Lecture notes on linear dynamical systems for neuroscience*

Il Memming Park and Matthew Dowling

November 14, 2022

1 Motivation

Our aim is to understand complex spatio-temporal phenomena that involve the nervous system. In the physical space, interaction among entities generates collective phenomena: there are many neurons, multiple subregions of a single neuron, multiple brain areas, and even multiple brains that are socially interacting. Various measurement technologies allow access to many pixels/voxels imaged, many electrodes that record electrical activities, and many magnetic field sensors. In the time domain, neural activity of interest in neuroscience span time scales from microseconds to years. Only through time, the neural systems can process information, store information, adapt to new context, learn, and generate meaningful behavior. Both for the practical data analysis, and for the theoretical study of the neural system, we often face the grand challenge of making sense of high-dimensional spatio-temporal dynamics. If we want to "understand" how the brain works, we need some intuition on high-dimensional spatio-temporal dynamics.

This may seem daunting at first, but we can stand on the shoulders of giants who developed various conceptual tools. In addition, fortunately, anyone can learn the mathematical intuition for making precise statements on those concepts. In this week, we will learn versatile and extremely useful framework of linear dynamical systems. There are two key intuitions covered within two subjects in mathematics.

Learning Objective 1

Intuition on high-dimensional spaces can be gained by first studying **linear algebra**.

Learning Objective 2

Intuition on lawful temporal changes can be gained by first studying **linear dynamical systems**.

Learning them benefits you because of their immense practicality. For example, data analysis methods such as principal components analysis (PCA), filtering, fourier transform, and linear regression are much faster than corresponding nonlinear extensions. The recent boost in deep neural networks is fueled by the fast linear operations implemented on modern computer architectures (GPGPU-computing).

Saves you time and the planet

Linear systems are computationally *fast* and can be made numerically stable.

As we will see, with linear dynamical systems theory, we can understand when systems shows stable behavior, and how quickly it forgets the past. It may be surprising that there are a finite categories of linear dynamical systems if we are only concerned with their asymptotic behavior.

*Available online: `https://github.com/memming/linear-algebra-and-dynamics-lectures`

Easy asymptotic theory

We can understand *all* possible (finite-dimensional) linear dynamical systems.

This allows us to theoretically understand roughly what is going to happen in systems at longer time scales thanks to powerful first-order approximation techniques.

Linearize (locally)

Even complex nonlinear systems usually have a meaningful linear component.

Of course, there are exceptions and limitations, however, there are extensions and generalizations that typically incorporates some form of linearity.

2 Discrete-time Linear Dynamical System

In most aspects discrete-time and continuous-time linear dynamical systems are analogous. Continuous-time systems require more abstract mathematical skills, namely calculus and its friends, while discrete-time systems have lower barriers since solutions can be straightforwardly computed. Discrete-time systems can be implemented with simple loops on computers. A simple *Euler integration* scheme "tightly" connects the two approaches, and allows most of the practical intuitions to be transferred.

$$\frac{\mathrm{d}x}{\mathrm{d}t} = f(x) \quad \Longleftrightarrow \quad x(t+\Delta) = x(t) + \Delta \cdot f(x) \tag{1}$$

for sufficiently small Δ. We will implicitly take discrete time steps of size Δ. Now, let us dive into discrete-time systems!

2.1 Memory traces in a 1-dimensional linear dynamical system

Consider an abstract leaky membrane potential model:

$$v(t+1) = a \cdot v(t) + I(t) \tag{2}$$

where $t = 0, 1, \ldots$ represents discrete time steps, $v(t) \in \mathbb{R}$ is the membrane potential, $a \in \mathbb{R}$ is a constant, and $I(t) \in \mathbb{R}$ is a current entering the neuron.

1. Given an initial condition $v(0)$ and the external current $I(t)$ for $t = 0, 1, \ldots$, write the general form of $v(t)$ using the summation notation (e.g. $\sum_{s=0}^{t}$). (Hint: try writing out $v(1), v(2), v(3)$ first and generalize.)

2. Sketch the solution over time from $t = 0$ to $t = 10$ for $a = 0.9$, $a = 1$, and $a = 2$ given $v(0) = 0$, $I(2) = 1$, and $I(t) = 0$ otherwise. Which one resembles the behavior of a neuronal membrane?

3. What is the value of a for an input pulse $I(0) \neq 0$ to decay to 10% of it original magnitude in 10 time steps (assume $v(0) = 0$ and $I(t > 0) = 0$)?

2.2 Linear network model

As we shall see later when we go to continuous time, $0 < a < 1$ is related to the time constant. The membrane potential time constant of a neuron is not very flexible; typically in the order of 10 milliseconds. Having many independent neurons does not help with the situation of forgetful membrane. But maybe we can create a network of (non-spiking) neurons that has more memory!

Consider a network of n linear neurons where each neuron's activity (analogous to membrane potential or firing rate) $x_i(t)$ evolves over time $t = 0, 1, 2, \ldots$ as,

$$x_i(t+1) = w_{i,i}x_i(t) + \sum_{j \neq i} w_{i,j}x_j(t) + b_i I(t) \tag{3}$$

where $w_{i,i} \in \mathbb{R}$ plays the role of a in (2), and $w_{i,j}$ is the influence of j-th neuron's activity directly onto i-th neuron's activity. Note that there is no synaptic dynamics in this super simplified model. Despite its simplicity, it is no doubt a very useful theoretical model of neural dynamics [2–4].

Let's write this in matrix form. Define the neural activity vector $\mathbf{x}(t)$, the connectivity matrix W, and the input gain vector $\mathbf{b}$.

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} \qquad W = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,n} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n,1} & w_{n,2} & \cdots & w_{n,n} \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{4}$$

Note that $W \in \mathbb{R}^{n \times n}$ is a square matrix, and $\mathbf{x}(t), \mathbf{b} \in \mathbb{R}^{n \times 1}$ are column vectors.

4. Write the matrix form difference equation for (3).

5. Write out the general form for $\mathbf{x}(t)$ using matrix power.

Sure, these are solutions, but not in a very transparent form. We desire more insights! First, let us recall some linear algebra to help us.

Matrix multiplication

Let A be an $n \times k$ matrix, and B be an $k \times m$ matrix. The i-th row and j-th column of A is denoted $A_{i,j}$. The matrix product $C = AB$ is $n \times m$, and its entries are given by,

$$\underbrace{\mathbf{C}}_{(n\times m)} = \underbrace{\mathbf{A}}_{(n\times k)} \underbrace{\mathbf{B}}_{(k\times m)} \qquad C_{i,j} = (AB)_{i,j} = \sum_k A_{i,k}B_{k,j} \tag{5}$$

Note that matrix products don't commute, i.e., $AB \neq BA$.

What does *matrix multiplication* do? Let us try out with some special matrices.

2.3 Some special matrices as connectivity

Diagonal matrices are non-zero only on the diagonal entries.

$$\text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n) = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & \lambda_n \end{bmatrix} \tag{6}$$

6. If $W = \text{diag}(\lambda_1, \ldots, \lambda_n)$, what is the general form for $\mathbf{x}(t)$?

Identity matrices are **diagonal matrices** with ones on the diagonal (zero otherwise):

$$I = \text{diag}(1, 1, \ldots, 1) = \begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix} \tag{7}$$

7. If $W = I$, what is the general form for $\mathbf{x}(t)$?

Cross diagonal matrices are non-zero only on the anti-diagonal entries.

$$C = \begin{bmatrix} 0 & \cdots & 0 & c_1 \\ \vdots & & \ddots & \vdots \\ 0 & c_{n-1} & \cdots & 0 \\ c_n & 0 & \cdots & 0 \end{bmatrix} \tag{8}$$

8. What is the general form for $\mathbf{x}(t)$ for the following cross-diagonal weight matrix?

$$W = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{9}$$

Strictly upper triangular matrices have the following zero entry pattern (X marks arbitrary value, example only shown in 3×3):

$$\begin{bmatrix} 0 & X & X \\ 0 & 0 & X \\ 0 & 0 & 0 \end{bmatrix} \tag{10}$$

9. What is the general form for $\mathbf{x}(t)$ for the following strictly upper triangular weight matrix?

$$W = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \tag{11}$$

10. Since (3) is causal $\mathbf{x}(t)$ only depends on the past input, i.e., $I(s)$ for $s < t$. How much memory of the past does $\mathbf{x}(t)$ have?
    (a) for (11)

    (b) for (7)

2.4 Orthonormal and Unitary Matrices

Recall that the Euclidean vector space $\mathbb{R}^n$ can be spanned by n linearly independent basis vectors. In particular, one can choose a set of unit length vectors that are orthogonal to each other. We can write this in linear algebra terms.

Transpose of a matrix (or vector) is defined as $(A^\top)_{i,j} = A_{j,i}$.

Two vectors $\mathbf{u}$ and $\mathbf{v}$ are **orthogonal** if $\mathbf{u}^\top \mathbf{v} = 0$.

$\mathbf{v} \in \mathbb{R}^{n \times 1}$ is a unit vector if $\|\mathbf{v}\| = \sqrt{\mathbf{v}^\top \mathbf{v}} = 1$, i.e., it is of unit length.

Let the collection of vectors $(\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n)$ form an *orthonormal basis* of $\mathbb{R}^n$, that is, (1) any vector $\mathbf{x} \in \mathbb{R}^{n \times 1}$ can be expressed as $\mathbf{x} = \sum_{i=1}^n \alpha_i \mathbf{u}_i$ where α_i are scalar, (2) if $i \neq j$, then $\mathbf{u}_i$ and $\mathbf{u}_j$ are orthogonal, and (3) $\mathbf{u}_i$ is a unit vector.

An **orthonormal matrix** is a real-valued square matrix where the collection of column vectors form an orthonormal basis.

$$U = \begin{bmatrix} | & & | \\ \mathbf{u}_1 & \cdots & \mathbf{u}_n \\ | & & | \end{bmatrix} \in \mathbb{R}^{n \times n} \tag{12}$$

11. Show that if U is orthonormal then $U^\top U = I$.

> **Fun Fact:** For square matrices A and B, if $AB = I$ then $BA = I$. B is the *matrix inverse* of A, that is, $B = A^{-1}$. (They are inverses of each other.)

12. Show that if U is orthonormal then $UU^\top = I$.

A complex-valued square matrix $U \in \mathbb{C}^{n\times n}$ is **unitary**, if $UU^\dagger = I$ where † denotes conjugate transpose operation. Orthonormal matrices are unitary.

13. Show that multiplying a unitary matrix to a column vector preserves its norm, i.e., it is an *isometric* transformation.

14. Let W in (3) be orthonormal. In the absence of input, i.e., $I(t) = 0$, what can you say about $\mathbf{x}(t)$ given $\mathbf{x}(0)$? (geometrically)

A **permutation matrix** is a matrix where each row and column has a single unity and zero otherwise. (Why is it called a permutation matrix?)

15. Show that a permutation matrix is a unitary matrix.

16. The following **rotation matrix** is a unitary matrix. Multiplying vectors in $\mathbb{R}^2$ results in a counter-clock rotation by θ.

$$W = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{13}$$

(a) Sketch out the general form of $\mathbf{x}(t)$ when $I(t) = 0$ and $\theta = \pi/4$.

(b) How is the dynamics qualitatively different when θ is a rational multiple of π or not?

17. Orthonormal matrices can also "flip" or reflect vectors with respect to axes. Consider

$$W = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{14}$$

(a) Show that W is unitary.

(b) What is the general form of $\mathbf{x}(t)$ when $I(t) = 0$?

18. Let U be a unitary matrix. Show that unitary matrix preserves the inner product structure, i.e., $\mathbf{x}^\top \mathbf{y} = (U\mathbf{x})^\top (U\mathbf{y})$.

Unitary or Orthonormal Matrix

It can rotate, permute, and reflect vectors while preserving their norm and inner product structure.

2.5 Spectral Decomposition

Now we reach the first and arguably one of the most important matrix decompositions.

A matrix is **symmetric** if $A = A^\top$. A complex matrix is **Hermitian** (or self-adjoint) if $A = A^\dagger$. Note that only square matrices can be symmetric or Hermitian.

Spectral theorem for Hermitian matrices

Any *real symmetric square (or Hermitian) matrix* A can be decomposed into the following form,

$$A = U\Lambda U^\dagger \qquad \text{(spectral decomposition)} \tag{15}$$

where $U = \begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_n\end{bmatrix}$ is an orthonormal (or unitary) matrix, and Λ is a *real*-valued diagonal matrix.

19. Let the connectivity matrix W be real symmetric. This means every pair of neurons in the network reciprocally activates each other with the same strength. Use the spectral decomposition (19) to answer the following questions.

    (a) What is WW?

    (b) What is W^n?

    (c) What is the general form of $\mathbf{u}_i{}^\top \mathbf{x}(t)$ in the absence of input ($I(t) = 0$) and with initial condition $\mathbf{x}(0) = \mathbf{u}_i$?

    (d) What is the general form of $\mathbf{x}(t)$ in the absence of input ($I(t) = 0$)?

    (e) Can linear neural networks with symmetric weight matrices oscillate without input?

    (f) What is the general form of $\mathbf{x}(t)$ with input?

Eigenvalues and eigenvectors

Given a matirx A, solutions to the following eigenvalue equation

$$A\mathbf{x}_i = \lambda_i \mathbf{x}_i, \tag{16}$$

are called the (right) eigenvectors ($\mathbf{x}_i$) and eigenvalues (λ_i) of A.

20. Given a spectral decomposition of $A = U\Lambda U^\dagger$, show that $\mathbf{u}_i$ is an eigenvector with corresponding eigenvalue $\Lambda_{i,i}$ of A.

As you have just seen, the spectral decomposition makes things easy. But are there other matrices that also allow such decomposition?

Normal matrices

A matrix W is called **normal**, if it commutes with its (conjugate)-transpose.

$$W^\dagger W = W W^\dagger$$

21. Determine if the following non-symmetric matrices are normal.

(a)

$$\begin{bmatrix} 0 & 2 \\ -2 & 0 \end{bmatrix} \tag{17}$$

(b)

$$\begin{bmatrix} 2 & 1 \\ -1 & 0 \end{bmatrix} \tag{18}$$

The class of normal matrices is pretty big, and includes diagonal, unitary, symmetric, and skew-symmetric ($A = -A^\top$) matrices. Normal matrices are exactly those that have *a complete set of* orthonormal *eigenvectors*. The spectral theorem extends to the class of normal matrices, but the eigenvalues may be complex-valued.

Spectral theorem for Normal matrices

Any *normal matrix* A can be decomposed into the following form,

$$A = U\Lambda U^\dagger \qquad \text{(spectral decomposition)} \tag{19}$$

where U is a unitary matrix, and Λ is a *complex*-valued diagonal matrix.

22. Show that if a normal matrix W is real-valued, then its complex eigenvalues come in complex conjugate pairs.

23. Show that the eigenvalues of a rotation matrix are of the form $e^{\pm i\theta} = \cos(\theta) \pm i\sin(\theta)$.

$$W_\theta = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \quad U = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ i & -i \end{bmatrix} \tag{20}$$

24. Show that if A is normal, so is $A + \sigma I$ for any $\sigma \in \mathbb{C}$.

In some sense, linear dynamics with normal matrices are easy to understand now. In the following section, we will attempt to understand the other cases.

2.6 Singular Value Decomposition

Another very powerful matrix decomposition that applies to any matrix is the SVD.

SVD

Any rectangular matrix $A \in \mathbb{C}^{m\times n}$ can be factored into the following form,

$$A = U\Sigma V^\dagger \qquad \text{(singular value decomposition)} \tag{21}$$

where $U \in \mathbb{C}^{m\times m}$ is a unitary matrix, $\Sigma \in \mathbb{R}^{m\times n}$ is diagonal matrix with non-negative values and $V \in \mathbb{C}^{n\times n}$ is a unitary matrix.

The diagonal entries in Σ are called the **singular values** of A, and the columns of U and V are called the left and right singular vectors. SVD provides a nice interpretation of matrix multiplication in general as a sequence of unitary transformation, scaling, and another (potentially different) unitary transformation.

25. Show the relationship between singular vectors and singular values of A and the eigenvectors of the square matrices $AA^\dagger$ and $A^\dagger A$.

26. Find a singular value decomposition of the following square matrix:

$$W_s = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{22}$$

27. Write the general form of $\mathbf{x}(t)$ in the absence of input using the SVD of W. Do the dynamics decouple into nice independent modes?

Rank of a matrix is the number of non-zero singular values.

28. Consider a rank-1 neural network with $W = \mathbf{u}\mathbf{v}^\top$. (Why is it rank-1?)

$$\mathbf{u} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad W = \tag{23}$$

Describe its time evolution $\mathbf{x}(t)$.

Frobenius norm of a matrix A is defined as $\|A\|_F = \sqrt{\sum_{i,j} |A_{i,j}|^2}$.

> **Low-rank approximation (Eckart–Young) theorem (practical):** The best low-rank approximation of a matrix (in terms of Frobenius norm) is given by dropping the smallest singular values.

29. According to the Eckart–Young theorem, replacing the smallest singular values with 0 is a good way to achieve low-rank approximation of a matrix. For example, SVD of the following matrix shows two rank-1 components.

$$W_c = \begin{bmatrix} 2 & 2 & 0 \\ 2 & 2 & 0 \\ 0 & 0 & 0.5 \end{bmatrix} = 4 \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \tag{24}$$

$$= 4 \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 1/2 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{25}$$

The best rank-1 approximation is the component with singular value 4.

$$\tilde{W}_c = \begin{bmatrix} 2 & 2 & 0 \\ 2 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{26}$$

Compare the two linear neural networks with weight matrix W_c and $\tilde{W}_c$, correspondingly.

2.7 Reading assignment for Day 1

Read Goldman [4] with the following questions in mind:

30. What is the form of recurrent weight matrix corresponding to the sequential activation network in Fig 1? Consider the problem in discrete time (3).

31. Take an orthonormal matrix U and left multiply to the weight matrix obtained from above. Can you rewrite the dynamics equation in a new coordiate system $\mathbf{y}(t) = U^\top \mathbf{x}(t)$? How does this relate to Fig 3C?

32. How does the eigenvector represent feedback only onto themselves and do not interact with each other?

2.8 Matrix Algebra Cheatsheet

Fun Fact: Every linear transformation is a matrix multiplication.

Matrix Fun Facts

Matrix multiplication: Vector inner product and outer products are special cases:

$$\mathbf{u}^\top \mathbf{v} = \begin{bmatrix} u_1, u_2, \cdots, u_n \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = \sum_k u_k v_k \qquad \textit{inner product} \tag{27}$$

$$\mathbf{u}\mathbf{v}^\top = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \begin{bmatrix} v_1, v_2, \cdots, v_m \end{bmatrix} = \begin{bmatrix} u_1 v_1 & u_1 v_2 & \cdots & u_1 v_m \\ u_2 v_1 & u_2 v_2 & \cdots & u_2 v_m \\ \vdots & \vdots & \ddots & \vdots \\ u_n v_1 & u_n v_2 & \cdots & u_n v_m \end{bmatrix} \qquad \textit{outer product} \tag{28}$$

Matrix outer product stacks:

$$\begin{bmatrix} | \\ \mathbf{u}_1 \\ | \end{bmatrix} \begin{bmatrix} — \mathbf{v}_1 — \end{bmatrix} + \begin{bmatrix} | \\ \mathbf{u}_2 \\ | \end{bmatrix} \begin{bmatrix} — \mathbf{v}_2 — \end{bmatrix} = \begin{bmatrix} | & | \\ \mathbf{u}_1 & \mathbf{u}_2 \\ | & | \end{bmatrix} \begin{bmatrix} — \mathbf{v}_1 — \\ — \mathbf{v}_2 — \end{bmatrix} \tag{29}$$

Matrix times diagonal matrix scales column vectors:

$$\begin{bmatrix} | & & | \\ \mathbf{u}_1 & \cdots & \mathbf{u}_n \\ | & & | \end{bmatrix} \cdot \begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_n \end{bmatrix} = \begin{bmatrix} | & & | \\ d_1 \mathbf{u}_1 & \cdots & d_n \mathbf{u}_n \\ | & & | \end{bmatrix} \tag{30}$$

$$\begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_n \end{bmatrix} \cdot \begin{bmatrix} — \mathbf{v}_1 — \\ \vdots \\ — \mathbf{v}_n — \end{bmatrix} = \begin{bmatrix} — d_1 \mathbf{v}_1 — \\ \vdots \\ — d_n \mathbf{v}_n — \end{bmatrix} \tag{31}$$

Fun Fact: $(AB)^\top = B^\top A^\top$.

Acknowledgements

Special thanks to Ayesha Vermani and Ábel Ságodi.

References

[1] Carmen Chicone. *Ordinary Differential Equations with Applications*. Springer Science & Business Media, September 2006. ISBN 9780387357942.

[2] Shaul Druckmann and Dmitri B Chklovskii. Neuronal circuits underlying persistent representations despite time varying activity. *Current biology: CB*, 22(22):2095–2103, November 2012. ISSN 0960-9822, 1879-0445. doi: 10.1016/j.cub.2012.08.058.

[3] Surya Ganguli, Dongsung Huh, and Haim Sompolinsky. Memory traces in dynamical systems. *Proceedings of the National Academy of Sciences*, 105(48):18970–18975, December 2008. ISSN 1091-6490. doi: 10.1073/pnas.0804451105.

[4] Mark S Goldman. Memory without feedback in a neural network. *Neuron*, 61(4):621–634, February 2009. ISSN 0896-6273, 1097-4199. doi: 10.1016/j.neuron.2008.12.012.

[5] Scott Linderman, Matthew Johnson, Andrew Miller, Ryan Adams, David Blei, and Liam Paninski. Bayesian Learning and Inference in Recurrent Switching Linear Dynamical Systems. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 914–922, Fort Lauderdale, FL, USA, 2017. PMLR.

[6] Josue Nassar, Scott W. Linderman, Monica Bugallo, and Il Memming Park. Tree-structured recurrent switching linear dynamical systems for multi-scale modeling. In *International Conference on Learning Representations (ICLR)*, November 2019.

[7] Yuan Zhao and Il Memming Park. Interpretable nonlinear dynamic modeling of neural trajectories. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.