t,k} = \frac{X_{i,t,k}}{\bar{X}_k}, \qquad \bar{X}_k = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T} X_{i,t,k}.$$

This operation removes trial-wide amplitude fluctuations while preserving within-trial structure across neurons and time. Such fluctuations are common in two-photon calcium imaging and can arise from photobleaching of the indicator and gradual physiological adaptation or arousal changes across the session.



(a) **(b)**

Figure 4.13: Per-neuron activity screening and quantile trimming. (a) Histogram of session-averaged activity per neuron. **(b)** Per-neuron mean activity (blue; neurons sorted by mean) with a shaded $\pm$s.d. band; dotted lines mark the quantile thresholds $Q_{2.5} = 0.747$ and $Q_{97.5} = 1.25$. The orange curve highlights neurons retained after quantile trimming to $[Q_{2.5}, Q_{97.5}]$, yielding 3964/4172 neurons ($\approx 95.0\%$) used in subsequent CP analyses.

To avoid the CP/PARAFAC decomposition fit being dominated by a handful of high-amplitude units, we first screened neuron-wise activity across the whole session. As shown in Fig. 4.13 panel (a), the distribution of per-neuron session means is tightly concentrated—with most values lying between 0.8 and 1.2 (arbitrary units after preprocessing)—and homogeneous standard deviations, except a few traces lying at the two extremes of the curve in Fig. 4.13 panel (b). To trim these extremes, we excluded neurons whose session mean fell outside the central quantile interval $[Q_{0.025}, Q_{0.975}]$ (i.e., below the 2.5th or above the 97.5th percentile of the empirical mean distribution). The resulting cleaned distribution is shown by the orange curve in Fig. 4.13 panel (b), and the filtered tensor was used for all subsequent CP analyses.

4.3.1 Rank Selection

The only parameter requiring tuning is the rank of the tensor decomposition, $\mathbf{R}$. Tuning R is challenging because it involves minimizing a loss function, which can lead to distinct local minima for the same number of factors $\mathbf{R}$. Therefore, we performed multiple optimization runs for each $\mathbf{R} \in [2,10]$ optimizing the factors for a maximum of 1000 iterations or untill the condition $|\Delta\text{fit}| < 10^{-7}$ (change in fit per iteration) was verified, to obtain stable results and mitigate the indeterminacy arising from distinct starting points and minima.

Medoid selection across restarts. Given the data tensor $\mathcal{X} \in \mathbb{R}^{N\times T\times K}_{\geq 0}$ (neurons × time × trials), we fit, for each rank $\mathbf{R}$, a family of $S = 50$ CP models by multiple random restarts:

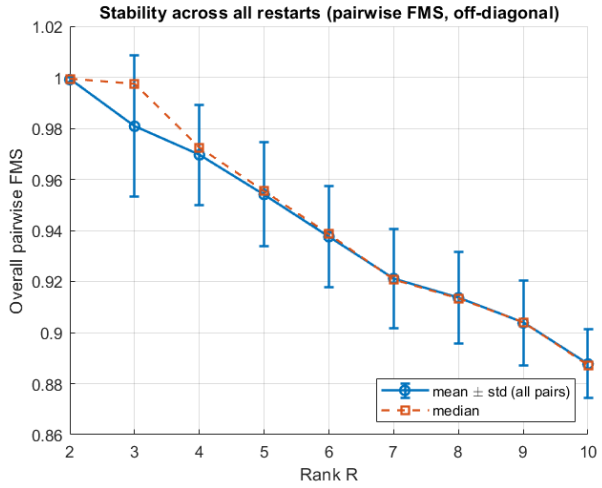
$$\widehat{\mathcal{X}}^{(s)} \;=\; \sum_{r=1}^{R} \lambda_r^{(s)}\, \mathbf{a}_r^{(s)} \circ \mathbf{b}_r^{(s)} \circ \mathbf{c}_r^{(s)}, \qquad s = 1, \ldots, S,$$

where $\lambda_r^{(s)} \geq 0$ and the factor matrices are $A^{(s)} = [\mathbf{a}_1^{(s)} \;\ldots\; \mathbf{a}_R^{(s)}] \in \mathbb{R}^{N\times R}$, $B^{(s)} \in \mathbb{R}^{T\times R}$, $C^{(s)} \in \mathbb{R}^{K\times R}$. To summarize the S solutions with a single representative, we select a *medoid* model $\widehat{\mathcal{X}}^{(s^\star)}$ as the one with maximal similarity to the consensus. As in previous works [25], similarity between two CP models is measured by the *Factor Match Score* (FMS), which aligns components via an optimal permutation π and multiplies per-mode cosine matches:

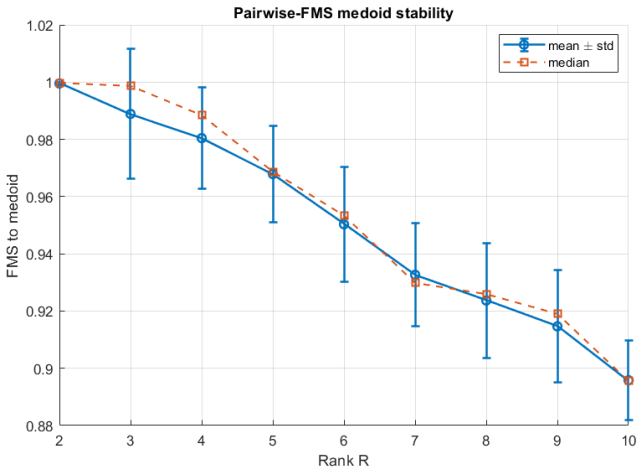
$$\mathrm{FMS}\Big(\mathcal{X}^{(i)}, \mathcal{X}^{(j)}\Big) \;=\; \max_{\pi\in S_R} \frac{1}{R} \sum_{r=1}^{R} \Big(|\langle \tilde{\mathbf{a}}_{\pi(r)}^{(i)}, \tilde{\mathbf{a}}_r^{(j)} \rangle|\, |\langle \tilde{\mathbf{b}}_{\pi(r)}^{(i)}, \tilde{\mathbf{b}}_r^{(j)} \rangle|\, |\langle \tilde{\mathbf{c}}_{\pi(r)}^{(i)}, \tilde{\mathbf{c}}_r^{(j)} \rangle| \Big),$$

where S_R is the permutation group, and tildes denote ℓ_2-normalized columns (e.g. $\tilde{\mathbf{a}} = \mathbf{a}/\|\mathbf{a}\|_2$). We then assembled the similarity matrix $F \in [0,1]^{S\times S}$, $F_{ij} = \mathrm{FMS}(i,j)$, and defined the *medoid* as the restart with the highest average similarity to all others. This exhaustive $O(S^2)$ procedure (here $S = 50$) yields a robust representative model for downstream diagnostics and visualization.

Stability across restarts. We quantified solution stability in two complementary ways. In Figure 4.14 (a), we summarize the *overall pairwise* consistency by computing all off–diagonal Factor Match Scores (FMS) between the S restarts at each rank R and plotting the mean±std with the median overlaid. In panel (b), we report the *FMS to the medoid*: for the run with the highest average similarity to all others. In both plots, scores remain close to 1 at low ranks ([2-4]) and decrease smoothly as R increases, indicating that higher ranks capture less reproducible structure.



(a) Overall stability: mean±std and median of all off–diagonal FMS values across restarts.



(b) Stability to the medoid: mean±std and median of across restarts.

Figure 4.14: Restart stability as a function of CP rank. Both metrics decrease gradually with R, while remaining high enough to indicate consistent recovery of the dominant components.

Diagnostics. The primary diagnostic metric considered is the reconstruction error

$$\mathrm{err}_{\mathrm{rel}} = \frac{\|\mathcal{X} - \widehat{\mathcal{X}}\|_F}{\|\mathcal{X}\|_F}$$

or equivalently, the fraction of variance explained:

$$\mathrm{FVE} = 1 - \mathrm{err}_{\mathrm{rel}}.$$

As shown in the Figure 4.15, the reconstruction error was not particularly informative in this case. Although a reconstruction rate of $\sim 73\%$ is considered good for neural data, this value did not increase significantly with higher ranks R.

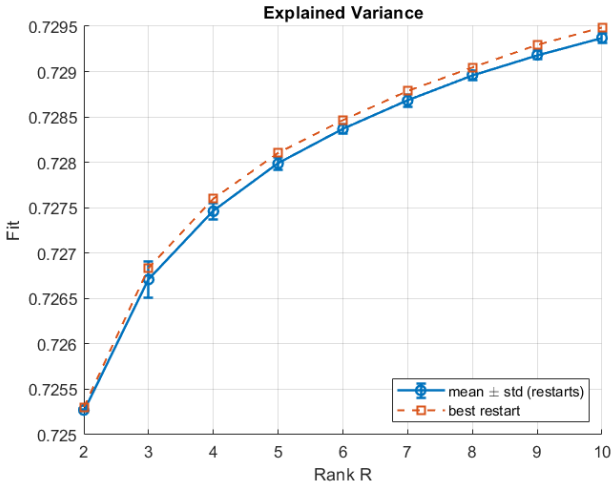


Figure 4.15: Explained variance achieved over the multiple runs for each rank R

This outcome is likely attributable to low intrinsic dimensionality. Because stimulus presentations are brief, the time mode contains only $T = 27$ samples, and responses are concentrated on a few shared temporal motifs. Once these motifs and their trial modulations are captured by a small number of components, increasing the rank primarily redistributes already-explained variance rather than introducing new directions. This often results in splitting neuron groups while reusing similar temporal kernels, which does not reduce the global loss on unseen trials.

This phenomenon is also evident in Figure 4.16, where intra-mode collinearity is reported. Given the medoid factor matrices $U^{(1)} = A^{(s^\star)} \in \mathbb{R}^{N\times R}$, $U^{(2)} = B^{(s^\star)} \in \mathbb{R}^{T\times R}$, $U^{(3)} = C^{(s^\star)} \in \mathbb{R}^{K\times R}$, we quantify how similar the components are *within* a mode $m \in \{1,2,3\}$ by the absolute cosine similarities between normalized columns. Let $\tilde{U}^{(m)} = [\tilde{\mathbf{u}}_1^{(m)} \ \dots \ \tilde{\mathbf{u}}_R^{(m)}]$ with $\tilde{\mathbf{u}}_r^{(m)} = \mathbf{u}_r^{(m)}/\|\mathbf{u}_r^{(m)}\|_2$. Define the off-diagonal cosine matrix

$$C^{(m)} = \left|\tilde{U}^{(m)\top}\tilde{U}^{(m)}\right| \in [0,1]^{R\times R}, \qquad C^{(m)}_{rr} = 0.$$

$$c^{(m)}_{ij} = \left|\langle \tilde{\mathbf{u}}^{(m)}_i, \tilde{\mathbf{u}}^{(m)}_j \rangle\right| \in [0,1], \qquad i \neq j.$$

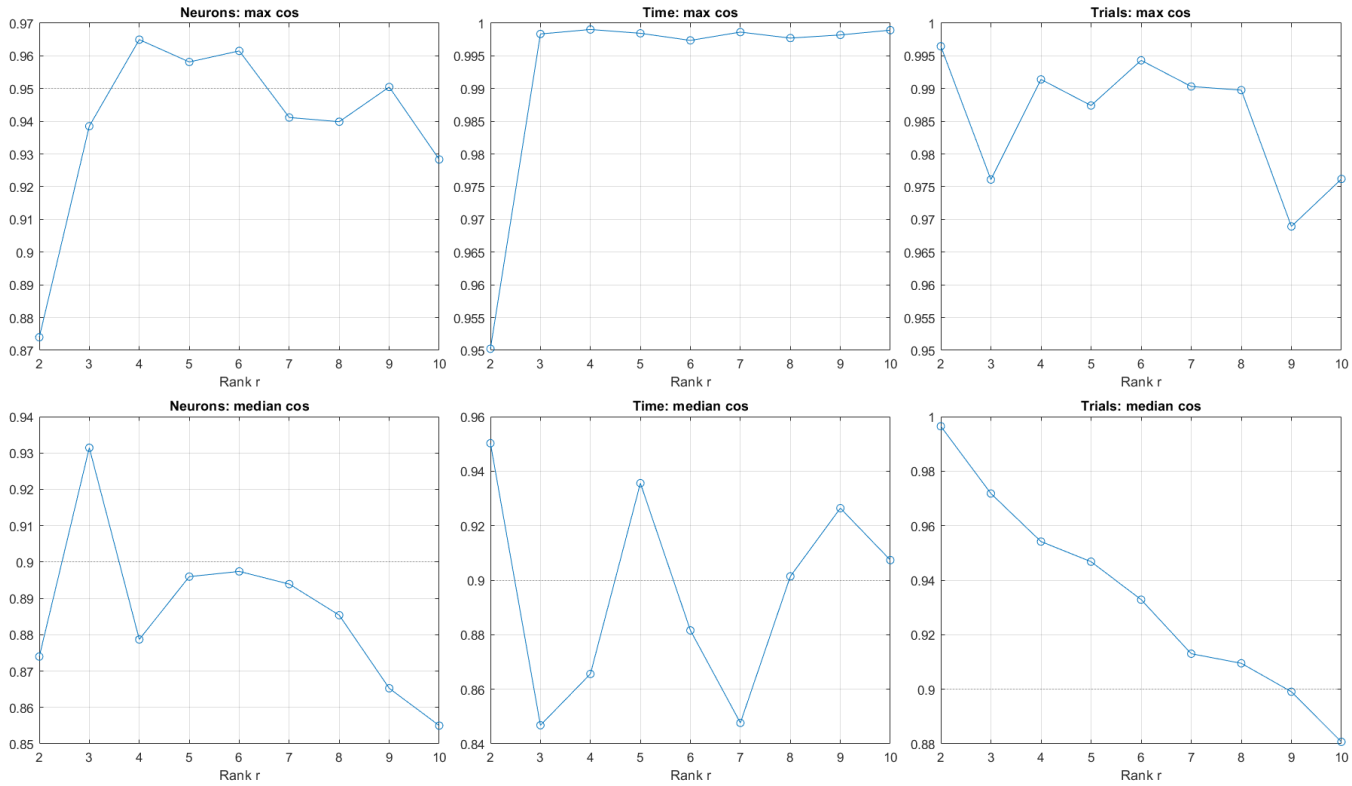


Figure 4.16: Cosine similarity metric between rank-R medoid factors

We report two summary statistics:

$$\mathrm{col}^{(m)}_{\max} \;=\; \max_{r\neq r'} C^{(m)}_{rr'}, \qquad \mathrm{col}^{(m)}_{\mathrm{med}} \;=\; \mathrm{median}\left(\{\, C^{(m)}_{rr'} : r < r' \,\}\right).$$

as R grows, a decreasing $\mathrm{col}^{(m)}_{\mathrm{med}}$ suggests that higher ranks partition the neuron-associated components that tend to become heterogeneous, while elevated values of $\mathrm{col}^{(m)}_{\max}$ indicate almost duplicated temporal components also at higher ranks. This suggests that the temporal motifs identified by the decomposition remain consistent without introducing new temporal kernels, but higher decomposition ranks can help disentangle the activity of distinct populations across the factors, even though the explained variance does not increase.

Hyperparameter tuning conclusions To balance the considerations provided by the reconstruction error—which, given the negligible increase in explained variance at higher ranks, advocates for parsimony in the choice of R—with the results of the collinearity analysis, we made a careful choice. The collinearity analysis indicates a potential benefit in selecting slightly higher ranks than the minimum

inspected, especially when the objective is to disentangle population dynamics as outlined earlier in this chapter. Therefore, we opted for a decomposition rank $\mathbf{R} = 4$. This choice, when considering both neuron and time-mode collinearity, represents the lowest value that ensures optimal performance with respect to the collinearity metrics while also achieving an exhaustive disentanglement of the responses to distinct stimuli across the trials dimension (see Figure 4.18); result that the more parsimonious model with $\mathbf{R} = 2$ fails to achieve, as illustrated in Figure 4.17.

The decompositions shown in Figures 4.17 and 4.18 confirm that the temporal motifs of the neural response are consistently reproduced across distinct models and factors within the same model. These motifs closely resemble the expected stimulus-evoked response of a neuron exhibiting slow calcium transient dynamics. In contrast, the components associated with neurons and trials differ between the first and second models. When the rank is limited to two, the trial components primarily capture simple increasing and decreasing trends in response strength across trials, with these trends distributed relatively homogeneously among neurons, as indicated by the coefficients in the bottom panels of the figure. In Figure 4.18, two additional trial-specific trends emerge when the model is allowed greater flexibility to distribute variance across more components. These new factors, which distinctly separate by stimulus type across trials, clearly reveal the temporal profile of neural responses and are markedly sparser across neurons, as evident from the steeper neuron-loading panels when compared with the other components.

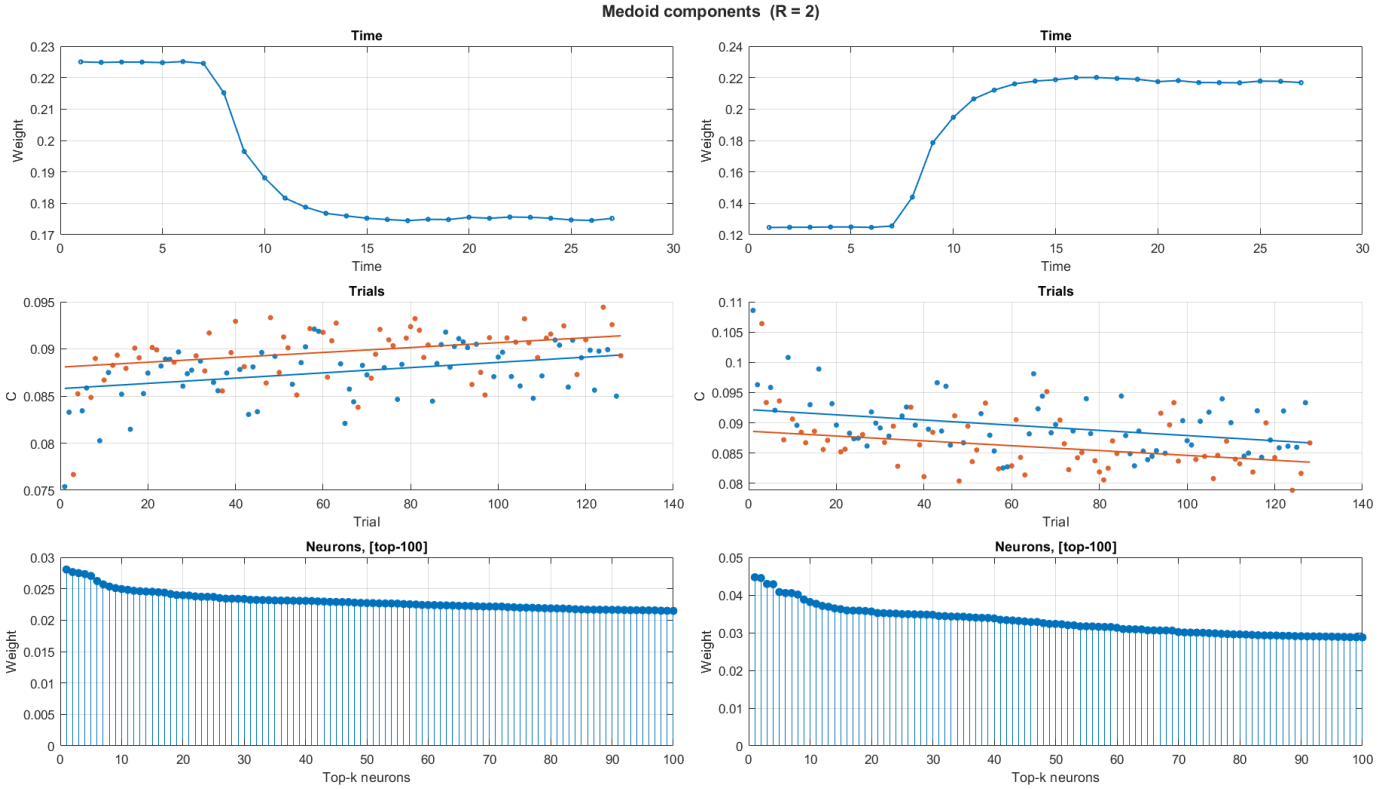


Figure 4.17: Factors resulting from rank $\mathbf{R} = 4$ CP decomposition. The top panels show the temporal profiles from the factors **B** decomposing the time dimension. The central panels report the response strength for each trial as encoded in the **C** factors; distinct colors discern trials by the stimulus presented (S_A or S_B). The bottom panels show the neuron factor **A** reordered by coefficient magnitude; the top 100 entries are displayed.

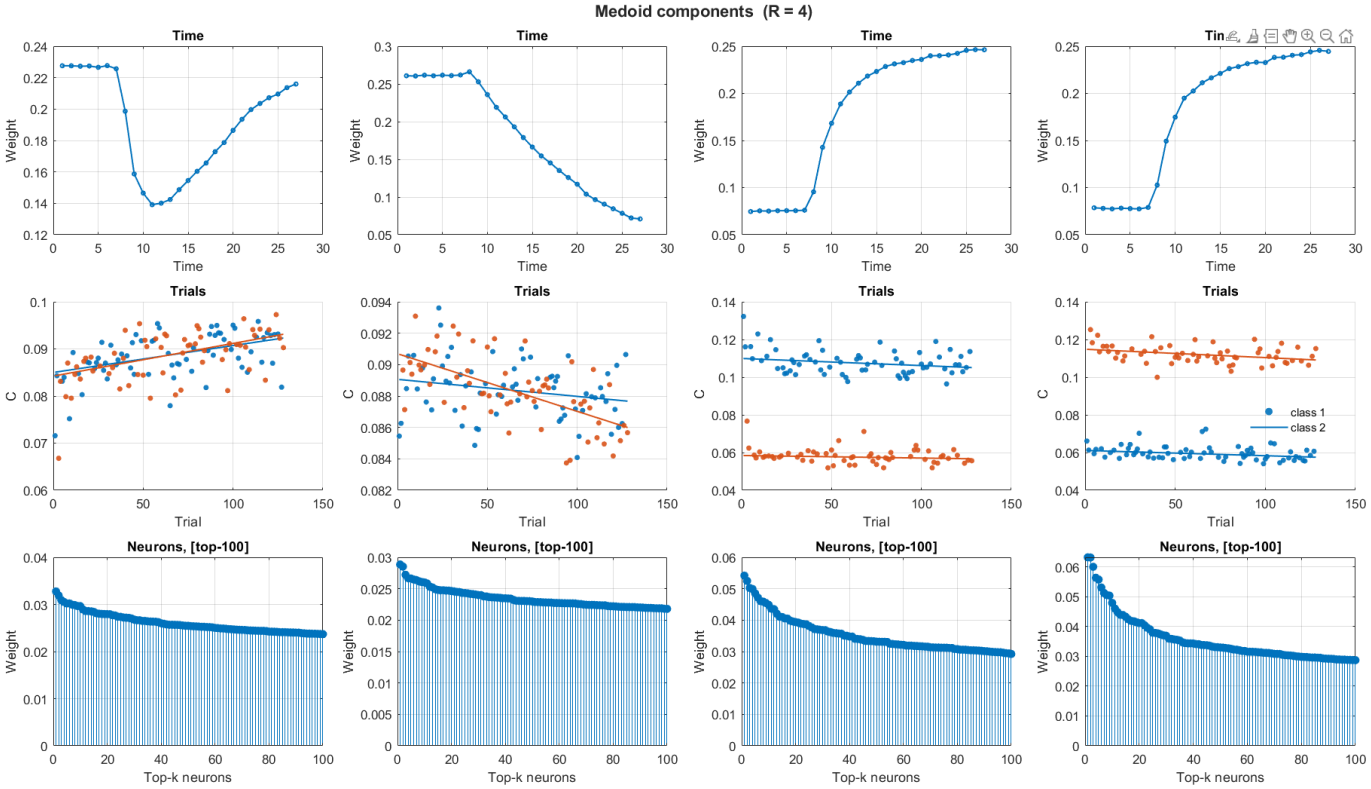


Figure 4.18: Factors resulting from rank $\mathbf{R} = 4$ CP decomposition. The top panels show temporal profiles from the **B** factors (time mode). The central panels report the trial-wise response strengths in **C**, with colors distinguishing stimulus identity (S_A vs. S_B). The bottom panels show the neuron factor **A**, reordered by coefficient magnitude; the top 100 entries are displayed.

Finally, to quantify component overlap in terms of neural population, for each factor we considered the top x% neurons in the neuron mode (ranked by loading) and measured pairwise similarity of the two sets via the Jaccard index,

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}.$$

Figures 4.19–4.21 report the overlap matrices for $x \in \{1,5,10\}$%. Across all thresholds, off-diagonal entries remain low, indicating that distinct components recruit largely non-overlapping neuron subsets. The model could successfully disentangle the populations encoding responses to the two stimuli S_A and S_B.

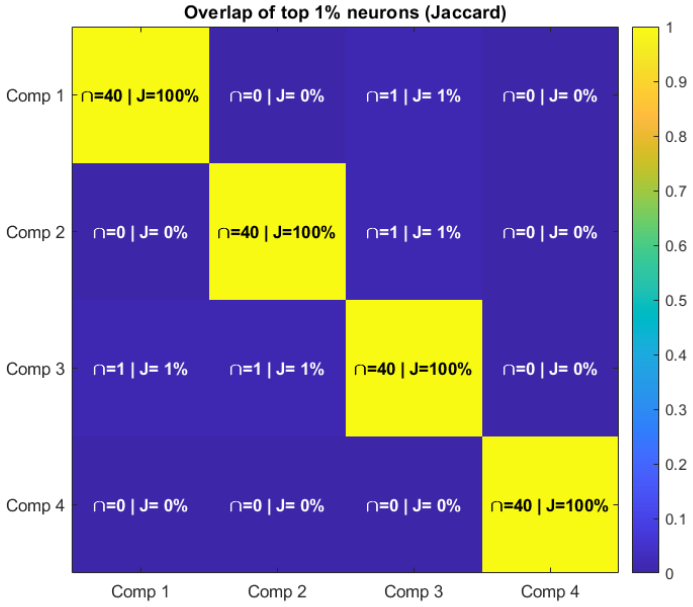


Figure 4.19: Overlap among top 1% neurons per component ordered by loadings. Cells display Jaccard similarity.

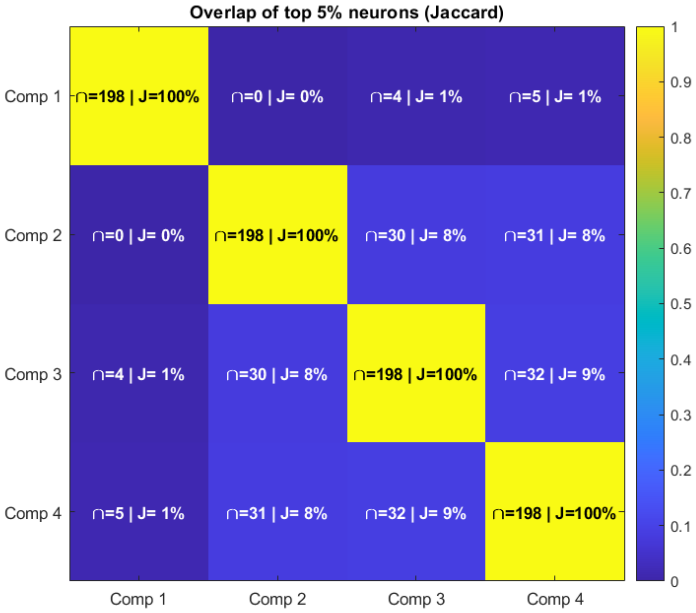


Figure 4.20: Overlap among top 5% neurons per component ordered by loadings. Cells display Jaccard similarity.

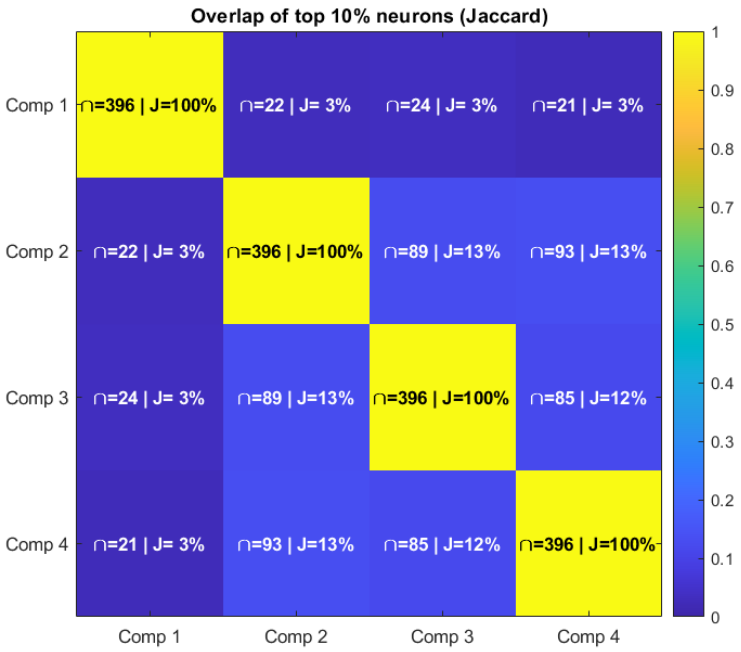


Figure 4.21: Overlap among top 10% neurons per component ordered by loadings. Cells display Jaccard similarity.

Chapter 5

Results

After decomposing the recordings with a rank-$R = 4$ CP/PARAFAC model we have reconstructed the tensor using all the factors. For each neuron i, we defined early (E) and late (L) blocks as the first and last $K = 5$ presentations per stimulus, and computed mean responses over time and trials. The stimulus-specific changes related to S_A and S_B are

$$\Delta x_{\mathrm{i}}^{A} = \bar{x}_i^{A,L} - \bar{x}_i^{A,E}, \qquad \Delta x_{\mathrm{i}}^{B} = \bar{x}_i^{B,L} - \bar{x}_i^{B,E}.$$

Each neuron's tuning would then be represented by an arrow from $(\bar{x}_i^{A,E}, \bar{x}_i^{B,E})$ to $(\bar{x}_i^{A,L}, \bar{x}_i^{B,L})$ in the (S_A, S_B) plane, as described in §3.3.2 and illustrated in Figures 3.2 and 3.3. Because the population is large and a neuron-level plot would be overcrowded, we summarize tuning directions with a 20-bin circular glyph weighted by arrow magnitude.

Directional weighted 20-bin glyph. For each neuron we form the change vector $(\Delta x_i^A, \Delta x_i^B)$ and convert it to polar form with angle θ_i and magnitute m_i:

$$\theta_i = \mathrm{atan2}(\Delta x_i^A, \Delta x_i^B) \in (-\pi, \pi], \quad m_i = \sqrt{(\Delta x_i^A)^2 + (\Delta x_i^B)^2}.$$

The circle is partitioned into 20 equal angular sectors by edges

$$\{-\pi = \phi_0 < \phi_1 < \cdots < \phi_{20} = \pi\}$$

with $\phi_k = -\pi + k \cdot (2\pi/20)$ and bin centers $\psi_k = (\phi_{k-1} + \phi_k)/2$. Each arrow contributes to exactly one bin $k = \min\{j : \theta_i \le \phi_j\}$. We draw from the origin, along direction ψ_k, a ray of length

$$L_k = \frac{s_k}{\sum_{i=1}^{N} m_i}, \qquad s_k = \sum_{i:\, \theta_i \in (\phi_{k-1},\, \phi_k]} m_i,$$

so that L_k encodes the *share of total movement* (vector-norm) for bin k. The 20 rays plotted at $\{\psi_k\}$ form a circular histogram whose radial extent reflects the distribution of change magnitudes across directions; dotted diagonals at $\pm 45°$ are there to provide visual reference. A synthetic example with unit-length rays is reported in Figure 5.1.

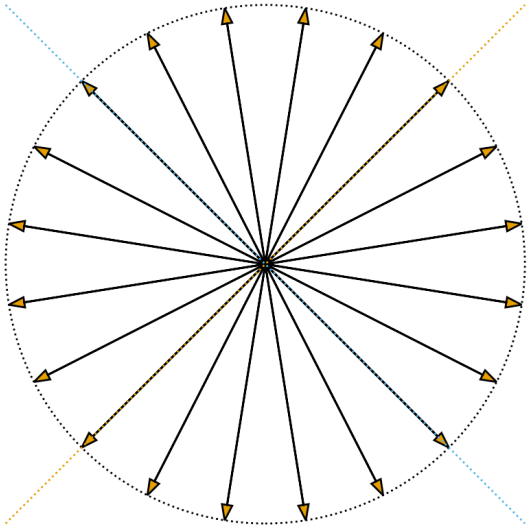


Figure 5.1: 20-bin weighted directional glyph

Under the single–learning-signal hypothesis H_1 derived in §3.3.2, both stimulus-locked synaptic weights are driven by the same scalar learning drive δ_i and therefore, as we derived, changes in activity must share the same sign:

$$\operatorname{sign}\left(\Delta x_i^A\right) \;=\; \operatorname{sign}\left(\Delta x_i^B\right)$$

Geometrically, these are precisely the arrows falling in the NE quadrant ($\Delta x_i^A > 0,\ \Delta x_i^B > 0$) or the SW quadrant ($\Delta x_i^A < 0,\ \Delta x_i^B < 0$). Arrows in the NW/SE quadrants ($\Delta x_i^A\, \Delta x_i^B < 0$) violate the same-sign constraint and are compatible with the alternative H_0.

5.1 Qualitative and quantitative evidence for H_0 vs H_1

Figure 5.2 shows the circular weighted glyph (all neurons) obtained from the CP decomposition $R = 4$ once factors were reassembled in a tensor. The distribution of

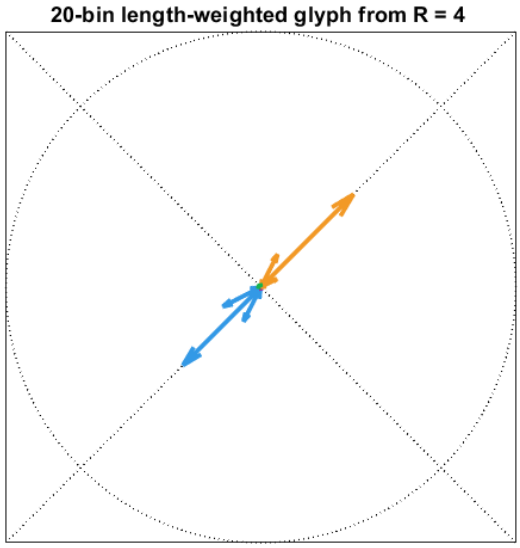


Figure 5.2: 20-bin weighted glyph resulting form the $R = 4$ CP model

arrow directions concentrates along the same–sign diagonals (NE and SO), whereas opposite–sign quadrants (SE and NO) are comparatively rare. Figure 5.3 shows a simplified four–bin glyph centered on NE, NO, SE, and SO, where arrow lengths are proportional to the *fraction of neurons* pointing in each quadrant, independently of change magnitude.

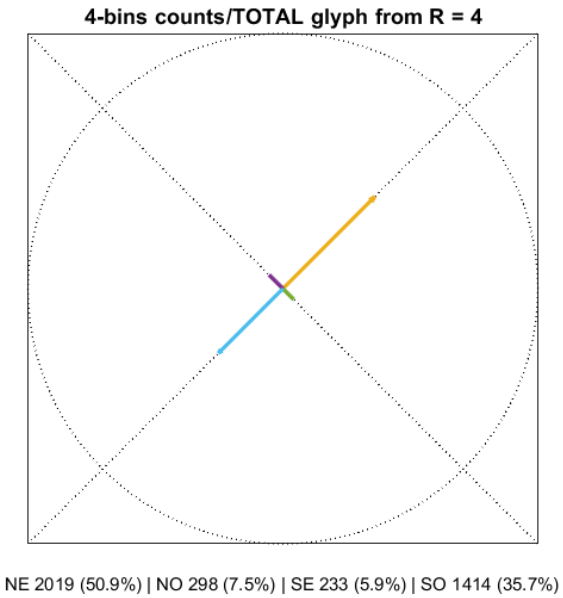


Figure 5.3: Four-bin empirical frequencies glyph from the $R = 4$ CP model. Arrows at NE, NO, SE, and SO encode the fraction of neurons whose changes $(\Delta x_i^A, \Delta x_i^B)$ fall in each quadrant; lengths reflect frequency only (not change magnitude).

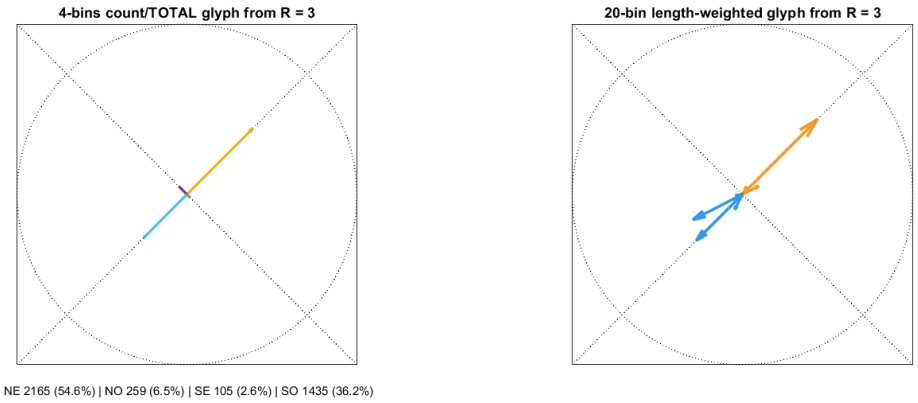
Quadrant	Count	%Count	SumLen	%Weighted
NE	2019	50.93%	15.300	45.27%
SE	233	5.88%	0.735	2.17%
NO	298	7.52%	1.250	3.69%
SO	1414	35.67%	16.600	48.87%
Total	**3964**	**100.00%**	**33.885**	**100.00%**

Table 5.1: Quadrant summary from the length-weighted glyph. NE: $(\Delta x_i^A > 0,\ \Delta x_i^B > 0)$, SE: $(\Delta x_i^A > 0,\ \Delta x_i^B < 0)$, NO: $(\Delta x_i^A < 0,\ \Delta x_i^B > 0)$, SO: $(\Delta x_i^A < 0,\ \Delta x_i^B < 0)$. SumLen is the total arrow length (movement magnitude) within a quadrant, %Weighted its share of the total movement.

Same–sign changes dominate (NE+SO = 86.6% by count), indicating that responses to A and B tend to evolve in the same direction across neurons. While NE contains the largest fraction of neurons (50.93%), SO contributes the largest share of total movement (48.87%). Thus, joint decreases (SO) are fewer than joint increases (NE) but are typically stronger in magnitude.

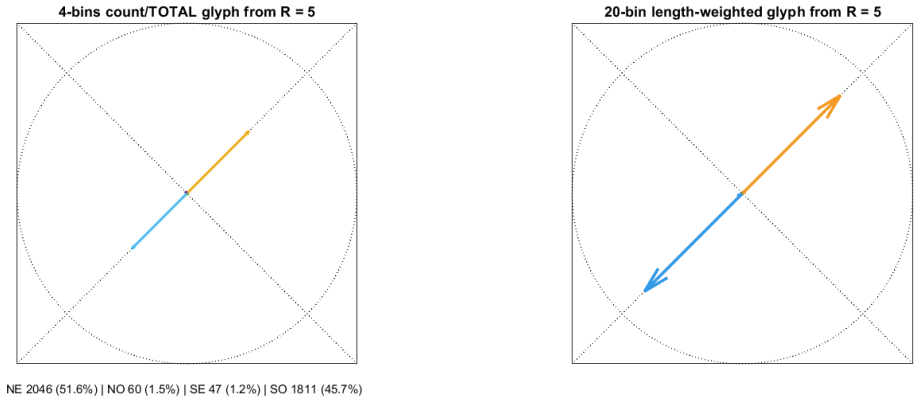
5.1.1 Robustness across model orders.

For completeness, we repeated the glyph analysis on the medoids obtained using ranks $R = 4 \pm 1$ (i.e., $R \in \{3,4,5\}$) and also at $R = 10$. Across these settings, the qualitative patterns of the glyphs remained consistent, indicating that our conclusions are not sensitive to moderate changes in the chosen CP rank.



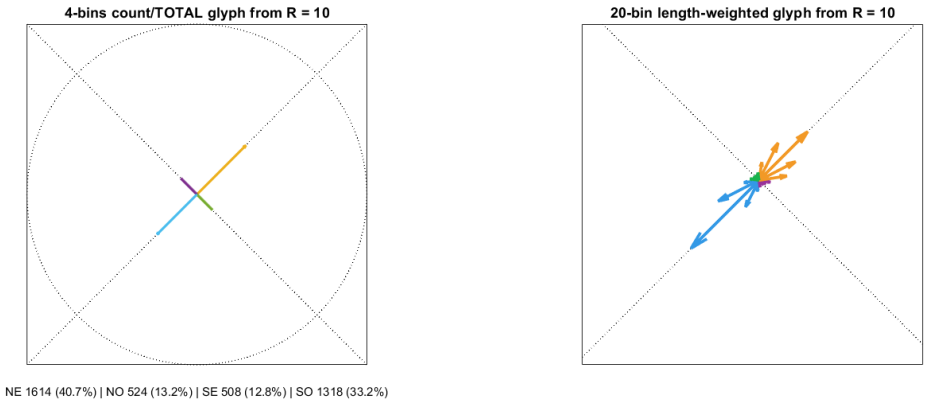
(a) **(b)**

Figure 5.4: Glyph analysis for the denoised tensor with CP rank $R = 3$: **(a)** counts per direction; **(b)** weighted glyph reflecting the distribution and relative strength of directions.



(a) **(b)**

Figure 5.5: Glyph analysis for the denoised tensor with CP rank $R = 5$: **(a)** counts per direction; **(b)** weighted glyph reflecting the distribution and relative strength of directions.



(a) **(b)**

Figure 5.6: Glyph analysis for the denoised tensor with CP rank $R = 10$: **(a)** counts per direction; **(b)** weighted glyph reflecting the distribution and relative strength of directions.

At higher rank ($R = 10$, Figure 5.6) we observe some very short arrows in the NW and SE quadrants. Such arrows could be due to over-factoring: the additional components start to model noise as small variations yielding tiny vector magnitudes that almost disappear when glyphs are length–weighted, i.e. when vectors are scaled by their effect size. Hence, the qualitative structure remains governed by the stable NE/SW pattern.

Chapter 6

Conclusions

This thesis pursued a dual aim: at a technical level, it advanced a tensor–decomposition view of calcium–imaging data that preserves the [neurons×time×trials] structure and cleanly disentangles stimulus–related populations; then, at a biological level we used those factors to probe how neurons update responses to multiple stimuli during learning.

A small number of nonnegative CP components captured the dominant variance without collapsing time, revealing concise temporal kernels, neuron–mode sparsity, and stimulus–separating trial structure. Components could recruit largely distinct neuron subsets for stimulus–related structure (low overlaps at top–x% neuron sets), clarifying which assemblies carry which parts of the code.

Starting from the observation from [7] that reactivations act as a *single* neuron-level learning drive, we instantiated it in the delta-rule model of plasticity, arriving at the sign-coupling constraint $\text{sign}(\Delta x^A_{\text{post}}) = \text{sign}(\Delta x^B_{\text{post}})$ (Eq. 3.7). This formalization provided clear, testable predictions for population drift under the null and alternative hypotheses. The distribution of per-neuron change vectors showed that most neurons adjust their responses to both stimuli with the same sign, supporting a single neuron-level learning drive that acts like a feature-selection mechanism, enhancing some assemblies while down-weighting others.

Limitations These conclusions are preliminary: the current evidence (one mouse, one session) lacks the scale and dependence–aware inference required for a definitive claim, and naïve independence assumptions (e.g., applying Rayleigh's test directly to raw angle data) can overstate evidence in the presence of shared covariance across neurons, time, and trials. Strengthening this result will require larger, replicated

datasets, inference that preserves dependence (permutation or block/bootstrap procedures) and explicitly accounts for correlation of neuronal populations.

Future work. Future work should expand the dataset across animals and sessions to quantify effect sizes with uncertainty at both neuron and animal levels; develop rigorously dependence–aware inference, test robustness systematically across reasonable ranks while exploring variants and evolutions of the CP/PARAFAC decomposition as the temporally flexible model PARAFAC2;

Appendix A

CP-structured logistic neural decoder

In this appendix, we report an additional analysis on the performance of a neural decoder built with the CP-structured logistic regression model illustrated in 2.6 (custom implementation), which infers the stimulus identity $[S_1, S_2]$ from the neuronal population's stimulus-evoked responses.

No definitive conclusion is intended here, as that would require a much more rigorous treatment. The goal of this paragraph is simply to showcase some indicative results that may spark interest in a class of decoders for computational neuroscience that has received comparatively little attention so far, overshadowed by simpler, more conventional techniques.

The purpose of these experiments is methodological: to assess the potential gains of tensor–aware decoders that preserve temporal structure and to compare their performance with the time–averaged logistic regressor used, for example, in [5]. However, they do not directly address the neuroscientific hypothesis we tested in the main chapters. For this reason, the CP–logistic results are presented here in the appendix rather than in the core Results and Discussion.

We compared the performance of a CP-structured logistic regressor with that of a baseline provided by a standard logistic regression model. In the baseline, each neuron's activity is collapsed into a time-averaged feature. The CP-structured regressor instead operates directly on the full [neurons $\times$ time $\times$ trials] activity tensor, using a Canonical Polyadic (CP) decomposition to impose a low-rank structure on the weights as illustrated in 2.6. In this way, the model can, in principle, exploit the rich temporal organization of responses and their interactions with stimulus identity.

Baseline. For the baseline decoder we used the standard ℓ_2-regularised logistic regression implemented in the `LogisticRegression` class of the `scikit-learn` library. Given features $\mathbf{x}_i \in \mathbb{R}^P$ and labels $y_i \in \{1, \dots, C\}$, the model parametrises class scores as in (2.1) and estimates $(\mathbf{W}, \mathbf{b})$ by minimising the regularised cross-entropy

$$\min_{\mathbf{W},\,\mathbf{b}} \; \frac{1}{n} \sum_{i=1}^{n} \Big[-\log p\big(y_i \mid \mathbf{x}_i\big) \Big] \; + \; \frac{1}{2C} \, \|\mathbf{W}\|_F^2,$$

where $C > 0$ is the inverse regularization strength used in the hyperparameter search and $\|\cdot\|_F$ denotes the Frobenius norm.

To construct a controlled test bench of classification problems, we proceeded as follows. First, we reshaped the activity tensor into a [trials $\times$ neurons $\times$ time] array and collapsed the temporal dimension by averaging each neuron's trace, yielding a [trials $\times$ neurons] design matrix[1]. Using this time–averaged representation, we estimated a decoding score for each neuron separately: for each neuron, we trained a univariate logistic regression model and computed its 5-fold stratified cross-validated accuracy, yielding one accuracy value per neuron. These single-neuron accuracies were used to rank neurons from worst to best.

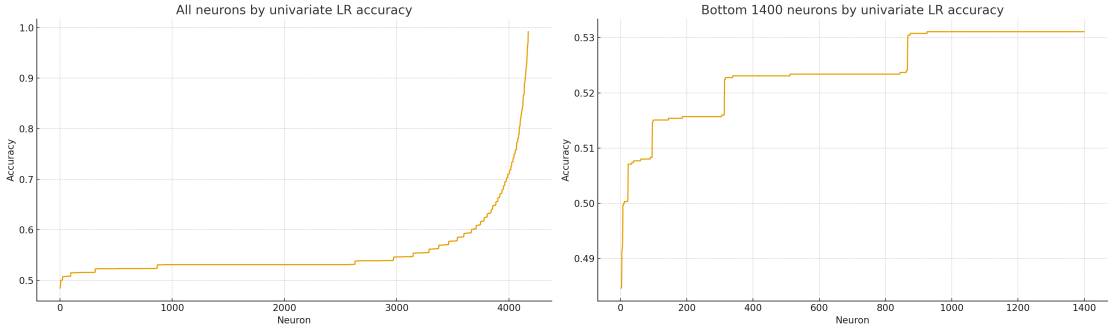
The hyperparameters for the univariate logistic regression baseline were chosen by a grid search, summarised in Table A.1. For each configuration, defined by the maximum number of iterations, the optimiser, and the inverse regularisation strength C, we computed the cross-validated decoding accuracy of the baseline classifier over the whole neuron set and then summarised its distribution across neurons (median, and quantiles). The final choice corresponds to the configuration achieving the highest median accuracy; among models with comparable median accuracy, we further selected the one with the largest lower 25% quantile. This criterion favours models that perform well not only on average, but also in a robust way across most neurons, avoiding those that performed well on a small subset of highly informative units while leaving a large fraction of neurons essentially uninformative.

In Figure A.1 we report the cross-validated decoding accuracies of the univariate classifier for the neurons, ordered from lowest to highest accuracy.

[1]To avoid data leakage, we did not apply the trace by trace 0–1 normalisation used in the main preprocessing pipeline. Instead, for each split of the 5-fold cross-validation, we fitted a per-neuron standardiser (zero mean, unit variance) using only the four training folds and then applied this transformation to the held-out fold.

Table A.1: Hyperparameter search for univariate LR

Parameter	Search space	Selected
C	$\{1, 0.1, 0.05, 0.01, 0.001\}$	0.01
Iters	$\{1000, 2000, 4000\}$	2000
Optimizer	$\{'lbfgs', 'liblinear'\}$	$'lbfgs'$



(a) **(b)**

Figure A.1: Sorted single-neuron decoding accuracies. **(a)** Classification accuracy of univariate logistic regressors trained on each neuron individually, sorted by performance across the full population; **(b)** Zoom on the 1400 lowest-scoring neurons; Notice that their accuracies cluster close to chance level.

Starting from this ranking, we defined families of baseline decoders that use only the first *k-worst* neurons in the ordered list. For each k, we trained a multivariate logistic regression on the corresponding subset of time–averaged features. We used the same 5-fold stratified cross-validation splits for all values of k. This yielded a curve of mean CV accuracy (with standard deviations across folds) as a function of the number of neurons included. This procedure produces a sequence of logistic benchmarks. We can then compare the performance of the CP-structured decoder that operates on the full tensor against these benchmarks.

Using exactly the same neuron subsets and the same CV folds, we then trained three CP–structured logistic regressors with ranks $R \in \{3,4,5\}$. In our analysis we keep the CP hyperparameters (rank, learning rate= 0.001, weight decay= 0.001, maximum number of epochs= 2500) fixed across all neuron subsets k, and likewise use a single hyperparameter setting for the logistic regression, so that performance as a function of k reflects the behavior of each model under a consistent configuration rather than subset-specific retuning.

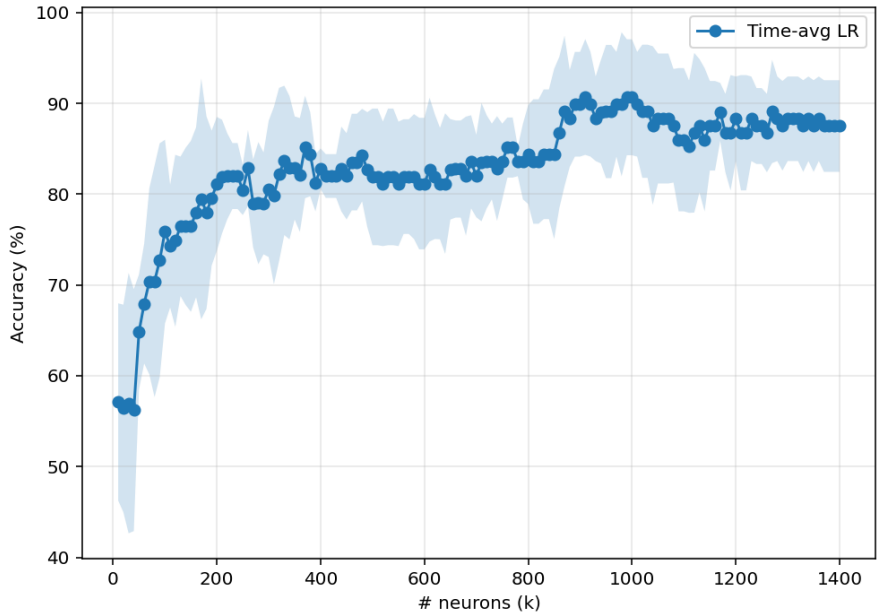


Figure A.2: Time–averaged LR baseline as a function of k. Mean±std CV accuracy of the time–averaged logistic regression decoder as the number of included neurons k increases (from 10 to 1400 in steps of 10). The same neuron ranking and outer CV folds used for the CP–logistic models are employed here.

Results. Figure A.2 shows the performance of the same time–averaged LR baseline when progressively including the first 1400 neurons in the ranking in steps of ten. The resulting mean accuracies ± std, of the three CP-structured logistic regressors, are reported in Figures A.3 - A.5, alongside the baseline classifier for comparison. It is evident that the CP-based decoder reaches higher accuracies with fewer neurons in the early subsets; it also achieves higher mean accuracies than the time-averaged logistic regression baseline in all three cases $R \in \{3,4,5\}$, and the standard deviations across the same cross-validation folds are generally smaller. However, the mean accuracy of the LR baseline appears more stable as a function of k, whereas the CP regressor, for a few isolated neuron subsets, performs markedly worse than logistic regression, exhibiting deep accuracy drops that become less pronounced as the rank increases.

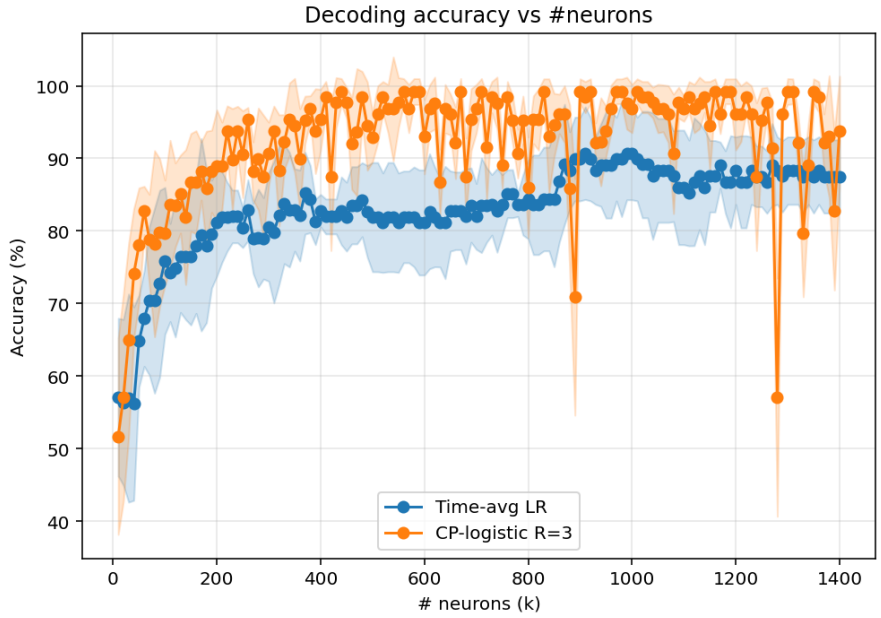


Figure A.3: CP–logistic decoder vs time–averaged LR, rank $R = 3$. Mean±std CV accuracy of the CP–logistic decoder as a function of the number of neurons k (from 10 to 1400 in steps of 10), together with the time–averaged LR baseline fitted on the same neuron subsets and outer folds.

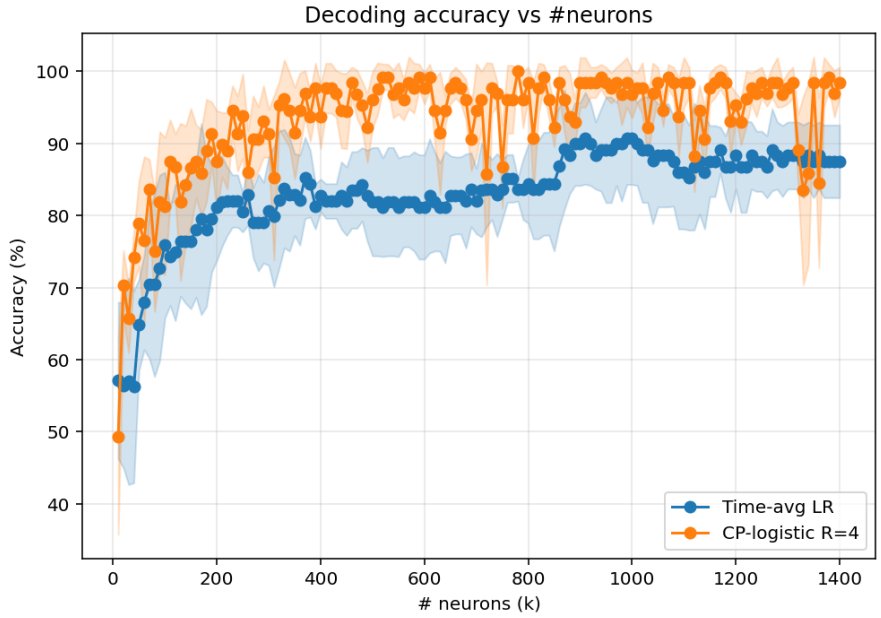


Figure A.4: CP–logistic decoder vs time–averaged LR, rank $R = 4$. Same analysis as in Fig. A.3, now for a CP–logistic decoder with rank $R = 4$.

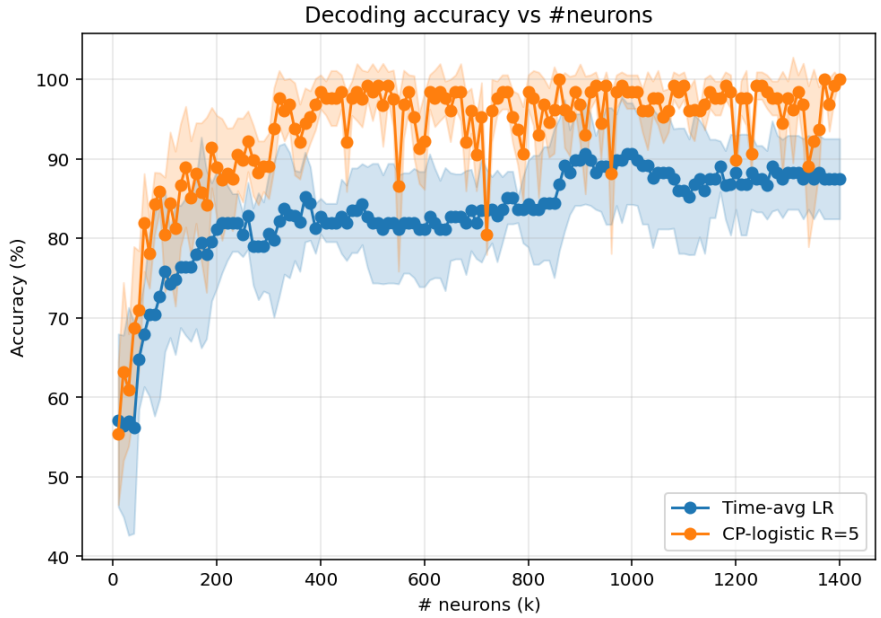


Figure A.5: CP–logistic decoder vs time–averaged LR, rank $R = 5$. Same analysis as in Fig. A.3, now for a CP–logistic decoder with rank $R = 5$.

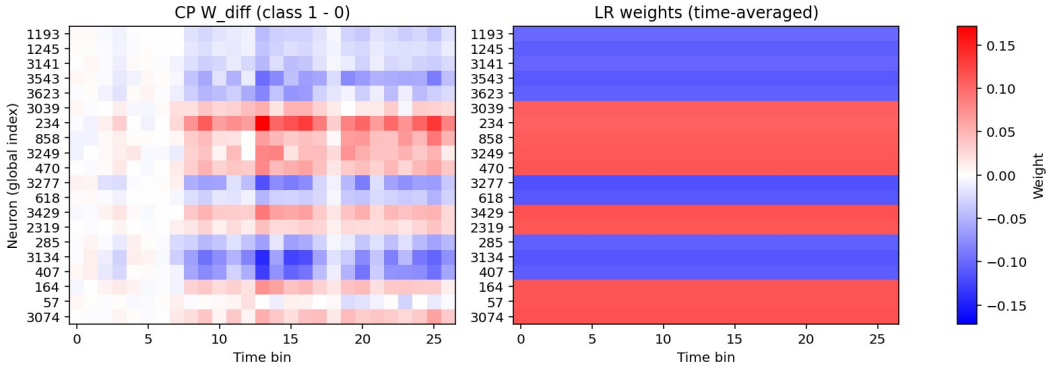
Model interpretation. One of the main reasons to use decoders in this context is to obtain interpretable models from which to draw conclusions about neural dynamics. The interpretability of the CP-structured regressor is not limited to identifying which neurons contribute most to each stimulus, as one can already do by inspecting the coefficients of a standard LR. In addition, it offers insight into the key features of the temporal dynamics that distinguish the responses to the two stimuli. For a two-class CP decoder with factors $A \in \mathbb{R}^{N\times R}$ and $B \in \mathbb{R}^{T\times R}$ and class weights $W_{\text{class}} \in \mathbb{R}^{R\times 2}$, the effective neuron–by–time weight matrix for class $c \in \{1,2\}$ is

$$W^{(c)}(n,t) \;=\; \sum_{r=1}^{R} W_{\text{class}}(r,c)\, A(n,r)\, B(t,r),$$

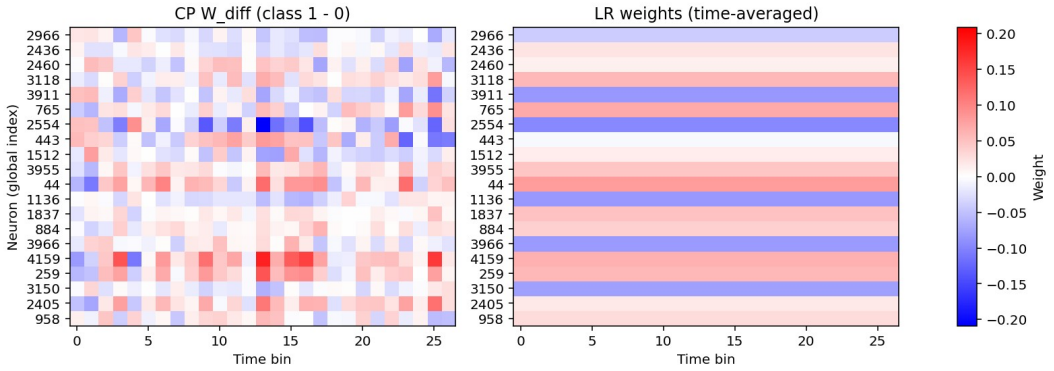
so that the logit for class c is the inner product between $W^{(c)}$ and the activity tensor. A simple and informative way to inspect the model is to visualise the *difference* matrix

$$\Delta W(n,t) \;=\; W^{(1)}(n,t) - W^{(2)}(n,t),$$

shown in Figures A.6a and A.6b for $k = 20$ neurons taken from the best and worst tails of the univariate ranking. Each pixel of ΔW reports how much activity of a given neuron at a given time bin pushes the decision towards stimulus 1 (red) rather than stimulus 2 (blue); the magnitude $|\Delta W(n,t)|$ encodes the strength of this effect. For the same subsets of neurons, we also plot the corresponding LR weights as a heatmap. This makes clear that the LR baseline can only assign a single, time-independent weight to each neuron (horizontal stripes), whereas the rank-5 CP model discovers structured patches of positive and negative contributions that are confined to specific temporal windows and subsets of neurons. In the best neurons, these patches form coherent motifs that align with the task-relevant epochs, while for the worst neurons ΔW is more diffuse and closer to zero, consistent with their low individual predictive power. Overall, the CP representation turns the decoder into a temporally resolved "importance map", providing a much richer and more interpretable description of how population activity supports the discrimination between the two stimuli.



(a) Difference matrix $\Delta W = W^{(1)} - W^{(2)}$ for the $k = 20$ *best* neurons in the univariate ranking(both decoders achieve 100% accuracy on this subset). Each pixel encodes how much activity of neuron n at time bin t pushes the decision towards stimulus 1 (red) rather than stimulus 2 (blue); the magnitude reflects the strength of this contribution. Most discriminative weights concentrate after the seventh time bin, suggesting that this late response segment carries the strongest stimulus-specific information.



(b) Same representation as in panel (a), now for the $k = 20$ *worst* neurons in the univariate ranking. Here the entries of ΔW are generally smaller in magnitude and less structured in time, consistent with the low individual predictive power of these units.

Figure A.6: CP decoder weight-difference maps for the two extremes of the neuron ranking. Panels (a) and (b) show the neuron–time difference matrix $\Delta W = W^{(1)} - W^{(2)}$ learned by the rank-5 CP-structured decoder for the 20 best and 20 worst neurons, respectively. These heatmaps summarise which neuron–time pairs are most discriminative for the two stimuli, and how this structure depends on the quality of the underlying units.

Bibliography

[1] João Sacramento, Rui Ponte Costa, Yoshua Bengio, and Walter Senn. «Dendritic cortical microcircuits approximate the backpropagation algorithm». In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2018. arXiv: 1810.11393. URL: https://papers.neurips.cc/paper/8089-dendritic-cortical-microcircuits-approximate-the-backpropagation-algorithm.pdf (cit. on p. 1).

[2] Alexandre Payeur, Jordan Guerguiev, Friedemann Zenke, Blake A. Richards, and Richard Naud. «Burst-dependent synaptic plasticity can coordinate learning in hierarchical circuits». In: *Nature Neuroscience* 24.7 (2021), pp. 1010–1019. DOI: 10.1038/s41593-021-00857-x. URL: https://www.nature.com/articles/s41593-021-00857-x (cit. on p. 1).

[3] Alexander Meulemans, Matilde Tristany Farinha, Javier García Ordóñez, Pau Vilimelis Aceituno, João Sacramento, and Benjamin F. Grewe. «Credit Assignment in Neural Networks through Deep Feedback Control». In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2021. arXiv: 2106.07887. URL: https://proceedings.neurips.cc/paper/2021/file/25048eb6a33209cb5a815bff0cf6887c-Paper.pdf (cit. on p. 1).

[4] Alexander U. Sugden et al. «Cortical reactivations of recent sensory experiences predict bidirectional network changes during learning». In: *Nature Neuroscience* 23.8 (2020), pp. 981–991. DOI: 10.1038/s41593-020-0651-5. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC7392804/ (cit. on p. 1).

[5] N. D. Nguyen et al. «Cortical reactivations predict future sensory responses». In: *Nature* 625.7993 (Jan. 2024). Epub 2023-12-13, pp. 110–118. DOI: 10.1038/s41586-023-06810-1 (cit. on pp. 1, 27, 28, 61).

[6] Pau Vilimelis Aceituno, Sander de Haan, Reinhard Loidl, and Benjamin F. Grewe. «Evidence for Target Learning in the Neocortex». In: *bioRxiv* (2024). Preprint; multiple revised versions exist under this DOI. DOI: 10.1101/2024.04.10.588837. URL: https://www.biorxiv.org/content/10.1101/2024.04.10.588837v5 (cit. on p. 1).

[7] Lhea Beumer. «Credit Assignment in Cortex: Backpropagation or Target Learning?» Research conducted at the Institute of Neuroinformatics (ETH Zürich & University of Zürich). Master's thesis. Groningen, The Netherlands: University of Groningen, 2025 (cit. on pp. 1, 21, 28, 58).

[8] Tamara G. Kolda and Brett W. Bader. «Tensor Decompositions and Applications». In: *SIAM Review* 51.3 (2009), pp. 455–500. DOI: 10.1137/07070111X (cit. on pp. 2, 17, 18).

[9] Grey Ballard and Tamara G. Kolda. *Tensor Decompositions for Data Science*. Cambridge, UK: Cambridge University Press, 2025. ISBN: 9781009471671. DOI: 10.1017/9781009471664. URL: https://www.cambridge.org/core/books/tensor-decompositions-for-data-science/640814D308696CD61CB9112EA57B2911 (cit. on p. 2).

[10] Joseph B. Kruskal. «Three-way arrays: Rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics». In: *Linear Algebra and its Applications* 18.2 (1977), pp. 95–138. DOI: 10.1016/0024-3795(77)90069-6 (cit. on pp. 2, 10).

[11] Hua Zhou, Lexin Li, and Hongtu Zhu. «Tensor Regression with Applications in Neuroimaging Data Analysis». In: *Journal of the American Statistical Association* 108.502 (2013), pp. 540–552. DOI: 10.1080/01621459.2013.776499 (cit. on pp. 2, 17).

[12] Arthur Pellegrino, Heike Stein, and N. Alex Cayco-Gajic. «Dimensionality reduction beyond neural subspaces with slice tensor component analysis». In: *Nature Neuroscience* 27.6 (2024), pp. 1199–1210. DOI: 10.1038/s41593-024-01626-2 (cit. on p. 2).

[13] Richard A. Harshman. «PARAFAC2: Mathematical and Technical Notes». In: *UCLA Working Papers in Phonetics* 22 (1972), pp. 30–44. URL: https://psychology.uwo.ca/faculty/harshman/wpppfac2.pdf (cit. on p. 2).

[14] Johan Håstad. «Tensor Rank is NP-Complete». In: *Journal of Algorithms* 11.4 (1990), pp. 644–654. DOI: 10.1016/0196-6774(90)90014-6 (cit. on p. 13).

[15] Max Welling and Markus Weber. «Positive tensor factorization». In: *Pattern Recognition Letters* 22.12 (2001). Selected Papers from the 11th Portuguese Conference on Pattern Recognition - RECPAD2000, pp. 1255–1261. ISSN: 0167-8655. DOI: https://doi.org/10.1016/S0167-8655(01)00070-8. URL: https://www.sciencedirect.com/science/article/pii/S0167865501000708 (cit. on p. 14).

[16] Daniel D. Lee and H. Sebastian Seung. «Learning the parts of objects by non-negative matrix factorization». In: *Nature* 401.6755 (1999), pp. 788–791. DOI: 10.1038/44565. URL: https://doi.org/10.1038/44565 (cit. on p. 14).

[17] J. Liu, C. Zhu, Z. Long, and Y. Liu. *Tensor Regression*. arXiv preprint. 2023. DOI: 10.1561/2200000087. URL: https://doi.org/10.1561/2200000087 (cit. on pp. 15, 16).

[18] Boyang Zang, Tao Sun, Yang Lu, Yuhang Zhang, Guihuai Wang, and Sen Wan. «Tensor-powered insights into neural dynamics». In: *Communications Biology* 8.1 (2025), p. 298. DOI: 10.1038/s42003-025-07711-x. URL: https://doi.org/10.1038/s42003-025-07711-x (cit. on p. 15).

[19] Matthew A. Wilson and Bruce L. McNaughton. «Reactivation of hippocampal ensemble memories during sleep». In: *Science* 265.5172 (1994), pp. 676–679. DOI: 10.1126/science.8036517 (cit. on p. 20).

[20] Daoyun Ji and Matthew A. Wilson. «Coordinated memory replay in the visual cortex and hippocampus during sleep». In: *Nature Neuroscience* 10.1 (2007), pp. 100–107. DOI: 10.1038/nn1825 (cit. on p. 20).

[21] Stephen N. Gomperts, Fabian Kloosterman, and Matthew A. Wilson. «VTA neurons coordinate with the hippocampal reactivation of spatial experience». In: *eLife* 4 (2015), e05360. DOI: 10.7554/eLife.05360 (cit. on p. 20).

[22] C. M. Tigaret, K. T. Tsaneva-Atanasova, G. L. Collingridge, and J. R. Mellor. «Wavelet Transform-Based De-Noising for Two-Photon Imaging of Synaptic Ca^{2+} Transients». In: *Biophysical Journal* 104 (2013), pp. 1006–1017. DOI: 10.1016/j.bpj.2013.01.015 (cit. on p. 36).

[23] Min Li, Wuhong Wang, Zhen Liu, Mingjun Qiu, and Dayi Qu. «Driver Behavior and Intention Recognition Based on Wavelet Denoising and Bayesian Theory». In: *Sustainability* 14.11 (2022), p. 6901. DOI: 10.3390/su14116901 (cit. on p. 36).

[24] Brett W. Bader, Tamara G. Kolda, Daniel M. Dunlavy, et al. *Tensor Toolbox for MATLAB*. Version 3.7. Sandia National Laboratories and MathSci.ai, Oct. 1, 2025. URL: https://www.tensortoolbox.org (cit. on p. 41).

[25] Christos Chatzis, Carla Schenker, Jérémy E. Cohen, and Evrim Acar. *dCMF: Learning interpretable evolving patterns from temporal multiway data*. 2025. arXiv: 2502.19367 [cs.LG]. URL: https://arxiv.org/abs/2502.19367 (cit. on p. 43).