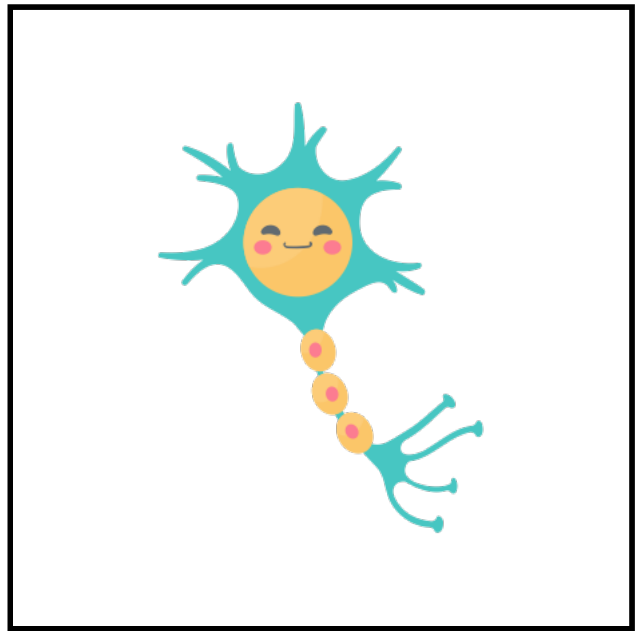
Brain Code Camp

Dimensionality Reduction

Itthi Chatnuntawech

Brain Code Camp

RGB Images



1024 x 1024 pixels

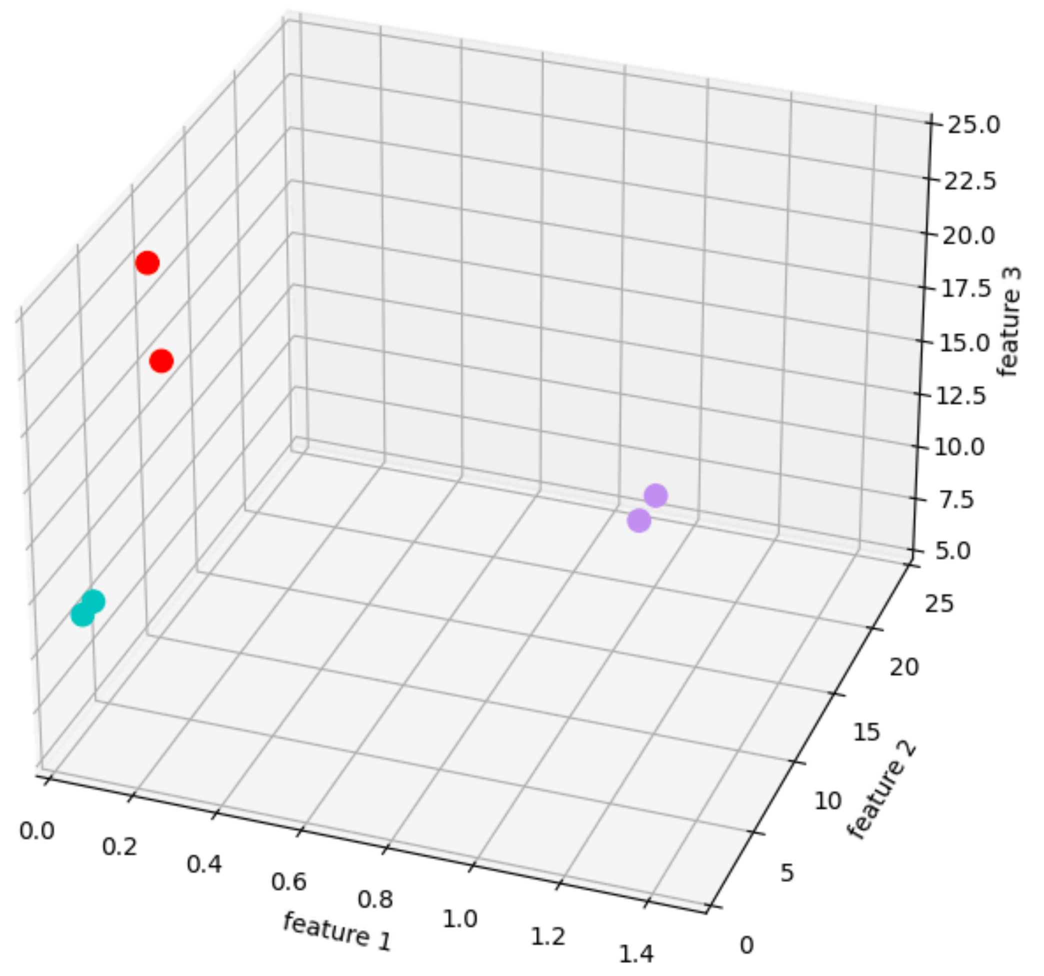
If we use raw pixels as features, we will have

1024	x	1024	x	3	=	3,145,728 features.
# of rows		# of columns		# of channels		

3,145,728-dimensional feature space!

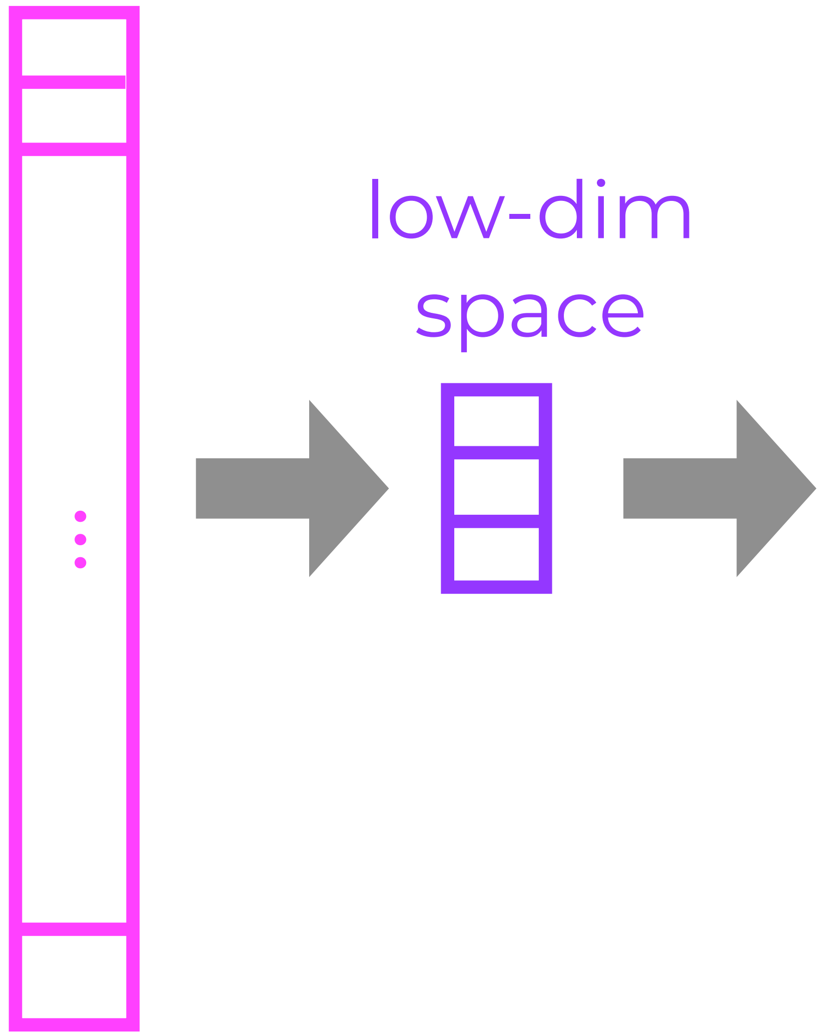
Dimensionality Reduction

data visualization



reduced computational resources

high-dim space

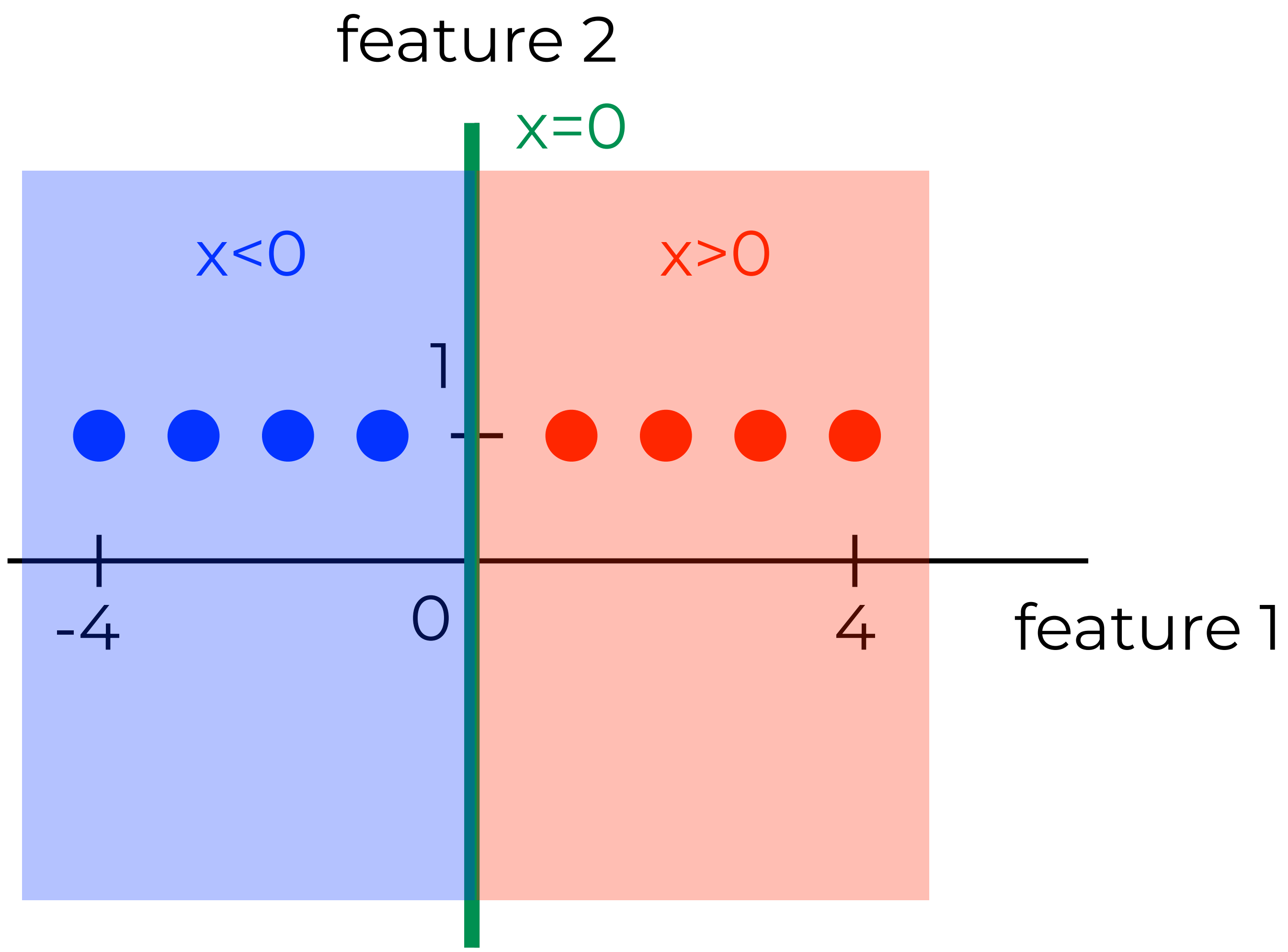


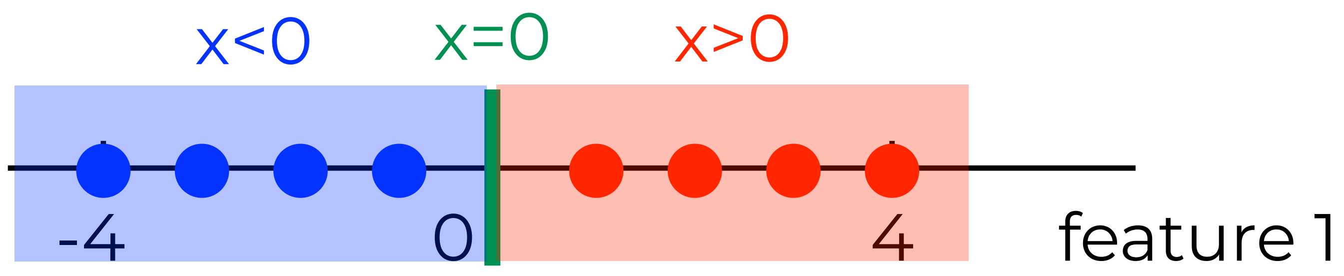
low-dim space

data analysis in lower dimensional space

A Simple Classification Example

	feature 1	feature 2
sample 1	-4	1
sample 2	-3	1
sample 3	-2	1
sample 4	-1	1
sample 5	1	1
sample 6	2	1
sample 7	3	1
sample 8	4	1

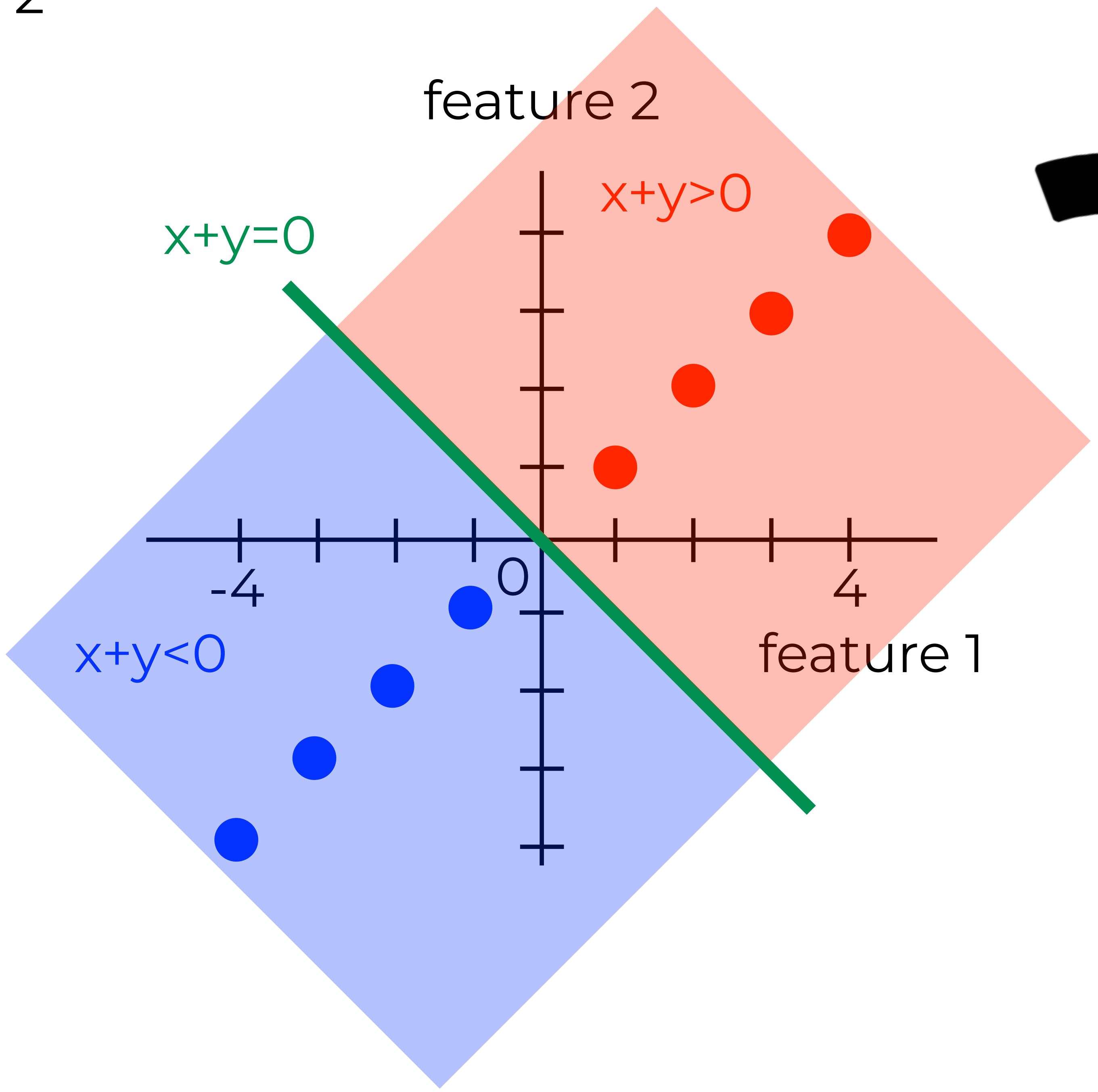


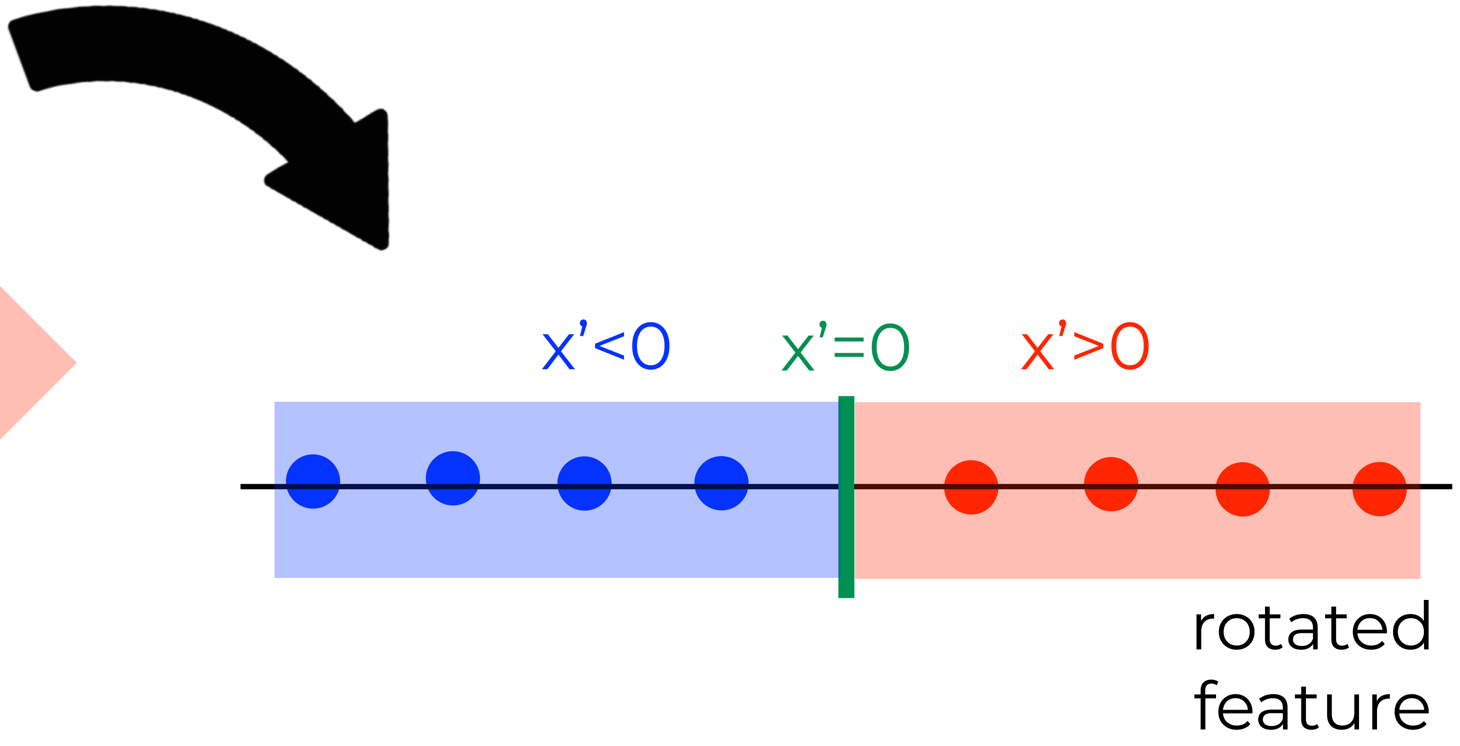


A Simple Classification Example

Original data

	feature 1	feature 2
sample 1	-4	-4
sample 2	-3	-3
sample 3	-2	-2
sample 4	-1	-1
sample 5	1	1
sample 6	2	2
sample 7	3	3
sample 8	4	4

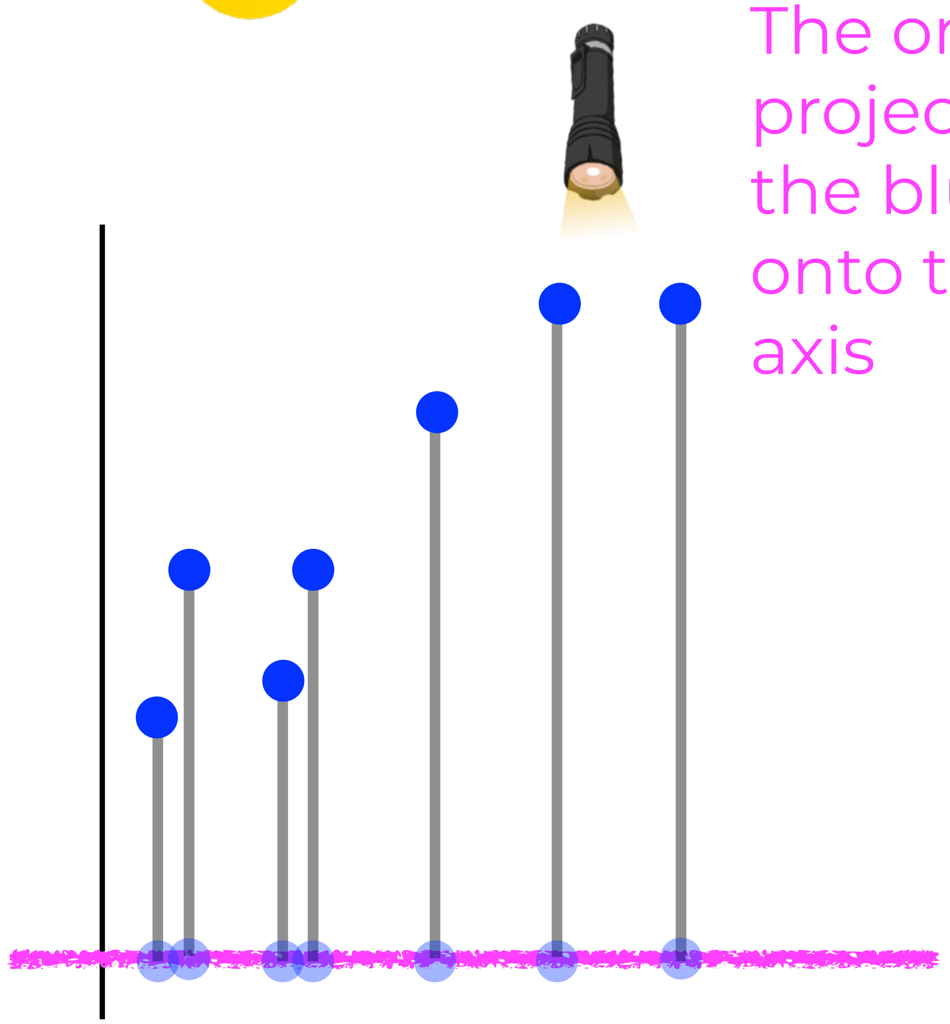




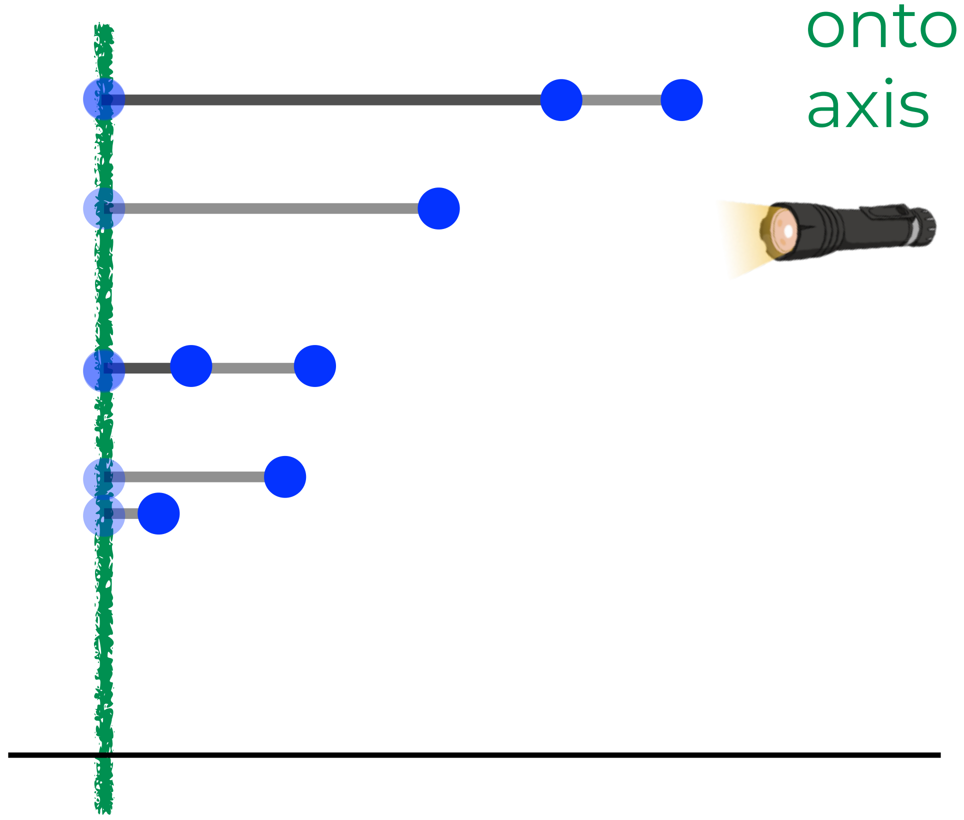
Brain Code Camp

Projection

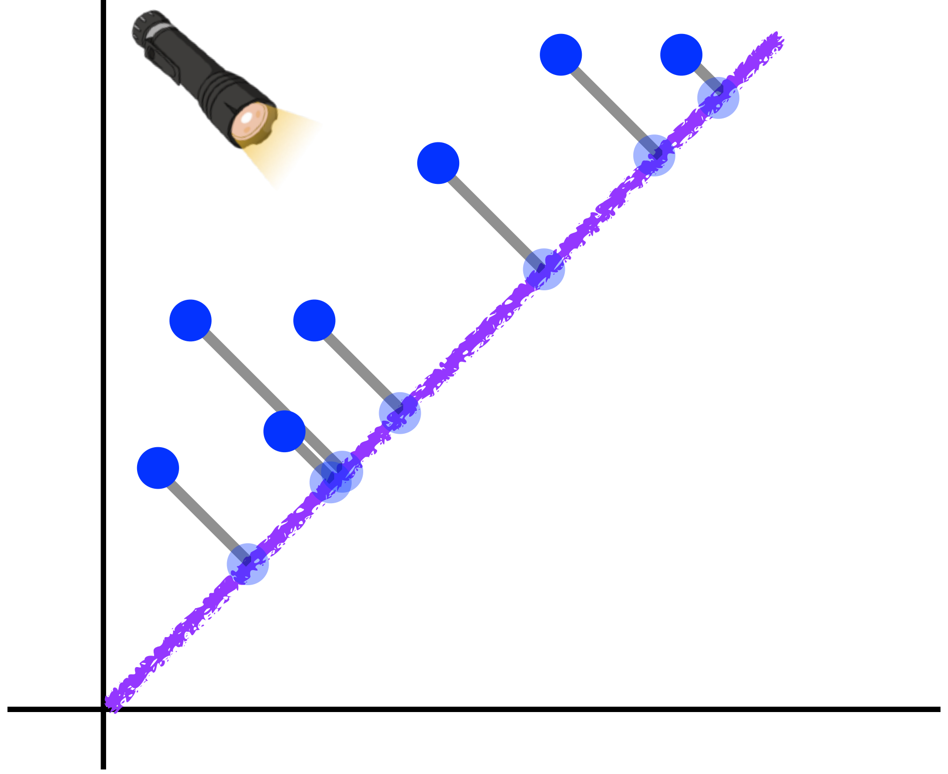
The orthogonal projection of the blue points onto the pink axis



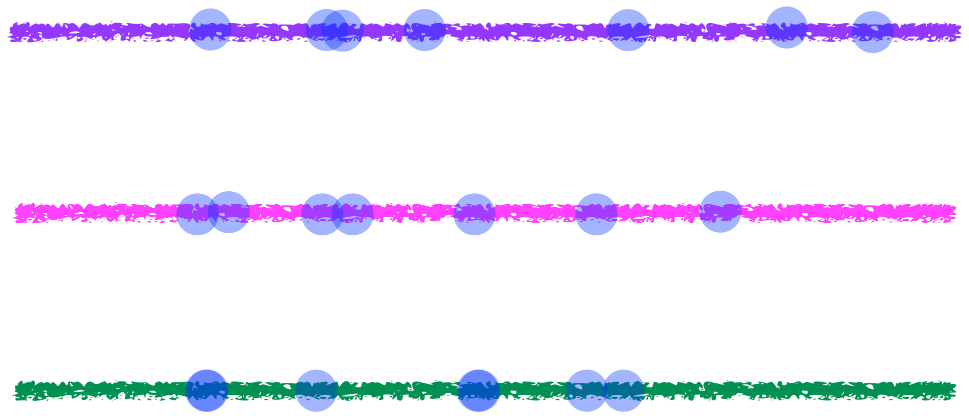
The orthogonal projection of the blue points onto the green axis



The orthogonal projection of the blue points onto the purple axis



How spread out the projected points are

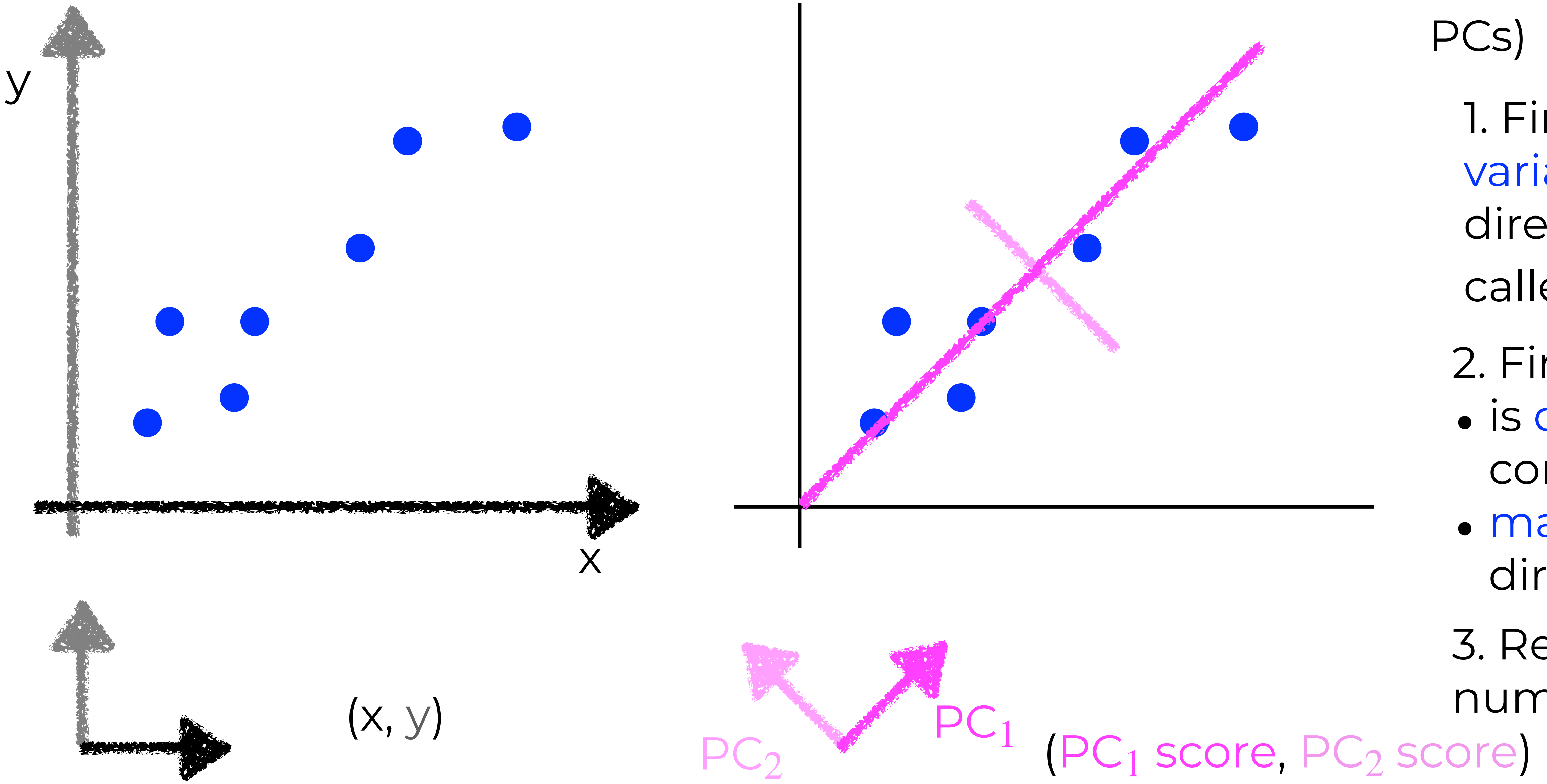


higher

lower

Principal Component Analysis (PCA)

Original space



How much you move in the x-direction

How much you move in the y-direction

How much you move in the PC_1 direction

How much you move in the PC_2 direction

For d-dimensional data, PCA can find up to d directions (principal components or PCs)

1. Find the direction that maximizes the variance of the projected points. This direction can be described by a vector called the first principal component (PC_1)
2. Find the next PC that
   - is orthogonal to the PCs already considered
   - maximizes the variance along the new direction
3. Repeat step 2 until you have the desired number of PCs

Principal Component Analysis (PCA)

sklearn.decomposition.PCA

class `sklearn.decomposition.`**`PCA`**(*n_components=None, *, copy=True, whiten=False, svd_solver='auto', tol=0.0, iterated_power='auto', n_oversamples=10, power_iteration_normalizer='auto', random_state=None*) [source]

Attributes:

components_ : *ndarray of shape (n_components, n_features)*

Principal axes in feature space, representing the directions of maximum variance in the data. Equivalently, the right singular vectors of the centered input data, parallel to its eigenvectors. The components are sorted by decreasing `explained_variance_`.

PC_1, PC_2, ...

explained_variance_ : *ndarray of shape (n_components,)*

The amount of variance explained by each of the selected components. The variance estimation uses `n_samples - 1` degrees of freedom.

Equal to n_components largest eigenvalues of the covariance matrix of X.

New in version 0.18.

λ_1, λ_2, ...

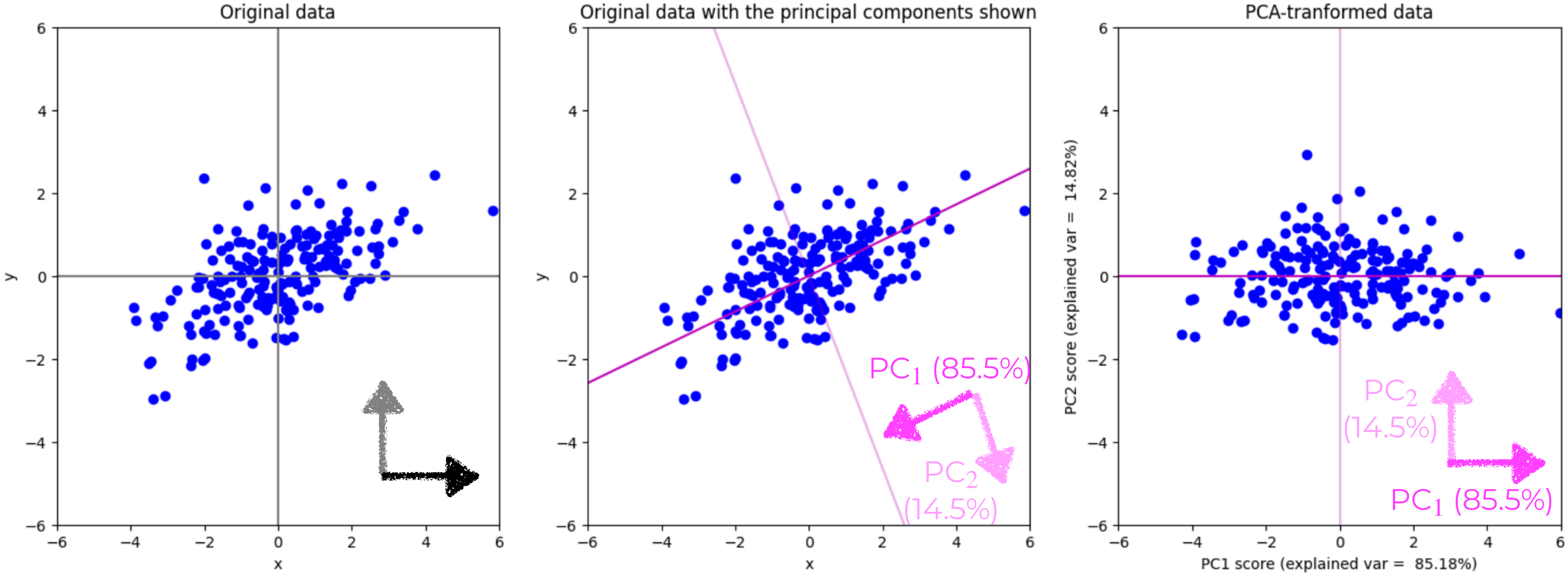
explained_variance_ratio_ : *ndarray of shape (n_components,)*

Percentage of variance explained by each of the selected components.

If `n_components` is not set then all components are stored and the sum of the ratios is equal to 1.0.

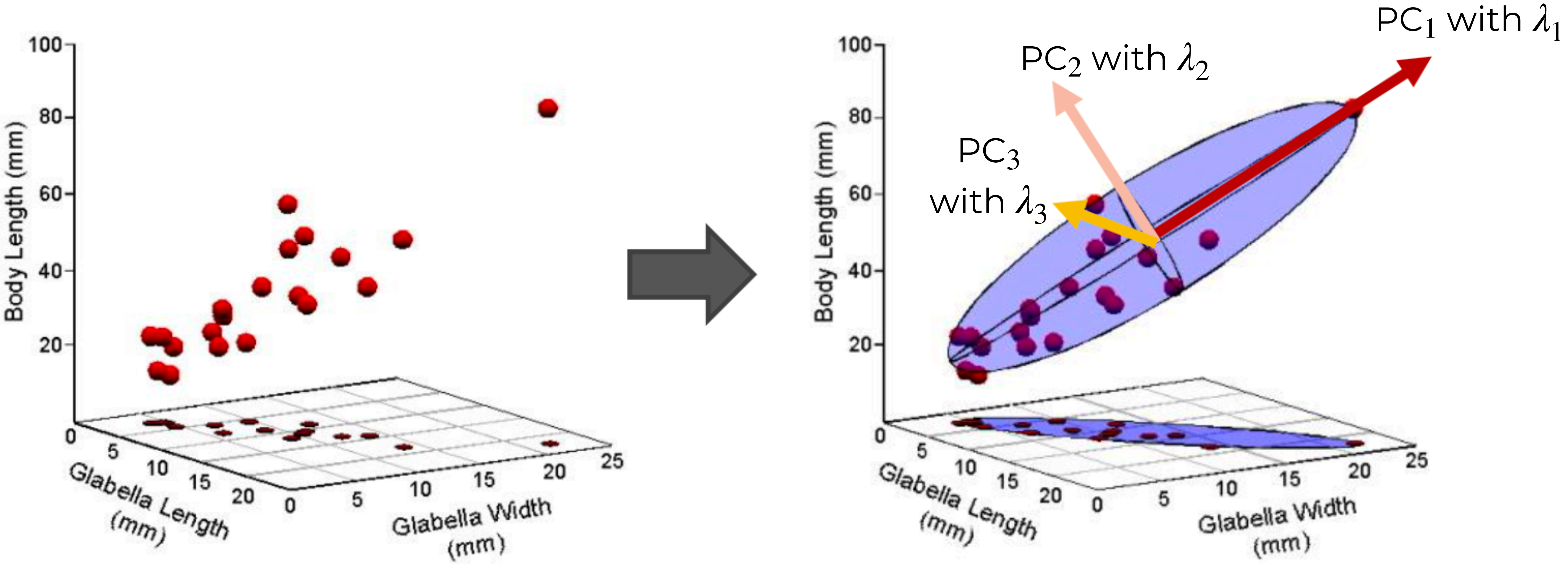
ex. 0.8, 0.2, ...

PCA in 2D



Original data
y
x
Original data with the principal components shown
y
x
PC1 (85.5%)
PC2 (14.5%)
PCA-tranformed data
PC2 score (explained var = 14.82%)
PC1 score (explained var = 85.18%)
PC2 (14.5%)
PC1 (85.5%)

PCA in 3D



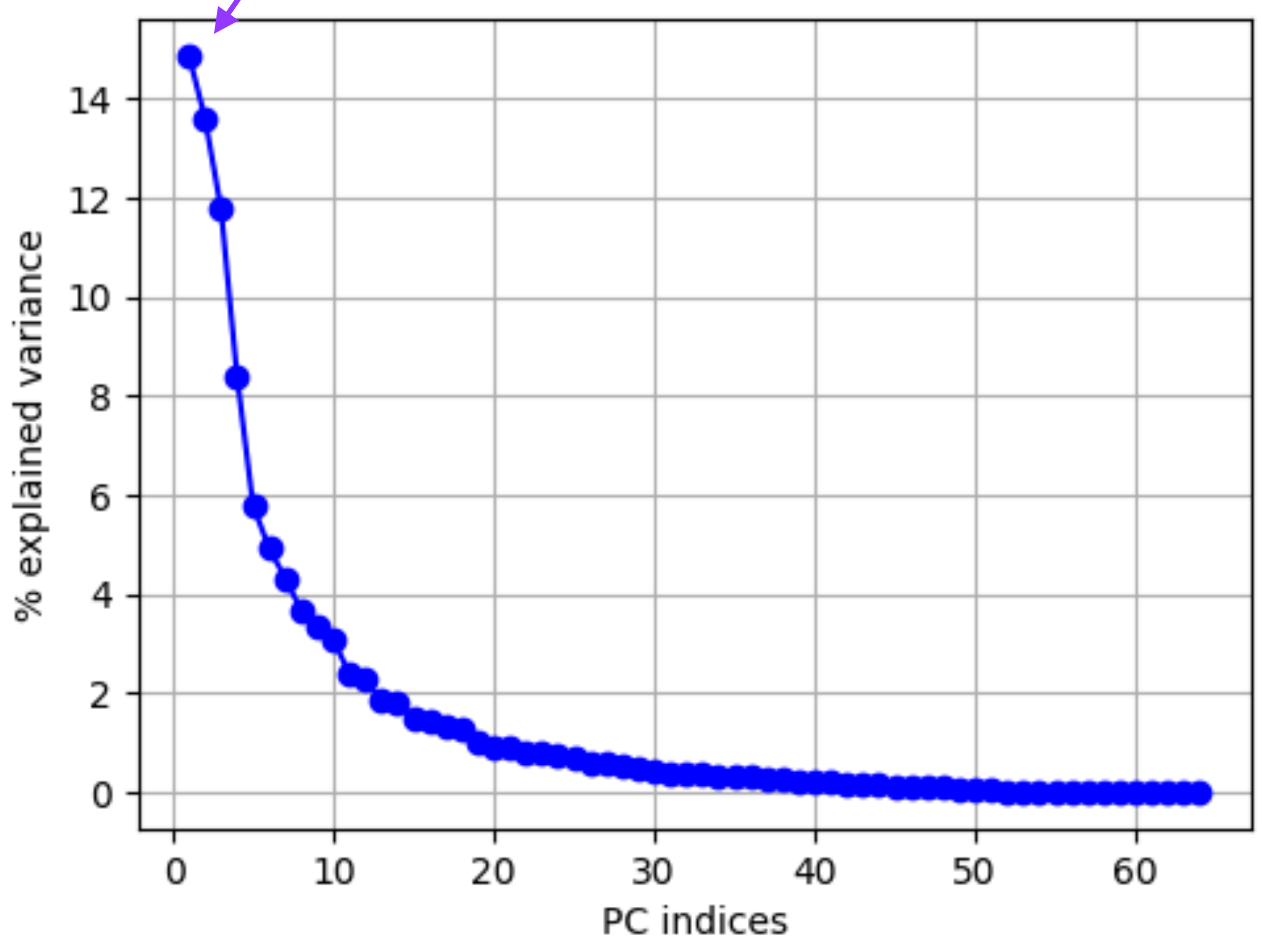
Source: the paleontological association

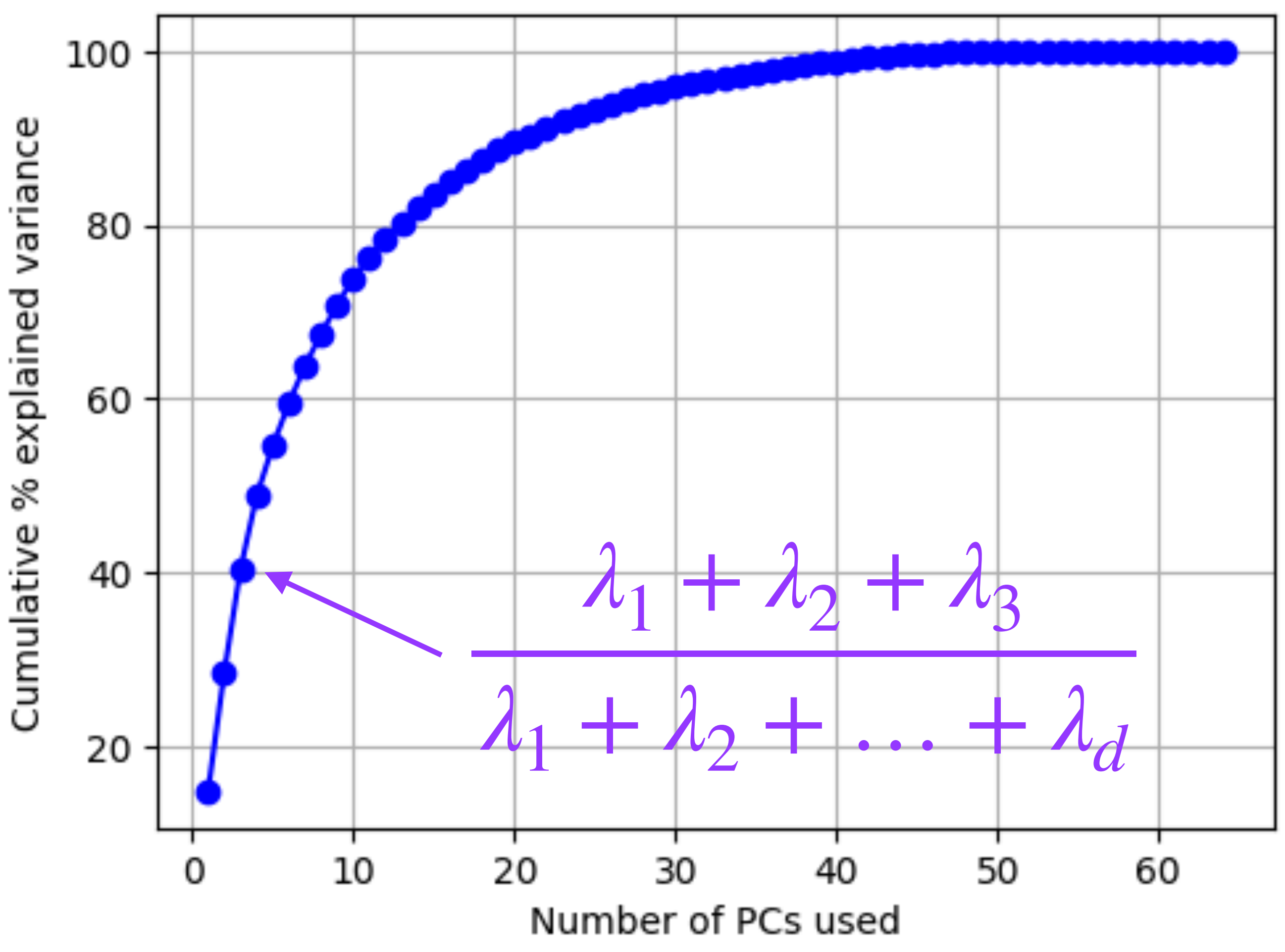
PCA in High-dimensional Space

d-dimensional space d principal components (PCs) $PC_1, \; PC_2 \; , \ldots , \; PC_d$

$$\frac{\lambda_1}{\lambda_1 + \lambda_2 + \ldots + \lambda_d}$$

explained variances $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$

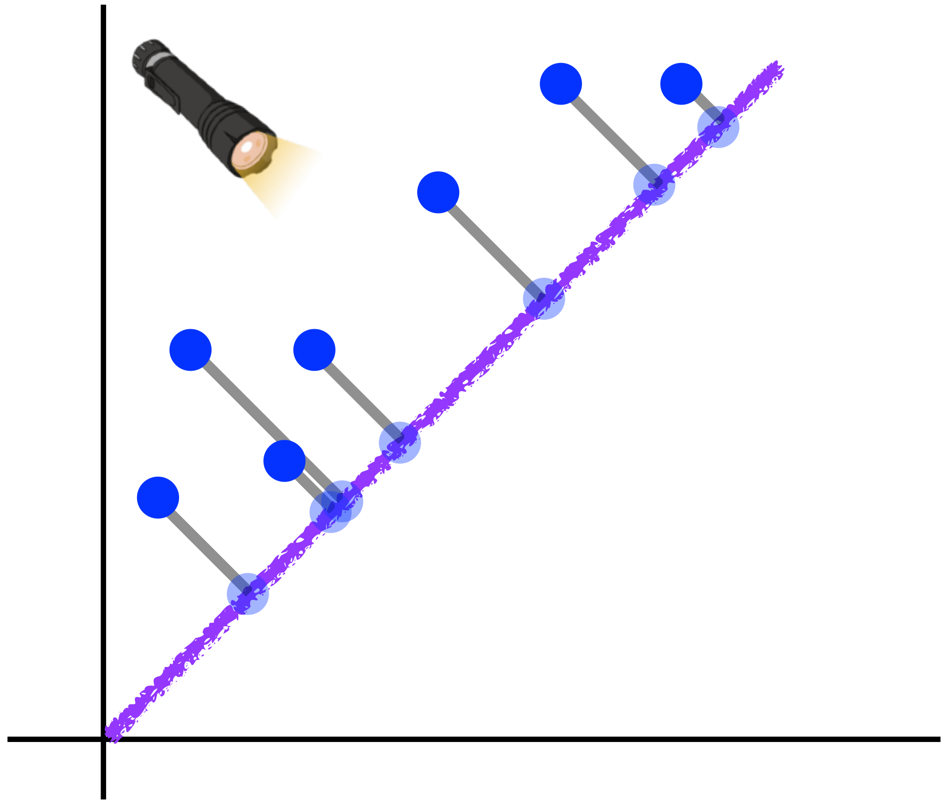




$$\frac{\lambda_1 + \lambda_2 + \lambda_3}{\lambda_1 + \lambda_2 + \ldots + \lambda_d}$$

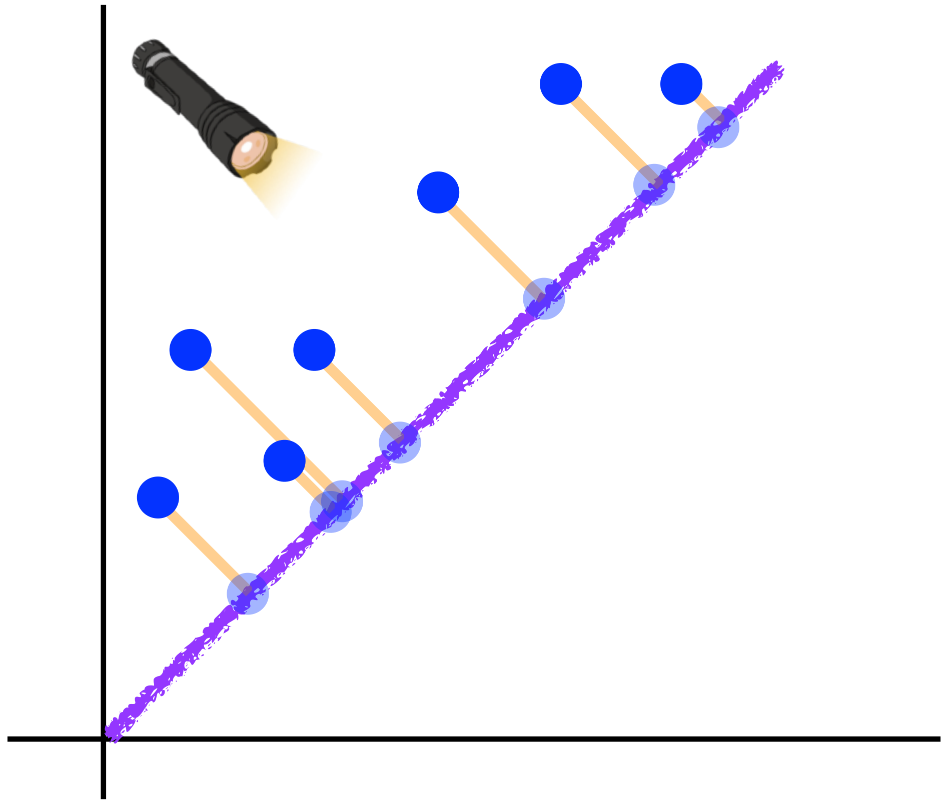
Two Common Definitions of PCA

Maximum variance formulation

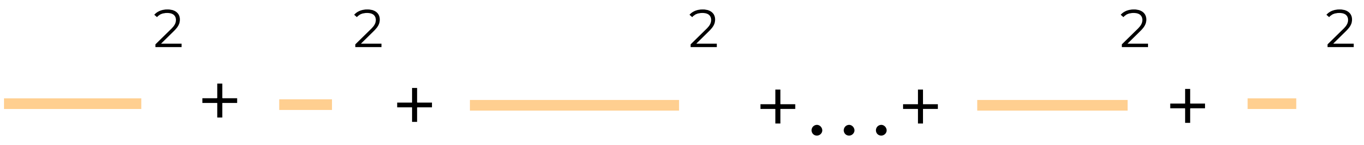


Maximizes the variance of the projected points

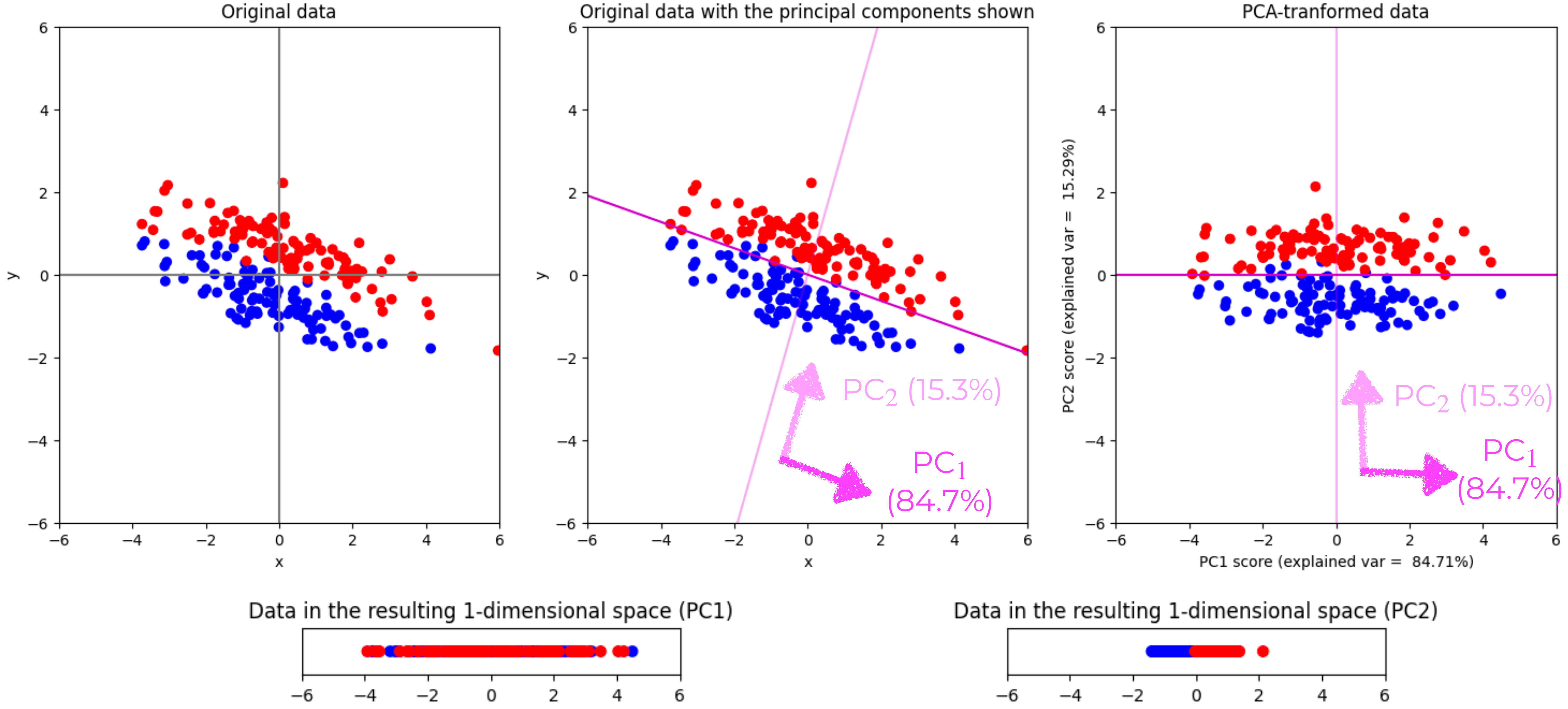
Minimum-error formulation



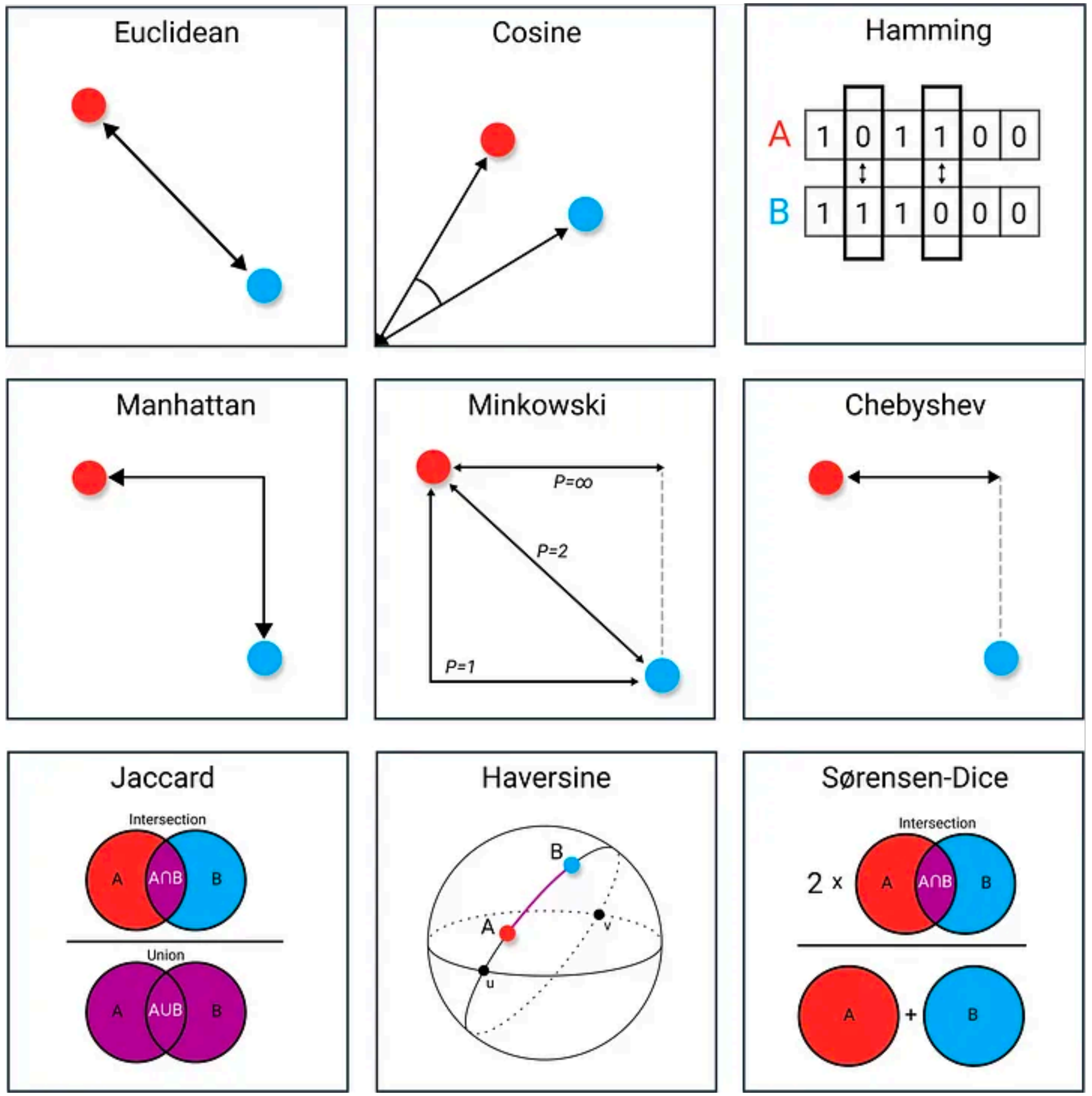
Minimizes the sum-of-squares of the projection errors



PCA Example

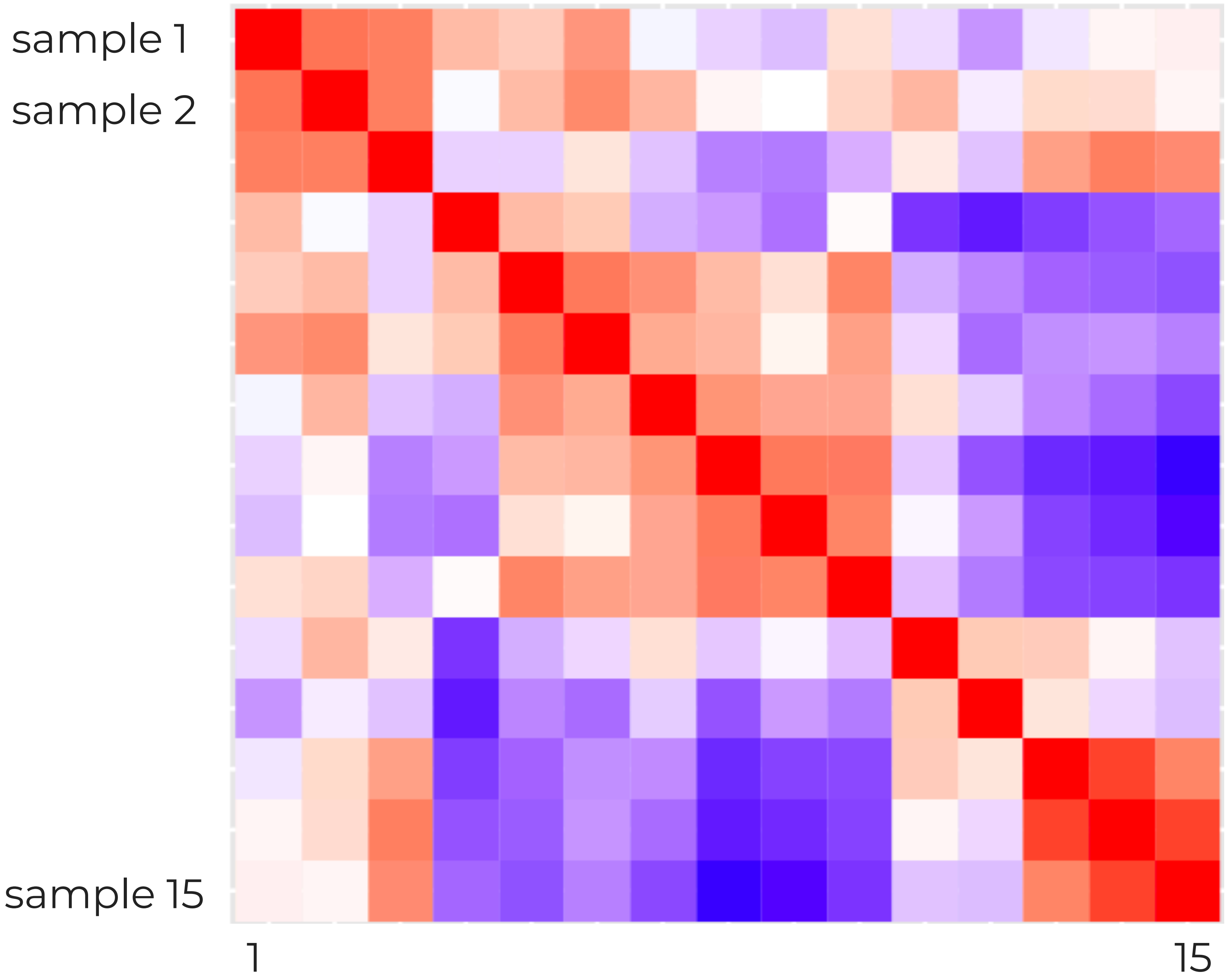

Original data
y
x
Original data with the principal components shown
y
x
PC₂ (15.3%)
PC₁
(84.7%)
PCA-tranformed data
PC2 score (explained var = 15.29%)
PC1 score (explained var = 84.71%)
PC₂ (15.3%)
PC₁
(84.7%)
Data in the resulting 1-dimensional space (PC1)
Data in the resulting 1-dimensional space (PC2)

Distance Measures



https://towardsdatascience.com/9-distance-measures-in-data-science-918109d069fa

Pairwise Distance Matrix



The element at row i and column j represents the distance between sample i and sample j

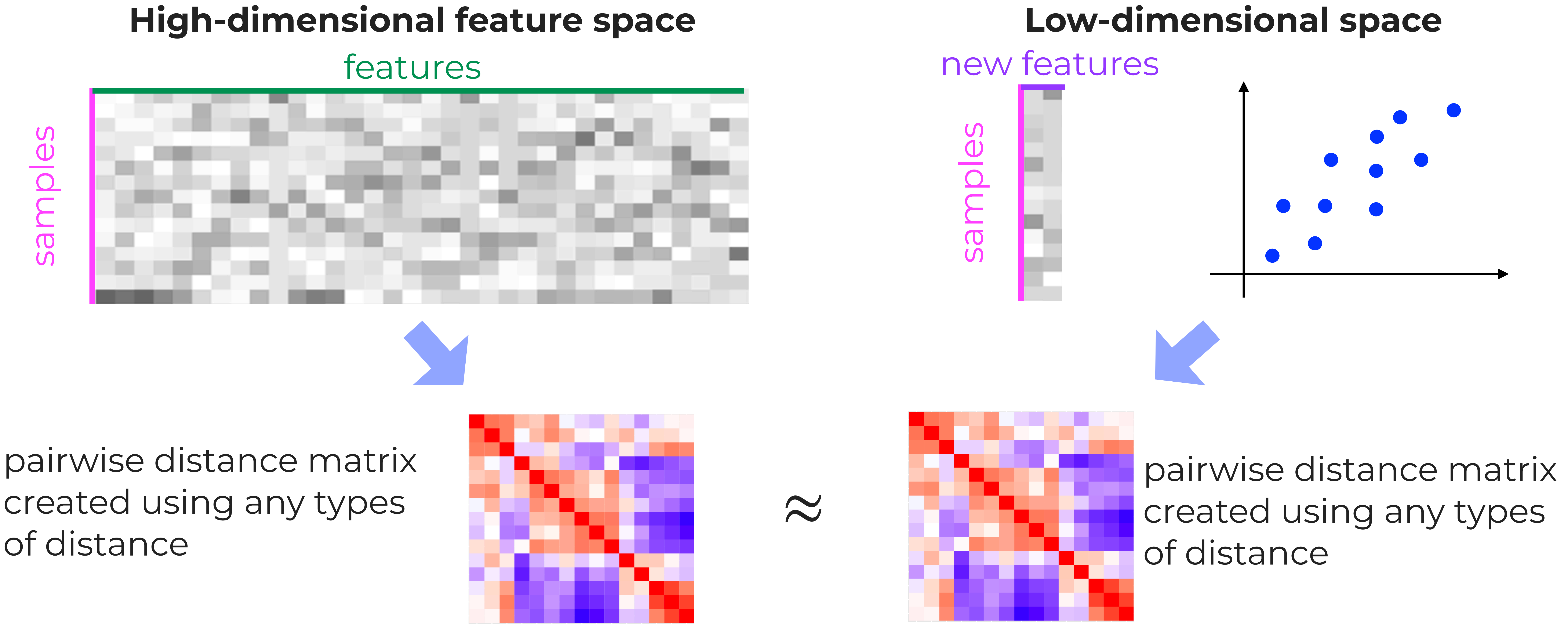
$$D_{ij} = D_{ji}$$

$$D_{ii} = 0$$

Modified from https://www.datanovia.com/en/lessons/clustering-distance-measures/

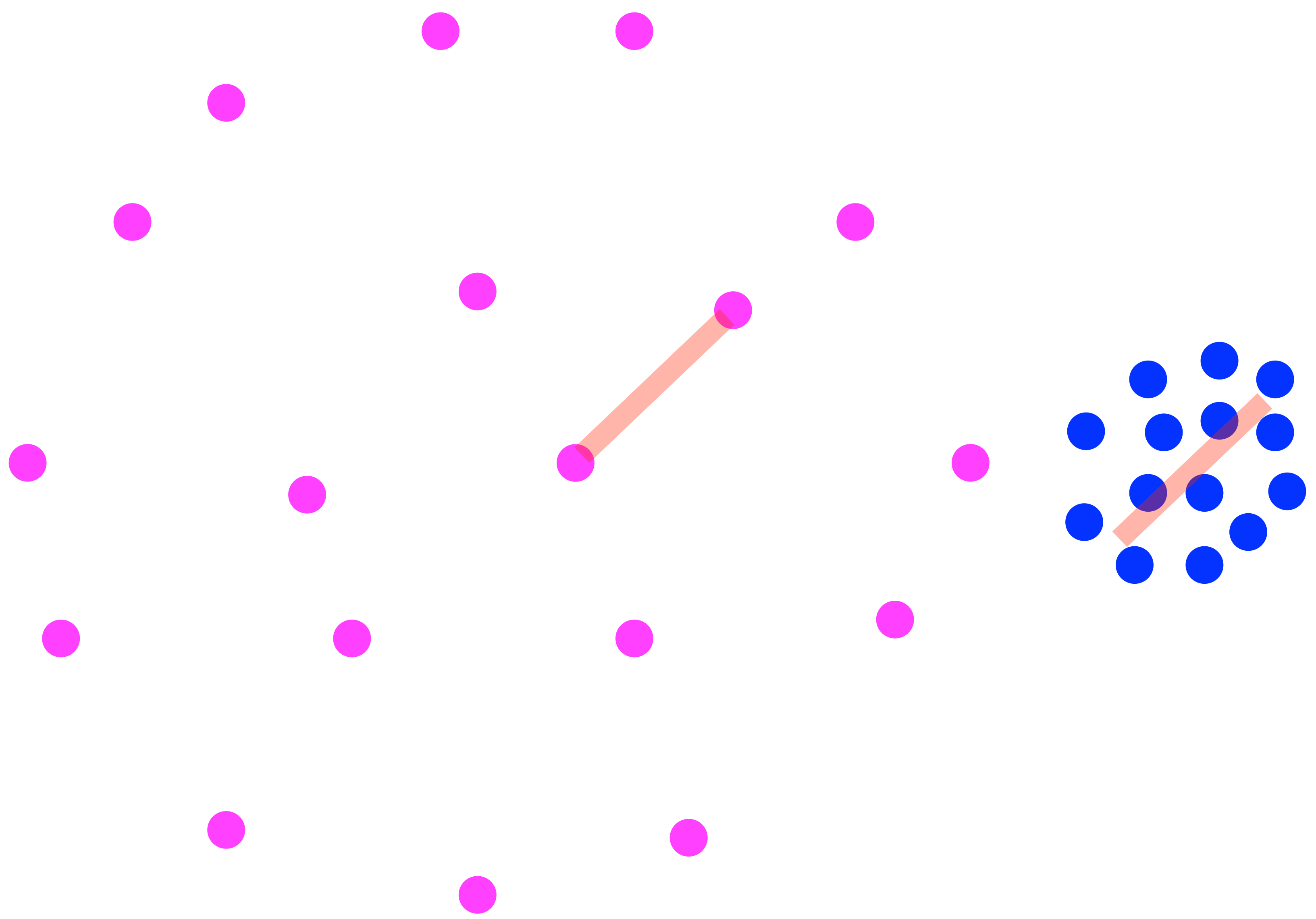
Brain Code Camp

Generalized Multi-Dimensional Scaling (MDS)



Brain
Code
Camp

Beyond MDS



Should we use distance metrics that take into account data density?

Brain Code Camp

Popular Nonlinear Dimensionality Reduction Methods

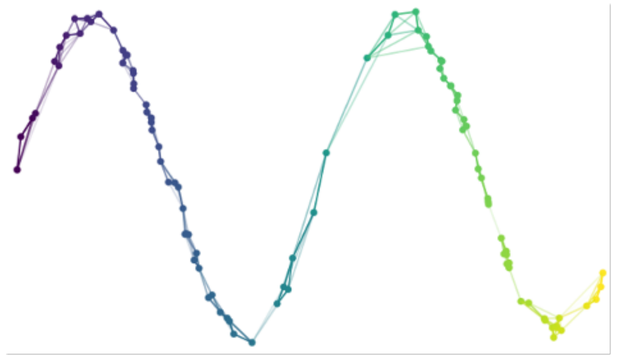
t-distributed Stochastic Neighbor Embedding (t-SNE)

sklearn.manifold.TSNE

class `sklearn.manifold.TSNE`(*n_components=2, *, perplexity=30.0, early_exaggeration=12.0, learning_rate='auto', n_iter=1000, n_iter_without_progress=300, min_grad_norm=1e-07, metric='euclidean', metric_params=None, init='pca', verbose=0, random_state=None, method='barnes_hut', angle=0.5, n_jobs=None*) [source]

Uniform Manifold Approximation and Projection (UMAP)

class `umap.umap_.UMAP`(*n_neighbors=15, n_components=2, metric='euclidean', metric_kwds=None, output_metric='euclidean', output_metric_kwds=None, n_epochs=None, learning_rate=1.0, init='spectral', min_dist=0.1, spread=1.0, low_memory=True, n_jobs=-1, set_op_mix_ratio=1.0, local_connectivity=1.0, repulsion_strength=1.0, negative_sample_rate=5, transform_queue_size=4.0, a=None, b=None, random_state=None, angular_rp_forest=False, target_n_neighbors=-1, target_metric='categorical', target_metric_kwds=None, target_weight=0.5, transform_seed=42, transform_mode='embedding', force_approximation_algorithm=False, verbose=False, unique=False, densmap=False, dens_lambda=2.0, dens_frac=0.3, dens_var_shift=0.1, output_dens=False, disconnection_distance=None*) [source]

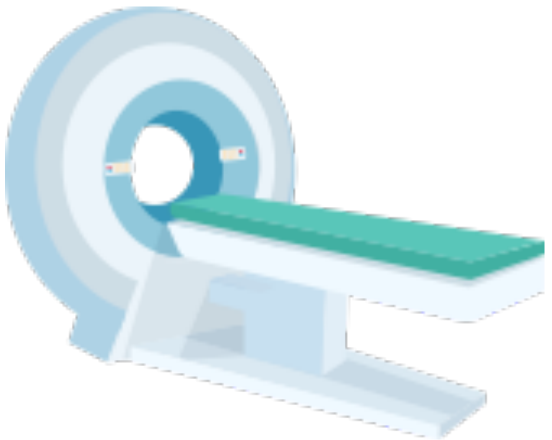


Single-Channel MRI

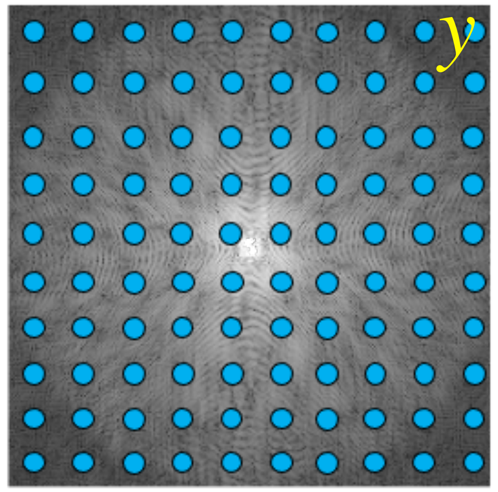
Acquired k-space data

Reconstructed data

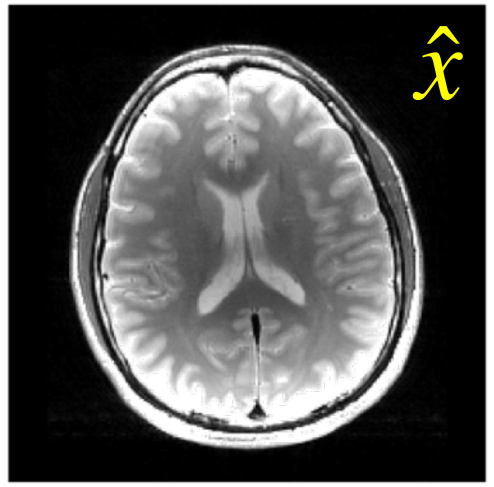
A

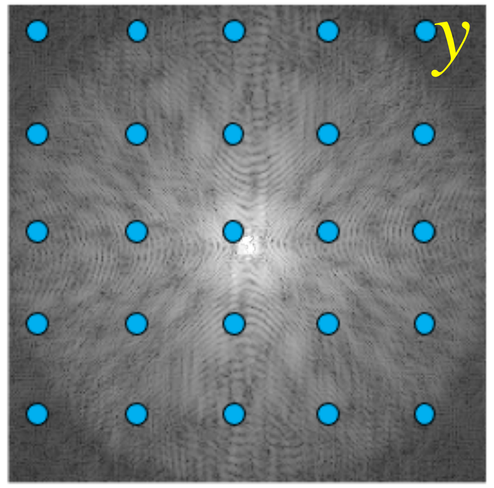


MR acquisition approximated using a linear operator

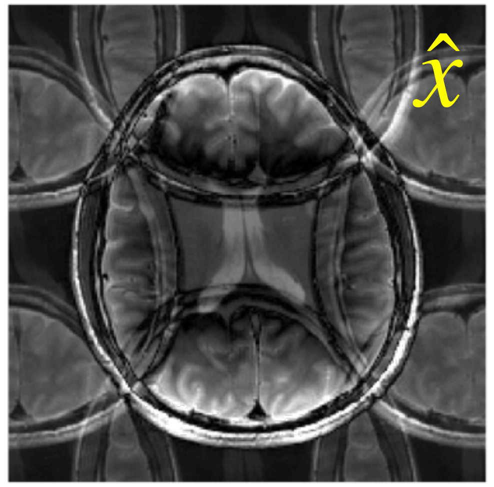


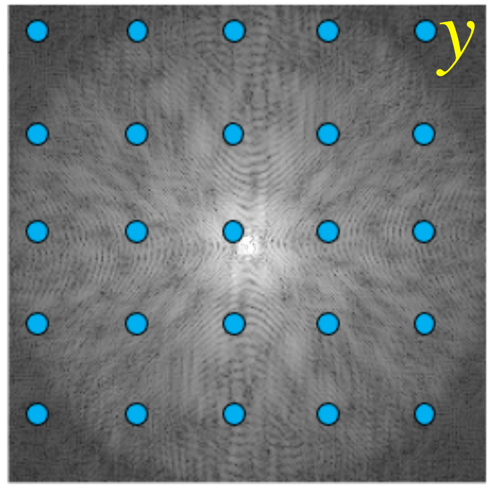
$$\hat{x} = \arg\min_{x} \frac{1}{2}\|Ax - y\|_2^2$$





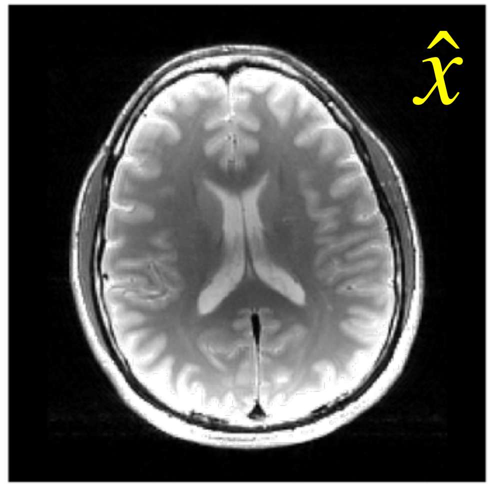
$$\hat{x} = \arg\min_{x} \frac{1}{2}\|Ax - y\|_2^2$$





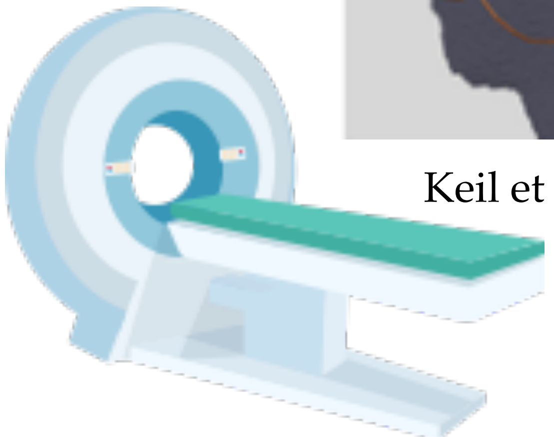
$$\hat{x} = \arg\min_{x} \frac{1}{2}\|Ax - y\|_2^2 + \lambda R(x)$$

Longer reconstruction time



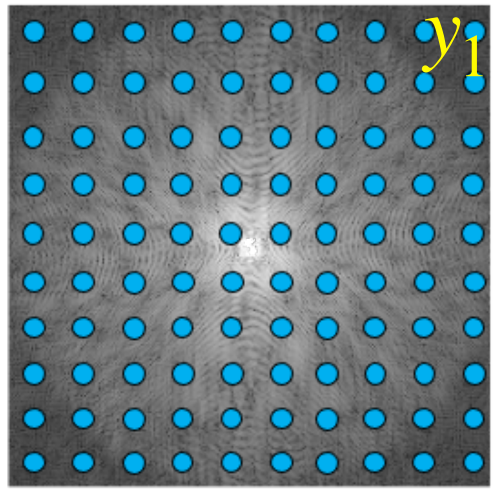
Multi-Channel MRI

Keil et al. JMR (2013)

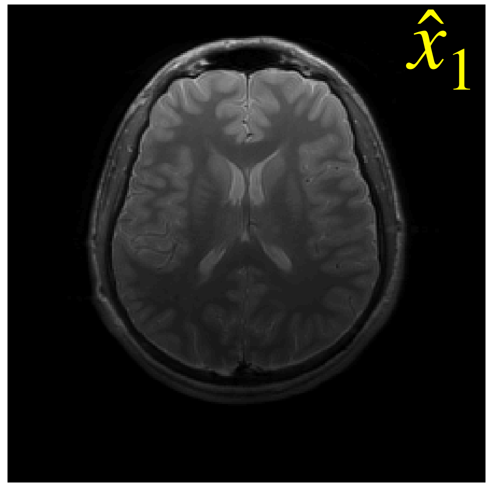


MR acquisition
approximated using
a linear operator

channel 1



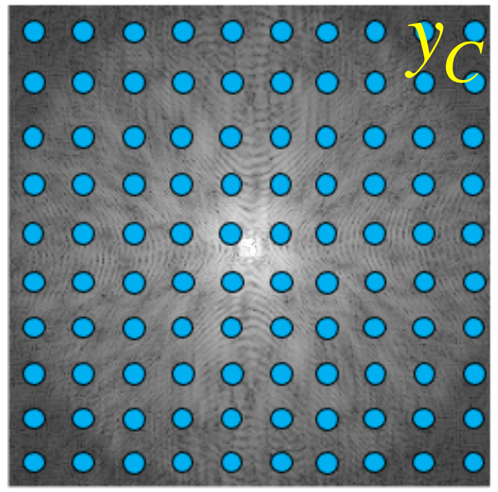
Reconstructed data

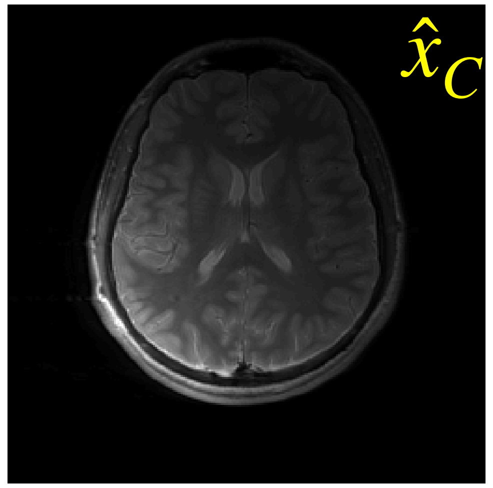


⋮

⋮

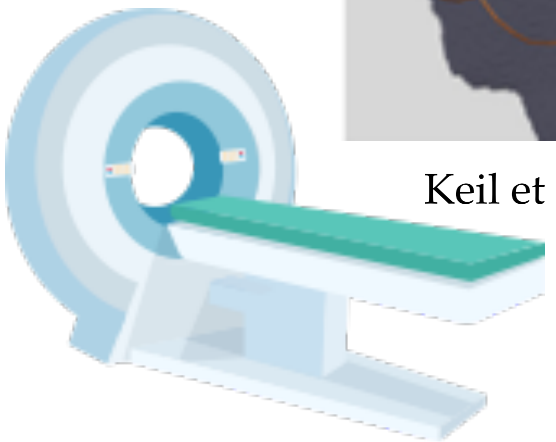
channel C

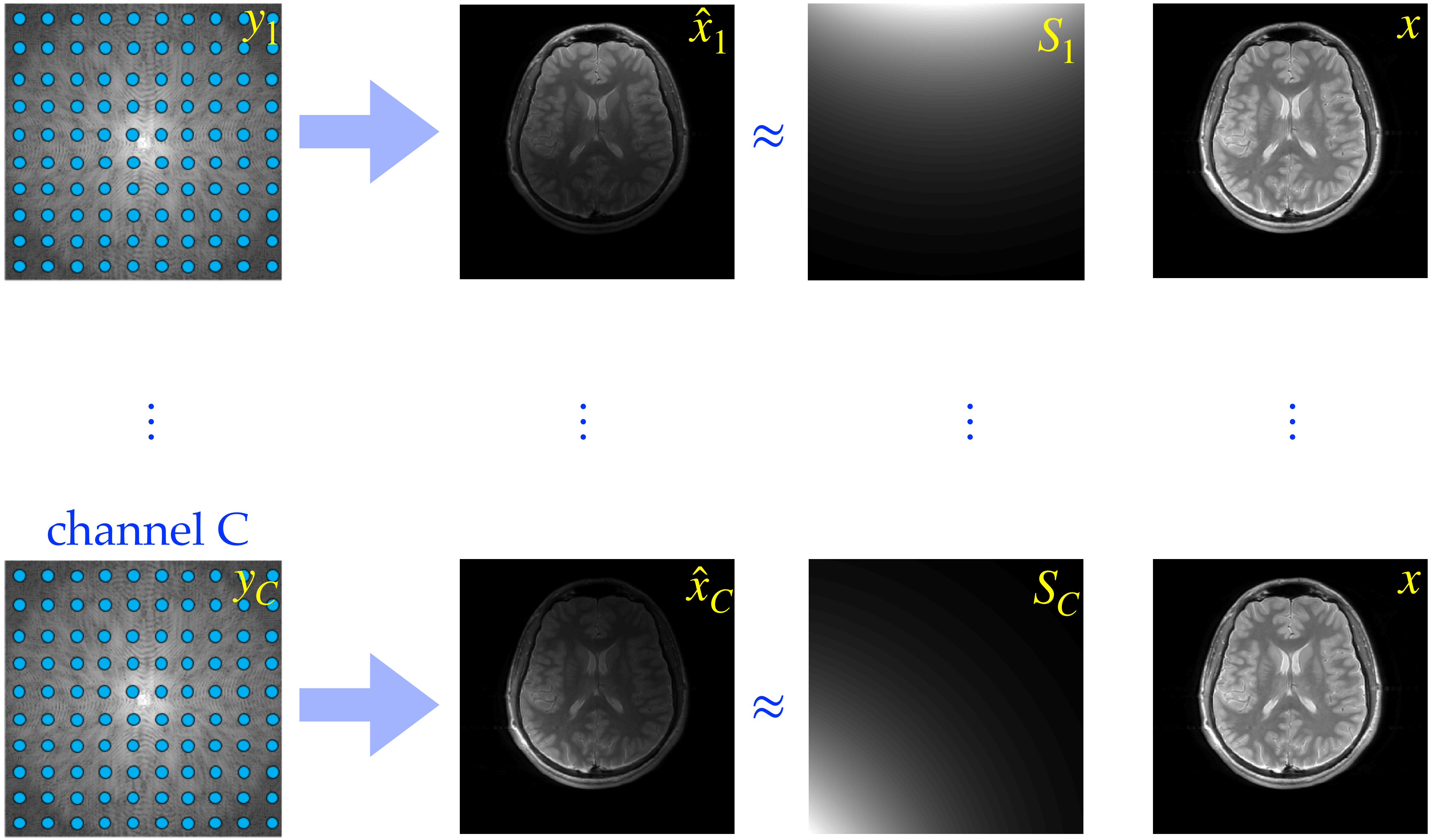




Multi-Channel MRI

Keil et al. JMR (2013)



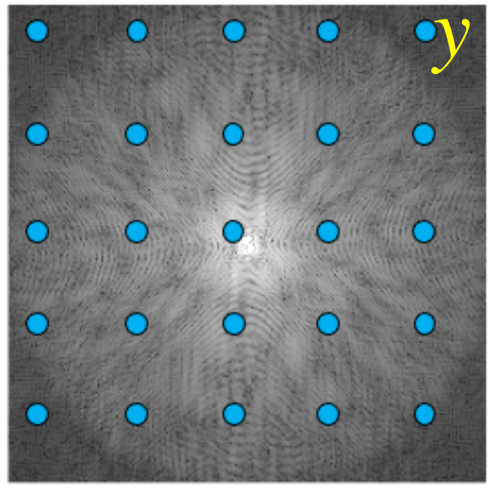


MR acquisition approximated using a linear operator

$A \rightarrow AS_c$

Multi-Channel MRI

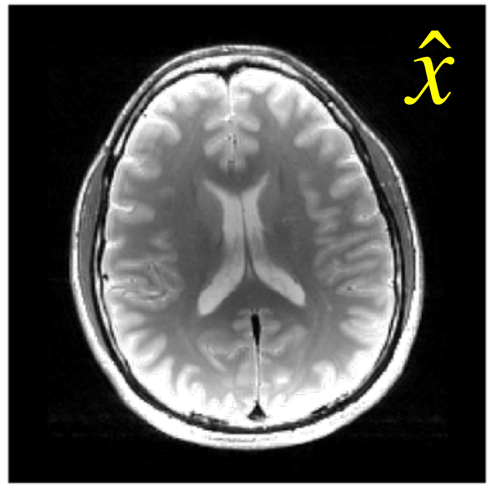
single-channel data



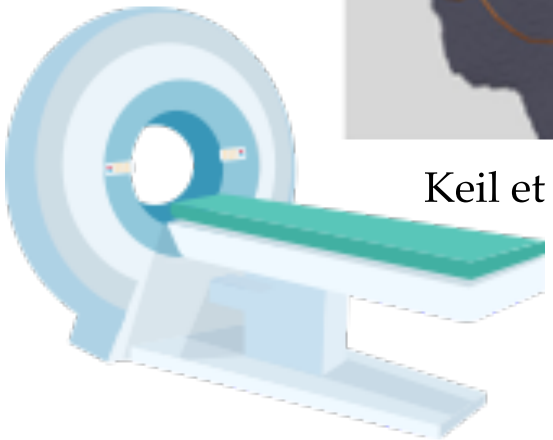
$$\min_x \frac{1}{2}\|Ax - y\|_2^2 + \lambda R(x)$$

single-channel reconstruction

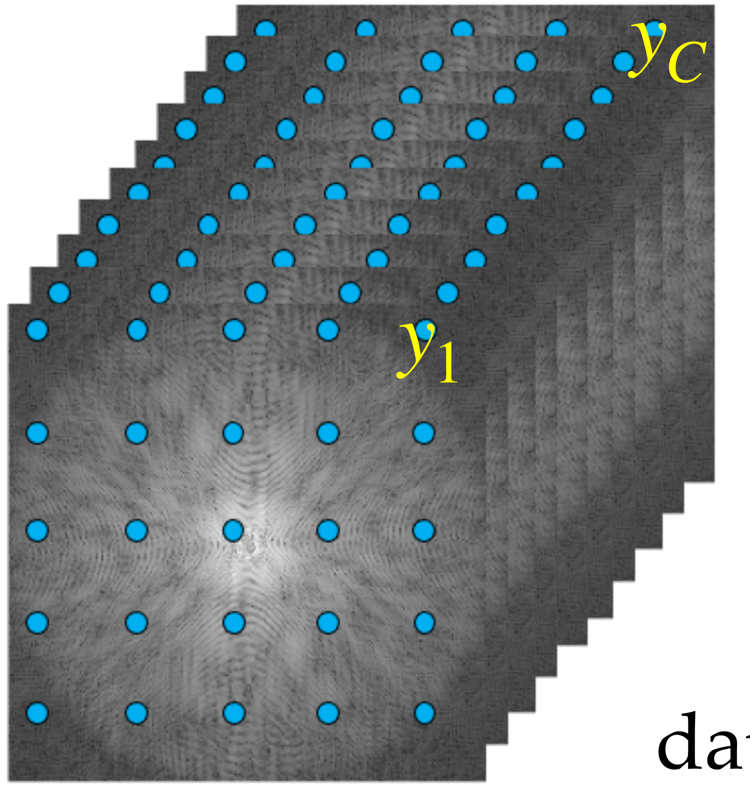
Reconstructed data



Keil et al. JMR (2013)

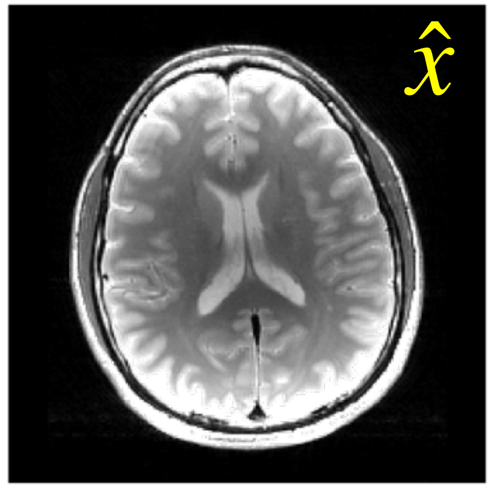


multi-channel data



$$\min_x \frac{1}{2}\sum_c \|AS_c x - y_c\|_2^2 + \lambda \tilde{R}(x)$$

multi-channel reconstruction



MR acquisition approximated using a linear operator

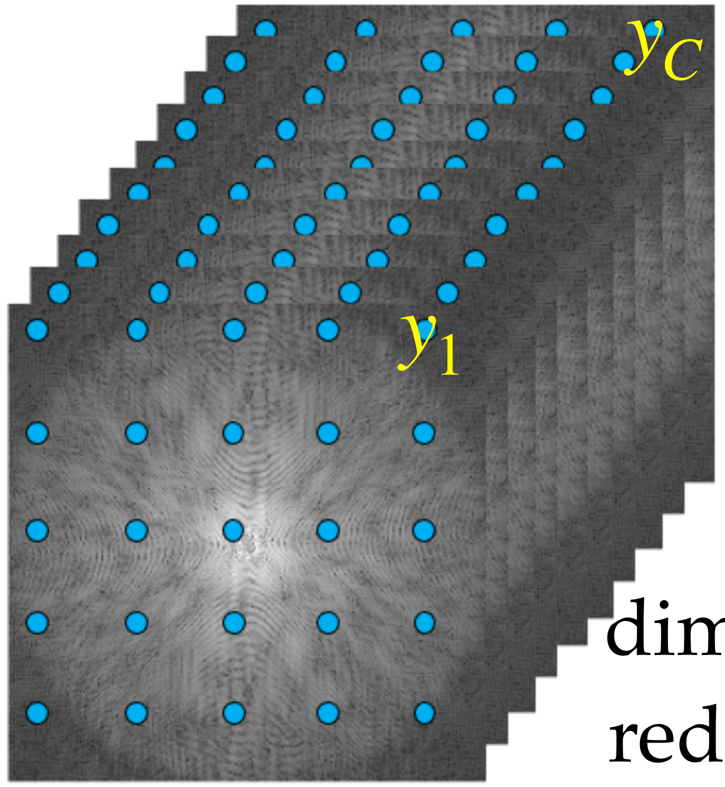
A → AS_c

data dimension: 512 x 512 x 64 x 64 x T

row col slice channels time points

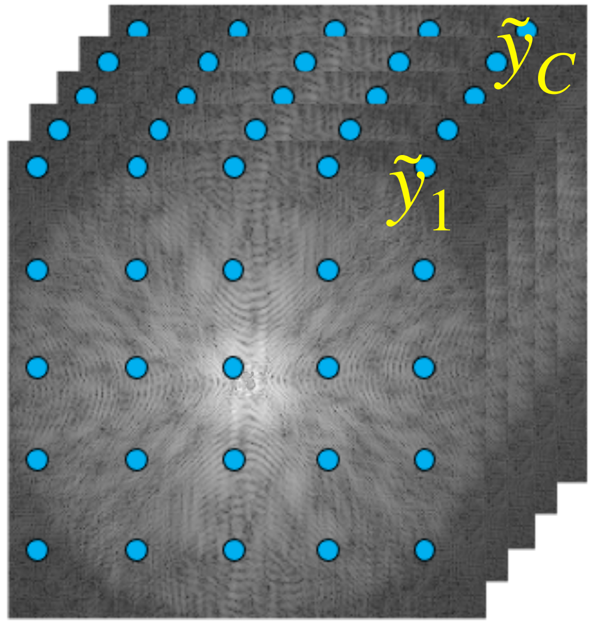
Multi-Channel MRI with Virtual Coils

multi-channel data



dimensionality reduction such as PCA

k-space data with fewer channels

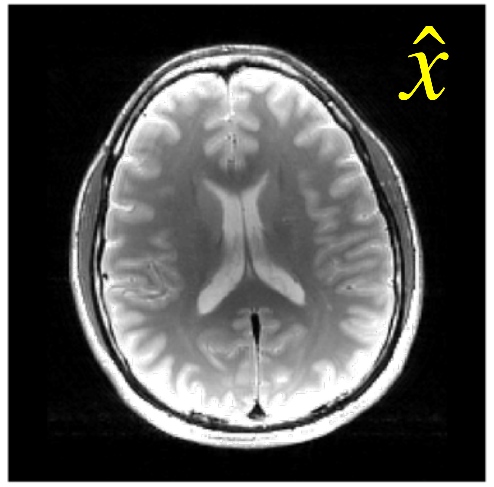


64 physical channels
→ 10 virtual channels

$$\min_x \frac{1}{2} \sum_c \|AS_c x - y_c\|_2^2 + \lambda \tilde{R}(x)$$

$$\min_x \frac{1}{2} \sum_c \|A\tilde{S}_c x - \tilde{y}_c\|_2^2 + \lambda \tilde{R}(x)$$

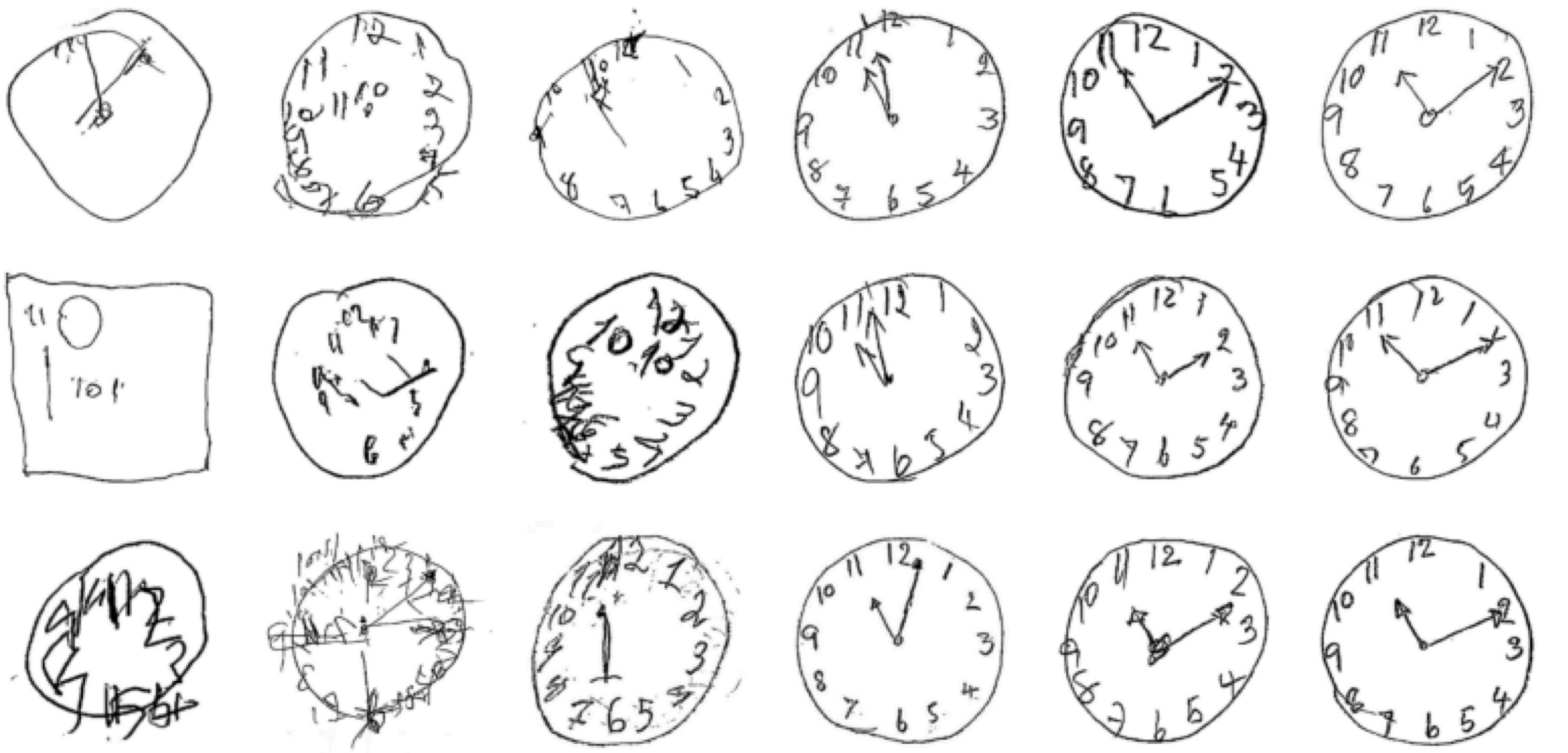
multi-channel reconstruction



Reduced computational resources

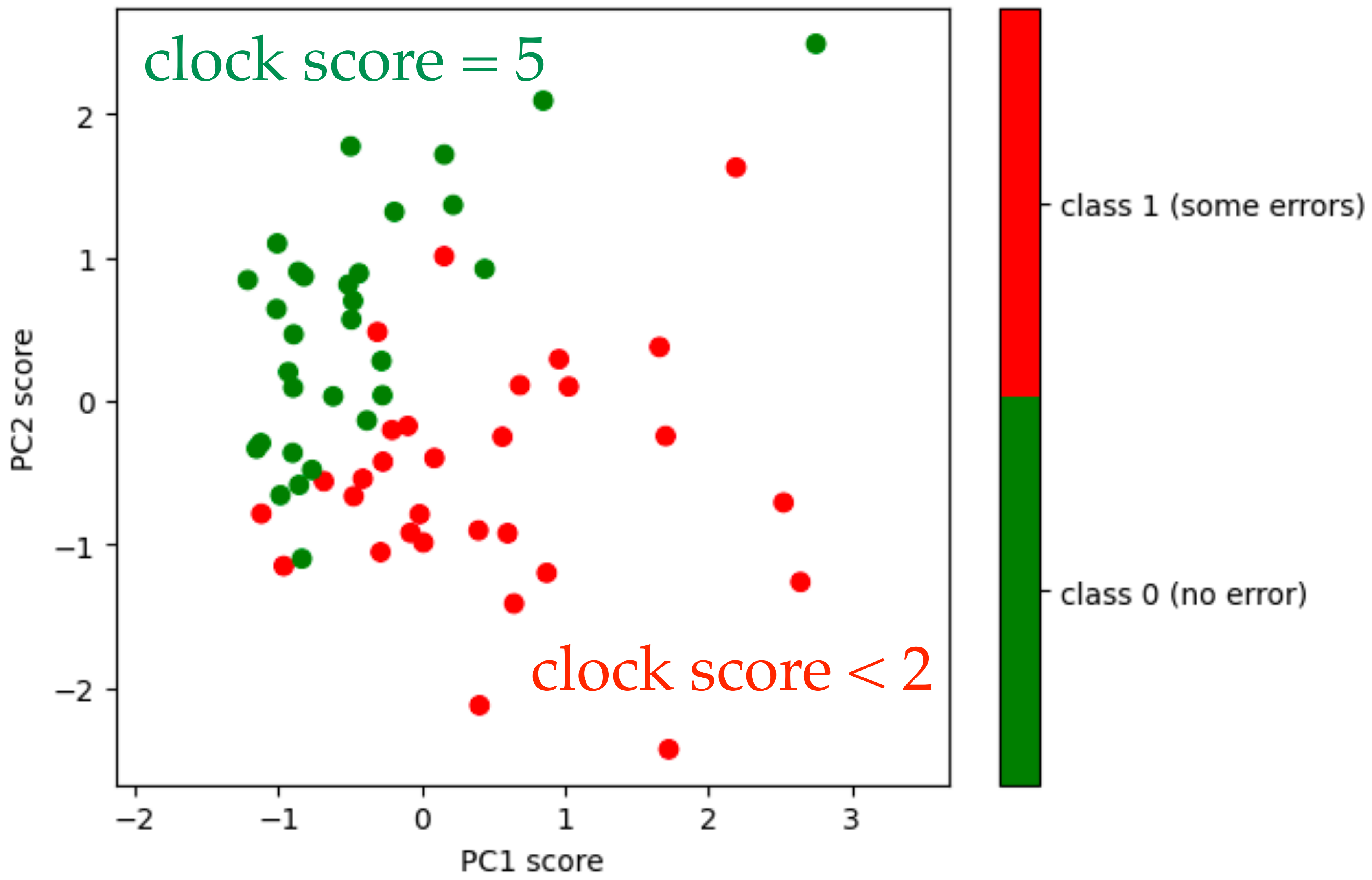
Clock Drawing Test

Score 0 (n=13) | Score 1 (n=20) | Score 2 (n=53) | Score 3 (n=352) | Score 4 (n=1047) | Score 5 (n=1623)



Clock Drawing Test

PCA



Ruengchaijatuporn, Natthanan, et al. "An explainable self-attention deep neural network for detecting mild cognitive impairment using multi-input digital drawing tasks." Alzheimer's Research & Therapy 14.1 (2022): 1-11.

Contrastive learning + deep learning → UMAP

Raksasat, Raksit, et al. "Attentive Pairwise Interaction Network for AI-assisted Clock Drawing Test Assessment of Early Visuospatial Deficits." *Available at SSRN 4327538*.