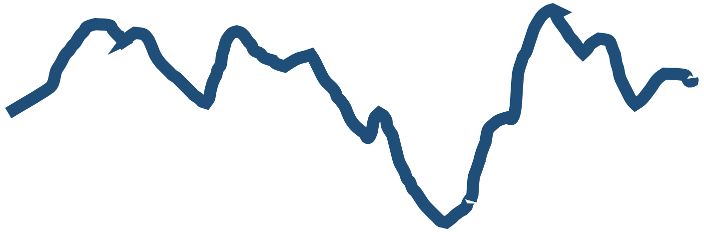
Dimensionality Reduction and Latent Variable Models

Bi23: Methods in Neural Data Analysis

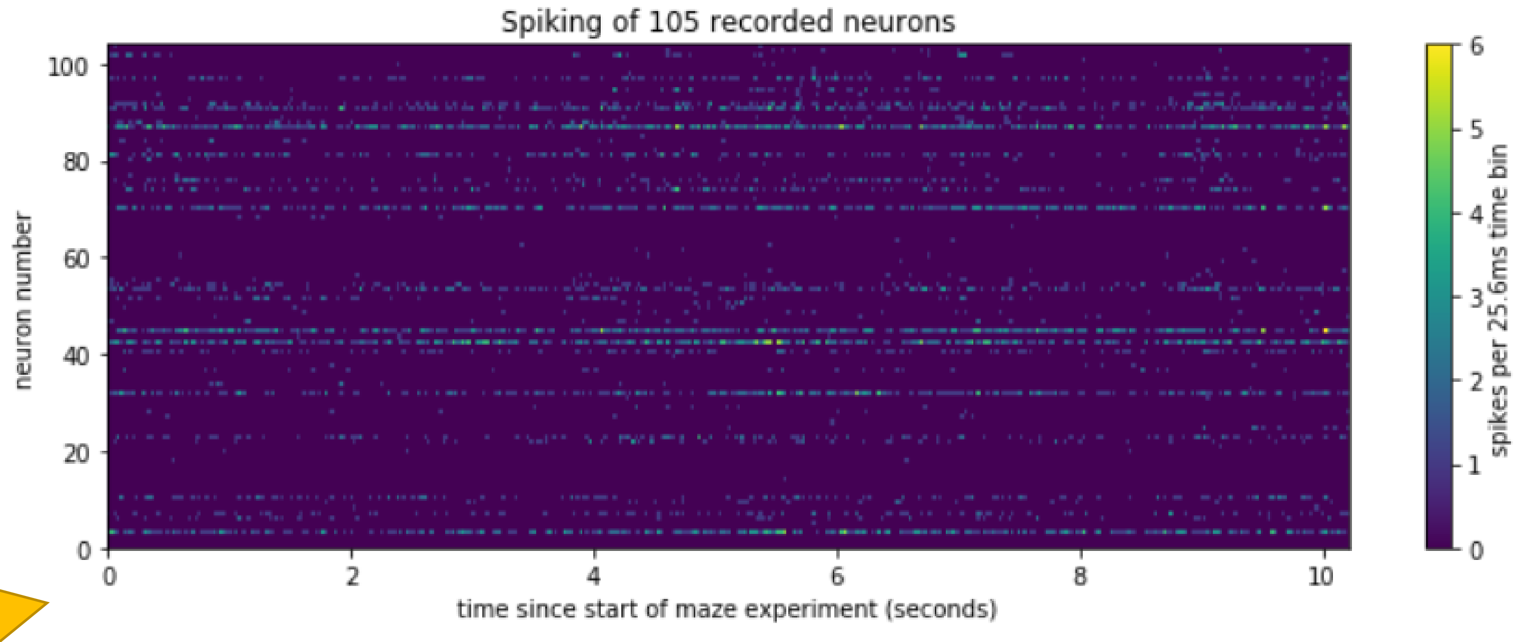
3/1/2019

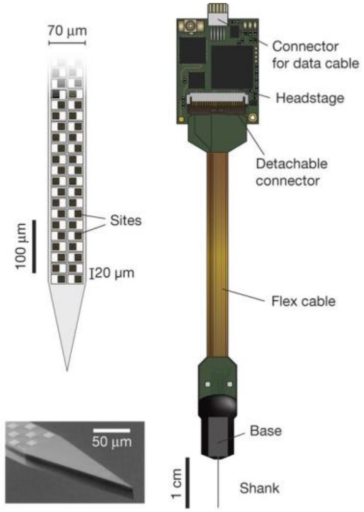
From single neurons to populations

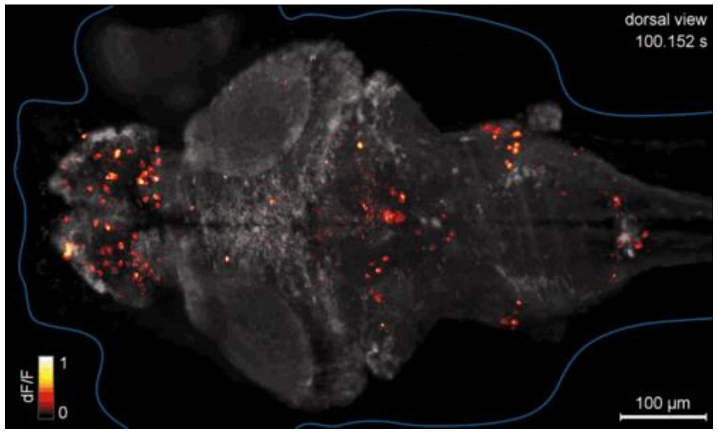
Stimulus feature **(x)**



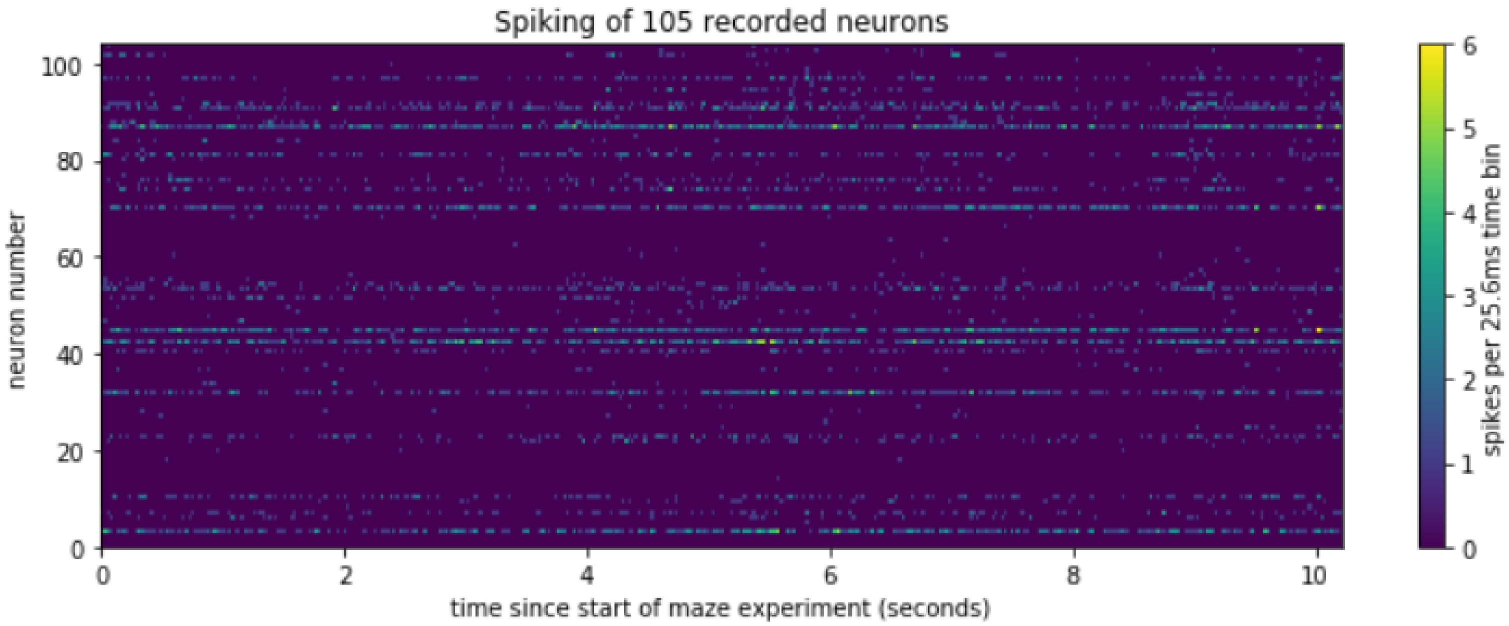
time →



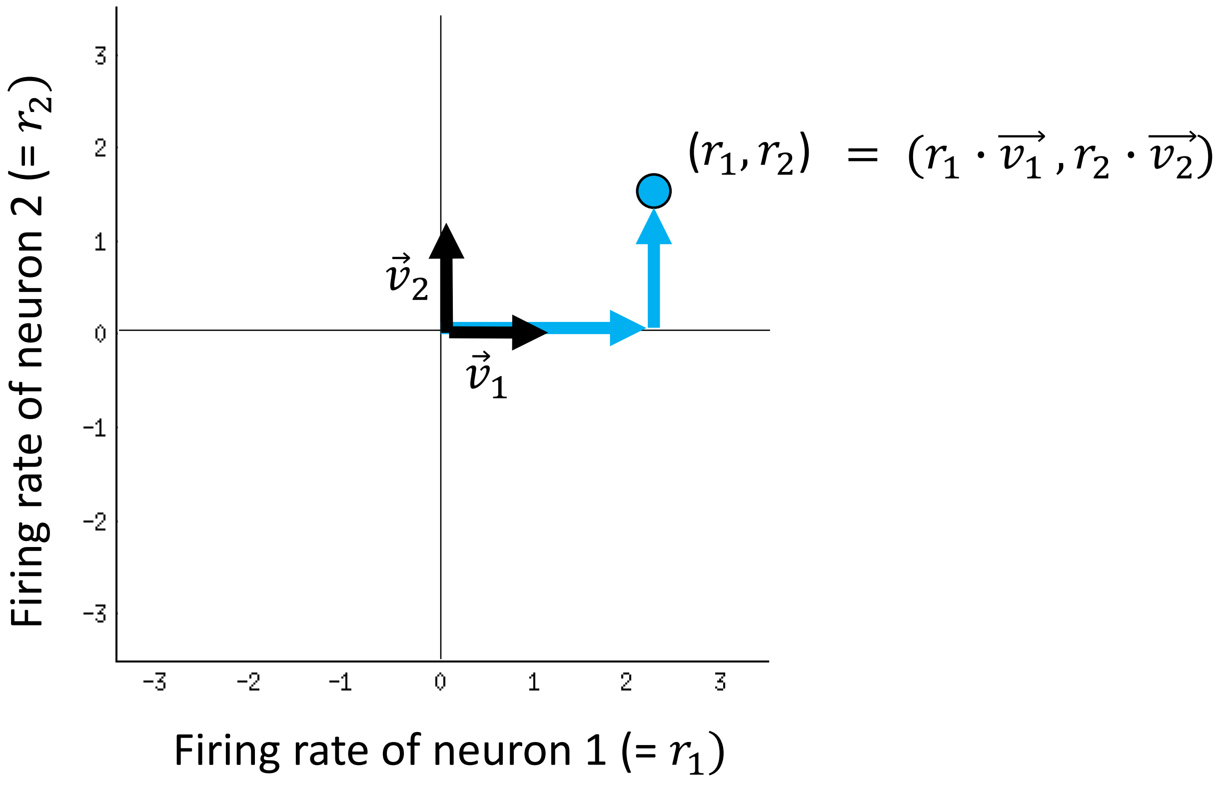




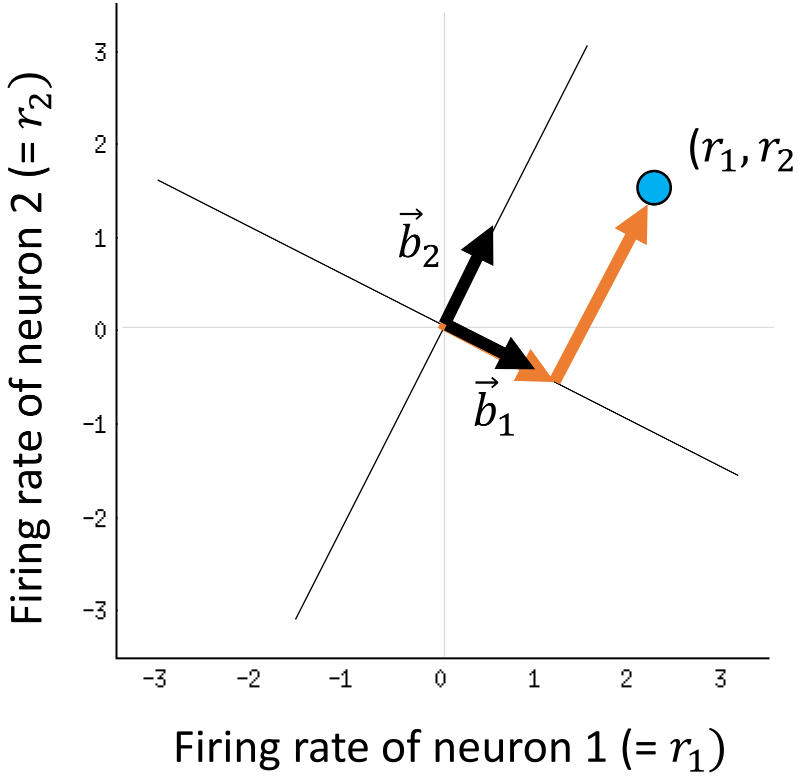
How do we summarize what this population of neurons is doing?



Linear algebra review: change of basis (ie, how are we summarizing what 2 neurons are doing?)

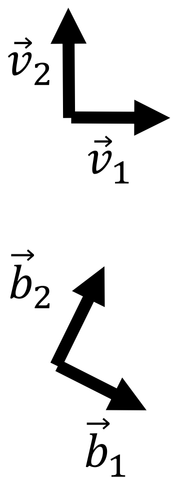


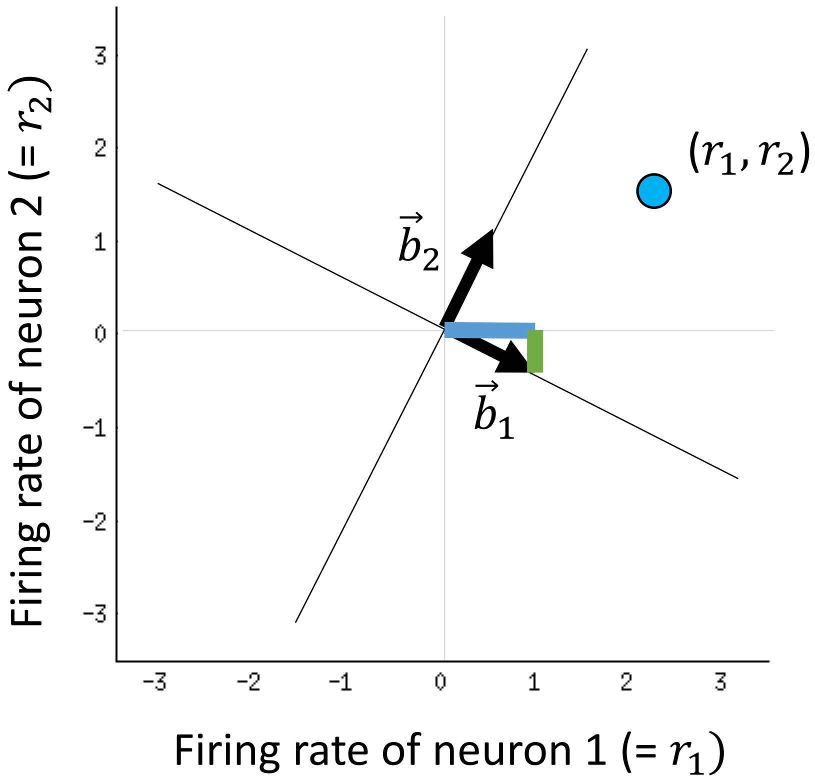
Linear algebra review: change of basis (ie, how are we summarizing what 2 neurons are doing?)



$(r_1, r_2) = (s_1 \cdot \vec{b}_1, s_2 \cdot \vec{b}_2)$

Linear algebra review: change of basis (ie, how are we summarizing what 2 neurons are doing?)

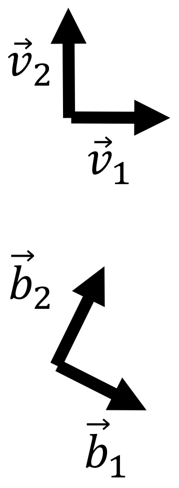


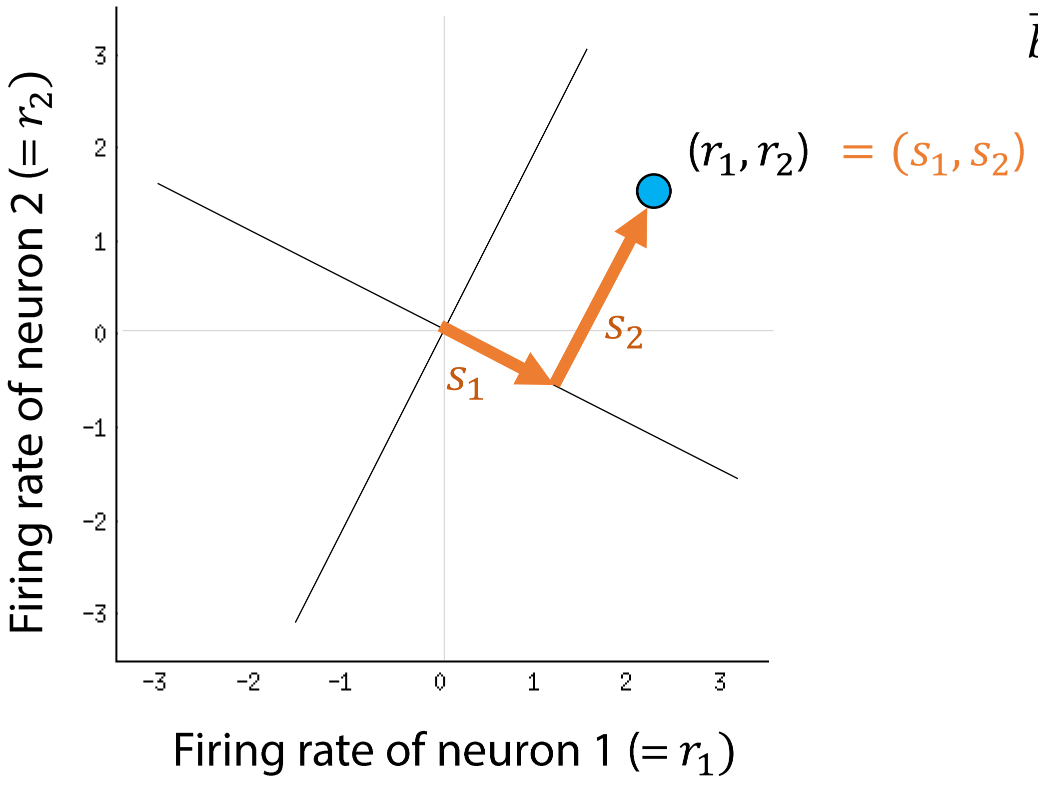


Change of basis from **v** to **b**:

$$\vec{b}_1 = \mathbf{w_1} \cdot \vec{v}_1 + \mathbf{w_2} \cdot \vec{v}_2$$
$$\vec{b}_2 = -w_2 \cdot \vec{v}_1 + w_1 \cdot \vec{v}_2$$

Linear algebra review: change of basis (ie, how are we summarizing what 2 neurons are doing?)





Change of basis from **v** to **b**:

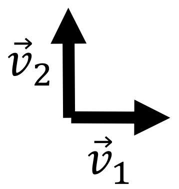
$$\vec{b}_1 = \mathbf{w_1} \cdot \vec{v}_1 + \mathbf{w_2} \cdot \vec{v}_2$$
$$\vec{b}_2 = -w_2 \cdot \vec{v}_1 + w_1 \cdot \vec{v}_2$$

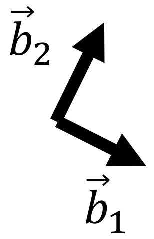
$$s_1 = w_1\, r_1 + w_2 r_2$$
$$= \mathbf{projection}((r_1, r_2), \vec{b}_1)$$

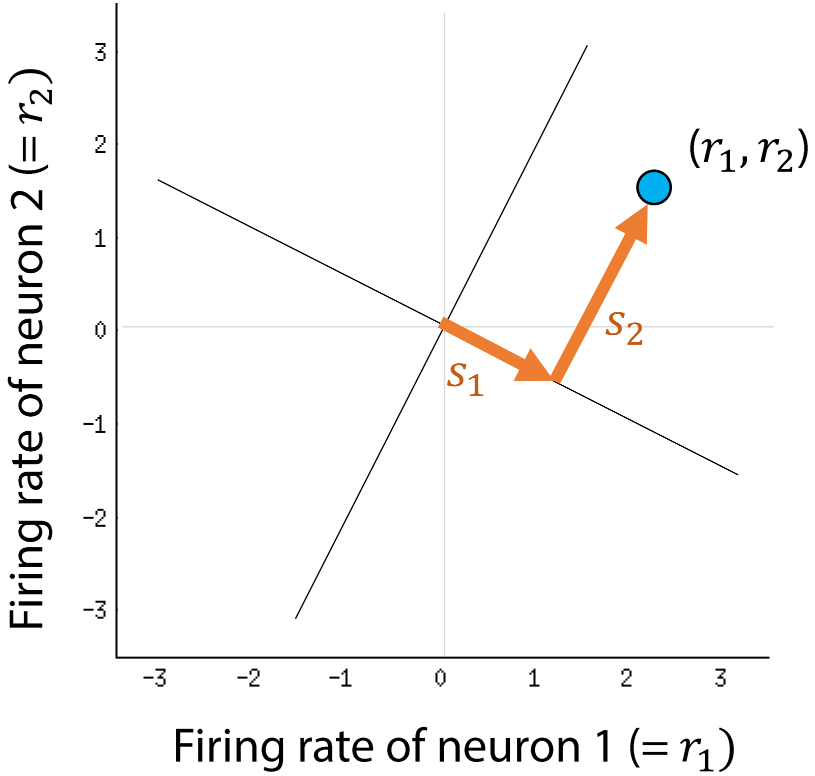
$$s_2 = -w_2 r_1 + w_1 r_2$$
$$= \mathbf{projection}((r_1, r_2), \vec{b}_2)$$

There are many different ways (bases) to communicate the same information!

Linear algebra review: change of basis (ie, how are we summarizing what 2 neurons are doing?)







Change of basis from **v** to **b**:

$$\vec{b}_1 = \quad w_1 \cdot \vec{v}_1 + w_2 \cdot \vec{v}_2$$
$$\vec{b}_2 = -w_2 \cdot \vec{v}_1 + w_1 \cdot \vec{v}_2$$

$$\begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} w_1 & w_2 \\ -w_2 & w_1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$$

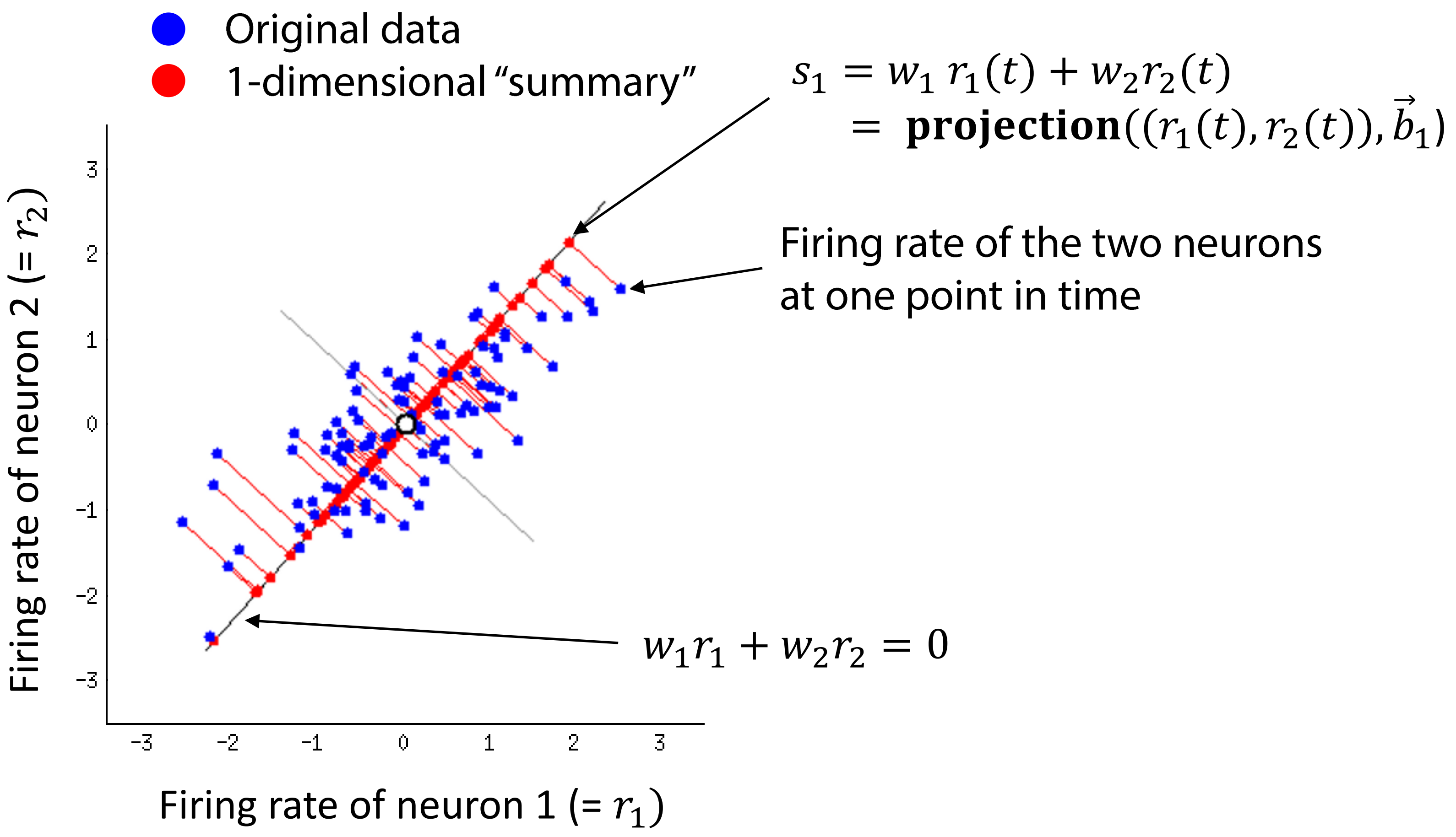
Change of basis from **b** back to **v**:

$$\vec{v}_1 = -w_1 \cdot \vec{b}_1 + w_2 \cdot \vec{b}_2$$
$$\vec{b}_2 = -w_2 \cdot \vec{b}_1 - w_1 \cdot \vec{b}_2$$

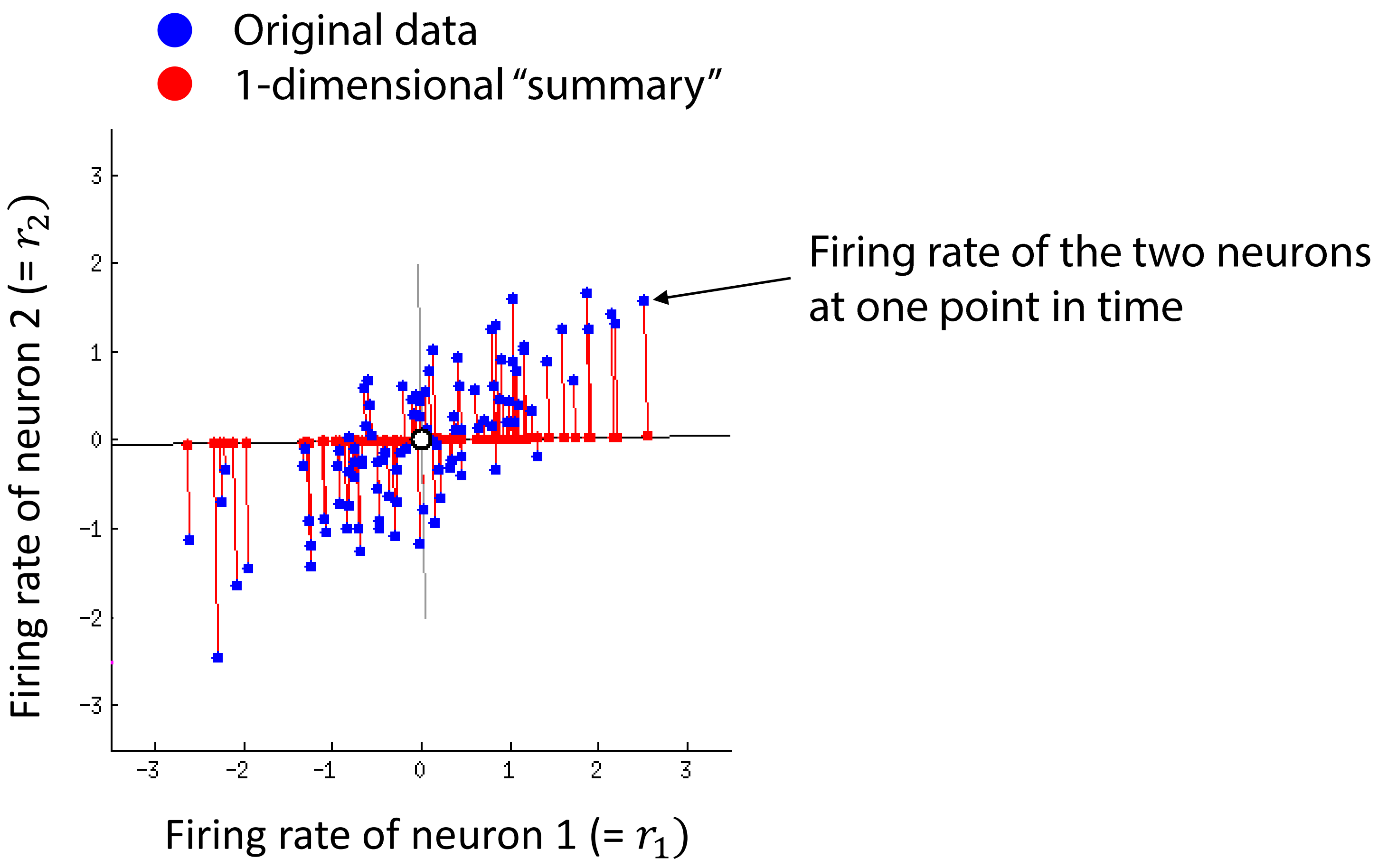
$$\begin{pmatrix} w_1 & w_2 \\ -w_2 & w_1 \end{pmatrix}^{-1} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$$

back to our problem of summarizing data

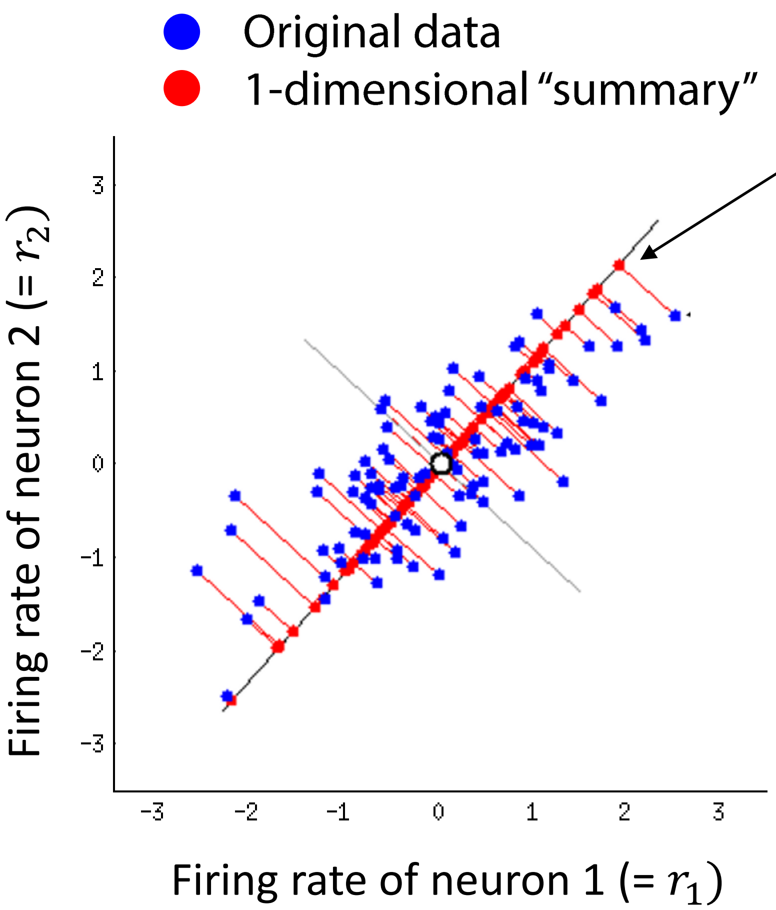
If we pick the right basis, we can simply from 2d to 1d without losing much information



Different bases produce different 1d summaries (representations) of the data



How much information do we lose by projecting down our data?



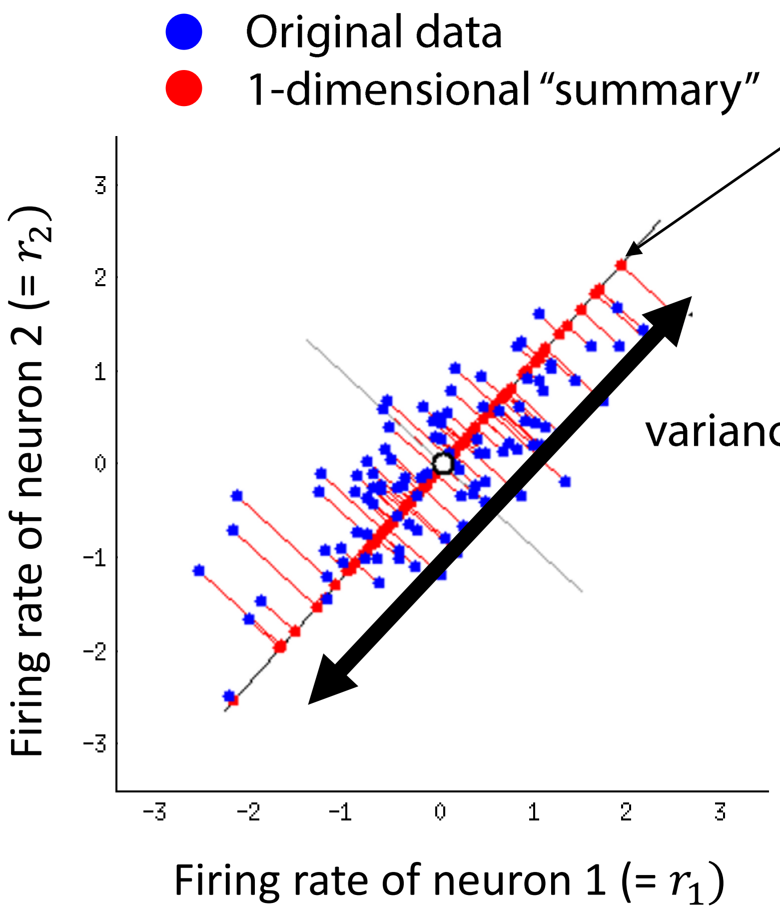
$s_1(t) = w_1\, r_1(t) + w_2 r_2(t)$

$$\begin{pmatrix} w_1 & w_2 \\ -w_2 & w_1 \end{pmatrix}^{-1} \begin{pmatrix} s_1(t) \\ s_2(t) \end{pmatrix} = \begin{pmatrix} r_1(t) \\ r_2(t) \end{pmatrix}$$

$$\begin{pmatrix} w_1 & w_2 \\ -w_2 & w_1 \end{pmatrix}^{-1} \begin{pmatrix} s_1(t) \\ 0 \end{pmatrix} = \begin{pmatrix} \hat{r}_1(t) \\ \hat{r}_2(t) \end{pmatrix}$$

$$\boldsymbol{MSE} = \sqrt{(\boldsymbol{r_1(t)} - \boldsymbol{\hat{r}_1(t)})^2 + (\boldsymbol{r_2(t)} - \boldsymbol{\hat{r}_2(t)})^2}$$

Which basis to pick? The one that maximizes *variance* of projected data

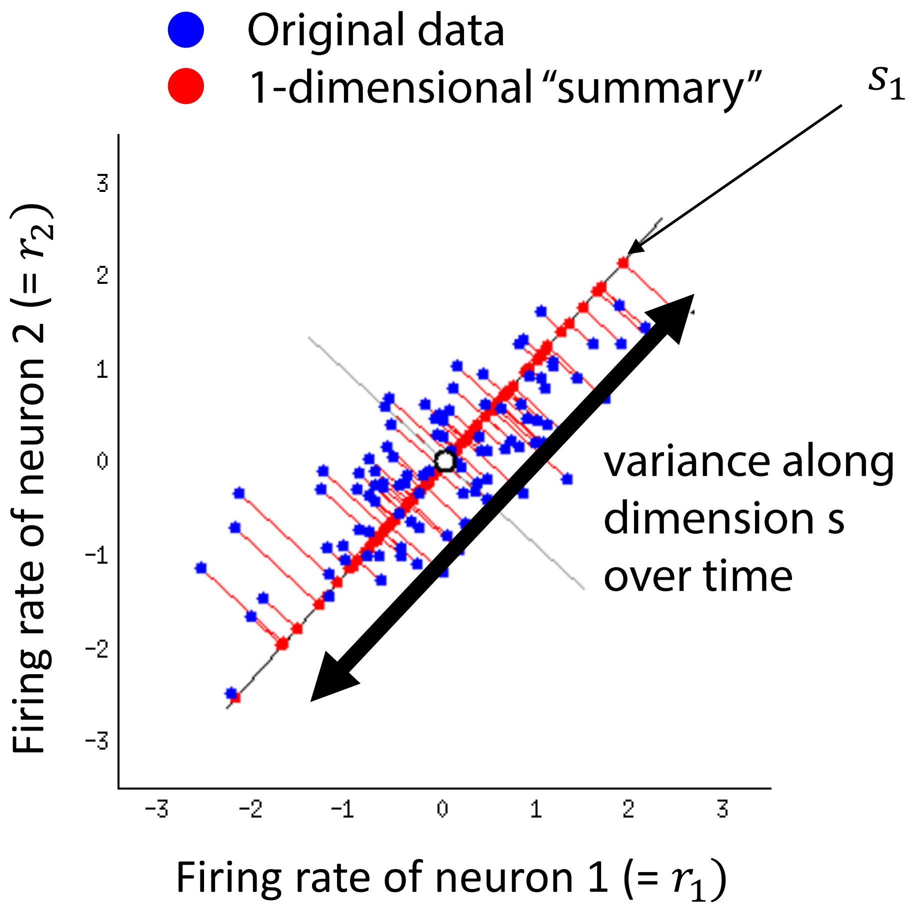


$s_1(t) = w_1\, r_1(\mathrm{t}) + w_2\, r_2(t)$

variance $= (s_1(0) - \bar{s}_1)^2 + (s_1(1) - \bar{s}_1)^2 + \ \ldots$

$= (\boldsymbol{s_1} - \bar{\boldsymbol{s}}_{\boldsymbol{1}})^T (\boldsymbol{s_1} - \bar{\boldsymbol{s}}_{\boldsymbol{1}})$

Which basis to pick? The one that maximizes *variance* of projected data



$$s_1(t) = w_1\, r_1(t) + w_2 r_2(t) = \boldsymbol{w} \cdot \boldsymbol{r}(\boldsymbol{t})$$

variance = $(s_1 - \bar{s}_1)(s_1 - \bar{s}_1)^T$

$$= \boldsymbol{w}(\boldsymbol{r} - \bar{\boldsymbol{r}})(\boldsymbol{r} - \bar{\boldsymbol{r}})^T \boldsymbol{w}^T$$

$$= \boldsymbol{w}\, \Sigma\, \boldsymbol{w}^T$$

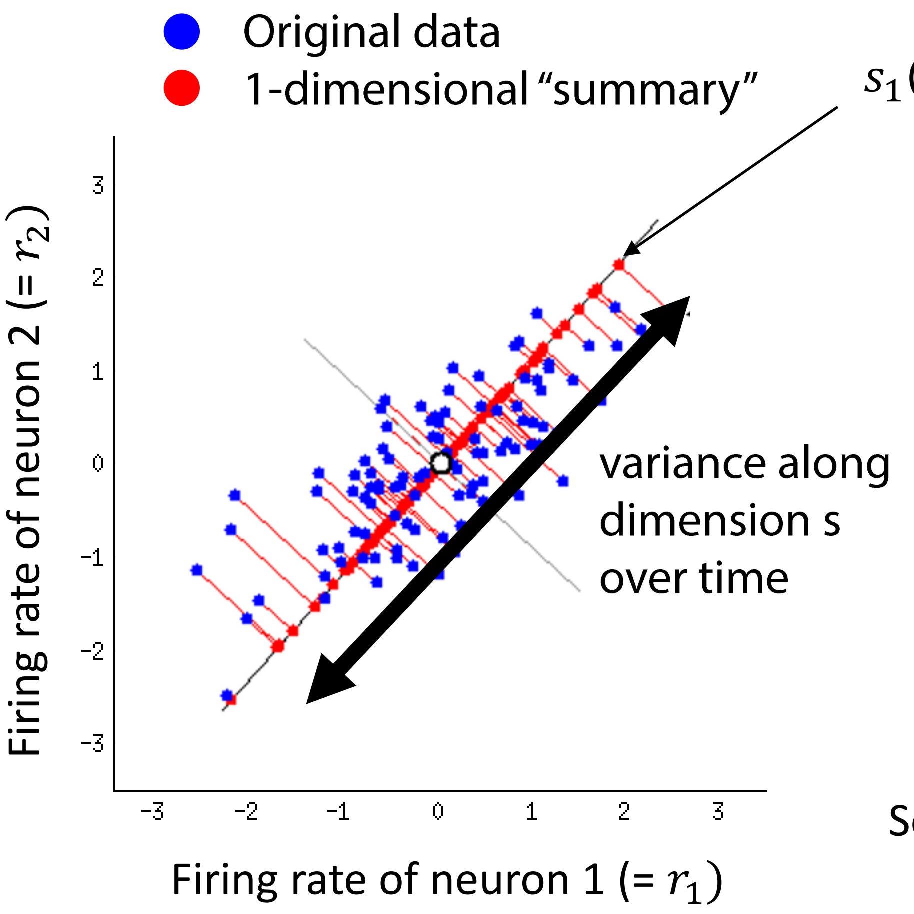
term we want to maximize

$$(AB)^T = B^T A^T$$

$$\Sigma = (\boldsymbol{r} - \bar{\boldsymbol{r}})(\boldsymbol{r} - \bar{\boldsymbol{r}})^T$$

Definition of the covariance matrix

Which basis to pick? The one that maximizes *variance* of projected data



$$s_1(t) = w_1\, r_1(t) + w_2 r_2(t) = \boldsymbol{w} \cdot \boldsymbol{r}(\boldsymbol{t})$$

$$\text{variance} = (s_1 - \bar{s}_1)(s_1 - \bar{s}_1)^T$$

$$= \boldsymbol{w}(\boldsymbol{r} - \bar{\boldsymbol{r}})(\boldsymbol{r} - \bar{\boldsymbol{r}})^T \boldsymbol{w}^T$$

$$= \boldsymbol{w}\, \Sigma\, \boldsymbol{w}^T$$

$(AB)^T = B^T A^T$

$\Sigma = (\boldsymbol{r} - \bar{\boldsymbol{r}})(\boldsymbol{r} - \bar{\boldsymbol{r}})^T$

Definition of the covariance matrix

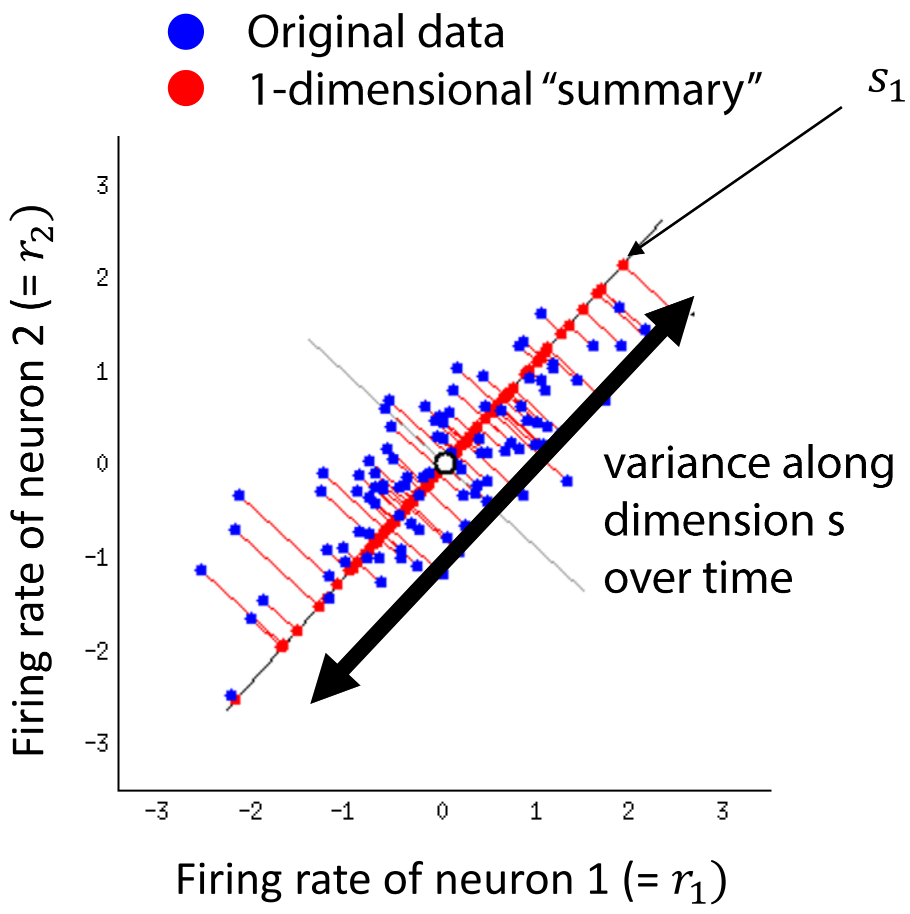
We can make $\boldsymbol{w}\, \Sigma\, \boldsymbol{w}^T$ arbitrarily large just by inflating w, so let's instead maximize it *subject to the constraint* $\boldsymbol{w}\, \boldsymbol{w}^T = 1$

So now we want to find the **w** that maximizes:

$$\boldsymbol{w}\, \Sigma\, \boldsymbol{w}^T - \boldsymbol{\lambda}(\boldsymbol{w}\boldsymbol{w}^T - 1)$$

Use a *Lagrange multiplier (λ)* to introduce an additional cost/constraint

Which basis to pick? The one that maximizes *variance* of projected data



$$s_1(t) = w_1\, r_1(t) + w_2 r_2(t) = \boldsymbol{w} \cdot \boldsymbol{r}(\boldsymbol{t})$$

$$\Sigma \;=(\boldsymbol{r}-\bar{\boldsymbol{r}})(\boldsymbol{r}-\bar{\boldsymbol{r}})^{T}$$

covariance matrix

Find the **w** that maximizes:

Lagrange multiplier

$$\boldsymbol{w}\,\Sigma\,\boldsymbol{w}^T - \boldsymbol{\lambda}(\boldsymbol{w}\boldsymbol{w}^T - 1)$$

Max occurs where derivative with respect to **w** = 0

$$\frac{\boldsymbol{d}}{\boldsymbol{dw}}\left(\boldsymbol{w}\,\Sigma\,\boldsymbol{w}^T - \boldsymbol{\lambda}(\boldsymbol{w}\boldsymbol{w}^T - 1)\right) = 0$$

$$(\,\Sigma\,\boldsymbol{w}^T - \boldsymbol{\lambda}\,\boldsymbol{w}^T) = 0$$

$$\Sigma\,\boldsymbol{w}^T = \boldsymbol{\lambda}\,\boldsymbol{w}^T$$

This should be recognizable as an eigenvector equation, where $\boldsymbol{w}$ is an eigenvector of Σ and $\boldsymbol{\lambda}$ is its corresponding eigenvalue.

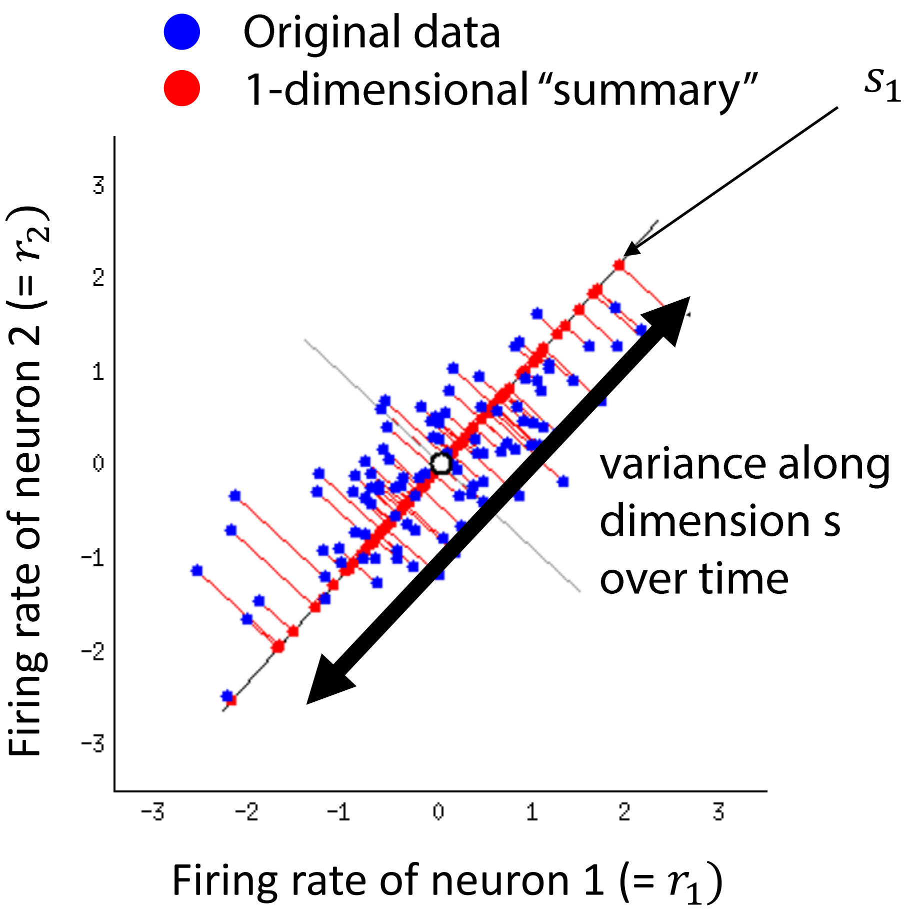
Which basis to pick? The one that maximizes *variance* of projected data

$\Sigma \ = (\boldsymbol{r} - \bar{\boldsymbol{r}})(\boldsymbol{r} - \bar{\boldsymbol{r}})^{\,T}$

covariance matrix

$\boldsymbol{\lambda}$ = Lagrange multiplier

$s_1(t) = w_1\, r_1(t) + w_2 r_2(t) = \boldsymbol{w} \ \cdot \ \boldsymbol{r}(\boldsymbol{t})$



Our solution for w is an eigenvector of the **covariance matrix** Σ ,

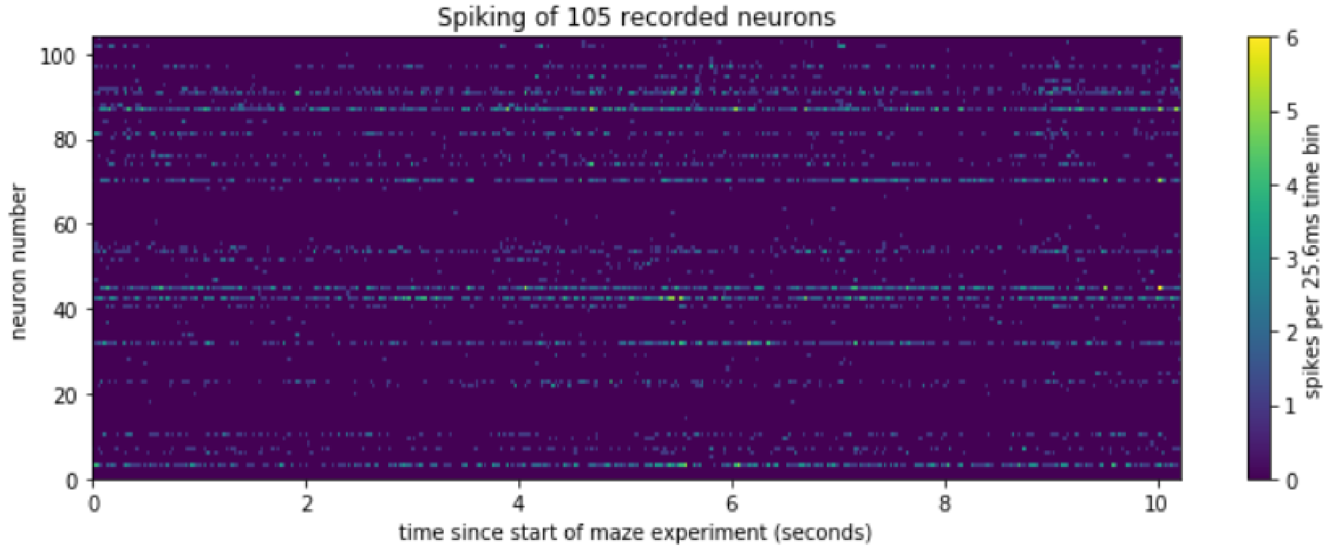
$$\Sigma\, \boldsymbol{w}^T \ = \boldsymbol{\lambda}\, \boldsymbol{w}^T$$

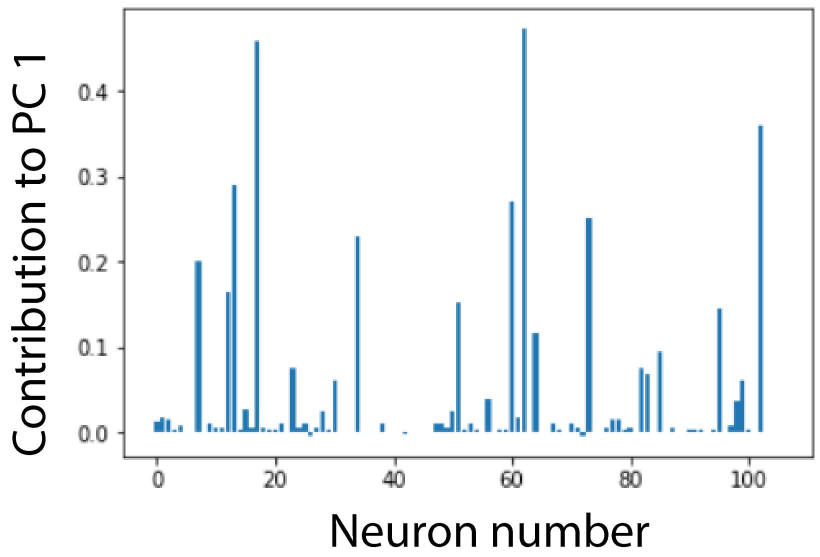
Which eigenvector (**w**) to use? Recall we want to maximize $\boldsymbol{w}\; \Sigma\; \boldsymbol{w}^T$, then since for any eigenvector $\boldsymbol{w_k}$ with eigenvalue $\boldsymbol{\lambda_k}$

$$\boldsymbol{w_k} \Sigma \boldsymbol{w_k}^T \ = \boldsymbol{w_k} \boldsymbol{\lambda_k} \boldsymbol{w_k}^T = \boldsymbol{\lambda_k} \boldsymbol{w_k} \boldsymbol{w_k}^T \ = \ \boldsymbol{\lambda_k}$$

Then the eigenvector $\boldsymbol{w_k}$ the maximizes $\boldsymbol{w_k} \Sigma \boldsymbol{w_k}^T$ is the one with the **LARGEST** eigenvalue $\boldsymbol{\lambda_k}$

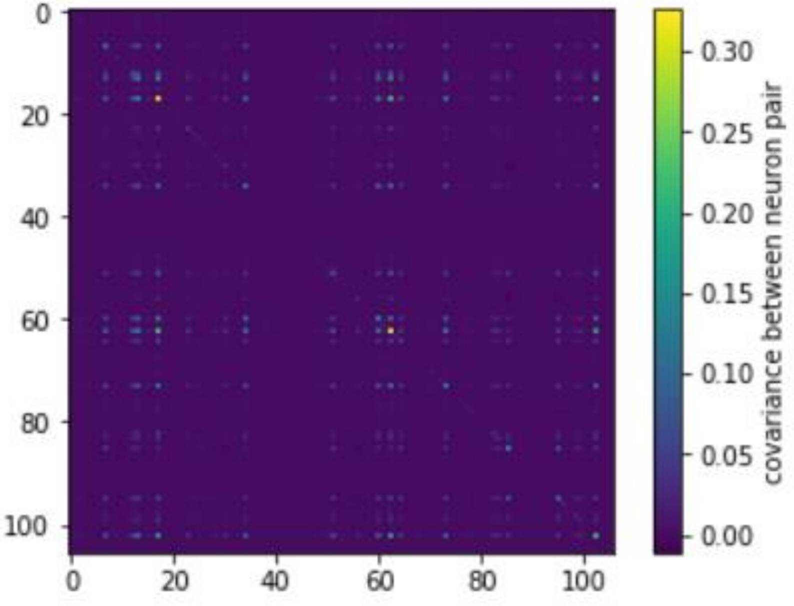
So: PCA is computing the *eigenvectors* of the *covariance matrix* (Σ)

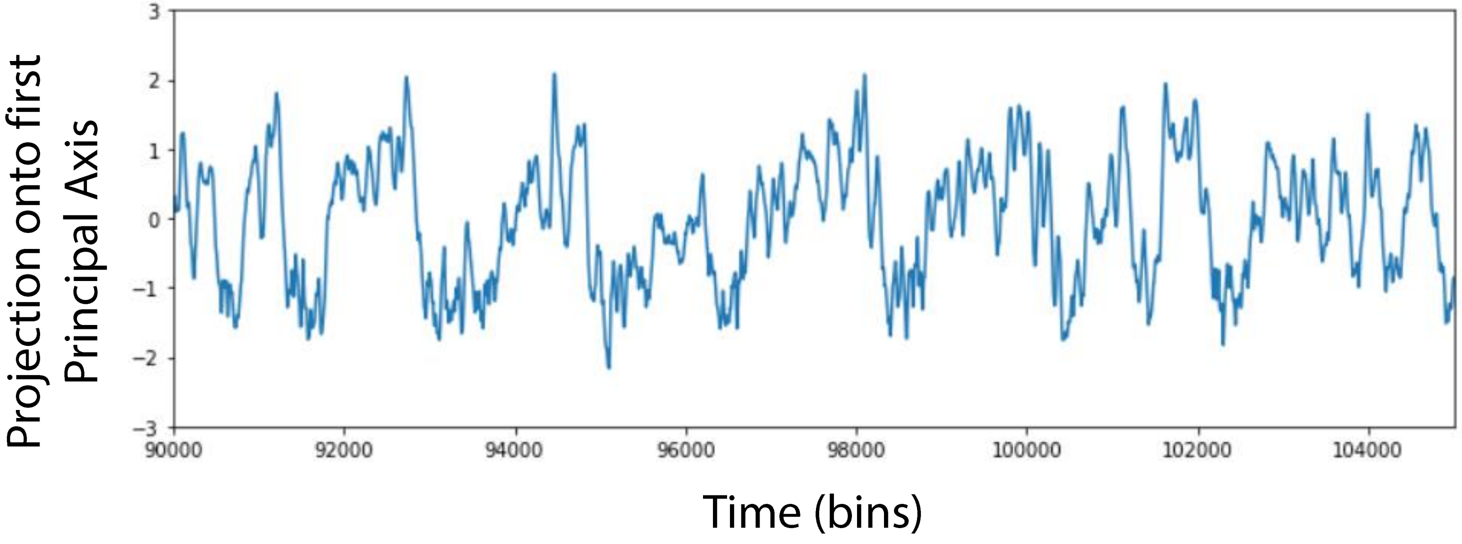




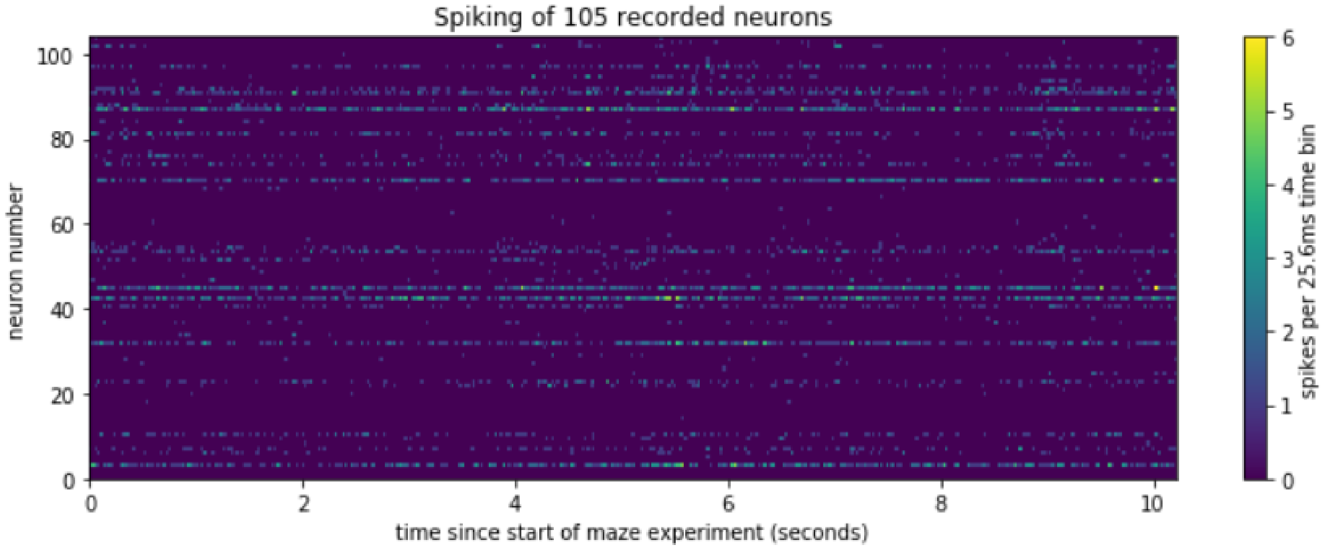
(aka **loadings**)

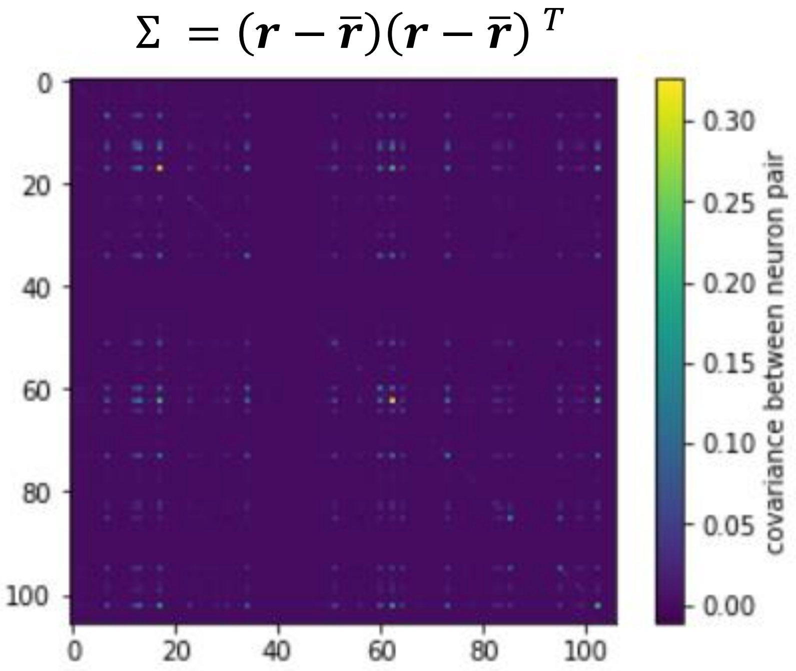
$$\Sigma \ = (\boldsymbol{r}-\bar{\boldsymbol{r}})(\boldsymbol{r}-\bar{\boldsymbol{r}})^{\,T}$$

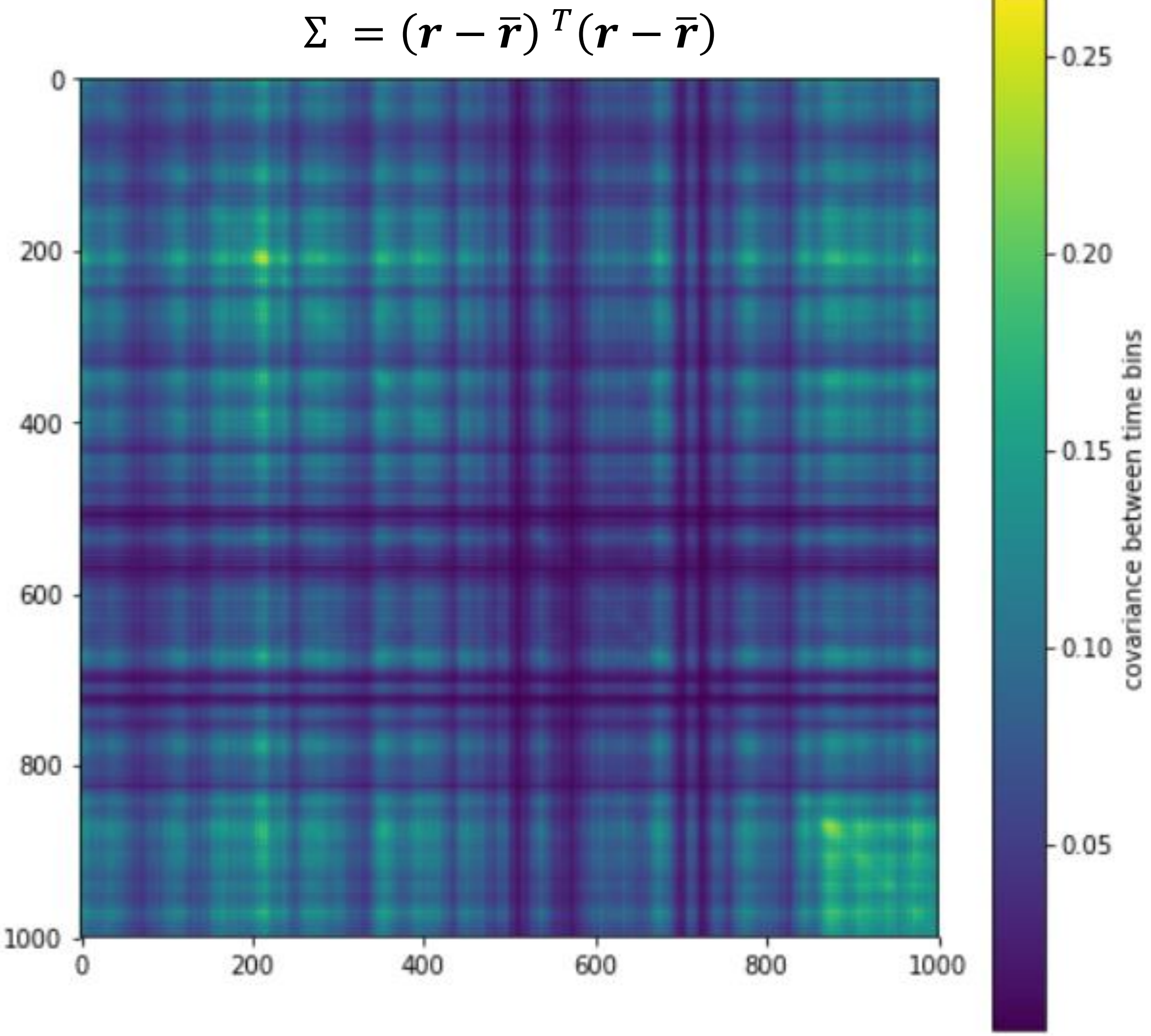




You can compute covariance over *neurons* or over *time* (neurons is easier)

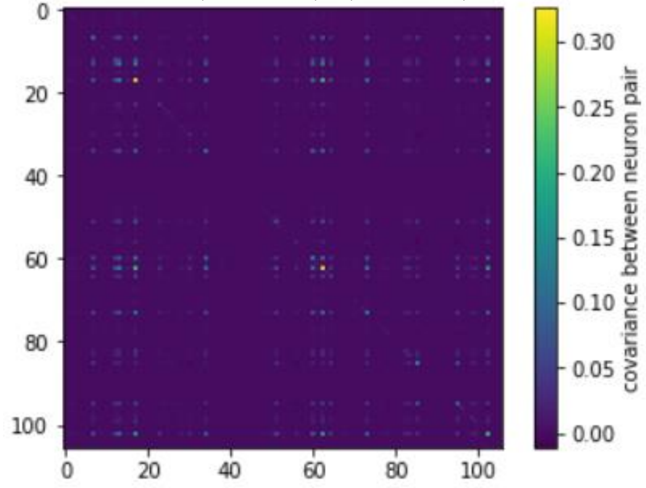




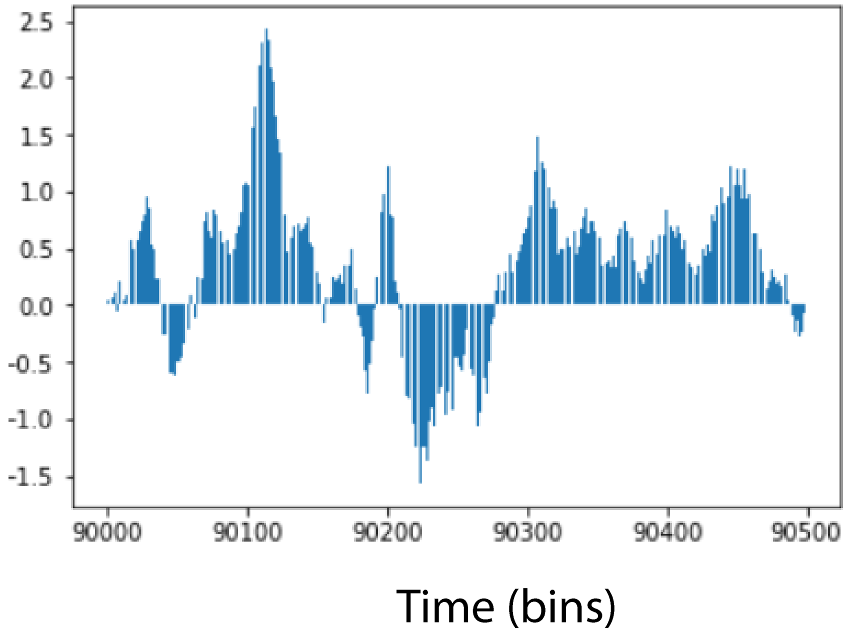


Projecting onto the PCs of the *neural* covariance matrix gives you the PCs of the time covariance matrix, and vice versa

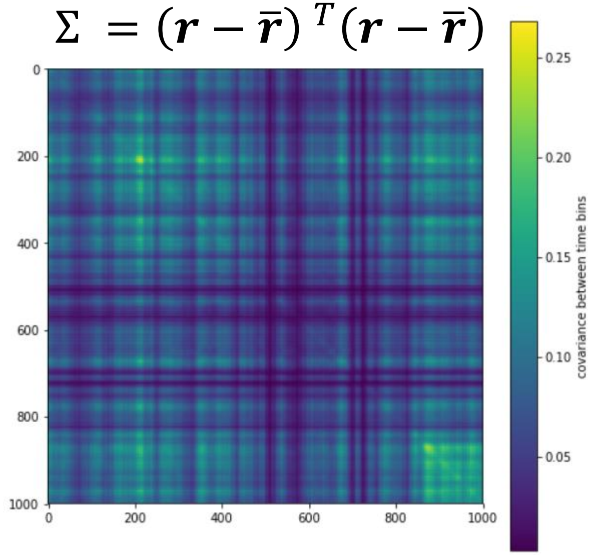
$$\Sigma = (\boldsymbol{r} - \bar{\boldsymbol{r}})(\boldsymbol{r} - \bar{\boldsymbol{r}})^T$$



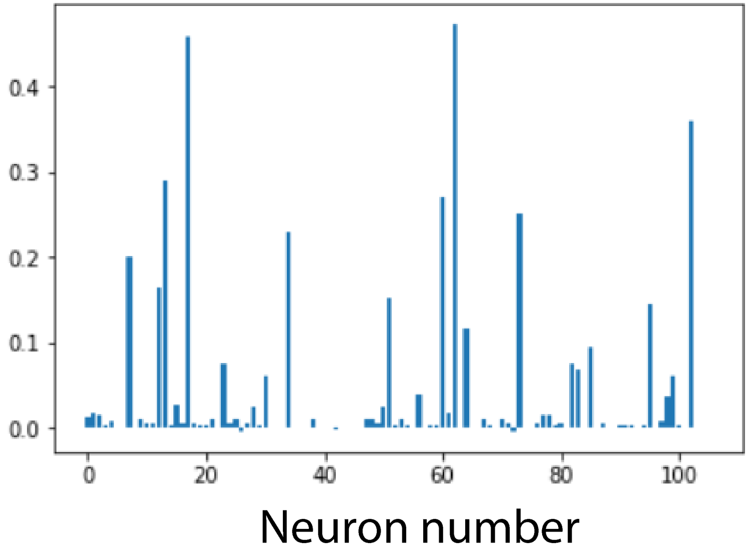
Contribution to PC 1 of temporal covariance matrix



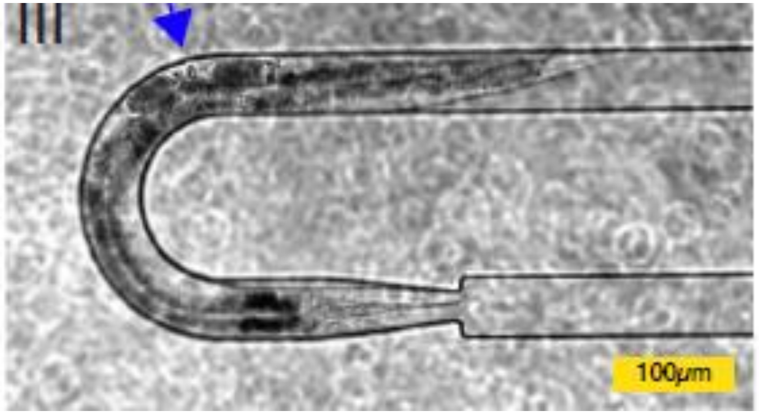
$$\Sigma = (\boldsymbol{r} - \bar{\boldsymbol{r}})^T(\boldsymbol{r} - \bar{\boldsymbol{r}})$$



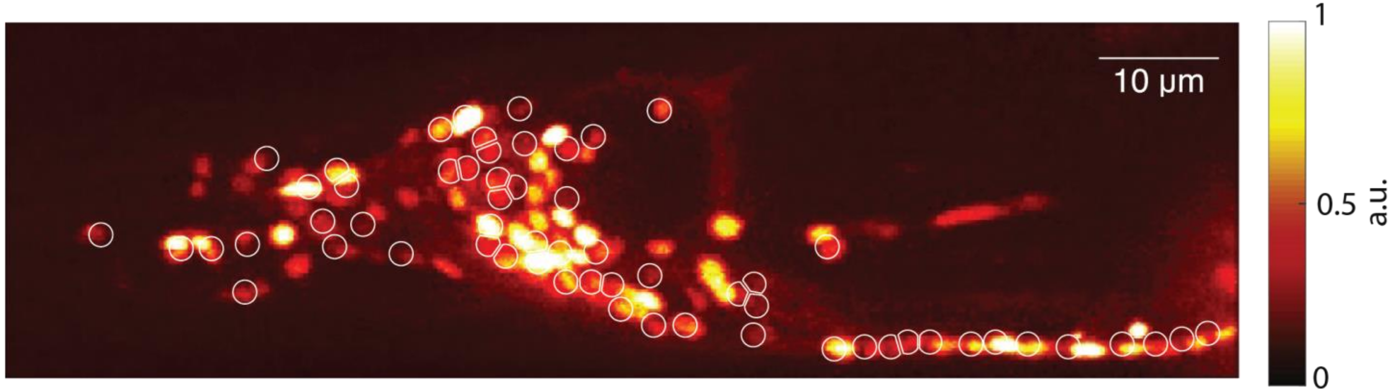
Projection onto first prinicipal axis of temporal covariance matrix



An example from neuroscience!



C elegans immobilized in microfluidic device.



Signal from pan-neuronal, nuclear-localized GCaMP5K in head ganglia, with example ROIs.

Image (most) neurons simultaneously in an immobilized worm, and visualize dynamics with PCA.

Kato et al 2015

record Ca^{2+} signal

↓

compute $\Delta F/F$

↓

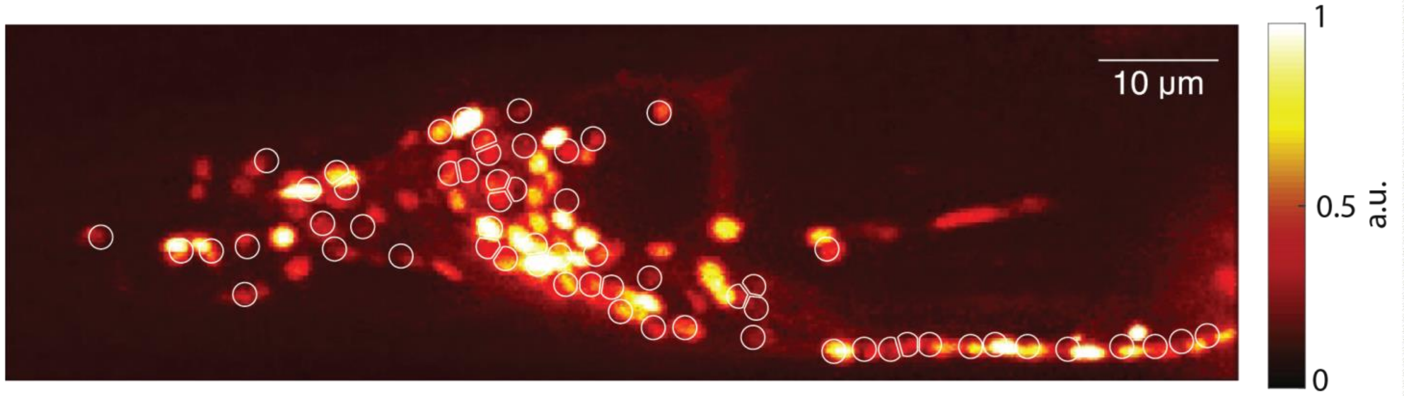
take derivative

↓

perform PCA on derivative

↓

project $\Delta F/F$ onto PC dimensions



Kato et al 2015

record Ca^{2+} signal

↓

compute ΔF/F

↓

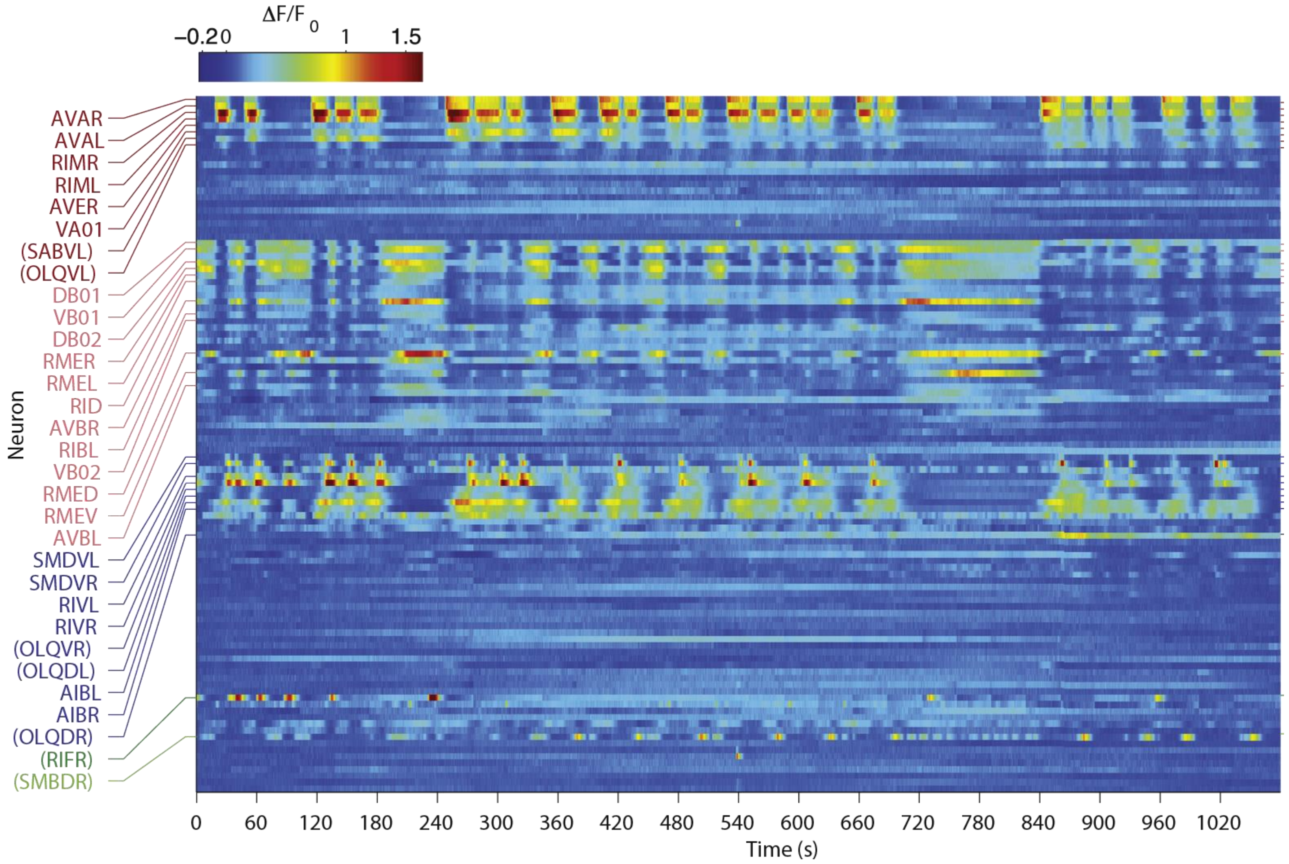
take derivative

↓

perform PCA on derivative

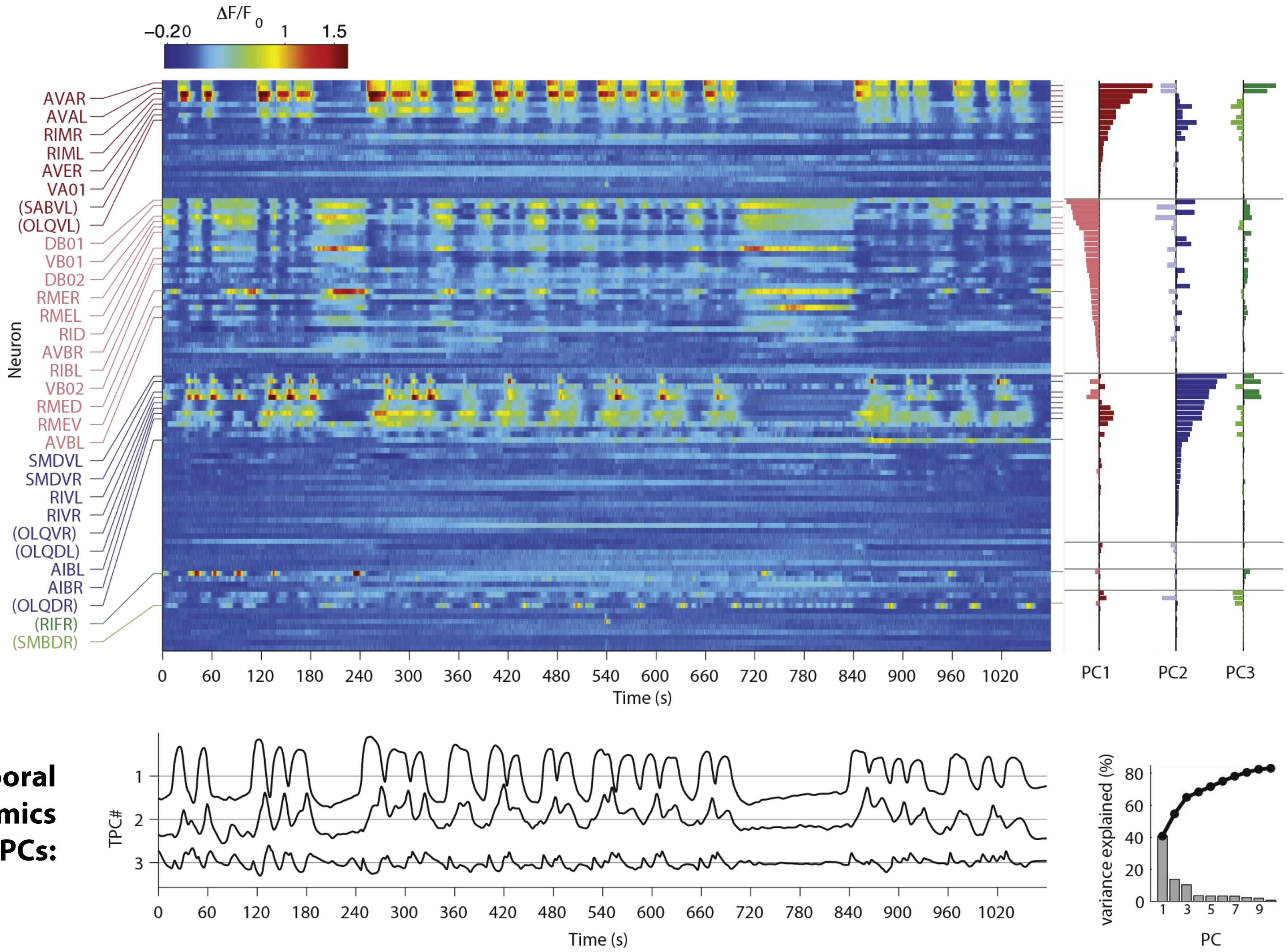
↓

project ΔF/F onto PC dimensions



n = 109 neurons
(in each of 5 worms)

Kato et al 2015



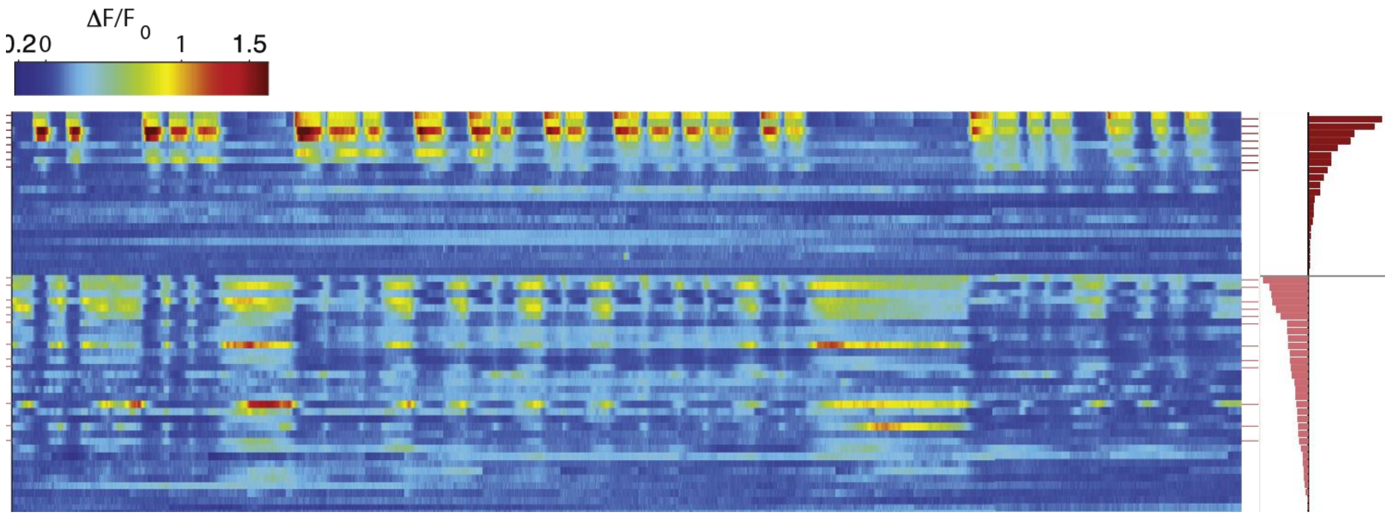
Neural contributions to first 3 PCs

Temporal dynamics of first 3 PCs:

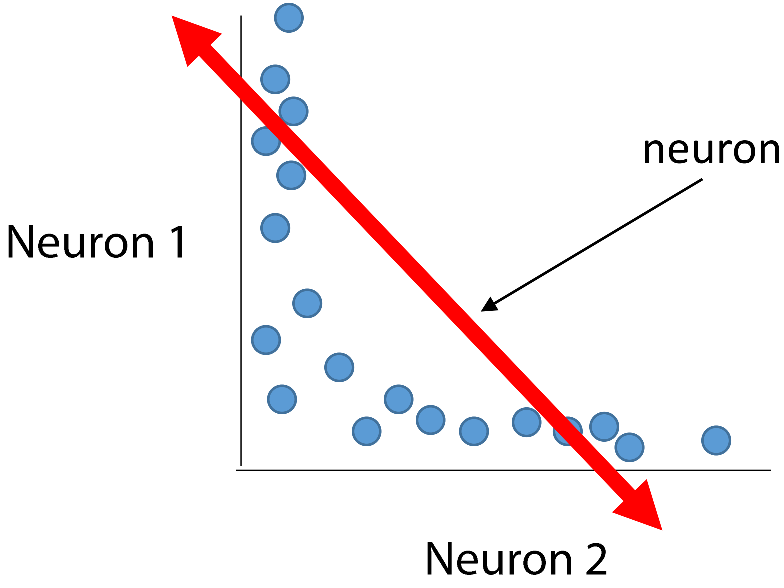
variance explained

Kato et al 2015

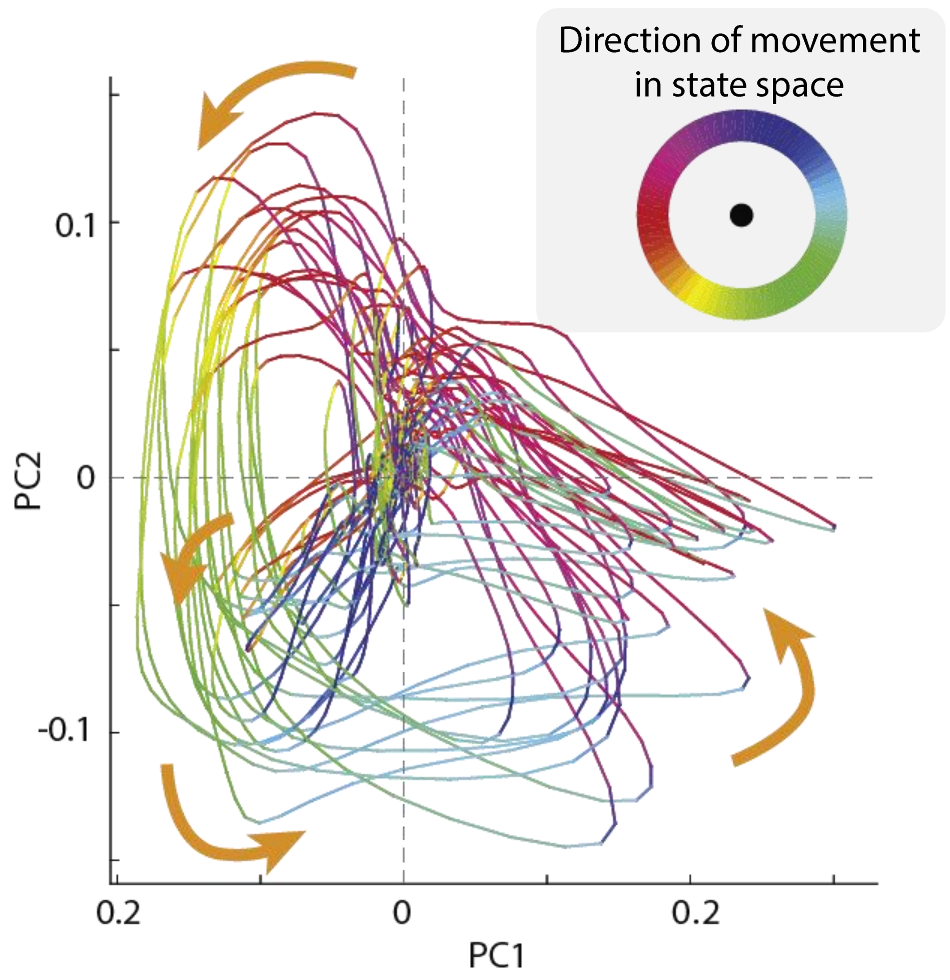
Important consideration: PCA allows *negative* contributions to PCs

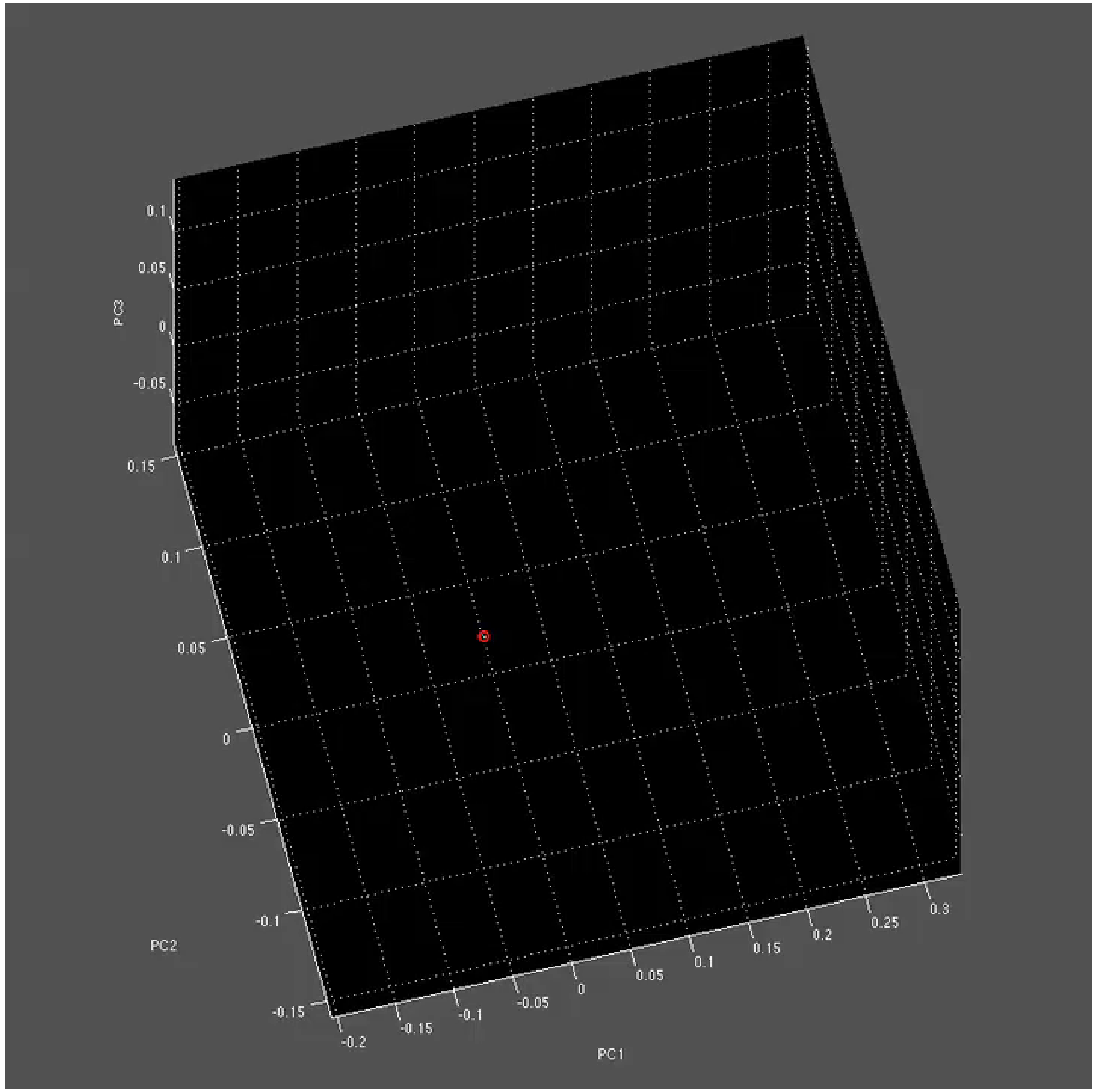


Neurons that most strongly contribute to PC 1



Kato et al 2015





Kato et al 2015