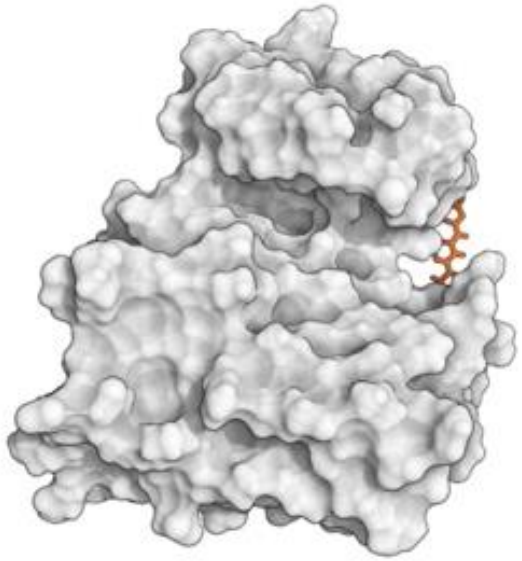
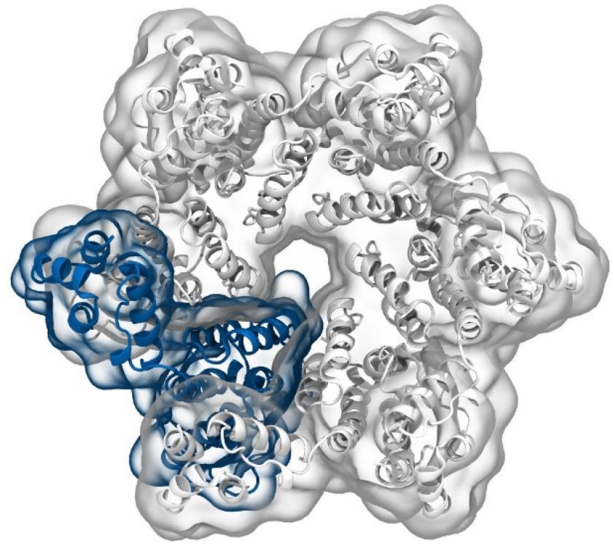
Simulation of Biomolecules



Dimensionality Reduction



Dr Matteo Degiacomi

Durham University

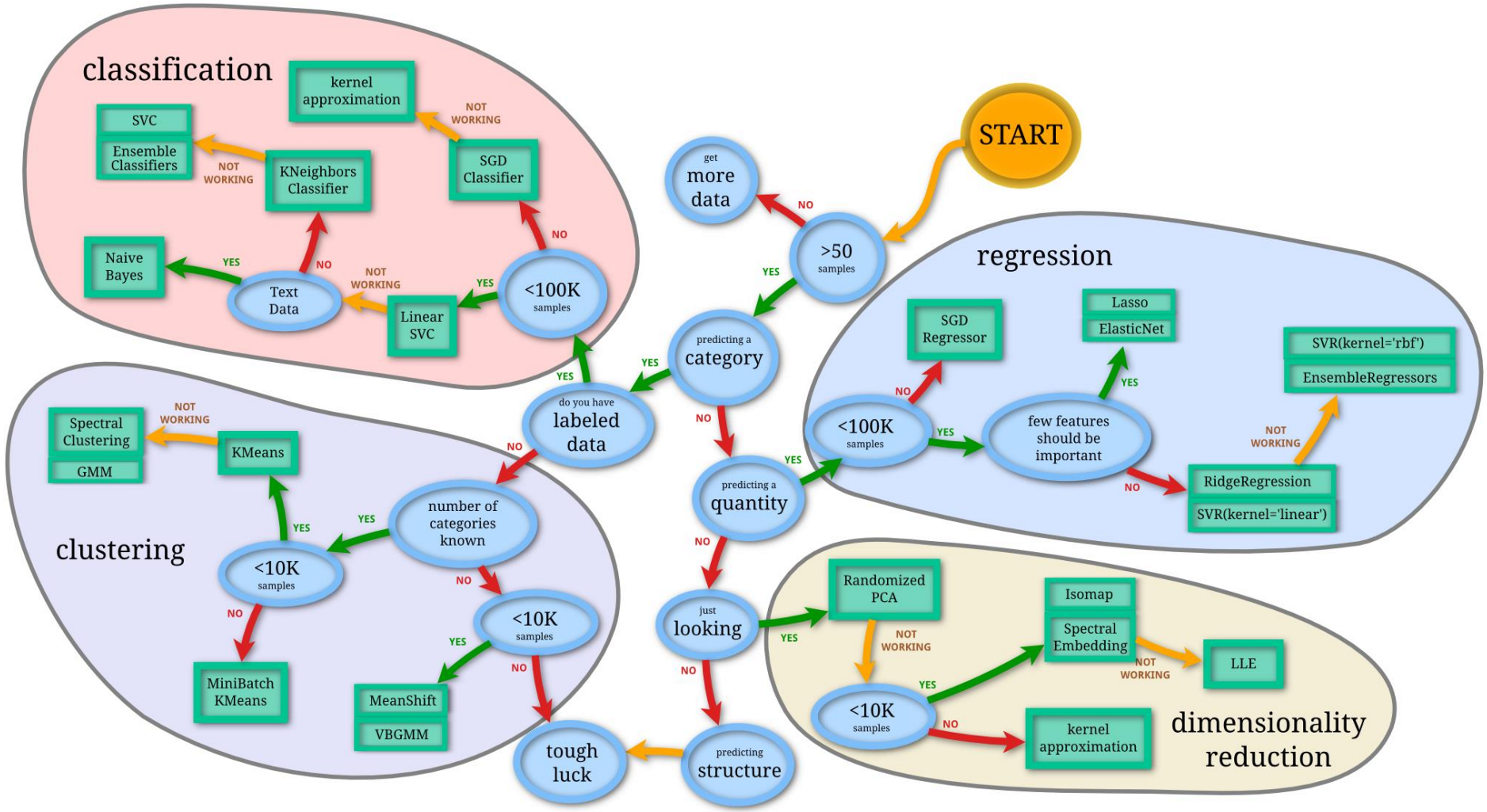
matteo.t.degiacomi@durham.ac.uk

Dr Antonia Mey

University of Edinburgh

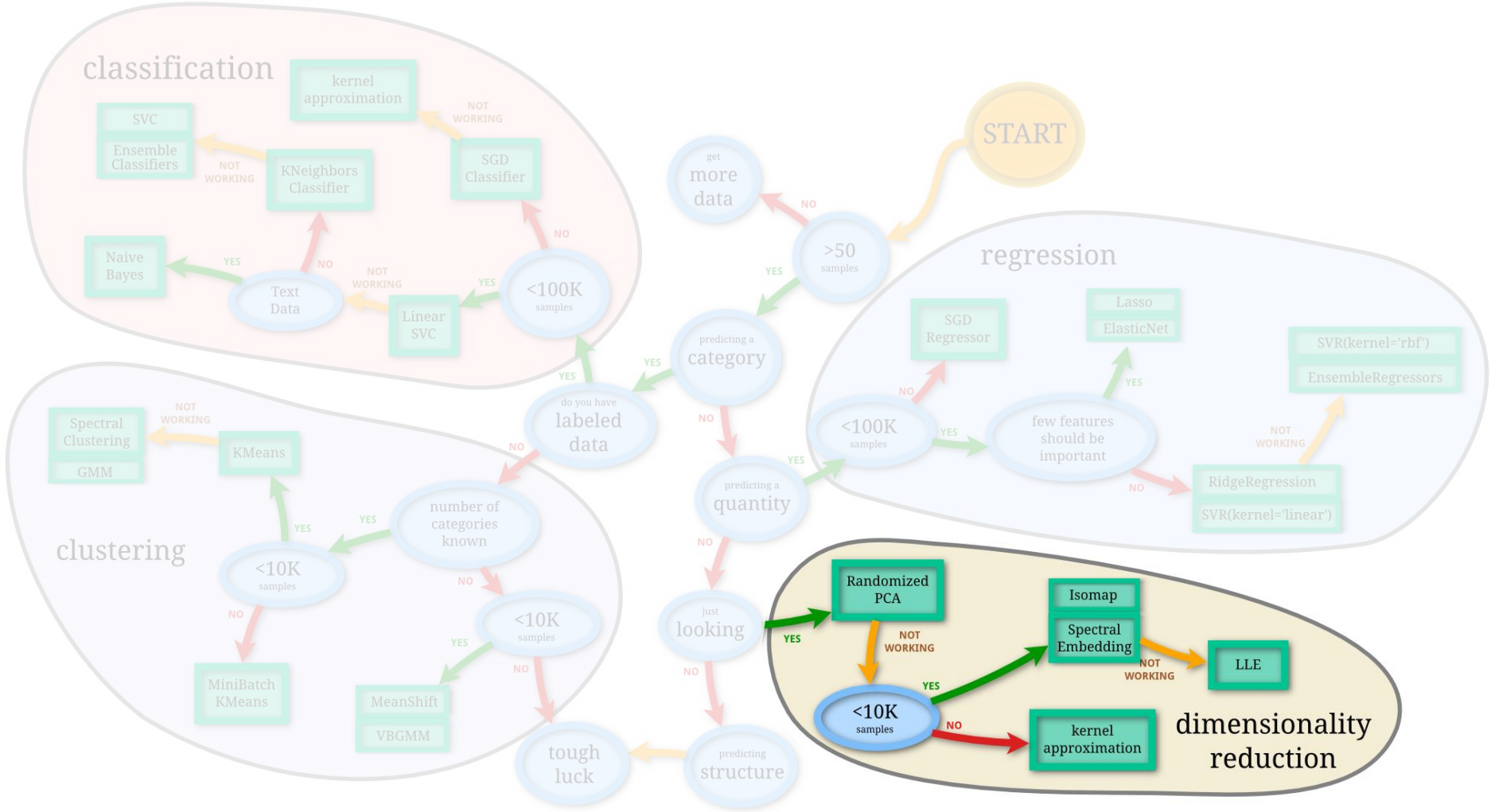
antonia.mey@ed.ac.uk

The Data Mining world



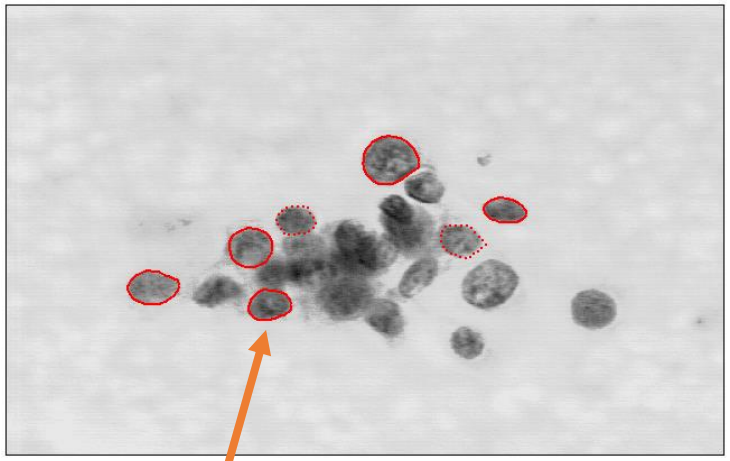
From **scikit-learn.org**

The Data Mining world



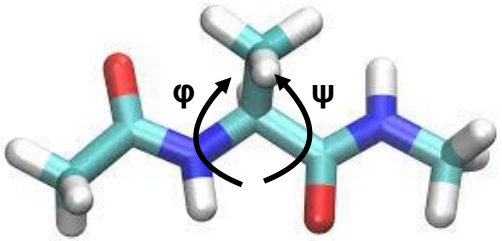
From **scikit-learn.org**

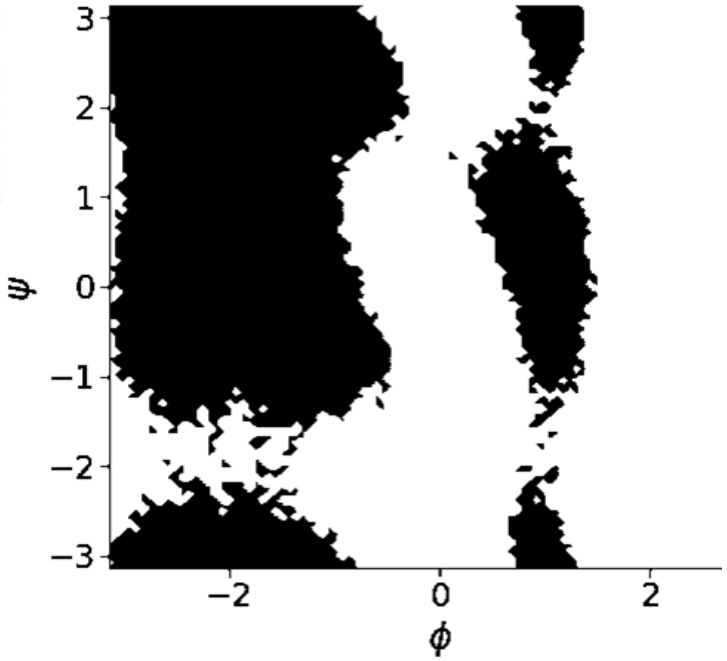
features are possible ways to represent data



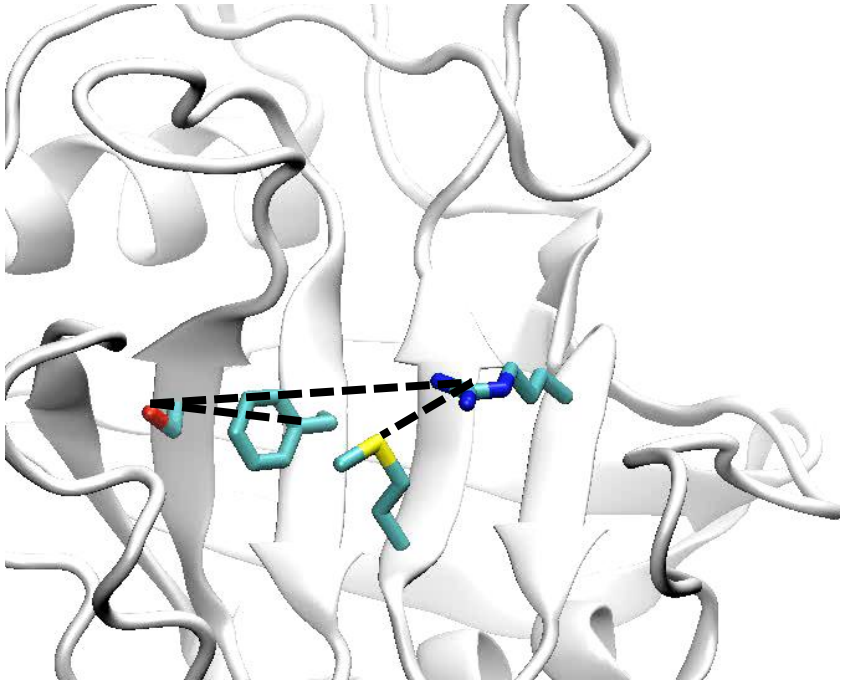
Sphericity (***1***)

Pixels colour
(28x28 = ***784***)





Torsional angles (***2***)



atomic positions (***459***)
atomic distances (***3***)

Not all features are useful

Task: predict the weather in Edinburgh using historical data

data = {X, Y, Z} → sun, rain, snow

{Temperature (C), ~~Temperature (K)~~, Humidity (g/m^3)}

2 decorrelated features

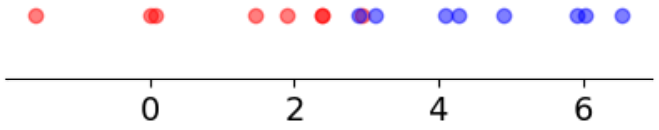
{Temperature (C), ~~Swiss cheese export (£)~~, Humidity (g/m^3)}

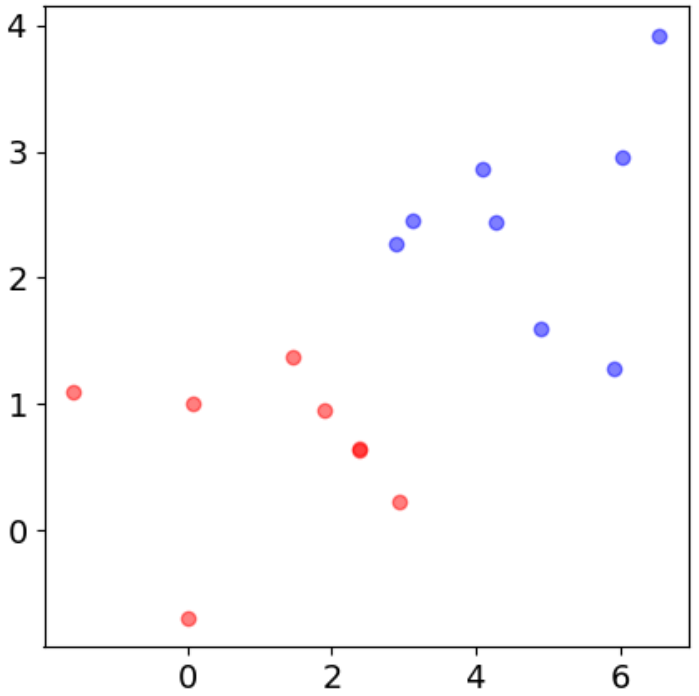
2 relevant features

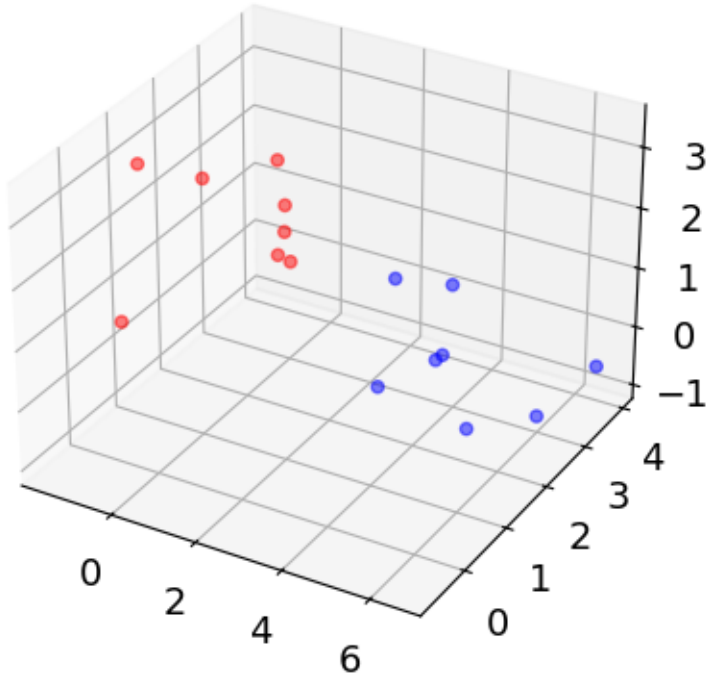
{Avg. gas expenditure (£), Heat strokes (#), Slipping accidents (#), Sunscreen sold (£)}

4 features connected to another quantity: temperature

Curse of dimensionality







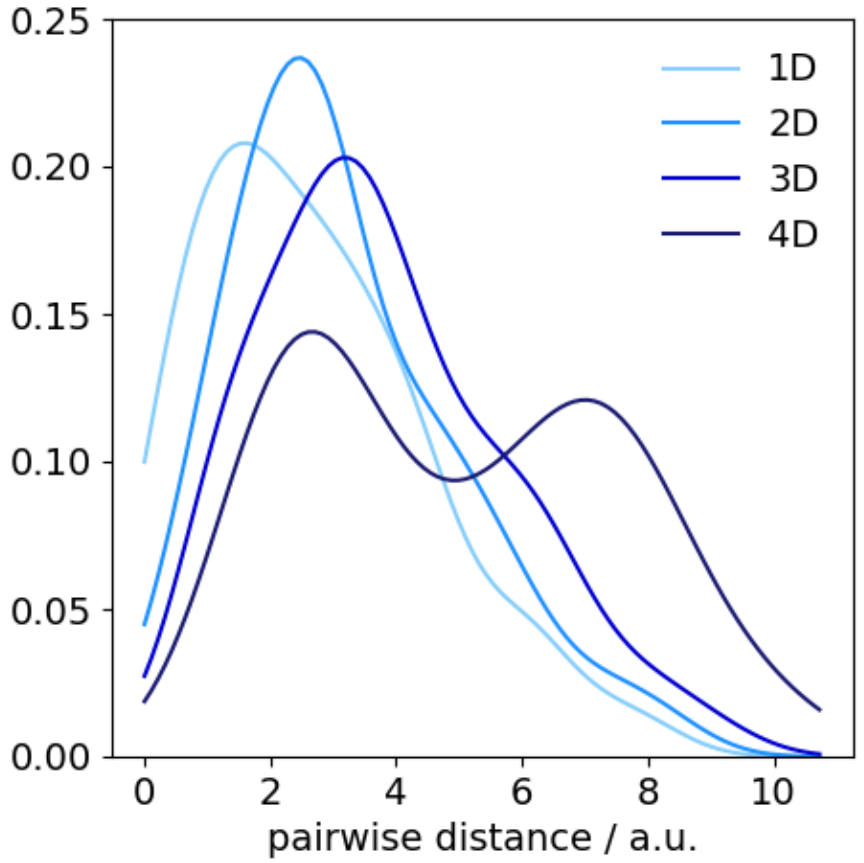
$$\|x\|_2 = \sqrt{\sum_{i=1}^{N} x_i^2}$$

Distances between *M* data points $\boldsymbol{x} \in \mathbb{R}^N$ increase, when *N* increases

Problem: less data density increases uncertainty on underlying data structure

[Extra] Curse of dimensionality

$$\|x\|_2 = \sqrt{\sum_{i=1}^{N} x_i^2}$$



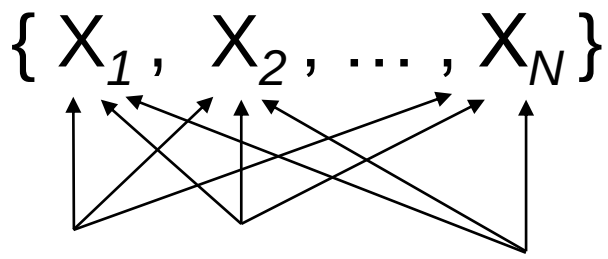
Distribution of pairwise distances between points shown in previous slide

Distances between *M* data points $\boldsymbol{x} \in \mathbb{R}^N$ increase, when *N* increases

Problem: less data density increases uncertainty on underlying data structure

Reducing features increases data density

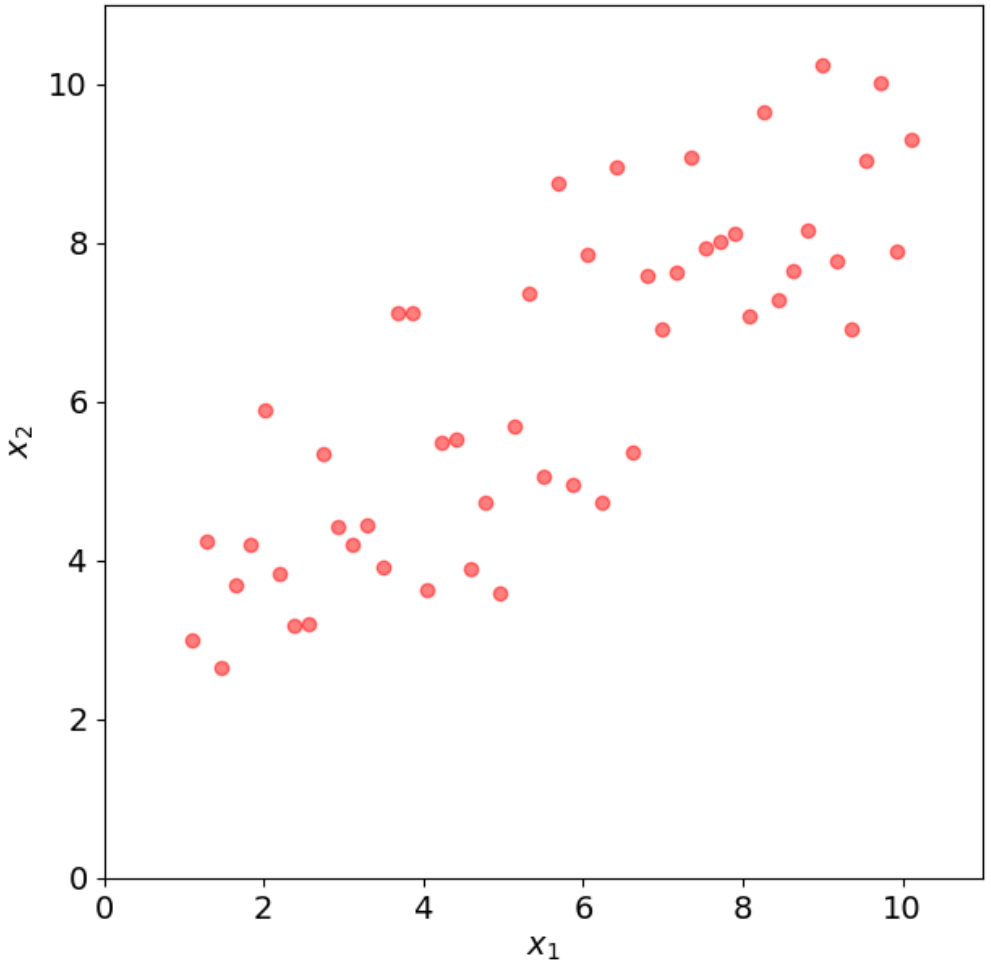
- Chose appropriate features [expert user]
- Remove features
- Find a lower dimensional representation *E* of features *X*



$E_i = f(\{X\})$
where $K < N$

Principal Components Analysis (PCA)

Let $\boldsymbol{X}$ a dataset of M datapoints in N dimensions (here, M=50 and N=2)



- Center data:

$$\boldsymbol{X}' = \boldsymbol{X} - \mu$$

- Compute data covariance matrix $\boldsymbol{C}$:

$$c_{i,j} = \frac{1}{M}\sum_{k=1}^{M} \boldsymbol{x}'_i \boldsymbol{x}'_j$$

- Calculate eigenvalue decomposition:

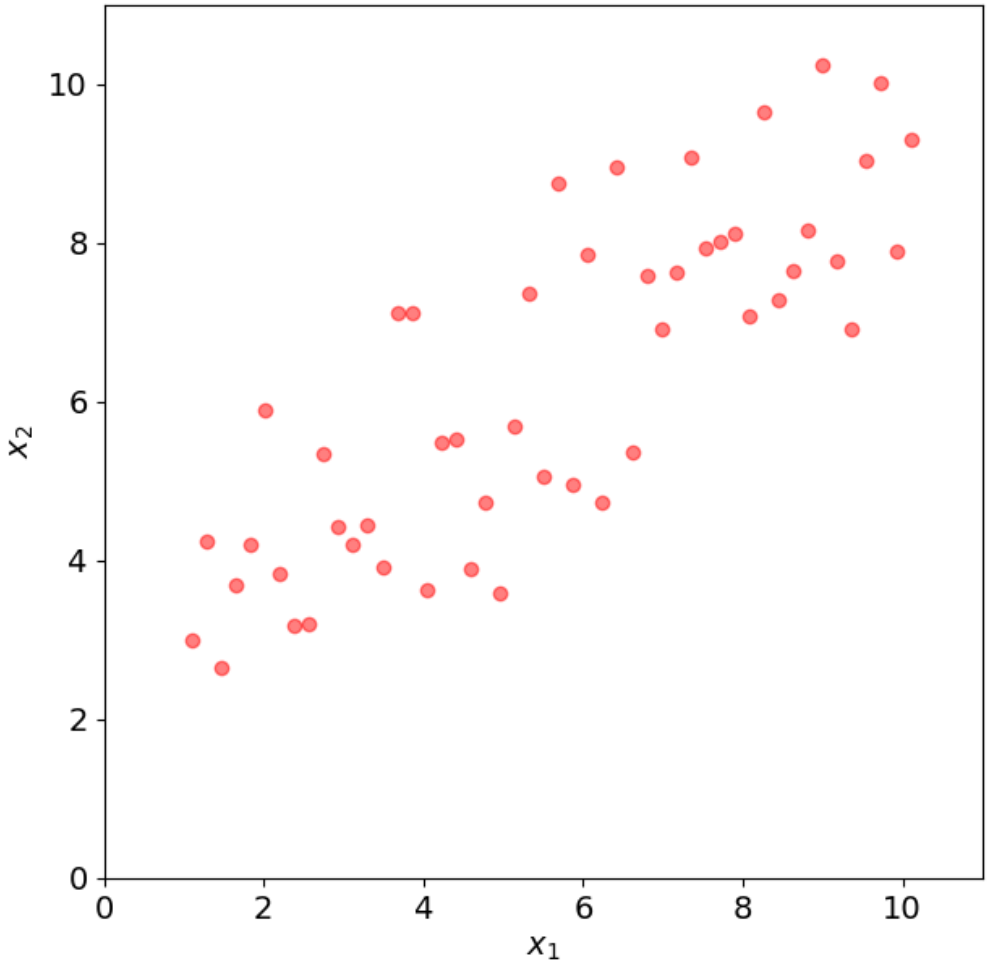
$$\boldsymbol{C} = \boldsymbol{V}\boldsymbol{\lambda}\boldsymbol{V}^{-1}$$

*N*x*N* matrix of ***eigenvectors***

*N*x*N* diagonal matrix of ***eigenvalues***

[Extra] Principal Components Analysis (PCA)

Let $\boldsymbol{X}$ a dataset of *M* datapoints in *N* dimensions (here, *M*=50 and *N*=2)



An eigenvector $\boldsymbol{v}$ of $\boldsymbol{C}$ respects:

$$\boldsymbol{Cv} = \boldsymbol{\lambda v}$$

Find eigenvalues as the roots of the characteristic polynomial:

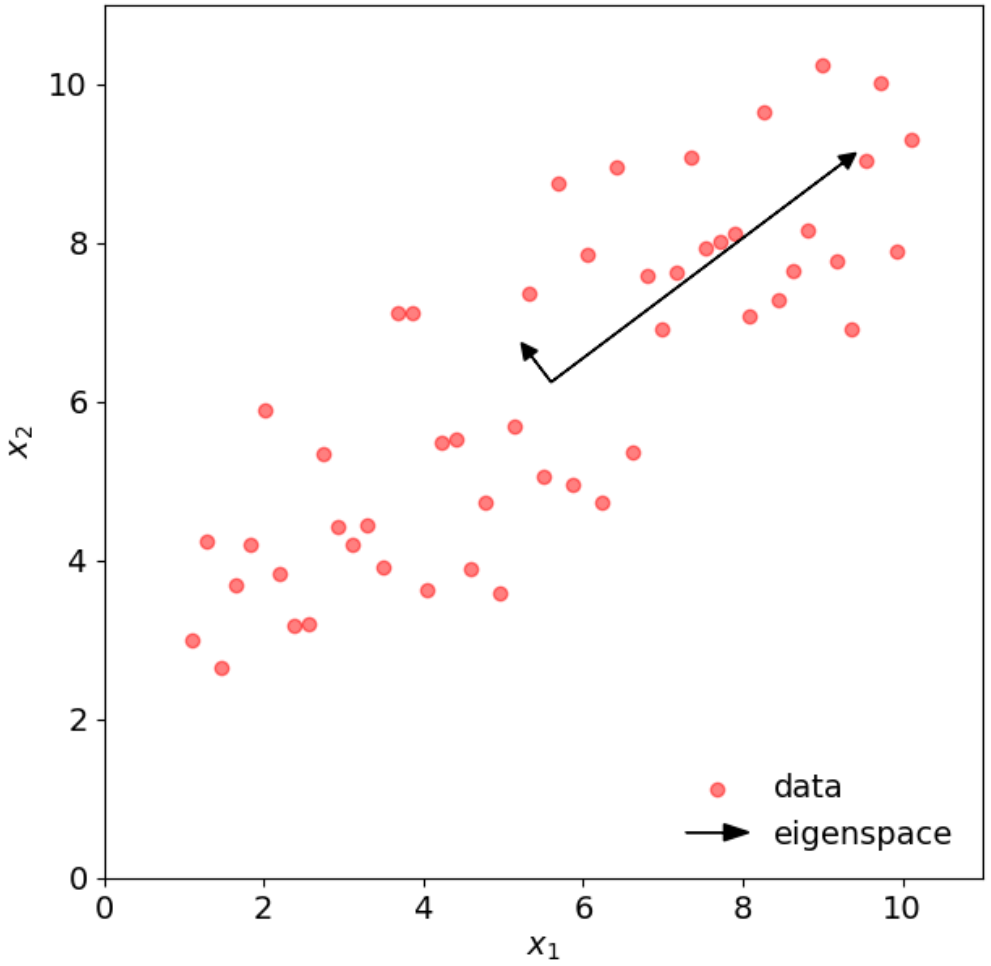
$$p(\boldsymbol{\lambda}) = \det(\boldsymbol{C} - \boldsymbol{\lambda I}) = 0$$

The *i*-th eigenvector $\boldsymbol{v_i}$ is found by solving:

$$\boldsymbol{Cv_i} = \boldsymbol{\lambda_i v_i}$$

Principal Components Analysis (PCA)

Let $\boldsymbol{X}$ a dataset of *M* datapoints in *N* dimensions (here, *M*=50 and *N*=2)



$$\boldsymbol{C = V\lambda V^{-1}}$$

$\boldsymbol{V = [v_1 \ldots v_N]}$ eigenvectors: orthonormal base

$\boldsymbol{\lambda}$ eigenvalues: scalars defining the importance of each eigenvector

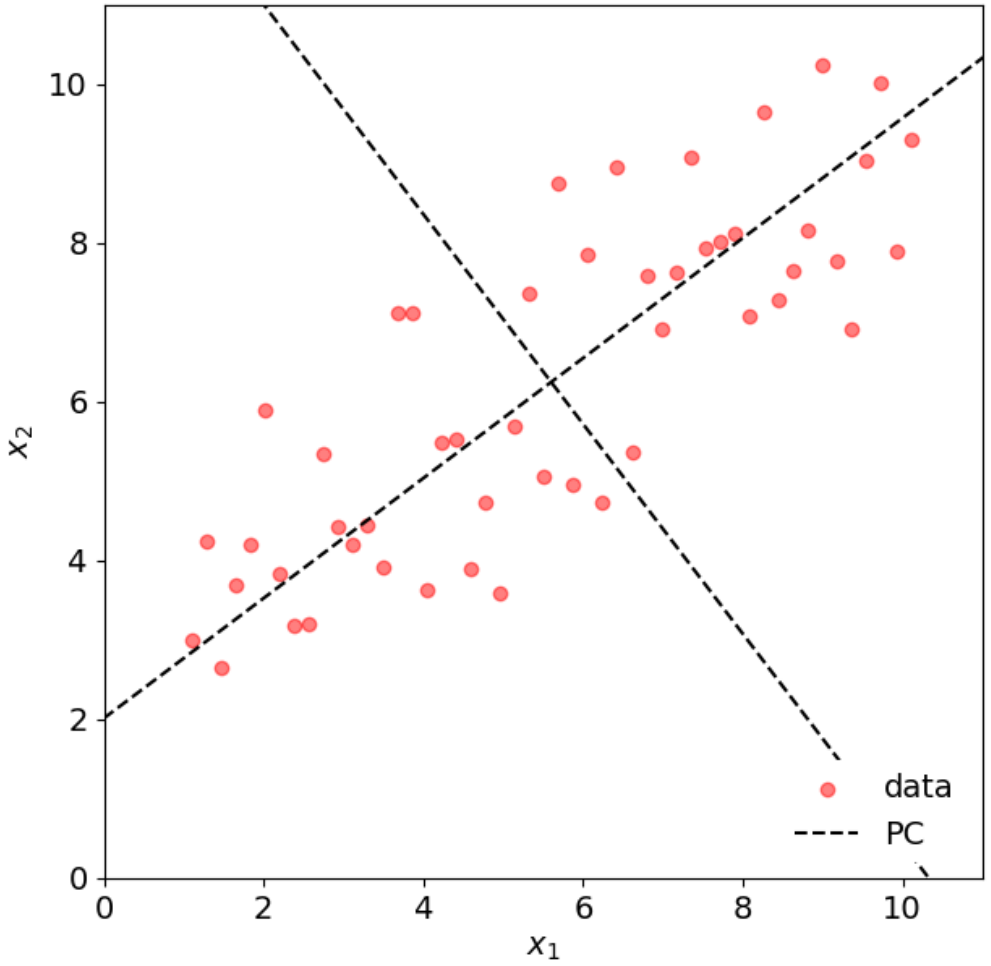
Importance $\boldsymbol{r_i}$ of each eigenvector $\boldsymbol{v_i}$:

$$\boldsymbol{r_i = \frac{\lambda_i}{\sum \lambda}}$$

Sort $\boldsymbol{V}$ *and* $\boldsymbol{\lambda}$ *according to* $\boldsymbol{\lambda}$

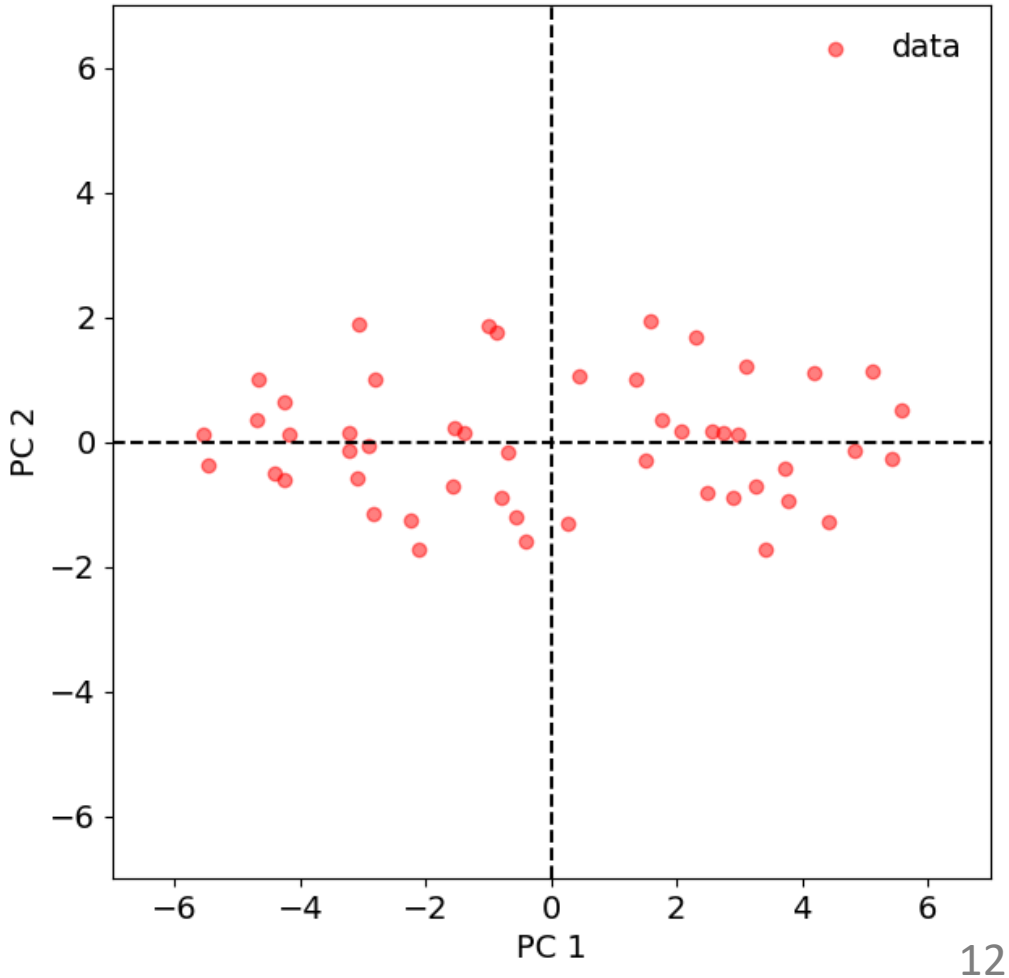
Projection into the eigenspace

Let **X** a dataset of *M* datapoints in *N* dimensions (here, *M*=50 and *N*=2)



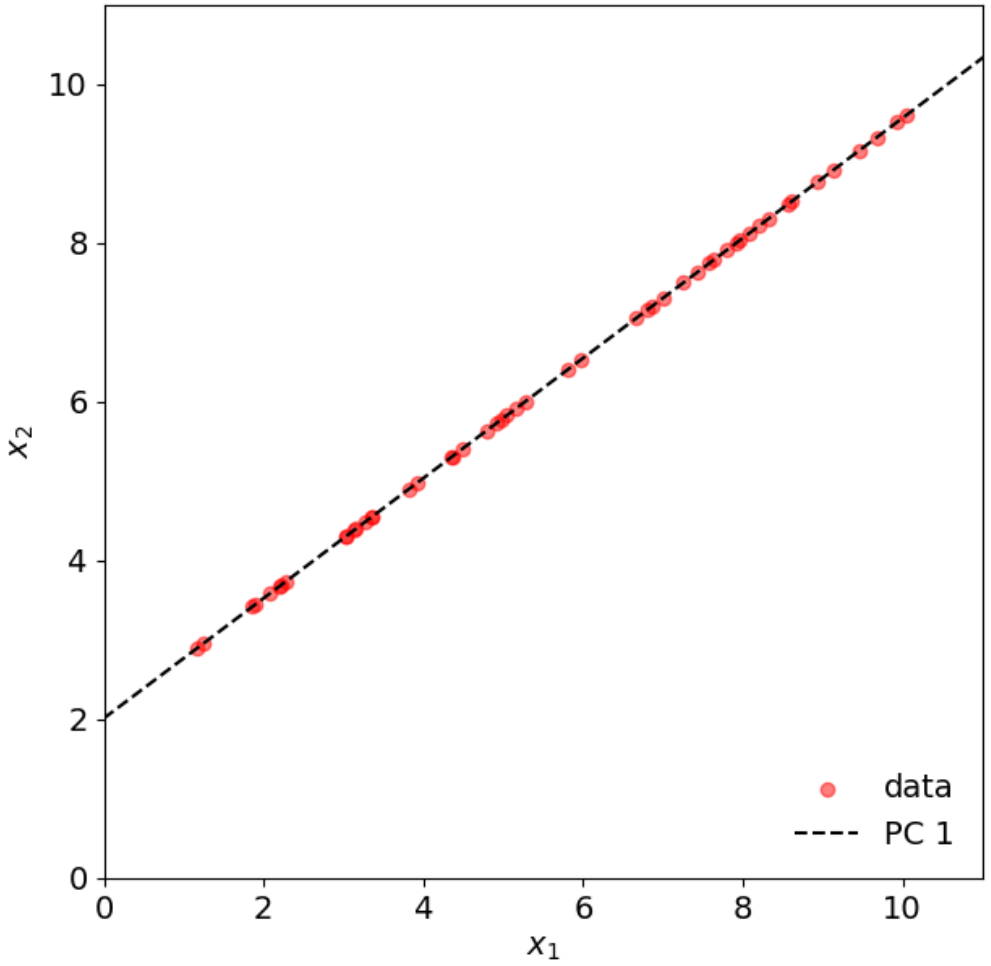
$$p = \boldsymbol{V}^T(\boldsymbol{x} - \mu)$$

transform



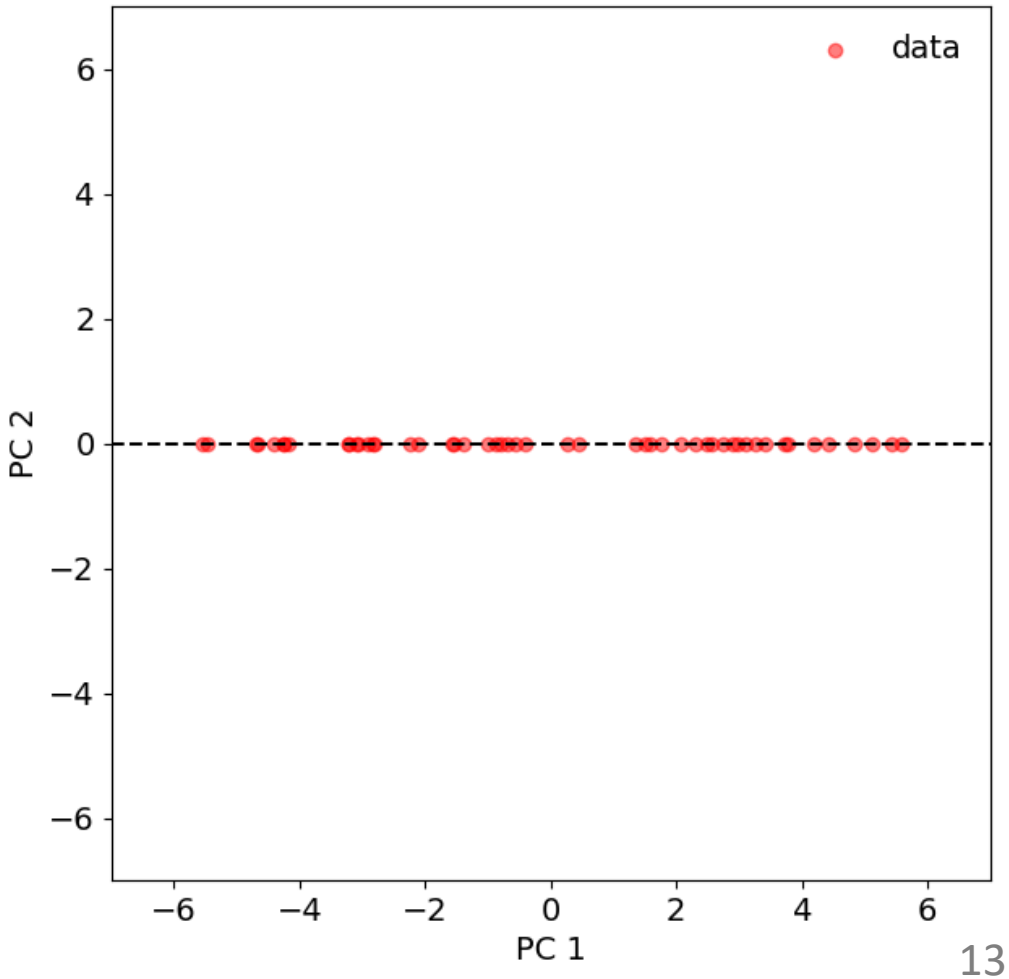
Dimensionality reduction

Remove dimensions that least contribute to data variance



$$p = \boldsymbol{V}^T(\boldsymbol{x} - \mu)$$

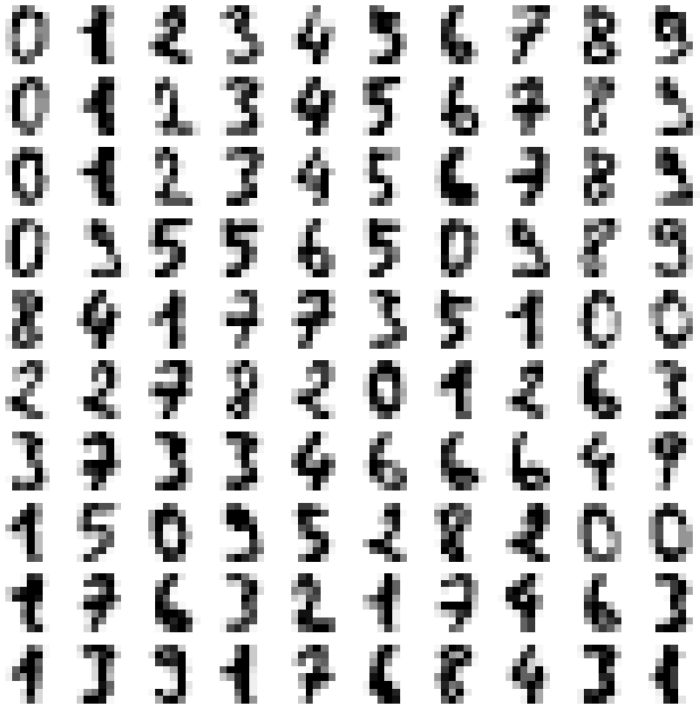
transform



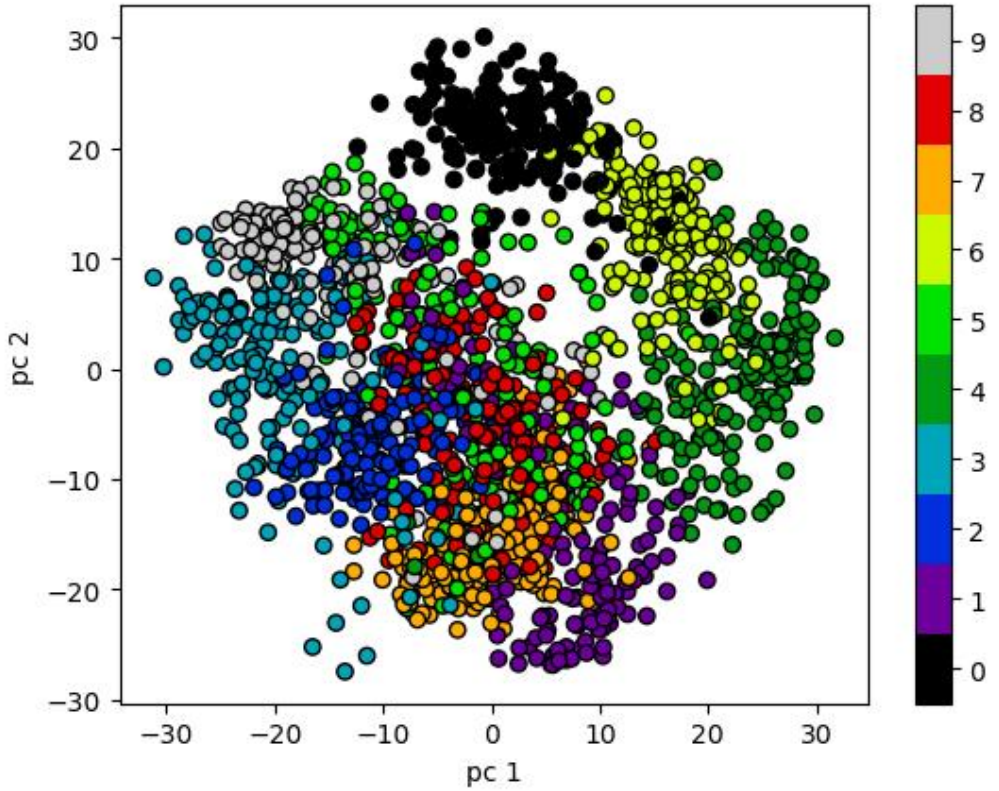
[Example 1] Representing written digits

MNIST: database of written digits

Input: 8x8 pixel images (64-dimensional vector)

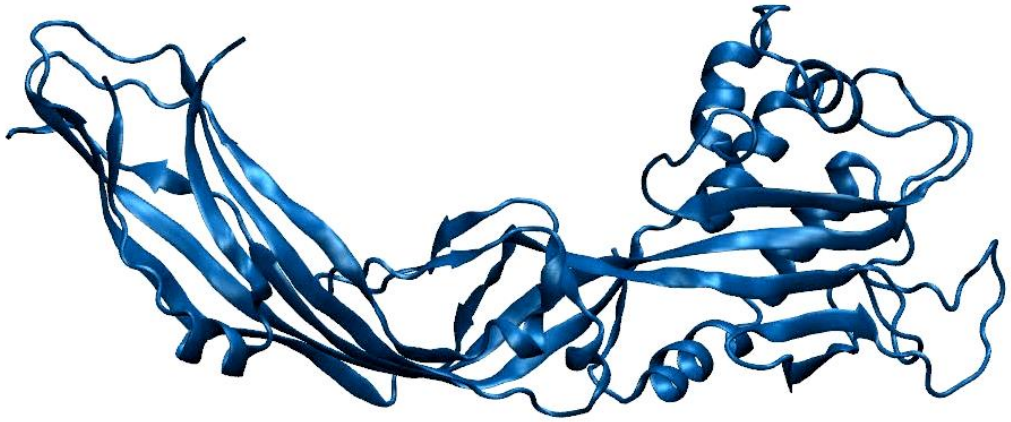


Projection in 2 principal components (PCs) separates some of the digits!

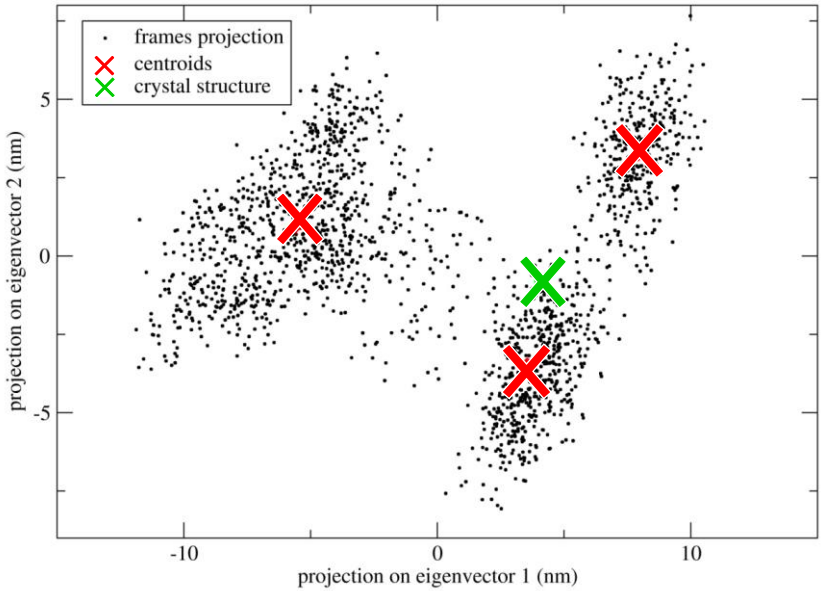


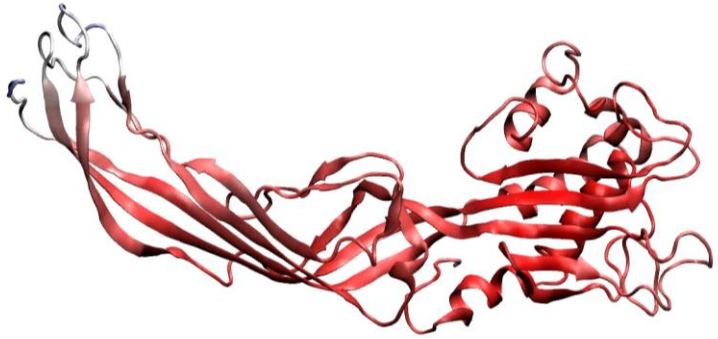
[Example 2] Identifying dominant motions in proteins

Protein MD simulation



Eigenspace of C_α coordinates



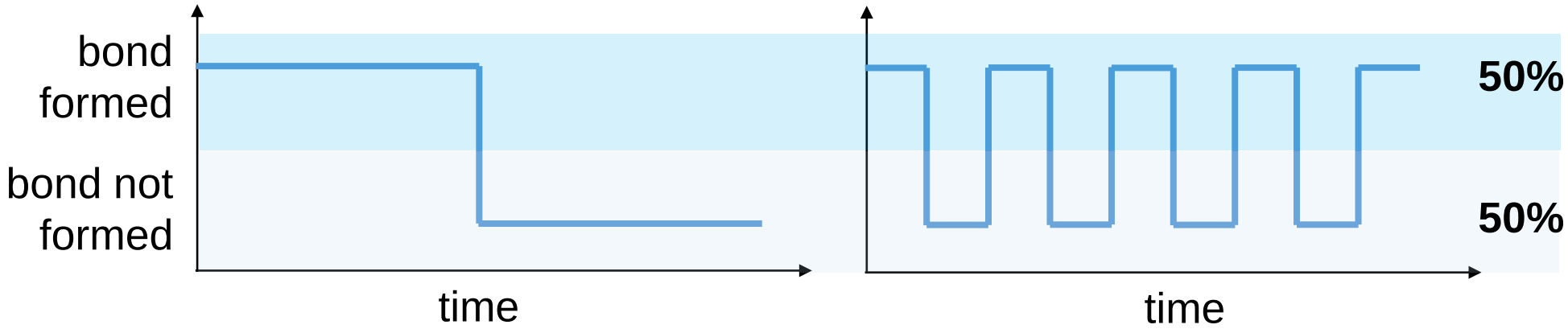


- Simulations are complex and noisy
- The first PCs capture large-scale collective motions, last ones capture noise

Degiacomi, M.T. & Dal Peraro, M.. *Macromolecular symmetric assembly prediction using swarm intelligence dynamic modeling*. ***Structure***, 2013

largest-scale motion ≠ slowest motion

Thought experiment reminder: typically hydrogen bond is considered established if donor-acceptor distance <2.5 Å, and donor-acceptor-hydrogen angle <20°.



- Biological function is often determied by kinetics of a processes.
- Two processes with different kinetics can have the same statistical distribution in terms of structure
- PCA separates in terms of structure, not timescales!

Time-lagged independent component analysis (tICA)

tICA is a **linear transform** similar to PCA

The transform is chosen such that, amongst all linear transforms, tICA **maximizes the autocorrelation** of transformed coordinates.

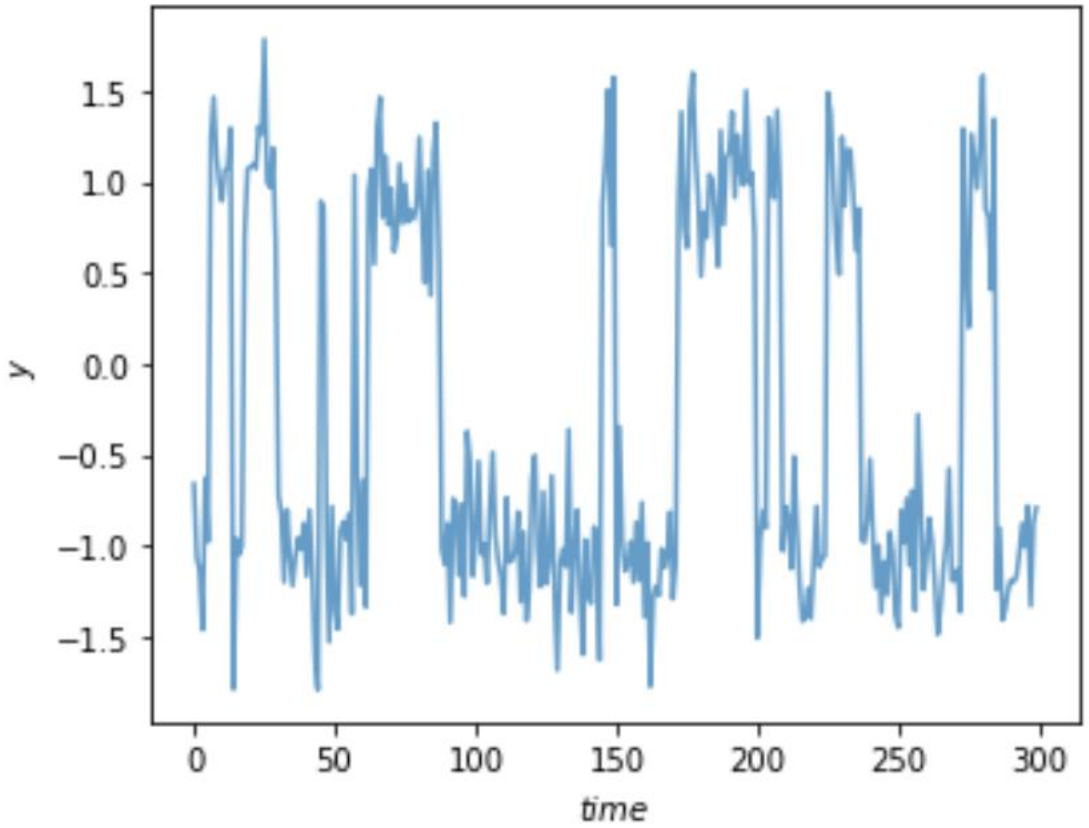
$\mathbf{r}(t) = (r_i(t))_{i=1,\ldots,D}$

D-dimensional input data vector that is mean free, i.e., $\mathbf{r}(t) = \mathbf{r}(t) - \langle \mathbf{r}(t) \rangle_t$

Computing the covariance of the data at $t = 0$ and $t = \tau$ which is the lag-time chosen

$c_{ij}(\tau) = \langle r_i(t) r_j(t+\tau) \rangle_t$

enables computing two covariance matrices: $\mathbf{C}(0)$ and $\mathbf{C}(\tau)$



F. Nüske, […], A. S. J. S. Mey and F. Noé: ***JCTC***, 2014

G. Perez-Hernandez, et al., ***JCTC***, 2013

C. R. Schwantes et al., ***JCTC***, 2013

Time-lagged independent component analysis (tICA)

tICA is a **linear transform** similar to PCA

The transform is chosen such that, amongst all linear transforms, tICA **maximizes the autocorrelation** of transformed coordinates.

Entries of the covariance matrix can be computed as:

$$c_{ij}(\tau) = \frac{1}{N-\tau-1}\sum_{t=1}^{N-\tau} r_i(t) r_j(t+\tau)$$

$\mathbf{C}(0)$ will be a symmetric matrix. The symmetry of $\mathbf{C}(\tau)$ will need to be enforced with:

$$\mathbf{C}(\tau) = \frac{1}{2}(\mathbf{C}_d(\tau) + \mathbf{C}_d^{\mathsf{T}}(\tau))$$

We can now solve the generalised eigenvalue problem:

$$\mathbf{C}(\tau)\mathbf{U} = \mathbf{C}(0)\mathbf{U}\mathbf{\Lambda}$$

Eigenvector matrix containing ICs

Diagonal matrix with eigenvalues

$$\mathbf{z}^{\mathsf{T}}(t) = \mathbf{r}^{\mathsf{T}}(t)\mathbf{U}$$

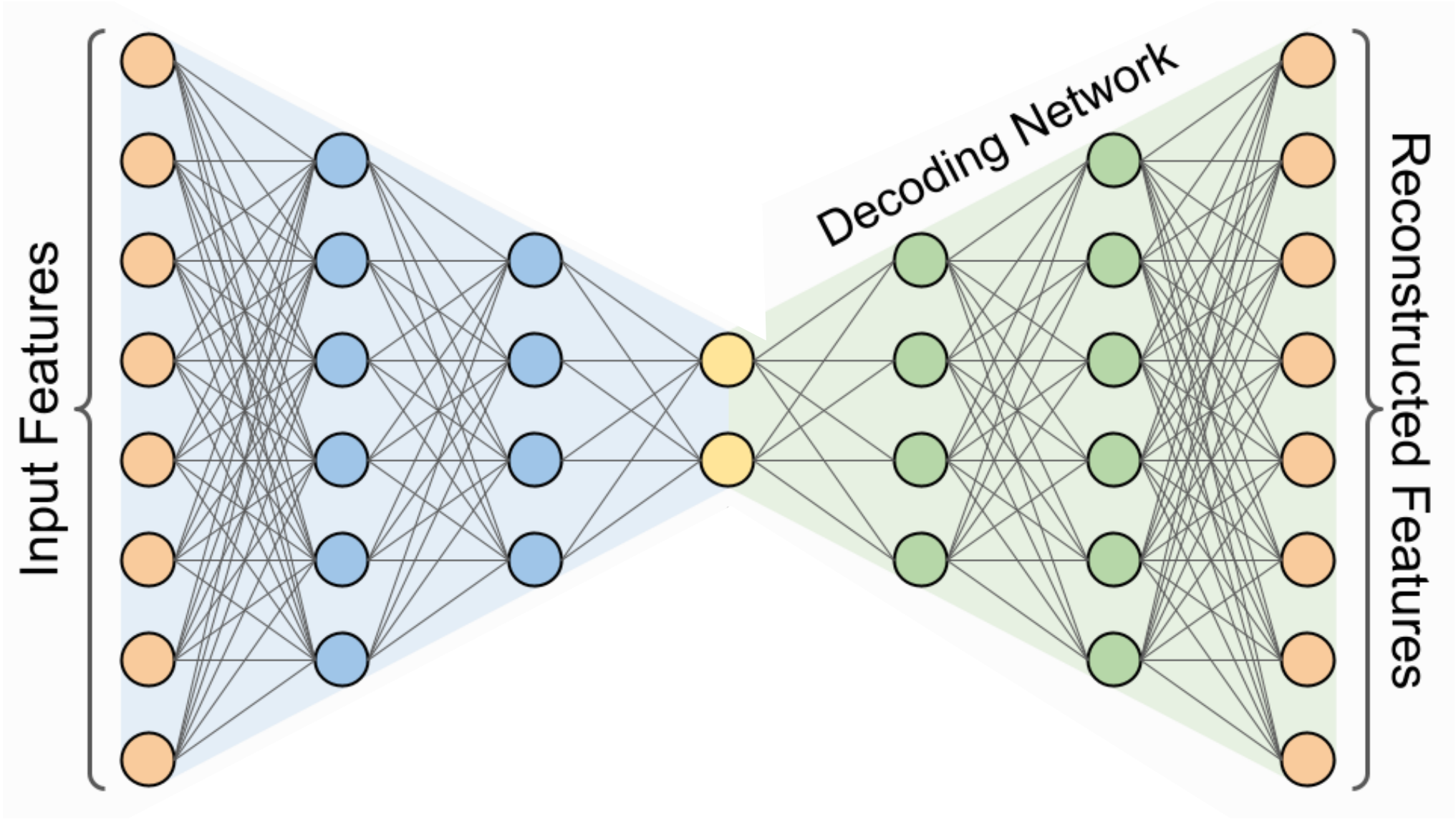
M columns of full rank U for DR

F. Nüske, […], A. S. J. S. Mey and F. Noé: ***JCTC***, 2014

G. Perez-Hernandez, et al., ***JCTC***, 2013

C. R. Schwantes et al., ***JCTC***, 2013

[Extra] Dimensionality reduction with neural networks



PCA dimensionality reduction is *interpretable*, that of t-SNE and neural networks is not.