

1 Part 1

The following series of exercises concern the application of LDS models to monkey neural data recorded while it was set to perform a task requiring motor control. More specifically, we constrain the model to be one that is rotational in state-space. The motivation for this is the periodicity that is often observed in real life activities for animals. With the given task for the monkey also exhibiting the repetitiveness that may have influenced evolved behaviors - an interesting hypothesis is to explore the validity of this rotational model in being the underlying driving force for the neural data.

To begin with, some basic plots of the neural data can be made and some qualitative assessments of the nature of the data can be made. The Peristimulus Time Histograms (PSTH) data is decomposed into a 3D array (containing data of all neural data across time and different task conditions) and a linear time array.

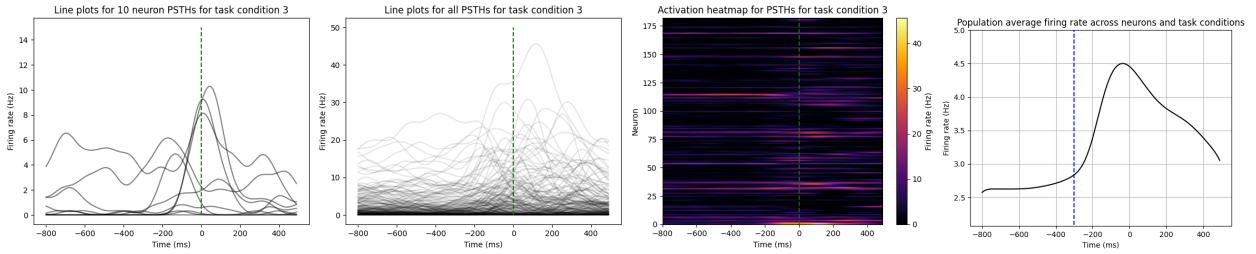


Figure 1: Neural data plots (from left to right: line plots of few neurons, all neurons for specified task, heat-map activation plot for specified task, population average across neurons and tasks)

Referring to figure 1, it can be observed that before the go cue, the neuron activity is stagnant, and somewhat constant. Individual neurons are either in a constant 'on' or 'off' state (corresponding to consistent firing or no firing). About 300 ms before the cue there is a transient stage where neurons change their firing rates. This transience is held until about 200 ms after the go cue after which there is another period of neurons stabilizing (presumably as the activity being performed is constant in that time frame - e.g. moving in a straight line to target).

The population average plot shows that on average, there is increased neural activity as we approach the 'go' cue. What is interesting is that from qualitative analysis - though this can be asserted by using something like an edge-identifying filter like a Laplacian of Gaussian - around -300 ms is the time when neuron activity changes significantly. This is interesting as it suggests a sort of non-causality between the timing of the go cue and the neuron activity. If one is to use a linear causal model to describe neuron activity, the non-causality can be attributed to mismatched calibration of the measuring instrument. More likely though, the monkey will be anticipating the go cue - and in expectation of the reward, neurons may begin firing well before the actual command. During training, the monkey could have picked up on clues and this could result in some sort of 'training-induced' anticipation. From the problem description - while the go cue is given at variable times, this is constrained to the interval (0.5, 1) seconds.

An interesting field of the literature to explore here is the effect of Reward Prediction Error signals (RPE) in dictating the anticipation. Given the training set-up (the task itself is non-probabilistic) this wouldn't really be a key factor to explore here. Though what could be happening is that the monkey is envisioning performing the action it needs to perform. Literature regarding RPE signals indicate that repetitive training induce beta oscillations in the monkey striatum which encode RPE signals - such that after target on-set but before the go cue, the neural activity may already transition to the initial task-execution state [1]. This is expanded on more in part 6.

2 Part 2

One can now work on reducing the dimensionality of the data, the goal is to go from 182 neurons across conditions and time bins to a lower dimension (for this report we set $M = 12$) 2D representation with the conditions and time bins axes concatenated. Before the steps for PCA are followed, some data pre-processing is performed. The data is normalized with respect to the maximum and minimum observed firing rates. This then constrains the firing rates to be between 0 and 1. The data is then mean centered (across neurons and time bins). Normalization ensures that each neuron is represented equally in the dataset and that the neural activity decomposition focuses on the *change* in neural activity rather than its magnitude (some neurons may be predisposed to be frequent firing). Mean centering then equally weights the 'on' and 'off' state of the neuron.

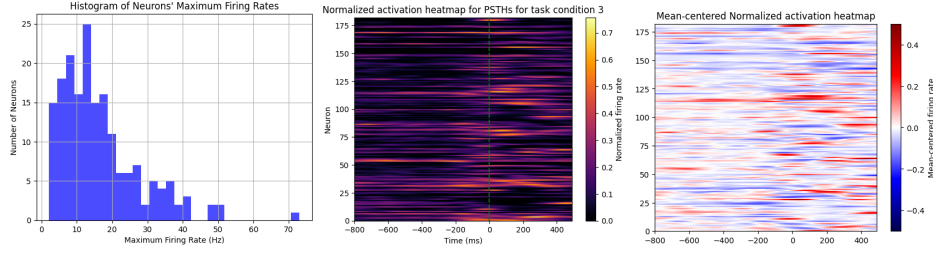


Figure 2: From left to right: PSTH neuron firing rate histogram, task-specific normalization, task-specific mean centering

PCA decomposition is performed then by performing a covariance operation over the data. The data is truncated to time bins corresponding to the $[-150 \text{ ms}, 300 \text{ ms}]$ interval, then reshaped across the conditions and time bins axes. It is then a 2D array of shape 182×4968 . The 182×182 covariance matrix is calculated by taking the matrix product of the 2D array with its transpose. Eigenvalue decomposition is performed on this matrix, then the last 12 eigenvectors (corresponding to the largest 12 eigenvalues) form the 12×182 matrix that is V_m this then multiplies with the reshaped truncated PSTH array to yield a 12×4928 array.

3 Part 3

The M-dimensional trajectories can further be distilled into the first two PCA components. The trajectories of these components can be plotted - per task - on a 2D plane. A color-mapper will map colors according to the maximum spread direction of the initial points. When plotted with markers - figure 3 is produced. One can see that per-task trajectories are somewhat separable, though they are often quite stochastic in their heading with no cohesive structure. A key observation is that the start and end points of these trajectories are somewhat distinctly clustered (and are close to each other).

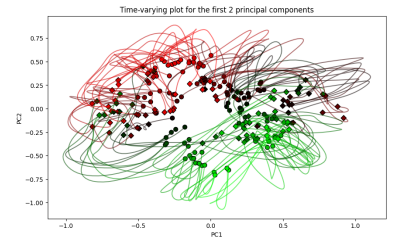


Figure 3: Trajectories of first 2 PCA components

4 Part 4

One can now begin with the required derivations to generate the maximum likelihood estimate for A. We start with the model definition. An attempt to fit an autonomous rotatory dynamical system to the neural data in projection space implies $z_{t+1} \approx R(\theta)z_t$ which - for small θ - can be simplified to $\Delta z_{t+1} \approx Az_t$ for an anti-symmetric A. This can be converted into an LDS equation: $\Delta z_{t+1} = Az_t + \sigma \epsilon_t$. So the maximum log-likelihood estimate for general A can be derived to be:

$$\mathcal{L} = \frac{-1}{2\sigma^2} \sum_t ((\Delta z_{t+1} - Az_t)^T (\Delta z_{t+1} - Az_t)) + D_{const} = \frac{-1}{2\sigma^2} \sum_t (\Delta z_{t+1}^T \Delta z_{t+1} + z_t^T A^T A z_t - 2\Delta z_{t+1}^T A z_t)$$

The log property converts the product of probabilities into sums of log-probabilities, for brevity the constant part of the log-likelihood is ignored in later parts (since we will be differentiating these anyway). Here Δz_{t+1} and z_t correspond to the the t^{th} time bin PCA projection. Taking the derivative of this with respect to the matrix A:

$$\frac{\partial \mathcal{L}}{\partial A} = \frac{\partial}{\partial A} \frac{-1}{2\sigma^2} \sum_t (z_t^T A^T A z_t - 2\Delta z_{t+1}^T A z_t) = \frac{-1}{\sigma^2} \sum_t (A z_t z_t^T - \Delta z_{t+1}^T z_t^T) = \frac{1}{\sigma^2} (\Delta Z - A Z) Z^T$$

The second step of this calculation can indeed be verified if we break down the matrix derivative into the derivative of the scalar with respect to the i^{th} and j^{th} components:

$$\frac{\partial \mathcal{L}}{\partial A_{ij}} = \frac{\partial}{\partial A_{ij}} \left(\frac{1}{\sigma^2} \sum_j \sum_i z_{t,j} A_{ij} \Delta z_{t+1,i} - \frac{1}{2\sigma^2} \sum_i \left(\sum_j A_{ij} z_{t,j} \right)^2 \right)$$

Which yields:

$$\frac{\partial \mathcal{L}}{\partial A_{ij}} = \frac{1}{\sigma^2} \left(z_{t,j} \Delta z_{t+1,i} - z_{t,j} \sum_k A_{ik} z_{t,k} \right)$$

Which - in matrix form - yields the same result as before. However, this is for general A. We do implement a constraint on the model - which is that it is a rotational dynamics model. From this - there is an added constraint that $A_{ij} = -A_{ji}$. To avoid performing a constrained optimization with a Lagrangian (which would be tedious with a matrix), one can instead convert this into an unconstrained problem by re-parameterizing A. We know that with only the upper-triangle part of the matrix (excluding the diagonal) we can reconstruct the entire matrix. In this sense, one can create a vector β which contains the data from the upper-triangle of the matrix- flattened out.

Therefore, the number of parameters for an $M \times M$ matrix reduces from $K = M^2$ to $K = \frac{M(M-1)}{2}$. Figure 4 demonstrates this process. More concisely, the reconstruction process can be performed by doing a tensor product between β and a 3D tensor H such that:

$$A_{ij} = \sum_{a=1}^K \beta_a H_{a ij}$$

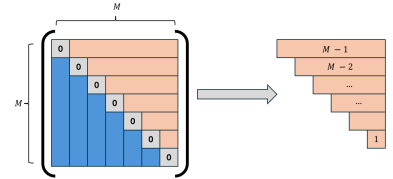


Figure 4: Diagram of constructing β vector

Here H is independent of A and only depends on the manner in which one flattens the upper triangle of an $M \times M$ matrix. We can now take the derivative of the log-likelihood with respect to β and perform unconstrained optimization - retrieve the optimal β and reconstruct A . We therefore need to represent the log-likelihood in terms of this β vector - this is easier when considering log-likelihood in element-wise fashion:

$$\mathcal{L} = D_{const} + \sum_t \left(\sum_i \left(\sum_j \sum_a \beta_a H_{a ij} z_{t,j} \right)^2 - 2 \sum_i \sum_j \sum_a \beta_a H_{a ij} \Delta z_{t,i} z_{t,j} \right)$$

This is a fairly tedious equation to work with, it is made easier with the introduction of an auxiliary tensor W where the elements of W are defined such that: $W_{ait} = \sum_j H_{a ij} z_{t,j}$. This then simplifies the above expression into:

$$\mathcal{L} = D_{const} + \sum_t \left(\sum_i \left(\sum_a \beta_a W_{ait} \right)^2 - 2 \sum_i \sum_a \beta_a W_{ait} \Delta z_{t,i} \right)$$

Then taking derivatives with respect to elements in $\beta \rightarrow \beta_k$:

$$\frac{\partial \mathcal{L}}{\partial \beta_k} = \frac{-1}{2\sigma^2} \sum_t \left(\sum_i 2 \left(\sum_a W_{ait} \beta_a \right) W_{kit} - 2 \sum_i W_{kit} u_{it} \right)$$

$$\frac{\partial \mathcal{L}}{\partial \beta_k} = \sum_t \sum_i u_{it} W_{kit} - \sum_t \sum_i \left(\sum_a W_{ait} \beta_a \right) W_{kit}$$

We see that this is an element wise representation of the form $b_i - (\beta Q)_i$, where the row-vector b and matrix Q are represented by tensor products such that:

$$b_k = \sum_i \sum_t W_{kit} u_{it} \quad ; \quad Q_{b,a} = \sum_i \sum_t W_{bit} W_{ait}$$

Then solving for the optimal (ML) β is simply solving the linear equation: $\beta_{optimal} = Q^{-1}b$. The Q matrix and b vector are solved for - in code - by taking tensor products across the relevant axes. Careful attention had to be given to ensuring that the shape is not mismatched in either term - so employing a strategy of regularly printing the shape of tensors as calculations were carried out was an important step in ensuring coherence of code. Matching matrix shapes is fairly crucial while debugging code in Python as Python has the undesirable behavior of sometimes switching row-wise and column-wise representations. Single vectors are printed as row-wise vectors but when eigenvalue decomposition is performed - for example - the eigenvectors are the columns of the returned matrix. the NumPy library fortunately preserves standard notation and matrix multiplicative rules.

From the MLE of β , the A matrix can simply be recovered by the equation involving it and the H tensor. Figure 5 gives the ML estimate of the rotational A matrix using the algorithm described thus far. From tests of known data an absolute mean difference of $2.02e - 6$ is observed, which is sufficiently small to confirm the workings of this algorithm. Now, the A matrix can be used to distill the data into 'rotation' hyper-planes in M -dimensional space.

5 Part 5

From theory, one can describe rotation in M -dimensional space as rotations in $M/2$ (for even M) 2D planes that are spanned by the real and imaginary parts of the eigen-vectors of the $M \times M$ matrix A . The eigen-values are purely imaginary (in executing code the real parts are in the order of magnitude of $e - 18$, whereas the imaginary parts are $e - 1$) and this is expected. They describe pure rotation in each of their planes.

Taking the eigenvectors corresponding to the eigenvalue of largest magnitude (in the complex plane) and decomposing it into a pair of normalized real and imaginary vectors - yields two orthogonal vectors that span a plane. One can think of this plane as projecting the neural data onto a isolated rotational plane. Using the largest imaginary eigenvalue's eigenvector would produce projections on the plane of fastest rotation (FR plane). Similarly, the plane of second, third fastest rotation can be found (see figure 6).

For the projection on the plane of fastest rotation (leftmost in figure 6) we see better path separation, and more 'extreme' trajectories. This makes sense given the projection onto the high-angular speed plane. There is more 'rotational movement' in this plane - which is expected. Naturally, the biggest difference is the overarching curved shape of all the trajectories - this homogeneity in task paths is not so apparent in the last trajectory plot. Green paths for example, seem to follow a segment of an ellipse and can quite clearly be discerned from the red and black paths. In one way - one can expect this to be a good representation of the monkey's motor dynamics, this is how the hand is actually moving. In a semantic sense, the trajectories may well indeed line up to the curvature of the hand path as the monkey responds to different tasks.

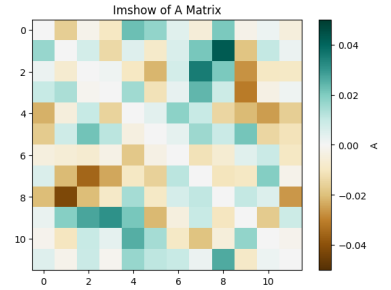


Figure 5: A matrix colormap

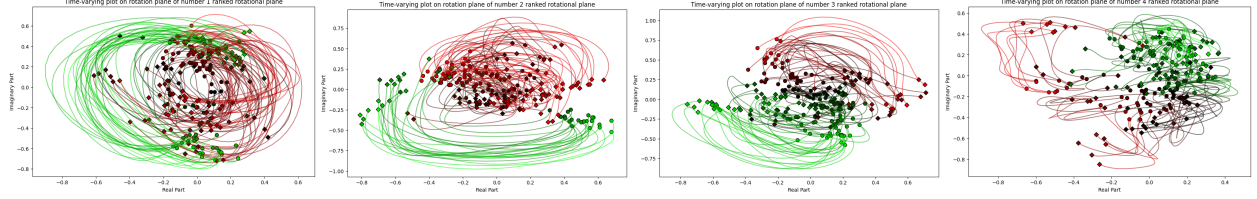


Figure 6: Projections onto the First, Second, Third, and Fourth Fastest Rotation Plane

As we filter down in the eigenvalues the structure gradually deteriorates, becoming less ellipsoidal and more stochastic. Task paths are not as discernible and do not follow a smooth rotational trajectory, and there are changes in the heading with respect to time - this is as we explore the planes for slower rotations. There could be a few explanations for the same, the faster eigenvalues dominate the matrix, the linearity of the operation means that on the faster plane they are smooth and ellipsoidal. For slower eigenvalues, these planes - while orthogonal to planes of faster rotation - are not driving the neural response, so are instead weakly rotational.

What is also key to note here is that similar colors have similar trajectories and different colors have separable trajectories - this is good evidence that this projection has captured some underlying neural data that corresponds to the monkeys motor movements. The trajectories are colored by the distribution of start points - and therefore - they roughly correspond to different tasks. So seeing the separation in this plane - task-wise - is a good indicator that the rotation dynamics have modeled the neural data well.

6 Part 6

What is also interesting to apply the PCA projection along with F_{PR} projection in the pre-movement period. In doing so - one captures what the anticipatory neural activity is presenting in the time before the movement. Figure 7 demonstrates this. A similar color-coding has been performed to distinguish pre-movement activity. It is quite clear that the pre-movement trajectories are not rotational, which supports evidence that the rotational nature of trajectories is due to condition on-set and is not the natural state of movement.

If we assume autonomous rotational dynamics one can infer the pre-movement interval as the anticipation interval. The monkey will be trained to recognize certain facts that precede the go cue. Whilst it may not be known exactly when the cue is going to be given, it knows the rough interval (one can infer that the monkey has built up an internal model of roughly how long it must wait before the go cue). The anticipation of acting could be presented as the neural activity projected on this plane gravitating towards the required task-specific framework to then execute the action. In humans, and from empirical evidence, the anticipation of executing an action or internalizing it can make one better trained to execute it better.

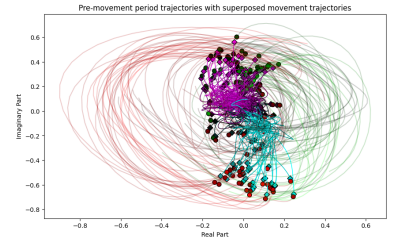


Figure 7: Pre-movement period trajectories

An interesting line of study would be to test the same monkey but periodically (or with some random probability) never give it the go cue - and record the neural activity past this pre-movement period. Whether the projected readings would remain in the same space or gradually return to the source region (roughly around 0,0 in the plane of fastest rotation) would indicate how anticipation plays into this rotational dynamics model. On this note, researchers [2] found that monkeys generally relied on a sense of elapsed time and prior probabilities on deciding when to initiate an anticipatory neural response. When the timing between on-set and the go-cue is increased, as expected, there is higher uncertainty and variability in the monkey's response.

7 Part 7

Naturally, one needs to question whether or not these dynamical models are simply over-fitted to the data (hallucinated) or are natural distillations of the latent space of the neural activity. To test the rigidity of the rotational dynamics model - the PSTH array is modified. For each neuron - at half of the task conditions (chosen at random) the firing rate is inverted along the time bin at -150 ms. This inversion, preserves continuity, but greatly increases the difference between the maximum and minimum values of the PSTH. Inverting movements for random tasks for each neuron essentially destroys the correlation between neurons (per task) and so this will corrupt latent features. In particular, the eigenvectors and values will change. This is sure to affect the rotational dynamic model if one assumes that this model underpinned neural activity. If the model was instead hallucinated and rotational trajectories were simply over-fitted, this would be invariant to the latent space correlations in the neural data.

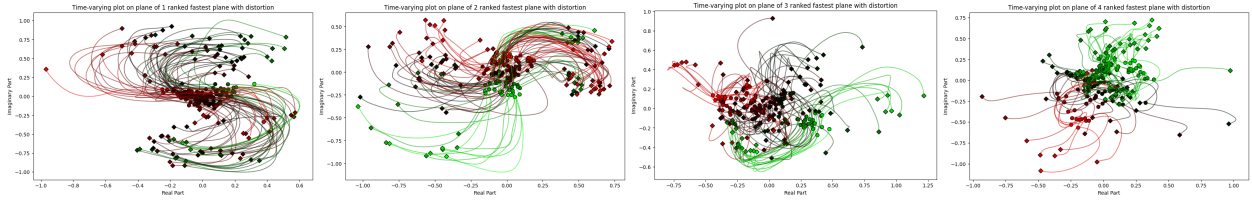


Figure 8: Trajectories on First, Second, Third, and Fourth Fastest Rotational Planes

What is interesting to observe is the fact that rotational trajectories are still present - though the trajectories themselves are far less separable. This means that even in the distorted data, the maximum likelihood estimate of the A matrix is able to discern some rotational underlying the data. And therefore the projection onto the plane of rotations still has some - albeit - weak rotational movements. The weak presence of this - is purely a qualitative statement, though it is apparent that trajectories are far less separable.

One can argue that this indicates a presence of some amount of hallucination and also some good representation of the latent model of the data. Again these can only be - so far - asserted qualitatively. Studies on rotational structures [3] have found quantitative measures for the 'strength' of presence of rotational dynamics and dubbed it a gyration number. Comprehensive studies of jPCA and other methods have concluded that these projective methods can often lead to "false positives". Another study [4] also stated: "we have concluded that developing a model of cortical dynamics based on such limited data may be an endless process, and debates about the nature of structural rotations may be unproductive". Attempts to use rotational dynamics in BCI and kinematic decoding systems have not been popularly adopted.

Yet, with all the results - one can make a good argument that rotational dynamical model is a good model for approximation. When compared to the pure MLE estimate of A, the rotational model is not too far. Computing the log-likelihood for the pure MLE estimate ($A_{MLE} = (\Delta ZZ^T)(ZZ^T)^{-1}$) yields **-6.484** while the A_{rot} matrix yields: **-7.326**. Naturally we expect it to be lower (due to the rotational a-priori assumption), but empirically the values are close, indicating strong evidence for the rotational model. Trajectory separability in the plane of fastest rotation also indicates goodness of the model. However, it is apparent that some amount of rotational trajectories are simply fitted onto data. The extent of this is not as strong as when there is no corruption, but it does indicate some amount of hallucination. The extent of this can be tested by applying this algorithm on synthetic datasets (with known underlying models) to generate a truth-table. Given the evidence with the data at hand, however, the rotational dynamics model can be said to be a strong descriptor for the underlying neural activity.

References

- [1] Basanisi, Ruggero, et al. “Beta Oscillations in Monkey Striatum Encode Reward Prediction Error Signals.” *Journal of Neuroscience*, Society for Neuroscience, 3 May 2023, doi.org/10.1523/JNEUROSCI.0952-22.2023. Accessed 20 Nov. 2024.
- [2] De Hemptinne, Coralie, et al. “How Do Primates Anticipate Uncertain Future Events?” *The Journal of Neuroscience*, vol. 27, no. 16, 18 Apr. 2007, pp. 4334–4341, doi:10.1523/jneurosci.0388-07.2007.
- [3] Kuzmina, Ekaterina, et al. “On the Rotational Structure in Neural Data.” *bioRxiv*, Cold Spring Harbor Laboratory, 1 Jan. 2023, doi.org/10.1101/2023.09.11.557230. Accessed 20 Nov. 2024.
- [4] Libby, Alexandra, and Timothy J. Buschman. “Rotational Dynamics Reduce Interference between Sensory and Memory Representations.” *Nature Neuroscience*, vol. 24, no. 5, 5 Apr. 2021, pp. 715–726, doi:10.1038/s41593-021-00821-9.