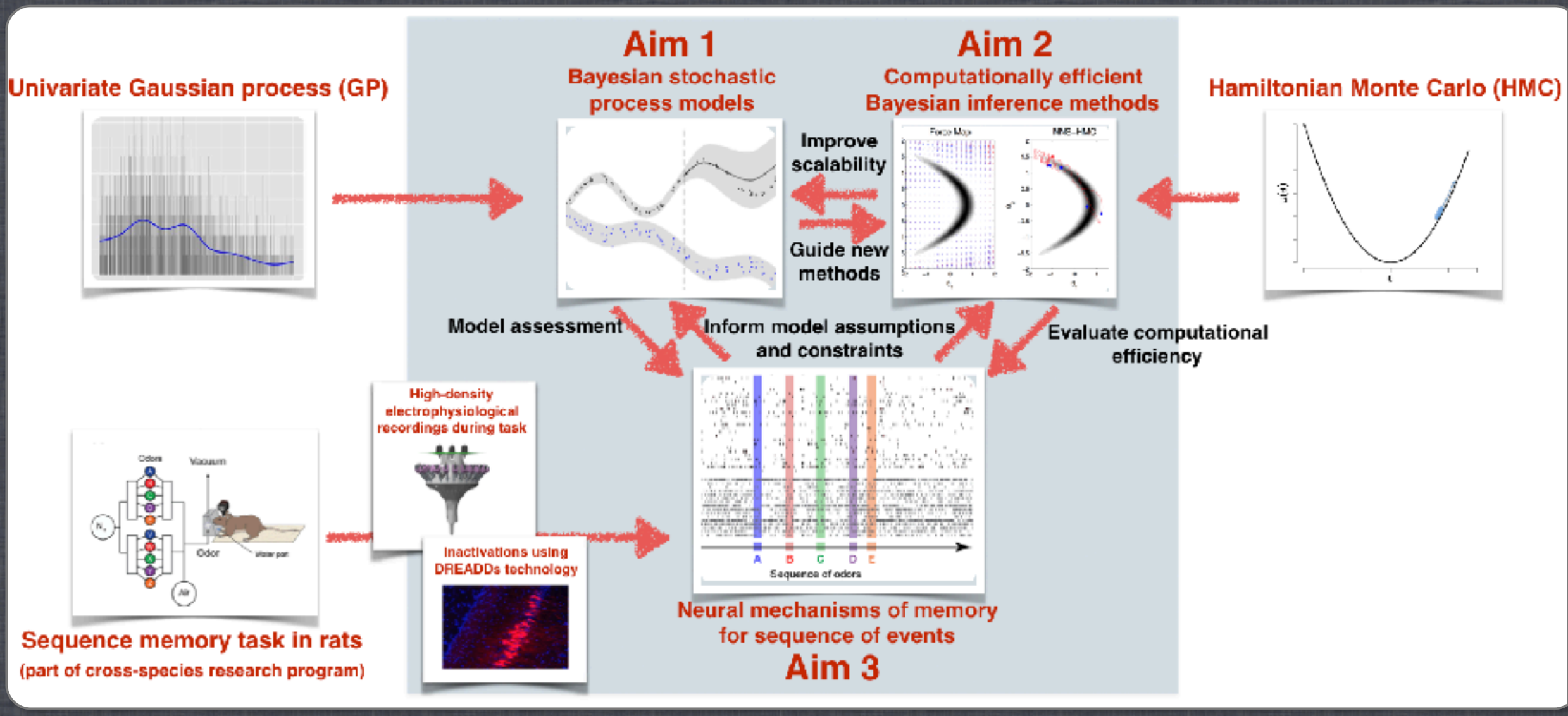
LATENT REPRESENTATION OF NEURAL DATA



Babak Shahbaba, PhD

Departments of Statistics and Computer Science

UC Irvine

IIS 2123366

R01MH115697

Outline

1. Neurophysiological experiment
2. Latent representation learning
3. Latent variable modeling
4. Future directions

Experiment

Memory for Sequence of Events

The memory for specific events is intrinsically tied to the **context** in which they occur, especially the spatial and temporal context

(Tulving, 1972; Griffiths et al.,1999; Eichenbaum & Fortin, 2005; Dere et al., 2006; Crystal, 2009; Eacott & Easton, 2009; Rattenborg & Martinez-Gonzalez, 2011; Allen & Fortin, 2013,…)

For example...

Event "**A**"

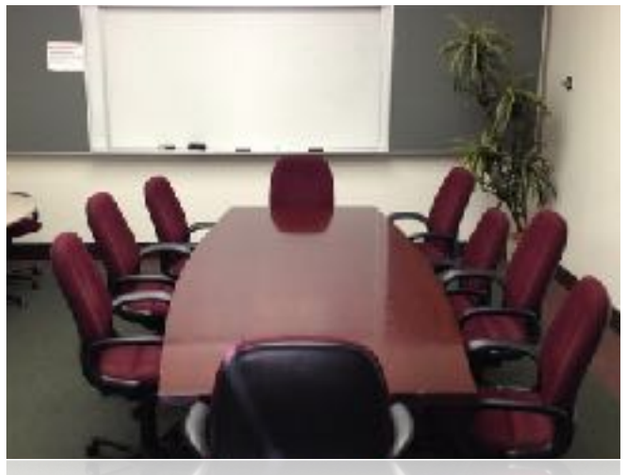
Catching up with emails

→

Event "**B**"

Getting some coffee

→



Event "**C**"

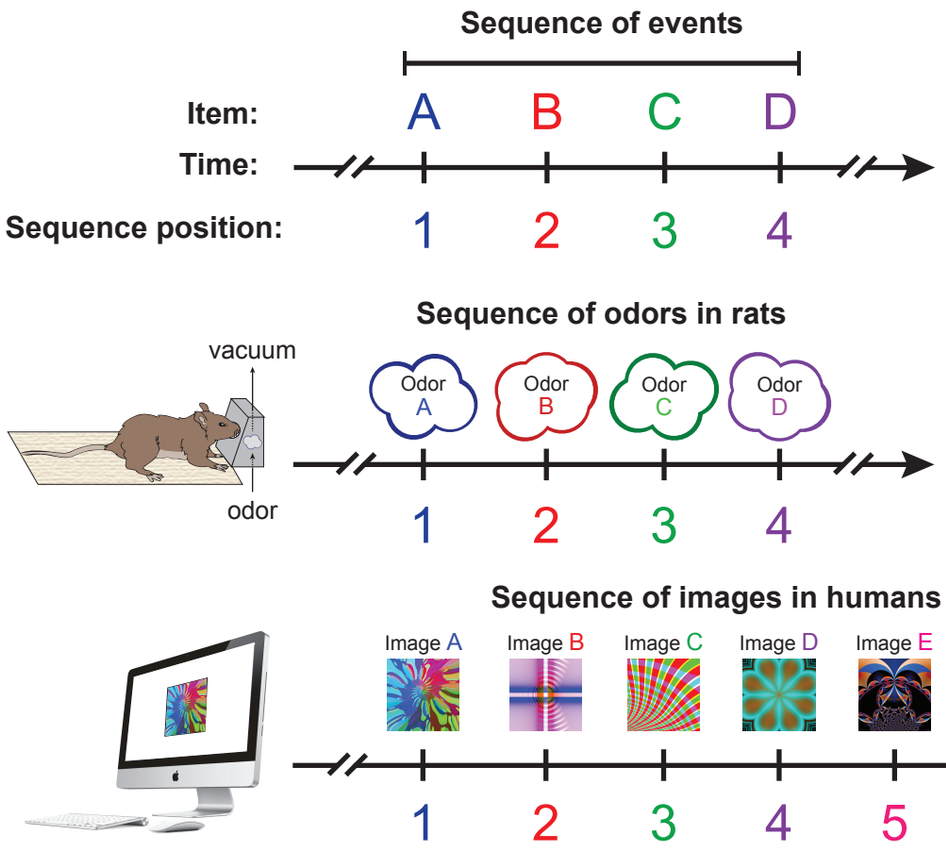
Going to a meeting

Our main focus is on the memory for sequences of events

Memory for Sequence of Events

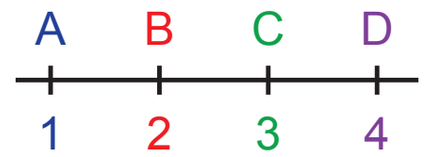
For each item in the sequence, subjects have to determine whether the item is presented “In Sequence” (InSeq) or “Out of Sequence” (OutSeq)

Task design



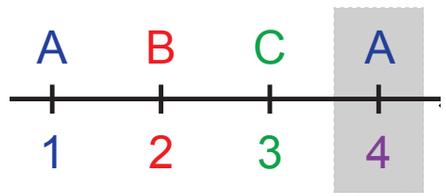
Each sequence is repeated many times during a session

Most of the time, all items are “In Sequence” (**InSeq**)

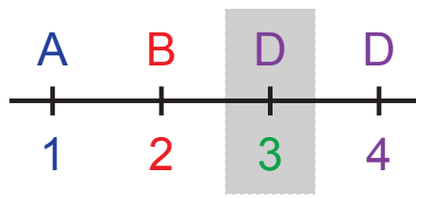


Occasionally, one item is “Out of Sequence” (**OutSeq**)

“Repeats”



“Skips”



If item is **InSeq** ⟶ **Hold** until signal (>1s)

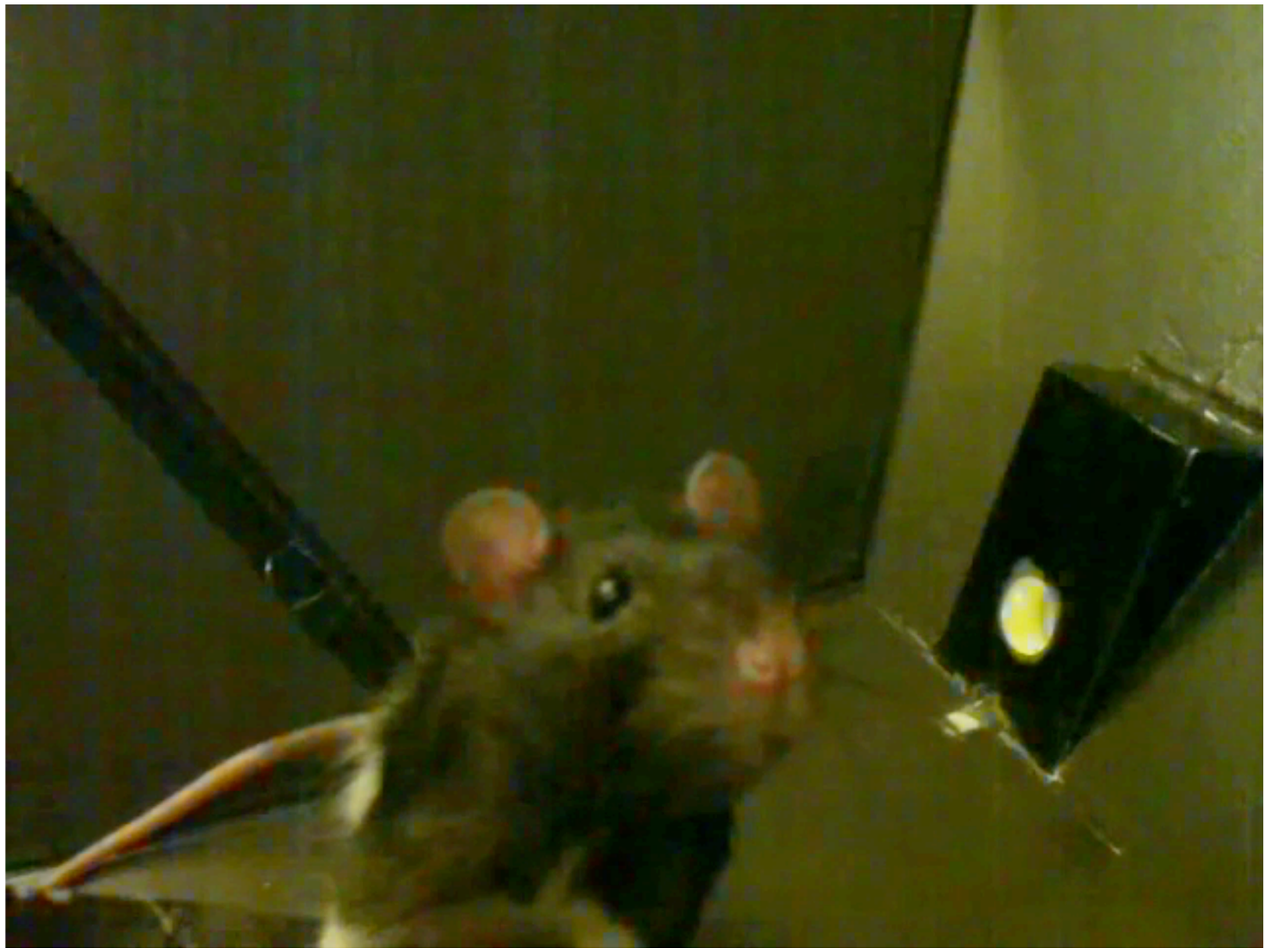
If item is **OutSeq** ⟶ **Withdraw** before signal (<1s)

Allen, Morris, Mattfeld, Stark & Fortin (2014) *Hippocampus*

Memory for Sequence of Events

In each session, rats are tested on a given sequence of 4 odors

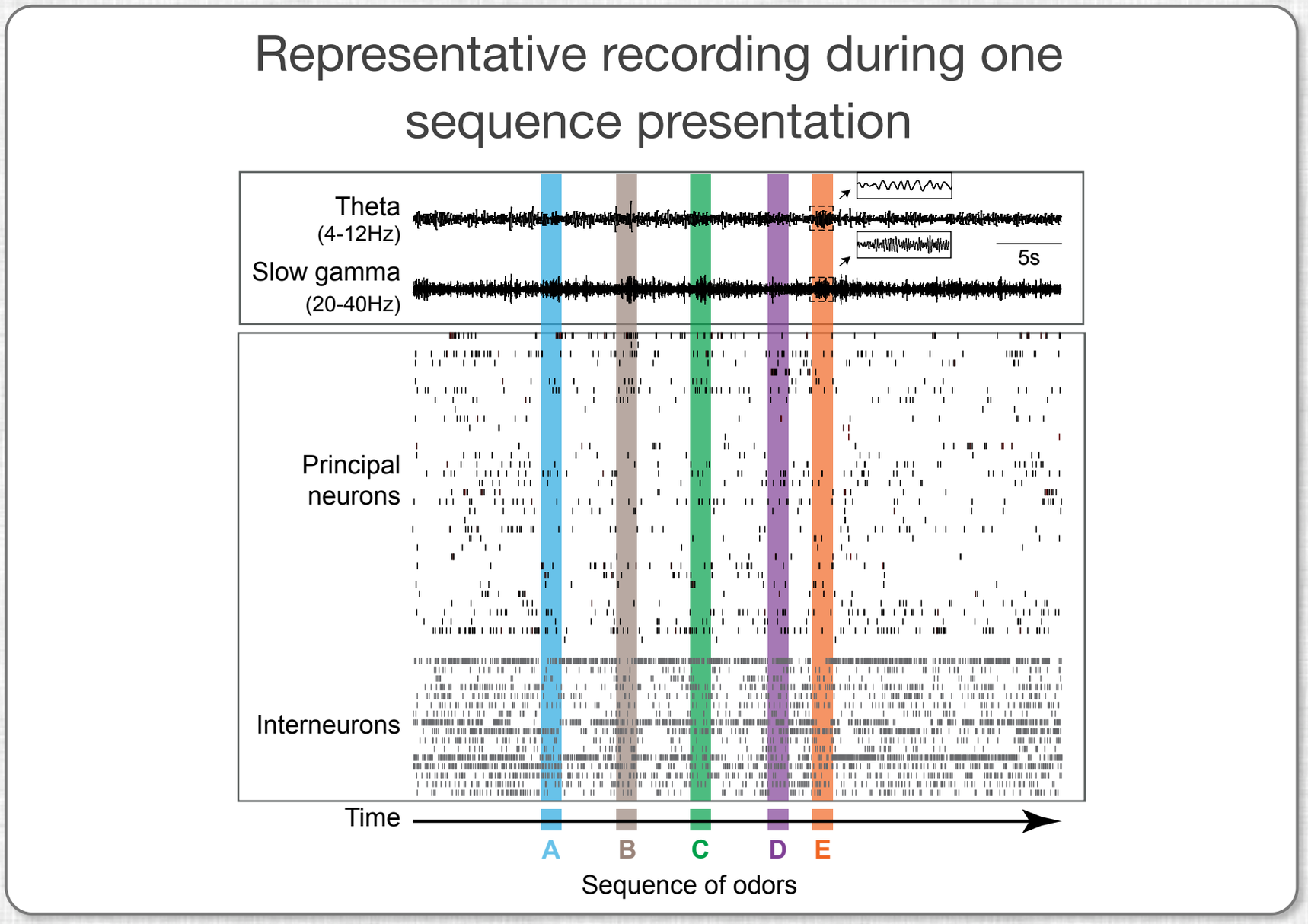
Sequence 1: A → B → C → D



Allen, Morris, Mattfeld, Stark & Fortin (2014) *Hippocampus*

Hipocampal Activity

We focus on a session consisted of **218** trials lasting anywhere from 0.48 to 1.74 seconds each. For each trial the data include LFP signals from **12** tetrodes and spike counts from **52** neurons.



Scientific Questions

- We want to answer the following scientific questions:
  - When exactly the rat identifies the odor and decides whether it is in the right sequence?
  - Can we decode the rat's response from the neuronal data and visualize the decoding process?
  - How the underlying neural process changes between InSeq and OutSeq trials?
  - Can we demonstrate nonspatial forms of sequence reactivation?
- We try to answer these questions through a set of *exploratory* tools based on **Deep Learning** algorithms and *scalable inferential* methods using **latent variable modeling**.

Latent Representation Learning

Generalized Linear Models (GLMs)

$$\mathbf{x}_i \in \mathbb{R}^d, \mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}$$

Link Function

$$g^{-1}(\mathbf{x}_i^T \mathbf{w} + b) = \mathbb{E}[\mathbf{y}_i | \mathbf{x}_i]$$

1- Notes are provided by Eric Nalisnick

Generalized Linear Models (GLMs)

$$\mathbf{x}_i \in \mathbb{R}^d, \mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}$$

Link Function

$$g^{-1}(\mathbf{x}_i^T \mathbf{w} + b) = \mathbb{E}[\mathbf{y}_i | \mathbf{x}_i]$$

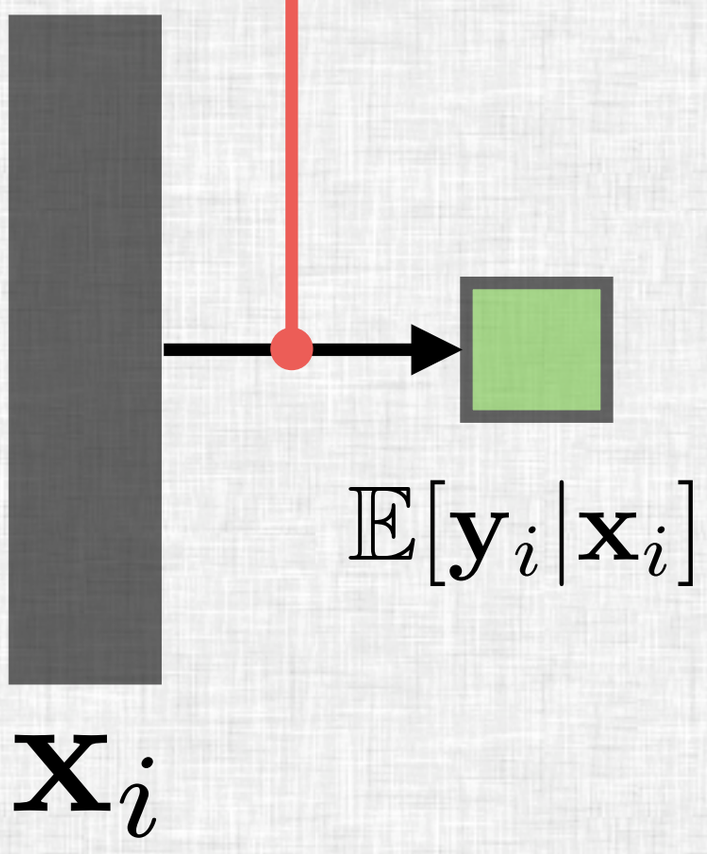
Example: Logistic Regression

$$\mathbf{y}_i \in \{0, 1\} \qquad g^{-1}(z) = \frac{1}{1 + e^{-z}}$$

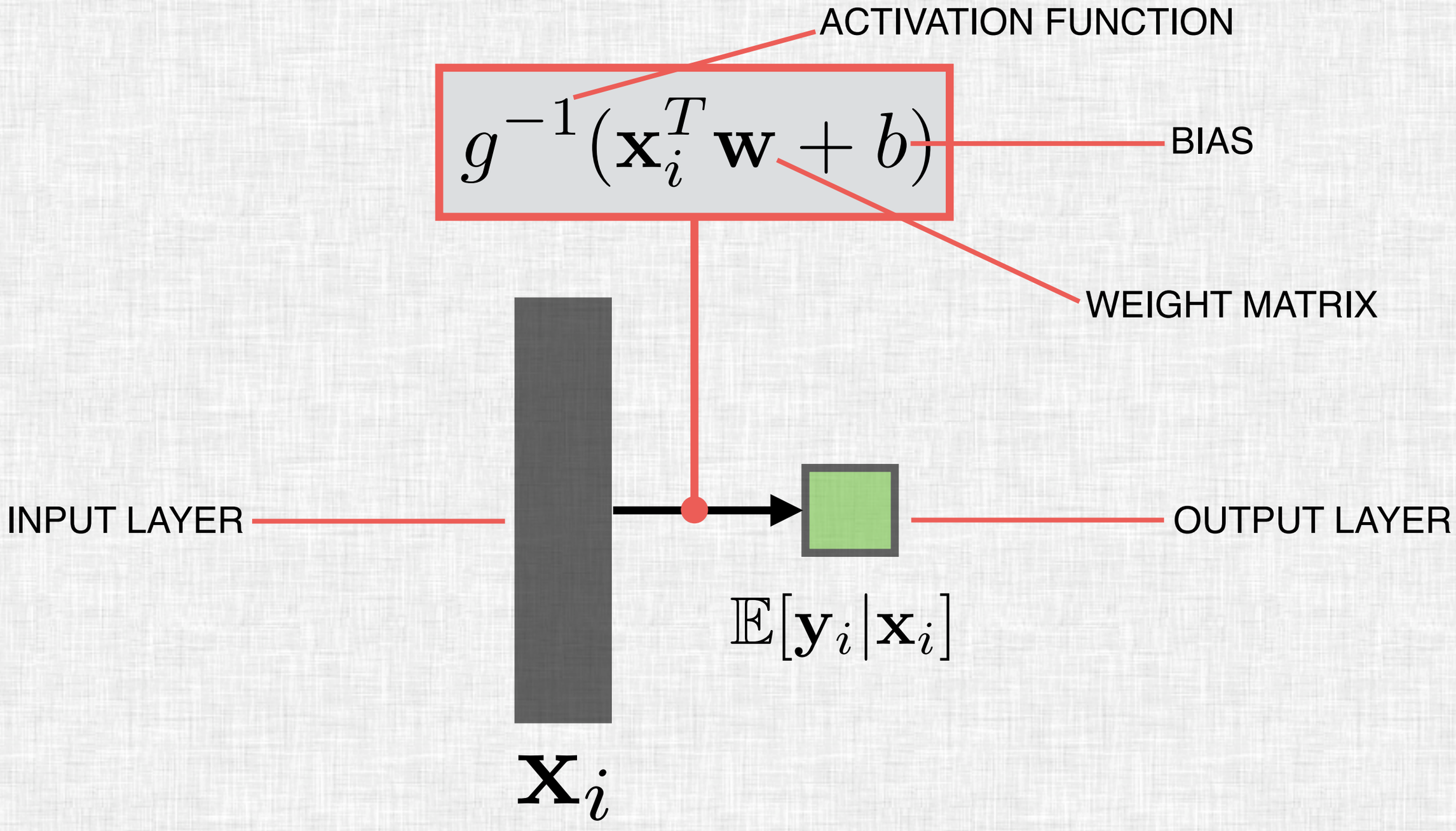
THE LOGISTIC FUNCTION

GLM as a simple neural network model

$$g^{-1}(\mathbf{x}_i^T \mathbf{w} + b)$$



GLM as a simple neural network model



Recursive GLMs

Define a latent feature via a GLM:

$$g_1^{-1}(\mathbf{x}_i^T \mathbf{w}_1 + b_1) = h_i$$

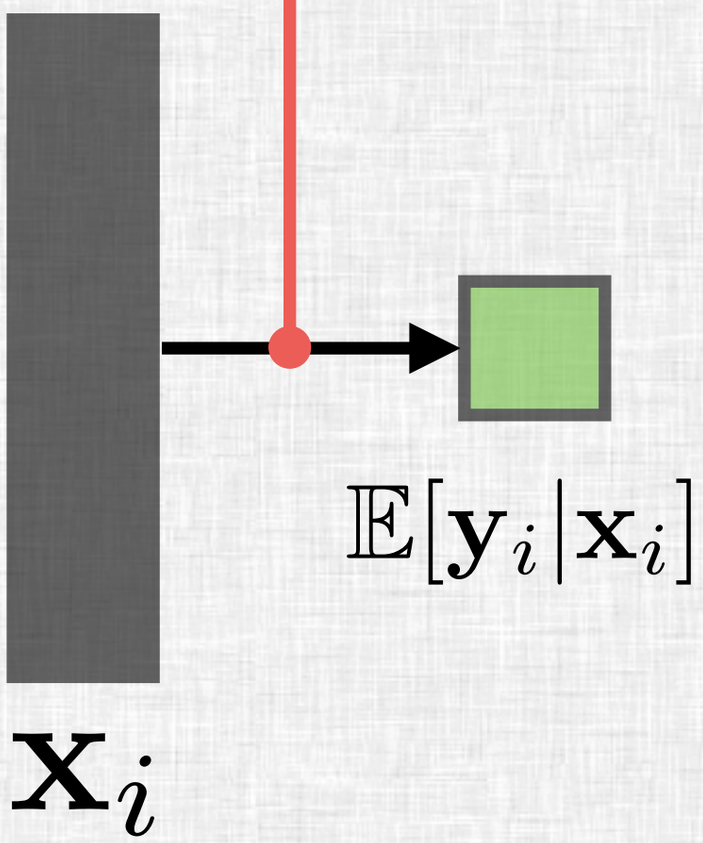
Define GLM on latent feature:

$$g_2^{-1}(h_i w_2 + b_2) = \mathbb{E}[\mathbf{y}_i | \mathbf{x}_i]$$

$$w_2 \in \mathbb{R}, b_2 \in \mathbb{R}$$

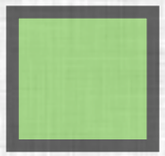
A Brief Overview of Deep Learning

$$g^{-1}(\mathbf{x}_i^T \mathbf{w} + b)$$



A Brief Overview of Deep Learning

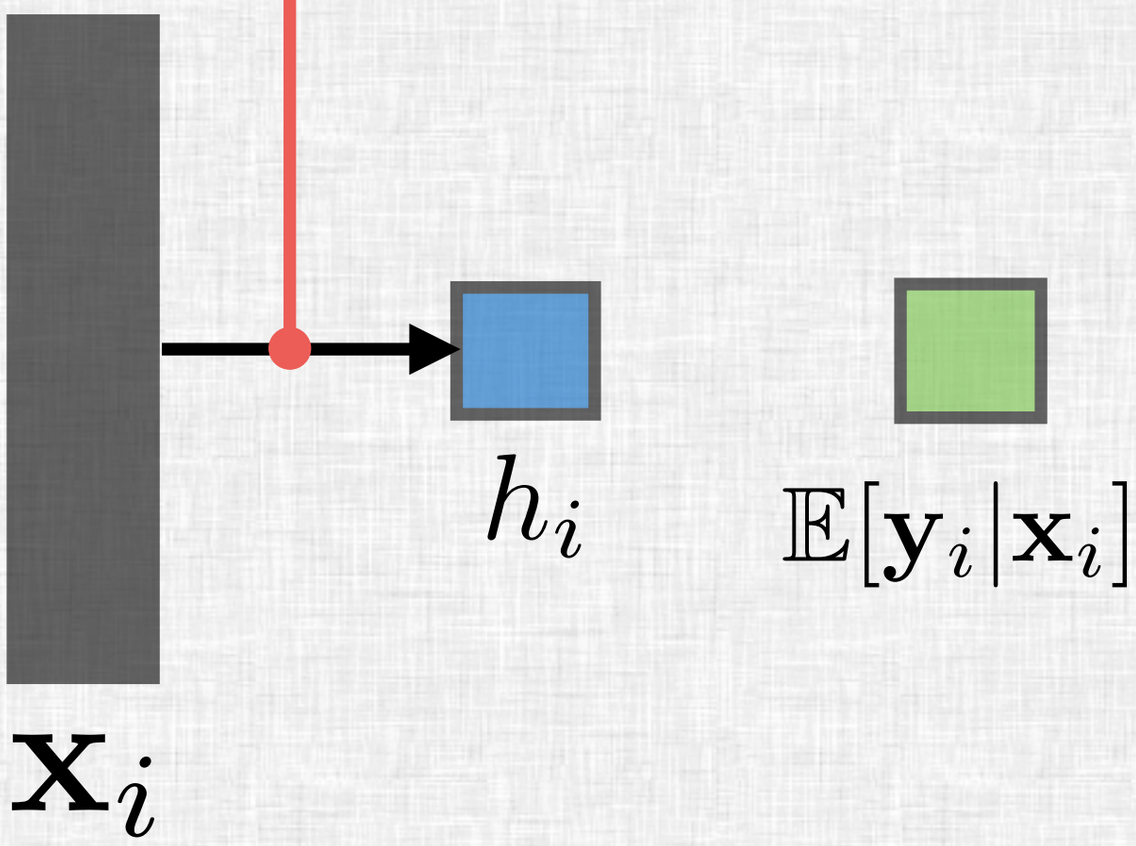
$\mathbf{x}_i$



$\mathbb{E}[\mathbf{y}_i|\mathbf{x}_i]$

A Brief Overview of Deep Learning

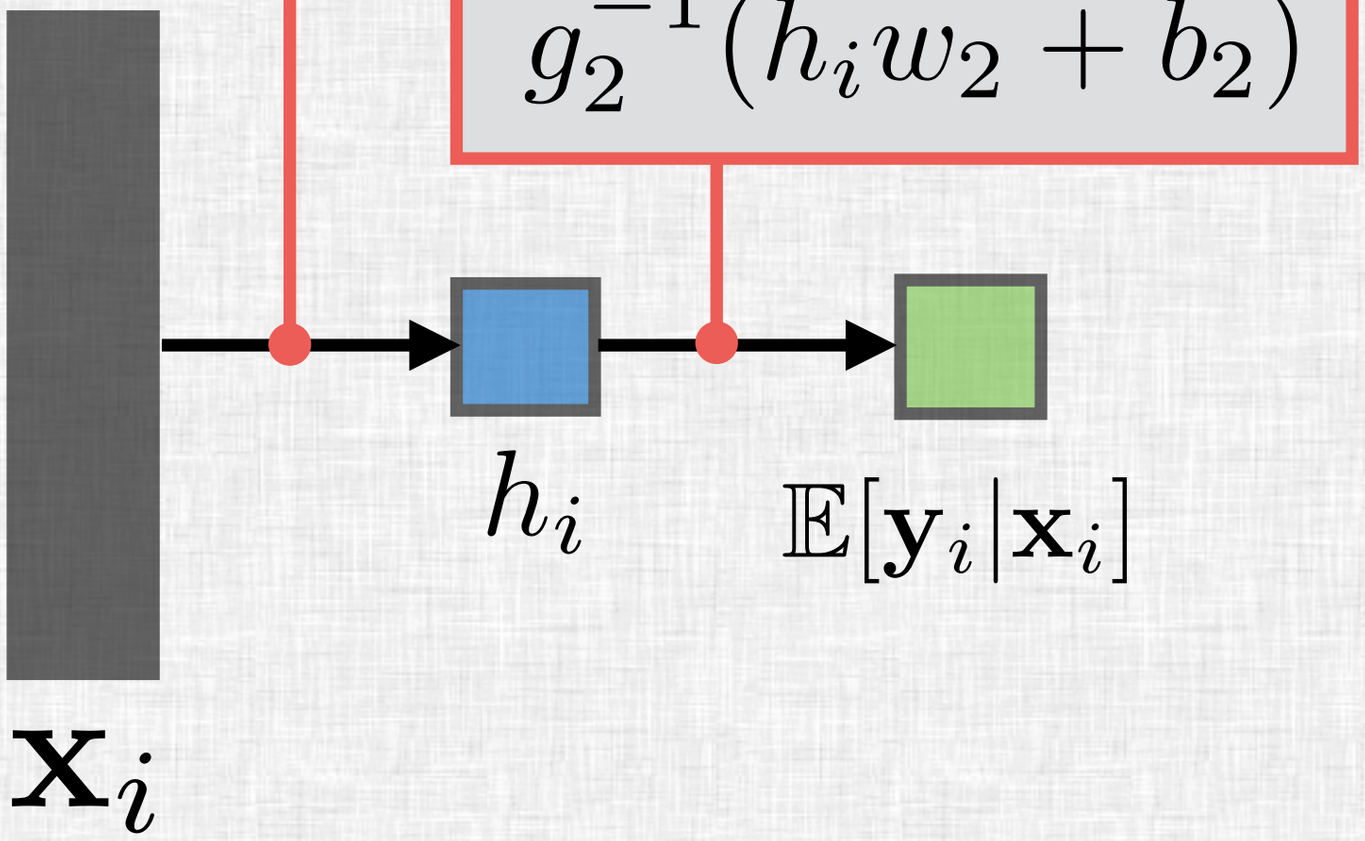
$$g_1^{-1}(\mathbf{x}_i^T \mathbf{w}_1 + b_1)$$



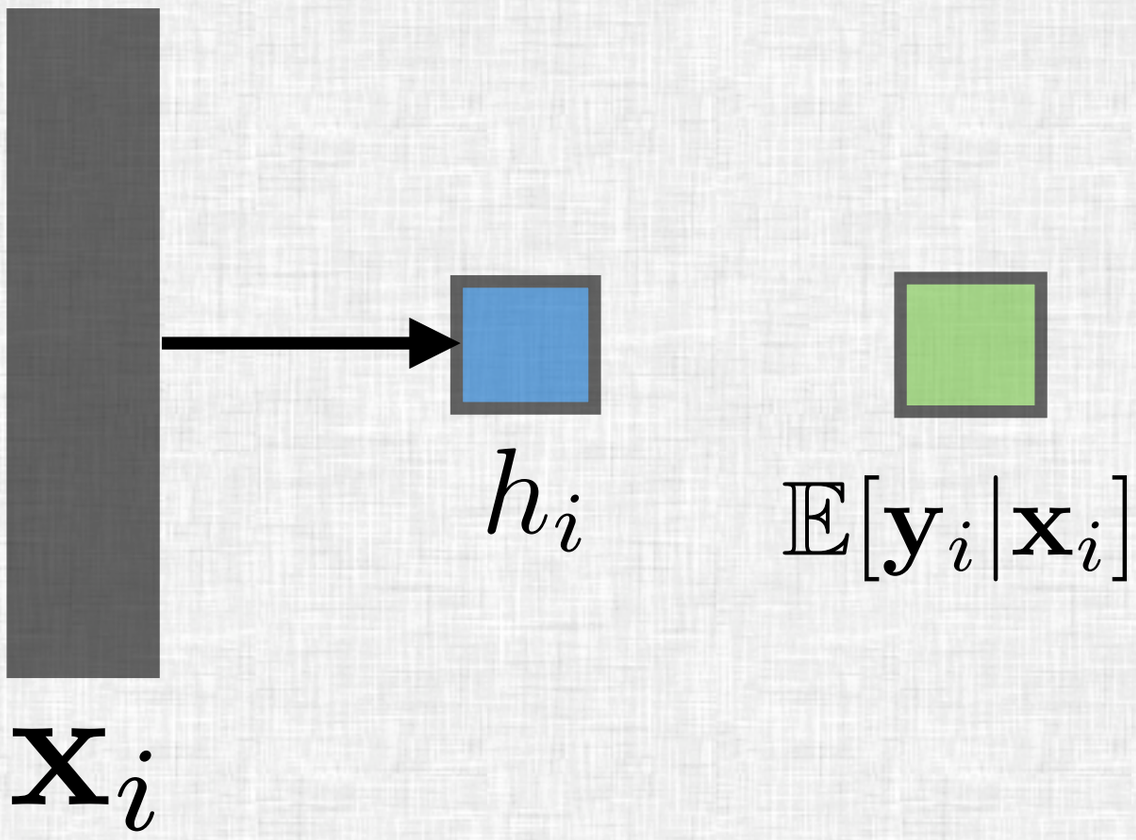
A Brief Overview of Deep Learning

$$g_1^{-1}(\mathbf{x}_i^T \mathbf{w}_1 + b_1)$$

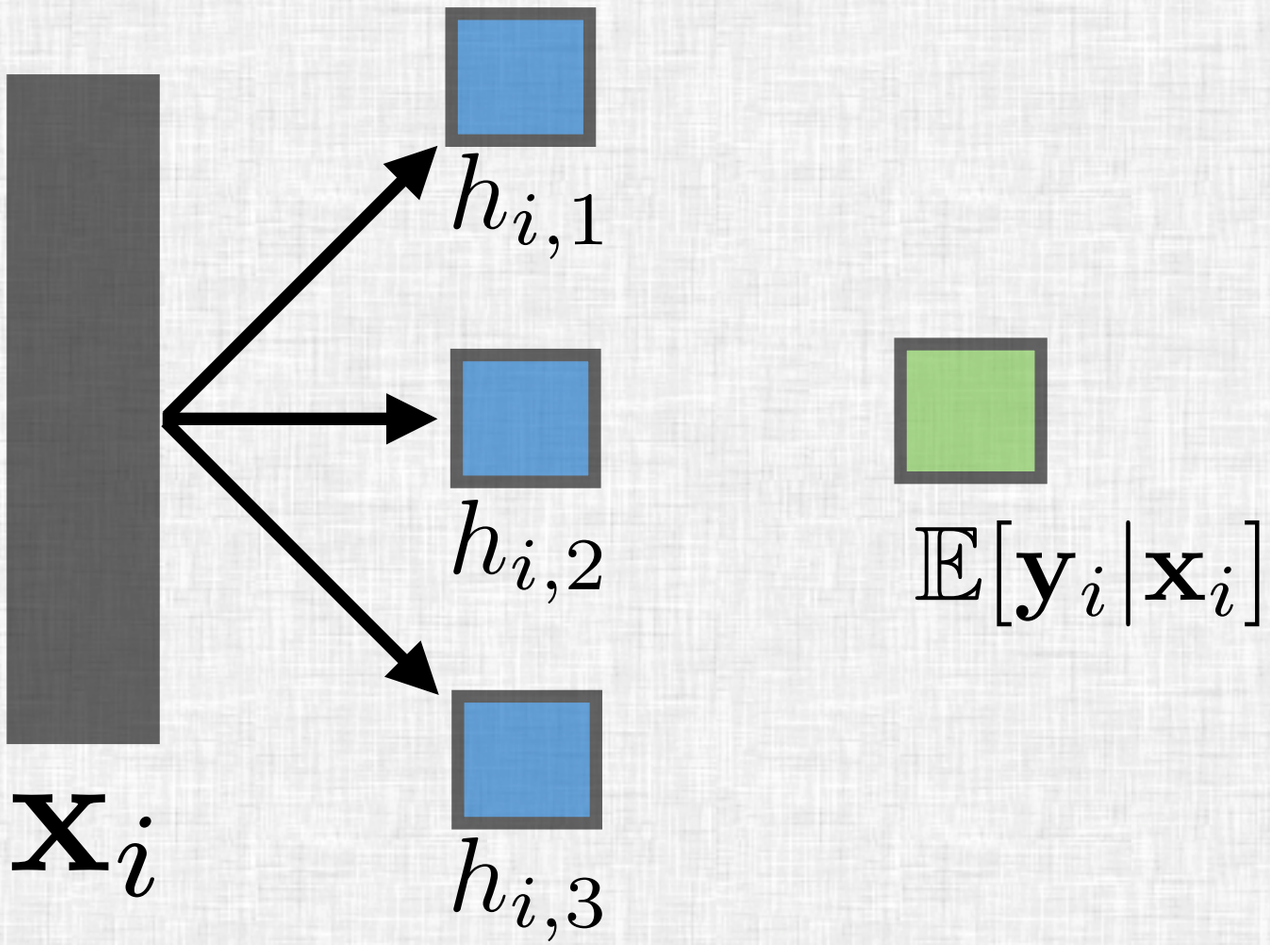
$$g_2^{-1}(h_i w_2 + b_2)$$



Building Neural Nets from Recursive GLMs

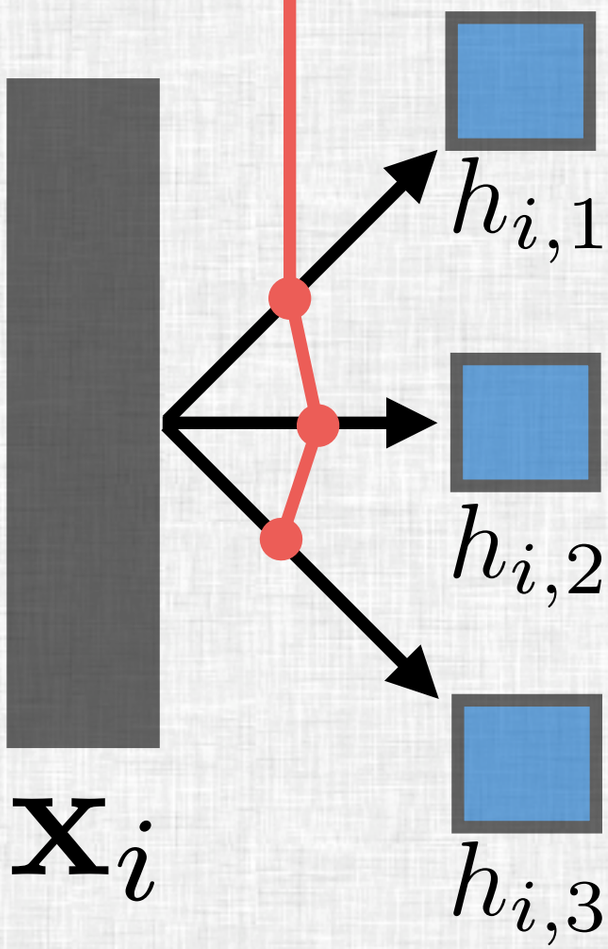


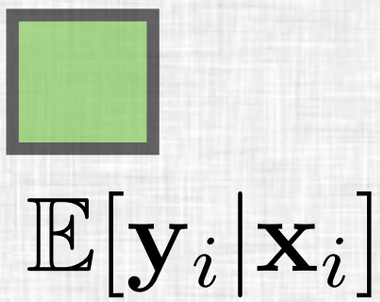
A Brief Overview of Deep Learning



A Brief Overview of Deep Learning

$$h_{i,j} = g_{1,j}^{-1}(\mathbf{x}_i^T \mathbf{w}_{1,j} + b_j)$$

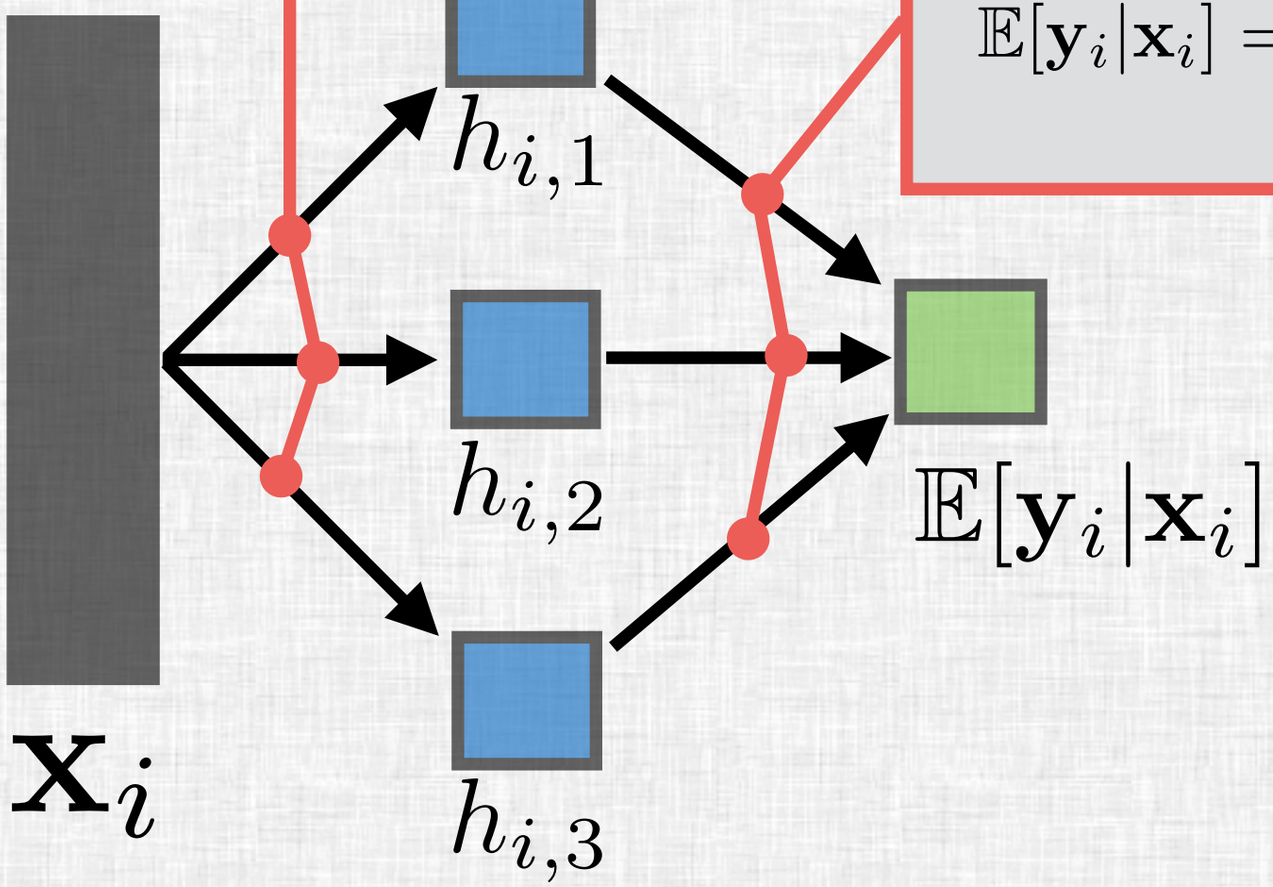




A Brief Overview of Deep Learning

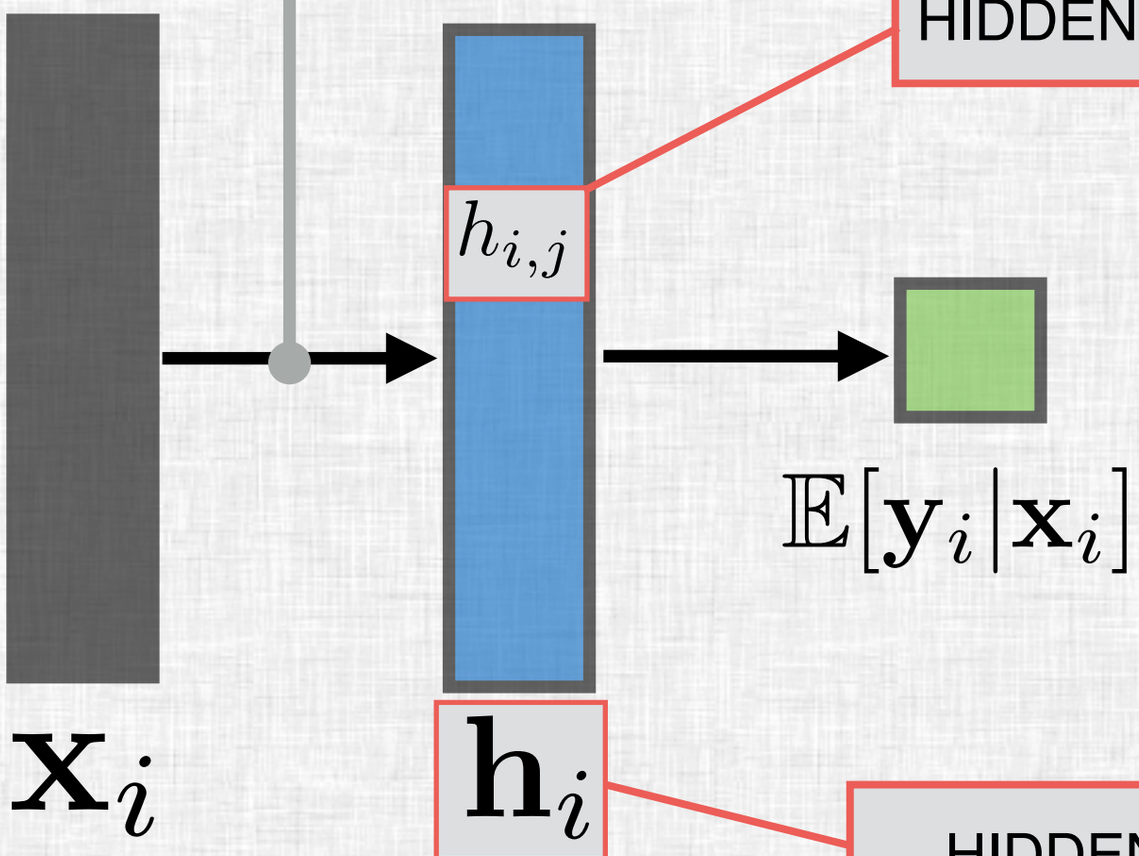
$$h_{i,j} = g_{1,j}^{-1}(\mathbf{x}_i^T \mathbf{w}_{1,j} + b_j)$$

$$\mathbb{E}[\mathbf{y}_i|\mathbf{x}_i] = g_2^{-1}\left(\sum_{j=1}^{H} h_{i,j} w_{2,j} + b_2\right)$$



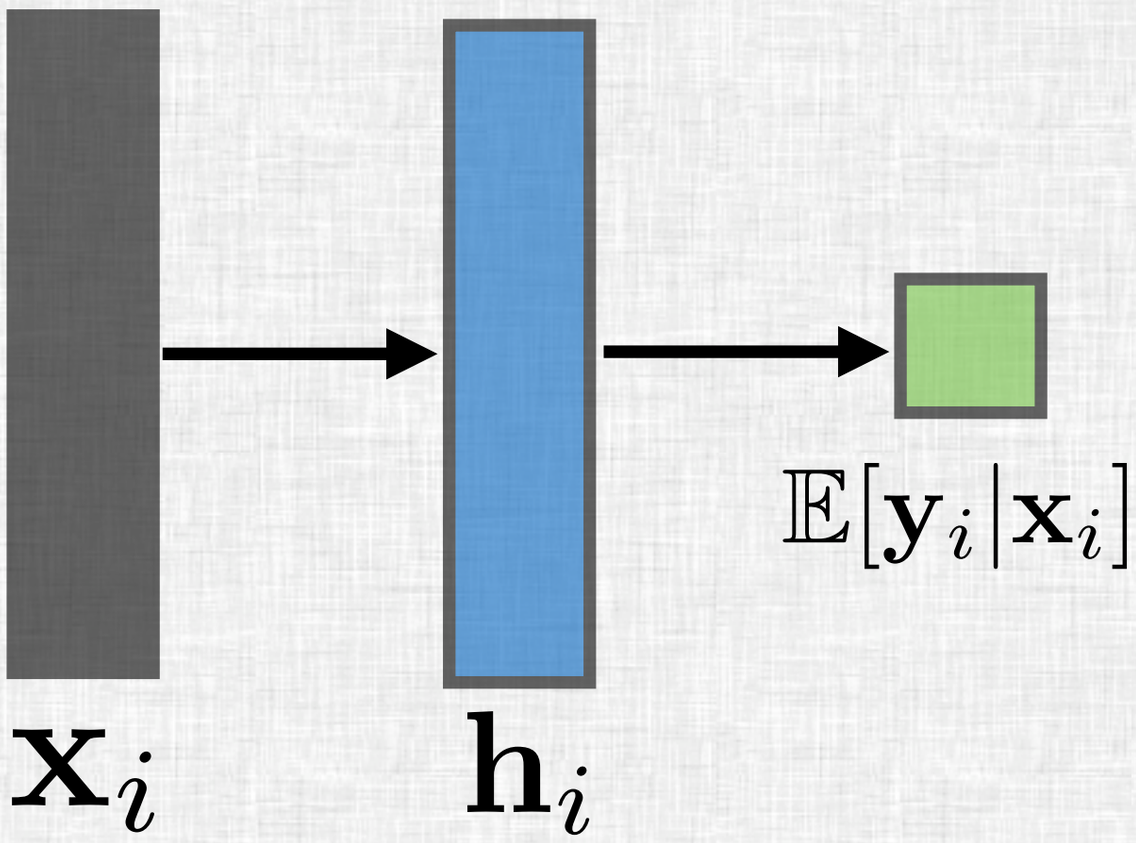
A Brief Overview of Deep Learning

$$\mathbf{h}_i = g_1^{-1}(\mathbf{x}_i^T \mathbf{W}_1 + \mathbf{b}_1)$$

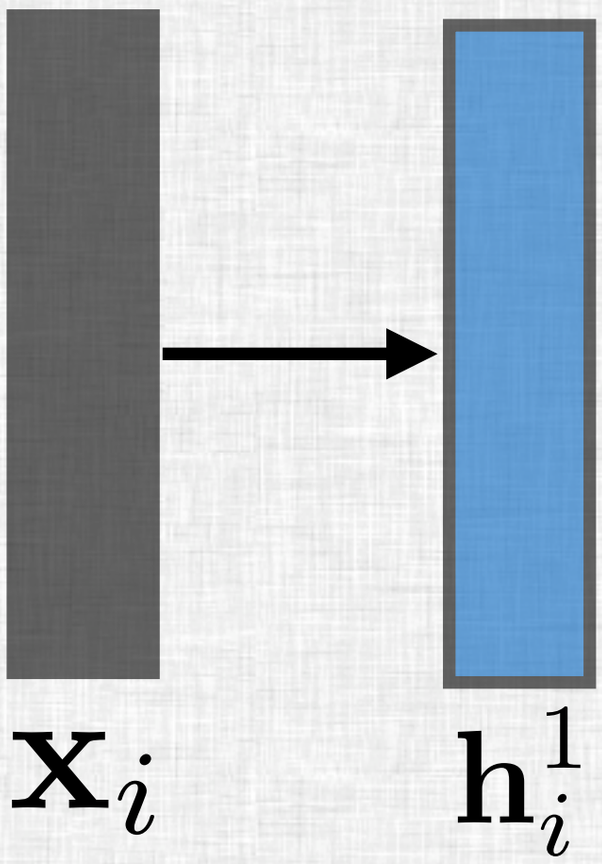


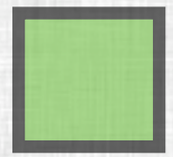
THIS IS A ONE-HIDDEN-LAYER ‘NEURAL NETWORK’!

Deep Neural Networks

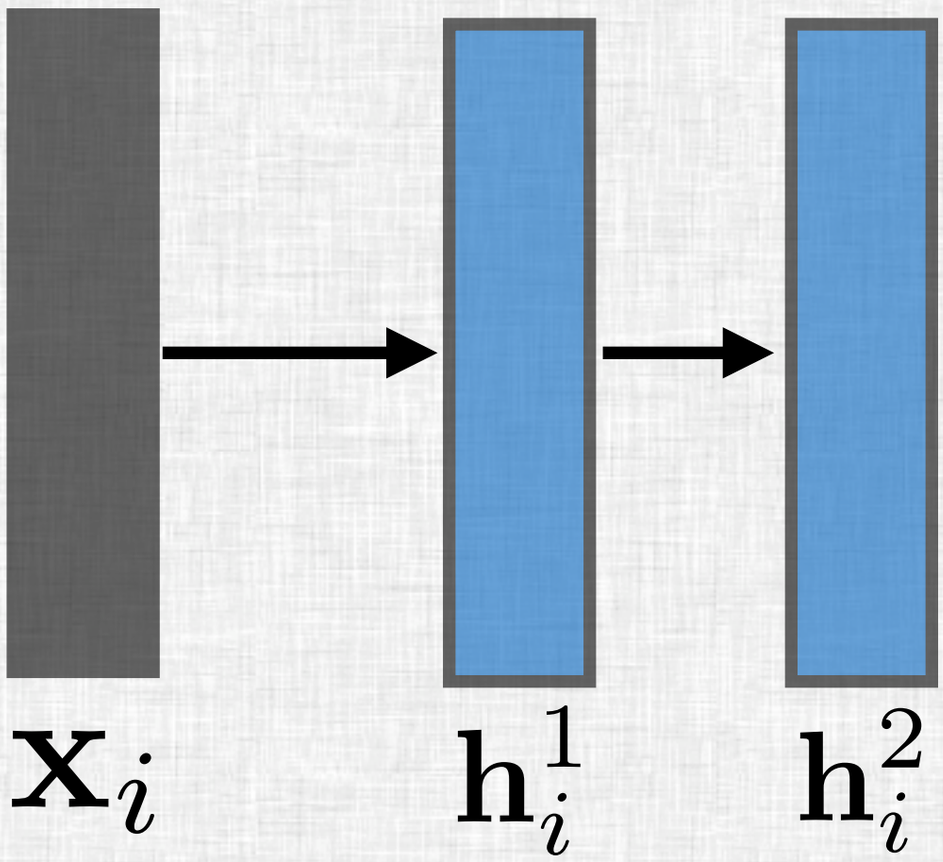


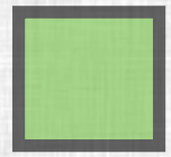
A Brief Overview of Deep Learning





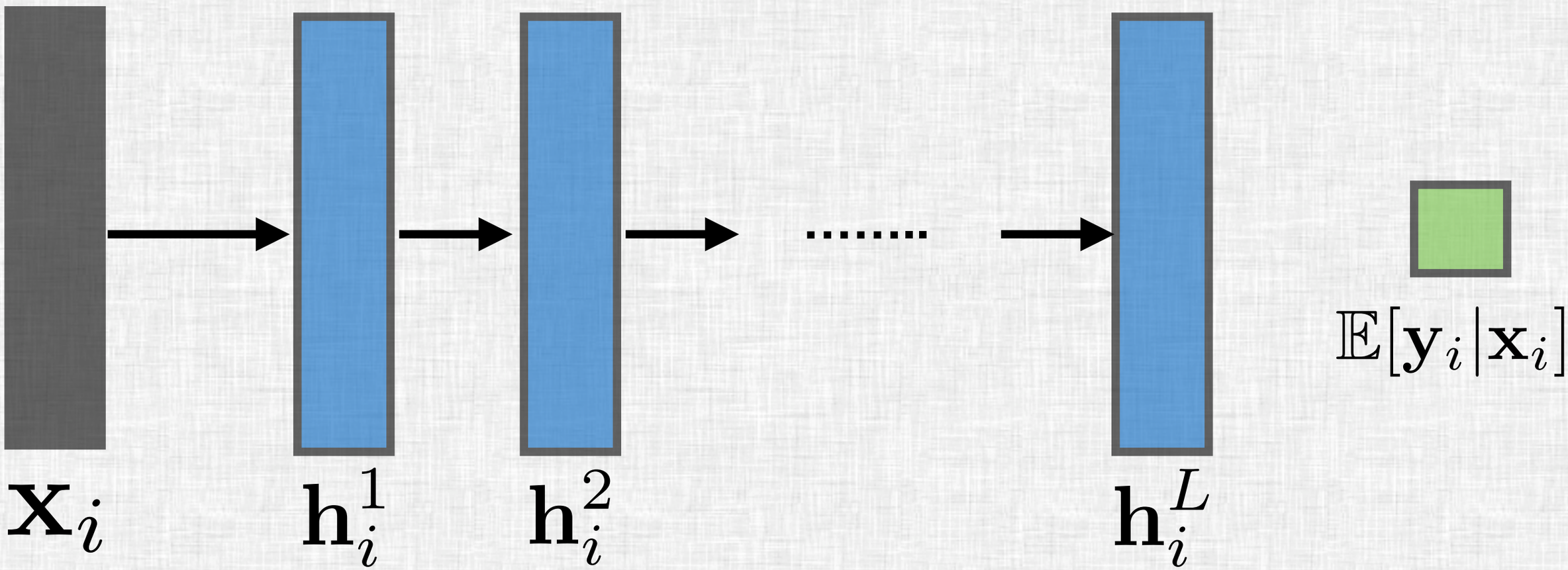
A Brief Overview of Deep Learning





$\mathbb{E}[\mathbf{y}_i|\mathbf{x}_i]$

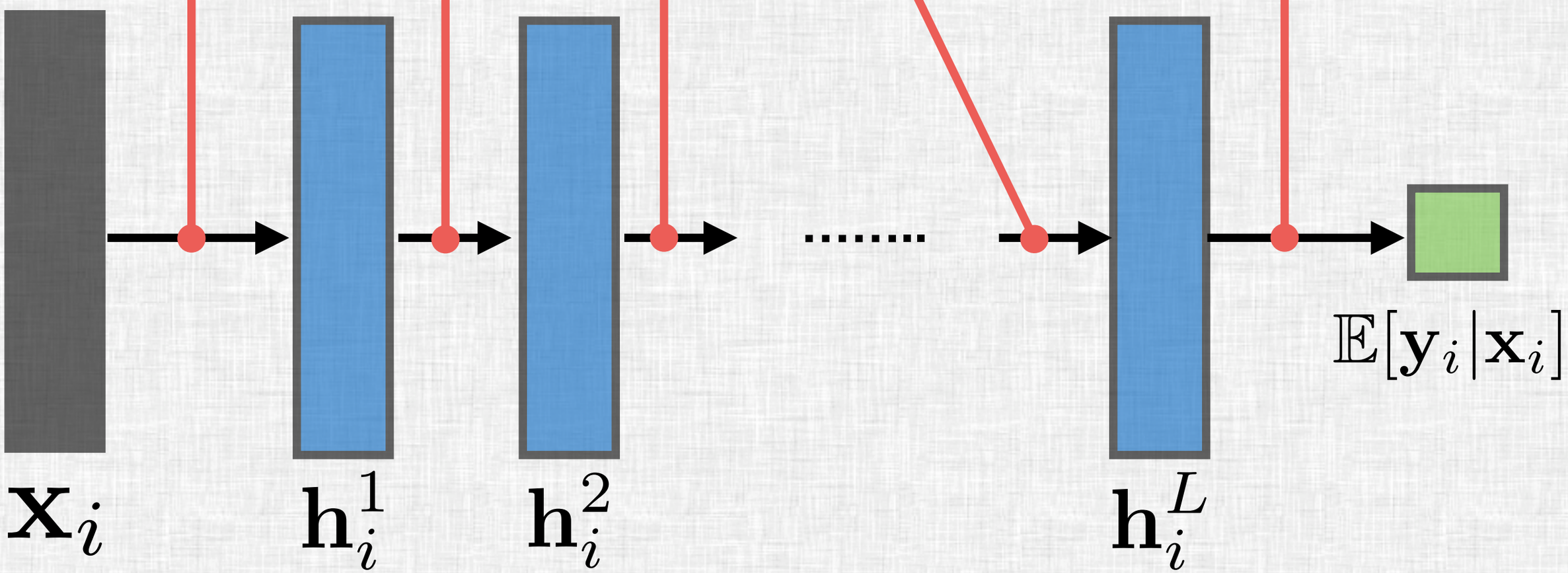
A Brief Overview of Deep Learning



Overview

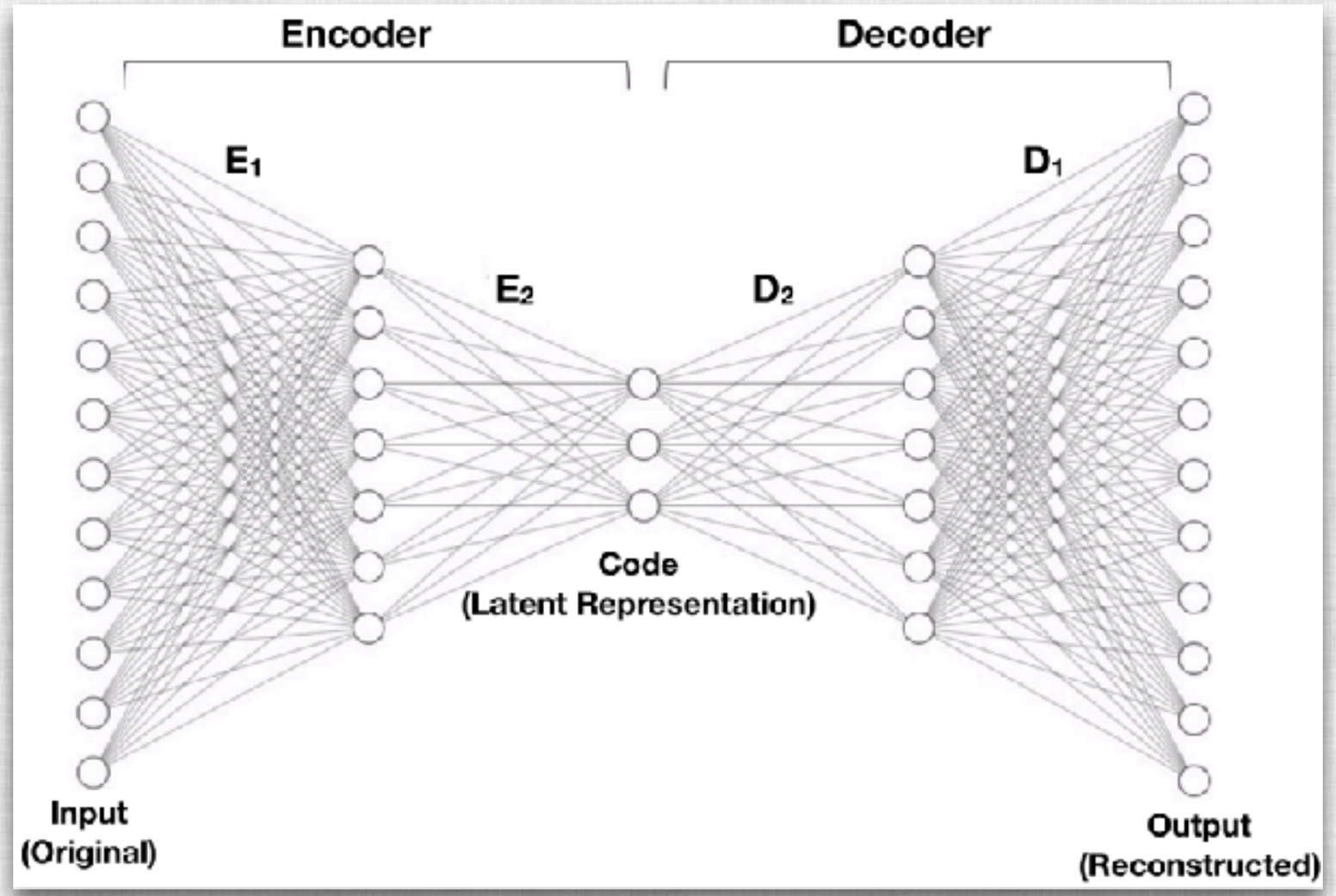
$$\mathbf{h}_i^l = f_l(\mathbf{h}_i^{l-1}\mathbf{W}_l + \mathbf{b}_l), \quad \mathbf{h}_i^0 = \mathbf{x}_i$$

$$\mathbb{E}[\mathbf{y}_i|\mathbf{x}_i] = g^{-1}(\mathbf{h}_i^L \mathbf{w}_o + b_o)$$

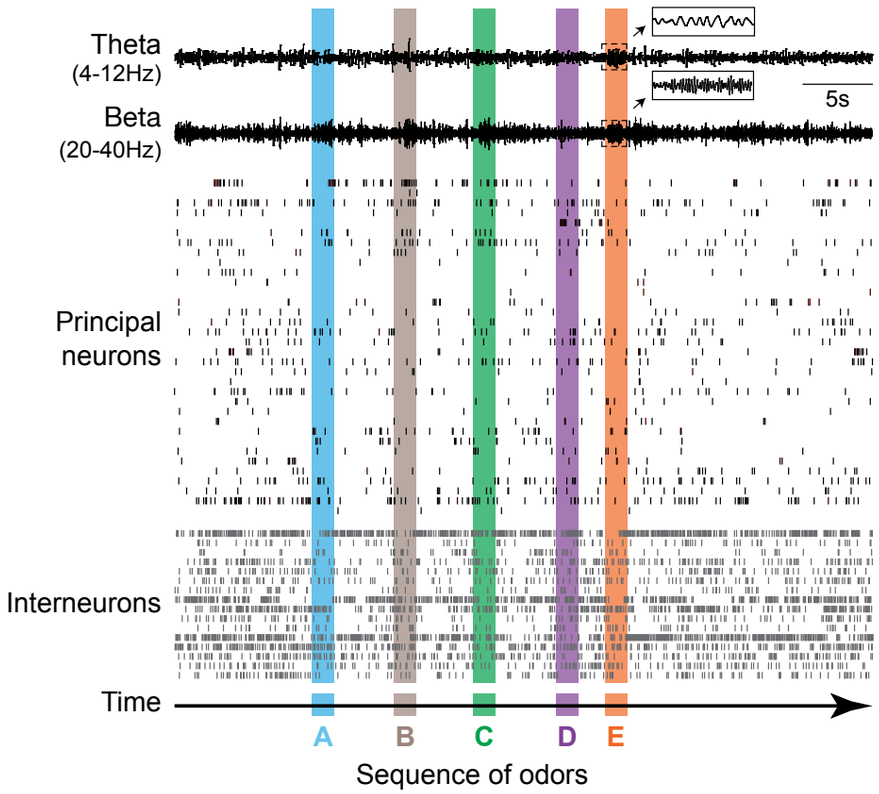


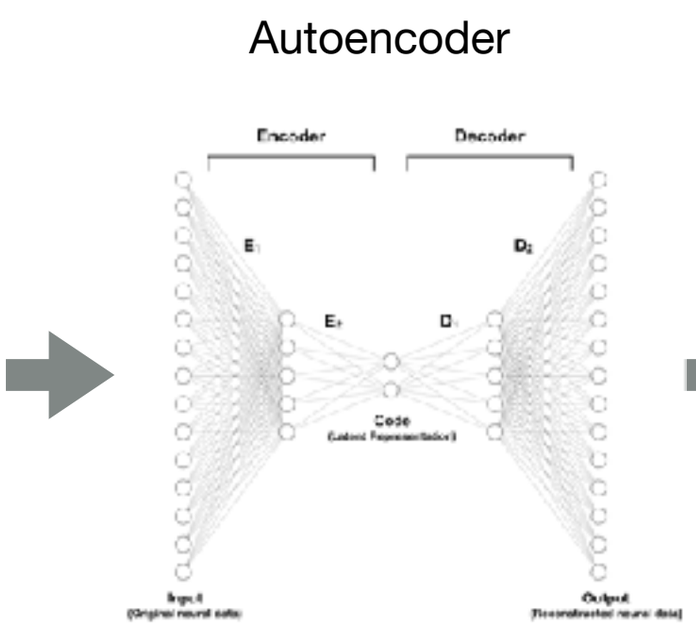
A Brief Overview of Deep Learning

- Autoencoder is a special type of neural network commonly used for learning latent representations of the data.
- You can think of it as a nonlinear alternative to PCA.



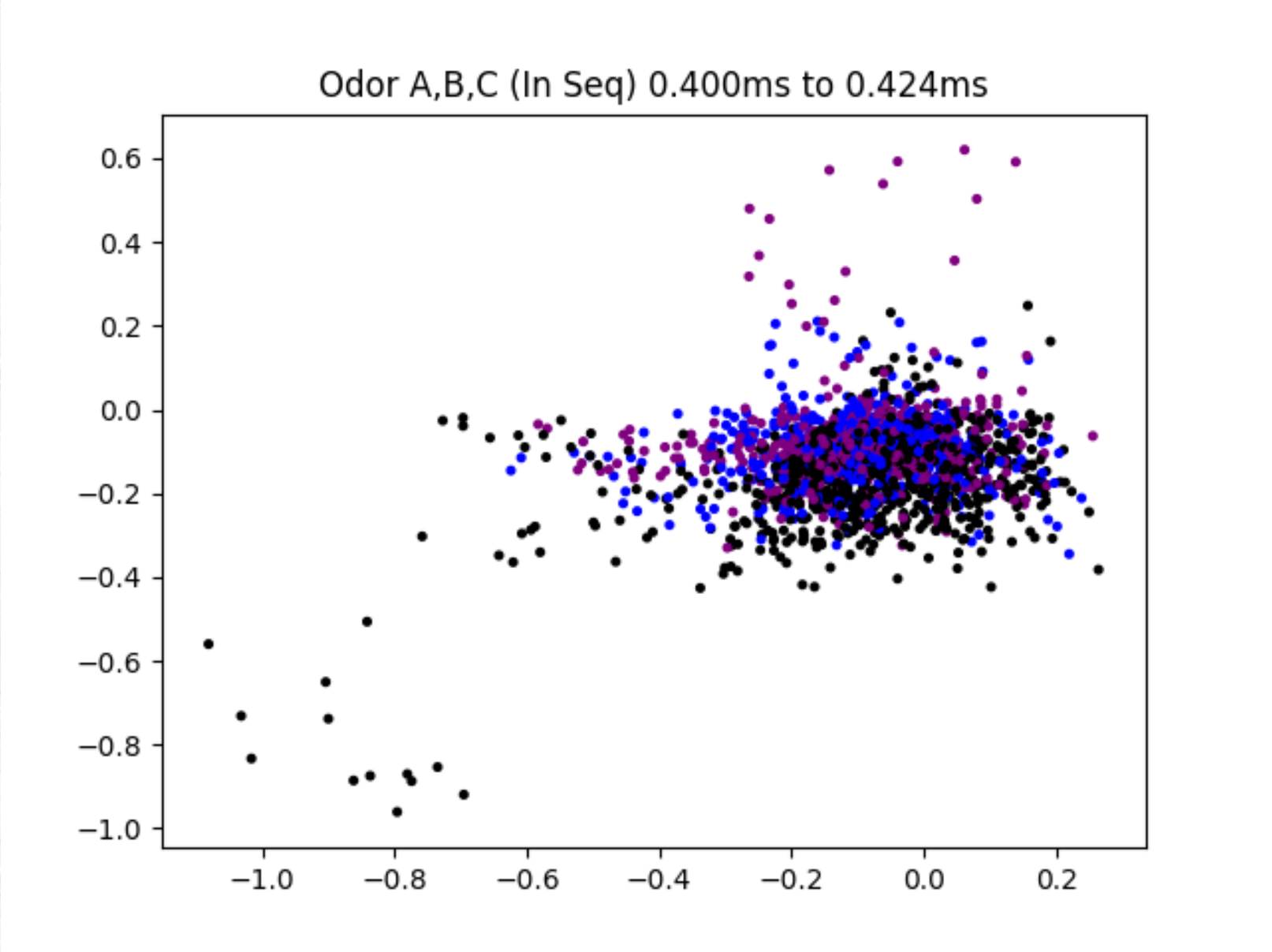
Unsupervised Latent Representation Learning



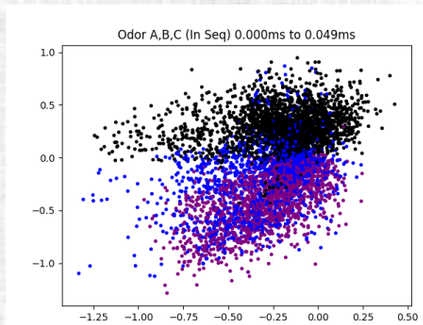


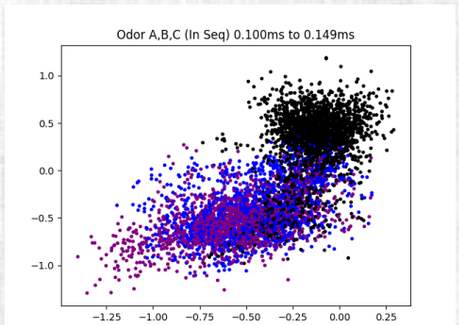
A → B → C → D

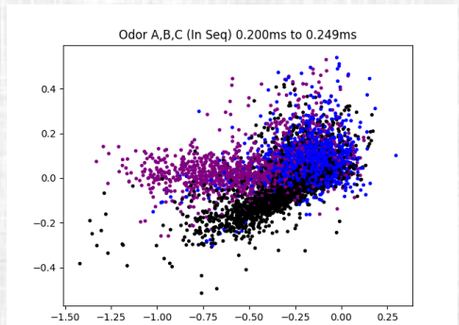
Unsupervised Latent Representation Learning

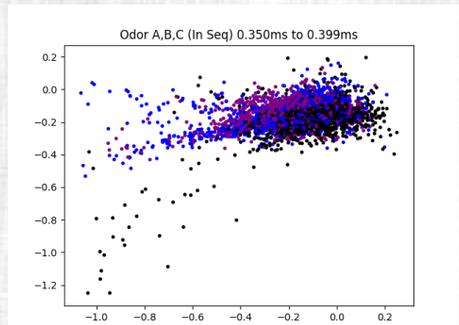


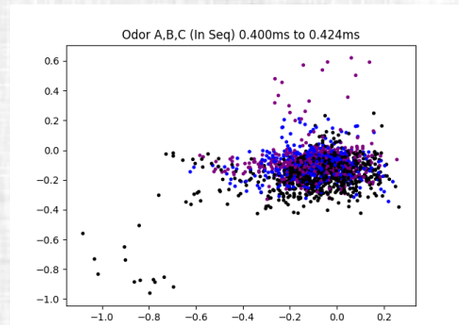
Unsupervised Latent Representation Learning

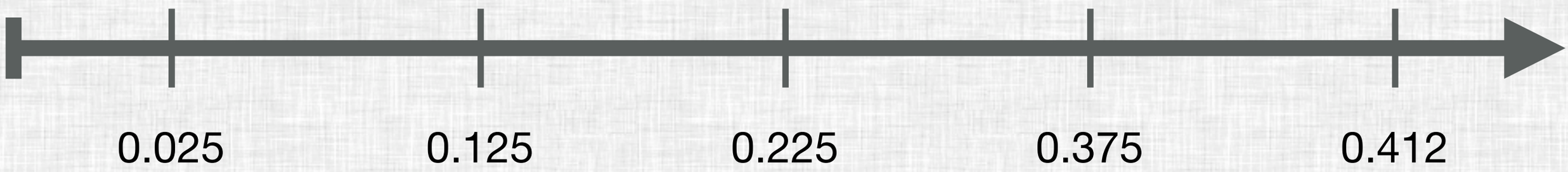




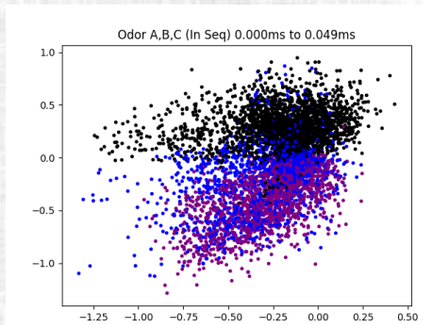


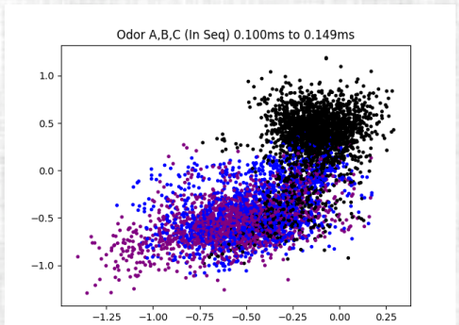


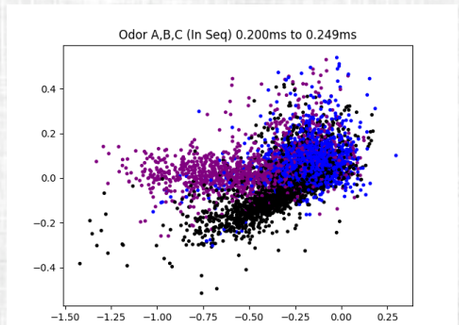


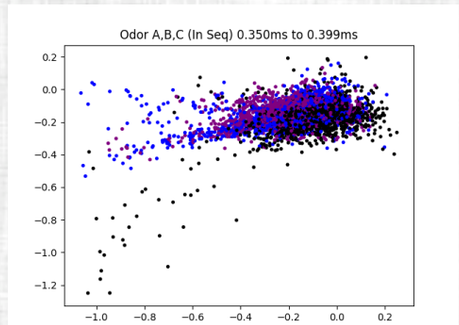


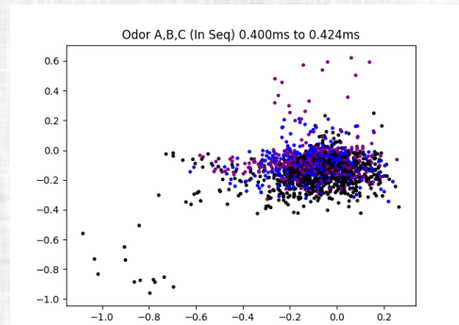
Unsupervised Latent Representation Learning

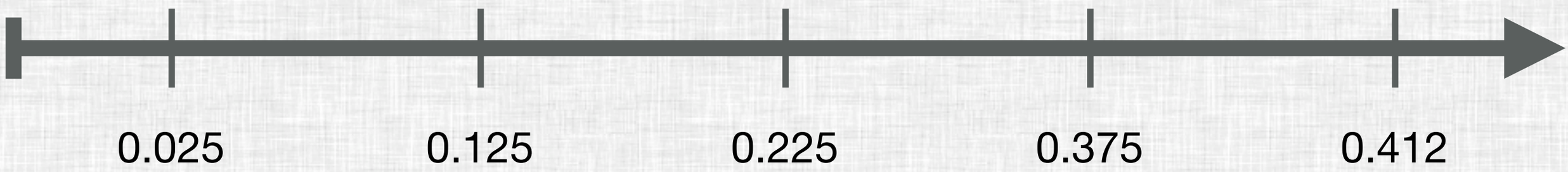




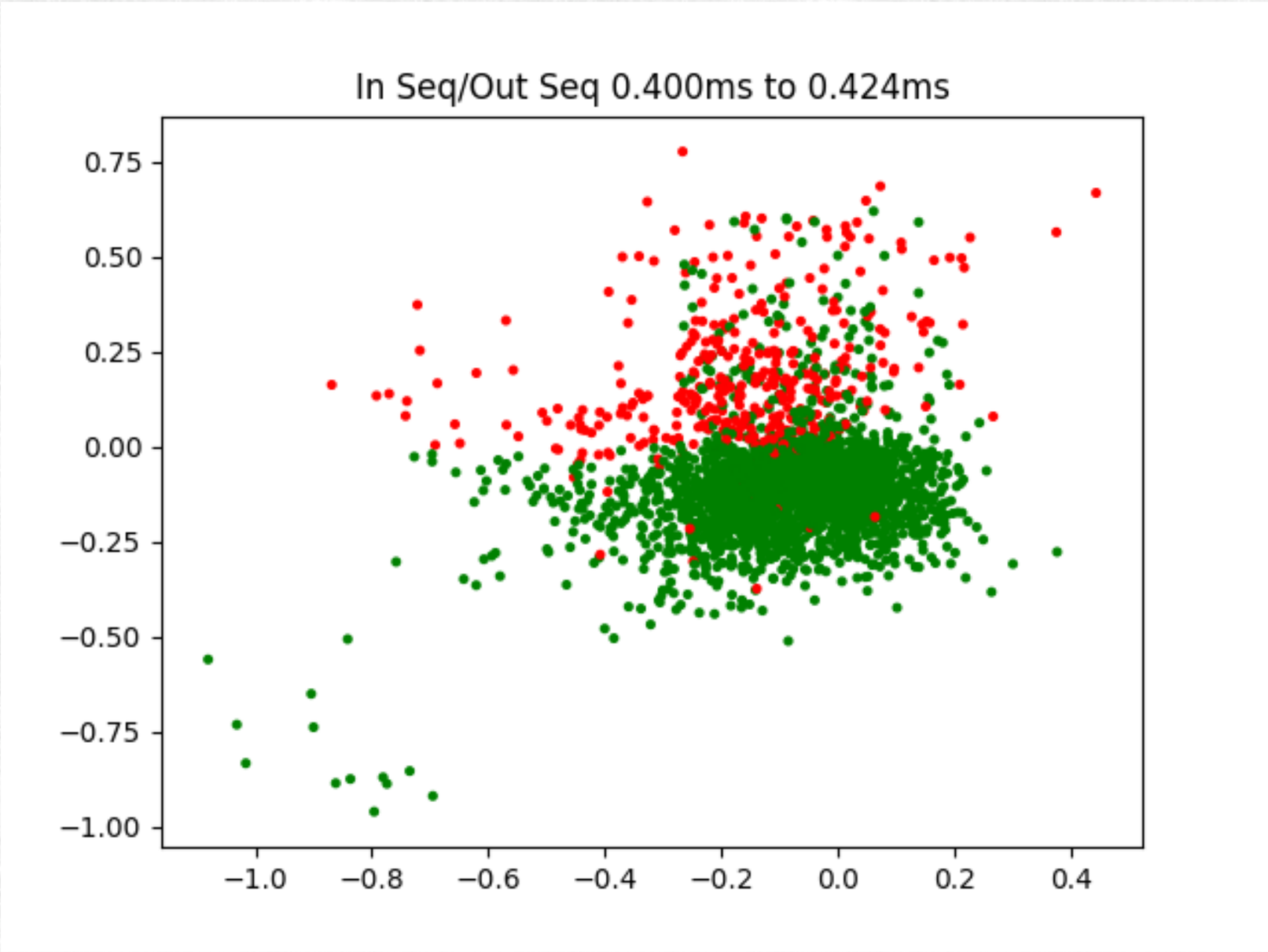




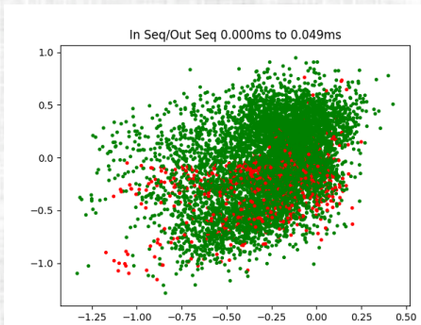


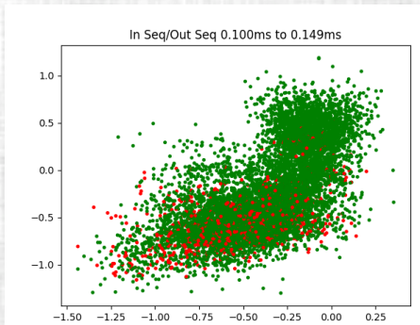


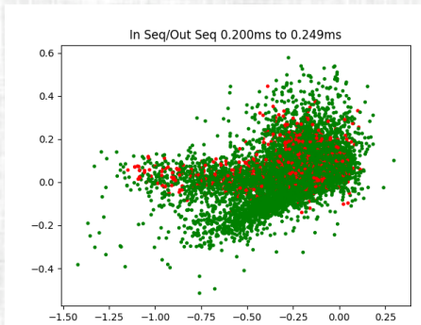
Unsupervised Latent Representation Learning

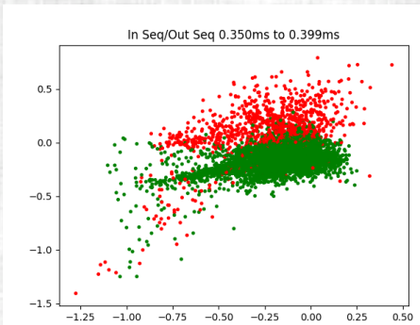


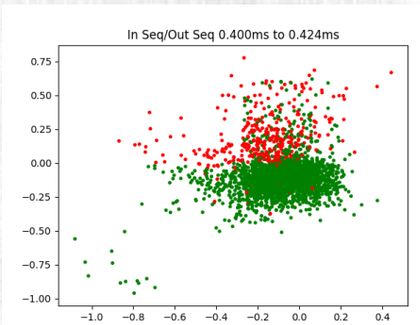
Unsupervised Latent Representation Learning

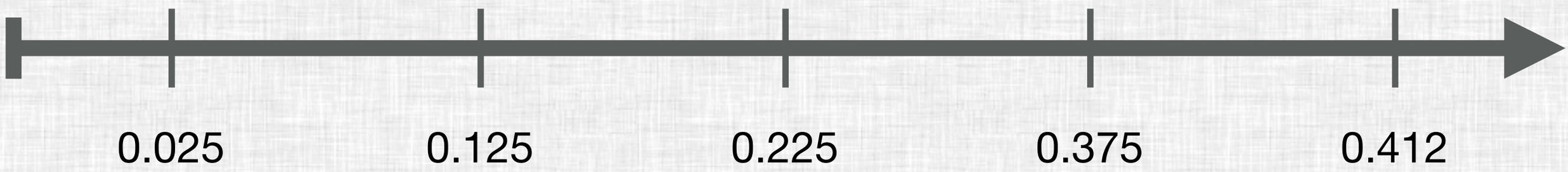




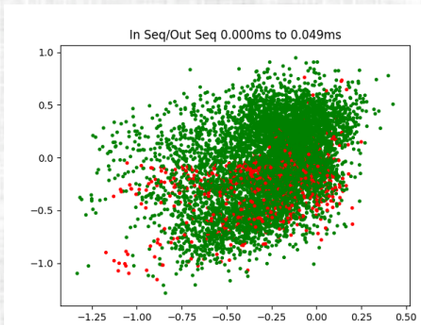


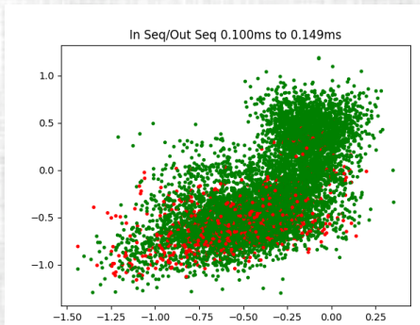


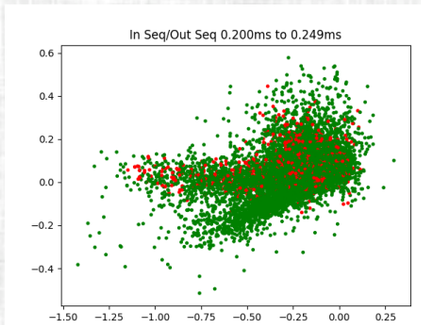


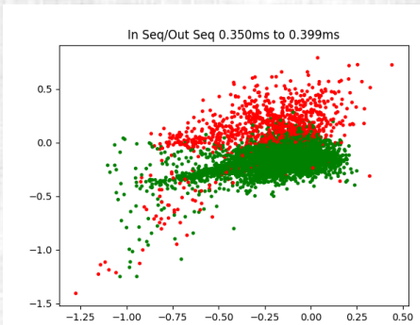


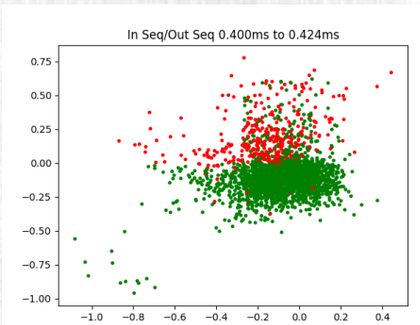
Unsupervised Latent Representation Learning

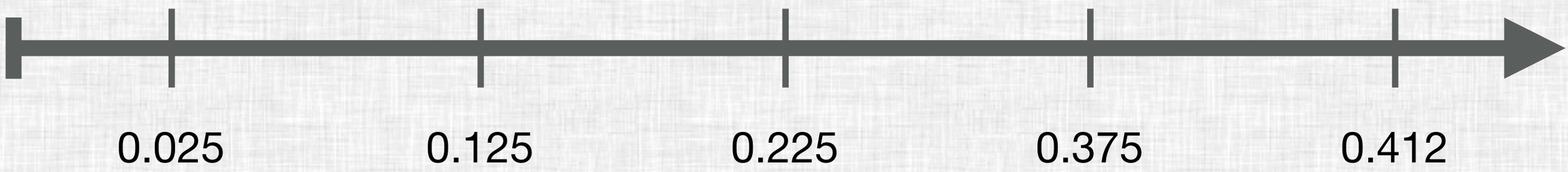








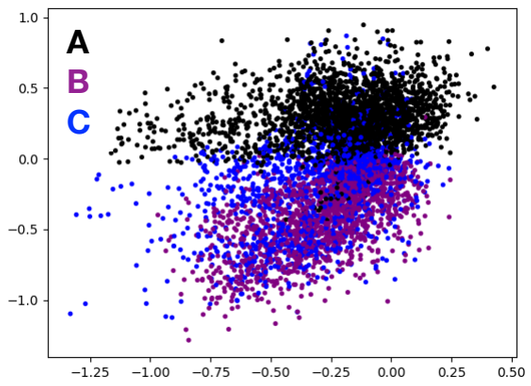




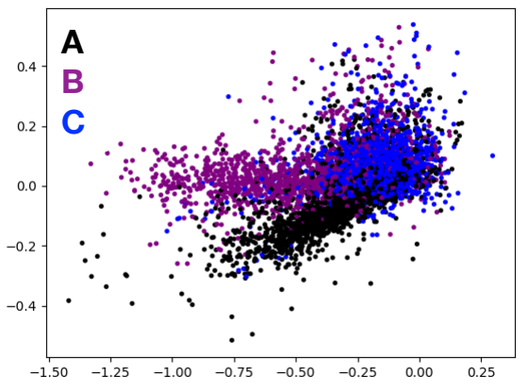
Unsupervised Latent Representation Learning

Odor and InSeq/OutSeq info is present at different moments within trials

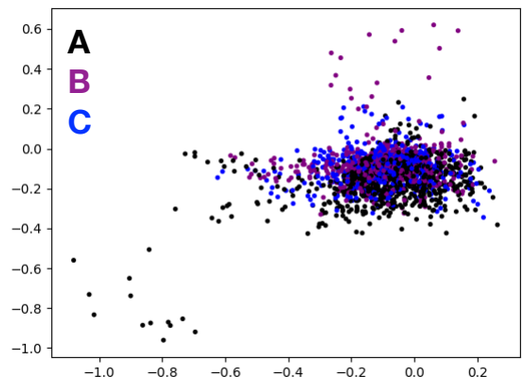
Poor odor differentiation

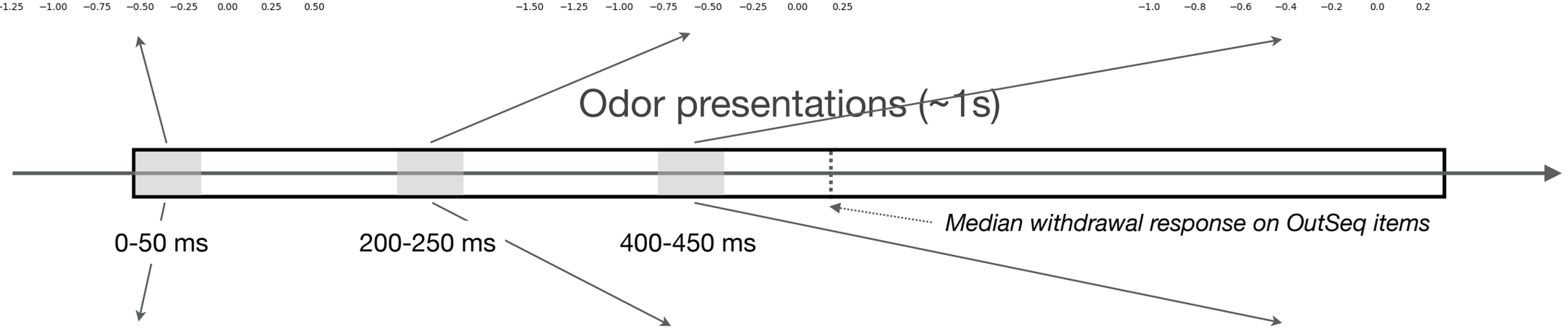


Strong odor differentiation

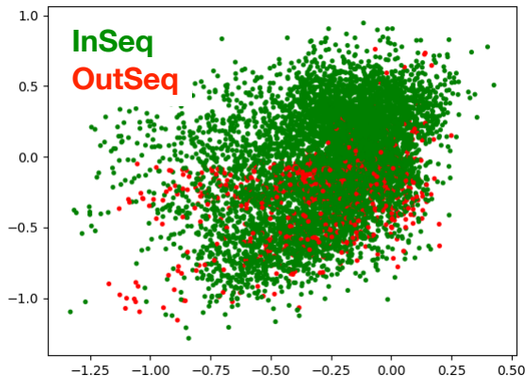


Poor odor differentiation

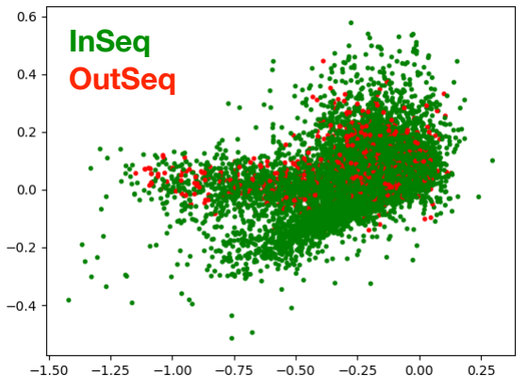




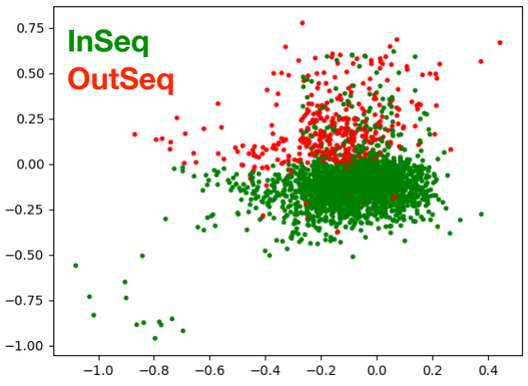
Poor InSeq/OutSeq differentiation



Poor InSeq/OutSeq differentiation



Strong inSeq/OutSeq differentiation

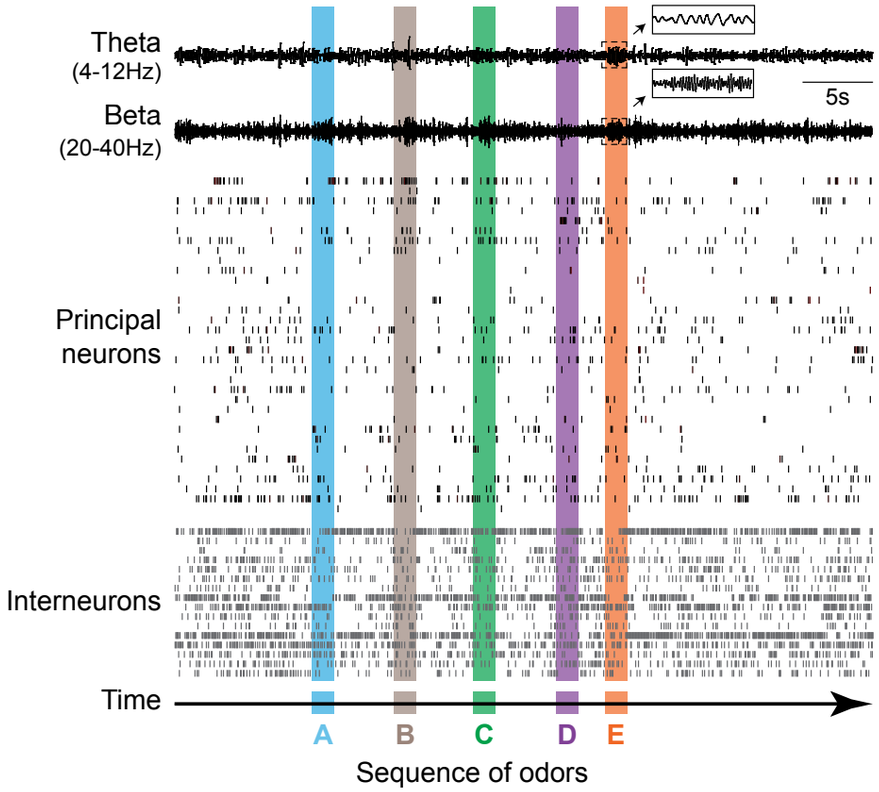


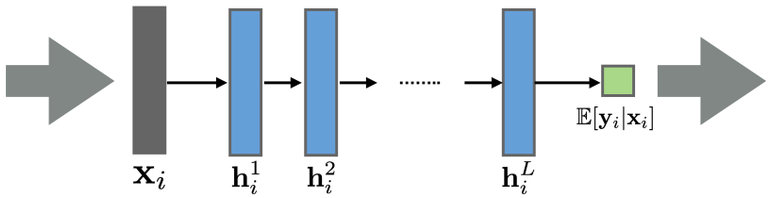
Shahbaba, Li, Agostinelli, Saraf, Cooper, Haghverdian, Elias, Baldi, Fortin (2021) *Nature Communications*

Supervised Latent Representation Learning

- Alternatively, we could process the data by **convolution** and **pooling**.
- Convolutional neural networks have been very successful in computer vision to extract features that are **location invariant**.
- Here we regard neural patterns during a trial as an image.
- We use time convolution filters that can extract features that are **time invariant**.

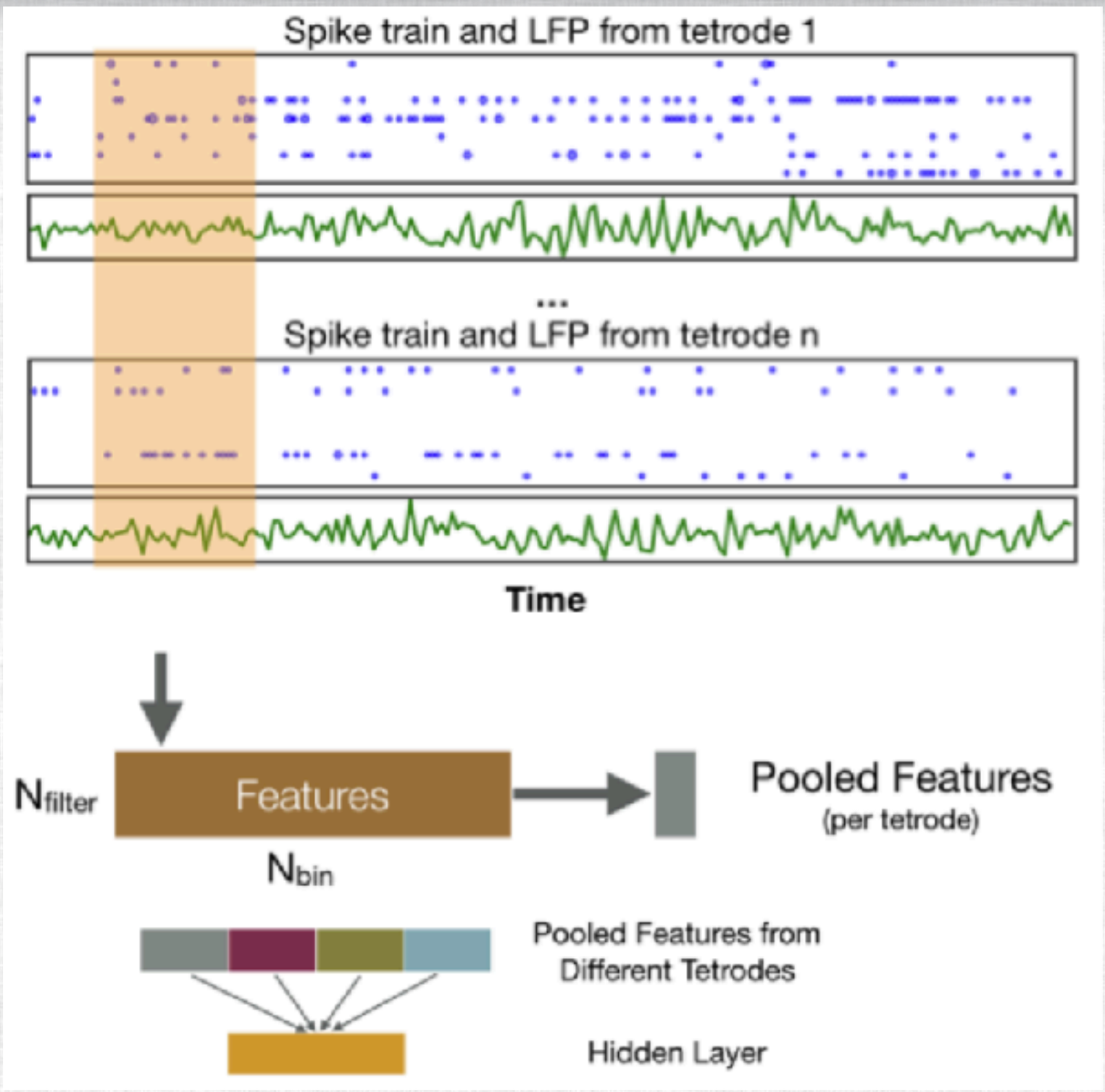
Supervised Latent Representation Learning





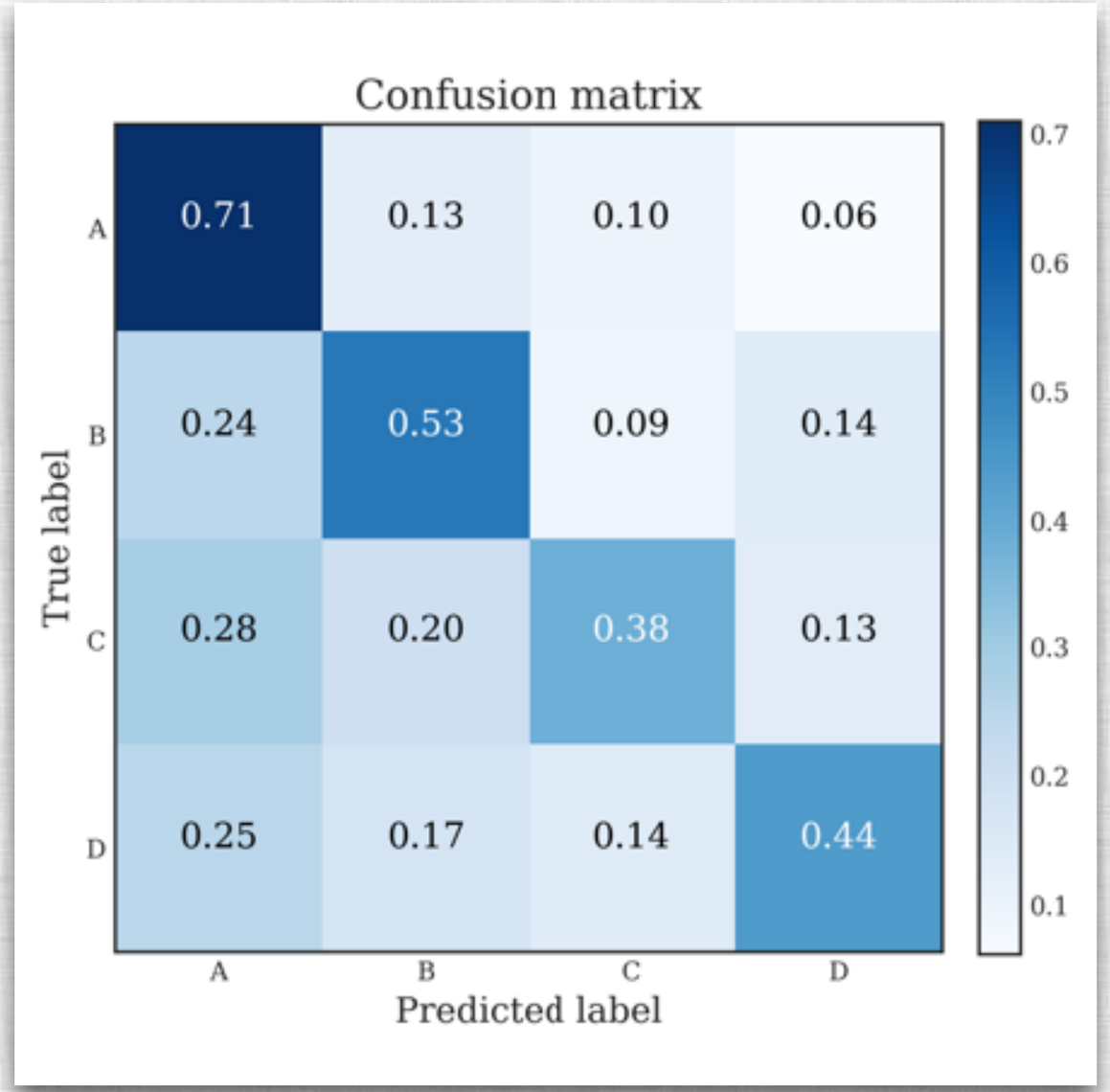
A → B → C → D

Supervised Latent Representation Learning

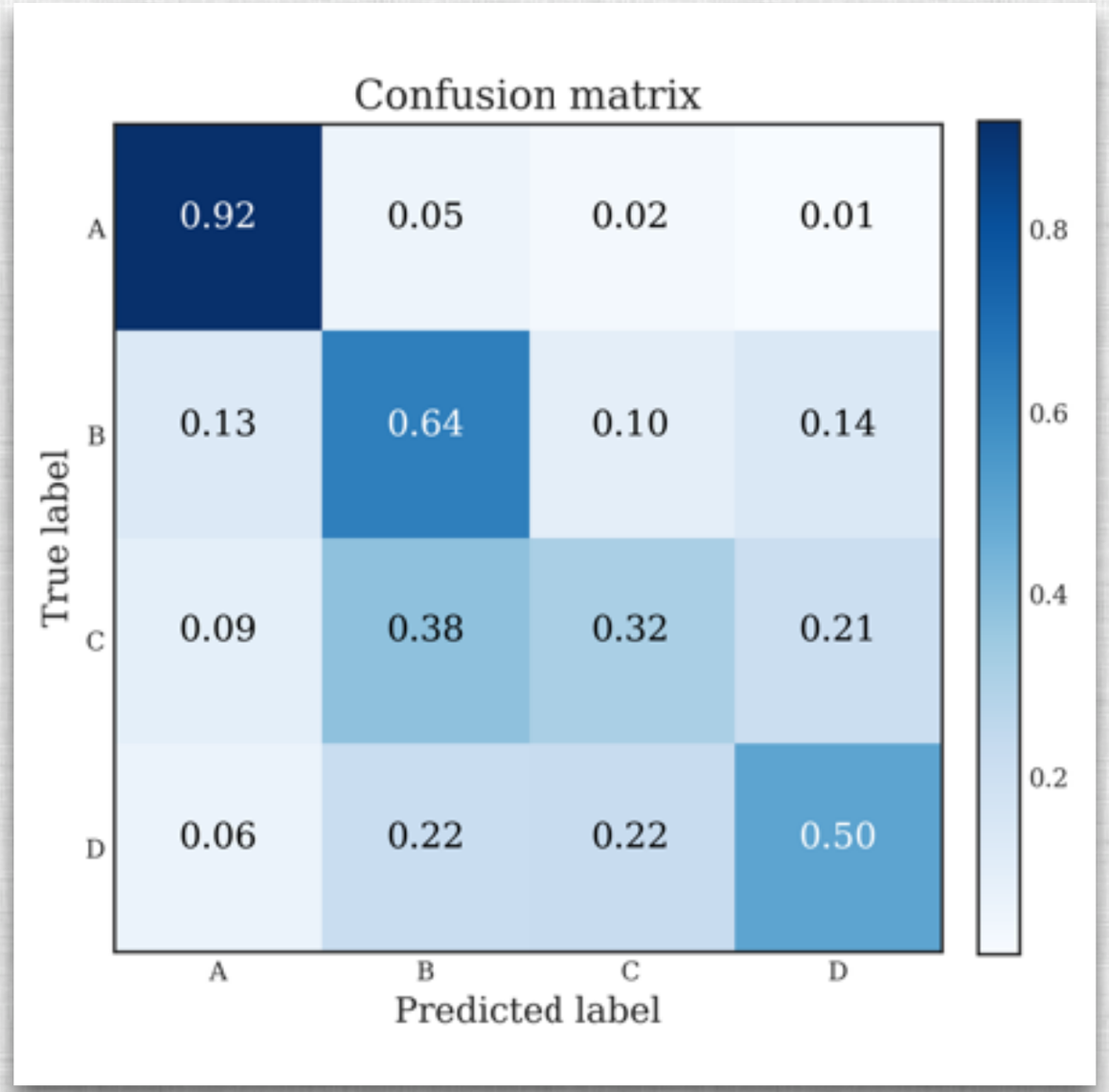


Supervised Latent Representation Learning

Lasso

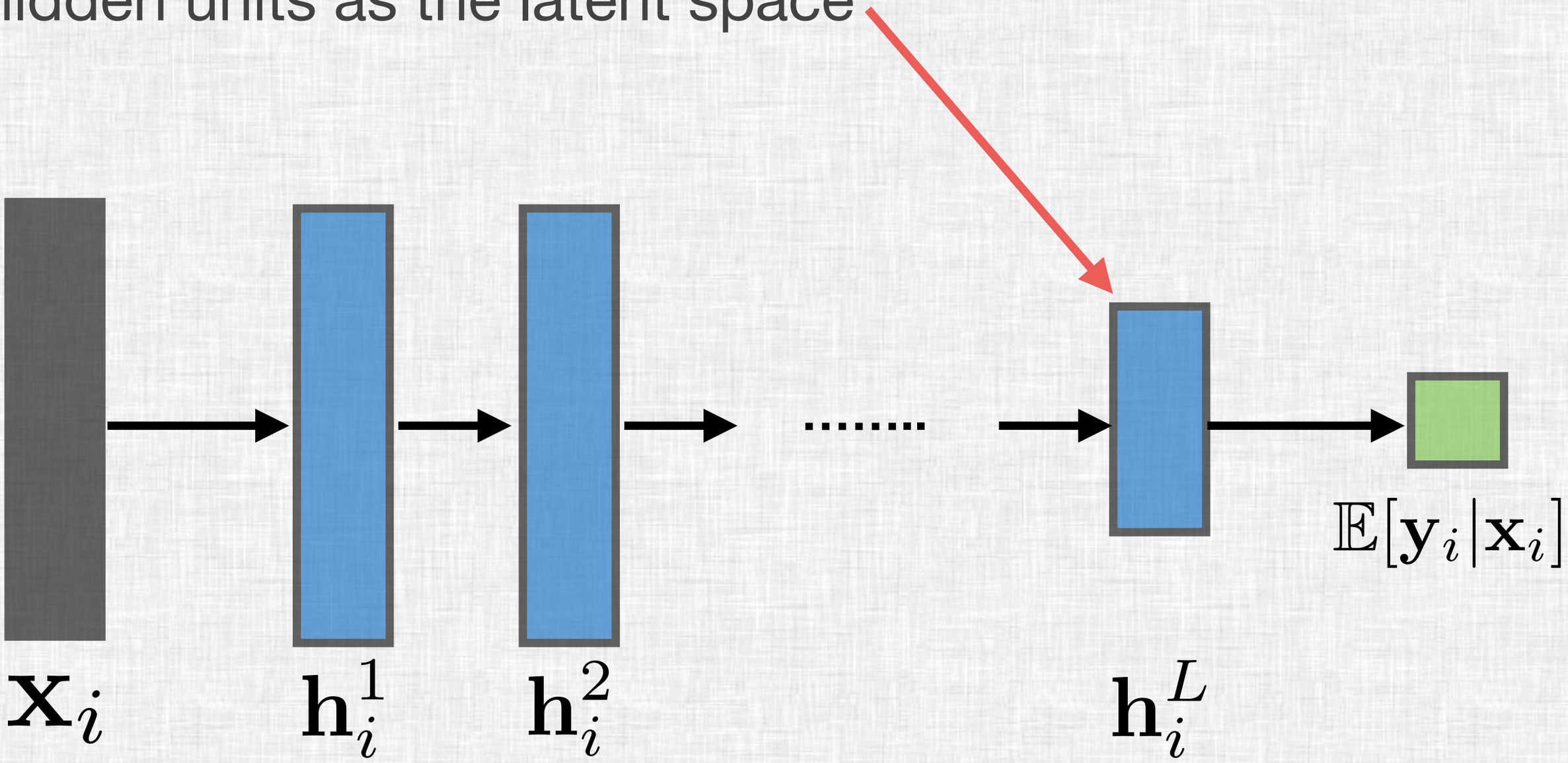


CNN



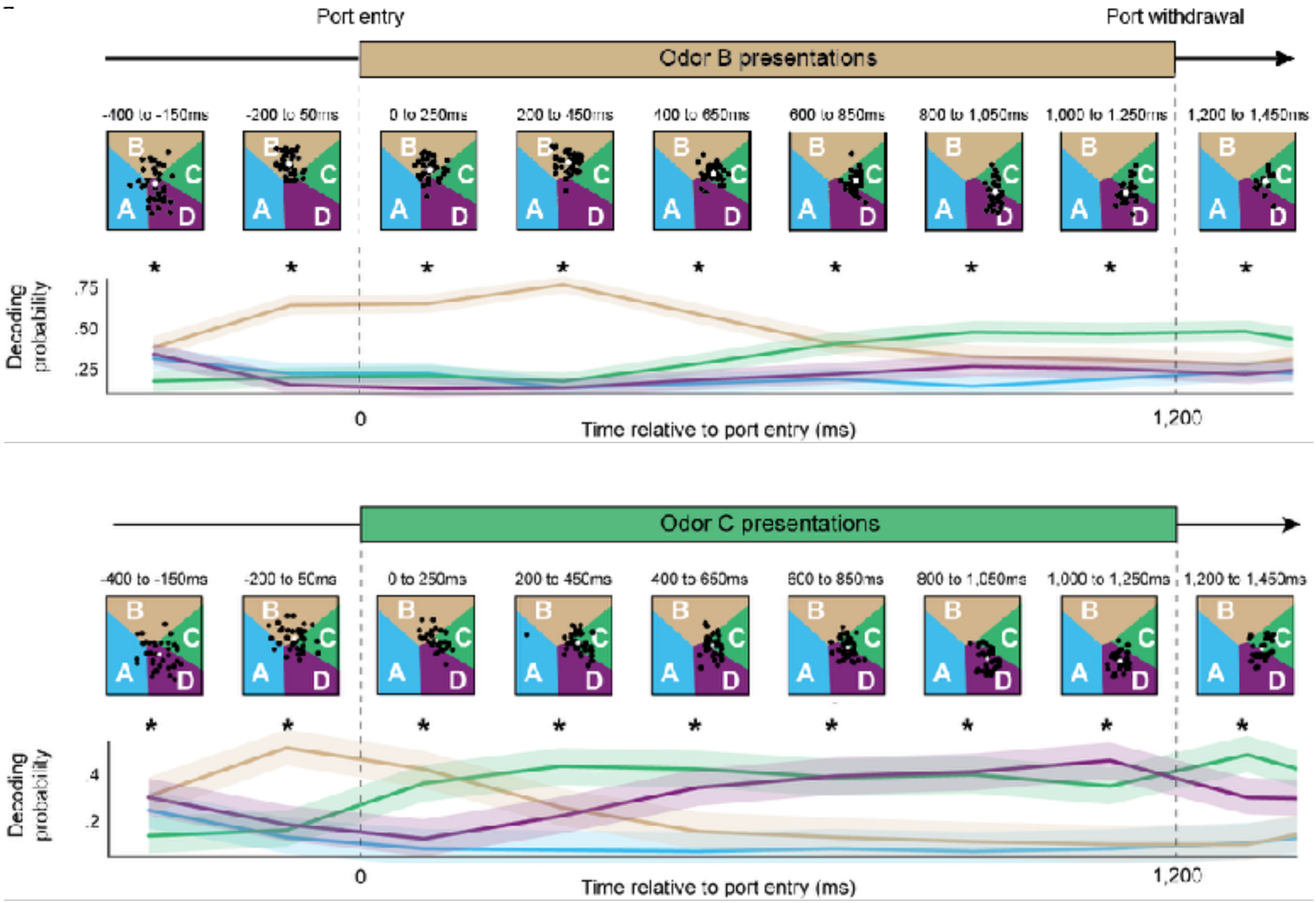
Supervised Latent Representation Learning

We use the last hidden layer with a relatively smaller number of hidden units as the latent space



Supervised Latent Representation Learning

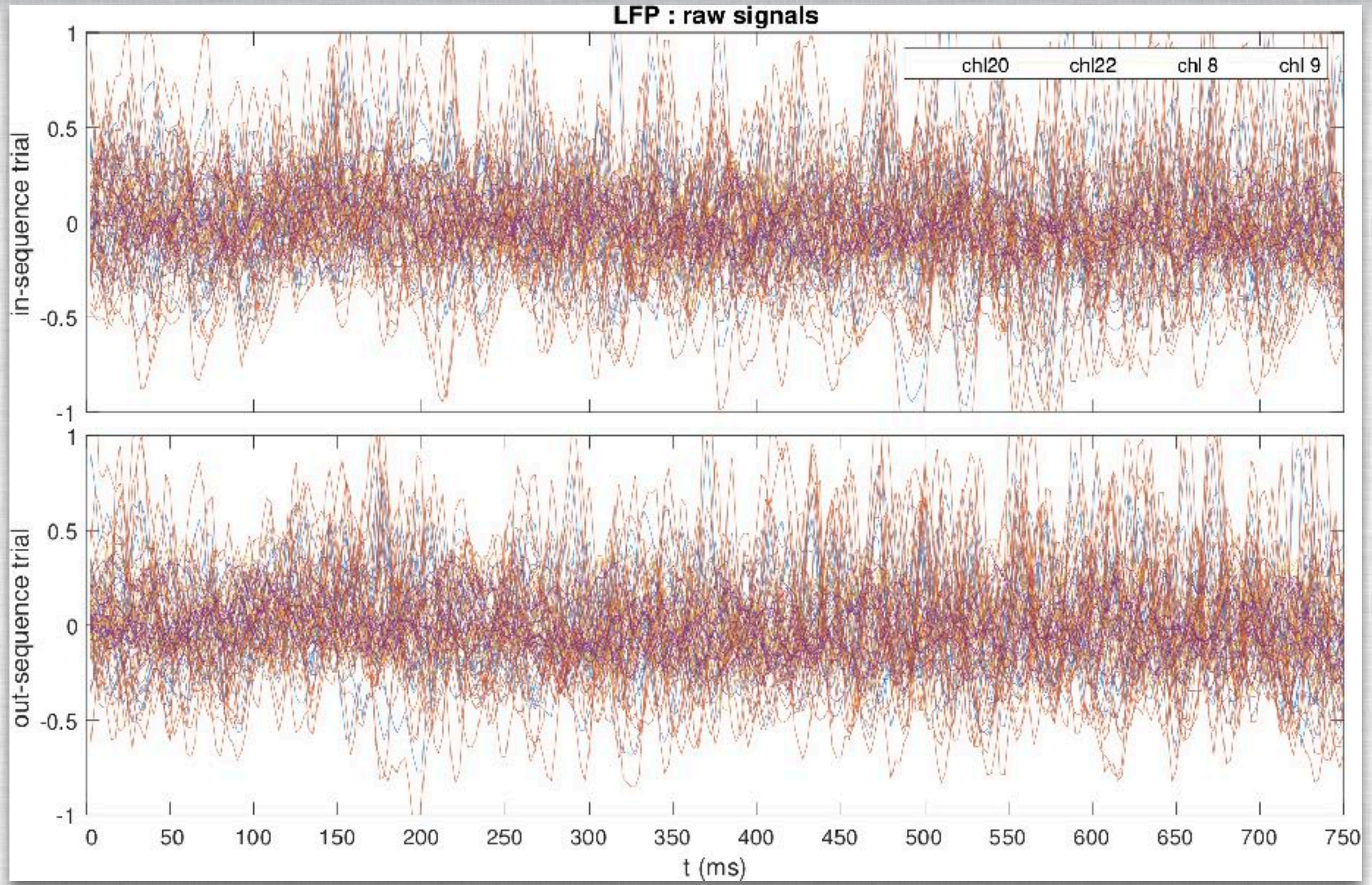
Representations of past, present, and future events are sequentially activated within each stimulus presentation



Shahbaba, Li, Agostinelli, Saraf, Cooper, Haghverdian, Elias, Baldi, Fortin (2021) *Nature Communications*

Latent Factor Modeling

Dynamic Functional Connectivity



Dynamic Functional Connectivity

- Dynamic functional connectivity is an active area of research in the neuroscience community
- Let $X_n(t)$ be a *D*-variate LFP time series for observations *n=1, …, N.*
- Let Σ_t be the *D* x *D* covariance matrix of $X_n(t)$ for observation *n* at time *t*.
- The questions of interest are
  - Does Σ_t varies significantly across *t*?
  - Does Σ_t differ significantly across experimental conditions?

Dynamic Functional Connectivity

- Previous modeling approaches include:
  - Sliding window (SW) methods, often with PCA (Lindquist et al, 2014, Leonardi et al., 2015)

    Easy to use and implement, but noisy and prone to spurious correlations

  - Variants of the Hidden Markov Model (HMM, Cabral et al., 2017)

    Great for capturing “switching-state” dynamics, but poor at capturing smoothly varying dynamics.

  - Latent factor models, such as Latent Factor Stochastic Volatility (LFSV, Lopes et al., 2004, Kastner et al, 2017)

    Stable and good for modeling volatility and smooth dynamics, but imposes a specific structure on the covariance that depends on the factor loadings on the observed signals.

Dynamic Functional Connectivity

- For multiple time series, we have

$$X_n(t) \sim \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t) \text{ where } \Sigma_t = [\sigma_{ij}(t)]_{D\times D} > 0$$

$$\mu_i(t) \sim \mathcal{GP}(0, \kappa_\mu(t,\theta)), \quad i = 1,\ldots,D$$

$$\log \sigma_{ii}(t) \sim \mathcal{GP}(0, \kappa_\sigma(t,\theta))$$

- Spatial dependence is coded in Σ_t.
- Temporal evolution is modeled by various Gaussian process (GP) models
- We need to ensure the positive-definiteness of the covariance matrix over time.
- We start by assuming $\Sigma_t = \Sigma$.

Dynamic Functional Connectivity

- We can use Cholesky decomposition of the covariance matrix:

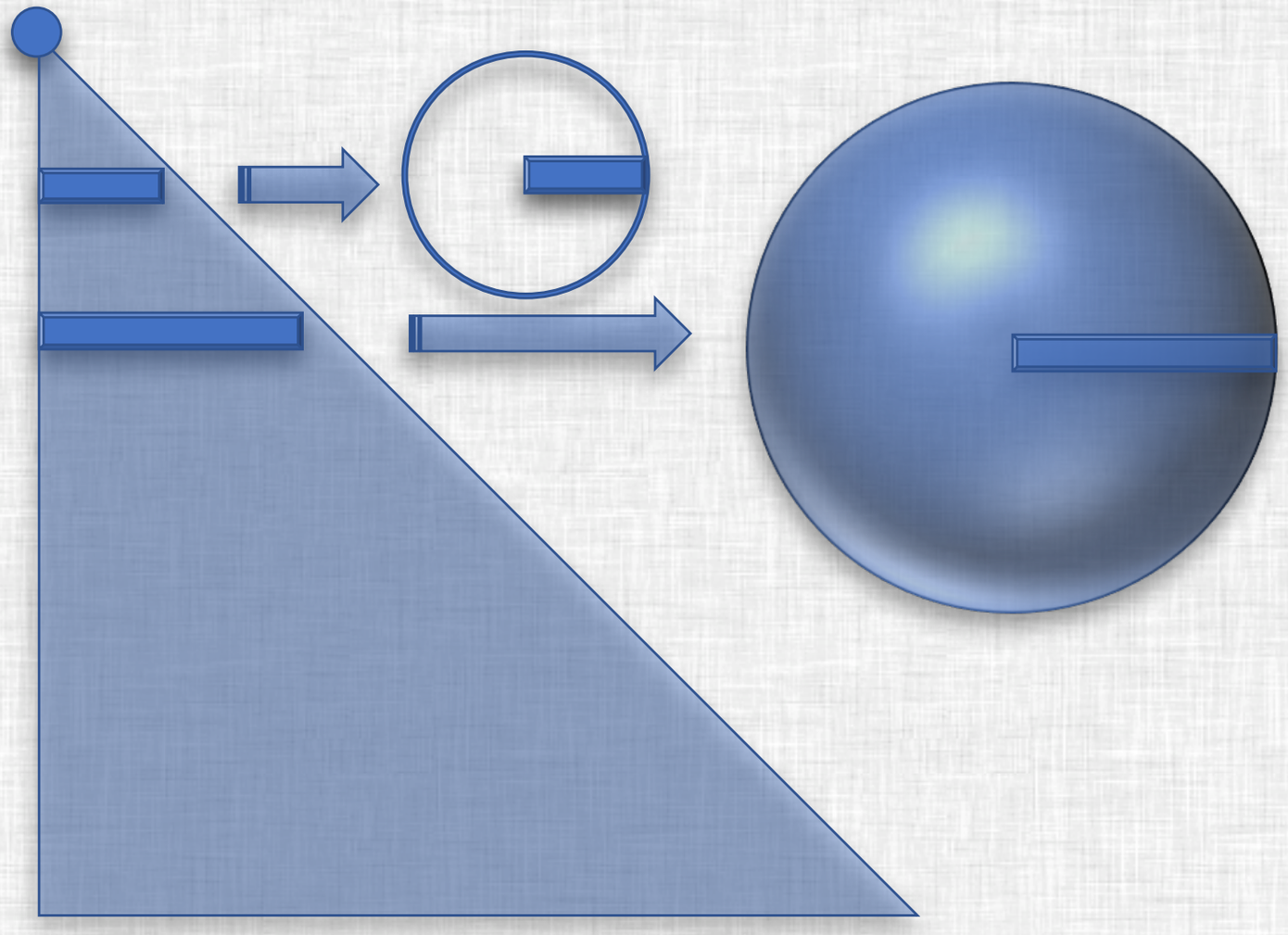
$$\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}^{\mathsf{T}}, \quad \sigma_{ij} = \sum_{k=1}^{\min\{i,j\}} l_{ik} l_{jk}, \quad \mathbf{L} = \begin{bmatrix} * & & \\ ** & & \\ * & * & * \end{bmatrix}$$

$$\sigma_i^2 := \sigma_{ii} = \sum_{k=1}^{i} l_{ik}^2 = \|\mathbf{l}_i\|^2, \quad \mathbf{L} = \begin{bmatrix} \mathbf{l}_1 \\ \vdots \\ \mathbf{l}_D \end{bmatrix}$$

- Or work on the correlation matrix instead:

$$\mathbf{P} := \operatorname{diag}(\boldsymbol{\Sigma})^{-\frac{1}{2}} \boldsymbol{\Sigma} \operatorname{diag}(\boldsymbol{\Sigma})^{-\frac{1}{2}} = \mathbf{L}^*(\mathbf{L}^*)^{\mathsf{T}}, \; \rho_{ij} = \sum_{k=1}^{\min\{i,j\}} l_{ik}^* l_{jk}^*$$

Dynamic Functional Connectivity



$$(\mathbf{l}_1, \mathbf{l}_2, \cdots, \mathbf{l}_D) \in \mathcal{S}_0^0(\sigma_1) \times \mathcal{S}_0^1(\sigma_2) \cdots \times \mathcal{S}_0^{D-1}(\sigma_D)$$

$$(\mathbf{l}_1^*, \mathbf{l}_2^*, \cdots, \mathbf{l}_D^*) \in \mathcal{S}_0^0 \times \mathcal{S}_0^1 \cdots \times \mathcal{S}_0^{D-1}$$

Lan, Holbrook, Elias, Fortin, Ombao, and Shahbaba (2019) *Bayesian Analysis*

Dynamic Functional Connectivity

- A natural spherical prior can be obtained by constraining a multivariate Gaussian random vector to have unit norm.
- Unit-vector Gaussian distribution:

$$p(\mathbf{l}_i \,|\, \|\mathbf{l}_i\|_2 = 1) = \frac{1}{(2\pi)^{\frac{i}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left\{ -\frac{1}{2} (\mathbf{l}_i - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{l}_i - \boldsymbol{\mu}) \right\}, \ \|\mathbf{l}_i\|_2 = 1$$

- Setting $\Sigma = I$, we obtain the von Mises-Fisher distribution as a special case.

Dynamic Functional Connectivity

- To model the covariance (or correlation) matrix dynamically, we use a time-varying Cholesky matrix, L_t
- Since each row of L_t has to be on a sphere of certain dimension, we consider a multivariate process, called unit-vector process (uvP), satisfying the unit-norm requirement

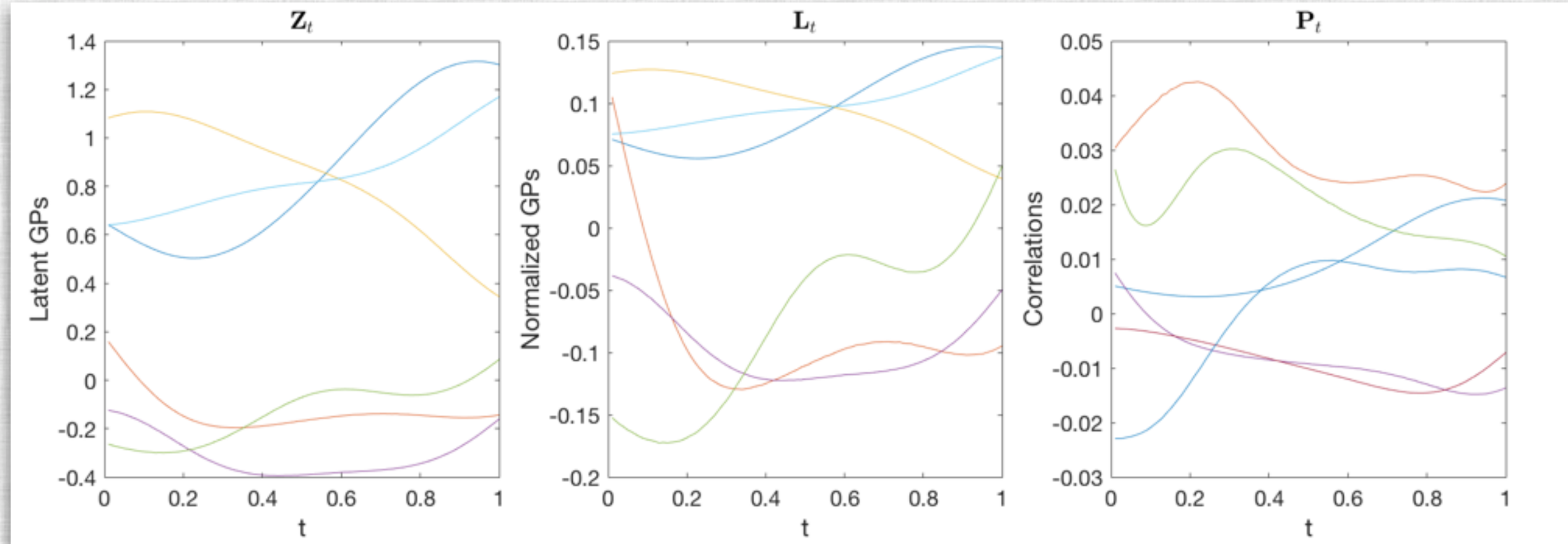
$$\|\mathbf{l}_i(t)\| \equiv 1, \quad \forall t \in \mathcal{T}$$

- More specifically, we are using A D-dimensional vector Gaussian process

$$\mathbf{Z}(x) \sim \mathcal{GP}_D(\boldsymbol{\mu}, \mathcal{C}, \mathbf{V}_{D\times D})$$

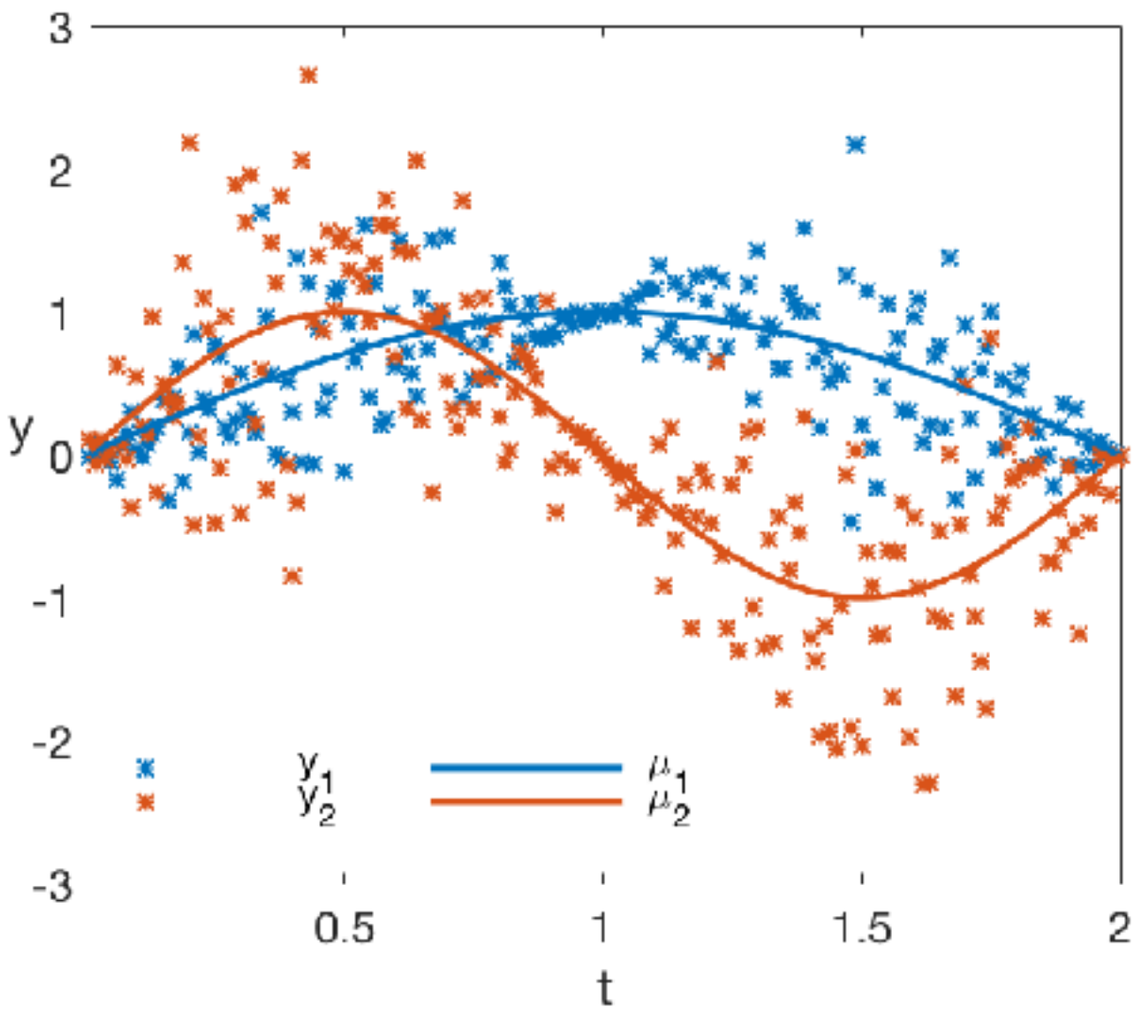
which is constrained to unit-sphere to obtain a unit-vector Gaussian process (uvGP)

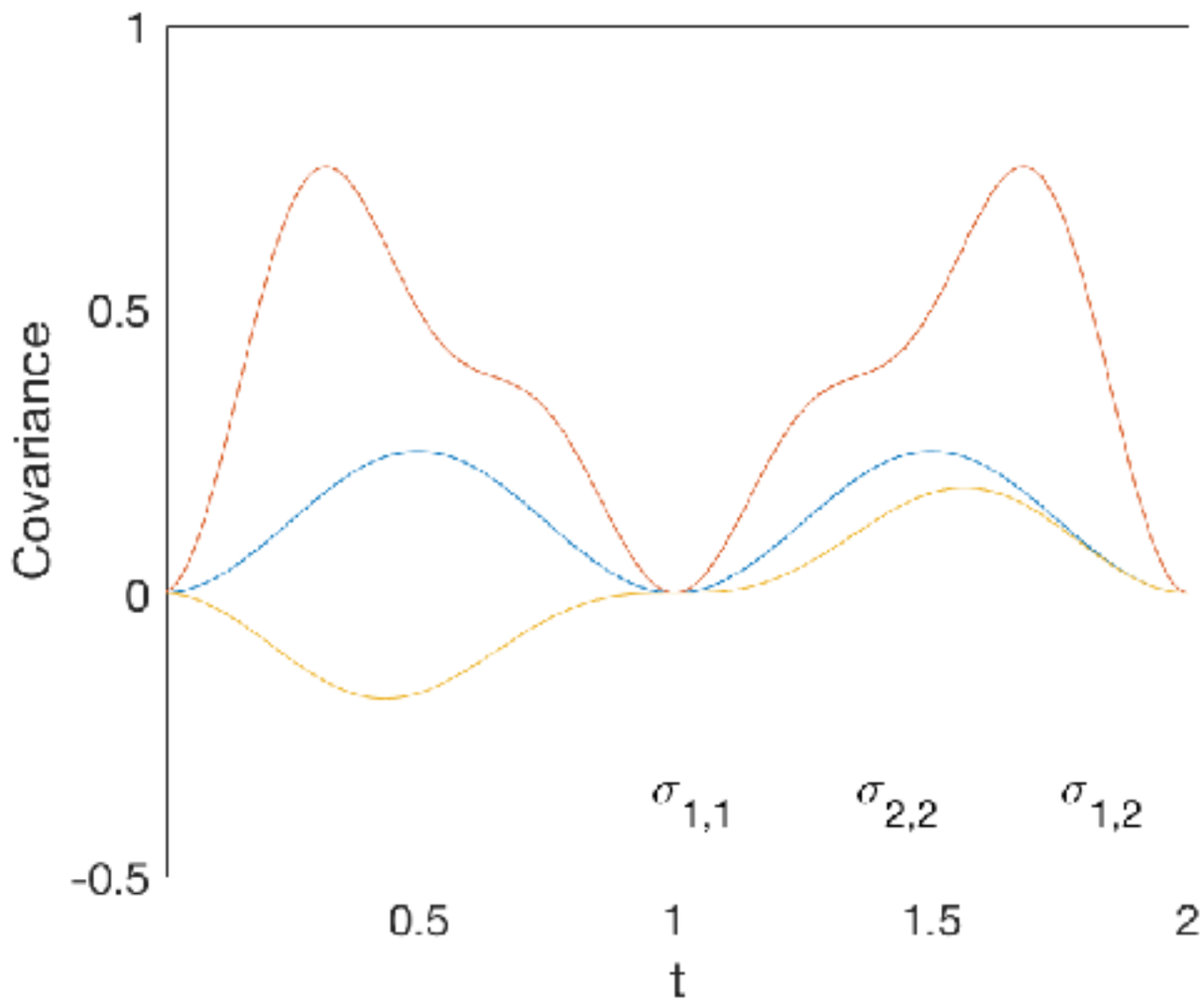
Dynamic Functional Connectivity



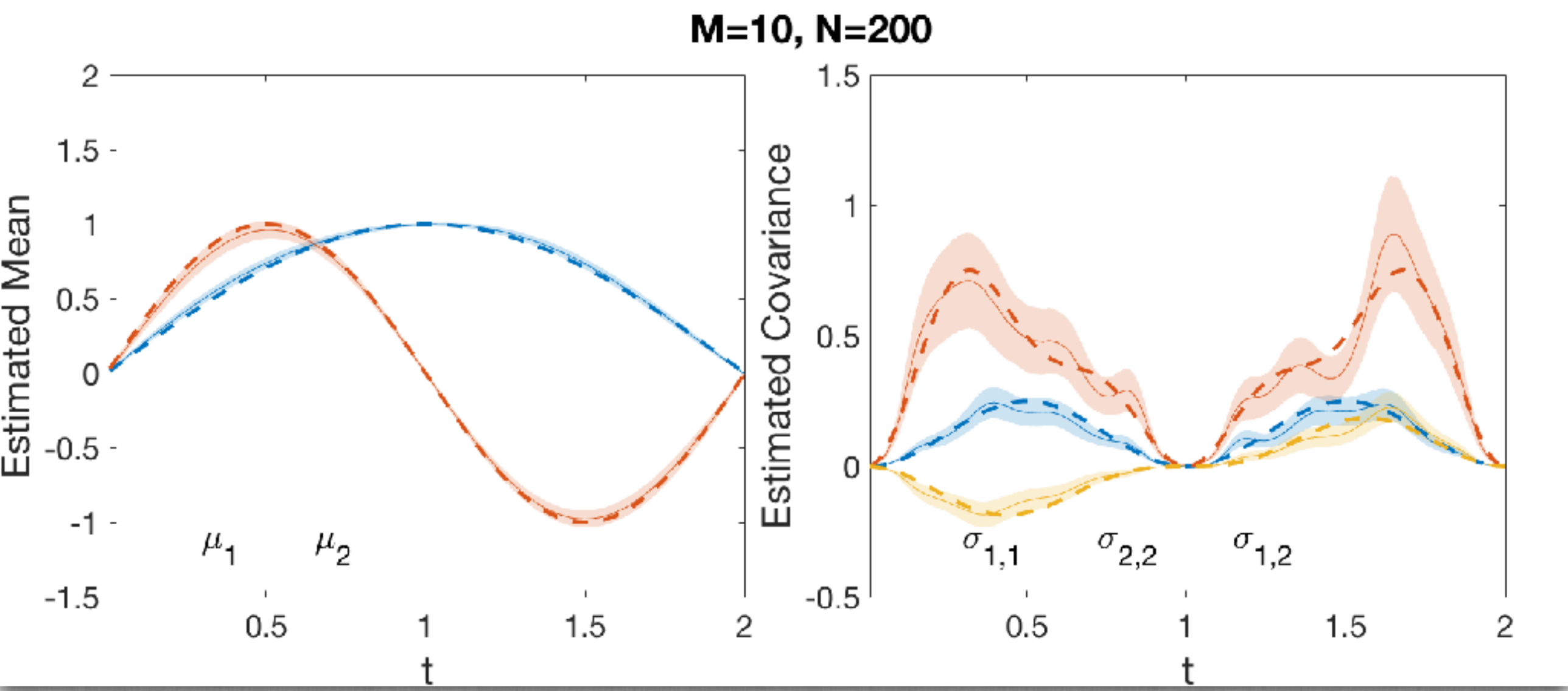
A realization of vector GP (left), its normalization (forming rows of) $\boldsymbol{L_t}$ (middle) and the induced correlation process.

Dynamic Functional Connectivity

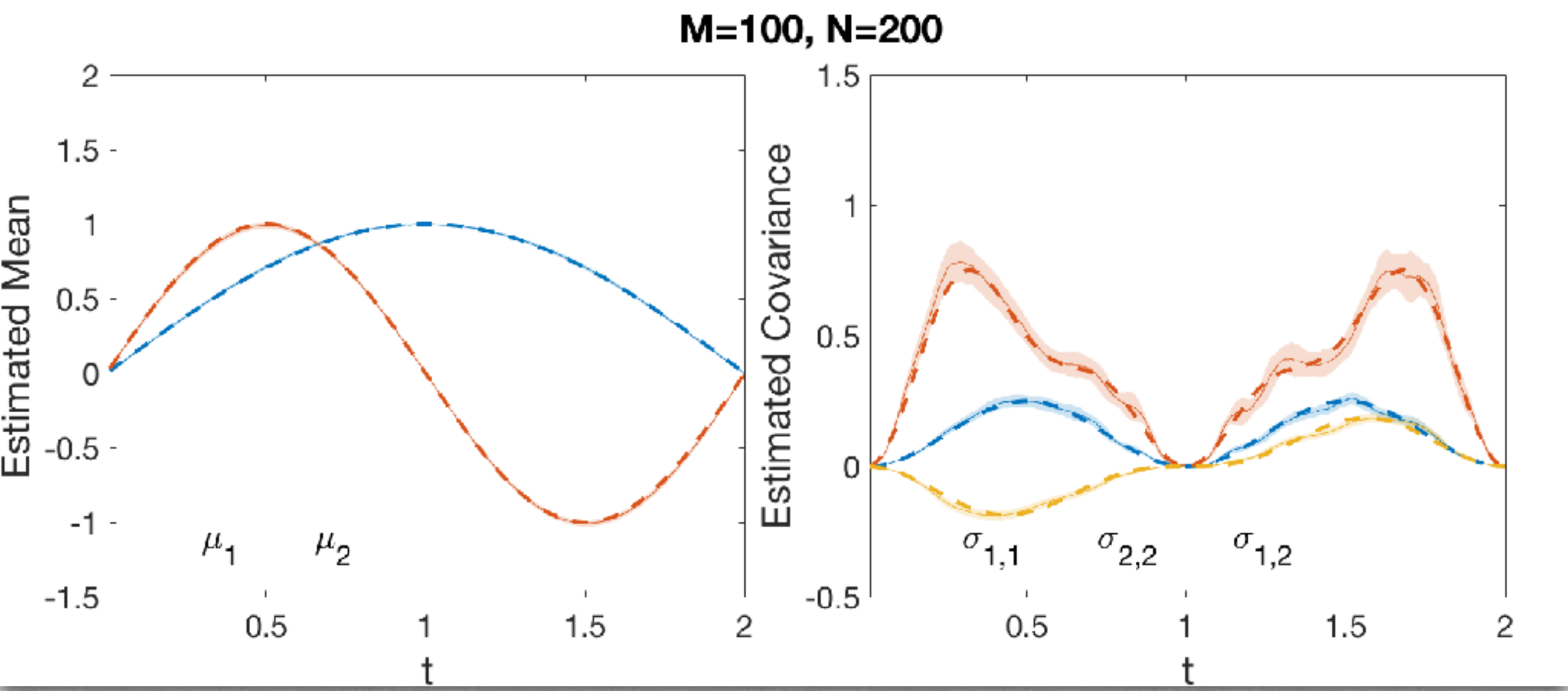




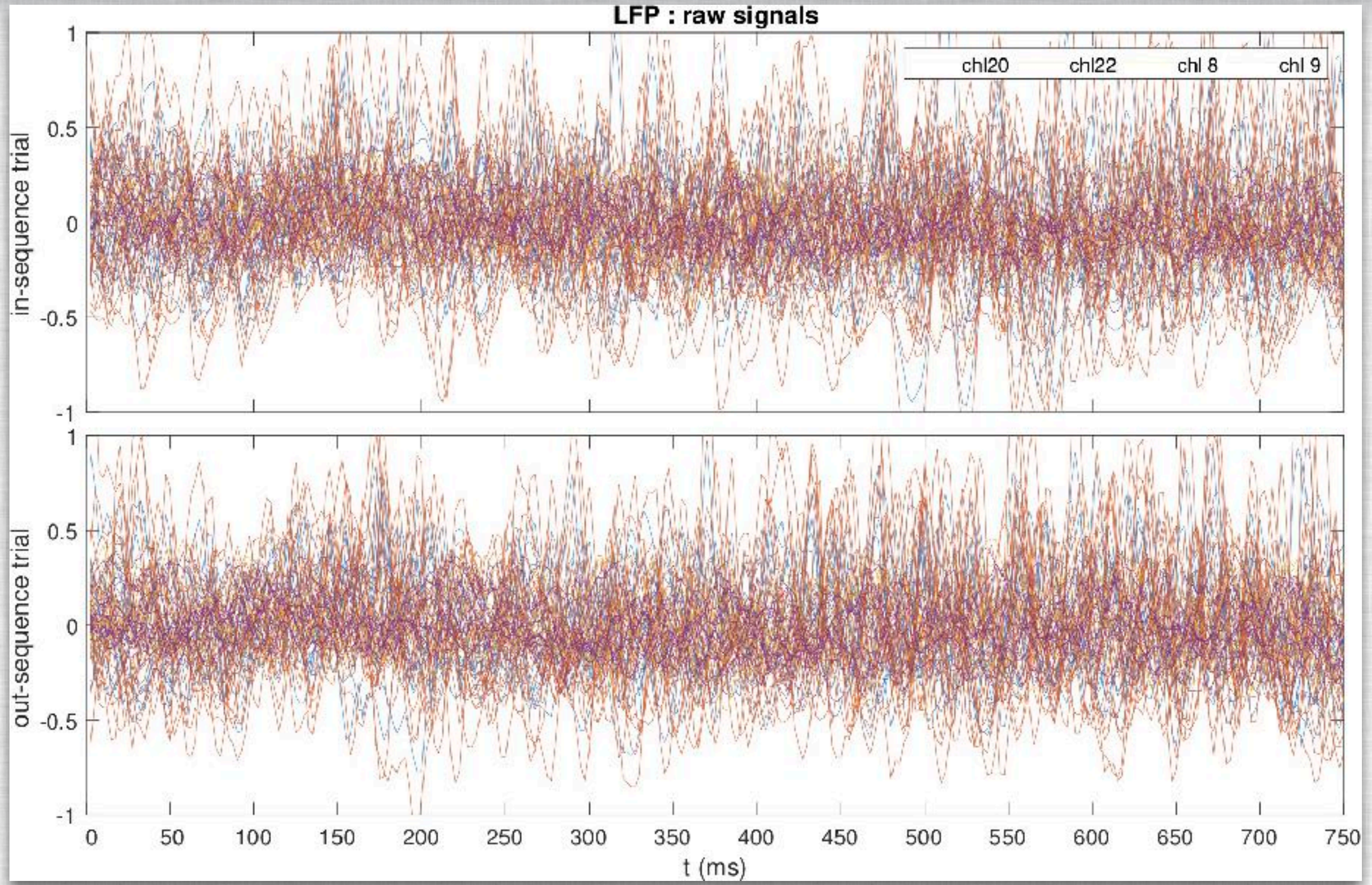
Dynamic Functional Connectivity



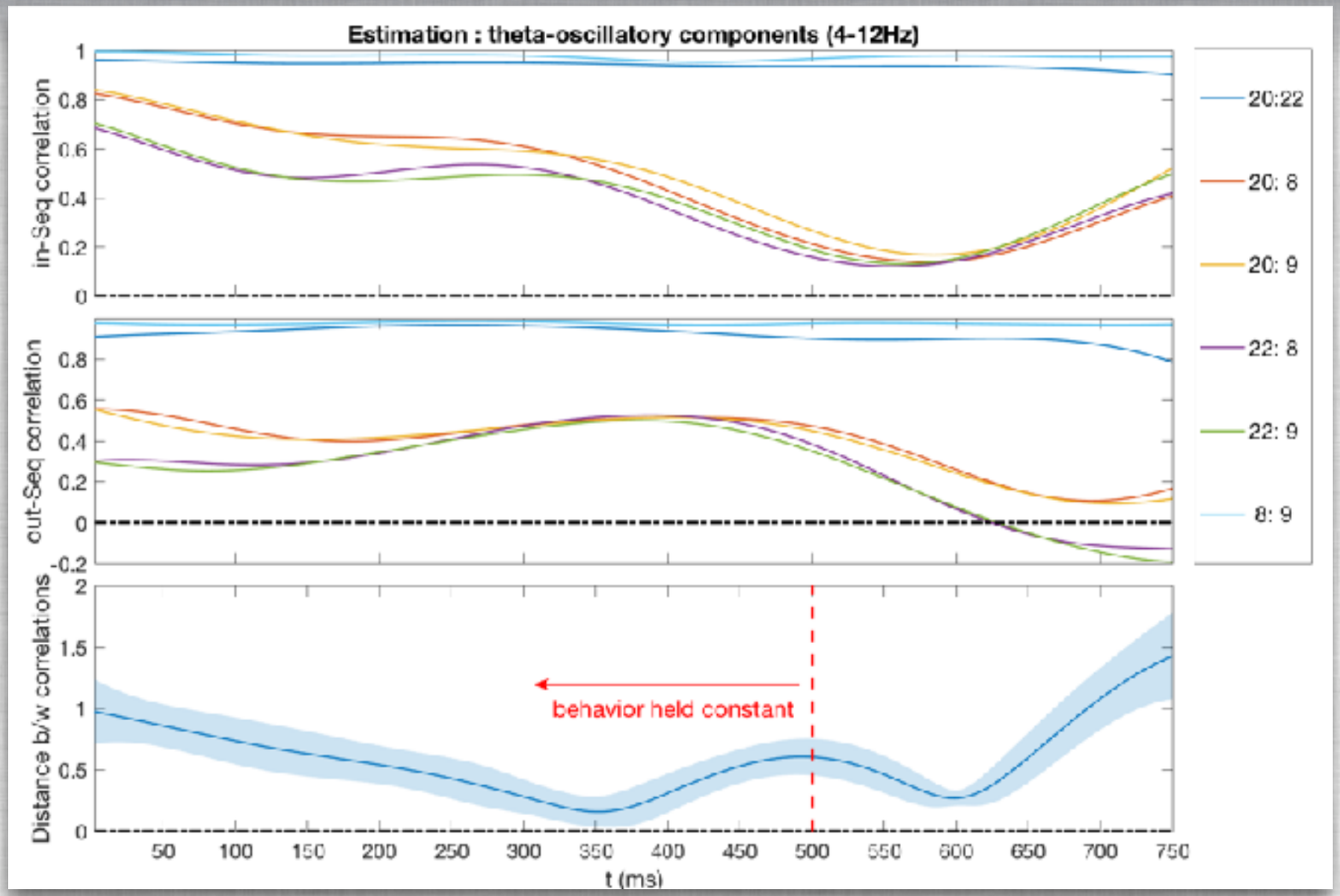
Dynamic Functional Connectivity



Dynamic Functional Connectivity

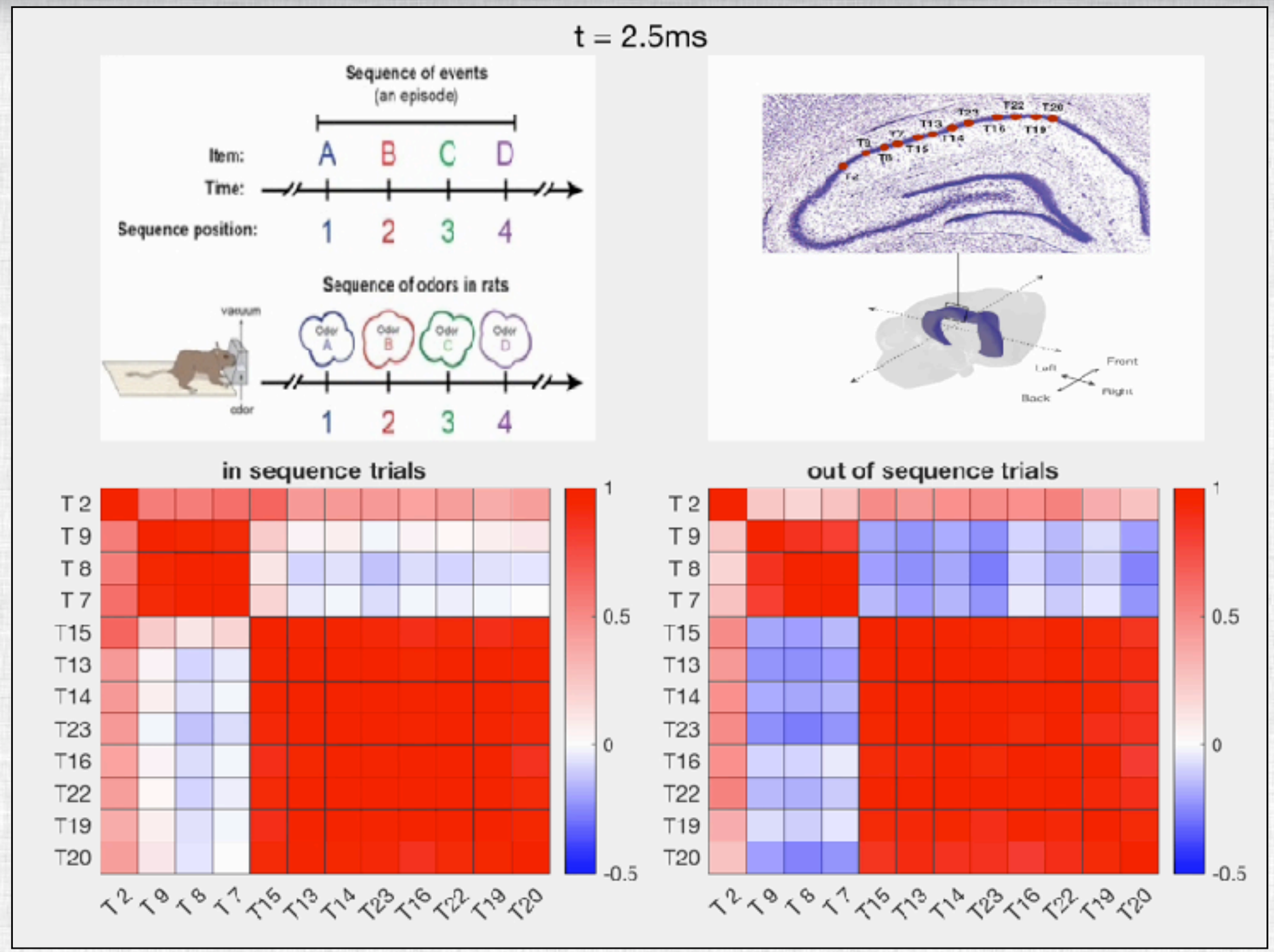


Dynamic Functional Connectivity



- Nearby electrodes (20:22 and 8:9) displayed remarkably higher correlations in LFP compared to distant pairs (20:8, 20:9, 22:8, and 22:9).
- InSeq and OutSeq activity was very similar at the beginning (e.g., before 350ms) but maximally different at the end.

Dynamic Functional Connectivity



Lan, Holbrook, Elias, Fortin, Ombao, and Shahbaba (2019) *Bayesian Analysis*

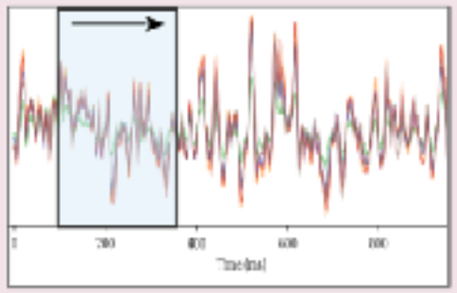
Latent Factor Gaussian Process Modeling

- The method discussed above does not scale well for high dimensional data
- Alternatively, we could use a latent factor model, where we represent the covariance process as a linear combination of factors:

1. Estimate the covariance process via sliding windows, which gives a time series of symmetric positive definite matrices.
2. Apply the matrix logarithm to the estimated covariance process, giving a time series of real symmetric matrices.
3. Fit a multivariate time series factor model to the vectorized upper triangle of the log-covariance process.

Latent Factor Gaussian Process Modeling

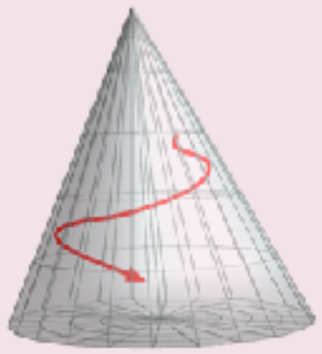
Observed LFP Signals



The covariance process from a non-stationary multivariate time series is obtained by **sliding window estimation**.

SW

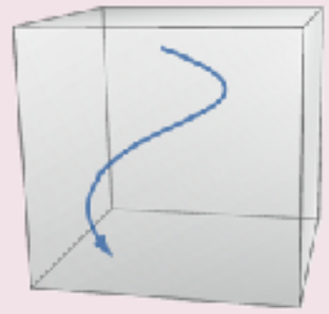
Cov. Process on $\mathbf{P}_\rho$



The estimated covariance process is a trajectory in the **manifold of symmetric positive definite matrices**.

Log

Log Process on $\mathbb{R}^q$



Applying the matrix logarithm maps to the covariance process to a **path in Euclidean space.**

LFGP

Latent Space Representation

Estimate a low-dimensional **latent space representation** of the log-covariance process.

Li, Pluta, Shahbaba, Fortin, Ombao, Baldi, (2019) *NeurIPS*

Latent Factor Gaussian Process Modeling

- The above model can be written as follows:

$$X_n(t) \sim \mathcal{D}(0, K_n(t)) \text{ where } K_n(t) = \exp(\vec{\mathbf{u}}^{-1}(Z_n(t)))$$

$$Z_n(t) = B \cdot F_n(t) + \epsilon_n \text{ where } \epsilon_n \overset{iid}{\sim} \mathcal{N}(0, \sigma^2 I)$$

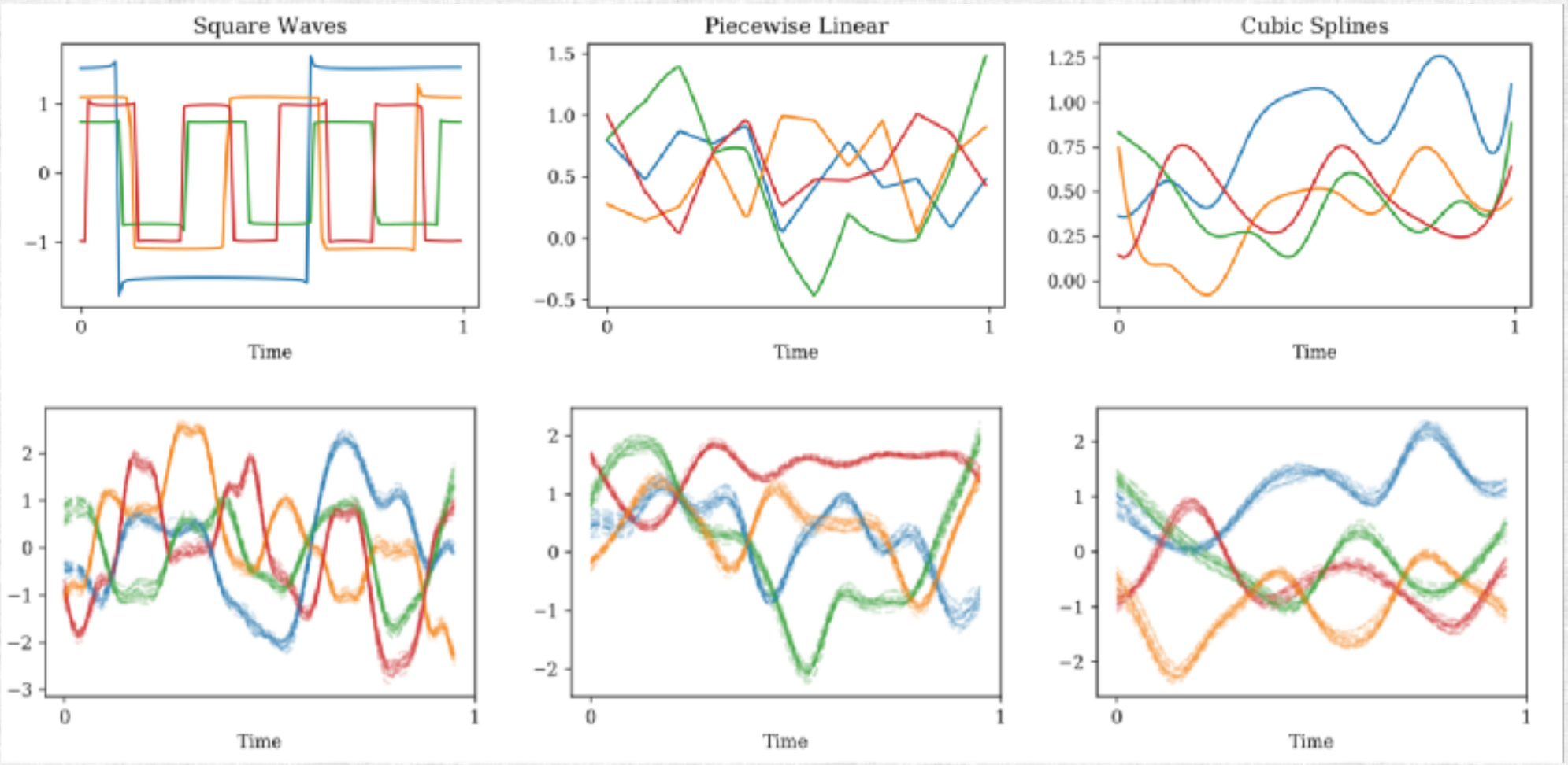
$$F_n(t) \sim \mathcal{GP}(0, \kappa(t; \theta))$$

$$B \sim p_1, \sigma^2 \sim p_2, \theta \sim p_3,$$

- Here, $Z_i(t)$ is the vectorized Log-covariance, $F_i(t)$ is the R-length factor vector, B *is* $D(D+1) \times R$ loading matrix, and p_1, p_2, p_3 are priors.

Latent Factor Gaussian Process Modeling

Simulation Results



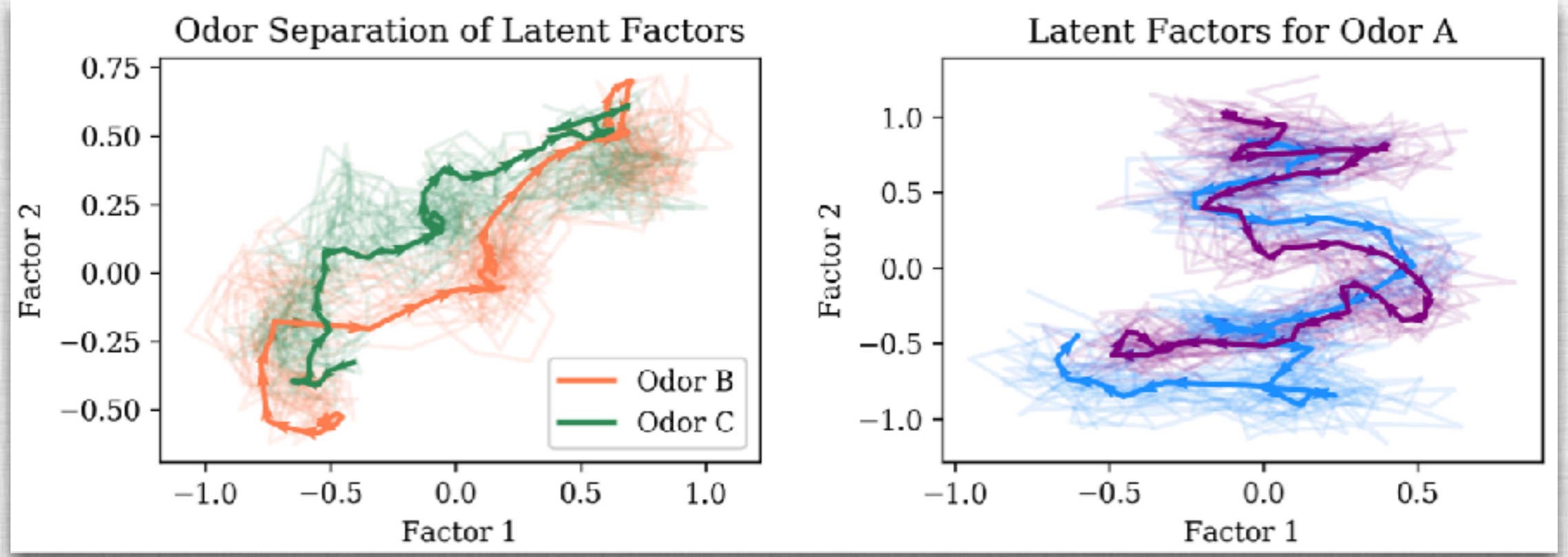
Median reconstruction loss (standard deviation) across 100 data sets

	SW-PCA	HMM	LFSV	LFGP
Square save	0.693 (0.499)	1.003 (1.299)	4.458 (2.416)	0.380 (0.420)
Piece-wise	0.034 (0.093)	0.130 (0.124)	0.660 (0.890)	0.027 (0.088)
Smooth spline	0.037 (0.016)	0.137 (0.113)	0.532 (0.400)	0.028 (0.123)

Latent Factor Gaussian Process Modeling

Real Data Analysis

- Posterior draws of median GP factors visualized as trajectories in latent space.
- For two different odors, the trajectories separate around 250ms after odor presentation.
- For similar odors, the trajectories remain close.



Li, Pluta, Shahbaba, Fortin, Ombao, Baldi, (2019) *NeurIPS*

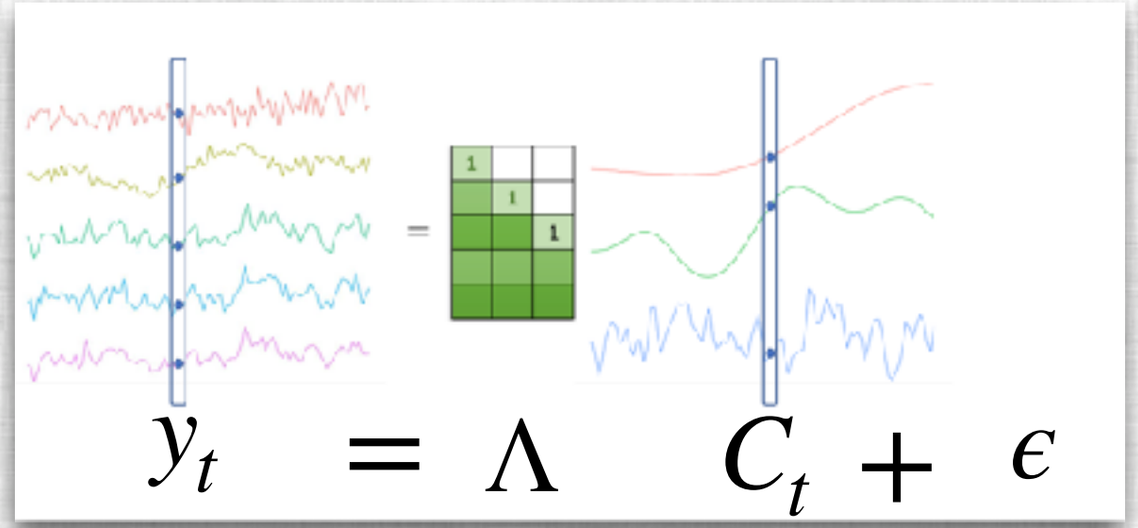
Future Directions

&

Summary

Data Integration Using LFGP

- We can model a set of time series, $Y(n \times t)$, as a linear combination of few latent Gaussian process factors.



- Λ is assumed to be lower triangular (Leorato and Mezzetti, 2020, Geweke and Zhou 1996), with all diagonal elements set to 1 to avoid sign and rotation unidentifiability.
- We assume $C_t \sim \mathscr{M}\mathscr{N}\left(0, \mathbb{I}_k, \Sigma_t\right)$, where Σ_t represents the covariance matrix of the corresponding GP.
- The above model implies $\Sigma_Y = \Lambda\Lambda^T + \Sigma_E$

Data Integration Using LFGP

- This model is flexible and could lead to interpretable results.
- It can also be interpreted as a state-space model.
- Shortcomings:
  - if t (number of time points) is large, the model estimation becomes infeasible
  - it can handle one data source at the time— no information sharing across different data sources

Data Integration Using LFGP

- To address these issues, we have extended the model for multiple sources of data as follows:

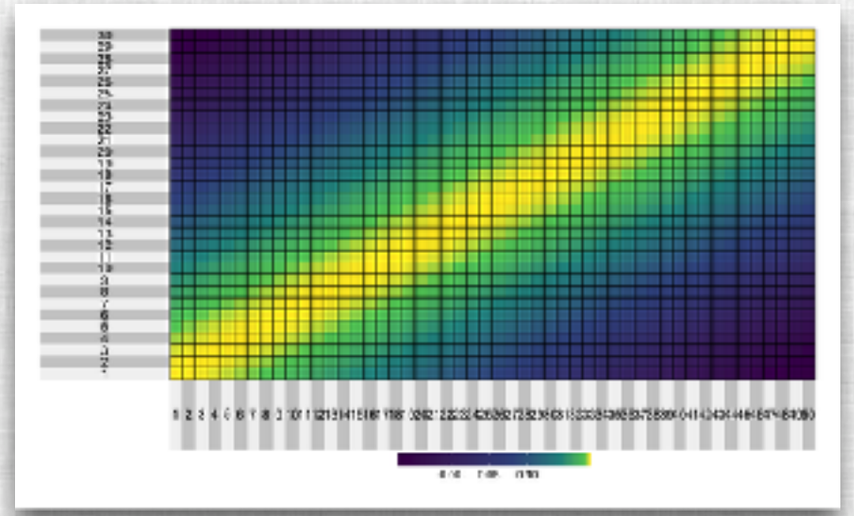
$$Y_g = \Lambda_g C \eta_g + \epsilon_g$$

- Λ_g $(n_g \times n_0)$ maps the n_g functional observations to a common number of functional factors n_0.
- C $(n_0 \times t_0)$ represents a **SHARED** collection of latent factors evaluated at t_0 time points.
- η_g $(t_0 \times t_g)$ "stretches" or "compress" the latent factors matching the original time scale.

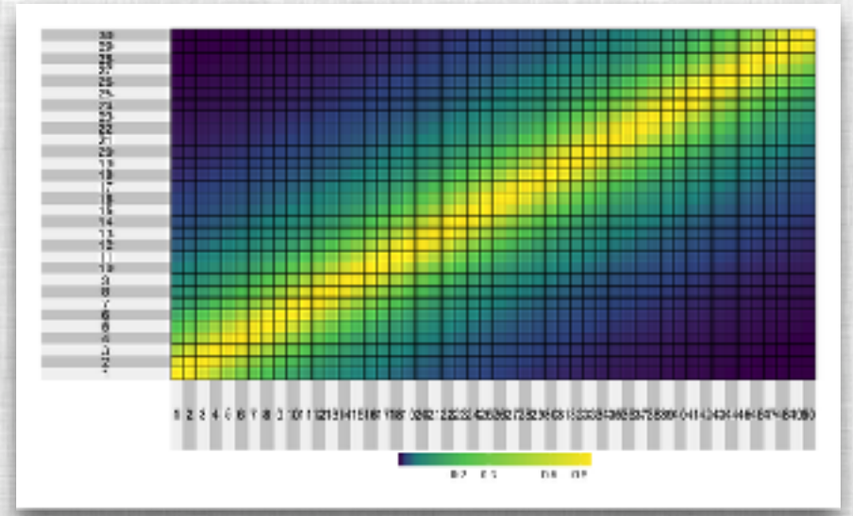
Data Integration Using LFGP

- Currently, the structure of η_g resembles a banded matrix, with values shrinking as they get far from the diagonal.
- This implies a “moving average” filtering effect on the functions in ΛC

t_0 t_1

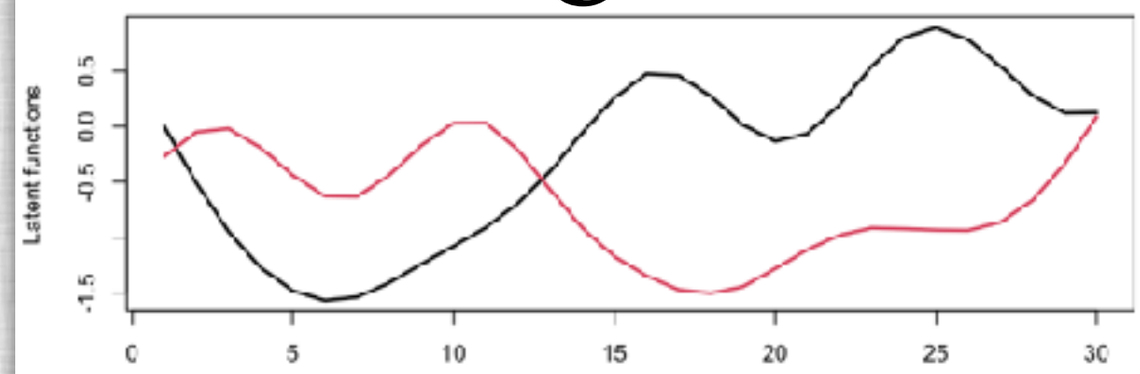


t_0 t_2

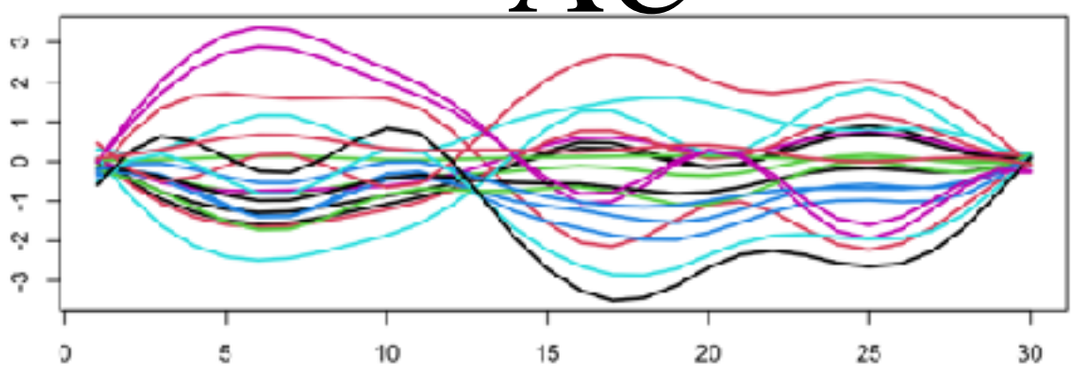


Data Integration Using LFGP

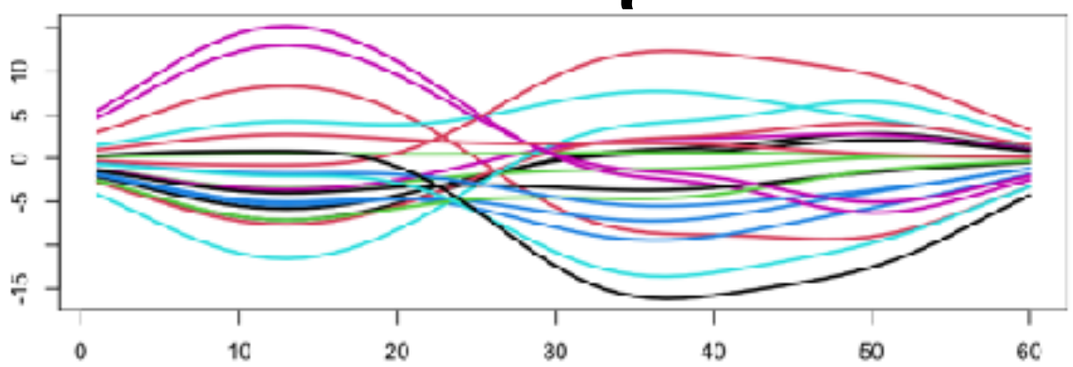
C



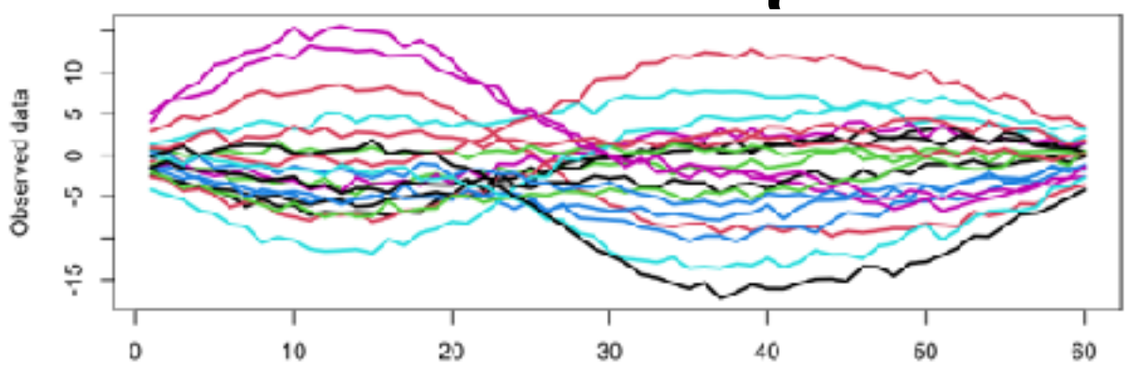
ΛC



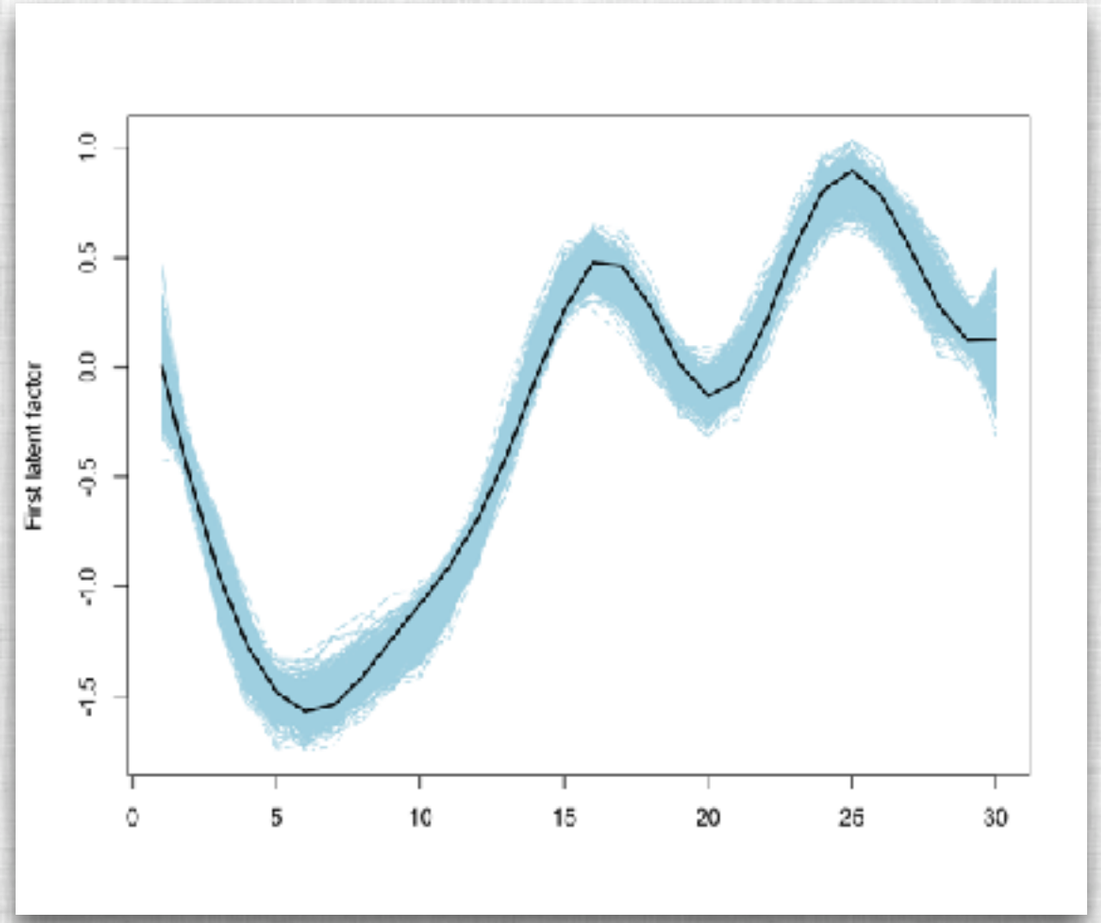
$\Lambda C\eta$

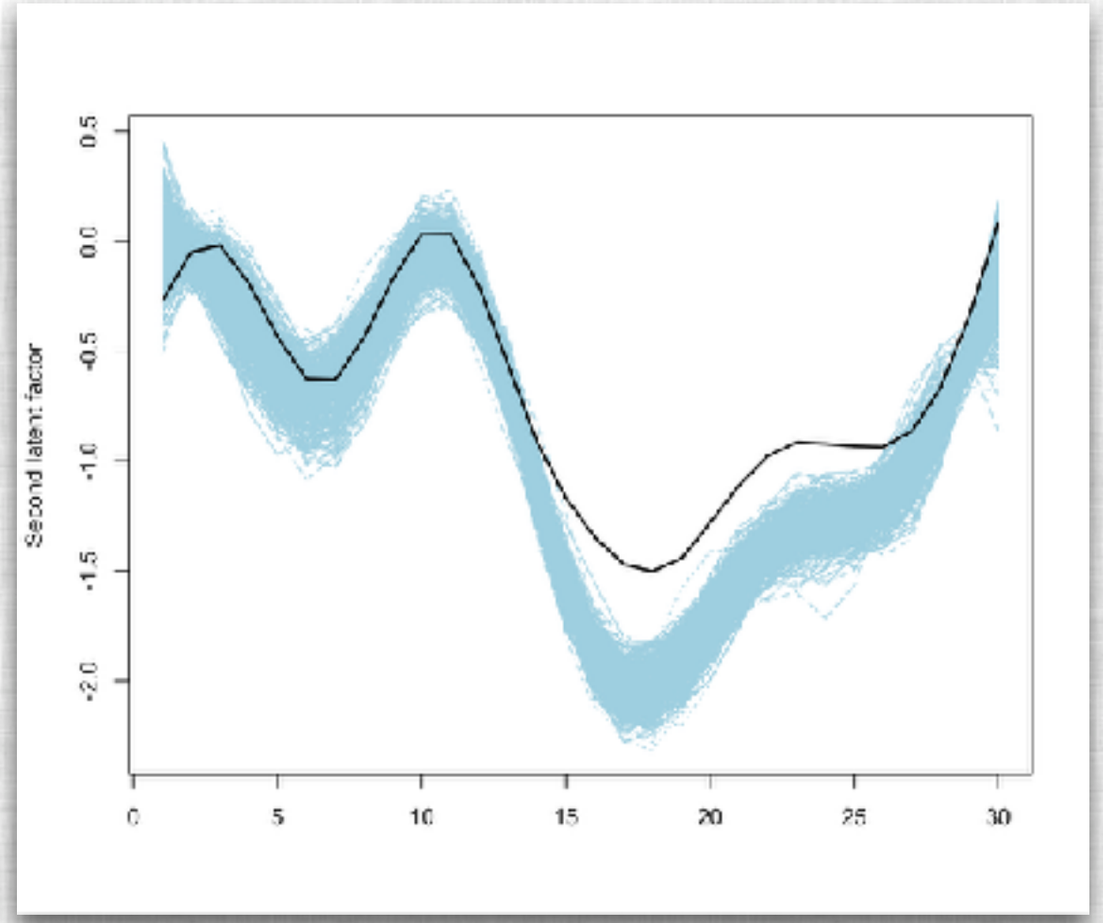


$Y = \Lambda C\eta + \epsilon$



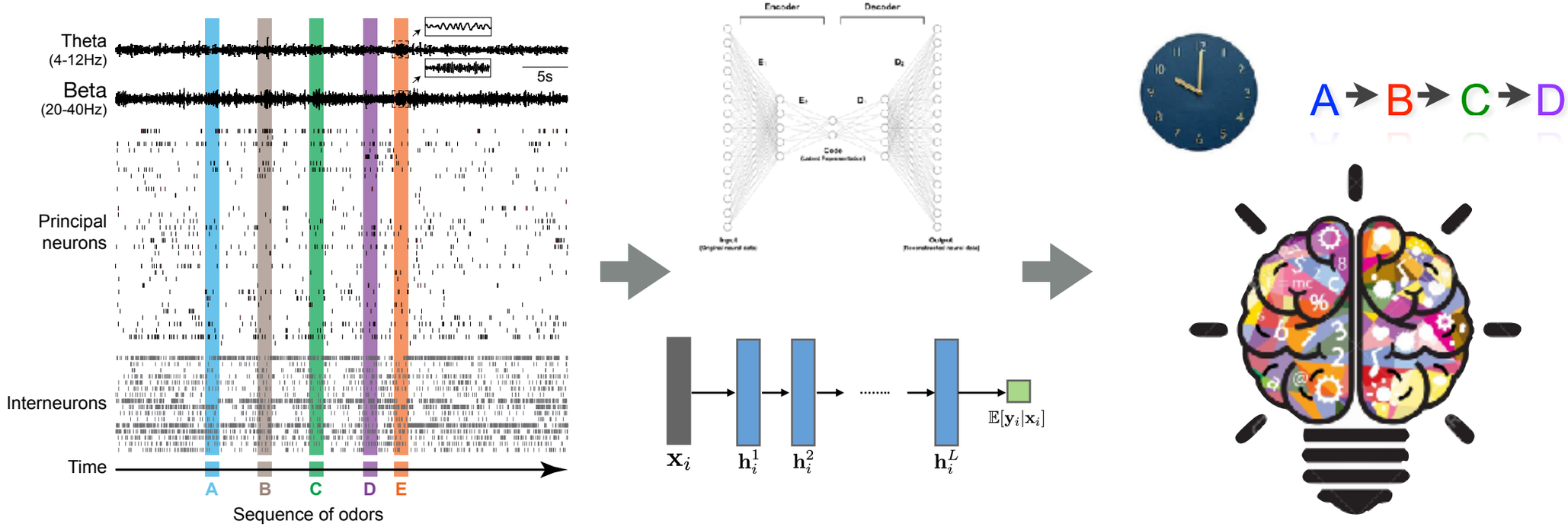
Data Integration Using LFGP



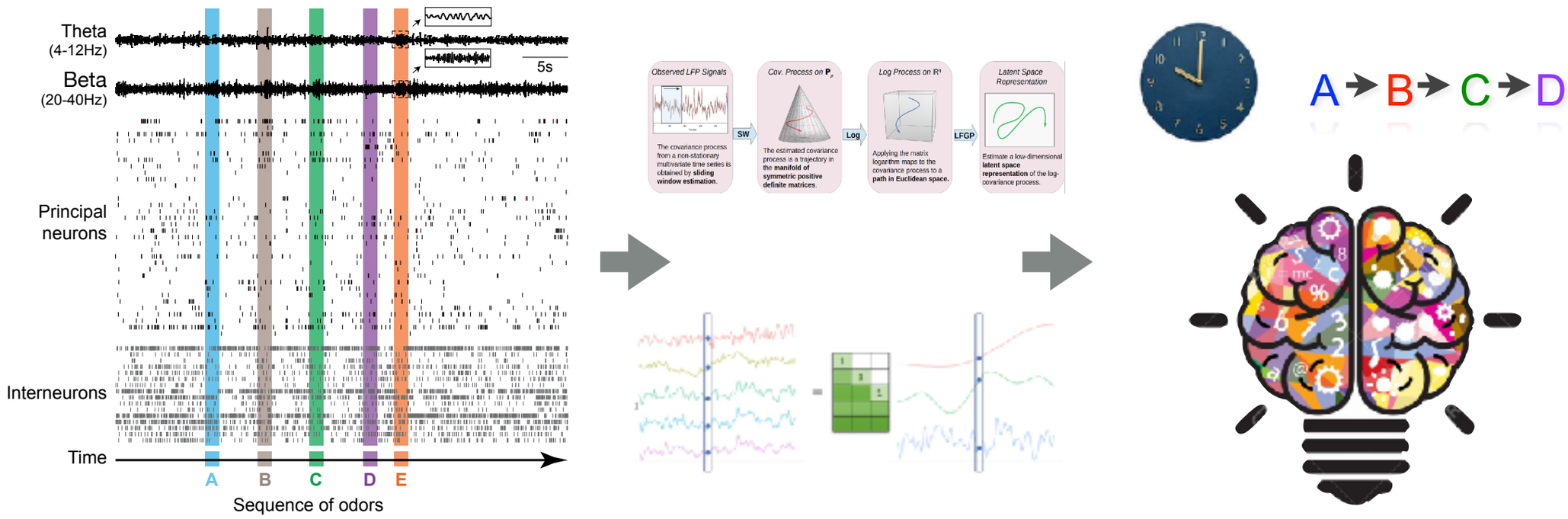


Latent Representation of Neural Data

Latent Representation Learning

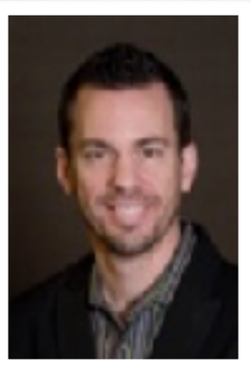


Latent Variable Modeling

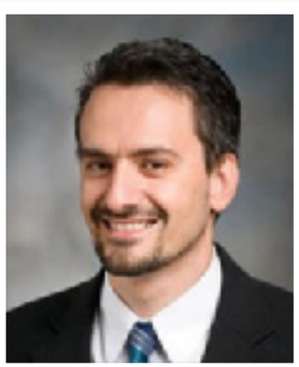


Acknowledgements

- Collaborators


Norbert Fortin


Shiwei Lan


Michele Guindani


Hernando Ombao

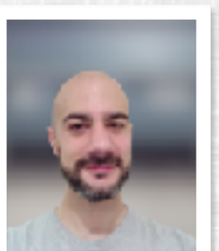

Pierre Baldi

- Postdocs and Students


Gabriel Elias
UCI


Mansi Saraf
UCI


Derenik Haghverdian
UCI


Francesco Denti
UCI


Dustin Pluta
Rice


Lingge Li
Facebook


Forest Agostinelli
U South Carolina

- Funding sources

IIS 2123366

R01MH115697

Thank You!