Probabilistic models for neural data
Session 6: Dimensionality reduction & state space models #1

To do before this session:

- **Read Williams et al. (2018) and prepare assigned presentation**
- **Read up on this session's statistical concepts**
- **Complete pre-session quiz**

In this session we will discuss Williams et al. (2018), which introduces tensor component analysis for dimensionality reduction of neural data. The paper extends PCA, and introduces a method that is a-priori not probabilistic. However, it operates on a well-defined loss function that can be interpreted as a negative log-likelihood, and so as emerging from a probabilistic model.

In the second part of this session we will consider a commonly used approximation used in probabilistic modeling known as the Laplace approximation. Furthermore, we will have a first look at state space models - Hidden Markov Models and the Kalman filter are special instances thereof - that will form the basis of the papers in the remaining sessions.

Paper: Williams et al. (2018). Unsupervised discovery of demixed, low-dimensional neural dynamics across multiple timescales through Tensor Component Analysis

When reading the paper, please focus on the following:

- What is the benefit of a low-dimensional decomposition of neural data?
- What downsides of PCA-based analyzes do the authors criticize?
- How does TCA differ from standard and trial-concatenated PCA?
- What are the benefits of TCA yielding unique, non-orthogonal solutions?
- (Advanced) What noise model does TCA assume? (see Eq. (7) of Methods)
- (Advanced) What are alternatives to the chosen neuron x time x trial decomposition?

The paper describes TCA and applies it to one simulated and two recorded datasets. Feel free to skip the macaque motor cortex application section. Two Methods sections are of particular interest to better understand the relationship between PCA and TCA. The *Matrix and Tensor Decomposition* section first discusses PCA, then introduces TCA as an extension, and describes its properties. The *Relationship Between PCA and Matrix Decomposition* section shows that PCA can be viewed as either minimizing the squared reconstruction error or finding the (linear) subspace of largest variability. Neither of these sections are compulsory reading, but provide complementary information to the main text by describing mathematically what the main text mostly describes in words.

Presentations:

1. PCA and its limitations (Text & Fig. 1A/B)
    a. Describe trial-averaged and trial-concatenated PCA applied to neural data and their interpretation (Text & Fig. 1A/B). How do either decompose the neural activity into components? (e.g., Eq. (1))

    b. What limitations of PCA are mentioned by the authors? (Text)
2. Introducing TCA (Text, Fig. 1C & Fig. 2F)
    a. Describe the decomposition of neural activity assumed by TCA (Text & Fig. 1C)
    b. Compare the decomposition to PCA by contrasting Eqs. (1) & (2). What is the benefit of adding 'trial factors'?
    c. Why do the authors consider it a benefit that TCA does not require orthogonality constraints to yield a solution? Can you think of situations when this non-orthogonality might be a disadvantage?
    d. Describe the method that the authors suggest to choose the number of TCA components (Fig. 2F).
    e. (Advanced) TCA is trained by minimizing the reconstruction error Eq. (7). Can you interpret this error probabilistically as noise? What distributional assumptions are made about this noise?
3. TCA applied to simulated non-linear networks (Fig. 3)
    a. Briefly describe the task that the network is trained to perform (Fig. 3A/B).
    b. Describe and interpret the single TCA component (Fig. 3C-E).
    c. Describe what TCA reveals about how synaptic weights within the network change over time, and the simpler network structure that this suggests (Fig. 3G/H).
4. TCA applied to neural activity during spatial navigation (Fig. 4)
    a. Describe the task the mice had to perform (Fig. 4A/B), and the collected neural data (Text, Fig. 4C-E).
    b. Describe their measures quantifying fit quality of TCA and nonnegative TCA (Fig. 4F-I,K). Why did they consider nonnegative TCA preferable to TCA for this dataset?
5. Relating TCA components to task variables (Fig. 6)
    a. On hand of this figure, revisit how the different factors are combined to model neural activity within and across trials (Fig. 6 & Eq. (2)).
    b. Describe how the authors relate task variables to TCA components. How can they be interpreted to modulate neural activity? Is it surprising that they find that some TCA factors to cluster by task condition?
6. Overfitting in TCA (Fig. 5)
    a. Describe the TCA cross-validation procedure (Fig. 5A).
    b. Compare the reported train/test set errors (Fig. 5B,C). What is the author's explanation for the lack of overfitting?

Statistical concepts: Laplace approximation and models for sequential data

Exact inference and parameter estimation for such models is intractable in most cases. Thus, they require the use of approximations. To understand the next session's paper, we need to go through the Laplace approximation, a common approximation to make inference tractable, and models for sequential data.

Laplace approximation (PRML 4.4)
When reading up on the Laplace approximation, which is a standard approximation in Bayesian modeling, make sure to understand the assumptions underlying this approximation, as well as when these assumptions are violated. Consider, for example, what happens when applying the Laplace approximation to a multimodal or a heavily skewed distribution. The application of the Laplace approximation (PRML 4.4.1) to model comparison is interesting, but not essential for this course. Feel free to skip this section.

Models for sequential data (PRML 13)
The essential concept that makes sequential models tractable are Markov Chains. Being able to describe the time-evolution of a system's state (e.g. a stimulus' location) by a Markov Chain implies that the system's future state is fully determined by its current state, and does not require any knowledge of the system's past states. In the language of conditional probabilities, *p*(future states | current state, past states) = p(future state | current state).

PRML 13 first introduces these Markov Chains (have a look at sections 13.0 and 13.1 for an introduction to Markov Chains), and then a pair of models (Hidden Markov Models and the Kalman Filter) in which the *latent* state sequence is only observable through noisy observations. Both models are conceptually the same, and only differ in their specific assumptions about latent states and their relations to the observables. They are closely related to neural state space models, which assume that an unobservable sequence of low-dimensional latent states can be observed by the activity of a neural population that encodes this state (noisily). In general, only the two types of such state space models discussed in PRML 13 are tractable without approximations.

PRML 13.2 discusses Hidden Markov Models (HMMs) in which the latent states are discrete (i.e., they can take one of a finite set of values). In this section, focus on the following:

- The basic definition of HMMs, as described in the section's introduction (PRML 13.2.0).
- The forward-backward algorithm (PRML 13.2.2). In particular, make sure to understand the two steps underlying the forward pass (i.e. to compute the α's) that represent predicting the next state given all past observations, and that update this prediction in the light of a new observation.
- You should understand the purpose of the Viterbi algorithm (PRML 13.2.5), but you can skip the math details.

You should read PRML 13.2.1 (an application of the EM algorithm to HMM parameter learning) and skim over PRML 13.2.3, but don't need to go into the details of the math, and you can skip PRML 13.2.4, which only becomes important if you want to implement the various algorithms yourself. PRML 13.2.6 provides an interesting discussion on possible HMM extensions, but is not required reading for the purpose of this course.

PRML 13.3 introduces Linear Dynamical Systems, which are better known as the Kalman Filter or Kalman Smoother. They assume a continuous latent state with linear-Gaussian dynamics, and linear-Gaussian observations. As conceptually the same as HMMs, you should have a read through the section on how to perform inference and identify parallels to HMMs, but can skip all remaining sections (PRML 13.3.2-13.3.4). In particular, you should focus on:

- The difference between Kalman Filtering and Kalman Smoothing, and their purpose.

- How these two inference modes relate to the forward pass and the forward-backward algorithm in HMMs.
- As for HMMs, the two steps underlying the forward pass / Kalman Filtering.

Don’t worry too much about the math underlying Kalman Filters/Smoothers. What is most important is to see the parallels to HMMs, which reveals the concepts underlying all state space models.