ICA = PP

ICA = PP

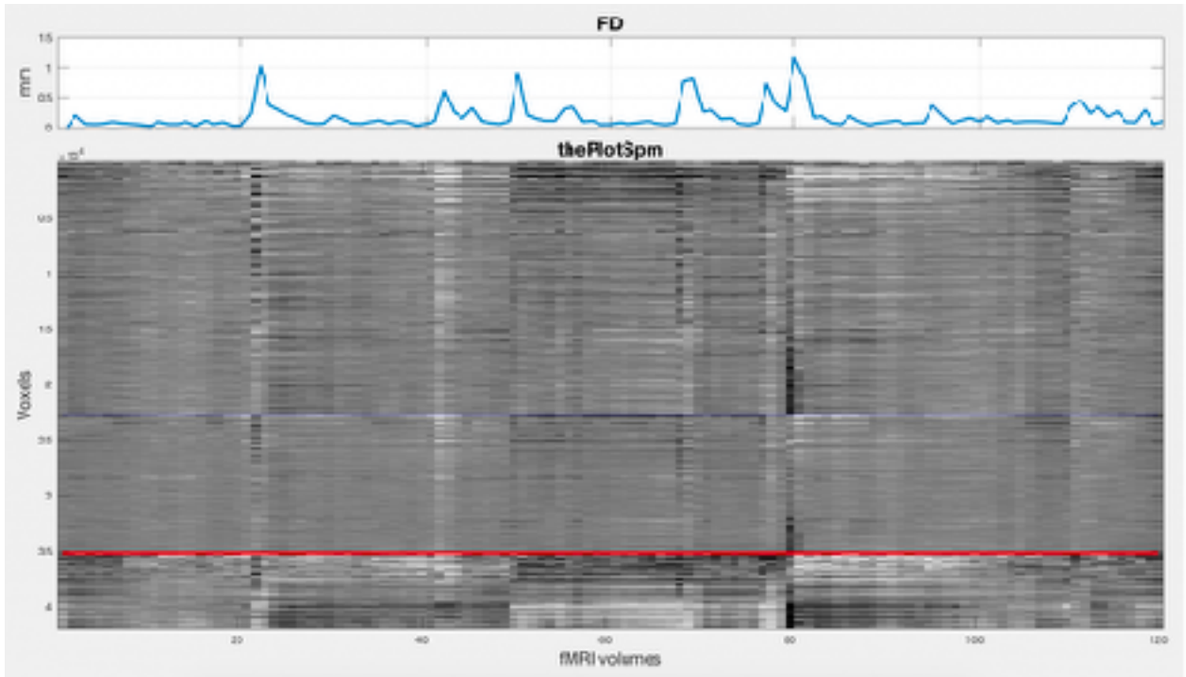
***M.C. JONES and ROBIN SIBSON**. What is projection Pursuit? Journal of the Royal Statistical Society. 1987*

***Aapo Hyvärinen**. Fast and Robust Fixed-Point Algorithms for Independent Component Analysis. IEEE TRANSACTIONS ON NEURAL NETWORKS, 1999*

How to make sense of a table of data?

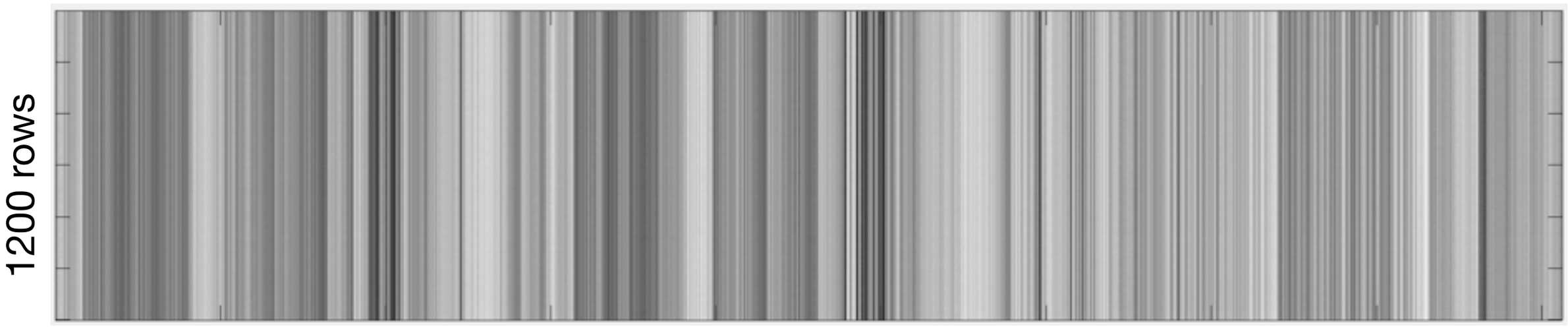
Images

FMRI data



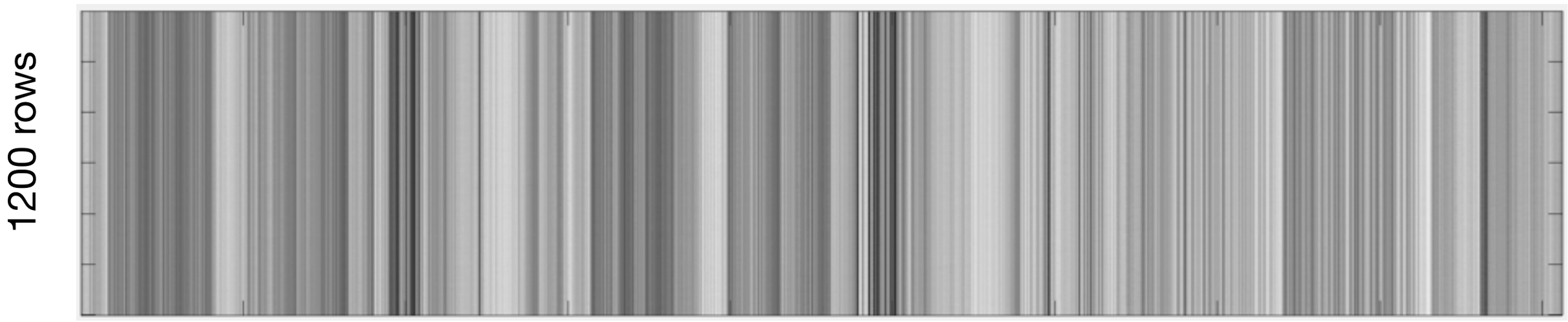
Jonathan Power “The Plot”

What do I do with this data?

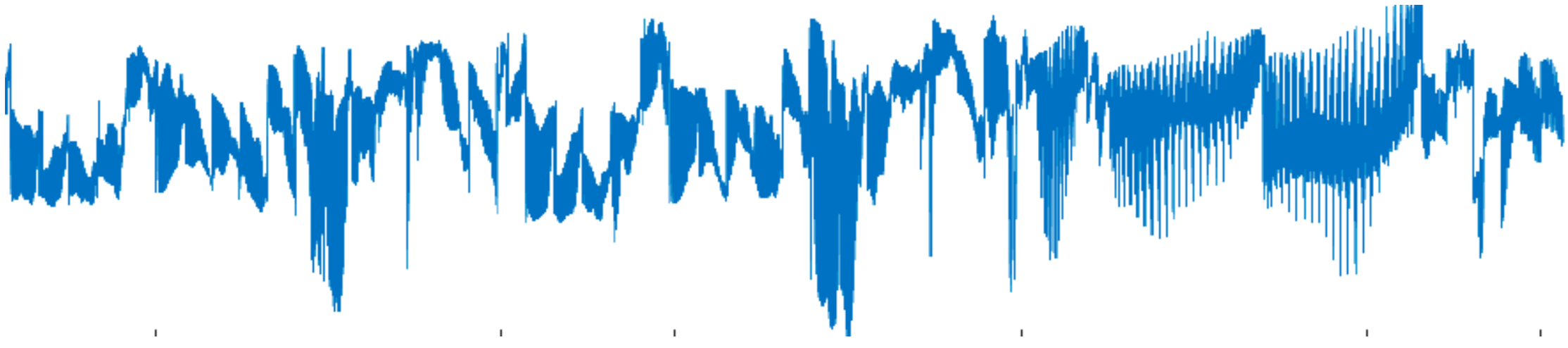


91282 columns

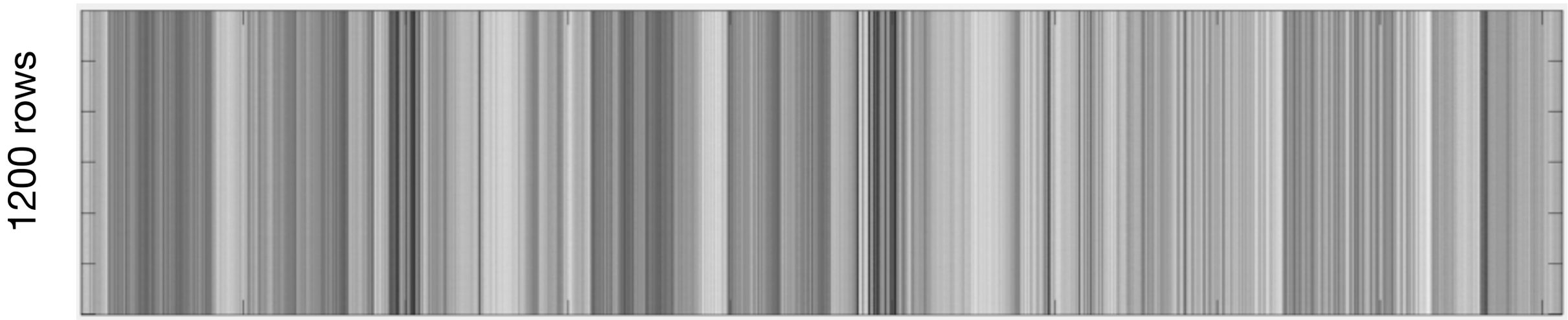
Take the mean?

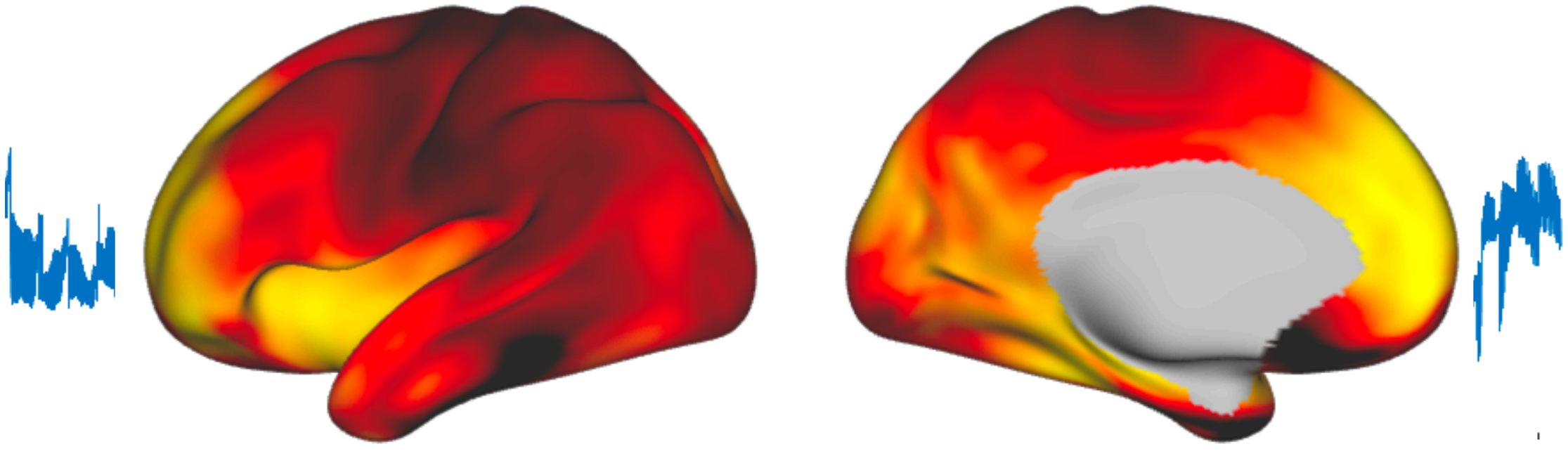


91282 columns

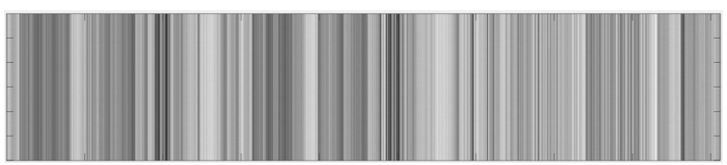
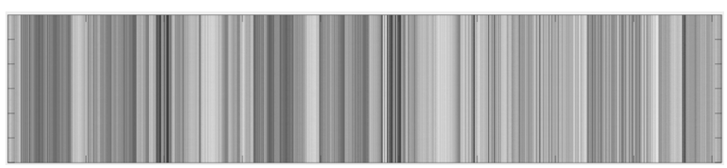


Take the mean?

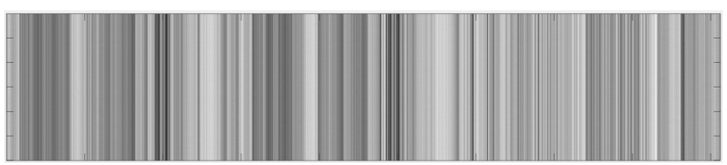
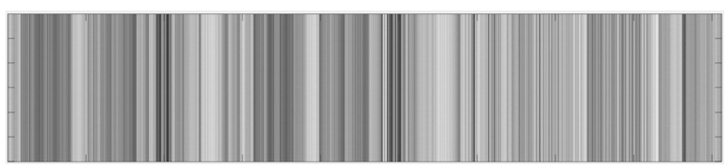




What have we done?

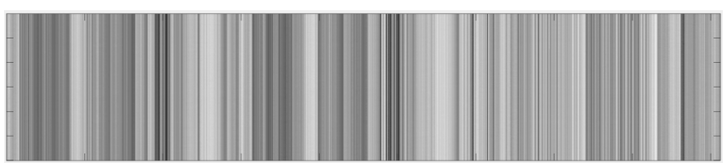
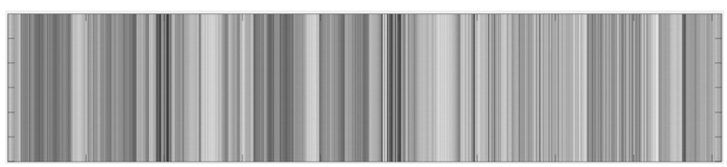
mean() = $\begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix}^T /n$ 

What have we done?

mean() = $\begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix}^T / n$ 

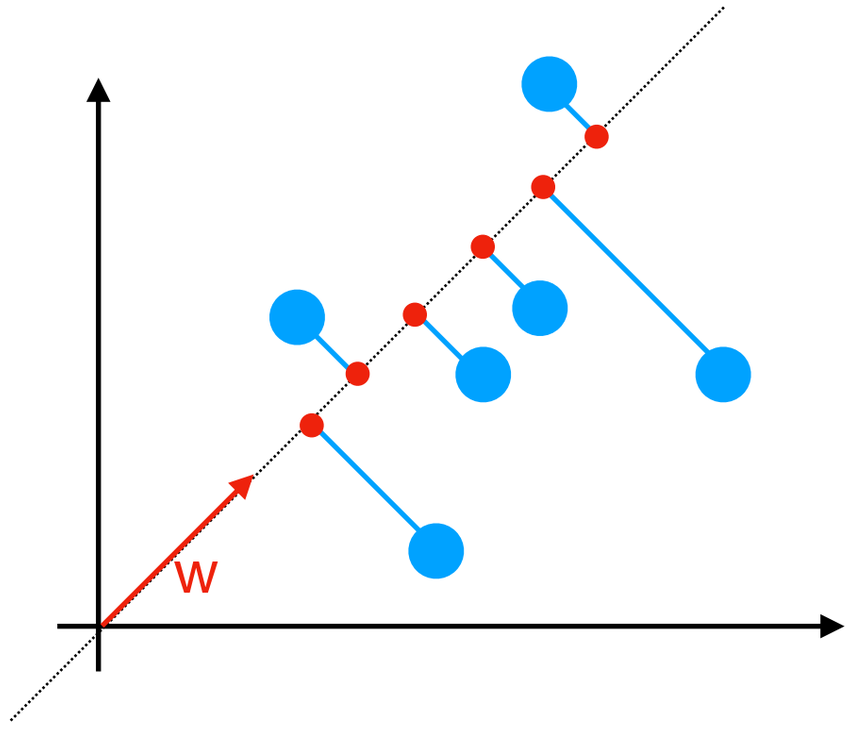
= $w^T X$ Orthogonal projection of X onto w

What have we done?

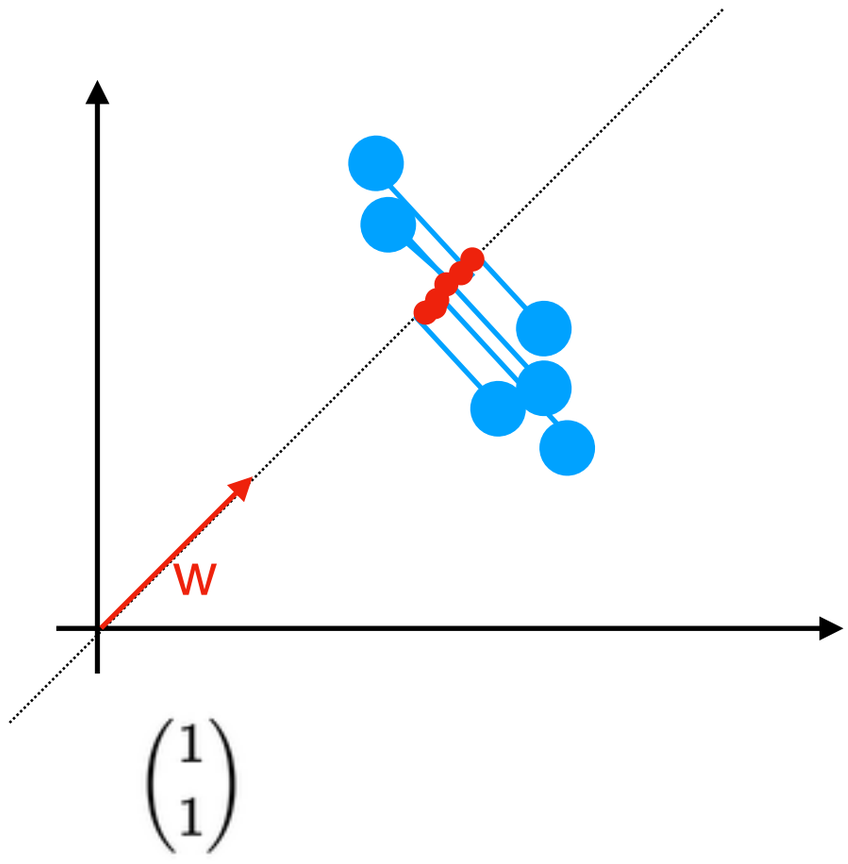
mean() = $\begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix}^T / n$ 

= $w^T X$ Orthogonal projection of X onto w

if X had 2 rows
and 6 columns:

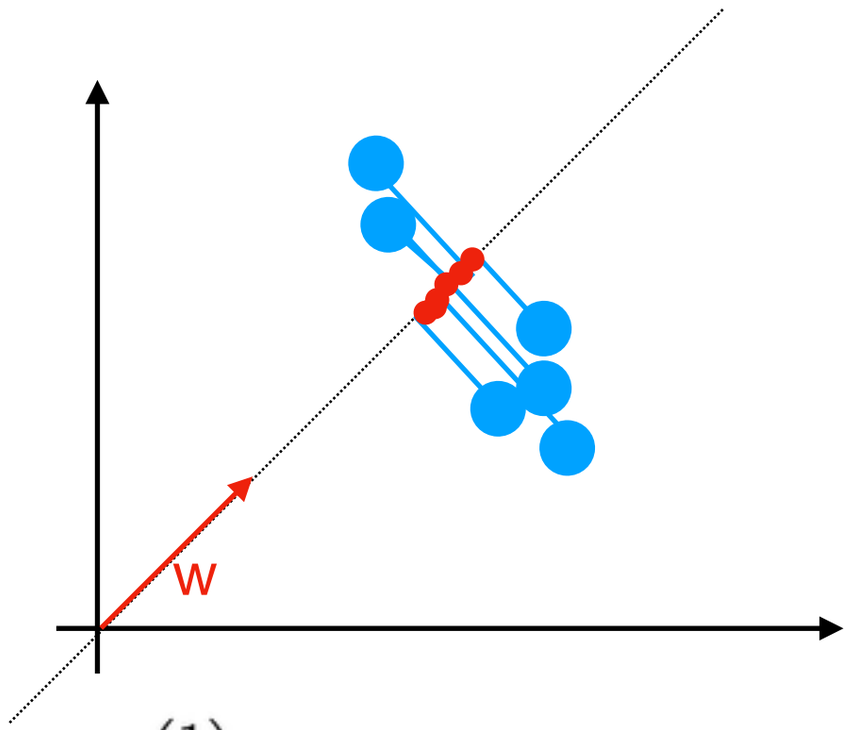


There is nothing special about $\begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix}$

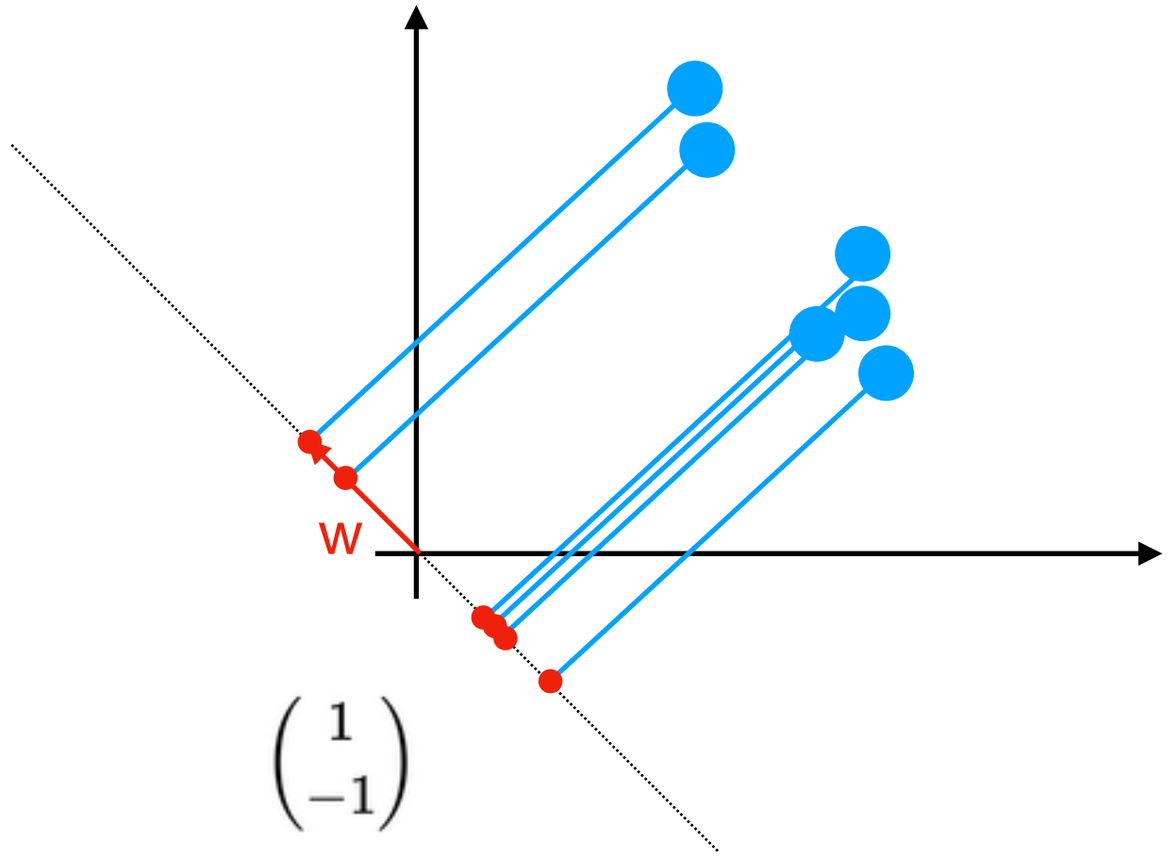


$\begin{pmatrix} 1 \\ 1 \end{pmatrix}$

There is nothing special about $\begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix}$



$\begin{pmatrix} 1 \\ 1 \end{pmatrix}$



$\begin{pmatrix} 1 \\ -1 \end{pmatrix}$

How do we find w?

- Solution: project onto all the direction and look at the histogram for some interesting features

Plan of talk:

- Defining "interestingness"
- PP
- ICA

Notation

- Data is a matrix X (dimensions: features x samples)
- Direction vector is w (features x 1)
- If we want more that one vector then it is W (features x p)

Interestingness

$$\mathrm{I}\left(\mathrm{w}^{T}\mathrm{X}\right)$$

some function TBD

1D version of X

Interestingness

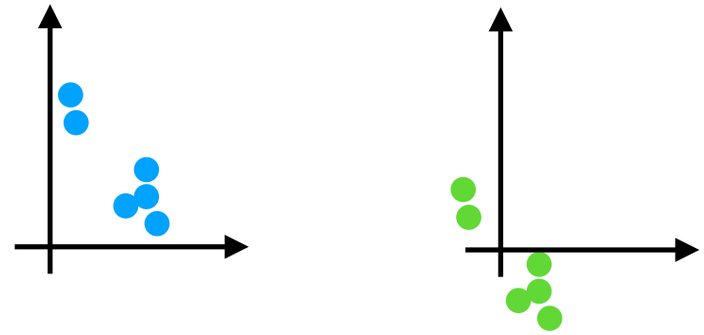
$$\mathrm{I}\left(\mathrm{w}^T\mathrm{X}\right)$$

some function TBD

1D version of X

- Location of data should not matter --> centering

$$\mathrm{X}_c = \mathrm{X}(\mathbf{I} - \mathbf{1}\mathbf{1}^T/\mathrm{n})$$



Interestingness

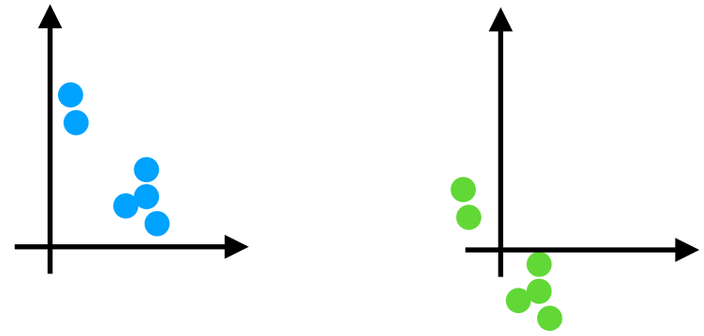
$$\mathrm{I}\left(\mathbf{w}^T\mathbf{X}\right)$$

some function TBD

1D version of X

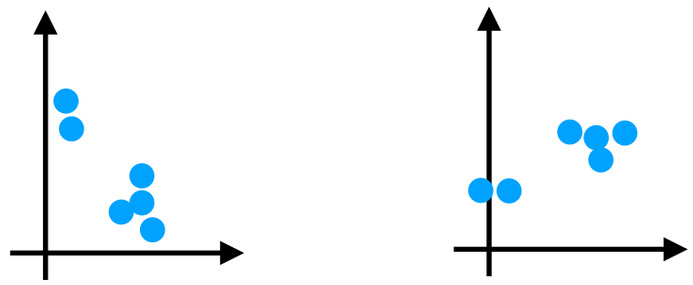
- Location of data should not matter --> centering

$$\mathbf{X}_c = \mathbf{X}(\mathbf{I} - \mathbf{1}\mathbf{1}^T/\mathrm{n})$$



- Orientation of data should not matter --> Ok.

$$(\mathbf{R}\mathbf{w})^T\mathbf{R}\mathbf{X}$$



Interestingness

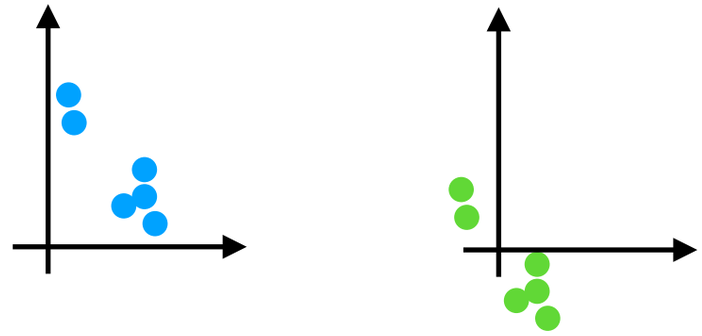
$$\mathrm{I}\left(\mathbf{w}^T\mathbf{X}\right)$$

some function TBD

1D version of X

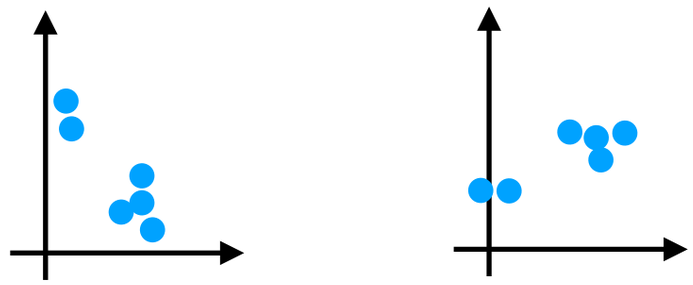
- Location of data should not matter --> centering

$$\mathbf{X}_c = \mathbf{X}(\mathbf{I} - \mathbf{1}\mathbf{1}^T/\mathrm{n})$$



- Orientation of data should not matter --> Ok.

$$(\mathbf{R}\mathbf{w})^T\mathbf{R}\mathbf{X}$$



- Variance?

Maximising the variance

$$\max \text{ Variance}(\mathbf{w}^T\mathbf{X}_c) = E[(\mathbf{w}^T\mathbf{X}_c)^2]$$

Maximising the variance

$$\max \text{ Variance}(\mathbf{w}^T\mathbf{X}_c) = E[(\mathbf{w}^T\mathbf{X}_c)^2]$$

$$= 1/n*(\mathbf{w}^T\mathbf{X}_c)(\mathbf{w}^T\mathbf{X}_c)^T$$

Maximising the variance

$$\max \text{ Variance}(\mathbf{w}^T\mathbf{X}_c) = E[(\mathbf{w}^T\mathbf{X}_c)^2]$$

$$= 1/n*(\mathbf{w}^T\mathbf{X}_c)(\mathbf{w}^T\mathbf{X}_c)^T$$

$$= 1/n*\mathbf{w}^T(\mathbf{X}_c\mathbf{X}_c^T)\mathbf{w}$$

Maximising the variance

$$\max \text{ Variance}(\mathbf{w}^T\mathbf{X}_c) = E[(\mathbf{w}^T\mathbf{X}_c)^2]$$

$$= 1/n*(\mathbf{w}^T\mathbf{X}_c)(\mathbf{w}^T\mathbf{X}_c)^T$$

$$= 1/n*\mathbf{w}^T(\mathbf{X}_c\mathbf{X}_c^T)\mathbf{w}$$

↓

1st eigenvector of covariance matrix

PCA!

Maximising the variance

$$\max \text{ Variance}(\mathbf{w}^T\mathbf{X}_c) = E[(\mathbf{w}^T\mathbf{X}_c)^2]$$

$$= 1/n*(\mathbf{w}^T\mathbf{X}_c)(\mathbf{w}^T\mathbf{X}_c)^T$$

$$= 1/n*\mathbf{w}^T(\mathbf{X}_c\mathbf{X}_c^T)\mathbf{w}$$

↓

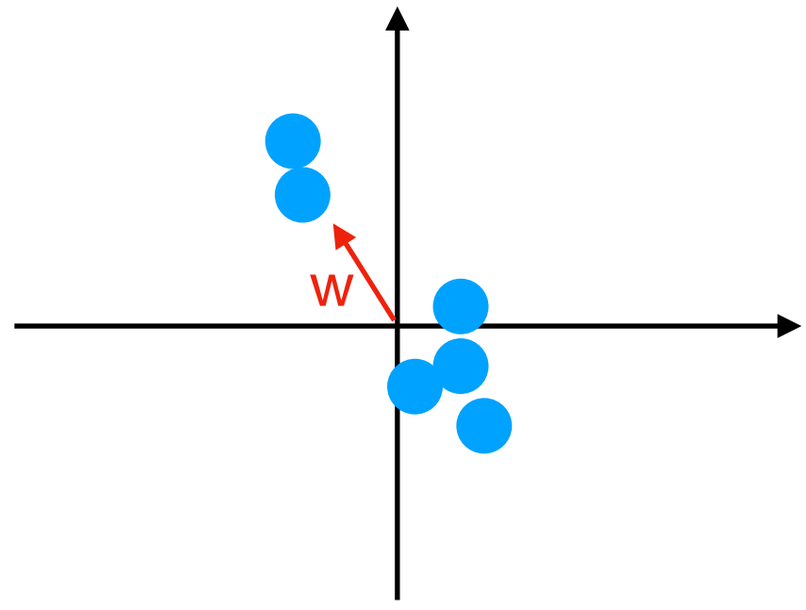
1st eigenvector of covariance matrix

PCA!

(theorem: a symmetric matrix M [like XX^T] has all the eigenvectors and one of them points in the direction of maximum x^TMX)

PCA

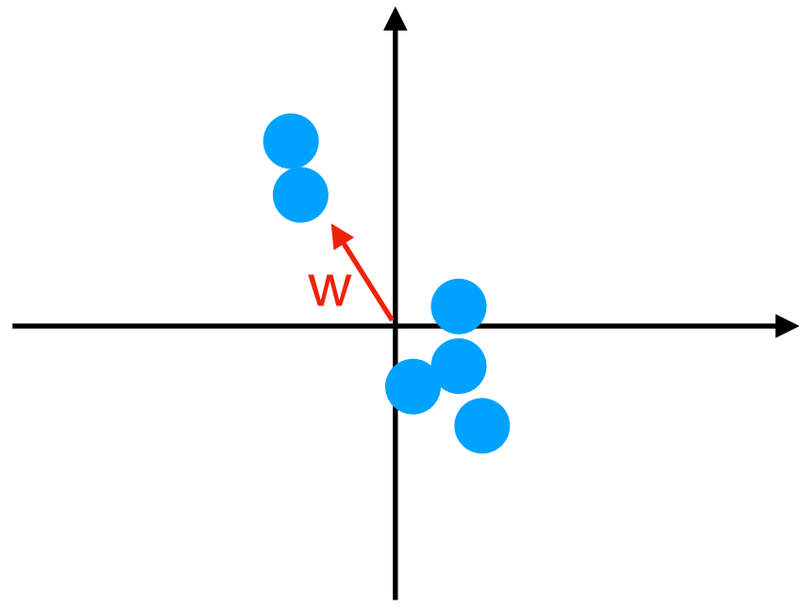
- Capture largest variance
- Incidentally, approximate the data best



PCA

- Capture largest variance
- Incidentally, approximate the data best

$w^T X$ 1D version of X

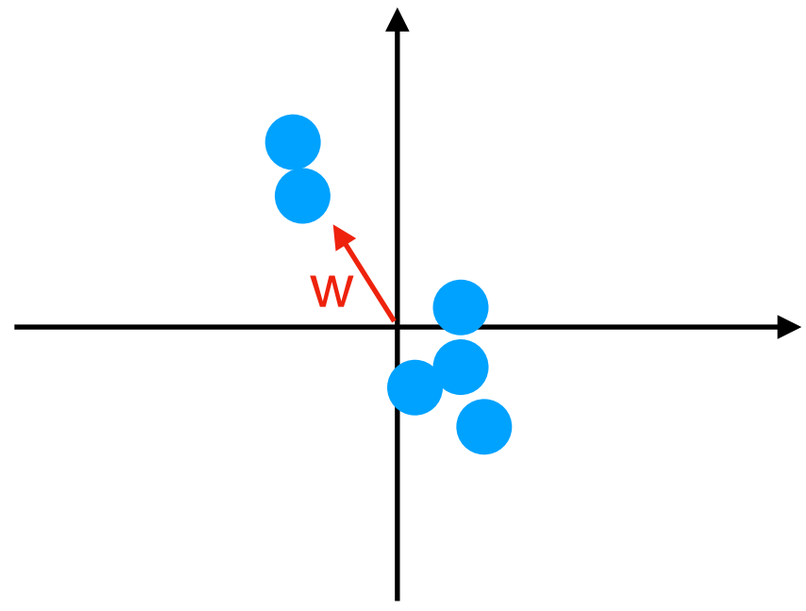


PCA

- Capture largest variance
- Incidentally, approximate the data best

w^TX 1D version of X

ww^TX same dimensions as X



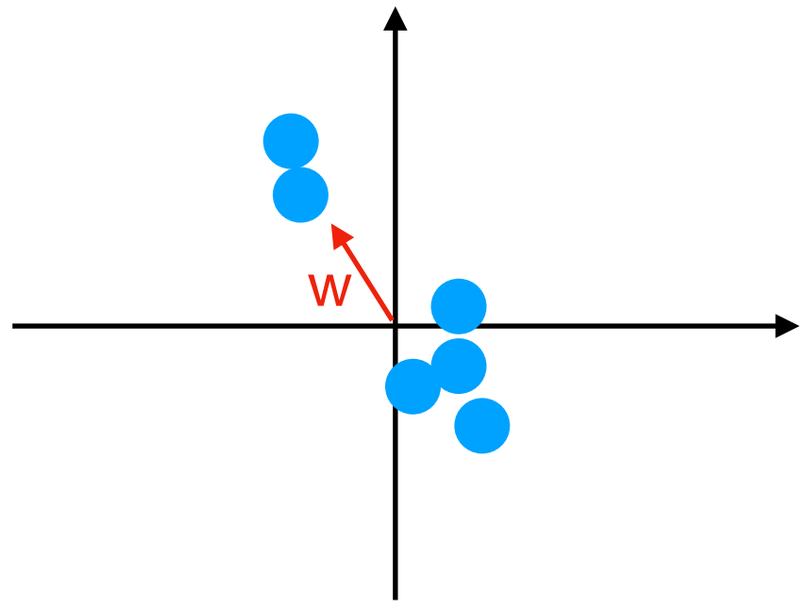
PCA

- Capture largest variance
- Incidentally, approximate the data best

$w^T X$ 1D version of X

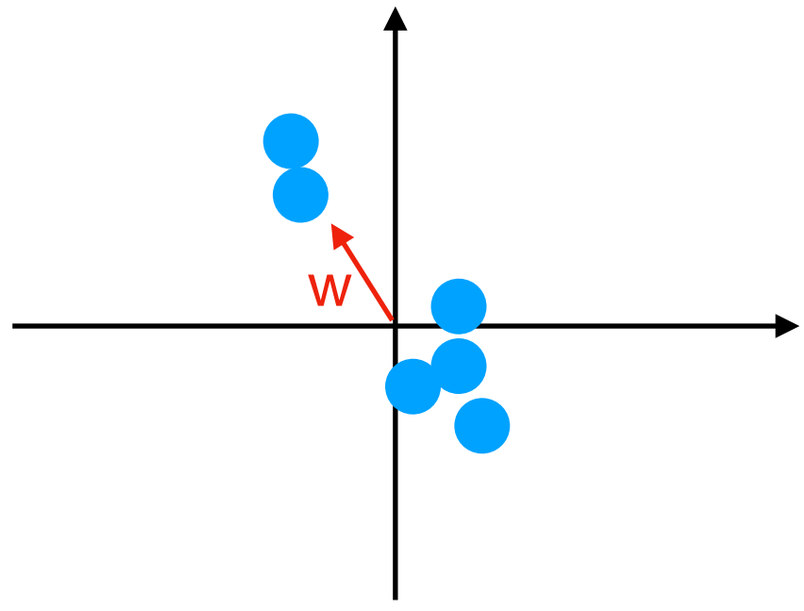
$ww^T X$ same dimensions as X

$|X - ww^T X|^2$ minimize this wrt w?



PCA

- Capture largest variance
- Incidentally, approximate the data best



$w^T X$ 1D version of X

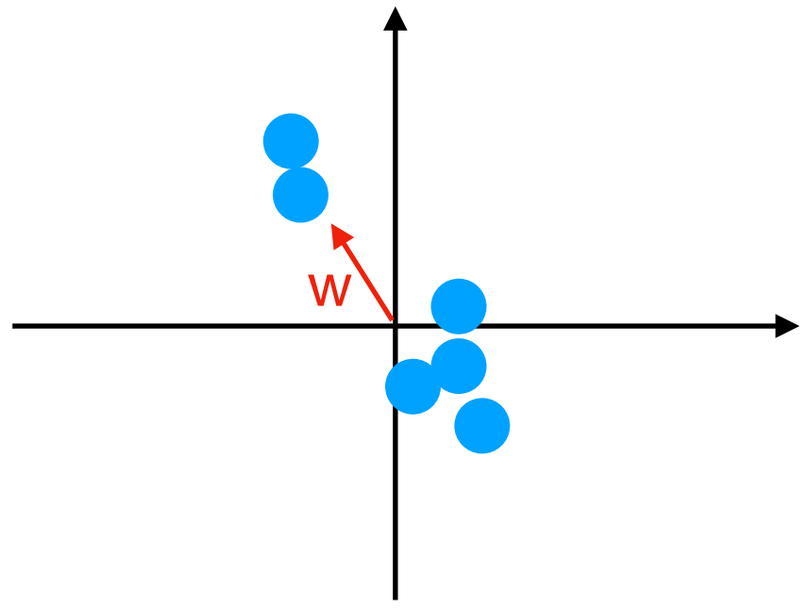
$ww^T X$ same dimensions as X

$|X-ww^T X|^2$ minimize this wrt w?

trace($(X-ww^T X)(X-ww^T X)^T$)

PCA

- Capture largest variance
- Incidentally, approximate the data best



$w^T X$ 1D version of X

$ww^T X$ same dimensions as X

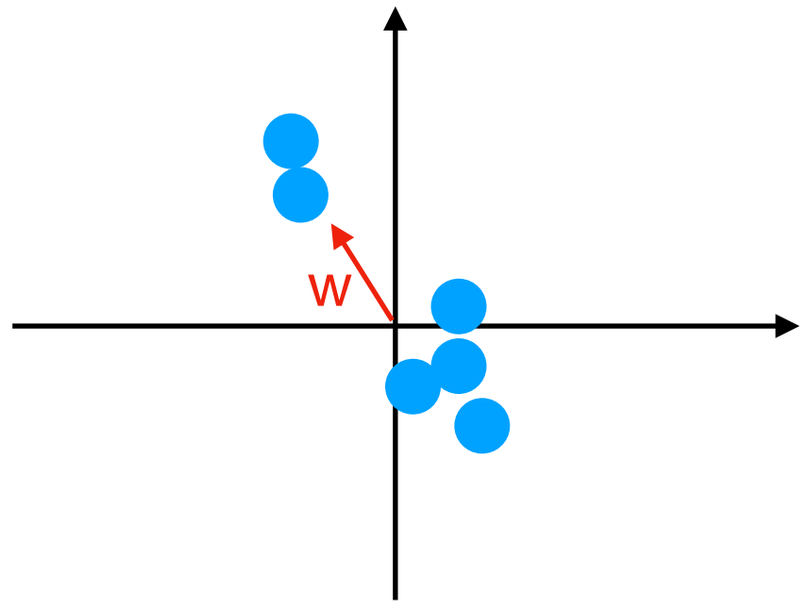
$|X-ww^T X|^2$ minimize this wrt w?

$\text{trace}((X-ww^T X)(X-ww^T X)^T)$

$\text{trace}(-ww^T XX^T \quad -XX^T ww^T \quad +ww^T XX^T ww^T)$

PCA

- Capture largest variance
- Incidentally, approximate the data best



$w^T X$ 1D version of X

$ww^T X$ same dimensions as X

$|X - ww^T X|^2$ minimize this wrt w?

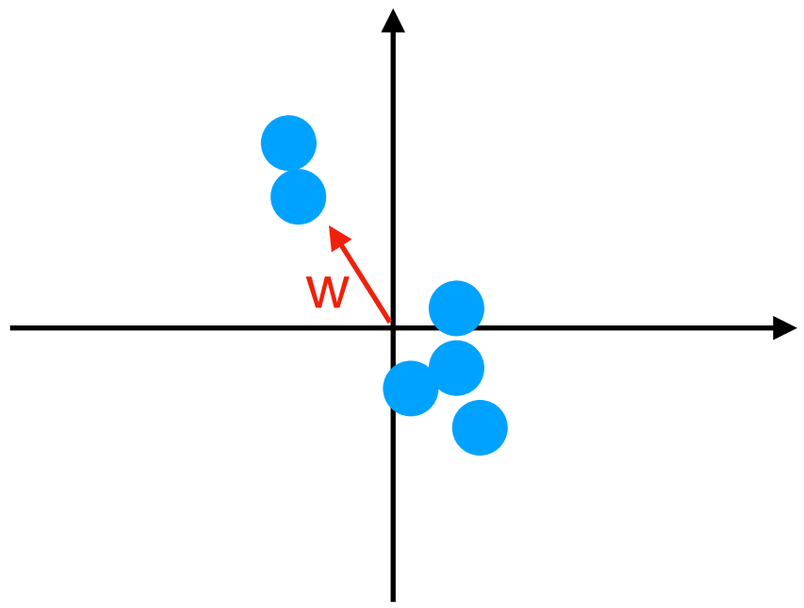
$$\text{trace}((X - ww^T X)(X - ww^T X)^T)$$

$$\text{trace}(-ww^T XX^T - XX^T ww^T + ww^T XX^T ww^T)$$

$$\text{trace}(-w^T XX^T w)$$

PCA interpretations

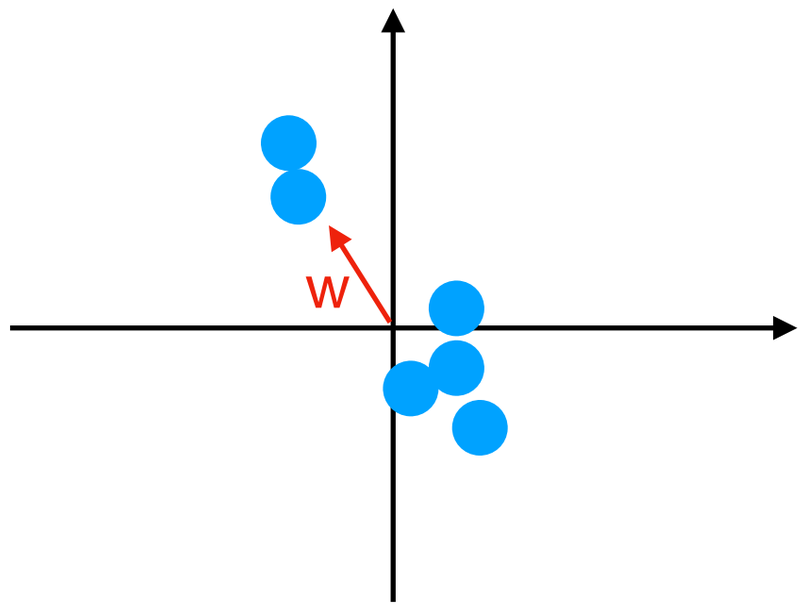
- Capture largest variance
- Best approximation to data
- Direction that is most aligned with data



PCA interpretations

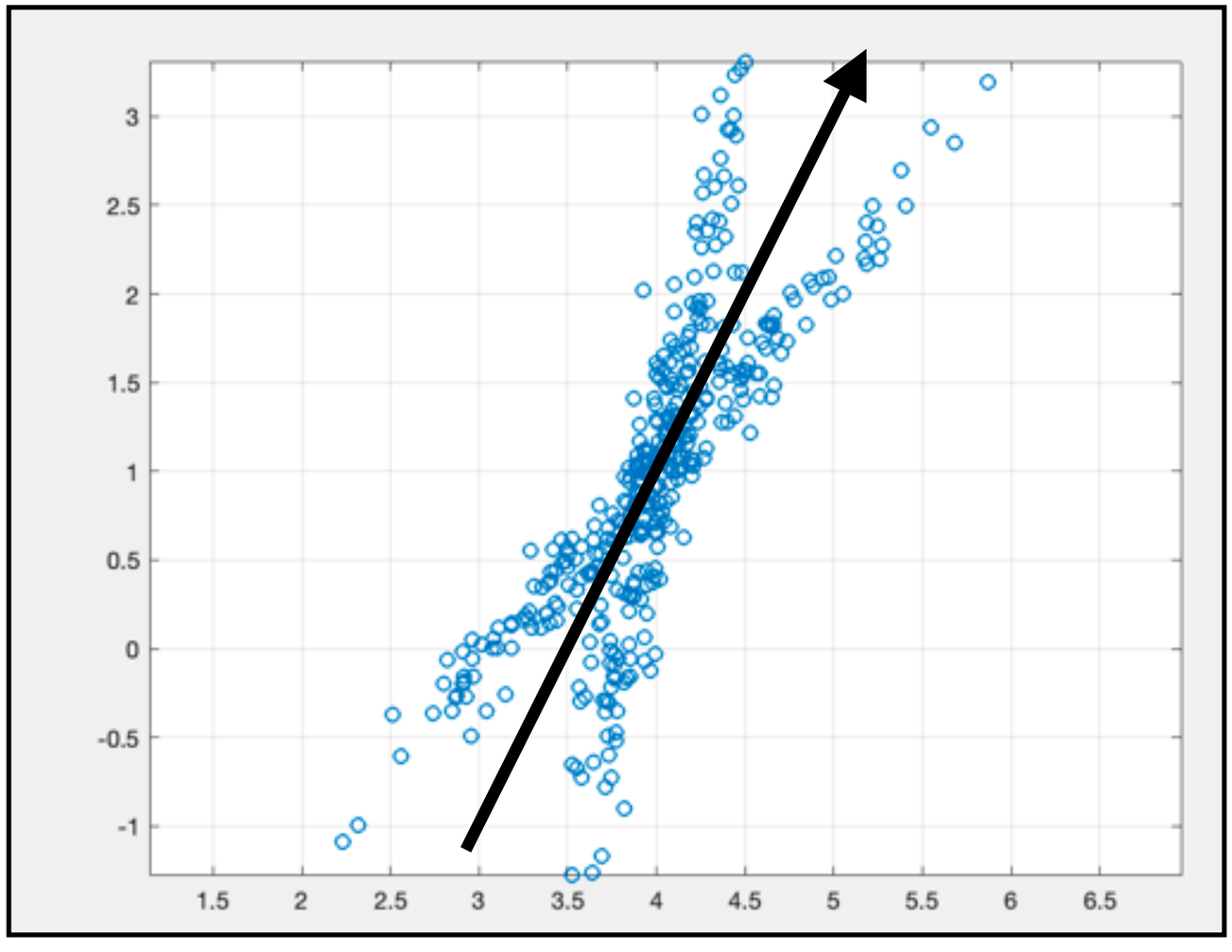
- Capture largest variance
- Best approximation to data
- Direction that is most aligned with data

$w^T X X^T w$ = sum_squares($X^T w$)



The problem with PCA

The direction of maximum variance is not always interesting

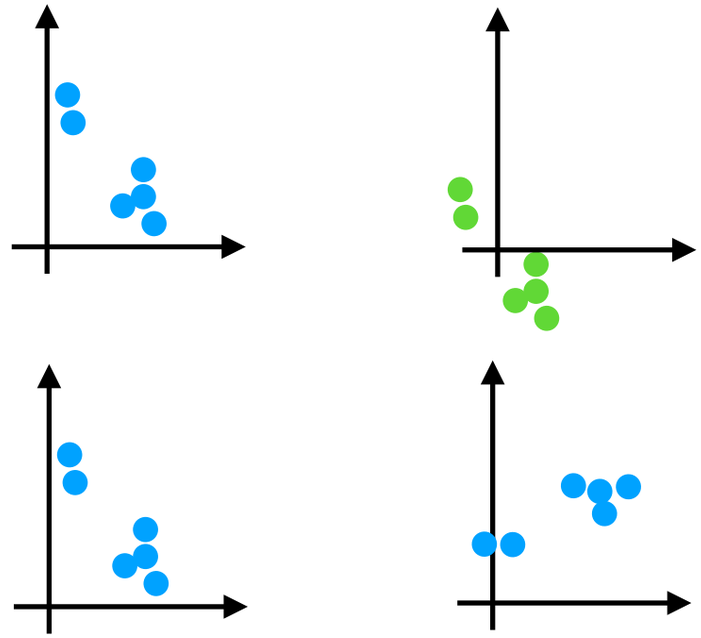


Interestingness

$$\mathrm{I}\left(\mathrm{w}^{T}\mathrm{X}\right)$$

- Location of data should not matter --> centering

$$\mathbf{X}_c = \mathbf{X}(\mathbf{I} - \mathbf{1}\mathbf{1}^T/\mathrm{n})$$



- Orientation of data should not matter --> Ok.

$$(\mathbf{R}\mathbf{w})^T\mathbf{R}\mathbf{X}$$

- ~~Variance?~~

Sphering

n.k.a. whitening

- How to make it so $\text{Variance}(\mathbf{w}^T\mathbf{X}_c) = 1$ for all $\mathbf{w}$?

Sphering

n.k.a. whitening

- How to make it so $\text{Variance}(\mathbf{w}^T\mathbf{X}_c) = 1$ for all $\mathbf{w}$?

$$\mathbf{X}_c \dashrightarrow \mathbf{X}_s = \mathbf{Q}\mathbf{X}_c$$

Sphering

n.k.a. whitening

- How to make it so $\text{Variance}(\mathbf{w}^T\mathbf{X}_c) = 1$ for all $\mathbf{w}$?

$$\mathbf{X}_c \dashrightarrow \mathbf{X}_s = \mathbf{Q}\mathbf{X}_c$$

$$n^* \text{ Variance}(\mathbf{w}^T\mathbf{Q}\mathbf{X}_c) = \mathbf{w}^T(\mathbf{Q}\mathbf{X}_c(\mathbf{Q}\mathbf{X}_c)^T)\mathbf{w} = \mathbf{w}^T\mathbf{Q}\mathbf{X}_c\mathbf{X}_c^T\mathbf{Q}^T\mathbf{w}$$

Sphering

n.k.a. whitening

- How to make it so $\text{Variance}(\mathbf{w}^T\mathbf{X}_c) = 1$ for all $\mathbf{w}$?

$$\mathbf{X}_c \dashrightarrow \mathbf{X}_s = \mathbf{Q}\mathbf{X}_c$$

$$n^* \text{ Variance}(\mathbf{w}^T\mathbf{Q}\mathbf{X}_c) = \mathbf{w}^T(\mathbf{Q}\mathbf{X}_c(\mathbf{Q}\mathbf{X}_c)^T)\mathbf{w} = \mathbf{w}^T\mathbf{Q}\mathbf{X}_c\mathbf{X}_c^T\mathbf{Q}^T\mathbf{w}$$

$$\mathbf{X}_c\mathbf{X}_c^T = \mathbf{U}\mathbf{D}\mathbf{U}^T \quad \text{eigenvalue decomposition}$$

Sphering

n.k.a. whitening

- How to make it so $\text{Variance}(\mathbf{w}^T\mathbf{X}_c) = 1$ for all $\mathbf{w}$?

$$\mathbf{X}_c \text{ --> } \mathbf{X}_s = \mathbf{Q}\mathbf{X}_c$$

$$n^* \text{ Variance}(\mathbf{w}^T\mathbf{Q}\mathbf{X}_c) = \mathbf{w}^T(\mathbf{Q}\mathbf{X}_c(\mathbf{Q}\mathbf{X}_c)^T)\mathbf{w} = \mathbf{w}^T\mathbf{Q}\mathbf{X}_c\mathbf{X}_c^T\mathbf{Q}^T\mathbf{w}$$

$$\mathbf{X}_c\mathbf{X}_c^T = \mathbf{U}\mathbf{D}\mathbf{U}^T \quad \text{eigenvalue decomposition}$$

$$\mathbf{w}^T(\mathbf{Q}\mathbf{X}_c\mathbf{X}_c^T\mathbf{Q}^T)\mathbf{w} = \mathbf{w}^T\mathbf{Q}\mathbf{U}\mathbf{D}\mathbf{U}^T\mathbf{Q}^T\mathbf{w}$$

Sphering

n.k.a. whitening

- How to make it so $\text{Variance}(\mathbf{w}^T\mathbf{X}_c) = 1$ for all $\mathbf{w}$?

$$\mathbf{X}_c \dashrightarrow \mathbf{X}_s = \mathbf{Q}\mathbf{X}_c$$

$$n^* \text{ Variance}(\mathbf{w}^T\mathbf{Q}\mathbf{X}_c) = \mathbf{w}^T(\mathbf{Q}\mathbf{X}_c(\mathbf{Q}\mathbf{X}_c)^T)\mathbf{w} = \mathbf{w}^T\mathbf{Q}\mathbf{X}_c\mathbf{X}_c^T\mathbf{Q}^T\mathbf{w}$$

$$\mathbf{X}_c\mathbf{X}_c^T = \mathbf{U}\mathbf{D}\mathbf{U}^T \quad \text{eigenvalue decomposition}$$

$$\mathbf{w}^T(\mathbf{Q}\mathbf{X}_c\mathbf{X}_c^T\mathbf{Q}^T)\mathbf{w} = \mathbf{w}^T\mathbf{Q}\mathbf{U}\mathbf{D}\mathbf{U}^T\mathbf{Q}^T\mathbf{w}$$

$$\mathbf{Q}\mathbf{U}\mathbf{D}\mathbf{U}^T\mathbf{Q}^T = \text{Identity}$$

Sphering

n.k.a. whitening

- How to make it so $\text{Variance}(\mathbf{w}^T\mathbf{X}_c) = 1$ for all $\mathbf{w}$?

$$\mathbf{X}_c \dashrightarrow \mathbf{X}_s = \mathbf{Q}\mathbf{X}_c$$

$$n^* \text{ Variance}(\mathbf{w}^T\mathbf{Q}\mathbf{X}_c) = \mathbf{w}^T(\mathbf{Q}\mathbf{X}_c(\mathbf{Q}\mathbf{X}_c)^T)\mathbf{w} = \mathbf{w}^T\mathbf{Q}\mathbf{X}_c\mathbf{X}_c^T\mathbf{Q}^T\mathbf{w}$$

$$\mathbf{X}_c\mathbf{X}_c^T = \mathbf{U}\mathbf{D}\mathbf{U}^T \quad \text{eigenvalue decomposition}$$

$$\mathbf{w}^T(\mathbf{Q}\mathbf{X}_c\mathbf{X}_c^T\mathbf{Q}^T)\mathbf{w} = \mathbf{w}^T\mathbf{Q}\mathbf{U}\mathbf{D}\mathbf{U}^T\mathbf{Q}^T\mathbf{w}$$

$$\mathbf{Q}\mathbf{U}\mathbf{D}\mathbf{U}^T\mathbf{Q}^T = \text{Identity}$$

$$\mathbf{Q} = \mathbf{U}\mathbf{D}^{-1/2}\mathbf{U}^T$$

Sphering

n.k.a. whitening

- How to make it so $\text{Variance}(\mathbf{w}^T\mathbf{X}_c) = 1$ for all $\mathbf{w}$?

$$\mathbf{X}_c \dashrightarrow \mathbf{X}_s = \mathbf{Q}\mathbf{X}_c$$

$$n^* \text{ Variance}(\mathbf{w}^T\mathbf{Q}\mathbf{X}_c) = \mathbf{w}^T(\mathbf{Q}\mathbf{X}_c(\mathbf{Q}\mathbf{X}_c)^T)\mathbf{w} = \mathbf{w}^T\mathbf{Q}\mathbf{X}_c\mathbf{X}_c^T\mathbf{Q}^T\mathbf{w}$$

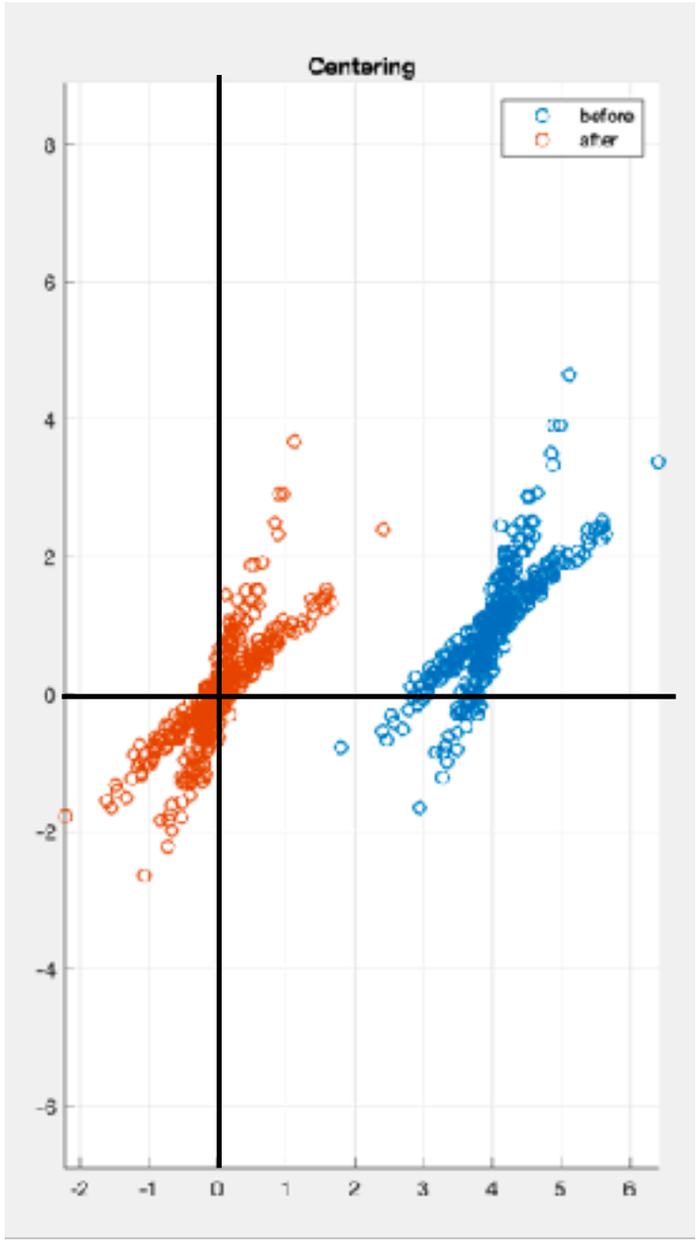
$$\mathbf{X}_c\mathbf{X}_c^T = \mathbf{U}\mathbf{D}\mathbf{U}^T$$ eigenvalue decomposition

$$\mathbf{w}^T(\mathbf{Q}\mathbf{X}_c\mathbf{X}_c^T\mathbf{Q}^T)\mathbf{w} = \mathbf{w}^T\mathbf{Q}\mathbf{U}\mathbf{D}\mathbf{U}^T\mathbf{Q}^T\mathbf{w}$$

$$\mathbf{Q}\mathbf{U}\mathbf{D}\mathbf{U}^T\mathbf{Q}^T = \text{Identity}$$

$$\mathbf{Q} = \mathbf{U}\mathbf{D}^{-1/2}\mathbf{U}^T$$ project onto eigenvectors
rescale (undo what X does)
project back

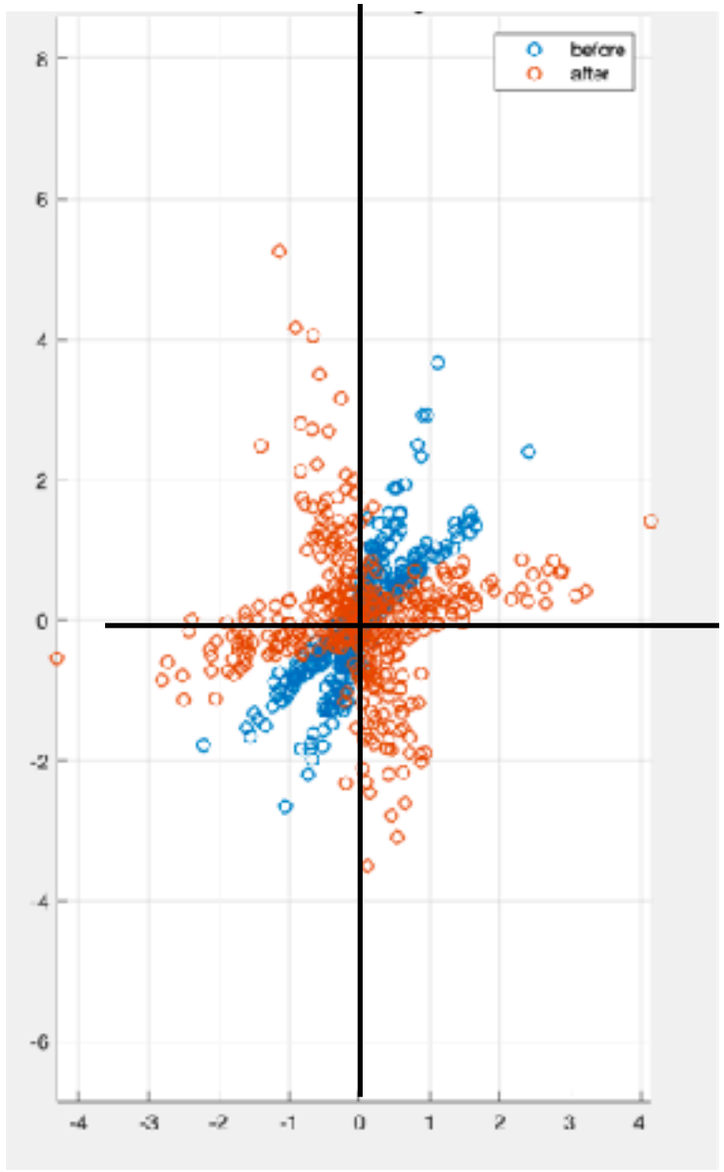
Centering



~~function X_c = center(X)~~
~~n = size(X,2);~~
~~C = eye(n) - ones(n,1)*ones(1,n)/n;~~
~~X_c = X*C;~~

```
function X_c = center(X)
  X_c = X - mean(X,2);
```

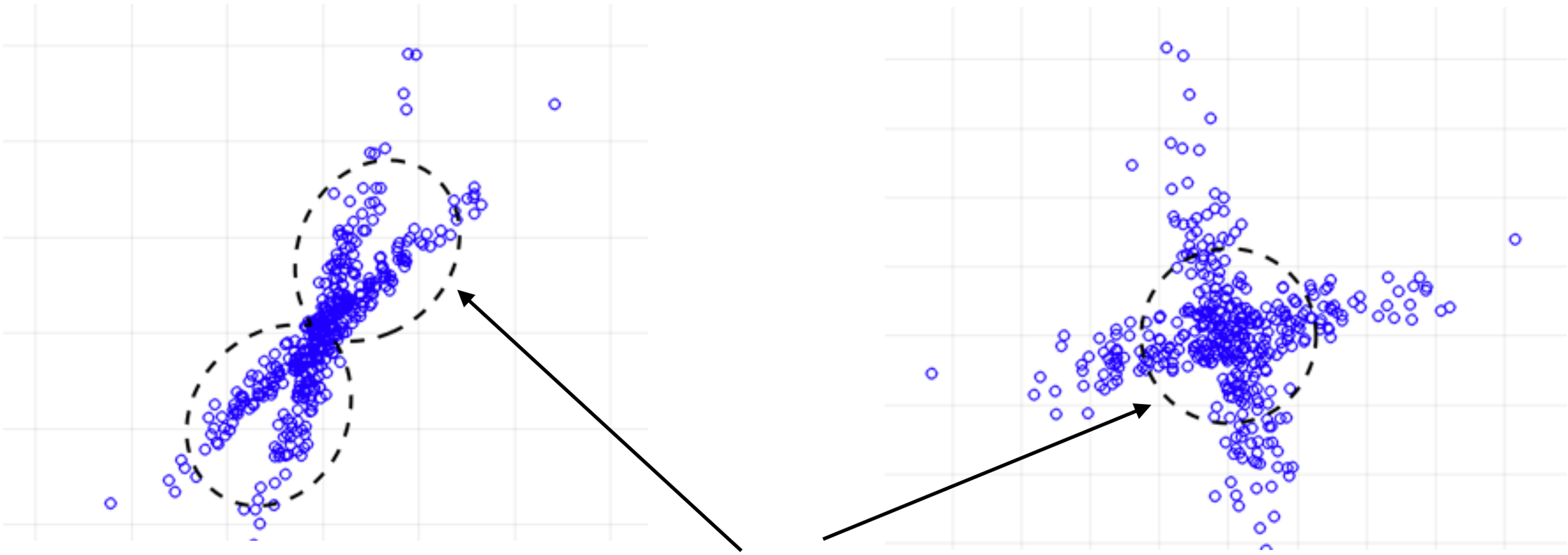
~~Sphering~~ whitening




```
function X_s = whiten(X)

   [V,D] = eig(cov(X'));
   Q = V*diag(1./sqrt(diag(D)))*V';
   X_s = Q*center(X);
```

centered

whitened



variance long different orientations

```
function plot_var(X)

angles = linspace(0,2*pi,1000);   % angles to calculate variance along
V      = zeros(size(angles));     % Variance
for i=1:length(angles)
    % get vector w along angle
    [w_x,w_y] = pol2cart(angles(i),1);
    w = [w_x;w_y];
    % calculate cf along w
    V(i) = var(w'*X);
end

figure,hold on
plot(X(1,:),X(2,:),'bo')

[xx,yy] = pol2cart(angles,V);
plot(xx,yy,'linewidth',2,'color','k','linestyle','--')
axis equal
grid on
```

Interestingness

$$\mathrm{I}\left(\mathrm{w}^{T}\mathrm{X}\right)$$

What is uninteresting?

Gaussian I am afraid

(everything is Gaussian by default)

Gaussians are boring

Gaussians are boring

- If you randomly project X, chances are you get a Gaussian (central limit theorem)

Gaussians are boring

- If you randomly project X, chances are you get a Gaussian (central limit theorem)
- Gaussian is the maximum entropy distribution (if you fix the mean and variance)

Gaussians are boring

- If you randomly project X, chances are you get a Gaussian (central limit theorem)
- Gaussian is the maximum entropy distribution (if you fix the mean and variance)
- So..... minimize entropy! (maximize neg-entropy). I.e. maximize non-gaussianity.

Maths of relative entropy and gaussians

Maths of relative entropy and gaussians

Relative entropy = -∫p.log(q/p)

Distance between p and q (>=0)

Maths of relative entropy and gaussians

Relative entropy = -∫p.log(q/p)

Distance between p and q (>=0)

Entropy = -∫p.log(1/p)

A bit like distance between p and 1 (no structure)

Maths of relative entropy and gaussians

Relative entropy = -∫p.log(q/p)

Distance between p and q (>=0)

Entropy = -∫p.log(1/p)

A bit like distance between p and 1 (no structure)

-∫p.log(q/p) = -∫p.log(q) -∫p.log(1/p)

Maths of relative entropy and gaussians

Relative entropy = -∫p.log(q/p)

Distance between p and q (>=0)

Entropy = -∫p.log(1/p)

A bit like distance between p and 1 (no structure)

-∫p.log(q/p) = -∫p.log(q) -∫p.log(1/p)

= -∫q.log(q) - Entropy(p)

Maths of relative entropy and gaussians

Relative entropy = -∫p.log(q/p)

Distance between p and q (>=0)

Entropy = -∫p.log(1/p)

A bit like distance between p and 1 (no structure)

-∫p.log(q/p) = -∫p.log(q) -∫p.log(1/p)

= -∫q.log(q) - Entropy(p)

q Gaussian -> log(q) is quadratic

Maths of relative entropy and gaussians

Relative entropy = $-\int p.\log(q/p)$

Distance between p and q (>=0)

Entropy = $-\int p.\log(1/p)$

A bit like distance between p and 1 (no structure)

$-\int p.\log(q/p) = -\int p.\log(q) - \int p.\log(1/p)$

$= -\int q.\log(q) - \text{Entropy}(p)$

q Gaussian -> log(q) is quadratic

-> $E_p[\log(q)] = E_q[\log(q)]$ = combination of 1st and 2nd moments!

Maths of relative entropy and gaussians

Relative entropy = $-\int p.\log(q/p)$

Distance between p and q (>=0)

Entropy = $-\int p.\log(1/p)$

A bit like distance between p and 1 (no structure)

$-\int p.\log(q/p) = -\int p.\log(q) - \int p.\log(1/p)$

$= -\int q.\log(q) - \text{Entropy}(p)$

q Gaussian -> log(q) is quadratic

-> $E_p[\log(q)] = E_q[\log(q)]$ = combination of 1st and 2nd moments!

$= \text{Entropy}(q) - \text{Entropy}(p)$

Maths of relative entropy and gaussians

Relative entropy = $-\int p.\log(q/p)$

Distance between p and q (>=0)

Entropy = $-\int p.\log(1/p)$

A bit like distance between p and 1 (no structure)

$-\int p.\log(q/p) = -\int p.\log(q) - \int p.\log(1/p)$

$= -\int q.\log(q) - \text{Entropy}(p)$

q Gaussian -> log(q) is quadratic

-> $E_p[\log(q)] = E_q[\log(q)]$ = combination of 1st and 2nd moments!

$= \text{Entropy}(q) - \text{Entropy}(p)$

>=0

Maths of relative entropy and gaussians

Relative entropy = $-\int p.\log(q/p)$

Distance between p and q (>=0)

Entropy = $-\int p.\log(1/p)$

A bit like distance between p and 1 (no structure)

$-\int p.\log(q/p) = -\int p.\log(q) - \int p.\log(1/p)$

$= -\int q.\log(q) - \text{Entropy}(p)$

q Gaussian -> log(q) is quadratic

-> $E_p[\log(q)] = E_q[\log(q)]$ = combination of 1st and 2nd moments!

$= \text{Entropy}(q) - \text{Entropy}(p)$

>=0

Gaussian = maximum entropy when mean and variance fixed

Ok, but how do you actually calculate entropy?

Ok, but how do you actually calculate entropy?

- hmm... we are given samples, not the actual data distribution

Ok, but how do you actually calculate entropy?

- hmm... we are given samples, not the actual data distribution
- kernel density not an option (we need to find w with optimization)

Ok, but how do you actually calculate entropy?

- hmm... we are given samples, not the actual data distribution

- kernel density not an option (we need to find w with optimization)

- approximation

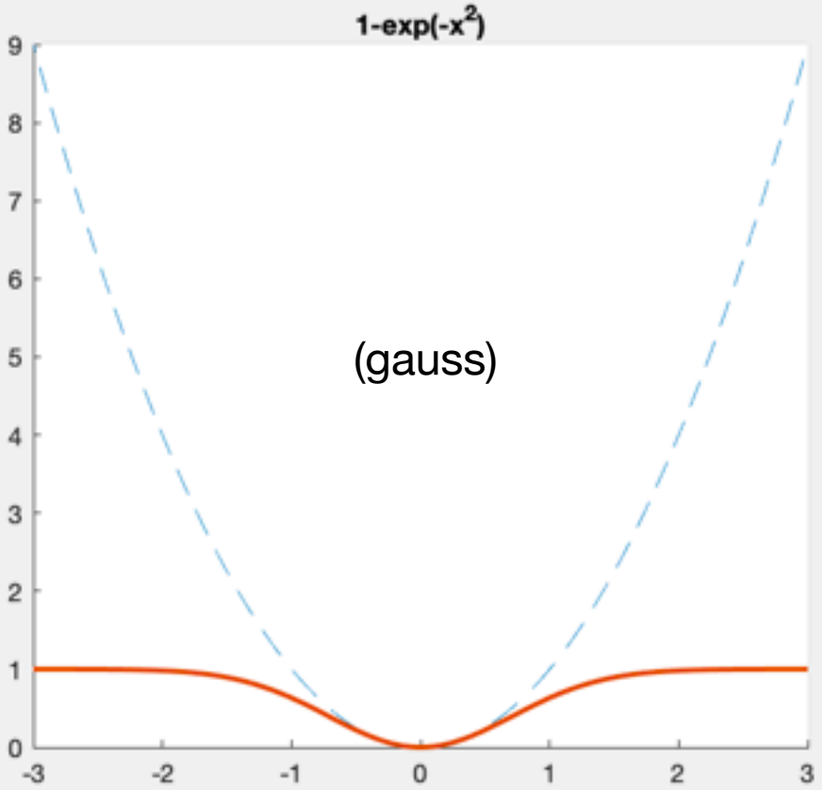
$$\mathbf{I}\left(\mathbf{w}^T\mathbf{X}\right) = E\left[\left(\mathbf{w}^T\mathbf{X}\right)^2\right]$$

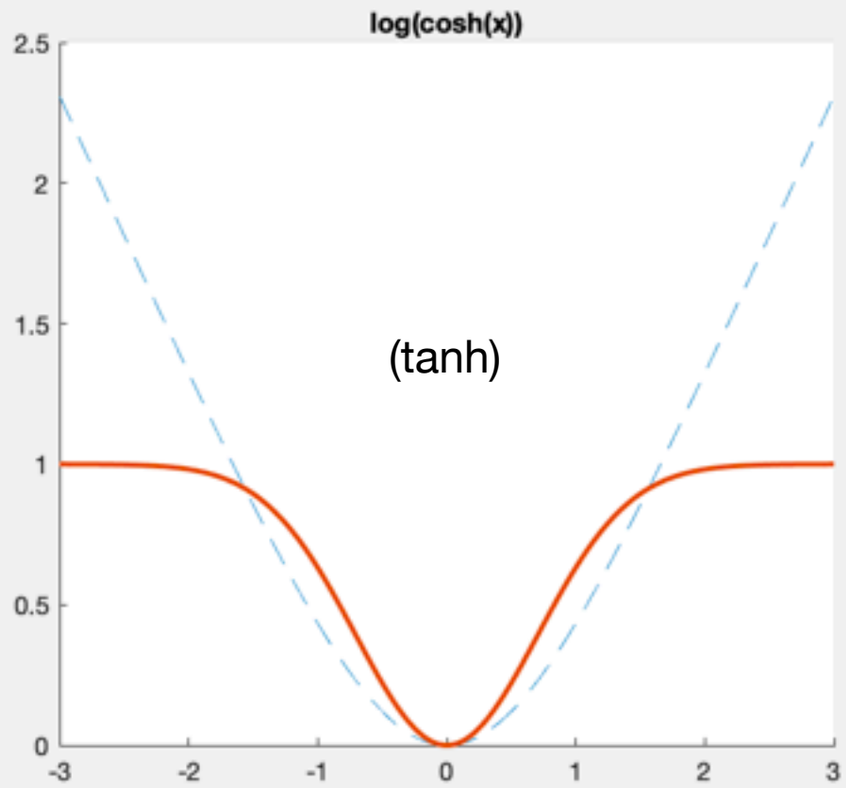
variance

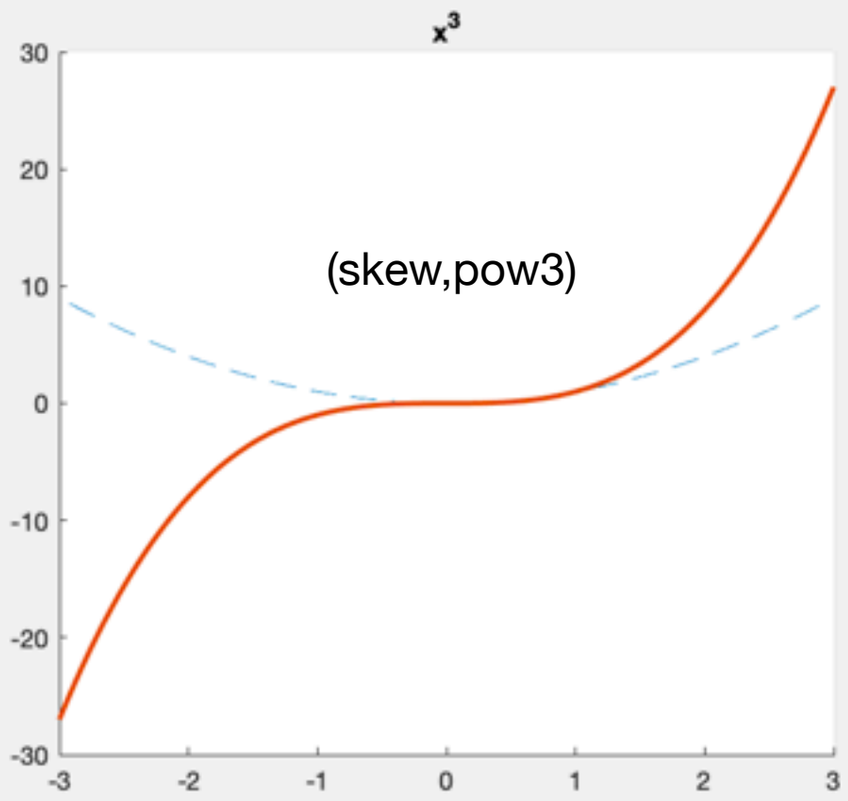
$$\mathbf{I}\left(\mathbf{w}^T\mathbf{X}\right) = E\left[g\left(\mathbf{w}^T\mathbf{X}\right)\right]$$

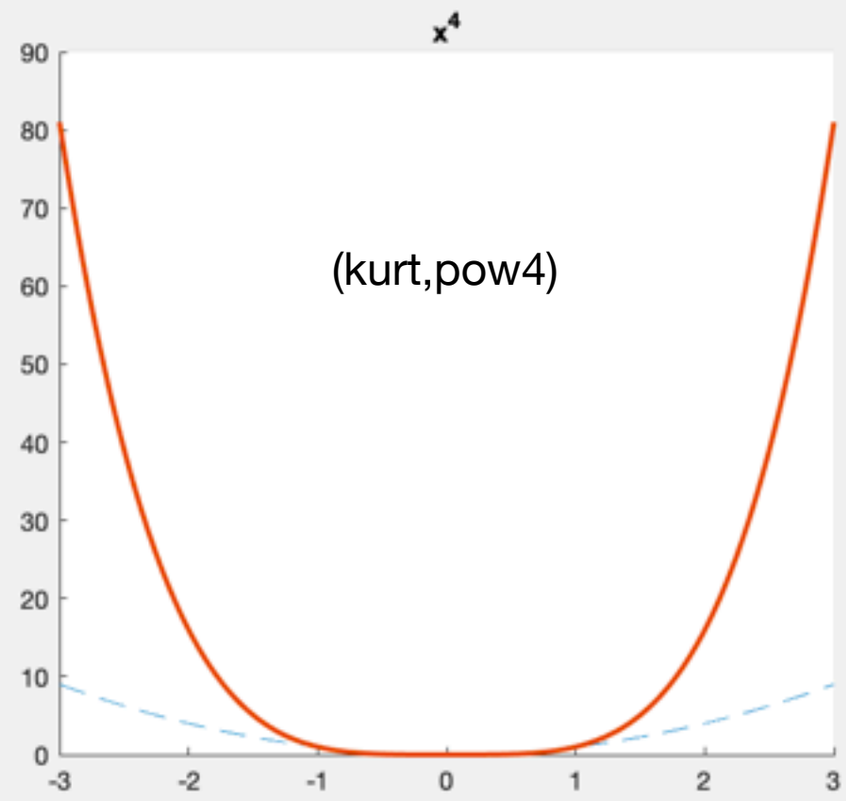
other (as long as g is not quadratic)

Advantages of approximation:
continuous function (good for optimization)
easy to calculate (unlike entropy)


1-exp(-x^2)
(gauss)

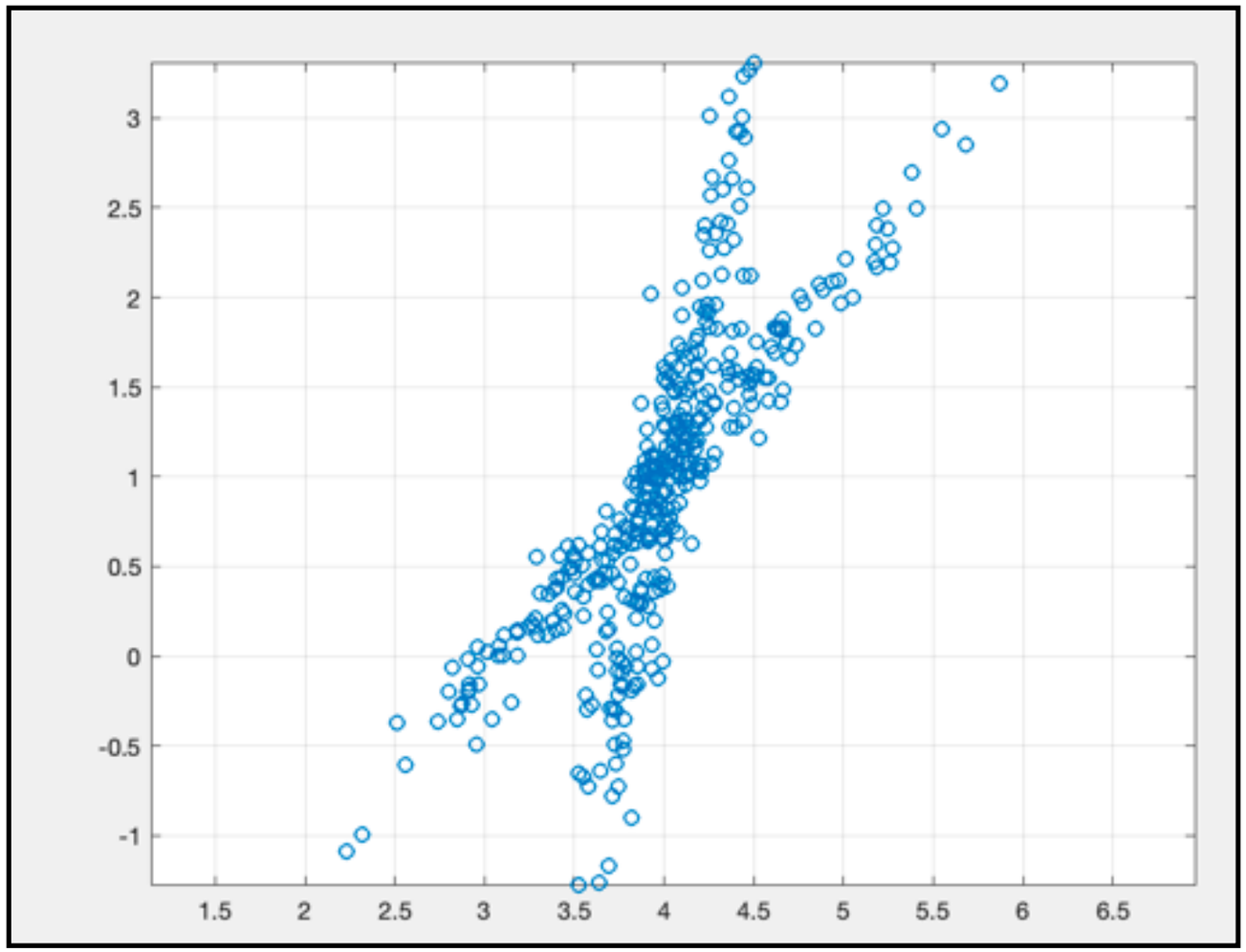

log(cosh(x))
(tanh)


x^3
(skew,pow3)


x^4
(kurt,pow4)

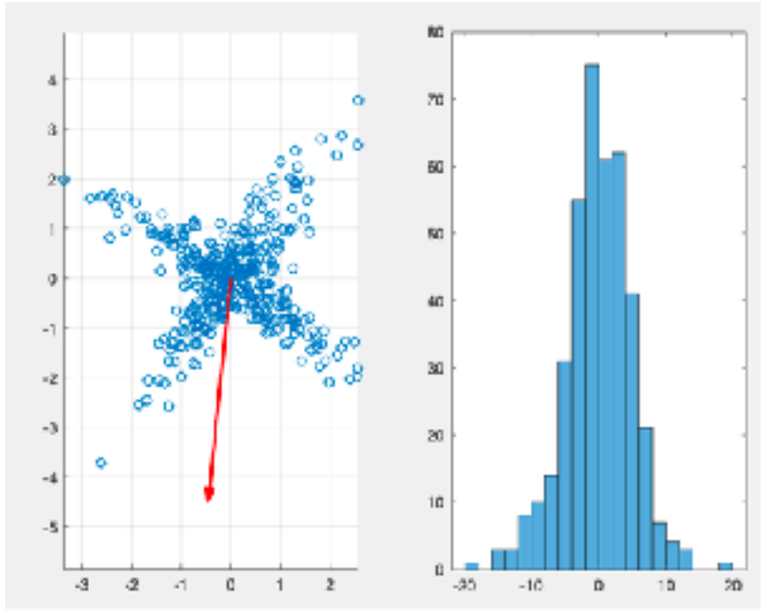
Let's try this

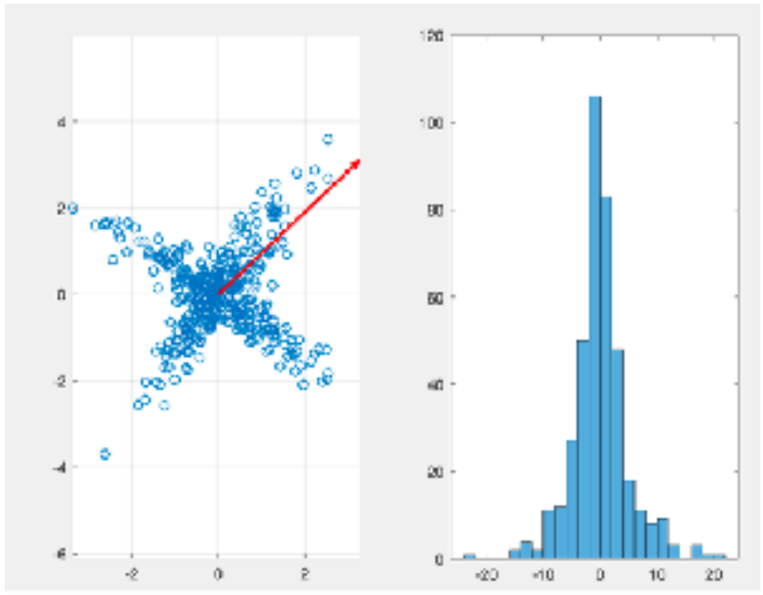
E.g. rFMRI spatial ICA:
each point = voxel
each axis = time point

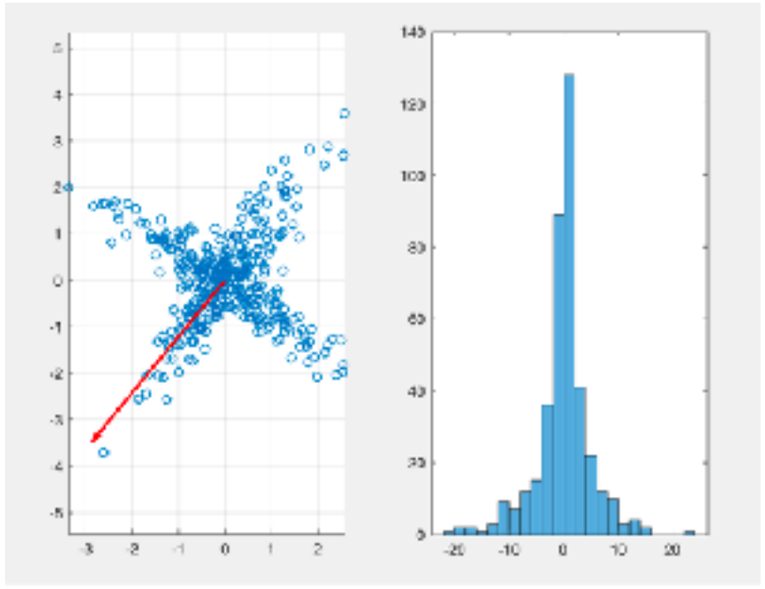


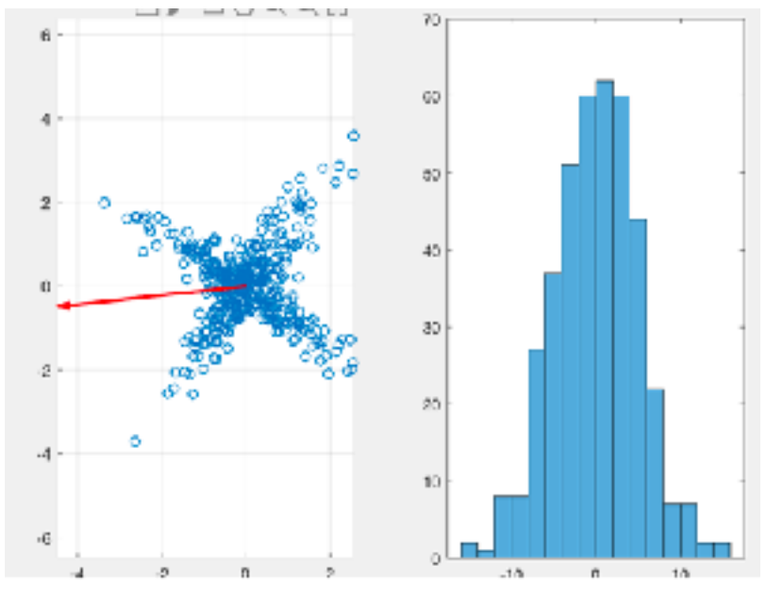

```
n = 400;  % number of samples
d = 2;    % dimensionality

w1 = randn(d,1);
w1 = w1/norm(w1);  % direction 1 is random
ang = 30;          % rotation angle
rot = [cosd(ang) sind(ang);   %rotation matrix
      -sind(ang) cosd(ang)];
w2 = rot*w1;       % direction 2 is rotated re direction 1

% Generate data along the two directions
% (multivariate Gaussians spread a little along w1 and w2
x1 = mvnrnd([4;1],w1*w1',n/2) + .1*randn(n/2,2);
x2 = mvnrnd([4;1],w2*w2',n/2) + .1*randn(n/2,2);

% Concatenate the two clouds of data
x = [x1;x2]';

figure
plot(x(1,:),x(2,:),'o')
grid on
axis equal
```

Random projection w'X

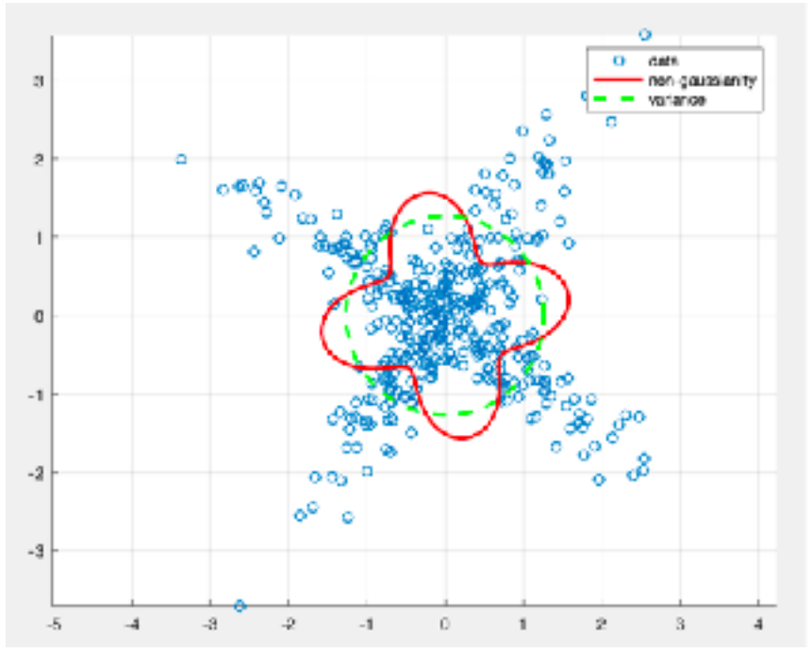




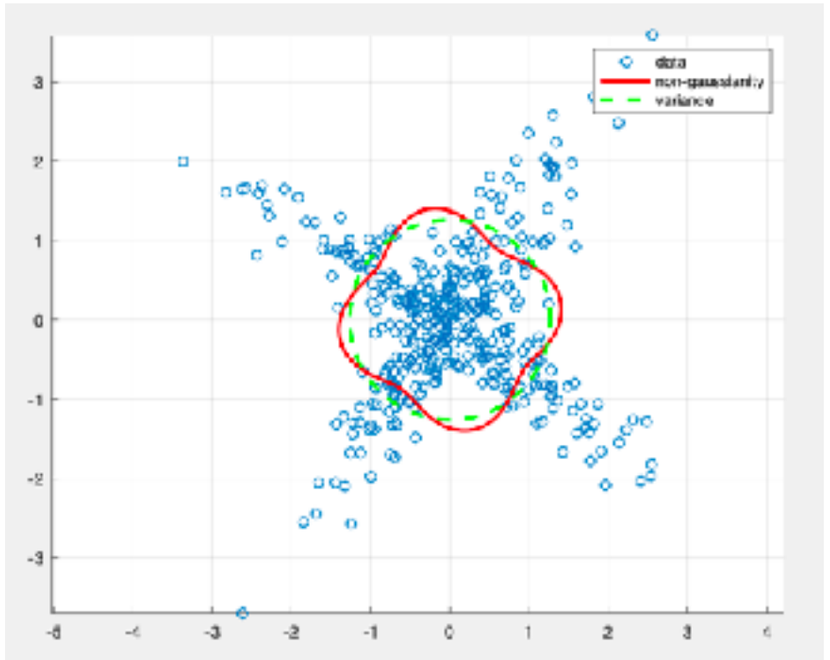




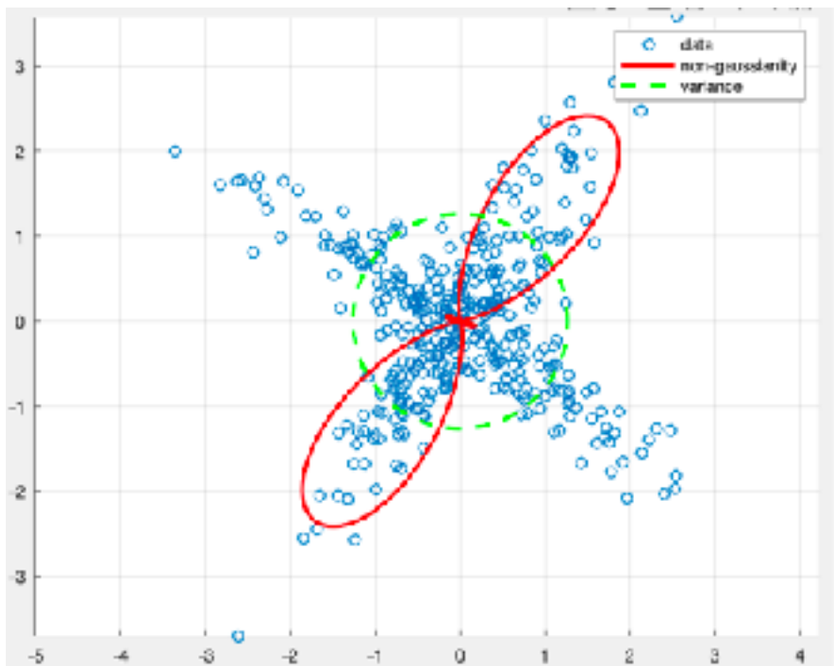
Gauss



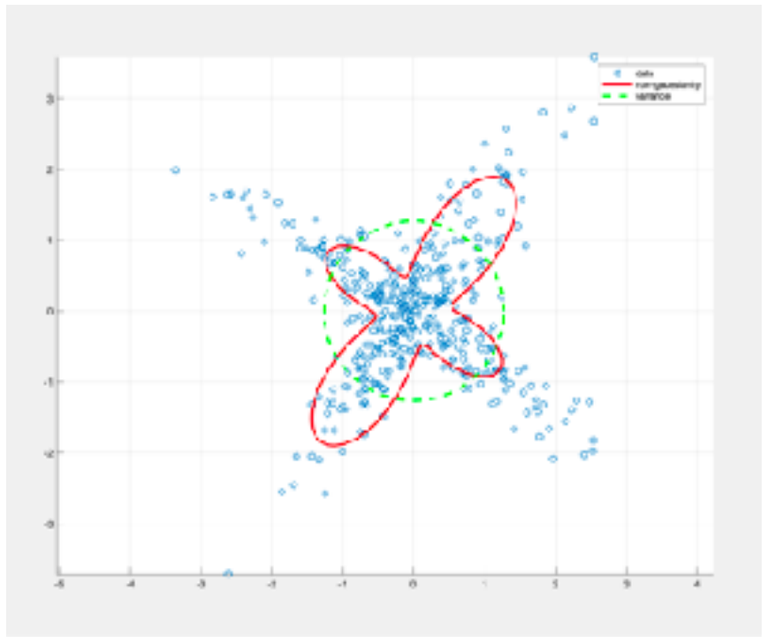
logcosh (tanh)



pow3 (skewness)



pow4 (kurtosis)



Beyond 2D

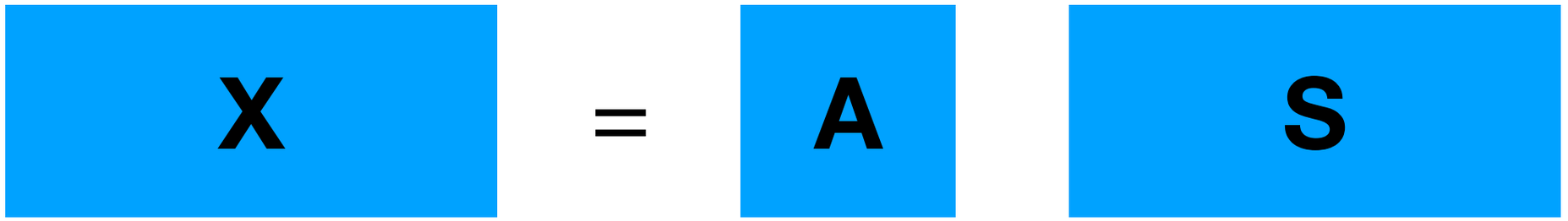
- We can't just look at the 2D projections
- We need an algorithm fo finding w
- We have a cost function! $E\left[g(\mathbf{w}^T\mathbf{X})\right]$ s.t. $\mathbf{w}^T\mathbf{w} = 1$
- FastICA = Newton-like algorithm

ICA

$$\underset{\text{data}}{\mathbf{X}} = \underset{\text{mixing matrix}}{\mathbf{A}}\underset{\text{mixed signals}}{\mathbf{S}}$$

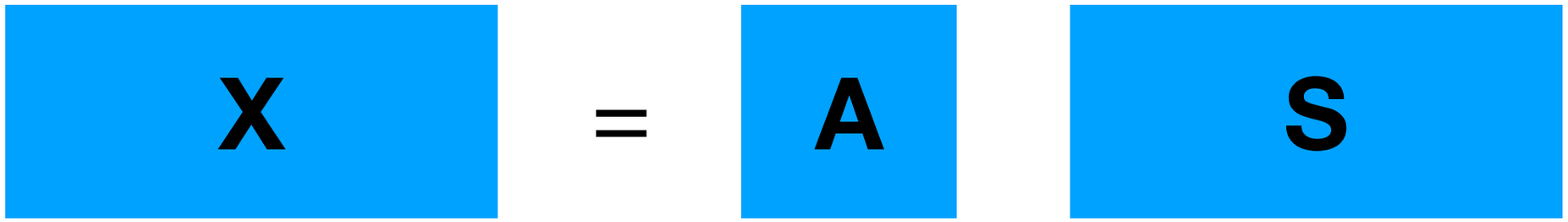
ICA

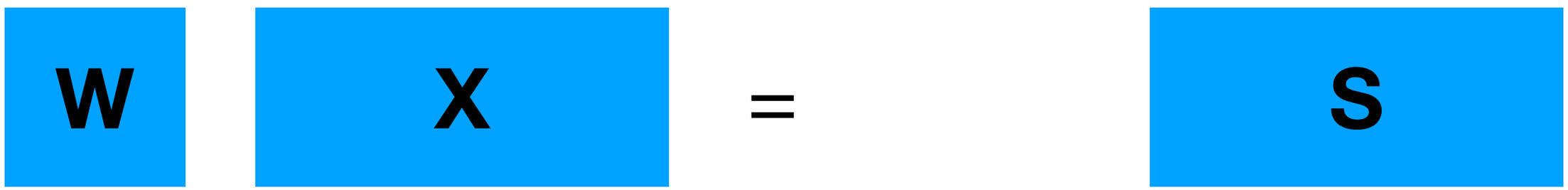
$$\underset{\text{data}}{X} = \underset{\text{mixing matrix}}{A}\underset{\text{mixed signals}}{S}$$



ICA

$$\underset{\text{data}}{X} = \underset{\text{mixing matrix}}{A}\underset{\text{mixed signals}}{S}$$

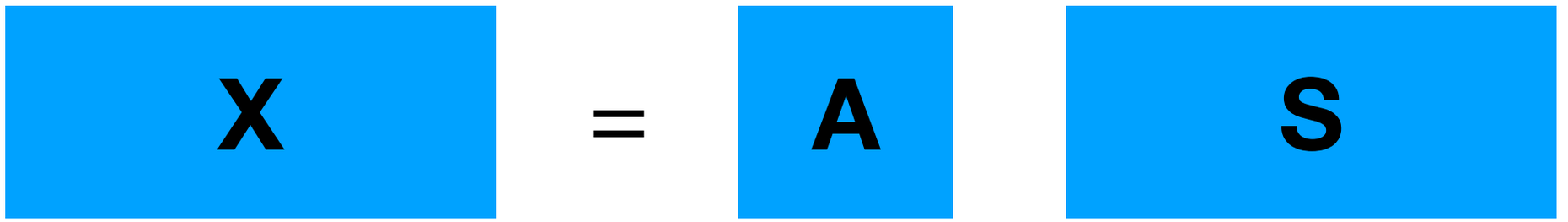


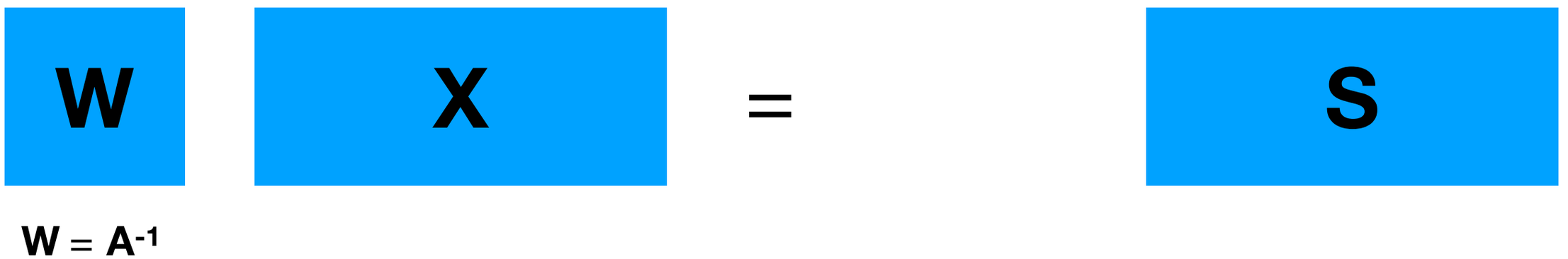


$W = A^{-1}$

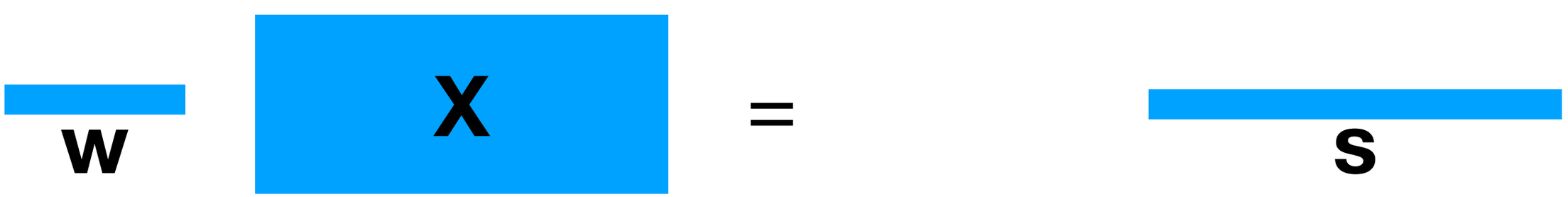
ICA

$$\underset{\text{data}}{X} = \underset{\text{mixing matrix}}{A}\underset{\text{mixed signals}}{S}$$





$W = A^{-1}$

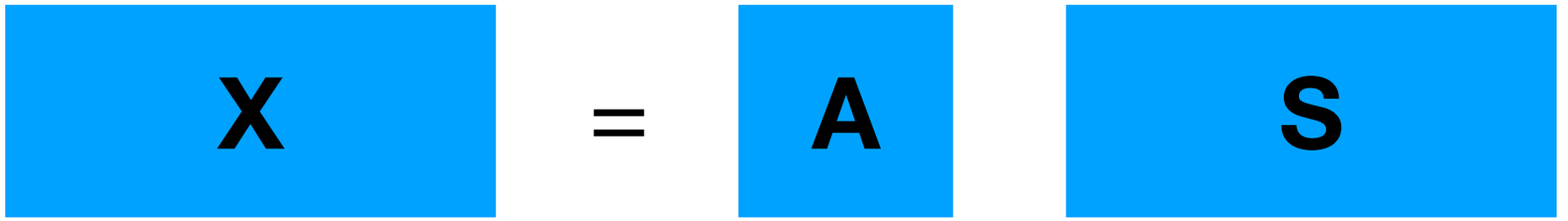


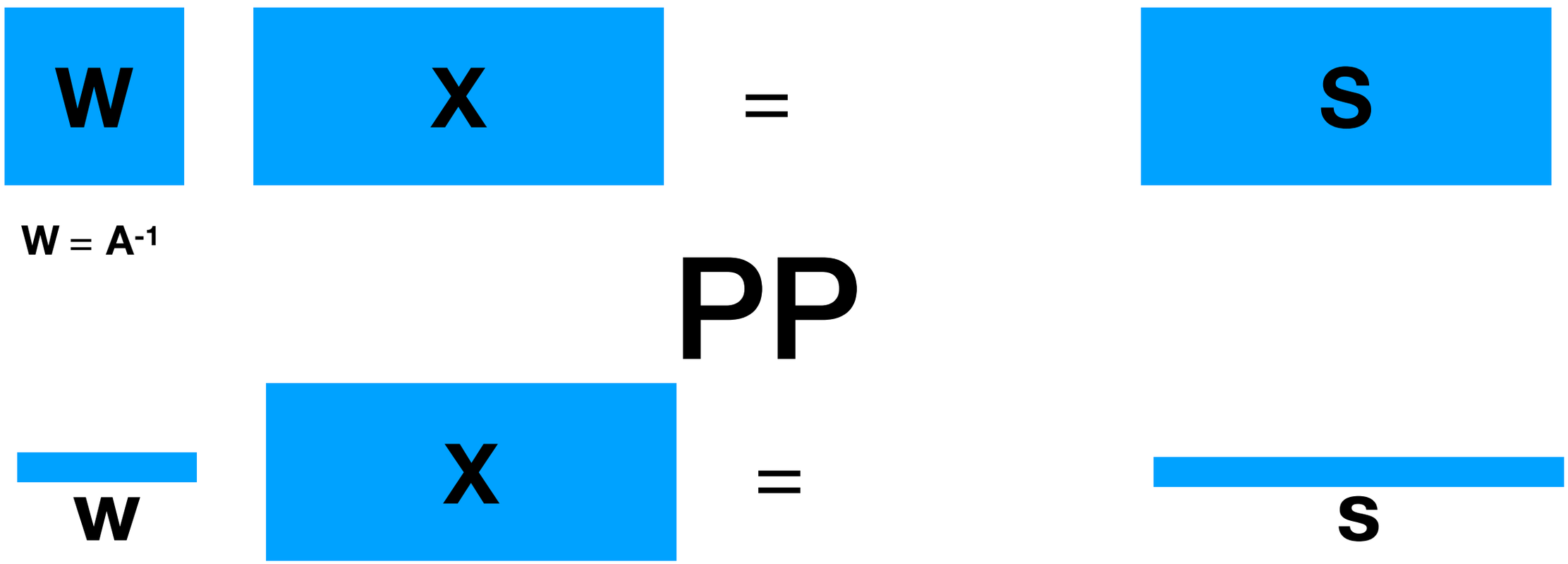
ICA

data mixed signals

$$X = AS$$

mixing matrix





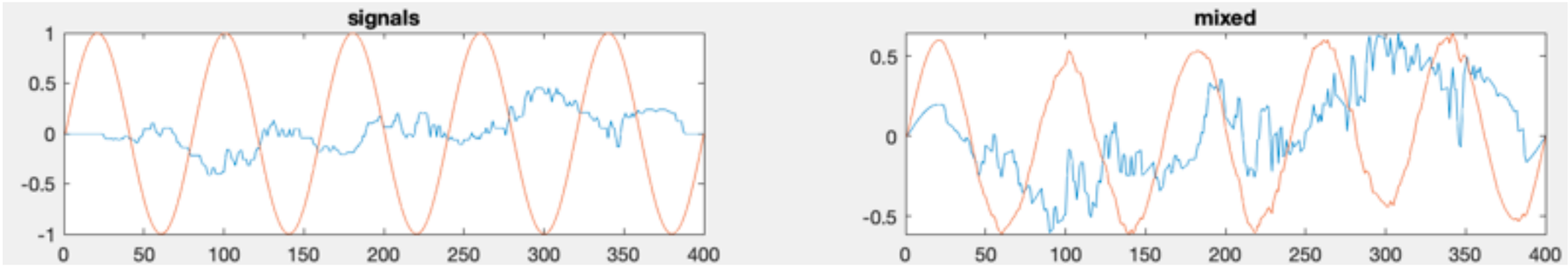
$W = A^{-1}$

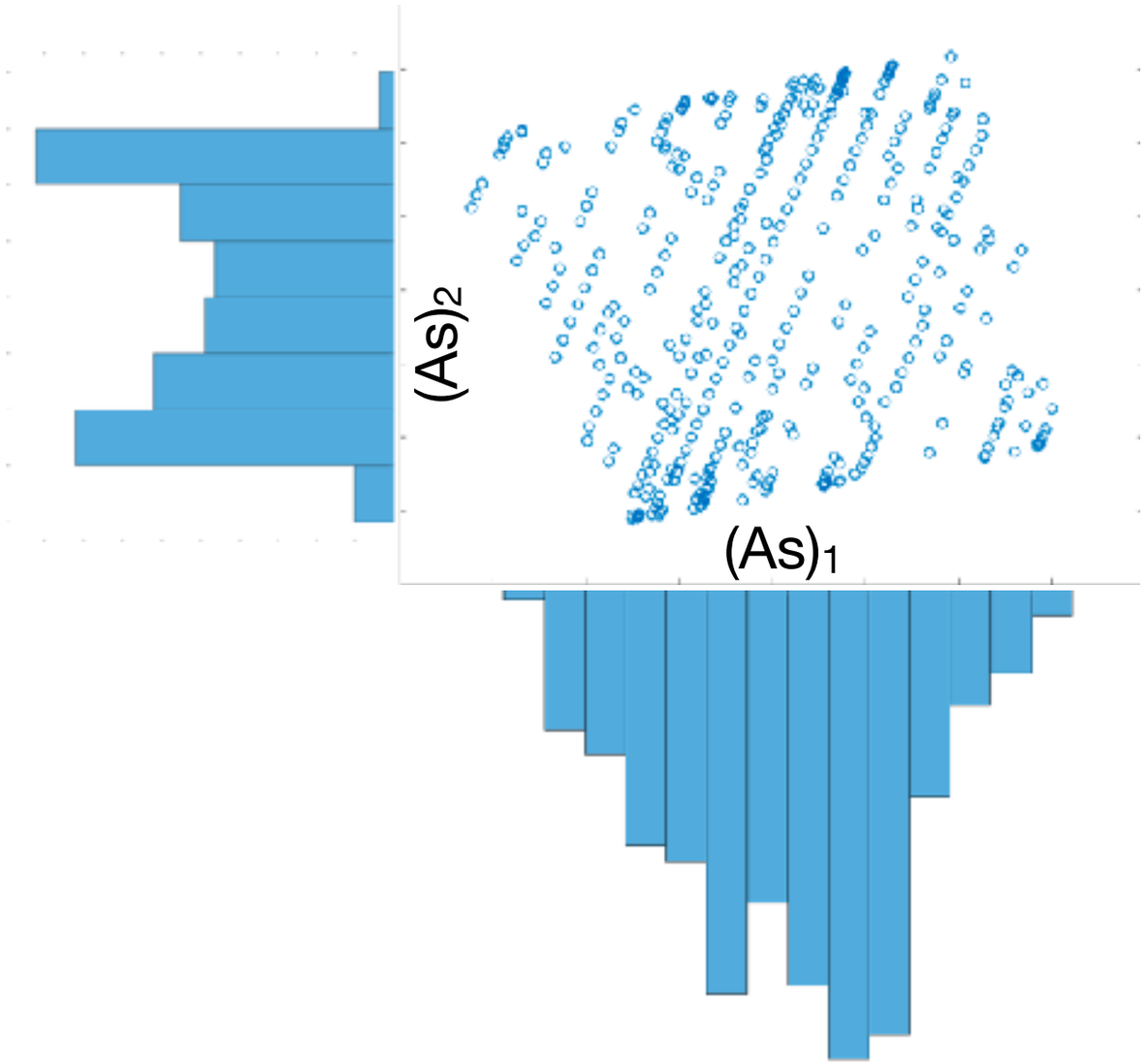
PP

Temporal PP

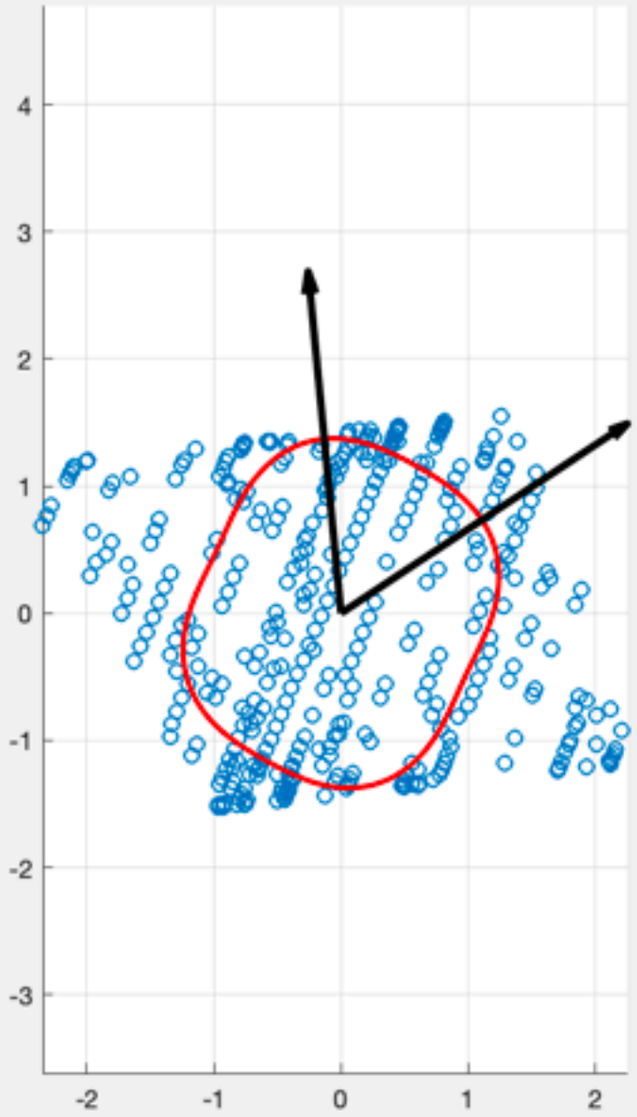
S (2xn)

AS

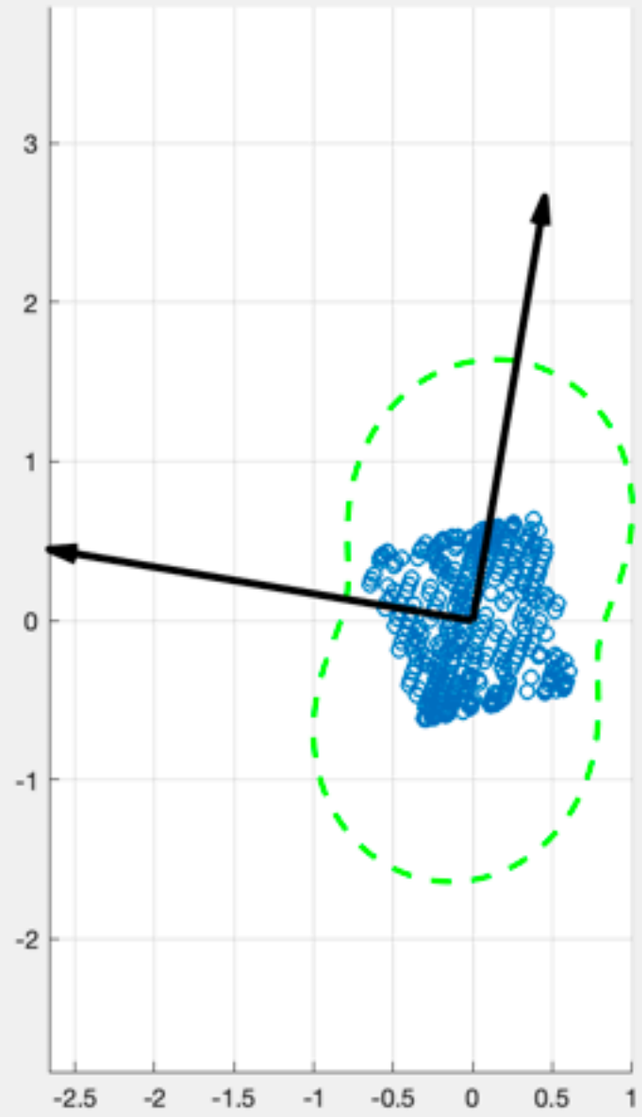


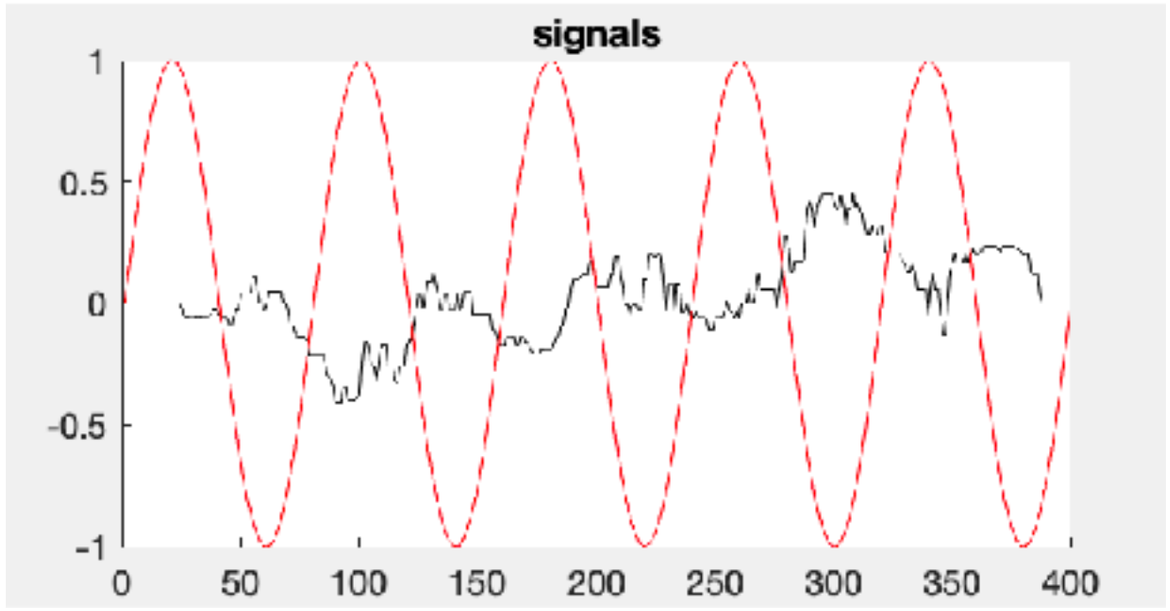


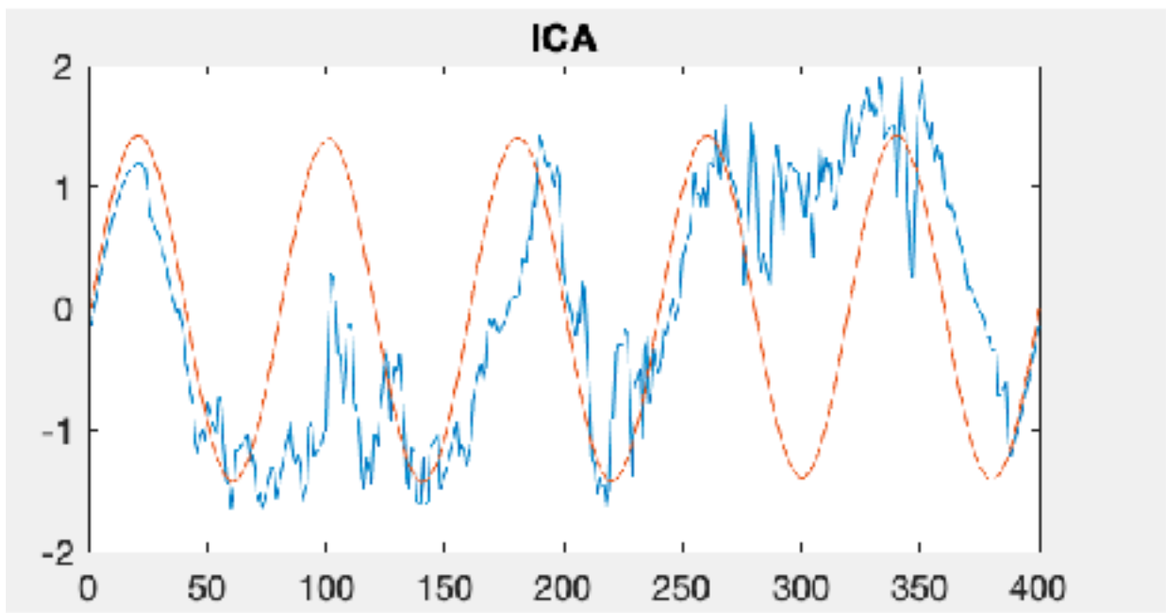
ICA (PP)
sphered



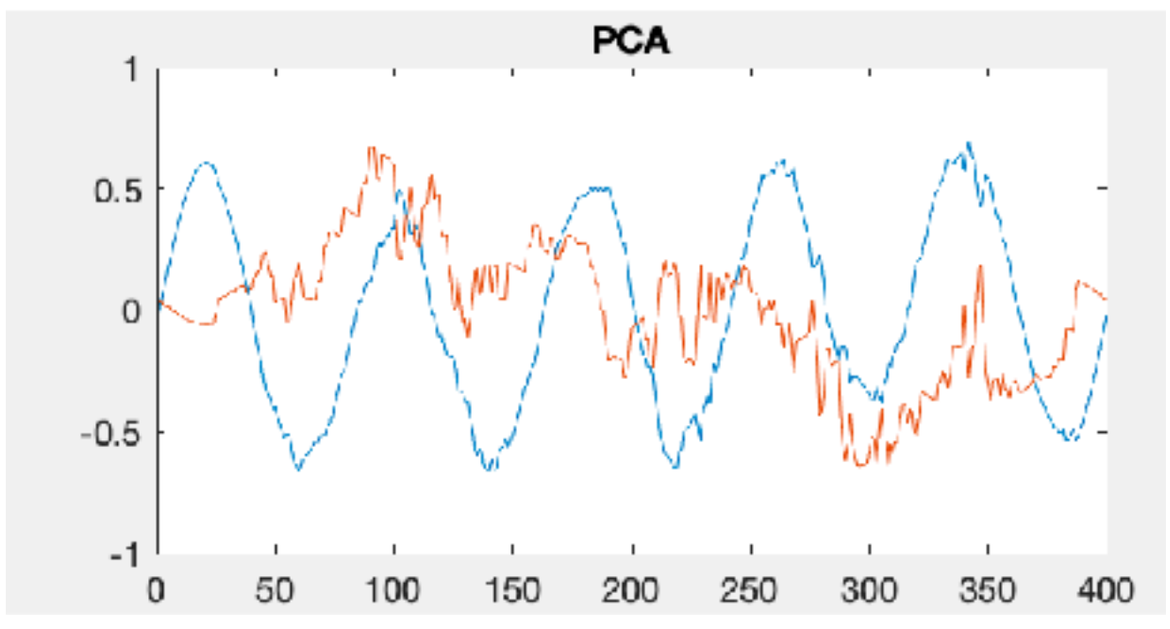
PCA
not sphered







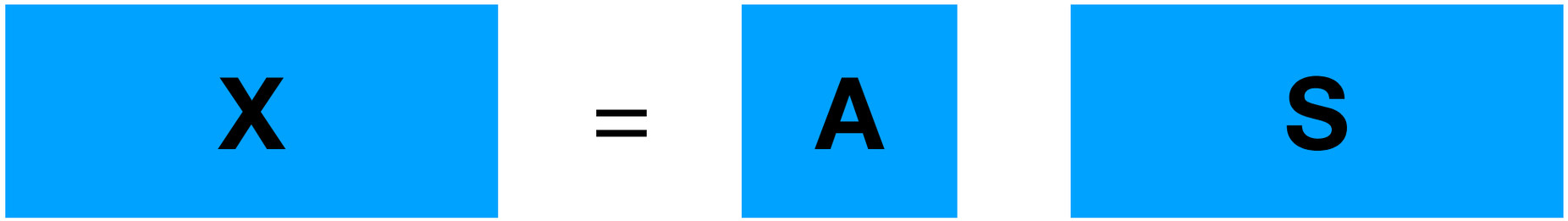
Successful unmixing



Still some mixing

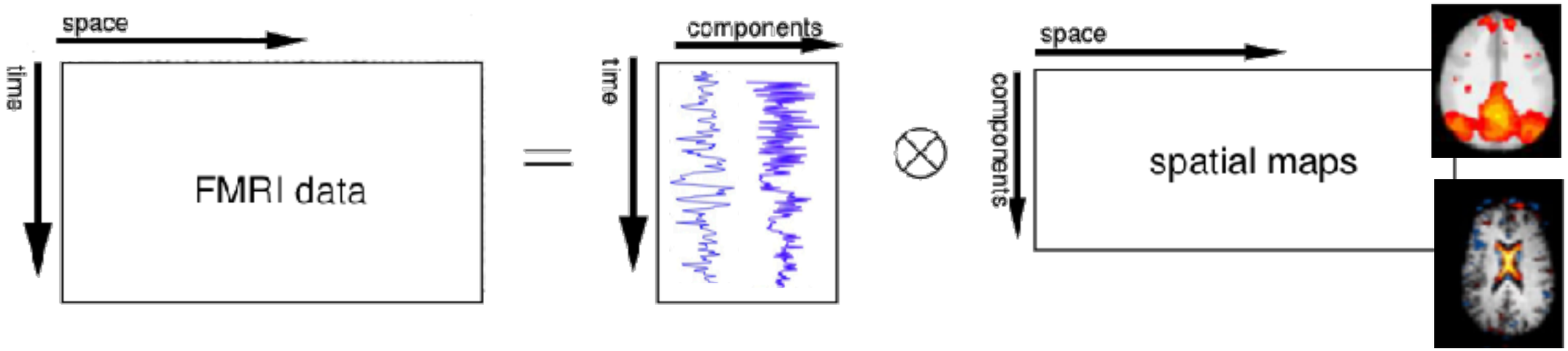
Spatial PP

Spatial PP



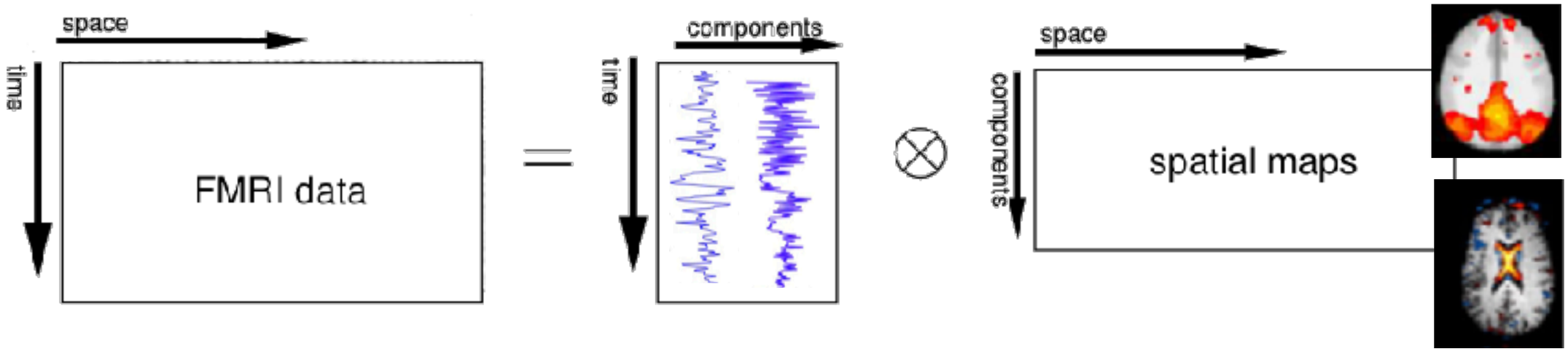
Spatial PP

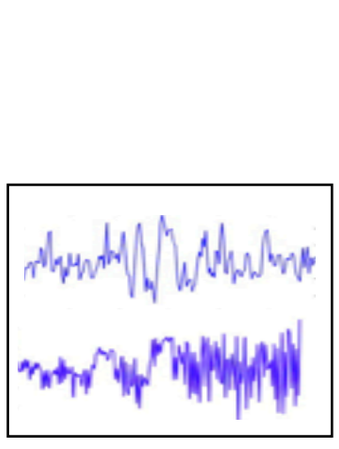
X = **A** **S**



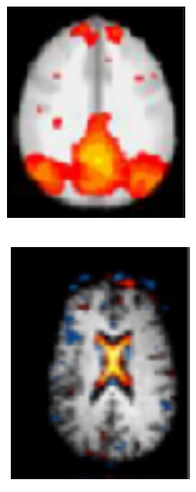
Spatial PP

X = **A** **S**

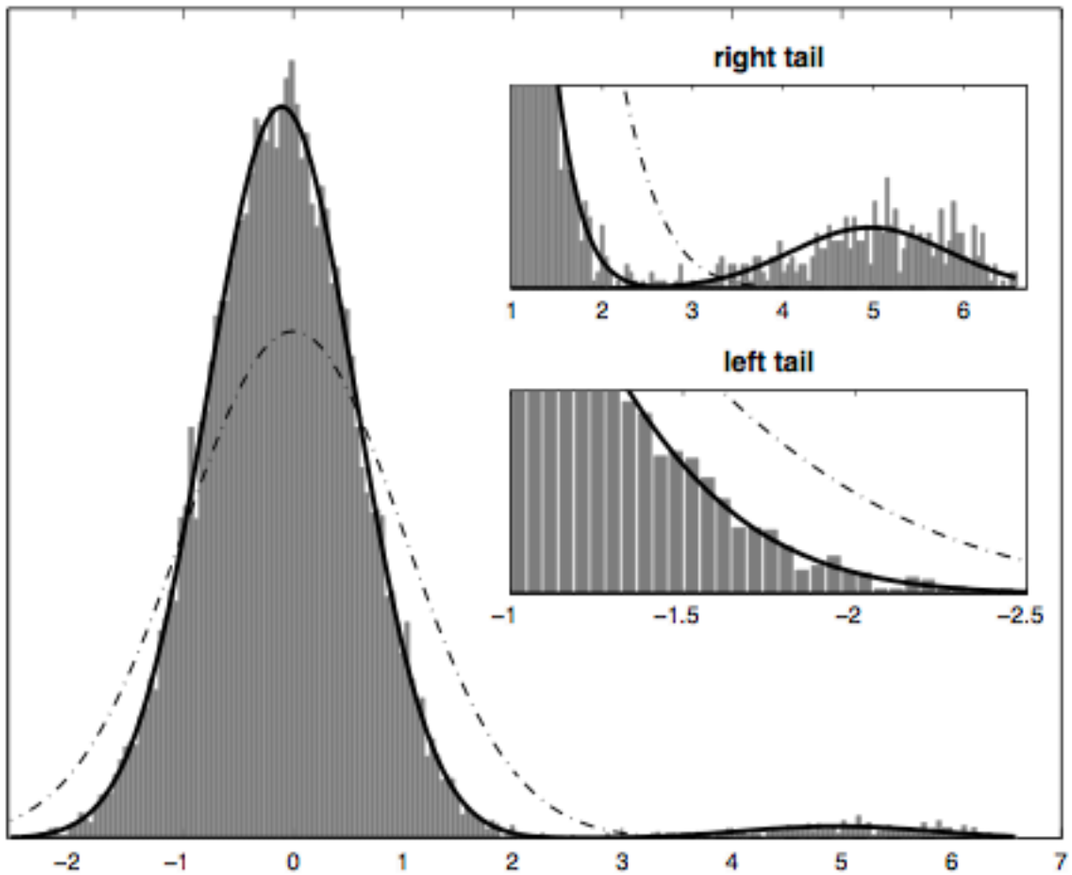




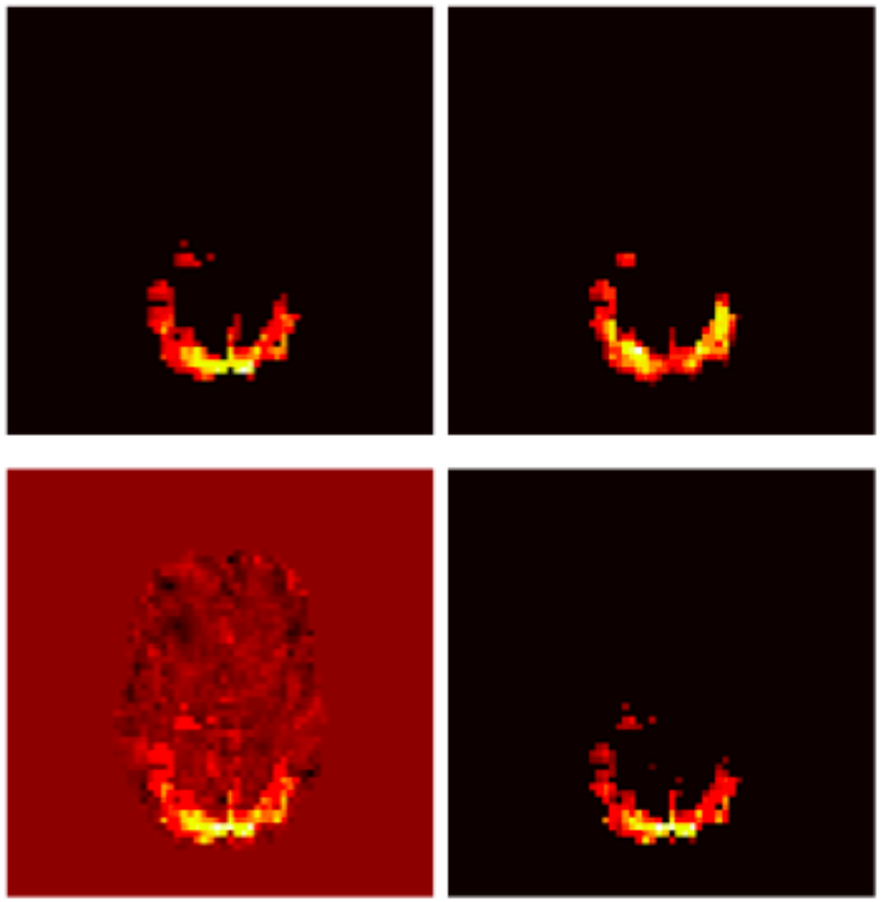
FMRI data = spatial maps



Interpretation: similarity between voxel time course and IC time course



(i)



(ii)

On orthogonality

On orthogonality

- When we talk about IC's what are we talking about?

On orthogonality

- When we talk about IC's what are we talking about?
- Remember: X=AS

On orthogonality

- When we talk about IC's what are we talking about?
- Remember: X=AS
- but X is sphered: QX=AS

On orthogonality

- When we talk about IC's what are we talking about?
- Remember: X=AS
- but X is sphered: QX=AS
- A = mixing matrix = W^{-1}

On orthogonality

- When we talk about IC's what are we talking about?
- Remember: X=AS
- but X is sphered: QX=AS
- A = mixing matrix = W^{-1}
- --> $X=Q^{-1}AS$

On orthogonality

- When we talk about IC's what are we talking about?
- Remember: X=AS
- but X is sphered: QX=AS
- A = mixing matrix = W^{-1}
- --> $X=Q^{-1}AS$
- rows of S (independent components) are actually orthogonal!

On orthogonality

- When we talk about IC's what are we talking about?
- Remember: X=AS
- but X is sphered: QX=AS
- A = mixing matrix = W^{-1}
- --> $X=Q^{-1}AS$
- rows of S (independent components) are actually orthogonal!
- This is because: $SS^T=(WQX)^TWQX=WQXX^TQ^TW^T=WW^T$=Identity

Difference between ICA and projection pursuit

Difference between ICA and projection pursuit

- ICA (like PCA) has an explicit model: X = A*S whereas PP only has data and tries to find w s.t. w'X has a funky distribution

Difference between ICA and projection pursuit

- ICA (like PCA) has an explicit model: X = A*S whereas PP only has data and tries to find w s.t. w'X has a funky distribution

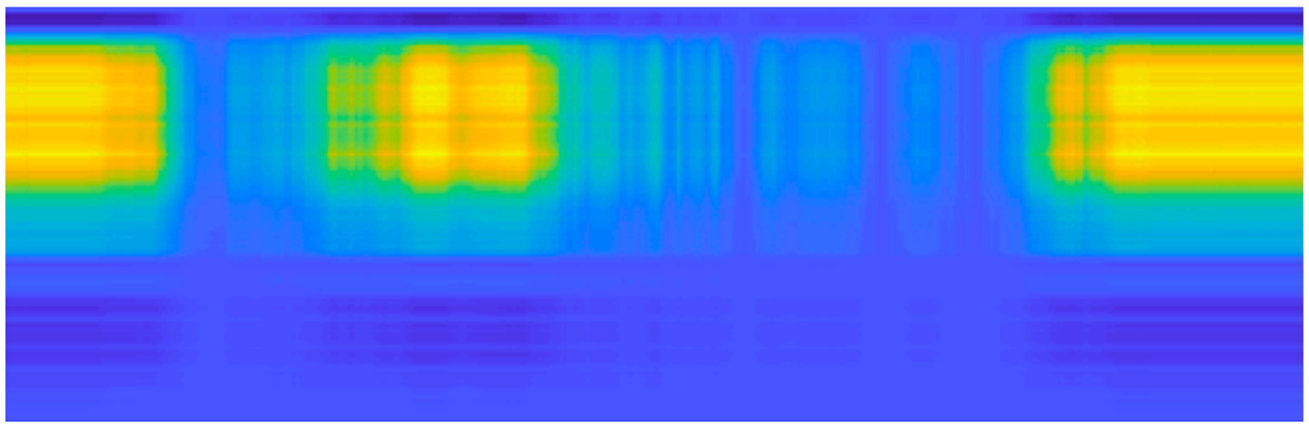
- Because ICA has a model of the data, it can be augmented with e.g. noise model (like Melodic), explicit model for S (e.g. mixture of gaussians like FLICA or other mixtures like PROFUMO), etc.

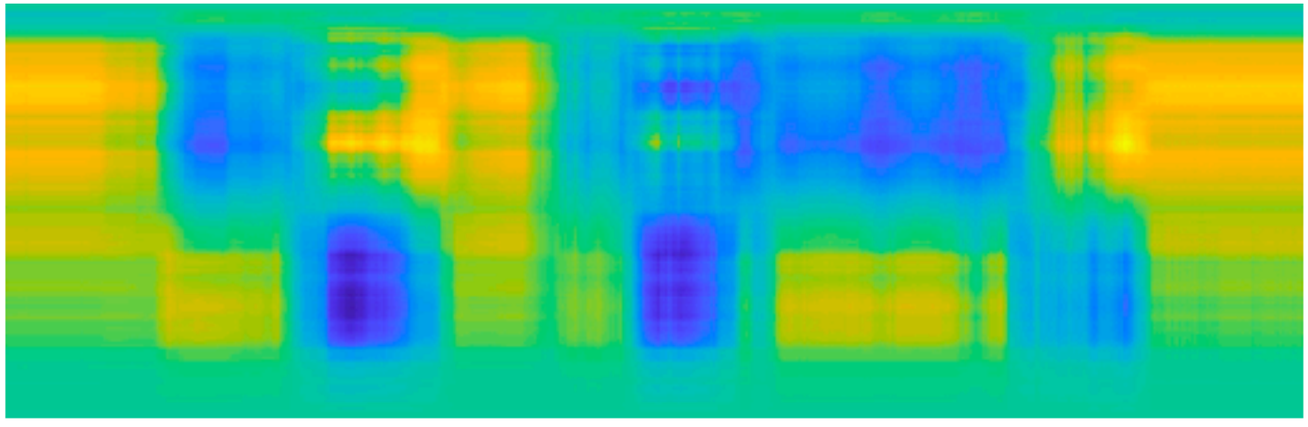
Difference between ICA and projection pursuit

- ICA (like PCA) has an explicit model: X = A*S whereas PP only has data and tries to find w s.t. w'X has a funky distribution

- Because ICA has a model of the data, it can be augmented with e.g. noise model (like Melodic), explicit model for S (e.g. mixture of gaussians like FLICA or other mixtures like PROFUMO), etc.

- But FastICA is PP with nice approximations to neg-entropy and an efficient algorithm for finding w





The end.