Lab 4 — 2D Regression problem

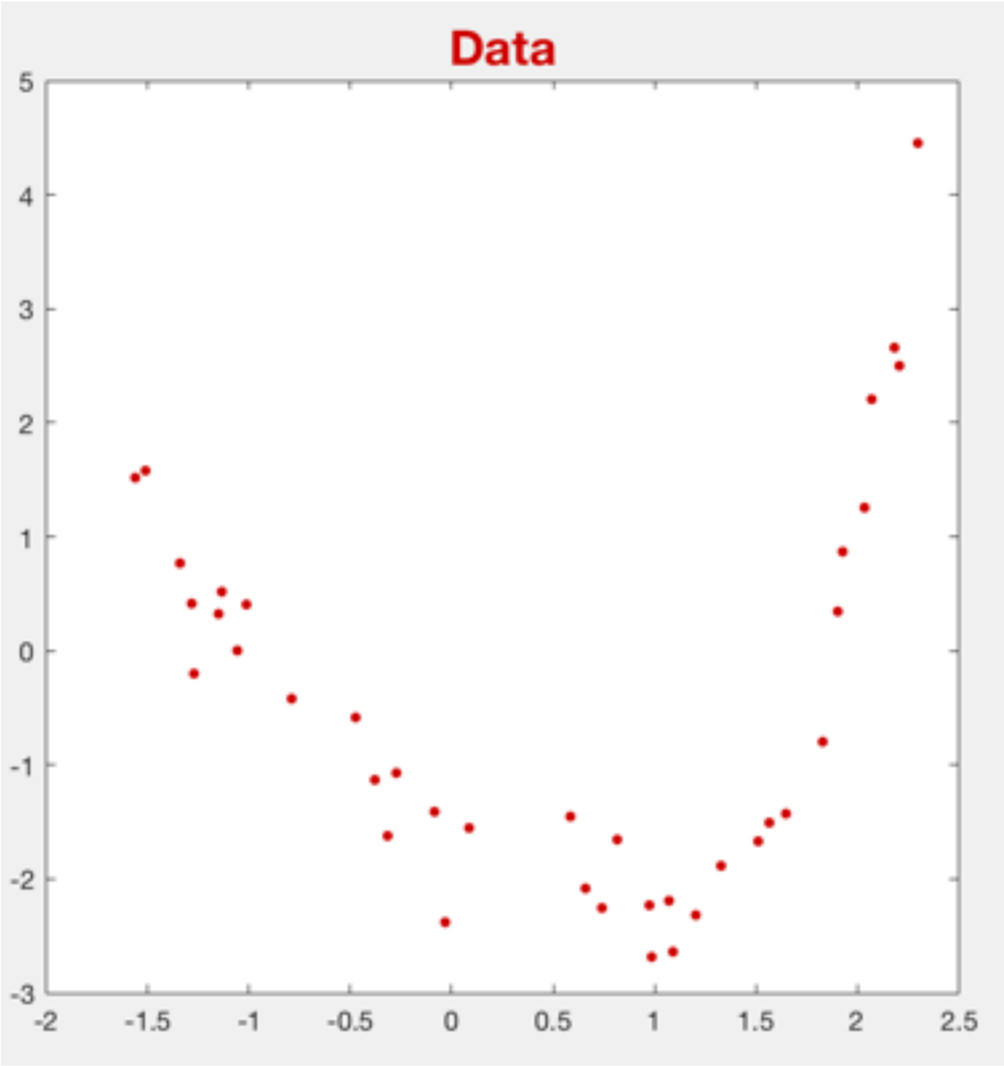
October 5th, 2018

Outline the problem:

- Load regress1.mat
- Plot Y as function of X
- Least squares fit of data with polynomial of order 0-5
  - Using SVD
- Plot the fit
- Plot the squared errors as function of order of poly

Plot x as function of y

- This is the data we want to fit

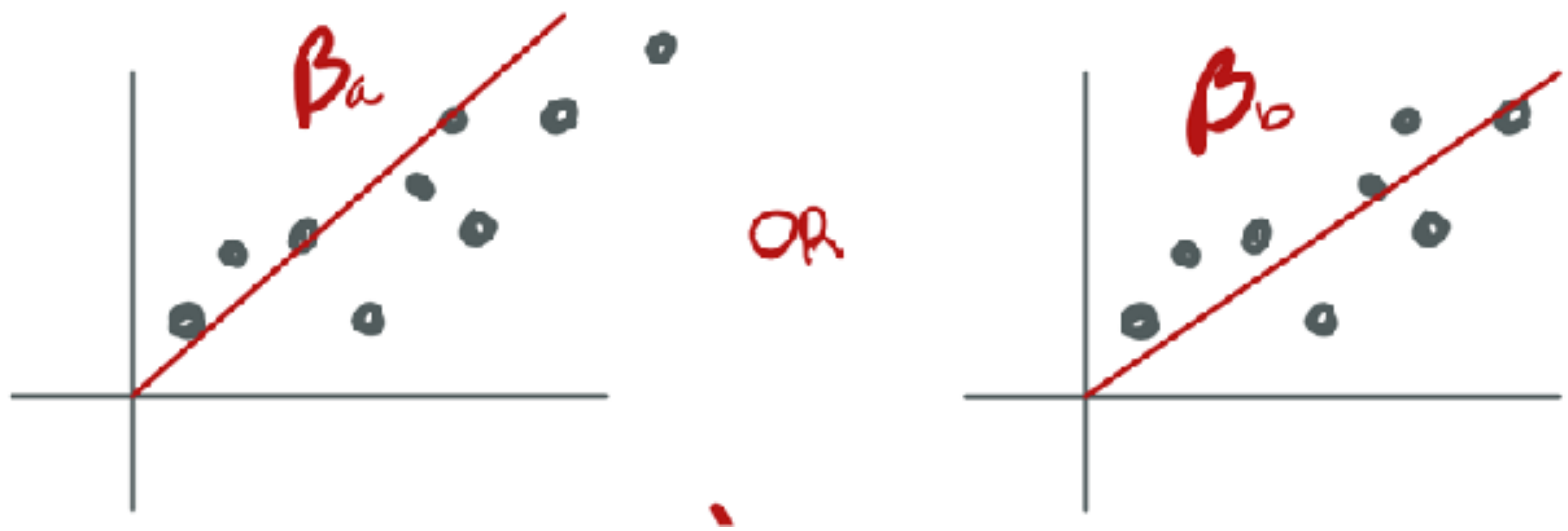


Outline the solution:

1. Think
2. Write the expression
3. X = USVt
4. Rotate into S-space
5. “Trim” to relevant expression
6. Solve for the combination of ßs that minimize the expression
7. Rotate back
8. Fit your data
9. Asses your work:
    1. Plot it
    2. Calculate the error

Step 1 — Thinking

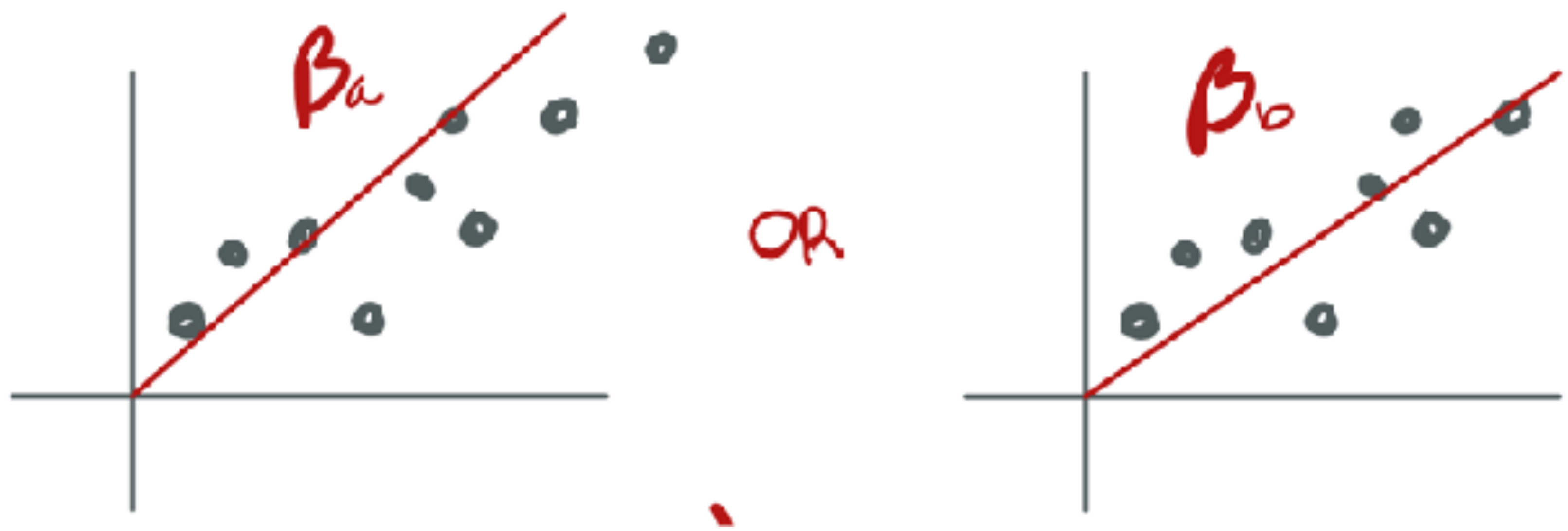
- Want to find an equation to fit our data.



- What's our question?
- Finding the beta that fits data best, produces the smallest error from the data points

Step 1 — Thinking

- Want to find an equation to fit our data.



- What's our question?
- Finding the beta that fits data best, produces the smallest error from the data points

$$\min_{\beta} ||\vec{y} - \beta\vec{x}||^2$$

Step 1 — Thinking

- 1st order example:

$$y = \underbrace{m}_{\text{constant}}x + \underbrace{b}_{\text{constant}}$$

High school notation

$$\vec{y} = \underbrace{\beta_1}_{\text{constant}}\vec{x} + \underbrace{\beta_0}_{\text{constant}}$$

Math tools notation

- Write it in polynomial notation:

$$\vec{y} = \beta_1 \vec{x}^1 + \beta_0 \vec{x}^0$$

Step 2 — writing expression

- Write it in polynomial notation:

$$\vec{y} = \beta_1 \vec{x}^1 + \beta_0 \vec{x}^0$$

- Higher orders:

$$y = \beta_2 x^2 + \beta_1 x^1 + \beta_0 x^0$$

- Re-write in matrix form:

$$\beta_2 \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}^2 + \beta_1 \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}^1 + \beta_0 \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ \vdots & \vdots & \vdots \\ x_n^2 & x_n & 1 \end{bmatrix}}_{\text{call this matrix } X} \underbrace{\begin{bmatrix} \beta_2 \\ \beta_1 \\ \beta_0 \end{bmatrix}}_{\text{call this vector } \vec{\beta}}$$

$$\Rightarrow X\vec{\beta}$$

$$\Rightarrow \boxed{\min_{\vec{\beta}} \|\vec{y} - X\beta\|^2}$$

Step 3 — SVD

- Our expression:

$$\min_{\beta} \|\vec{y} - X\vec{\beta}\|^2$$

- Do svd of matrix X:

$$= \min_{\beta} \|\vec{y} - USV^T\vec{\beta}\|^2$$

- Rearrange:

$$= \min_{\beta} \|U^T\vec{y} - SV^T\vec{\beta}\|^2$$

- Why do we want to rotate y by Ut?

Step 4 — rotate into S-space

- We rotated y and ß to the "S space"

$$= \min_{\beta} \| \underbrace{U^T \vec{y}}_{\vec{y}^*} - S \underbrace{V^T \vec{\beta}}_{\vec{\beta}^*} \|^2$$

Don't lose track of your variables!

$$U^T \vec{y} = \vec{y}^*$$

$$V^T \vec{\beta} = \vec{\beta}^*$$

- Rewrite:

$$= \min_{\beta} \| \vec{y}^* - S \vec{\beta}^* \|^2$$

- Notice:
  - How many non-zero values does S have for a first order polynomial?

Step 5 — Trim

- Only 2 non-zero S values in this example:

$$\vec{y}^* - S\vec{\beta}^*$$

$$\begin{bmatrix} y_1^* \\ y_2^* \\ \vdots \\ y_n^* \end{bmatrix} - \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \\ \vdots & \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \beta_1^* \\ \beta_2^* \end{bmatrix} = \begin{bmatrix} y_1^* \\ y_2^* \\ \vdots \\ y_n^* \end{bmatrix} - \begin{bmatrix} s_1\beta_1^* \\ s_2\beta_2^* \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Don't lose track of your variables!

$$U^T\vec{y} = \vec{y}^*$$

$$V^T\vec{\beta} = \vec{\beta}^*$$

- The betas we choose will only affect y*_1 and y*_2

$$\vec{y}^* - S\vec{\beta}^*$$

$$\begin{bmatrix} y_1^* \\ y_2^* \end{bmatrix} - \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix} \begin{bmatrix} \beta_1^* \\ \beta_2^* \end{bmatrix}$$

- How much can we minimize this expression?

Step 6 — solve for ßopt

- How much can we minimize this expression?

$$\vec{y}^* - S\vec{\beta}^*$$

$$\begin{bmatrix} y_1^* \\ y_2^* \end{bmatrix} \quad \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix} \begin{bmatrix} \beta_1^* \\ \beta_2^* \end{bmatrix}$$

Don't lose track of your variables!

$$U^T\vec{y} = \vec{y}^*$$

$$V^T\vec{\beta} = \vec{\beta}^*$$

$$\vec{y}^* - S\vec{\beta}^* = 0$$

$$\vec{y}^* - S\vec{\beta}^*_{opt} = 0$$

$$\vec{y}^* = S\vec{\beta}^*_{opt}$$

$$S^{-1}\vec{y}^* = \vec{\beta}^*_{opt}$$

$$\frac{y_1^*}{s_1} = \beta^*_{opt\,1}$$

$$\frac{y_2^*}{s_2} = \beta^*_{opt\,2}$$

Step 7 — going back

- Good thing we kept track of our variables!

Don't lose track of your variables!

$$U^T\vec{y} = \vec{y}^*$$

$$V^T\vec{\beta} = \vec{\beta}^*$$

$$\vec{\beta} = V\vec{\beta}^*$$

- Remember: ß is the solution, not ß*
- The green ßopt is what we plug in our original expression:

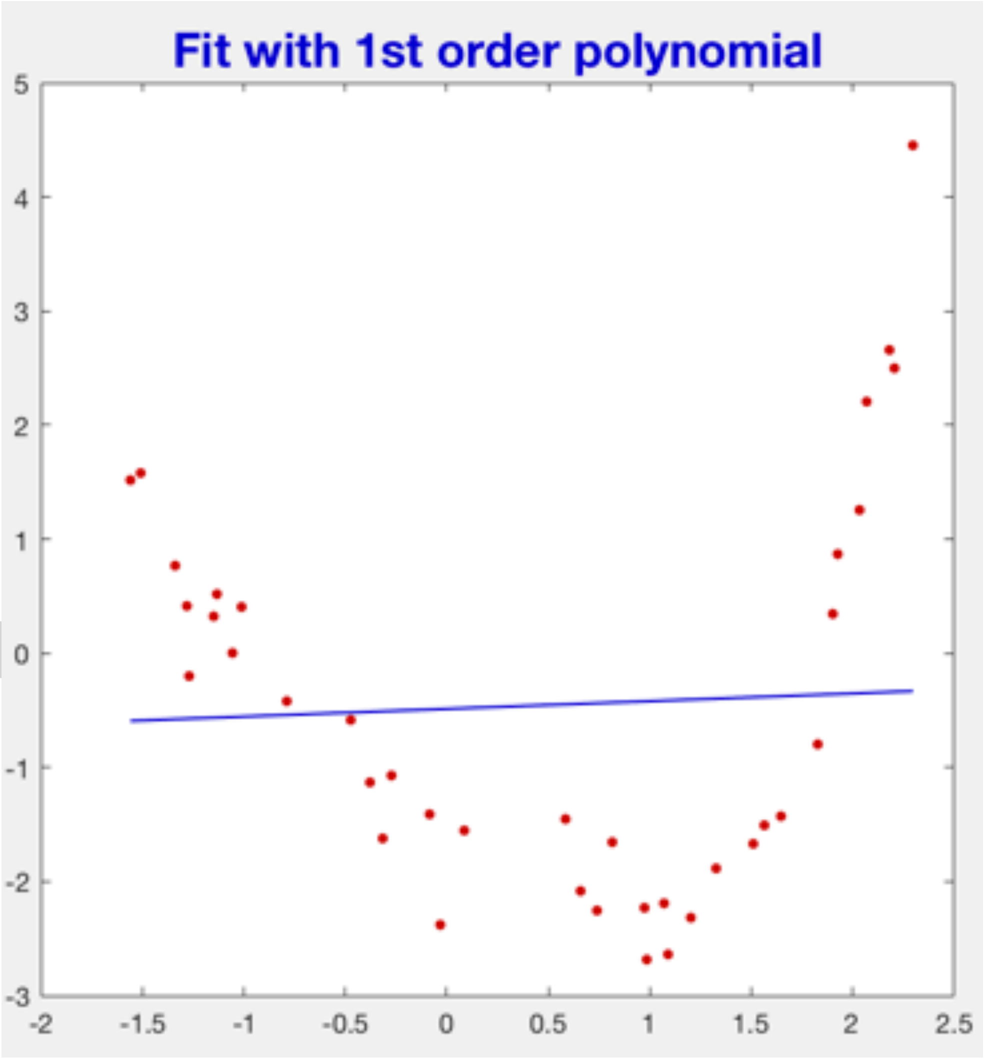
$$\min_{\beta} \|\vec{y} - X\vec{\beta}\|^2$$

Summary of steps:

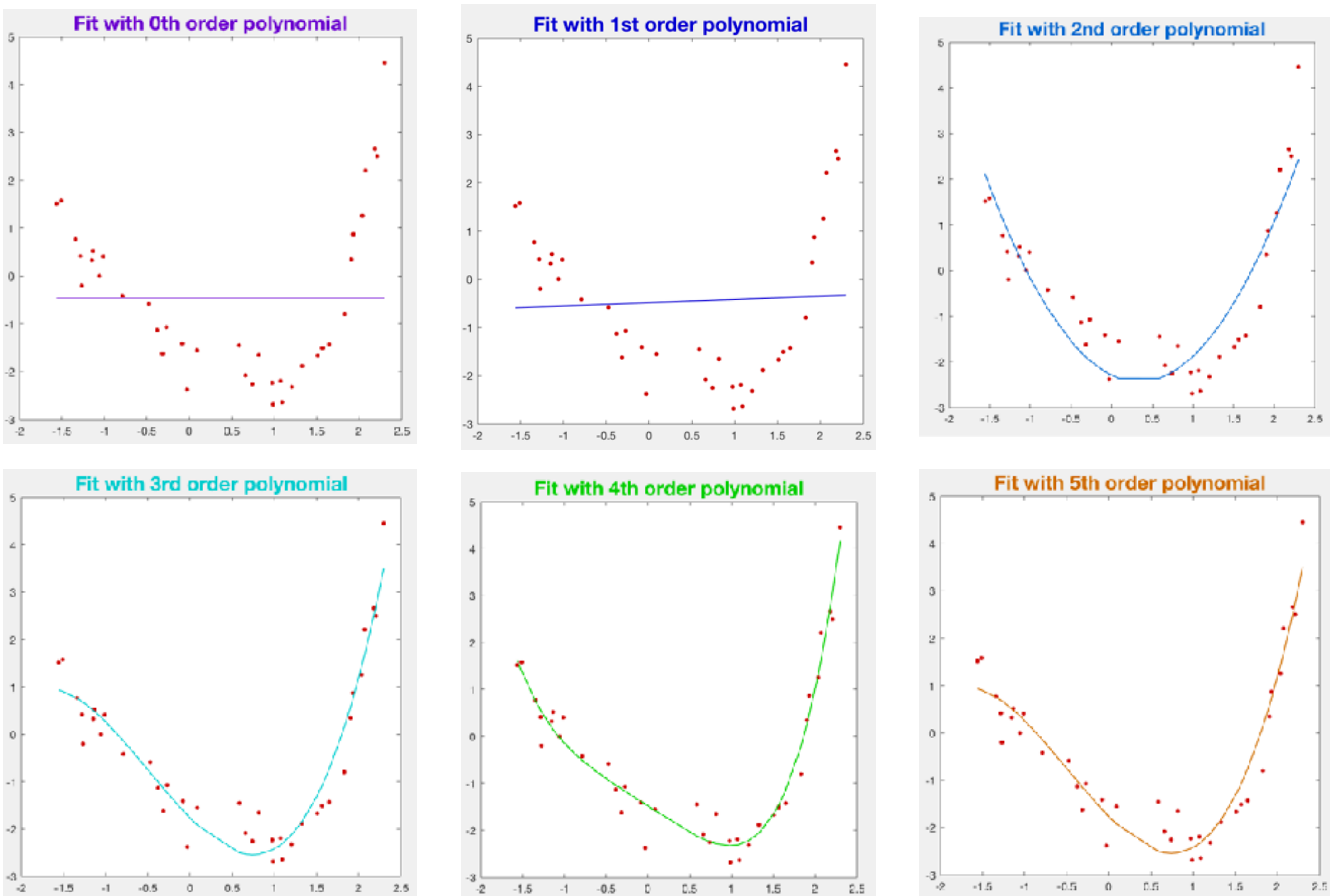
1. Think
2. Write the expression
3. X = USVt
4. Rotate into S-space
5. “Trim” to relevant expression
6. Solve for the combination of ßs that minimize the expression
7. Rotate back
8. Fit your data
9. Asses your work:
    1. Plot it
    2. Calculate the error

Summary of steps:

1. Think
2. Write the expression
3. X = USVt
4. Rotate into S-space
5. “Trim” to relevant expression
6. Solve for the combination of ßs t
   expression
7. Rotate back
8. Fit your data
9. Asses your work:
   1. Plot it
   2. Calculate the error

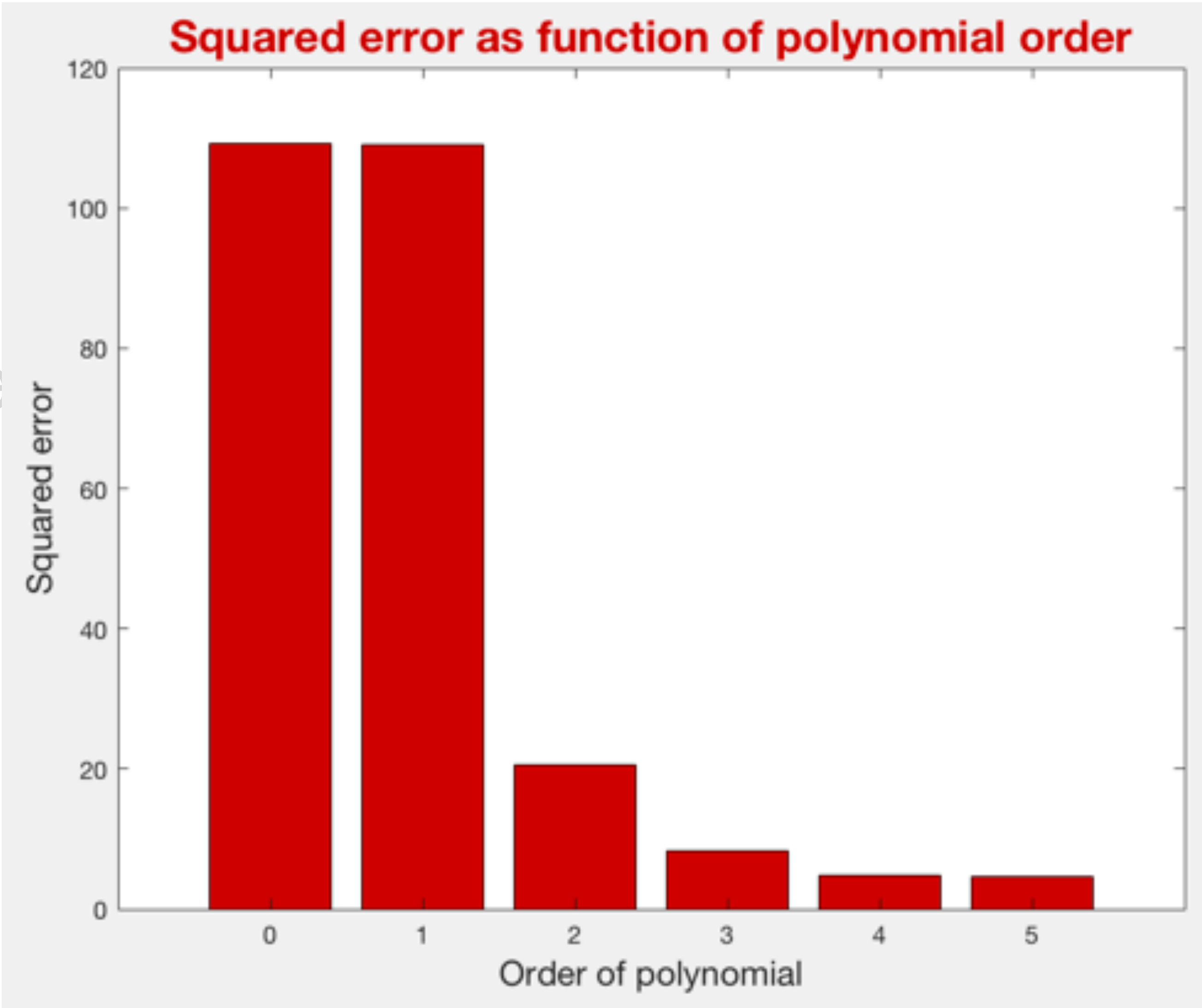


Different orders:



Error:

1. Think
2. Write the expression
3. X = USVt
4. Rotate into S-space
5. "Trim" to relevant expre
6. Solve for the combinati expression
7. Rotate back
8. Fit your data
9. Asses your work:
    1. Plot it
    2. Calculate the error

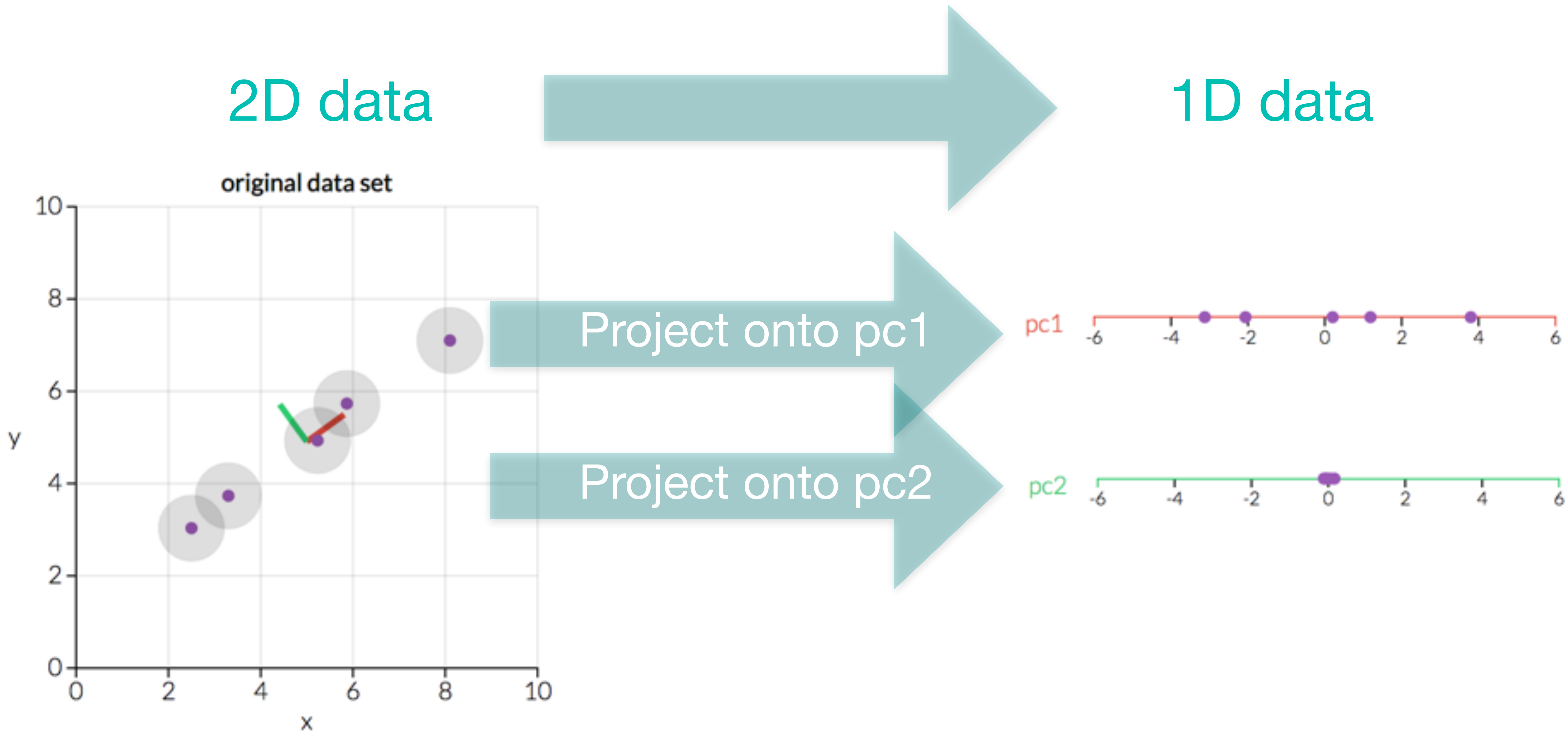


Lab 4 — PCA overview

October 5th, 2018

Reducing dimensionality

- Complicated to work with high dimensional data
- Reducing dimensionality is good
- Example for visualization: 2d —> 1d



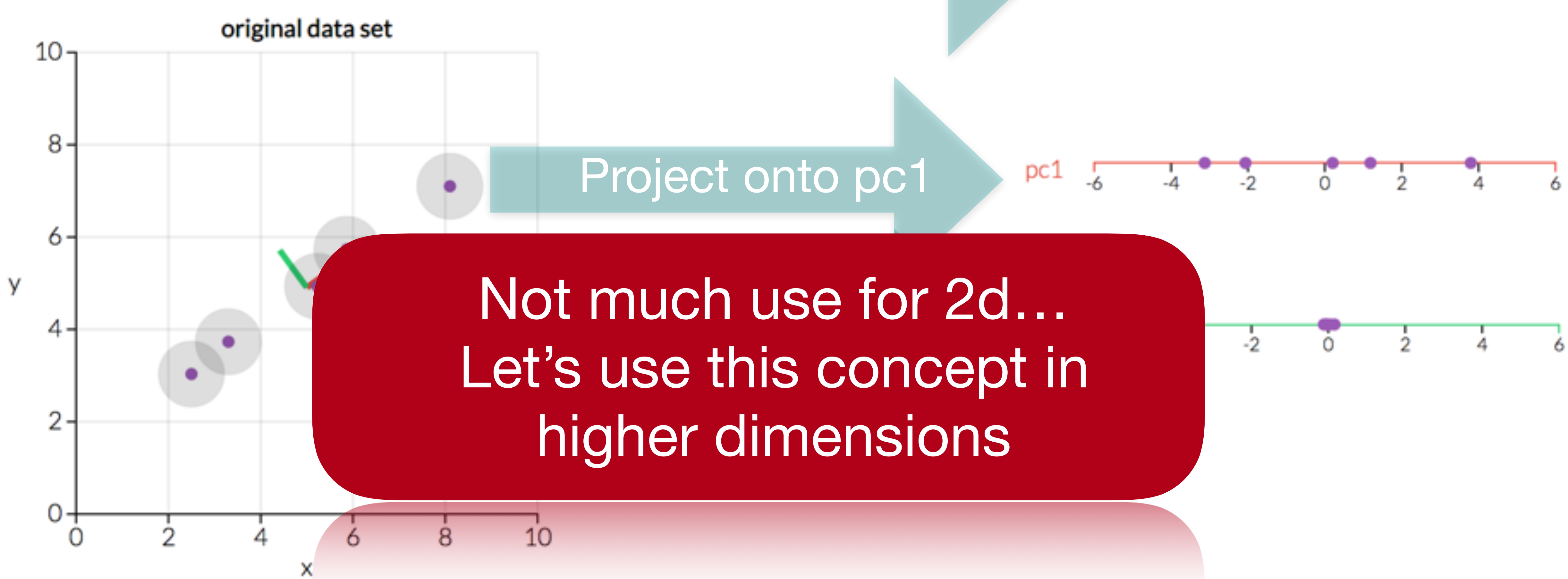
Check out the animation! http://setosa.io/ev/principal-component-analysis/

Reducing dimensionality

- Complicated to work with high dimensional data
- Reducing dimensionality is good
- Example for visualization: 2d —> 1d

2D data

1D data

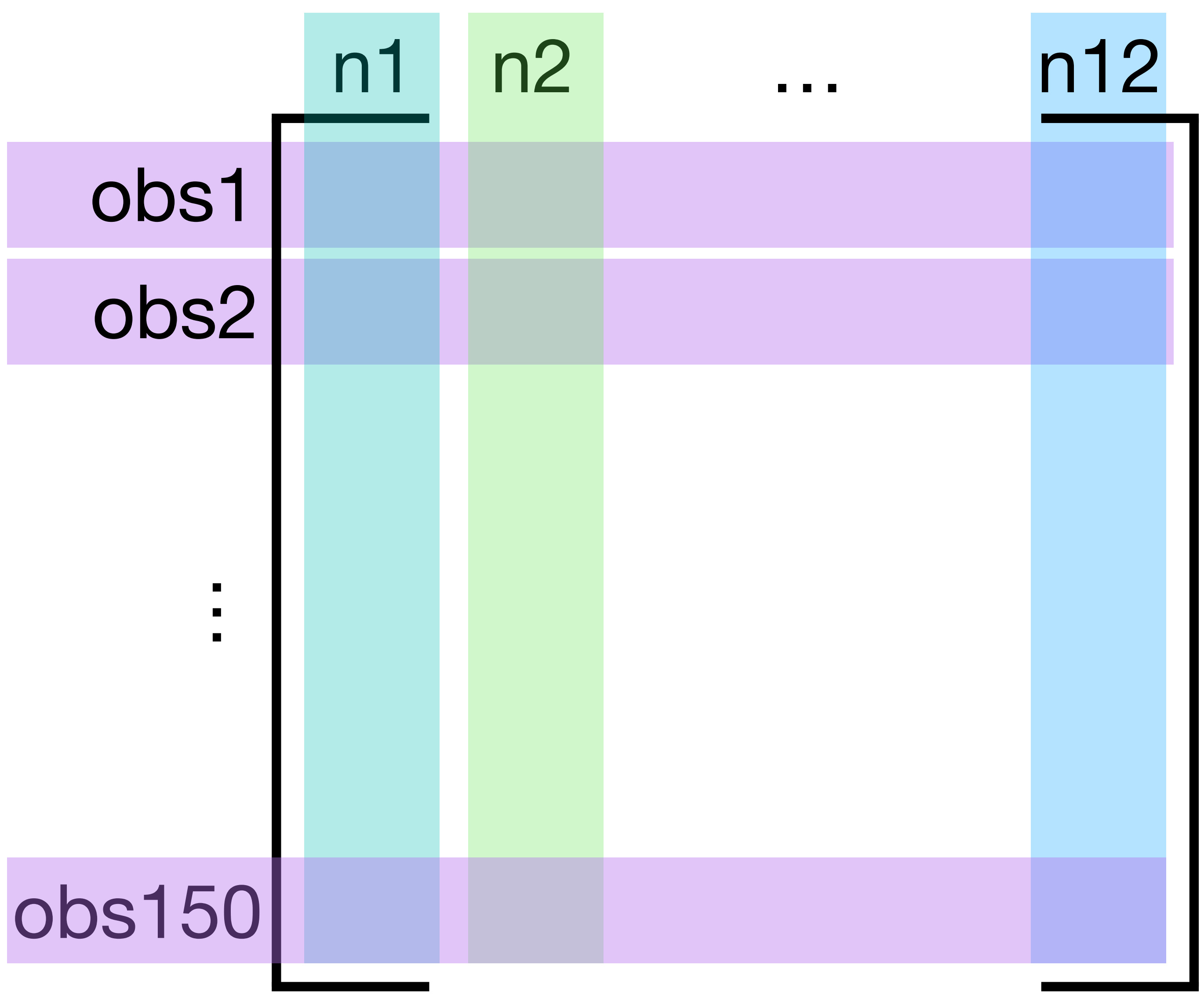


Not much use for 2d…
Let's use this concept in
higher dimensions

Check out the animation! http://setosa.io/ev/principal-component-analysis/

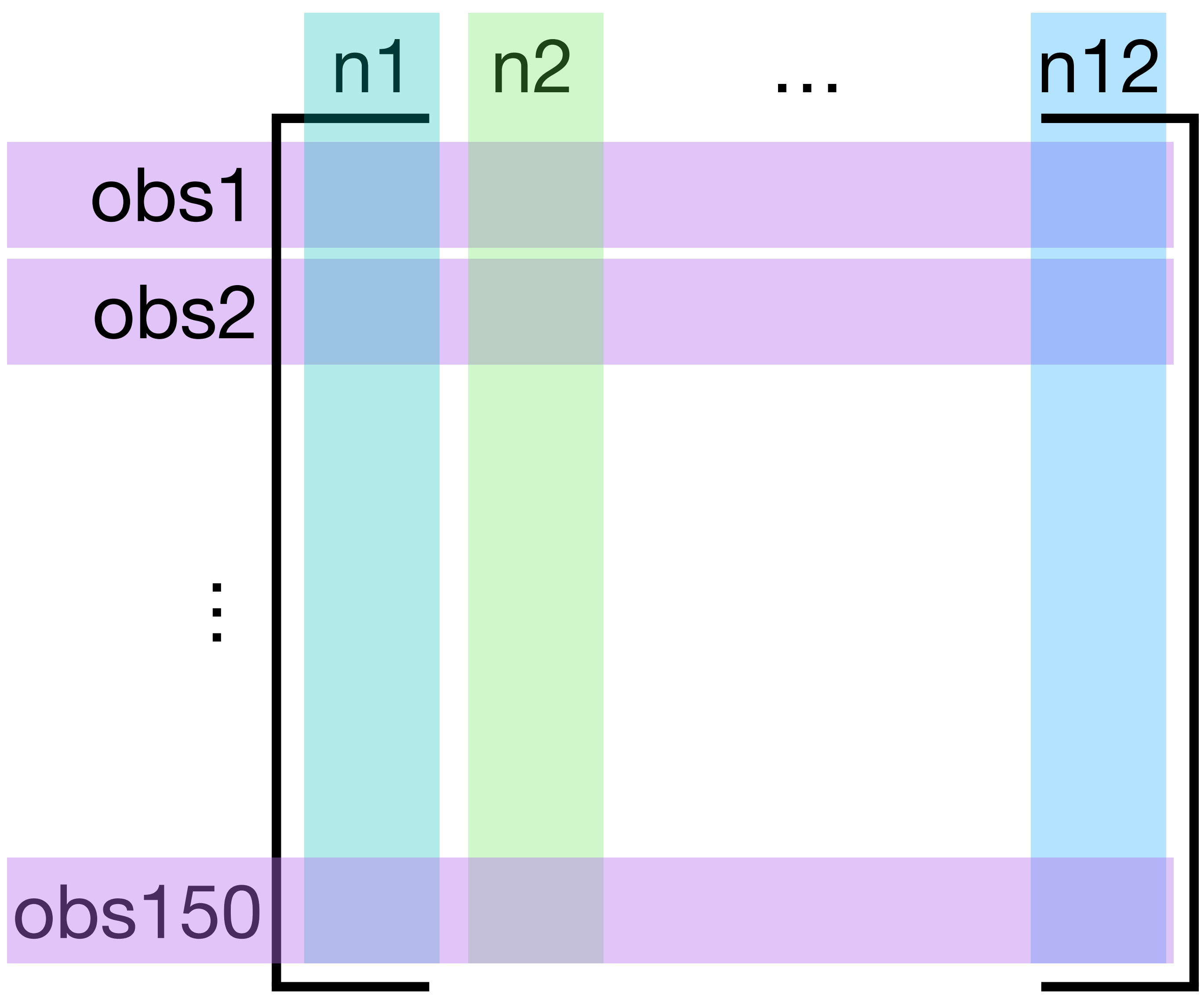
12 Neurons — intro

- Recording from 12 n simultaneously
- Record 150 observations



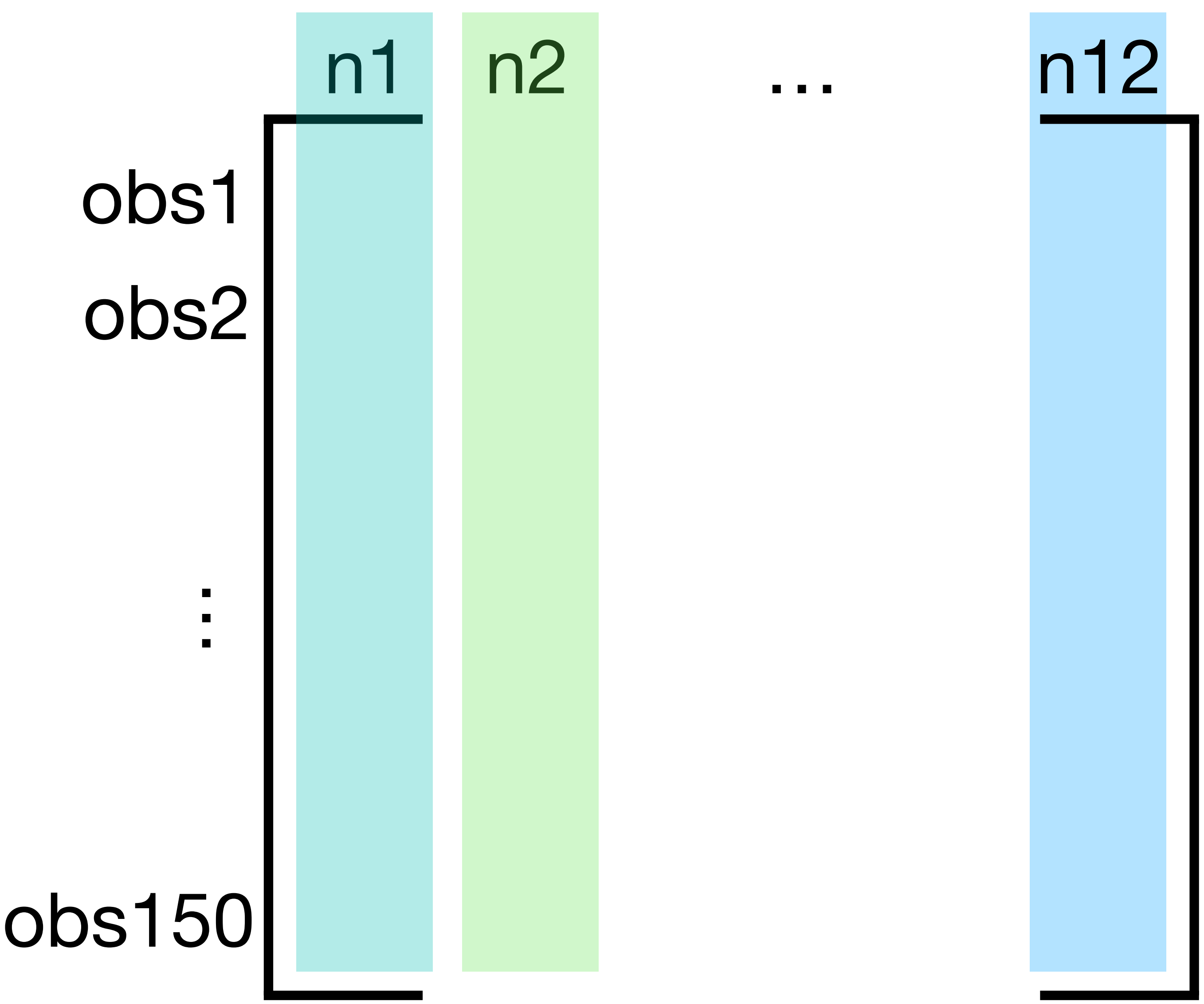
Reducing dimensionality

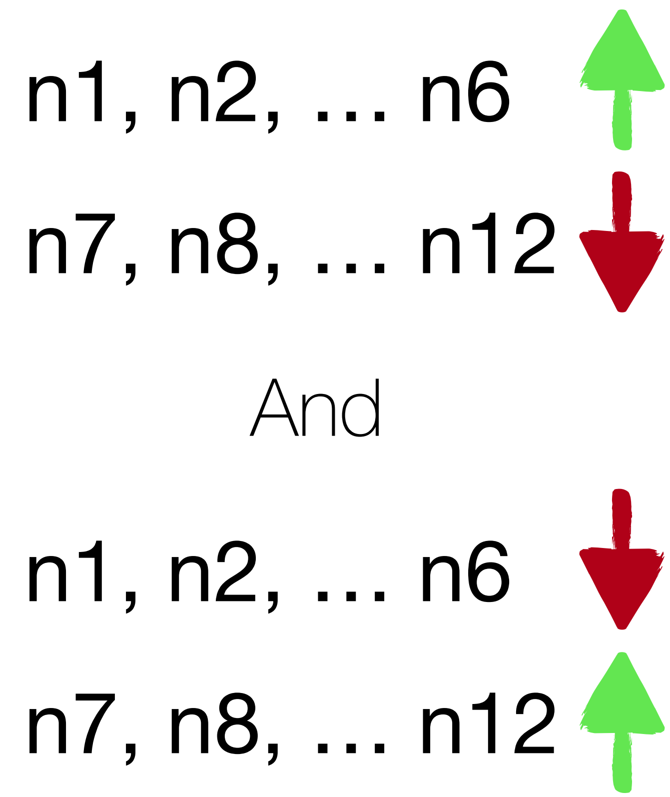
- Low dim representation of the 12 neurons
- OR low dim representation of the 150 observations



Thinking… it's important

- What would it mean to reduce the dimensionality of the 12 neurons?
- Simple ideal example:

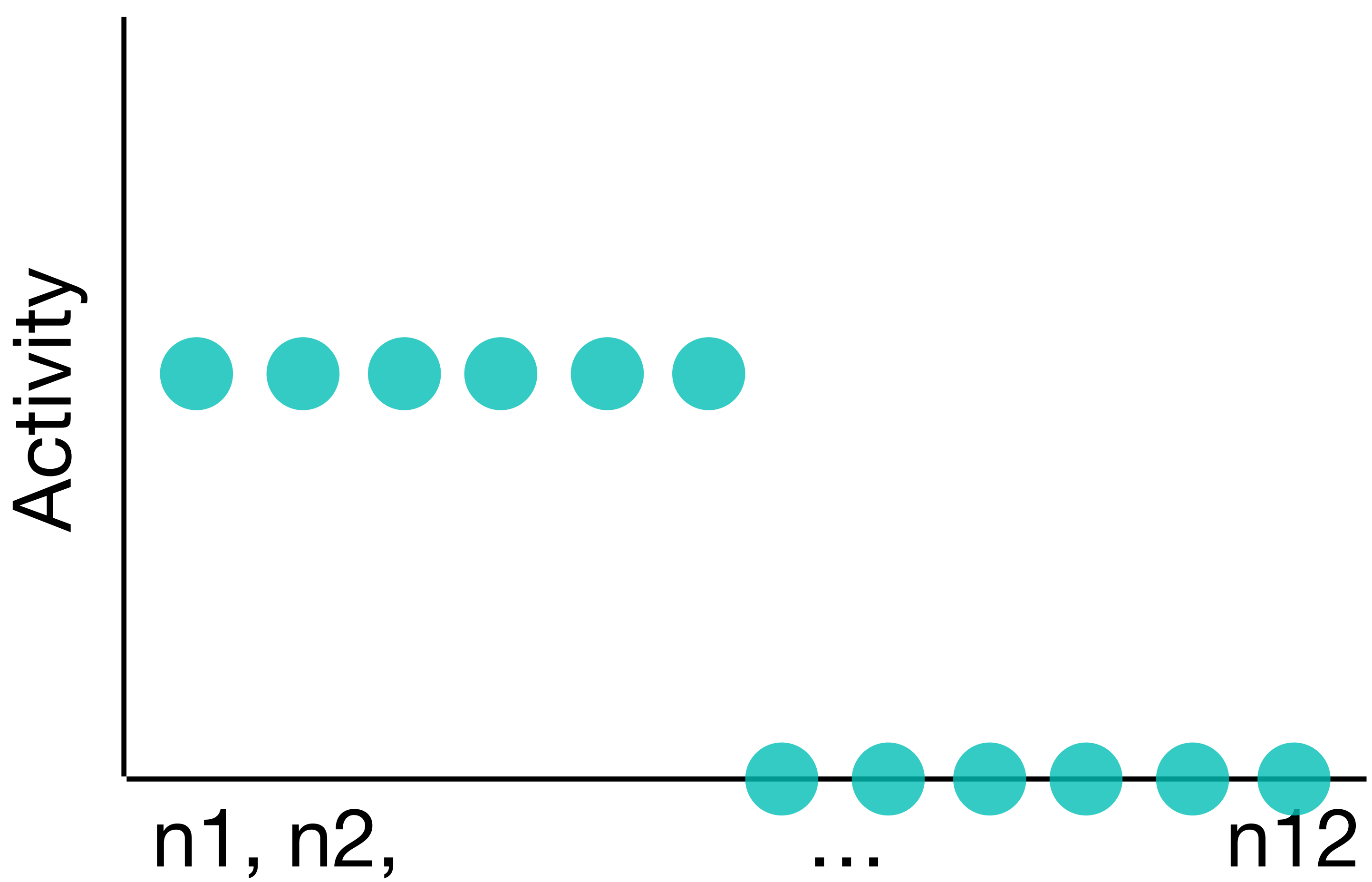




Thinking… it's important

- What would it mean to reduce the dimensionality of the 12 neurons?
- Simple ideal example:

At some time point:



n1, n2, … n6 ↑

n7, n8, … n12 ↓

And

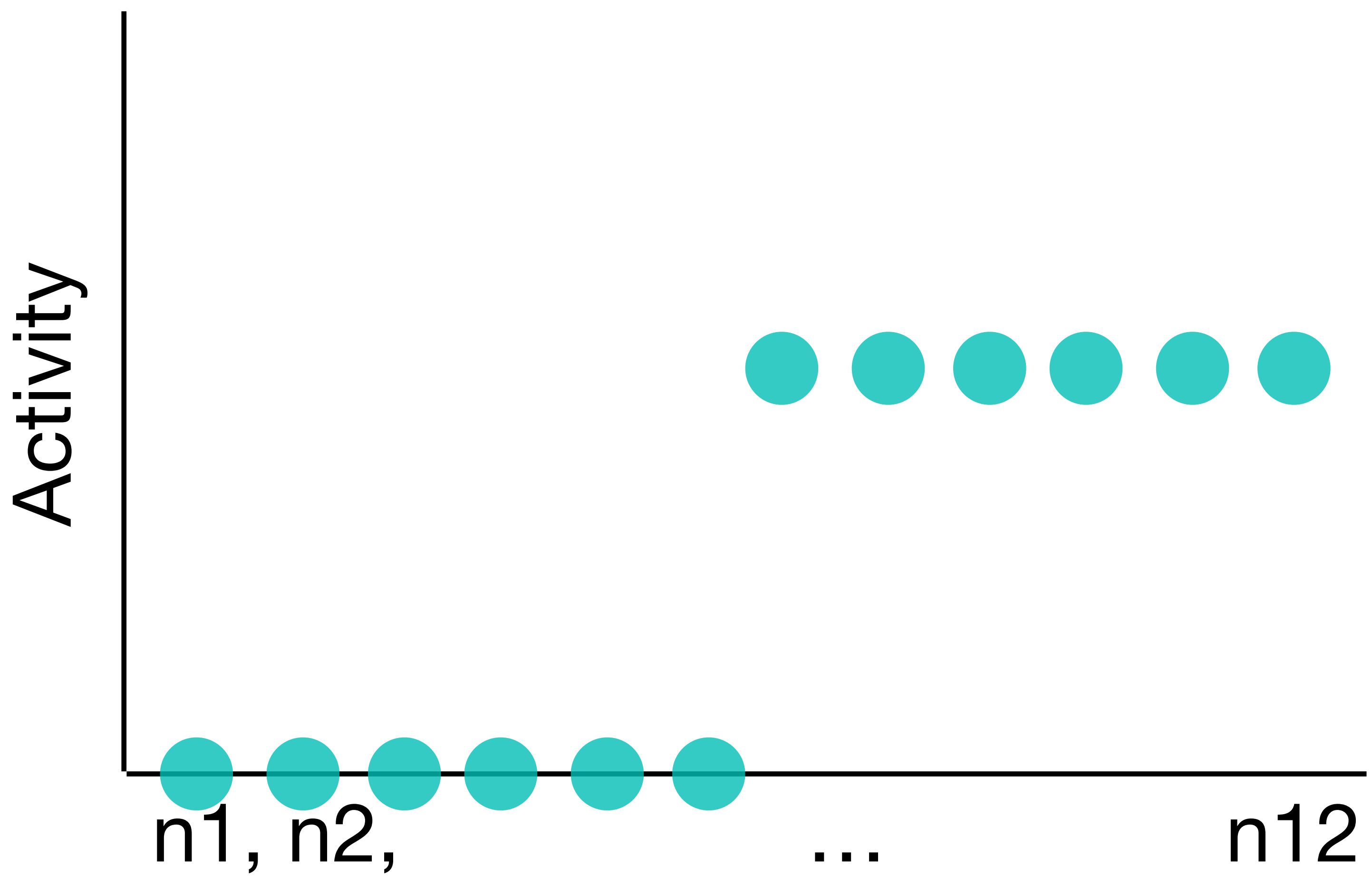
n1, n2, … n6 ↓

n7, n8, … n12 ↑

Thinking… it's important

- What would it mean to reduce the dimensionality of the 12 neurons?
- Simple ideal example:

At some other time point:



n1, n2, … n6

n7, n8, … n12

And

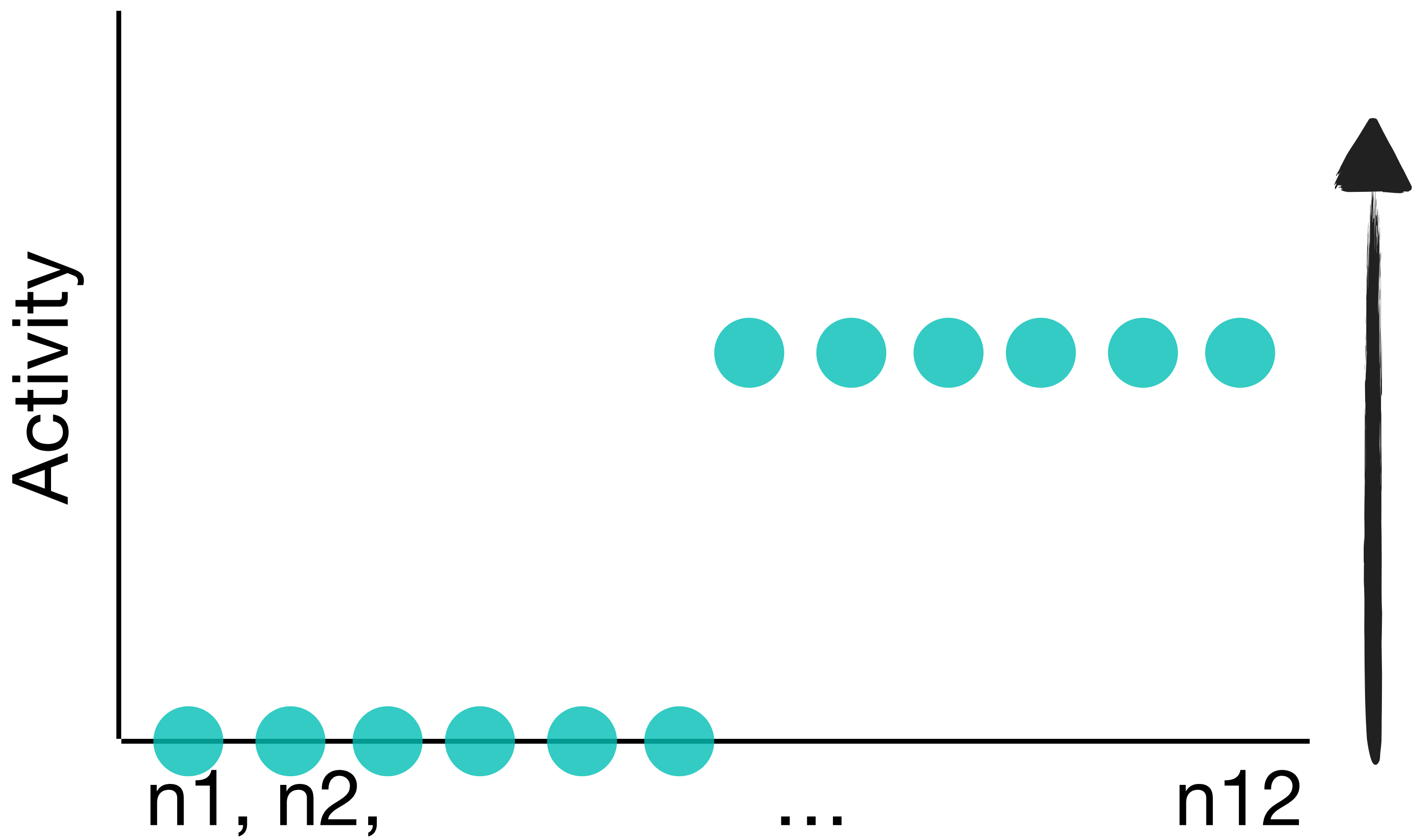
n1, n2, … n6

n7, n8, … n12

Thinking… it's important

- What would it mean to reduce the dimensionality of the 12 neurons?
- Simple ideal example:

All you need is 1D:



n1, n2, … n6 ↑
n7, n8, … n12 ↓

And

n1, n2, … n6 ↓
n7, n8, … n12 ↑

Thinking… it's important

- What would it mean to reduce the dimensionality of the 12 neurons?
- Simple ideal example:

n1, n2, … n6 ↑

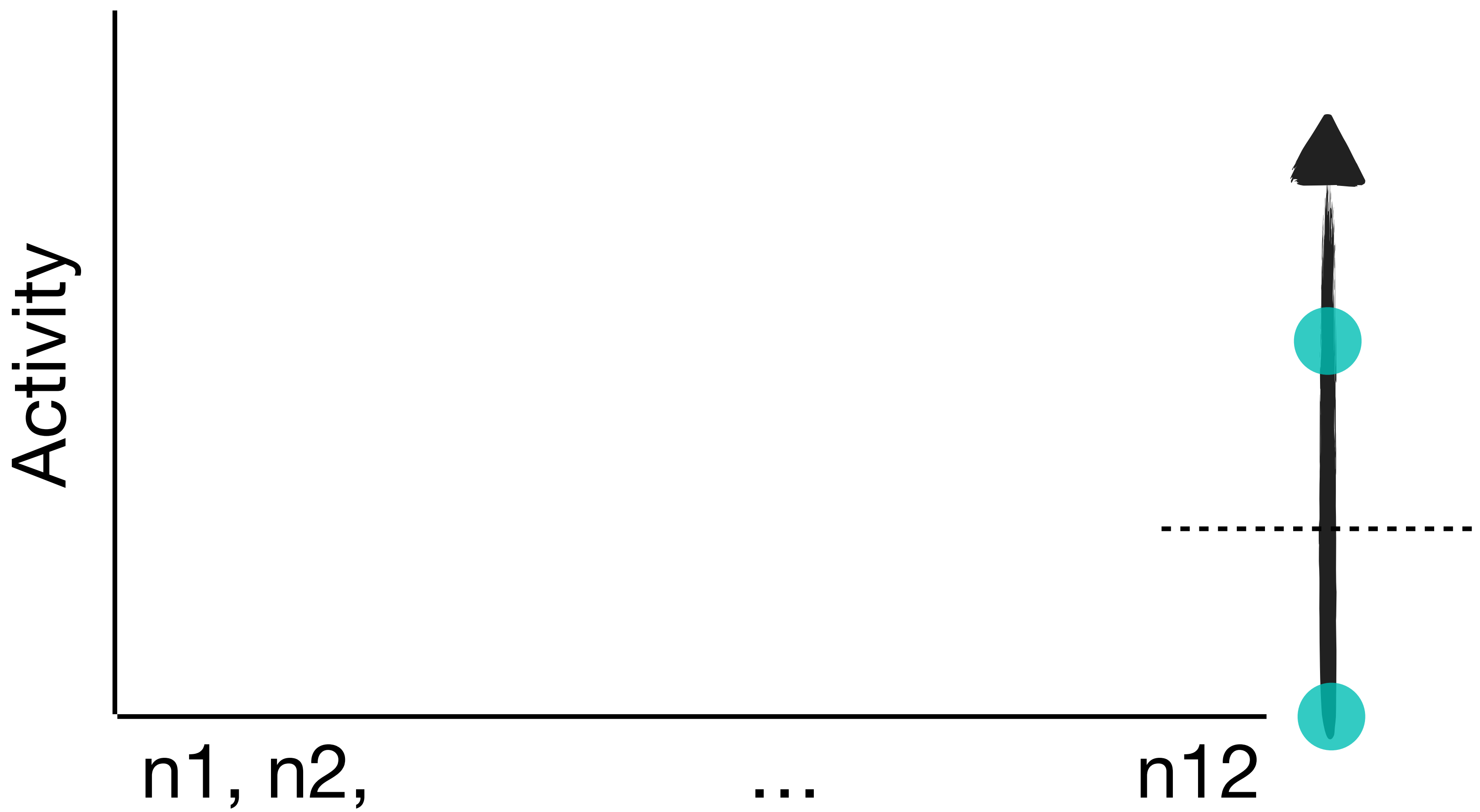
n7, n8, … n12 ↓

All you need is 1D:

And

n1, n2, … n6 ↓

n7, n8, … n12 ↑



Thinking… it's important

- What would it mean to reduce the dimensionality of the 12 neurons?
- Simple ideal example:

We can say…

- Neurons 1:6 are highly correlated
- Neurons 7:12 are highly correlated
- Neurons 1:6 are not at all correlated with neurons 7:12

n1, n2, … n6 $\uparrow$

n7, n8, … n12 $\downarrow$

And

n1, n2, … n6 $\downarrow$

n7, n8, … n12 $\uparrow$

Thinking… it's important

- What would it mean to reduce the dimensionality of the 12 neurons?
- Simple ideal example:

n1, n2, … n6 ↑

n7, n8, … n12 ↓

And

n1, n2, … n6 ↓

n7, n8, … n12 ↑

We can say…

- Neurons 1:6 are highly correlated
- Neurons 7:12 are highly correlated
- Neurons 1:6 are not at all correlated with neurons 7:12

How does each neuron correlate with all the other 11 neurons?

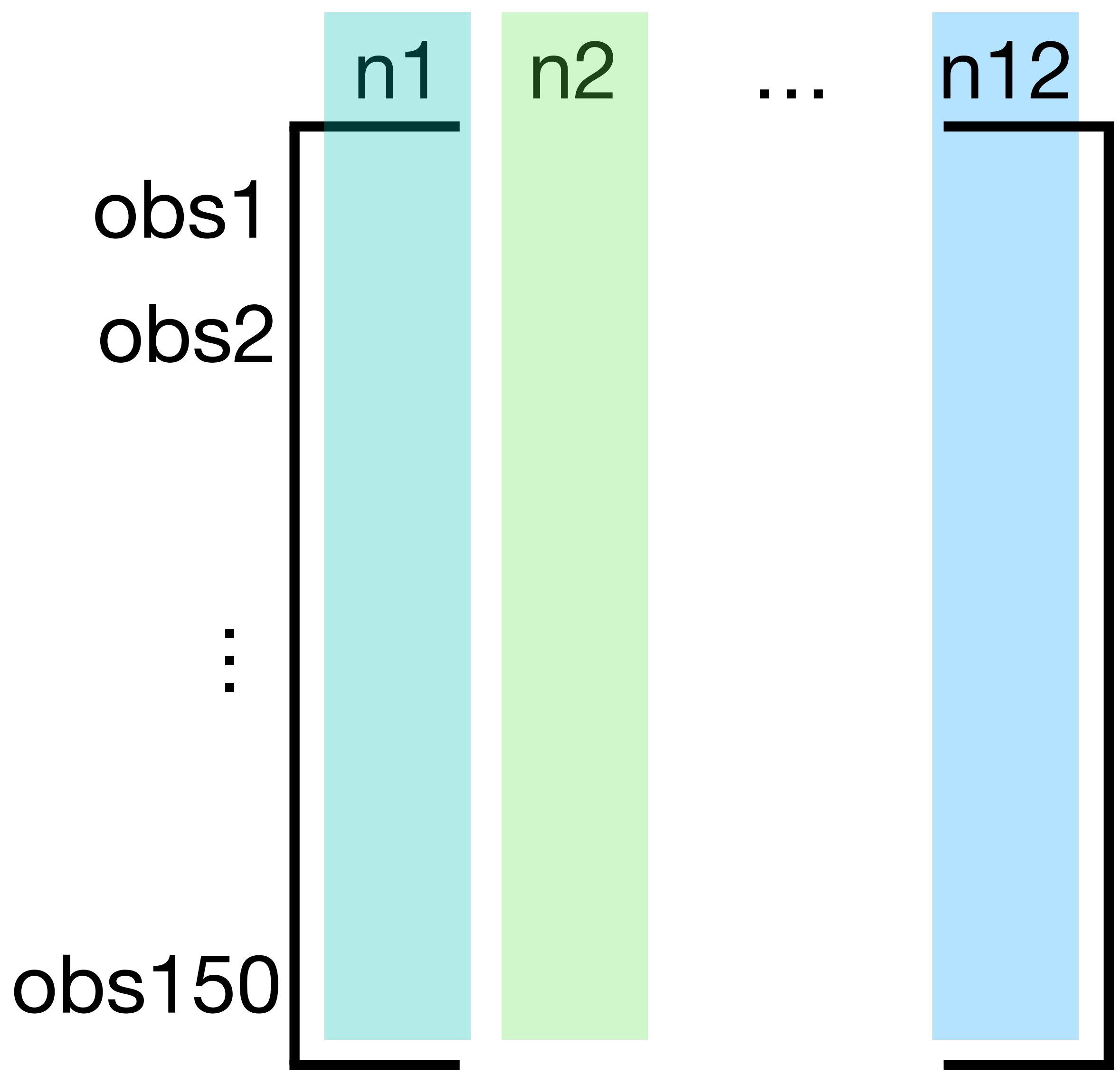
Correlation across neurons:

- 12x12 table
- How can we write this in linear algebra form?

	A	B	C	D	E	F	G	H	I	J	K	L	M
1		**n1**	**n2**	**n3**	**n4**	**n5**	**n6**	**n7**	**n8**	**n9**	**n10**	**n11**	**n12**
2	**n1**	n1n1	n1n2										n1n12
3	**n2**	n2n1	n2n2										n2n12
4	**n3**												
5	**n4**												
6	**n5**												
7	**n6**												
8	**n7**												
9	**n8**												
10	**n9**												
11	**n10**												
12	**n11**												
13	**n12**	n12n1	n12n2										n12n12

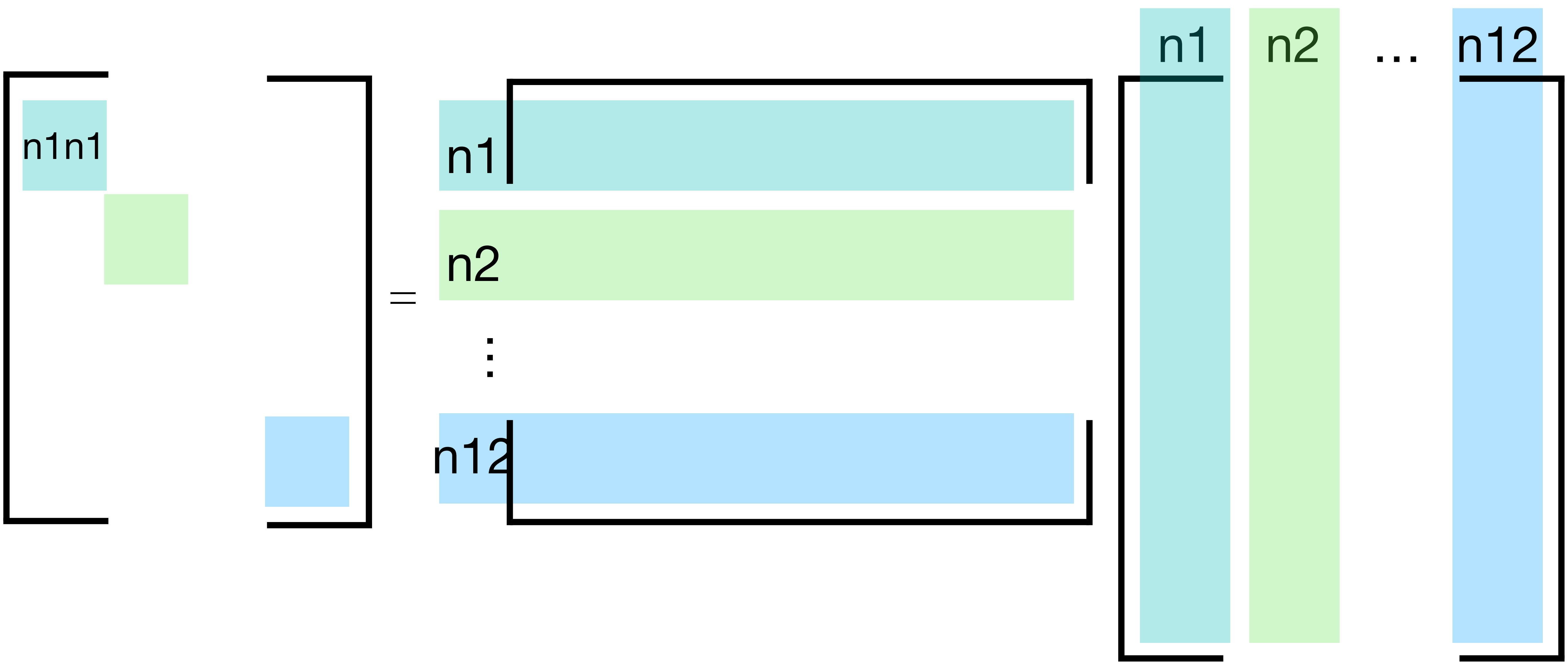
Correlation across neurons:

- 12x12 table
- How can we write this in linear algebra form?



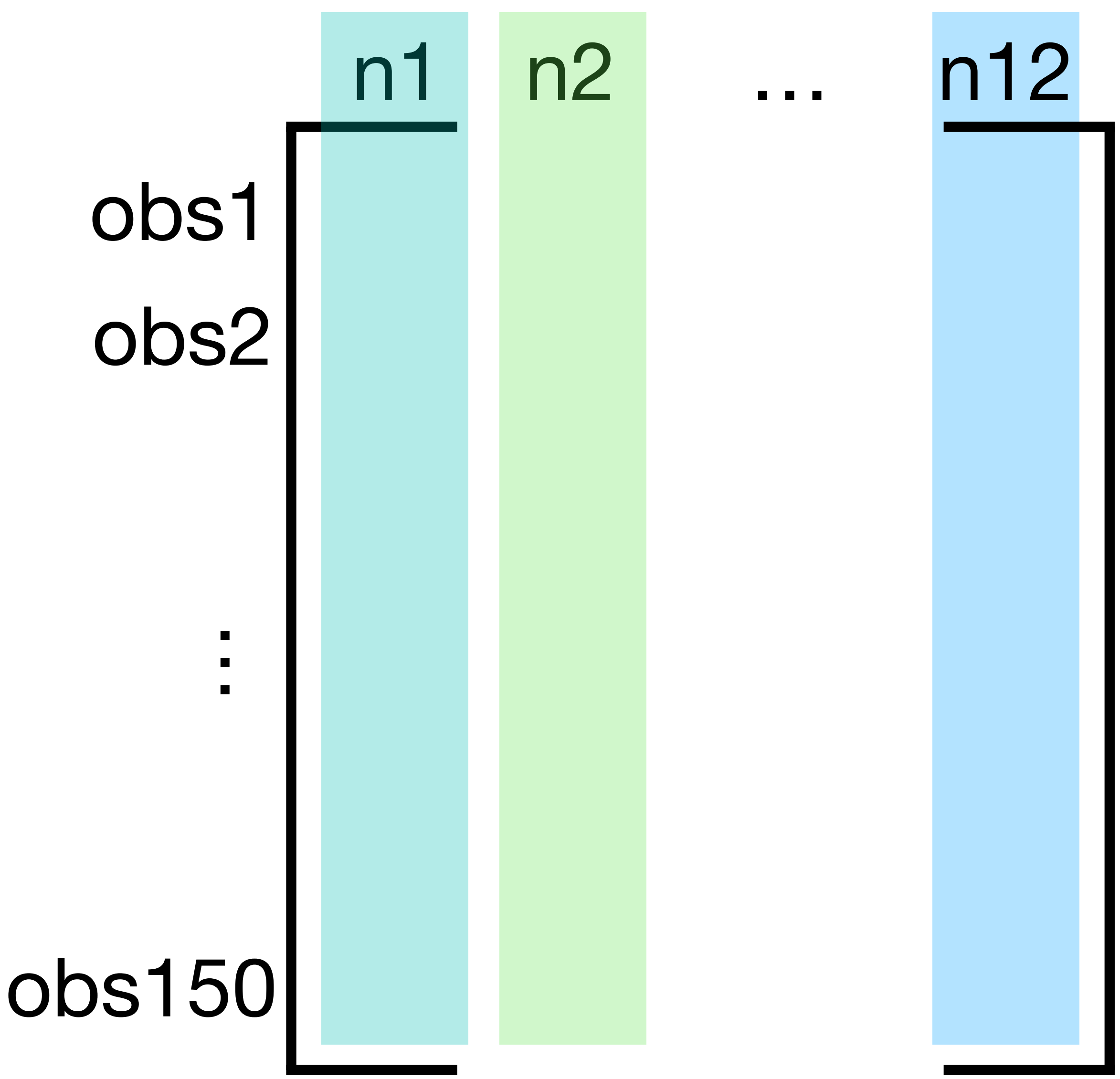
Correlation across neurons:

- 12x12 table
- How can we write this in linear algebra form?
- MtM — covariance matrix of M



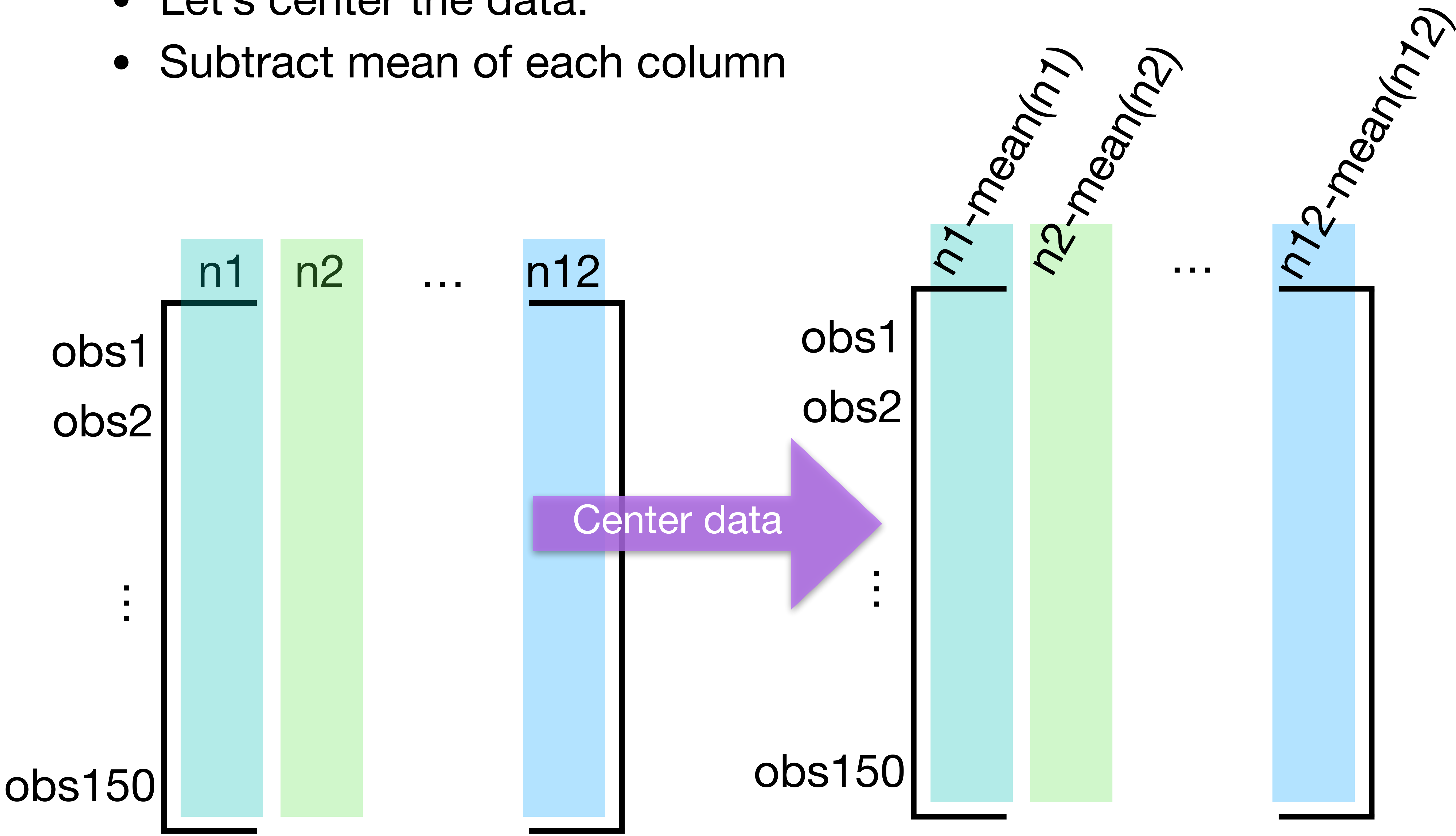
Important first step!

- What if the the baseline of a neuron is higher?
- Do we care about baseline?
- Probably not… here we care about the correlation across neurons



Important first step!

- Let’s center the data:
- Subtract mean of each column



What would you do if you want to cluster the 150 observations?

- How would you center the data?
- How would you get the covariance matrix?

Back to neurons

- Want a lower dimensional representation:
  - How many dimensions? Which dimensions?
- Ready for eigen decomposition of Mt*M
- eig(MtM) = V * lambda * Vt
- Where V is an orthogonal matrix, Vt is the transpose of V
- Lambda is a diagonal matrix

$$\begin{array}{cccc} & V & \Lambda & Vt \\ MtM = & \begin{bmatrix} \text{Orthogonal} \end{bmatrix} & \begin{bmatrix} \text{Diagonal} \end{bmatrix} & \begin{bmatrix} \text{Orthogonal} \end{bmatrix} \end{array}$$

Back to neurons

- Want a lower dimensional representation:
  - How many dimensions? Which dimensions?
- Ready for eigen decomposition of Mt*M
- eig(MtM) = V * lambda * Vt
- Where V is a … pose of V
- Lambda is a …

This must look familiar…

$$MtM = \underbrace{V}_{\text{Orthogonal}} \; \underbrace{\Lambda}_{\text{Diagonal}} \; \underbrace{Vt}_{\text{Orthogonal}}$$

SVD vs. Eigen decomp

- Clear similarities (decomposing a matrix into 3, 2 ortho 1 diagonal)
- SVD can decompose any matrix — Eigen can't
  - Matrix must be square for Eigen
  - Eigen decomposition is a special case of SVD

$$MtM = V \quad \Lambda \quad Vt$$

Orthogonal Diagonal Orthogonal

$$M = U \quad S \quad Vt$$

SVD vs. Eigen decomp

- You can get to Eigen decomposition of Mt*M through the SVD of M:

Define symmetric matrix:

$$
\begin{aligned}
C &= M^T M \\
&= (USV^T)^T (USV^T) \\
&= VS^T U^T USV^T \\
&= V(S^T S)V^T
\end{aligned}
$$

$$
MtM \quad = \quad \begin{array}{c} V \\ \left[\ \text{Orthogonal}\ \right] \end{array} \begin{array}{c} \Lambda \\ \left[\ \text{Diagonal}\ \right] \end{array} \begin{array}{c} Vt \\ \left[\ \text{Orthogonal}\ \right] \end{array}
$$

SVD vs. Eigen decomp

- Columns of V are the Eigen vectors of Mt*M
- What about the S matrix from svd(M)? How does it relate to Eigen decomposition?

Define symmetric matrix:

$$\begin{aligned} C &= M^T M \\ &= (USV^T)^T (USV^T) \\ &= VS^T U^T USV^T \\ &= V(S^T S)V^T \end{aligned}$$

- “rotate, stretch, rotate back”
- matrix C “summarizes” the shape of the data

Eigen decomp

- Eigen vector 1 (e1) gets scaled by eigenvalue s1^2

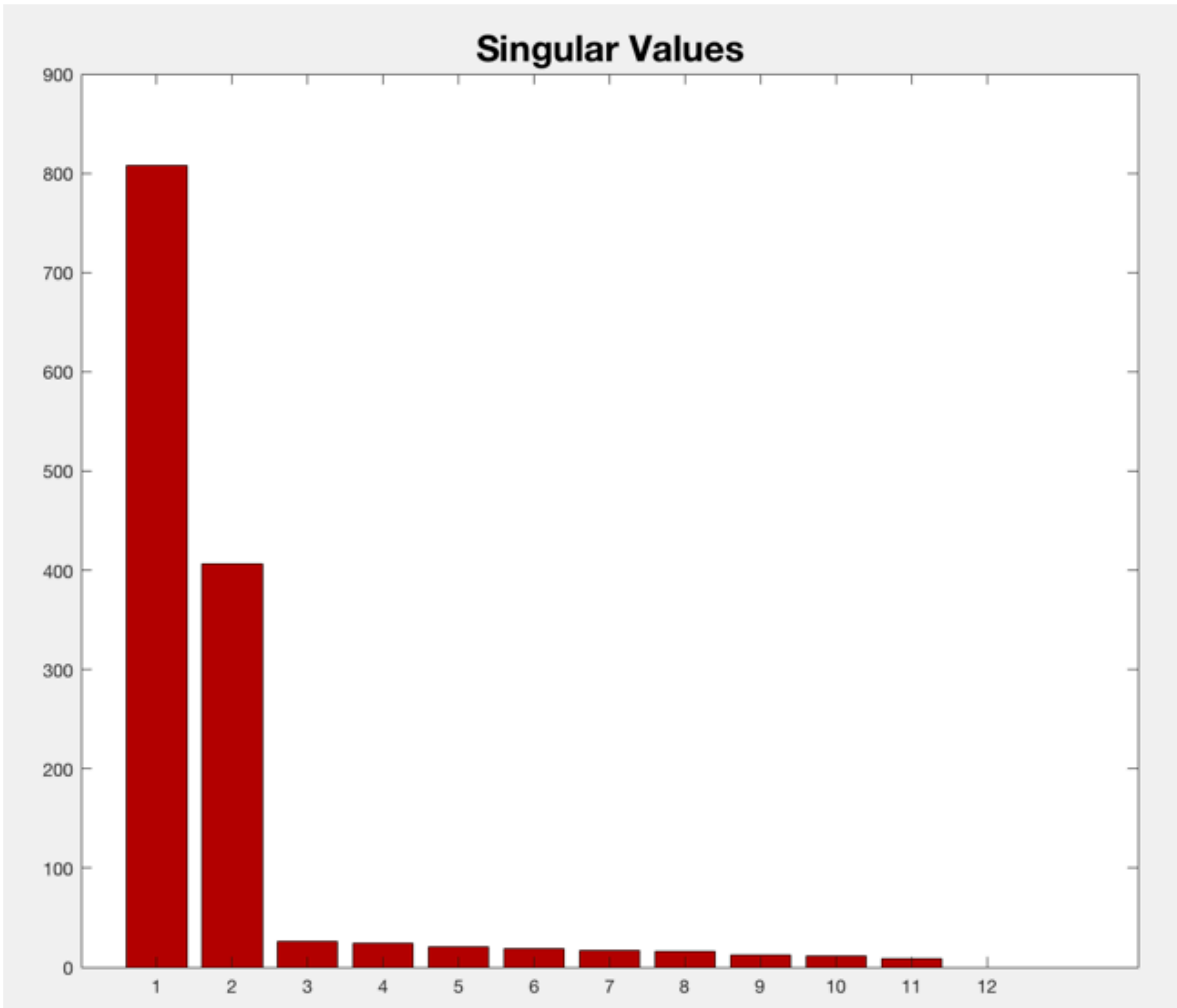
$\vec{v}_k$, the *k*th columns of V,
is called an *eigenvector* of C:

$$\begin{aligned} C\vec{v}_k &= V(S^T S)V^T\vec{v}_k \\ &= V(S^T S)\hat{e}_k \\ &= s_k^2 V\hat{e}_k \\ &= s_k^2\vec{v}_k \end{aligned}$$

- output is a rescaled copy of input
- scale factor s_k^2 is called the *eigenvalue* associated with $\vec{v}_k$

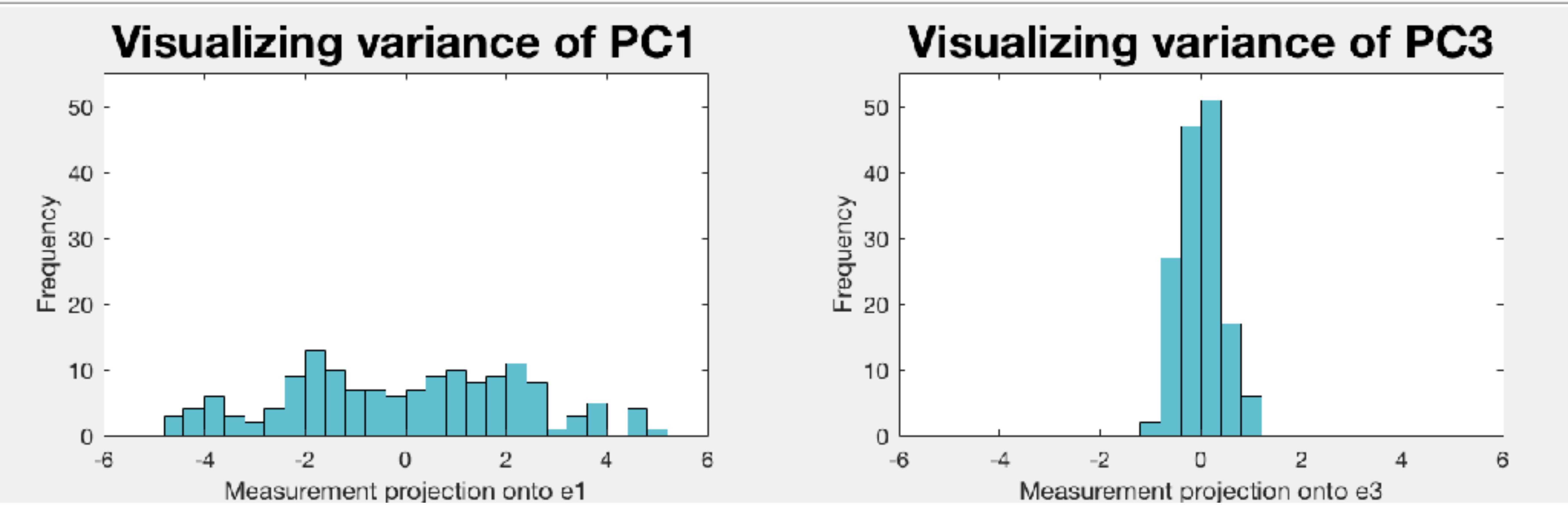
Reducing dimensionality

- What reduced dimensionality do we want?



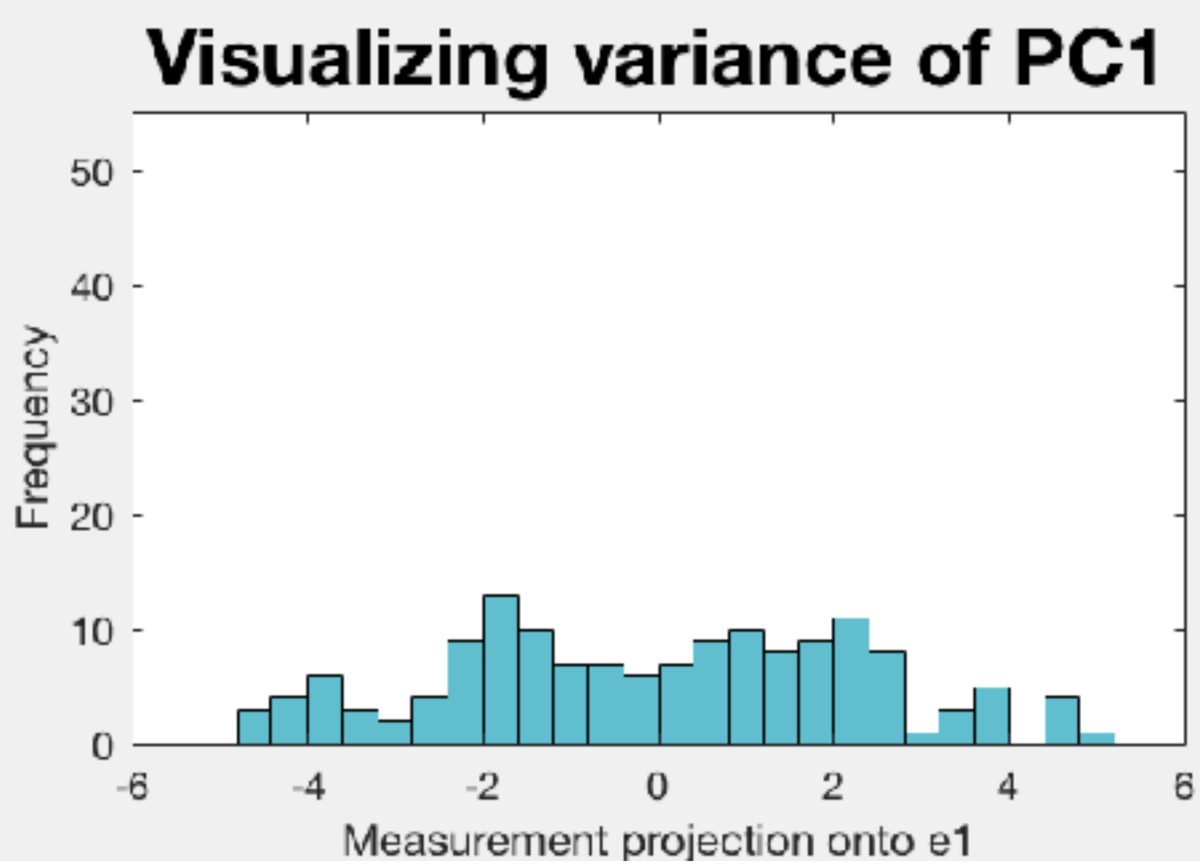
Reducing dimensionality

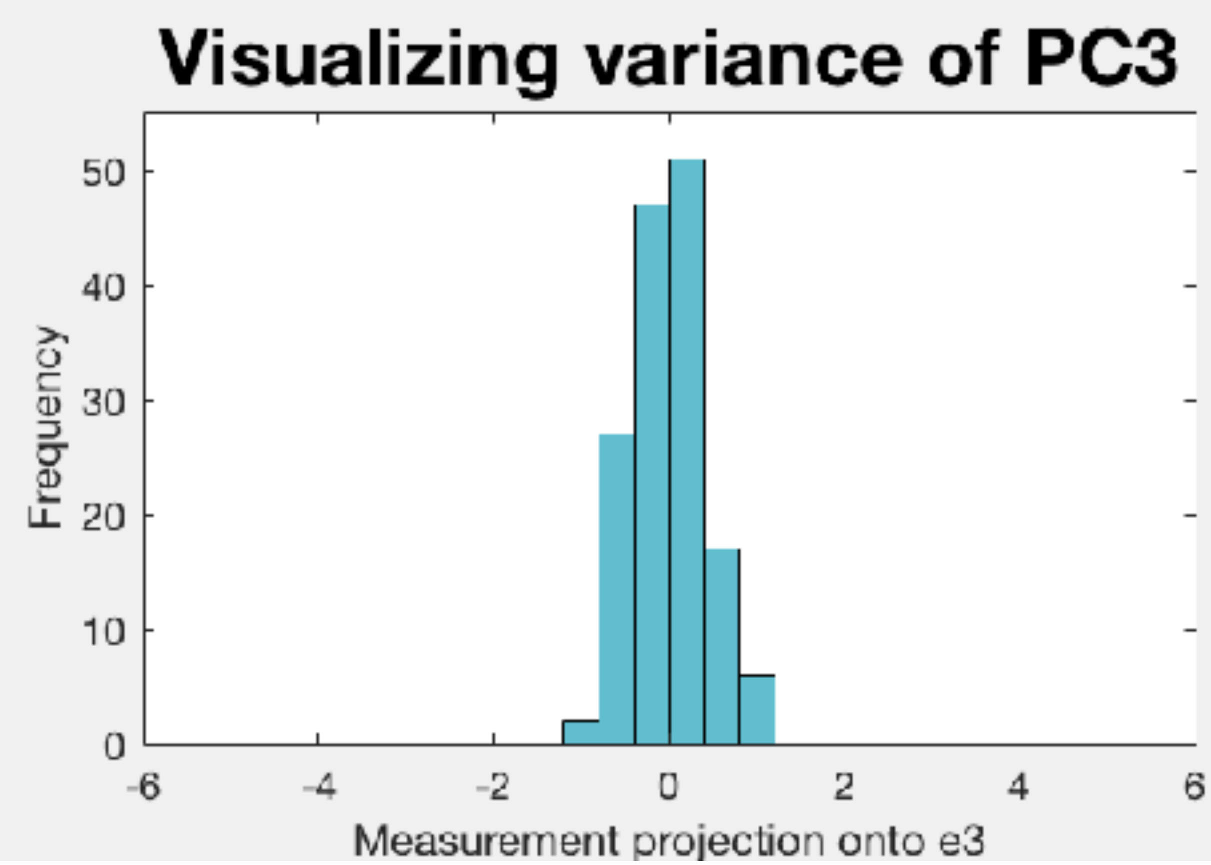
- Project data onto e1 — get most variance
- Project data onto e3 (small eigenvalue in this example) — get much less variance.



Reducing dimensionality

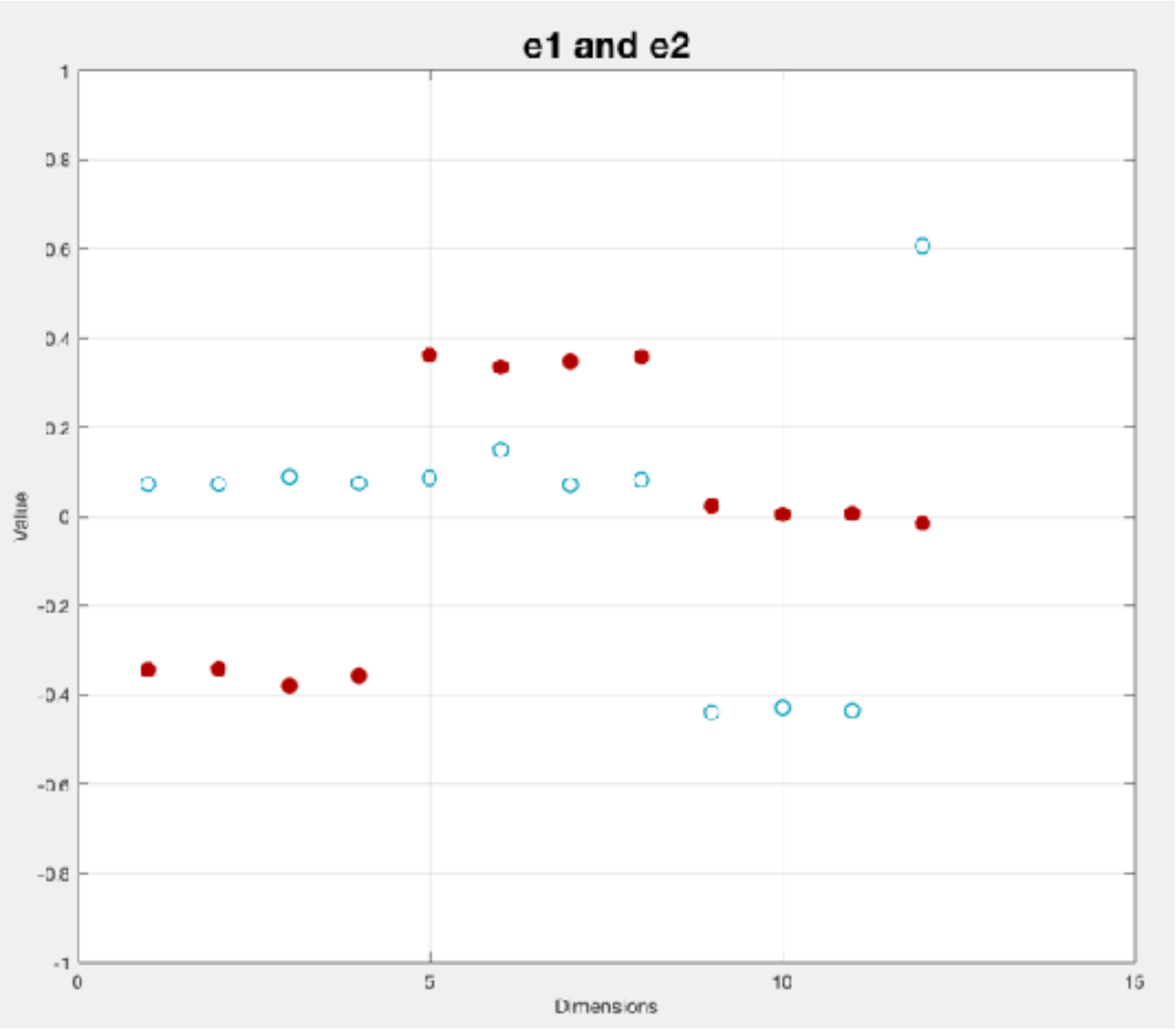
- In this example, because eigenv1 and eigenv2 >> eigenv3 and the others, we can plot our data in 2 dimensions





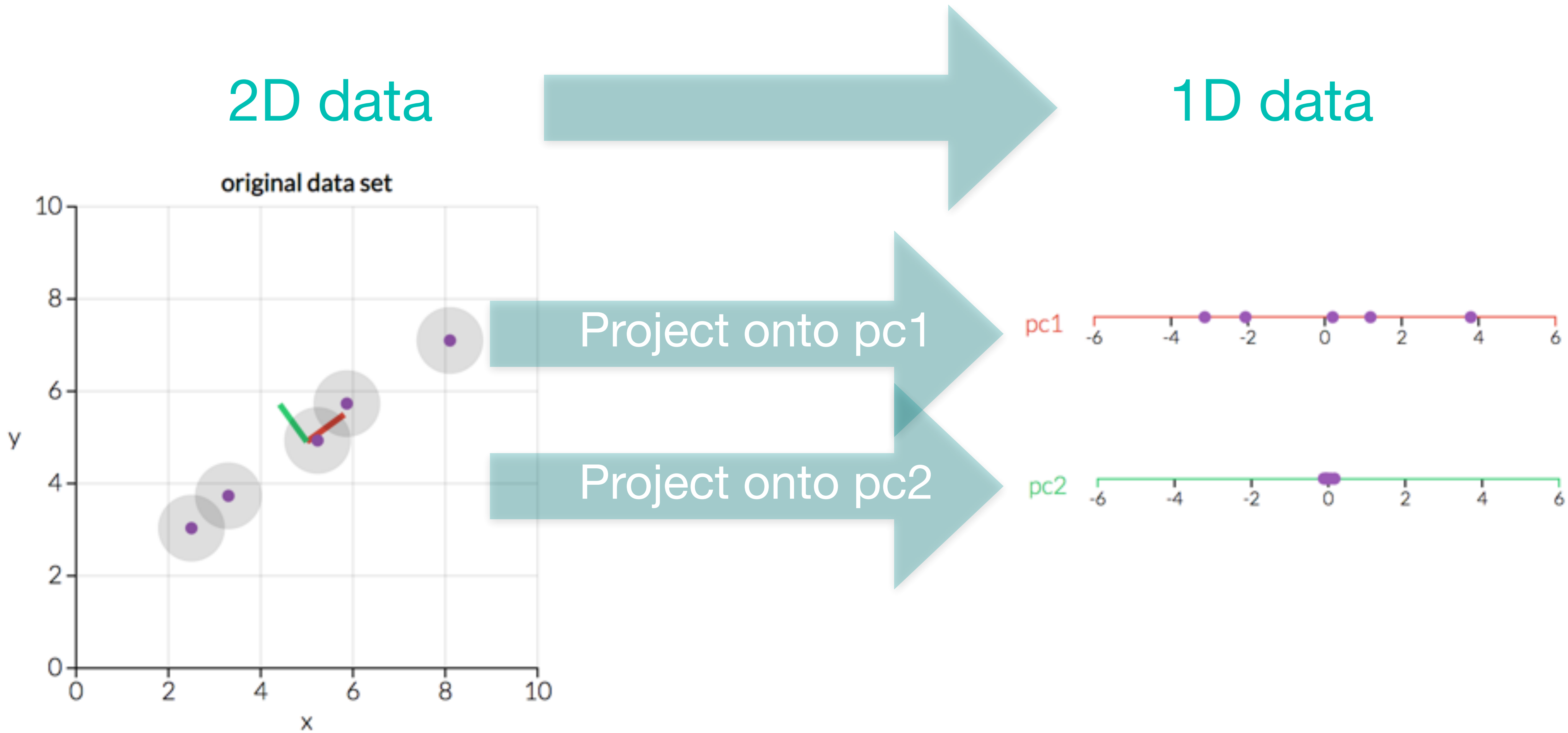
Plotting the Eigenvectors:

- Spike plot of 12d vectors:
- What does each element/index represent?



Reducing dimensionality

- Complicated to work with high dimensional data
- Reducing dimensionality is good
- Example for visualization: 2d —> 1d



Check out the animation! http://setosa.io/ev/principal-component-analysis/

Summary

Data: 150x12 matrix of 12 neurons

1. Think about row and columns
2. What do you want to know?
3. Center data
4. Compute the right covariance matrix
5. Eigen decomposition
6. Visualize S values — choose reduced dimensionality
7. Answer questions about your data:
    1. Plot it in the reduced dimensionality — think about what this means
    2. Plot eigenvectors — think about what they mean

Happy Friday — the weekend is so close!