# PCA, CCA & Systems Identification

Lucas Pinto
Assistant Professor
Department of Neuroscience
Northwestern University

NUIN 443, 03/27/2024

# Outline

- SVD & Eigendecomposition
- Principal components analysis
- Primer: canonical correlation analysis
- Systems identification

# Refresher: SVD and PCA

# Singular Value Decomposition

Any complex matrix *M* of size *m* x *n* can be decomposed as:

$$M = U\Sigma V^*$$ (or $V^T$ if *M* is real)

$m \times n$

$m \times m$

$U^*U = UU^* = I$

columns of *U*:

*left singular vectors*

(orthogonal basis)

$n \times n$

$V^*V = VV^* = I$

columns of *V*:

*right singular vectors*

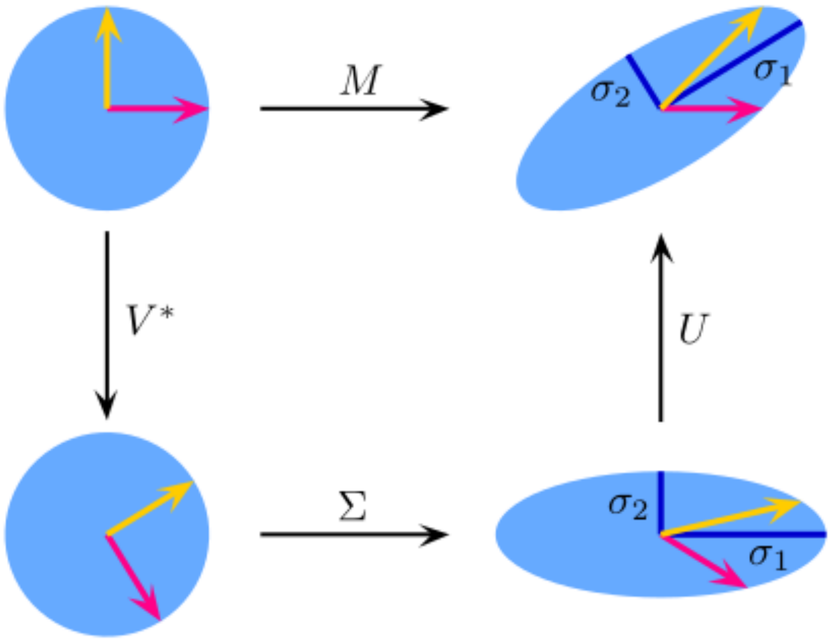
(orthogonal basis)

$$\underset{m \times n}{\Sigma} = \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & \sigma_n \\ \vdots & \vdots & \vdots \end{pmatrix}$$

\Sigma is uniquely determined by *M* -> \sigma_i are the *singular values*

The number of \sigma_i > 0 is the rank of *M*

**Geometric interpretation**




*wikipedia*

# Eigendecomposition

If *M* is square (*n* x *n*) and diagonalizable

$$M = Q\Lambda Q^{-1}$$

*eigenvectors*

$$Q = \begin{pmatrix} \vec{q_1} & \vec{q_2} & \dots & \vec{q_n} \end{pmatrix}$$

*(orthogonal)*

*eigenvalues*

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & \lambda_n \end{pmatrix}$$

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$$

A vector **q** is said to be an eigenvector of M if

$$M\vec{q} = \lambda\vec{q}$$

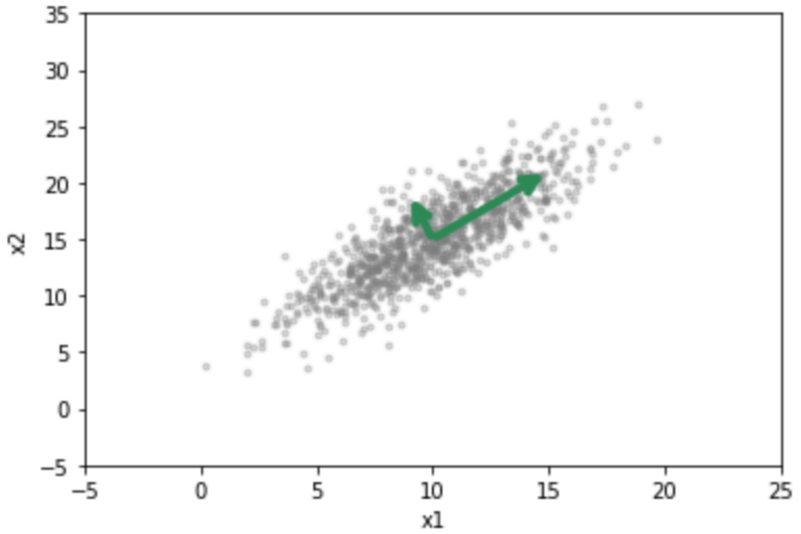
Equivalently, for a matrix of **q**'s

$$MQ = Q\Lambda$$

$$M = Q\Lambda Q^{-1}$$

# PCA: overview

We want to linearly transform (rotate) the coordinate system into a set of orthogonal dimensions.




We do this by successively finding projecting the data along the axis of highest variance, given by the vector *z*.

$$z = Xw$$

Assuming the data area centered

$$\underset{||w||=1}{\text{argmax}}\{||Xw||^2\}$$

$$\underset{||w||=1}{\text{argmax}}\{(Xw)^T Xw\}$$

$$\underset{||w||=1}{\text{argmax}}\{w^T X^T Xw\}$$

- It can be shown that *w* is the largest eigenvector of the covariance matrix $X^TX$, and is the first PC of the data

The next PC is found by repeating the procedure on a matrix with PC1 subtracted (and so forth)

$$X_2 = X - Xw^T w$$

- Note that the centering means that we rotate x about the mean

# PCA via eigendecomposition of $X^TX$

More generally, $Z = XW$

$n \times d$

$n \times d$

$Z = \begin{pmatrix} \overrightarrow{z_1} & \overrightarrow{z_2} & \cdots & \overrightarrow{z_n} \end{pmatrix}$

(a.k.a. PC scores)

$d \times d$

Columns of *W*, the PCs are the eigenvectors of the data covariance matrix

$$X^TX = C = W\Lambda W^{-1} = W\Lambda W^T \quad \text{(because } C \text{ is real symmetric)}$$

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & \lambda_n \end{pmatrix} = \begin{pmatrix} \sigma_1^2 & 0 & 0 \\ 0 & \sigma_2^2 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & \sigma_n^2 \end{pmatrix}$$

The loading of each datapoint onto each PC is given by $\overrightarrow{w_i} \times \sqrt{\lambda_i}$

So we can simply compute *C* and perform eigendecomposition. But computationally expensive!

# PCA via SVD of *X*

A much more common implementation is via SVD, bypassing the expensive computation of $X^TX$

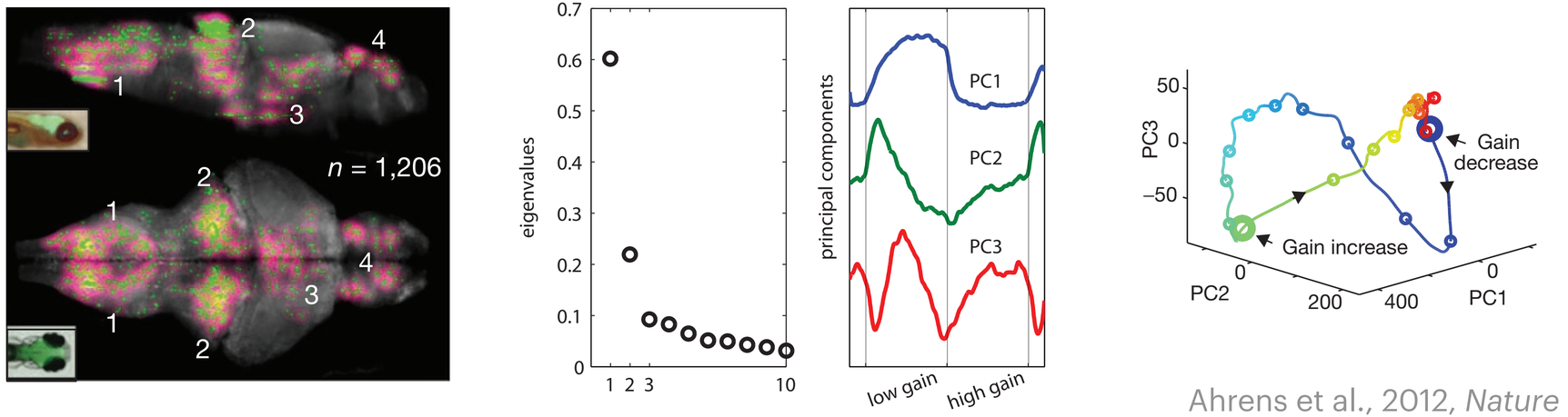
$$X = U\Sigma W^T$$

$$\begin{aligned} X^TX &= (U\Sigma W^T)^T U\Sigma W^T \\ &= W\Sigma^T U^T U\Sigma W^T \\ &= W\Sigma^T\Sigma W^T \\ &= W\Sigma^T\Sigma W^T \\ &= W\hat{\Sigma}^2 W^T \end{aligned}$$

- \hat{\Sigma^2} is a square diagonal matrix without the bottom rows of 0's, with the singular values of *X*, equal to the sqrt() of the eigenvalues of $X^TX$
- W is equivalent to eigenvectors of $X^TX$
- And the PC scores: $Z = XW = U\Sigma W^TW = U\Sigma$

# PCA as dimensionality reduction

Neural dynamics are often much lower-dimensional than full activity space (but that needs to be interpreted w.r.t. behavior)




Ahrens et al., 2012, *Nature*

Because we transform the data into components of progressively smaller variance, we can discard the trailing PCs to reduce the dimensionality of the data

Similarly, we can project the data down to k dimensions to reconstruct, denoise etc.

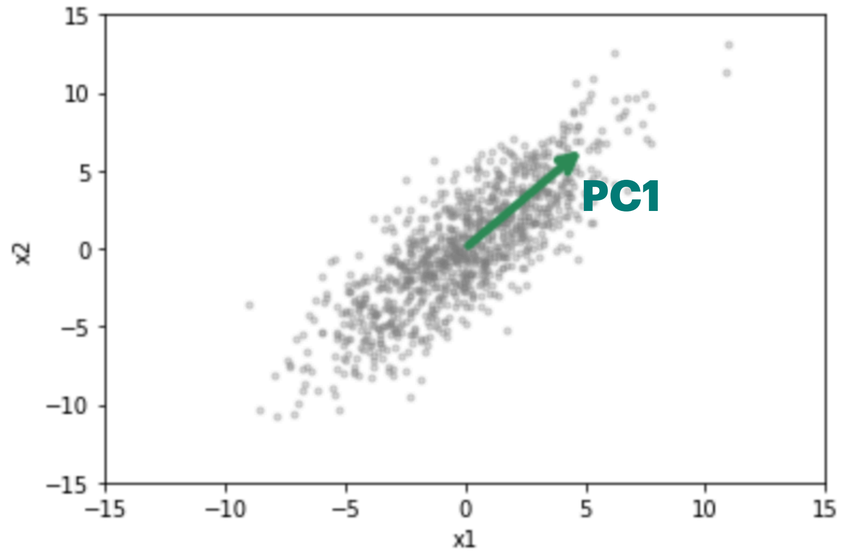
$$Z = XW \qquad \tilde{X} = Z_k W_k^T$$

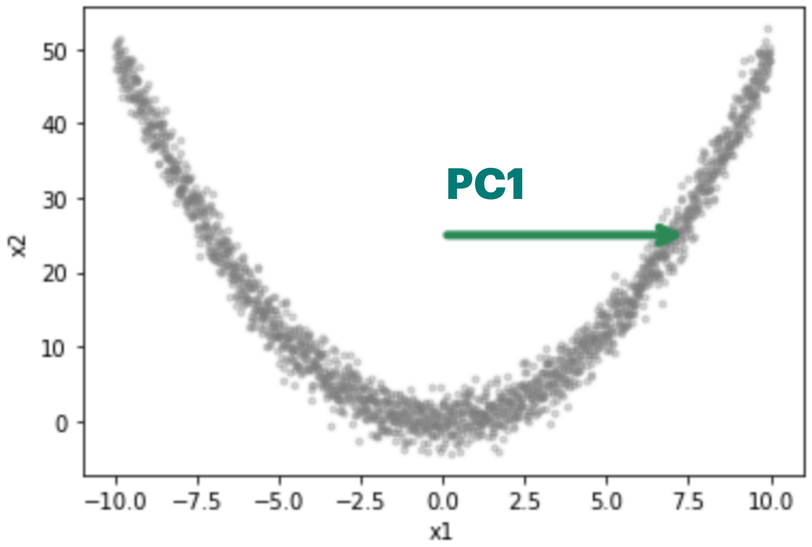
$$n \times d \;\; n \times d \;\; d \times d \qquad n \times d \;\; n \times k \;\; k \times d$$

# Some applications in neuroscience

- PCA: dimensionality reduction for clustering, e.g. spike sorting
- PCA/SVD: dimensionality reduction for feature selecting of high-D datasets going into a GLM (e.g. movies)
- PCA: visualizing / analyzing high-D neural dynamics projected onto a low-D space
- SVD/Eigen: analyzing connectivity matrices of RNNs
- SVD/Eigen: Systems identification (next)
- And many others

# Reminder: these are linear methods!



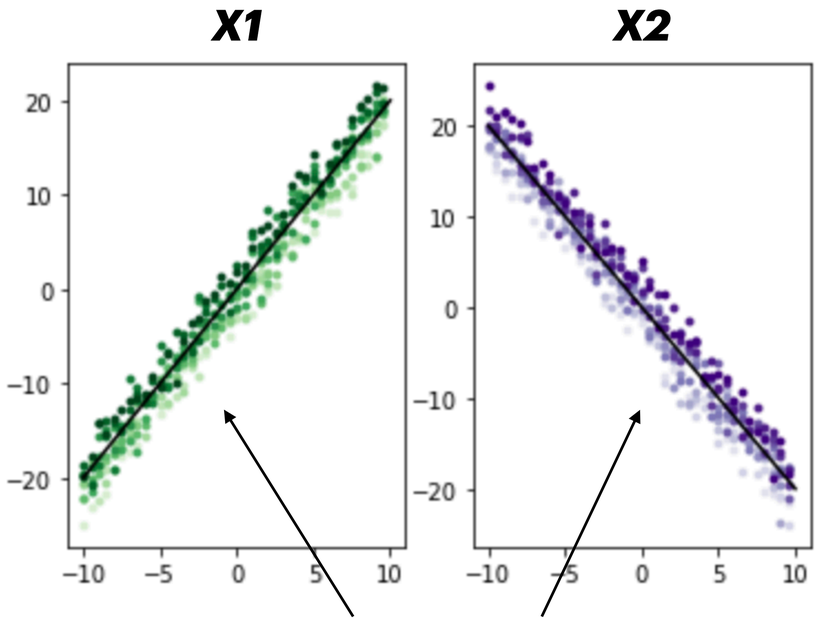

- Josh Glaser will discuss some non-linear dimensionality reduction techniques

# CCA primer

# CCA: Intuition

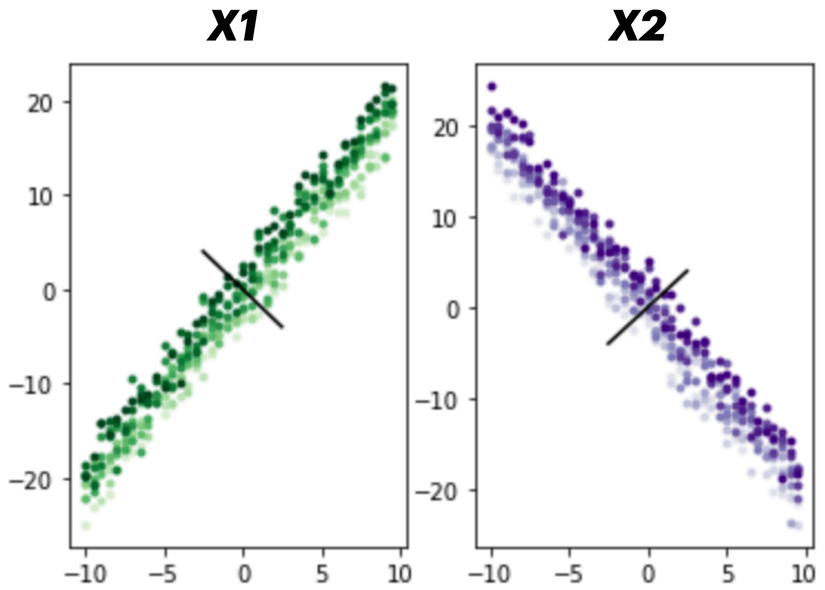
CCA is an extension of PCA for two variables that aims to find the axes along which these variables are maximally correlated

Separately estimated




color intensity is correlated

Jointly estimated




In other words, CCA jointly reduces the dimensionality of *X1* and *X2*

# CCA: Computation

The separate PCAs would be given by maximizing the covariance

$$\text{argmax}\left\{ \frac{w_1^T X_1^T X_1 w_1}{w_1^T w_1} \right\} \qquad \text{argmax}\left\{ \frac{w_2^T X_2^T X_2 w_2}{w_2^T w_2} \right\}$$

Maximizing the cross-covariance

$$\text{argmax}\left\{ \frac{w_1^T X_1^T X_2 w_2}{\sqrt{w_1^T w_1}\sqrt{w_2^T w_2}} \right\}$$

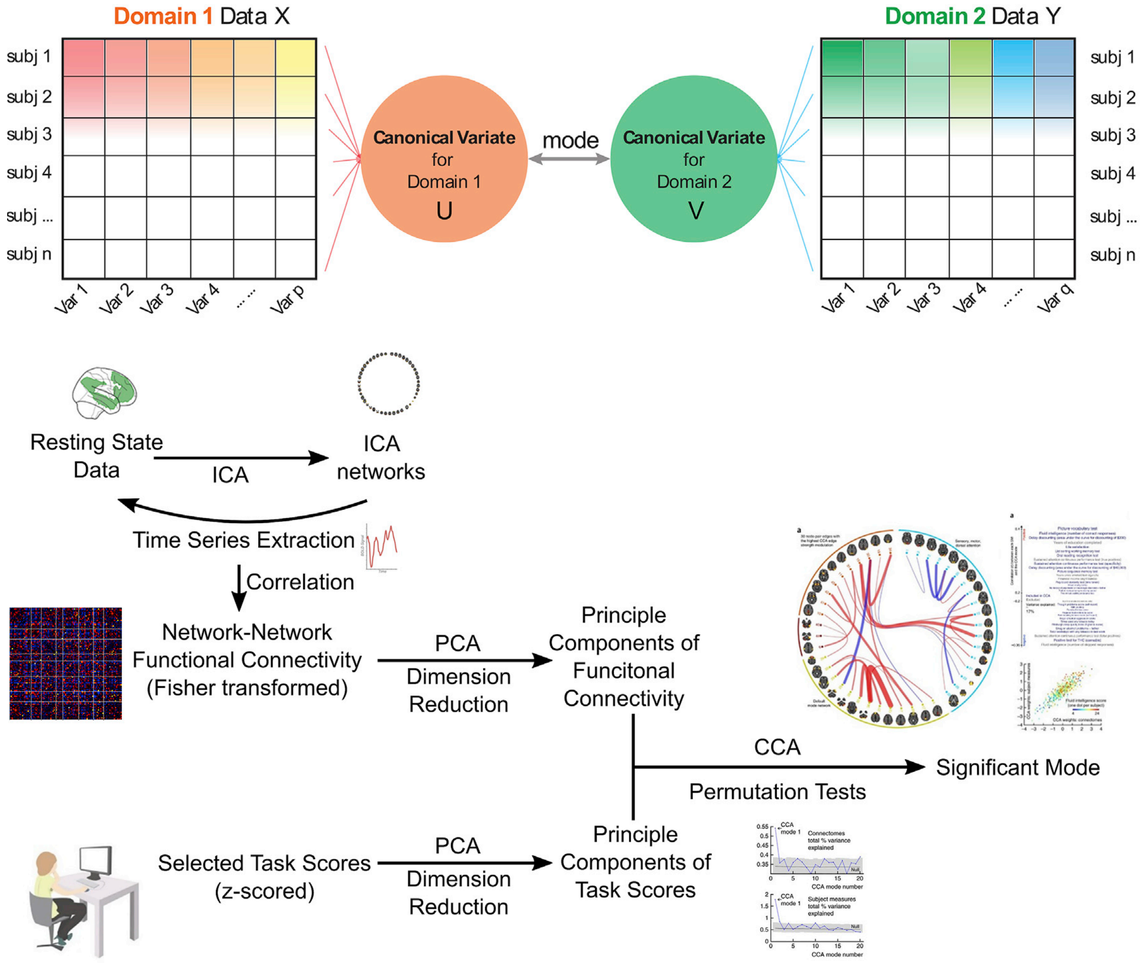
And the correlation (CCA)

$$\text{argmax}\left\{ \frac{w_1^T X_1^T X_2 w_2}{\sqrt{w_1^T X_1^T X_1 w_1} + \sqrt{w_2^T X_2^T X_2 w_2}} \right\}$$

- Similarly solved w/ eigendecomposition of correlation matrix

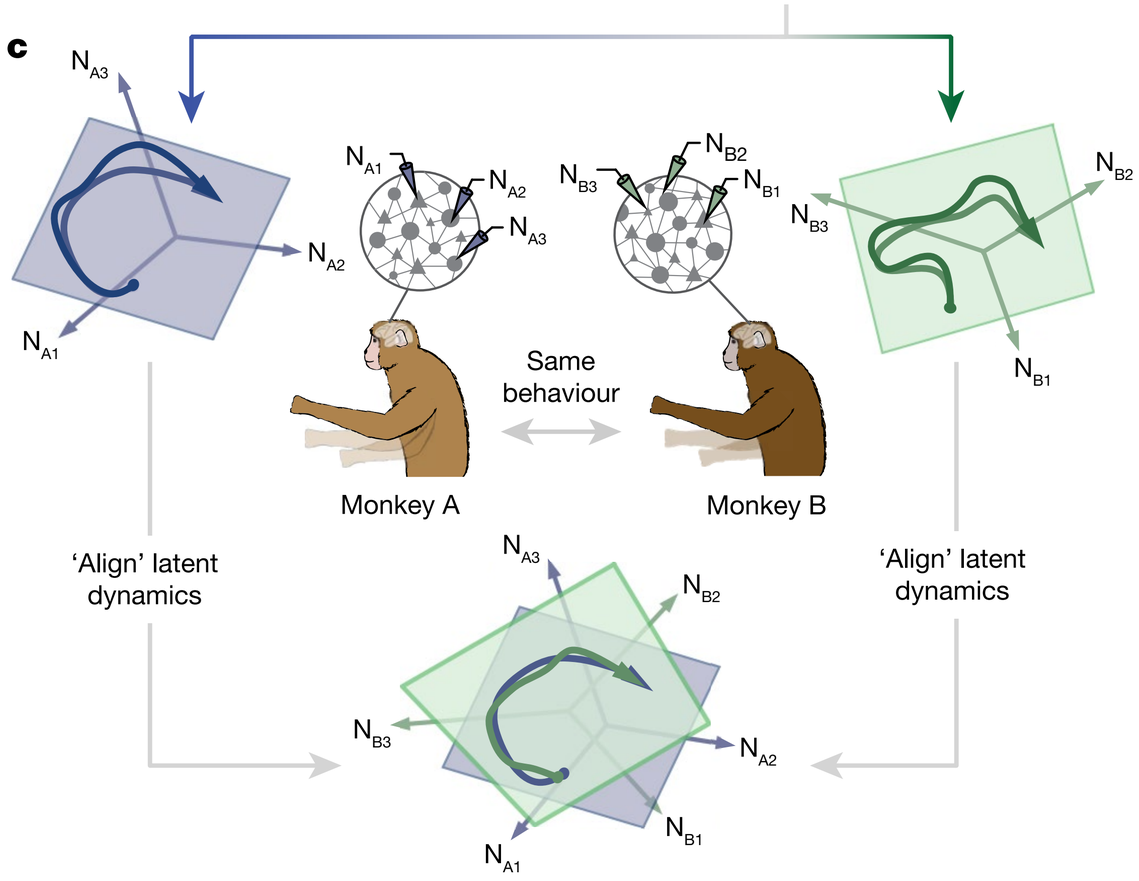
# CCA: application examples

- Identify common sources of variation in multimodal data (e.g. non-simultaneous imaging <-> behavior)




Wang et al., 2020, *Neuroimage*

- Align dynamics across subjects, species, areas




Safaie et al., 2022, *Nature*

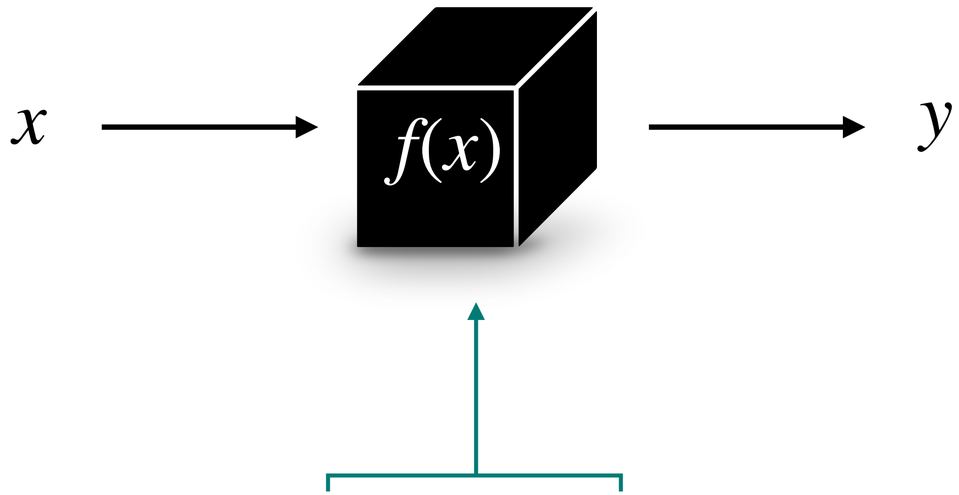
# Systems Identification

# A common approach

**Input**
(e.g., visual stimulus)

**Filter**
(e.g., receptive field)

**Response**
(e.g., spikes)



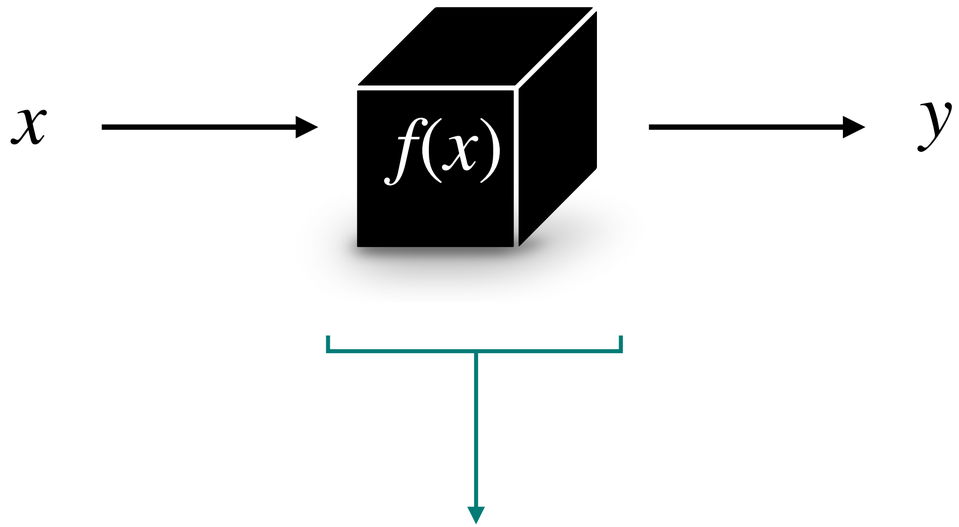

We design stimuli based on
a hypothesis about $f(x)$ (e.g., oriented bars),
and average $y$ over many presentations of the same
set of stimuli

# Goal of systems identification

**Input**
(e.g., visual stimulus)

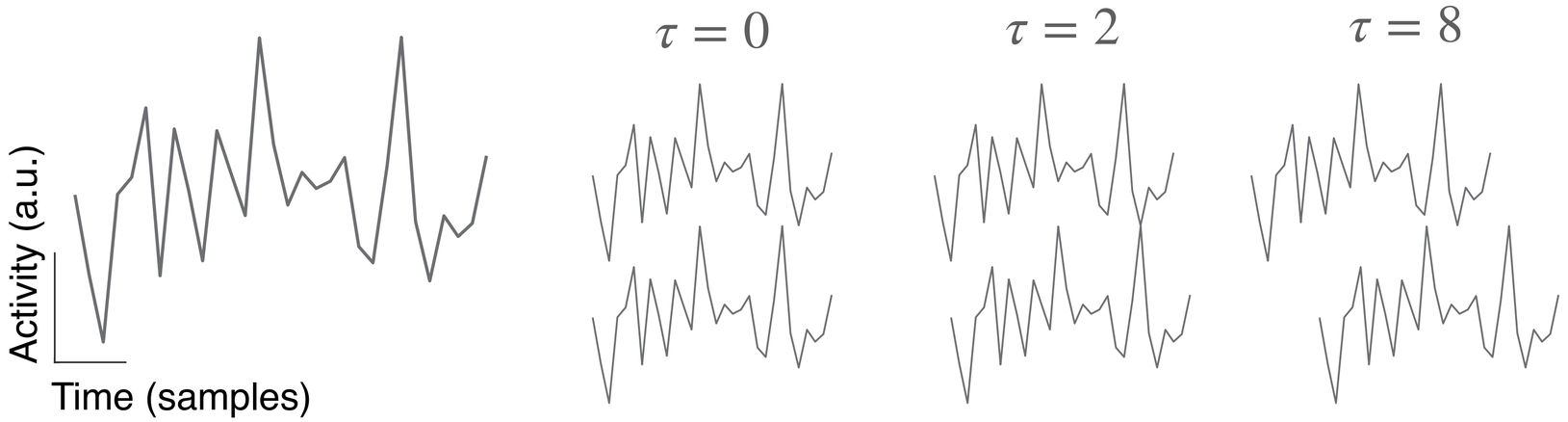
**Filter**
(e.g., receptive field)

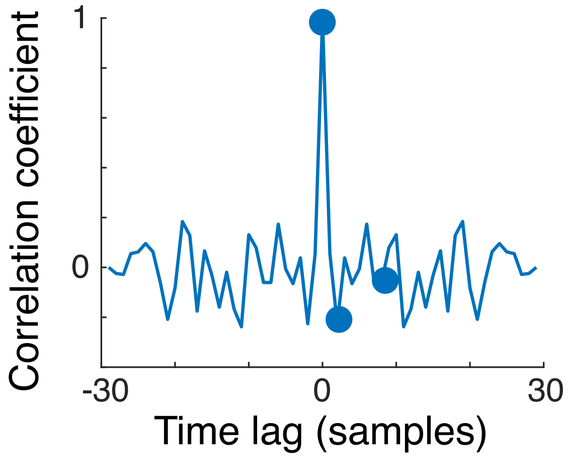
**Response**
(e.g., spikes)




We want to estimate $f(x)$ in a data-driven way, without opening the black box

# Auto- and cross-correlation




More formally,

Auto-correlation $$\phi_{xx}(\tau) = \int_0^\infty x(t)x(t-\tau)dt$$

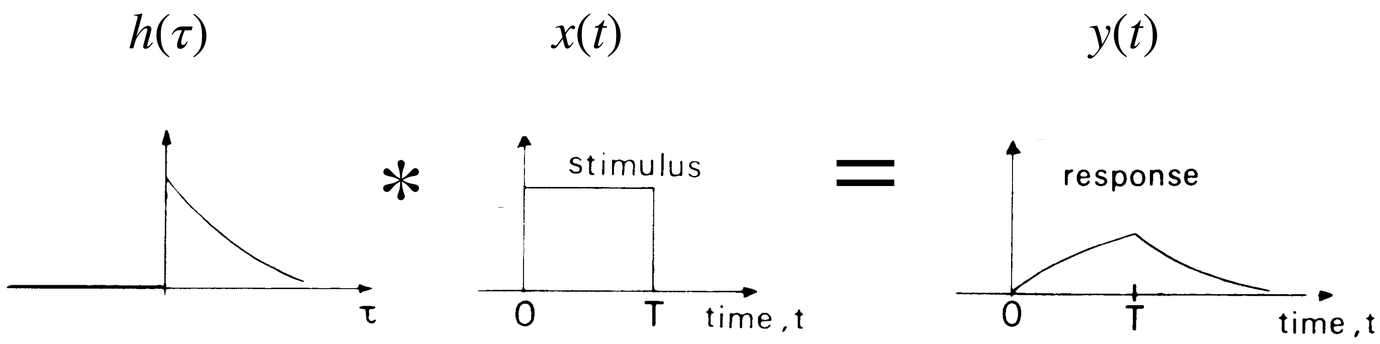
Cross-correlation $$\phi_{xy}(\tau) = \int_0^\infty y(t)x(t-\tau)dt$$

- function is symmetrical around x=0 but we usually just want the causal delays for systems identification
- Notice that the autocorrelation for gaussian (white) noise is a delta function

# Impulse response function (time domain)

The response *y* of a filter to an input *x* is given by the convolution of its kernel *h* and *x*.

$$y(t) = \int_0^\infty h(\tau)x(t-\tau)d\tau$$

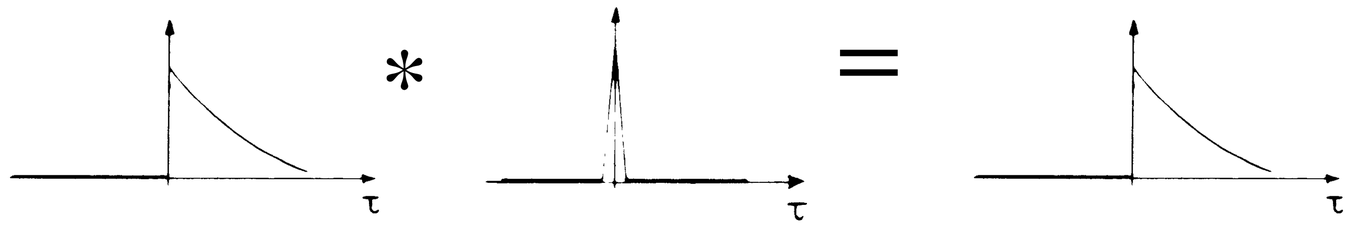



The response *y* of a filter to a delta input *x* is equal to its kernel *h*.

$$\delta(t) = \begin{cases} 1 & \text{if } t = 0 \\ 0 & \text{otherwise} \end{cases}$$

$$y(t) = \int_0^\infty h(\tau)\delta(t-\tau)d\tau$$

$$y(t) = h(\tau)$$




Marmarelis & Marmarelis, 1978

# Relationship to auto- (cross-) correlation

A useful property: if this system is linear, time invariant, and has finite memory, the same relationship is true for the auto (cross-) correlation functions

$$\phi_{xy}(\tau) = \int_0^\infty y(t)x(t-\tau)dt$$

$$= \int_0^\infty \int_0^\infty h(\lambda)x(t-\lambda)x(\tau-t)dtd\lambda$$

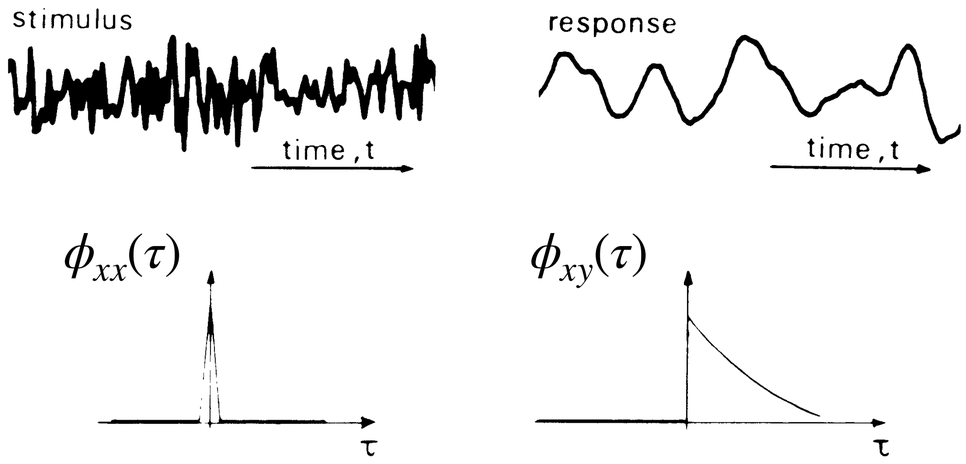
$$= \int_0^\infty h(\lambda)\phi_{xx}(\tau-\lambda)d\lambda$$

Similarly:

$$\phi_{yy}(\tau) = \int_0^\infty y(t)y(t-\tau)dt$$

$$= \int_0^\infty h(\mu)\left[\int_0^\infty h(\lambda)\phi_{xx}(\tau-\mu-\lambda)d\lambda\right]d\mu$$

If the autocorrelation is a delta function (e.g., white noise), *h()* is just a scaled version of the cross-correlation function *phi*$_{xy}$




Marmarelis & Marmarelis, 1978

# Frequency domain

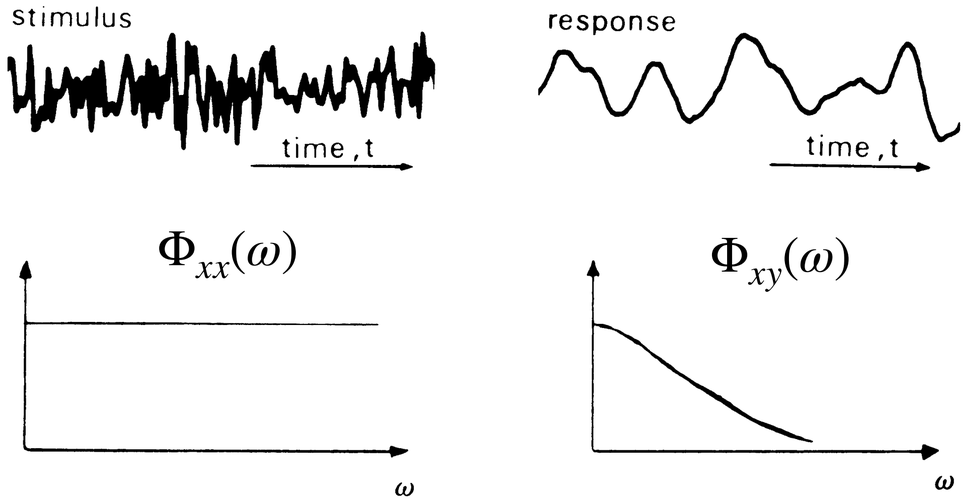
It is often more convenient to do these operations in frequency domain, where all the integrals just become simpler algebra

The power spectrum of a signal is equivalent to the Fourier Transform of its autocorrelation function

$$\phi_{xx}(\tau) = \int_0^\infty x(t)x(t-\tau)dt$$

$$\Phi_{xx}(\omega) = \int_0^\infty \phi_{xx}(\tau)e^{-i\omega\tau}d\tau$$

White noise has a flat power spectrum, so you just multiply *H(\omega)* by a constant




$$H(\omega) = c\Phi_{xy}(\omega)$$

Equivalently (from the previous slide)

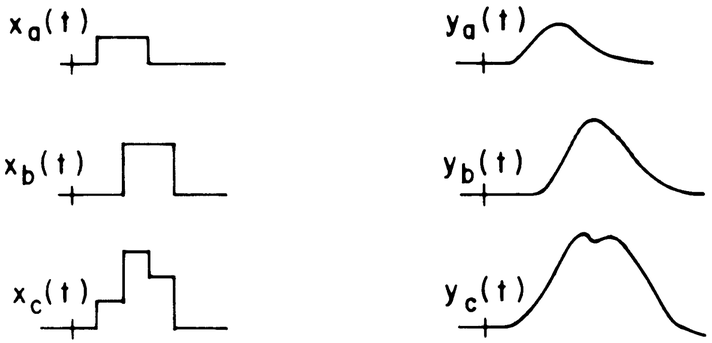
$$\Phi_{xy}(\omega) = H(\omega)\Phi_{xx}(\omega) \longrightarrow H(\omega) = \frac{\Phi_{xy}(\omega)}{\Phi_{xx}(\omega)}$$

$$\Phi_{yy}(\omega) = |H(\omega)|^2\Phi_{xx}(\omega)$$

Marmarelis & Marmarelis, 1978

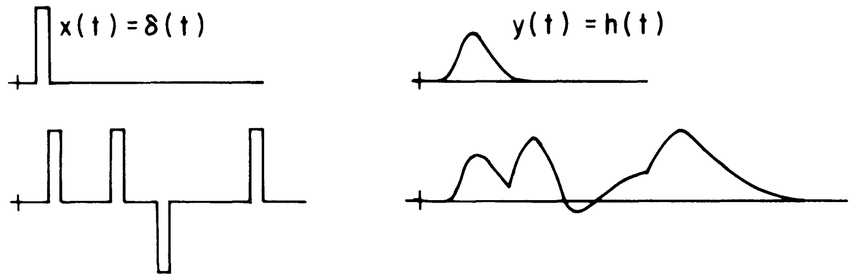
# Estimating a linear system

Principle of superposition: A linear system's response to a combined input is the a linear combination of the response to individual inputs




If your input is a train of deltas, the output is just the sum of kernels over *N* pulses, scaled by their amplitude

$$x(t) = \sum_{k=1}^{N} a_k \delta(t - t_k) \qquad y(t) = \sum_{k=1}^{N} a_k h(t - t_k)$$




Marmarelis & Marmarelis, 1978

In practice, the crosscorrelation \phi$_{xy}$ is usually estimated with averaging, indicated by <>, using the time of the input as reference

$$\phi_{xy}(\tau) = \left\langle x(t)y(t+\tau) \right\rangle \ , \ \tau > 0$$

An equivalent way of doing it is going in the other direction, i.e. picking the response time as a reference and asking what average stimulus preceded that response

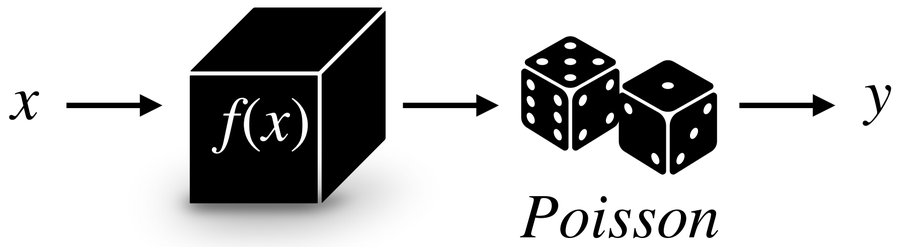
$$\mu = t + \tau$$

$$\phi_{xy}(\tau) = \left\langle x(\mu - \tau)y(\mu) \right\rangle = \left\langle x(t)y(t+\tau) \right\rangle$$

- In neuro, this is often referred to as *reverse correlation* (but note we're still estimating the same \phi$_{xy}$)

# Estimating a linear neural system

Neurons are not linear because of spiking non-linearity, but in some cases that can be thought of as a non-linearity on top of a linear operation




In this case it can be shown that the cross-correlation is still proportional to *h()*:

$$\phi_{xy}(\tau) = k\left\langle x(\mu - \tau)y(\mu)\right\rangle$$

So, when x is a delta function, h() is obtained by computing the average stimulus that preceded each spike occurring at time s

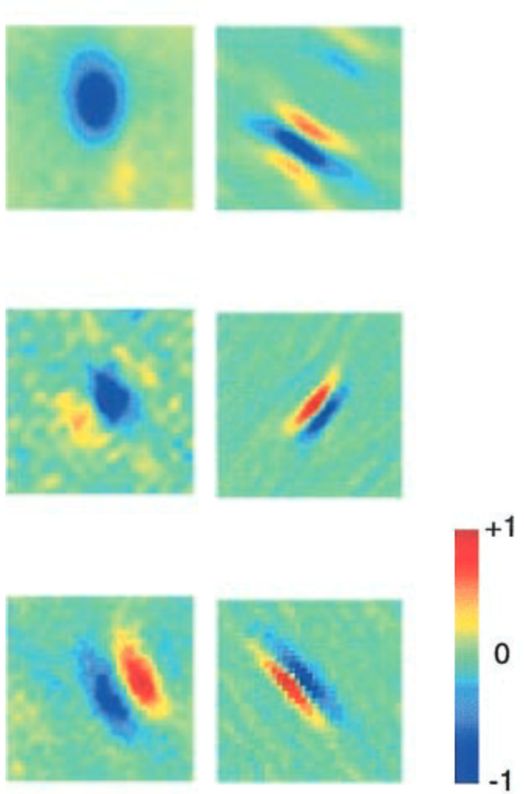
$$\phi_{xy}(\tau) \approx \frac{1}{N}\sum_{i=1}^{N} x(s_i - \tau)$$

- This is known as the *spike-triggered average* (STA)

- x does not need to be unidimensional. E.g., for a visual receptive field, what we typically want to estimate is *h*(*x*,*y*,*t*). But everything still applies

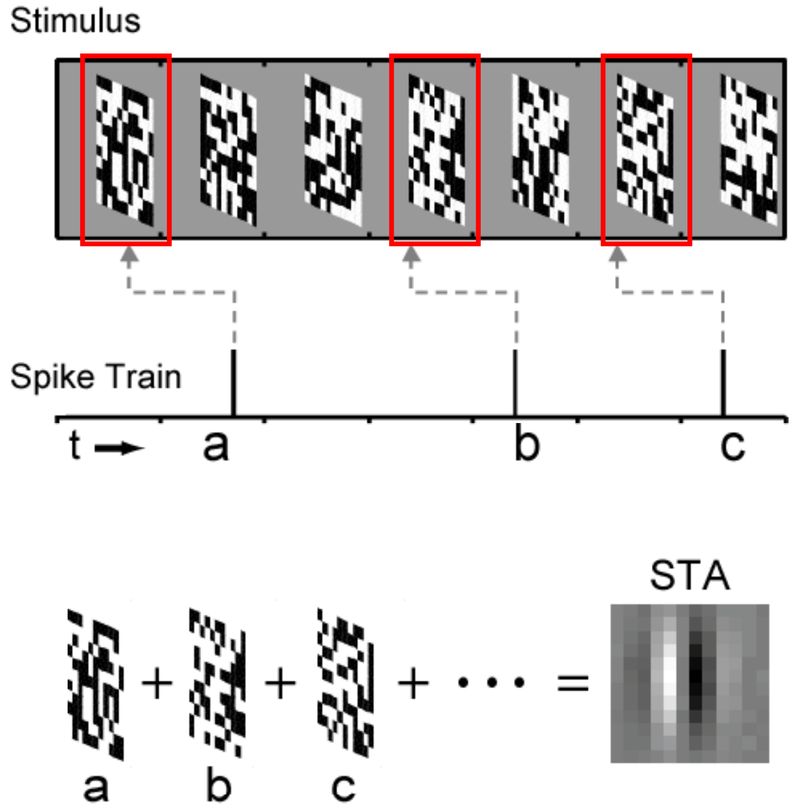
# Estimating a linear neural system: a V1 simple cell

A simple cell has separate excitatory and inhibitory subregions that combine linearly to determine its response to a stimulus inside its receptive field (Responses are phase dependent)




Ringach, 2005, *J Neurosci*

STA with white noise

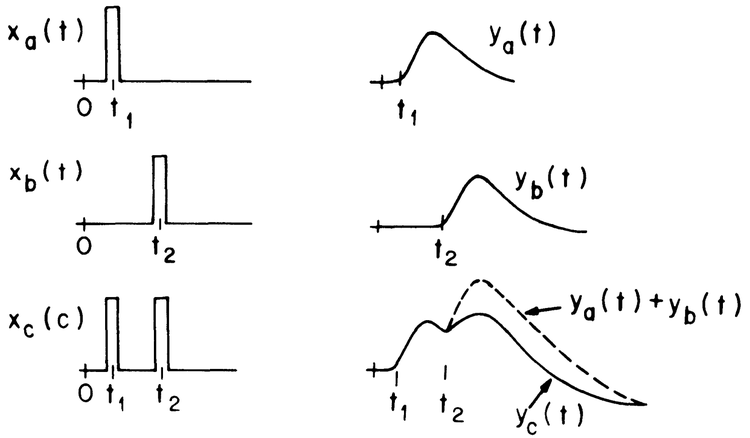



From Yang Dan

(sparse or dense binary, grayscale white noise, basis sets of sinusoidal gratings, all work)

# Estimating a non-linear system

A non-linear system does not follow the principle of superposition




Marmarelis & Marmarelis, 1978

Linear:

$$x_c(t) = x_a(t) + x_b(t)$$
$$y_c(t) = y_a(t) + y_b(t)$$

Non-linear:

$$x_c(t) = x_a(t) + x_b(t)$$
$$y_c(t) \neq y_a(t) + y_b(t)$$
$$y_c(t) = y_a(t) + y_b(t) + \theta_{t_1 t_2}(t)$$

interaction term

More generally, a time-invariant system with finite memory can be approximated by the sum of its nth-order kernels (interaction terms)

$$y(t) = \int_0^\infty h_1(\tau)x(t-\tau)d\tau + \int_0^\infty \int_0^\infty h_2(\tau_1,\tau_2)x(t-\tau_1)x(t-\tau_2)d\tau_1 d\tau_2 + \int_0^\infty \int_0^\infty \int_0^\infty h_3(\tau_1,\tau_2,\tau_3)x(t-\tau_1)x(t-\tau_2)x(t-\tau_3)d\tau_1 d\tau_2 d\tau_3 \ \ldots$$

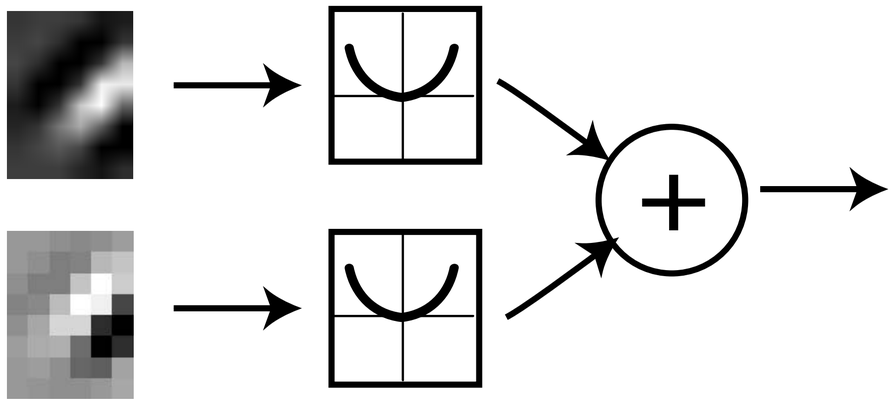
This is known as the *Volterra Series*

A more helpful formulation for white noise is given by *Wiener Theory*, which is very related, but in which the nth order terms are orthogonal to each other w.r.t. white noise. You can find an extended treatment of this in Chapter 4 of Marmarelis and Marmarelis (1978).

# Estimating a non-linear neural system: a V1 complex cell

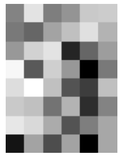
Directly estimating Volterra/Wiener kernels is often impractical and very data hungry. But *eigendecomposition* can come to the rescue as an alternative estimate of the second-order term

E.g. take a complex cell, whose responses to visual stimuli are phase invariant and therefore non-linear. A sum of quadratic non-linearities is often used to model these cells



Schwartz et al., 2006, *J. Vis.*

Now we do a white-noise experiment as before and compute the STA, which returns noise



The second mode of the distribution can be estimated from the covariance matrix *C* of the spike-triggered stimuli:

$$\hat{C}(t) = \frac{1}{N}\sum_{n=1}^{N}(\vec{s}_n - \overrightarrow{STA})^T(\vec{s}_n - \overrightarrow{STA})$$

Number of spikes

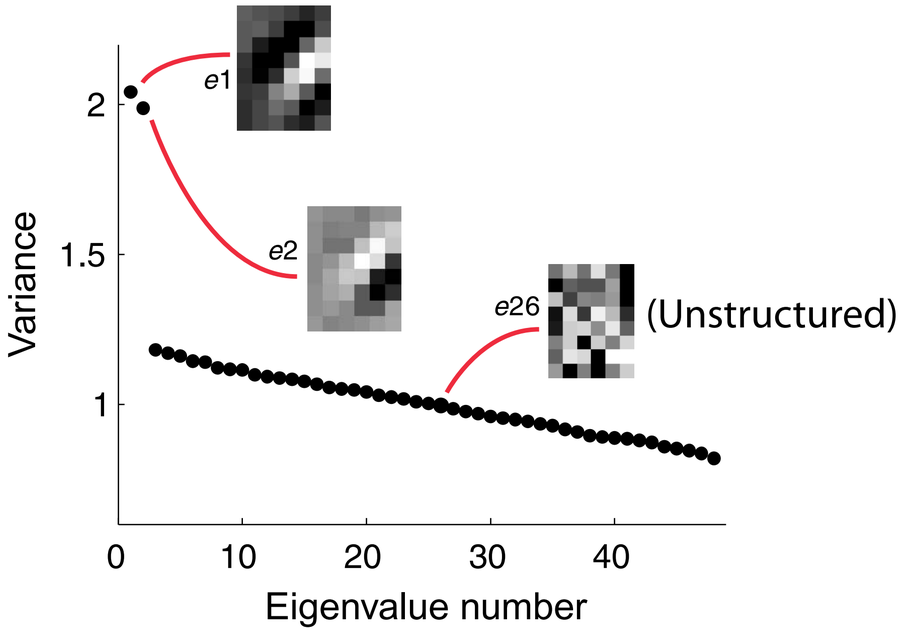
Vector of stimulus values triggered by spike n

Subtract the average (STA) as defined previously

- In practice we bin spike data and multiply each term by spike counts

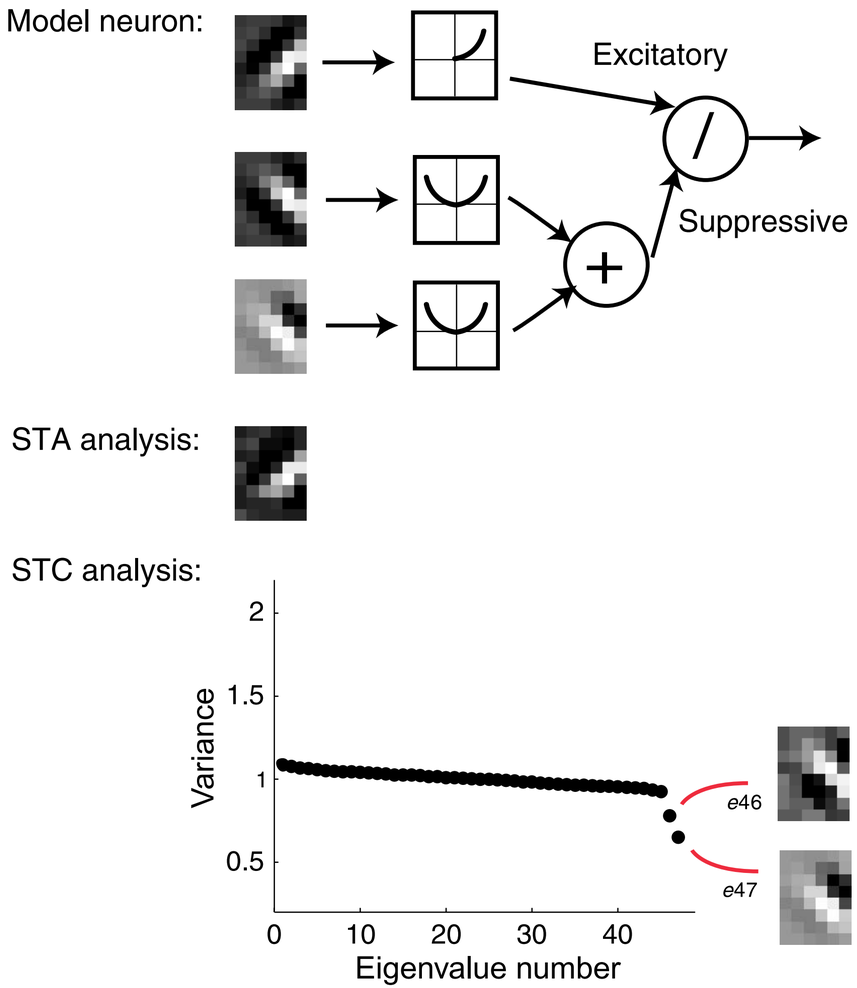
# Estimating a non-linear neural system: a V1 complex cell

Finally, we can do eigendecomposition of *C* to obtain its significant orthogonal components as estimates of receptive field subunits




Eigenvector significance is usually estimated with shuffling / bootstrapping procedures

And it can get more complicated, too!

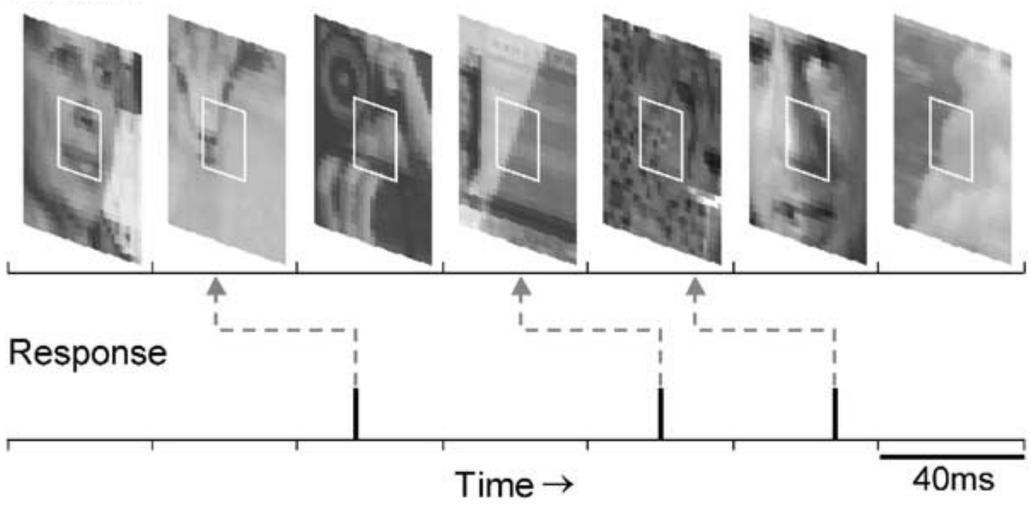



Schwartz et al., 2006, *J. Vis.*

# Beyond white noise

White noise has many convenient properties, but systems identification can also generalize beyond the Gaussian noise distribution

For example, V1 response properties differ when probed with natural stimuli, which have different statistics than white noise or gratings (pink noise, $1/f^2$, because nearby pixels are correlated)




Touryan et al., 2005, *Neuron*

But remember that the kernel depends on both the stimulus-response cross-correlation and the stimulus auto-correlation. So we can still use non-white stimuli of we correct for its statistical (spectral) properties:

$$H(\omega) = \frac{\Phi_{xy}(\omega)}{\Phi_{xx}(\omega)}$$

One of our journal club papers (Touryan et al., 2005) will go over this in detail so I won't give any more spoilers

Also, for an interesting statistical (Bayesian) interpretation of systems identification, refer to the Wu et al. (2006) paper in the suggested reading list

# Summary

- Eigendecomposition decomposes a square matrices into an orthogonal basis set
- PCA is the eigendecomposition of the covariance matrix, rotates the data around the mean to obtain orthogonal dimensions
- PCA (& eigenvalue decomposition) has countless uses in neuroscience: linear dimensionality reduction, systems identification, regression, visualization
- CCA is a PCA extension that finds the most correlated axes between two variables
- CCA is very useful for e.g., multi-area, multi-day, multi-animal comparisons
- Systems identification estimates the operations a black box performs on inputs without opening the box
- White noise has many convenient properties that allow systems identification directly from auto- and cross-correlation functions
- STA and STC are common implementations of systems identification in sensory receptive field mapping

# Suggested reading


- Hastie, Tibshirani & Friedman (2017). The Elements of Statistical Learning, 2nd edition. Chapter 14 (PCA section).
- Wang et al. (2020). Finding the needle in a high-dimensional haystack: Canonical correlation analysis for neuroscientists. *Neuroimage* 216: 166745.
- Marmarelis & Marmarelis (1978). Analysis of Physiological Systems: the white-noise approach. Springer, New York NY https://link.springer.com/book/10.1007/978-1-4613-3970-0, esp. chapters 2-4
- Ringach & Shapley (2004). Reverse correlation in neurophysiology. *Cog. Sci.* 28:147-166
- Schwartz et al. (2006). Spike-triggered neural characterization. *J. Vis.* 6:484-507
- Wu et al. (2006). Complete functional characterization of sensory neurons by systems identification. *Annu. Rev. Neurosci.* 29:477-505
- Journal club papers on canvas

# The end